Import and Reading Data

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_excel (r'C:\Users\aqsa\Desktop\Data Analytics Work\Global Superstore.xls')
```

In [3]:
```
df.head()
```

Out[3]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product ID	Category	Sub-Category	Produc Nam
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	New York	...	TEC-AC-10003033	Technology	Accessories	Plantronic CS510 Over-the Hea monaur Wir.
1	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporate	Wollongong	New South Wales	...	FUR-CH-10003950	Furniture	Chairs	Novime Executiv Leathe Armchai Blac
2	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer	Brisbane	Queensland	...	TEC-PH-10004664	Technology	Phones	Noki Sma Phone with Calle I
3	13524	ES-2013-1579342	2013-01-28	2013-01-30	First Class	KM-16375	Katherine Murray	Home Office	Berlin	Berlin	...	TEC-PH-10004583	Technology	Phones	Motorol Sma Phone Cordles
4	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	RH-9495	Rick Hansen	Consumer	Dakar	Dakar	...	TEC-SHA-10000501	Technology	Copiers	Shar Wireles Fax, High Spee

5 rows × 24 columns

In [5]:
```
df.shape
```

Out[5]:
```
(51290, 24)
```

In [4]:
```
df.columns
```

Out[4]:
```
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')
```

In [5]:
```
df.dtypes
```

Out[5]:
```
Row ID                  int64
Order ID               object
Order Date     datetime64[ns]
Ship Date      datetime64[ns]
Ship Mode              object
Customer ID            object
Customer Name          object
Segment                object
City                   object
State                  object
Country                object
Postal Code           float64
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
```

In [6]:
```
df.describe()
```

Out[6]:

	Row ID	Postal Code	Sales	Quantity	Discount	Profit	Shipping Cost
count	51290.00000	9994.000000	51290.000000	51290.000000	51290.000000	51290.000000	51290.000000
mean	25645.50000	55190.379428	246.490581	3.476545	0.142908	28.610982	26.375818
std	14806.29199	32063.693350	487.565361	2.278766	0.212280	174.340972	57.296810
min	1.00000	1040.000000	0.444000	1.000000	0.000000	-6599.978000	0.002000
25%	12823.25000	23223.000000	30.758625	2.000000	0.000000	0.000000	2.610000
50%	25645.50000	56430.500000	85.053000	3.000000	0.000000	9.240000	7.790000
75%	38467.75000	90008.000000	251.053200	5.000000	0.200000	36.810000	24.450000
max	51290.00000	99301.000000	22638.480000	14.000000	0.850000	8399.976000	933.570000

In [7]:
```
df['Category'].value_counts()
```

Out[7]:
```
Office Supplies    31273
Technology         10141
Furniture           9876
Name: Category, dtype: int64
```

In [8]:
```
df['Sub-Category'].value_counts()
```

Out[8]:
```
Binders        6152
Storage        5059
Art            4883
Paper          3538
Chairs         3434
Phones         3357
Furnishings    3170
Accessories    3075
Labels         2606
Envelopes      2435
Supplies       2425
Fasteners      2420
Bookcases      2411
```

```
Copiers        2223
Appliances     1755
Machines       1486
Tables          861
Name: Sub-Category, dtype: int64
```

Wrangling and Manipulating Data

In [9]:
```
df.info()
```

```
<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  datetime64[ns]
 3   Ship Date       51290 non-null  datetime64[ns]
 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: datetime64[ns](2), float64(5), int64(2), object(15)
memory usage: 9.4+ MB
```

In [10]:
```
missing_data = df.isnull()
```

```
missing_data.head()
```

Out[10]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Dis
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	

5 rows × 24 columns

In [11]:

```
for column in missing_data.columns.values.tolist():
    print(column)
    print (missing_data[column].value_counts())
    print ("")
```

```
Row ID
False    51290
Name: Row ID, dtype: int64

Order ID
False    51290
Name: Order ID, dtype: int64

Order Date
False    51290
Name: Order Date, dtype: int64

Ship Date
False    51290
Name: Ship Date, dtype: int64

Ship Mode
False    51290
Name: Ship Mode, dtype: int64

Customer ID
```

```
False    51290
Name: Customer ID, dtype: int64

Customer Name
False    51290
Name: Customer Name, dtype: int64

Segment
False    51290
Name: Segment, dtype: int64

City
False    51290
Name: City, dtype: int64

State
False    51290
Name: State, dtype: int64

Country
False    51290
Name: Country, dtype: int64

Postal Code
True     41296
False     9994
Name: Postal Code, dtype: int64

Market
False    51290
Name: Market, dtype: int64

Region
False    51290
Name: Region, dtype: int64

Product ID
False    51290
Name: Product ID, dtype: int64

Category
False    51290
Name: Category, dtype: int64

Sub-Category
```

```
False    51290
Name: Sub-Category, dtype: int64

Product Name
False    51290
Name: Product Name, dtype: int64

Sales
False    51290
Name: Sales, dtype: int64

Quantity
False    51290
Name: Quantity, dtype: int64

Discount
False    51290
Name: Discount, dtype: int64

Profit
False    51290
Name: Profit, dtype: int64

Shipping Cost
False    51290
Name: Shipping Cost, dtype: int64

Order Priority
False    51290
Name: Order Priority, dtype: int64
```

In [12]:
```
df.dropna (subset = ["Postal Code"], axis=0)
```

Out[12]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product ID	Category	Sub-Category	
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	New York	...	TEC-AC-10003033	Technology	Accessories	Pla O m

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product ID	Category	Sub-Category	
8	40155	CA-2014-135909	2014-10-14	2014-10-21	Standard Class	JW-15220	Jane Waco	Corporate	Sacramento	California	...	OFF-BI-10003527	Office Supplies	Binders	Plast
9	40936	CA-2012-116638	2012-01-28	2012-01-31	Second Class	JH-15985	Joseph Holt	Consumer	Concord	North Carolina	...	FUR-TA-10000198	Furniture	Tables	Ch B W Co
10	34577	CA-2011-102988	2011-04-05	2011-04-09	Second Class	GM-14695	Greg Maxwell	Corporate	Alexandria	Virginia	...	OFF-SU-10002881	Office Supplies	Supplies	Ma C Let
16	36178	CA-2014-143567	2014-11-03	2014-11-06	Second Class	TB-21175	Thomas Boland	Corporate	Henderson	Kentucky	...	TEC-AC-10004145	Technology	Accessories	diNo K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
51270	38414	CA-2011-143168	2011-10-18	2011-10-23	Second Class	IG-15085	Ivan Gibson	Consumer	Seattle	Washington	...	OFF-BI-10003784	Office Supplies	Binders	C In
51276	31558	US-2014-155299	2014-06-09	2014-06-13	Standard Class	DI-13600	Dorris liebe	Corporate	Pasadena	Texas	...	OFF-AP-10002203	Office Supplies	Appliances	Dis Vib
51277	37361	CA-2012-111780	2012-12-25	2012-12-30	Second Class	RA-19285	Ralph Arnett	Consumer	San Diego	California	...	OFF-PA-10001667	Office Supplies	Paper	Gre M R Pap

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product ID	Category	Sub-Category	
51286	35398	US-2014-102288	2014-06-20	2014-06-24	Standard Class	ZC-21910	Zuschuss Carroll	Consumer	Houston	Texas	...	OFF-AP-10002906	Office Supplies	Appliances	Repl Co Gu
