GlobalSuperSales - Data Analysis¶

Data set Link : https://www.kaggle.com/code/tingray/global-superstore-eda

GitHub Link : https://github.com/Official-Vivek-Singh?tab=repositories

LinkedIn : https://www.linkedin.com/in/vivekvishwas/

```
#Importing libraries
import pandas as pd

import numpy as np

import datetime as dt

import matplotlib.pyplot as plt

import seaborn as sns

# setting the Float Data type display format

pd.set_option('display.float_format', lambda x: '%.3f' %x)
print('Float format Set-up Done !!')
```

```
Float format Set-up Done !!
```

```
# importing Dataset from local machine
df= pd.read_csv('D:\Vivek_Stuff\Learning Stuff\Python_Work\Python
Project\superstore_dataset2011-2015.csv', encoding = 'ISO-8859-1')
print('Dataset loaded !!')
```

```
Dataset loaded !!
```

Show top 5 rows of your dataset¶

```
# .head()
df.head()
```

```
   Row ID         Order ID Order Date Ship Date       Ship Mode
Customer ID  \
0   42433     AG-2011-2040   1/1/2011  6/1/2011  Standard Class    TB-
11280
1   22253    IN-2011-47883   1/1/2011  8/1/2011  Standard Class    JH-
15985
2   48883     HU-2011-1220   1/1/2011  5/1/2011    Second Class
AT-735
3   11731  IT-2011-3647632   1/1/2011  5/1/2011    Second Class    EM-
```

```
14140
4   22255    IN-2011-47883   1/1/2011  8/1/2011  Standard Class    JH-
15985

     Customer Name      Segment         City            State  ...  \
0  Toby Braunhardt     Consumer  Constantine      Constantine  ...
1      Joseph Holt     Consumer  Wagga Wagga  New South Wales  ...
2    Annie Thurman     Consumer     Budapest         Budapest  ...
3     Eugene Moren  Home Office    Stockholm        Stockholm  ...
4      Joseph Holt     Consumer  Wagga Wagga  New South Wales  ...

          Product ID         Category Sub-Category  \
0  OFF-TEN-10000025  Office Supplies      Storage
1   OFF-SU-10000618  Office Supplies     Supplies
2  OFF-TEN-10001585  Office Supplies      Storage
3   OFF-PA-10001492  Office Supplies        Paper
4   FUR-FU-10003447        Furniture  Furnishings

                  Product Name    Sales Quantity Discount   Profit  \
0          Tenex Lockers, Blue  408.300        2      0.0  106.140
1     Acme Trimmer, High Speed  120.366        3      0.1   36.036
2      Tenex Box, Single Width   66.120        4      0.0   29.640
3  Enermax Note Cards, Premium   44.865        3      0.5  -26.055
4   Eldon Light Bulb, Duo Pack  113.670        5      0.1   37.770

   Shipping Cost  Order Priority
0          35.46          Medium
1           9.72          Medium
2           8.17            High
3           4.82            High
4           4.70          Medium

[5 rows x 24 columns]
```

Show last 5 rows of your dataset

```
# Show Dataset last 5 Rows
df.tail()
```

```
       Row ID        Order ID  Order Date Ship Date       Ship Mode  \
51285   32593  CA-2014-115427  31-12-2014  4/1/2015  Standard Class
51286   47594    MO-2014-2560  31-12-2014  5/1/2015  Standard Class
51287    8857  MX-2014-110527  31-12-2014  2/1/2015    Second Class
51288    6852  MX-2014-114783  31-12-2014  6/1/2015  Standard Class
51289   36388  CA-2014-156720  31-12-2014  4/1/2015  Standard Class

      Customer ID     Customer Name    Segment       City
State  \
```

```
51285    EB-13975        Erica Bern  Corporate  Fairfield
California
51286     LP-7095         Liz Preis   Consumer     Agadir  Souss-
Massa-Draâ
51287    CM-12190  Charlotte Melton   Consumer    Managua
Managua
51288    TD-20995     Tamara Dahlen   Consumer     Juárez
Chihuahua
51289    JM-15580     Jill Matthias   Consumer   Loveland
Colorado

       ...         Product ID         Category Sub-Category  \
51285  ...   OFF-BI-10002103  Office Supplies      Binders
51286  ...  OFF-WIL-10001069  Office Supplies      Binders
51287  ...   OFF-LA-10004182  Office Supplies       Labels
51288  ...   OFF-LA-10000413  Office Supplies       Labels
51289  ...   OFF-FA-10003472  Office Supplies    Fasteners

                                              Product Name   Sales
Quantity  \
51285  Cardinal Slant-D Ring Binder, Heavy Gauge Vinyl  13.904
2
51286          Wilson Jones Hole Reinforcements, Clear   3.990
1
51287           Hon Color Coded Labels, 5000 Label Set  26.400
3
51288           Hon Legal Exhibit Labels, Alphabetical   7.120
1
51289                                Bagged Rubber Bands   3.024
3

      Discount   Profit  Shipping Cost  Order Priority
51285      0.2   4.5188           0.89          Medium
51286      0.0   0.4200           0.49          Medium
51287      0.0  12.3600           0.35          Medium
51288      0.0   0.5600           0.20          Medium
51289      0.2  -0.6048           0.17          Medium

[5 rows x 24 columns]
```

Show size of your dataset

```
df.shape

# .shape is used to find the size of dataset

(51290, 24)
```

Show Headers/Columns of your dataset

```
df.columns

# .columns returns the headers of Dataset
```

```
Index(['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'],
      dtype='object')
```

Show All information of your dataset

```
df.info()

# .info() returnt the complete information of dataset , it returns the
column wise Total value count, datatype etc..
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          51290 non-null  int64
 1   Order ID        51290 non-null  object
 2   Order Date      51290 non-null  object
 3   Ship Date       51290 non-null  object
 4   Ship Mode       51290 non-null  object
 5   Customer ID     51290 non-null  object
 6   Customer Name   51290 non-null  object
 7   Segment         51290 non-null  object
 8   City            51290 non-null  object
 9   State           51290 non-null  object
 10  Country         51290 non-null  object
 11  Postal Code     9994 non-null   float64
 12  Market          51290 non-null  object
 13  Region          51290 non-null  object
 14  Product ID      51290 non-null  object
 15  Category        51290 non-null  object
 16  Sub-Category    51290 non-null  object
 17  Product Name    51290 non-null  object
 18  Sales           51290 non-null  float64
 19  Quantity        51290 non-null  int64
 20  Discount        51290 non-null  float64
```

```
 21  Profit          51290 non-null  float64
 22  Shipping Cost   51290 non-null  float64
 23  Order Priority  51290 non-null  object
dtypes: float64(5), int64(2), object(17)
memory usage: 9.4+ MB
```

Show Null/Missing Values of your dataset

```
df.isnull()
df.isna()

# isna() and isnull() both returns the True/False
```

```
       Row ID  Order ID  Order Date  Ship Date  Ship Mode  Customer ID
\
0       False     False       False      False      False        False

1       False     False       False      False      False        False

2       False     False       False      False      False        False

3       False     False       False      False      False        False

4       False     False       False      False      False        False

...       ...       ...         ...        ...        ...          ...

51285   False     False       False      False      False        False

51286   False     False       False      False      False        False

51287   False     False       False      False      False        False

51288   False     False       False      False      False        False

51289   False     False       False      False      False        False


       Customer Name  Segment   City  State  ...  Product ID  Category
\
0              False    False  False  False  ...       False     False

1              False    False  False  False  ...       False     False

2              False    False  False  False  ...       False     False

3              False    False  False  False  ...       False     False

4              False    False  False  False  ...       False     False
```

```
...              ...      ...    ...    ...  ...         ...       ...
51285          False    False  False  False  ...       False     False
51286          False    False  False  False  ...       False     False
51287          False    False  False  False  ...       False     False
51288          False    False  False  False  ...       False     False
51289          False    False  False  False  ...       False     False

       Sub-Category  Product Name  Sales  Quantity  Discount
Profit  \
0              False         False  False     False     False   False
1              False         False  False     False     False   False
2              False         False  False     False     False   False
3              False         False  False     False     False   False
4              False         False  False     False     False   False
...              ...           ...    ...       ...       ...     ...
51285          False         False  False     False     False   False
51286          False         False  False     False     False   False
51287          False         False  False     False     False   False
51288          False         False  False     False     False   False
51289          False         False  False     False     False   False

       Shipping Cost  Order Priority
0              False           False
1              False           False
2              False           False
3              False           False
4              False           False
...              ...             ...
51285          False           False
51286          False           False
51287          False           False
51288          False           False
51289          False           False
```

```
[51290 rows x 24 columns]
```

```
df.isna()
```

```
# .isna() returnt he Null status in True/False
```

```
       Row ID  Order ID  Order Date  Ship Date  Ship Mode  Customer ID
\
0       False     False       False      False      False        False

1       False     False       False      False      False        False

2       False     False       False      False      False        False

3       False     False       False      False      False        False

4       False     False       False      False      False        False

...       ...       ...         ...        ...        ...          ...

51285   False     False       False      False      False        False

51286   False     False       False      False      False        False

51287   False     False       False      False      False        False

51288   False     False       False      False      False        False

51289   False     False       False      False      False        False


       Customer Name  Segment   City  State  ...  Product ID  Category
\
0              False    False  False  False  ...       False     False

1              False    False  False  False  ...       False     False

2              False    False  False  False  ...       False     False

3              False    False  False  False  ...       False     False

4              False    False  False  False  ...       False     False

...              ...      ...    ...    ...  ...         ...       ...

51285          False    False  False  False  ...       False     False

51286          False    False  False  False  ...       False     False

51287          False    False  False  False  ...       False     False
```

```
51288          False    False  False  False  ...       False     False

51289          False    False  False  False  ...       False     False


       Sub-Category  Product Name  Sales  Quantity  Discount
Profit  \
0             False         False  False     False     False   False

1             False         False  False     False     False   False

2             False         False  False     False     False   False

3             False         False  False     False     False   False

4             False         False  False     False     False   False

...             ...           ...    ...       ...       ...     ...

51285         False         False  False     False     False   False

51286         False         False  False     False     False   False

51287         False         False  False     False     False   False

51288         False         False  False     False     False   False

51289         False         False  False     False     False   False


       Shipping Cost  Order Priority
0              False           False
1              False           False
2              False           False
3              False           False
4              False           False
...              ...             ...
51285          False           False
51286          False           False
51287          False           False
51288          False           False
51289          False           False

[51290 rows x 24 columns]
```

Show Count of Missing/Null Values of your dataset

```
df.isnull().sum()

# .sum() to add the value ...
```

```
Row ID                0
Order ID              0
Order Date            0
Ship Date             0
Ship Mode             0
Customer ID           0
Customer Name         0
Segment               0
City                  0
State                 0
Country               0
Postal Code       41296
Market                0
Region                0
Product ID            0
Category              0
Sub-Category          0
Product Name          0
Sales                 0
Quantity              0
Discount              0
Profit                0
Shipping Cost         0
Order Priority        0
dtype: int64
```

Percenteage of Missing Value in Each Column

```
## Percenteage of Missing Value in Each Column
(df.isna().sum() / df.shape[0]) * 100
```

```
Row ID            0.00000
Order ID          0.00000
Order Date        0.00000
Ship Date         0.00000
Ship Mode         0.00000
Customer ID       0.00000
Customer Name     0.00000
Segment           0.00000
City              0.00000
```

```
State               0.00000
Country             0.00000
Postal Code        80.51472
Market              0.00000
Region              0.00000
Product ID          0.00000
Category            0.00000
Sub-Category        0.00000
Product Name        0.00000
Sales               0.00000
Quantity            0.00000
Discount            0.00000
Profit              0.00000
Shipping Cost       0.00000
Order Priority      0.00000
dtype: float64
```

Show Over ALl Stastistics of your dataset

```
df.describe() # for Numerical data only


# .describe() returns the stastistic information
# By default it works on Numerical data
```

```
             Row ID   Postal Code          Sales      Quantity
Discount  \
count  51290.00000   9994.000000  51290.000000  51290.000000
51290.000000
mean   25645.50000  55190.379428    246.490581      3.476545
0.142908
std    14806.29199  32063.693350    487.565361      2.278766
0.212280
min        1.00000   1040.000000      0.444000      1.000000
0.000000
25%    12823.25000  23223.000000     30.758625      2.000000
0.000000
50%    25645.50000  56430.500000     85.053000      3.000000
0.000000
75%    38467.75000  90008.000000    251.053200      5.000000
0.200000
max    51290.00000  99301.000000  22638.480000     14.000000
0.850000

              Profit  Shipping Cost
count  51290.000000   51290.000000
mean      28.610982      26.375915
std      174.340972      57.296804
min    -6599.978000       0.000000
25%        0.000000       2.610000
```

```
50%        9.240000       7.790000
75%       36.810000      24.450000
max     8399.976000     933.570000


# To show the Stastistics of all
# for all data types we can use as below code

df.describe(include='all')
```

```
              Row ID        Order ID  Order Date   Ship Date       Ship
Mode  \
count   51290.00000           51290       51290       51290
51290
unique          NaN           25035        1430        1464
4
top             NaN  CA-2014-100111  18-06-2014  22-11-2014  Standard
Class
freq            NaN              14         135         130
30775
mean    25645.50000             NaN         NaN         NaN
NaN
std     14806.29199             NaN         NaN         NaN
NaN
min         1.00000             NaN         NaN         NaN
NaN
25%     12823.25000             NaN         NaN         NaN
NaN
50%     25645.50000             NaN         NaN         NaN
NaN
75%     38467.75000             NaN         NaN         NaN
NaN
max     51290.00000             NaN         NaN         NaN
NaN

       Customer ID    Customer Name   Segment           City
State  ...  \
count        51290            51290     51290          51290
51290  ...
unique        1590              795         3           3636
1094  ...
top       PO-18850  Muhammed Yedwab  Consumer  New York City
California  ...
freq            97              108     26518            915
2001  ...
mean           NaN              NaN       NaN            NaN
NaN  ...
std            NaN              NaN       NaN            NaN
NaN  ...
min            NaN              NaN       NaN            NaN
```

```
NaN  ...
25%            NaN               NaN       NaN             NaN
NaN  ...
50%            NaN               NaN       NaN             NaN
NaN  ...
75%            NaN               NaN       NaN             NaN
NaN  ...
max            NaN               NaN       NaN             NaN
NaN  ...

              Product ID         Category Sub-Category Product Name  \
count              51290            51290        51290        51290
unique             10292                3           17         3788
top     OFF-AR-10003651  Office Supplies      Binders      Staples
freq                  35            31273         6152          227
mean                 NaN              NaN          NaN          NaN
std                  NaN              NaN          NaN          NaN
min                  NaN              NaN          NaN          NaN
25%                  NaN              NaN          NaN          NaN
50%                  NaN              NaN          NaN          NaN
75%                  NaN              NaN          NaN          NaN
max                  NaN              NaN          NaN          NaN

                Sales      Quantity      Discount        Profit
Shipping Cost  \
count   51290.000000  51290.000000  51290.000000  51290.000000
51290.000000
unique           NaN           NaN           NaN           NaN
NaN
top              NaN           NaN           NaN           NaN
NaN
freq             NaN           NaN           NaN           NaN
NaN
mean      246.490581      3.476545      0.142908     28.610982
26.375915
std       487.565361      2.278766      0.212280    174.340972
57.296804
min         0.444000      1.000000      0.000000  -6599.978000
0.000000
25%        30.758625      2.000000      0.000000      0.000000
2.610000
50%        85.053000      3.000000      0.000000      9.240000
7.790000
75%       251.053200      5.000000      0.200000     36.810000
24.450000
max     22638.480000     14.000000      0.850000   8399.976000
933.570000

        Order Priority
count            51290
```

```
unique                4
top              Medium
freq              29433
mean                NaN
std                 NaN
min                 NaN
25%                 NaN
50%                 NaN
75%                 NaN
max                 NaN

[11 rows x 24 columns]
```

Check Duplicate data if any

```
df.duplicated().any()

# .duplcated() is used to find the duplicate data in dataset
# .any() , we have used here to find any duplicate if any there is in
dataset

False
```

drop unneccessery colomns

```
# showing Columns

df.columns

Index(['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'],
      dtype='object')
```

drop columns

```
df= df.drop(['Row ID', 'Order ID','Customer ID','Postal Code'],
axis=1)

print('Columns Dropped !!')
```

```
# drop() is used to drop the data from dataset
# axis=1 is used to difine that we are dropping the columns , for Rows we can use axis=0
# we have passed the column names in List bcz we have to drop many columns
```

```
Columns Dropped !!
```

```
# showing columns
```

```
df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority'],
      dtype='object')
```

Task >>>

show Category wise Profit

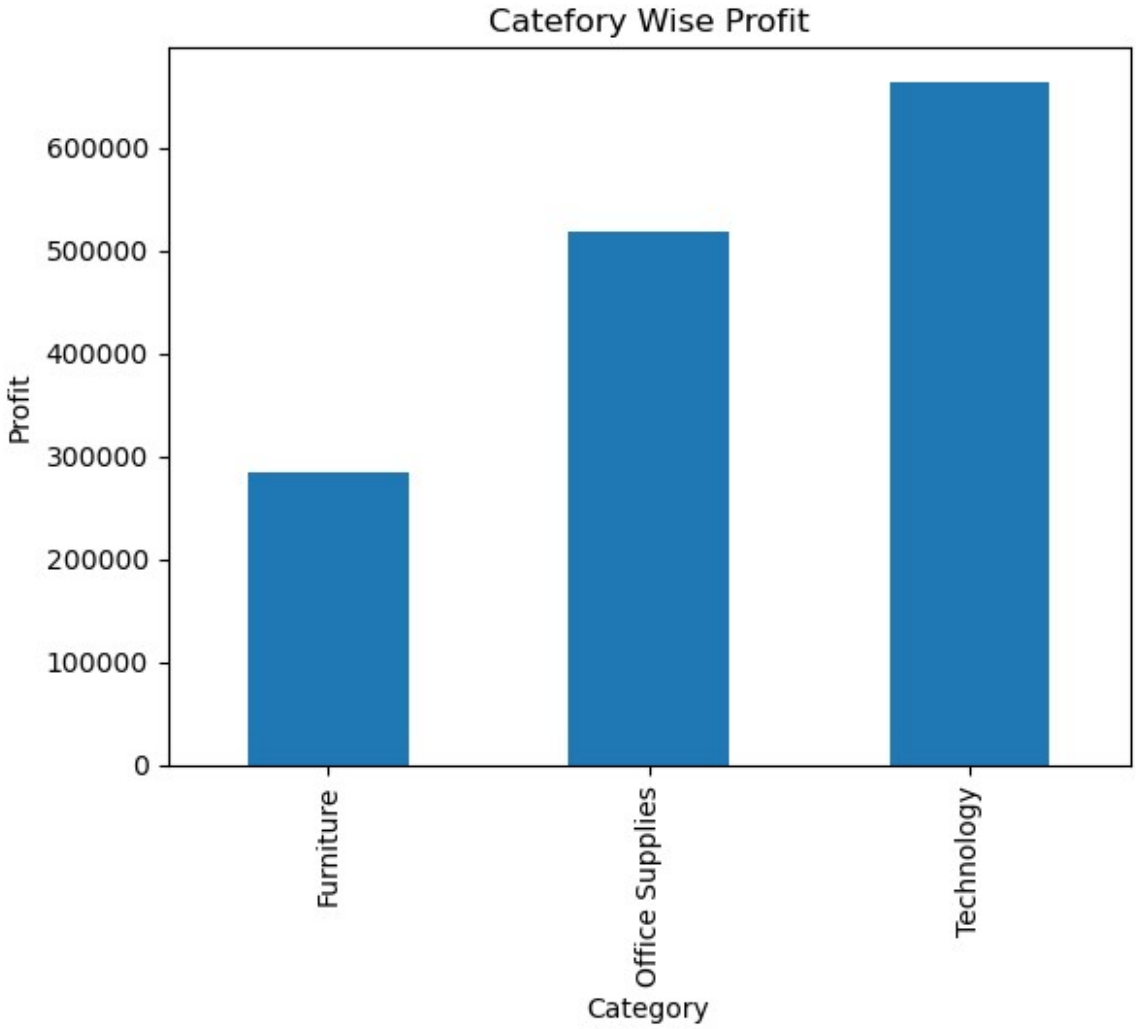
```
# showing columns
df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority'],
      dtype='object')
```

```
cat_profit = df.groupby('Category')['Profit'].sum()

# show result
cat_profit


# .groupby() is used to performing the group wise aggregation,
# .sum() returns the total
```

```
Category
Furniture          285204.72380
Office Supplies    518473.83430
Technology         663778.73318
Name: Profit, dtype: float64
```

```
cat_profit.plot(kind='bar')
plt.title('Catefory Wise Profit')
plt.xlabel('Category')
plt.ylabel('Profit')
plt.show()
```



Show Region wise sales

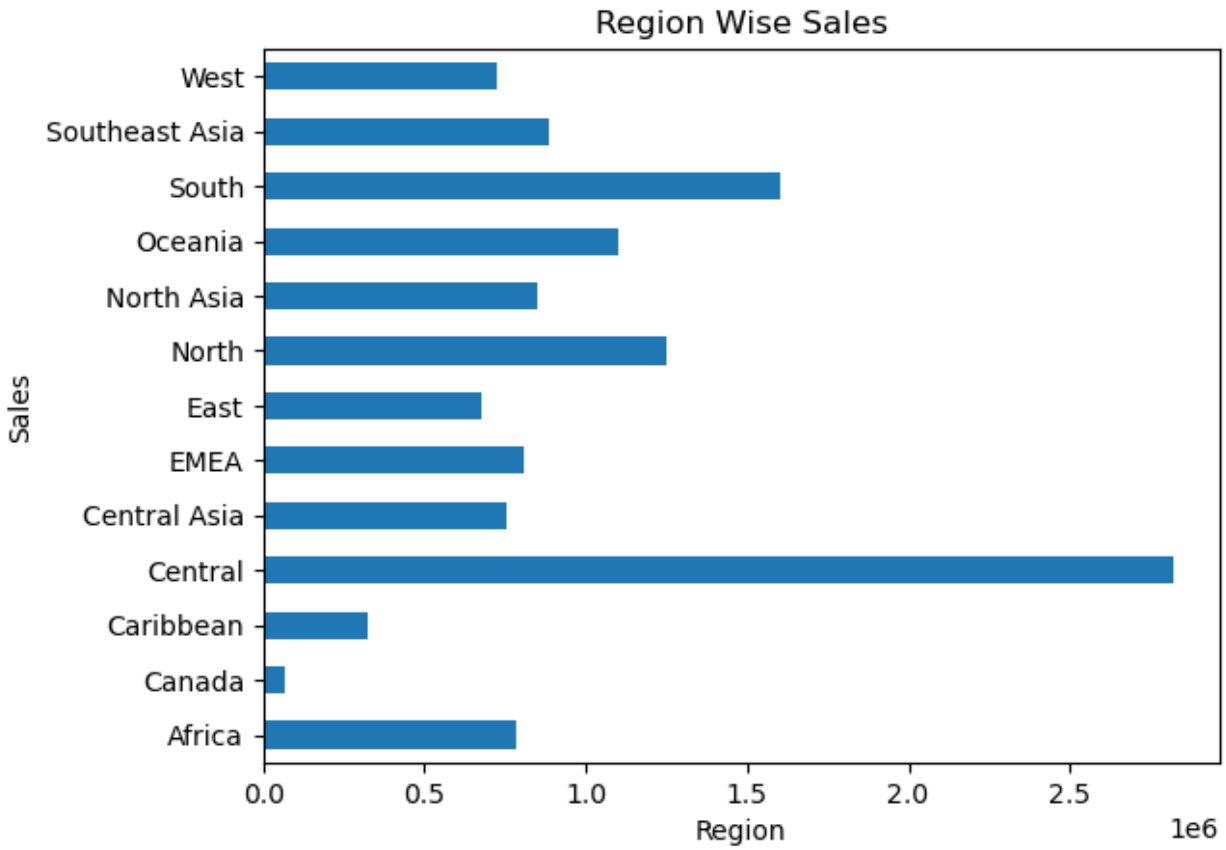
```
# showing columns

df.columns

Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
```

```
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority'],
      dtype='object')
```

```
reg_sales = df.groupby('Region')['Sales'].sum()


# .groupby() is used to performing the group wise aggregation,
# .sum() returns the total

# show result
reg_sales
```

```
Region
Africa             7.837732e+05
Canada             6.692817e+04
Caribbean          3.242809e+05
Central            2.822303e+06
Central Asia       7.528266e+05
EMEA               8.061613e+05
East               6.787812e+05
North              1.248166e+06
North Asia         8.483098e+05
Oceania            1.100185e+06
South              1.600907e+06
Southeast Asia     8.844232e+05
West               7.254578e+05
Name: Sales, dtype: float64
```

```
reg_sales.plot(kind='barh')
plt.title('Region Wise Sales')
plt.xlabel('Region')
plt.ylabel('Sales')
plt.show()
```



Show MOnthly Sales to find that which month has the highest sales

```
# showing columns

df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority'],
      dtype='object')
```

```
# showing top 5 rows of order date
df['Order Date'].head()
```

```
0    1/1/2011
1    1/1/2011
2    1/1/2011
3    1/1/2011
```

```
4    1/1/2011
Name: Order Date, dtype: object
```

```
# Create New column as Order_Month on the basis of Order date

df['Order_Mnth'] =  pd.DatetimeIndex(df['Order Date']).month

print('column created !!')
```

```
column created !!
```

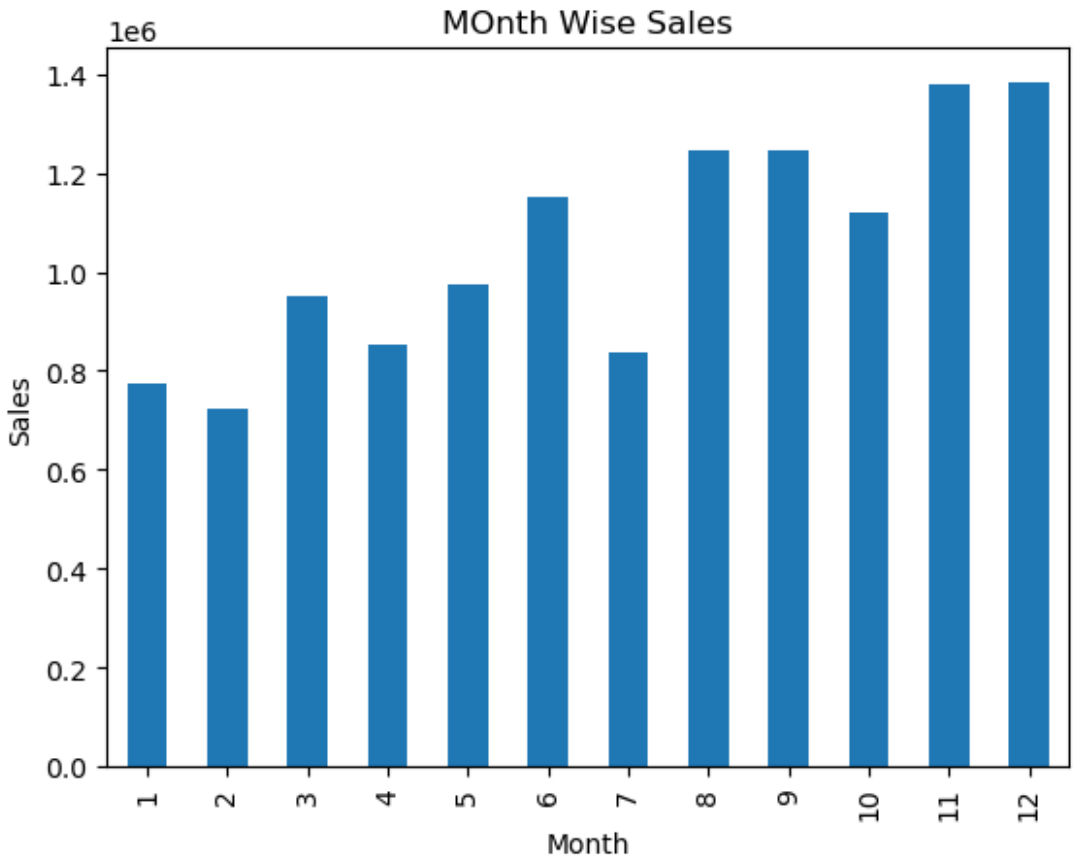
```
# showing column

df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth'],
      dtype='object')
```

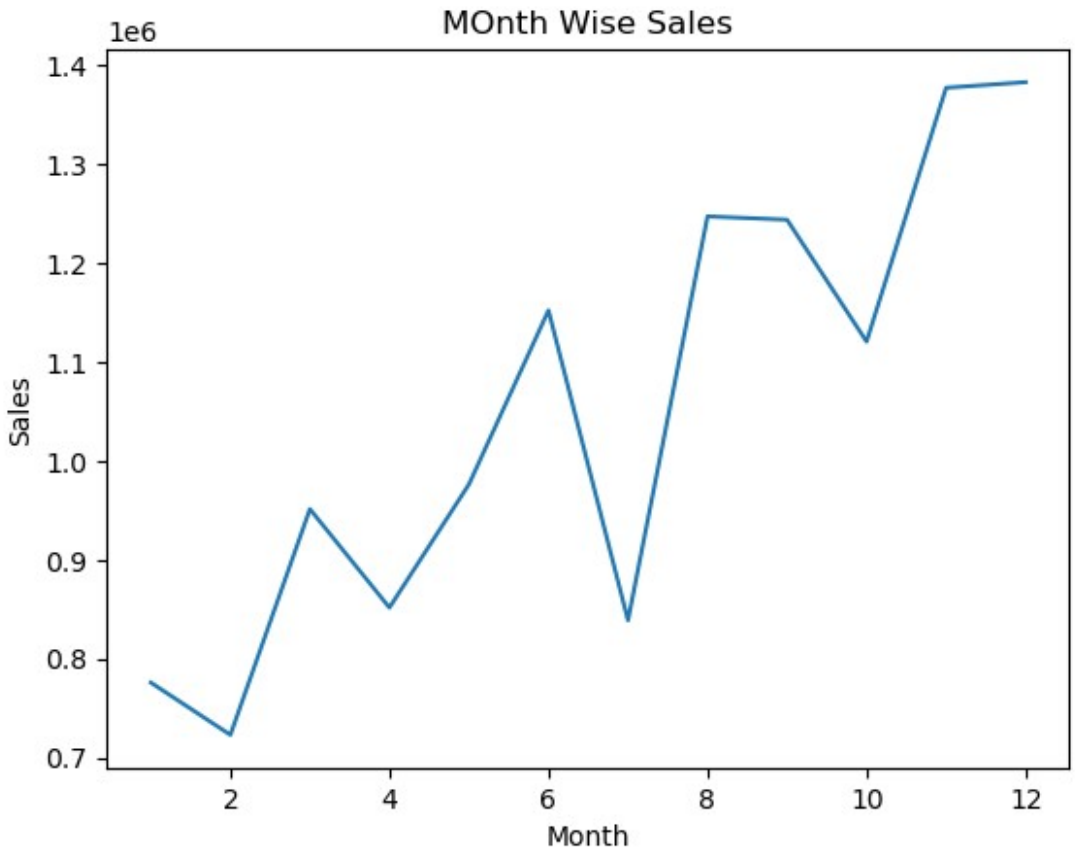
```
monthly_sales = df.groupby('Order_Mnth')['Sales'].sum()

# show Result
monthly_sales
```

```
Order_Mnth
1     7.757669e+05
2     7.228532e+05
3     9.513331e+05
4     8.516173e+05
5     9.764157e+05
6     1.152368e+06
7     8.387436e+05
8     1.247501e+06
9     1.244140e+06
10    1.120777e+06
11    1.377651e+06
12    1.383335e+06
Name: Sales, dtype: float64
```

```
monthly_sales.plot(kind='bar')
plt.title('MOnth Wise Sales')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.show()
```




```
monthly_sales.plot(kind='line')
plt.title('MOnth Wise Sales')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.show()
```



is WeekDay more profitable that Weekend

```
# Showing Columns

df.columns

Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth'],
      dtype='object')

df['Order_day'] = pd.DatetimeIndex(df['Order Date']).day_name()

print('Columns Craeted !!')

Columns Craeted !!
```

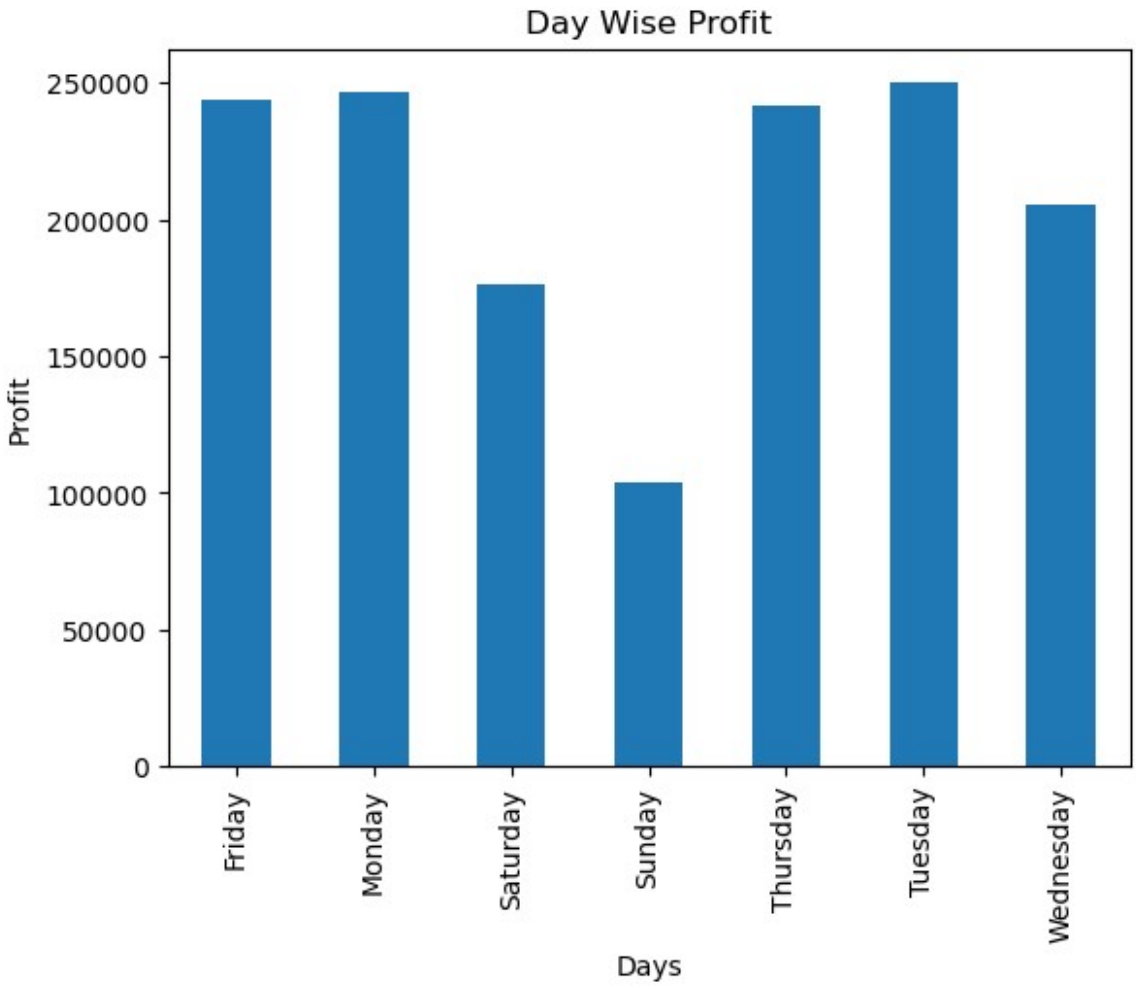
```
# Show the Order_DAy column Unique data only

df['Order_day'].unique()
```

```
array(['Saturday', 'Sunday', 'Monday', 'Tuesday', 'Thursday',
'Friday',
       'Wednesday'], dtype=object)
```

```
daily_profit = df.groupby('Order_day')['Profit'].sum()

# Show Result

daily_profit
```

```
Order_day
Friday       243802.83544
Monday       246526.55710
Saturday     176486.55222
Sunday       104117.90698
Thursday     241183.07994
Tuesday      249788.05098
Wednesday    205552.30862
Name: Profit, dtype: float64
```

```
daily_profit.plot(kind='bar')
plt.title('Day Wise Profit')
plt.xlabel('Days')
plt.ylabel('Profit')
plt.show()
```



SHow the top 10 Perfroming Product By Sales

```
# showing columns

df.columns
```

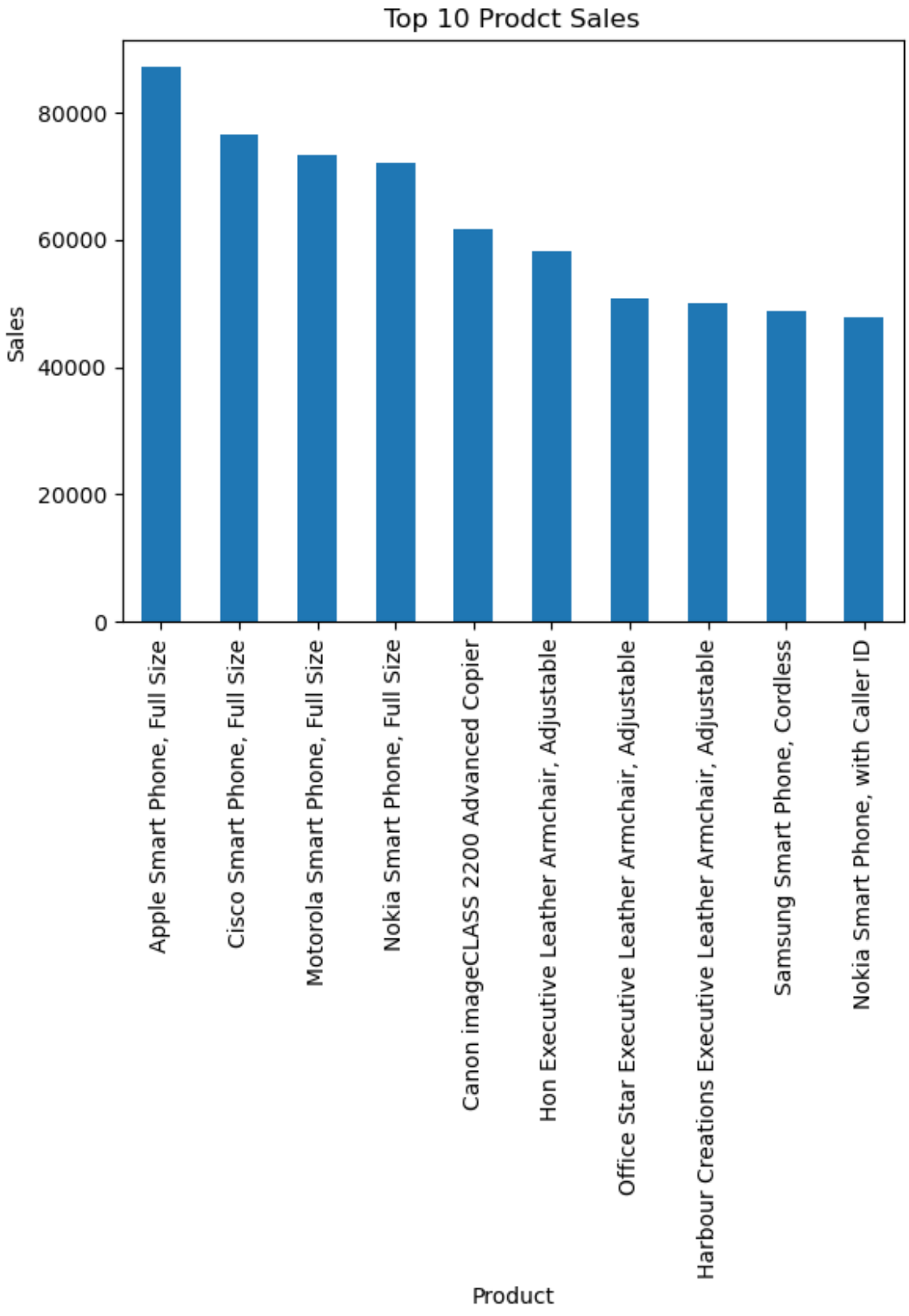
```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day'],
      dtype='object')
```

```
top10_prod_sales = df.groupby('Product Name')
['Sales'].sum().sort_values(ascending = False).head(10)
```

```
# show result
top10_prod_sales
```

```
Product Name
Apple Smart Phone, Full Size                                86935.7786
Cisco Smart Phone, Full Size                                76441.5306
Motorola Smart Phone, Full Size                             73156.3030
Nokia Smart Phone, Full Size                                71904.5555
Canon imageCLASS 2200 Advanced Copier                       61599.8240
Hon Executive Leather Armchair, Adjustable                  58193.4841
Office Star Executive Leather Armchair, Adjustable          50661.6840
Harbour Creations Executive Leather Armchair, Adjustable    50121.5160
Samsung Smart Phone, Cordless                               48653.4600
Nokia Smart Phone, with Caller ID                           47877.7857
Name: Sales, dtype: float64
```

```
# Setting up the graph & its type
top10_prod_sales.plot(kind='bar') # Setting up the Plot Type = Bar

plt.title('Top 10 Prodct Sales') # Setting up the Title
plt.xlabel('Product') # Setting up the X-label title
plt.ylabel('Sales') # Setting up the X-label title

# Show the graph
plt.show()
```


Top 10 Prodct Sales
Sales
80000
60000
40000
20000
0
Apple Smart Phone, Full Size
Cisco Smart Phone, Full Size
Motorola Smart Phone, Full Size
Nokia Smart Phone, Full Size
Canon imageCLASS 2200 Advanced Copier
Hon Executive Leather Armchair, Adjustable
Office Star Executive Leather Armchair, Adjustable
Harbour Creations Executive Leather Armchair, Adjustable
Samsung Smart Phone, Cordless
Nokia Smart Phone, with Caller ID
Product

How Many type sof Shipment available ?

```
# showung Columns

df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day'],
      dtype='object')
```

```
# showing Ship Mode data

df['Ship Mode']
```

```
0         Standard Class
1         Standard Class
2           Second Class
3           Second Class
4         Standard Class
               ...
51285     Standard Class
51286     Standard Class
51287       Second Class
51288     Standard Class
51289     Standard Class
Name: Ship Mode, Length: 51290, dtype: object
```

```
# Lets extract the Unique Ship Mode

df['Ship Mode'].unique().tolist()
```

```
['Standard Class', 'Second Class', 'First Class', 'Same Day']
```

