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

In [2]:
```
df = pd.read_excel('superstore_sales.xlsx')
```

In [ ]:
```
### ADUDIT OF DATA
```

In [4]:
```
df.head()
```

Out[4]:

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

5 rows × 21 columns

In [5]:
```
df.tail()
```

Out[5]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	re
51285	CA-2014-115427	2014-12-31	2015-01-04	Standard Class	Erica Bern	Corporate	California	United States	US	V
51286	MO-2014-2560	2014-12-31	2015-01-05	Standard Class	Liz Preis	Consumer	Souss-Massa-Draâ	Morocco	Africa	A
51287	MX-2014-110527	2014-12-31	2015-01-02	Second Class	Charlotte Melton	Consumer	Managua	Nicaragua	LATAM	Ce
51288	MX-2014-114783	2014-12-31	2015-01-06	Standard Class	Tamara Dahlen	Consumer	Chihuahua	Mexico	LATAM	N
51289	CA-2014-156720	2014-12-31	2015-01-04	Standard Class	Jill Matthias	Consumer	Colorado	United States	US	V

5 rows × 21 columns

In [6]:
```python
df.shape
```
Out[6]:
```
(51290, 21)
```
In [7]:
```python
df.isnull().sum().sum()
```
Out[7]:
```
0
```
In [8]:
```python
df.columns
```
Out[8]:
```
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 [11]:
```python
#summary of datset
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 [12]:
```python
## descriptive statistic of dataset
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
[illegible]	[illegible]25	2.000000	0.000000	0.000000	2.610000	2012.000000

Loading [MathJax]/extensions/Safe.js

	sales	quantity	discount	profit	shipping_cost	year
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

EXPLORATORY DATA ANALYSIS

What is the overall sales trend ?

In [17]:

```
df['order_date'].min()
```

Out[17]:

```
Timestamp('2011-01-01 00:00:00')
```

In [18]:

```
df['order_date'].max()
```

Out[18]:

```
Timestamp('2014-12-31 00:00:00')
```

In [22]:

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

Out[22]:

```
0        2011-01
1        2011-01
2        2011-01
3        2011-01
4        2011-01
          ...
51285    2014-12
51286    2014-12
51287    2014-12
51288    2014-12
51289    2014-12
Name: month_year, Length: 51290, dtype: object
```

In [24]:

```
## grouping month year
df_trend = df.groupby('month_year').sum()['sales'].reset_index()
df_trend
```

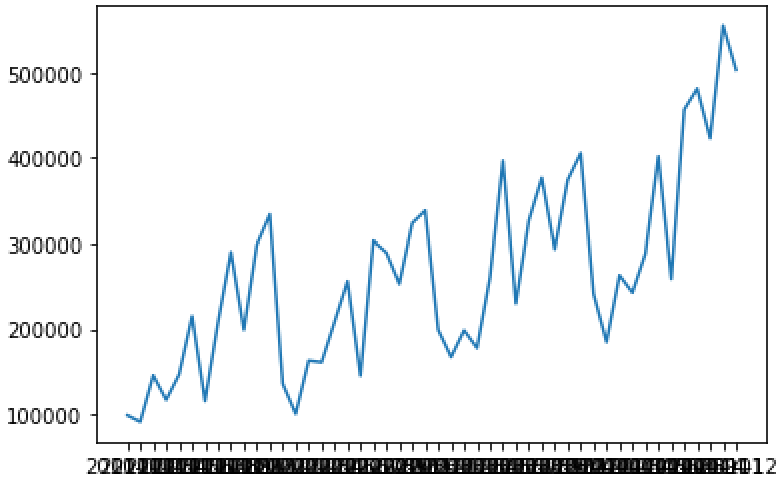
Out[24]:

	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

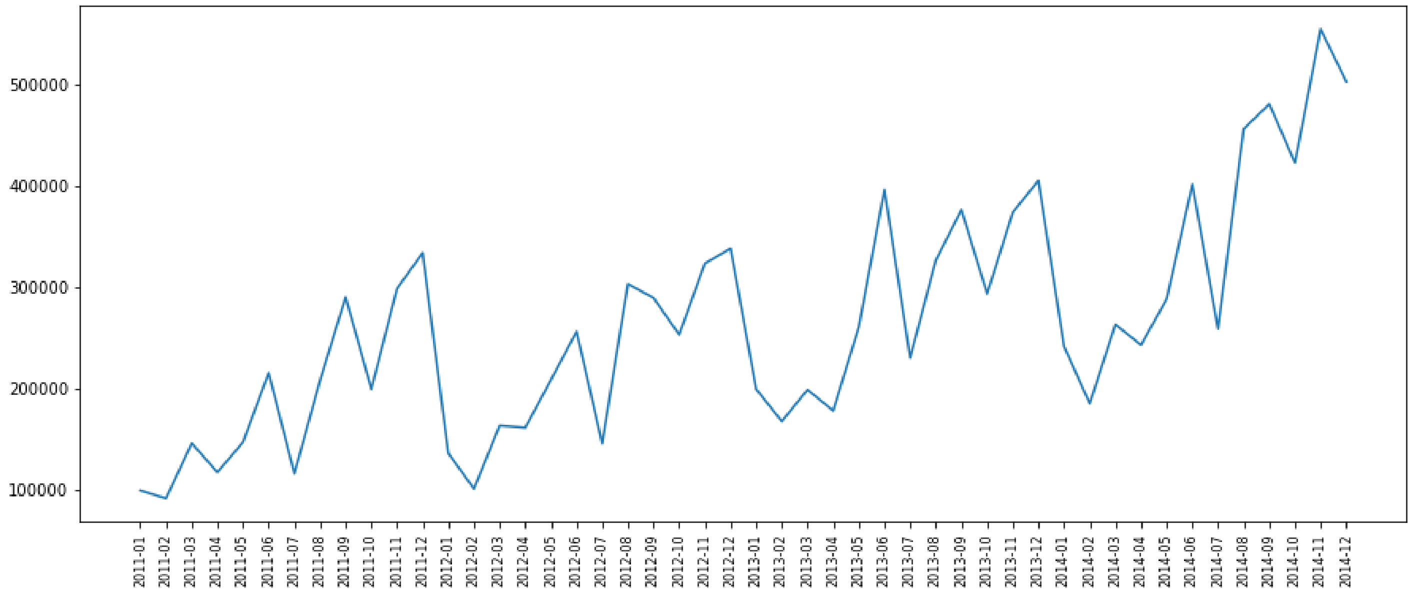
	month_year	sales
9	2011-10	199071.26404
10	2011-11	298496.53752
11	2011-12	333925.73460
12	2012-01	135780.72024
13	2012-02	100510.21698
14	2012-03	163076.77116
15	2012-04	161052.26952
16	2012-05	208364.89124
17	2012-06	256175.69842
18	2012-07	145236.78512
19	2012-08	303142.94238
20	2012-09	289389.16564
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
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 [25]:
```
plt.plot(df_trend['month_year'],df_trend['sales'])
```

Out[25]:
```
[<matplotlib.lines.Line2D at 0x26f7cf671c0>]
```



In [30]:
```
## setting the figure size
plt.figure(figsize=(15,6))
plt.plot(df_trend['month_year'],df_trend['sales'])
plt.xticks(rotation='vertical', size=8)
plt.show()
```



which are the top 10 products by sales

In [31]:
```
df.head()
```

Out[31]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Africa	Africa
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania

Loading [MathJax]/extensions/Safe.js

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EMEA	EMEA
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	EU	North
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania

5 rows × 22 columns

In [37]:
```
## grouping product name column
prod_sales= pd.DataFrame(df.groupby('product_name').sum()['sales'])
prod_sales= prod_sales.sort_values('sales',ascending=False)
prod_sales
```

Out[37]:

	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
...	...
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 [38]:
```
prod_sales[:10]
```

Out[38]:

	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

Loading [MathJax]/extensions/Safe.js

	sales
product_name	
Samsung Smart Phone, Cordless	48653.4600
Nokia Smart Phone, with Caller ID	47877.7857

which are the most selling products

In [42]:
```
## grouping product name
most_sell_product = pd.DataFrame(df.groupby('product_name').sum()['quantity'])
most_sell_product = most_sell_product.sort_values('quantity',ascending=False)
```

In [44]:
```
most_sell_product[:10]
```

Out[44]:

	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

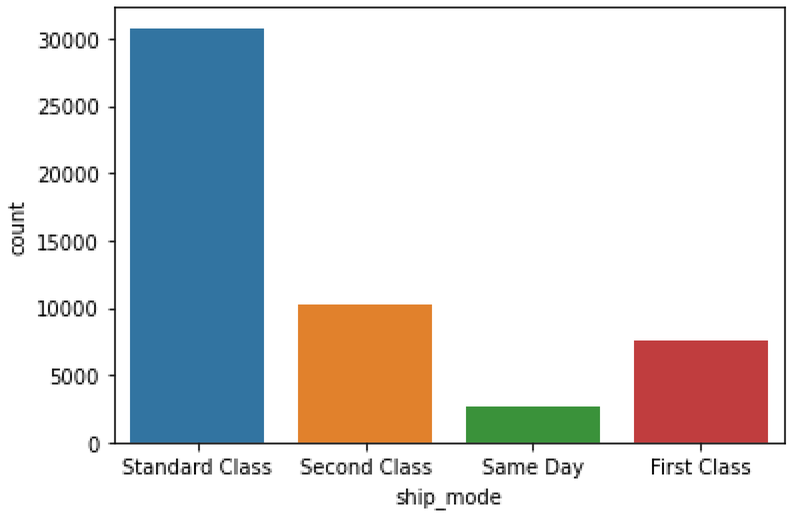
most preferred shipped product

In [46]:
```
## plotting shipmode
sns.countplot(df['ship_mode'])
```

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

Out[46]:
```
<AxesSubplot:xlabel='ship_mode', ylabel='count'>
```

Loading [MathJax]/extensions/Safe.js



which are the most profitable category and sub-category

In [51]:
```
cat_subcat = pd.DataFrame(df.groupby(['category','sub_category']).sum()['profit'])
cat_subcat = cat_subcat.sort_values(['category','sub_category'], ascending=False)
cat_subcat
```

Out[51]:

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

In [ ]:

Loading [MathJax]/extensions/Safe.js