51287	40470	US-2013-155768	2013-12-02	2013-12-02	Same Day	LB-16795	Laurel Beltran	Home Office	Oxnard	California	...	OFF-EN-10001219	Office Supplies	Envelopes	#10- Sec E

9994 rows × 24 columns

In [13]:
```
import datetime
```

In [14]:
```
df['Month'] = df['Order Date'].dt.month
df['Month'] = df ['Order Date'].dt.strftime('%m')
```

In [15]:
```
df ['Day'] = df ['Order Date'].dt.month
df ['Day'] = df ['Order Date'].dt.strftime ('%d')
```

In [16]:
```
df ['Price'] = df ['Sales'] / df ['Quantity']
```

In [17]:
```
df.head()
```

Out[17]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product Name	Sales	Quantity	Discount	
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	New York	...	Plantronics CS510 - Over-the-Head monaural Wir...	2309.650	7	0.0	7

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product Name	Sales	Quantity	Discount	
1	26341	IN-2013-77878	02-05	02-07	Class	JR-16210	Ritter	Corporate	Wollongong	Wales	...	Novimex Executive Leather Black	3709.395	9	0.1	-2
2	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer	Brisbane	Queensland	...	Nokia Smart Phone, with Caller ID	5175.171	9	0.1	9
3	13524	ES-2013-1579342	2013-01-28	2013-01-30	First Class	KM-16375	Katherine Murray	Home Office	Berlin	Berlin	...	Motorola Smart Phone, Cordless	2892.510	5	0.1	-
4	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	RH-9495	Rick Hansen	Consumer	Dakar	Dakar	...	Sharp Wireless Fax, High-Speed	2832.960	8	0.0	3

5 rows × 27 columns

What was the best month for sales?

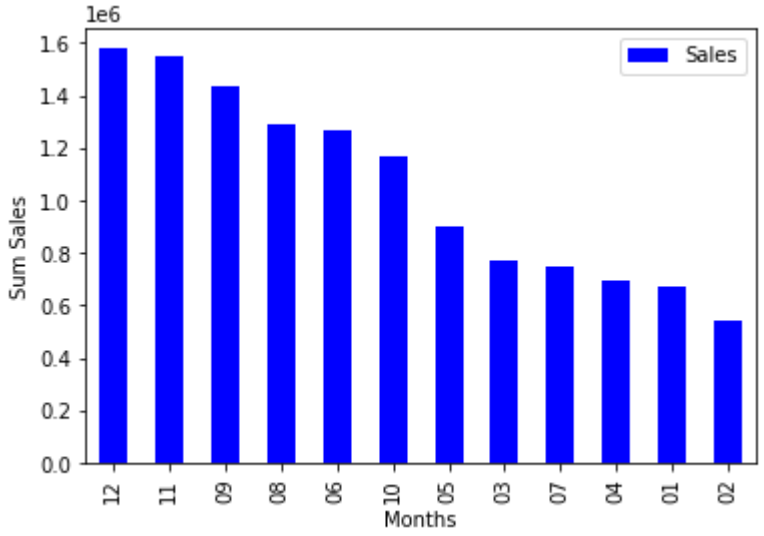
In [18]:
```
df_month = df [['Month', 'Sales']]
grouped_month = df_month.groupby("Month").sum().sort_values ('Sales', ascending = False).head(12)
grouped_month
```

Out[18]:

Month	Sales
12	1.580781e+06
11	1.551277e+06
09	1.437380e+06

Month	Sales
08	1.293833e+06
06	1.269717e+06
10	1.168184e+06
05	9.040123e+05
03	7.705009e+05
07	7.493818e+05
04	6.985612e+05
01	6.751337e+05
02	5.437394e+05

In [19]:
```
grouped_month.plot.bar (color ="b")
plt.ylabel ('Sum Sales')
plt.xlabel ('Months')
plt.show()
```



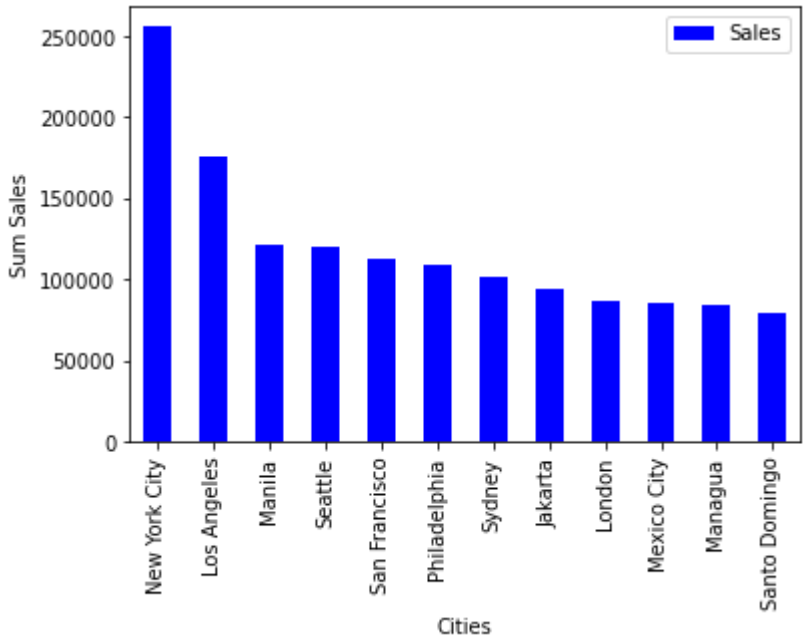
What city sold the most product?

In [20]:
```
df_gpcity = df [['City', 'Sales']]
grouped_city = df_gpcity.groupby (['City']).sum().sort_values ('Sales', ascending = False).head (12)
grouped_city
```

Out[20]:

City	Sales
New York City	256368.16100
Los Angeles	175851.34100
Manila	120886.94850
Seattle	119540.74200
San Francisco	112669.09200
Philadelphia	109077.01300
Sydney	101945.51700
Jakarta	94321.32420
London	86945.80500
Mexico City	85728.55176
Managua	83707.49804
Santo Domingo	78713.66344

In [21]:
```
grouped_city.plot.bar (color = "b")
plt.ylabel ("Sum Sales")
plt.xlabel ("Cities")
plt.show ()
```



What state sold the most product?

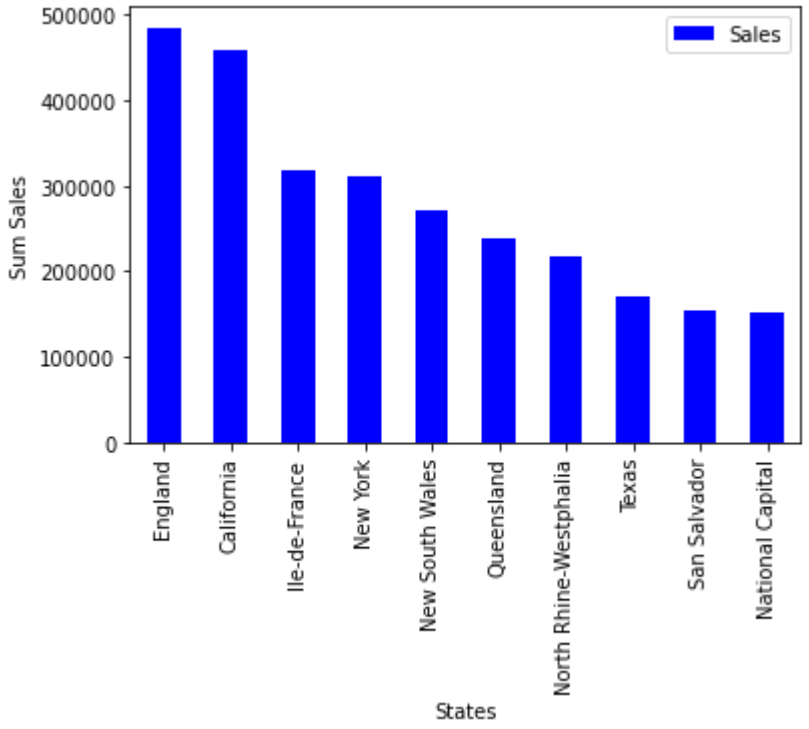
In [22]:
```
df_gpstate = df [['State','Sales']]
grouped_state = df_gpstate.groupby("State").sum().sort_values("Sales", ascending = False).head(10)
grouped_state
```

Out[22]:

State	Sales
England	485170.9710
California	457687.6315
Ile-de-France	317822.5440
New York	310876.2710
New South Wales	270487.1040
Queensland	238312.7340

	Sales
State	
North Rhine-Westphalia	216451.8510
