Importing Lib.

In [1]:

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

Import CSV file

In [2]:

```
df = pd.read_csv('Sales Data.csv' , encoding= 'unicode_escape')
```

In [4]:

```
df.shape
```

Out[4]:

```
(11251, 15)
```

In [5]:

```
df.head(10)
```

Out[5]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh
7	1002092	Shivangi	P00273442	F	55+	61	0	Maharashtra
8	1003224	Kushal	P00205642	M	26-35	35	0	Uttar Pradesh
9	1003650	Ginny	P00031142	F	26-35	26	1	Andhra Pradesh

In [6]:

```
df.tail(10)
```

Out[6]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	Sta
11241	1003032	Matthias	P00058042	F	26-35	33	0	De
11242	1004344	Hildebrand	P00185442	F	26-35	27	1	De
11243	1005446	Sheetal	P00297742	M	51-55	53	0	Guja
11244	1005446	Sheetal	P00297742	M	51-55	53	0	Madh Prade
11245	1004140	Bertelson	P00057442	F	26-35	31	1	De
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharash
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Harya
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madh Prade
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnata
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharash

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
```

Drop blank columns

In [8]:

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

In [10]:

```
pd.isnull(df).sum()
```

Loading [MathJax]/extensions/Safe.js

```
Out[10]: 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
```

Drop null value

```
In [11]: df.dropna(inplace=True)
```

```
In [12]: pd.isnull(df).sum()
```

```
Out[12]: 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
```

Change Data Type

```
In [14]: df['Amount']=df['Amount'].astype(int)
```

```
In [17]: df.dtypes
```

```
Out[17]: 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
```

```
In [19]: df.columns
```

```
Out[19]: Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
                'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
                'Orders', 'Amount'],
               dtype='object')
```

Rename Column

```
In [22]: df.rename(columns = {'Marital_Status' : 'Vivah'})
```

Out[22]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Vivah	State
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat
...	...	...	...	...	...	...	...	...
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra

11239 rows × 13 columns

Loading [MathJax]/extensions/Safe.js

Count , Mean , Std. , etc

In [23]:
```
df.describe()
```

Out[23]:

	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 [24]:
```
df[['Age' , 'Orders' , 'Amount']].describe()
```

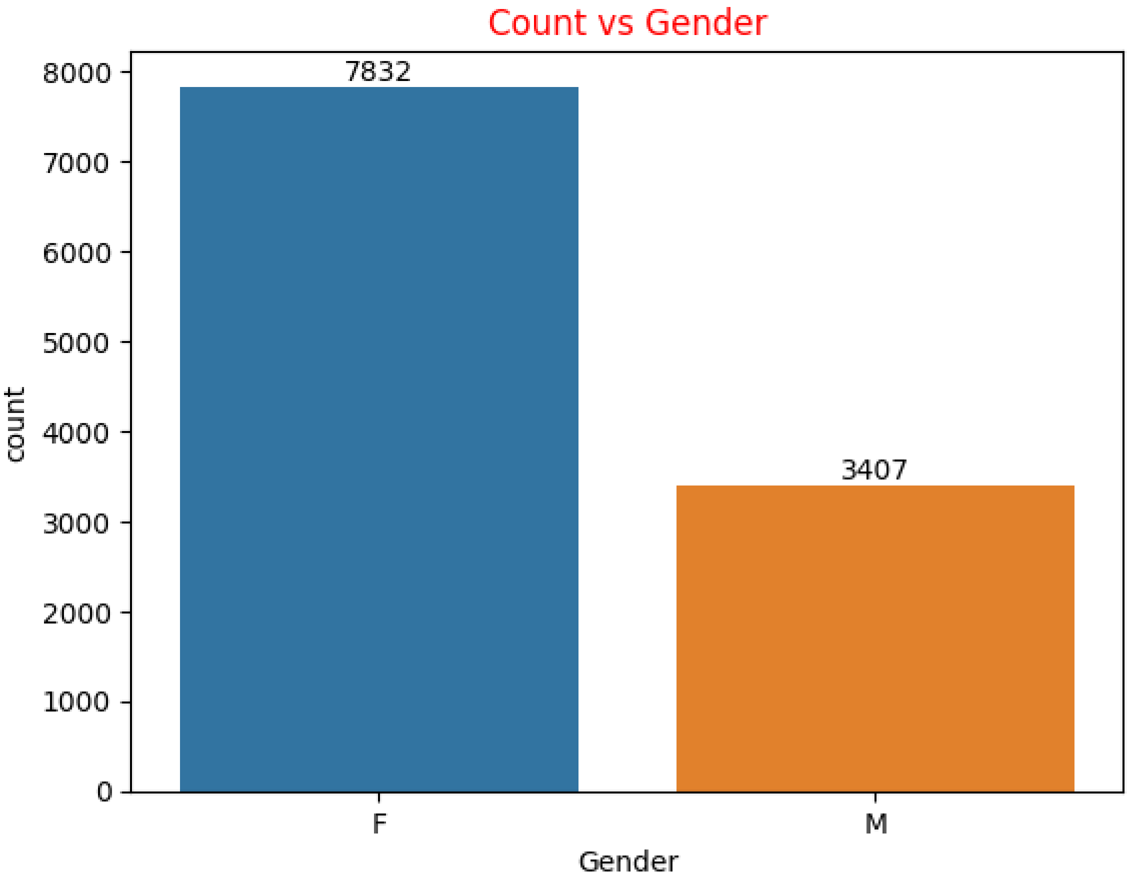
Out[24]:

	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

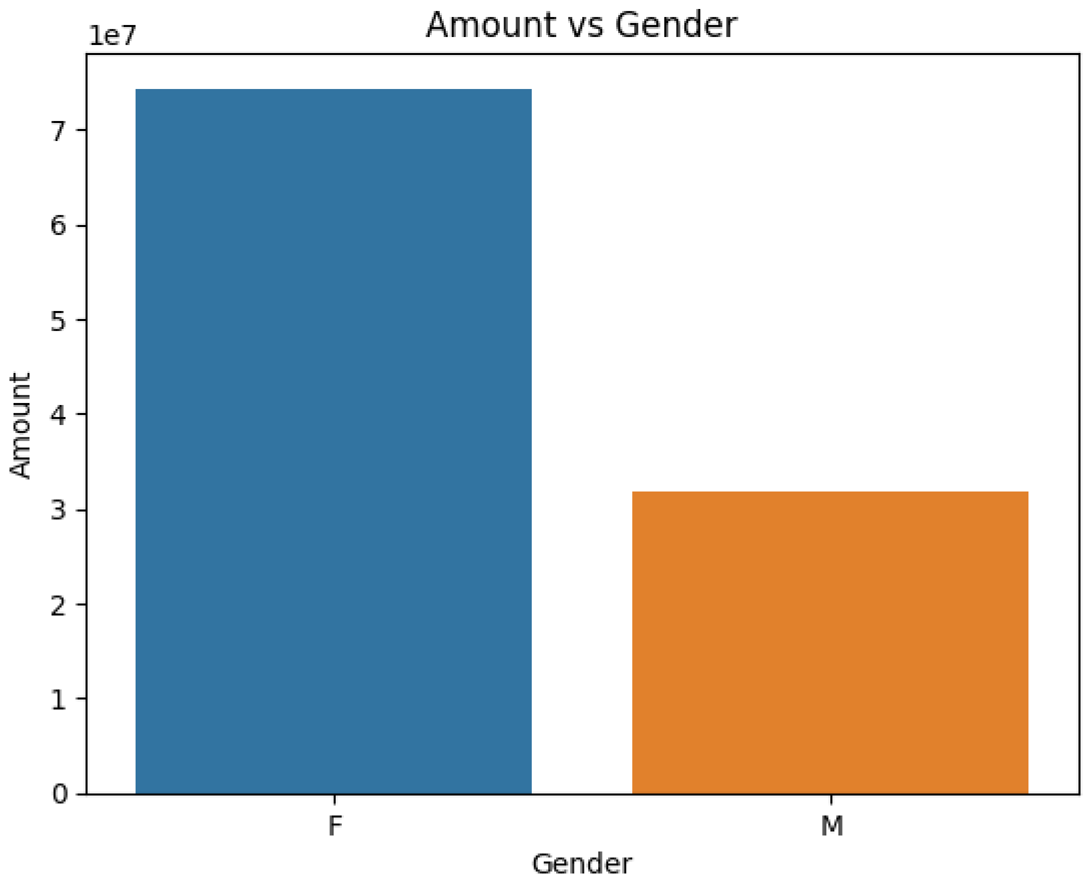
In [28]:
```
ax = sns.countplot(x = 'Gender',data = df)

for bars in ax.containers:
    ax.bar_label(bars)
    plt.title('Count vs Gender' , color='red')
```



```
In [27]: sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_valu

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

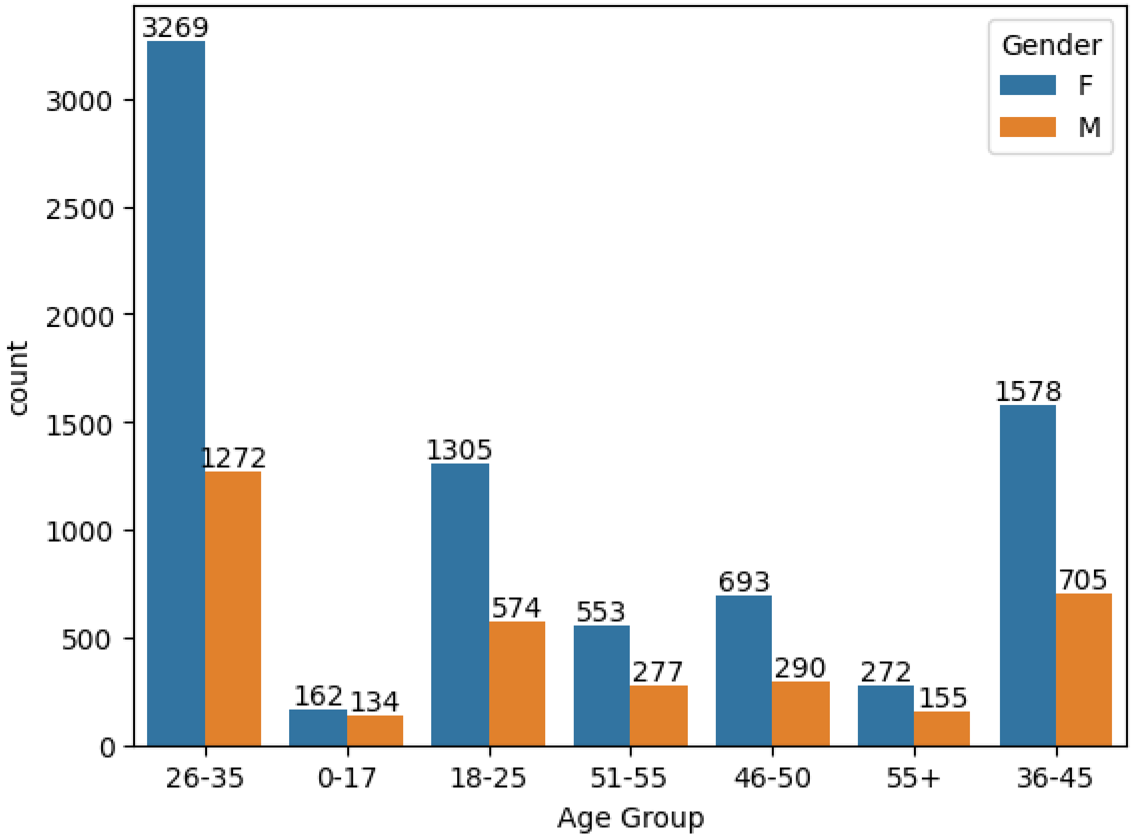
```
Out[27]: Text(0.5, 1.0, 'Amount vs Gender')
```

Loading [MathJax]/extensions/Safe.js



Age Group

```
In [29]: ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

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

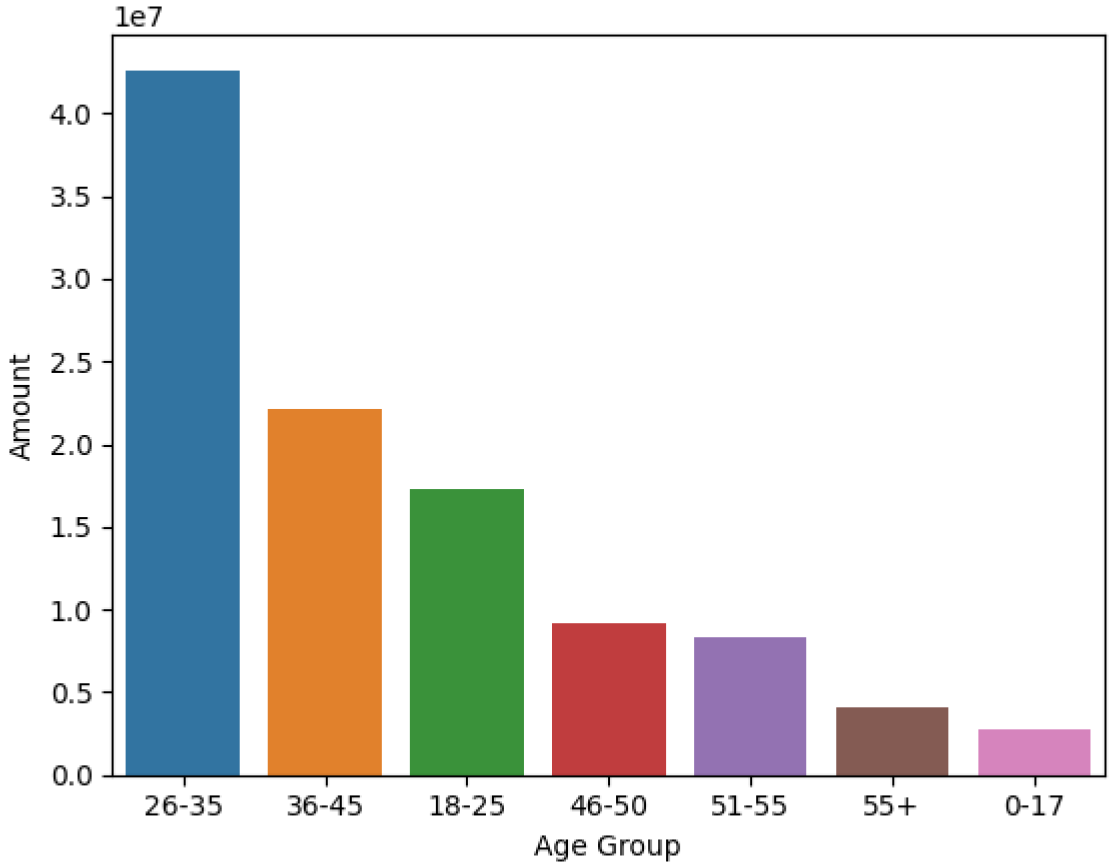


Amount vs Age Group

```
In [30]: sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_v

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

```
Out[30]: <Axes: xlabel='Age Group', ylabel='Amount'>
```

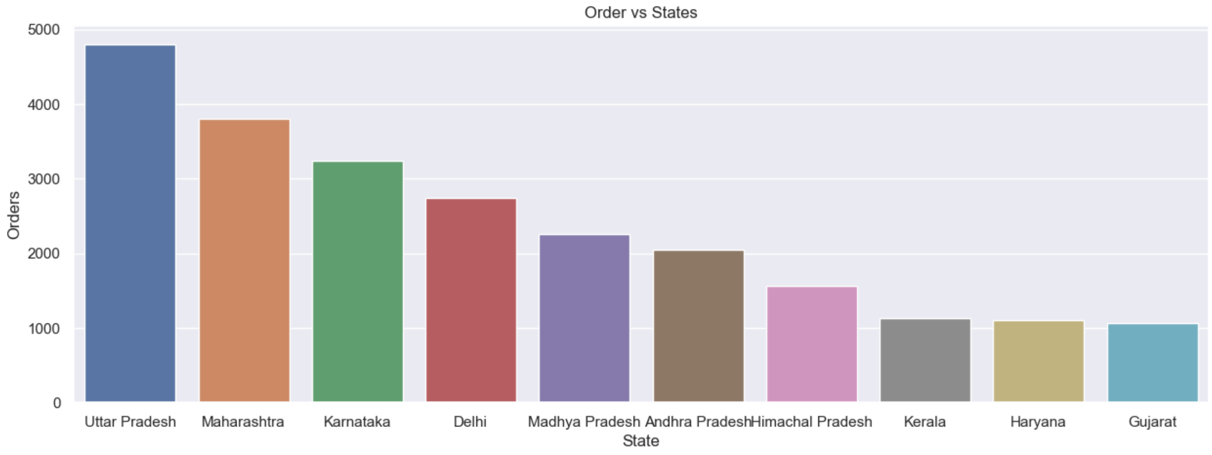
Loading [MathJax]/extensions/Safe.js



States

In [47]:
```
sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_val

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

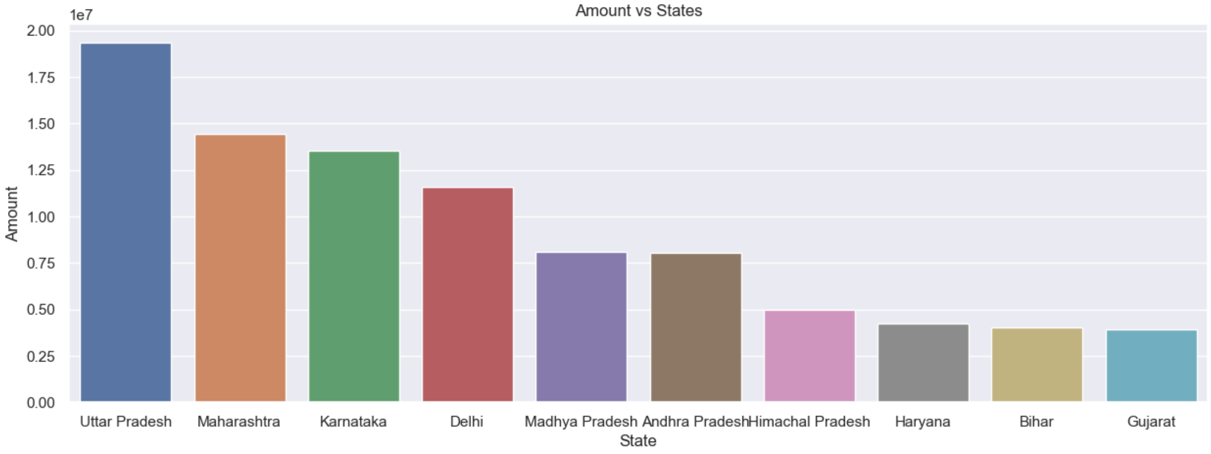
Out[47]:
```
Text(0.5, 1.0, 'Order vs States')
```



In [46]:
```
sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_val
```

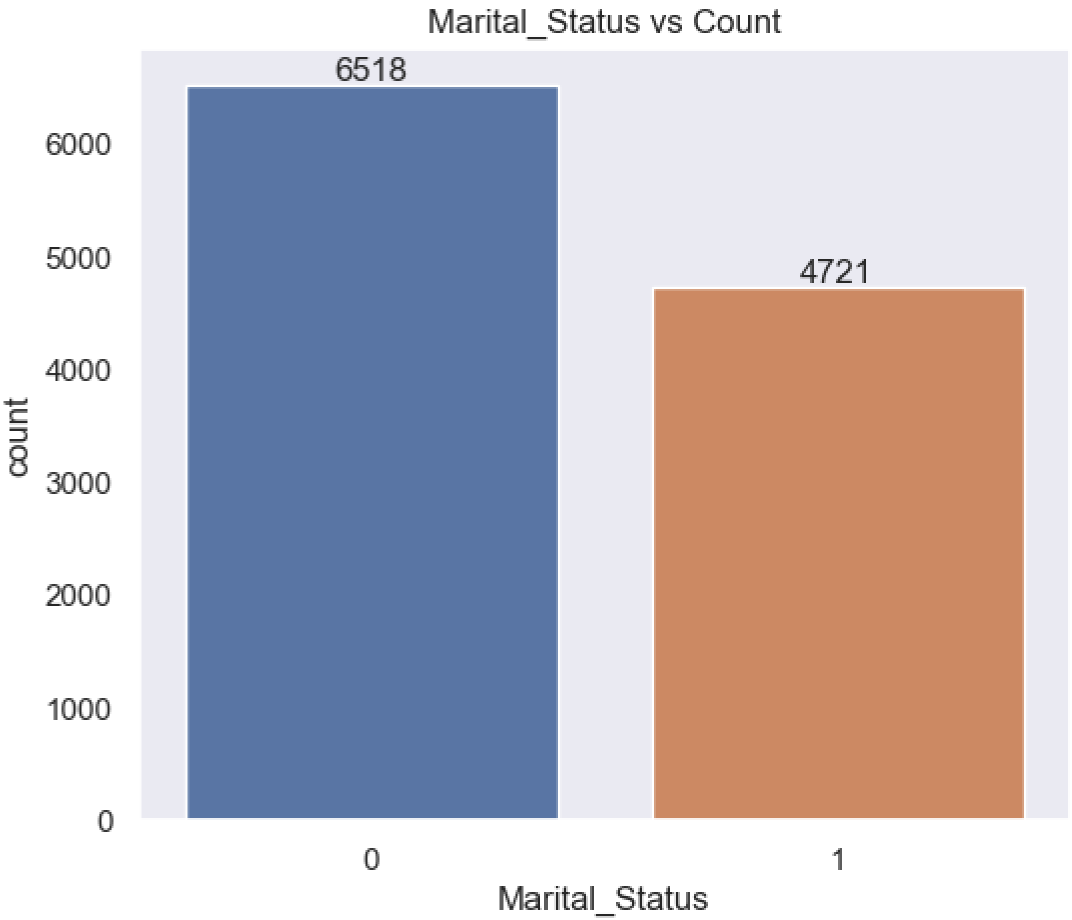
```
sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
plt.title('Amount vs States')
```

```
Out[46]: Text(0.5, 1.0, 'Amount vs States')
```



Marital Status

```
In [43]: ax = sns.countplot(data = df, x = 'Marital_Status')

         sns.set(rc={'figure.figsize':(7,5)})
         for bars in ax.containers:
             ax.bar_label(bars)
             ax.grid(False)
             plt.title('Marital_Status vs Count')
```

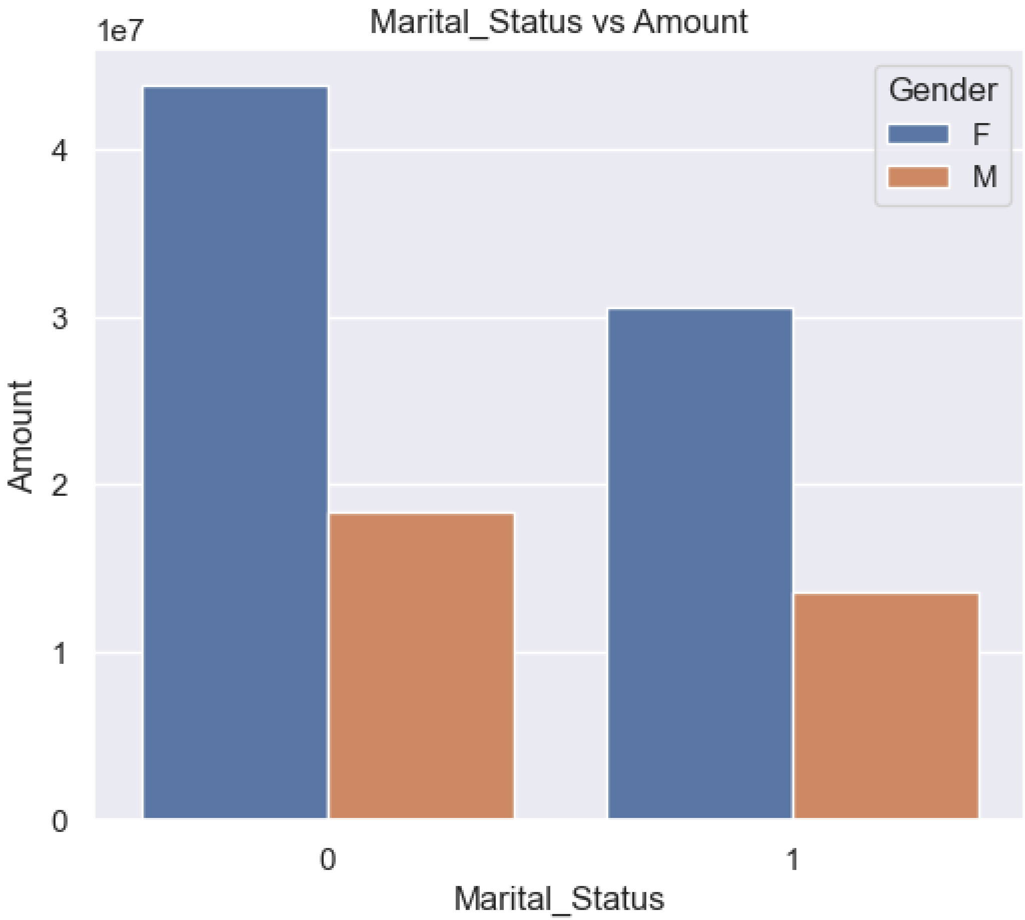


In [42]:

```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amou

sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(data = sales_state, x = 'Marital_Status',y= 'Amount', hue='Gende
ax.grid(False)
plt.title('Marital_Status vs Amount')
```

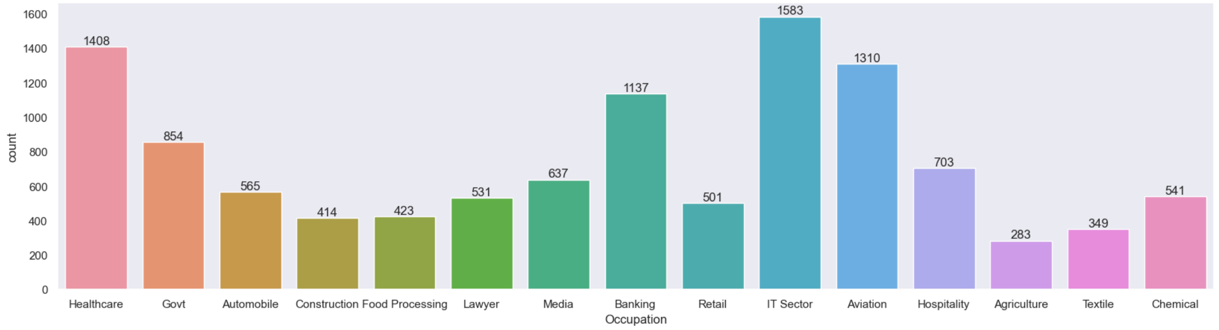
Out[42]:

```
Text(0.5, 1.0, 'Marital_Status vs Amount')
```



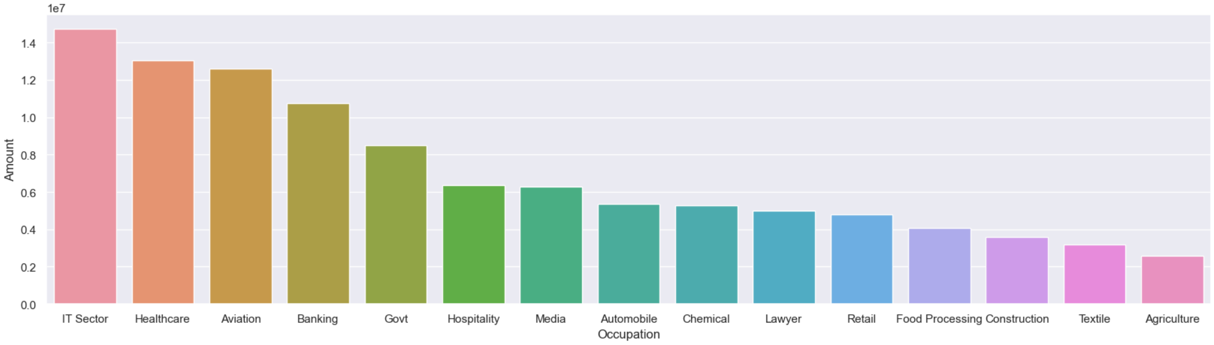
Occupations

In [40]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')
ax.grid(False)
for bars in ax.containers:
    ax.bar_label(bars)
```



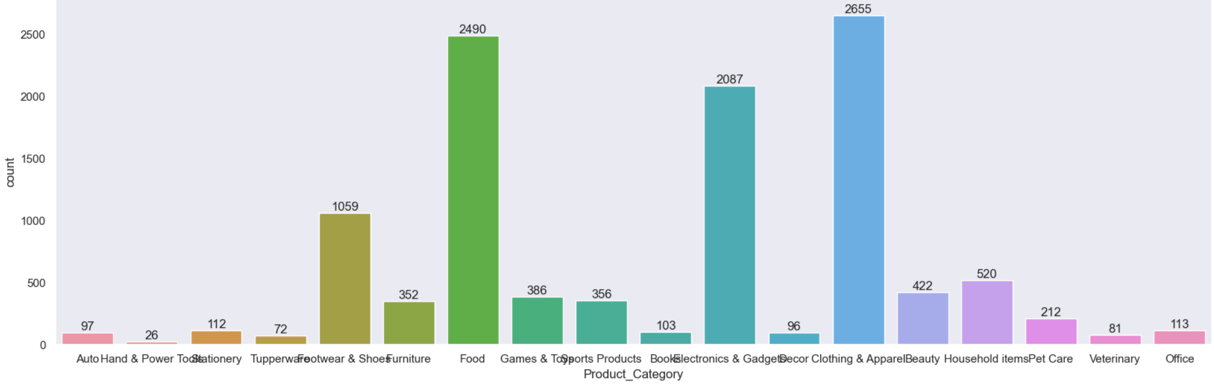
In [50]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sor

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

Loading [MathJax]/extensions/Safe.js

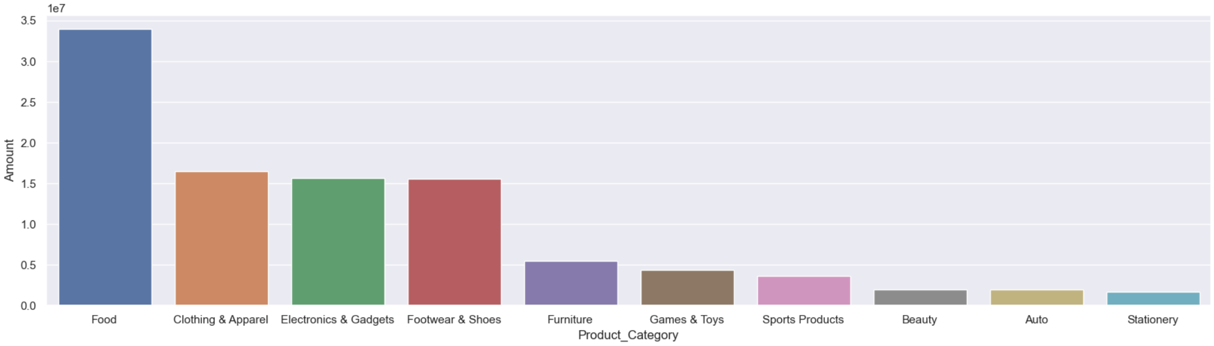


Product Category

In [52]:
```
sns.set(rc={'figure.figsize':(20,6)})
ax = sns.countplot(data = df, x = 'Product_Category')

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

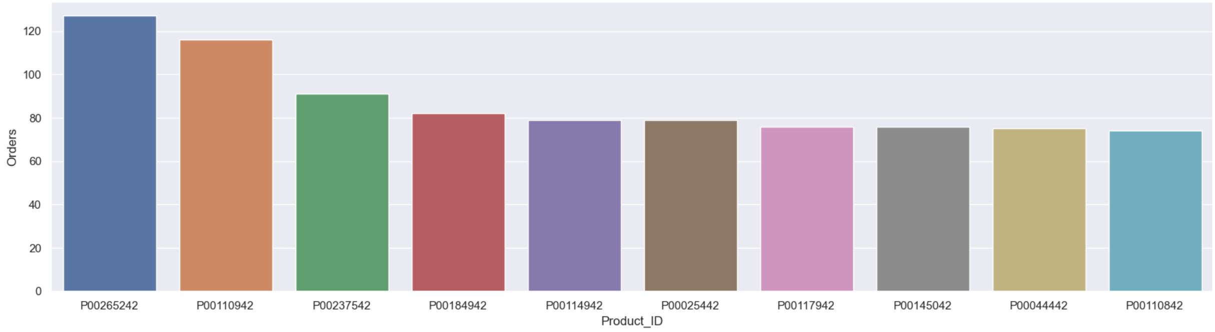


In [54]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum

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



In [55]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sor

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

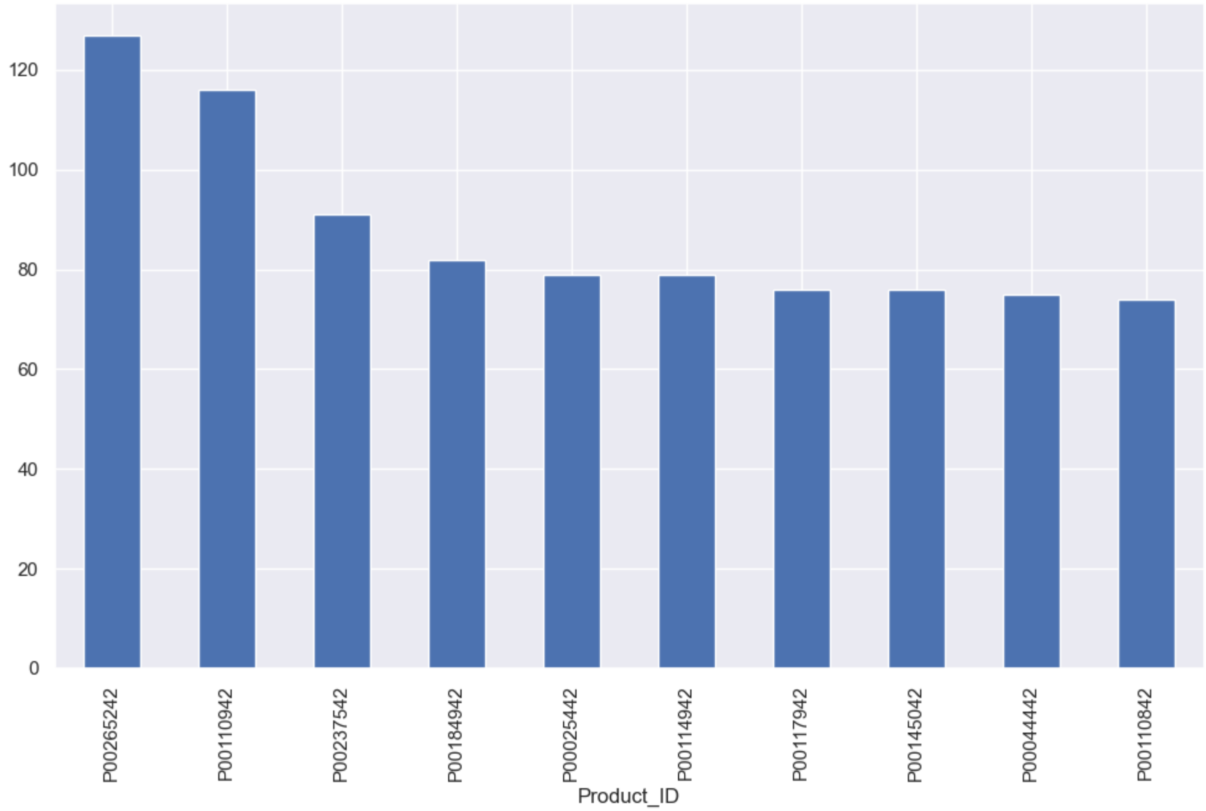
```
Out[55]: <Axes: xlabel='Product_ID', ylabel='Orders'>
```



Top 10 Most Selling Product

```
In [58]: fig1, ax1 = plt.subplots(figsize=(12,7))
         df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=
```

```
Out[58]: <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