In [2]:
```
import numpy as np                #for numerical calculation
import pandas as pd               #for data loading
import matplotlib.pyplot as plt   #for ploting graph
%matplotlib inline
import seaborn as sns             #for visualization
```

In [3]:
```
df = pd.read_csv(r'C:\Users\ASUS\Downloads\EDA in Python\Python_Diwali_Sales_Analysis-main\Python_Diwali
```

```
# df as dataframe where we load data with use of pd, 1st copying location of data from downloads then
adding an name next to it with .csv
```

In [4]:
```
df
```

Out[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Ca
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	
...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	Ve
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	

11251 rows × 15 columns

In [5]:
```
df.shape
```

Out[5]:
```
(11251, 15)
```

In [6]:
```
df.head(10)
```

Out[6]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Categor
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Aut
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Aut
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Aut
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Aut
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Aut
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh	Northern	Food Processing	Aut
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh	Central	Lawyer	Aut
7	1002092	Shivangi	P00273442	F	55+	61	0	Maharashtra	Western	IT Sector	Aut
8	1003224	Kushal	P00205642	M	26-35	35	0	Uttar Pradesh	Central	Govt	Aut
9	1003650	Ginny	P00031142	F	26-35	26	1	Andhra Pradesh	Southern	Media	Aut

In [7]:
```
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
```

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

In [9]:
```
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
```

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

Out[10]:

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

11251 rows × 13 columns

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
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 [12]:
```
df.shape
```

Out[12]:
```
(11251, 13)
```

In [13]:
```
#dropna function drop null values
df.dropna(inplace=True) #inplace is for saving the changes
```

In [14]:
```
df.shape
```

Out[14]:
```
(11239, 13)
```

In [15]:
```
import pandas as pd
# lets creating an list
data_test=[['vaishali',28],['shailesh',],['tilak',23]]
# creat an pandas dataframe using list
df_test=pd.DataFrame(data_test,columns=['Name','Age'])
df_test
```

Out[15]:

	Name	Age
0	vaishali	28.0
1	shailesh	NaN
2	tilak	23.0

In [16]:
```
df_test.dropna(inplace=True) #beacuse of inplace function its possible to save the changes
```

In [17]:
```
df_test
```

Out[17]:

	Name	Age
0	vaishali	28.0
2	tilak	23.0

In [18]:
```
#change data type of amount
df['Amount']=df['Amount'].astype('int')
```

In [19]:
```
df['Amount'].dtype
```

Out[19]:
```
dtype('int32')
```

In [20]:
```
df.columns
```

Out[20]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [21]:
```
#for renaming any column
df.rename(columns={'Marital_Status':'shadi'})
```

Out[21]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	shadi	State	Zone	Occupation	Product_Category
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto
...	...	...	...	...	...	...	...	...	...	...	...
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	Office
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	Veterinary
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	Office
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	Office
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	Office

11239 rows × 13 columns

In [22]:
```
#its changes columns name as maritial status to shadi. its just for an example
```

In [23]:
```
#for renaming any column
df.rename(columns={'Marital_Status':'Marital_Status'})
```

Out[23]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Ca
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	
...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	Ve
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	

11239 rows × 13 columns

In [24]:
```
#describe function give brief description if data presesnt in dataset in the form of min max mean mode m
df.describe()
```

Out[24]:

	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 [25]:
```
#above data is describe base on numerical values of all columns, but we can select an selective columns
df[['Age','Orders','Amount']].describe()
```

Out[25]:

	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

Exploratory Data Analysis

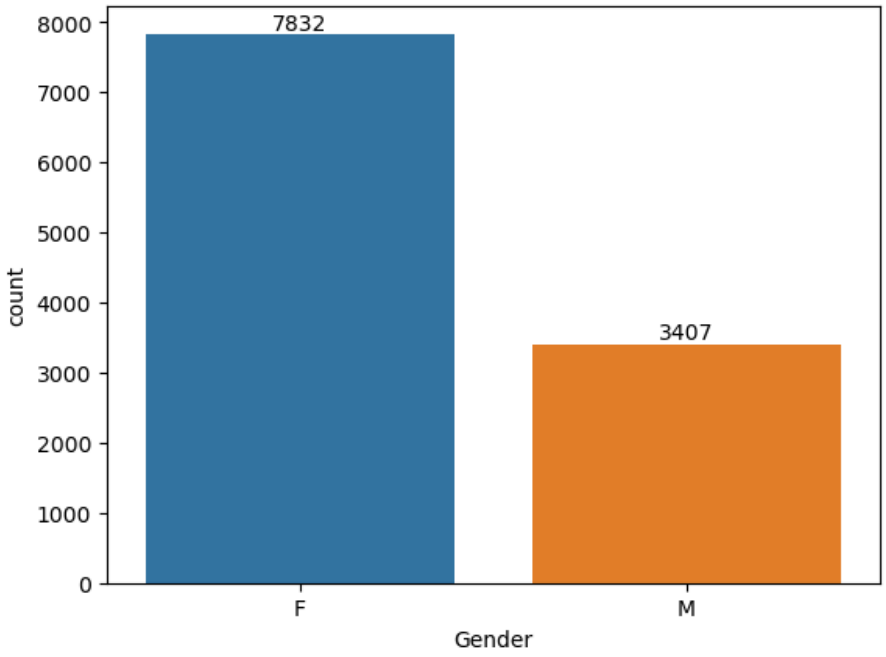
Above we did an data processing and data cleaning process here in EDA we doing an important column wise analysis by ploting an graph. by visualization its easy to differentiate by considering different state,age,product & occupation wise orders,purachsing rate,sales and demand so its easy for imporvement.

Gender

In [27]:
```
df.columns
```

Out[27]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [28]:
```
ax = sns.countplot(x= 'Gender',data=df) #by this we only get an bar
for bars in ax.containers:              #by this loop we get and amount on tap of the bar
    ax.bar_label(bars)
```



In [33]:
```
sales_gen= df.groupby(['Gender'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False
```

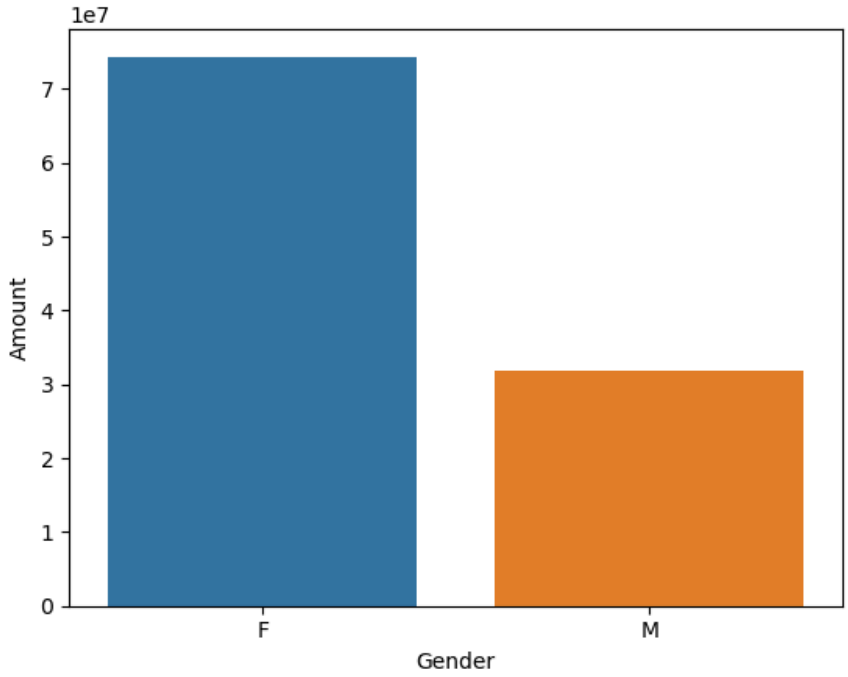
In [34]:
```
sales_gen
```

Out[34]:

	Gender	Amount
0	F	74335853
1	M	31913276

In [35]:
```
sales_gen= df.groupby(['Gender'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False
sns.barplot(x='Gender',y='Amount',data=sales_gen)
```

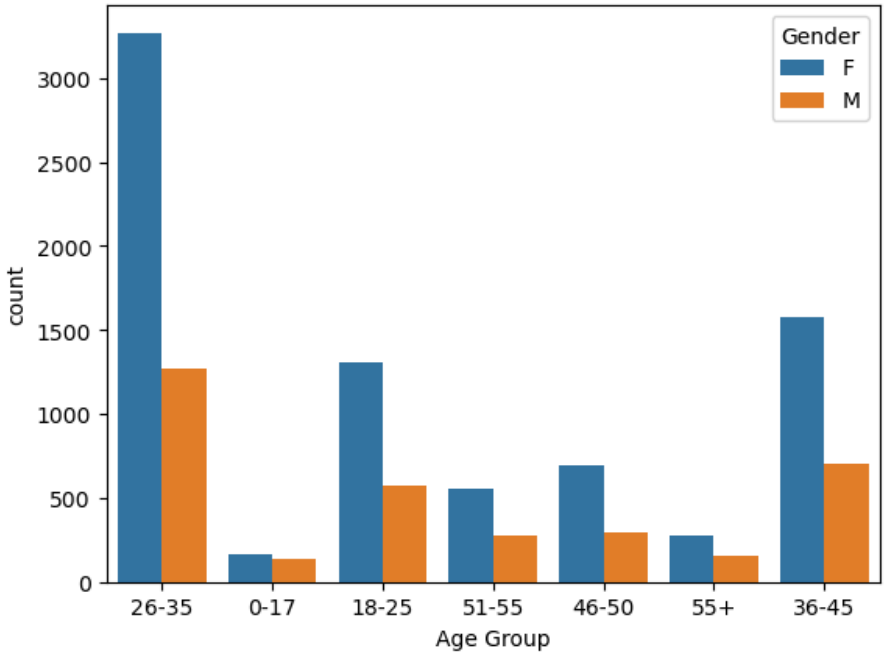
Out[35]: <Axes: xlabel='Gender', ylabel='Amount'>



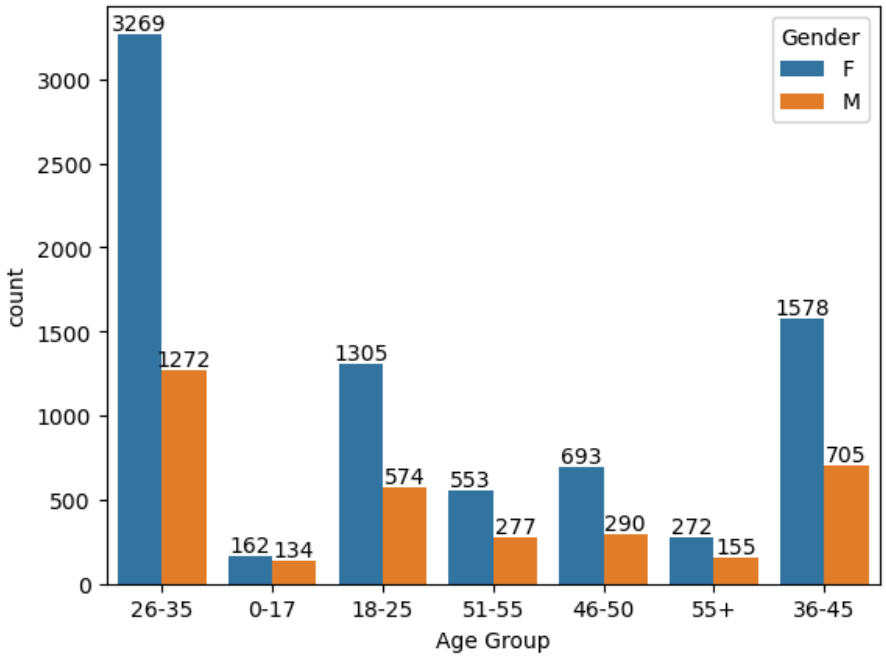
from above we analyz that purchasing rate of female is more than male.

Age

In [36]:
```
ax = sns.countplot(x= 'Age Group',data=df,hue='Gender') #by this we only get an bar without values
```



In [37]:
```
ax = sns.countplot(x= 'Age Group',data=df,hue='Gender')
for bars in ax.containers:              #by this loop we get and amount value on tap of the bar
    ax.bar_label(bars)
```



In [39]:
```
sales_Age= df.groupby(['Age Group'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=Fa
```

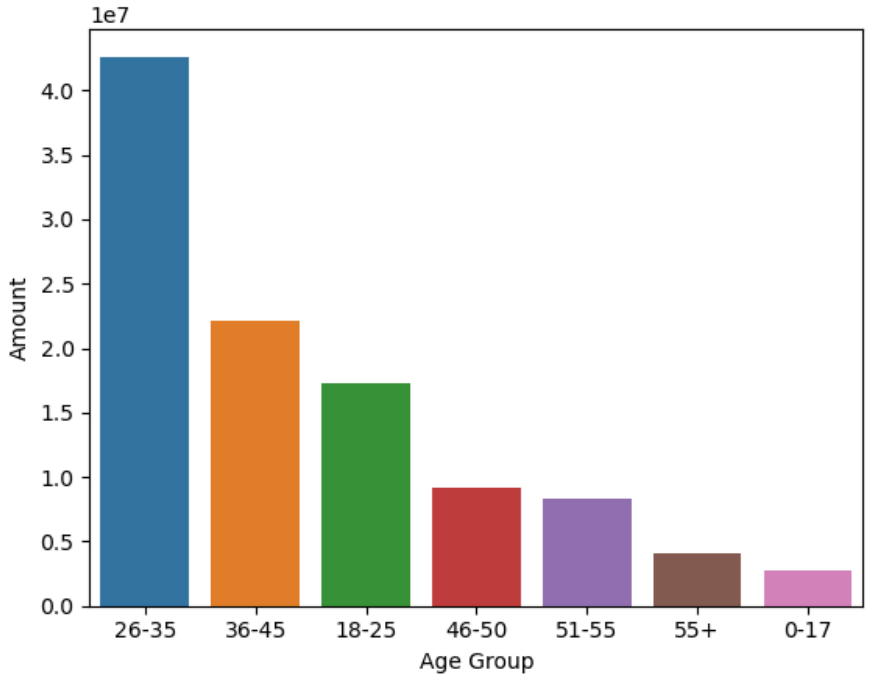
In [40]:
```
sales_Age
```

Out[40]:

	Age Group	Amount
2	26-35	42613442
3	36-45	22144994
1	18-25	17240732
4	46-50	9207844
5	51-55	8261477
6	55+	4080987
0	0-17	2699653

In [41]:
```
sales_Age= df.groupby(['Age Group'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=Fa
sns.barplot(x='Age Group',y='Amount',data=sales_Age)
```

Out[41]: <Axes: xlabel='Age Group', ylabel='Amount'>

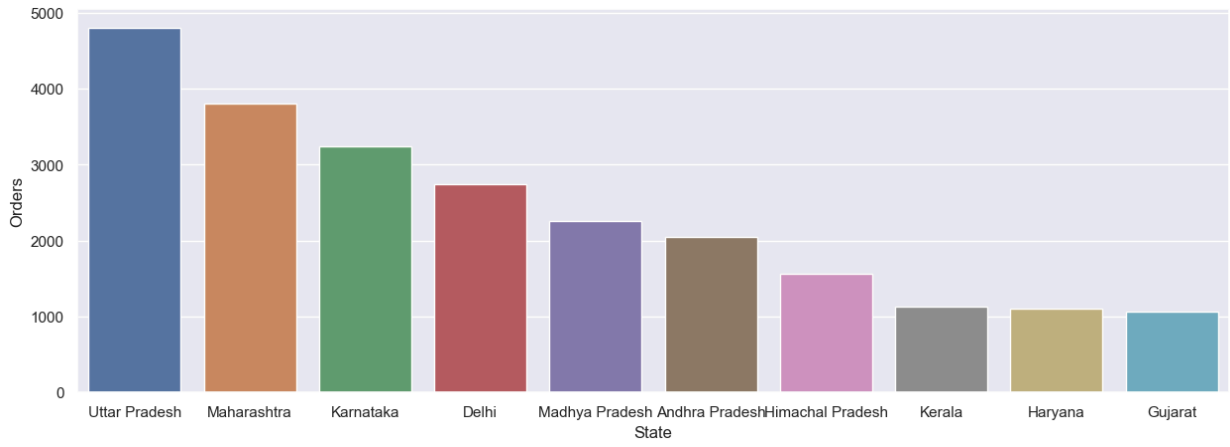


From above graph we can see that Most of buyer from age group 26-35 year female are most

State

In [54]:
```
sales_State= df.groupby(['State'],as_index=False)['Orders'].sum().sort_values(by='Orders',ascending=Fals
sns.set(rc={'figure.figsize':(15,5)})   # for providing space between an column that is size
sns.barplot(x='State',y='Orders',data=sales_State)
```

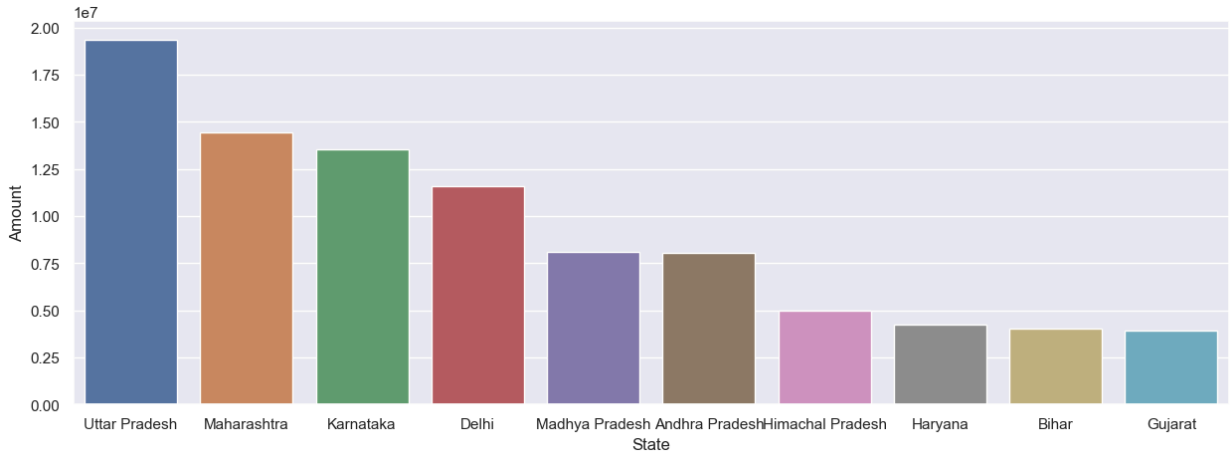
Out[54]: <Axes: xlabel='State', ylabel='Orders'>



Above we can see state wise orders. where we can find UP,Maharashtra, Karnataka,Delhi are the top most cities who have ordered.

In [55]:
```
sales_State= df.groupby(['State'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=Fals
sns.set(rc={'figure.figsize':(15,5)})   # for providing space between an column that is size
sns.barplot(x='State',y='Amount',data=sales_State)
```

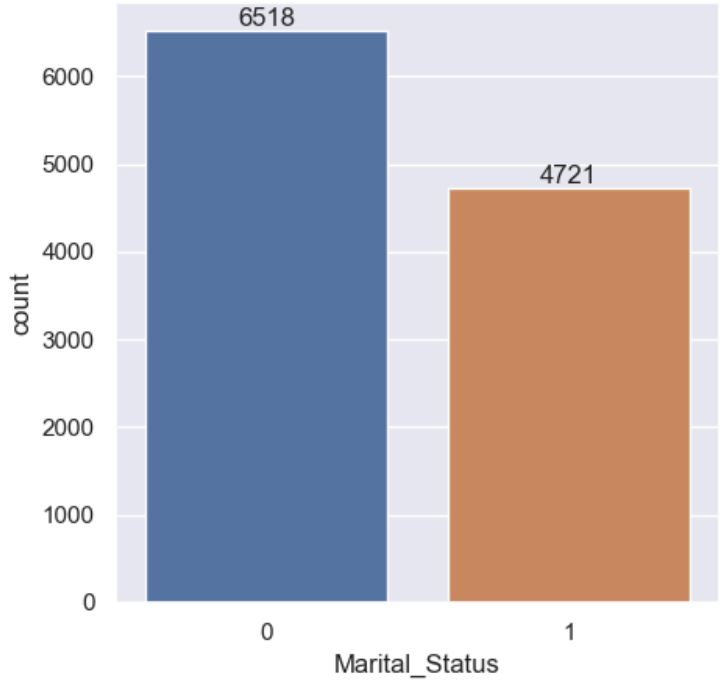
Out[55]: <Axes: xlabel='State', ylabel='Amount'>



we can see that most of purchasing orderswise as well amountwise states are up, maharashtra,karnataka, delhi etc, but we found at end amountwise states are change when we compare with orders graph.

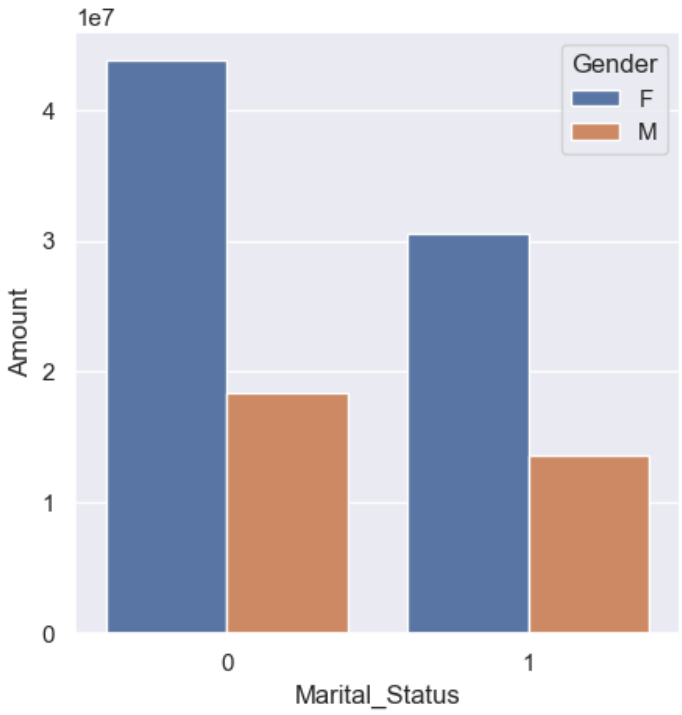
Marital_Status

In [64]:
```
ax = sns.countplot(x= 'Marital_Status',data=df)
sns.set(rc={'figure.figsize':(5,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```



In [69]:
```
sales_Marital_Status= df.groupby(['Marital_Status','Gender'],as_index=False)['Amount'].sum().sort_values
sns.set(rc={'figure.figsize':(5,5)})   # for providing space between an column that is size
sns.barplot(x='Marital_Status',y='Amount',data=sales_Marital_Status,hue='Gender')
```

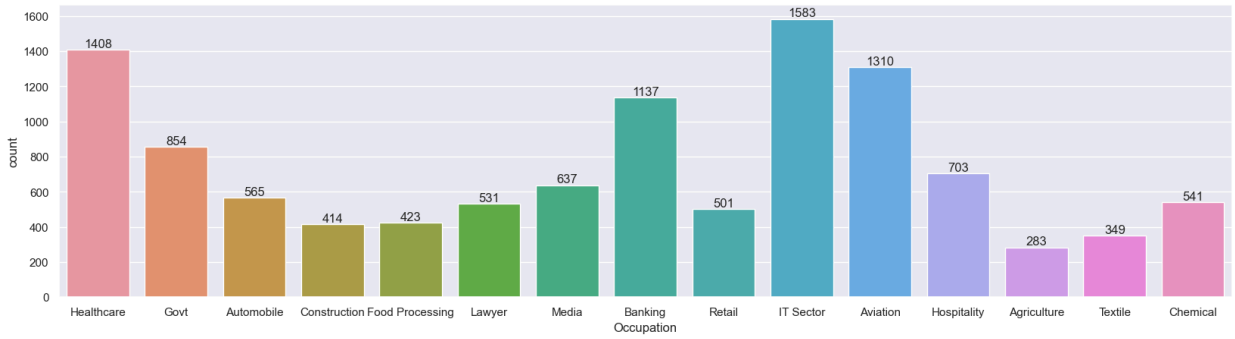
Out[69]: <Axes: xlabel='Marital_Status', ylabel='Amount'>



from above graph we can see purchasing power of married women is higher.

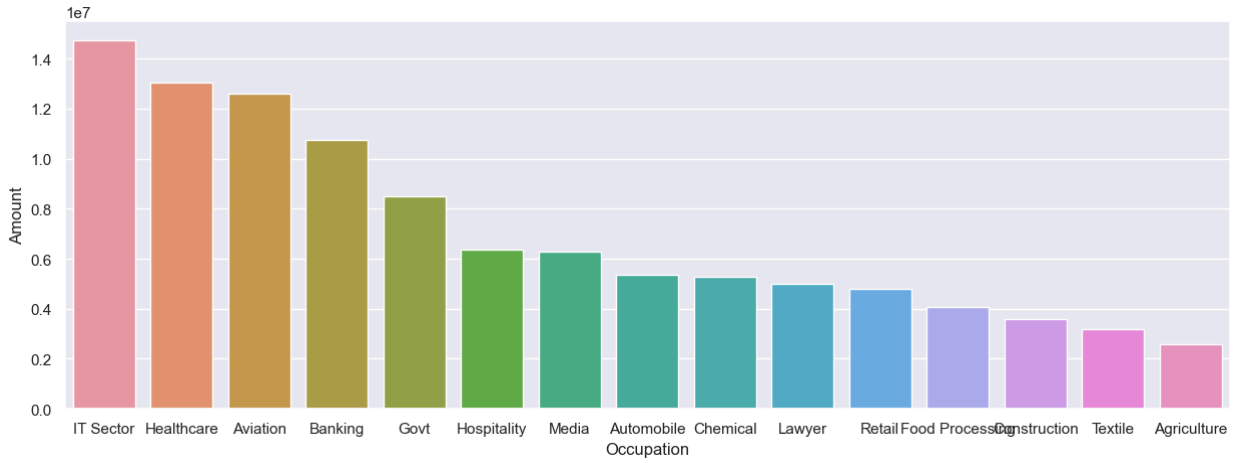
Occupation

In [71]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax=sns.countplot(data=df,x='Occupation')
for bars in ax.containers:
    ax.bar_label(bars)
```



In [72]:
```
sales_State= df.groupby(['Occupation'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending
sns.set(rc={'figure.figsize':(15,5)})   # for providing space between an column that is size
sns.barplot(x='Occupation',y='Amount',data=sales_State)
```

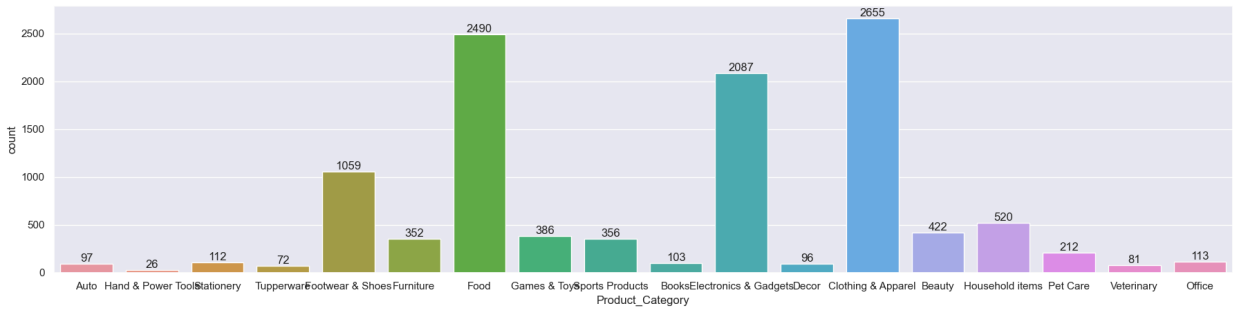
Out[72]: <Axes: xlabel='Occupation', ylabel='Amount'>



from above graph we can see purchasing power of IT Sector, Healthcare, Aviation and Banking is highest as compare to others.

Product_Category

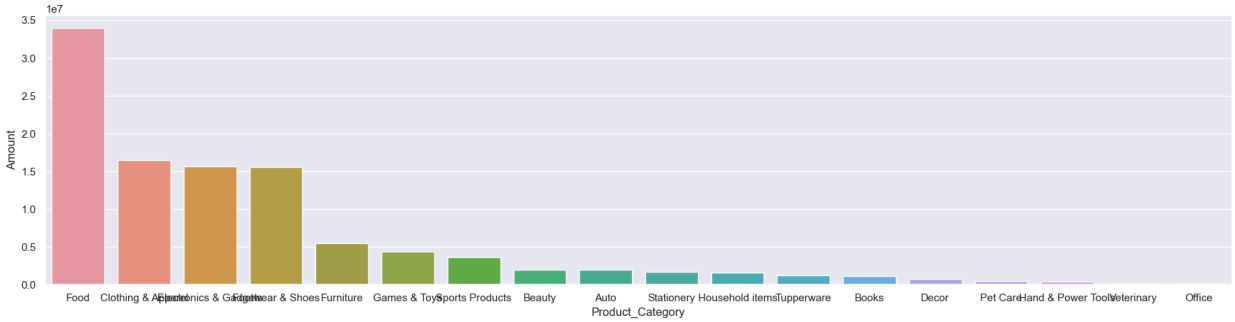
In [75]:
```
sns.set(rc={'figure.figsize':(22,5)})
ax = sns.countplot(data = df, x= 'Product_Category')
for bars in ax.containers:
    ax.bar_label(bars)
```



From Above graph we can see that most of product order for clothing & appearances, food and Electronic & Gadgets

In [82]:
```
sales_State= df.groupby(['Product_Category'],as_index=False)['Amount'].sum().sort_values(by='Amount',asc
sns.set(rc={'figure.figsize':(22,5)})   # for providing space between an column that is size
sns.barplot(x='Product_Category',y='Amount',data=sales_State)
```

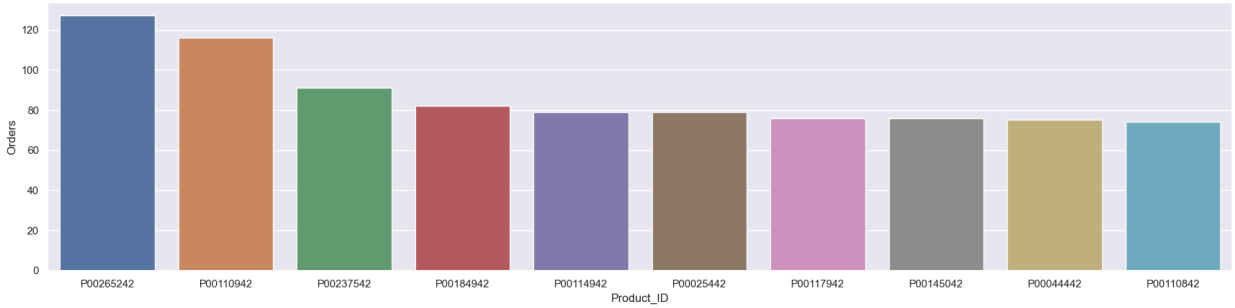
Out[82]: <Axes: xlabel='Product_Category', ylabel='Amount'>



From Above graph we can see that most of amount spend on food,clothing & appearances and Electronic & Gadgets.

In [83]:
```
sales_State= df.groupby(['Product_ID'],as_index=False)['Orders'].sum().sort_values(by='Orders',ascending
sns.set(rc={'figure.figsize':(22,5)})   # for providing space between an column that is size
sns.barplot(x='Product_ID',y='Orders',data=sales_State)
```

Out[83]: <Axes: xlabel='Product_ID', ylabel='Orders'>



Conclusion:

so from above analysis we conclude that age group of 26-35 years married women belonging from UP,Maharashtra,Karnataka, Delhi who are working for IT Sector,Healthcar and Aviation having an highest purchasing rate for products like Clothes and appearance ,Food and Electronic & Gadgest.