Texas	170188.0458
San Salvador	153639.3970
National Capital	152175.3555

In [23]:
```
grouped_state.plot.bar (color = "blue")
plt.ylabel ("Sum Sales")
plt.xlabel ("States")
plt.show()
```



What time should we display advertisements to maximize the likelihood of customers buying product?

In [24]:
```
df_gpday = df[['Day', 'Sales']]
group_day = df_gpday.groupby("Day").count().sort_values("Day", ascending = False)
group_day
```

Out[24]:

Day	Sales
31	983
30	1432
29	1480
28	1705
27	1767
26	1770
25	1785
24	1715
23	1595
22	1647
21	1694
20	1771
19	1676
18	1851
17	1658
16	1577
15	1714
14	1619

Day	Sales
13	1784
12	1763
11	1763
10	1656
09	1671
08	1571
07	1698
06	1670
05	1661
04	1684
03	1806
02	1635
01	1489

In [25]:
```
group_day.plot.line (color = "blue")
plt.ylabel = ("Count Sales")
plt.xlabel = ("Days")
plt.grid ()
plt.show()
# The best days for advertising is from the third day to the 28th of the month.
```

What categories are most often sold togather?

In [26]:
```
df = df [df['Order ID'].duplicated(keep = False)]
df.head()
```

Out[26]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product Name	Sales	Quantity	Discount	
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	New York	...	Plantronics CS510 - Over-the-Head monaural Wir...	2309.650	7	0.0	76
1	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporate	Wollongong	New South Wales	...	Novimex Executive Leather Armchair, Black	3709.395	9	0.1	-28

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Product Name	Sales	Quantity	Discount	
2	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer	Brisbane	Queensland	...	Nokia Smart Phone, with Caller ID	5175.171	9	0.1	91
5	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	JM-15655	Jim Mitchum	Corporate	Sydney	New South Wales	...	Samsung Smart Phone, with Caller ID	2862.675	5	0.1	76
6	30570	IN-2011-81826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Consumer	Porirua	Wellington	...	Novimex Executive Leather Armchair, Adjustable	1822.080	4	0.0	56

5 rows × 27 columns

In [27]:
```
df['Grouped'] = df.groupby ('Order ID') ['Category'].transform (lambda x:','.join(x))
```

In [28]:
```
df.head()
```

Out[28]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Sales	Quantity	Discount	Profit	Shi
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	New York	...	2309.650	7	0.0	762.1845	9
1	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporate	Wollongong	New South Wales	...	3709.395	9	0.1	-288.7650	9
2	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer	Brisbane	Queensland	...	5175.171	9	0.1	919.9710	9

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	...	Sales	Quantity	Discount	Profit	Shi
5	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	JM-15655	Jim Mitchum	Corporate	Sydney	New South Wales	...	2862.675	5	0.1	763.2750	8
6	30570	IN-2011-81826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Consumer	Porirua	Wellington	...	1822.080	4	0.0	564.8400	8

5 rows × 28 columns

In [29]:
```
df = df.drop_duplicates (subset=['Order ID','Grouped'])
```

In [30]:
```
from itertools import combinations
```

In [31]:
```
from collections import Counter
```

In [32]:
```
count = Counter ()

for row in df['Grouped']:
    row_list = row.split (',')
    count.update(Counter(combinations(row_list,3)))
for key, value in count.most_common(10):
    print (key,value)
```

```
('Office Supplies', 'Office Supplies', 'Office Supplies') 13414
('Technology', 'Office Supplies', 'Office Supplies') 9136
('Furniture', 'Office Supplies', 'Office Supplies') 7702
('Technology', 'Technology', 'Office Supplies') 3036
('Furniture', 'Technology', 'Office Supplies') 2712
('Technology', 'Furniture', 'Office Supplies') 2643
('Office Supplies', 'Furniture', 'Office Supplies') 2617
('Office Supplies', 'Technology', 'Office Supplies') 2612
('Furniture', 'Furniture', 'Office Supplies') 2481
('Office Supplies', 'Office Supplies', 'Furniture') 1225
```

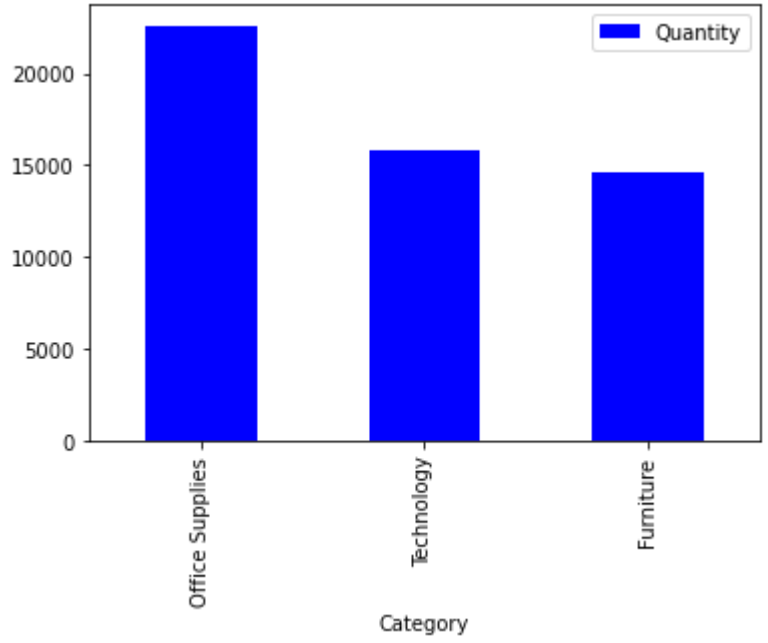
What category sold the most?

In [33]:
```
df_gpcategory  = df[['Category', 'Quantity']]
df_gpquantity = df_gpcategory.groupby ("Category").sum().sort_values("Quantity", ascending = False)
df_gpquantity
```

Out[33]:

Category	Quantity
Office Supplies	22595
Technology	15805
Furniture	14593

In [34]:
```
df_gpquantity.plot.bar (color = "blue")
plt.show()
```



What sub-category sold the most?

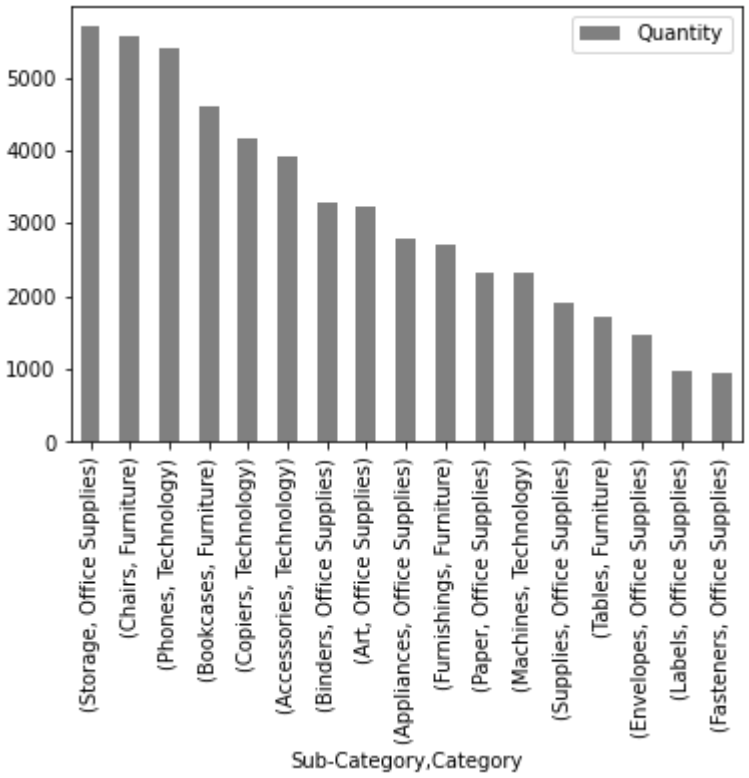
In [35]:
```
df_subcategory = df [['Category','Sub-Category','Quantity']]
df_groupby = df_subcategory.groupby(['Sub-Category','Category']).sum().sort_values("Quantity", ascending=False)
df_groupby
```

Out[35]:

		Quantity
Sub-Category	**Category**	
Storage	**Office Supplies**	5692
Chairs	**Furniture**	5551
Phones	**Technology**	5406
Bookcases	**Furniture**	4607
Copiers	**Technology**	4174
Accessories	**Technology**	3915
Binders	**Office Supplies**	3275
Art	**Office Supplies**	3239
Appliances	**Office Supplies**	2788
Furnishings	**Furniture**	2714
Paper	**Office Supplies**	2322
Machines	**Technology**	2310
Supplies	**Office Supplies**	1899
Tables	**Furniture**	1721
Envelopes	**Office Supplies**	1459
Labels	**Office Supplies**	984
Fasteners	**Office Supplies**	937

In [36]:
```
df_groupby.plot.bar(color = "grey")
```

Out[36]:
```
<AxesSubplot:xlabel='Sub-Category,Category'>
```



What is the relation between Sub-Category, Quantity and Price?

In [37]:
```
prices = df [['Sub-Category', 'Quantity']]
pricegp = prices.groupby(["Sub-Category"]).sum().sort_values("Quantity",ascending = False)
```

In [38]:
```
subcategory_group = df.groupby('Sub-Category')
quantity_order = subcategory_group.sum()['Quantity']
category = [category for category, df in subcategory_group]
```

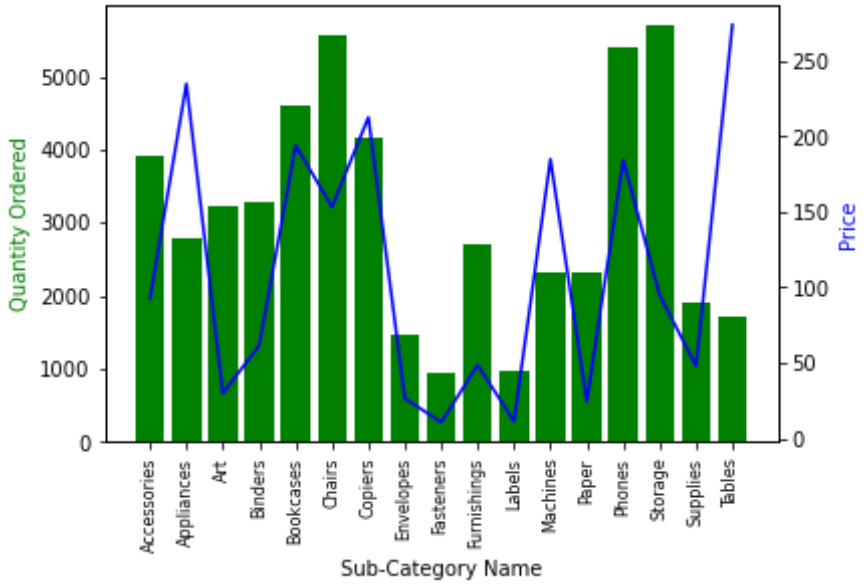
In [39]:
```
prices = df.groupby ('Sub-Category').mean()['Price']
```

In [40]:
```
ig, ax1 =plt.subplots()
```

```
ax2 = ax1.twinx()
ax1.bar(category,quantity_order, color="green")
ax2.plot(category,prices, 'b-')

ax1.set_xlabel('Sub-Category Name')
ax1.set_ylabel ('Quantity Ordered', color="green")
ax2.set_ylabel('Price', color='blue')
ax1.set_xticklabels (category, rotation = "vertical", size = 8)
plt.show()
```

```
<ipython-input-40-939dba4eb7fc>:10: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax1.set_xticklabels (category, rotation = "vertical", size = 8)
```



What is the relation between Category, Quantity and Price?

In [45]:
```
category_group = df.groupby('Category')
quantity_ordered = category_group.sum()['Quantity']
categories = [product for product, df in category_group]
```

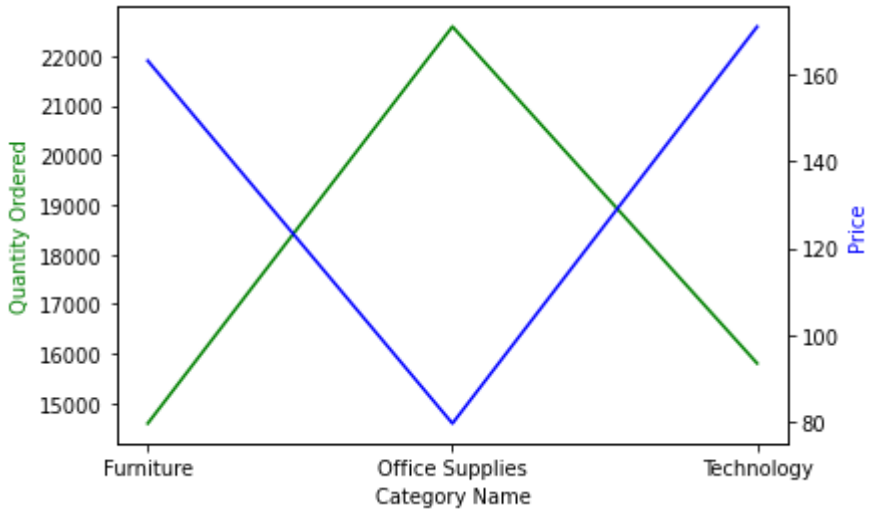
In [46]:
```
price = df.groupby ('Category').mean()['Price']
```

In [47]:
```
fig, ax1 = plt.subplots()

ax2 = ax1. twinx()
ax1.plot (categories, quantity_ordered, color = "green")
ax2.plot (categories,price, 'b-')

ax1.set_xlabel ('Category Name')
ax1.set_ylabel ('Quantity Ordered', color = 'green')
ax2.set_ylabel ('Price', color = 'blue')
```

Out[47]:
```
Text(0, 0.5, 'Price')
```



In [ ]: