Exploratory Data Analysis on Global Superstore

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

import warnings
warnings.filterwarnings("ignore")
```

```
df = pd.read_csv('Global_SuperStore.csv', encoding= 'latin1')
#we use encoding= 'latin1' for handling text data
df.head()
```

```
   Row ID         Order ID  Order Date   Ship Date     Ship Mode Customer ID
\
0   32298   CA-2012-124891  31-07-2012  31-07-2012      Same Day    RH-19495
1   26341    IN-2013-77878  05-02-2013  07-02-2013  Second Class    JR-16210
2   25330    IN-2013-71249  17-10-2013  18-10-2013   First Class    CR-12730
3   13524  ES-2013-1579342  28-01-2013  30-01-2013   First Class    KM-16375
4   47221     SG-2013-4320  05-11-2013  06-11-2013      Same Day     RH-9495

      Customer Name      Segment           City            State  ...  \
0       Rick Hansen     Consumer  New York City         New York  ...
1     Justin Ritter    Corporate     Wollongong  New South Wales  ...
2      Craig Reiter     Consumer       Brisbane       Queensland  ...
3  Katherine Murray  Home Office         Berlin           Berlin  ...
4       Rick Hansen     Consumer          Dakar            Dakar  ...

         Product ID    Category Sub-Category  \
0   TEC-AC-10003033  Technology  Accessories
1   FUR-CH-10003950   Furniture       Chairs
2   TEC-PH-10004664  Technology       Phones
3   TEC-PH-10004583  Technology       Phones
4  TEC-SHA-10000501  Technology      Copiers

                                        Product Name     Sales Quantity  \
0  Plantronics CS510 - Over-the-Head monaural Wir...  2309.650        7
1          Novimex Executive Leather Armchair, Black  3709.395        9
2                  Nokia Smart Phone, with Caller ID  5175.171        9
3                     Motorola Smart Phone, Cordless  2892.510        5
4                     Sharp Wireless Fax, High-Speed  2832.960        8

  Discount    Profit  Shipping Cost  Order Priority
0      0.0  762.1845         933.57        Critical
1      0.1 -288.7650         923.63        Critical
2      0.1  919.9710         915.49          Medium
3      0.1  -96.5400         910.16          Medium
```

```
4      0.0  311.5200         903.04        Critical

[5 rows x 24 columns]
```

```
df.shape
```

```
(51290, 24)
```

A). Data Preprocessing

1). Check Null-values

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

```
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
```

```
df['Postal Code']
```

```
0        10024.0
1            NaN
2            NaN
3            NaN
4            NaN
          ...
51285        NaN
51286    77095.0
```

```
51287    93030.0
51288        NaN
51289        NaN
Name: Postal Code, Length: 51290, dtype: float64
```

```
print(f'Postal Code contains {41296*100/51290} % null values')
```

```
Postal Code contains 80.51472021836615 % null values
```

Inference

We can drop the Postal Code column as it contains null values more than 80%.

```
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')
```

```
df.drop('Postal Code',axis=1, inplace=True)
df.columns
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', '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')
```

2). Check Datatype

```
df.dtypes
```

```
Row ID             int64
Order ID          object
Order Date        object
Ship Date         object
Ship Mode         object
Customer ID       object
Customer Name     object
Segment           object
City              object
State             object
Country           object
Market            object
Region            object
Product ID        object
Category          object
Sub-Category      object
Product Name      object
```

```
Sales             float64
Quantity            int64
Discount          float64
Profit            float64
Shipping Cost     float64
Order Priority     object
dtype: object
```

3). Check Duplicates

```
df.duplicated().sum()
```

```
0
```

4). Extract all categorical columns and numerical columns

```
cat_cols=df.select_dtypes(include='object').columns
num_cols=df.select_dtypes(exclude='object').columns
print(cat_cols)
print(num_cols)
```

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

B). Group By Operations based on EDA

1). Find the Sub Category wise sum of Sales and depict it on the bar chart and depict highest and lowest shipping cost with diffference colors.

```
df.columns
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', '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')
```

```
a1=df.groupby(['Sub-Category'])['Sales'].sum()
a1
```

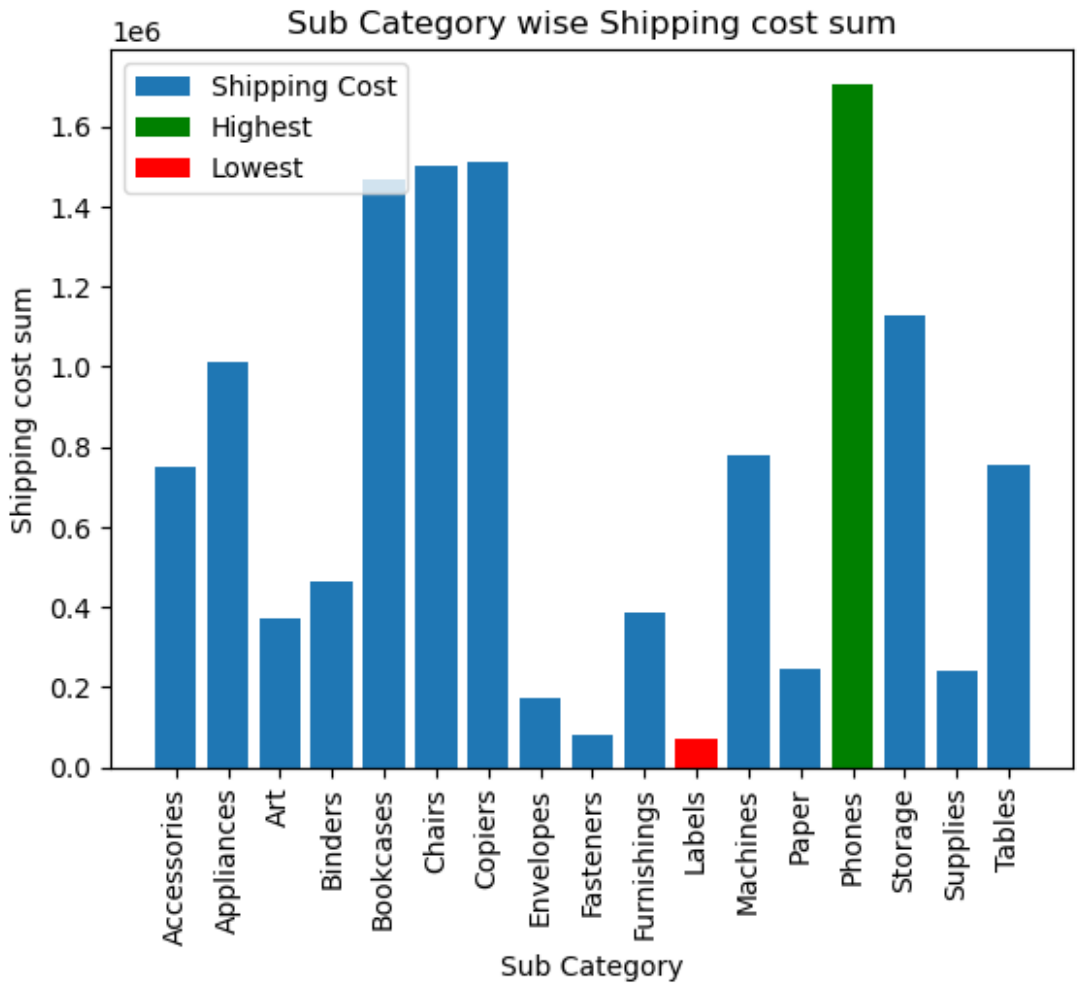
```
Sub-Category
Accessories    7.492370e+05
Appliances     1.011064e+06
Art            3.720920e+05
Binders        4.619115e+05
Bookcases      1.466572e+06
```

```
Chairs         1.501682e+06
Copiers        1.509436e+06
Envelopes      1.709043e+05
Fasteners      8.324232e+04
Furnishings    3.855783e+05
Labels         7.340403e+04
Machines       7.790601e+05
Paper          2.442917e+05
Phones         1.706824e+06
Storage        1.127086e+06
Supplies       2.430742e+05
Tables         7.570419e+05
Name: Sales, dtype: float64
```

```
sc= a1.index.tolist()
s_sales=a1.values.tolist()
print(sc)
print(s_sales)
```

```
['Accessories', 'Appliances', 'Art', 'Binders', 'Bookcases', 'Chairs', 'Copie
rs', 'Envelopes', 'Fasteners', 'Furnishings', 'Labels', 'Machines', 'Paper',
'Phones', 'Storage', 'Supplies', 'Tables']
[749237.0185, 1011064.305, 372091.9659, 461911.5057, 1466572.2418, 1501681.76
42, 1509436.27328, 170904.3016, 83242.3159, 385578.2559, 73404.03, 779060.067
1, 244291.7194, 1706824.1392, 1127085.8614, 243074.2206, 757041.9244]
```

```
plt.bar(sc,s_sales,label='Shipping Cost')
plt.bar(sc[s_sales.index(max(s_sales))],max(s_sales),color='green',label='Hig
hest')
plt.bar(sc[s_sales.index(min(s_sales))],min(s_sales),color='red',label='Lowes
t')
plt.legend()
plt.title('Sub Category wise Shipping cost sum')
plt.xlabel('Sub Category')
plt.ylabel('Shipping cost sum')
plt.legend(loc=2)
plt.xticks(rotation=90)
plt.show()
```

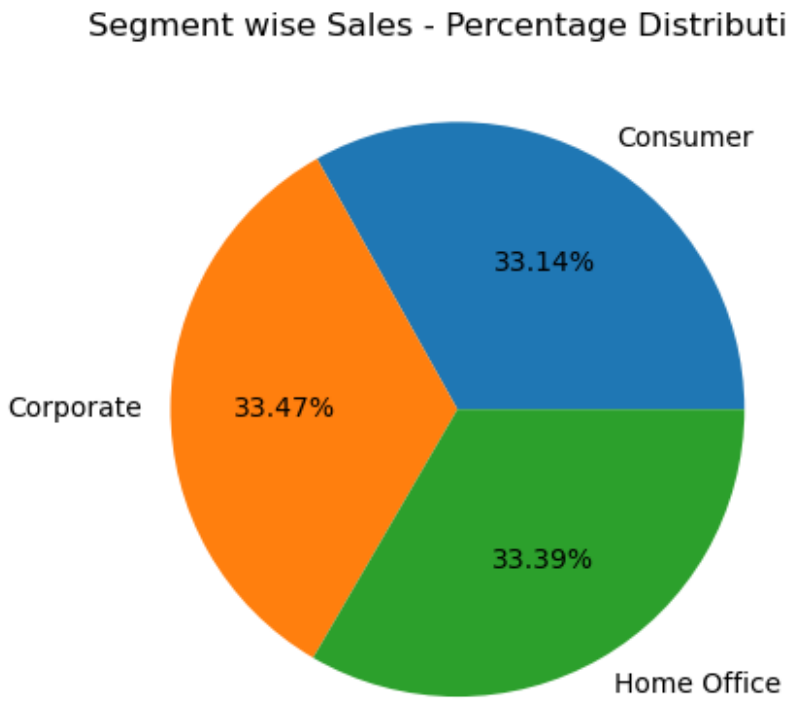


2) Find Segment wise mean of sales and depict it on a pie chart.

```
b2=df.groupby(['Segment'])['Sales'].mean()
b2
```

```
Segment
Consumer        245.416299
Corporate       247.890176
Home Office     247.228403
Name: Sales, dtype: float64
```

```
plt.pie(b2.values, labels=b2.index,autopct='%.2f%%')
plt.title("Segment wise Sales - Percentage Distribution")
plt.show()
```

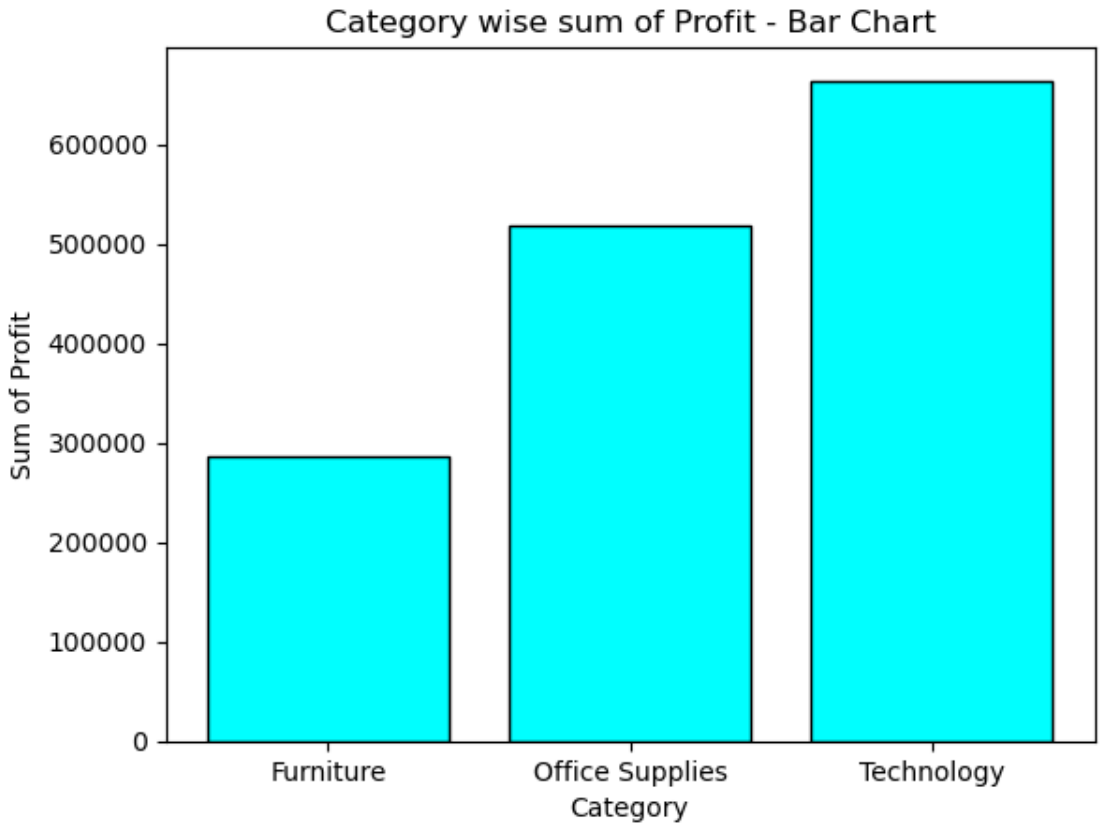


3). Find Category wise sum of profit and depict it on a bar chart and pie chart.

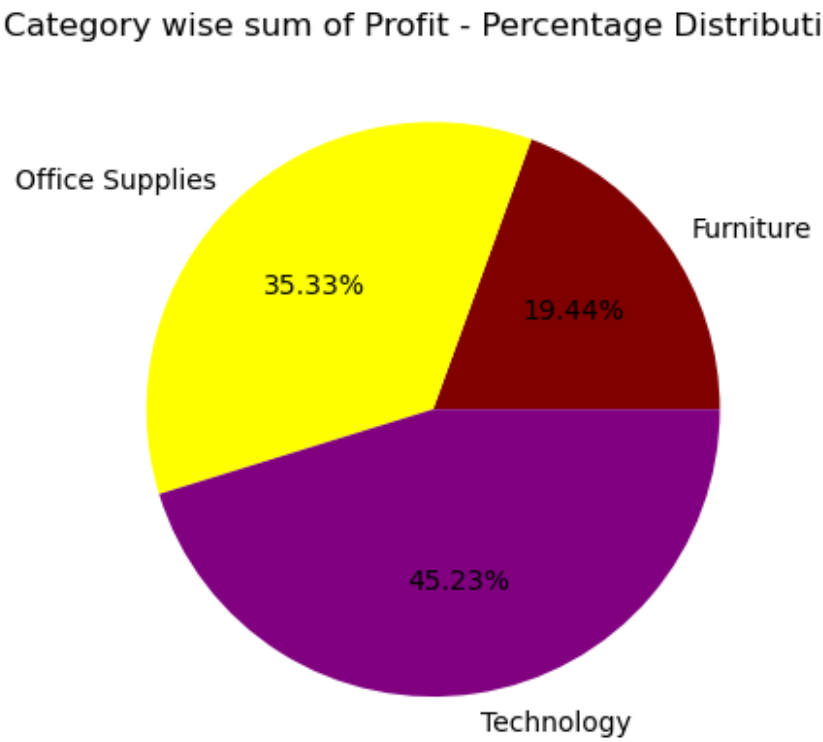
```
b3=df.groupby(['Category'])['Profit'].sum()
b3
```

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

```
plt.bar(b3.index, b3.values,color='cyan',edgecolor= 'black')
plt.title("Category wise sum of Profit - Bar Chart")
plt.xlabel('Category')
plt.ylabel('Sum of Profit')
plt.show()
```




```
plt.pie(b3.values, labels=b3.index,colors=['maroon','yellow','purple'],autopc
t='%.2f%%')
plt.title("Category wise sum of Profit - Percentage Distribution")
plt.show()
```

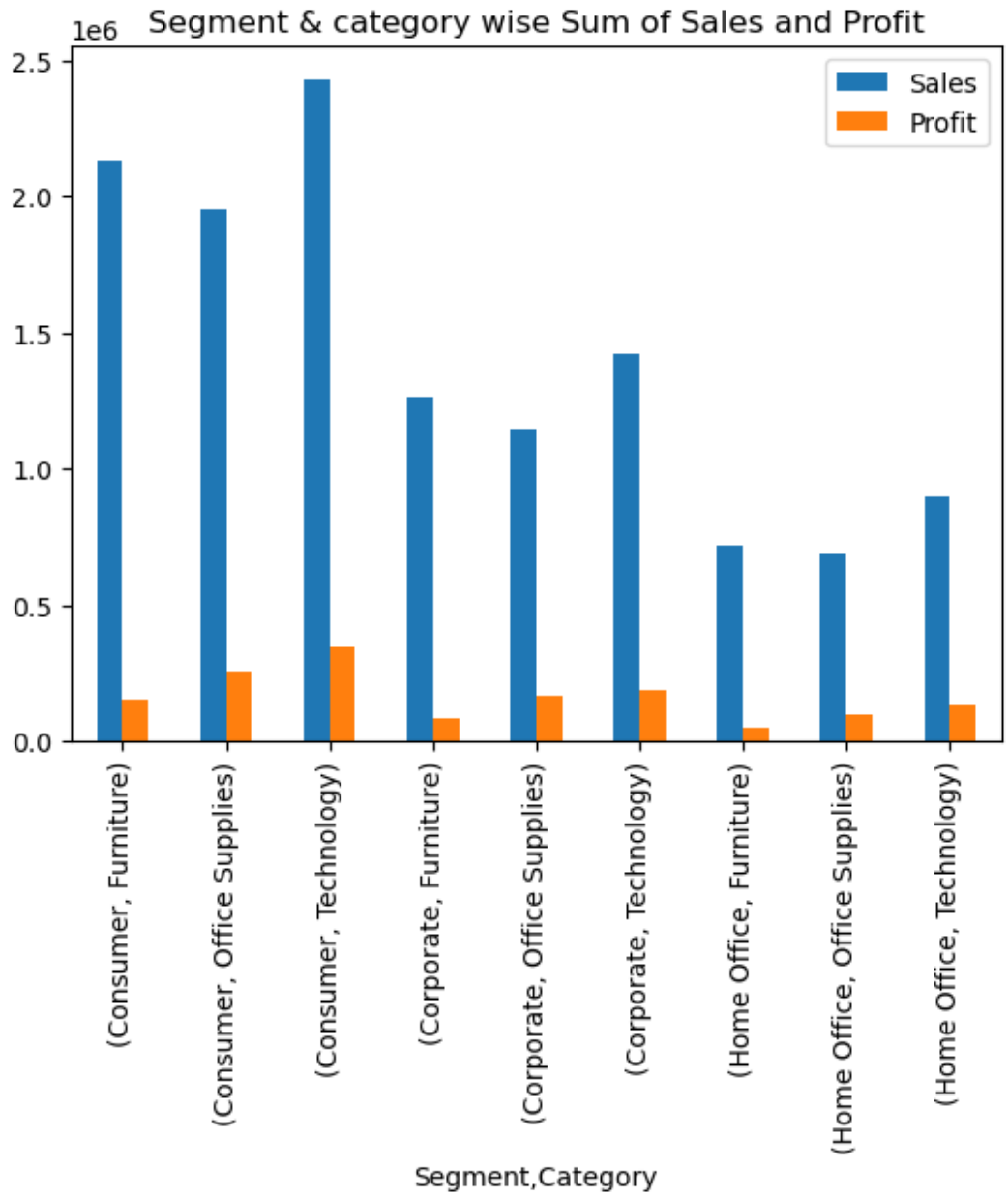


4). Find Segment and Category wise sum of sales and sum of profit and depict it on a clustered bar chart.

```
d4=df.groupby(['Segment','Category'])['Sales','Profit'].sum()
d4
```

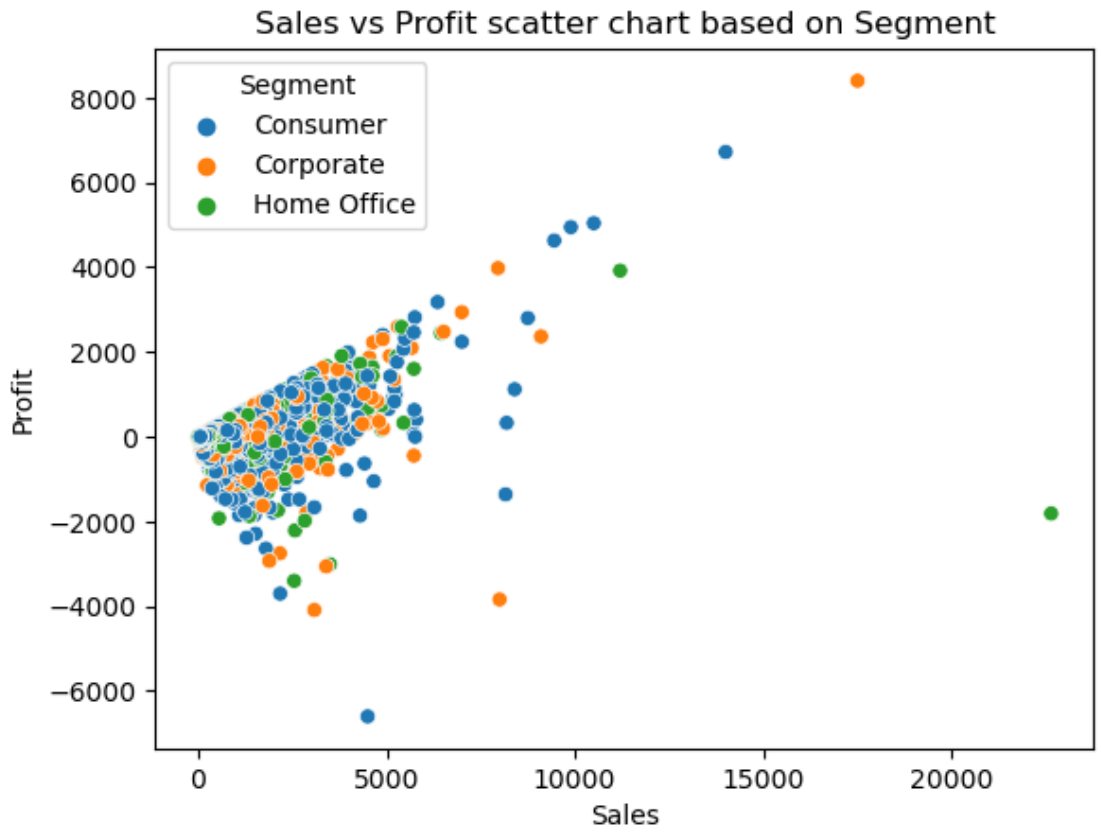
		Sales	Profit
Segment	Category		
Consumer	Furniture	2.128396e+06	153734.39350
	Office Supplies	1.952514e+06	253059.98460
	Technology	2.427040e+06	342445.40396
Corporate	Furniture	1.264520e+06	83731.91800
	Office Supplies	1.142386e+06	167581.22530
	Technology	1.417791e+06	189895.18536
Home Office	Furniture	7.179586e+05	47738.41230
	Office Supplies	6.921702e+05	97832.62440
	Technology	8.997261e+05	131438.14386

```
d4.plot(kind='bar')
plt.title('Segment & category wise Sum of Sales and Profit')
plt.xticks(rotation=90)
plt.show()
```



5). Depict Sales and profit in a Scatter chart based on Segment.

```
sns.scatterplot(x=df['Sales'],y=df['Profit'],hue=df['Segment'])
plt.title('Sales vs Profit scatter chart based on Segment')
plt.show()
```

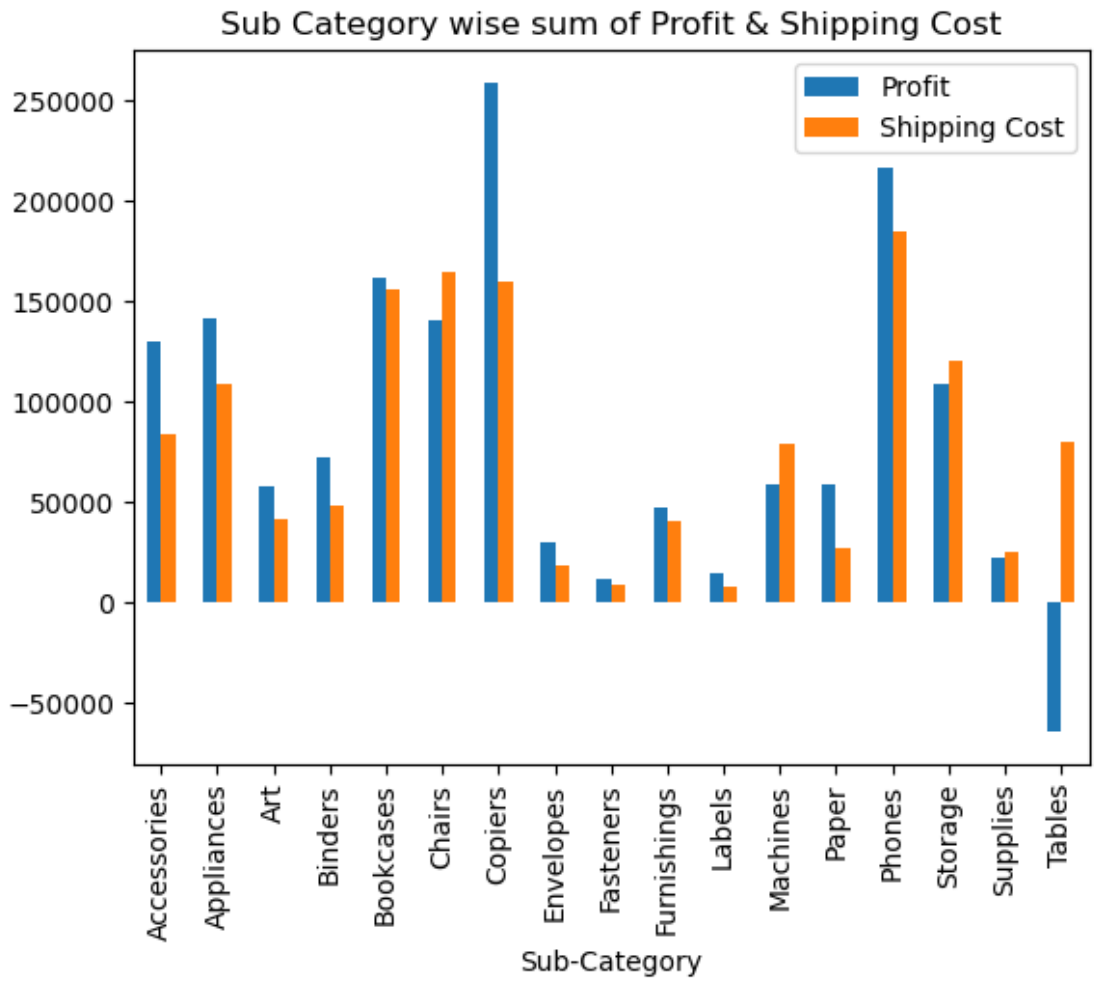


6). Find sum of Profit and Shipping cost based on Segment and display it on a Stacked bar chart

```
b5=df.groupby(['Sub-Category'])[['Profit','Shipping Cost']].sum()
b5
```

Sub-Category	Profit	Shipping Cost
Accessories	129626.30620	83513.3340
Appliances	141680.58940	108300.5860
Art	57953.91090	41287.1420
Binders	72449.84600	48181.7120
Bookcases	161924.41950	155481.9670
Chairs	140396.26750	164229.3520
Copiers	258567.54818	159496.2049
Envelopes	29601.11630	18547.4880
Fasteners	11525.42410	9053.3380
Furnishings	46967.42550	40746.7660
Labels	15010.51200	8059.6750
Machines	58867.87300	79135.8485
Paper	59207.68270	26660.8450
Phones	216717.00580	184902.4920
Storage	108461.48980	120546.0320
Supplies	22583.26310	24811.5270
Tables	-64083.38870	79861.3940

```
b5.plot(kind='bar')
plt.title('Sub Category wise sum of Profit & Shipping Cost')
plt.show()
```



E). Creating New Columns

1). Create new columns Year,Quarter,Month and Day based on Order date.

```
df.dtypes
```

```
Row ID            int64
Order ID         object
Order Date       object
Ship Date        object
Ship Mode        object
Customer ID      object
Customer Name    object
Segment          object
City             object
State            object
```

```
Country            object
Market             object
Region             object
Product ID         object
Category           object
Sub-Category       object
Product Name       object
Sales             float64
Quantity            int64
Discount          float64
Profit            float64
Shipping Cost     float64
Order Priority     object
dtype: object
```

```
df['Order Date']= pd.to_datetime(df['Order Date'])
df.dtypes
```

```
Row ID                     int64
Order ID                  object
Order Date        datetime64[ns]
Ship Date                 object
Ship Mode                 object
Customer ID               object
Customer Name             object
Segment                   object
City                      object
State                     object
Country                   object
Market                    object
Region                    object
Product ID                object
Category                  object
Sub-Category              object
Product Name              object
Sales                    float64
Quantity                   int64
Discount                 float64
Profit                   float64
Shipping Cost            float64
Order Priority            object
dtype: object
```

```
df['Year']=df['Order Date'].dt.year
df['Qtr']=df['Order Date'].dt.quarter
df['Month']=df['Order Date'].dt.month
df['Day']=df['Order Date'].dt.day
df.head()
```

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

```
0   32298   CA-2012-124891 2012-07-31  31-07-2012      Same Day    RH-19495
1   26341    IN-2013-77878 2013-05-02  07-02-2013  Second Class    JR-16210
2   25330    IN-2013-71249 2013-10-17  18-10-2013   First Class    CR-12730
3   13524  ES-2013-1579342 2013-01-28  30-01-2013   First Class    KM-16375
4   47221     SG-2013-4320 2013-05-11  06-11-2013      Same Day     RH-9495

      Customer Name      Segment           City            State  ...  \
0       Rick Hansen     Consumer  New York City         New York  ...
1     Justin Ritter    Corporate     Wollongong  New South Wales  ...
2      Craig Reiter     Consumer       Brisbane       Queensland  ...
3  Katherine Murray  Home Office         Berlin           Berlin  ...
4       Rick Hansen     Consumer          Dakar            Dakar  ...

      Sales Quantity Discount    Profit Shipping Cost Order Priority  Year  \
0  2309.650        7      0.0  762.1845        933.57       Critical  2012
1  3709.395        9      0.1 -288.7650        923.63       Critical  2013
2  5175.171        9      0.1  919.9710        915.49         Medium  2013
3  2892.510        5      0.1  -96.5400        910.16         Medium  2013
4  2832.960        8      0.0  311.5200        903.04       Critical  2013

   Qtr  Month  Day
0    3      7   31
1    2      5    2
2    4     10   17
3    1      1   28
4    2      5   11

[5 rows x 27 columns]
```

E). EDA On Conditional/Filtered data plots

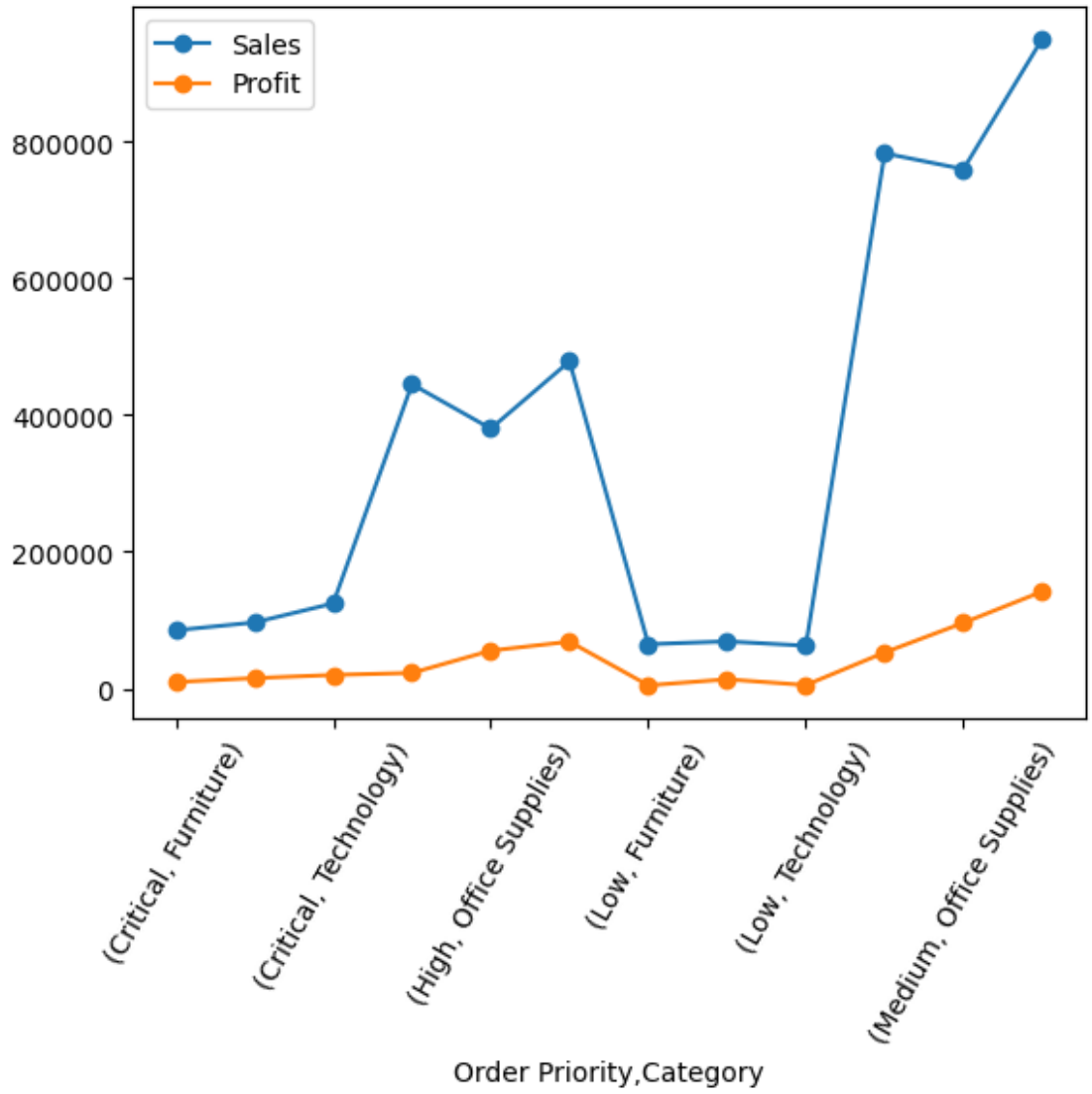
1). Find Order Priority and Category wise sum of Profit and Sales for the Year 2014 and depict it on a line chart and a bar chart.

```
c1= df[df['Year']==2014].groupby(['Order Priority','Category'])[['Sales','Profit']].sum()
c1
```

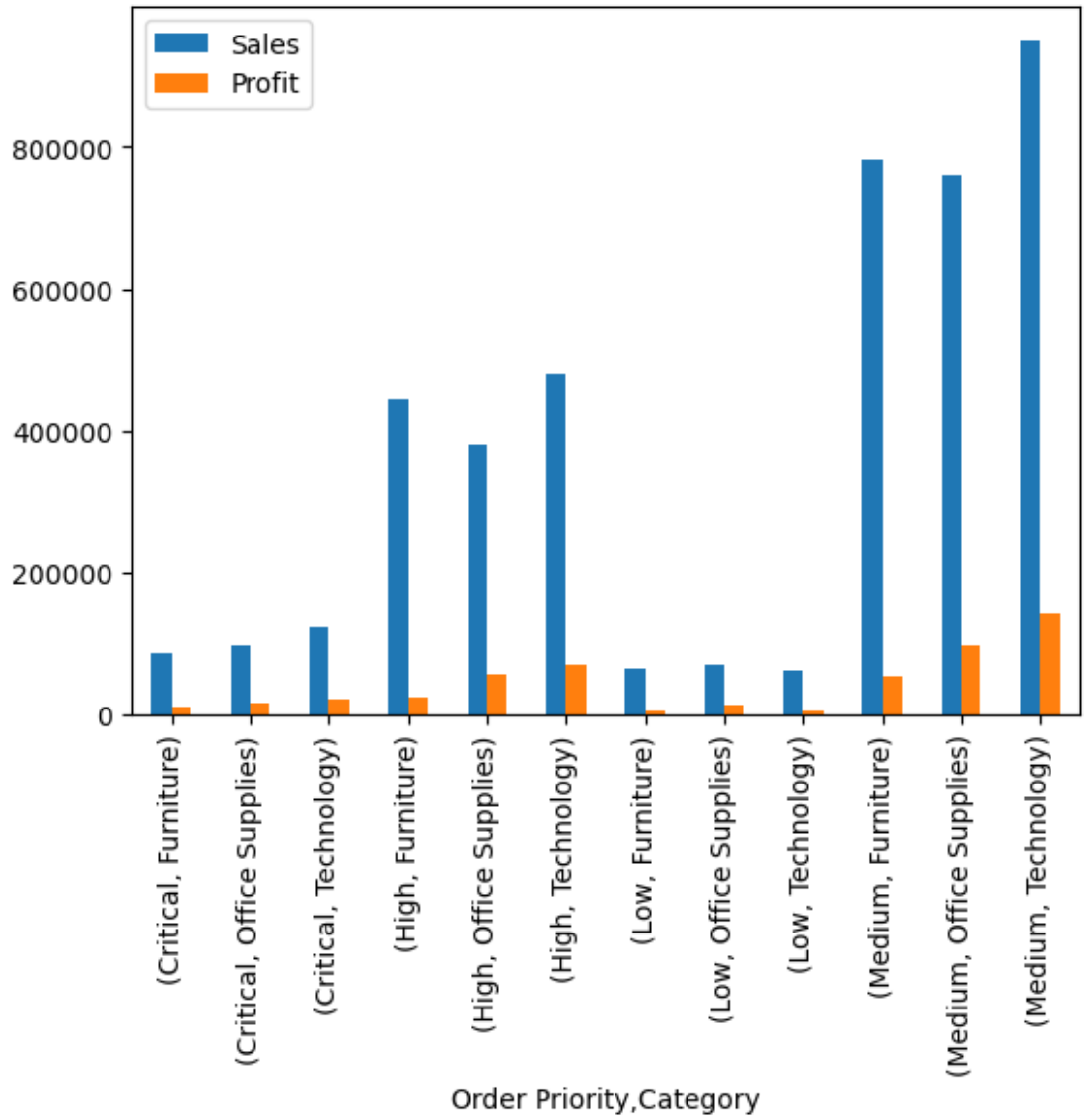
```
                                    Sales        Profit
Order Priority Category
Critical       Furniture         85052.56400    9378.68100
               Office Supplies   96599.38640   15107.06820
               Technology       124422.26100   19753.05740
High           Furniture        445196.84650   22687.80670
               Office Supplies  380209.23600   55363.11820
               Technology       478685.75262   68457.60102
Low            Furniture         64724.59080    4556.62740
               Office Supplies   68876.07050   13393.66080
               Technology        62548.54258    4920.15388
```

```
Medium          Furniture         783081.68950   52688.94780
                Office Supplies   759966.81350   96062.20920
                Technology        950502.11716  141797.03886
```

```
c1.plot(kind='line',marker='o')
plt.xticks(rotation=60)
plt.show()
```




```
c1.plot(kind='bar')
plt.xticks(rotation=90)
plt.show()
```



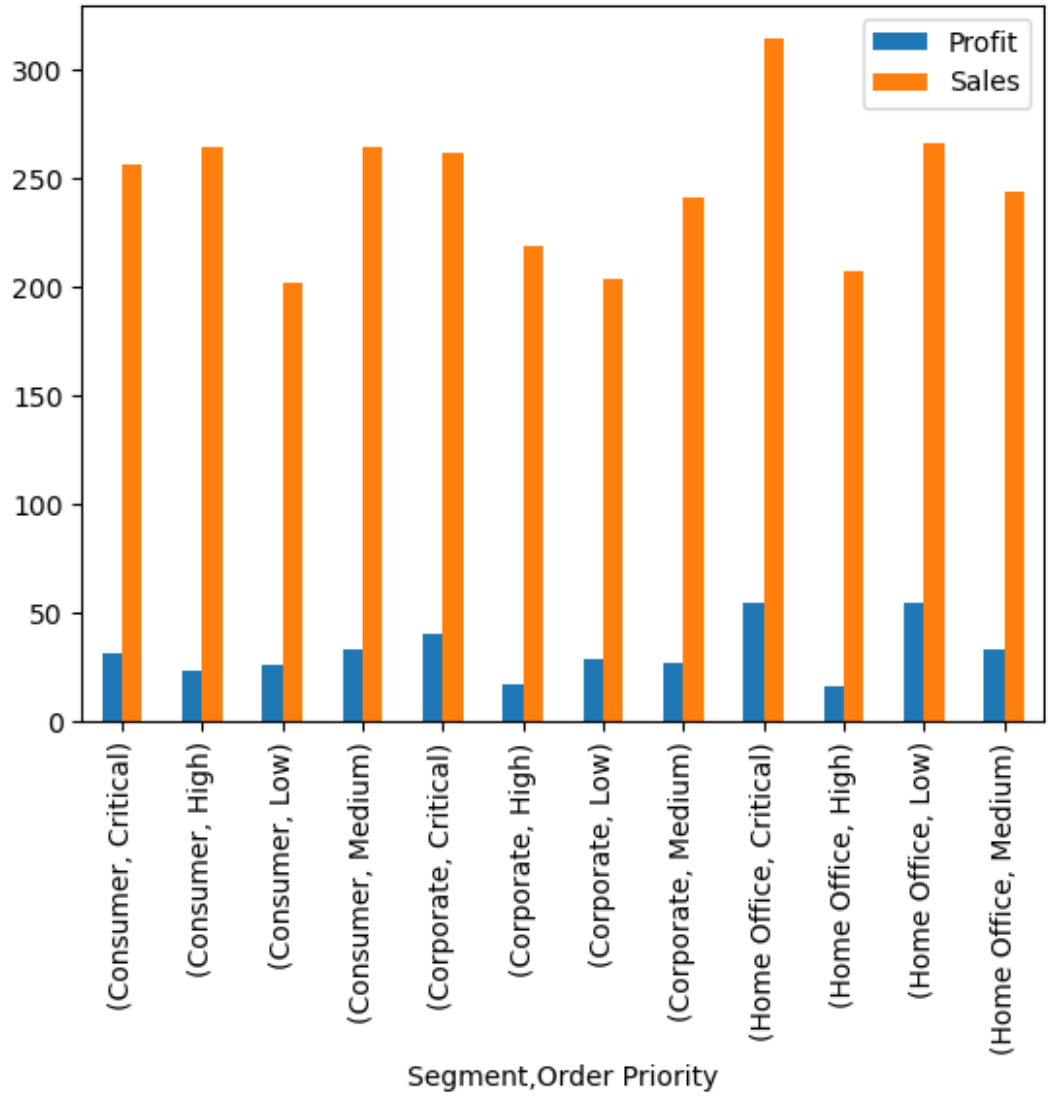
Q2). Find Segment and Order Priority wise sum of Profit and mean of sales for the month of December and depict on a clustered bar chart

```
c2=df[df['Month']==12].groupby(['Segment','Order Priority'])[['Profit','Sales
']].mean()
c2
```

Segment	Order Priority	Profit	Sales
Consumer	Critical	31.498554	257.066859
	High	23.937476	264.474709
	Low	26.709415	202.300394
	Medium	33.769130	264.739044
Corporate	Critical	40.750820	261.919750
	High	17.192785	219.531257
	Low	29.033422	204.060205

```
            Medium          27.469863  241.818547
Home Office Critical        55.128831  314.256471
            High            16.460995  207.233081
            Low             54.493600  266.589644
            Medium          33.375318  243.973392
```

```
c2.plot(kind='bar')
plt.xticks(rotation=90)
plt.show()
```



Q3). For the 'Corporate' Segment and 'Office Supplies' Category group the dataframe based on Year and find sum of Quantity,sum and mean of Discount, max profit.

```
c3 = df[(df['Category']=="Office Supplies") & (df['Segment']=="Corporate")].g
roupby(['Year']).agg(
{'Quantity':sum,'Discount':[sum,np.mean],'Profit':max})
```

```
c3
```

	Quantity	Discount		Profit
	sum	sum	mean	max
Year				
2011	5736	219.60	0.133902	2476.44
2012	7007	272.82	0.134859	1989.54
2013	8588	338.07	0.134850	1444.59
2014	10905	451.36	0.141315	3979.08

Q4) For the Order date between '2013-03-08' and between '2015-11-21', find the sum of Quantity, max discount, Mean Sales based on Category, Order Priority and Segment

```
c4 = df[df["Order Date"].between('2013-03-08','2015-11-21',inclusive=True)].g
roupby(['Category','Order Priority','Segment']).agg(
{'Quantity':sum,'Discount':max,'Sales':np.mean})
c4
```

			Quantity	Discount	Sales
Category	Order Priority	Segment			
Furniture	Critical	Consumer	618	0.7	356.530193
		Corporate	486	0.7	392.580107
		Home Office	284	0.6	412.247782
	High	Consumer	3054	0.8	404.531186
		Corporate	1876	0.7	416.740125
		Home Office	1273	0.7	404.789669
	Low	Consumer	391	0.7	425.010705
		Corporate	325	0.7	415.600220
		Home Office	242	0.7	491.326581
	Medium	Consumer	6101	0.8	424.003336
		Corporate	3547	0.8	412.820540
		Home Office	1948	0.8	388.229448
Office Supplies	Critical	Consumer	2449	0.8	113.849558
		Corporate	1381	0.8	131.433895
		Home Office	884	0.7	147.853150
	High	Consumer	9615	0.8	119.957403
		Corporate	5273	0.8	118.407848
		Home Office	3617	0.8	114.047460
	Low	Consumer	1494	0.8	112.385158
		Corporate	907	0.8	156.374730
		Home Office	600	0.8	85.155849
	Medium	Consumer	18526	0.8	118.215619
		Corporate	10667	0.8	119.711206
		Home Office	6948	0.8	124.429498
Technology	Critical	Consumer	816	0.7	453.354040
		Corporate	500	0.7	583.811480
		Home Office	274	0.7	455.095833
	High	Consumer	3329	0.7	470.731279

```
                    Corporate        1648       0.7  451.522301
                    Home Office      1128       0.7  508.254655
Low                 Consumer          385       0.7  426.104973
                    Corporate         338       0.7  453.576093
                    Home Office       200       0.7  397.810141
Medium              Consumer         5615       0.7  448.815381
                    Corporate        3637       0.7  488.145524
                    Home Office      2391       0.7  512.132876
```

Q5) For the Category Technology or Furniture and for the Quarter 4, depict Quantity vs segment on a boxplot

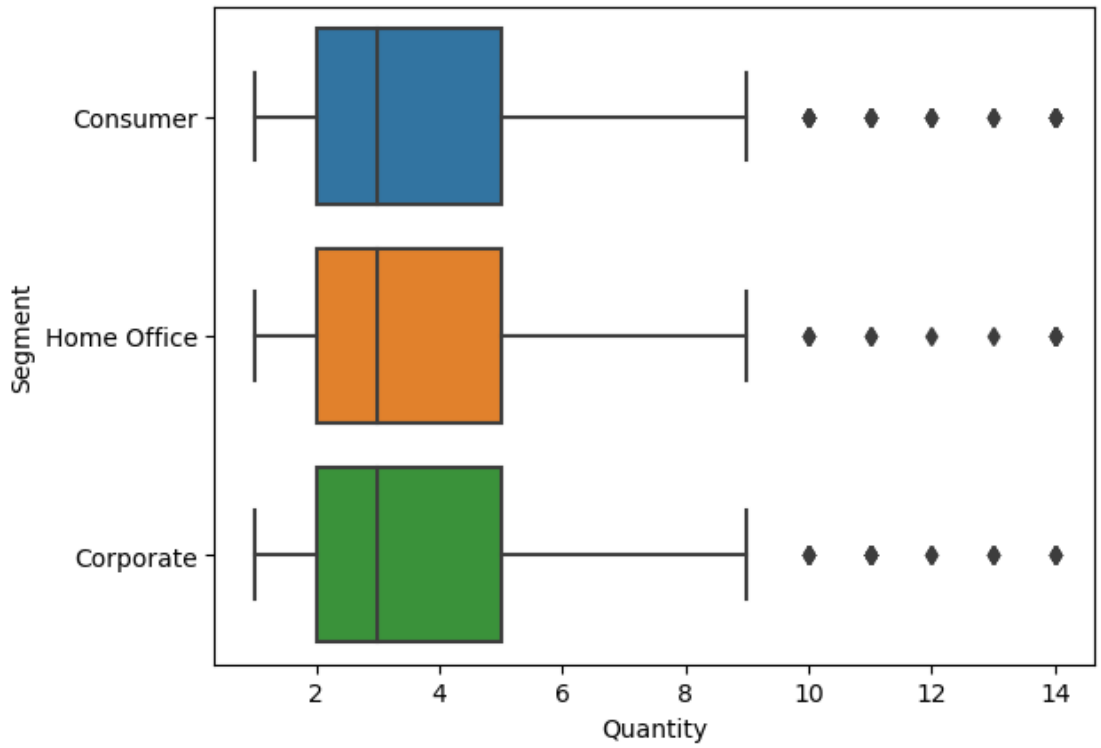
```
df['Qtr'].value_counts()
```

```
4    15661
3    13523
2    12329
1     9777
Name: Qtr, dtype: int64
```

```
c5 = df[(df['Category'].isin(['Technology','Furniture'])) & (df['Qtr']==4)]
c5.shape
```

```
(6205, 27)
```

```
sns.boxplot(x=c5['Quantity'],y=c5['Segment'])
plt.show()
```



6). EDA using Sub-plots

Q1) For 'Consumer' Segment and 'Medium' Order Priority, find Ship Mode wise mean of Sales and Profit

Create a 2x2 subplot depicting

a) Bar Chart and Line chart for Ship Mode vs Sales

b) Bar Chart and Line chart for Ship Mode vs Profit

c) Scatter Chart for Profit vs Sales

d) Line Chart for Ship Mode vs Profit and Ship Mode vs Sales

```
d1 = df[(df['Segment']=='Consumer') & (df['Order Priority']=='Medium')].group
by(['Ship Mode'])[['Sales','Profit']].mean()
d1
```

```
                     Sales     Profit
Ship Mode
First Class     250.842119  33.145865
Same Day        247.102313  38.281658
Second Class    236.487664  26.300652
Standard Class  245.587219  29.234086
```

```
fig,((ax1,ax2),(ax3,ax4))= plt.subplots(2,2,figsize=(10,8))

ax1.bar(d1.index,d1['Sales'],color='cyan')  # Bar chart for Sales vs Ship Mod
e
ax1.plot(d1.index,d1['Sales'],marker='*', color='red')    # Line chart for Sa
les vs Ship Mode
ax1.set_xlabel('Ship Mode')
ax1.set_ylabel('Sales')
ax1.set_title('Ship Mode vs Sales')

ax2.bar(d1.index,d1['Profit'],color='maroon')  # Bar chart for Profit vs Ship
Mode
ax2.plot(d1.index,d1['Profit'],marker="o")     # Line chart for Profit vs Shi
p Mode
ax2.set_xlabel('Ship Mode')
ax2.set_ylabel('Profit')
ax2.set_title('Ship Mode vs Profit')

ax3.scatter(d1['Sales'],d1['Profit'],marker='o',s=100,color='maroon')
ax3.set_xlabel('Sales')
ax3.set_ylabel('Profit')


ax4.plot(d1.index,d1['Sales'],marker="o",label="Sales")      # Line chart for
```

```
Sales vs Ship Mode
ax4.plot(d1.index,d1['Profit'],marker="*",label="Profit")   # Line chart for
Profit vs Ship Mode
ax4.legend()
ax4.set_xlabel('Ship Mode')
ax4.set_ylabel('Profit')

plt.show()
```