In [2]:
```
# import python libraries

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

In [4]:
```
# import csv file
df = pd.read_csv('Diwali Sales Data.csv', encoding= 'unicode_escape')
```

In [5]:
```
df.shape
```

Out[5]: (11251, 15)

In [6]:
```
df.head()
```

Out[6]:

| | User_ID | Cust_name | Product_ID | Gender | Age Group | Age | Marital_Status | Stat |
|---|---|---|---|---|---|---|---|---|
| **0** | 1002903 | Sanskriti | P00125942 | F | 26-35 | 28 | 0 | Maharashtı |
| **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 | Gujara |

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
```

In [8]:
```
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 [9]:
```
#drop unrelated/blank columns
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

In [10]:
```
#check for null values
pd.isnull(df).sum()
```

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
```

In [11]:
```
#check how many rows and columns
df.shape
```

Out[11]:
```
(11251, 13)
```

In [12]:
```
# drop null values(for save any operation we have to put inplace=true!
df.dropna(inplace=True)
```

In [13]:
```
df.shape
```

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

In [14]:
```
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              0
dtype: int64
```

In [15]:
```
# change data type
df['Amount'] = df['Amount'].astype('int')
```

In [16]:
```
df['Amount'].dtypes
```

Out[16]:
```
dtype('int32')
```

In [17]:
```
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 [18]:
```
#Check column
df.columns
```

Out[18]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [19]:
```
#rename column
df.rename(columns= {'Marital_Status':'vibaha',"Cust_name":"customer_name"})
```

Out[19]:

| | User_ID | customer_name | Product_ID | Gender | Age Group | Age | vibaha | S |
|---|---|---|---|---|---|---|---|---|
| **0** | 1002903 | Sanskriti | P00125942 | F | 26-35 | 28 | 0 | Maharas |
| **1** | 1000732 | Kartik | P00110942 | F | 26-35 | 35 | 1 | Andhra Prac |
| **2** | 1001990 | Bindu | P00118542 | F | 26-35 | 35 | 1 | Uttar Prac |
| **3** | 1001425 | Sudevi | P00237842 | M | 0-17 | 16 | 0 | Karna |
| **4** | 1000588 | Joni | P00057942 | M | 26-35 | 28 | 1 | Guj |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **11246** | 1000695 | Manning | P00296942 | M | 18-25 | 19 | 1 | Maharas |
| **11247** | 1004089 | Reichenbach | P00171342 | M | 26-35 | 33 | 0 | Hary |
| **11248** | 1001209 | Oshin | P00201342 | F | 36-45 | 40 | 0 | Mac Prac |
| **11249** | 1004023 | Noonan | P00059442 | M | 36-45 | 37 | 0 | Karna |
| **11250** | 1002744 | Brumley | P00281742 | F | 18-25 | 19 | 0 | Maharas |

11239 rows × 13 columns

In [20]:
```
# describe() method returns description of the data in the DataFrame (i.e. count
df.describe()
```

Out[20]:

| | 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 [21]:
```
# use describe() for specific columns
df[['Age', 'Orders', 'Amount']].describe()
```

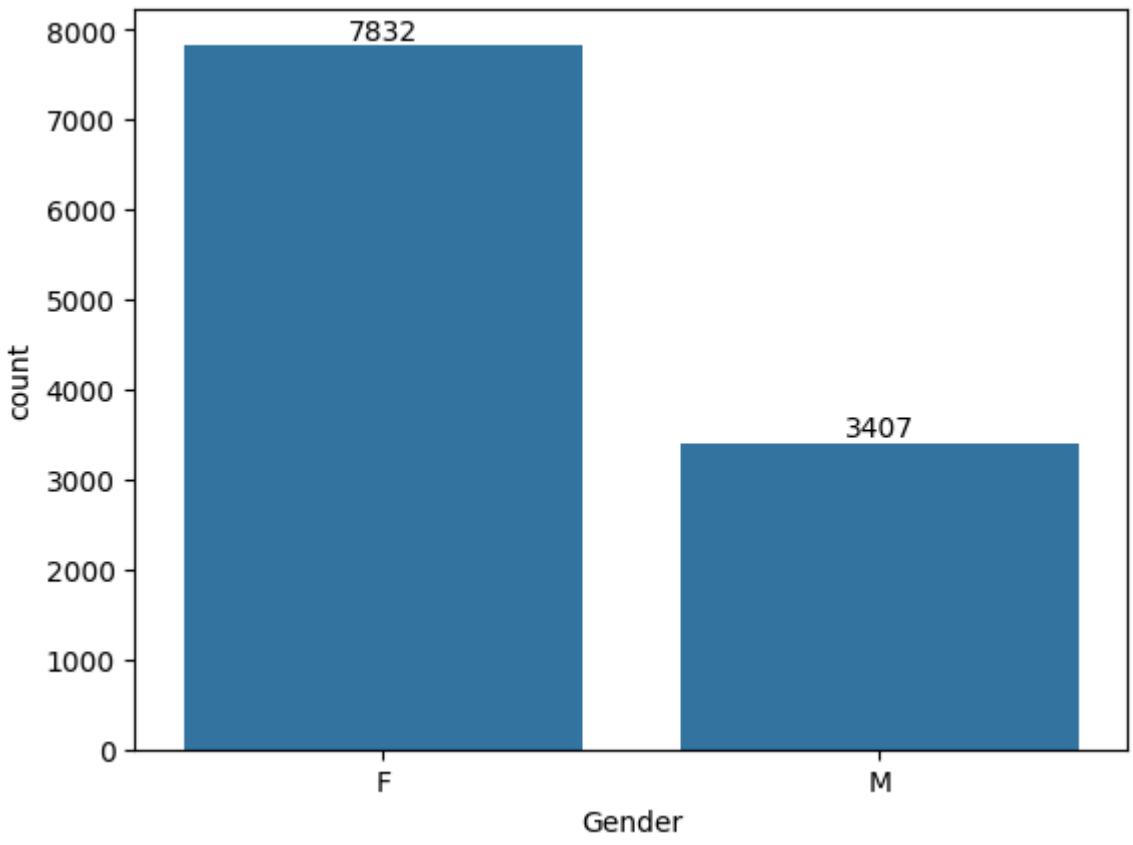
Out[21]:

| | 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

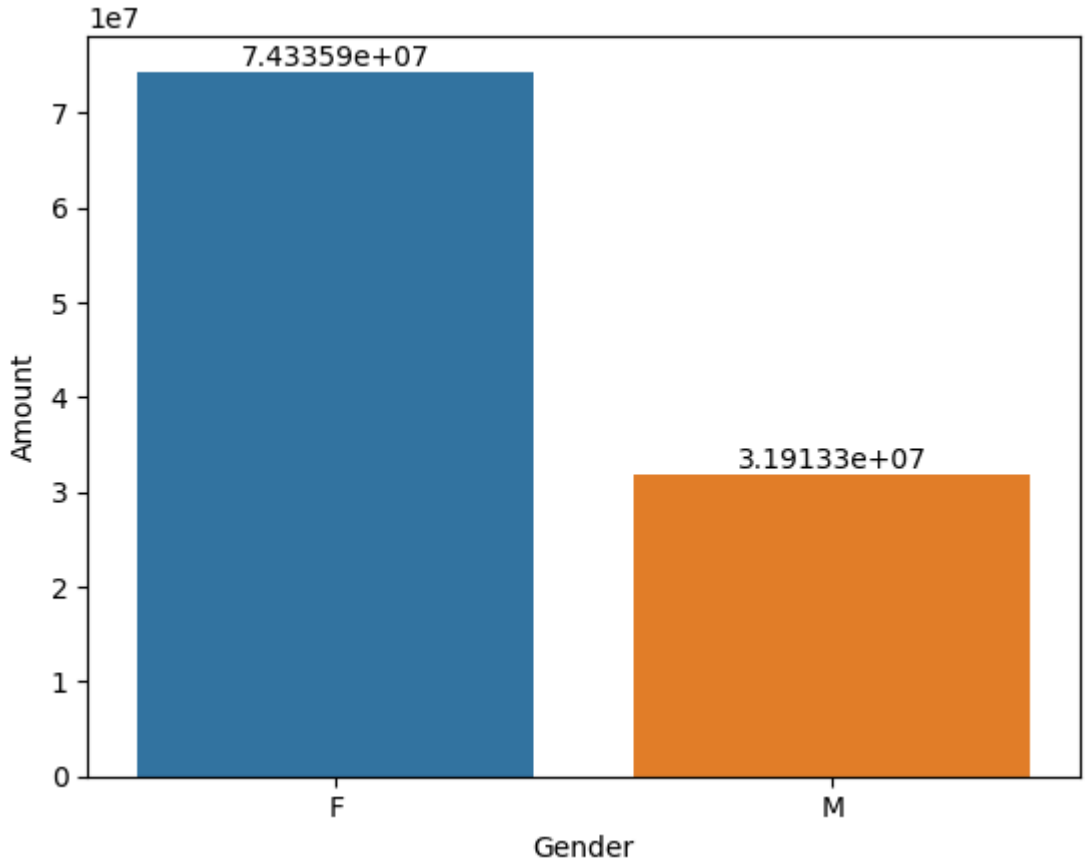
## Gender

In [22]:
```
# plotting a bar chart for Gender and it's count(for getting amount on above bar

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

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

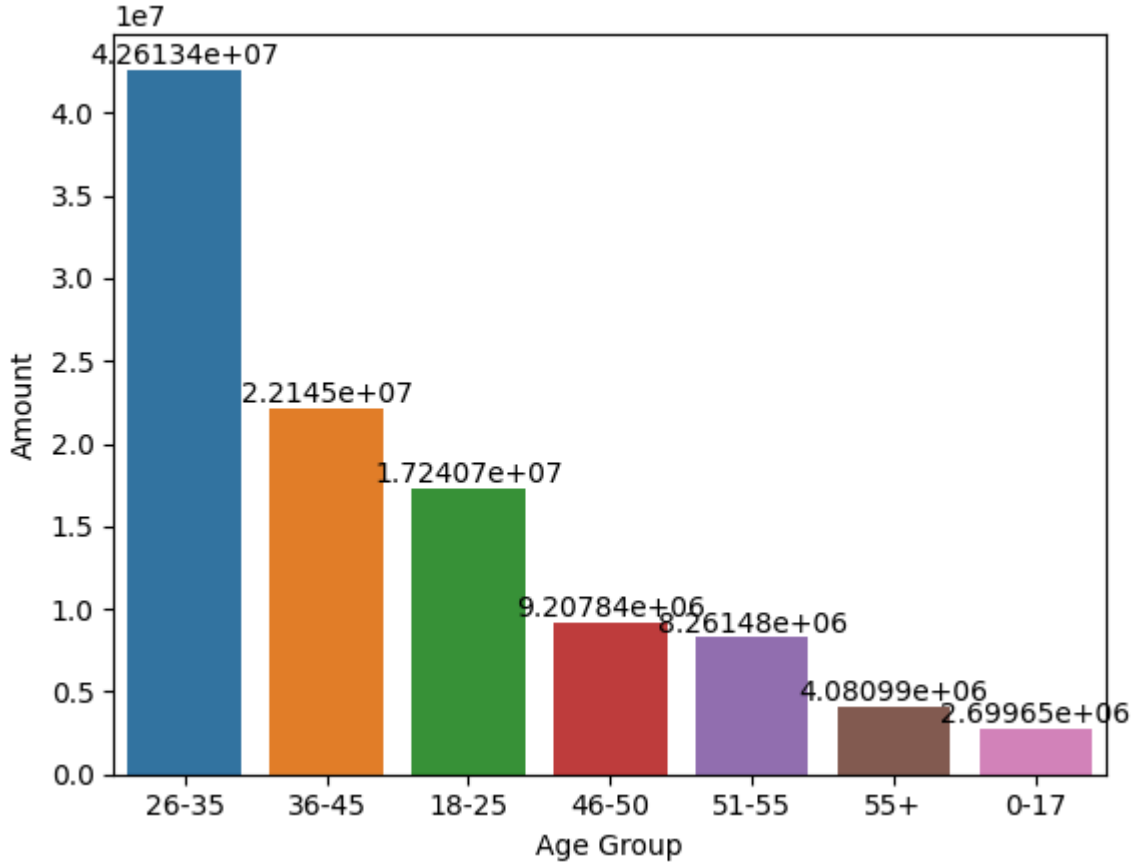



In [25]:
```
# plotting a bar chart for gender vs total amount
```

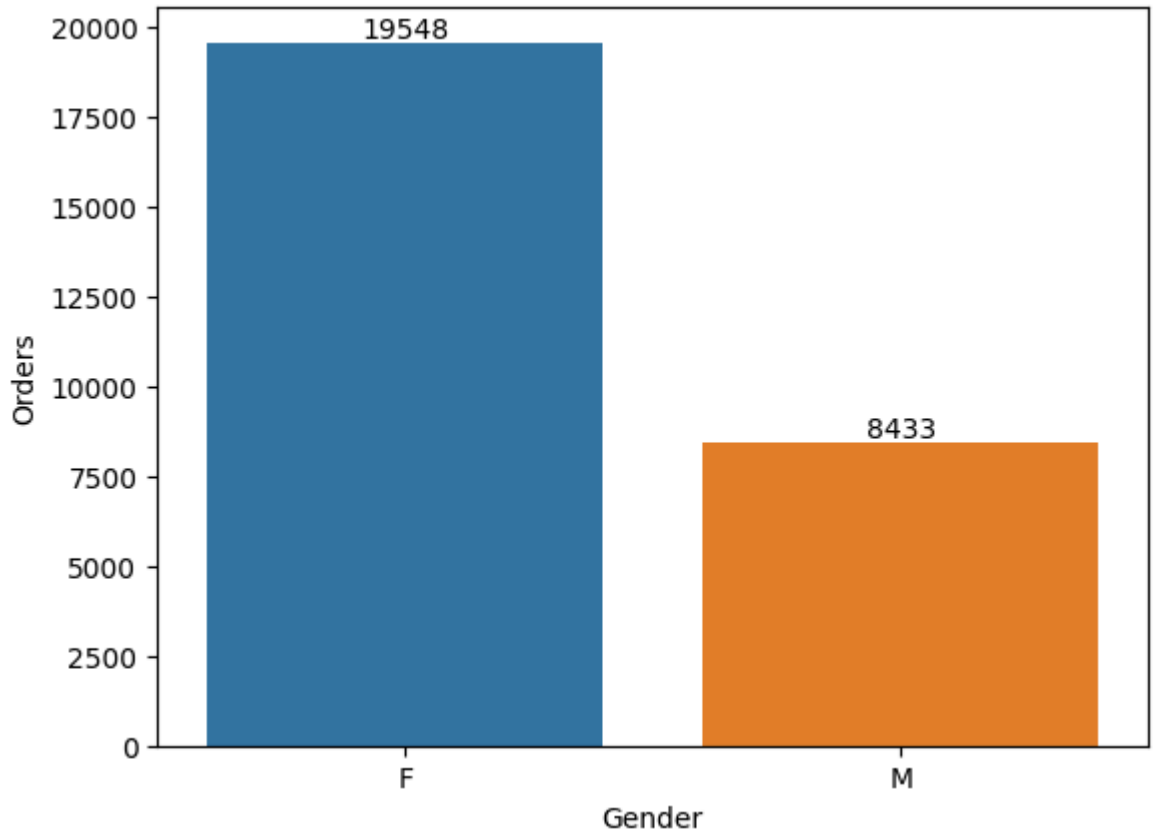
```
sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(b

ab = sns.barplot(x = 'Gender',y= 'Amount' ,data = sales_gen, hue="Gender")
for bars in ab.containers:
    ab.bar_label(bars)
plt.show()
```




In [26]:
```
#Ploting a bar chart for age group vs amount!
ay = df.groupby(["Age Group"], as_index=False)["Amount"].sum().sort_values(by="A
dh = sns.barplot(x="Age Group", y="Amount", data=ay,hue="Age Group")
for bars in dh.containers:
    dh.bar_label(bars)
plt.show()
```

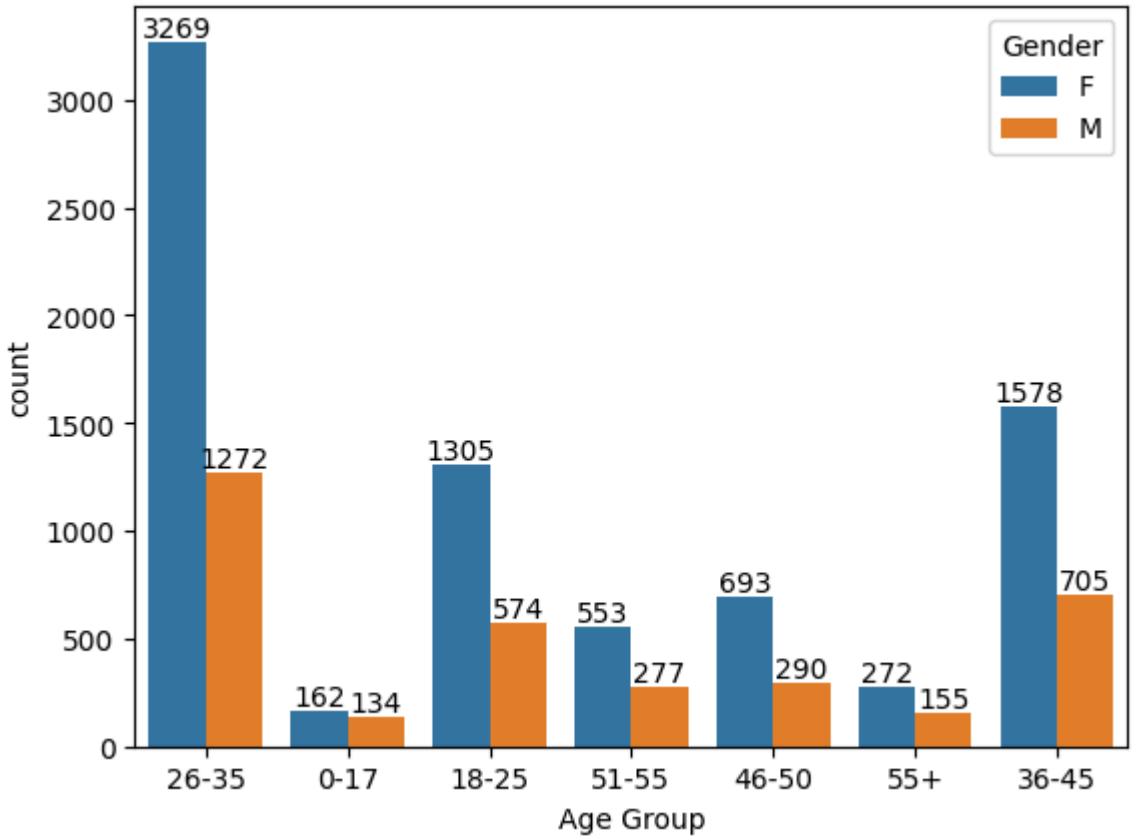



In [23]:
```
#create a bar plot for genderVS orders
ab = df.groupby(["Gender"], as_index=False)["Orders"].sum().sort_values(by="Orde
ac = sns.barplot(x="Gender", y="Orders", data=ab, hue="Gender")
for i in ac.containers:
    ac.bar_label(i)
```

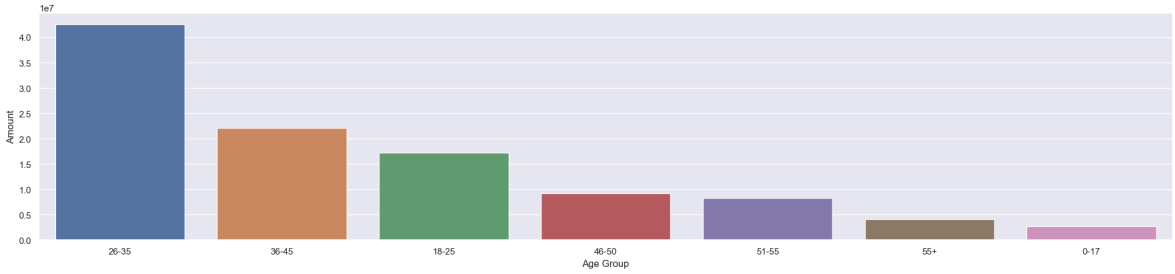


## Age

In [24]:
```
#plot a bar chart for age group vs gender(find out maximum age group of customer
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

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




In [31]:
```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_value

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

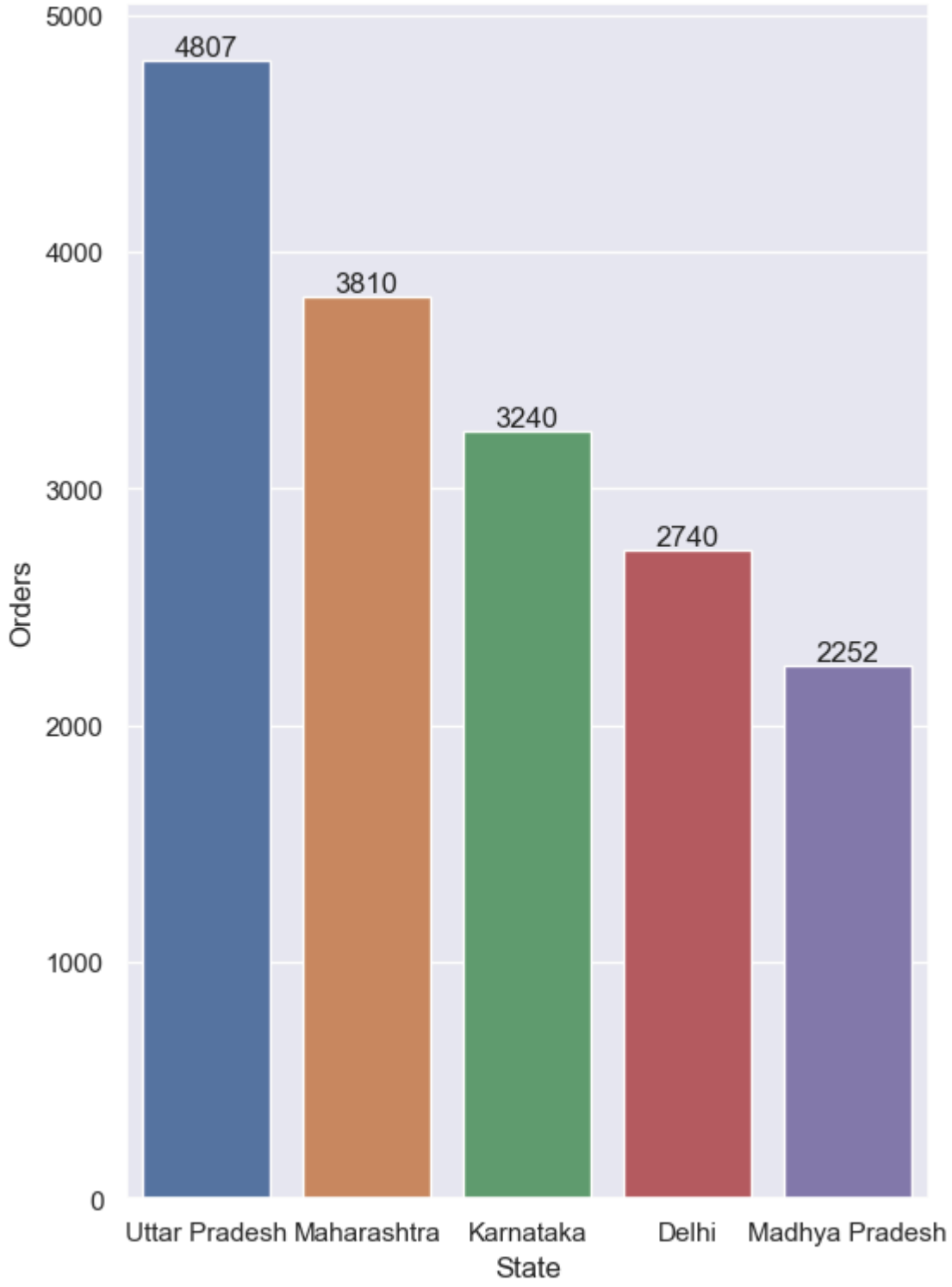



*From above graphs we can see that most of the buyers are of age group between 26-35 yrs female*

## State

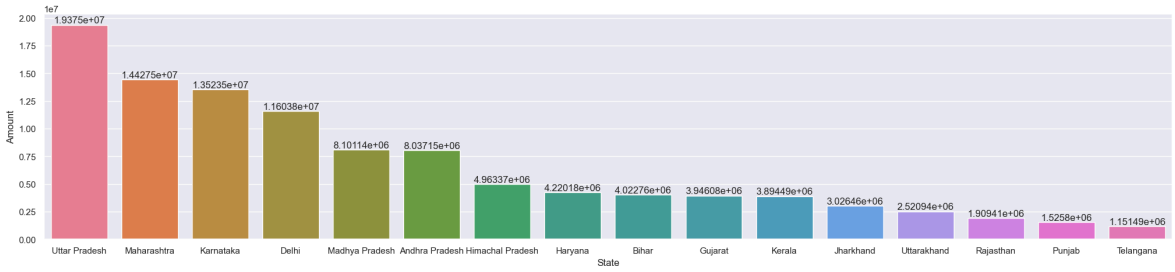
In [35]:
```
# total number of orders from top 10 states(for set the length and with we use t

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

```
plt.figure(figsize=(6,9))
af = sns.barplot(data = sales_state, x = 'State',y= 'Orders', hue="State")
for i in af.containers:
    af.bar_label(i)

plt.show()
```

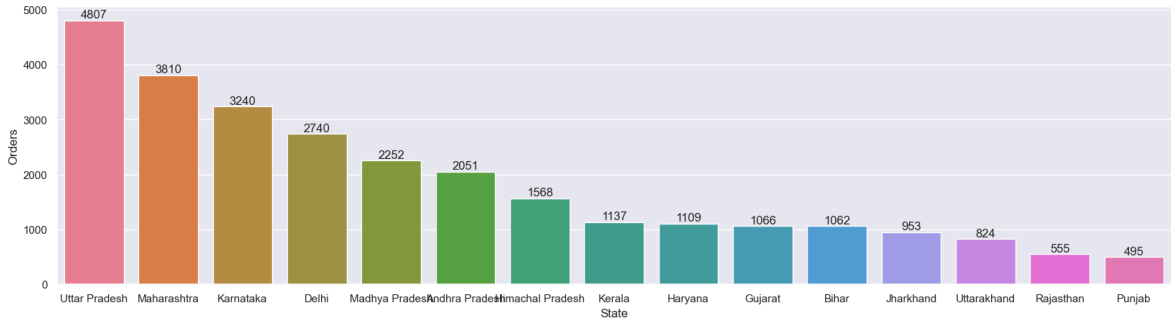



In [32]:
```
# total amount/sales from top 10 states

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

sns.set(rc={'figure.figsize':(25,5)})
al = sns.barplot(data = sales_state, x = 'State',y= 'Amount', hue="State")
```

```
for i in al.containers:
    al.bar_label(i)
```

