PANDAS SALES ANALYSIS

OBJECTIVE

Upon intitial inspection of the data, we can start thinking of some questions about it that we would want to answer.

- what is the overall sales trend?
- what are the top 10 products by sales?
- what are the most selling products?
- which is the most preferred ship Mode?
- which are the most profitable category and sub-category?

Improting Required libraries

In [19]:
```
# Data Manipulation
import pandas as pd

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

import seaborn as sns
```

Importing the Dataset

In [21]:
```
df = pd.read_excel('superstore_sales.xlsx')
```

Data Audit

In [26]:
```
# First five rows of the dataset
df.head()
```

Out[26]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state	co
0	AG-2011-2040	2011-01-01	2011-01-06	Standard Class	Toby Braunhardt	Consumer	Constantine	A
1	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Au
2	HU-2011-1220	2011-01-01	2011-01-05	Second Class	Annie Thurman	Consumer	Budapest	Hu
3	IT-2011-3647632	2011-01-01	2011-01-05	Second Class	Eugene Moren	Home Office	Stockholm	Sw
4	IN-2011-47883	2011-01-01	2011-01-08	Standard Class	Joseph Holt	Consumer	New South Wales	Au

5 rows × 21 columns

In [28]:
```
# last five rows of the dataset
df.tail()
```

Out[28]:

	order_id	order_date	ship_date	ship_mode	customer_name	segment	state
51285	CA-2014-115427	2014-12-31	2015-01-04	Standard Class	Erica Bern	Corporate	California
51286	MO-2014-2560	2014-12-31	2015-01-05	Standard Class	Liz Preis	Consumer	Souss-Massa-Draâ
51287	MX-2014-110527	2014-12-31	2015-01-02	Second Class	Charlotte Melton	Consumer	Managua
51288	MX-2014-114783	2014-12-31	2015-01-06	Standard Class	Tamara Dahlen	Consumer	Chihuahua
51289	CA-2014-156720	2014-12-31	2015-01-04	Standard Class	Jill Matthias	Consumer	Colorado

5 rows × 21 columns

In [36]:
```
# shape of the dataset
df.shape
```

Out[36]:
```
(51290, 21)
```

In [38]:
```
# columns present in the dataset
df.columns
```

Out[38]:
```
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 [42]:
```
# 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 [44]:
```
# checking missing values
df.isnull().sum()
```

Out[44]:
```
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 [48]:
```
# Getting descriptive statistics summary
df.describe()
```

Out[48]:

	order_date	ship_date	sales	quantity	discount	
count	51290	51290	51290.000000	51290.000000	51290.000000	51
mean	2013-05-11 21:26:49.155780864	2013-05-15 20:42:42.745174528	246.490581	3.476545	0.142908	
min	2011-01-01 00:00:00	2011-01-03 00:00:00	0.444000	1.000000	0.000000	-6
25%	2012-06-19 00:00:00	2012-06-23 00:00:00	30.758625	2.000000	0.000000	
50%	2013-07-08 00:00:00	2013-07-12 00:00:00	85.053000	3.000000	0.000000	
75%	2014-05-22 00:00:00	2014-05-26 00:00:00	251.053200	5.000000	0.200000	
max	2014-12-31 00:00:00	2015-01-07 00:00:00	22638.480000	14.000000	0.850000	8
std	NaN	NaN	487.565361	2.278766	0.212280	

EXPLORATORY DATA ANALYSIS

WHAT IS THE OVERALL SALES TREND?

In [56]:
```
print(df['order_date'].min())
df['order_date'].max()
```

