In [ ]:
```
#SALES ANALYSIS.
```

In [5]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt #  for visualizing.
%matplotlib inline
import seaborn as sns
```

In [6]:
```
df = pd.read_csv(r'C:\Users\SAMAD\Downloads\Datasets\Python_Diwali_Sales_Analysis\
```

In [ ]:
```
# DATA INFORMATION.
```

In [7]:
```
df.shape
```

Out[7]:
```
(11251, 15)
```

In [8]:
```
df.head()
```

Out[8]:

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

In [9]:
```
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 [ ]:
```
# DATA CLEANING.
```

In [10]:
```
#drop null/blank columns.
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

In [11]:
```
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 [14]:
```
# CHECKING FOR NULL VALUES.
pd.isnull(df).sum()
```

Out[14]:
```
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 [17]:
```
df.shape
```

Out[17]:
```
(11239, 13)
```

In [18]:
```
# DROPING NULL VALUES.
df.dropna(inplace=True) # USING INPLACE TO SAVE DATA.
```

In [20]:
```
pd.isnull(df).sum()
```

Out[20]:
```
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
```

In [21]:
```
# CONVERTING DATA TYPE.
df['Amount'] = df['Amount'].astype('int') # USING ASTYPE FOR CONVERSION.
```

In [22]:
```
df['Amount'].dtypes
```

Out[22]:
```
dtype('int32')
```

In [23]:
```
df.columns # FOR CHECKING COLUMN NAMES.
```

Out[23]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [25]:
```
# describe() METHOD RETURNS DESCRIPTION OF DATA IN DATAFRAME.
df.describe()
```

Out[25]:

	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 [27]:
```
# USING describe() FOR SPECIFIC DATA OR COLUMN.
df[['Age', 'Orders', 'Amount']].describe()
```

Out[27]:

	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

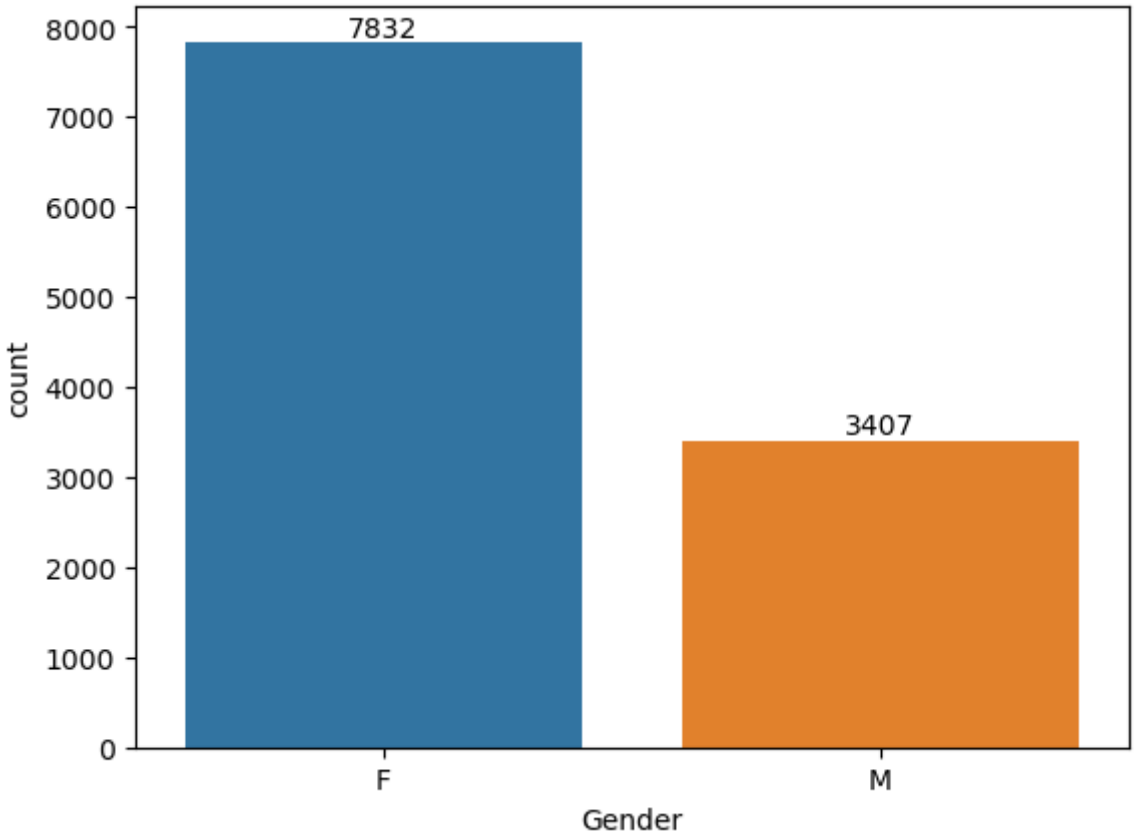
Exploratory Data Analysis.

Gender

In [28]:
```
df.columns
```

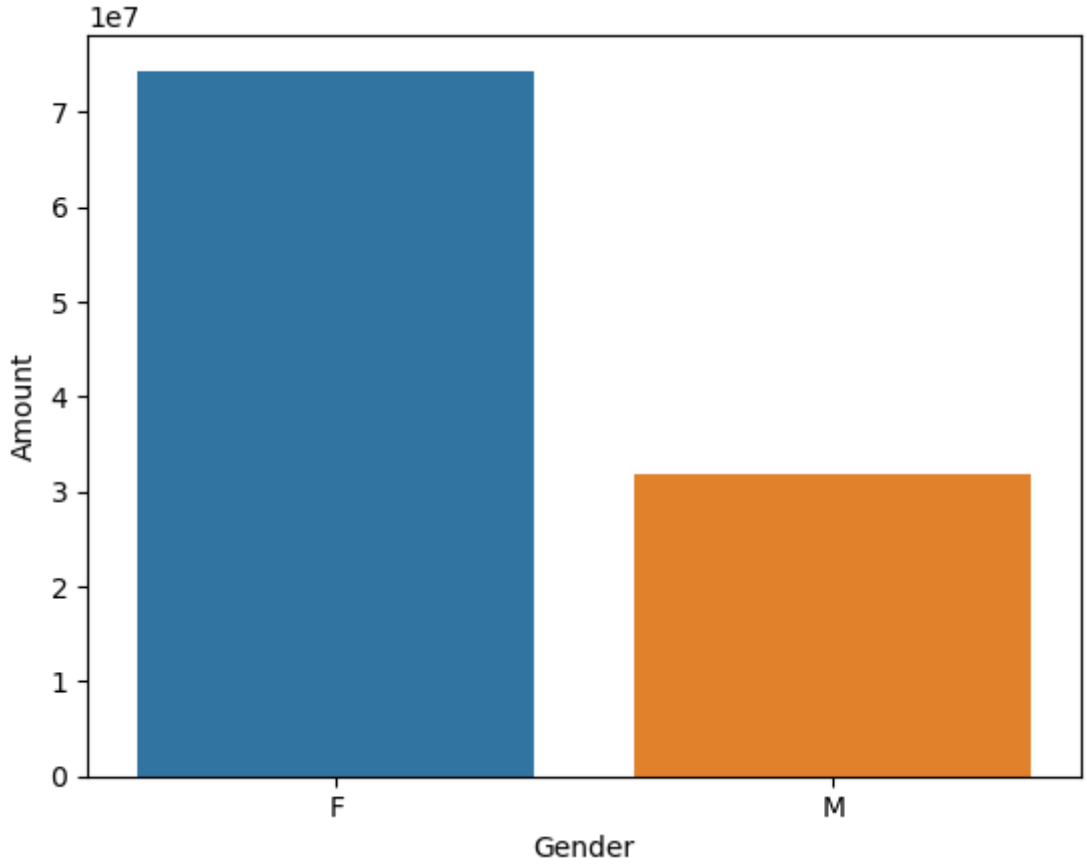
Out[28]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [29]:
```
# PLOTTING A BAR CHART FOR IT'S GENDER AND COUNT.

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

for bars in ax.containers: # FOR BAR LABELS.
    ax.bar_label(bars)
```



In [33]:
```
# plotting a bar chart for gender vs total amount

sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(by=
#using sort_values for sorting values

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

Out[33]:
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



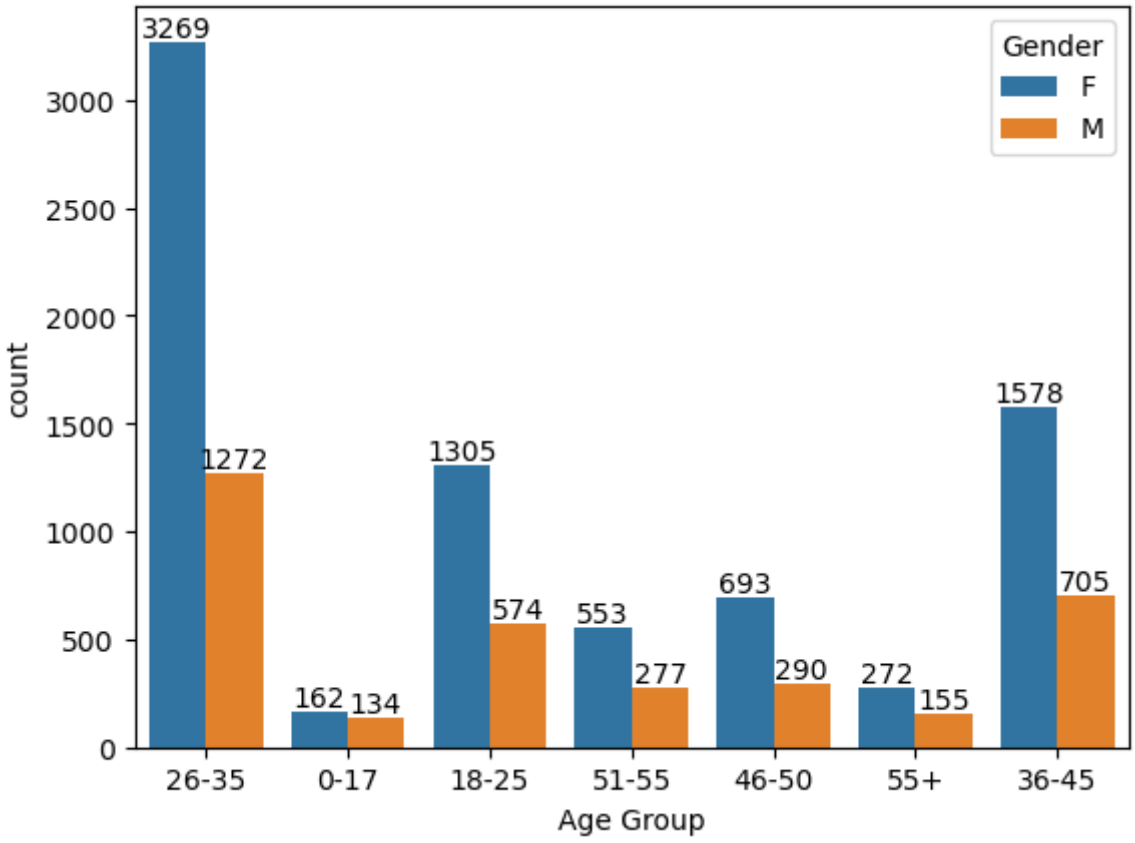
Above graphs showcase that purcahsing power of females are very high in compare to males.

Age

In [35]:
```
df.columns
```

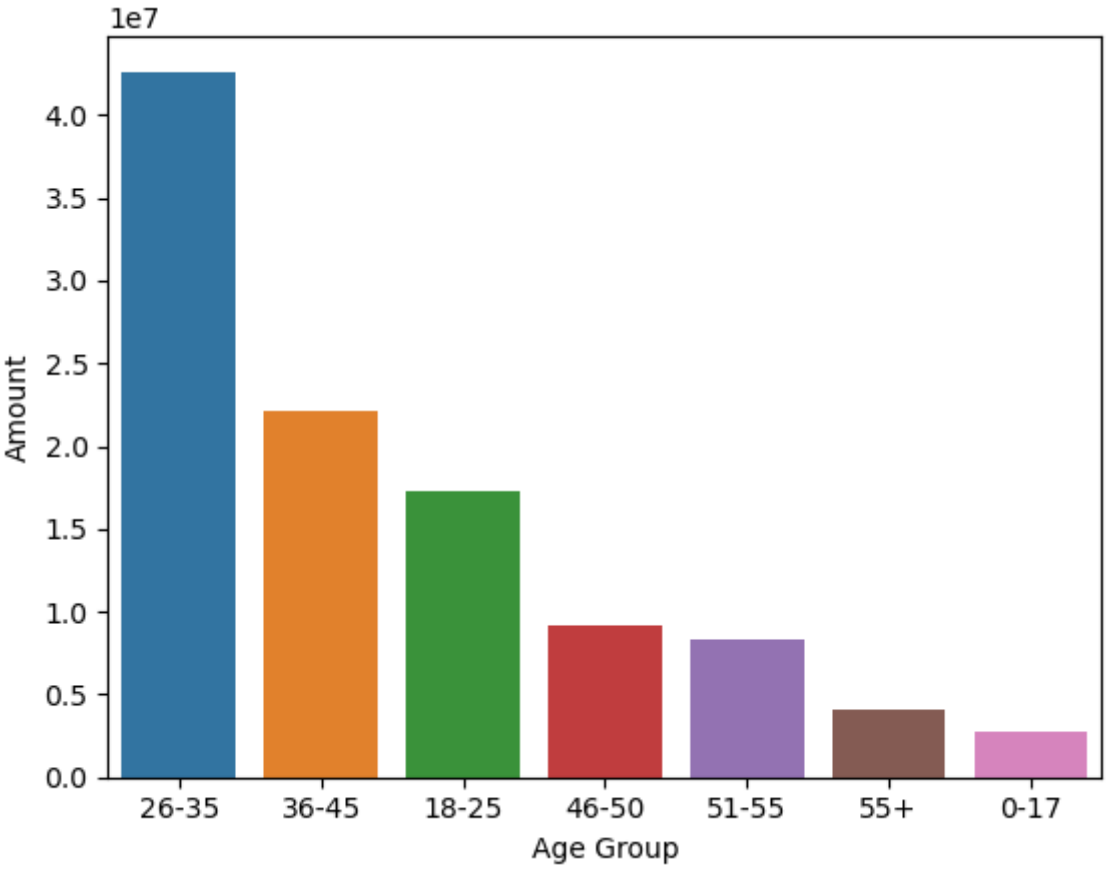
Out[35]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [38]:
```
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender') #USING hue FOR VISU

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



In [43]:
```
# AMOUNT IN COMPARE TO AGE.
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(

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

Out[43]:
```
<Axes: xlabel='Age Group', ylabel='Amount'>
```

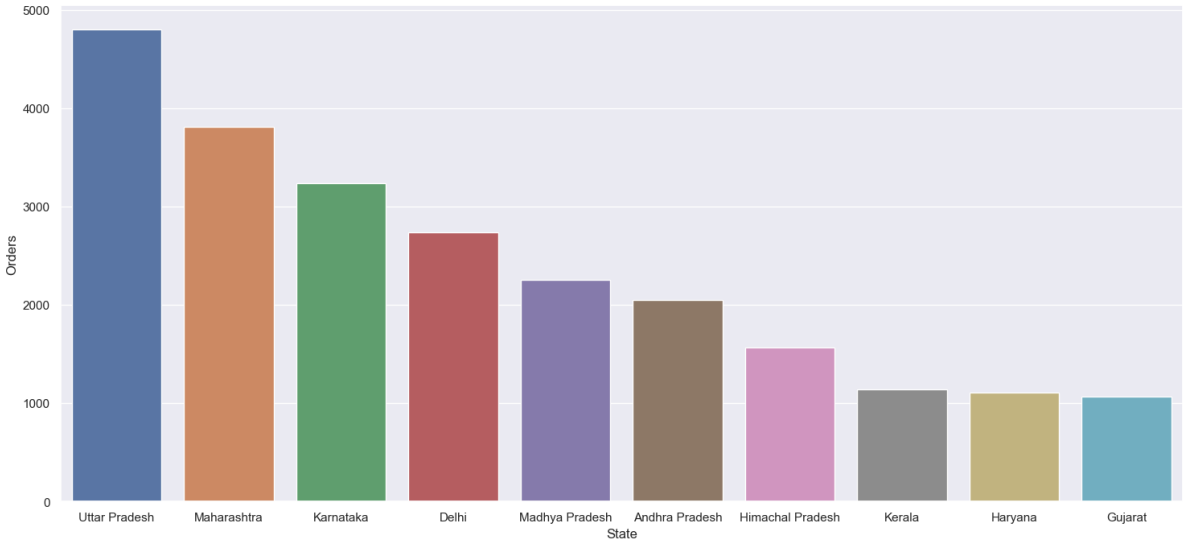


From above data we can see that maximum buyers are female with the age group 26-35 years.

State

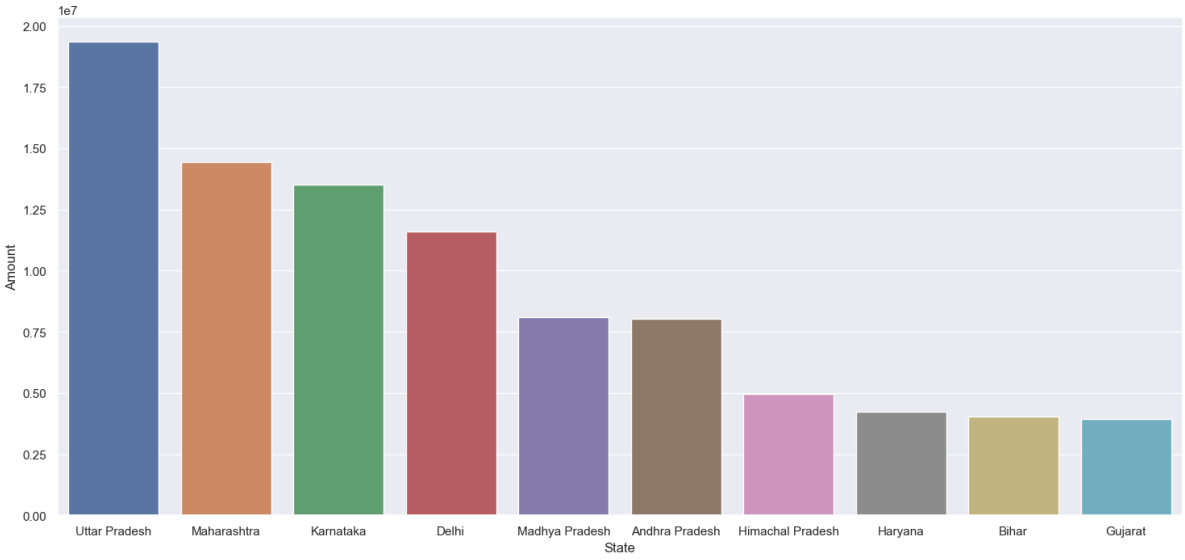
In [52]:
```
# TOTAL COUNT OF ORDERS FROM TOP 10 STATES.

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

sns.set(rc={'figure.figsize':(18,8)}) # USING figure.figsize FOR FITTING THE SIZE (
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

Out[52]:
```
<Axes: xlabel='State', ylabel='Orders'>
```



In [56]:
```
# TOTAL AMOUNT/SALES FROM TOP 10 SALES.

sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_values(by=
#USING head() FOR GETTING TOP 10.
sns.set(rc={'figure.figsize':(18,8)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

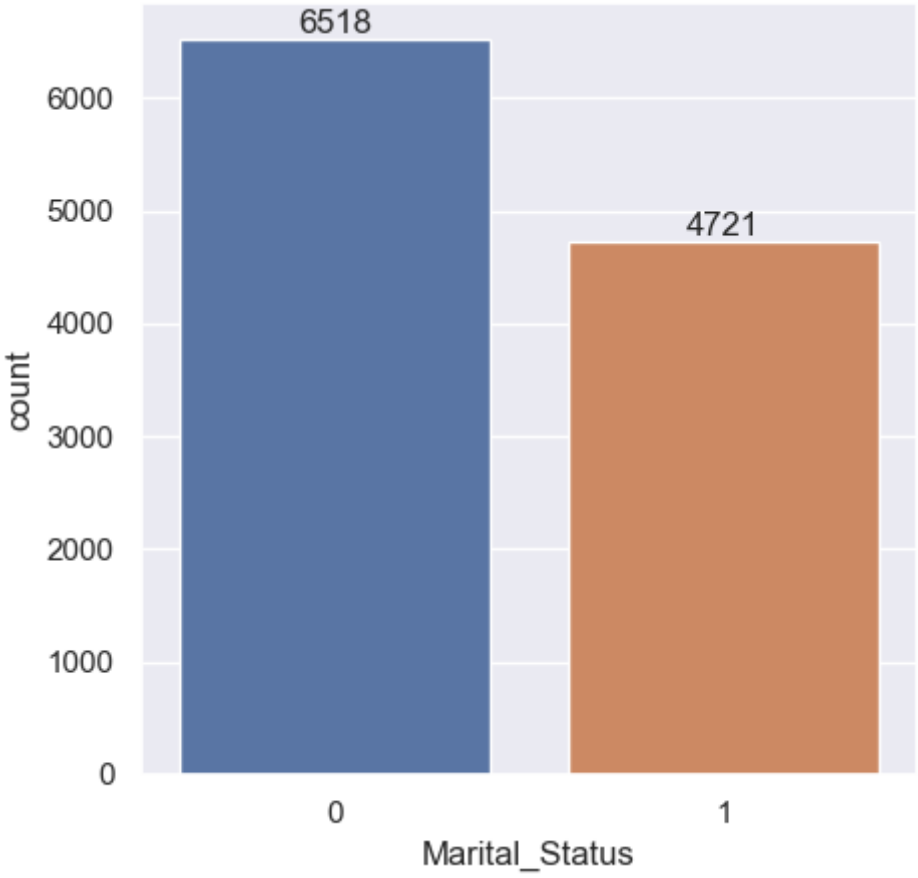
Out[56]:
```
<Axes: xlabel='State', ylabel='Amount'>
```



Top 3 state contributers are from Uttar Pradesh, Maharashtra and Karnataka respectively

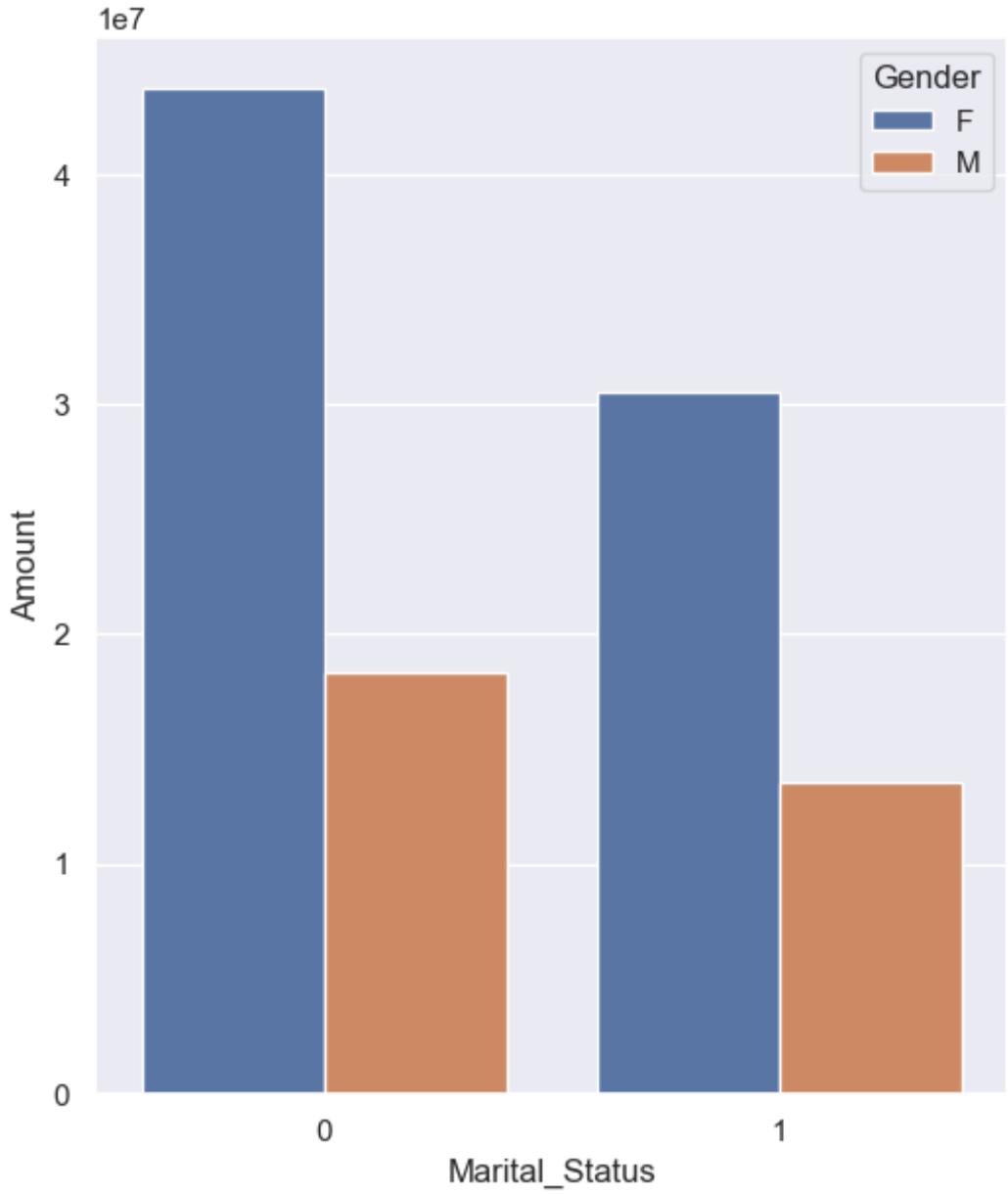
Marital Status.

In [61]:
```
ax = sns.countplot(data = df, x = 'Marital_Status')

sns.set(rc={'figure.figsize':(6,7)})
for bars in ax.containers:
    ax.bar_label(bars)
```



In [67]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].su

sns.set(rc={'figure.figsize':(6,7)})
sns.barplot(data = sales_state, x = 'Marital_Status',y= 'Amount', hue='Gender')
```

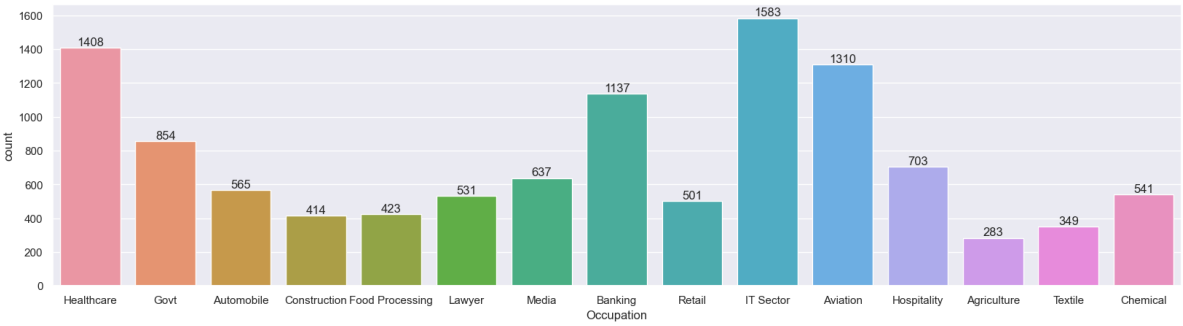
Out[67]:
```
<Axes: xlabel='Marital_Status', ylabel='Amount'>
```



From above data we can see that married womens has contributed highly in terms of purchasing power.

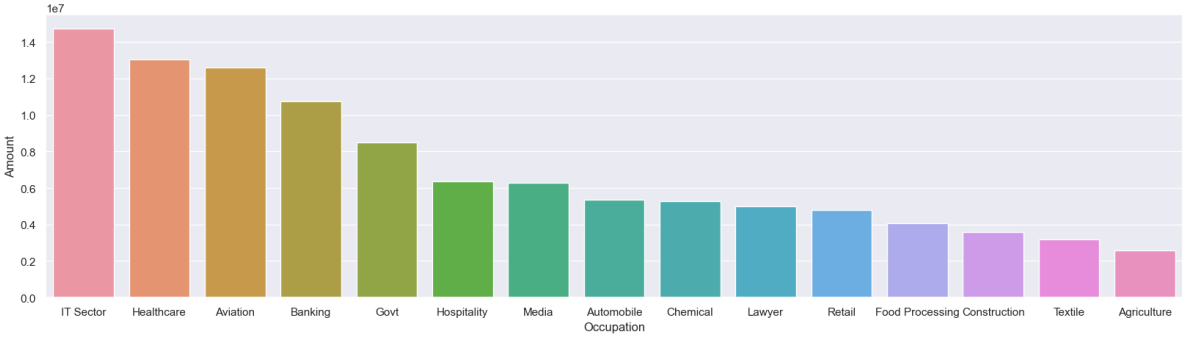
Occupation

In [70]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

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



In [73]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_valu

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

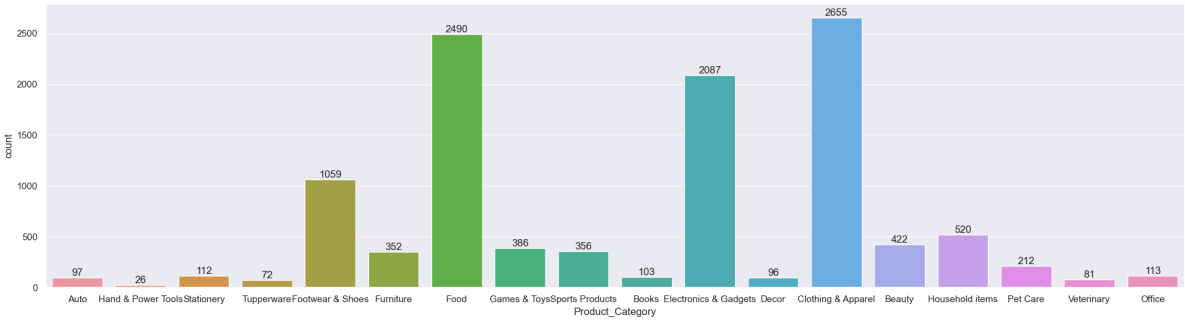
Out[73]:
```
<Axes: xlabel='Occupation', ylabel='Amount'>
```



Top Contributers are working in IT, Healthcare and Aviation sector

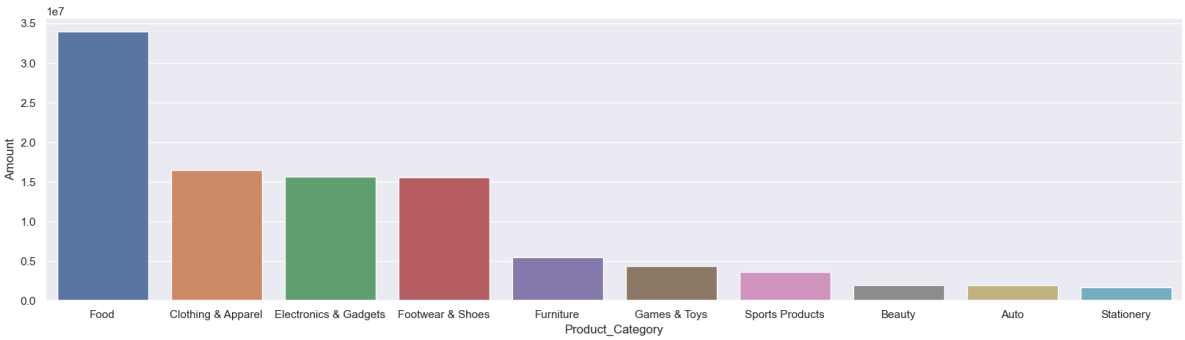
Product Category

In [79]:
```
sns.set(rc={'figure.figsize':(24,6)})
ax = sns.countplot(data = df, x = 'Product_Category')

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



In [82]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().sor

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

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



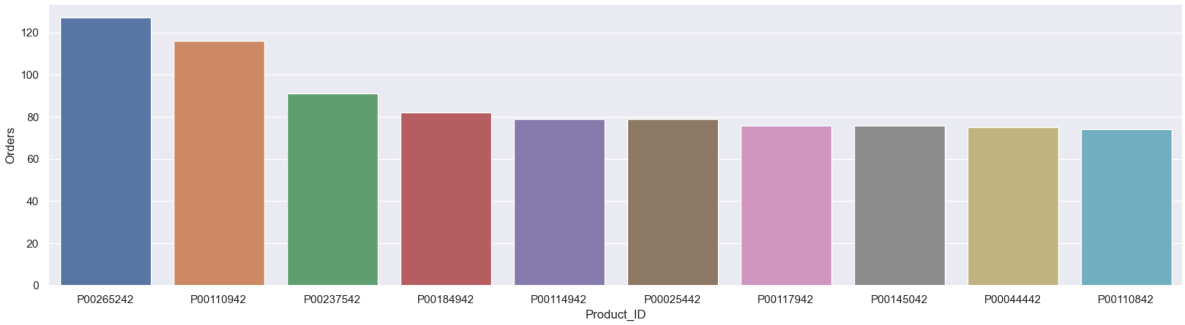
We can see that most of the sold products are from Food, Clothing and Electronics category

In [85]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_valu
```

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

Out[85]:

```
<Axes: xlabel='Product_ID', ylabel='Orders'>
```



In [88]:

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

Out[88]:

```
<Axes: xlabel='Product_ID'>
```