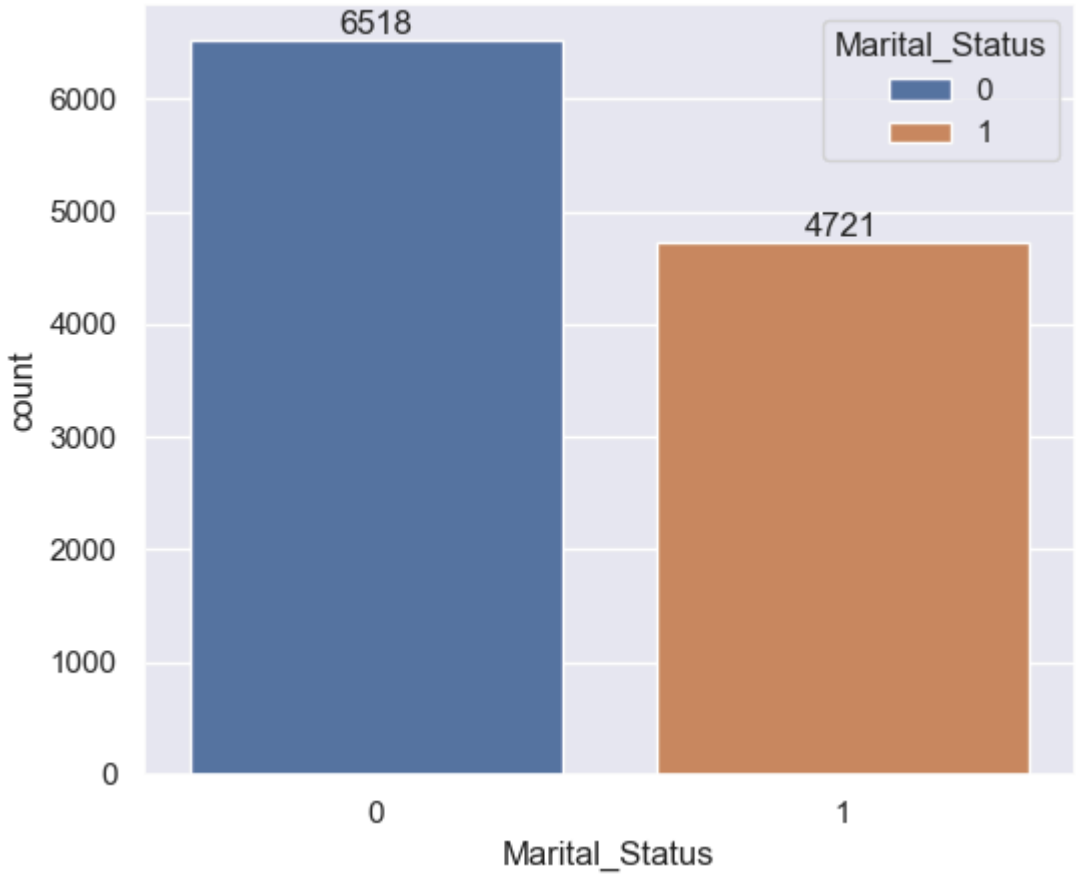



In [36]:
```
#create a bar plot for state vs orders(find out maximum orders state)
ff = df.groupby(["State"], as_index=False)["Orders"].sum().sort_values(by="Order
sns.set(rc={"figure.figsize":(20,5)})
ak = sns.barplot(data=ff,x="State", y="Orders", hue="State")
for i in ak.containers:
    ak.bar_label(i)
```

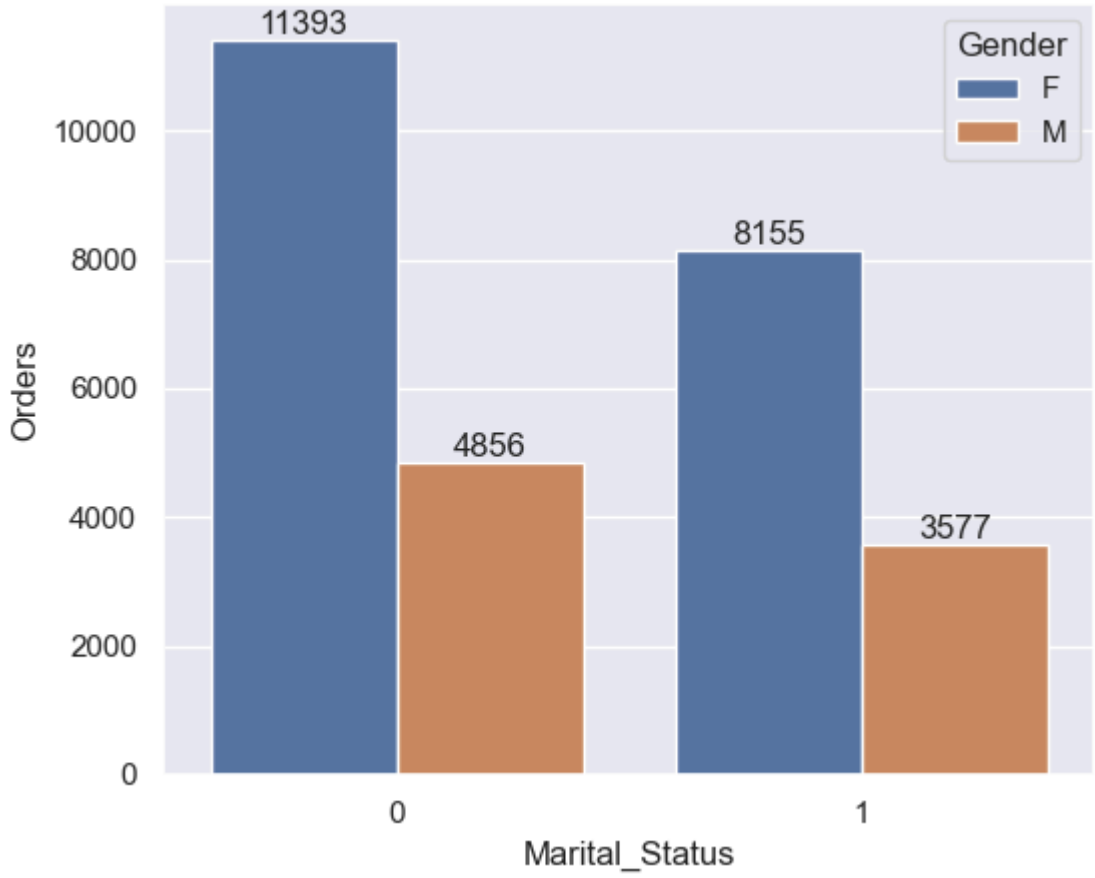



## Marital Status

In [38]:
```
#count martial status
ax = sns.countplot(data = df, x = 'Marital_Status', hue="Marital_Status")

sns.set(rc={'figure.figsize':(25,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```

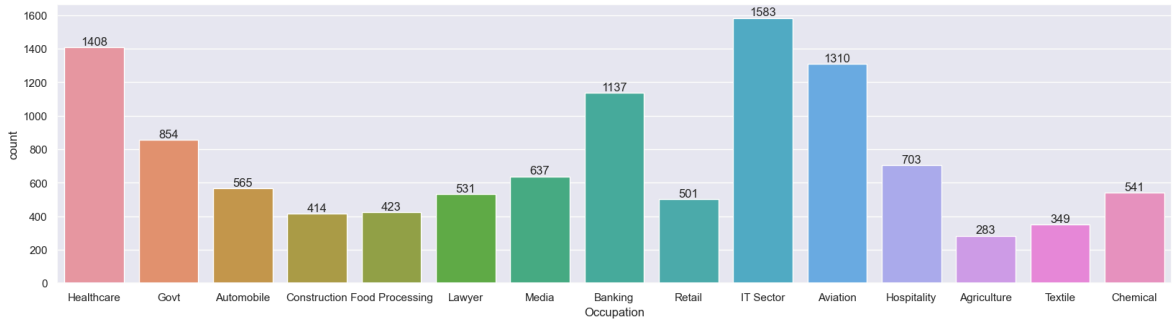



In [39]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Orders']

sns.set(rc={'figure.figsize':(6,5)})
aj = sns.barplot(data = sales_state, x = 'Marital_Status',y= 'Orders', hue='Gend
for i in aj.containers:
    aj.bar_label(i)
plt.show()
```

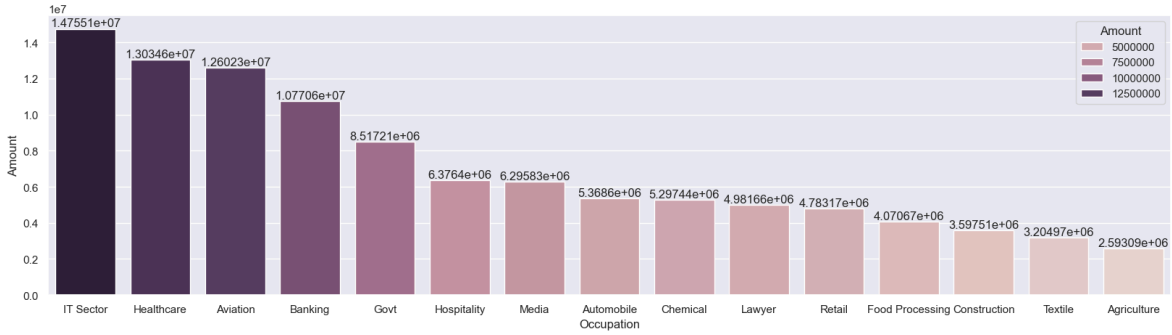



## Occupation

In [23]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

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



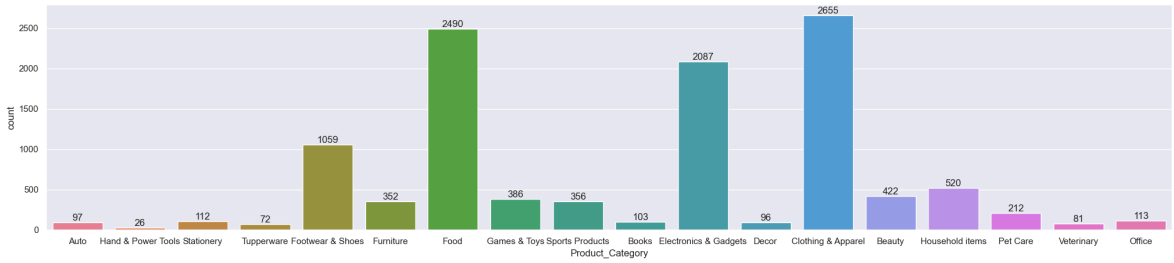

In [40]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_va

sns.set(rc={'figure.figsize':(20,5)})
pl = sns.barplot(data = sales_state, x = 'Occupation',y= 'Amount', hue="Amount")
for i in pl.containers:
    pl.bar_label(i)
plt.show()
```



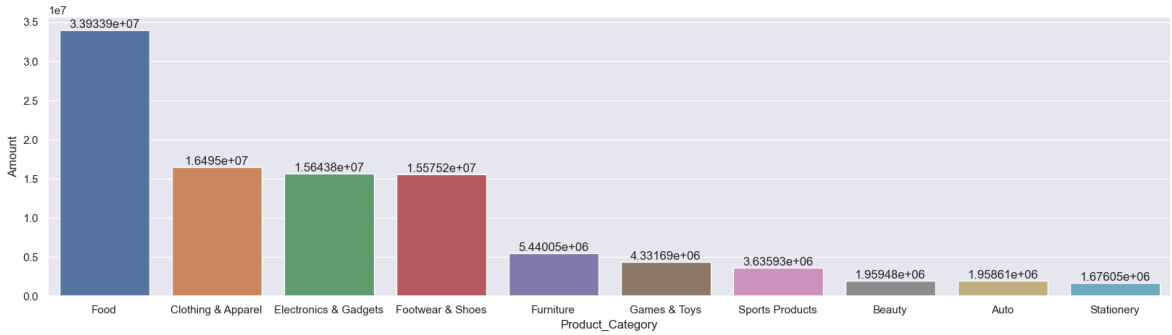

*From above graphs we can see that most of the buyers are working in IT, Healthcare and Aviation sector*

## Product Category

In [41]:
```
sns.set(rc={'figure.figsize':(25,5)})
ax = sns.countplot(data = df, x = 'Product_Category', hue="Product_Category")

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



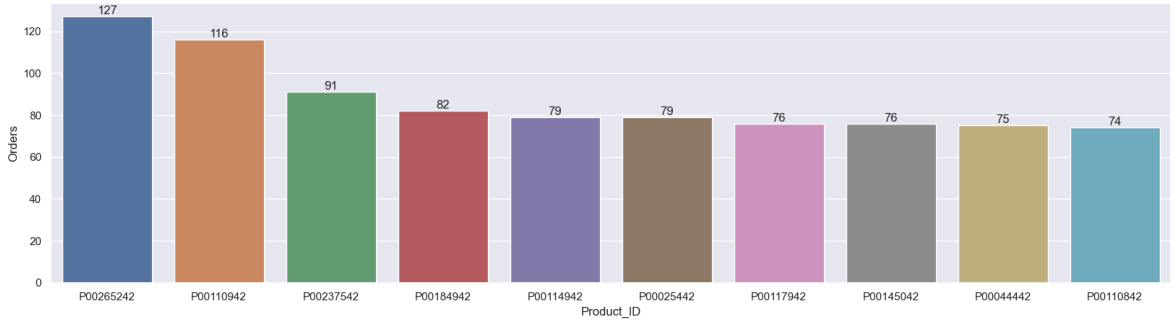

In [42]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().s

sns.set(rc={'figure.figsize':(20,5)})
ff = sns.barplot(data = sales_state, x = 'Product_Category',y= 'Amount', hue="Pr
for i in ff.containers:
    ff.bar_label(i)
plt.show()
```



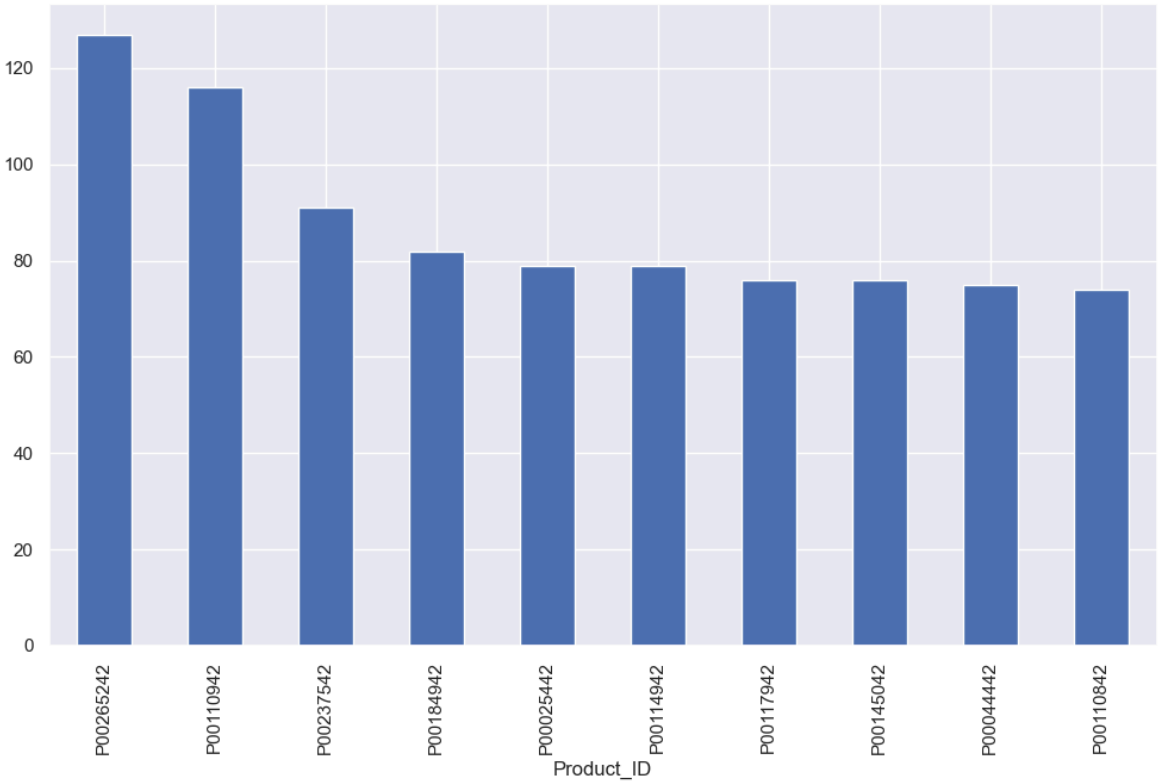

*From above graphs we can see that most of the sold products are from Food, Clothing and Electronics category*

In [44]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_va

sns.set(rc={'figure.figsize':(20,5)})
ss = sns.barplot(data = sales_state, x = 'Product_ID',y= 'Orders', hue="Product_
for i in ss.containers:
```

```
    ss.bar_label(i)
plt.show()
```




In [45]:
```
# 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=Fals
plt.show()
```




# 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*

Thank you!