In [1]:

```python
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:

```python
market = pd.read_excel('C:/Users/Raghavendra K/Downloads/superstore_sales.xlsx')
```

In [3]:

```python
market.head(5)
```

Out[3]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	category	sub_category
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Africa	Africa	...	Office Supplies	Storage
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Office Supplies	Supplies
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EMEA	EMEA	...	Office Supplies	Storage
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	EU	North	...	Office Supplies	Paper
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Furniture	Furnishings

5 rows × 21 columns

In [47]:

```python
market.tail()
```

Out[47]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	category	sub_cate
51285	CA-2014-115427	2014-12-31	2015-01-04	Standard Class	Erica Bern	Corporate	California	United States	US	West	...	Office Supplies	Bir
51286	MO-2014-2560	2014-12-31	2015-01-05	Standard Class	Liz Preis	Consumer	Souss-Massa-Draâ	Morocco	Africa	Africa	...	Office Supplies	Bir
51287	MX-2014-110527	2014-12-31	2015-01-02	Second Class	Charlotte Melton	Consumer	Managua	Nicaragua	LATAM	Central	...	Office Supplies	La
51288	MX-2014-114783	2014-12-31	2015-01-06	Standard Class	Tamara Dahlen	Consumer	Chihuahua	Mexico	LATAM	North	...	Office Supplies	La
51289	CA-2014-156720	2014-12-31	2015-01-04	Standard Class	Jill Matthias	Consumer	Colorado	United States	US	West	...	Office Supplies	Faste

5 rows × 21 columns

In [48]:

```python
market.shape
```

Out[48]:

```
(51290, 21)
```

In [49]:

```python
market.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype         
---  ------          --------------  -----         
 0   order_id        51290 non-null  object        
 1   order_date      51290 non-null  datetime64[ns]
 2   ship_date       51290 non-null  datetime64[ns]
 3   ship_mode       51290 non-null  object        
 4   customer_name   51290 non-null  object        
 5   segment         51290 non-null  object        
 6   state           51290 non-null  object        
 7   country         51290 non-null  object        
 8   market          51290 non-null  object        
 9   region          51290 non-null  object        
 10  product_id      51290 non-null  object        
 11  category        51290 non-null  object        
 12  sub_category    51290 non-null  object        
 13  product_name    51290 non-null  object        
 14  sales           51290 non-null  float64       
 15  quantity        51290 non-null  int64         
 16  discount        51290 non-null  float64       
 17  profit          51290 non-null  float64       
 18  shipping_cost   51290 non-null  float64       
 19  order_priority  51290 non-null  object        
 20  year            51290 non-null  int64         
dtypes: datetime64[ns](2), float64(4), int64(2), object(13)
memory usage: 8.2+ MB
```

In [4]:
```
market.describe()
```

Out[4]:

	sales	quantity	discount	profit	shipping_cost	year
count	51290.000000	51290.000000	51290.000000	51290.000000	51290.000000	51290.000000
mean	246.490581	3.476545	0.142908	28.641740	26.375818	2012.777208
std	487.565361	2.278766	0.212280	174.424113	57.296810	1.098931
min	0.444000	1.000000	0.000000	-6599.978000	0.002000	2011.000000
25%	30.758625	2.000000	0.000000	0.000000	2.610000	2012.000000
50%	85.053000	3.000000	0.000000	9.240000	7.790000	2013.000000
75%	251.053200	5.000000	0.200000	36.810000	24.450000	2014.000000
max	22638.480000	14.000000	0.850000	8399.976000	933.570000	2014.000000

In [21]:
```
market.isnull().sum()
```

Out[21]:
```
order_id          0
order_date        0
ship_date         0
ship_mode         0
customer_name     0
segment           0
state             0
country           0
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
year              0
dtype: int64
```

In [23]:
```
market.columns
```

Out[23]:
```
Index(['order_id', 'order_date', 'ship_date', 'ship_mode', 'customer_name',
       'segment', 'state', 'country', 'market', 'region', 'product_id',
       'category', 'sub_category', 'product_name', 'sales', 'quantity',
       'discount', 'profit', 'shipping_cost', 'order_priority', 'year'],
      dtype='object')
```

In [30]:
```
market['order_date'].max()
```

Out[30]:
```
Timestamp('2014-12-31 00:00:00')
```

In [31]:
```
market['order_date'].min()
```

Out[31]:
```
Timestamp('2011-01-01 00:00:00')
```

```
In [14]: market['country'].max()
```

```
Out[14]: 'Zimbabwe'
```

```
In [16]: market['country'].min()
```

```
Out[16]: 'Afghanistan'
```

```
In [69]: market['shipping_cost'].max()
```

```
Out[69]: 933.57
```

```
In [70]: market['shipping_cost'].min()
```

```
Out[70]: 0.002
```

```
In [35]: market['month_year']=market['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

```
In [73]: market['shipping_cost'].mode()
```

```
Out[73]: 0    0.35
         Name: shipping_cost, dtype: float64
```

```
In [75]: market['sales'].max()
```

```
Out[75]: 22638.48
```

```
In [10]: market['order_date']
```

```
Out[10]: 0       2011-01-01
         1       2011-01-01
         2       2011-01-01
         3       2011-01-01
         4       2011-01-01
                    ...
         51285   2014-12-31
         51286   2014-12-31
         51287   2014-12-31
         51288   2014-12-31
         51289   2014-12-31
         Name: order_date, Length: 51290, dtype: datetime64[ns]
```

```
In [37]: market['month_year']=market['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

```
In [44]: market.groupby('month_year').sum()
```

Out[44]:

	sales	quantity	discount	profit	shipping_cost	year
month_year						
11-01	98898.48886	1463	68.758	8321.80096	10544.78800	870763
11-02	91152.15698	1224	52.252	12417.90698	10681.16300	760158
11-03	145729.36736	1836	74.212	15303.56826	13096.18550	1083929
11-04	116915.76418	2020	80.782	12902.32438	12954.52000	1134204
11-05	146747.83610	2013	82.382	12183.82870	16443.20600	1138226
11-06	215207.38022	3112	159.534	23415.24702	23813.10900	1844087
11-07	115510.41912	1774	80.086	5585.00352	11844.47600	995445
11-08	207581.49122	3035	121.462	23713.66772	22001.13600	1765658
11-09	290214.45534	3707	137.678	35776.88394	29664.85100	2115572
11-10	199071.26404	2727	110.192	25963.41834	21380.08200	1556514
11-11	298496.53752	4039	178.836	32709.17772	34701.99800	2290529
11-12	333925.73460	4493	187.220	40647.98400	37144.83100	2539893
12-01	135780.72024	1845	74.454	10401.63764	13665.74900	1084468
12-02	100510.21698	1473	62.784	15000.09618	11393.72600	863148
12-03	163076.77116	2237	101.682	17992.91756	16170.78500	1331944
12-04	161052.26952	2250	93.248	17366.96722	16767.86200	1321884
12-05	208364.89124	2921	114.272	29876.70374	23801.61700	1690080
12-06	256175.69842	3671	168.284	34407.15362	28155.90000	2285632
12-07	145236.78512	2321	104.404	15585.38842	17334.43500	1325908
12-08	303142.94238	3818	136.166	43573.87858	32038.73000	2178996
12-09	289389.16564	4205	169.070	27776.18034	28023.17800	2460676
12-10	252939.85020	3563	135.866	30662.88270	25085.74000	1991880
12-11	323512.41690	5193	215.868	31820.72180	33489.74100	2937520
12-12	338256.96660	4614	172.676	32950.75130	37563.36100	2583408
13-01	199185.90738	2413	91.442	26810.55968	21677.43200	1427217
13-02	167239.65040	2102	78.012	25340.02610	16911.85000	1217865
13-03	198594.03012	2686	114.384	23433.77462	21268.01000	1541958
13-04	177821.31684	2688	116.116	19462.03844	19133.23400	1580205
13-05	260498.56470	3808	153.092	28495.69410	28315.21100	2127741
13-06	396519.61190	5327	213.642	45478.41340	42814.02600	3079890
13-07	229928.95200	3252	125.644	28863.82720	24501.84236	1862025
13-08	326488.78936	4934	202.640	31023.66846	35673.08800	2902746
13-09	376619.24568	5793	240.674	38905.66778	38488.40000	3385866
13-10	293406.64288	3883	160.860	42433.22258	31174.68400	2214300
13-11	373989.36010	5556	215.324	48062.99670	41407.16700	3212748
13-12	405454.37802	5694	223.692	50202.87112	43183.80000	3224826
14-01	241268.55566	3122	127.928	28001.38626	24870.80100	1848852
14-02	184837.35556	2482	111.126	19751.69996	19525.80000	1522584
14-03	263100.77262	3722	142.016	37357.26052	26838.63554	2150952
14-04	242771.86130	3594	164.000	23782.30120	26272.71800	2116714
14-05	288401.04614	4300	188.986	33953.55774	31882.58300	2585976
14-06	401814.06310	6009	251.462	43778.60280	41894.07600	3520472
14-07	258705.68048	3637	163.512	28035.87258	29581.73300	2189218
14-08	456619.94236	5824	217.672	53542.89496	46759.35300	3373450
14-09	481157.24370	6837	272.094	67979.45110	53485.43000	4064252
14-10	422766.62916	5876	233.752	58209.83476	44622.41400	3274764
14-11	555279.02700	7706	304.384	62856.58790	59918.35500	4324058
14-12	503143.69348	7513	335.106	46916.52068	54853.89100	4336142

In [45]:
```
market['month_year']=market['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

In [46]:
```
market.groupby('month_year').sum()['sales']
```

Out[46]:
```
month_year
11-01     98898.48886
11-02     91152.15698
11-03    145729.36736
11-04    116915.76418
11-05    146747.83610
11-06    215207.38022
11-07    115510.41912
11-08    207581.49122
11-09    290214.45534
11-10    199071.26404
11-11    298496.53752
11-12    333925.73460
12-01    135780.72024
12-02    100510.21698
12-03    163076.77116
12-04    161052.26952
12-05    208364.89124
12-06    256175.69842
12-07    145236.78512
12-08    303142.94238
12-09    289389.16564
12-10    252939.85020
12-11    323512.41690
12-12    338256.96660
13-01    199185.90738
13-02    167239.65040
13-03    198594.03012
13-04    177821.31684
13-05    260498.56470
13-06    396519.61190
13-07    229928.95200
13-08    326488.78936
13-09    376619.24568
13-10    293406.64288
13-11    373989.36010
13-12    405454.37802
14-01    241268.55566
14-02    184837.35556
14-03    263100.77262
14-04    242771.86130
14-05    288401.04614
14-06    401814.06310
14-07    258705.68048
14-08    456619.94236
14-09    481157.24370
14-10    422766.62916
14-11    555279.02700
14-12    503143.69348
Name: sales, dtype: float64
```

In [47]:
```
market['month_year']=market['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

In [48]:
```
market.groupby('month_year').sum()['sales'].reset_index()
```

Out[48]:

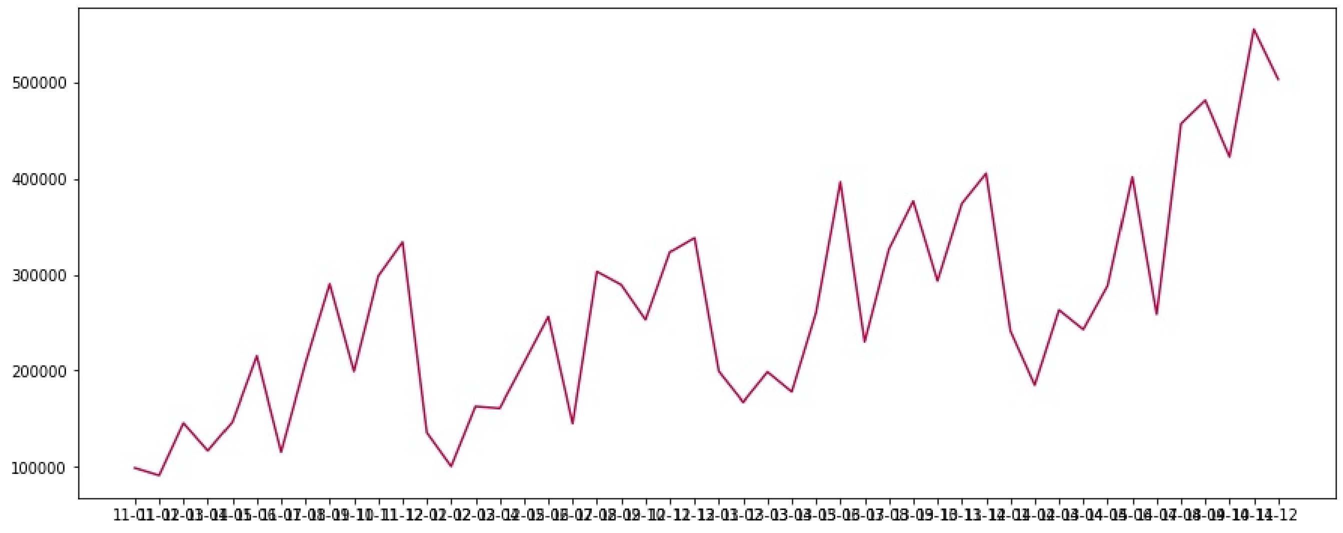
	month_year	sales
0	11-01	98898.48886
1	11-02	91152.15698
2	11-03	145729.36736
3	11-04	116915.76418
4	11-05	146747.83610
5	11-06	215207.38022
6	11-07	115510.41912
7	11-08	207581.49122
8	11-09	290214.45534
9	11-10	199071.26404
10	11-11	298496.53752
11	11-12	333925.73460
12	12-01	135780.72024
13	12-02	100510.21698
14	12-03	163076.77116
15	12-04	161052.26952
16	12-05	208364.89124
17	12-06	256175.69842
18	12-07	145236.78512
19	12-08	303142.94238
20	12-09	289389.16564
21	12-10	252939.85020
22	12-11	323512.41690
23	12-12	338256.96660
24	13-01	199185.90738
25	13-02	167239.65040
26	13-03	198594.03012
27	13-04	177821.31684
28	13-05	260498.56470
29	13-06	396519.61190
30	13-07	229928.95200
31	13-08	326488.78936
32	13-09	376619.24568
33	13-10	293406.64288
34	13-11	373989.36010
35	13-12	405454.37802
36	14-01	241268.55566
37	14-02	184837.35556
38	14-03	263100.77262
39	14-04	242771.86130
40	14-05	288401.04614
41	14-06	401814.06310
42	14-07	258705.68048
43	14-08	456619.94236
44	14-09	481157.24370
45	14-10	422766.62916
46	14-11	555279.02700
47	14-12	503143.69348

In [49]:
```
market['month_year']=market['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

In [50]:
```
market_trend = market.groupby('month_year').sum()['sales'].reset_index()
```

In [73]:
```
plt.figure(figsize=(15,6))
plt.plot(market_trend['month_year'],market_trend['sales'], color= '#b80045')
```

Out[73]:
```
[<matplotlib.lines.Line2D at 0x24592806770>]
```



In [81]:
```python
pd.DataFrame(market.groupby('product_name').sum()['sales'])
```

Out[81]:

	sales
product_name	
"While you Were Out" Message Book, One Form per Page	25.228
#10 Gummed Flap White Envelopes, 100/Box	41.300
#10 Self-Seal White Envelopes	108.682
#10 White Business Envelopes,4 1/8 x 9 1/2	488.904
#10- 4 1/8" x 9 1/2" Recycled Envelopes	286.672
...	...
iKross Bluetooth Portable Keyboard + Cell Phone Stand Holder + Brush for Apple iPhone 5S 5C 5, 4S 4	477.660
iOttie HLCRIO102 Car Mount	215.892
iOttie XL Car Mount	223.888
invisibleSHIELD by ZAGG Smudge-Free Screen Protector	442.554
netTALK DUO VoIP Telephone Service	1112.788

3788 rows × 1 columns

In [82]:
```python
prod_sales = pd.DataFrame(market.groupby('product_name').sum()['sales'])
```

In [86]:
```python
prod_sales = prod_sales.sort_values('sales', ascending=False)
```

In [89]:
```python
prod_sales.head(10)
```

Out[89]:

	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

In [18]:
```python
market.groupby('product_name').sum()['quantity']
```

Out[18]:
```
product_name
"While you Were Out" Message Book, One Form per Page                                  8
#10 Gummed Flap White Envelopes, 100/Box                                             11
#10 Self-Seal White Envelopes                                                        10
#10 White Business Envelopes,4 1/8 x 9 1/2                                           32
#10- 4 1/8" x 9 1/2" Recycled Envelopes                                              37
                                                                                     ..
iKross Bluetooth Portable Keyboard + Cell Phone Stand Holder + Brush for Apple iPhone 5S 5C 5, 4S 4    24
iOttie HLCRIO102 Car Mount                                                           12
iOttie XL Car Mount                                                                  14
invisibleSHIELD by ZAGG Smudge-Free Screen Protector                                 29
netTALK DUO VoIP Telephone Service                                                   26
Name: quantity, Length: 3788, dtype: int64
```

In [19]:
```
pd.DataFrame(market.groupby('product_name').sum()['quantity'])
```

Out[19]:

	quantity
product_name	
"While you Were Out" Message Book, One Form per Page	8
#10 Gummed Flap White Envelopes, 100/Box	11
#10 Self-Seal White Envelopes	10
#10 White Business Envelopes,4 1/8 x 9 1/2	32
#10- 4 1/8" x 9 1/2" Recycled Envelopes	37
...	...
iKross Bluetooth Portable Keyboard + Cell Phone Stand Holder + Brush for Apple iPhone 5S 5C 5, 4S 4	24
iOttie HLCRIO102 Car Mount	12
iOttie XL Car Mount	14
invisibleSHIELD by ZAGG Smudge-Free Screen Protector	29
netTALK DUO VoIP Telephone Service	26

3788 rows × 1 columns

In [23]:
```
most_sales_prod=pd.DataFrame(market.groupby('product_name').sum()['quantity'])
```

In [24]:
```
most_sales_prod = most_sales_prod.sort_values('quantity', ascending=False)
```

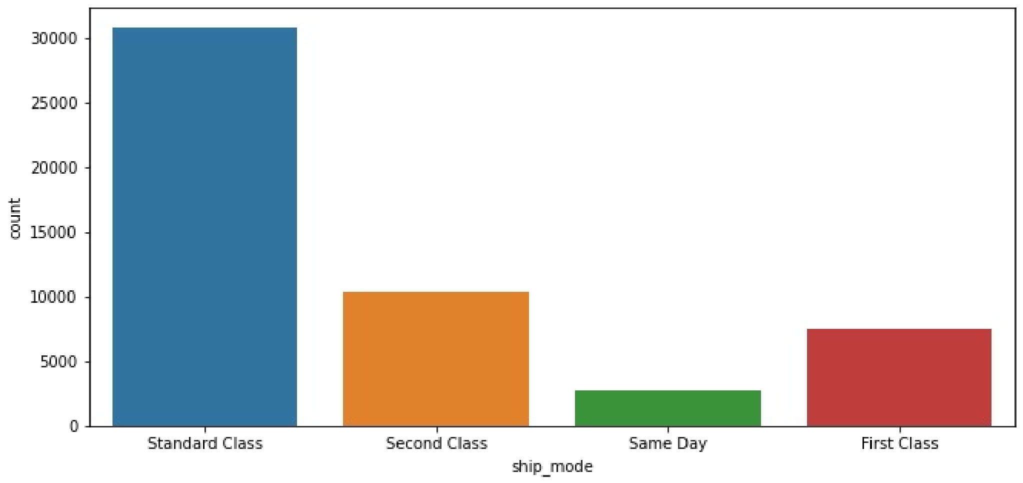
In [26]:
```
most_sales_prod[:10]
```

Out[26]:

	quantity
product_name	
Staples	876
Cardinal Index Tab, Clear	337
Eldon File Cart, Single Width	321
Rogers File Cart, Single Width	262
Sanford Pencil Sharpener, Water Color	259
Stockwell Paper Clips, Assorted Sizes	253
Avery Index Tab, Clear	252
Ibico Index Tab, Clear	251
Smead File Cart, Single Width	250
Stanley Pencil Sharpener, Water Color	242

In [30]:
```
plt.figure(figsize=(10.8,5))
sns.countplot(market['ship_mode'])
plt.show()
```

```
C:\Users\Raghavendra K\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argum
ent will be `data`, and passing other arguments without an explicit keyword will result in an error or misinter
pretation.
  warnings.warn(
```



In [33]:
```
market.groupby(['category','sub_category']).sum()
```

Out[33]:

category	sub_category	sales	quantity	discount	profit	shipping_cost	year
Furniture	Bookcases	1.466572e+06	8310	370.710	161924.41950	155481.9670	4852847
	Chairs	1.501682e+06	12336	560.120	141973.79750	164229.3520	6911889
	Furnishings	3.855783e+05	11225	478.880	46967.42550	40746.7660	6380451
	Tables	7.570419e+05	3083	250.320	-64083.38870	79861.3940	1732979
Office Supplies	Appliances	1.011064e+06	6078	248.700	141680.58940	108300.5860	3532371
	Art	3.720920e+05	16301	573.080	57953.91090	41287.1420	9828413
	Binders	4.619115e+05	21429	1102.480	72449.84600	48181.7120	12382700
	Envelopes	1.709043e+05	8380	320.810	29601.11630	18547.4880	4901146
	Fasteners	8.324232e+04	8390	340.240	11525.42410	9053.3380	4870955
	Labels	7.340403e+04	9322	313.890	15010.51200	8059.6750	5245285
	Paper	2.442917e+05	12822	387.300	59207.68270	26660.8450	7121179
	Storage	1.127086e+06	16917	700.490	108461.48980	120546.0320	10182612
	Supplies	2.430742e+05	8543	310.200	22583.26310	24811.5270	4881018
Technology	Accessories	7.492370e+05	10946	370.480	129626.30620	83513.3340	6189269
	Copiers	1.509436e+06	7454	260.418	258567.54818	159496.2049	4474471
	Machines	7.790601e+05	4906	252.000	58867.87300	79135.8485	2990958
	Phones	1.706824e+06	11870	489.610	216717.00580	184902.4920	6756800

In [35]:
```
market.groupby(['category','sub_category']).sum()['profit']
```

Out[35]:
```
category         sub_category
Furniture        Bookcases       161924.41950
                 Chairs          141973.79750
                 Furnishings      46967.42550
                 Tables          -64083.38870
Office Supplies  Appliances      141680.58940
                 Art              57953.91090
                 Binders          72449.84600
                 Envelopes        29601.11630
                 Fasteners        11525.42410
                 Labels           15010.51200
                 Paper            59207.68270
                 Storage         108461.48980
                 Supplies         22583.26310
Technology       Accessories     129626.30620
                 Copiers         258567.54818
                 Machines         58867.87300
                 Phones          216717.00580
Name: profit, dtype: float64
```

In [7]:
```
market.groupby(['category']).sum()['profit']
```

In [7]:
```
market.groupby(['category']).sum()['profit']
```

Out[7]:
```
category
Furniture          286782.25380
Office Supplies    518473.83430
Technology         663778.73318
Name: profit, dtype: float64
```

In [8]:
```
pd.DataFrame(market.groupby(['category','sub_category']).sum()['profit'])
```

Out[8]:

		profit
category	**sub_category**	
Furniture	**Bookcases**	161924.41950
	Chairs	141973.79750
	Furnishings	46967.42550
	Tables	-64083.38870
Office Supplies	**Appliances**	141680.58940
	Art	57953.91090
	Binders	72449.84600
	Envelopes	29601.11630
	Fasteners	11525.42410
	Labels	15010.51200
	Paper	59207.68270
	Storage	108461.48980
	Supplies	22583.26310
Technology	**Accessories**	129626.30620
	Copiers	258567.54818
	Machines	58867.87300
	Phones	216717.00580

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js