```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import style
%matplotlib inline
style.use("ggplot")

df = pd.read_excel ('superstore.xls')

print(df.head(5))
```

```
   Row ID        Order ID Order Date  Ship Date       Ship Mode
Customer ID  \
0       1  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-
12520
1       2  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-
12520
2       3  CA-2016-138688 2016-06-12 2016-06-16    Second Class    DV-
13045
3       4  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-
20335
4       5  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-
20335

     Customer Name    Segment        Country             City  ...  \
0      Claire Gute   Consumer  United States        Henderson  ...
1      Claire Gute   Consumer  United States        Henderson  ...
2  Darrin Van Huff  Corporate  United States      Los Angeles  ...
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...

  Postal Code  Region       Product ID         Category Sub-
Category  \
0       42420   South  FUR-BO-10001798        Furniture    Bookcases

1       42420   South  FUR-CH-10000454        Furniture       Chairs

2       90036    West  OFF-LA-10000240  Office Supplies       Labels

3       33311   South  FUR-TA-10000577        Furniture       Tables

4       33311   South  OFF-ST-10000760  Office Supplies      Storage


                                     Product Name     Sales
Quantity  \
0                   Bush Somerset Collection Bookcase  261.9600
2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400
3
```

```
2  Self-Adhesive Address Labels for Typewriters b...   14.6200
2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775
5
4                     Eldon Fold 'N Roll Cart System   22.3680
2
```

```
   Discount    Profit
0      0.00   41.9136
1      0.00  219.5820
2      0.00    6.8714
3      0.45 -383.0310
4      0.20    2.5164

[5 rows x 21 columns]
```

```
df.columns
```

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

```
column_list = list(df.columns)
for col in column_list[17:]:
    print(df[col].describe())
    print("\n")
```

```
count     9994.000000
mean       229.858001
std        623.245101
min          0.444000
25%         17.280000
50%         54.490000
75%        209.940000
max      22638.480000
Name: Sales, dtype: float64


count    9994.000000
mean        3.789574
std         2.225110
min         1.000000
25%         2.000000
50%         3.000000
75%         5.000000
max        14.000000
Name: Quantity, dtype: float64
```

```
count    9994.000000
mean        0.156203
std         0.206452
min         0.000000
25%         0.000000
50%         0.200000
75%         0.200000
max         0.800000
Name: Discount, dtype: float64
```

```
count    9994.000000
mean       28.656896
std       234.260108
min     -6599.978000
25%         1.728750
50%         8.666500
75%        29.364000
max      8399.976000
Name: Profit, dtype: float64
```

```
df.isna().sum()
```

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

```
import datetime as dt
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['Ship Date'] = pd.to_datetime(df['Ship Date'])

price_per_unit = df["Sales"] / df["Quantity"]
df["Price/Unit"] = price_per_unit

df.head()
```

```
   Row ID        Order ID Order Date  Ship Date       Ship Mode
Customer ID  \
0       1  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-
12520
1       2  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-
12520
2       3  CA-2016-138688 2016-06-12 2016-06-16    Second Class    DV-
13045
3       4  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-
20335
4       5  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-
20335

     Customer Name    Segment        Country             City  ...
Region  \
0      Claire Gute   Consumer  United States        Henderson  ...
South
1      Claire Gute   Consumer  United States        Henderson  ...
South
2  Darrin Van Huff  Corporate  United States      Los Angeles  ...
West
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...
South
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...
South

        Product ID         Category Sub-Category  \
0  FUR-BO-10001798        Furniture    Bookcases
1  FUR-CH-10000454        Furniture       Chairs
2  OFF-LA-10000240  Office Supplies       Labels
3  FUR-TA-10000577        Furniture       Tables
4  OFF-ST-10000760  Office Supplies      Storage

                                         Product Name     Sales
Quantity  \
0                  Bush Somerset Collection Bookcase  261.9600
2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400
3
2  Self-Adhesive Address Labels for Typewriters b...   14.6200
2
```

```
3      Bretford CR4500 Series Slim Rectangular Table  957.5775
5
4                     Eldon Fold 'N Roll Cart System   22.3680
2

   Discount    Profit  Price/Unit
0      0.00   41.9136    130.9800
1      0.00  219.5820    243.9800
2      0.00    6.8714      7.3100
3      0.45 -383.0310    191.5155
4      0.20    2.5164     11.1840

[5 rows x 22 columns]
```

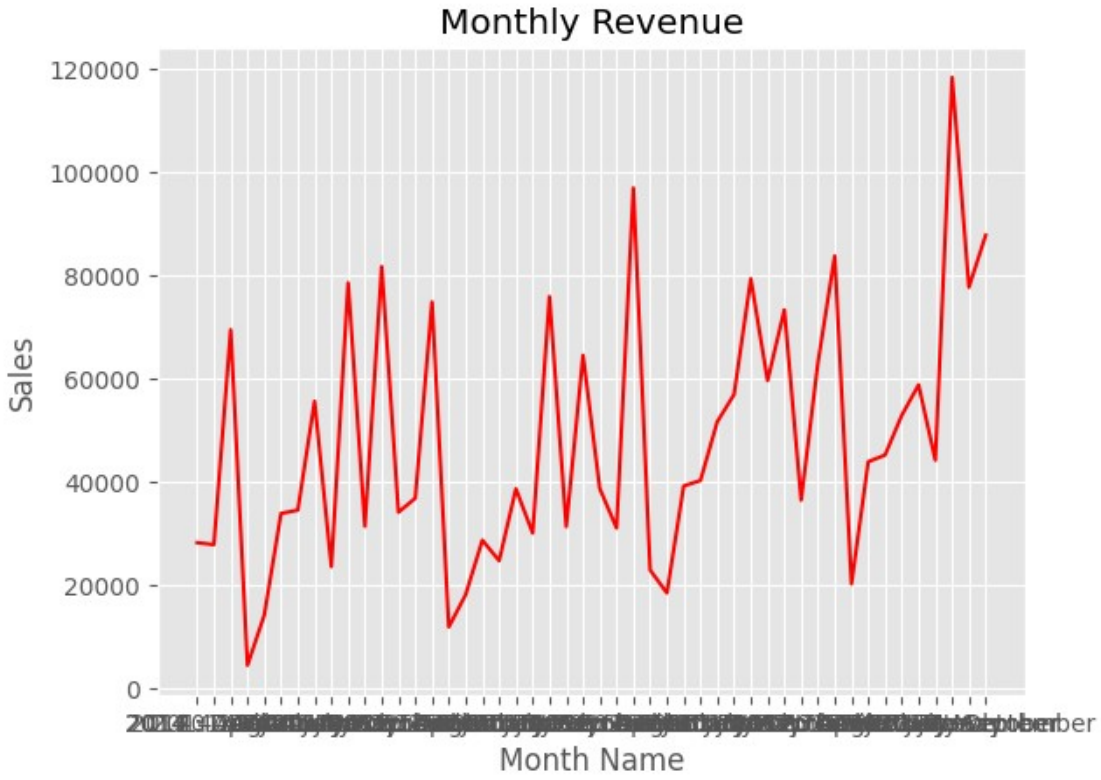
```
df['month_year'] = df['Order Date'].dt.to_period('M')

monthly_revenue =
pd.DataFrame(df.groupby(df['month_year'].dt.strftime('%Y : %B'))
['Sales'].sum())
monthly_revenue
```

```
                        Sales
month_year
2014 : April       28295.3450
2014 : August      27909.4685
2014 : December    69545.6205
2014 : February     4519.8920
2014 : January     14236.8950
2014 : July        33946.3930
2014 : June        34595.1276
2014 : March       55691.0090
2014 : May         23648.2870
2014 : November    78628.7167
2014 : October     31453.3930
2014 : September   81777.3508
2015 : April       34195.2085
2015 : August      36898.3322
2015 : December    74919.5212
2015 : February    11951.4110
2015 : January     18174.0756
2015 : July        28765.3250
2015 : June        24797.2920
2015 : March       38726.2520
2015 : May         30131.6865
2015 : November    75972.5635
2015 : October     31404.9235
2015 : September   64595.9180
2016 : April       38750.0390
2016 : August      31115.3743
2016 : December    96999.0430
2016 : February    22978.8150
```

```
2016 : January     18542.4910
2016 : July        39261.9630
2016 : June        40344.5340
2016 : March       51715.8750
2016 : May         56987.7280
2016 : November    79411.9658
2016 : October     59687.7450
2016 : September   73410.0249
2017 : April       36521.5361
2017 : August      63120.8880
2017 : December    83829.3188
2017 : February    20301.1334
2017 : January     43971.3740
2017 : July        45264.4160
2017 : June        52981.7257
2017 : March       58872.3528
2017 : May         44261.1102
2017 : November   118447.8250
2017 : October     77776.9232
2017 : September   87866.6520
```

```
plt.title('Monthly Revenue')
plt.xlabel('Month Name')
plt.ylabel('Sales')
plt.plot(monthly_revenue, 'r')
plt.show()
```



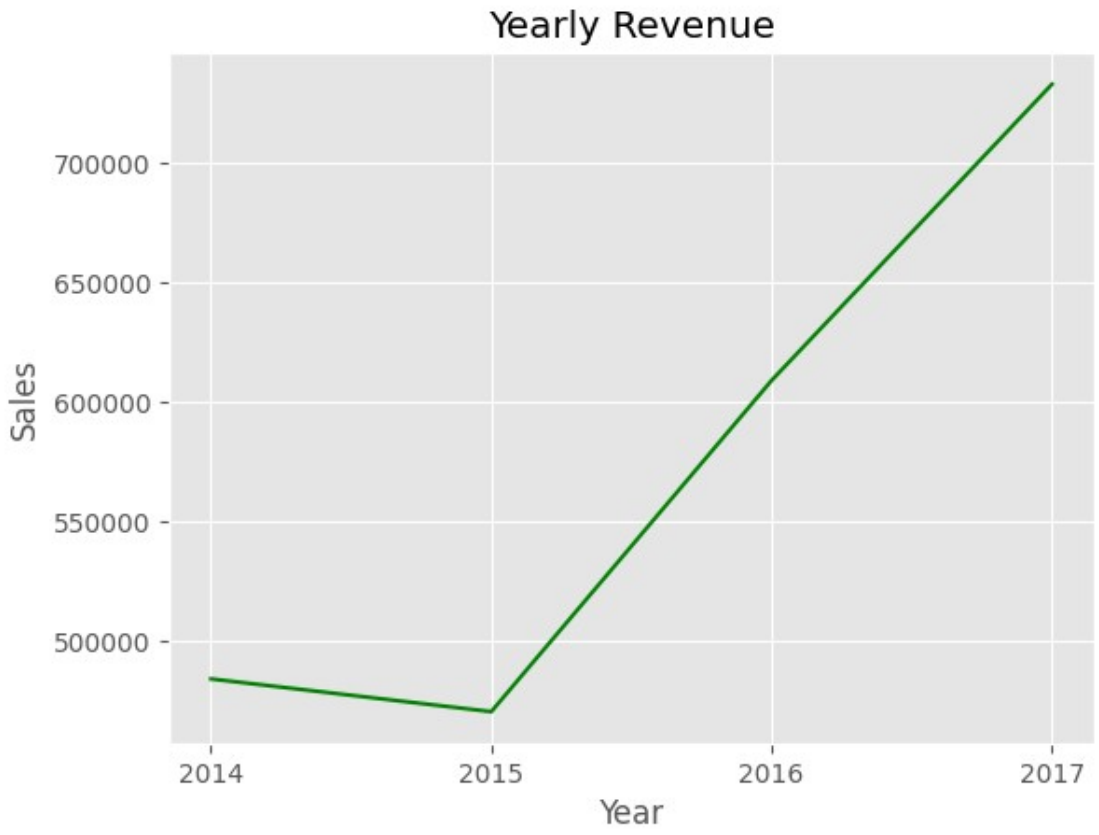
```
monthly_revenue.describe()
```

```
              Sales
count      48.000000
mean    47858.351256
std     25195.890700
min      4519.892000
25%     29790.096125
50%     39803.248500
75%     65833.343625
max    118447.825000
```

```
yearly_revenue =
pd.DataFrame(df.groupby(df['month_year'].dt.strftime('%Y'))
['Sales'].sum())
yearly_revenue
```

```
                  Sales
month_year
2014        484247.4981
2015        470532.5090
2016        609205.5980
2017        733215.2552
```

```
plt.title('Yearly Revenue')
plt.xlabel('Year')
plt.ylabel('Sales')
plt.plot(yearly_revenue, 'g')
plt.show()
```




```
yearly_revenue.describe()
```

```
                Sales
count        4.000000
mean    574300.215075
std     122949.318005
min     470532.509000
25%     480818.750825
50%     546726.548050
75%     640208.012300
max     733215.255200
```

```
monthly_revenue["Monthly Growth Rate"]=(monthly_revenue['Sales']-
monthly_revenue['Sales'].shift())/monthly_revenue['Sales'].shift()*100
monthly_revenue
```

```
                        Sales  Monthly Growth Rate
month_year
2014 : April       28295.3450                  NaN
2014 : August      27909.4685            -1.363746
2014 : December    69545.6205           149.182891
2014 : February     4519.8920           -93.500824
2014 : January     14236.8950           214.983079
2014 : July        33946.3930           138.439583
2014 : June        34595.1276             1.911056
```

```
2014 : March       55691.0090           60.979343
2014 : May         23648.2870          -57.536616
2014 : November    78628.7167          232.492230
2014 : October     31453.3930          -59.997576
2014 : September   81777.3508          159.995323
2015 : April       34195.2085          -58.184989
2015 : August      36898.3322            7.904978
2015 : December    74919.5212          103.043110
2015 : February    11951.4110          -84.047668
2015 : January     18174.0756           52.066359
2015 : July        28765.3250           58.276688
2015 : June        24797.2920          -13.794501
2015 : March       38726.2520           56.171295
2015 : May         30131.6865          -22.193125
2015 : November    75972.5635          152.135119
2015 : October     31404.9235          -58.662809
2015 : September   64595.9180          105.687232
2016 : April       38750.0390          -40.011629
2016 : August      31115.3743          -19.702341
2016 : December    96999.0430          211.739920
2016 : February    22978.8150          -76.310266
2016 : January     18542.4910          -19.306148
2016 : July        39261.9630          111.740499
2016 : June        40344.5340            2.757302
2016 : March       51715.8750           28.185580
2016 : May         56987.7280           10.193878
2016 : November    79411.9658           39.349240
2016 : October     59687.7450          -24.837845
2016 : September   73410.0249           22.990113
2017 : April       36521.5361          -50.249934
2017 : August      63120.8880           72.831964
2017 : December    83829.3188           32.807572
2017 : February    20301.1334          -75.782777
2017 : January     43971.3740          116.595661
2017 : July        45264.4160            2.940645
2017 : June        52981.7257           17.049396
2017 : March       58872.3528           11.118224
2017 : May         44261.1102          -24.818513
2017 : November   118447.8250          167.611509
2017 : October     77776.9232          -34.336554
2017 : September   87866.6520           12.972651
```

```
print(np.max(monthly_revenue['Monthly Growth Rate']))
```

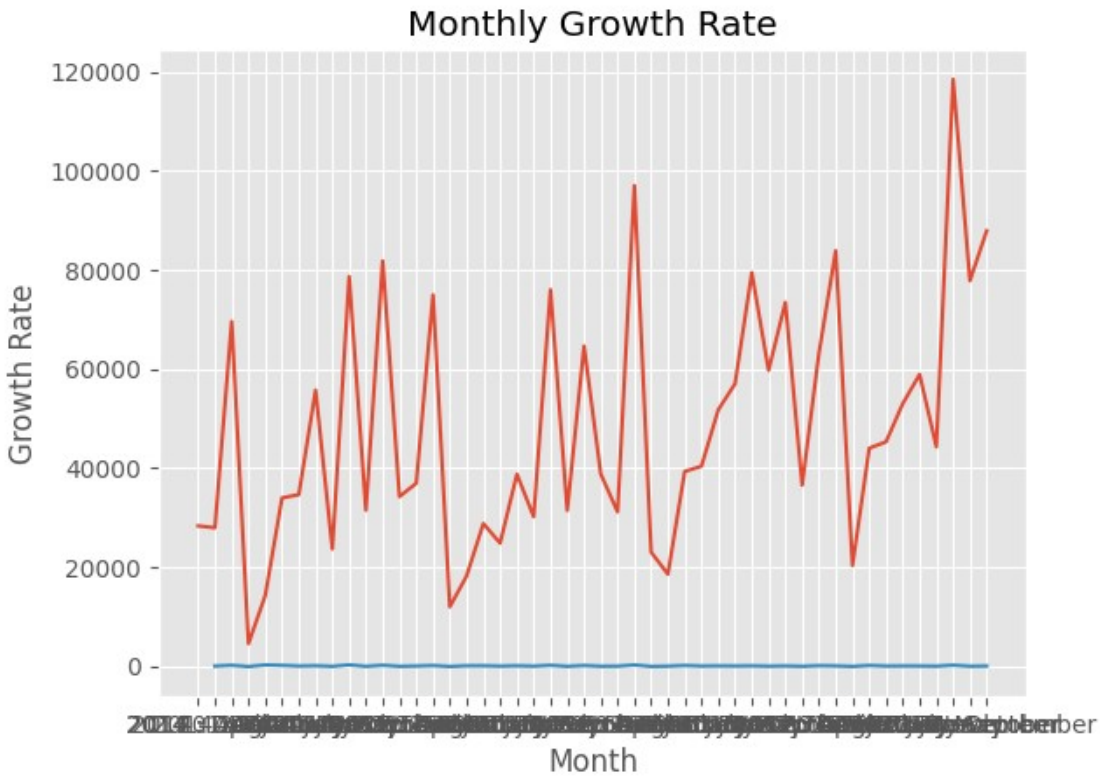
```
232.4922295640272
```

```
monthly_revenue.loc[monthly_revenue['Monthly Growth Rate'] ==
232.4922295640271]
```

```
Empty DataFrame
Columns: [Sales, Monthly Growth Rate]
Index: []
```

```
plt.title('Monthly Growth Rate')
plt.xlabel('Month')
plt.ylabel('Growth Rate')
plt.plot(monthly_revenue)
```

```
[<matplotlib.lines.Line2D at 0x23c329dac20>,
 <matplotlib.lines.Line2D at 0x23c329dabc0>]
```




```
monthly_revenue.describe()
```

	Sales	Monthly Growth Rate
count	48.000000	47.000000
mean	47858.351256	32.755629
std	25195.890700	85.070021
min	4519.892000	-93.500824
25%	29790.096125	-24.828179
50%	39803.248500	11.118224
75%	65833.343625	87.937537
max	118447.825000	232.492230

Finding out the most and least sold product id

```
product_id_df = pd.DataFrame(df.groupby(df["Product ID"])
["Quantity"].sum())
product_id_df
```

```
                 Quantity
Product ID
FUR-BO-10000112         9
FUR-BO-10000330        10
FUR-BO-10000362        14
FUR-BO-10000468        21
FUR-BO-10000711        12
...                   ...
TEC-PH-10004912        11
TEC-PH-10004922        19
TEC-PH-10004924         8
TEC-PH-10004959         4
TEC-PH-10004977        32

[1862 rows x 1 columns]
```

```
print(np.max(product_id_df["Quantity"]))
```

```
75
```

```
product_id_df.loc[product_id_df['Quantity'] == 75]
```

```
                 Quantity
Product ID
TEC-AC-10003832        75
```

Finding out the customer who bought most and least from us in terms of quantity

```
customer_name_df = pd.DataFrame(df.groupby(df["Customer Name"])
["Quantity"].sum())
customer_name_df
```

```
                    Quantity
Customer Name
Aaron Bergman             13
Aaron Hawkins             54
Aaron Smayling            48
Adam Bellavance           56
Adam Hart                 75
...                      ...
Xylona Preis             100
Yana Sorensen             58
Yoseph Carroll            31
Zuschuss Carroll         105
Zuschuss Donatelli        32

[793 rows x 1 columns]
```

```
print(np.max(customer_name_df["Quantity"]))
```

```
150
```

```
customer_name_df.loc[customer_name_df['Quantity'] == 150]
```

```
                  Quantity
Customer Name
Jonathan Doherty       150
```

Finding out the customer who bought most and least from us in terms of value

```
customer_by_value = pd.DataFrame(df.groupby(df["Customer Name"])
["Sales"].sum())
customer_by_value
```

```
                       Sales
Customer Name
Aaron Bergman         886.156
Aaron Hawkins        1744.700
Aaron Smayling       3050.692
Adam Bellavance      7755.620
Adam Hart            3250.337
...                       ...
Xylona Preis         2374.658
Yana Sorensen        6720.444
Yoseph Carroll       5454.350
Zuschuss Carroll     8025.707
Zuschuss Donatelli   1493.944

[793 rows x 1 columns]
```

```
print(np.max(customer_by_value["Sales"]))
```

```
25043.05
```

```
customer_by_value.loc[customer_by_value['Sales'] == 25043.05]
```

```
                  Sales
Customer Name
Sean Miller    25043.05
```

Finding out the majority and minority customer cities on basis of Number of customers

```
city_customer_group=df.groupby("City").count()['Customer Name']
city_customer_group = pd.DataFrame(city_customer_group)
city_customer_group['Customer Count'] = city_customer_group['Customer
Name']
city_customer_group.drop('Customer Name',  axis='columns',
inplace=True)
city_customer_group
```

```
             Customer Count
City
Aberdeen                  1
Abilene                   1
Akron                    21
Albuquerque              14
Alexandria               16
...                     ...
Woonsocket                4
Yonkers                  15
York                      5
Yucaipa                   1
Yuma                      4

[531 rows x 1 columns]
```

```
print(np.max(city_customer_group["Customer Count"]))
```

```
915
```

```
city_customer_group.loc[city_customer_group['Customer Count'] == 915]
```

```
               Customer Count
City
New York City             915
```

```
print(np.min(city_customer_group["Customer Count"]))
```

```
1
```

```
city_customer_group.loc[city_customer_group['Customer Count'] == 1]
```

```
                   Customer Count
City
Aberdeen                        1
Abilene                         1
Antioch                         1
Arlington Heights               1
Atlantic City                   1
...                           ...
Vacaville                       1
Waterloo                        1
Waukesha                        1
Whittier                        1
Yucaipa                         1

[70 rows x 1 columns]
```

```
city_by_sale = pd.DataFrame(df.groupby(df["City"])["Sales"].sum())
city_by_sale
```

```
                Sales
City
Aberdeen       25.500
Abilene         1.392
Akron        2729.986
Albuquerque  2220.160
Alexandria   5519.570
...               ...
Woonsocket    195.550
Yonkers      7657.666
York          817.978
Yucaipa        50.800
Yuma          840.865

[531 rows x 1 columns]
```

```
print(np.max(city_by_sale["Sales"]))
```

```
256368.161
```

```
city_by_sale.loc[city_by_sale['Sales'] == 256368.161]
```

```
                   Sales
City
New York City  256368.161
```

```
print(np.min(city_by_sale["Sales"]))
```

```
1.3919999999999997
```

```
city_by_sale.loc[city_by_sale['Sales'] == 1.3919999999999997]
```

```
         Sales
City
Abilene  1.392
```

Number of Quantity Sold

```
city_by_quantity = pd.DataFrame(df.groupby(df["City"])
["Quantity"].sum())
city_by_quantity
```

```
             Quantity
City
Aberdeen            3
Abilene             2
Akron              65
Albuquerque        65
Alexandria         84
...               ...
Woonsocket         15
Yonkers            57
```

```
York               19
Yucaipa             5
Yuma               22

[531 rows x 1 columns]
```

```
print(np.max(city_by_quantity["Quantity"]))
```

```
3417
```

```
city_by_quantity.loc[city_by_quantity['Quantity'] == 3417]
```

```
               Quantity
City
New York City      3417
```

```
print(np.min(city_by_quantity["Quantity"]))
```

```
1
```

```
city_by_quantity.loc[city_by_quantity['Quantity'] == 1]
```

```
             Quantity
City
Elyria              1
Iowa City           1
Jupiter             1
Lindenhurst         1
Littleton           1
Port Orange         1
```

Find out the most and least sold product category from the store Quantity based

```
product_category_by_quantity = pd.DataFrame(df.groupby(df["Category"])
["Quantity"].sum())
product_category_by_quantity
```

```
                 Quantity
Category
Furniture            8028
Office Supplies     22906
Technology           6939
```

```
print(np.max(product_category_by_quantity["Quantity"]))
```

```
22906
```

```
product_category_by_quantity.loc[product_category_by_quantity['Quantit
y'] == 22906]
```

```
                 Quantity
Category
Office Supplies     22906
```

```
print(np.min(product_category_by_quantity["Quantity"]))
```

```
6939
```

```
product_category_by_quantity.loc[product_category_by_quantity['Quantit
y'] == 6939]
```

```
            Quantity
Category
Technology      6939
```

Value based

```
product_category_by_value = pd.DataFrame(df.groupby(df["Category"])
["Sales"].sum())
product_category_by_value
```

```
                      Sales
Category
Furniture        741999.7953
Office Supplies  719047.0320
Technology       836154.0330
```

```
print(np.max(product_category_by_value["Sales"]))
```

```
836154.033
```

```
product_category_by_value.loc[product_category_by_value['Sales'] ==
836154.0329999966]
```

```
Empty DataFrame
Columns: [Sales]
Index: []
```

```
print(np.min(product_category_by_value["Sales"]))
```

```
719047.032
```

```
product_category_by_value.loc[product_category_by_value['Sales'] ==
719047.0320000029]
```

```
Empty DataFrame
Columns: [Sales]
Index: []
```

Find out the most and least sold product sub category from the store By Quantity

```
subcat_by_quantity = pd.DataFrame(df.groupby(df["Sub-Category"])
["Quantity"].sum())
subcat_by_quantity
```

```
              Quantity
Sub-Category
Accessories       2976
```

```
Appliances        1729
Art               3000
Binders           5974
Bookcases          868
Chairs            2356
Copiers            234
Envelopes          906
Fasteners          914
Furnishings       3563
Labels            1400
Machines           440
Paper             5178
Phones            3289
Storage           3158
Supplies           647
Tables            1241
```

```
print(np.max(subcat_by_quantity["Quantity"]))
```

```
5974
```

```
subcat_by_quantity.loc[subcat_by_quantity['Quantity'] == 5974]
```

```
             Quantity
Sub-Category
Binders          5974
```

```
print(np.min(subcat_by_quantity["Quantity"]))
```

```
234
```

```
subcat_by_quantity.loc[subcat_by_quantity['Quantity'] == 234]
```

```
             Quantity
Sub-Category
Copiers           234
```

By Value

```
subcat_by_value = pd.DataFrame(df.groupby(df["Sub-Category"])
["Sales"].sum())
subcat_by_value
```

```
                   Sales
Sub-Category
Accessories   167380.3180
Appliances    107532.1610
Art            27118.7920
Binders       203412.7330
Bookcases     114879.9963
Chairs        328449.1030
Copiers       149528.0300
Envelopes      16476.4020
```

```
Fasteners       3024.2800
Furnishings    91705.1640
Labels         12486.3120
Machines      189238.6310
Paper          78479.2060
Phones        330007.0540
Storage       223843.6080
Supplies       46673.5380
Tables        206965.5320
```

```
print(np.max(subcat_by_value["Sales"]))
```

```
330007.054
```

```
subcat_by_value.loc[subcat_by_value['Sales'] == 330007.0540000001]
```

```
Empty DataFrame
Columns: [Sales]
Index: []
```

```
print(np.min(subcat_by_value["Sales"]))
```

```
3024.28
```

```
subcat_by_value.loc[subcat_by_value['Sales'] == 3024.2799999999997]
```

```
Empty DataFrame
Columns: [Sales]
Index: []
```