In [1]:
```
# Importing libraries

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

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

In [8]:
```
df.shape
```

Out[8]: (11251, 14)

In [9]:
```
df.head()
```

Out[9]:

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

In [10]:
```
df.tail()
```

Out[10]:

	User_ID	Cust_name	Product_ID	Gender	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orde
11246	1000695	Manning	P00296942	M	19	1	Maharashtra	Western	Chemical	Office	
11247	1004089	Reichenbach	P00171342	M	33	0	Haryana	Northern	Healthcare	Veterinary	
11248	1001209	Oshin	P00201342	F	40	0	Madhya Pradesh	Central	Textile	Office	
11249	1004023	Noonan	P00059442	M	37	0	Karnataka	Southern	Agriculture	Office	
11250	1002744	Brumley	P00281742	F	19	0	Maharashtra	Western	Healthcare	Office	

In [11]:
```
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               11251 non-null  int64  
 5   Marital_Status    11251 non-null  int64  
 6   State             11251 non-null  object 
 7   Zone              11251 non-null  object 
 8   Occupation        11251 non-null  object 
 9   Product_Category  11251 non-null  object 
 10  Orders            11251 non-null  int64  
 11  Amount            11239 non-null  float64
 12  Status            0 non-null      float64
 13  Unnamed           0 non-null      float64
dtypes: float64(3), int64(4), object(7)
memory usage: 1.2+ MB
```

In [13]:
```
#Dropping columns
df.drop(['Status','Unnamed'],axis=1,inplace=True)
```

In [15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 12 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               11251 non-null  int64  
 5   Marital_Status    11251 non-null  int64  
 6   State             11251 non-null  object 
 7   Zone              11251 non-null  object 
 8   Occupation        11251 non-null  object 
 9   Product_Category  11251 non-null  object 
 10  Orders            11251 non-null  int64  
 11  Amount            11239 non-null  float64
dtypes: float64(1), int64(4), object(7)
memory usage: 1.0+ MB
```

In [16]:
```
pd.isnull(df)
```

Out[16]:

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

11251 rows × 12 columns

In [17]:
```
pd.isnull(df).sum()
```

Out[17]:
```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
Product_Category    0
Orders              0
Amount             12
dtype: int64
```

In [19]:
```
#getting null rows
row_with_nulls= df[df.isnull().any(axis=1)]
print("Rows with Nulls: ")
print(row_with_nulls)
```

```
Rows with Nulls:
     User_ID Cust_name Product_ID Gender  Age  Marital_Status          State  \
7    1002092  Shivangi  P00273442      F   61               0    Maharashtra
14   1003858      Cano  P00293742      M   46               1   Madhya Pradesh
16   1005447       Amy  P00275642      F   48               1   Andhra Pradesh
109  1005265    Sakshi  P00296242      F   48               1          Delhi
111  1005261   Apoorva  P00057942      F   41               1          Delhi
184  1005538    Kartik  P00269542      F   49               1      Karnataka
293  1000326  Jonathan  P00120542      M   53               0        Gujarat
344  1002507   Lakshmi  P00045842      F   35               1        Gujarat
345  1004498   Srishti  P00030842      F   55               0          Delhi
452  1004601    Gaurav  P00014442      F   40               1   Madhya Pradesh
464  1004528    Anurag  P00338442      F   33               1    Uttar Pradesh
493  1002994    Hemant   P0009942      F   38               0    Uttar Pradesh

         Zone   Occupation  Product_Category  Orders  Amount
7     Western    IT Sector              Auto       1     NaN
14    Central  Hospitality              Auto       3     NaN
16   Southern    IT Sector              Auto       3     NaN
109   Central      Banking  Footwear & Shoes       1     NaN
111   Central    IT Sector  Footwear & Shoes       2     NaN
184  Southern      Banking  Footwear & Shoes       1     NaN
293   Western    IT Sector  Footwear & Shoes       3     NaN
344   Western     Chemical         Furniture       1     NaN
345   Central      Textile  Footwear & Shoes       1     NaN
452   Central  Hospitality              Food       4     NaN
464   Central   Automobile              Food       2     NaN
493   Central    IT Sector              Food       4     NaN
```

In [20]:
```
#Calculating mean for respective category and fill NA with mean using groupby
df['Amount']=df.groupby('Product_Category')['Amount'].transform(lambda x:x.fillna(x.mean()))
```

