```
In [1]: import os
```

```
In [2]: os.getcwd()
```

```
Out[2]: 'C:\\Users\\Kalekye'
```

```
In [3]: os.chdir('C:\\Users\\Kalekye\\Desktop\\Microsoft Excel Files')
```

```
In [4]: os.getcwd()
```

```
Out[4]: 'C:\\Users\\Kalekye\\Desktop\\Microsoft Excel Files'
```

```
In [5]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sns
```

```
In [7]: df = pd.read_excel('superstore_sales.xlsx')
        df.head(10)
```

Out[7]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	mar
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Af
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EN
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
5	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
6	CA-2011-1510	2011-01-02	2011-01-06	Standard Class	Magdelene Morse	Consumer	Ontario	Canada	Can
7	IN-2011-79397	2011-01-03	2011-01-03	Same Day	Kean Nguyen	Corporate	New South Wales	Australia	Al
8	ID-2011-80230	2011-01-03	2011-01-09	Standard Class	Ken Lonsdale	Consumer	Auckland	New Zealand	Al
9	IZ-2011-4680	2011-01-03	2011-01-07	Standard Class	Lindsay Williams	Corporate	Ninawa	Iraq	EN

10 rows × 21 columns

In [9]:
```
cat_features = [i for i in df.columns if df[i].dtype =='O']
cat_features
```

Out[9]:
```
['order_id',
 'ship_mode',
 'customer_name',
 'segment',
 'state',
 'country',
 'market',
 'region',
 'product_id',
 'category',
 'sub_category',
 'product_name',
 'order_priority']
```

In [10]:
```
#Summary of the dataset
df.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 [11]:
```
#check for missing values
df.isnull().sum()
```

Out[11]:
```
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 [12]:
```
#getting descriptive staristics summary
df.describe()
```

Out[12]:

	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 [13]:
```
#Exploratory Data Analysis
#1. What is the overall sales trend?
df['order_date'].min()
```

Out[13]:
```
Timestamp('2011-01-01 00:00:00')
```

In [14]:
```
df['order_date'].max()
```

Out[14]:
```
Timestamp('2014-12-31 00:00:00')
```

In [15]:
```
#Getting month year from the dataset
df['month_year'] = df['order_date'].apply(lambda x: x.strftime('%Y-%m'))
```

In [21]:
```
#Grouping month year
df = df.groupby('month_year').sum()['sales'].reset_index()
```

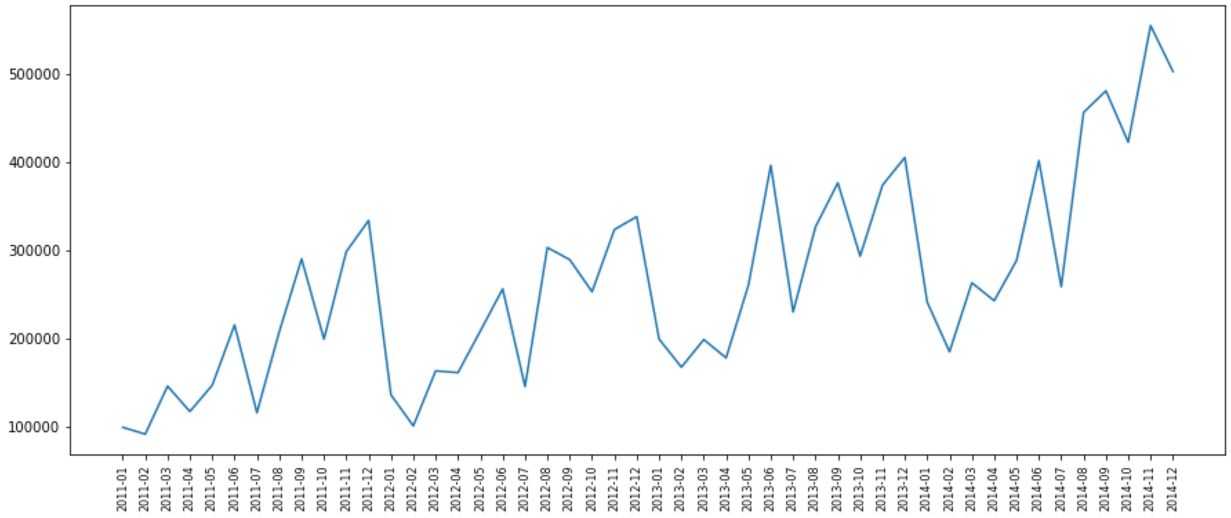
In [22]:
```
df
```

Out[22]:

	month_year	sales
0	2011-01	98898.48886
1	2011-02	91152.15698
2	2011-03	145729.36736
3	2011-04	116915.76418
4	2011-05	146747.83610
5	2011-06	215207.38022
6	2011-07	115510.41912
7	2011-08	207581.49122
8	2011-09	290214.45534
9	2011-10	199071.26404
10	2011-11	298496.53752
11	2011-12	333925.73460
12	2012-01	135780.72024
13	2012-02	100510.21698
14	2012-03	163076.77116
15	2012-04	161052.26952
16	2012-05	208364.89124
17	2012-06	256175.69842
18	2012-07	145236.78512
19	2012-08	303142.94238
20	2012-09	289389.16564
21	2012-10	252939.85020
22	2012-11	323512.41690
23	2012-12	338256.96660
24	2013-01	199185.90738
25	2013-02	167239.65040
26	2013-03	198594.03012
27	2013-04	177821.31684
28	2013-05	260498.56470
29	2013-06	396519.61190
30	2013-07	229928.95200
31	2013-08	326488.78936
32	2013-09	376619.24568

	month_year	sales
33	2013-10	293406.64288
34	2013-11	373989.36010
35	2013-12	405454.37802
36	2014-01	241268.55566
37	2014-02	184837.35556
38	2014-03	263100.77262
39	2014-04	242771.86130
40	2014-05	288401.04614
41	2014-06	401814.06310
42	2014-07	258705.68048
43	2014-08	456619.94236
44	2014-09	481157.24370
45	2014-10	422766.62916
46	2014-11	555279.02700
47	2014-12	503143.69348

In [24]:
```
#setting the figure size
plt.figure(figsize=(15,6))
plt.plot(df['month_year'], df['sales'])
plt.xticks(rotation='vertical', size=8)
plt.show()
```



In [28]:
```
#2. What are the top 10 products by sales
df = pd.read_excel('superstore_sales.xlsx')
df.head(10)
```

Out[28]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	mar
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Af
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EN
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
5	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	Al
6	CA-2011-1510	2011-01-02	2011-01-06	Standard Class	Magdelene Morse	Consumer	Ontario	Canada	Can
7	IN-2011-79397	2011-01-03	2011-01-03	Same Day	Kean Nguyen	Corporate	New South Wales	Australia	Al
8	ID-2011-80230	2011-01-03	2011-01-09	Standard Class	Ken Lonsdale	Consumer	Auckland	New Zealand	Al
9	IZ-2011-4680	2011-01-03	2011-01-07	Standard Class	Lindsay Williams	Corporate	Ninawa	Iraq	EN

10 rows × 21 columns

In [32]:
```
#Grouping product name column
prod_sales = pd.DataFrame(df.groupby('product_name').sum()['sales'])
```

In [36]:
```
#sorting prod_sales column in descending order
prod_sales.sort_values('sales',ascending=False)
```

Out[36]:

product_name	sales
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
...	...
Avery Hi-Liter Pen Style Six-Color Fluorescent Set	7.7000
Grip Seal Envelopes	7.0720
Xerox 20	6.4800
Avery 5	5.7600
Eureka Disposable Bags for Sanitaire Vibra Groomer I Upright Vac	1.6240

3788 rows × 1 columns

In [37]:

Out[37]:

product_name	sales
"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
#10- 4 1/8" x 9 1/2" Security-Tint Envelopes	146.688
#10-4 1/8" x 9 1/2" Premium Diagonal Seam Envelopes	176.288
#6 3/4 Gummed Flap White Envelopes	71.280
1.7 Cubic Foot Compact "Cube" Office Refrigerators	2706.080
1/4 Fold Party Design Invitations & White Envelopes, 24 8-1/2" X 11" Cards, 25 Env./Pack	49.980

In [42]:

```
#which are the most selling products
#Grouping product name
most_sell_prod = pd.DataFrame(df.groupby('product_name').sum()['quantity'])
most_sell_prod
```

Out[42]:

product_name	quantity
"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

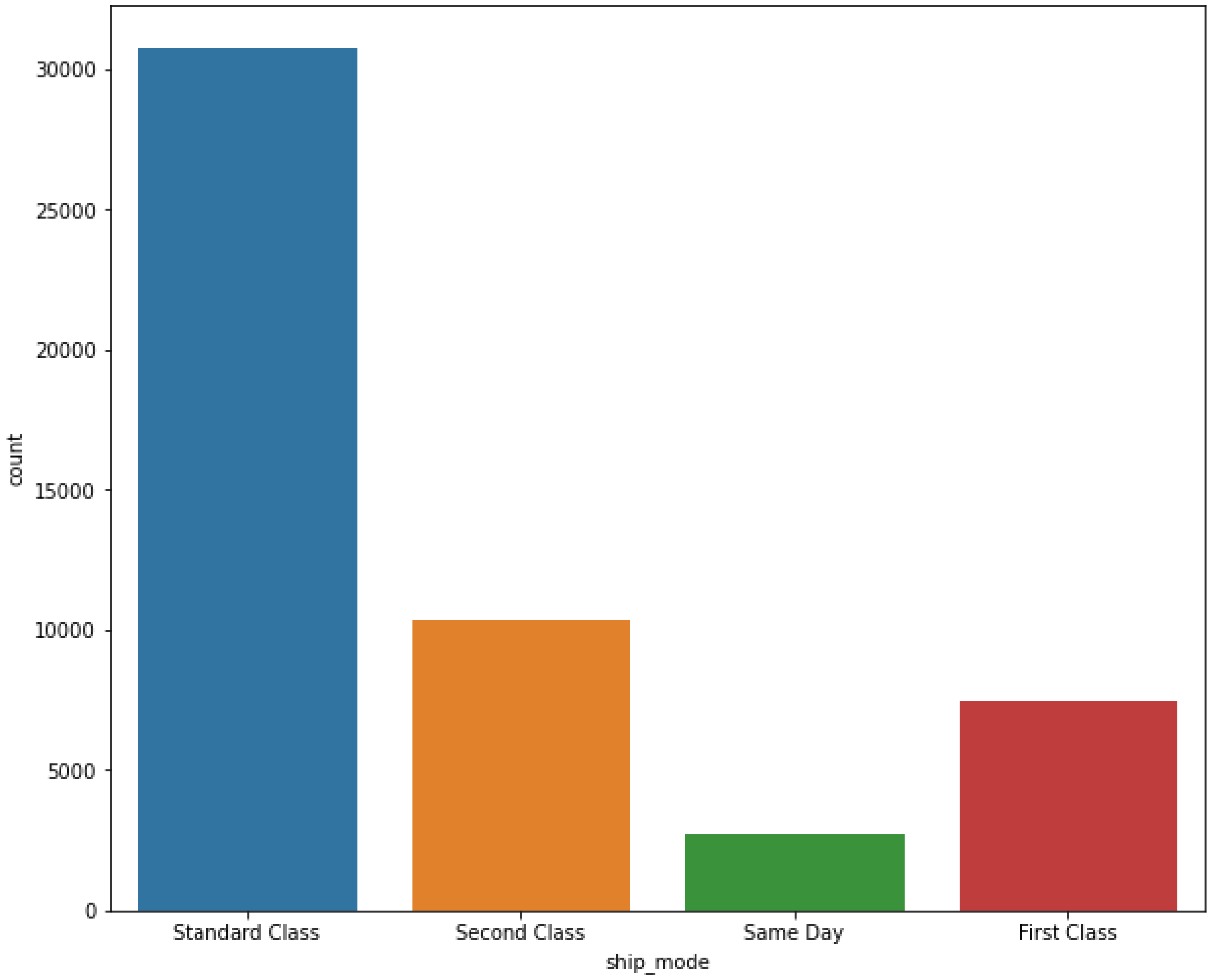
In [44]:
```
#sorting most_sell_products
most_sell_prod = most_sell_prod.sort_values('quantity',ascending=False)
most_sell_prod[:10]
```

Out[44]:

product_name	quantity
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 [47]:
```
#3. What is the preferred ship mode
#Setting figure size
plt.figure(figsize=(10,8.5))
#ploting shipmode
sns.countplot(df['ship_mode'])
plt.show()
```

```
C:\Users\Kalekye\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarnin
g: Pass the following variable as a keyword arg: x. From version 0.12, the only valid
positional argument will be `data`, and passing other arguments without an explicit k
eyword will result in an error or misinterpretation.
  warnings.warn(
```



In [53]:

```
# Which are the most profitable category and sub_category
#Grouping category and sub_category
#then sort in descending order
cat_subcat_profit = pd.DataFrame(df.groupby(['category','sub_category']).sum()['profit
cat_subcat_profit.sort_values(['category','profit'],ascending=False)
```

Out[53]:

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

In [ ]: