In [110]:
```
#import libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt #visualizing Data
%matplotlib inline
import seaborn as sns
```

In [111]:
```
#import Csv File
df = pd.read_csv('Diwali Sales Data.csv', encoding = 'unicode_escape')
```

In [112]:
```
#Checking the rows and columns
df.shape
```

Out[112]: (11251, 15)

In [113]:
```
#using Head method to visualize the top 5 data in the csv file
df.head()
```

Out[113]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zo
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	West
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	South
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Cen
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	South
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	West

In [114]:
```
# Info() method used to find the object type of the data in the csv file
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 [115]:
```
# Describe method used to find out the datas descriptive analysis statistics o
#It's not important in this analysis
df.describe()
```

Out[115]:

	User_ID	Age	Marital_Status	Orders	Amount	Status	unnamed
count	1.125100e+04	11251.000000	11251.000000	11251.000000	11239.000000	0.0	0.
mean	1.003004e+06	35.421207	0.420318	2.489290	9453.610858	NaN	Na
std	1.716125e+03	12.754122	0.493632	1.115047	5222.355869	NaN	Na
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000	NaN	Na
25%	1.001492e+06	27.000000	0.000000	1.500000	5443.000000	NaN	Na
50%	1.003065e+06	33.000000	0.000000	2.000000	8109.000000	NaN	Na
75%	1.004430e+06	43.000000	1.000000	3.000000	12675.000000	NaN	Na
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000	NaN	Na

In [116]:
```
# Drop method used to remove the unfilled/Blanks in the column
df.drop(['Status','unnamed1'],  axis=1, inplace=True)
```

In [117]:
```
#isnull function check the columns has null values or not (Removed the blanks)
pd.isnull(df).sum()
```

Out[117]:
```
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 [118]:
```
#dropna method used to remove the null values
# inplace=True in pandas used for giving permission to make the change permane
df.dropna(inplace=True)
```

In [119]:
```
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 [120]:
```
#Here Amount we want total No chances like .23 and all.
#so we are changing the datatype of the Amount Column as Integer
#using astype function used to change particularly column select datatype
df['Amount'] =df['Amount'].astype('int')
```

In [121]:
```
#checking the Amount column data type
df['Amount'].dtypes
```

Out[121]:
```
dtype('int32')
```

In [122]:
```
#shows that the columns in the csv file
df.columns
```

Out[122]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [123]:
```
#rename the column by using rename() method
df.rename(columns = {'Cust_name':'Customer_name'})
```

Out[123]:

	User_ID	Customer_name	Product_ID	Gender	Age Group	Age	Marital_Status	Sta
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharasht
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Prades
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Prades
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnatak
4	1000588	Joni	P00057942	M	26-35	28	1	Gujar
...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharasht
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryar
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhy Prades
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnatak
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharasht

11239 rows × 13 columns

In [124]:
```
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  int32 
dtypes: int32(1), int64(4), object(8)
memory usage: 1.2+ MB
```

In [125]:
```
df[['Age','Orders','Amount']].describe()
```

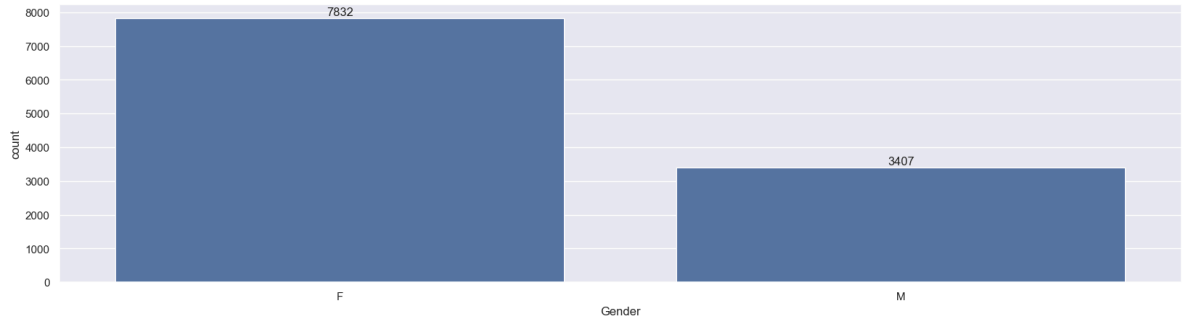
Out[125]:

	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

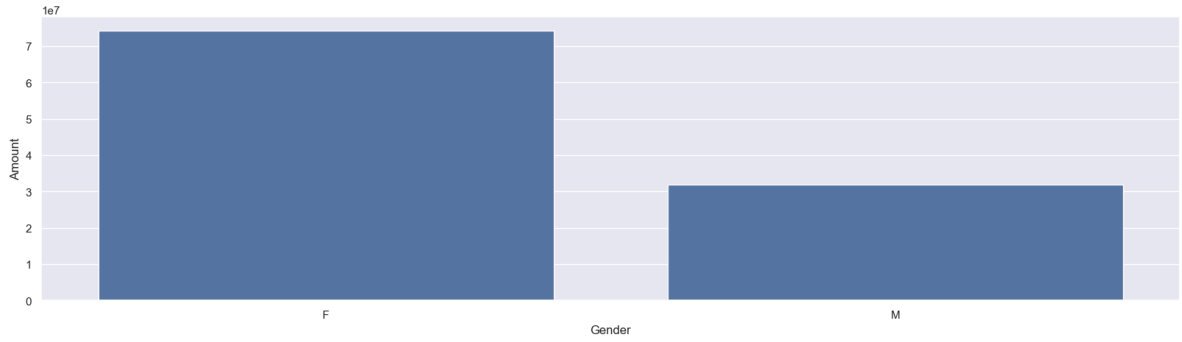
In [126]:
```
# plotting a bar chart for Gender and it's count

ax = sns.countplot(x = 'Gender',data = df)

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



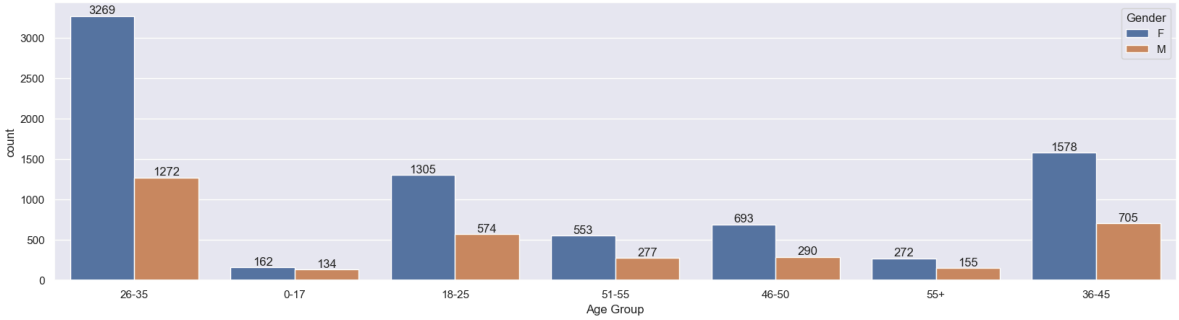
Age

In [127]:
```
# plotting a bar chart for gender vs total amount

sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values

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

Out[127]:
```
<Axes: xlabel='Gender', ylabel='Amount'>
```

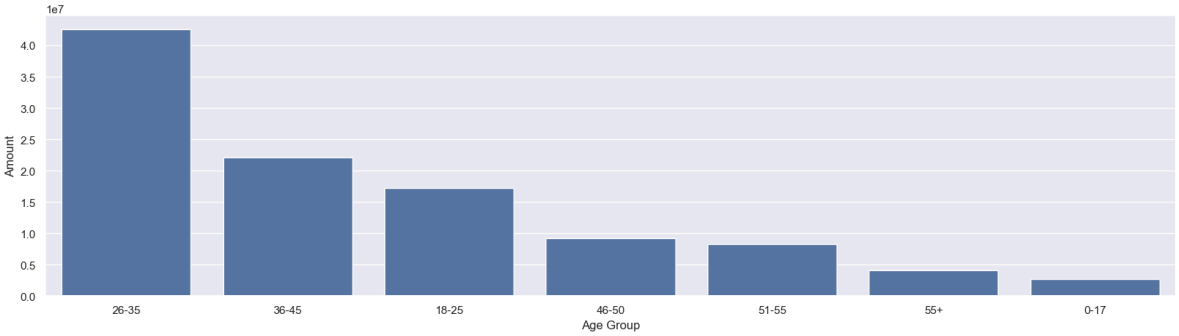


In [128]:
```
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

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



In [129]:
```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_val

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

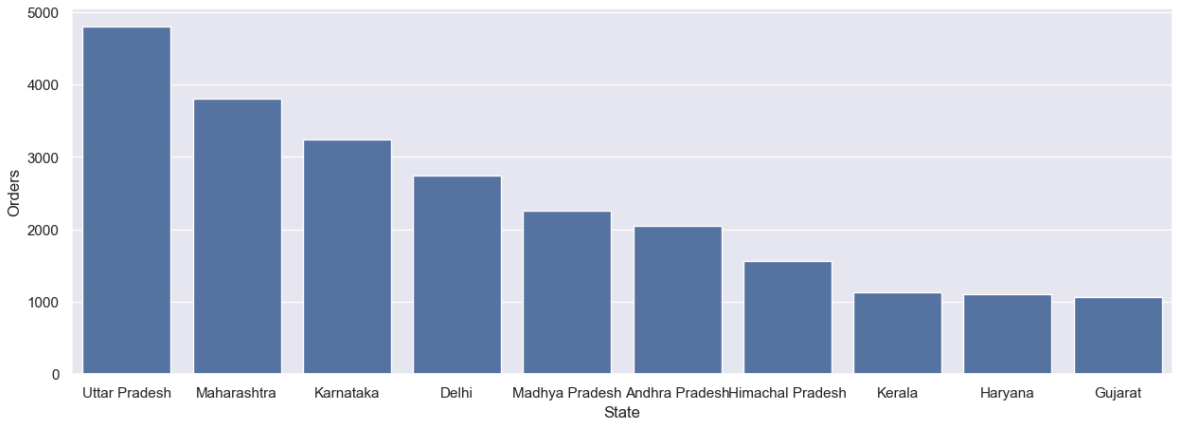
Out[129]: <Axes: xlabel='Age Group', ylabel='Amount'>



State

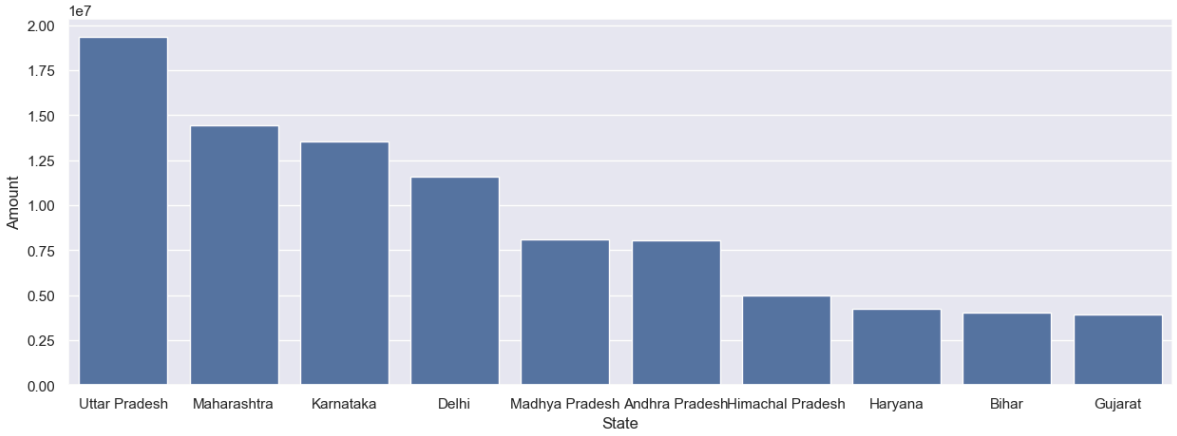
In [130]:
```
# total number of orders from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_value

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

Out[130]: <Axes: xlabel='State', ylabel='Orders'>



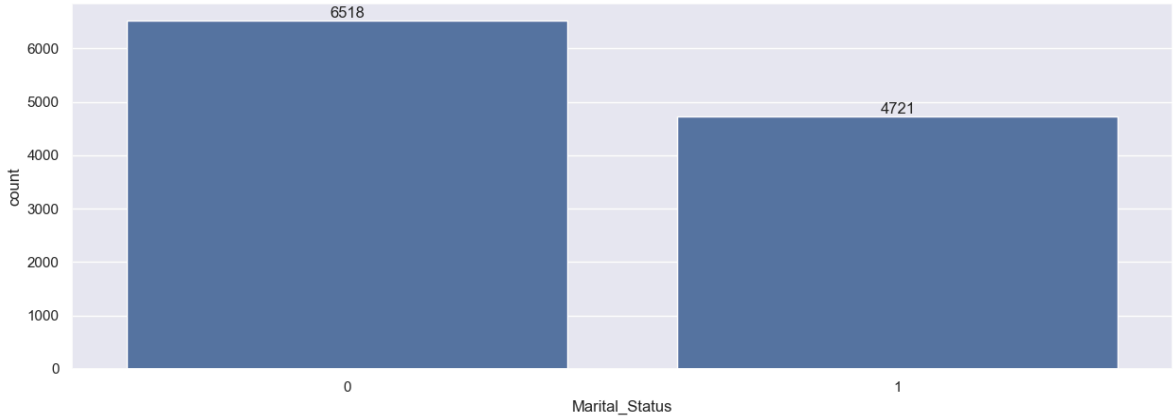
In [131]:
```
# total amount/sales from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_value

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

Out[131]: <Axes: xlabel='State', ylabel='Amount'>



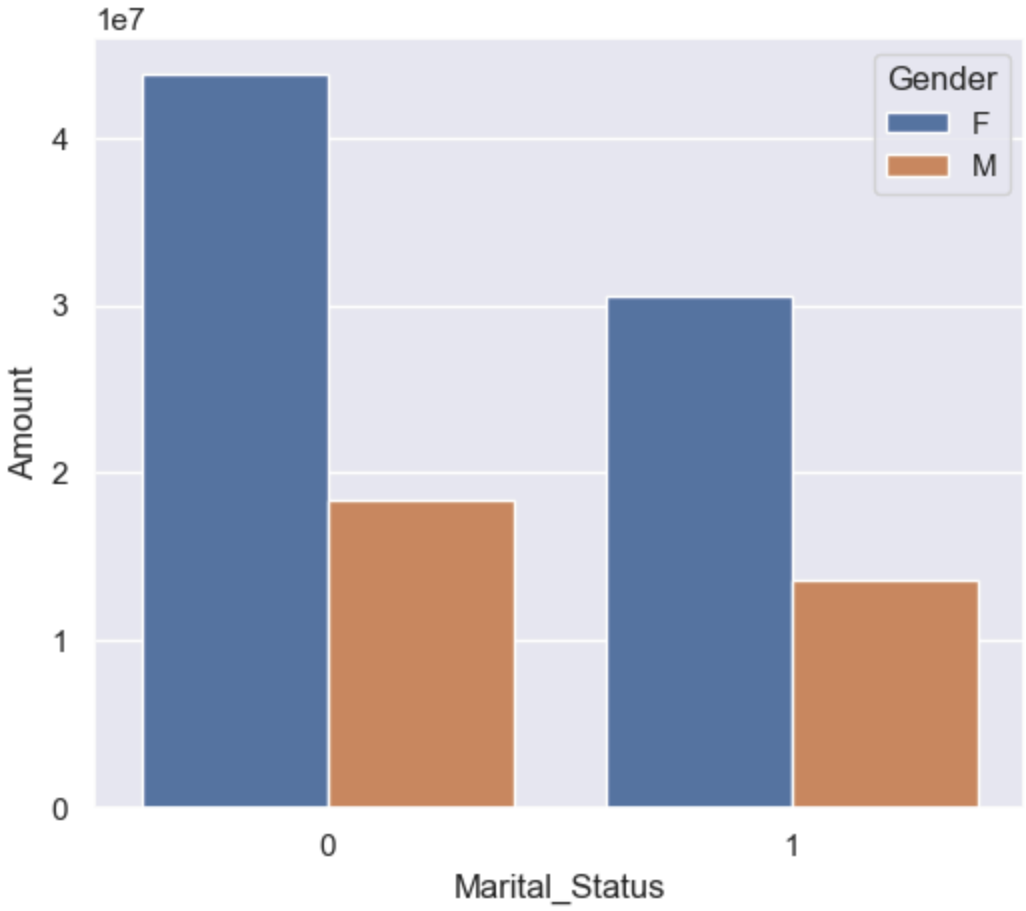
Marital Status

In [132]:
```
ax = sns.countplot(data = df, x = 'Marital_Status')
sns.set(rc={'figure.figsize':(7,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```



In [133]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount

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

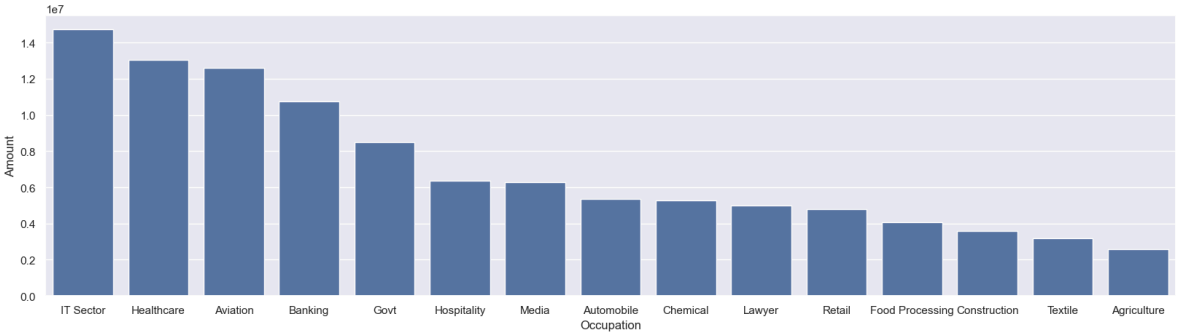
Out[133]: <Axes: xlabel='Marital_Status', ylabel='Amount'>



Occupation

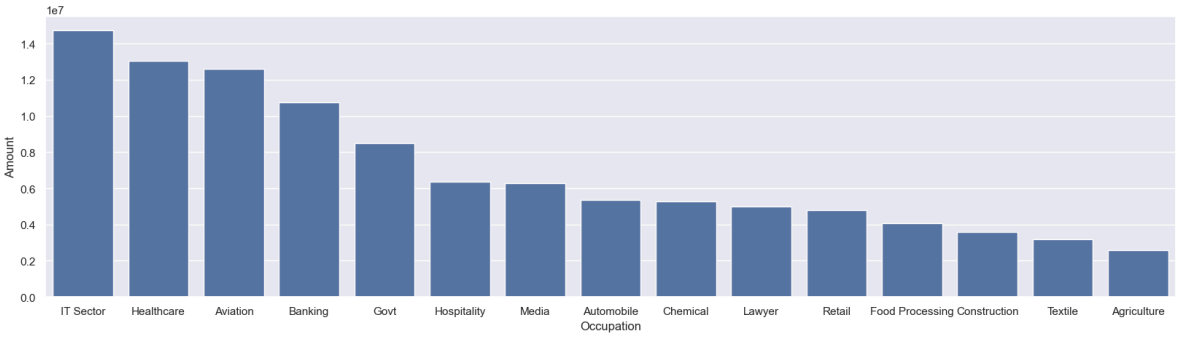
In [134]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_

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

Out[134]: <Axes: xlabel='Occupation', ylabel='Amount'>



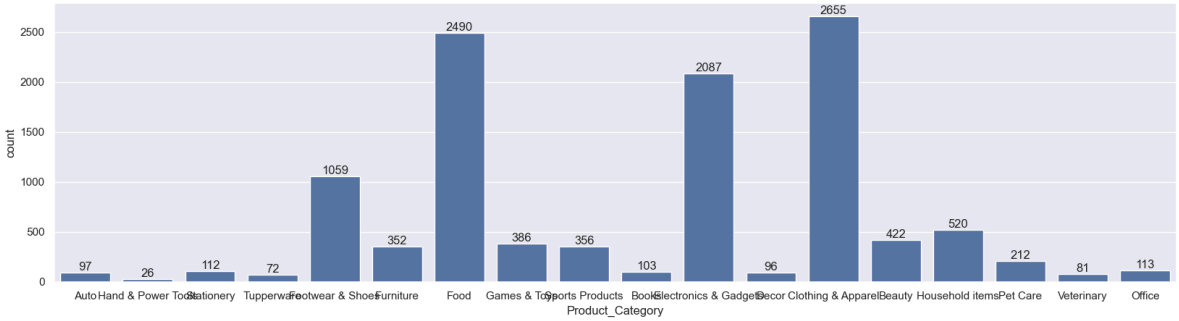
In [135]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_

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

Out[135]: <Axes: xlabel='Occupation', ylabel='Amount'>



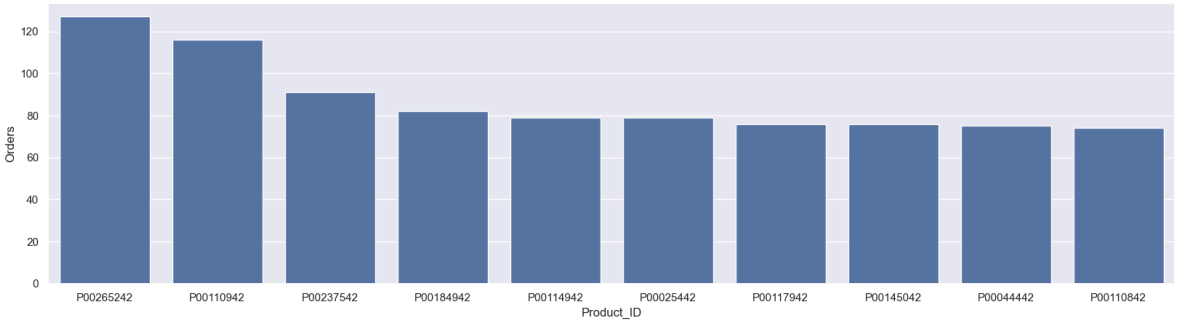
Product Category

In [136]:
```
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 [137]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_

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

Out[137]: <Axes: xlabel='Product_ID', ylabel='Orders'>



In [138]:
```
# 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=Fa
```

Out[138]:
```
<Axes: xlabel='Product_ID'>
```

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 [ ]: