In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [9]:
```
df = pd.read_csv('Diwali Sales Data.csv',encoding = 'unicode_escape')
df.shape
```

Out[9]: (11251, 15)

In [12]:
```
df.head()
```

Out[12]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare
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

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5195 entries, 0 to 5194
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           5195 non-null   int64
 1   Cust_name         5195 non-null   object
 2   Product_ID        5195 non-null   object
 3   Gender            5195 non-null   object
 4   Age Group         5195 non-null   object
 5   Age               5195 non-null   int64
 6   Marital_Status    5195 non-null   int64
 7   State             5195 non-null   object
 8   Zone              5195 non-null   object
 9   Occupation        5195 non-null   object
 10  Product_Category  5195 non-null   object
 11  Orders            5195 non-null   int64
 12  Amount            5190 non-null   float64
 13  status            0 non-null      float64
 14  unnamed1          0 non-null      float64
dtypes: float64(3), int64(4), object(8)
memory usage: 608.9+ KB
```

In [6]:
```
#drop unrelated/blank columns
df.drop(['status', 'unnamed1'], axis=1, inplace=True)
```

In [7]:
```
#check for null values
pd.isnull(df).sum()
```

Out[7]:
```
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              5
dtype: int64
```

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

Out[8]:
```
(5195, 13)
```

In [9]:
```
# drop null values
df.dropna(inplace=True)
```

In [10]:
```
df.shape
```

Out[10]:
```
(5190, 13)
```

In [11]:
```
# initialize list of lists
data_test = [['madhav', 11], ['Gopi', 15], ['Keshav', ], ['Lalita', 16]]

# Create the pandas DataFrame using list
df_test = pd.DataFrame(data_test, columns=['Name', 'Age'])

df_test
```

Out[11]:

	Name	Age
0	madhav	11.0
1	Gopi	15.0
2	Keshav	NaN
3	Lalita	16.0

In [12]:
```
df_test.dropna()
```

Out[12]:

	Name	Age
0	madhav	11.0
1	Gopi	15.0
3	Lalita	16.0

In [13]:
```
df_test
```

Out[13]:

	Name	Age
0	madhav	11.0
1	Gopi	15.0
2	Keshav	NaN
3	Lalita	16.0

both are same thing

df_test.dropna(inplace=True)

```
df_test = df_test.dropna()
```

In [14]:
```
# change data type
df['Amount'] = df['Amount'].astype('int')
```

In [15]:
```
df['Amount'].dtypes
```

Out[15]:
```
dtype('int32')
```

In [16]:
```
df.columns
```

Out[16]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [17]:
```
#rename column
df.rename(columns= {'Marital_Status':'Shaadi'})
```

Out[17]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Shaadi	State	Zone	Occupation	Produ
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	
...	...	...	...	...	...	...	...	...	...	...	
5188	1000611	Yogesh	P00233942	F	36-45	37	0	Rajasthan	Northern	IT Sector	
5190	1003661	Nandita	P00122442	F	36-45	38	1	Delhi	Central	Banking	Footv
5191	1004447	Ajay	P00281942	F	46-50	47	0	Uttarakhand	Central	Aviation	
5192	1005998	Nitant	P00318742	F	18-25	21	0	Himachal Pradesh	Northern	IT Sector	
5193	1002428	Lori	P00118542	F	46-50	50	0	Jharkhand	Eastern	Govt	

5190 rows × 13 columns

In [18]:
```
# describe() method returns description of the data in the DataFrame (i.e. count, mean, std, etc)
df.describe()
```

Out[18]:

	User_ID	Age	Marital_Status	Orders	Amount
count	5.190000e+03	5190.000000	5190.000000	5190.000000	5190.000000
mean	1.002999e+06	35.724277	0.414644	2.473410	14065.260116
std	1.704431e+03	12.768756	0.492708	1.111775	3797.154500
min	1.000003e+06	12.000000	0.000000	1.000000	8630.000000
25%	1.001501e+06	27.000000	0.000000	1.000000	10792.250000
50%	1.003064e+06	33.000000	0.000000	2.000000	13158.000000
75%	1.004412e+06	44.000000	1.000000	3.000000	16516.750000
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000

In [19]:
```
# use describe() for specific columns
df[['Age', 'Orders', 'Amount']].describe()
```

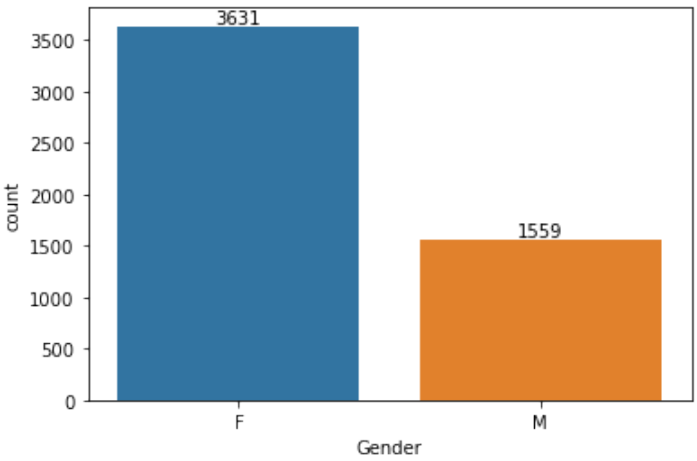
Out[19]:

	Age	Orders	Amount
count	5190.000000	5190.000000	5190.000000
mean	35.724277	2.473410	14065.260116
std	12.768756	1.111775	3797.154500
min	12.000000	1.000000	8630.000000
25%	27.000000	1.000000	10792.250000
50%	33.000000	2.000000	13158.000000
75%	44.000000	3.000000	16516.750000
max	92.000000	4.000000	23952.000000

Exploratory Data Analysis

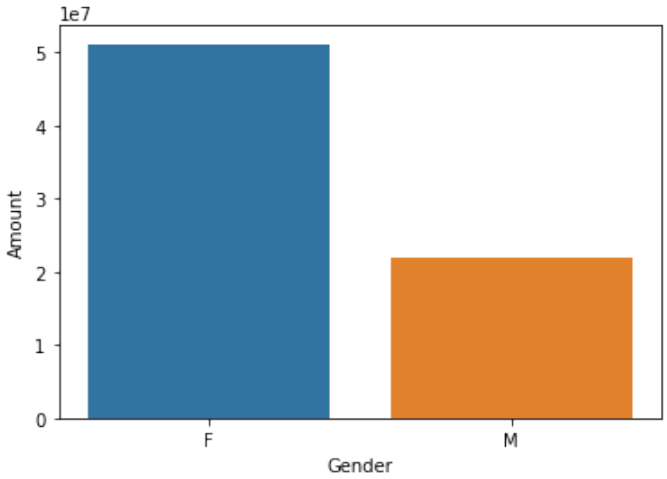
Gender

In [20]:
```
ax = sns.countplot(x = 'Gender',data = df)

for bars in ax.containers:
    ax.bar_label(bars)
```



In [21]:
```
sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=Fals

sns.barplot(x = 'Gender',y= 'Amount' ,data = sales_gen)
```

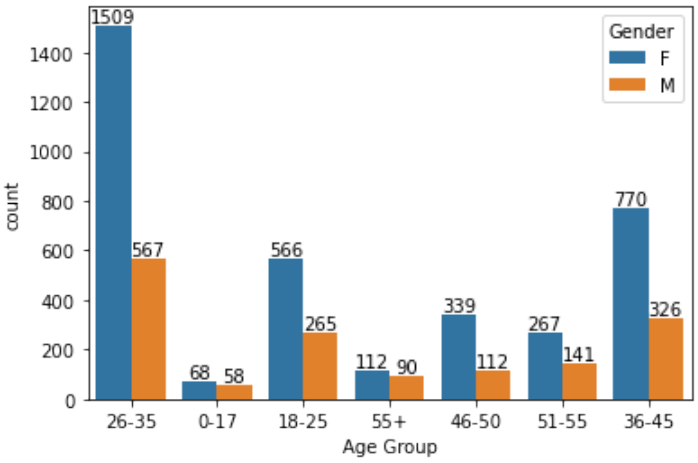
Out[21]:
```
<AxesSubplot:xlabel='Gender', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men

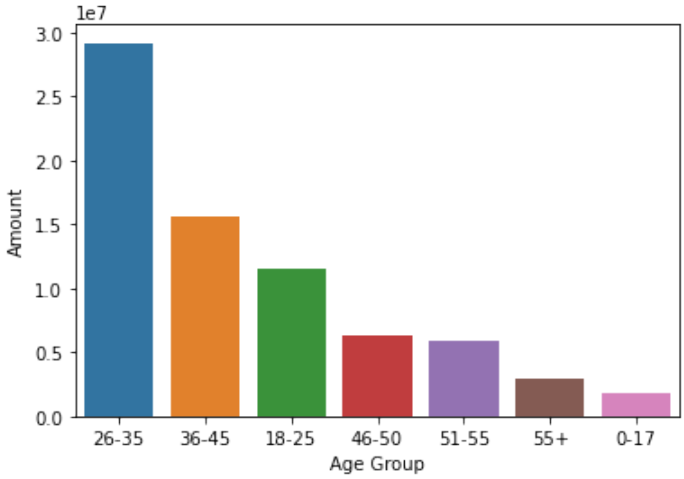
Age

In [22]:
```
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

for bars in ax.containers:
    ax.bar_label(bars)
```



In [23]:
```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=F

sns.barplot(x = 'Age Group',y= 'Amount' ,data = sales_age)
```

Out[23]:
```
<AxesSubplot:xlabel='Age Group', ylabel='Amount'>
```

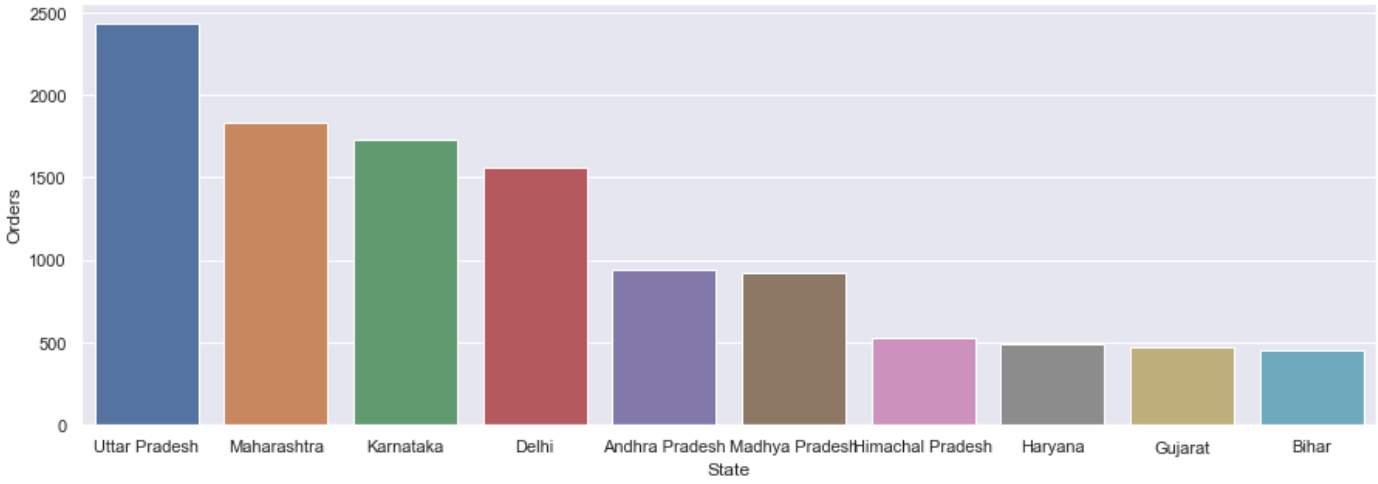


From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

State

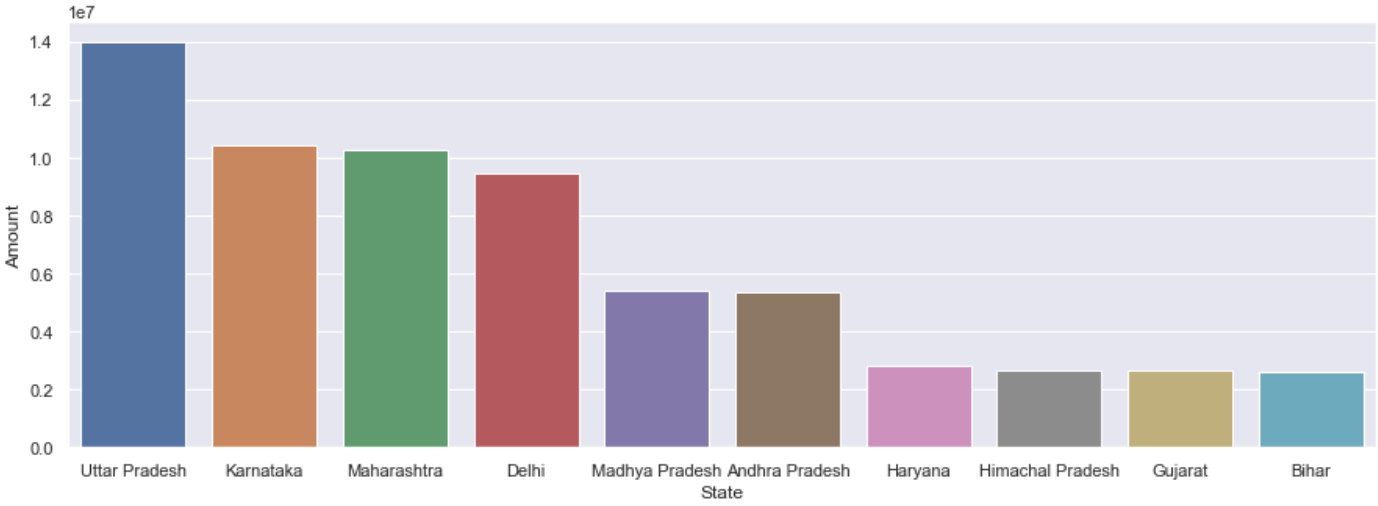
In [24]:
```
# total number of orders from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=Fal

sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

Out[24]:
```
<AxesSubplot:xlabel='State', ylabel='Orders'>
```



In [25]:
```
# total amount/sales from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=Fal

sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

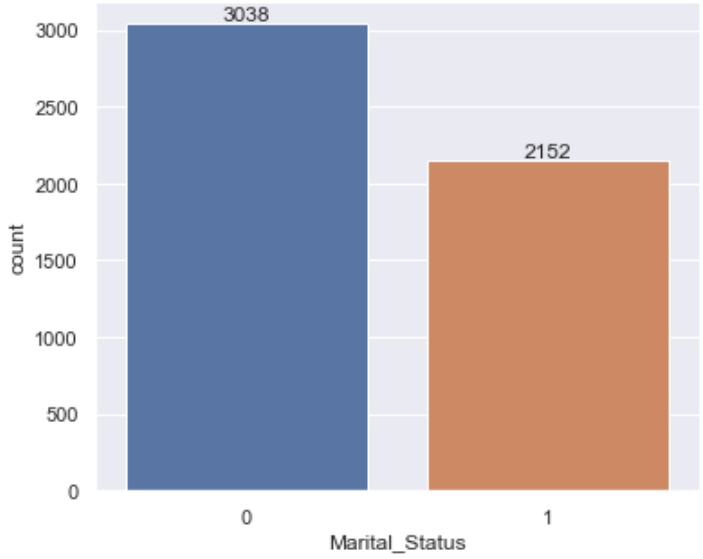
Out[25]: <AxesSubplot:xlabel='State', ylabel='Amount'>



From above graphs we can see that unexpectedly most of the orders are from Uttar Pradesh, Maharashtra and Karnataka respectively but total sales/amount is from UP, Karnataka and then Maharashtra

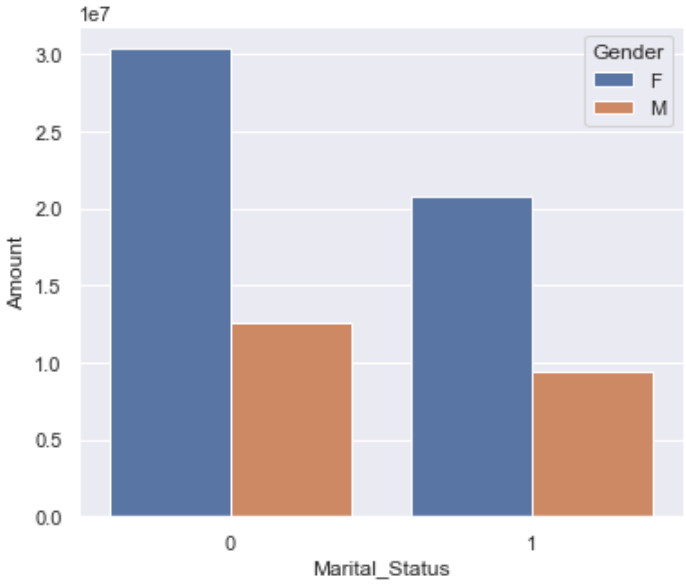
Marital Status

In [31]:
```
ax = sns.countplot(data = df, x = 'Marital_Status')

for bars in ax.containers:
    ax.bar_label(bars)
```



In [71]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].sum().sort_values(by='Amo

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

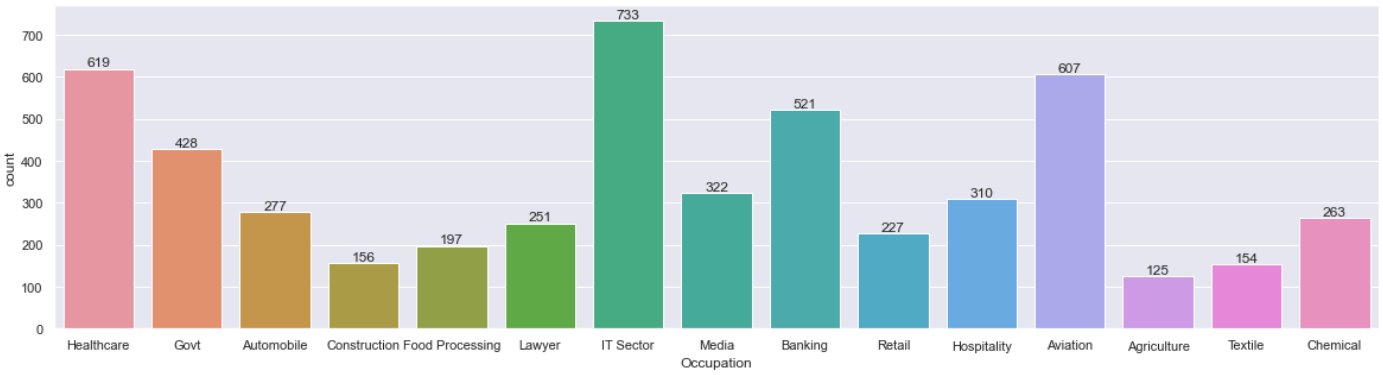
Out[71]: <AxesSubplot:xlabel='Marital_Status', ylabel='Amount'>



From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

Occupation

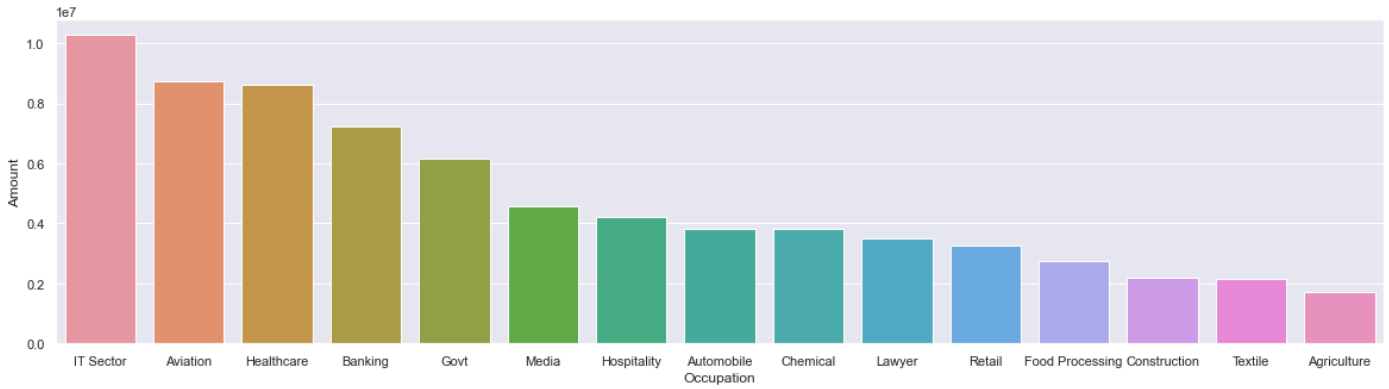
In [34]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

for bars in ax.containers:
    ax.bar_label(bars)
```



In [36]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascendir
```

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

Out[36]: <AxesSubplot:xlabel='Occupation', ylabel='Amount'>



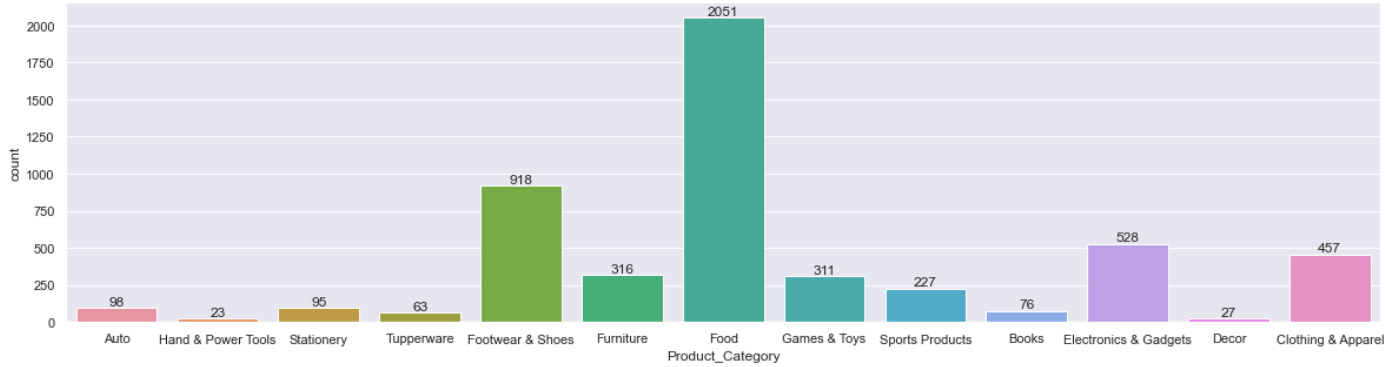
From above graphs we can see that most of the buyers are working in IT, Aviation and Healthcare sector

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

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

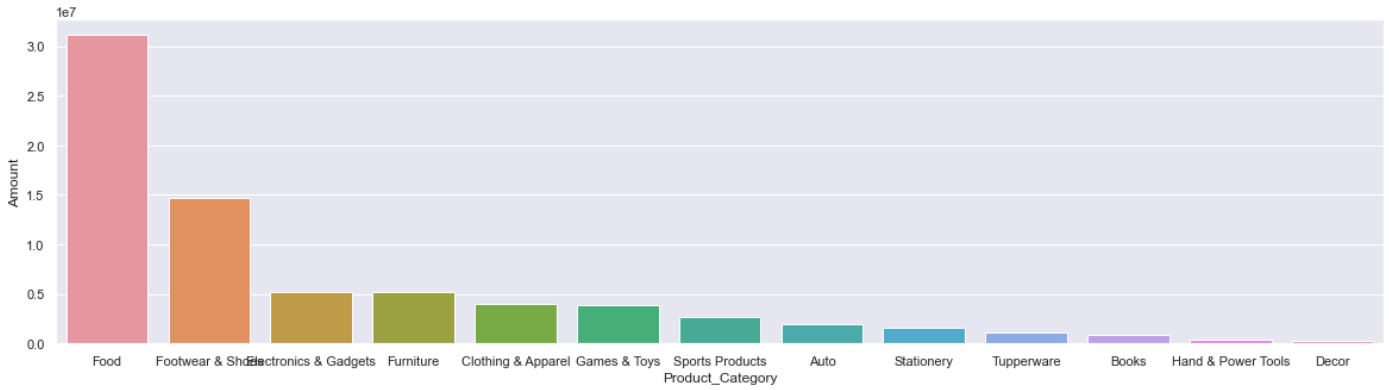
Product Category

In [42]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Product_Category')

for bars in ax.containers:
    ax.bar_label(bars)
```



In [43]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().sort_values(by='Amount', as

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

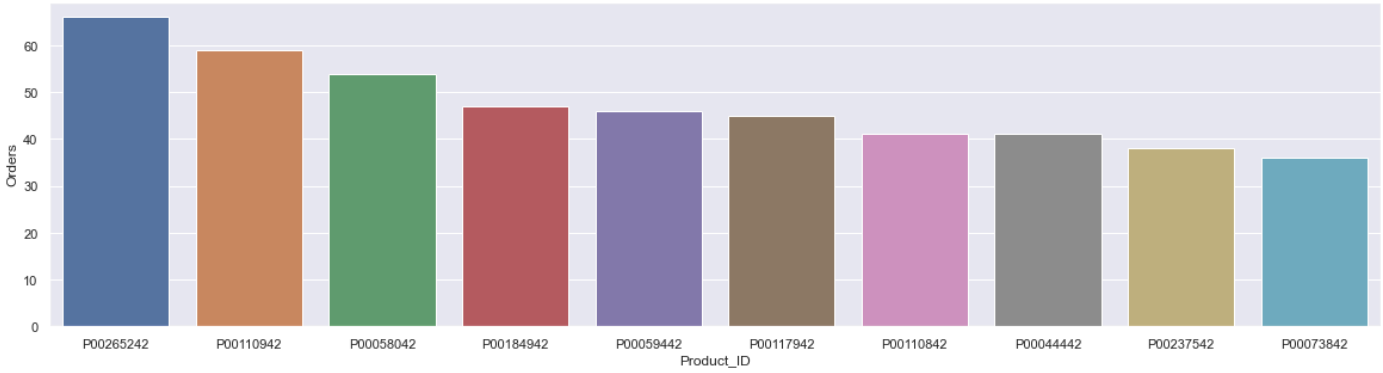
Out[43]: <AxesSubplot:xlabel='Product_Category', ylabel='Amount'>



From above graphs we can see that most of the sold products are from Food, Footwear and Electronics category

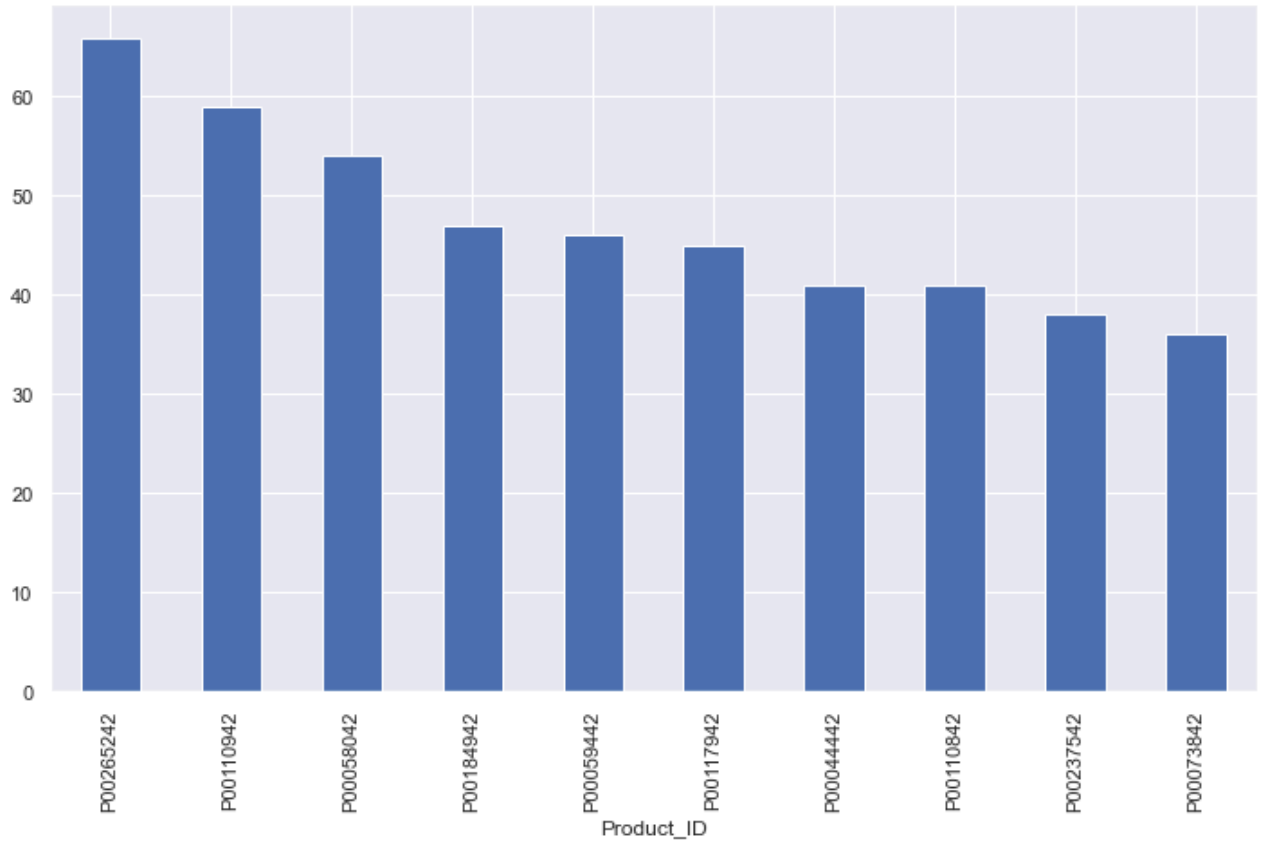
In [46]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascendir

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

Out[46]: <AxesSubplot:xlabel='Product_ID', ylabel='Orders'>



In [67]:
```
# top 10 most sold products (same thing as above)

fig1, ax1 = plt.subplots(figsize=(12,7))
df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='bar')
```

Out[67]: <AxesSubplot:xlabel='Product_ID'>



Conclusion:

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