___NAME- Mrityunjay Josh!

_ PROJECT NAME- "New Year Sales Data 2024"...

⬇️

🟥 - A company has provided New Year Sales data of Year 2024, they want us to analyze it, or at the end, we share a summary with them, & With whose help the company-

➡️ 1- Improve Customer Experience,

➡️ 2- Increase Revenue or Sales ammount.

In [ ]:

```

```

Import Python Libraries :

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

IMPORT & READ FILE :

In [2]:

```
df=pd.read_csv('New Year Sales Data 2024.csv',encoding= 'unicode_escape')
```

In [3]:

```
df.index
```

Out[3]:

```
RangeIndex(start=0, stop=11251, step=1)
```

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Cust_name', 'Product_ID', 'condition', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Status', 'Orders', 'Amount', 'unnamed1', 'Unnamed: 15', 'User_ID'],
      dtype='object')
```

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

Out[5]:
```
191267
```

In [6]:
```
df.shape
```

Out[6]:
```
(11251, 17)
```

In [7]:
```
df.head(3)
```

Out[7]:

	Cust_name	Product_ID	condition	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Status	Orders
0	Sanskriti	P00125942	NaN	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	NaN	1
1	Kartik	P00110942	NaN	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	NaN	3
2	Bindu	P00118542	NaN	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	NaN	3

MODIFYING & CLEANING PROCESS :

In [8]:
```
data=df.iloc[:,[16,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]]
```

In [9]:
```
data.head(3)
```

Out[9]:

	User_ID	Cust_name	Product_ID	condition	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Status
0	1002903	Sanskriti	P00125942	NaN	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	NaN
1	1000732	Kartik	P00110942	NaN	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	NaN
2	1001990	Bindu	P00118542	NaN	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	NaN

In [10]:
```
data.tail(3)
```

Out[10]:

	User_ID	Cust_name	Product_ID	condition	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Statu
11248	1001209	Oshin	P00201342	NaN	F	36-45	40	0	Madhya Pradesh	Central	Textile	Office	Na
11249	1004023	Noonan	P00059442	NaN	M	36-45	37	0	Karnataka	Southern	Agriculture	Office	Na
11250	1002744	Brumley	P00281742	NaN	F	18-25	19	0	Maharashtra	Western	Healthcare	Office	Na

In [11]:
```
pd.isnull(data).sum()   # check null values:
```

Out[11]:
```
User_ID                0
Cust_name              0
Product_ID             0
condition          11251
Gender                 0
Age Group              0
Age                    0
Marital_Status         0
State                  0
Zone                   0
Occupation             4
Product_Category       1
Status             11251
Orders                 0
Amount                12
unnamed1           11251
Unnamed: 15        11251
dtype: int64
```

In [12]:
```
data.drop(['condition','Status','unnamed1','Unnamed: 15'], axis=1,inplace=True)   # drop blank columns:
```

```
C:\Users\19mri\AppData\Local\Temp\ipykernel_9856\209070940.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data.drop(['condition','Status','unnamed1','Unnamed: 15'], axis=1,inplace=True)   # drop blank columns:
```

In [13]:
```
data.head(3)
```

Out[13]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0

In [14]:
```
pd.isnull(data).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           4
Product_Category     1
Orders               0
Amount              12
dtype: int64
```

In [15]:
```
data.dropna(inplace=True)  # drop null values:
```

```
C:\Users\19mri\AppData\Local\Temp\ipykernel_9856\758851116.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data.dropna(inplace=True)  # drop null values:
```

In [16]:
```
pd.isnull(data).sum()
```

Out[16]:
```
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 [17]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11235 entries, 0 to 11250
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   User_ID           11235 non-null  int64  
 1   Cust_name         11235 non-null  object 
 2   Product_ID        11235 non-null  object 
 3   Gender            11235 non-null  object 
 4   Age Group         11235 non-null  object 
 5   Age               11235 non-null  int64  
 6   Marital_Status    11235 non-null  int64  
 7   State             11235 non-null  object 
 8   Zone              11235 non-null  object 
 9   Occupation        11235 non-null  object 
 10  Product_Category  11235 non-null  object 
 11  Orders            11235 non-null  int64  
 12  Amount            11235 non-null  float64
dtypes: float64(1), int64(4), object(8)
memory usage: 1.2+ MB
```

In [18]:
```
data.rename(columns={'Product_Category':'Pro Category'},inplace=True) # Change Or Short Column Name:
```

```
C:\Users\19mri\AppData\Local\Temp\ipykernel_9856\4070226324.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data.rename(columns={'Product_Category':'Pro Category'},inplace=True) # Change Or Short Column Name:
```

In [19]:
```
data['Amount']=data['Amount'].astype(int)    # change Data Type:
```

```
C:\Users\19mri\AppData\Local\Temp\ipykernel_9856\2570367169.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Amount']=data['Amount'].astype(int)    # change Data Type:
```

In [20]:
```
data['Amount'].dtypes
# OR
```

Out[20]:
```
dtype('int32')
```

In [21]:
```
data.dtypes          # Check Data Types:
```

Out[21]:
```
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
Pro Category     object
Orders            int64
Amount            int32
dtype: object
```

In [22]:
```
pd.unique(data['Pro Category'])   # Check Unique Value's Name in Column:
```

Out[22]:
```
array(['Auto', 'Hand & Power Tools', 'Stationery', 'Tupperware',
       'Footwear & Shoes', 'Furniture', 'Food', 'Games & Toys',
       'Sports Products', 'Books', 'Electronics & Gadgets', 'Decor',
       'Clothing & Apparel', 'Beauty', 'Household items', 'Pet Care',
       'Veterinary', 'Office'], dtype=object)
```

In [23]:
```
data.nunique()   # Check Count of Unique Values in Data:
```

Out[23]:
```
User_ID          3752
Cust_name        1250
Product_ID       2350
Gender              2
Age Group           7
Age                81
Marital_Status      2
State              16
Zone                5
Occupation         15
Pro Category       18
Orders              4
Amount           6580
dtype: int64
```

In [24]:
```
# use describe() for specific columns:
data[['Age','Orders','Amount']].describe()    # descriptive statistics:
```

Out[24]:

	Age	Orders	Amount
count	11235.000000	11235.000000	11235.000000
mean	35.410236	2.489453	9449.753538
std	12.754911	1.115044	5219.281678
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	12671.000000
max	92.000000	4.000000	23952.000000

In [25]:
```
data.describe(include=object)
```

Out[25]:

	Cust_name	Product_ID	Gender	Age Group	State	Zone	Occupation	Pro Category
count	11235	11235	11235	11235	11235	11235	11235	11235
unique	1250	2350	2	7	16	5	15	18
top	Vishakha	P00265242	F	26-35	Uttar Pradesh	Central	IT Sector	Clothing & Apparel
freq	42	53	7829	4540	1944	4288	1583	2655

Exploratory Data Analysis :

In [26]:
```
sns.set(style="darkgrid")
```

In [27]:
```
data.head(3)
```

Out[27]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Pro Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924

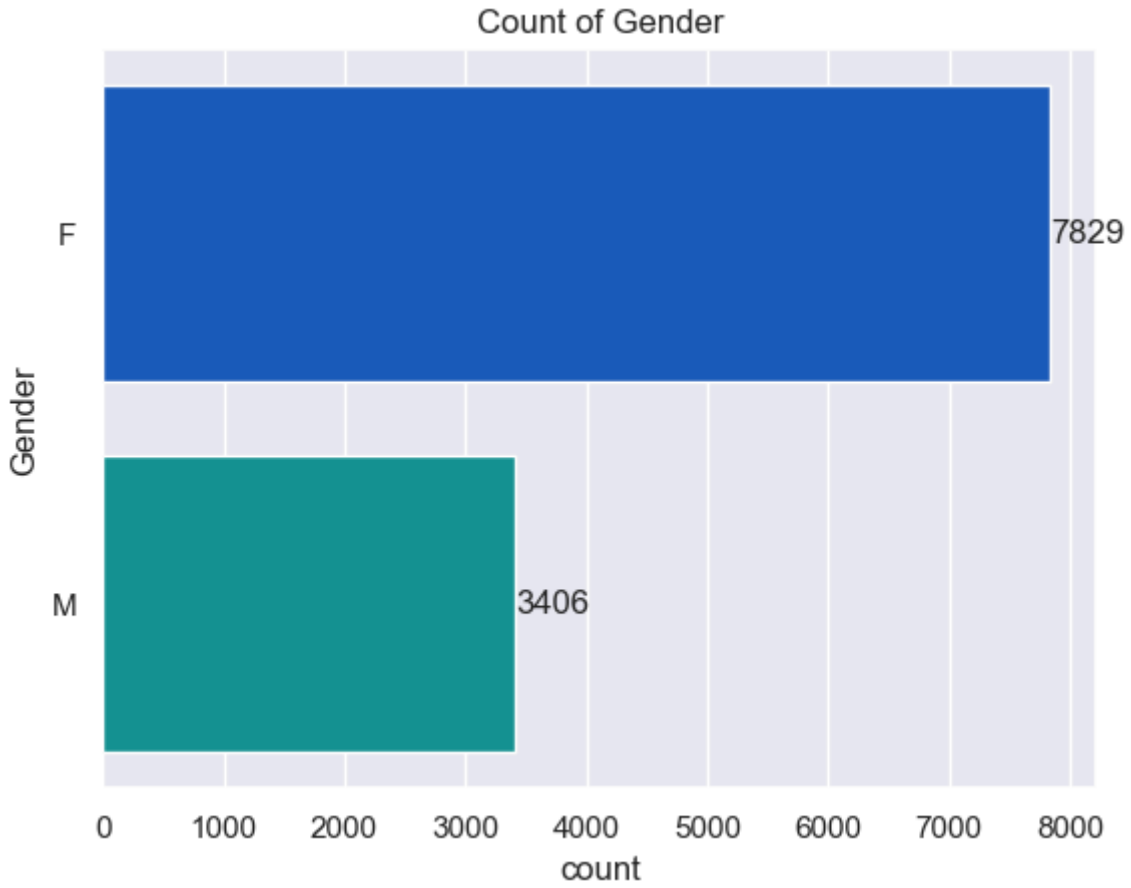
GENDER:_

In [28]:
```
data.groupby(['Gender'])['Gender'].count().reset_index(name='sum')
```

Out[28]:

	Gender	sum
0	F	7829
1	M	3406

In [29]:
```
ax=sns.countplot(y='Gender',data=data,palette='winter')
for bars in ax.containers:
    ax.bar_label(bars)
plt.title('Count of Gender')
plt.show()
```



In [30]:
```
gr=data.groupby(['Gender'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False)
```

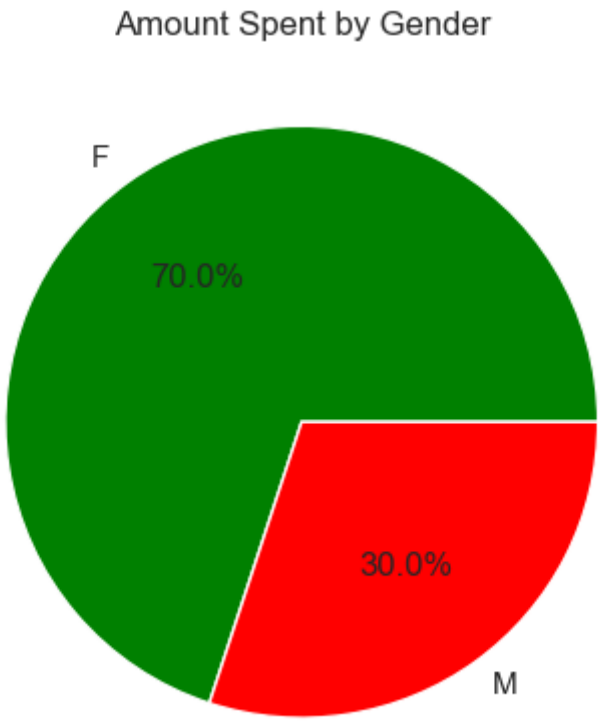
In [31]:
```
gr
```

Out[31]:

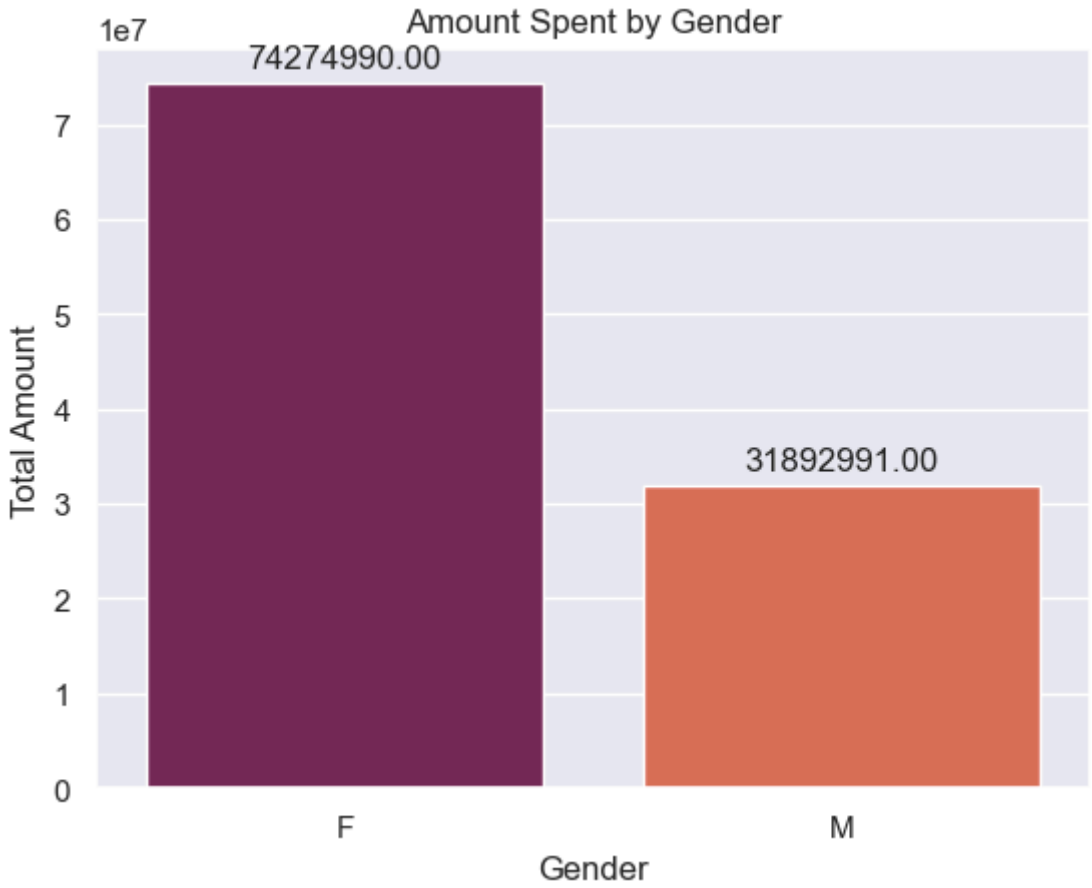
	Gender	sum
0	F	74274990
1	M	31892991

In [32]:
```
colors=['green','red']
```

In [33]:
```
plt.pie(x='sum',labels='Gender',data=gr,colors=colors,autopct='%1.1f%%')
plt.title('Amount Spent by Gender')
plt.show()
```



In [34]:
```
ax=sns.barplot(x='Gender',y='sum',data=gr,palette='rocket')
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.2f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 9),
                textcoords = 'offset points')
plt.title('Amount Spent by Gender')
plt.xlabel('Gender')
plt.ylabel('Total Amount');
```



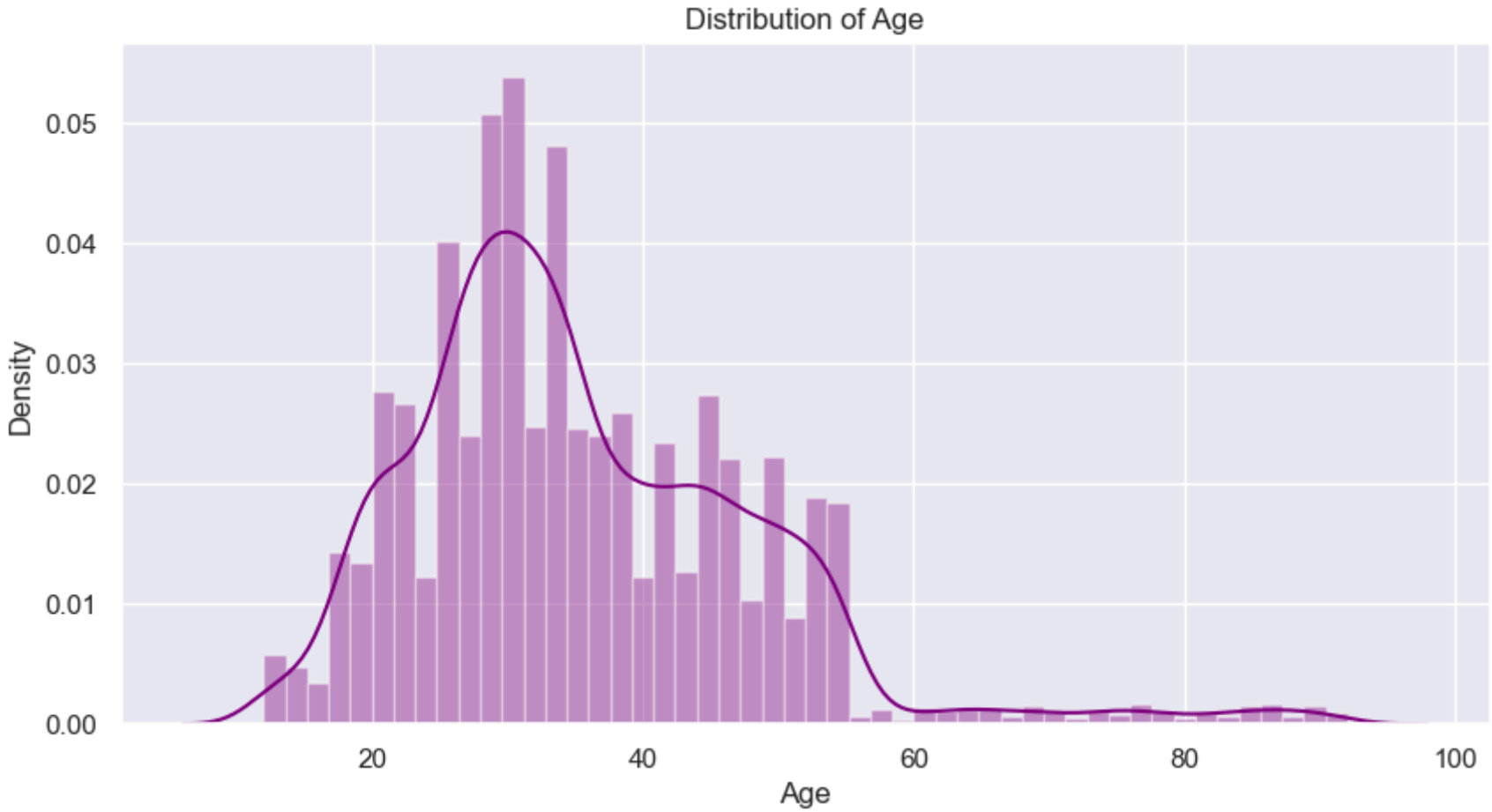
🔍 *From above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men...*

In [ ]:

AGE & AGE GROUP:__

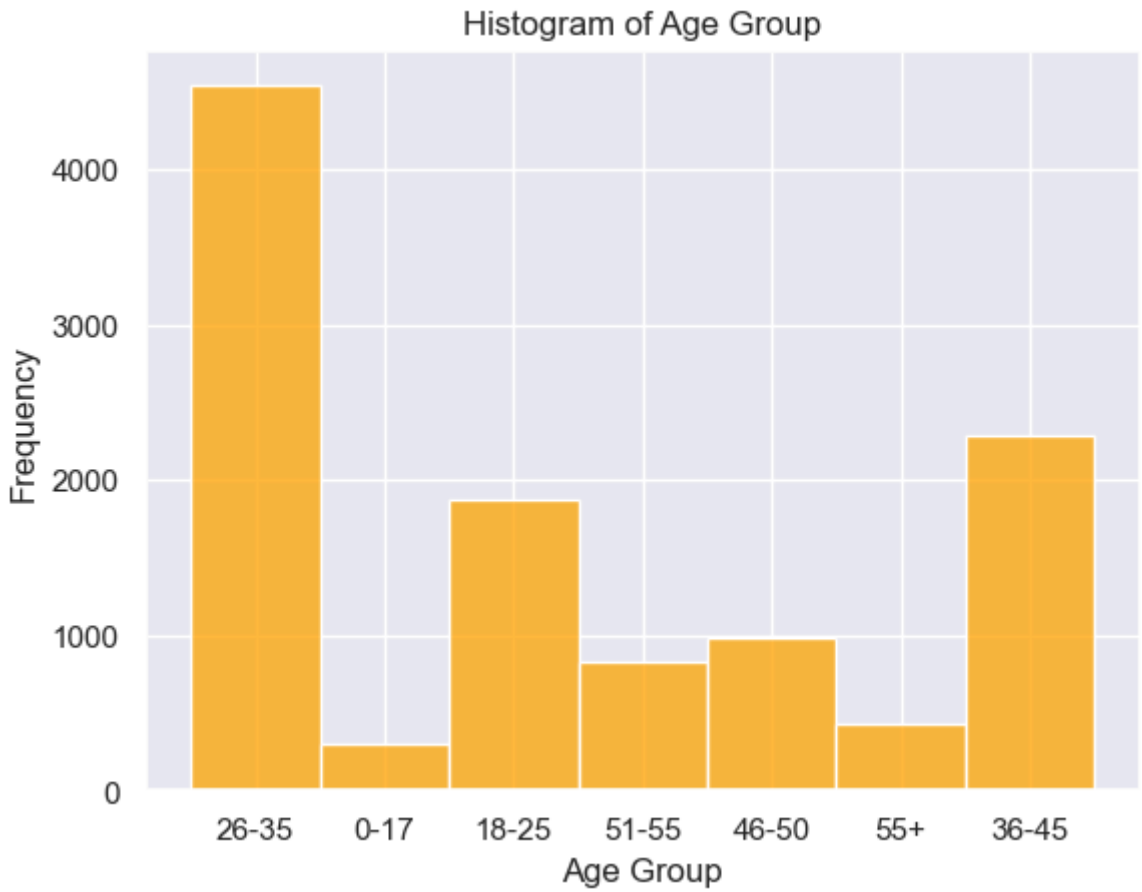
In [35]:
```
plt.figure(figsize=(10,5))
sns.distplot(data['Age'],color='purple')
plt.title('Distribution of Age')
plt.show()
```

```
C:\Users\19mri\AppData\Local\Temp\ipykernel_9856\151055996.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Age'],color='purple')
```



In [36]:
```
sns.histplot(data['Age Group'],color='orange')
plt.title('Histogram of Age Group')
```

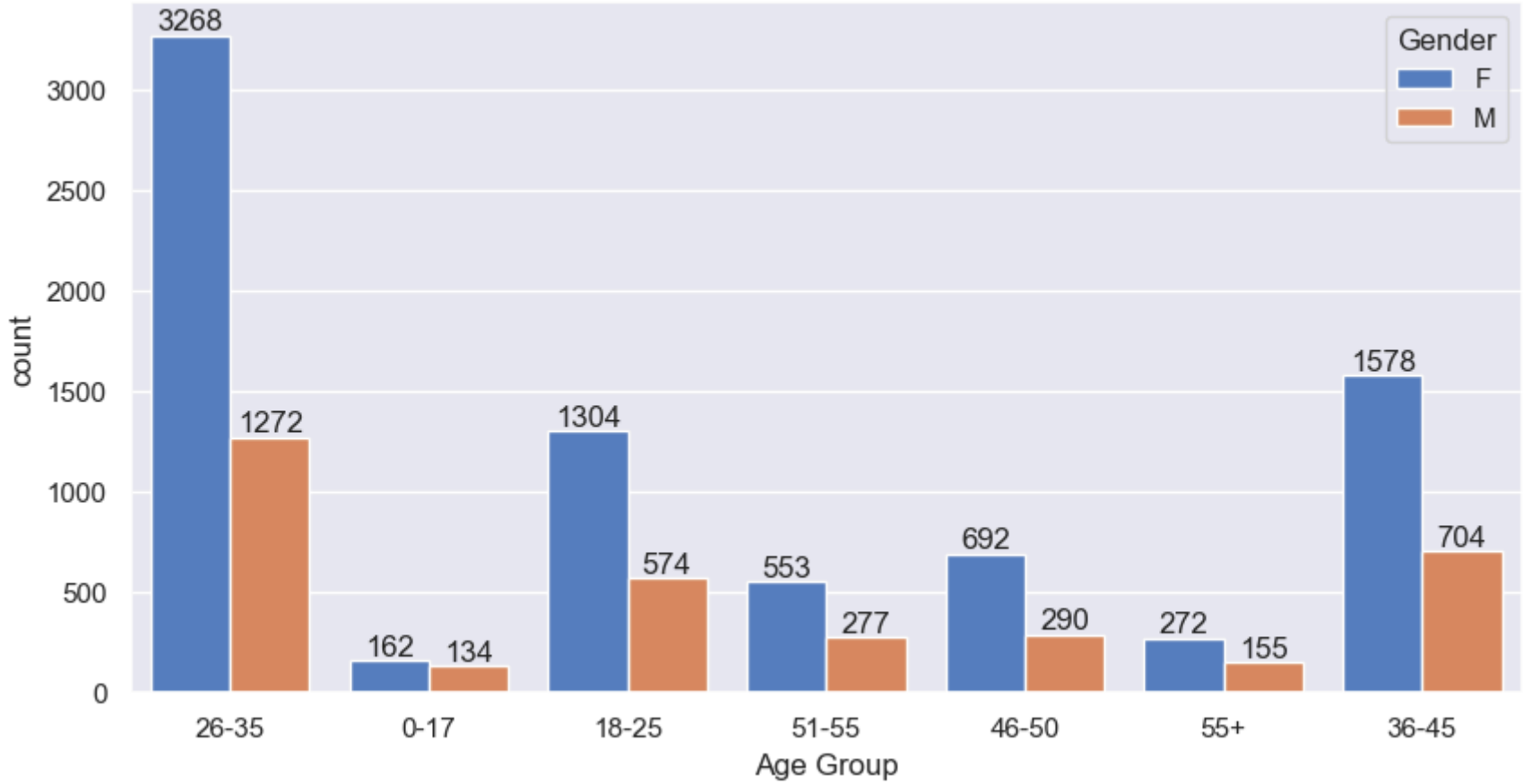
```
plt.xlabel('Age Group')
plt.ylabel('Frequency')
plt.show()
```



🔍 *Here Peoples between Age of 26 To 35, do More Shopping Compare to Others*:

In [ ]:
```
```

In [37]:
```
plt.figure(figsize=(10,5))
ax=sns.countplot(x='Age Group',hue='Gender',data=data,palette='muted')
for bars in ax.containers:
    ax.bar_label(bars)
```



In [38]:
```
gr=data.groupby(['Age Group','Gender'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False)
```

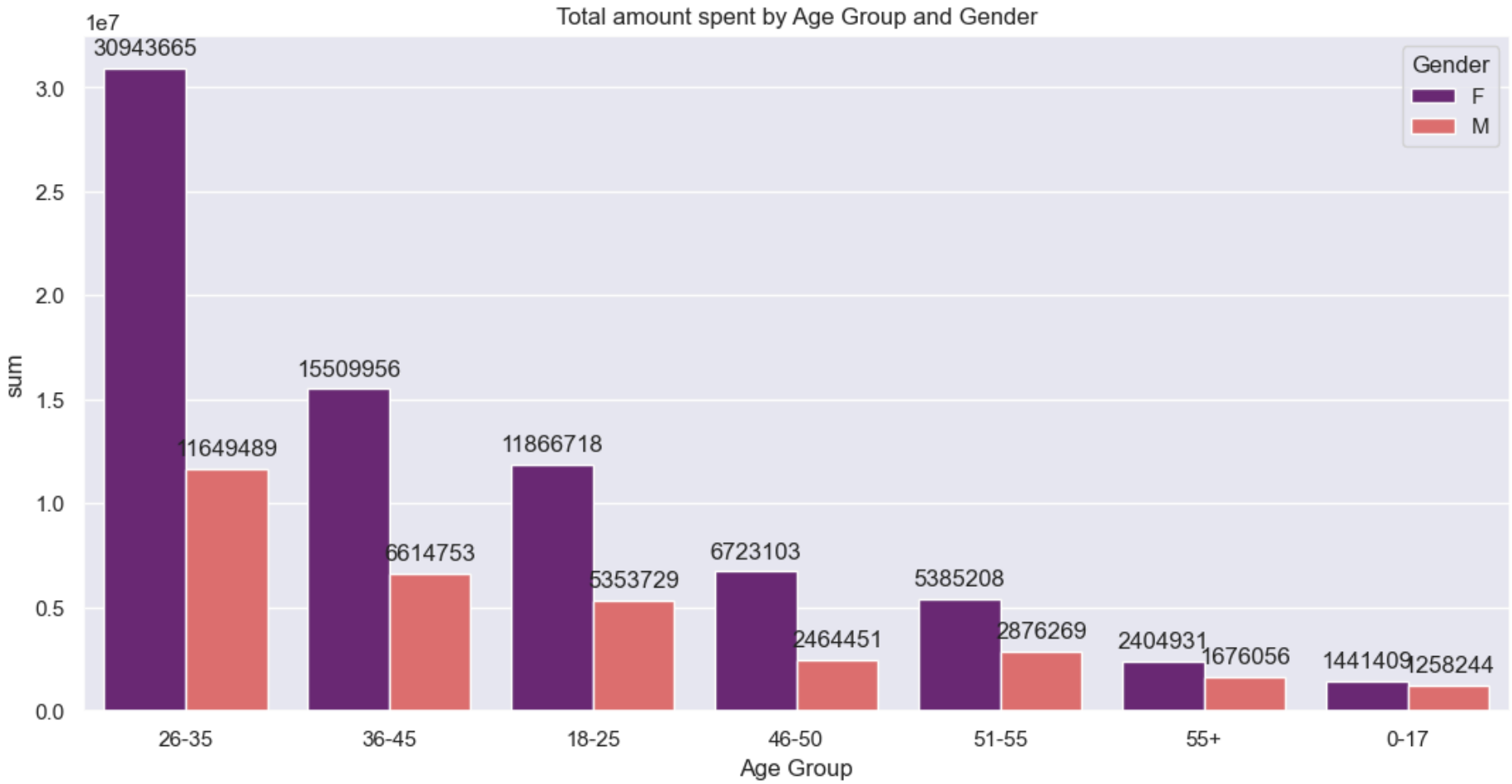
In [39]:
```
gr
```

Out[39]:

	Age Group	Gender	sum
4	26-35	F	30943665
6	36-45	F	15509956
2	18-25	F	11866718
5	26-35	M	11649489
8	46-50	F	6723103
7	36-45	M	6614753
10	51-55	F	5385208
3	18-25	M	5353729
11	51-55	M	2876269
9	46-50	M	2464451
12	55+	F	2404931
13	55+	M	1676056
0	0-17	F	1441409
1	0-17	M	1258244

In [40]:

```
plt.figure(figsize=(13, 6.2))
ax=sns.barplot(x='Age Group',y='sum',hue='Gender',data=gr,palette='magma')
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 10),
                textcoords = 'offset points')
plt.title('Total amount spent by Age Group and Gender')
plt.show()
```



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

In [ ]:

STATE:___

In [41]:
```
data.head(2)
```

Out[41]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Pro Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934

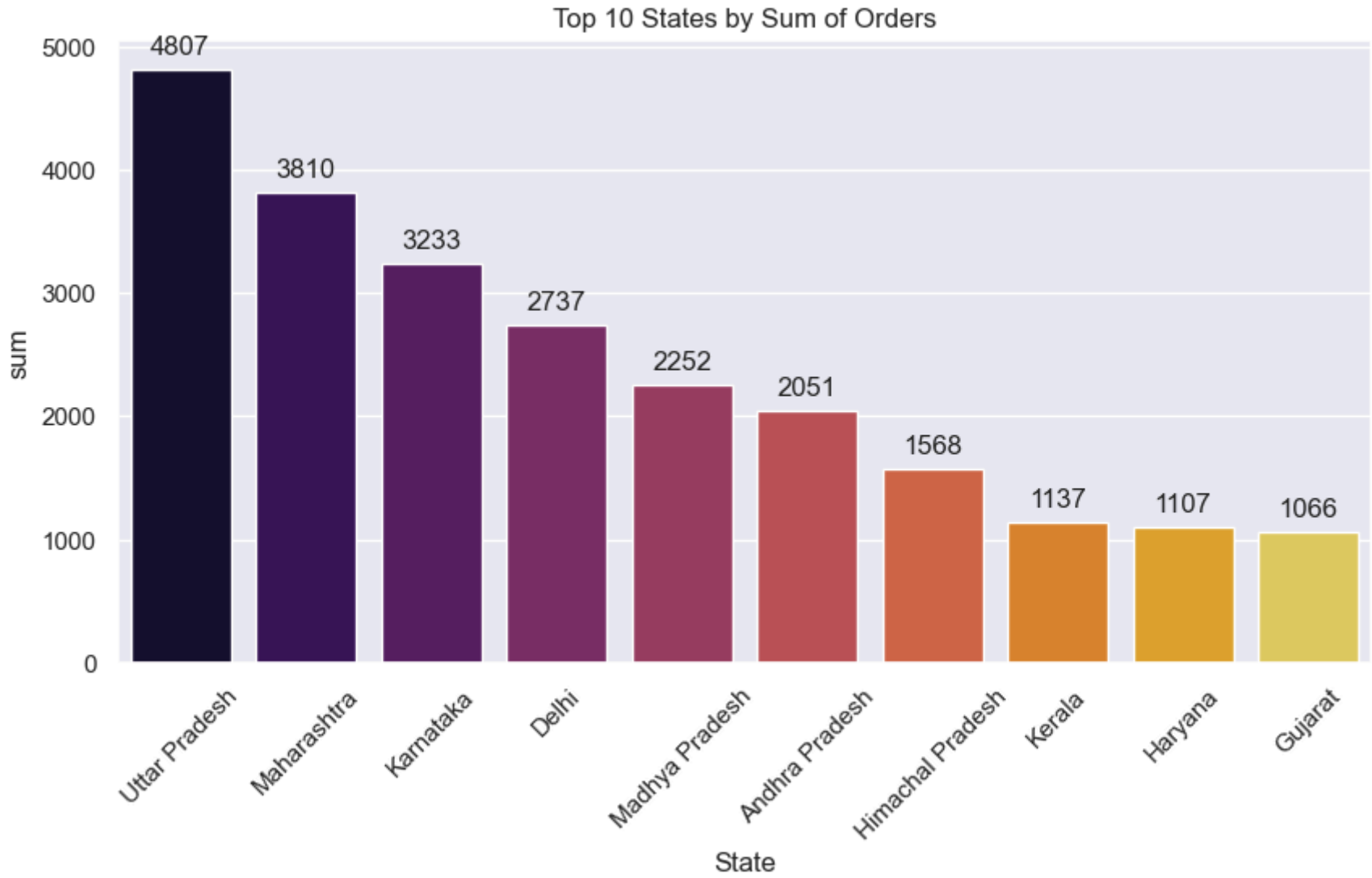
In [42]:
```
gr=data.groupby(['State'])['Orders'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False).head(10)
```

In [43]:
```
gr
```

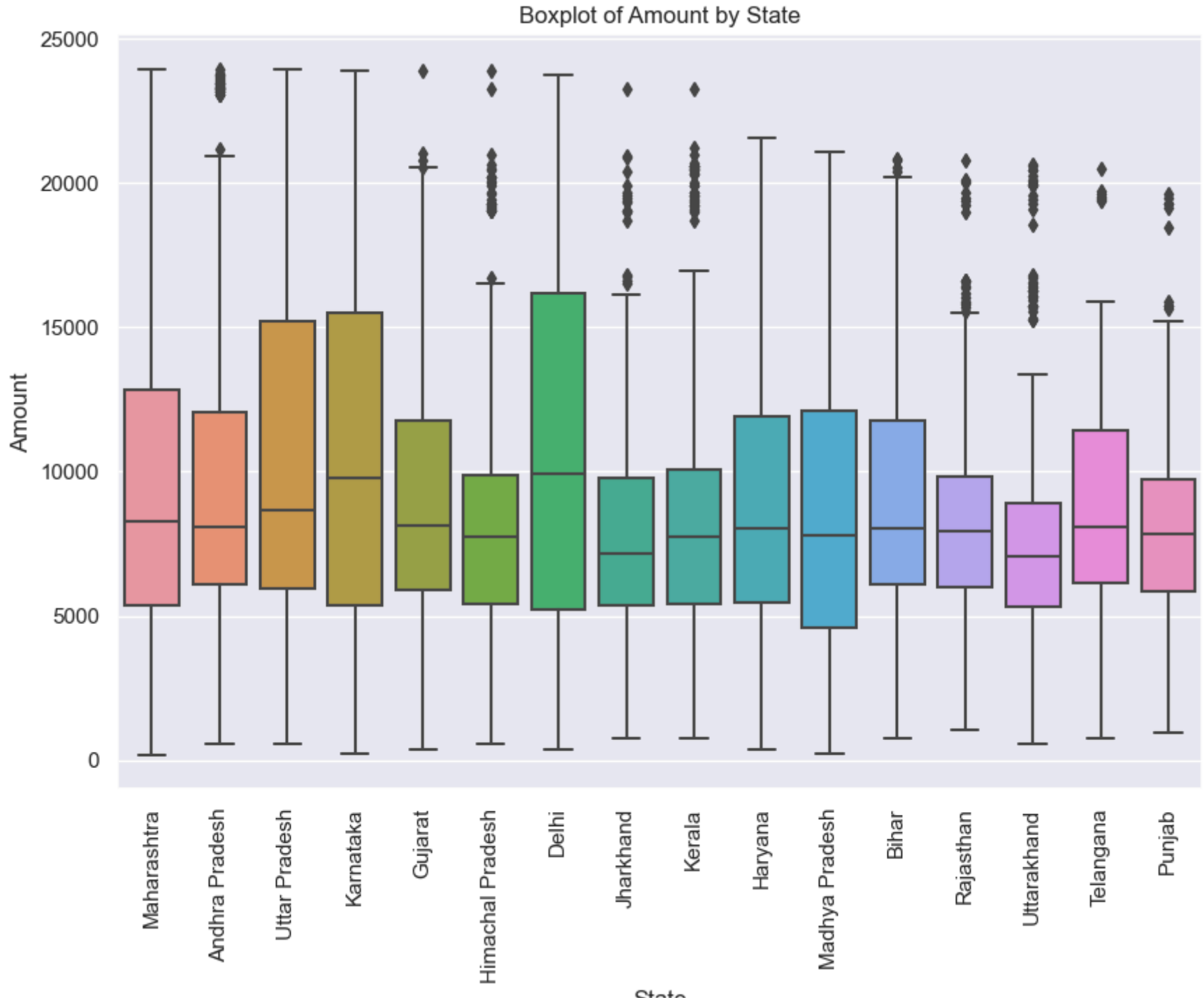
Out[43]:

	State	sum
14	Uttar Pradesh	4807
10	Maharashtra	3810
7	Karnataka	3233
2	Delhi	2737
9	Madhya Pradesh	2252
0	Andhra Pradesh	2051
5	Himachal Pradesh	1568
8	Kerala	1137
4	Haryana	1107
3	Gujarat	1066

In [44]:
```
plt.figure(figsize=(10,5))
ax=sns.barplot(x='State',y='sum',data=gr,palette='inferno')
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 10),
                textcoords = 'offset points')
plt.title('Top 10 States by Sum of Orders')
plt.xticks(rotation=45)
plt.show()
```



In [45]:
```
plt.figure(figsize=(10,7))
sns.boxplot(x='State',y='Amount',data=data)
plt.title('Boxplot of Amount by State')
plt.xticks(rotation=90)
plt.show()
```


Boxplot of Amount by State
Amount
25000
20000
15000
10000
5000
0
Maharashtra
Andhra Pradesh
Uttar Pradesh
Karnataka
Gujarat
Himachal Pradesh
Delhi
Jharkhand
Kerala
Haryana
Madhya Pradesh
Bihar
Rajasthan
Uttarakhand
Telangana
Punjab
State

State

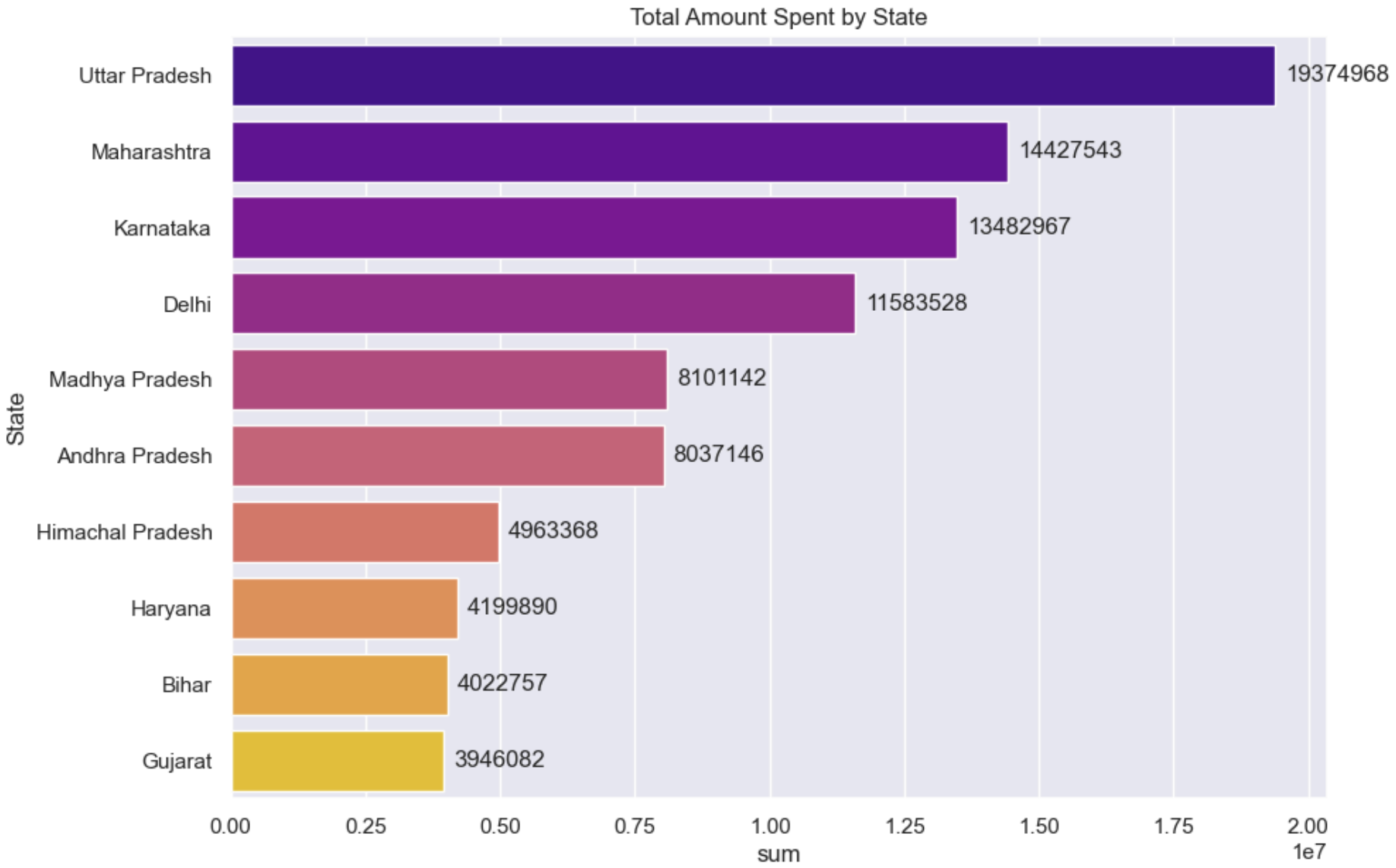
In [46]:
```
gr=data.groupby(['State'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False).head(10)
```

In [47]:
```
gr
```

Out[47]:

	State	sum
14	Uttar Pradesh	19374968
10	Maharashtra	14427543
7	Karnataka	13482967
2	Delhi	11583528
9	Madhya Pradesh	8101142
0	Andhra Pradesh	8037146
5	Himachal Pradesh	4963368
4	Haryana	4199890
1	Bihar	4022757
3	Gujarat	3946082

In [48]:
```
plt.figure(figsize=(10,7))
ax = sns.barplot(y='State', x='sum', data=gr, palette='plasma')
for p in ax.patches:
    ax.annotate(format(p.get_width(), '.0f'),
                (p.get_width(), p.get_y() + p.get_height() / 2.),
                xytext=(5, 0),
                textcoords='offset points',
                ha='left',
                va='center')

plt.title('Total Amount Spent by State')
plt.show()
```



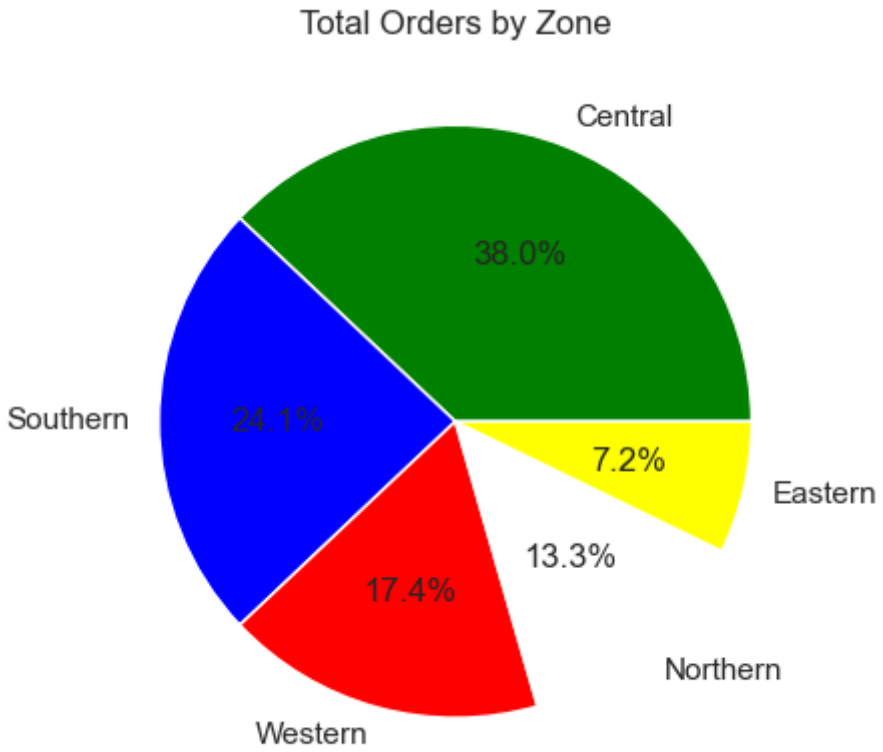
🔍 *From above graphs we can see that most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra & Karnataka respectively...*

In [ ]:

Zone:_

In [49]:
```
pie_chart=data.groupby(['Zone'])['Orders'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False)
```

In [50]:
```
colors=['green','blue','red','white','yellow']
plt.pie(x=pie_chart['sum'], labels=pie_chart['Zone'],autopct='%1.1f%%',colors=colors)
plt.title('Total Orders by Zone')
plt.show()
```



🔍 *Mostly Orders are from Central Zone & South Zone...*

In [ ]:
```

```

Marital Status:_

In [51]:
```
gr=data.groupby(['Marital_Status'])['Marital_Status'].count().reset_index(name='count')
```

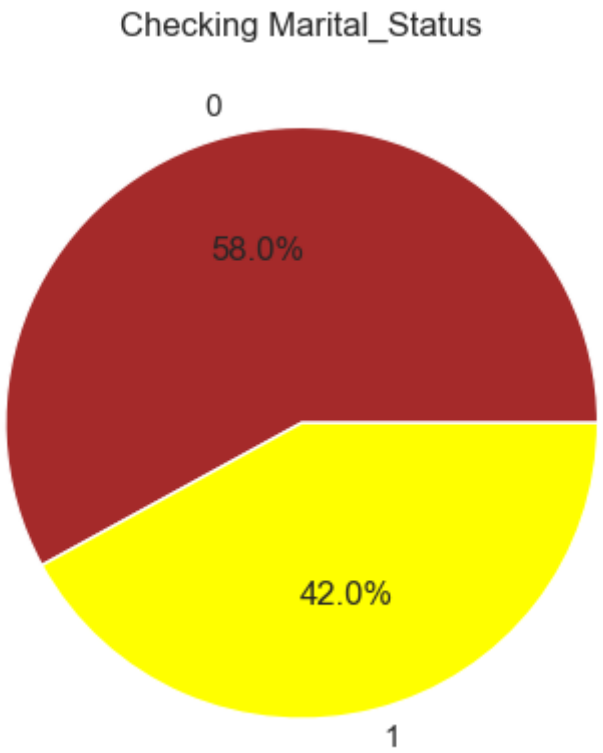
In [52]:
```
gr
```

Out[52]:

	Marital_Status	count
0	0	6515
1	1	4720

In [53]:
```
colors=['brown','yellow']
```

In [54]:
```
plt.pie(x='count',labels='Marital_Status',data=gr,autopct='%1.1f%%',colors=colors);
plt.title('Checking Marital_Status')
plt.show()
```



🔍 *From above Figure, Most of buyers are Unmarried...*

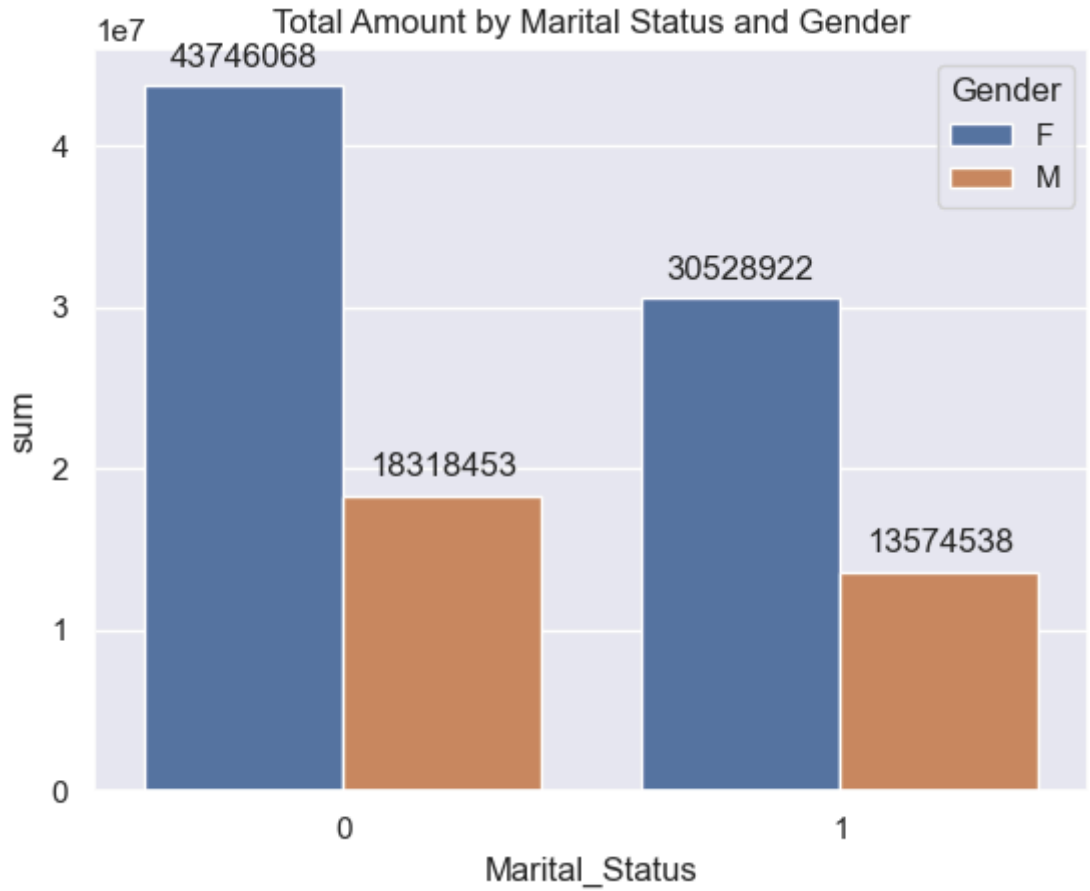
In [55]:
```
gr=data.groupby(['Marital_Status','Gender'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False)
```

In [56]:
```
gr
```

Out[56]:

	Marital_Status	Gender	sum
0	0	F	43746068
2	1	F	30528922
1	0	M	18318453
3	1	M	13574538

In [57]:
```
ax = sns.barplot(x='Marital_Status', hue='Gender', y='sum', data=gr)
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center',
                xytext=(0, 10),
                textcoords='offset points')
title = 'Total Amount by Marital Status and Gender'
plt.title(title)
plt.show()
```



🔍 *From above graphs we can see that most of the buyers are Unmarried (women) and they have high purchasing power...*

In [ ]:

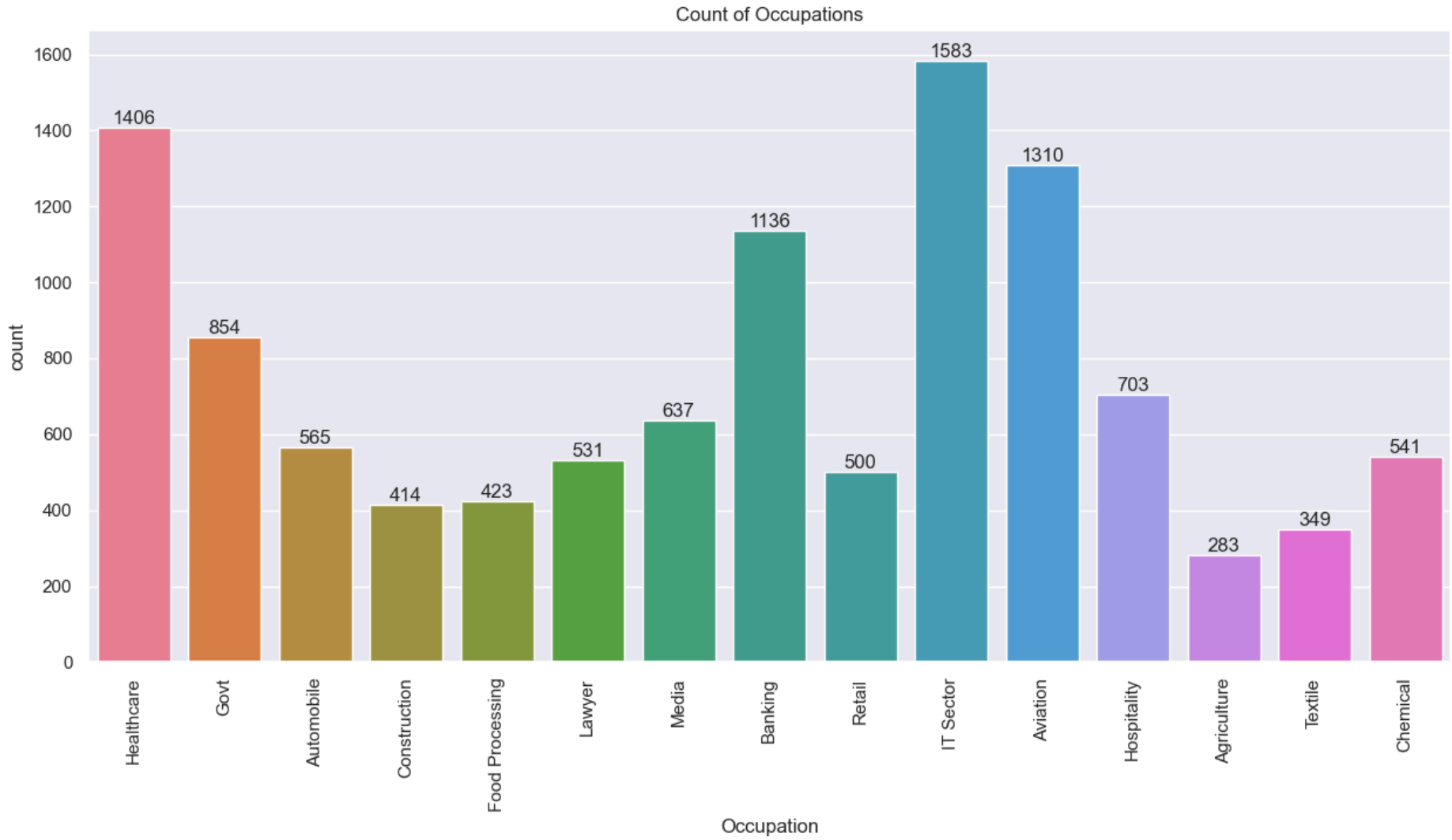
Occupation:___

In [58]:
```
data.head(2)
```

Out[58]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Pro Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934

In [59]:
```
plt.figure(figsize=(15,7))
ax=sns.countplot(x='Occupation',data=data,palette='husl')
for bars in ax.containers:
    ax.bar_label(bars)
plt.title('Count of Occupations')
plt.xticks(rotation=90.1)
plt.show()
```



In [60]:
```
gr=data.groupby(['Occupation'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False).head(10)
```

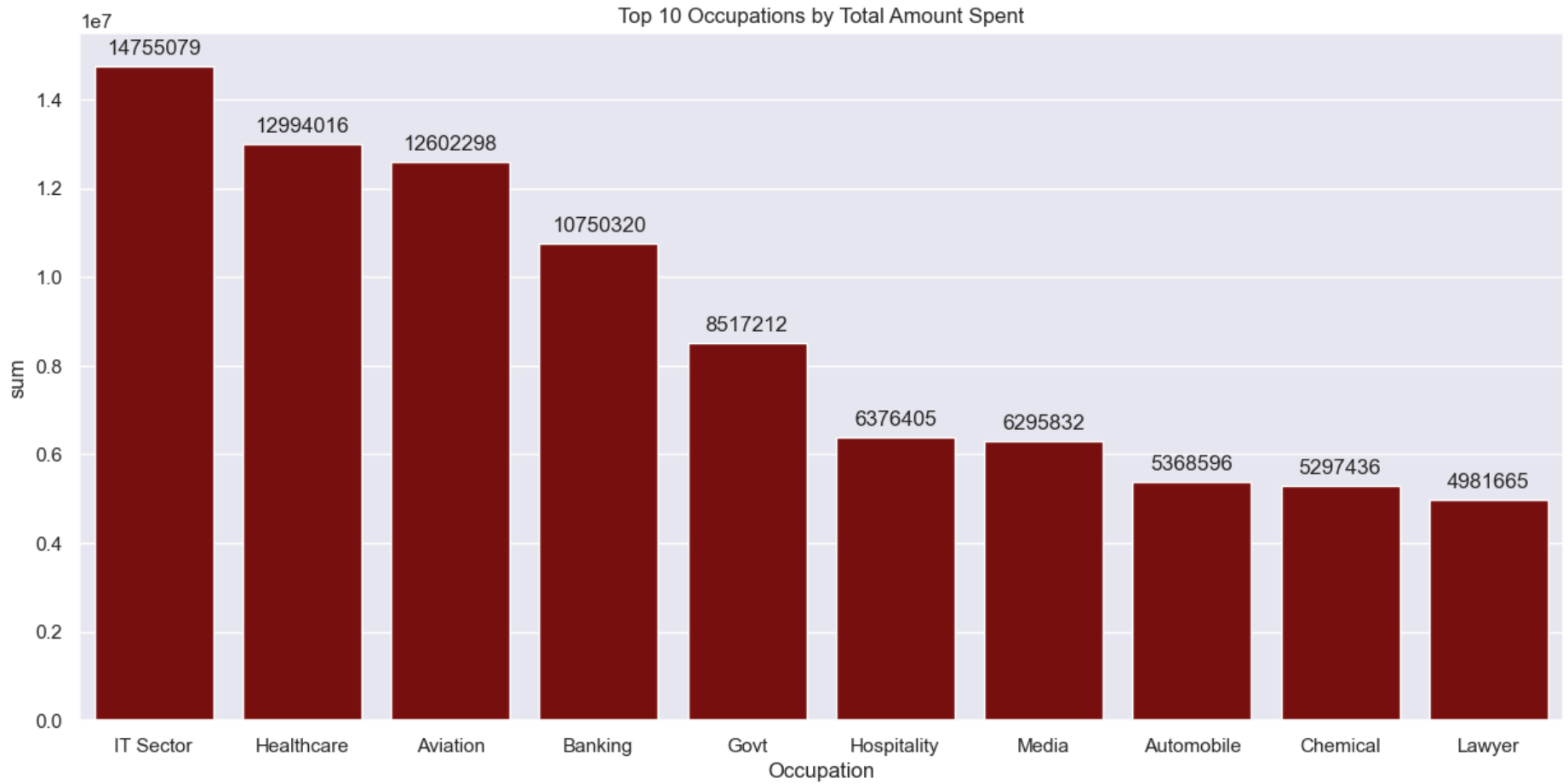
In [61]:
```
gr
```

Out[61]:

	Occupation	sum
10	IT Sector	14755079
8	Healthcare	12994016
2	Aviation	12602298
3	Banking	10750320
7	Govt	8517212
9	Hospitality	6376405
12	Media	6295832
1	Automobile	5368596
4	Chemical	5297436
11	Lawyer	4981665

In [62]:

```
plt.figure(figsize=(15,7))
ax=sns.barplot(x='Occupation',y='sum',data=gr,color='darkred')
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center',
                xytext=(0, 10),
                textcoords='offset points')
plt.title('Top 10 Occupations by Total Amount Spent')
plt.show()
```



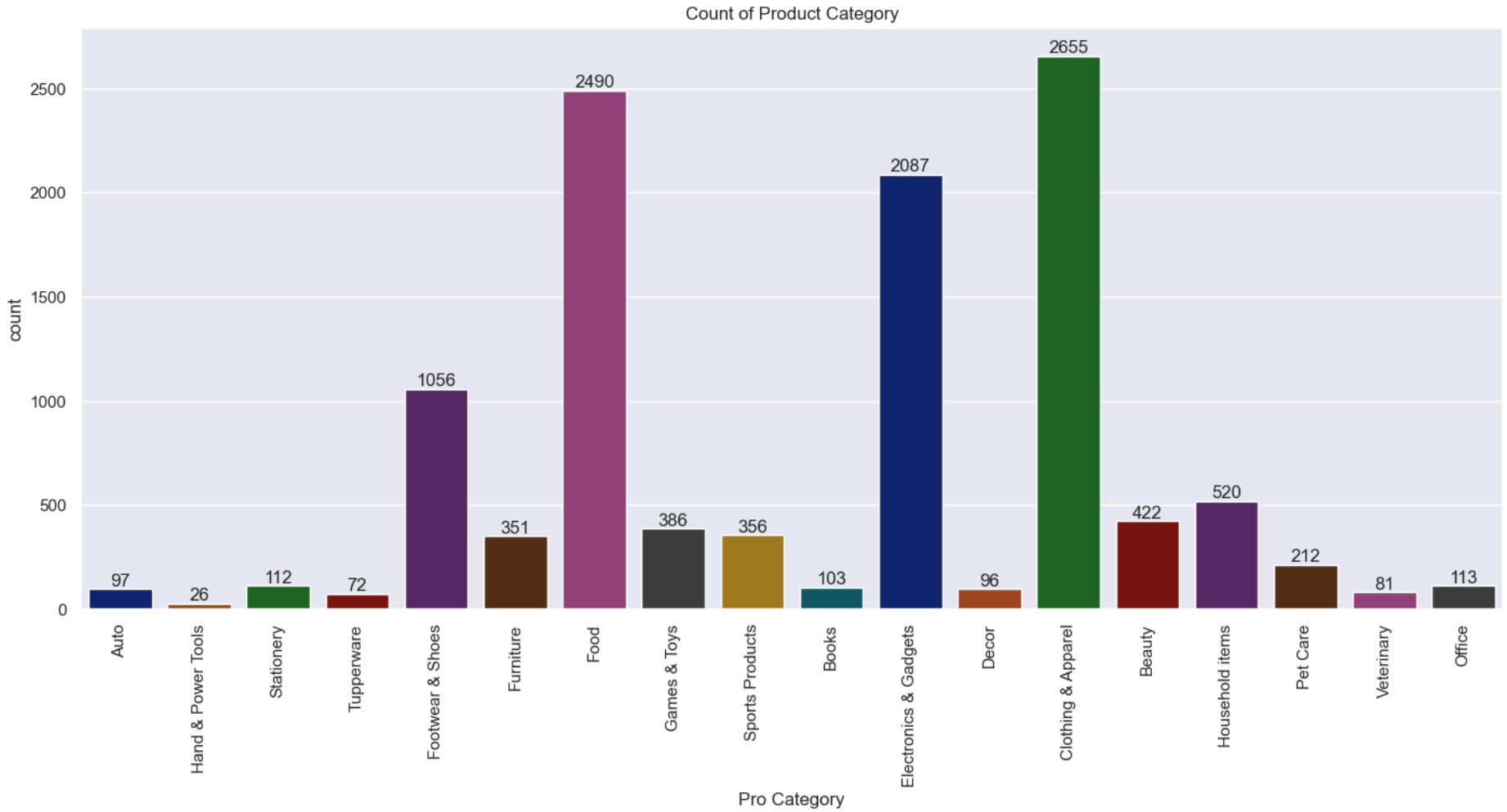
🔍 *From above graphs we can see that most of the buyers are working in IT, Healthcare, Aviation, & Banking sector...*

In [ ]:

Product_Category:___

In [63]:
```
plt.figure(figsize=(17,7))
ax=sns.countplot(x='Pro Category',data=data,palette='dark')
for bars in ax.containers:
    ax.bar_label(bars)
```

```
plt.title('Count of Product Category')
plt.xticks(rotation=90)
plt.show()
```



In [64]:
```
gr=data.groupby(['Pro Category'])['Amount'].sum().reset_index(name='sum').sort_values(by='sum',ascending=False).head(10)
```

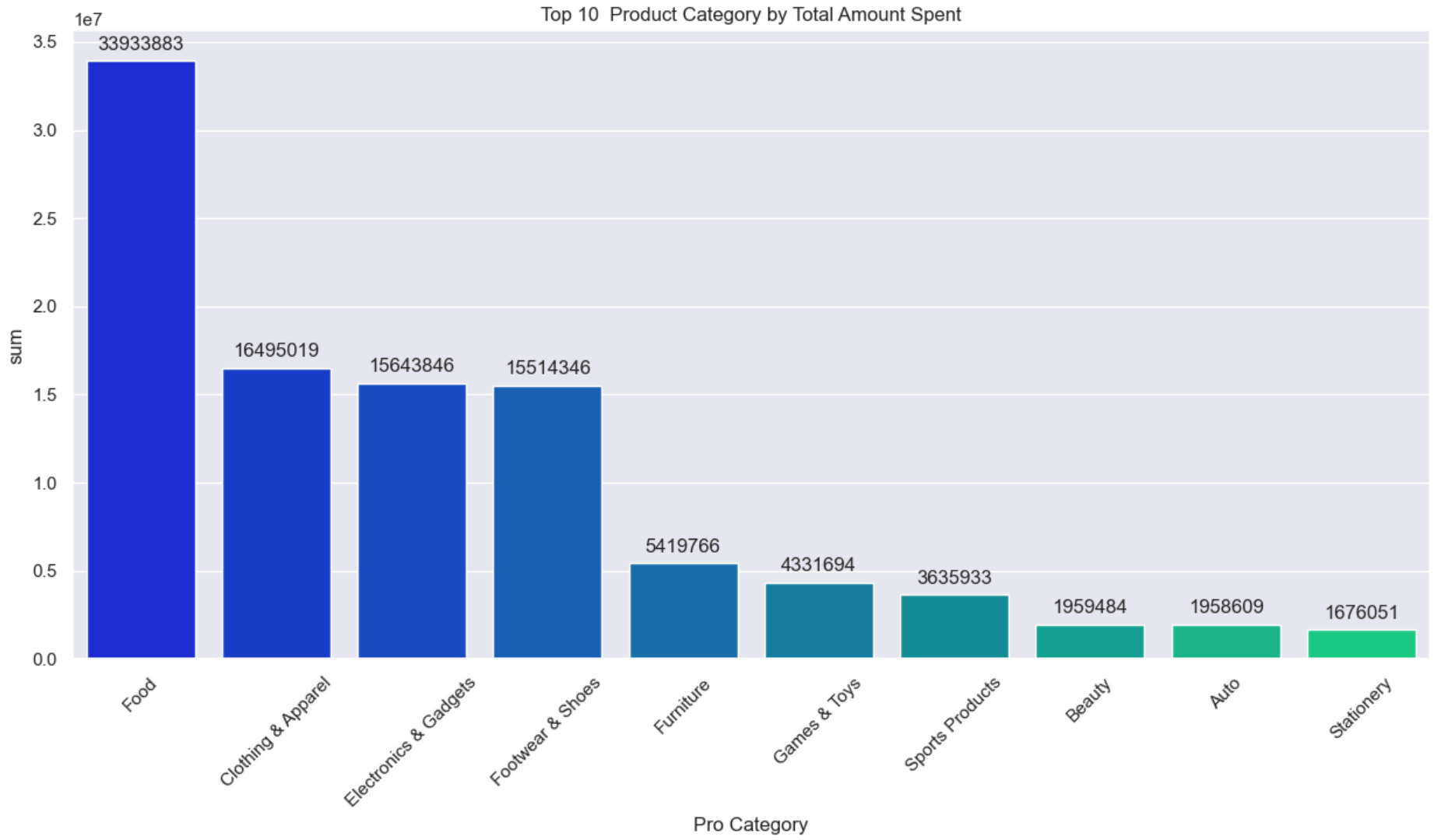
In [65]:
```
gr
```

Out[65]:

	Pro Category	sum
6	Food	33933883
3	Clothing & Apparel	16495019
5	Electronics & Gadgets	15643846
7	Footwear & Shoes	15514346
8	Furniture	5419766
9	Games & Toys	4331694
14	Sports Products	3635933
1	Beauty	1959484
0	Auto	1958609
15	Stationery	1676051

In [66]:

```
plt.figure(figsize=(15,7))
ax=sns.barplot(x='Pro Category',y='sum',data=gr,palette='winter')
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center',
                xytext=(0, 10),
                textcoords='offset points')
plt.title('Top 10  Product Category by Total Amount Spent')
plt.xticks(rotation=45)
plt.show()
```

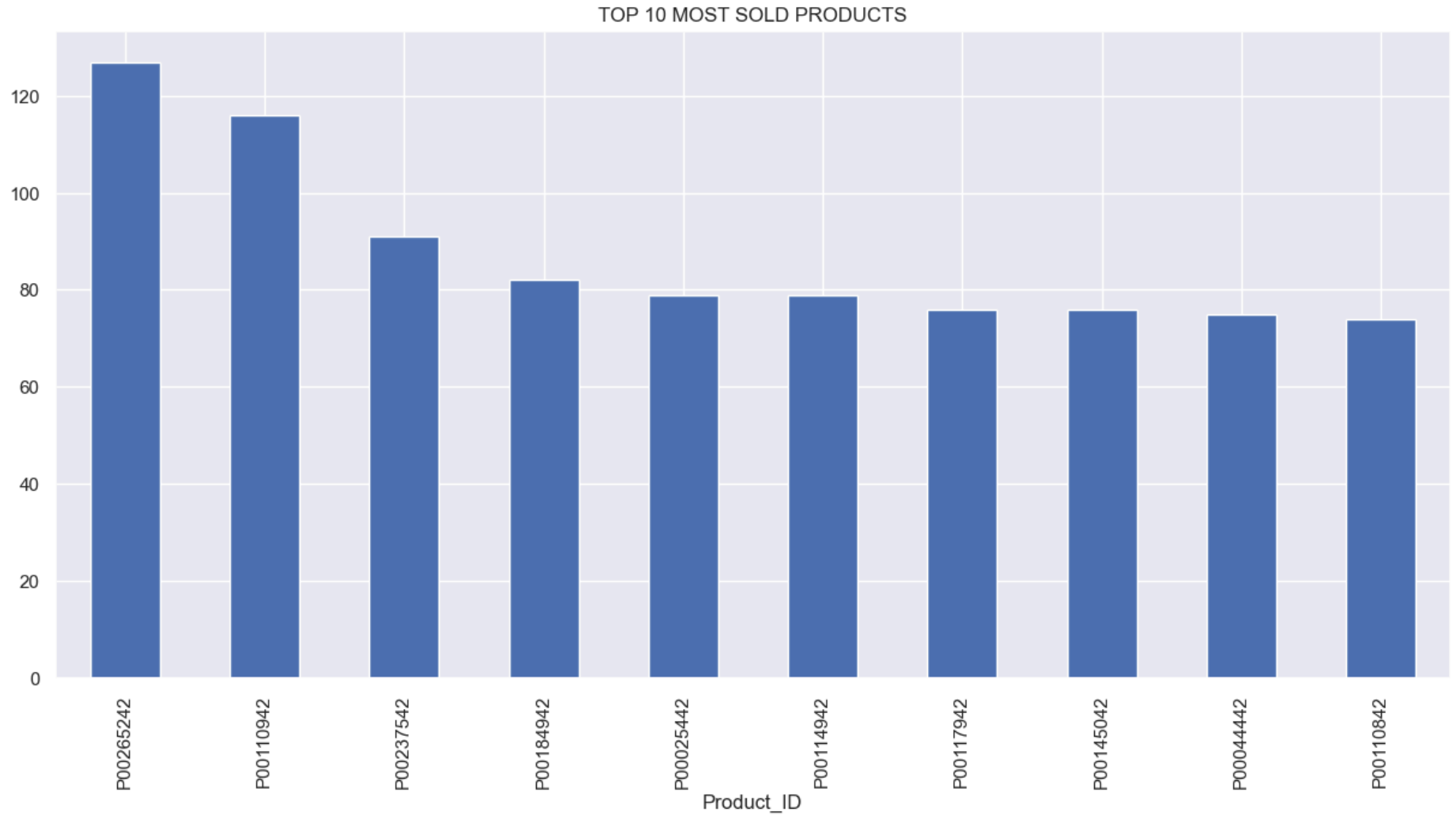


🔍 ***From above graphs we can see that most of the sold products are from Food, Clothing , Electronics & Footware Category...***

In [ ]:

In [67]:
```
plt.figure(figsize=(15,7))
data.groupby(['Product_ID'])['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='bar')
```

```
plt.title('TOP 10 MOST SOLD PRODUCTS');
```



In [68]:
```
data.head(2)
```

Out[68]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Pro Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934

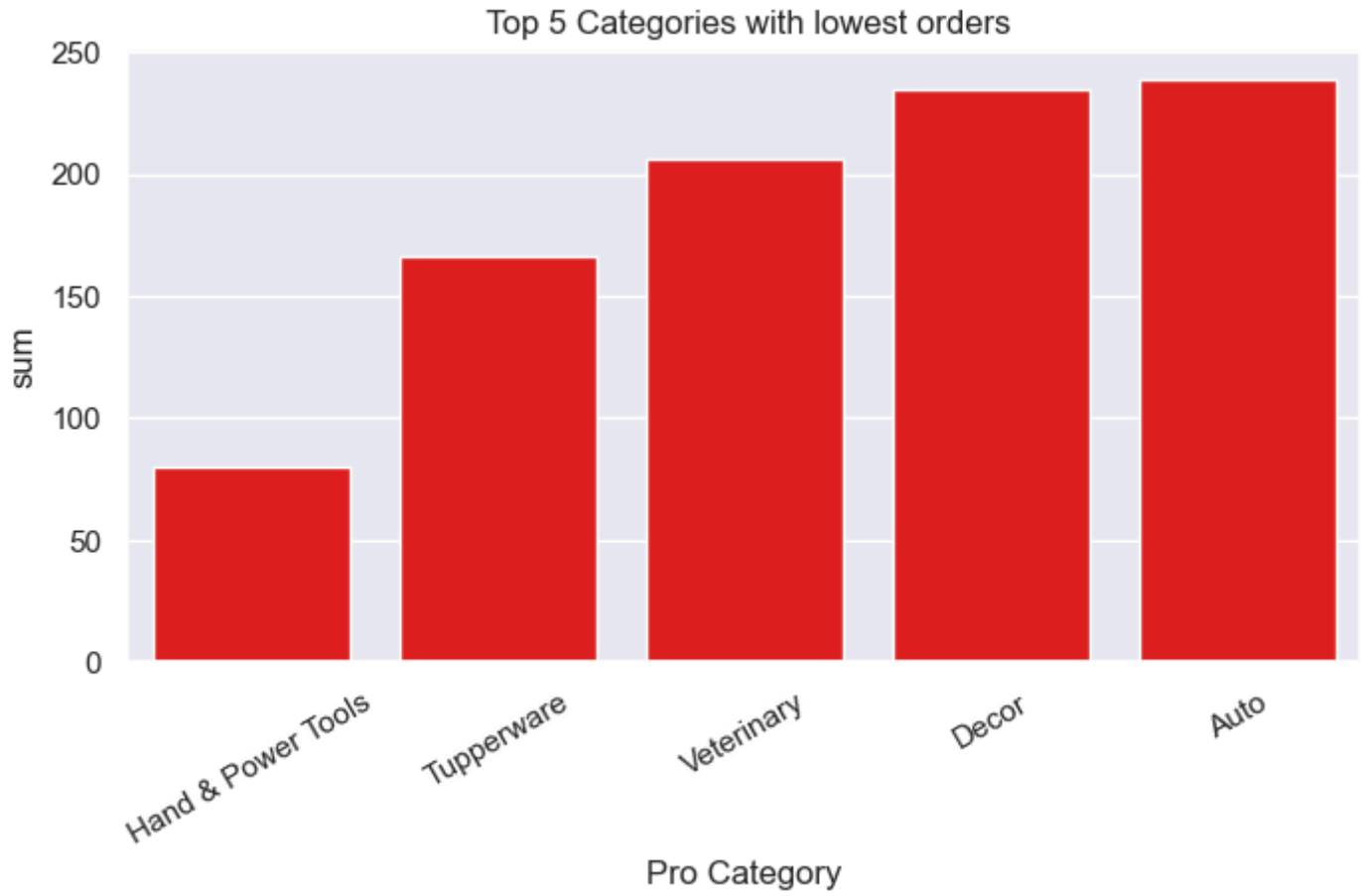
In [69]:
```
gr=data.groupby(['Pro Category'])['Orders'].sum().reset_index(name='sum').sort_values(by='sum',ascending=True).head(5)
```

In [70]:
```
gr
```

Out[70]:

	Pro Category	sum
10	Hand & Power Tools	80
16	Tupperware	166
17	Veterinary	206
4	Decor	235
0	Auto	239

In [71]:
```
plt.figure(figsize=(8,4))
sns.barplot(x='Pro Category',y='sum',data=gr,color='red')
plt.xticks(rotation=30)
plt.title(' Top 5 Categories with lowest orders');
```



CHECK RELATIONS:

In [72]:
```
data.head(3)
```

Out[72]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Pro Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924

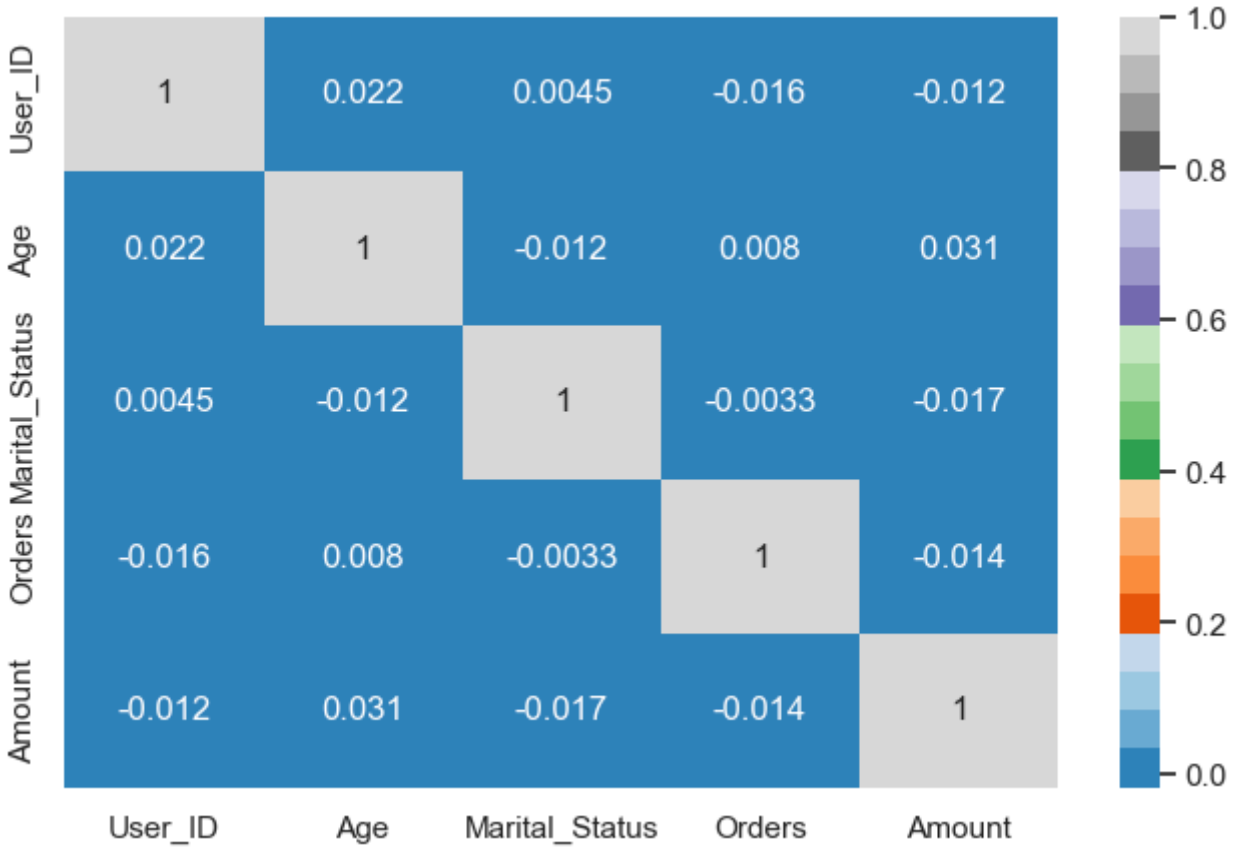
In [73]:
```
data.corr(numeric_only=True)
```

Out[73]:

	User_ID	Age	Marital_Status	Orders	Amount
User_ID	1.000000	0.022069	0.004504	-0.016181	-0.012066
Age	0.022069	1.000000	-0.012150	0.007968	0.030948
Marital_Status	0.004504	-0.012150	1.000000	-0.003270	-0.017255
Orders	-0.016181	0.007968	-0.003270	1.000000	-0.013533
Amount	-0.012066	0.030948	-0.017255	-0.013533	1.000000

In [74]:
```
plt.figure(figsize=(8,5))
sns.heatmap(data.corr(numeric_only=True),cmap="tab20c",annot=True)
```

Out[74]:
```
<Axes: >
```



🔍 *There are no any type of relations between these columns...*

In [ ]:

FINAL CONCLUSION ⚡ ⚡ :

➡️ UnMarried womens & mens, mostely Orders are from Central Zone & South Zone, age group 25-36 yrs from Uttar Pradesh, Maharastra and Karnataka working in Information Technology, Healthcare ,Aviation and Banking are more likely to buy products from Food, Clothing and Electronics category.

➡️ On the other side Hand - Power Tools,Tupperware, Veterinary, Decor & Auto are Top 5 Categories with very lowest orders...

THANK YOU ! ❤️

⬆️