```
2011-01-01 00:00:00
```

Out[56]: Timestamp('2014-12-31 00:00:00')

In [108…
```
# Extract 'month_year' from 'order_date'
df['month_year'] = df['order_date'].apply(lambda x: x.strftime('%y-%m'))
```

In [110…
```
# Group by 'month_year' and sum 'sales'
df_trend = df.groupby('month_year')['sales'].sum().reset_index()
```

In [117…
```
# Setting the figure size
plt.figure(figsize=(15, 6))
plt.plot(df_trend['month_year'], df_trend['sales'], color='#b80045')
plt.xticks(rotation='vertical', size=8)
plt.show()
```

WHICH ARE THE TOP 10 PRODUCTS BY SALES?

In [123…
```
# Grouping product name column
prod_sales = pd.DataFrame(df.groupby('product_name')['sales'].sum() )
```

In [127…
```
# sorting prod_sales column
prod_sales = prod_sales.sort_values('sales', ascending = False)
```

In [129…
```
# Top 10 products by sales
prod_sales[:10]
```

Out[129…

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

WHICH ARE THE MOST SELLING PRODUCTS

In [146…
```
# Grouping product name
most_sell_prod = pd.DataFrame(df.groupby('product_name')['quantity'].sum())
```

In [148…
```
# sorting most_sell_products
most_sell_prod = most_sell_prod.sort_values('quantity', ascending = False)
```

In [150…
```
most_sell_prod[:10]
```

Out[150…

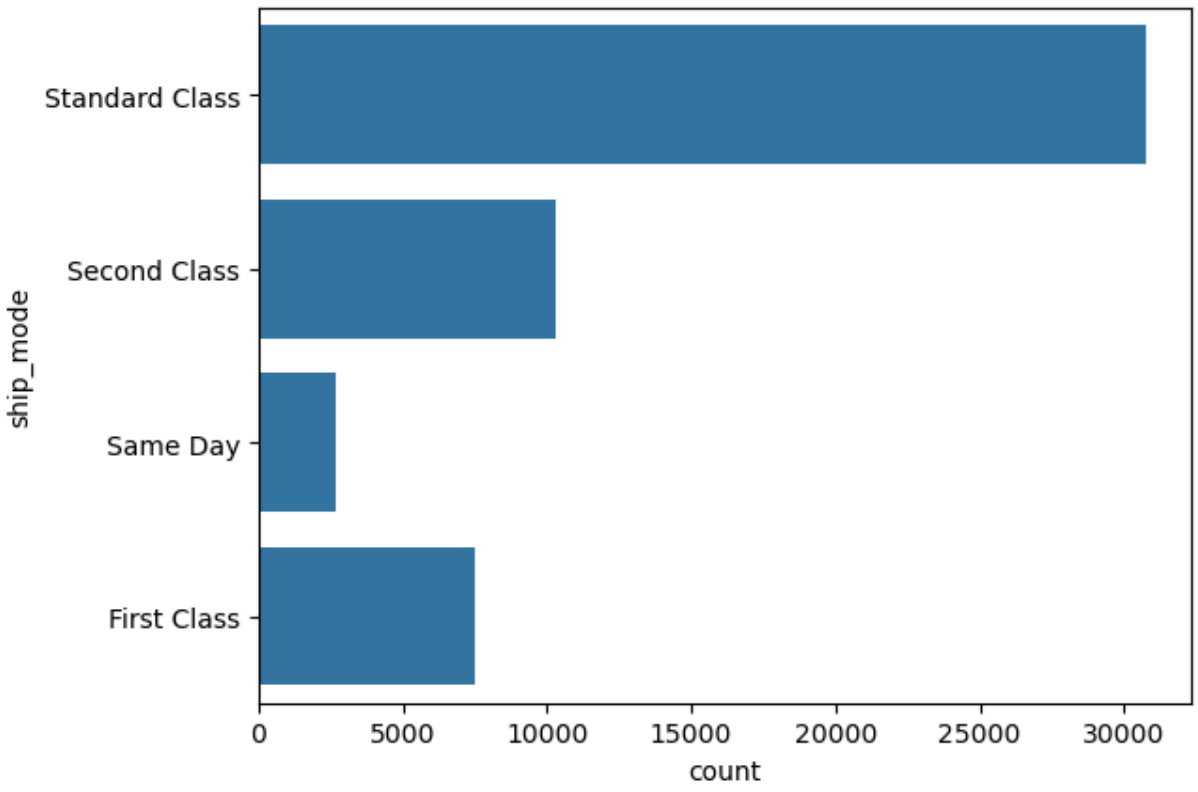
product_name	quantity
Staples	876
Cardinal Index Tab, Clear	337
Eldon File Cart, Single Width	321
Rogers File Cart, Single Width	262
Sanford Pencil Sharpener, Water Color	259
Stockwell Paper Clips, Assorted Sizes	253
Avery Index Tab, Clear	252
Ibico Index Tab, Clear	251
Smead File Cart, Single Width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

WHAT IS THE MOST PREFERRED SHIP MODE?

In [167…

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

plt.show()
```



WHICH ARE THE MOST PROFITABE CATEGORY AND SUB-CATEGORY

In [208…

```
# Grouping category and sub_category
cat_subcat_profit = pd.DataFrame(df.groupby(['category', 'sub_category'])['profit']

cat_subcat_profit.sort_values(['category', 'profit'], ascending=False)
```

Out[208…

		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 [ ]: