In [1]:
```
import numpy as np
import pandas as pd
```

Import csv file

In [2]:
```
df = pd.read_csv('Diwali Sales Data.csv', encoding = 'unicode_escape')
```

In [3]:
```
# Check How many rows and column in csv file
df.shape
```

Out[3]:
```
(11251, 15)
```

In [4]:
```
df.head()
```

Out[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed1
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0	NaN	NaN
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0	NaN	NaN
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0	NaN	NaN
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.0	NaN	NaN
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.0	NaN	NaN

Data Cleaning

In [5]:
```
#Checking Null Values
```

In [6]:
```
pd.isna(df).sum()
```

Out[6]:
```
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
Status             11251
unnamed1           11251
dtype: int64
```

Result

- in amount column 12 null values,
- status and unnamed1 column are fully null Values

In [8]:
```
# Drop Blank Columns
df.drop(['Status','unnamed1'],axis=1, inplace=True)
```

In [9]:
```
pd.isna(df).sum()
```

Out[9]:
```
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
```

In [10]:
```
#Drop Null Values
```

In [11]:
```
df.dropna(inplace=True)
```

In [15]:
```
#Verify Null Values are removed or not
pd.isna(df).sum()
```

Out[15]:
```
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
```

All Null value are removed

In [16]:
```
df.head()
```

Out[16]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.0
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.0

In [17]:
```
# summary of dataframe
```

In [18]:
```
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  float64
dtypes: float64(1), int64(4), object(8)
memory usage: 1.2+ MB
```

In [19]:
```
# Change data type of Amount Column Float to int
df['Amount'] = df['Amount'].astype('int')
```

In [20]:
```
df['Amount'].dtypes
```

Out[20]:
```
dtype('int32')
```

In [21]:
```
# View All Column names in CSV File
df.columns
```

Out[21]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [ ]:
```

```

In [22]:
```
# Description of the dataframe
df.describe()
```

Out[22]:

	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

In [23]:
```
df.head()
```

Out[23]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877

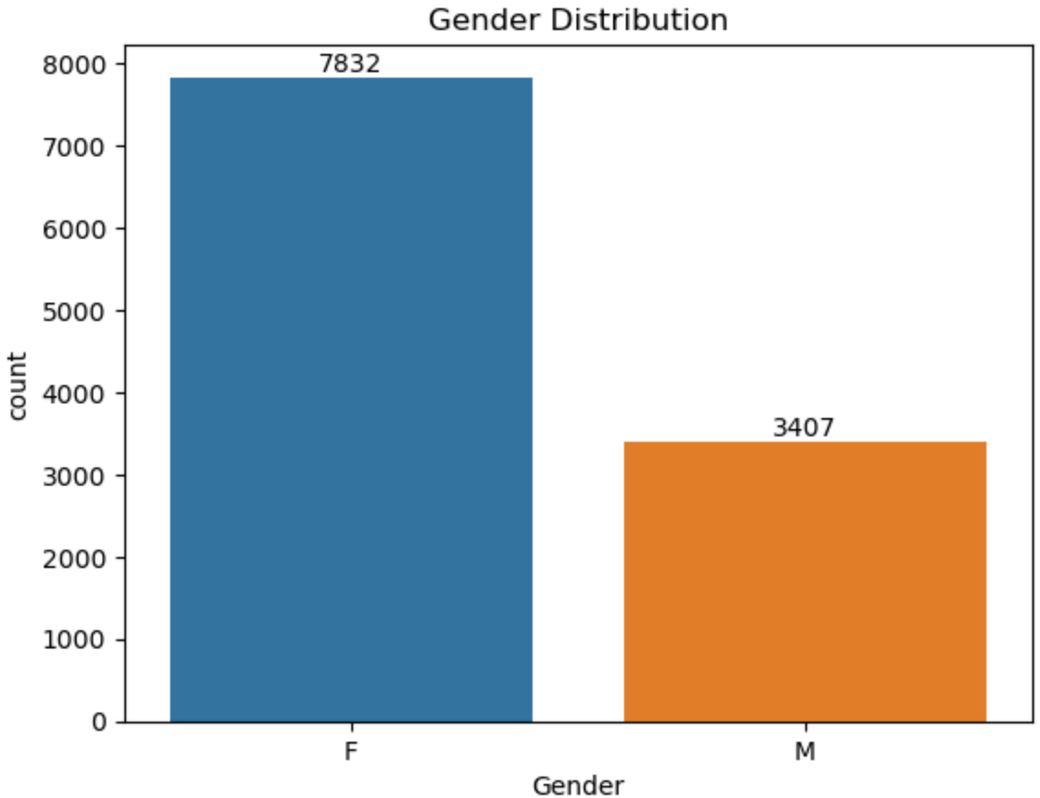
Exploratory Data Analysis

In [24]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

Gender Distribution

In [27]:
```
ax = sns.countplot(x= 'Gender', data =df)
for bars in ax.containers:
    ax.bar_label(bars)

plt.title('Gender Distribution')
plt.show()
```



Gender Comparison in Total Sales Amount

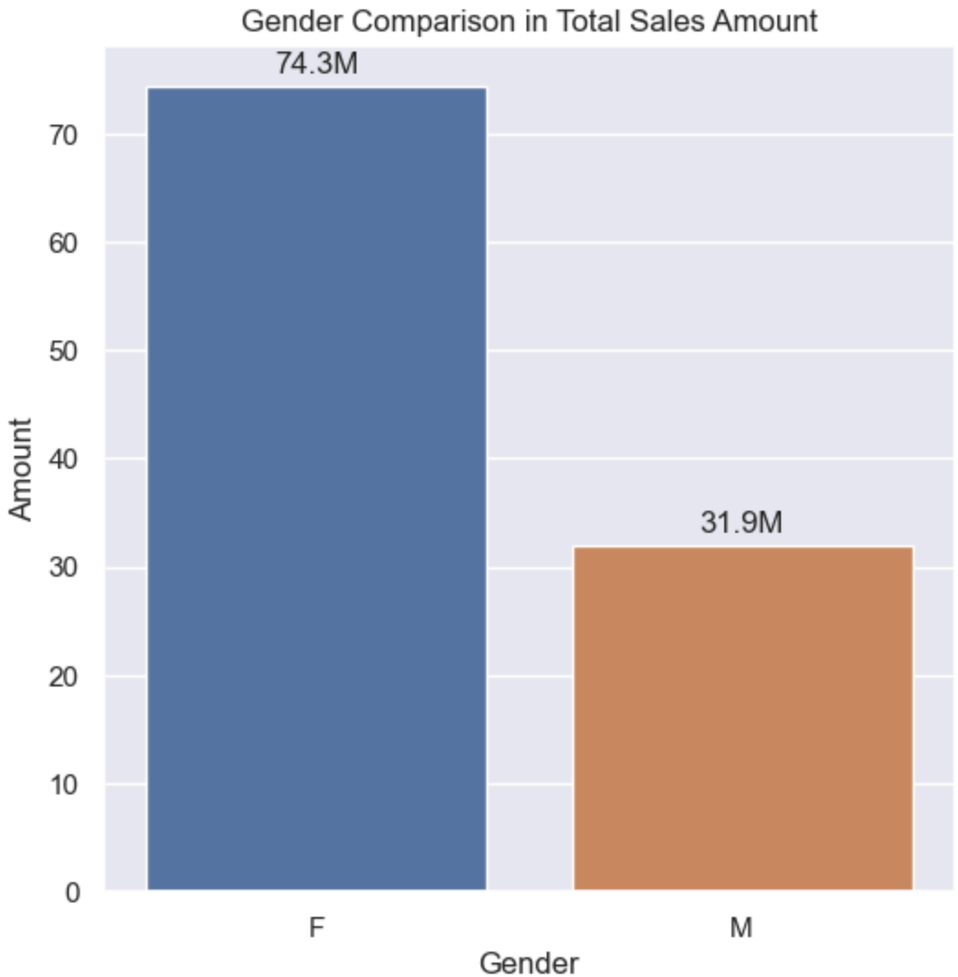
In [92]:
```
gen_wise_sales = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
gen_wise_sales['Amount'] = gen_wise_sales['Amount'] / 1_000_000

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

ax = sns.barplot(x='Gender', y='Amount', data=gen_wise_sales)

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)
```

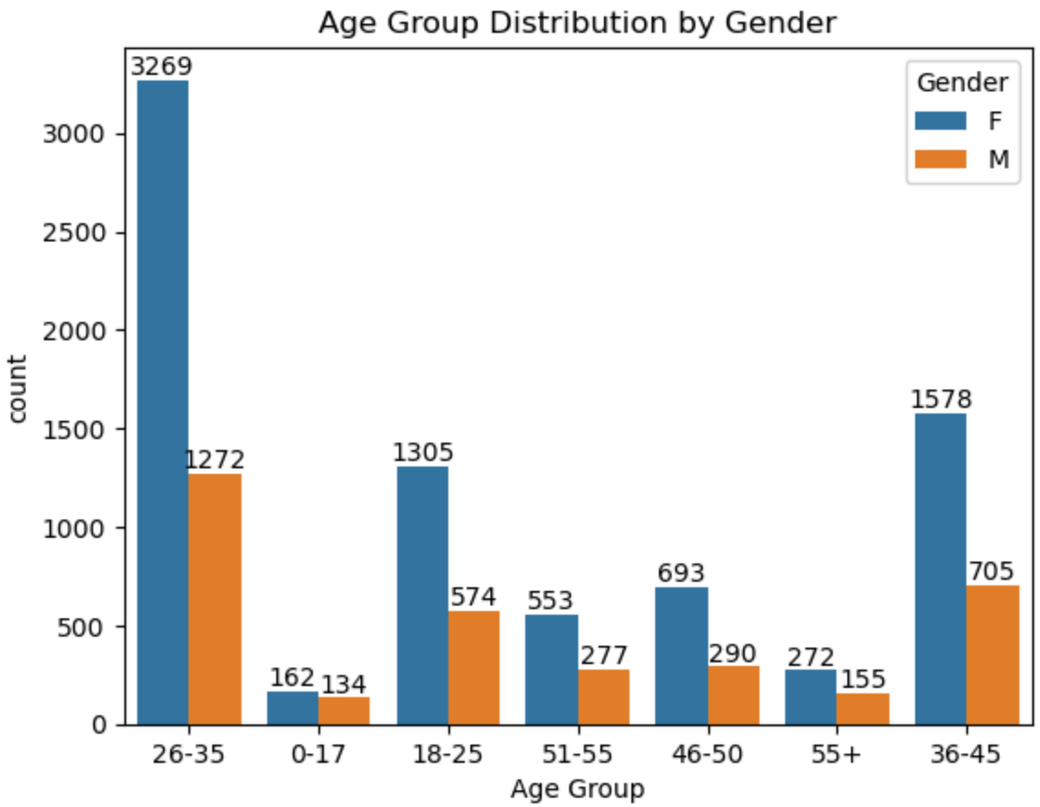
```
plt.title('Gender Comparison in Total Sales Amount')
plt.show()
```



Observation - The graph demonstrates that females are the predominant buyers and exhibit higher purchasing power compared to their male counterparts.

Age

In [39]:
```
ax = sns.countplot(data=df, x='Age Group', hue='Gender')
for container in ax.containers:
    ax.bar_label(container)
plt.title('Age Group Distribution by Gender')
plt.show()
```



In [ ]:
```

```

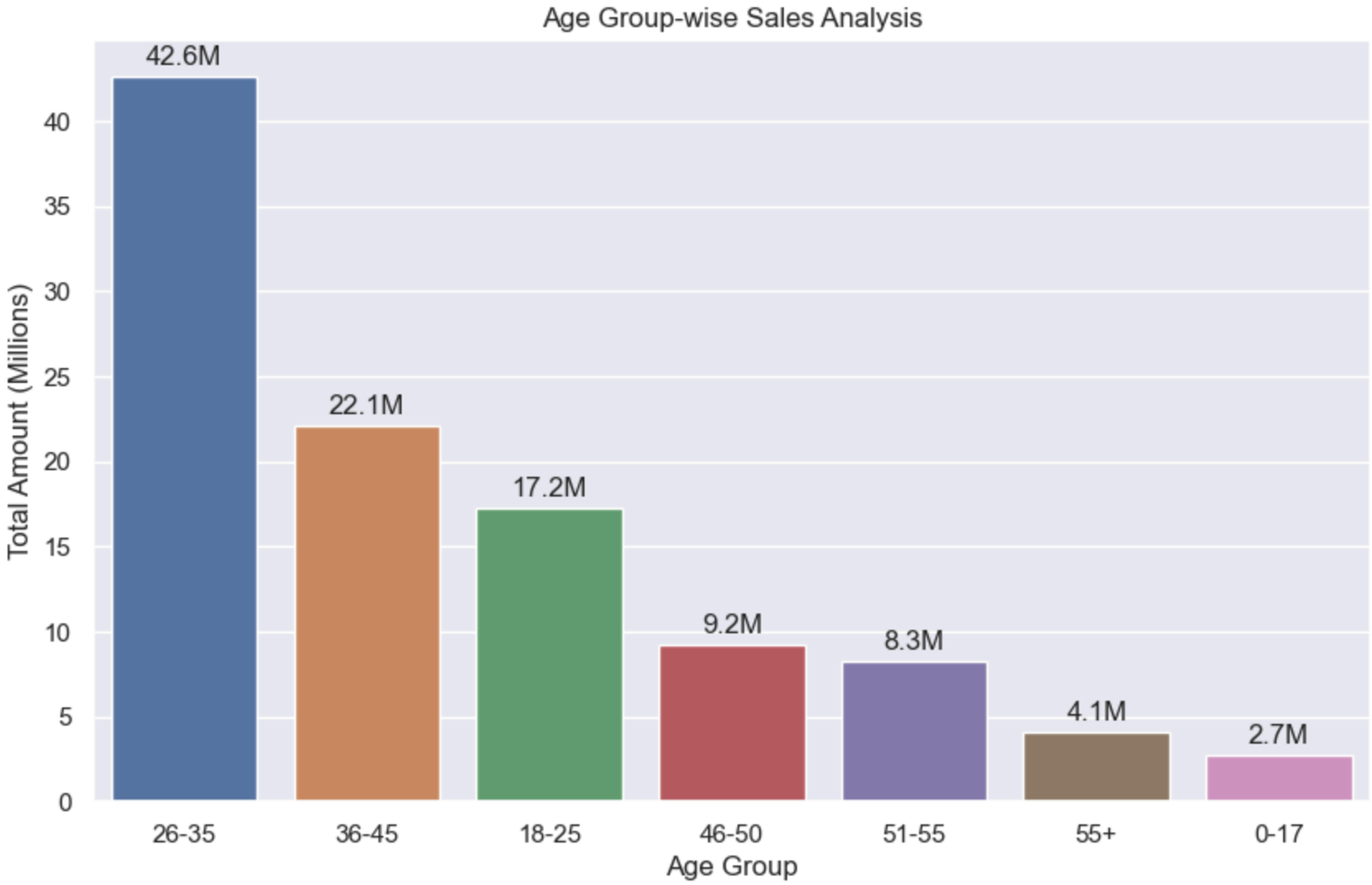
Age Group-wise Sales Analysis

In [89]:
```
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sales_age['Amount'] = sales_age['Amount'] / 1_000_000
```

```
plt.figure(figsize=(10, 6))
ax = sns.barplot(x='Age Group', y='Amount', data=sales_age)

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)

plt.title('Age Group-wise Sales Analysis')
plt.xlabel('Age Group')
plt.ylabel('Total Amount (Millions)')
plt.show()
```



Observation -The analysis of the graphs indicates that the predominant buyer demographic is females aged between 26-35 years.

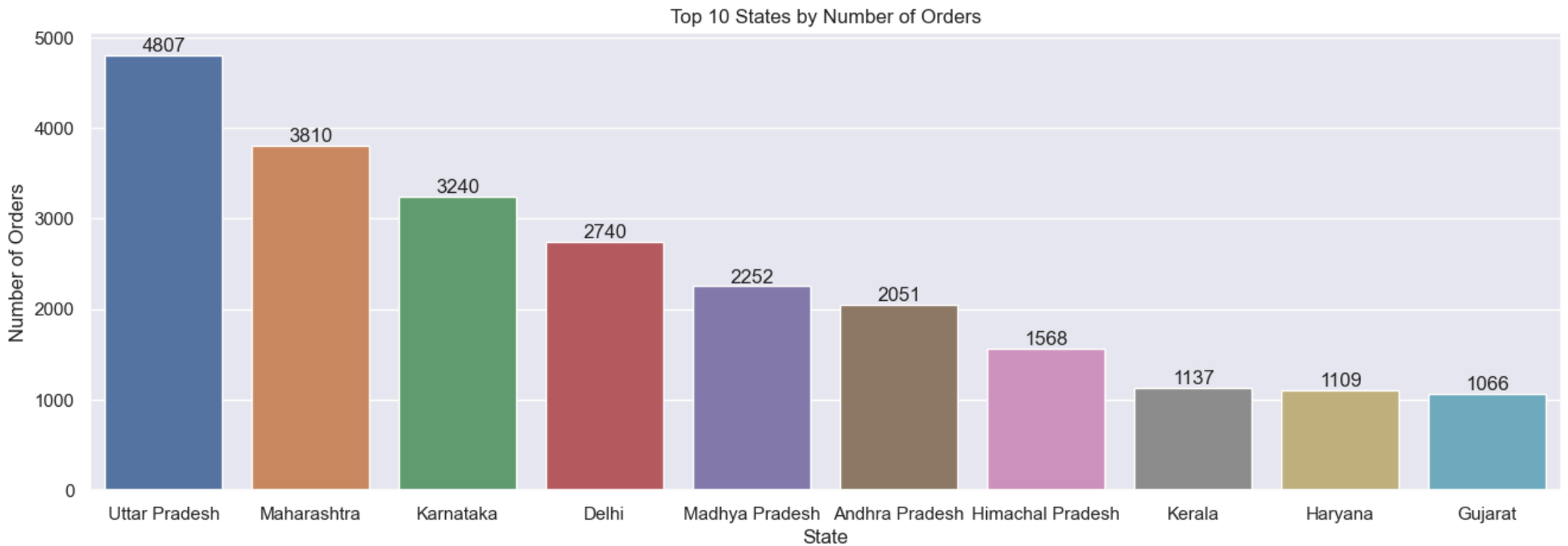
State

In [47]:
```
#Top 10 States by Number of Orders
```

In [46]:
```
sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(16,5)})
ax = sns.barplot(data=sales_state, x='State', y='Orders')
for container in ax.containers:
    ax.bar_label(container)

ax.set_title('Top 10 States by Number of Orders')
ax.set_xlabel('State')
ax.set_ylabel('Number of Orders')
plt.show()
```



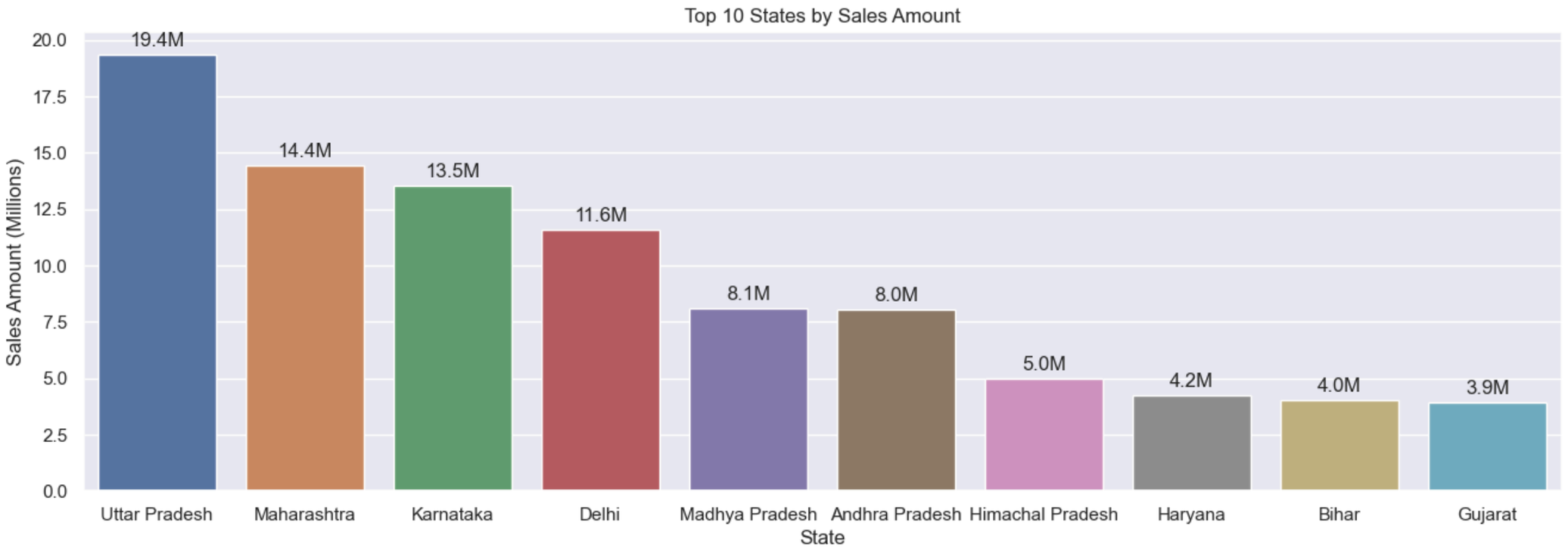
In [48]:
```
# Top 10 States by Sales Amount
```

In [64]:
```
sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)
sales_state['Amount'] = sales_state['Amount'] / 1_000_000
```

```
sns.set(rc={'figure.figsize':(16,5)})
ax = sns.barplot(data=sales_state, x='State', y='Amount')

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)

ax.set_title('Top 10 States by Sales Amount')
ax.set_xlabel('State')
ax.set_ylabel('Sales Amount (Millions)')
plt.show()
```



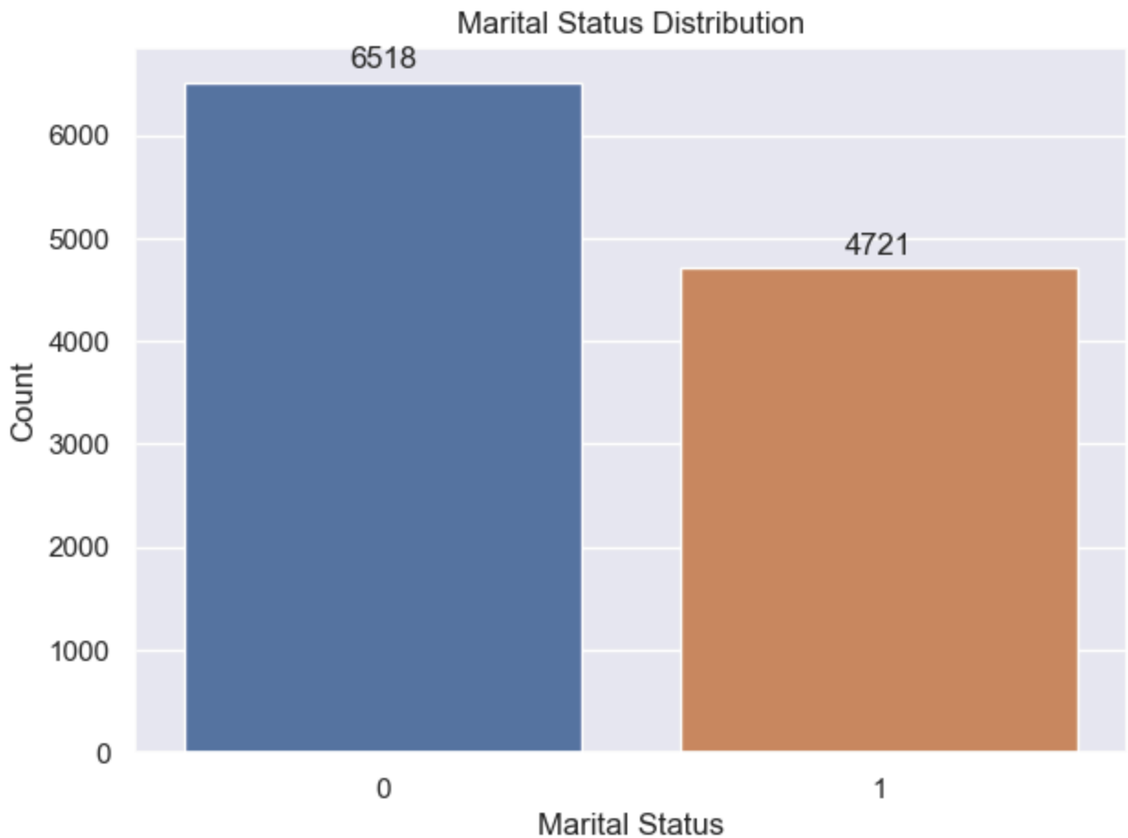
Observation - The graphs indicate that the highest volume of orders and total sales are concentrated in Uttar Pradesh, Maharashtra, and Karnataka, respectively.

Marital Status

In [54]:
```
ax = sns.countplot(data=df, x='Marital_Status')
sns.set(rc={'figure.figsize':(7,5)})

for container in ax.containers:
    ax.bar_label(container, fmt='%d', padding=3)

ax.set_title('Marital Status Distribution')
ax.set_xlabel('Marital Status')
ax.set_ylabel('Count')
plt.show()
```



In [ ]:

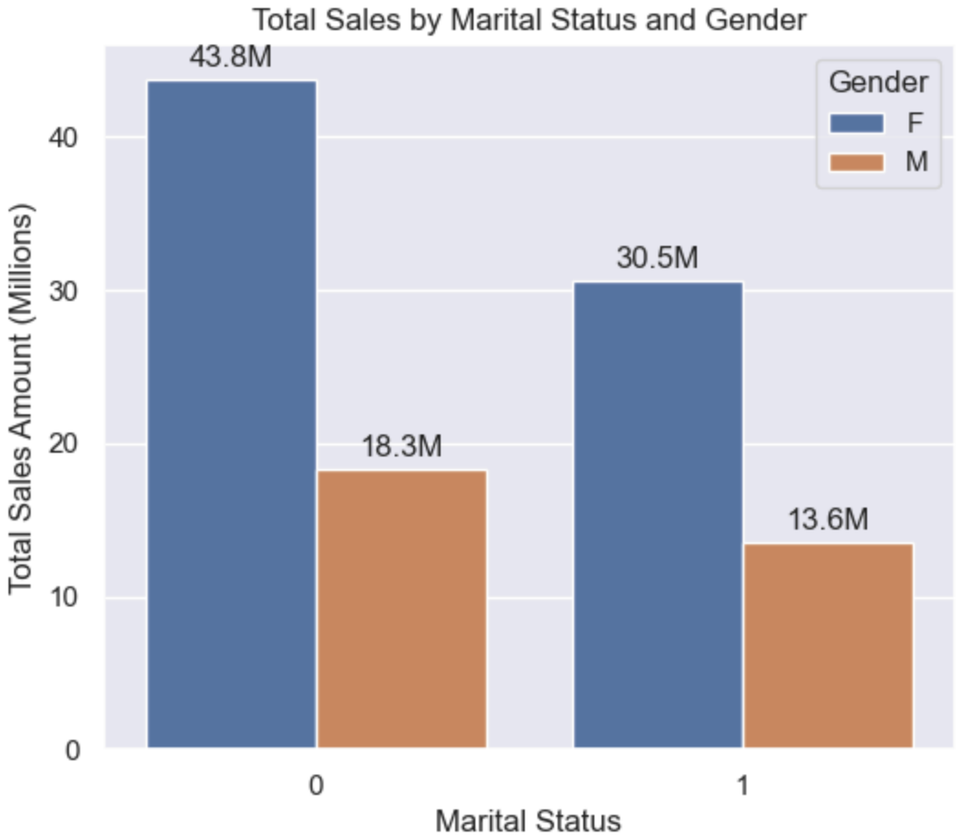
In [65]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sales_state['Amount'] = sales_state['Amount'] / 1_000_000

sns.set(rc={'figure.figsize':(6,5)})
ax = sns.barplot(data=sales_state, x='Marital_Status', y='Amount', hue='Gender')

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)

ax.set_title('Total Sales by Marital Status and Gender')
ax.set_xlabel('Marital Status')
ax.set_ylabel('Total Sales Amount (Millions)')
plt.show()
```

```
C:\Users\vinay\AppData\Local\Temp\ipykernel_8740\565062681.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to True in a f
uture version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True
in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_vals = vals.groupby(grouper)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True
in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_vals = vals.groupby(grouper)
```



Observation - The graphs indicate that the majority of buyers are married women, who demonstrate higher purchasing power.

In [ ]:

Occupation

In [67]:
```
#Number of Orders by Occupation
```

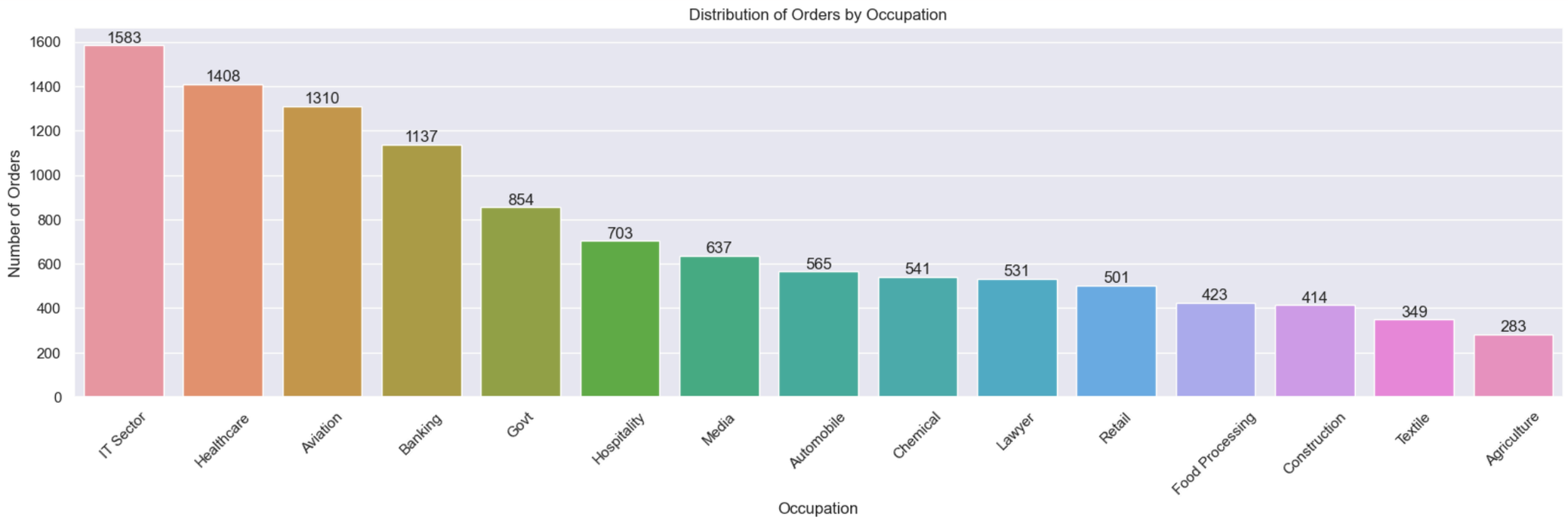
In [88]:
```
occupation_counts = df['Occupation'].value_counts().sort_values(ascending=False)
sorted_df = df[df['Occupation'].isin(occupation_counts.index)]
sorted_df['Occupation'] = pd.Categorical(sorted_df['Occupation'], categories=occupation_counts.index, ordered=True)
```

```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data=sorted_df, x='Occupation', order=occupation_counts.index)

for container in ax.containers:
    ax.bar_label(container)

ax.set_title('Distribution of Orders by Occupation')
ax.set_xlabel('Occupation')
ax.set_ylabel('Number of Orders')
plt.xticks(rotation=45)
plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_vals = vals.groupby(grouper)
```



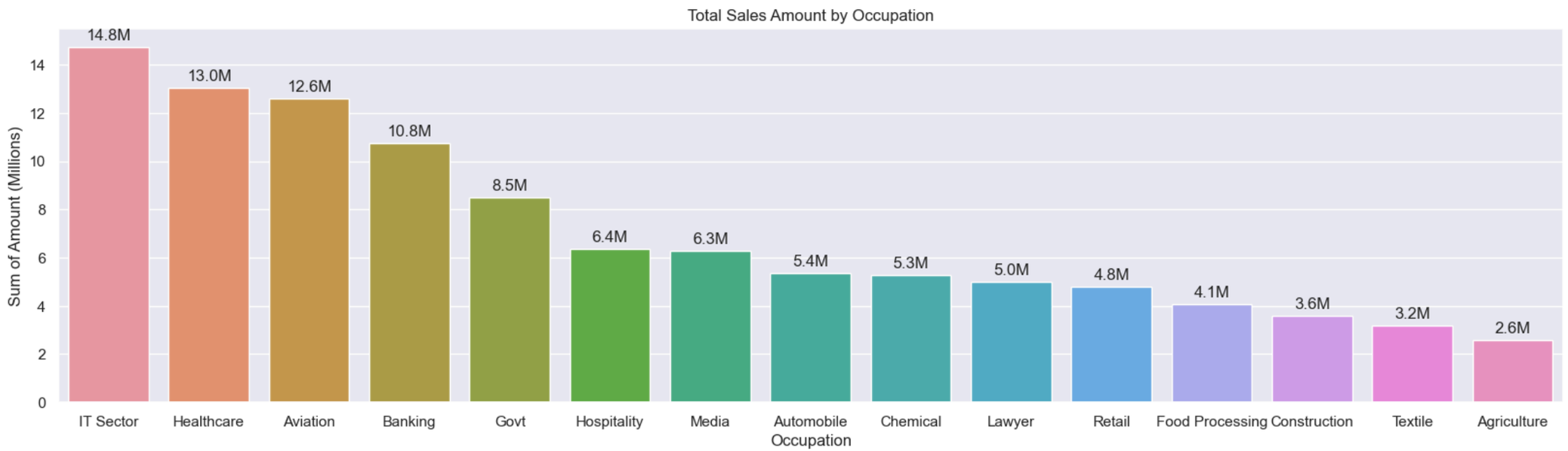
In [ ]:

In [87]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sales_state['Amount'] = sales_state['Amount'] / 1_000_000

sns.set(rc={'figure.figsize':(20,5)})
ax = sns.barplot(data=sales_state, x='Occupation', y='Amount')

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)

ax.set_title('Total Sales Amount by Occupation')
ax.set_xlabel('Occupation')
ax.set_ylabel('Sum of Amount (Millions)')
plt.show()
```



Observation - The graphs indicate that the majority of buyers are employed in the IT, Healthcare, and Aviation sectors.

In [ ]:
```
```

Product Category

In [79]:
```
# Top 10 Product Categories by Number of Orders
```

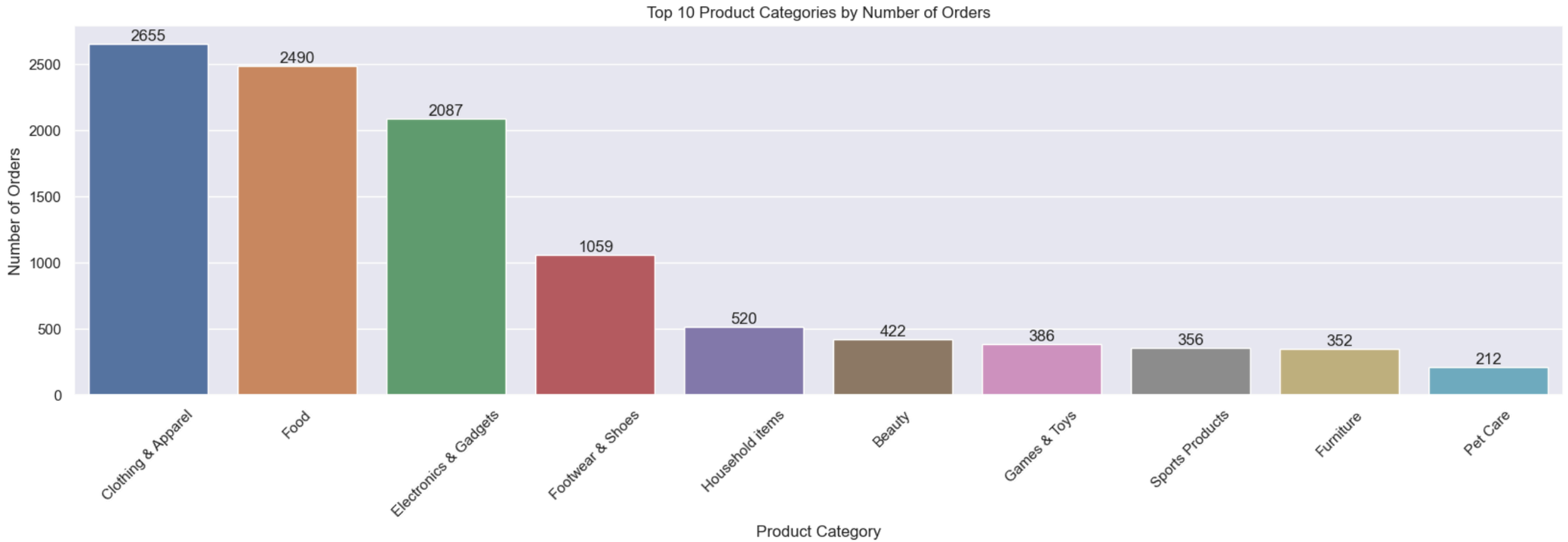
In [76]:
```
category_counts = df['Product_Category'].value_counts()
top_categories = category_counts.head(10).index
top_categories_df = df[df['Product_Category'].isin(top_categories)]
top_categories_counts = top_categories_df['Product_Category'].value_counts().reindex(top_categories)
top_categories_df_for_plot = top_categories_counts.reset_index()
top_categories_df_for_plot.columns = ['Product_Category', 'Number_of_Orders']

sns.set(rc={'figure.figsize':(20,5)})
ax = sns.barplot(data=top_categories_df_for_plot, x='Product_Category', y='Number_of_Orders')

for container in ax.containers:
    ax.bar_label(container)

ax.set_title('Top 10 Product Categories by Number of Orders')
ax.set_xlabel('Product Category')
ax.set_ylabel('Number of Orders')

plt.xticks(rotation=45)
plt.show()
```



Observation - The highest Number of order are attributed to the Clothing, Food and Electronics categories.

In [78]:
```
# Top 10 Product Categories by Total Sales Amount
```

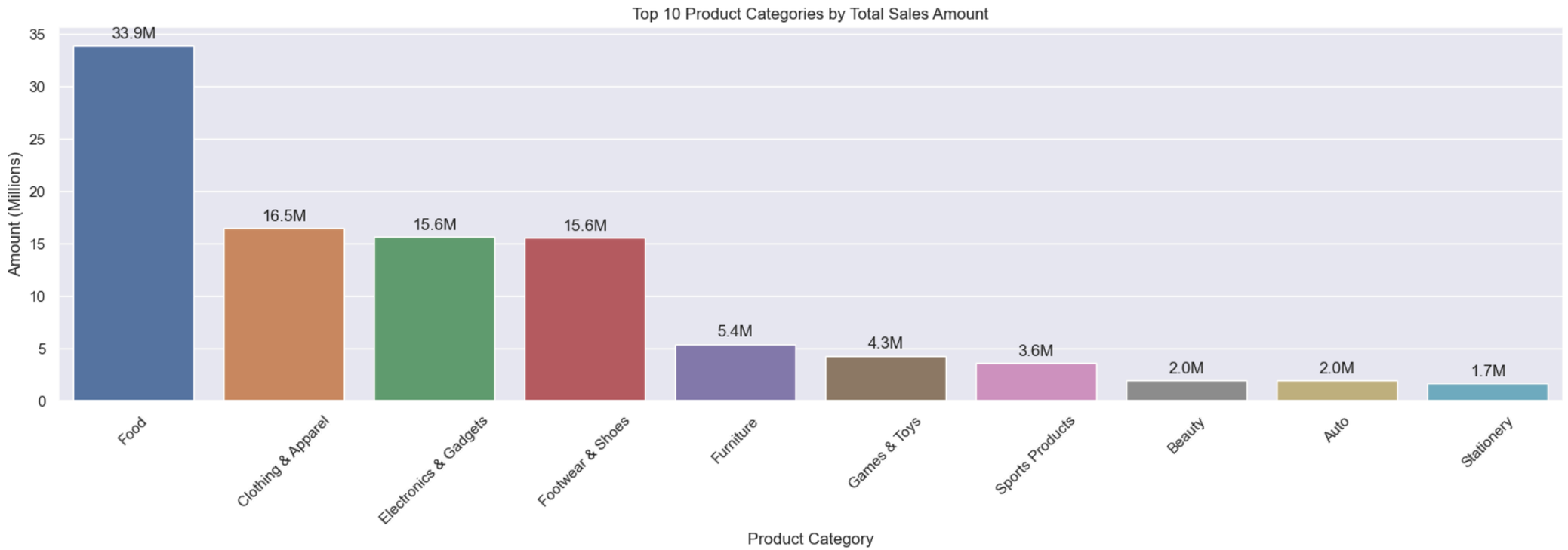
In [75]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)
sales_state['Amount'] = sales_state['Amount'] / 1_000_000

sns.set(rc={'figure.figsize':(20,5)})
ax = sns.barplot(data=sales_state, x='Product_Category', y='Amount')

for container in ax.containers:
    ax.bar_label(container, fmt='%.1fM', padding=3)

ax.set_title('Top 10 Product Categories by Total Sales Amount')
ax.set_xlabel('Product Category')
ax.set_ylabel('Amount (Millions)')
```

```
plt.xticks(rotation=45)
plt.show()
```



Observation - The highest sales amounts are attributed to the Food, Clothing, and Electronics categories.

In [ ]:

In [81]:
```
# Top 10 Products by Number of Orders
```

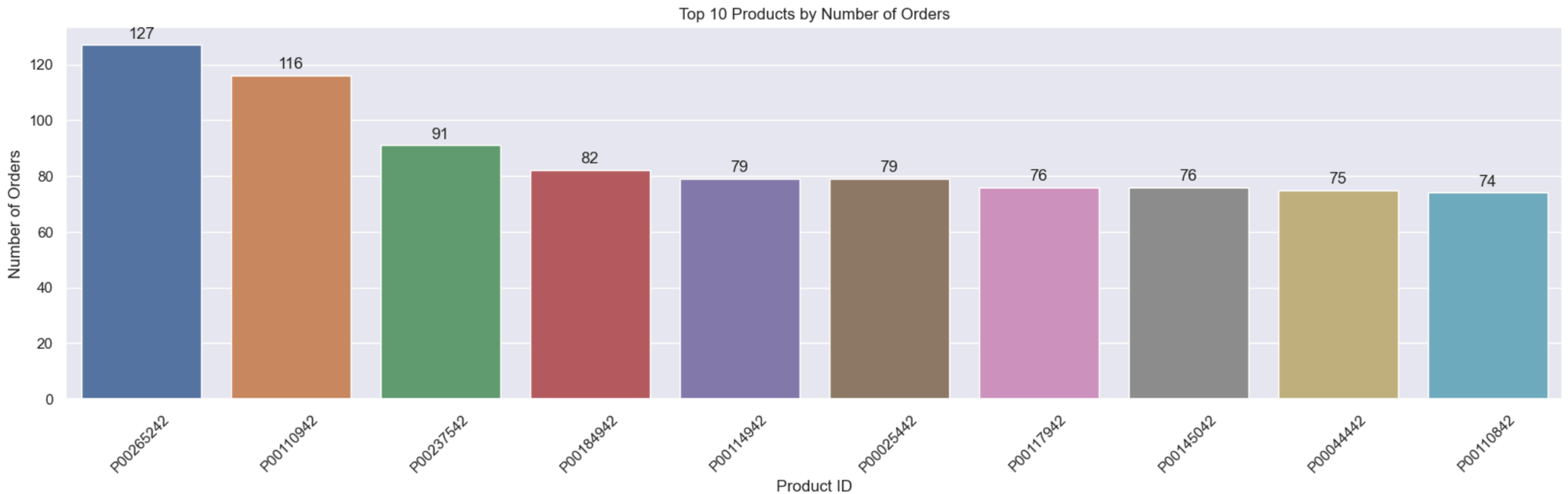
In [82]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
ax = sns.barplot(data=sales_state, x='Product_ID', y='Orders')

for container in ax.containers:
```

```
    ax.bar_label(container, fmt='%d', padding=3)

ax.set_title('Top 10 Products by Number of Orders')
ax.set_xlabel('Product ID')
ax.set_ylabel('Number of Orders')
plt.xticks(rotation=45)
plt.show()
```



In [ ]:

In [ ]:

Final Conclusion

- The analysis of the provided graphs yields several key insights into the purchasing behaviors and demographics of buyers:
- Gender and Purchasing Power: The data demonstrates that females are the predominant buyers

and exhibit higher purchasing power compared to their male counterparts.

- Age Demographics: The analysis reveals that the primary buyer demographic is females aged between 26-35 years.
- Geographic Concentration: The highest volume of orders and total sales are concentrated in Uttar Pradesh, Maharashtra, and Karnataka.
- Marital Status: The data indicates that the majority of buyers are married women, who also show higher purchasing power.
- Occupational Distribution: Most buyers are employed in the IT, Healthcare, and Aviation sectors.
- Product Categories: The highest number of orders are for Clothing, Food, and Electronics categories,

and these categories also account for the highest sales amounts.

These observations provide valuable insights for targeted marketing strategies and inventory management, emphasizing the importance of catering to female buyers, especially those aged 26-35, and focusing efforts on key geographic regions and product categories.

In [ ]:

Contact Information

GitHub: Vinay Kumar Panika's
LinkedIn: Vinay Kumar Panika's