```
!pip install pandas

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

```
Requirement already satisfied: pandas in c:\users\guru jadhav\anaconda3\lib\site-packages (2.1.4)
Requirement already satisfied: numpy<2,>=1.23.2 in c:\users\guru jadhav\anaconda3\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\guru jadhav\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\guru jadhav\anaconda3\lib\site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\guru jadhav\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\guru jadhav\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.
```

```
df = pd.read_csv(r"C:/Users/GURU JADHAV/Downloads/Diwali Sales Data.csv", encoding='latin1')
print(df.shape)
```

```
(11251, 15)
```

```
df.head(10)
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Am
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	239
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	239
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	239
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	239
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	238
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh	Northern	Food Processing	Auto	1	238
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh	Central	Lawyer	Auto	4	238
7	1002092	Shivangi	P00273442	F	55+	61	0	Maharashtra	Western	IT Sector	Auto	1	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11251 non-null  int64
 1   Cust_name         11251 non-null  object
 2   Product_ID        11251 non-null  object
 3   Gender            11251 non-null  object
 4   Age Group         11251 non-null  object
 5   Age               11251 non-null  int64
 6   Marital_Status    11251 non-null  int64
 7   State             11251 non-null  object
 8   Zone              11251 non-null  object
 9   Occupation        11251 non-null  object
 10  Product_Category  11251 non-null  object
 11  Orders            11251 non-null  int64
 12  Amount            11239 non-null  float64
 13  Status            0 non-null      float64
 14  unnamed1          0 non-null      float64
dtypes: float64(3), int64(4), object(8)
memory usage: 1.3+ MB
```

```
df.drop(['unnamed1'], axis=1, inplace=True, errors='ignore')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
```

```
 0   User_ID           11251 non-null  int64
 1   Cust_name         11251 non-null  object
 2   Product_ID        11251 non-null  object
 3   Gender            11251 non-null  object
 4   Age Group         11251 non-null  object
 5   Age               11251 non-null  int64
 6   Marital_Status    11251 non-null  int64
 7   State             11251 non-null  object
 8   Zone              11251 non-null  object
 9   Occupation        11251 non-null  object
 10  Product_Category  11251 non-null  object
 11  Orders            11251 non-null  int64
 12  Amount            11239 non-null  float64
 13  Status            0 non-null      float64
dtypes: float64(2), int64(4), object(8)
memory usage: 1.2+ MB
```

```
df.drop(["Status"], axis=1, inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11251 non-null  int64
 1   Cust_name         11251 non-null  object
 2   Product_ID        11251 non-null  object
 3   Gender            11251 non-null  object
 4   Age Group         11251 non-null  object
 5   Age               11251 non-null  int64
 6   Marital_Status    11251 non-null  int64
 7   State             11251 non-null  object
 8   Zone              11251 non-null  object
 9   Occupation        11251 non-null  object
 10  Product_Category  11251 non-null  object
 11  Orders            11251 non-null  int64
 12  Amount            11239 non-null  float64
dtypes: float64(1), int64(4), object(8)
memory usage: 1.1+ MB
```

```
pd.isnull(df)
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
11246	False	False	False	False	False	False	False	False	False	False	False	False	False
11247	False	False	False	False	False	False	False	False	False	False	False	False	False
11248	False	False	False	False	False	False	False	False	False	False	False	False	False
11249	False	False	False	False	False	False	False	False	False	False	False	False	False
11250	False	False	False	False	False	False	False	False	False	False	False	False	False

11251 rows × 13 columns

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

```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age Group           0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
```

```
Product_Category     0
Orders               0
Amount              12
dtype: int64
```

```
df.dropna(inplace=True)
```

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

```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age Group           0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
Product_Category    0
Orders              0
Amount              0
dtype: int64
```

```
df['Amount']=df['Amount'].astype('int')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11239 entries, 0 to 11250
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   User_ID           11239 non-null  int64 
 1   Cust_name         11239 non-null  object
 2   Product_ID        11239 non-null  object
 3   Gender            11239 non-null  object
 4   Age Group         11239 non-null  object
 5   Age               11239 non-null  int64 
 6   Marital_Status    11239 non-null  int64 
 7   State             11239 non-null  object
 8   Zone              11239 non-null  object
 9   Occupation        11239 non-null  object
 10  Product_Category  11239 non-null  object
 11  Orders            11239 non-null  int64 
 12  Amount            11239 non-null  int32 
dtypes: int32(1), int64(4), object(8)
memory usage: 1.2+ MB
```

```
df.columns
```

```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

```
df.describe()
```

	User_ID	Age	Marital_Status	Orders	Amount
count	1.123900e+04	11239.000000	11239.000000	11239.000000	11239.000000
mean	1.003004e+06	35.410357	0.420055	2.489634	9453.610553
std	1.716039e+03	12.753866	0.493589	1.114967	5222.355168
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000
25%	1.001492e+06	27.000000	0.000000	2.000000	5443.000000
50%	1.003064e+06	33.000000	0.000000	2.000000	8109.000000
75%	1.004426e+06	43.000000	1.000000	3.000000	12675.000000
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000

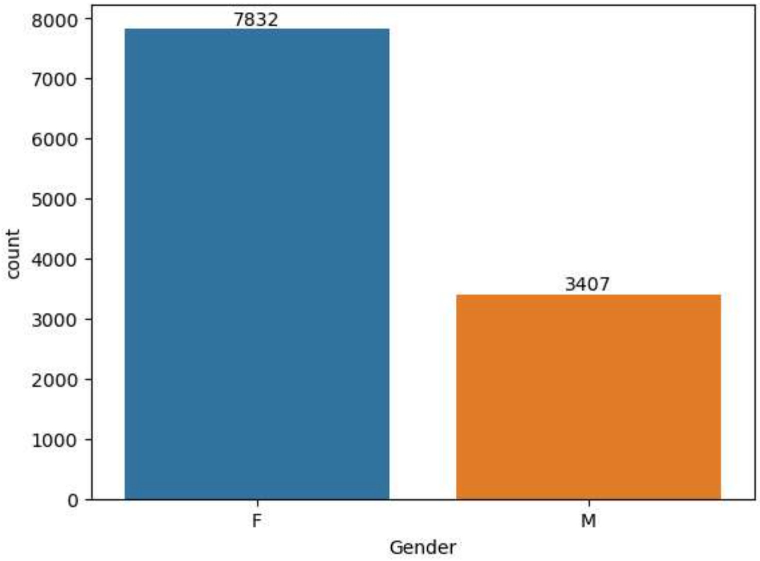
```
df[['Age','Orders','Amount']].describe()
```

	Age	Orders	Amount
count	11239.000000	11239.000000	11239.000000
mean	35.410357	2.489634	9453.610553
std	12.753866	1.114967	5222.355168
min	12.000000	1.000000	188.000000
25%	27.000000	2.000000	5443.000000
50%	33.000000	2.000000	8109.000000
75%	43.000000	3.000000	12675.000000
max	92.000000	4.000000	23952.000000

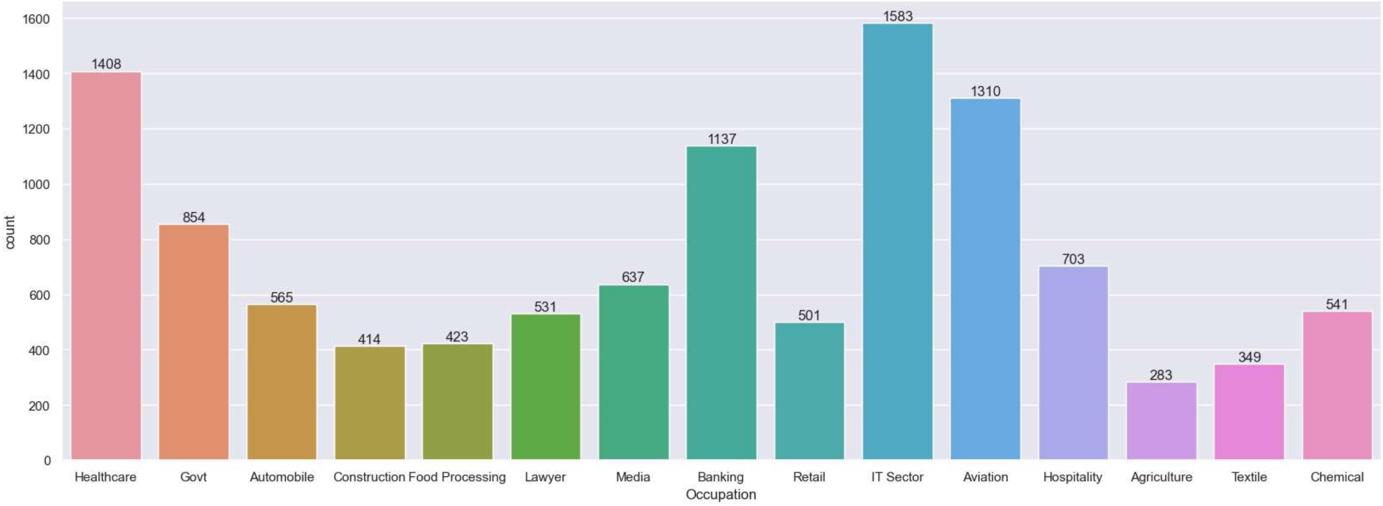
```
df.columns
```

```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

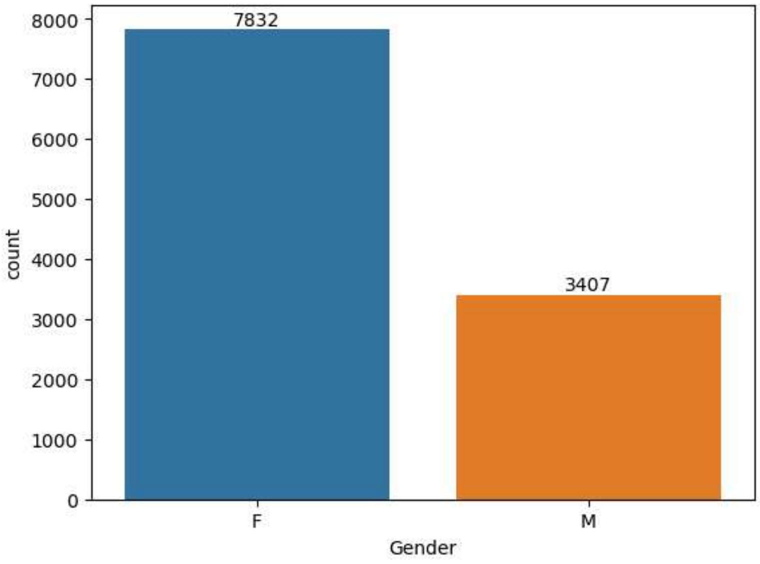
```
ax = sns.countplot(x='Gender' , data= df)
for y in ax.containers:
      ax.bar_label(y)
```



```
ax = sns.countplot(x='Occupation' , data= df)
sns.set(rc={'figure.figsize' : (50,7)})
for y in ax.containers:
      ax.bar_label(y)
```



```
ax = sns.countplot(x='Gender' , data= df)
for y in ax.containers:
    ax.bar_label(y)
```



```
gen = df.groupby(['Gender'] )['Amount'].count()
gen
```

```
Gender
F    7832
M    3407
Name: Amount, dtype: int64
```

```
df.shape
```

```
(11239, 13)
```

```
df['Gender'].count()
```

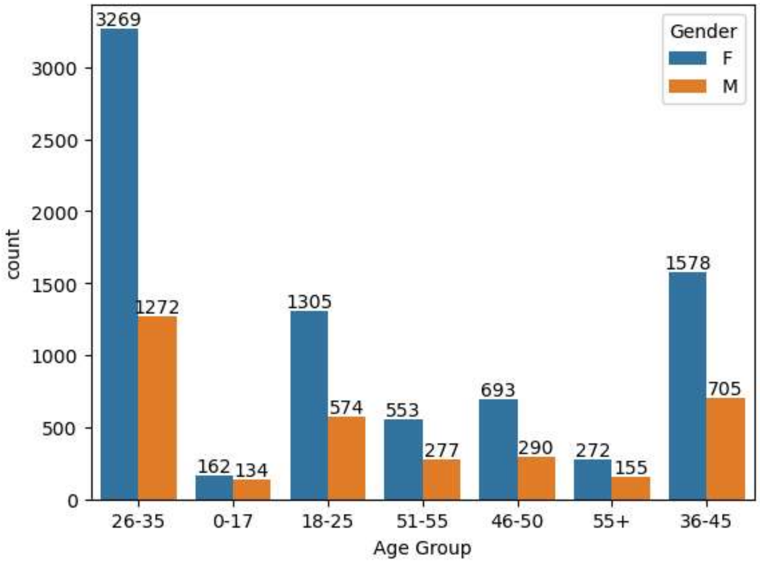
```
11239
```

```
df.groupby(['Occupation'])['Amount'].count()
```

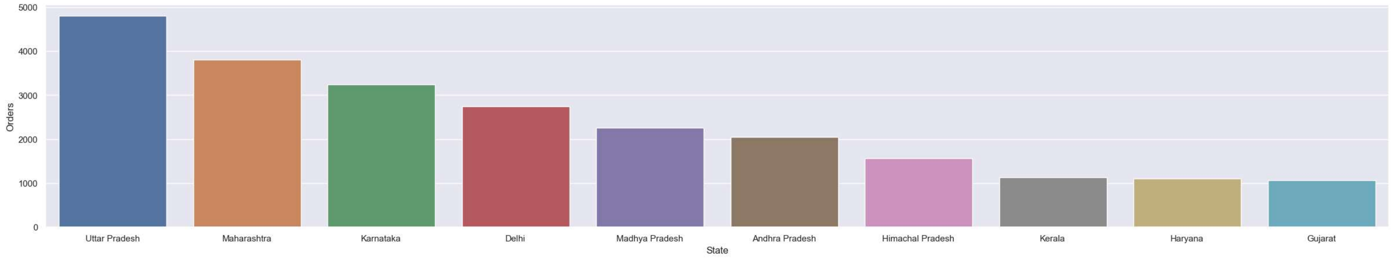
```
Occupation
Agriculture         283
Automobile          565
Aviation           1310
Banking            1137
Chemical            541
Construction        414
Food Processing     423
Govt                854
Healthcare         1408
Hospitality         703
IT Sector          1583
Lawyer              531
Media               637
Retail              501
Textile             349
Name: Amount, dtype: int64
```

Double-click (or enter) to edit

```
ax= sns.countplot(x ='Age Group', data= df , hue ='Gender')
for m in ax.containers:
    ax.bar_label(m)
```



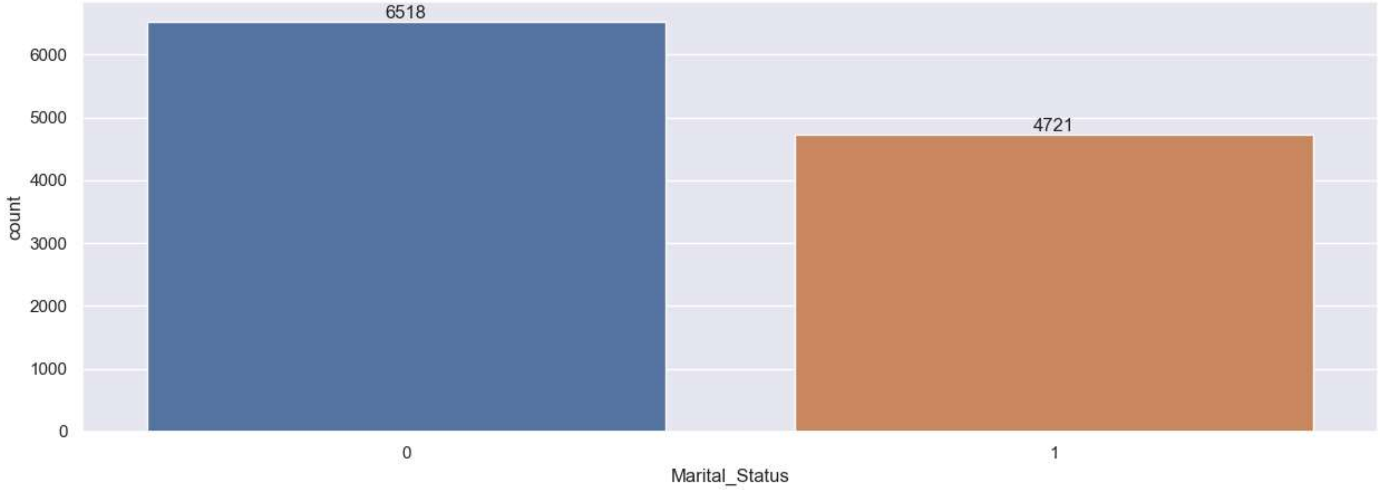
```
sales_state= df.groupby(['State'])['Orders'].sum().sort_values(ascending =False).head(10)
sales_state
sales_state = sales_state.reset_index()
sns.barplot( x='State', y='Orders' , data= sales_state)

sns.set(rc={'figure.figsize': (30, 10)})
```



```
df.columns
```

```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
           'Orders', 'Amount'],
          dtype='object')
```

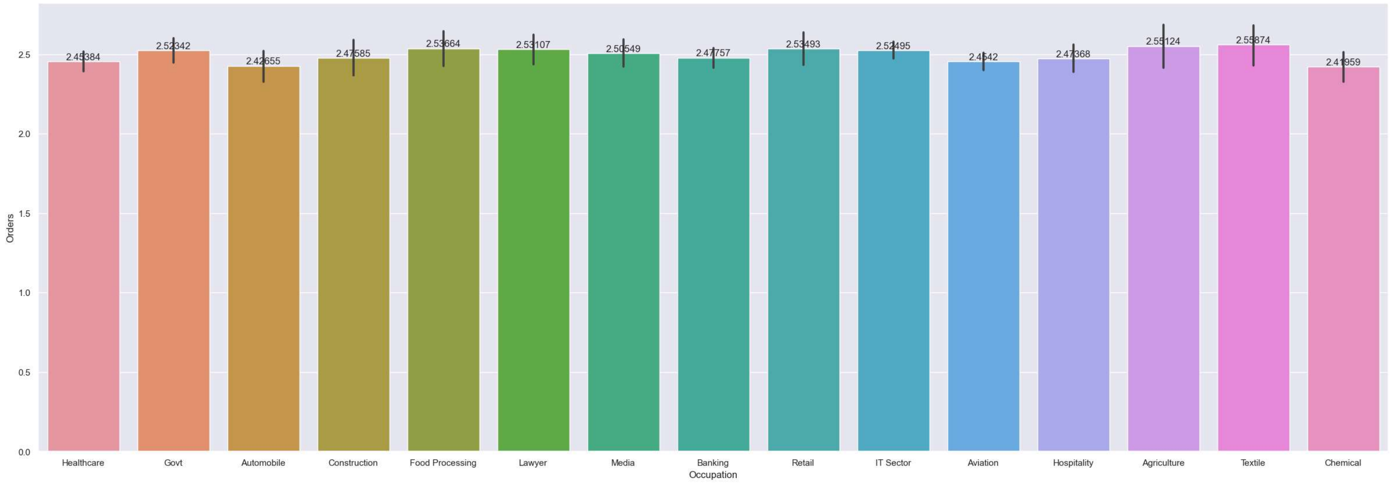
```
ax = sns.countplot(x='Marital_Status' , data=df)
sns.set(rc={'figure.figsize' : (7,15)})
for m in ax.containers:
    ax.bar_label(m)
```



```
df.groupby(['Marital_Status'] )['Amount'].count()
```

```
Marital_Status
0    6518
1    4721
Name: Amount, dtype: int64
```

```
s= df.groupby(['Occupation'])['Orders'].count()
ax = sns.barplot(x='Occupation' , y='Orders' ,data=df
          )
sns.set(rc={'figure.figsize':(30,10)})
for m in ax.containers:
    ax.bar_label(m)
```



```
Conclusion Summary of Data Analysis Project using Python on Diwali Sales Data
```

```
Project Overview:
The data analysis project focused on examining the Diwali sales dataset to uncover patterns, trends, and insights that can help in formulatin

Data Cleaning:
The dataset contained several null values. These missing values were handled using appropriate techniques, such as filling with mean/median/m

Data Type Transformation:
Certain columns in the dataset had incorrect data types that could hinder the analysis. These columns were converted to appropriate data type

Exploratory Data Analysis (EDA) with Seaborn:
EDA was performed using Seaborn to visualize various aspects of the sales data. Graphs such as bar plots, histograms, and box plots were used

Gender-based Customer Analysis:
The analysis revealed that the rate of female customers was higher compared to male customers. This insight is crucial for targeted marketing

Occupation-based Sales Analysis:
The occupation column was analyzed, and it was found that customers from the IT sector had the highest sales figures. This information can be

Age Group Analysis:
        Customers in the age group of 26-35 years were the most prominent shoppers, with a significant portion being female. Understanding th

        State-wise Sales Analysis:
        The state-wise analysis indicated that Uttar Pradesh had the highest sales among all states. This regional insight can guide the allo

Marital Status and Sales:
The analysis between different marital statuses revealed insights into spending patterns. Bachelors and married couples were compared, provid

Strategy to Increase Sales:
Based on the analysis, several strategies can be recommended to increase sales:

Targeted Marketing: Focus on female customers and IT professionals with customized offers and promotions.
Age-specific Campaigns: Create marketing campaigns aimed at the 26-35 age group, emphasizing products and deals that appeal to this demograph
Regional Promotions: Develop state-specific promotions, especially targeting Uttar Pradesh, to leverage the high sales potential.
```