OBJECTIVE

1.What is the overall sales trend?

2.Which are the Top 10 products by sales?

3.Which are the Most Selling Products?

4.Which is the most preferred Ship Mode?

5.Which are the Most Profitable Category and Sub-Category?

IMPORTING REQUIRED LIBRARIES

In [6]:
```
# Data Manipulation
import pandas as pd

# Data Visualisation
import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
```

In [7]:
```
# Importing dataset
df = pd.read_excel('superstore_sales.xlsx')
```

In [9]:
```
# To display First five rows of the dataset
df.head()
```

Out[9]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	category	sub_category	product_name	sales	quantity	discount	pro
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Africa	Africa	...	Office Supplies	Storage	Tenex Lockers, Blue	408.300	2	0.0	106.1
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Office Supplies	Supplies	Acme Trimmer, High Speed	120.366	3	0.1	36.0
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EMEA	EMEA	...	Office Supplies	Storage	Tenex Box, Single Width	66.120	4	0.0	29.6
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	EU	North	...	Office Supplies	Paper	Enermax Note Cards, Premium	44.865	3	0.5	-26.0
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Furniture	Furnishings	Eldon Light Bulb, Duo Pack	113.670	5	0.1	37.7

5 rows × 21 columns

In [10]:
```
# To display Last five rows of the dataset
df.tail()
```

Out[10]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	category	sub_category	product_name	sales	quantity	discount
51285	CA-2014-115427	2014-12-31	2015-01-04	Standard Class	Erica Bern	Corporate	California	United States	US	West	...	Office Supplies	Binders	Cardinal Slant-D Ring Binder, Heavy Gauge Vinyl	13.904	2	0.2
51286	MO-2014-2560	2014-12-31	2015-01-05	Standard Class	Liz Preis	Consumer	Souss-Massa-Draâ	Morocco	Africa	Africa	...	Office Supplies	Binders	Wilson Jones Hole Reinforcements, Clear	3.990	1	0.0
51287	MX-2014-110527	2014-12-31	2015-01-02	Second Class	Charlotte Melton	Consumer	Managua	Nicaragua	LATAM	Central	...	Office Supplies	Labels	Hon Color Coded Labels, 5000 Label Set	26.400	3	0.0
51288	MX-2014-114783	2014-12-31	2015-01-06	Standard Class	Tamara Dahlen	Consumer	Chihuahua	Mexico	LATAM	North	...	Office Supplies	Labels	Hon Legal Exhibit Labels, Alphabetical	7.120	1	0.0
51289	CA-2014-156720	2014-12-31	2015-01-04	Standard Class	Jill Matthias	Consumer	Colorado	United States	US	West	...	Office Supplies	Fasteners	Bagged Rubber Bands	3.024	3	0.2

5 rows × 21 columns

In [11]:
```
# To display Shape of the dataset(No. of rows and columns)
df.shape
```

Out[11]:
```
(51290, 21)
```

In [12]:
```
# Columns present in the dataset
df.columns
```

Out[12]:
```
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 [13]:
```
# A concise 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 [16]:
```
#To check if there are any missing values in the entire data set we use the isnull function, then see if there are any values.
df.isna().sum()
```

Out[16]:
```
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 [ ]:
```
#luckily there are no null values
```

Some descriptive statistics on the data set.

Notice, it only shows the statistics on the numerical columns. From here you can see the following statistics:

1.Row count, which aligns to what the shape attribute showed us.

2.The mean, or average.

3.The standard deviation, or how spread out the data is.

4.The minimum and maximum value of each column

5.The number of items that fall within the first, second, and third percentiles.

In [17]:
```
df.describe().round()
```

Out[17]:

	sales	quantity	discount	profit	shipping_cost	year
count	51290.0	51290.0	51290.0	51290.0	51290.0	51290.0
mean	246.0	3.0	0.0	29.0	26.0	2013.0
std	488.0	2.0	0.0	174.0	57.0	1.0
min	0.0	1.0	0.0	-6600.0	0.0	2011.0
25%	31.0	2.0	0.0	0.0	3.0	2012.0
50%	85.0	3.0	0.0	9.0	8.0	2013.0
75%	251.0	5.0	0.0	37.0	24.0	2014.0
max	22638.0	14.0	1.0	8400.0	934.0	2014.0

EXPLORATORY DATA ANALYSIS :

1.WHAT IS THE OVERALL SALES TREND?

In [18]:
```
# the first date of our data set is..
df['order_date'].min()
```

Out[18]:
```
Timestamp('2011-01-01 00:00:00')
```

In [19]:
```
# the last date of our data set is...
df['order_date'].max()
```

Out[19]:
```
Timestamp('2014-12-31 00:00:00')
```

In [21]:
```
# Getting month and year from order_date
df['month_year'] = df['order_date'].apply(lambda x: x.strftime('%Y-%m'))
df['month_year']
```

Out[21]:
```
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 [22]:
```
df.groupby('month_year').sum()
#by grouping data based on "month_year" column and sum() gives count of sales,quantity.....per month
```

Out[22]:

month_year	sales	quantity	discount	profit	shipping_cost	year
2011-01	98898.48886	1463	68.758	8321.80096	10544.78800	870763
2011-02	91152.15698	1224	52.252	12417.90698	10681.16300	760158
2011-03	145729.36736	1836	74.212	15303.56826	13096.18550	1083929
2011-04	116915.76418	2020	80.782	12902.32438	12954.52000	1134204
2011-05	146747.83610	2013	82.382	12183.82870	16443.20600	1138226
2011-06	215207.38022	3112	159.534	23415.24702	23813.10900	1844087
2011-07	115510.41912	1774	80.086	5585.00352	11844.47600	995445
2011-08	207581.49122	3035	121.462	23713.66772	22001.13600	1765658
2011-09	290214.45534	3707	137.678	35776.88394	29664.85100	2115572
2011-10	199071.26404	2727	110.192	25963.41834	21380.08200	1556514
2011-11	298496.53752	4039	178.836	32709.17772	34701.99800	2290529
2011-12	333925.73460	4493	187.220	40647.98400	37144.83100	2539893
2012-01	135780.72024	1845	74.454	10401.63764	13665.74900	1084468
2012-02	100510.21698	1473	62.784	15000.09618	11393.72600	863148
2012-03	163076.77116	2237	101.682	17992.91756	16170.78500	1331944
2012-04	161052.26952	2250	93.248	17366.96722	16767.86200	1321884
2012-05	208364.89124	2921	114.272	29876.70374	23801.61700	1690080
2012-06	256175.69842	3671	168.284	34407.15362	28155.90000	2285632
2012-07	145236.78512	2321	104.404	15585.38842	17334.43500	1325908
2012-08	303142.94238	3818	136.166	43573.87858	32038.73000	2178996
2012-09	289389.16564	4205	169.070	27776.18034	28023.17800	2460676
2012-10	252939.85020	3563	135.866	30662.88270	25085.74000	1991880
2012-11	323512.41690	5193	215.868	31820.72180	33489.74100	2937520
2012-12	338256.96660	4614	172.676	32950.75130	37563.36100	2583408
2013-01	199185.90738	2413	91.442	26810.55968	21677.43200	1427217
2013-02	167239.65040	2102	78.012	25340.02610	16911.85000	1217865
2013-03	198594.03012	2686	114.384	23433.77462	21268.01000	1541958
2013-04	177821.31684	2688	116.116	19462.03844	19133.23400	1580205
2013-05	260498.56470	3808	153.092	28495.69410	28315.21100	2127741
2013-06	396519.61190	5327	213.642	45478.41340	42814.02600	3079890
2013-07	229928.95200	3252	125.644	28863.82720	24501.84236	1862025
2013-08	326488.78936	4934	202.640	31023.66846	35673.08800	2902746
2013-09	376619.24568	5793	240.674	38905.66778	38488.40000	3385866
2013-10	293406.64288	3883	160.860	42433.22258	31174.68400	2214300
2013-11	373989.36010	5556	215.324	48062.99670	41407.16700	3212748
2013-12	405454.37802	5694	223.692	50202.87112	43183.80000	3224826
2014-01	241268.55566	3122	127.928	28001.38626	24870.80100	1848852
2014-02	184837.35556	2482	111.126	19751.69996	19525.80000	1522584
2014-03	263100.77262	3722	142.016	37357.26052	26838.63554	2150952
2014-04	242771.86130	3594	164.000	23782.30120	26272.71800	2116714
2014-05	288401.04614	4300	188.986	33953.55774	31882.58300	2585976
2014-06	401814.06310	6009	251.462	43778.60280	41894.07600	3520472
2014-07	258705.68048	3637	163.512	28035.87258	29581.73300	2189218
2014-08	456619.94236	5824	217.672	53542.89496	46759.35300	3373450
2014-09	481157.24370	6837	272.094	67979.45110	53485.43000	4064252
2014-10	422766.62916	5876	233.752	58209.83476	44622.41400	3274764
2014-11	555279.02700	7706	304.384	62856.58790	59918.35500	4324058
2014-12	503143.69348	7513	335.106	46916.52068	54853.89100	4336142

In [23]:
```
#since we only need sales we specify to sum up only sales column and reset the index values
df.groupby('month_year').sum()['sales'].reset_index()
```

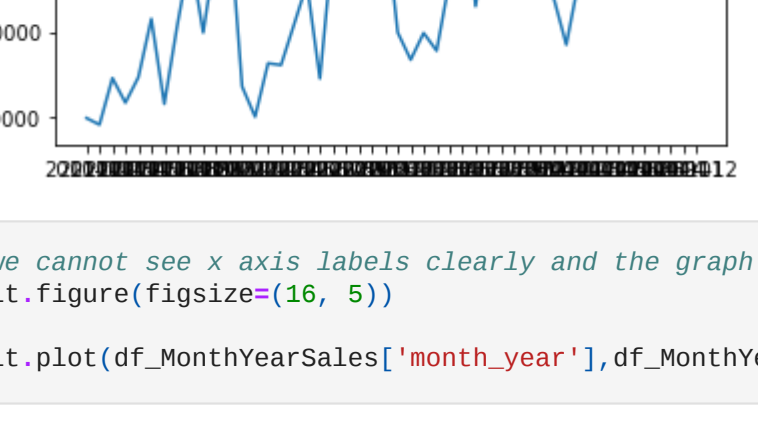
Out[23]:

	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
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 [26]:
```
#rename the new data set that we made now as df_MonthYearSales
df_MonthYearSales=df.groupby('month_year').sum()['sales'].reset_index()
```

In [27]:
```
#plotting: plt.plot(x,y)
plt.plot(df_MonthYearSales['month_year'],df_MonthYearSales['sales'])
```

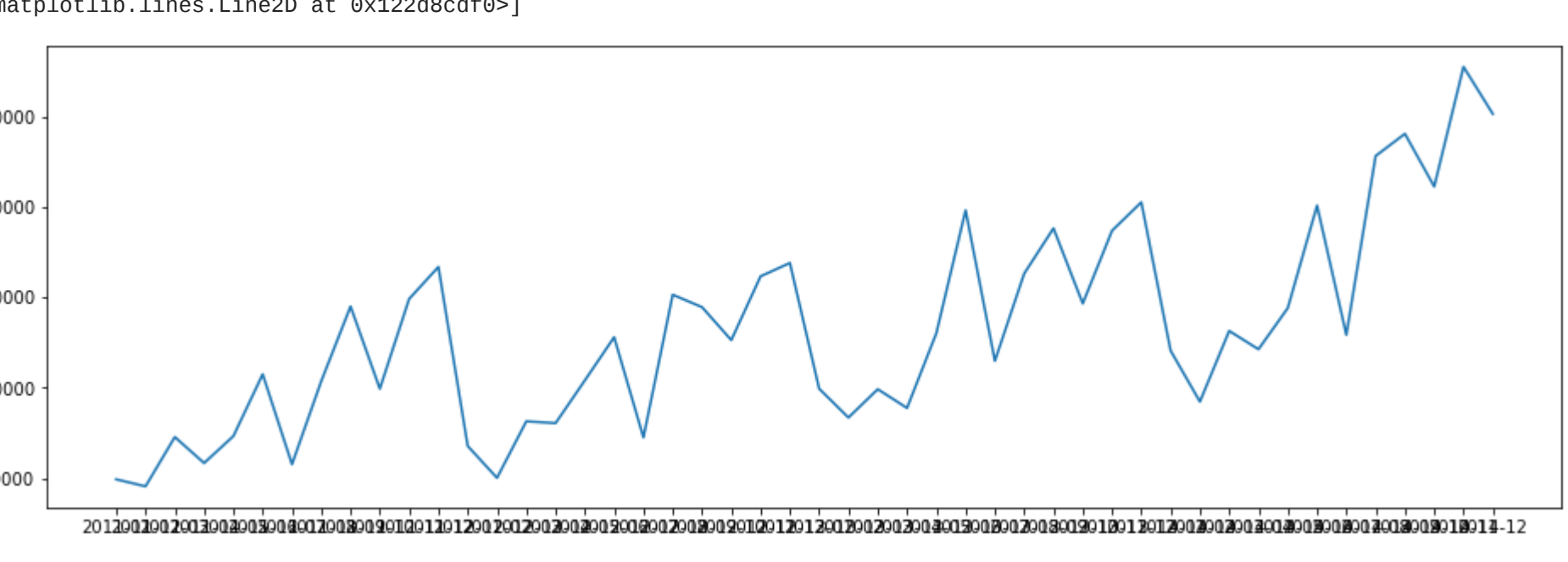
Out[27]:
```
[<matplotlib.lines.Line2D at 0x1217ed4f0>]
```



In [29]:
```
#we cannot see x axis labels clearly and the graph is aligned too close(in the above line graph)so set the figure size
plt.figure(figsize=(16, 5))

plt.plot(df_MonthYearSales['month_year'],df_MonthYearSales['sales'])
```

Out[29]:
```
[<matplotlib.lines.Line2D at 0x122d8cdf0>]
```



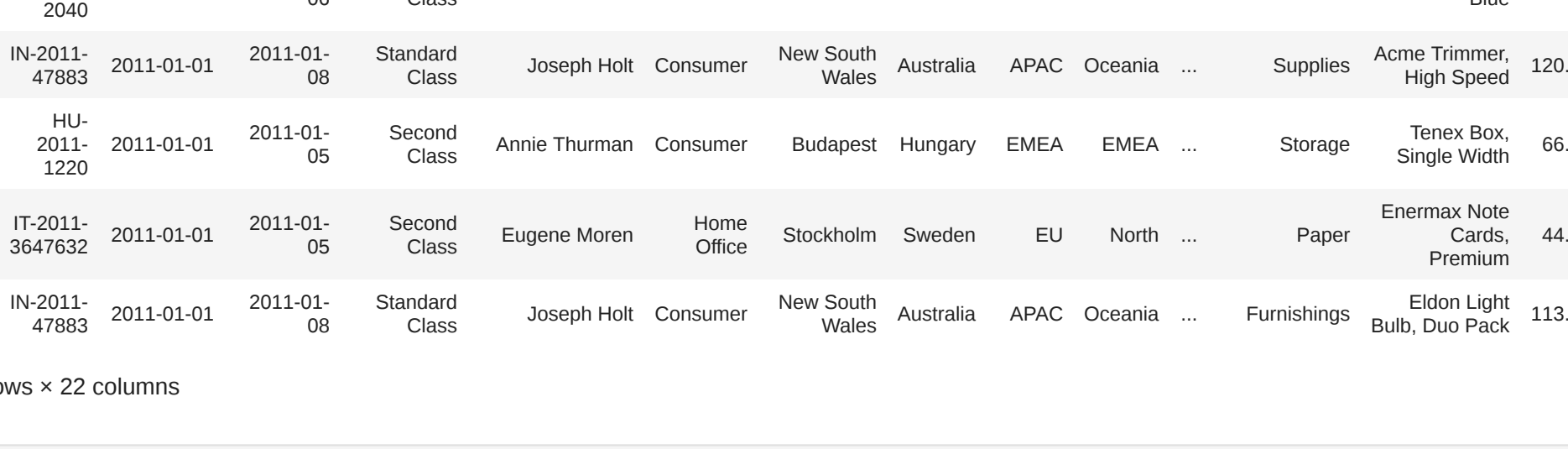
In [31]:
```
#the graph is aligned properly but since x axis is still not clear hence we tilt the labels to 90 degrees vertical by using xticks(rotation)

plt.figure(figsize=(16, 5))

plt.xticks(rotation='vertical', size=8)

plt.plot(df_MonthYearSales['month_year'],df_MonthYearSales['sales'])

plt.show()
```



2.WHICH ARE THE TOP 10 PRODUCTS BY SALES?

In [32]:
```
df.head()
```

Out[32]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	sub_category	product_name	sales	quantity	discount	profit	shippi
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Africa	Africa	...	Storage	Tenex Lockers, Blue	408.300	2	0.0	106.140	
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Supplies	Acme Trimmer, High Speed	120.366	3	0.1	36.036	
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EMEA	EMEA	...	Storage	Tenex Box, Single Width	66.120	4	0.0	29.640	
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	EU	North	...	Paper	Enermax Note Cards, Premium	44.865	3	0.5	-26.055	
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Furnishings	Eldon Light Bulb, Duo Pack	113.670	5	0.1	37.770	

5 rows × 22 columns

In [36]:
```
#the two columns we need to answer the question are "sales" and "product_name"
df['product_name']
```

Out[36]:
```
0                               Tenex Lockers, Blue
1                          Acme Trimmer, High Speed
2                           Tenex Box, Single Width
3                       Enermax Note Cards, Premium
4                        Eldon Light Bulb, Duo Pack
                            ...
51285    Cardinal Slant-D Ring Binder, Heavy Gauge Vinyl
51286           Wilson Jones Hole Reinforcements, Clear
51287            Hon Color Coded Labels, 5000 Label Set
51288            Hon Legal Exhibit Labels, Alphabetical
51289                               Bagged Rubber Bands
Name: product_name, Length: 51290, dtype: object
```

In [39]:
```
# Grouping products by sales and creating a new data frame
prod_sales = pd.DataFrame(df.groupby('product_name').sum()['sales'])
prod_sales
```

Out[39]:

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
...	...
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 [41]:
```
# Sorting the dataframe in descending order
prod_sales.sort_values(by=['sales'], ascending=False)
```

Out[41]:

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 [42]:
```
#to get top 10 products by sale (i.e top 10 rows of the above data frame)
prod_sales[:10]
```

Out[42]:

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
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

3.WHICH ARE THE MOST SELLING PRODUCTS?

In [43]:
```
df.head()
```

Out[43]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	country	market	region	...	sub_category	product_name	sales	quantity	discount	profit	shippi
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	Algeria	Africa	Africa	...	Storage	Tenex Lockers, Blue	408.300	2	0.0	106.140	
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Supplies	Acme Trimmer, High Speed	120.366	3	0.1	36.036	
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hungary	EMEA	EMEA	...	Storage	Tenex Box, Single Width	66.120	4	0.0	29.640	
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sweden	EU	North	...	Paper	Enermax Note Cards, Premium	44.865	3	0.5	-26.055	
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Australia	APAC	Oceania	...	Furnishings	Eldon Light Bulb, Duo Pack	113.670	5	0.1	37.770	

5 rows × 22 columns

In [45]:
```
#as per question we need two columns for this scenario i.e product_name and quantity columns
# Grouping products by Quantity
best_selling_prods = pd.DataFrame(df.groupby('product_name').sum()['quantity'])

# Sorting the dataframe in descending order
best_selling_prods.sort_values(by=['quantity'], ascending=False)
```

Out[45]:

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
...	...
Penpower WorldCard Pro Card Scanner	1
Bush Saratoga Collection 5-Shelf Bookcase, Hanover Cherry, *Special Order	1
Boston 1900 Electric Pencil Sharpener	1
Xerox 20	1
Global Enterprise Series Seating Low-Back Swivel/Tilt Chairs	1

3788 rows × 1 columns

In [46]:
```
# Most selling products
best_selling_prods[:10]
```

Out[46]:

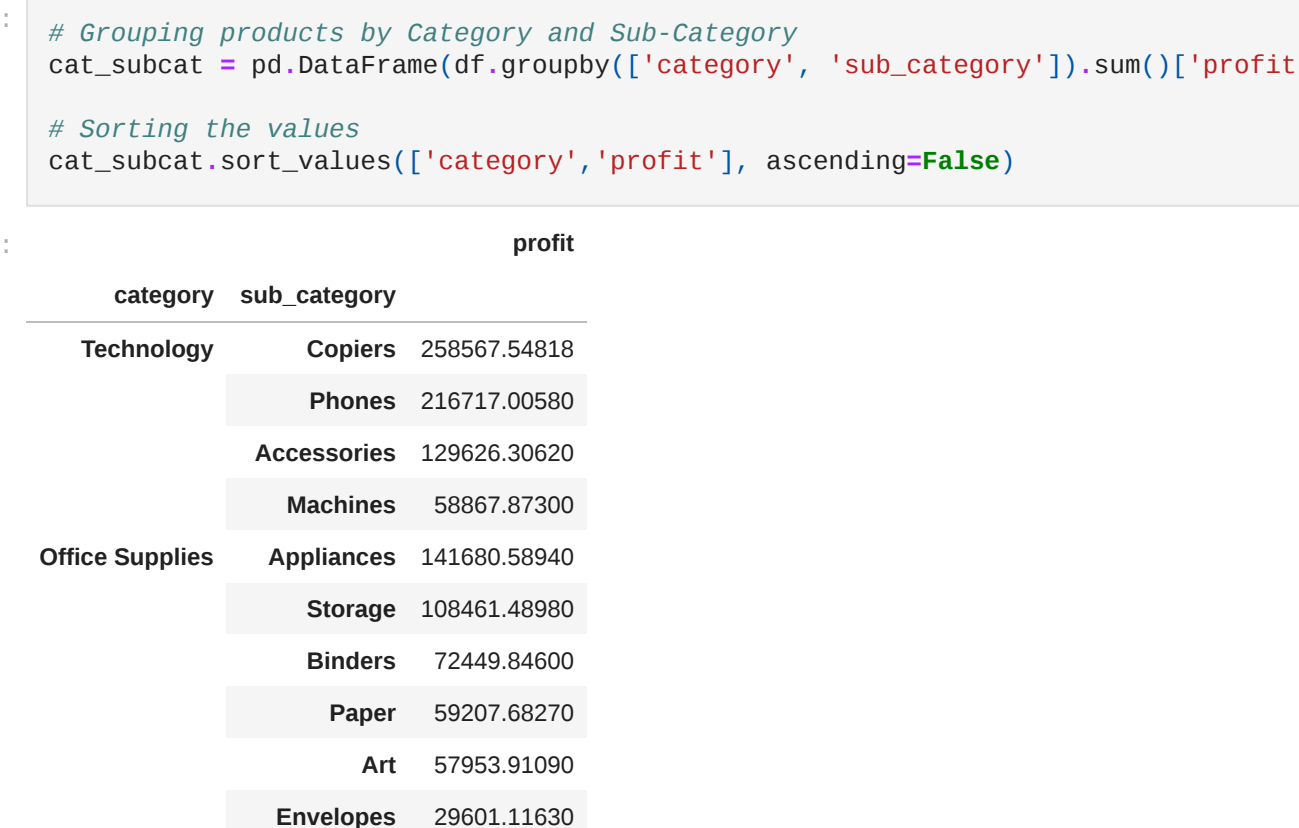
product_name	quantity
"While you Were Out" Message Book, One Form per Page	8
#10 Gummed Flap White Envelopes, 100/Box	11
#10 Self-Seal White Envelopes	10
#10 White Business Envelopes,4 1/8 x 9 1/2	32
#10- 4 1/8" x 9 1/2" Recycled Envelopes	37
#10- 4 1/8" x 9 1/2" Security-Tint Envelopes	21
#10-4 1/8" x 9 1/2" Premium Diagonal Seam Envelopes	13
#6 3/4 Gummed Flap White Envelopes	9
1.7 Cubic Foot Compact "Cube" Office Refrigerators	14
1/4 Fold Party Design Invitations & White Envelopes, 24 8-1/2" X 11" Cards, 25 Env./Pack	7

4.WHAT IS THE MOST PREFERRED SHIP MODE?

In [51]:
```
# Setting the figure size
plt.figure(figsize=(10, 8))

# countplot: Show the counts of observations in each categorical bin using bars
sns.countplot(x='ship_mode', data=df)

# Display the figure
plt.show()
```



5.Which are the Most Profitable Category and Sub-Category?

In [52]:
```
# Grouping products by Category and Sub-Category
cat_subcat = pd.DataFrame(df.groupby(['category', 'sub_category']).sum()['profit'])

# Sorting the values
cat_subcat.sort_values(['category','profit'], ascending=False)
```

Out[52]:

category	sub_category	profit
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 [ ]: