Let's import python libraries First

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

import csv file

In [3]:
```
df=pd.read_csv('Sales_Data.csv',encoding = 'unicode_escape')
```

Show Top 5 Rows

In [4]:
```
df.head()
```

Out[4]:

	User_ID	Cust_name	Product_ID	Age	Age Group	Gender	State	Zone	Zipco
0	1003650.0	Meena	P00031142	26.0	26-35	Female	Andhra Pradesh	South	Na
1	1003829.0	Harsh	P00200842	34.0	26-35	M	Delhi	Central	Na
2	1000214.0	Raji	P00119142	20.0	18-25	Female	Andhra Pradesh	South	Na
3	1004035.0	Shiva	P00080342	20.0	18-25	Female	Andhra Pradesh	South	Na
4	1001680.0	Vasudev	P00324942	26.0	26-35	M	Andhra Pradesh	South	Na

Field details and Data type

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11247 entries, 0 to 11246
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11236 non-null  float64
 1   Cust_name         11236 non-null  object
 2   Product_ID        11236 non-null  object
 3   Age               11236 non-null  float64
 4   Age Group         11236 non-null  object
 5   Gender            11236 non-null  object
 6   State             11236 non-null  object
 7   Zone              11236 non-null  object
 8   Zipcode           0 non-null      float64
 9   Profession        11236 non-null  object
 10  Product_Category  11236 non-null  object
 11  Orders            11236 non-null  float64
 12  Amount            11236 non-null  float64
dtypes: float64(5), object(8)
memory usage: 1.1+ MB
```

Let's Start Data Cleaning

Deleting blank column

In [6]:
```
df.drop(['Zipcode'],axis=1,inplace=True)
```

List of column Available

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

Out[7]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Age', 'Age Group', 'Gender',
       'State', 'Zone', 'Profession', 'Product_Category', 'Orders', 'Amount'],
      dtype='object')
```

check for null values

In [8]:
```
pd.isnull(df).sum()
```

Out[8]:
```
User_ID             11
Cust_name           11
Product_ID          11
Age                 11
Age Group           11
Gender              11
State               11
Zone                11
Profession          11
Product_Category    11
Orders              11
Amount              11
dtype: int64
```

Drop null Values

In [9]:
```
df.dropna(how='any',inplace=True)
```

Replace Values for Gender Column

In [10]:
```
df['Gender']=df['Gender'].replace('M','Male')
```

View only Male Gender

In [11]:
```
df[df['Gender']=='Male']
```

Out[11]:

	User_ID	Cust_name	Product_ID	Age	Age Group	Gender	State	Zone
1	1003829.0	Harsh	P00200842	34.0	26-35	Male	Delhi	Central
4	1001680.0	Vasudev	P00324942	26.0	26-35	Male	Andhra Pradesh	South
5	1003858.0	Ravinath	P00293742	46.0	46-50	Male	Madhya Pradesh	Central
9	1001883.0	Rajveer	P00029842	54.0	51-55	Male	Uttar Pradesh	Central
10	1001883.0	Vinod	P00029842	54.0	51-55	Male	Uttar Pradesh	Central
...	...	...	...	...	...	...	...	...
11239	1005446.0	Sheetal	P00297742	53.0	51-55	Male	Gujarat	West
11240	1005446.0	Sheetal	P00297742	53.0	51-55	Male	Madhya Pradesh	Central
11242	1000695.0	Manning	P00296942	19.0	18-25	Male	Maharashtra	West
11243	1004089.0	Reichenbach	P00171342	33.0	26-35	Male	Haryana	Northern
11245	1004023.0	Noonan	P00059442	37.0	36-45	Male	Karnataka	South

3404 rows × 12 columns

Now Let's Learn EDA - Exploratory Data Analysis

describe() method returns description of the data in the DataFrame

In [12]:
```
df.describe()
```

Out[12]:

	User_ID	Age	Orders	Amount
count	1.123600e+04	11236.000000	11236.000000	11236.000000
mean	1.003005e+06	35.419989	3.500267	9449.925284
std	1.716252e+03	12.755547	1.714074	5218.902380
min	1.000001e+06	12.000000	1.000000	188.000000
25%	1.001494e+06	27.000000	2.000000	5443.000000
50%	1.003067e+06	33.000000	4.000000	8109.000000
75%	1.004430e+06	43.000000	5.000000	12671.500000
max	1.006040e+06	92.000000	6.000000	29000.000000

In [13]:

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

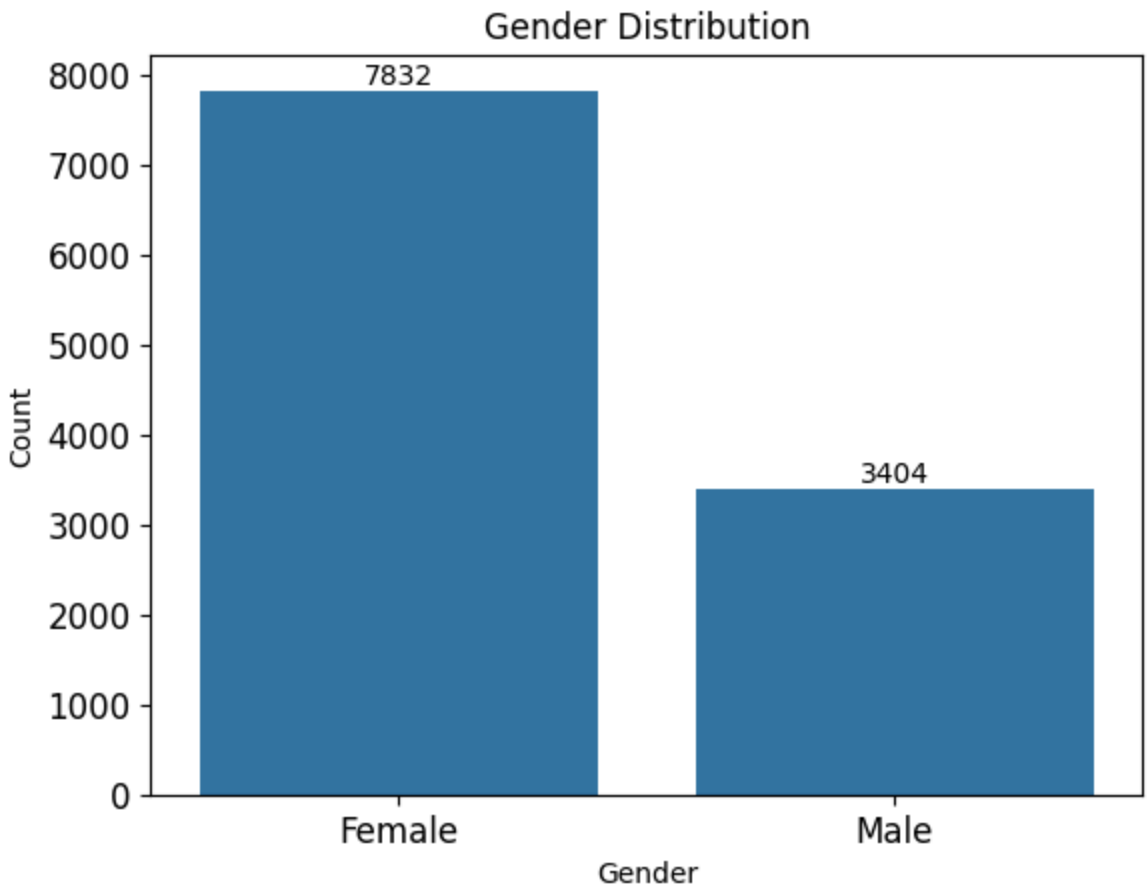
Out[13]:

	Age	Orders	Amount
count	11236.000000	11236.000000	11236.000000
mean	35.419989	3.500267	9449.925284
std	12.755547	1.714074	5218.902380
min	12.000000	1.000000	188.000000
25%	27.000000	2.000000	5443.000000
50%	33.000000	4.000000	8109.000000
75%	43.000000	5.000000	12671.500000
max	92.000000	6.000000	29000.000000

Total Transaction count by Gender Wise in bar char

In [14]:

```
ax=sns.countplot(x='Gender',data=df)
for bars in ax.containers:
    ax.bar_label(bars)
    plt.title('Gender Distribution')
    plt.xlabel('Gender')
    plt.ylabel('Count')
    plt.tick_params(axis='both',labelsize=12)
    plt.show()
```



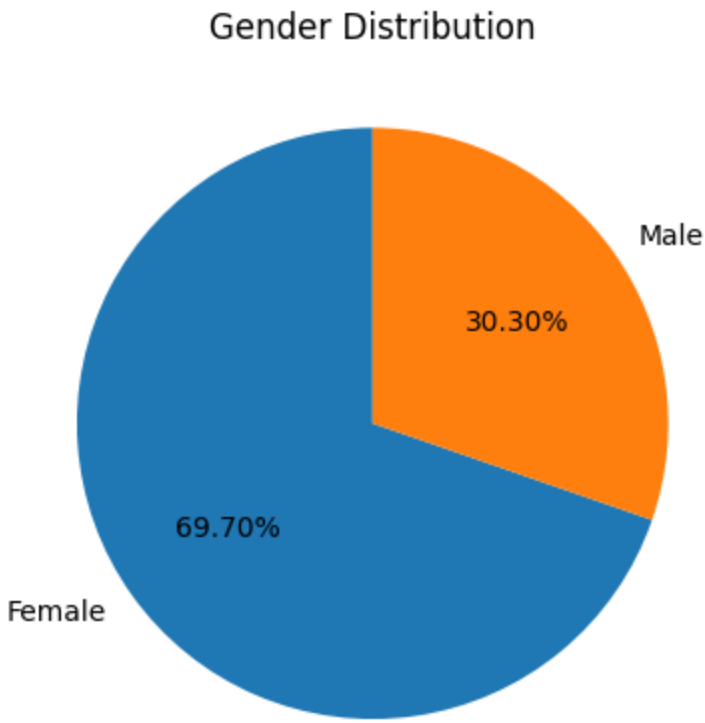
Total Transactions Count by Gender wise

In [15]:
```
Gender_counts = df['Gender'].value_counts()
print(Gender_counts)
```

```
Gender
Female    7832
Male      3404
Name: count, dtype: int64
```

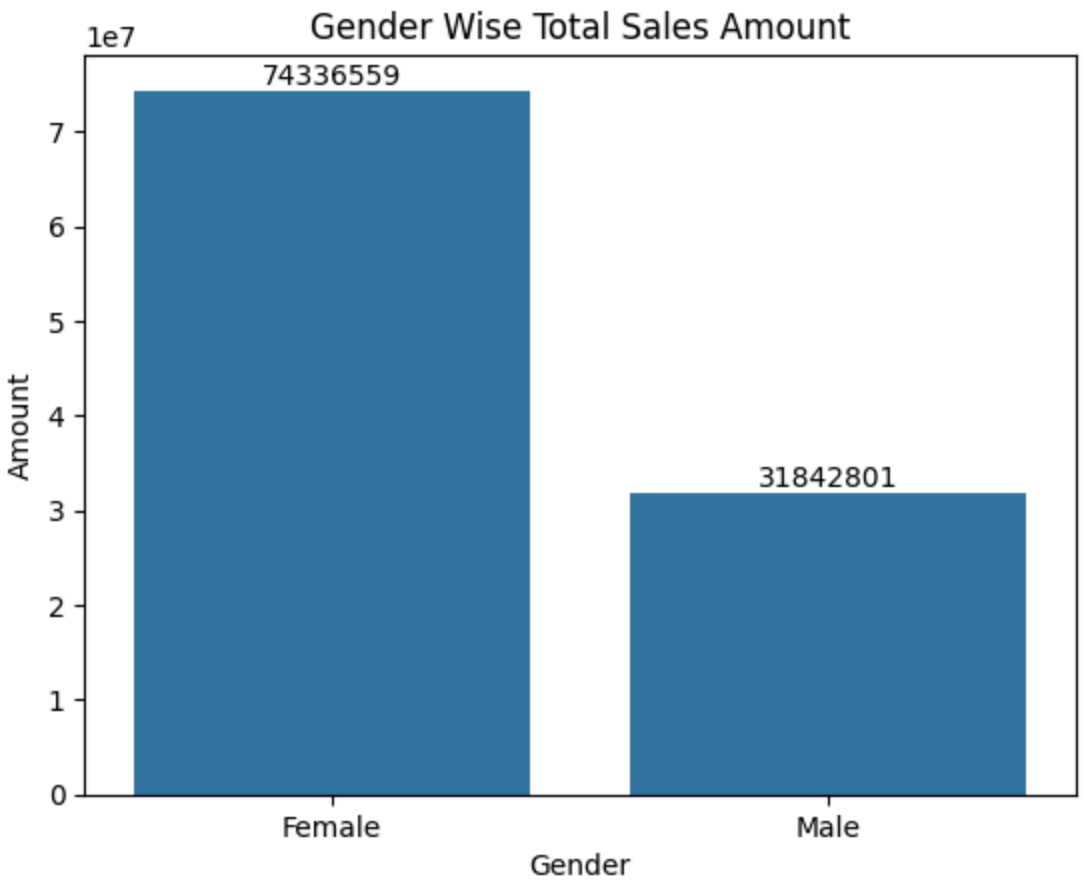
Total Transactions Count by Gender wise in Pie Chart

In [16]:
```
Gender_counts=df['Gender'].value_counts()
plt.pie(Gender_counts,labels=Gender_counts.index,autopct='%1.2f%%',startangle=90)
plt.title('Gender Distribution')
plt.show()
```



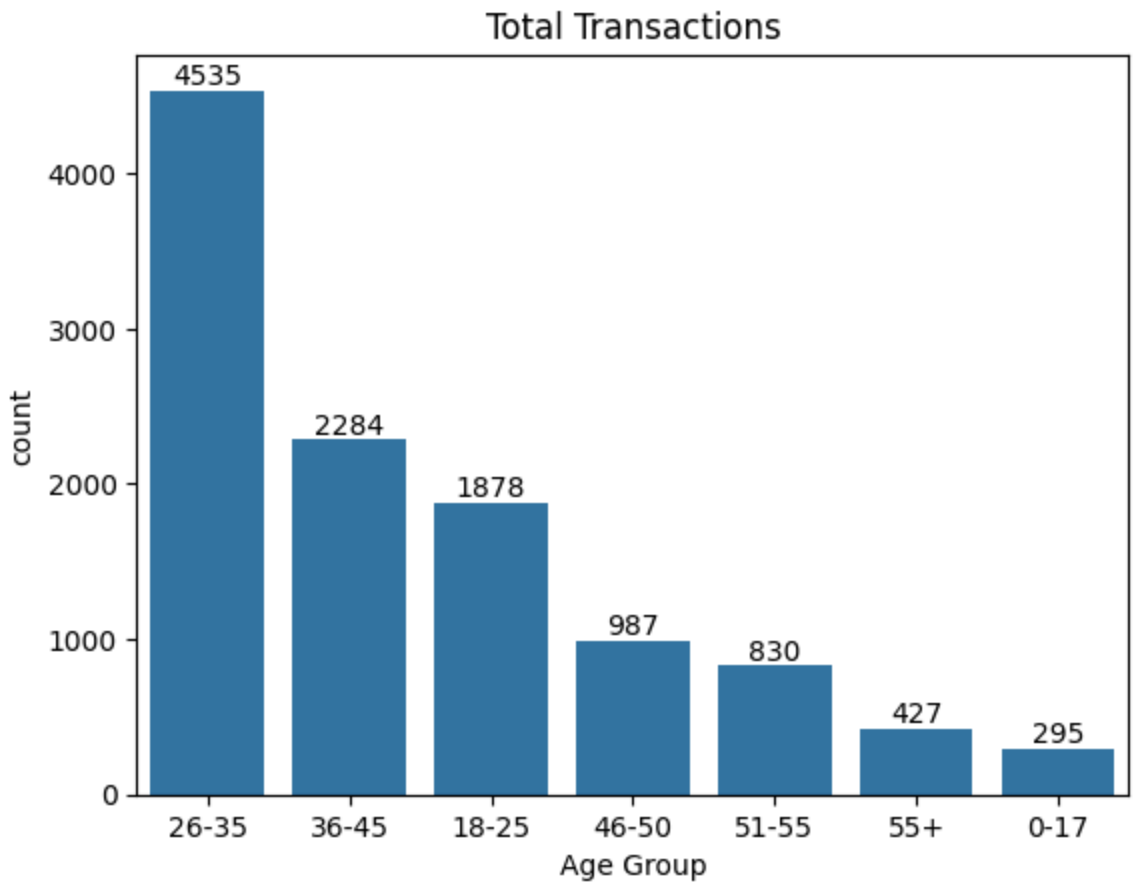
Gender wise Total Sales Amount

In [17]:
```
Gen_Wise_Sales=df.groupby('Gender',as_index=False)['Amount'].sum().sort_values(by='
ax=sns.barplot(x='Gender',y='Amount',data=Gen_Wise_Sales)
for bar in ax.containers:
    ax.bar_label(bar,fmt='%.0f')
    plt.title('Gender Wise Total Sales Amount')
    plt.show()
```



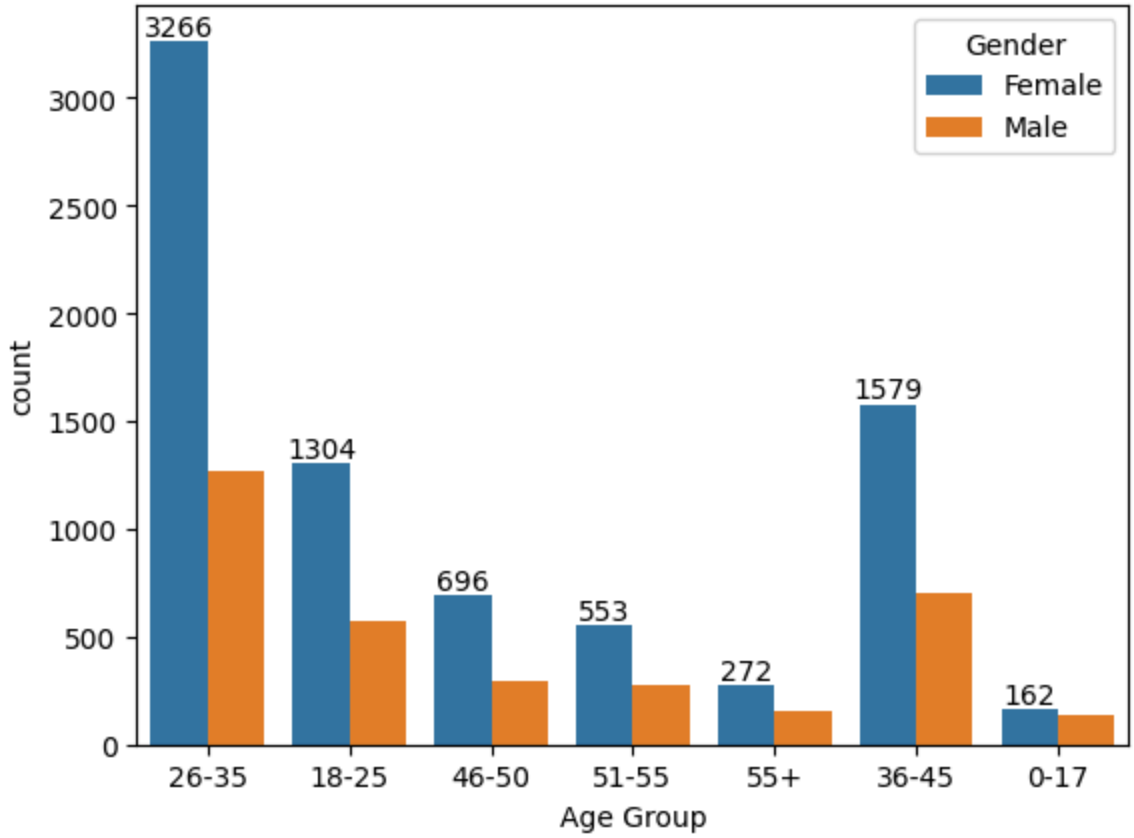
Age group wise total transactions

In [18]:
```
age_group_count=df['Age Group'].value_counts().sort_values(ascending=False)
sns_order=age_group_count.index
ax=sns.countplot(x='Age Group',data=df,order=sns_order)
for bar in ax.containers:
    ax.bar_label(bar)
    plt.title('Total Transactions')
    plt.show()
```



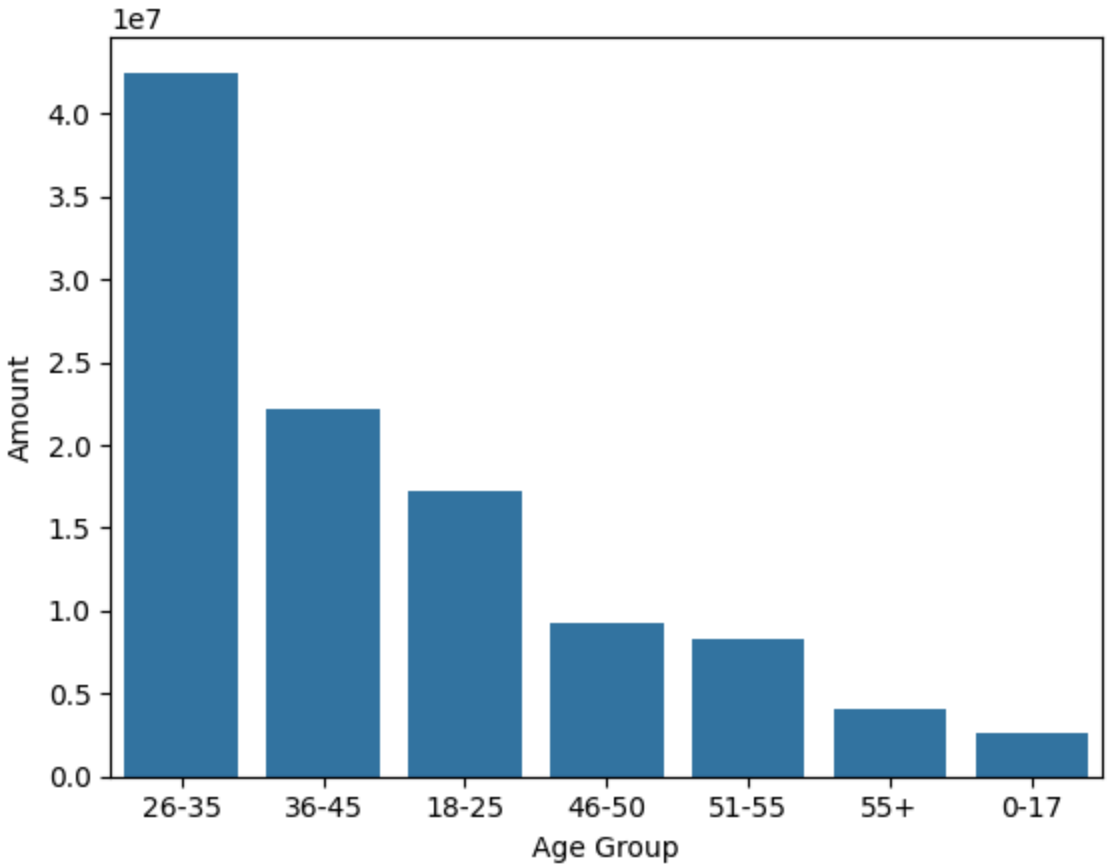
Age group and Gender wise Transactions distribution

In [19]:
```
ax=sns.countplot(data=df,x='Age Group',hue='Gender')
for bar in ax.containers:
    ax.bar_label(bar)
    plt.show()
```



Age Group wise Total Amount

In [20]:
```
sales_age=df.groupby(['Age Group'],as_index=False)['Amount'].sum().sort_values(by='
sns.barplot(x='Age Group',y='Amount',data=sales_age)
plt.show()
```



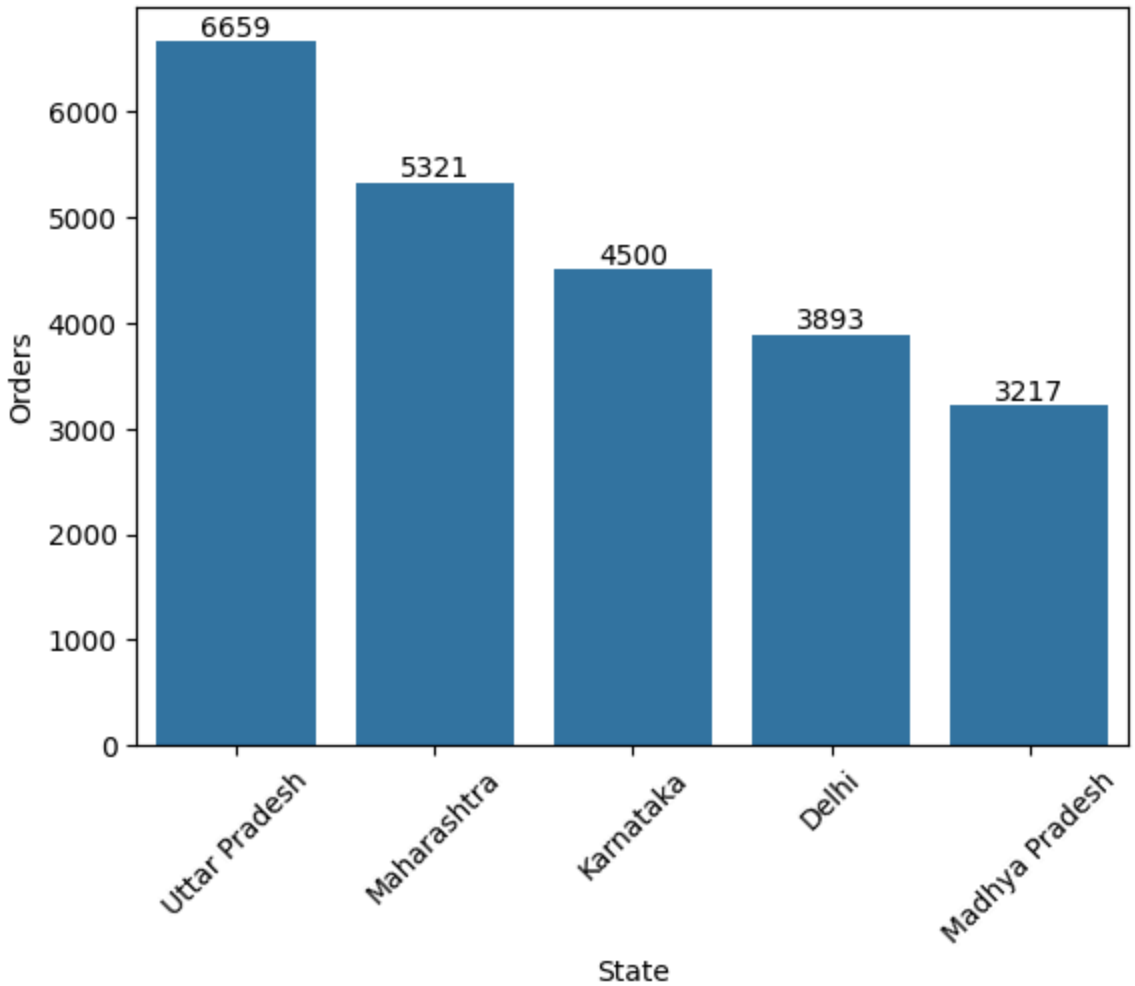
State wise analysis

In [21]:
```
order_state = df.groupby(['State'],as_index=False)['Orders'].sum().sort_values(by='
print(order_state)
```

```
            State  Orders
14   Uttar Pradesh  6659.0
10     Maharashtra  5321.0
7        Karnataka  4500.0
2            Delhi  3893.0
9   Madhya Pradesh  3217.0
```

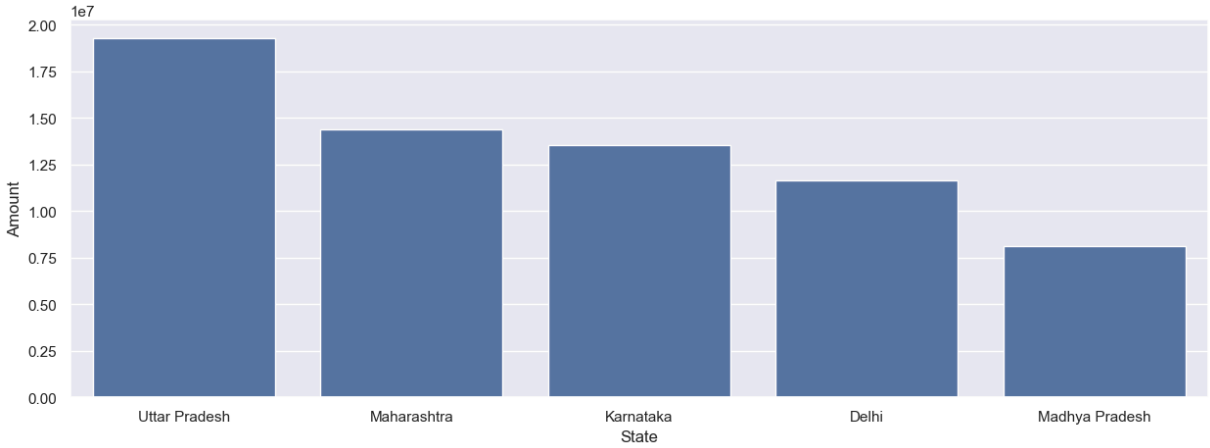
Order wise Top 5 state

In [22]:
```
order_state = df.groupby(['State'],as_index=False)['Orders'].sum().sort_values(by='
ax=sns.barplot(data=order_state,x='State',y='Orders')
for bar in ax.containers:
    ax.bar_label(bar)
    plt.xticks(rotation=45)
    plt.show()
```



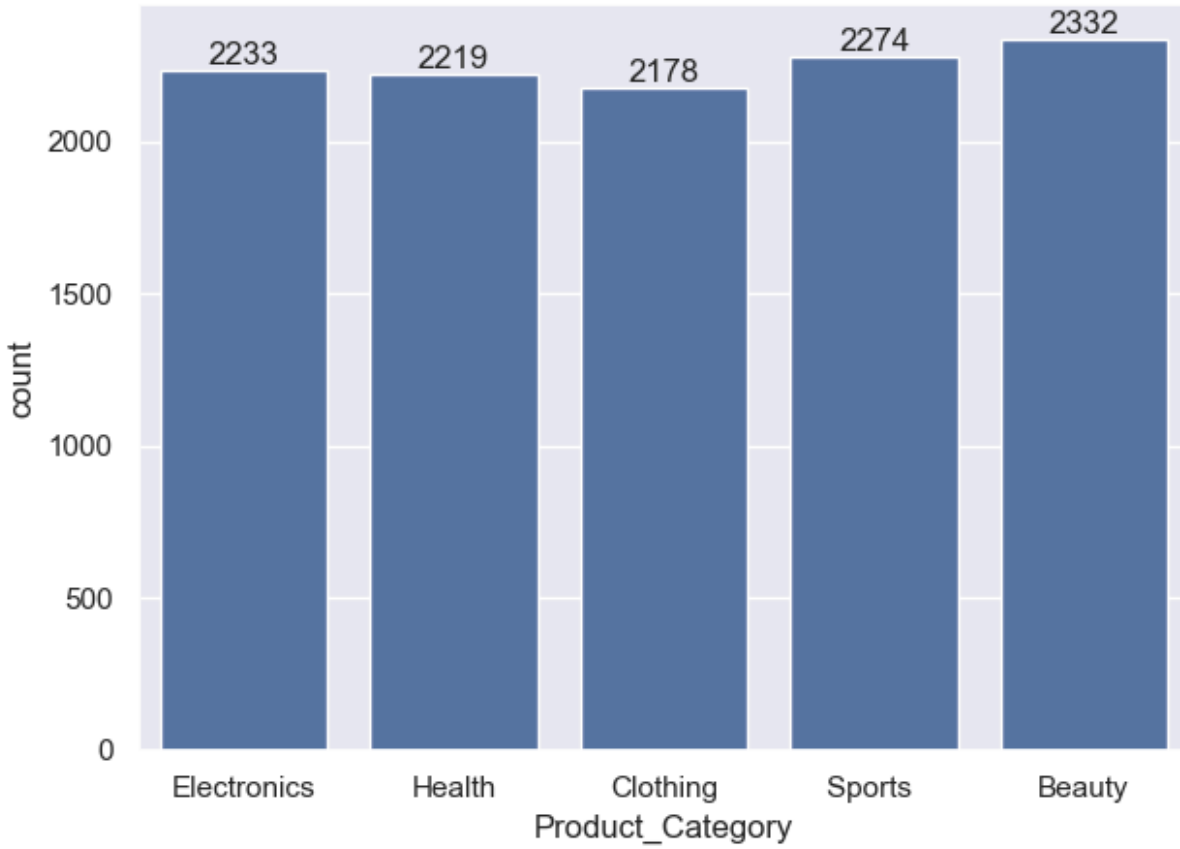
Amount Wise top 5 state

In [23]:
```
sales_state = df.groupby(['State'],as_index=False)['Amount'].sum().sort_values(by='
sns.set(rc={'figure.figsize':(15,5)})
ax=sns.barplot(x='State',y='Amount',data=sales_state)
plt.show()
```



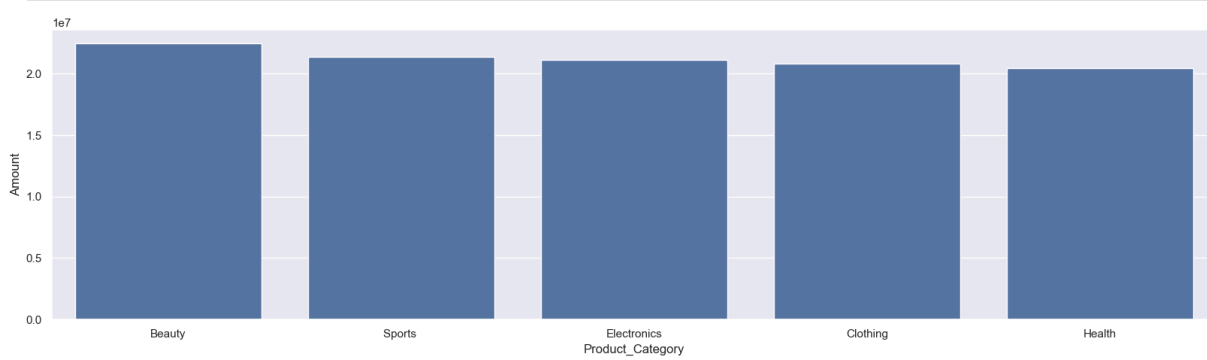
Product category wise Transactions count

In [27]:
```
ax = sns.countplot(data = df, x = 'Product_Category')
sns.set(rc={'figure.figsize':(8,6)})
for bars in ax.containers:
    ax.bar_label(bars)
plt.show()
```



Amount wise Product Category in Bar Chart

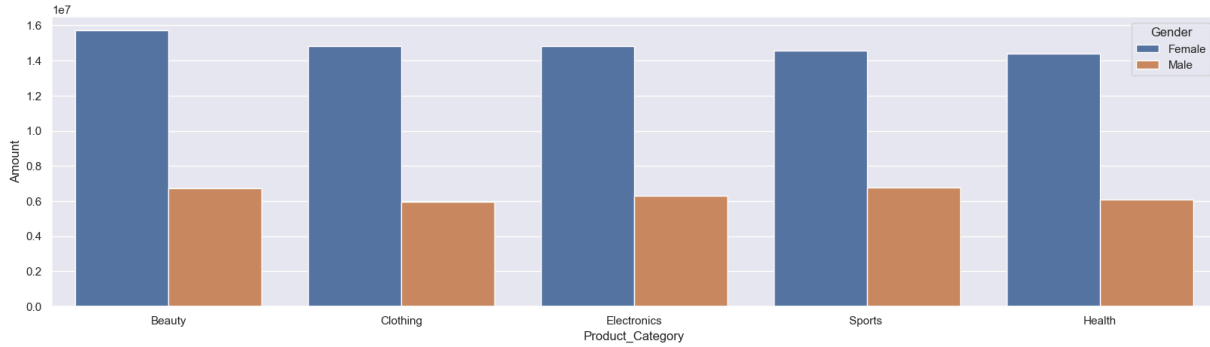
In [29]:
```
Salse_state = df.groupby(['Product_Category'],as_index=False)['Amount'].sum().sort_
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(x='Product_Category',data=Salse_state,y='Amount')
plt.show()
```



Product category and Gender wise Transactions count

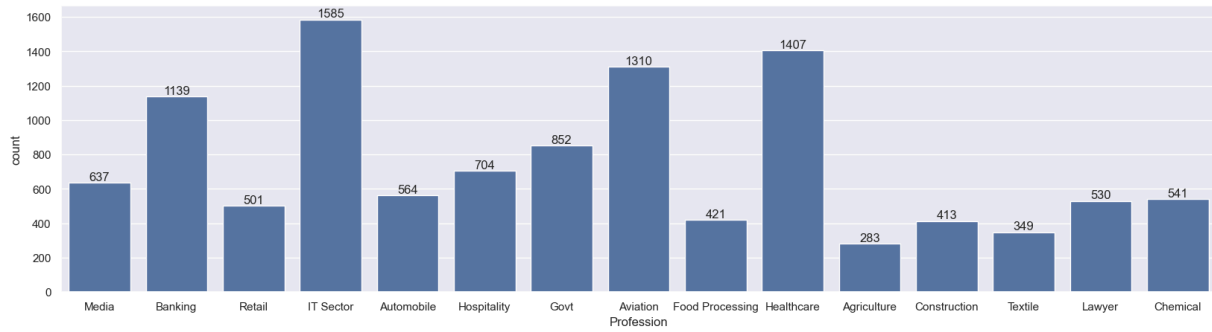
In [32]:
```
salary_pro = df.groupby(['Product_Category','Gender'],as_index=False)['Amount'].sum
sns.barplot(x='Product_Category',y='Amount',data=salary_pro,hue='Gender')
```

```
plt.show()
```



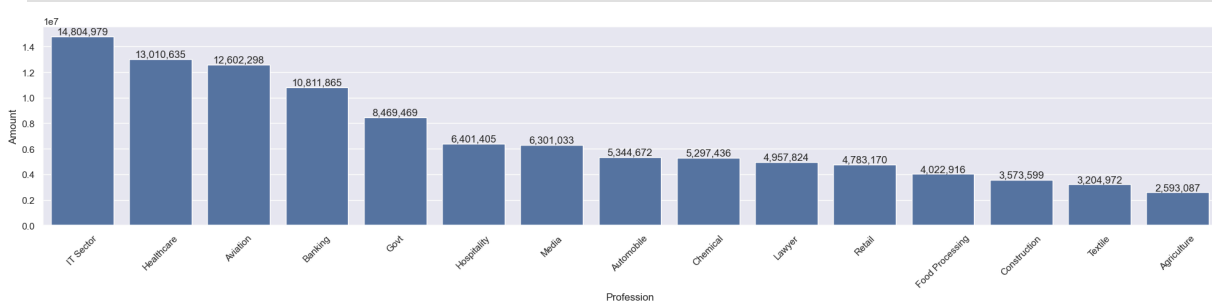
Profession Wise Transaction Count

In [33]:
```
ax=sns.countplot(data=df,x='Profession')
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```



Amount Wise Top Professions

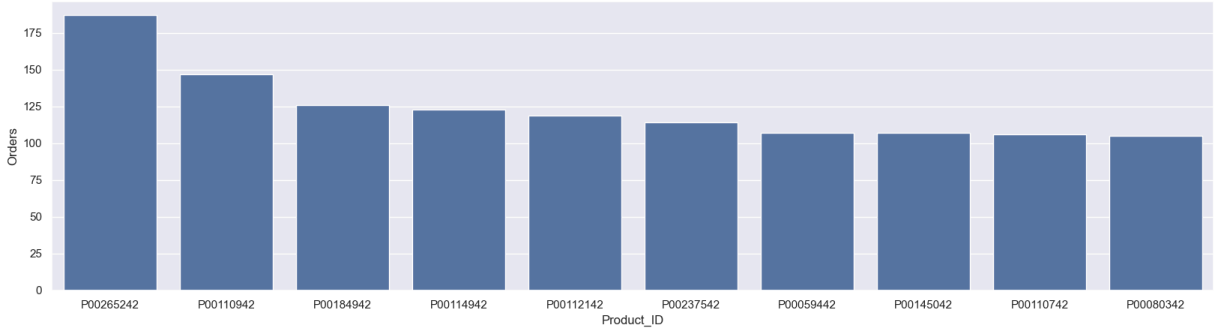
In [42]:
```
sales_state=df.groupby('Profession',as_index=False)['Amount'].sum().sort_values(by=
ax=sns.barplot(x='Profession',y='Amount',data=sales_state)
for bar in ax.containers:
    ax.bar_label(bar, labels=[f'{int(x.get_height()):,}' for x in bar])
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Order wise Top 10 Product

In [ ]:
```
Sales_Order=df.groupby(['Product_ID'],as_index=False)['Orders'].sum().sort_values(b
sns.barplot(x='Product_ID',y='Orders',data=Sales_Order)
```

```
plt.show()
```



Conclusion: Female age group 26-35 yrs from UP, Maharastra and Karnataka working in IT, Healthcare and Aviation are more likely to buy products from Beauty, Sports and Electronics