SUPERSTORE SALES TREND DATA ANALYSIS

Business Objective - Analyze sales data to identify trends, seasonal patterns, and anomalies.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df = pd.read_csv('/content/Superstore.csv.csv', encoding='cp1252')
df.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category
0	1	CA-2016-152156	11-08-2016	11-11-2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture
1	2	CA-2016-152156	11-08-2016	11-11-2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture
2	3	CA-2016-138688	06-12-2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies
3	4	US-2015-108966	10-11-2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	FUR-TA-10000577	Furniture
4	5	US-2015-108966	10-11-2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	OFF-ST-10000760	Office Supplies

5 rows × 21 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   object
 3   Ship Date      9994 non-null   object
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

```
df.describe()
```

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55190.379428	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

```
df.isnull().sum()
```

	0
Row ID	0
Order ID	0
Order Date	0
Ship Date	0
Ship Mode	0
Customer ID	0
Customer Name	0
Segment	0
Country	0
City	0
State	0
Postal Code	0
Region	0
Product ID	0
Category	0
Sub-Category	0
Product Name	0
Sales	0
Quantity	0
Discount	0
Profit	0

dtype: int64

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df1 = df.copy()
```

```
df1['Order Date'] = pd.to_datetime(df1['Order Date'], format='mixed')
df1['Ship Date'] = pd.to_datetime(df1['Ship Date'], format='mixed')
```

Changing the datatype i.e., 'Order Date', 'Ship Date' from object to datetime variables.

```
df1.columns
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State',
       'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category',
       'Product Name', 'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: datetime64[ns](2), float64(3), int64(3), object(13)
memory usage: 1.6+ MB
```

```
df1.describe()
```

	Row ID	Order Date	Ship Date	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994	9994	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	2016-04-30 00:07:12.259355648	2016-05-03 23:06:58.571142912	55190.379428	229.858001	3.789574	0.156203	28.656896
min	1.000000	2014-01-03 00:00:00	2014-01-07 00:00:00	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	2015-05-23 00:00:00	2015-05-27 00:00:00	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	2016-06-26 00:00:00	2016-06-29 00:00:00	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	2017-05-14 00:00:00	2017-05-18 00:00:00	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	2017-12-30 00:00:00	2018-01-05 00:00:00	99301.000000	22638.480000	14.000000	0.800000	8399.976000
std	2885.163629	NaN	NaN	32063.693350	623.245101	2.225110	0.206452	234.260108

```
df2 = df1.copy()

# Assessing Shipping Duration by using logistics KPI
df2['Shipping Duration'] = (df2['Ship Date'] - df2['Order Date']).dt.days

# Profit Margin
df2['Profit Margin'] = df2['Profit'] / df2['Sales']
df2['Profit Margin'] = df2['Profit Margin'].replace([float('inf'), -float('inf')], 0)


# Sales per Item
df2['Sales per Item'] = df2['Sales'] / df2['Quantity']


# Year and Month Columns following the trend
df2['Order Date'] = pd.to_datetime(df2['Order Date'], format='mixed', dayfirst=False)
df2['Order Year'] = df2['Order Date'].dt.year
df2['Order Month'] = df2['Order Date'].dt.month
print(df2.columns, df2.head())
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State',
       'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category',
       'Product Name', 'Sales', 'Quantity', 'Discount', 'Profit',
       'Shipping Duration', 'Profit Margin', 'Sales per Item', 'Order Year',
       'Order Month'],
      dtype='object')    Row ID        Order ID Order Date  Ship Date       Ship Mode Customer ID  \
0       1  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
1       2  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
2       3  CA-2016-138688 2016-06-12 2016-06-16    Second Class    DV-13045
3       4  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335
4       5  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335

     Customer Name    Segment        Country             City  ...  \
0      Claire Gute   Consumer  United States        Henderson  ...
1      Claire Gute   Consumer  United States        Henderson  ...
2  Darrin Van Huff  Corporate  United States      Los Angeles  ...
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...
```

```
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...

                                        Product Name     Sales Quantity  \
0                 Bush Somerset Collection Bookcase  261.9600        2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400        3
2  Self-Adhesive Address Labels for Typewriters b...   14.6200        2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775        5
4                     Eldon Fold 'N Roll Cart System   22.3680        2

  Discount    Profit Shipping Duration Profit Margin  Sales per Item  \
0     0.00   41.9136                 3        0.1600        130.9800
1     0.00  219.5820                 3        0.3000        243.9800
2     0.00    6.8714                 4        0.4700          7.3100
3     0.45 -383.0310                 7       -0.4000        191.5155
4     0.20    2.5164                 7        0.1125         11.1840

   Order Year  Order Month
0        2016           11
1        2016           11
2        2016            6
3        2015           10
4        2015           10

[5 rows x 26 columns]
```

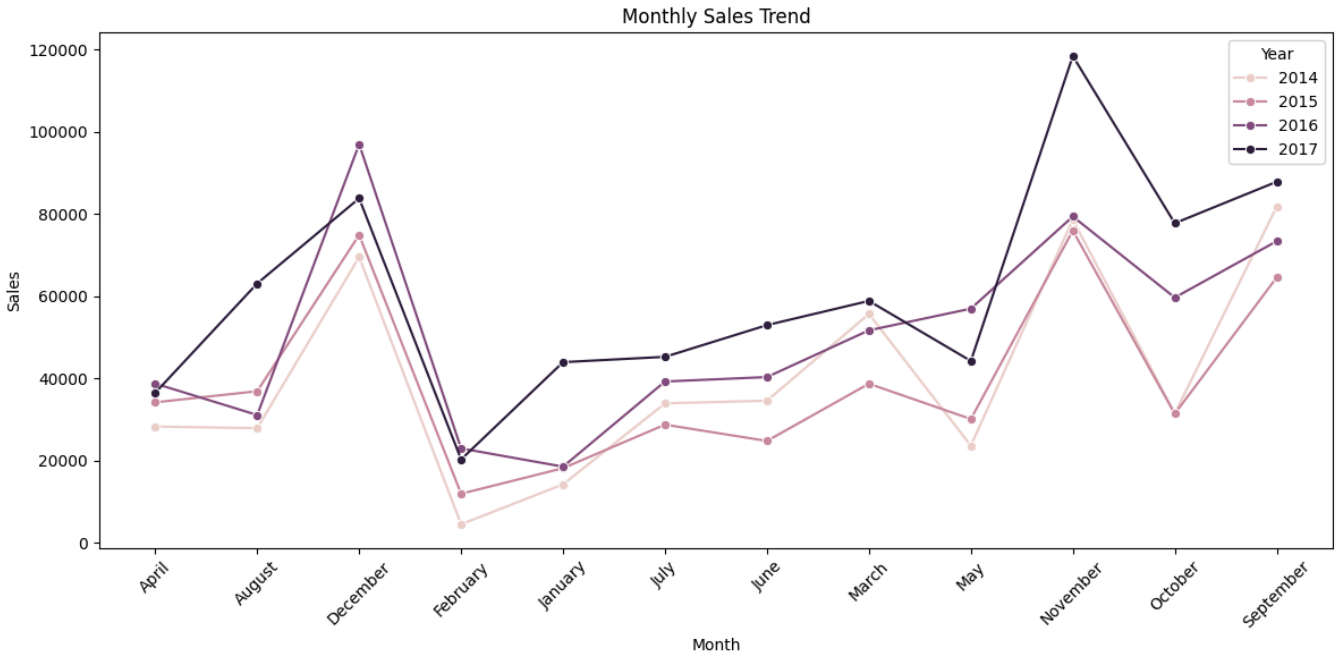
```
df2.to_csv("cleaned_superstore.csv", index=False)
```

```
# Extract month and year
df2['Month'] = df2['Order Date'].dt.month_name()
df2['Year'] = df2['Order Date'].dt.year
```

```
total_sales = df['Sales'].sum()
total_profit = df['Profit'].sum()
total_quantity = df['Quantity'].sum()

print(f"Total Sales: ${total_sales:,.2f}")
print(f"Total Profit: ${total_profit:,.2f}")
print(f"Total Quantity: {total_quantity}")
```

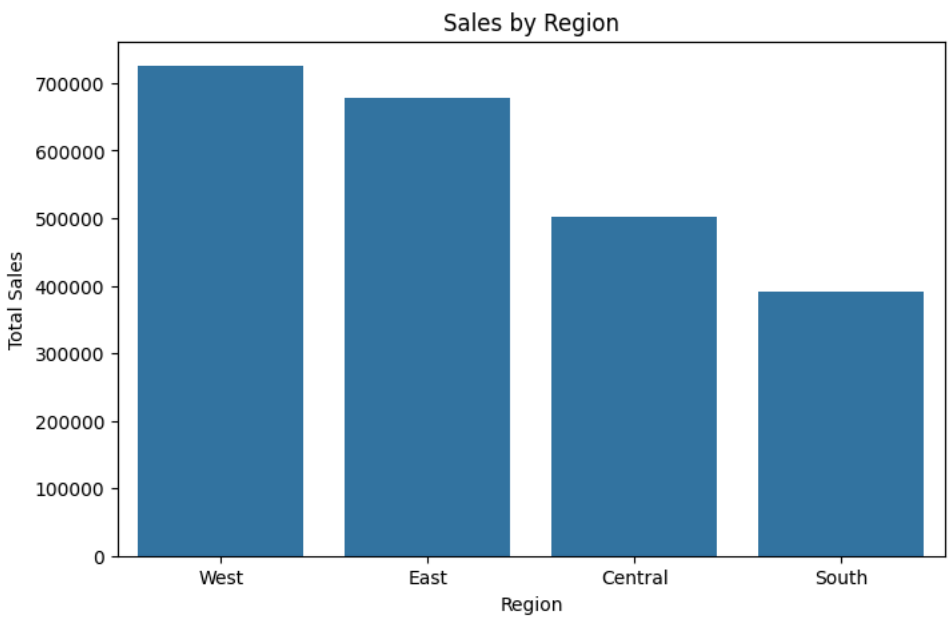
```
Total Sales: $2,297,200.86
Total Profit: $286,397.02
Total Quantity: 37873
```

```
monthly_sales = df2.groupby(['Year', 'Month'])['Sales'].sum().reset_index()

plt.figure(figsize=(12,6))
sns.lineplot(data=monthly_sales, x='Month', y='Sales', hue='Year', marker='o')
plt.title('Monthly Sales Trend')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



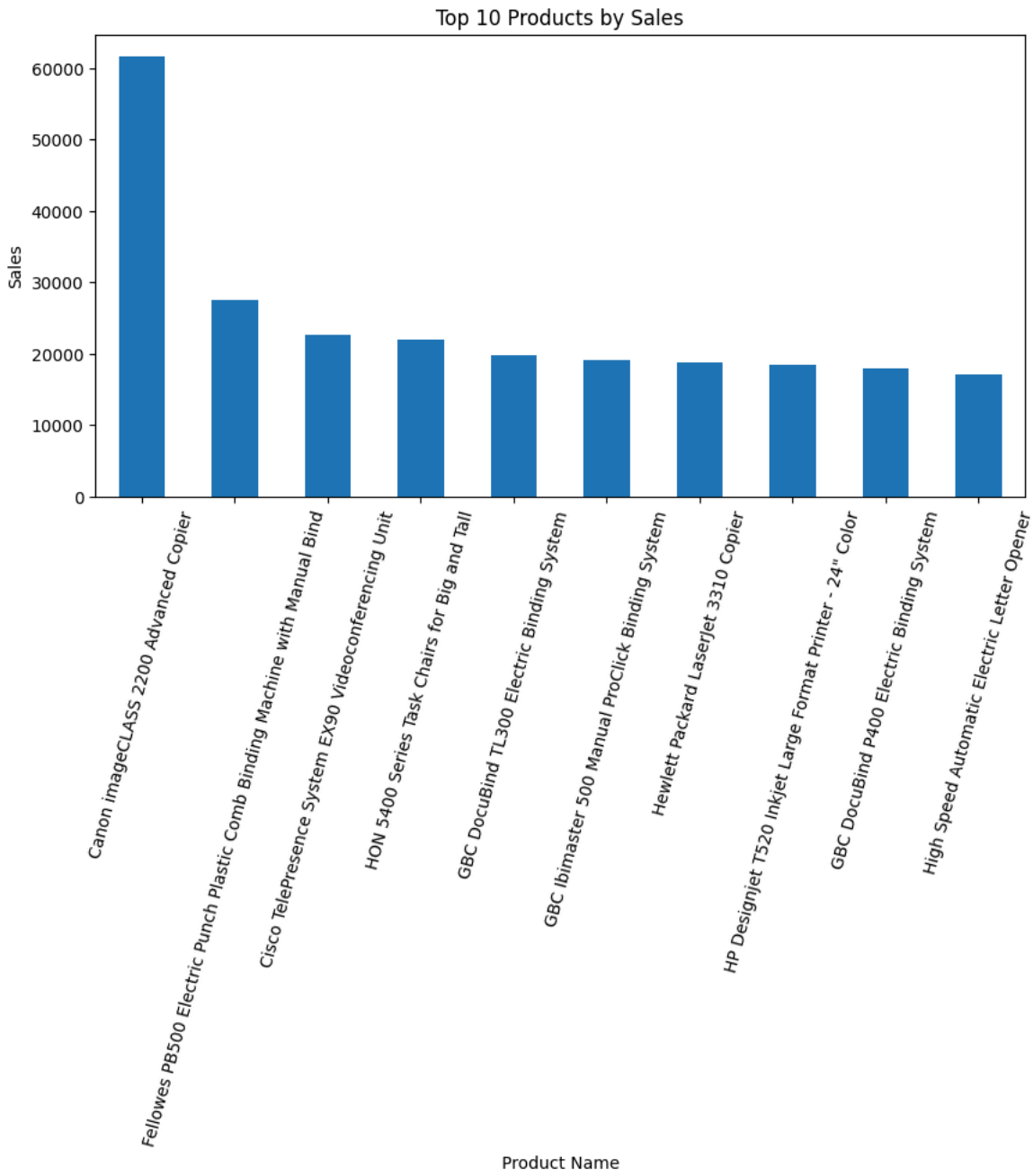
From the above plot, we can say that in the month of August, October, November, December; there are uplift movements of line chart, showing that monthly sales trend is increased whereas in the month of February, MAy, September; there is a sharp fall. So, we can said that the seasonal trend aligns on last 3 months before yearending. It can also be concluded that the highest value showing the trend of monthly sales, lies on 2017.

```
region_sales = df.groupby('Region')['Sales'].sum().sort_values(ascending=False)

plt.figure(figsize=(8,5))
sns.barplot(x=region_sales.index, y=region_sales.values)
plt.title('Sales by Region')
plt.ylabel('Total Sales')
plt.xlabel('Region')
plt.show()
```

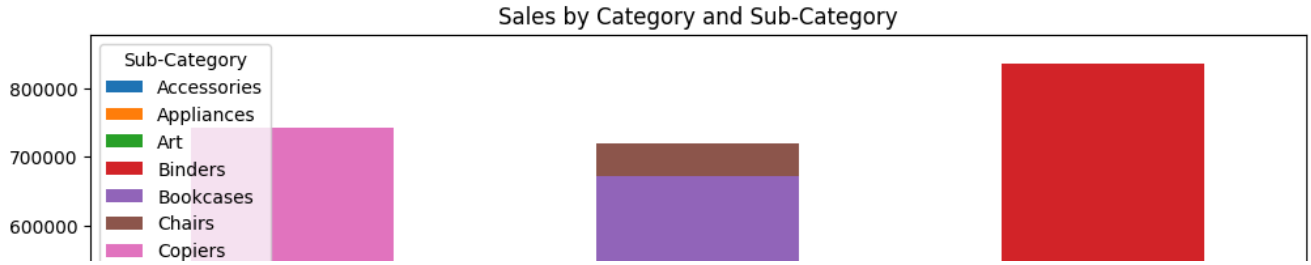



```
top_products = df.groupby('Product Name')['Sales'].sum().sort_values(ascending=False).head(10)

plt.figure(figsize=(10,5))
top_products.plot(kind='bar')
plt.title('Top 10 Products by Sales')
plt.ylabel('Sales')
```

```
plt.xticks(rotation=75)
plt.show()
```




```
# Sales by Category and Sub-Category
df.groupby(['Category', 'Sub-Category'])['Sales'].sum().unstack().plot(kind='bar', stacked=True, figsize=(12,6))
plt.title('Sales by Category and Sub-Category')
plt.ylabel('Sales')
plt.show()
```


Sales by Category and Sub-Category
Sub-Category
Accessories
Appliances
Art
Binders
Bookcases
Chairs
Copiers
800000
700000
600000