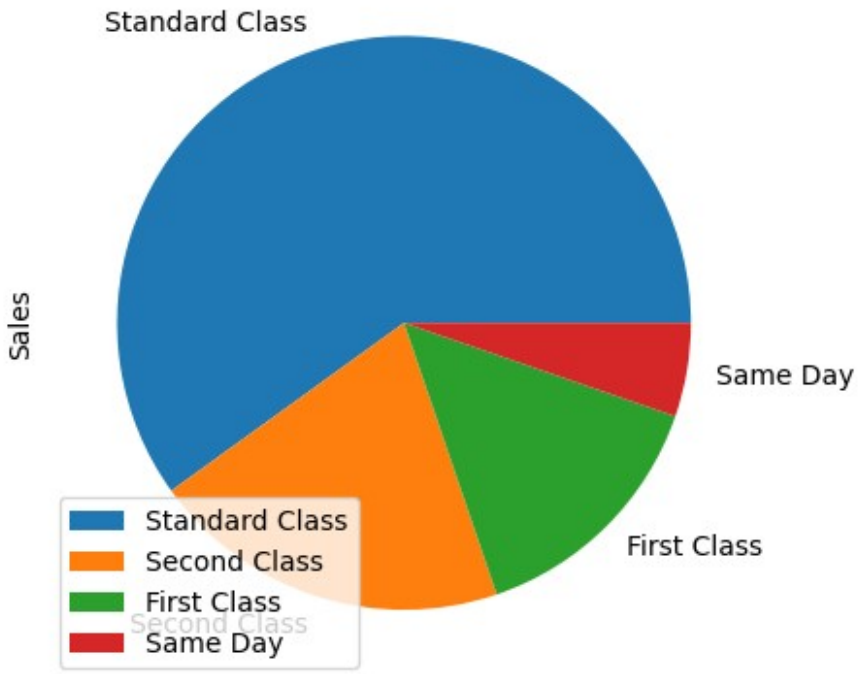
show Ship Mode Wise Total Sales

```
ship_mode_sales = df.groupby('Ship Mode')
['Sales'].sum().sort_values(ascending=False)

# show result
ship_mode_sales
```

```
Ship Mode
Standard Class   7578652.107
```

```
Second Class     2565671.681
First Class      1830976.138
Same Day          667201.984
Name: Sales, dtype: float64
```

```
ship_mode_sales.plot(kind='pie')

plt.legend()
plt.show()
```




```
# Showing Top 2 Rows

df.head(2)
```

```
  Order Date Ship Date       Ship Mode    Customer Name   Segment  \
0   1/1/2011  6/1/2011  Standard Class  Toby Braunhardt  Consumer
1   1/1/2011  8/1/2011  Standard Class      Joseph Holt  Consumer

          City            State    Country  Market   Region  ... Sub-
Category  \
0  Constantine      Constantine    Algeria  Africa   Africa  ...
Storage
1  Wagga Wagga  New South Wales  Australia    APAC  Oceania  ...
Supplies

               Product Name   Sales Quantity  Discount  Profit
Shipping Cost  \
```

```
0       Tenex Lockers, Blue 408.300         2     0.000 106.140
35.460
1  Acme Trimmer, High Speed 120.366         3     0.100  36.036
9.720

   Order Priority  Order_Mnth Order_day
0          Medium           1  Saturday
1          Medium           1  Saturday

[2 rows x 22 columns]
```

Show Sub_Category Wise Total Sales & Profit and Sold QTY

```
# Showing Columns
df.columns

Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day'],
      dtype='object')

# Grouping the data
df.groupby('Sub-Category')[['Sales','Profit','Quantity']].sum()

                    Sales     Profit  Quantity
Sub-Category
Accessories   749237.019 129626.306     10946
Appliances   1011064.305 141680.589      6078
Art           372091.966  57953.911     16301
Binders       461911.506  72449.846     21429
Bookcases    1466572.242 161924.419      8310
Chairs       1501681.764 140396.267     12336
Copiers      1509436.273 258567.548      7454
Envelopes     170904.302  29601.116      8380
Fasteners      83242.316  11525.424      8390
Furnishings   385578.256  46967.425     11225
Labels         73404.030  15010.512      9322
Machines      779060.067  58867.873      4906
Paper         244291.719  59207.683     12822
Phones       1706824.139 216717.006     11870
Storage      1127085.861 108461.490     16917
```

```
Supplies      243074.221  22583.263      8543
Tables        757041.924 -64083.389      3083
```

Show the List of Top Performing Customer who making More profit

```
df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day'],
      dtype='object')
```

```
top_customer = df.groupby('Customer Name')
['Profit'].sum().sort_values(ascending= False).head(10)

# show Result
top_customer
```

```
Customer Name
Tamara Chand      8672.899
Raymond Buch      8453.050
Sanjit Chand      8205.380
Hunter Lopez      7816.568
Bill Eplett       7410.005
Harry Marie       6958.286
Susan Pistek      6484.407
Mike Gockenbach   6458.676
Adrian Barton     6417.284
Tom Ashbrook      6311.979
Name: Profit, dtype: float64
```

Show Yearly Sales Profit & Total Sold Qty

```
df.columns
```

```
Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
```

```
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day'],
      dtype='object')

# Creating Year column


df['Order_Year'] =  pd.DatetimeIndex(df['Order Date']).year
print('column created')

column created

df.columns

Index(['Order Date', 'Ship Date', 'Ship Mode', 'Customer Name',
'Segment',
       'City', 'State', 'Country', 'Market', 'Region', 'Product ID',
       'Category', 'Sub-Category', 'Product Name', 'Sales',
'Quantity',
       'Discount', 'Profit', 'Shipping Cost', 'Order Priority',
'Order_Mnth',
       'Order_day', 'Order_Year'],
      dtype='object')

yearly_report = df.groupby('Order_Year')
[['Sales','Profit','Quantity']].sum()

# show result
yearly_report

                  Sales     Profit  Quantity
Order_Year
2011       2259450.896 248940.812     31443
2012       2677438.694 307415.279     38111
2013       3405746.449 406935.230     48136
2014       4299865.871 504165.970     60622
```