In [22]:
```
#checking above step is applied or not
selected_rows= df[(df['User_ID']==1004498)&(df['Product_ID']=='P00030842')]
print(selected_rows)
```

```
     User_ID Cust_name Product_ID Gender  Age  Marital_Status  State     Zone  \
345  1004498   Srishti  P00030842      F   55               0  Delhi  Central

    Occupation  Product_Category  Orders        Amount
345    Textile  Footwear & Shoes       1  14707.468791
```

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

Out[23]:
```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
Product_Category    0
Orders              0
Amount              0
dtype: int64
```

In [24]:
```
df['Amount']=df['Amount'].astype('int')
```

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 12 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               11251 non-null  int64
 5   Marital_Status    11251 non-null  int64
 6   State             11251 non-null  object
 7   Zone              11251 non-null  object
 8   Occupation        11251 non-null  object
 9   Product_Category  11251 non-null  object
 10  Orders            11251 non-null  int64
 11  Amount            11251 non-null  int32
dtypes: int32(1), int64(4), object(7)
memory usage: 1011.0+ KB
```

In [28]:
```
df.columns
```

Out[28]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age', 'Marital_Status',
       'State', 'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount'],
      dtype='object')
```

In [29]:
```
df.rename(columns={'Marital_Status':'Married'})
```

Out[29]:

	User_ID	Cust_name	Product_ID	Gender	Age	Married	State	Zone	Occupation	Product_Category	Orders
0	1002903	Sanskriti	P00125942	F	28	0	Maharashtra	Western	Healthcare	Auto	1
1	1000732	Kartik	P00110942	F	35	1	Andhra Pradesh	Southern	Govt	Auto	3
2	1001990	Bindu	P00118542	F	35	1	Uttar Pradesh	Central	Automobile	Auto	3
3	1001425	Sudevi	P00237842	M	16	0	Karnataka	Southern	Construction	Auto	2
4	1000588	Joni	P00057942	M	28	1	Gujarat	Western	Food Processing	Auto	2
...	...	...	...	...	...	...	...	...	...	...	...
11246	1000695	Manning	P00296942	M	19	1	Maharashtra	Western	Chemical	Office	4
11247	1004089	Reichenbach	P00171342	M	33	0	Haryana	Northern	Healthcare	Veterinary	3
11248	1001209	Oshin	P00201342	F	40	0	Madhya Pradesh	Central	Textile	Office	4
11249	1004023	Noonan	P00059442	M	37	0	Karnataka	Southern	Agriculture	Office	3
11250	1002744	Brumley	P00281742	F	19	0	Maharashtra	Western	Healthcare	Office	3

11251 rows × 12 columns

In [34]:
```
#summarizing the data
df[['Age','Orders','Amount']].describe()
```

Out[34]:

	Age	Orders	Amount
count	11251.000000	11251.000000	11251.000000
mean	35.421207	2.489290	9460.454626
std	12.754122	1.115047	5224.434220
min	12.000000	1.000000	188.000000
25%	27.000000	1.500000	5443.500000
50%	33.000000	2.000000	8110.000000
75%	43.000000	3.000000	12691.500000
max	92.000000	4.000000	23952.000000

In [35]:
```
#Categorizing the age group
def categorize_age(age):
    if age<= 17:
        return '0-17'
    elif age<=25:
        return '18-25'
    elif age<=35:
        return '26-35'
    elif age<=45:
        return '36-45'
    elif age<=50:
        return '46-50'
    elif age<=55:
        return '51-55'
    else:
        return '+55'
#Applying above conditions and creating new coloumn
df['Age_Group']=df['Age'].apply(categorize_age)
print(df)
```

```
       User_ID    Cust_name Product_ID Gender  Age  Marital_Status  \
0      1002903    Sanskriti  P00125942      F   28               0
1      1000732       Kartik  P00110942      F   35               1
2      1001990        Bindu  P00118542      F   35               1
3      1001425       Sudevi  P00237842      M   16               0
4      1000588         Joni  P00057942      M   28               1
...        ...          ...        ...    ...  ...             ...
11246  1000695      Manning  P00296942      M   19               1
11247  1004089  Reichenbach  P00171342      M   33               0
11248  1001209        Oshin  P00201342      F   40               0
11249  1004023       Noonan  P00059442      M   37               0
11250  1002744      Brumley  P00281742      F   19               0

                State      Zone       Occupation Product_Category  Orders  \
0         Maharashtra   Western       Healthcare             Auto       1
1      Andhra Pradesh  Southern             Govt             Auto       3
2       Uttar Pradesh   Central       Automobile             Auto       3
3           Karnataka  Southern     Construction             Auto       2
4             Gujarat   Western  Food Processing             Auto       2
...               ...       ...              ...              ...     ...
11246     Maharashtra   Western         Chemical           Office       4
11247         Haryana  Northern       Healthcare       Veterinary       3
11248  Madhya Pradesh   Central          Textile           Office       4
11249       Karnataka  Southern      Agriculture           Office       3
11250     Maharashtra   Western       Healthcare           Office       3

       Amount Age_Group
0       23952     26-35
1       23934     26-35
2       23924     26-35
3       23912      0-17
4       23877     26-35
...       ...       ...
11246     370     18-25
11247     367     26-35
11248     213     36-45
11249     206     36-45
11250     188     18-25

[11251 rows x 13 columns]
```

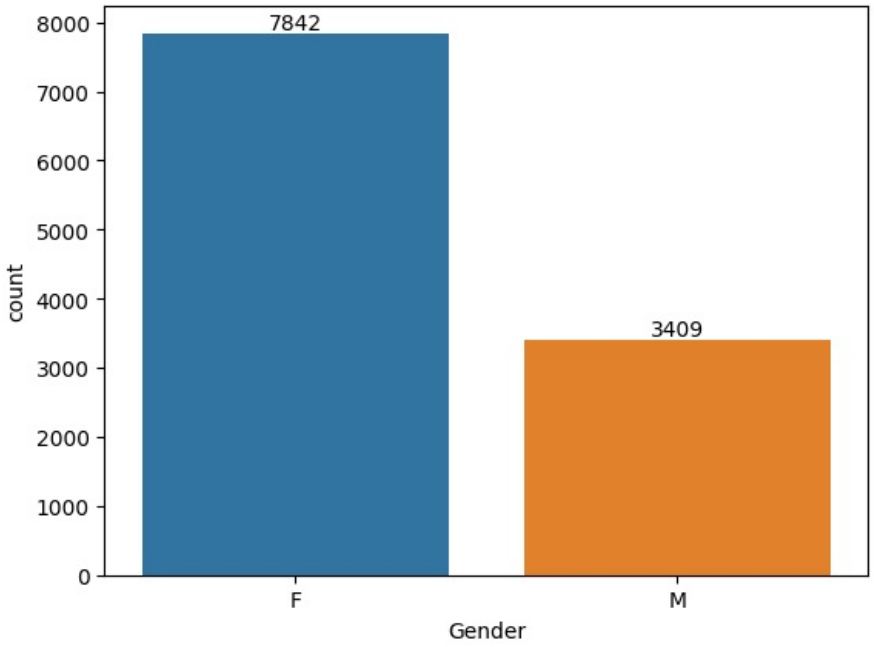
EDA-Exploratory Data Analysis

In [37]:
```
df.columns
```

Out[37]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age', 'Marital_Status',
       'State', 'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount',
       'Age_Group'],
      dtype='object')
```

GENDER

In [39]:
```
gen=sns.countplot(x='Gender',data=df)
for bars in gen.containers:
    gen.bar_label(bars)
```



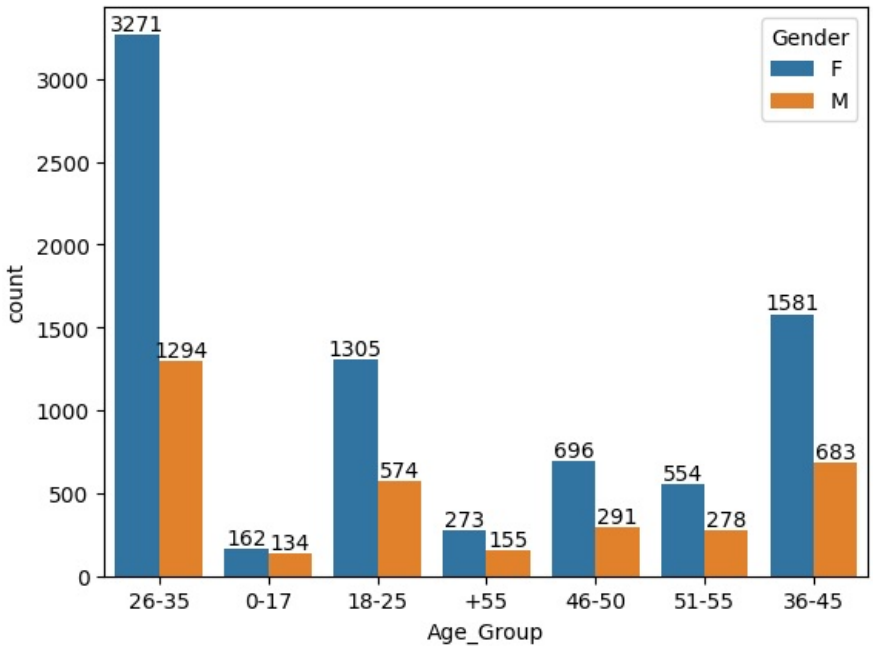
Number of orders of Females is higher compared to Male

AGE

In [40]:
```
df.columns
```

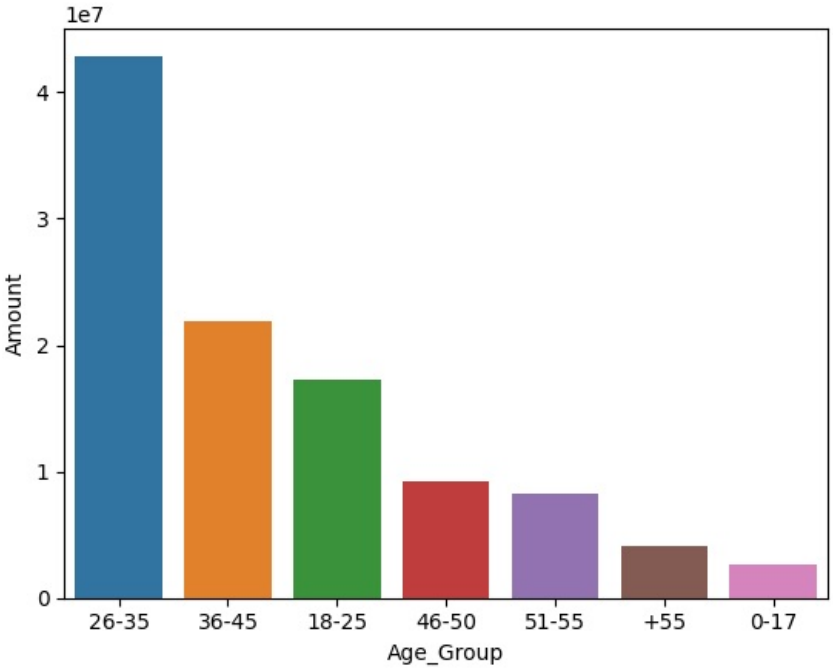
Out[40]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age', 'Marital_Status',
       'State', 'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount',
       'Age_Group'],
      dtype='object')
```

In [42]:
```
age=sns.countplot(data=df,x='Age_Group',hue='Gender')
for bars in age.containers:
    age.bar_label(bars)
```



In [49]:
```
#Amount vs Age_group by groupby
sales_age=df.groupby(['Age_Group'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False)
sns.barplot(x='Age_Group',y='Amount',data=sales_age)
```

Out[49]:
```
<Axes: xlabel='Age_Group', ylabel='Amount'>
```



From the above graph we can say purchasing power of age_group between 26-35 is greater than anyother group

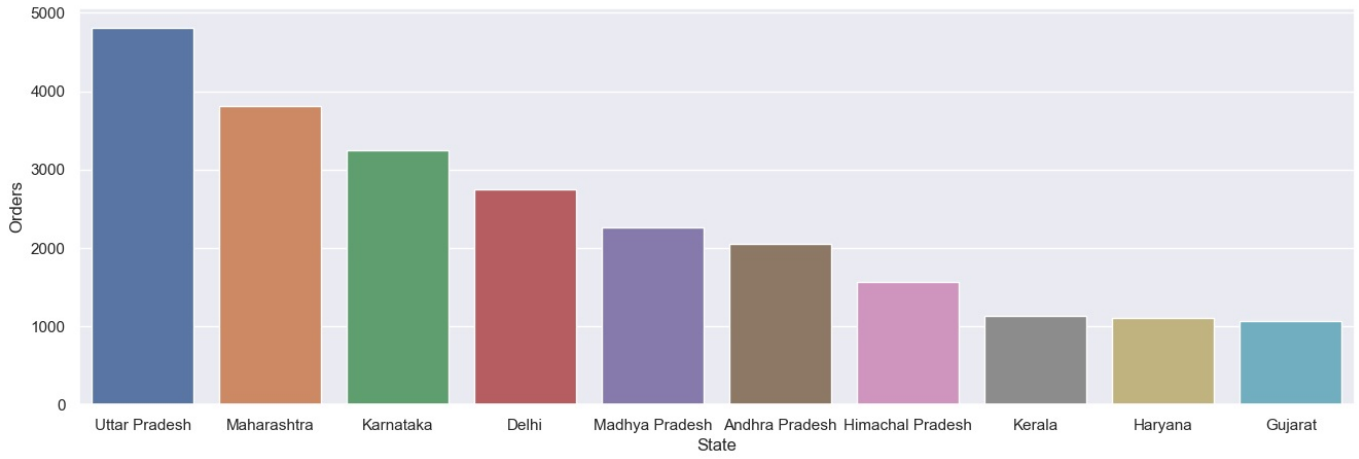
STATES

In [50]:
```
df.columns
```

Out[50]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age', 'Marital_Status',
       'State', 'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount',
       'Age_Group'],
      dtype='object')
```

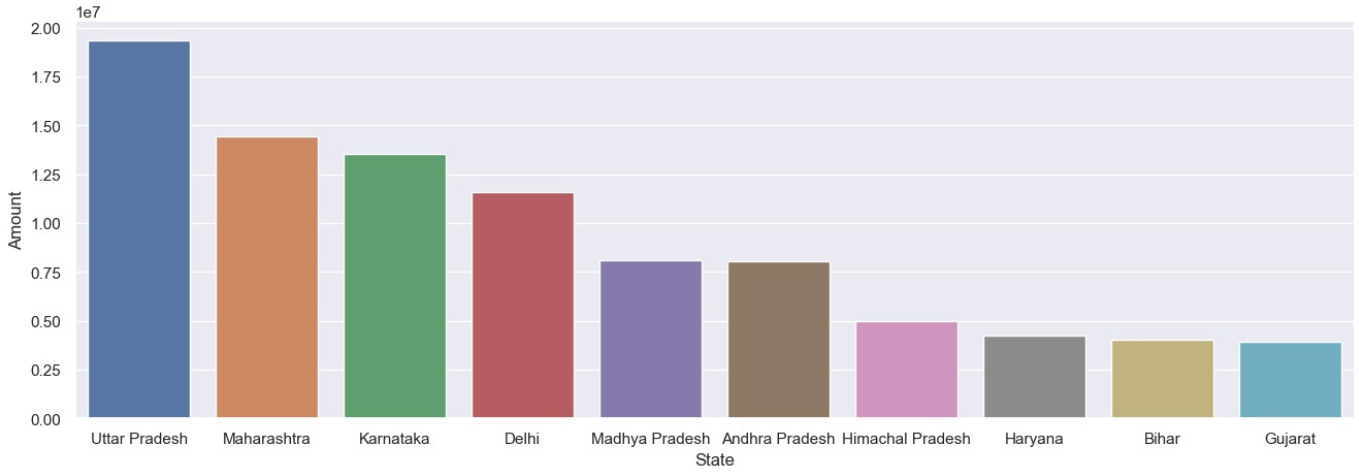
In [6]:
```
sales_state=df.groupby(['State'],as_index=False)['Orders'].sum().sort_values(by='Orders',ascending=False).head(
sns.set(rc={'figure.figsize':(16,5)})
sns.barplot(x='State',y='Orders',data=sales_state)
```

Out[6]:
```
<Axes: xlabel='State', ylabel='Orders'>
```



In [8]:
```
#Comparing State vs Amount
sales_state=df.groupby(['State'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False).head(
sns.set(rc={'figure.figsize':(16,5)})
sns.barplot(x='State',y='Amount',data=sales_state)
```

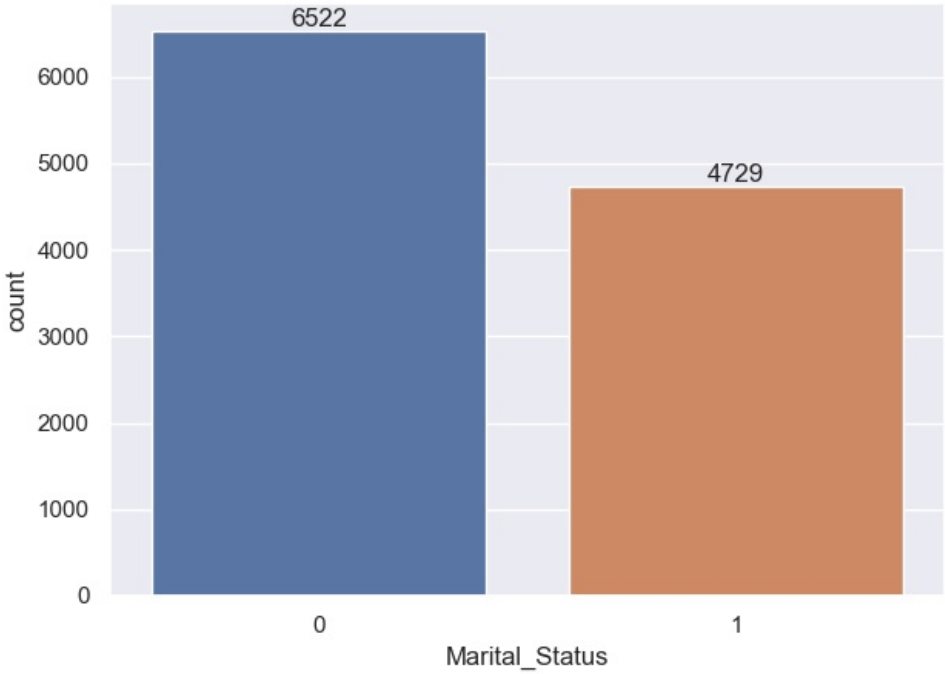
Out[8]:
```
<Axes: xlabel='State', ylabel='Amount'>
```



From above analysis Uttar Pradesh ,Maharashtra and Karnataka are among the highest in terms of Amount and Sales

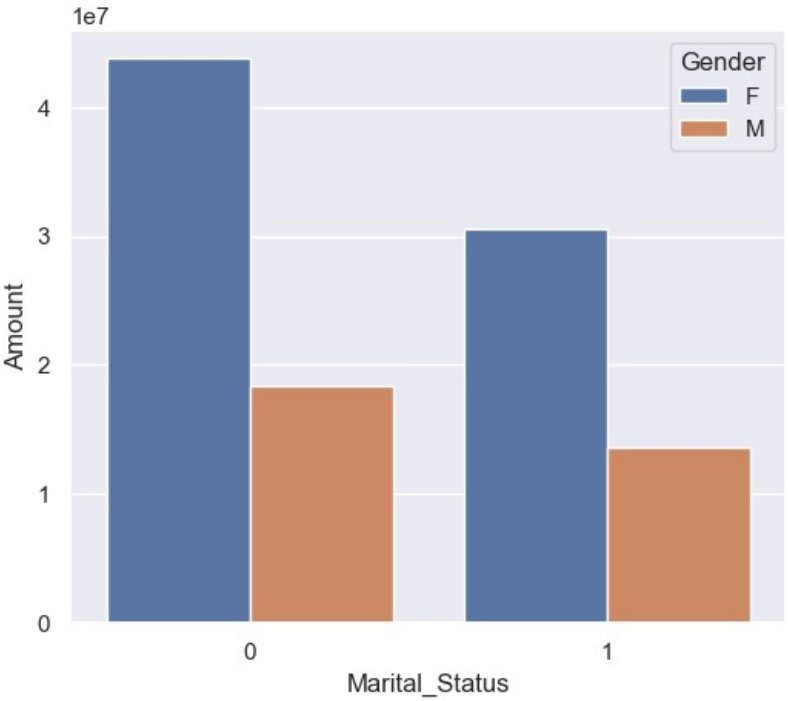
Marital Status

In [15]:
```
marital=sns.countplot(x='Marital_Status',data=df)
sns.set(rc={'figure.figsize':(7,5)})
for bars in marital.containers:
    marital.bar_label(bars)
```



In [19]:
```
#Marital Status wrt amount and orders
sales_status=df.groupby(['Marital_Status','Gender'],as_index=False)['Amount'].sum().sort_values(by='Amount',asc
sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(x='Marital_Status',y='Amount',hue='Gender',data=sales_status)
```

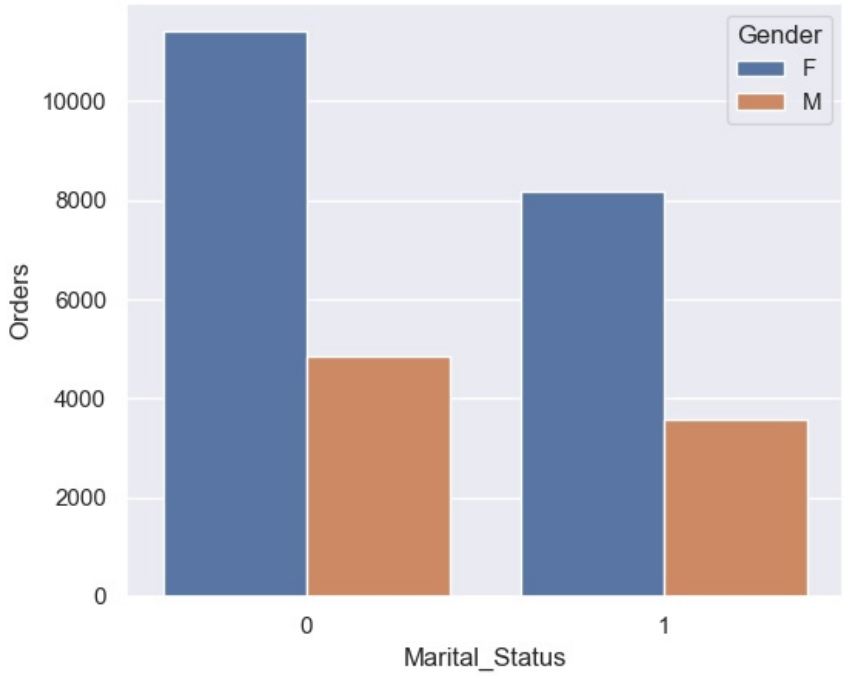
Out[19]: <Axes: xlabel='Marital_Status', ylabel='Amount'>



In [22]:
```
sales_status=df.groupby(['Marital_Status','Gender'],as_index=False)['Orders'].sum().sort_values(by='Orders',asc
sns.set(rc={'figure.figsize':(6,5)})

sns.barplot(x='Marital_Status',y='Orders',hue='Gender',data=sales_status)
```

Out[22]: <Axes: xlabel='Marital_Status', ylabel='Orders'>



- From the above graphs we can get insights thet Women(Married) have higher purchasing power *

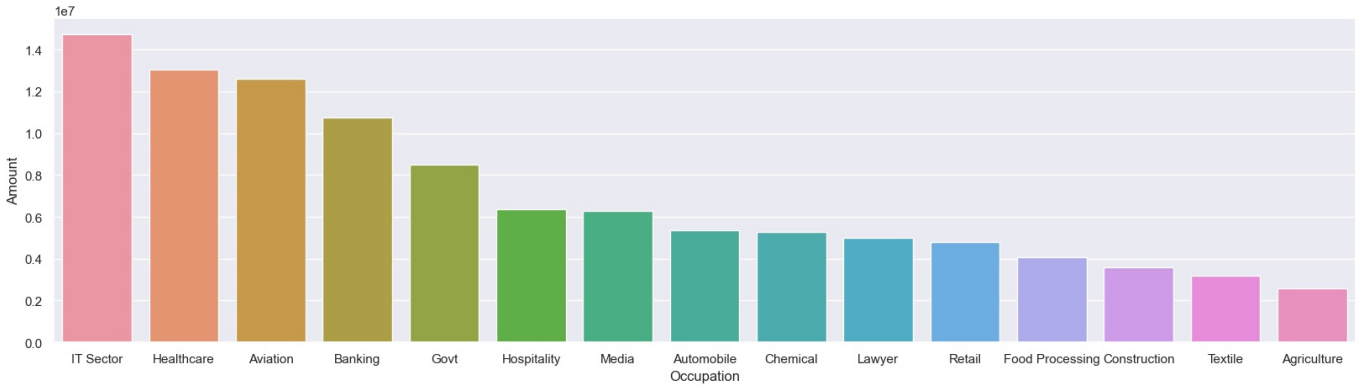
Occupation

In [23]:
```
df.columns
```

Out[23]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age', 'Marital_Status',
       'State', 'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount',
       'Status', 'Unnamed'],
      dtype='object')
```

In [24]:
```
sales_data=df.groupby(['Occupation'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False)
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(x='Occupation',y='Amount',data=sales_data)
```

Out[24]:
```
<Axes: xlabel='Occupation', ylabel='Amount'>
```

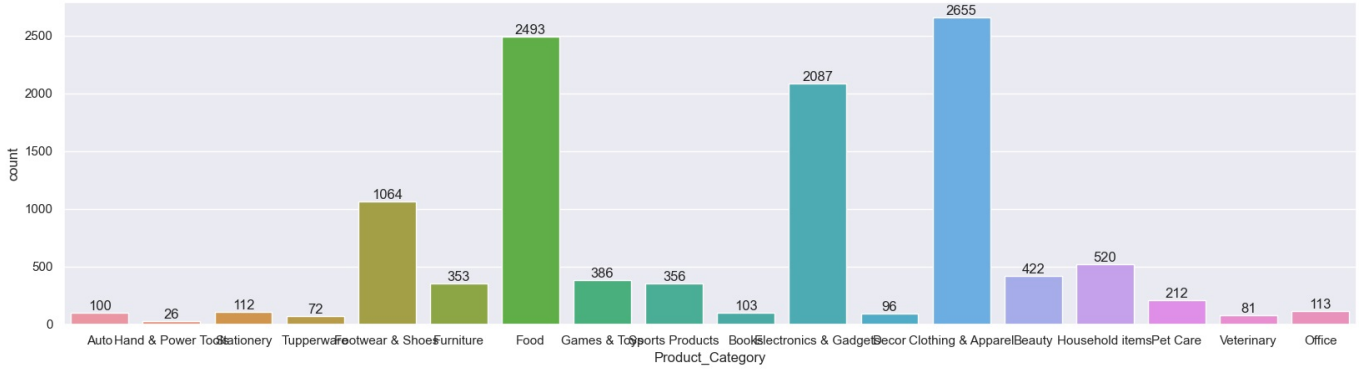


In [25]:
```
sns.set(rc={'figure.figsize':(20,5)})
sales_count=sns.countplot(x='Occupation',data=df)
for bars in sales_count.containers:
    sales_count.bar_label(bars)
```

From the above graphs most of the buyers are from IT Sector,Healthcare amd Aviation

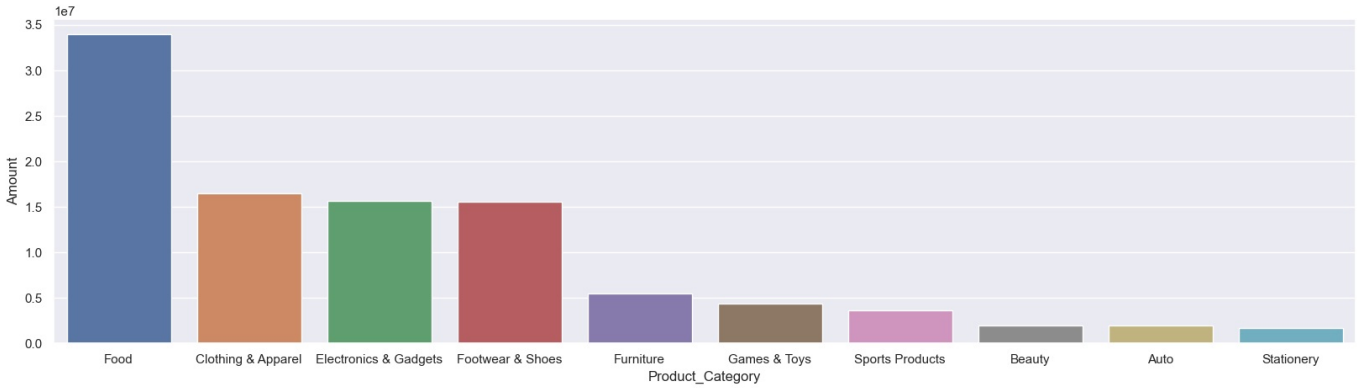
Product Category

In [28]:
```
sns.set(rc={'figure.figsize':(20,5)})
product=sns.countplot(x='Product_Category',data=df)
for bars in product.containers:
    product.bar_label(bars)
```



In [29]:
```
product=df.groupby(['Product_Category'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(x='Product_Category',y='Amount',data=product)
```

Out[29]:
```
<Axes: xlabel='Product_Category', ylabel='Amount'>
```



From the above graphs most of the products purchased is from Food Category

Conclusion:

Married women age group 26-35 yrs from UP, Maharastra and Karnataka working in IT, Healthcare and Aviation are more likely to buy products from Food, Clothing and Electronics category

In [ ]: