Import python libraries

In [43]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # visualizing data
%matplotlib inline
import seaborn as sns
```

In [44]:
```
pwd
```
Out[44]:
```
'C:\\Users\\hp\\Desktop\\pythonproject\\Exploratory data analysis'
```

Import csv file

In [45]:
```
df = pd.read_csv('Diwali Sales Data.csv', encoding='unicode_escape')
```

Data Cleaning

Viewing top 5 rows in dataframe

In [46]:
```
df.head()
```
Out[46]:

	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

In [47]:
```
df.shape
```
Out[47]:
```
(11251, 15)
```

In [48]:
```
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
```

Deleting the null columns present in the dataframe

In [49]:
```
df.drop(['Status','unnamed1'], axis=1, inplace=True)
```

Checking if deleted columns are still there in the given data

In [50]:
```
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
```

Check for null rows in the given data

In [51]:
```
df.isnull().sum()
```
Out[51]:
```
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
```

We can infer from here that we have null values in amount column which cannot be replaced by the corresponding mean/median of the data. So, we will drop the rows having null values in the amount column.

Dropping the null values

In [52]:
```
df.dropna(inplace=True)
```

In [53]:
```
df.isnull().sum()
```
Out[53]:
```
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
```

We don't have any null values in our data now.

Changing data type

In [54]:
```
df['Amount'] = df['Amount'].astype('int')
df['Amount'].dtype
```
Out[54]:
```
dtype('int32')
```

Checking the data types of all columns

In [55]:
```
df.dtypes
```
Out[55]:
```
User_ID              int64
Cust_name           object
Product_ID          object
Gender              object
Age Group           object
Age                  int64
Marital_Status       int64
State               object
Zone                object
Occupation          object
Product_Category    object
Orders               int64
Amount               int32
dtype: object
```

Description of data

In [56]:
```
df[['Age','Orders','Amount']].describe()
```
Out[56]:

	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

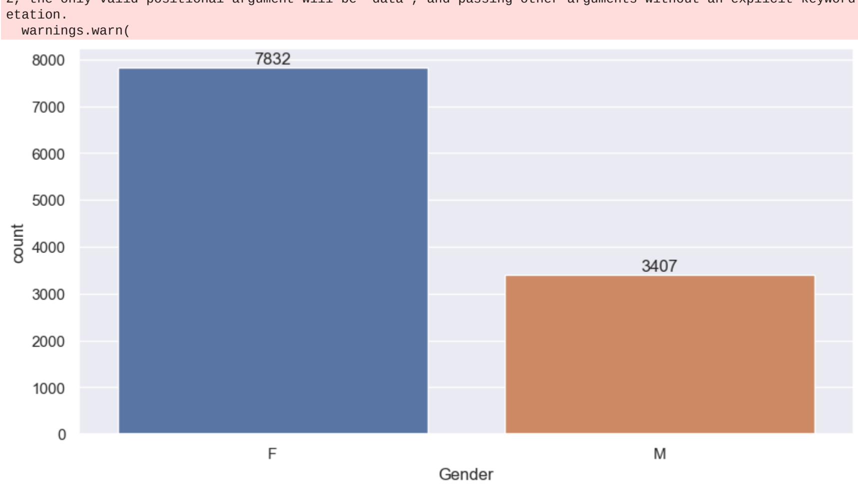
Exploratory Data Analysis

Gender column

In [57]:
```
df.columns
```
Out[57]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

Plotting a bar plot for gender and its count

In [58]:
```
ax = sns.countplot(df['Gender'])
sns.set(rc={'figure.figsize':(10,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```
```
C:\Users\hp\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

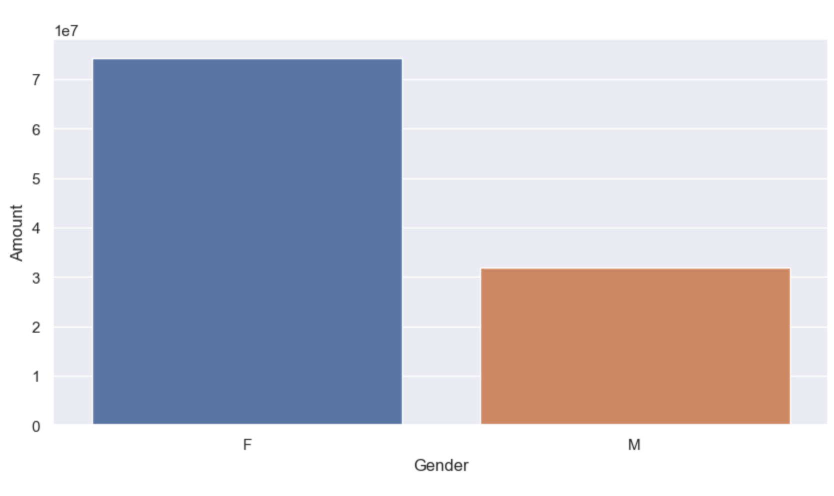


From this graph, we can infer that female customers have placed more orders than Males

In [59]:
```
sales_gen = pd.DataFrame(df.groupby(['Gender'])['Amount'].sum().sort_values(ascending =False))
sales_gen= sales_gen.reset_index()
sales_gen
```
Out[59]:

	Gender	Amount
0	F	74335853
1	M	31913276

In [60]:
```
# sales_gen.plot(kind='bar', x='Gender',  rot=True)
sns.barplot(x='Gender', y='Amount', data=sales_gen)
```
Out[60]:
```
<AxesSubplot:xlabel='Gender', ylabel='Amount'>
```

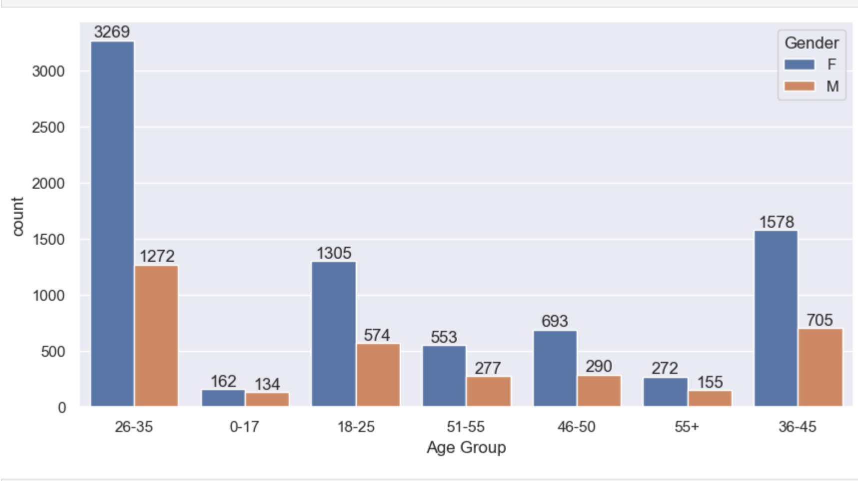


From the above 2 graphs, we can infer that the most of the buyers are females and even the purchasing power of females are greater than males

Age

In [61]:
```
df.columns
```
Out[61]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

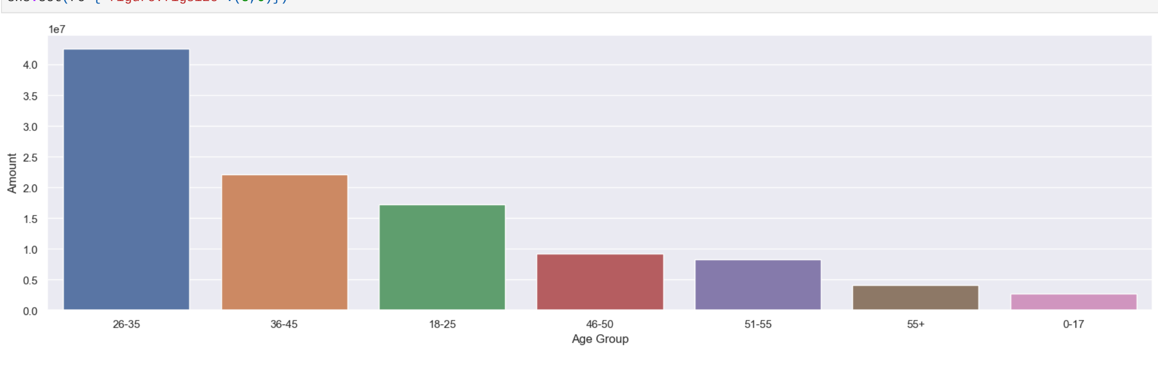
In [62]:
```
ax= sns.countplot(x='Age Group', data=df, hue='Gender')
sns.set(rc={'figure.figsize':(10,6)})
for bars in ax.containers:
    ax.bar_label(bars)
```



In [63]:
```
sales_age = pd.DataFrame(df.groupby(['Age Group'])['Amount'].sum().sort_values(ascending =False))
sales_age = sales_age.reset_index()
sales_age
```
Out[63]:

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

In [84]:
```
sns.barplot(x='Age Group', y='Amount', data=sales_age)
sns.set(rc={'figure.figsize':(8,6)})
```



From above 2 graphs, most of the buyers and maximum purchase is done by females of age-group 26-35

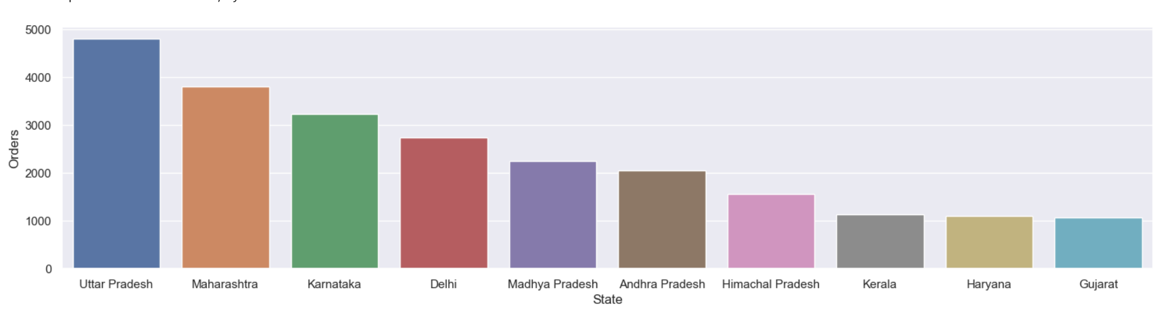
State

Total number of orders from top 10 states

In [65]:
```
df.columns
```
Out[65]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [68]:
```
State_orders = pd.DataFrame(df.groupby(['State'])['Orders'].sum().sort_values(ascending=False).head(10))
State_orders= State_orders.reset_index()
```

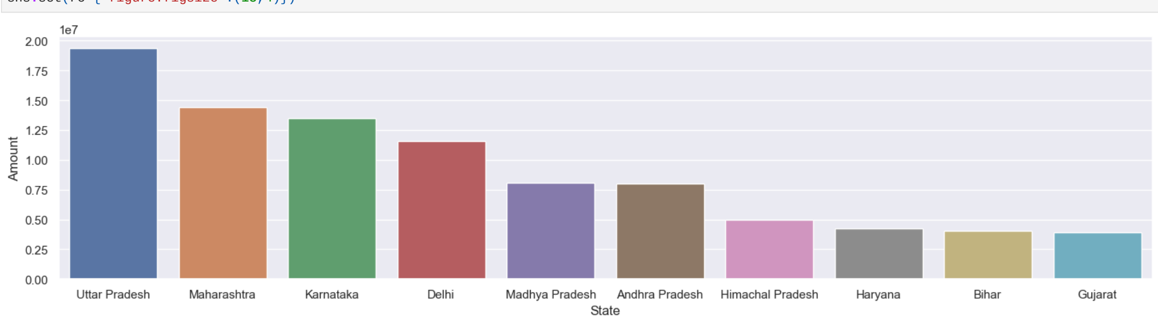
In [69]:
```
sns.set(rc={'figure.figsize':(18,4)})
sns.barplot(x='State', y='Orders', data=State_orders)
```
Out[69]:
```
<AxesSubplot:xlabel='State', ylabel='Orders'>
```



Total amount/sales from top 10 states

In [70]:
```
State_sales = pd.DataFrame(df.groupby(['State'])['Amount'].sum().sort_values(ascending=False).head(10)).reset_index()
#State_sales
```

In [71]:
```
#State_sales.plot(kind='barh', x='State', rot=True)
sns.barplot(x='State', y='Amount', data=State_sales)
sns.set(rc={'figure.figsize':(18,4)})
```

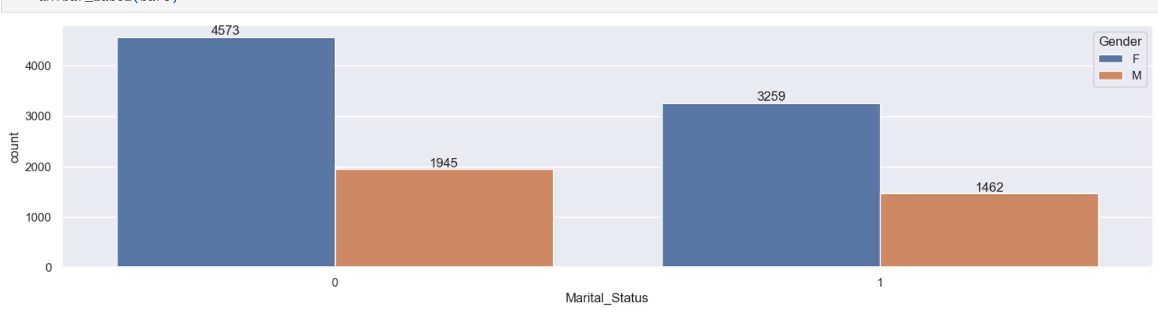


From above graphs, we can infer that the most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively.

Marital Status

In [72]:
```
df.columns
```
Out[72]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [73]:
```
ax = sns.countplot(x='Marital_Status', data=df,hue='Gender')
sns.set(rc={'figure.figsize':(7,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```

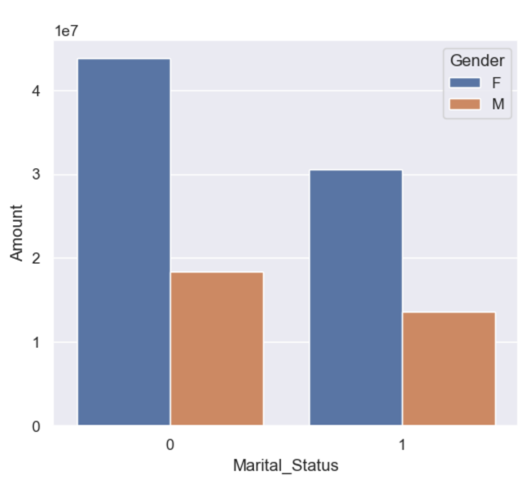


In [74]:
```
sales_Marital = pd.DataFrame(df.groupby(['Marital_Status', 'Gender'])['Amount'].sum().sort_values(ascending=False)).reset_index()

sales_Marital
```
Out[74]:

	Marital_Status	Gender	Amount
0	0	F	43786646
1	1	F	30549207
2	0	M	18338738
3	1	M	13574538

In [75]:
```
sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(data = sales_Marital, x = 'Marital_Status',y= 'Amount', hue='Gender')
```
Out[75]:
```
<AxesSubplot:xlabel='Marital_Status', ylabel='Amount'>
```



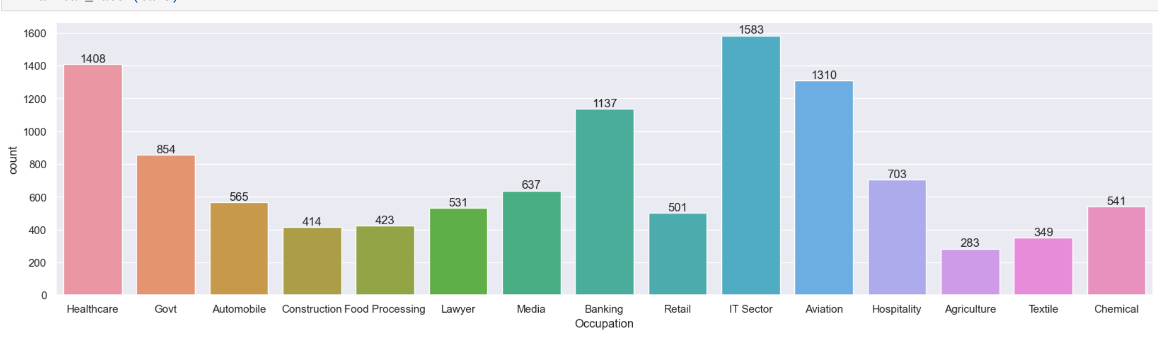
From above graphs, we can infer that the most of the buyers are married (women) and they have high purchasing power

Occupation

Total orders vs Occupation type

In [76]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

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

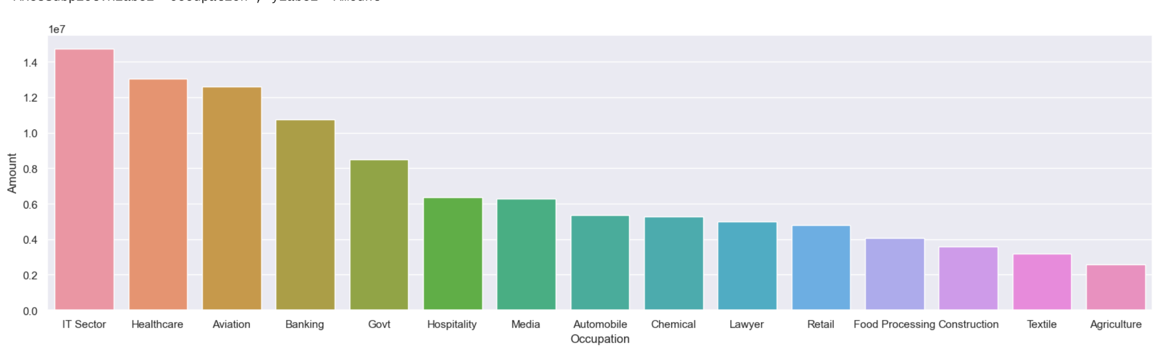


Total Sales vs Occupation

In [77]:
```
sales_occu = pd.DataFrame(df.groupby(['Occupation'])['Amount'].sum().sort_values(ascending=False)).reset_index()
sales_occu
```
Out[77]:

	Occupation	Amount
0	IT Sector	14755079
1	Healthcare	13034586
2	Aviation	12602298
3	Banking	10770610
4	Govt	8517212
5	Hospitality	6376405
6	Media	6295832
7	Automobile	5368596
8	Chemical	5297436
9	Lawyer	4981665
10	Retail	4783170
11	Food Processing	4070670
12	Construction	3597511
13	Textile	3204972
14	Agriculture	2593087

In [78]:
```
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_occu, x = 'Occupation',y= 'Amount')
```
Out[78]:
```
<AxesSubplot:xlabel='Occupation', ylabel='Amount'>
```



From above graphs, we can infer that the most of the buyers & customers with maximum purchasing power are working in IT, Healthcare, and Aviation sector

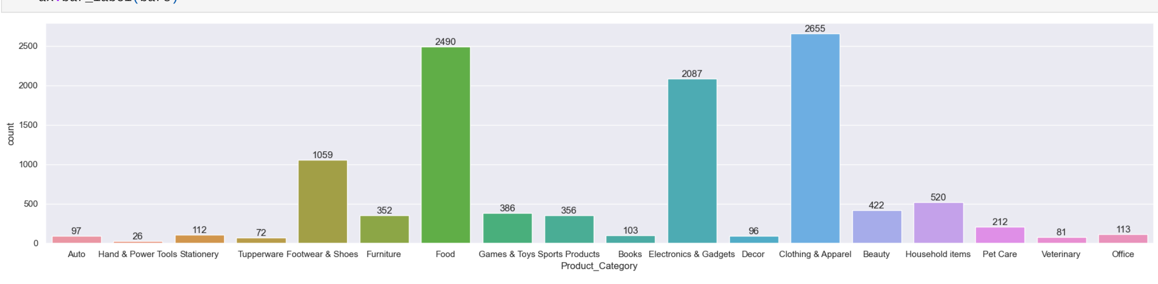
Product Category

In [79]:
```
df.columns
```
Out[79]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

Total orders vs Product category

In [80]:
```
sns.set(rc={'figure.figsize':(25,5)})
ax = sns.countplot(data = df, x = 'Product_Category')

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

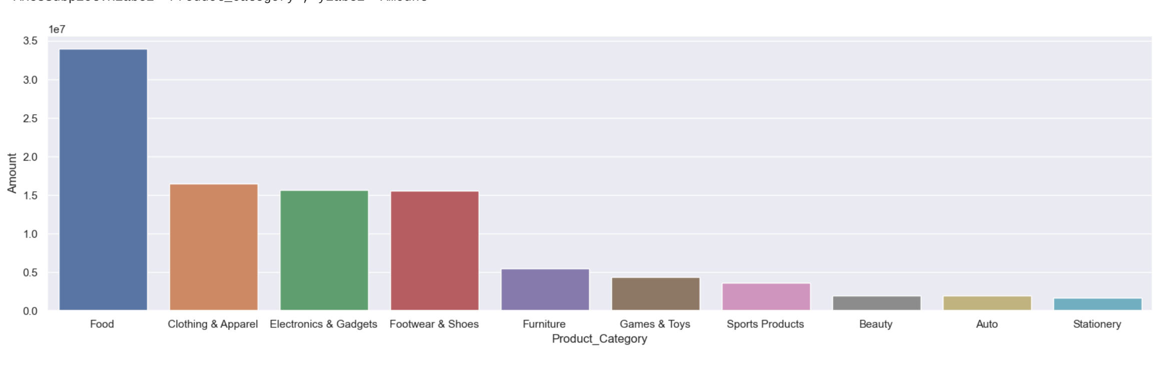


Top 10 product categories with highest sales

In [81]:
```
sales_product_categ = pd.DataFrame(df.groupby(['Product_Category'])['Amount'].sum().sort_values(ascending=False)).head(10).reset_index()
sales_product_categ
```
Out[81]:

	Product_Category	Amount
0	Food	33933883
1	Clothing & Apparel	16495019
2	Electronics & Gadgets	15643846
3	Footwear & Shoes	15575209
4	Furniture	5440051
5	Games & Toys	4331694
6	Sports Products	3635933
7	Beauty	1959484
8	Auto	1958609
9	Stationery	1676051

In [82]:
```
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_product_categ, x = 'Product_Category',y= 'Amount')
```
Out[82]:
```
<AxesSubplot:xlabel='Product_Category', ylabel='Amount'>
```

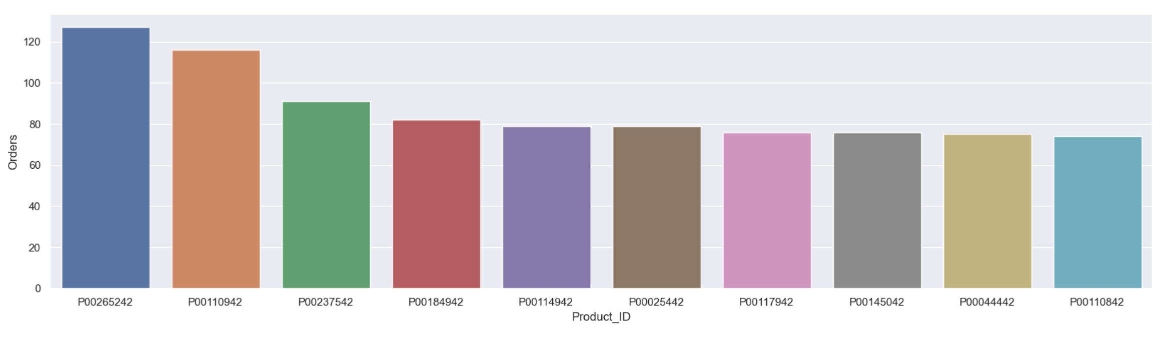


From above graphs, we can infer that most of the sold products are from Food, Clothing and Electronics category

Top 10 most sold products

In [83]:
```
Product_orders = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = Product_orders, x = 'Product_ID',y= 'Orders')
```
Out[83]:
```
<AxesSubplot:xlabel='Product_ID', ylabel='Orders'>
```



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