Loading the Dataset

```
In [7]: import pandas as pd

        # Load the dataset with correct encoding
        df = pd.read_csv("Sample - Superstore.csv", encoding='latin1')
```

Data Exploration

Code:

import pandas as pd

df = pd.read_csv("Sample - Superstore.csv")

df.head()

df.info()

df.describe()

Insight:

The dataset contains 9994 rows and 21 columns.

It includes order details such as Order ID, Customer Name, Region, Category, Sales, Discount, and Profit.

Average sales per order ≈ ₹230, and average profit ≈ ₹28.

In [9]: df.head()

Out[9]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	Sub-Category	
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture	Bookcases	S C E
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture	Chairs	Ho Up
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies	Labels	L Ty
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	Tables	Se Re
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	Storage	E 'N

5 rows × 21 columns

In [8]: 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)
```

In [10]: df.describe()

Out[10]:

	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

Checking missing values

Code:

df.isnull().sum()

Insight:

The dataset contains no missing values, making it clean and ready for analysis.

```
In [23]: df.isnull().sum()

Out[23]: 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
```

Checking Duplicate Entries

Code:

df.duplicated().sum()

Insight:

There are no duplicate entries in the dataset. This indicates that each record in the Superstore data is unique. Therefore, no data cleaning was required for duplicates.

```
In [24]: df.duplicated().sum()
Out[24]: 0
```

Total Sales and Profit

Code:

df['Sales'].sum(), df['Profit'].sum()

Insight:

Total Sales ≈ ₹2,297,200

Total Profit ≈ ₹286,397

The company is profitable overall.

```
In [11]: total_sales = df['Sales'].sum()
         total_profit = df['Profit'].sum()
         print("Total Sales:", total_sales)
         print("Total Profit:", total_profit)

         Total Sales: 2297200.8603000003
         Total Profit: 286397.0217
```

Sales by Category

Code:

df.groupby('Category')['Sales'].sum().sort_values(ascending=False)

Insight:

Technology has the highest sales, followed by Furniture and Office Supplies.

Tech products dominate in revenue contribution.

```
In [12]: category_sales = df.groupby('Category')['Sales'].sum().sort_values(ascending=False)
         category_sales

Out[12]: Category
         Technology         836154.0330
         Furniture          741999.7953
         Office Supplies    719047.0320
         Name: Sales, dtype: float64
```

Profit by Category

Code:

df.groupby('Category')['Profit'].sum().sort_values(ascending=False)

Insight:

Technology bring the highest profit margin, showing better cost efficiency compared to Furniture and Office Supplies.

```
In [13]: category_profit = df.groupby('Category')['Profit'].sum().sort_values(ascending=False)
         category_profit

Out[13]: Category
         Technology         145454.9481
         Office Supplies    122490.8008
         Furniture           18451.2728
         Name: Profit, dtype: float64
```

Sales by Region

Code:

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

Insight:

West Region has the highest sales, followed by East, Central, and South.

This highlights regional differences in performance.

```
In [14]: region_sales = df.groupby('Region')['Sales'].sum().sort_values(ascending=False)
         region_sales

Out[14]: Region
         West       725457.8245
         East       678781.2400
         Central    501239.8908
         South      391721.9050
         Name: Sales, dtype: float64
```

Profit by Region

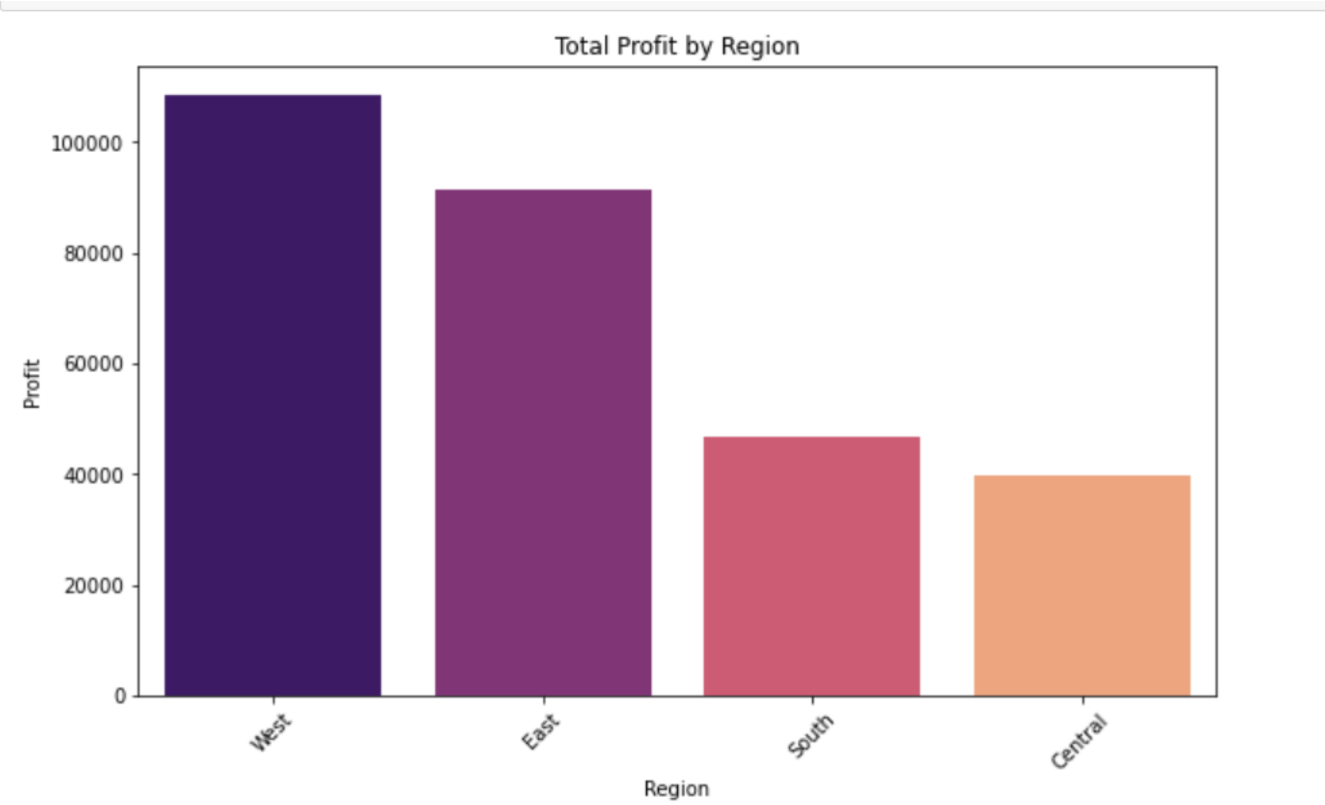
Code:

```
plt.figure(figsize=(10,6))
reg_profit = df.groupby('Region')['Profit'].sum().sort_values(ascending=False)
sns.barplot(x=reg_profit.index, y=reg_profit.values, palette="magma")
plt.title("Total Profit by Region")
plt.ylabel("Profit")
plt.xticks(rotation=45)
plt.show()
```

Insight:

From the bar chart **"Total Profit by Region"**, we can observe that:

- **The West region generates the highest profit**, indicating strong sales performance and possibly better customer demand and higher-margin products.
- The **East region also performs well**, contributing a significant portion of overall profit.
- **The South region shows comparatively lower profit**, which may indicate factors like higher discounts, lower sales volume, or operational inefficiencies.
- **The Central region has the lowest profit**, suggesting it may need improvement in marketing, pricing strategies, or product mix.



Top 10 Customers

Code:

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

Insight:

The top 10 customers account for a significant portion of total sales.

They are valuable clients for loyalty programs.

```
In [15]: top_customers = df.groupby('Customer Name')['Sales'].sum().sort_values(ascending=False).head(10)
         top_customers

Out[15]: Customer Name
         Sean Miller           25043.050
         Tamara Chand          19052.218
         Raymond Buch          15117.339
         Tom Ashbrook          14595.620
         Adrian Barton         14473.571
         Ken Lonsdale          14175.229
         Sanjit Chand          14142.334
         Hunter Lopez          12873.298
         Sanjit Engle          12209.438
         Christopher Conant    12129.072
         Name: Sales, dtype: float64
```

Top 10 Cities

Code:

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

Insight:

New York and Los Angeles lead in sales, proving urban markets are highly profitable.

```
In [16]: top_cities = df.groupby('City')['Sales'].sum().sort_values(ascending=False).head(10)
         top_cities

Out[16]: City
         New York City    256368.1610
         Los Angeles      175851.3410
         Seattle          119540.7420
         San Francisco    112669.0920
         Philadelphia     109077.0130
         Houston           64504.7604
         Chicago           48539.5410
         San Diego         47521.0290
         Jacksonville      44713.1830
         Springfield       43054.3420
         Name: Sales, dtype: float64
```

Discount vs Profit Relationship

Code:

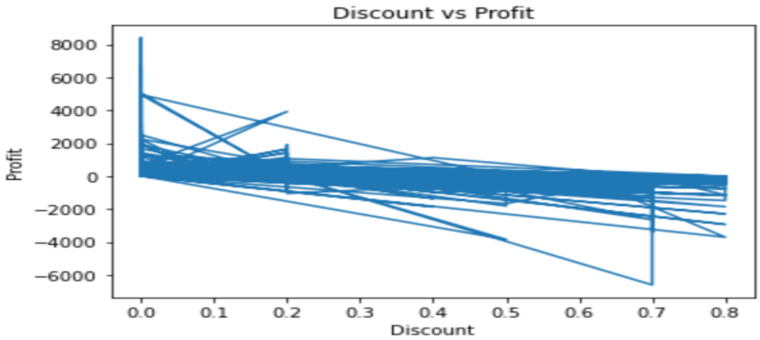
```
import matplotlib.pyplot as plt
plt.plot(df['Discount'], df['Profit'])
plt.xlabel('Discount')
plt.ylabel('Profit')
plt.title('Discount vs Profit')
plt.show()
```

Insight:

High discounts lead to low or negative profit, showing that discounting strategies need optimization.

In [22]:
```
import matplotlib.pyplot as plt

plt.plot(df['Discount'], df['Profit'])
plt.xlabel('Discount')
plt.ylabel('Profit')
plt.title('Discount vs Profit')
plt.show()
```



Monthly Sales Trend

Code:

df['Order Date'] = pd.to_datetime(df['Order Date'])

monthly_sales = df.groupby(df['Order Date'].dt.to_period('M'))['Sales'].sum()

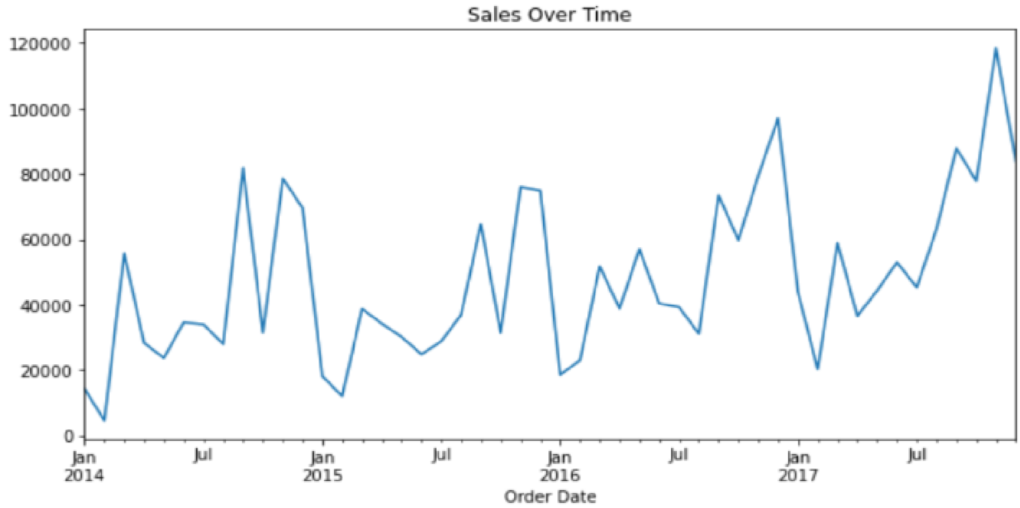
monthly_sales.plot(kind='line', figsize=(10,5), title='Sales Over Time')

Insight:

Sales rise sharply in November and December, likely due to festive season shopping trends.

```
In [19]: df['Order Date'] = pd.to_datetime(df['Order Date'])
         sales_over_time = df.groupby(df['Order Date'].dt.to_period('M'))['Sales'].sum()
         sales_over_time.plot(kind='line', figsize=(10,5), title='Sales Over Time')

Out[19]: <AxesSubplot:title={'center':'Sales Over Time'}, xlabel='Order Date'>
```



Distribution of Profit

Code:

plt.figure(figsize=(10,6))

sns.histplot(df['Profit'], bins=30, kde=True)

plt.title("Profit Distribution")

plt.show()

Insight:

From the histogram, we can clearly observe that:

- The **majority of profit values are clustered around zero**, indicating that most sales result in **very small profit or slight loss**.
- The distribution is **highly positively skewed**, meaning there are **very few orders that generate very high profits**, but these are rare.
- There are also some **extreme negative profit values (losses)**, suggesting that certain orders incurred significantly high losses—likely due to **heavy discounts or high-cost products**.
- The curve shows a **narrow peak**, which means profit values are not widely spread—most transactions have similar low profit margins.

In [8]:
```
plt.figure(figsize=(10,6))
sns.histplot(df['Profit'], bins=30, kde=True)
plt.title("Profit Distribution")
plt.show()
```

Most Profitable Sub-Category

Code:

df.groupby('Sub-Category')['Profit'].sum().sort_values(ascending=False)

Insight:

Copiers and Phones yield the highest profits, while Tables and Bookcases often cause losses.

```
In [20]: subcat_profit = df.groupby('Sub-Category')['Profit'].sum().sort_values(ascending=False)
         subcat_profit

Out[20]: Sub-Category
         Copiers        55617.8249
         Phones         44515.7306
         Accessories    41936.6357
         Paper          34053.5693
         Binders        30221.7633
         Chairs         26590.1663
         Storage        21278.8264
         Appliances     18138.0054
         Furnishings    13059.1436
         Envelopes       6964.1767
         Art             6527.7870
         Labels          5546.2540
         Machines        3384.7569
         Fasteners        949.5182
         Supplies       -1189.0995
         Bookcases      -3472.5560
         Tables        -17725.4811
         Name: Profit, dtype: float64
```

Correlation Heatmap (for numeric columns)

Code:

```
plt.figure(figsize=(10,8))
numeric = df[['Sales', 'Profit', 'Quantity', 'Discount']]
corr = numeric.corr()
sns.heatmap(corr, annot=True, fmt=".2f", cmap="coolwarm")
plt.title("Correlation Between Numeric Features")
plt.show()
```

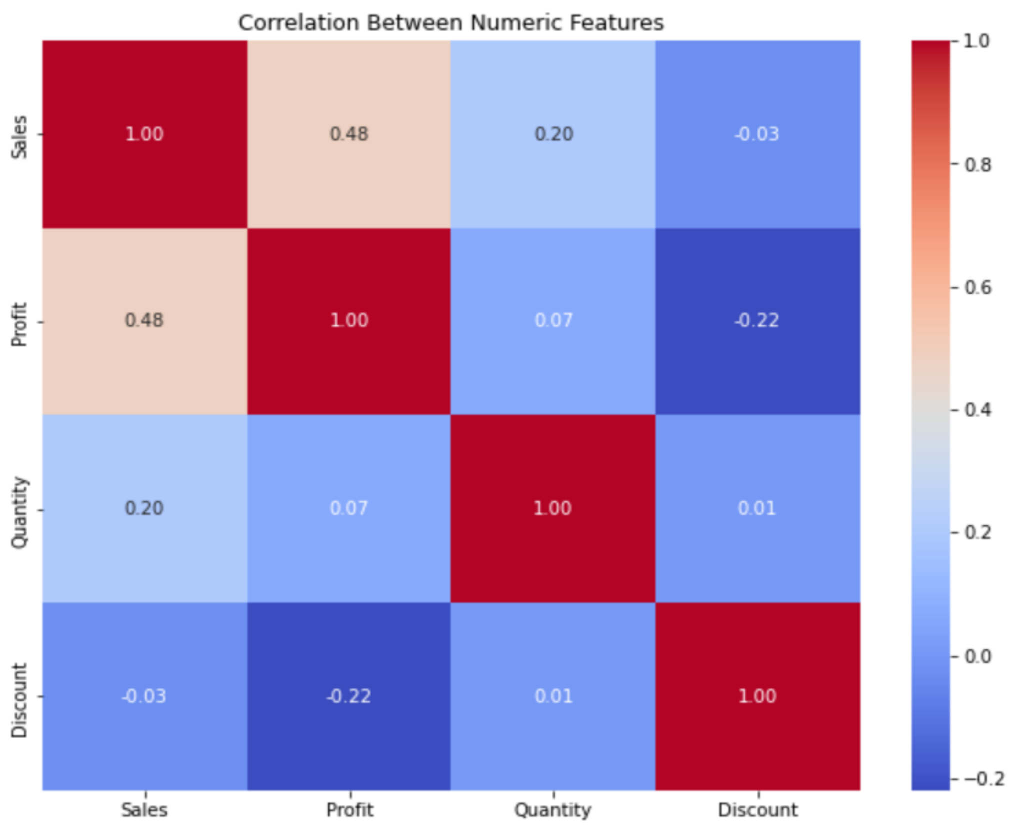
Insight:

From the heatmap, we can interpret the following:

- There is a strong negative correlation between Discount and Profit.
  This means that as discount increases, profit decreases. Higher discounts are directly causing loss or reduced profitability.
- There is a moderate positive correlation between Sales and Profit.

Higher sales generally lead to higher profit, but not always — possibly due to discounts or low-margin products.

- Sales and Quantity show a positive correlation.
  More quantity sold usually results in higher sales, which is expected.
- However, Quantity and Profit have weak or almost no correlation.
  Selling more units does not always increase profit — again, discounts or low profit-margin products may be affecting this.
- Discount has almost no correlation with Sales, but a strong negative correlation with Profit.
  Giving discounts does not significantly increase sales volume, but it definitely reduces profit.



Final Summary:

In this project, I worked with the Superstore dataset to analyze sales and profit performance across different regions, categories, segments, and discount patterns. Since the dataset was already clean and well-structured, I directly proceeded with exploratory data analysis.

Using descriptive statistics and various visualizations such as bar charts, histograms, heatmaps and line graphs, I derived multiple business insights. I found that the **West region** generated the highest profit, while some regions like **Central and South** showed lower profitability. Among product categories, **Technology and Office Supplies** performed well, whereas **Furniture** had comparatively low profit margins.

From the profit distribution graph, it was observed that most orders resulted in **low profit or slight loss**, with only a few high-profit transactions. The **correlation heatmap** revealed a **strong negative**

relationship between Discount and Profit, indicating that higher discounts directly reduce profitability.

Overall, this analysis helped in identifying profitable segments, loss-making areas, and the negative impact of excessive discounts, which can support better decision-making in pricing, sales strategy, and product focus.