In [72]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
import calendar
```

In [2]:
```
# cwd
cwd = os.getcwd()
cwd
```

Out[2]:
```
'C:\\Users\\Joel Che'
```

In [91]:
```
df = pd.read_excel(r'C:\Users\Joel Che\python_projects\Superstore.xlsx')
```

In [94]:
```
df.head()
```

Out[94]:

	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	year	month
0	1	CA-2017-152156	2017-11-08	2017-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420.0	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.9600	2017	Nov
1	2	CA-2017-152156	2017-11-08	2017-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420.0	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400	2017	Nov
2	3	CA-2017-138688	2017-06-12	2017-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036.0	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters b...	14.6200	2017	Jun
3	4	US-2016-108966	2016-10-11	2016-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311.0	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	2016	Oct
4	5	US-2016-108966	2016-10-11	2016-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311.0	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	22.3680	2016	Oct

Add year and month column

In [100]:
```
df['year'] = pd.DatetimeIndex(df['Order Date']).year
df['month'] = pd.DatetimeIndex(df['Order Date']).month
df['month'] = df['month'].apply(lambda x: calendar.month_abbr[x])
sorted_months = ['Jan', 'Feb', 'Mar','Apr', 'May','Jun','Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
df['month'] = pd.Categorical(df['month'], categories = sorted_months, ordered = True)
```

Which year had the highest sales?

In [95]:
```
df[['year', 'Sales']].groupby('year').sum()
```

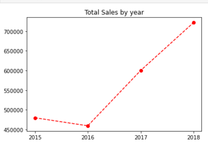
Out[95]:

	Sales
year	
2015	479856.2081
2016	459436.0054
2017	600192.5500
2018	722052.0192

In [271]:
```
df_Ysales = df[['year', 'Sales']].groupby('year').sum()
x = [2015,2016,2017,2018]
y = df_Ysales['Sales']
plot = plt.plot(x,y, 'ro--')


plt.title('Total Sales by year')
plt.xticks([2015,2016,2017,2018])

plt.show()
```



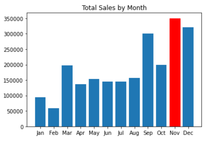
Which month had the highest sales?

In [101]:
```
month_sales = df[['Category','month','year', 'Sales']].groupby(['Category','month', 'year']).sum()
month_sales.query("year == 2018 & Category == 'Furniture'")
```

Out[101]:

			Sales
Category	**month**	**year**	
Furniture	**Jan**	**2018**	5930.1620
	Feb	**2018**	6774.3774
	Mar	**2018**	10893.4448
	Apr	**2018**	9065.9581
	May	**2018**	16957.5582
	Jun	**2018**	17032.3867
	Jul	**2018**	11375.8220
	Aug	**2018**	15419.1220
	Sep	**2018**	28516.7060
	Oct	**2018**	21884.0682
	Nov	**2018**	37056.7150
	Dec	**2018**	31407.4668

In [353]:
```
M_sales = df[['month','Sales']].groupby(['month']).sum()
x = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
y = M_sales['Sales']
bars = plt.bar(x,y)
bars[10].set_color('red')
plt.title('Total Sales by Month')
plt.show()
```



Which Category had the highest sales?

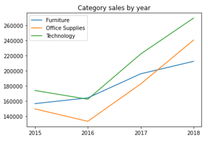
In [342]:
```
Cat_sales = df[['Category', 'year', 'Sales']].groupby(['Category','year'], as_index=False).sum()
Pvt = Cat_sales.pivot(index='year',columns='Category', values='Sales')
Pvt
```

Out[342]:

Category	Furniture	Office Supplies	Technology
year			
2015	156477.8811	149512.820	173865.507
2016	164053.8674	133124.407	162257.731
2017	195813.0400	182417.566	221961.944
2018	212313.7872	240367.541	269370.691

In [352]:
```
x = [2015,2016,2017,2018]
y1 = Pvt['Furniture']
y2 = Pvt['Office Supplies']
y3 = Pvt['Technology']

plt.xticks([2015,2016,2017,2018])
plt.plot(x,y1, label = 'Furniture')
plt.plot(x,y2, label = 'Office Supplies')
plt.plot(x,y3, label = 'Technology')
plt.legend()
plt.title('Category sales by year')
plt.show()
```



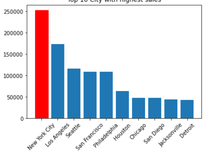
Which City had the highest sales?

In [355]:
```
C_Sales = df[['City','Sales']].groupby(['City']).sum()
C_Sales1 = C_Sales.sort_values(by='Sales', ascending = False).head(10)
C_Sales1
```

Out[355]:

	Sales
City	
New York City	252462.5470
Los Angeles	173420.1810
Seattle	116106.3220
San Francisco	109041.1200
Philadelphia	108841.7490
Houston	63956.1428
Chicago	47820.1330
San Diego	47521.0290
Jacksonville	44713.1830
Detroit	42446.9440

In [356]:
```
x = ['New York City',' Los Angeles', 'Seattle', 'San Francisco', 'Philadelphia', 'Houston', 'Chicago', 'San Diego', 'Jacksonville', 'Detroit']
y = C_Sales1['Sales']
bars = plt.bar(x,y)
bars[0].set_color('red')
plt.xticks(rotation=45)
plt.title('Top 10 City with highest sales')
plt.show()
```



Which products were sold together?

In [362]:
```
#ref: https://www.youtube.com/watch?v=eMOAIpPVUc4&t=195s&ab_channel=KeithGalli

df_p = df[df['Order ID'].duplicated(keep=False)]

df_p['Grouped'] = df_p.groupby('Order ID')['Sub-Category'].transform(lambda x: ','.join(x))

df_p = df_p[['Order ID', 'Grouped']].drop_duplicates()

df_p.head()
```

```
C:\Users\Joel Che\AppData\Local\Temp\ipykernel_5192\300955537.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_p['Grouped'] = df_p.groupby('Order ID')['Sub-Category'].transform(lambda x: ','.join(x))
```

Out[362]:

	Order ID	Grouped
0	CA-2017-152156	Bookcases,Chairs
3	US-2016-108966	Tables,Storage
5	CA-2015-115812	Furnishings,Art,Phones,Binders,Appliances,Tabl...
14	US-2016-118983	Appliances,Binders
18	CA-2015-143336	Art,Phones,Binders

In [369]:
```
from itertools import combinations
from collections import Counter

count = Counter()

for row in df_p['Grouped']:
    row_list = row.split(',')
    count.update(Counter(combinations(row_list, 2)))

for key, value in count.most_common(10):
    print(key,value)
```

```
('Binders', 'Binders') 222
('Paper', 'Binders') 202
('Paper', 'Paper') 191
('Binders', 'Paper') 184
('Furnishings', 'Binders') 146
('Phones', 'Binders') 144
('Binders', 'Storage') 143
('Paper', 'Storage') 134
('Storage', 'Binders') 132
('Paper', 'Furnishings') 125
```