"Analysis of Customer Behavior and Sales Patterns: Uncovering Insights and Opportunities for Business Growth"

In [20]:
```
# import python libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [21]:
```
# import csv file
df = pd.read_csv(r"C:\Users\Gracy Pauluse Ingle\Downloads\Python_Diwali_Sal
```

In [22]:
```
df.head()
```

Out[22]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Mah
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar
3	1001425	Sudevi	P00237842	M	0-17	16	0	Ka
4	1000588	Joni	P00057942	M	26-35	28	1	

"Data Cleaning: Preparing and Refining the Dataset"

In [23]:
```
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 [24]:
```
#drop unrelated/blank columns
df.drop(["Status", "unnamed1"], axis = 1 ,inplace = True)
```

In [25]:
```
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 [26]:
```
#check for null values
pd.isnull(df).sum()
```

Out[26]:
```
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 [27]:
```
# drop null values
df.dropna(inplace = True)
```

In [28]:
```
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  float64
dtypes: float64(1), int64(4), object(8)
memory usage: 1.2+ MB
```

"Converting 'Amount' Column from Float to Integer".

Explanation: In this step, we will convert the 'Amount' column in the DataFrame from a float data type to an integer data type.

In [29]:
```
df["Amount"]=df["Amount"].astype(int)
```

In [30]:
```
df["Amount"].dtypes
```

Out[30]:
```
dtype('int32')
```

In [114…
```
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 [115…

```
df.columns
```

Out[115]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Categor
y',
       'Orders', 'Amount'],
      dtype='object')
```

In [116…
```
#rename column
df =df.rename(columns={"Marital_Status" : "Marital"})
```

In [117…
```
df.columns
```

Out[117]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital', 'State', 'Zone', 'Occupation', 'Product_Category', 'Orde
rs',
       'Amount'],
      dtype='object')
```

In [118…
```
df.describe()
```

Out[118]:

	User_ID	Age	Marital	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

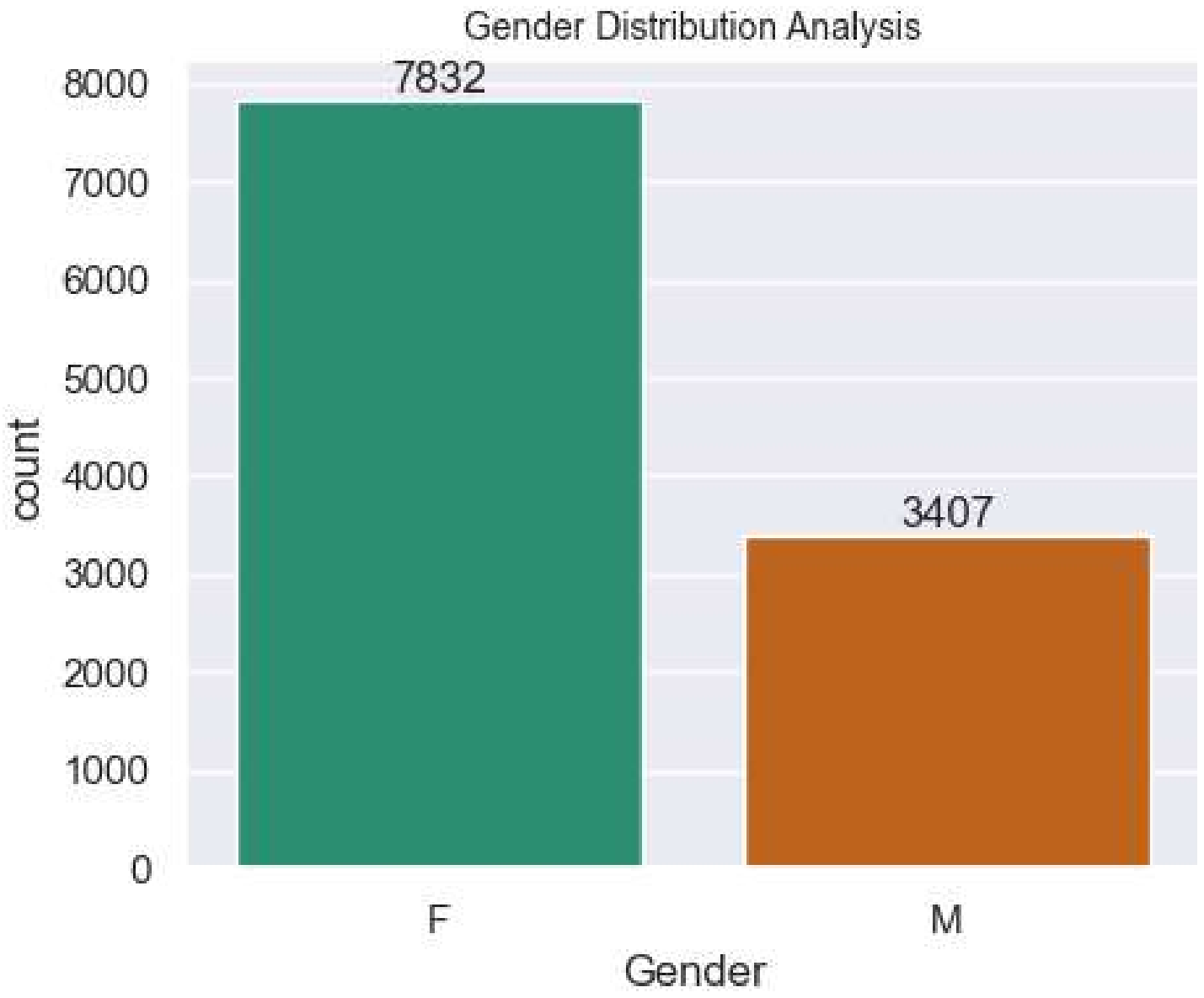
Exploratory data analysis (EDA)

"Gender: Understanding Patterns and Insights"

In [139…

```
ax = sns.countplot(x ="Gender", data = df)
sns.set(rc={'figure.figsize':(5,4)})
```

```
sns.set_palette("Dark2")
plt.title("Gender Distribution Analysis"  , fontsize = 10)
for bars in ax.containers:
    ax.bar_label(bars)
```

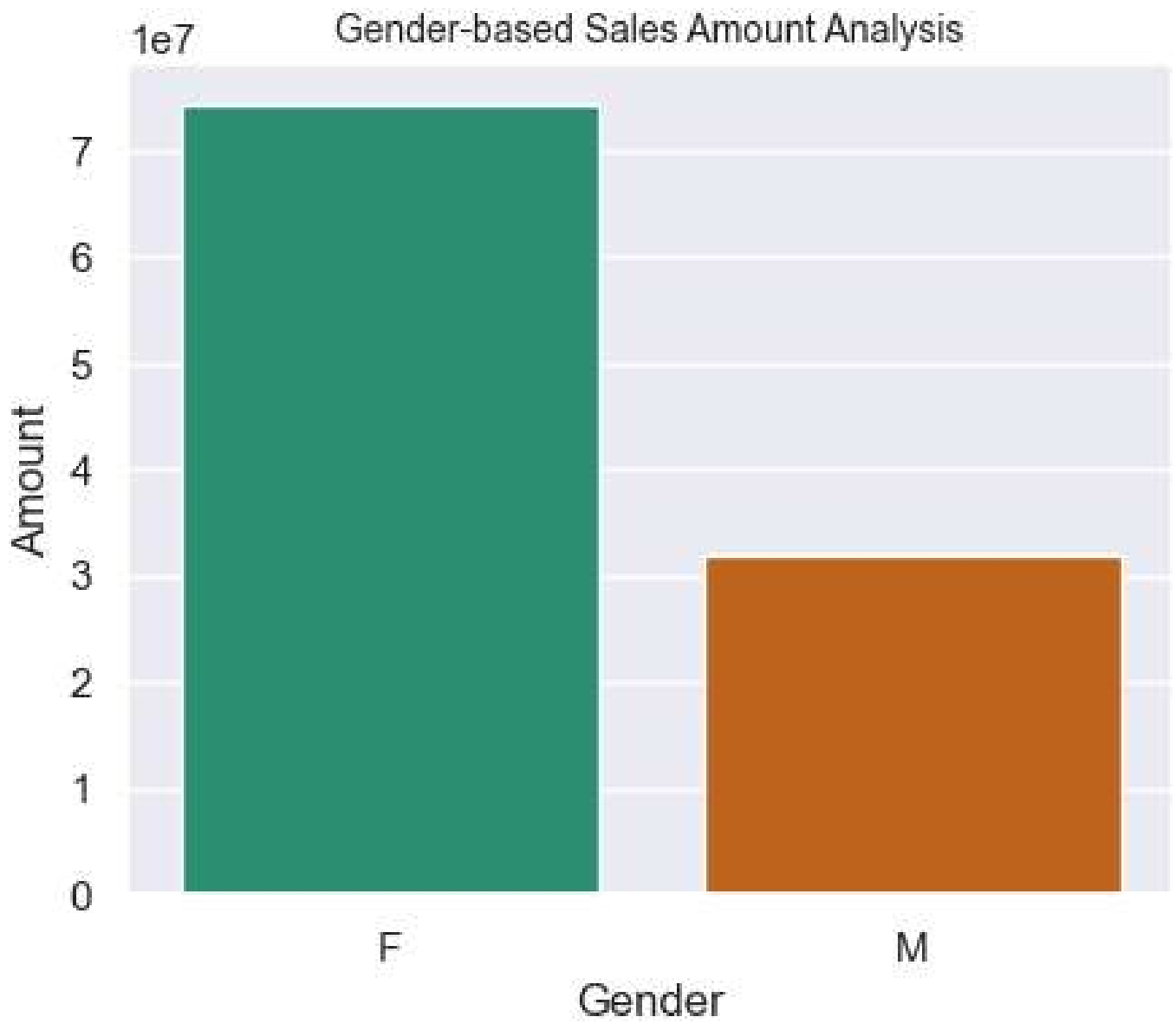


In [120…

```
sales_gen = df.groupby(['Gender'], as_index = False)['Amount'].sum().sort_v
sns.set(rc={'figure.figsize':(5,4)})
```

```
sns.set_palette("Dark2")
plt.title( "Gender-based Sales Amount Analysis" , fontsize = 10)
sns.barplot(x = 'Gender', y = 'Amount', data = sales_gen)
```

Out[120]: <Axes: title={'center': 'Gender-based Sales Amount Analysis'}, xlabel='Gender', ylabel='Amount'>

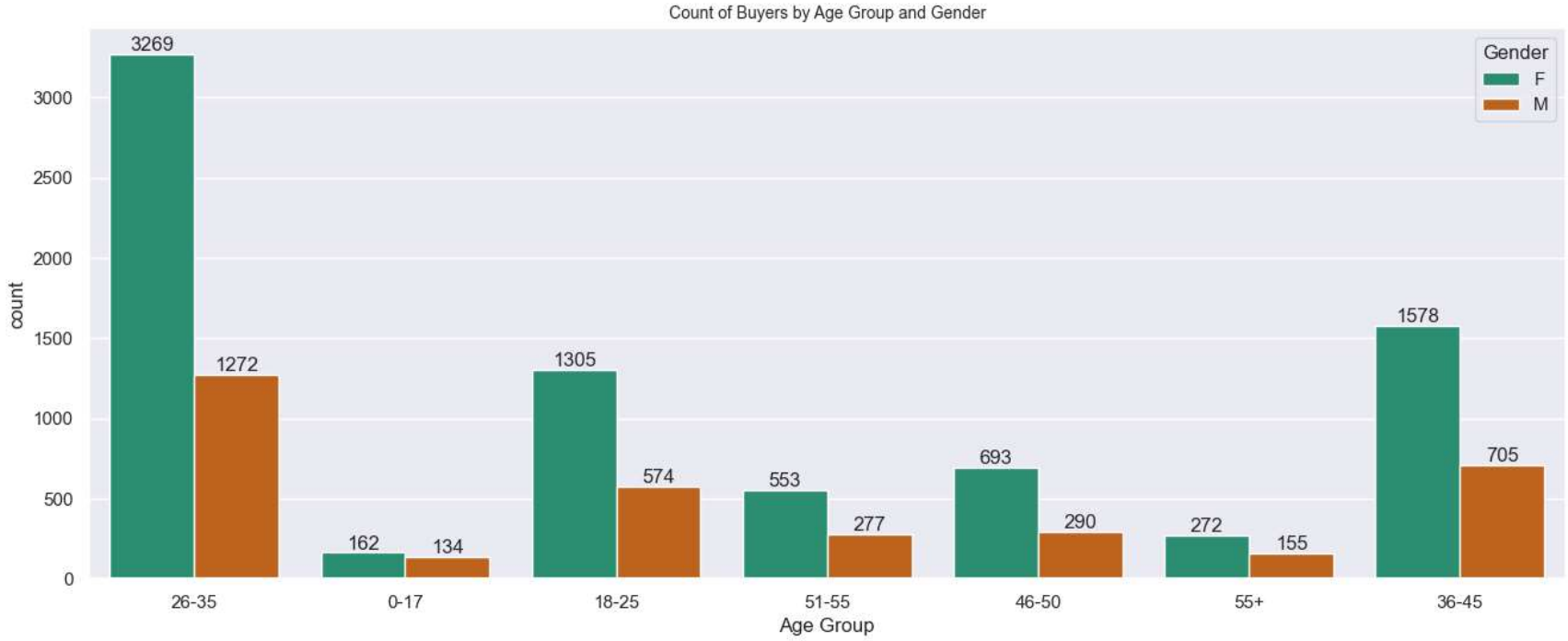

Gender-based Sales Amount Analysis
1e7
7
6
5
4
3
2
1
0
Amount
F
M
Gender

"From above graphs we can see that "Analysis of Gender: Females as the Dominant Buyers with Strong Purchasing Power"

"Age : Uncovering Trends and Patterns"

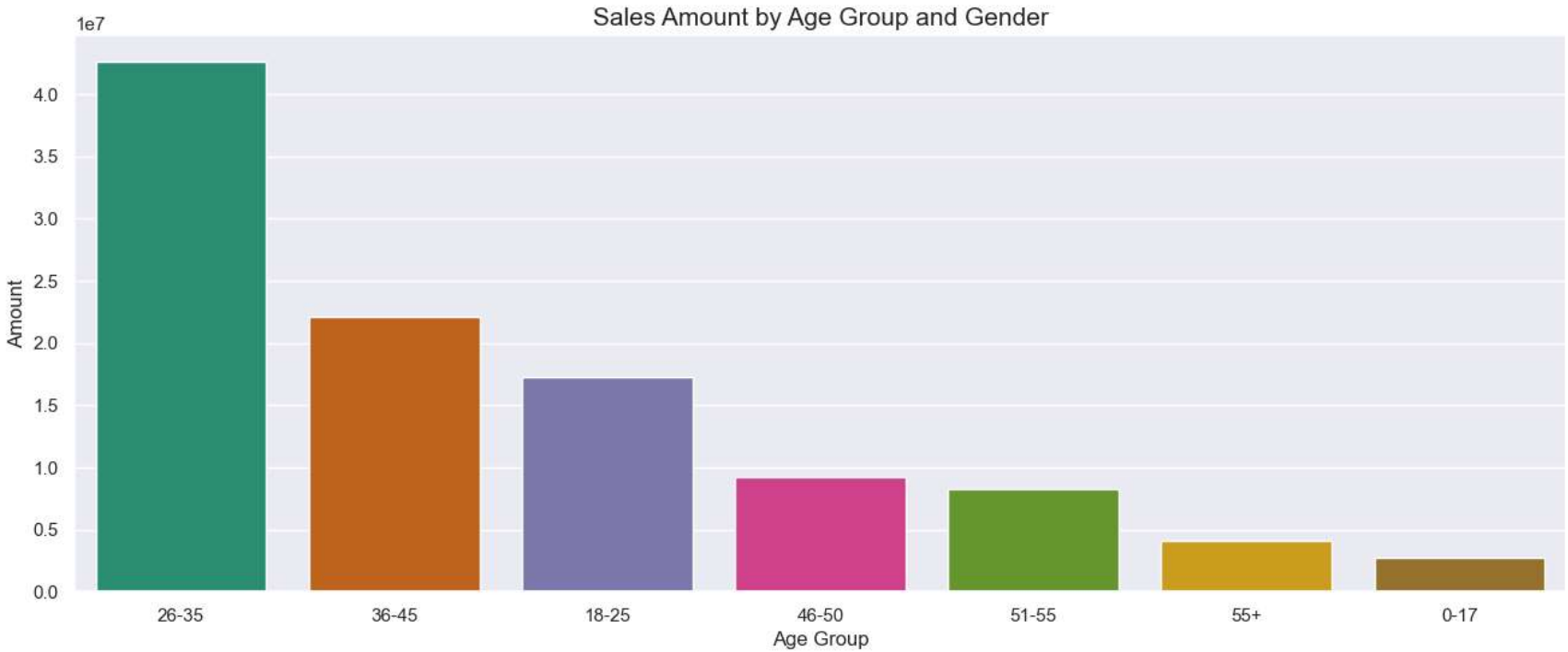
In [142…

```
ax = sns.countplot(data = df, x ="Age Group", hue = 'Gender')
sns.set(rc={'figure.figsize':(16,6)})
plt.title( "Count of Buyers by Age Group and Gender" , fontsize = 10)
sns.set_palette("Dark2")
for bars in ax.containers:
    ax.bar_label(bars)
```



In [122…
```
sales_Age = df.groupby(['Age Group'], as_index = False)['Amount'].sum().sor
sns.barplot(x = 'Age Group', y = 'Amount', data = sales_Age)
sns.set(rc={'figure.figsize':(16,6)})
sns.set_palette("Dark2")
plt.title("Sales Amount by Age Group and Gender", fontsize = 15)
```

Out[122]:
```
Text(0.5, 1.0, 'Sales Amount by Age Group and Gender')
```



"Age Analysis: The Largest Proportion of Female Buyers Falls within the 26-35 Year Age Group"

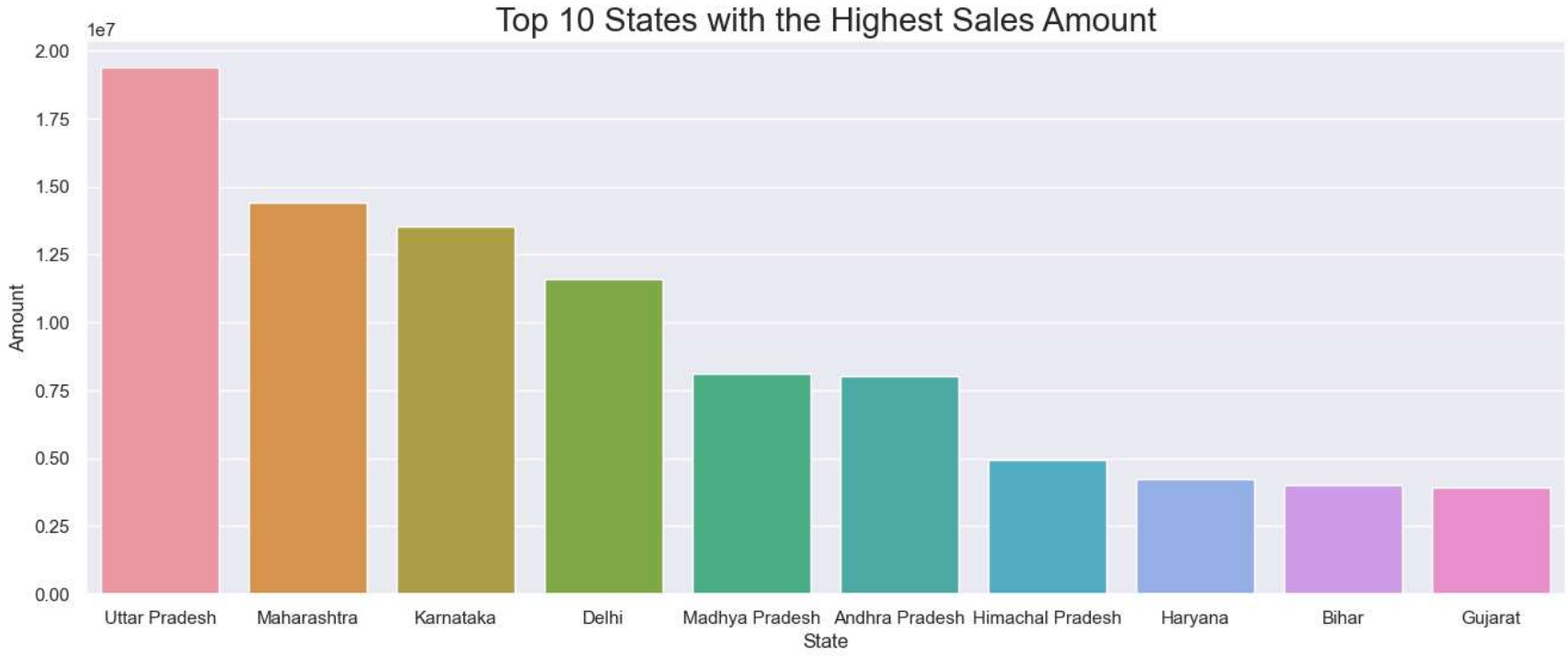
"State: Analyzing Geographical Patterns and Insights"

In [123…

```
# "Top 10 States with the Highest Sales"
sales_state = df.groupby(['State'], as_index = False)['Amount'].sum().sort_

sns.set(rc={'figure.figsize':(16,6)})
sns.set_palette("Dark2")
```

```
plt.title("Top 10 States with the Highest Sales Amount", fontsize = 20)
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

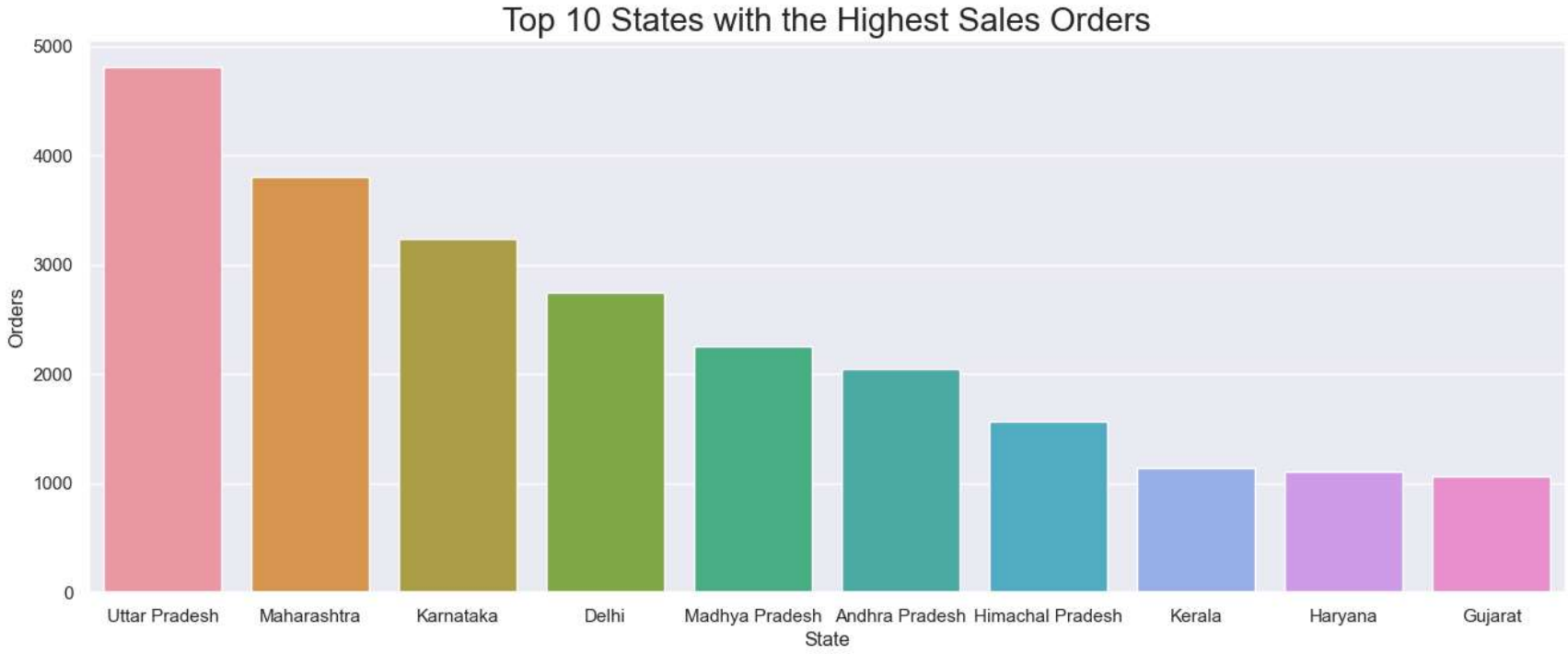
Out[123]:
```
<Axes: title={'center': 'Top 10 States with the Highest Sales Amount'}, xl
abel='State', ylabel='Amount'>
```



In [124…
```
#Top 10 States with the Highest Order Count"
sales_state = df.groupby(['State'], as_index = False)['Orders'].sum().sort_

sns.set(rc={'figure.figsize':(16,6)})
sns.set_palette("Dark2")
```

```
plt.title( "Top 10 States with the Highest Sales Orders" , fontsize = 20)
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

Out[124]: <Axes: title={'center': 'Top 10 States with the Highest Sales Orders'}, xlabel='State', ylabel='Orders'>

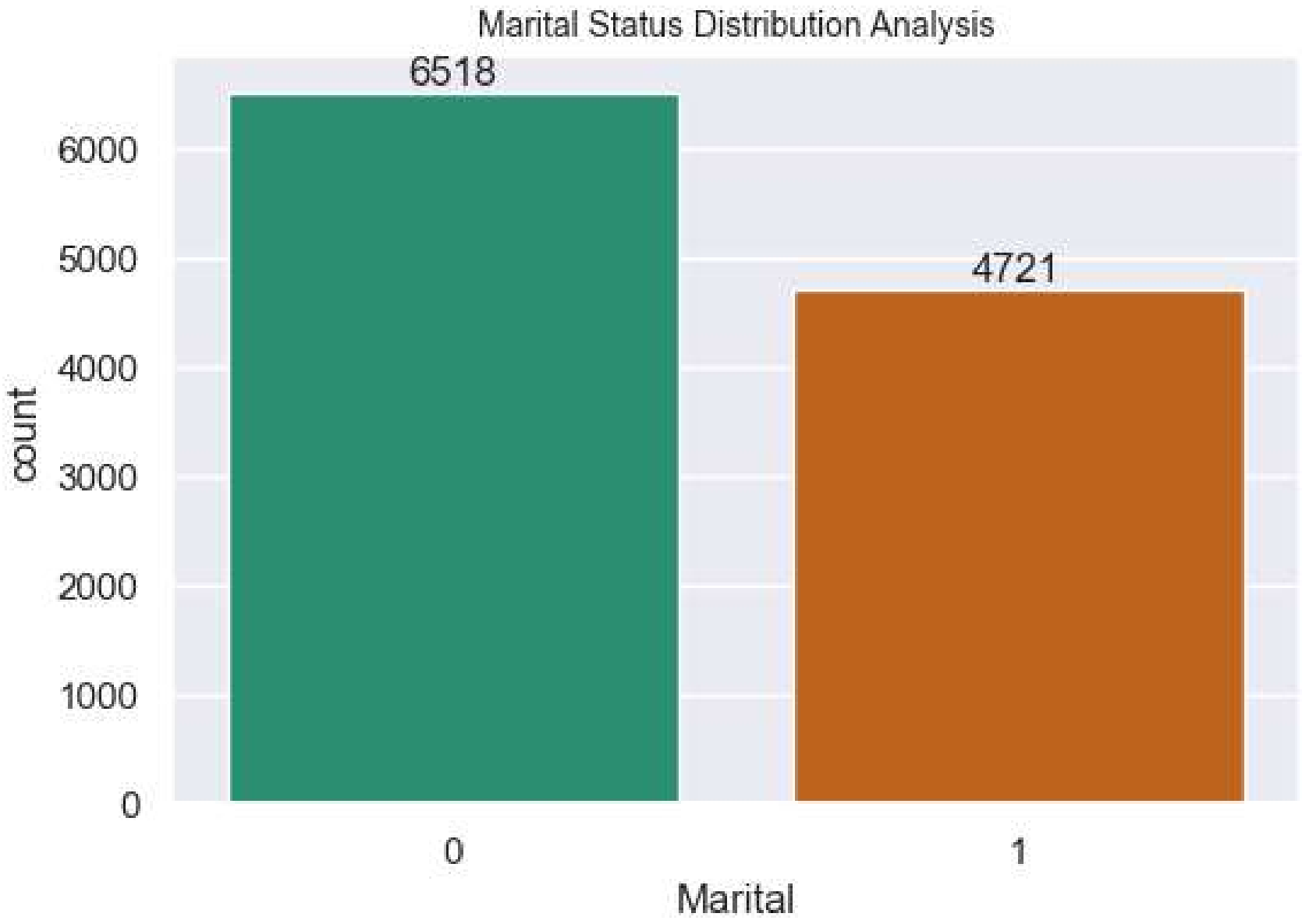


The graphs indicate that Uttar Pradesh, Maharashtra, and Karnataka have the highest number of orders and contribute the most to the total sales or amount.

"Marital: Understanding Relationship Patterns and Implications"

In [147…

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

sns.set(rc={'figure.figsize':(10,4)})
sns.set_palette("Dark2")
plt.title( "Marital Status Distribution Analysis", fontsize = 10)
for bars in ax.containers:
    ax.bar_label(bars)
```

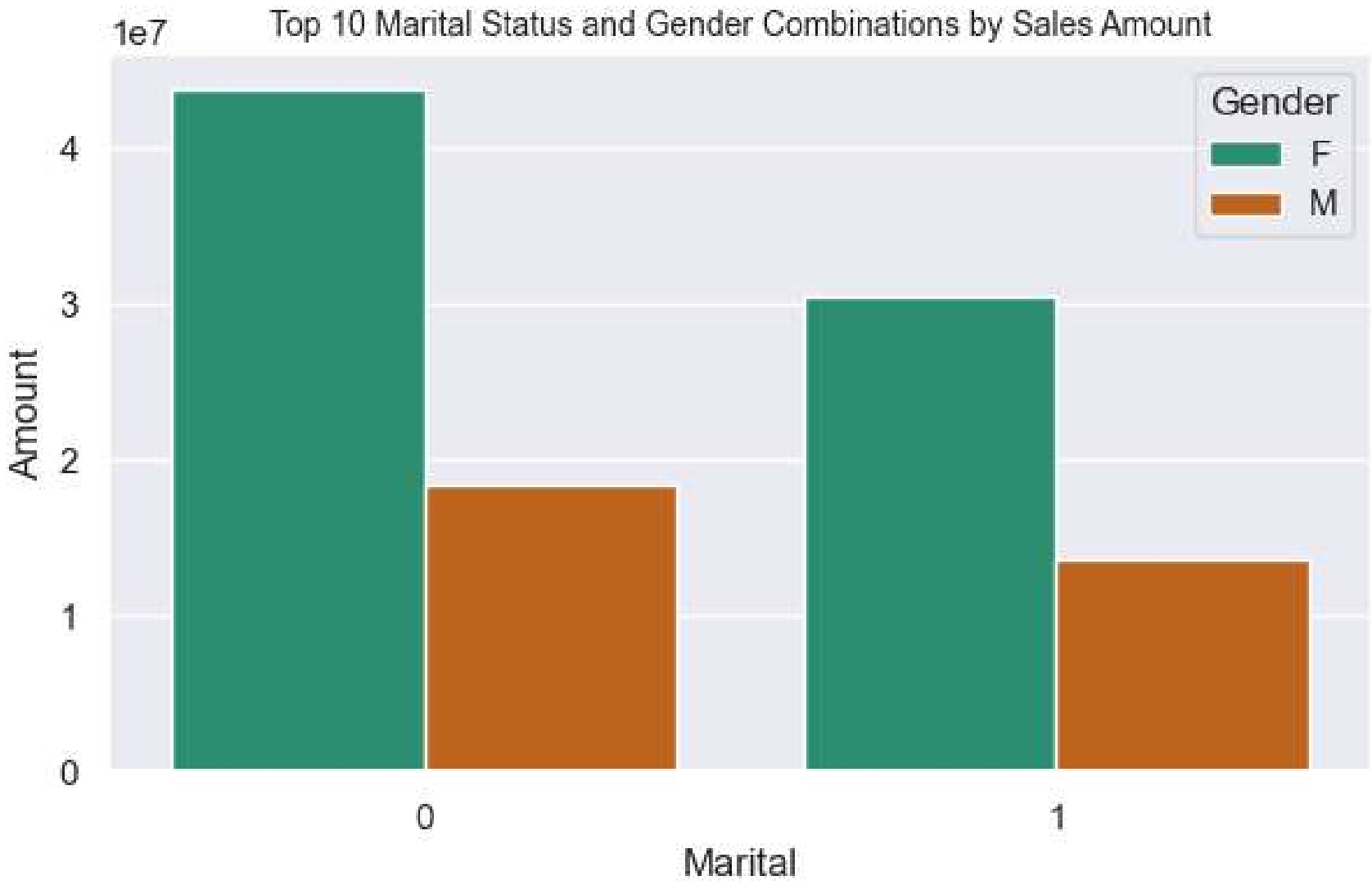


In [126…

```
sales_Marital = df.groupby(["Marital", "Gender"], as_index = False)['Amount
sns.set(rc={'figure.figsize':(7,4)})
```

```
sns.set_palette("Dark2")
plt.title( "Top 10 Marital Status and Gender Combinations by Sales Amount"
sns.barplot(data = sales_Marital, x = 'Marital',y= 'Amount', hue = "Gender"
```

Out[126]: <Axes: title={'center': 'Top 10 Marital Status and Gender Combinations by Sales Amount'}, xlabel='Marital', ylabel='Amount'>

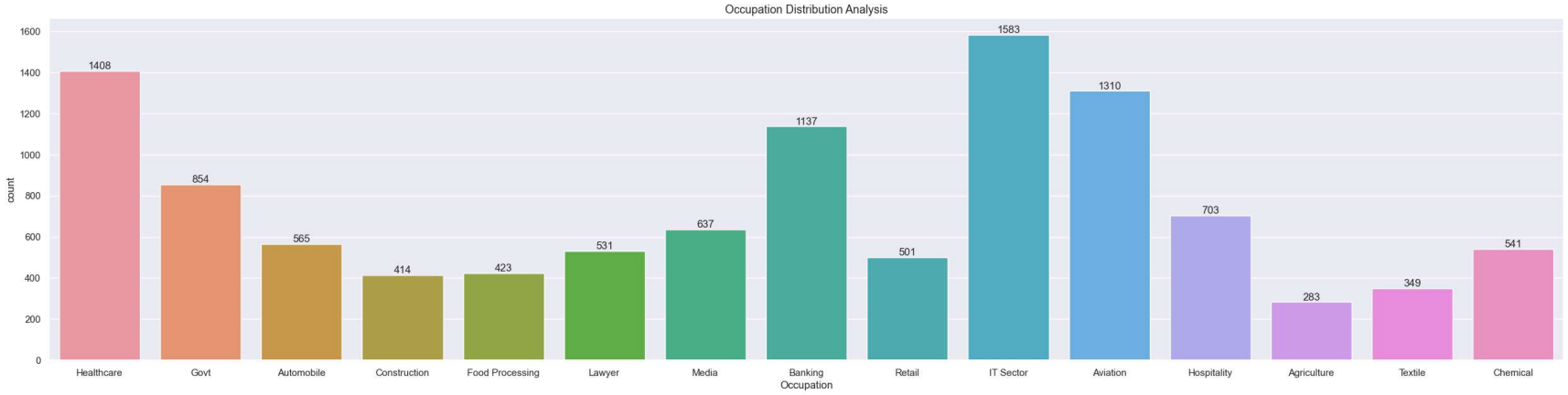


Based on the graphs, it is evident that the majority of buyers are married women, and they exhibit a significant purchasing power.

"Occupation : Understanding Relationship Patterns and Implications"

In [152…

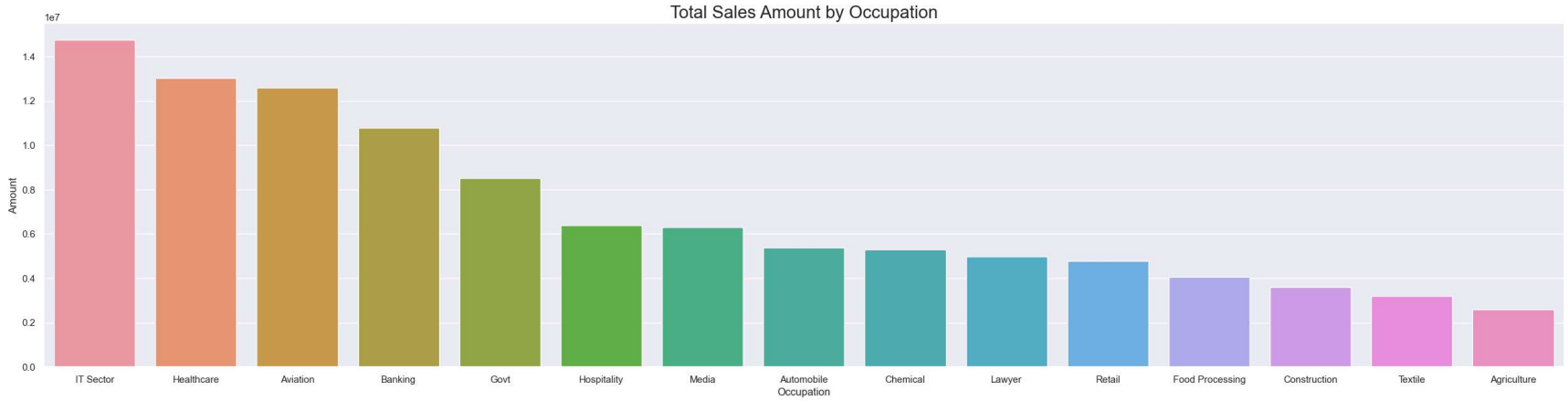
```
ax = sns.countplot(data = df, x = 'Occupation')

sns.set(rc={'figure.figsize':(31,7)})
sns.set_palette("Dark2")
plt.title( "Occupation Distribution Analysis" , fontsize = 13)
for bars in ax.containers:
    ax.bar_label(bars)
```



In [128…

```
sales_Occupation = df.groupby(["Occupation"], as_index = False)['Amount'].s
sns.barplot(data = sales_Occupation , x = "Occupation", y = "Amount")
plt.title("Total Sales Amount by Occupation", fontsize = 20)
```

Out[128]: Text(0.5, 1.0, 'Total Sales Amount by Occupation')

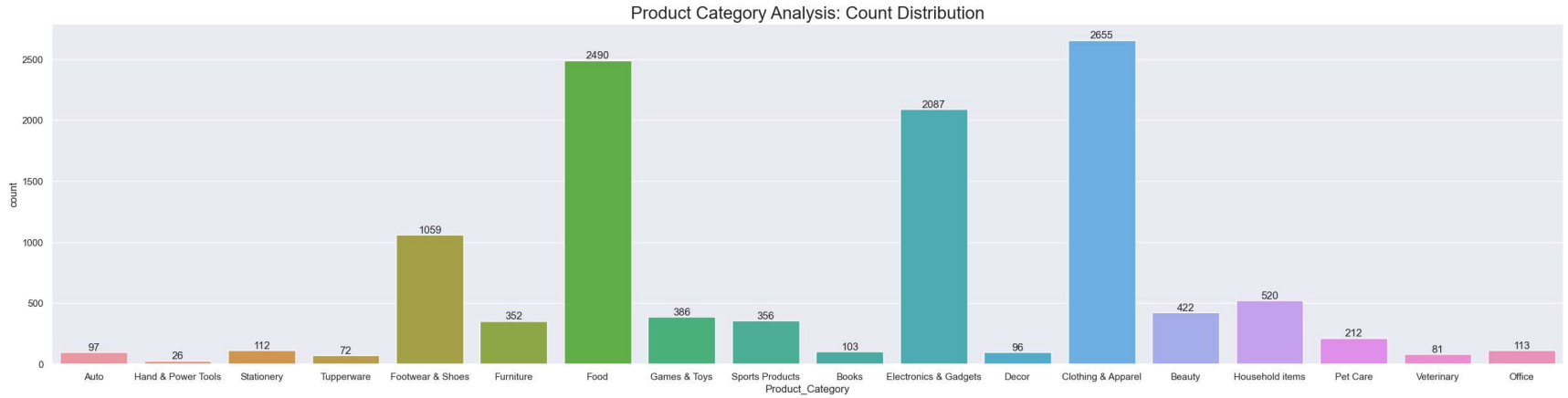


"The graphs indicate that a significant number of buyers are employed in the IT, healthcare, and aviation sectors"

"Product_Category : Understanding Relationship Patterns and Implications"

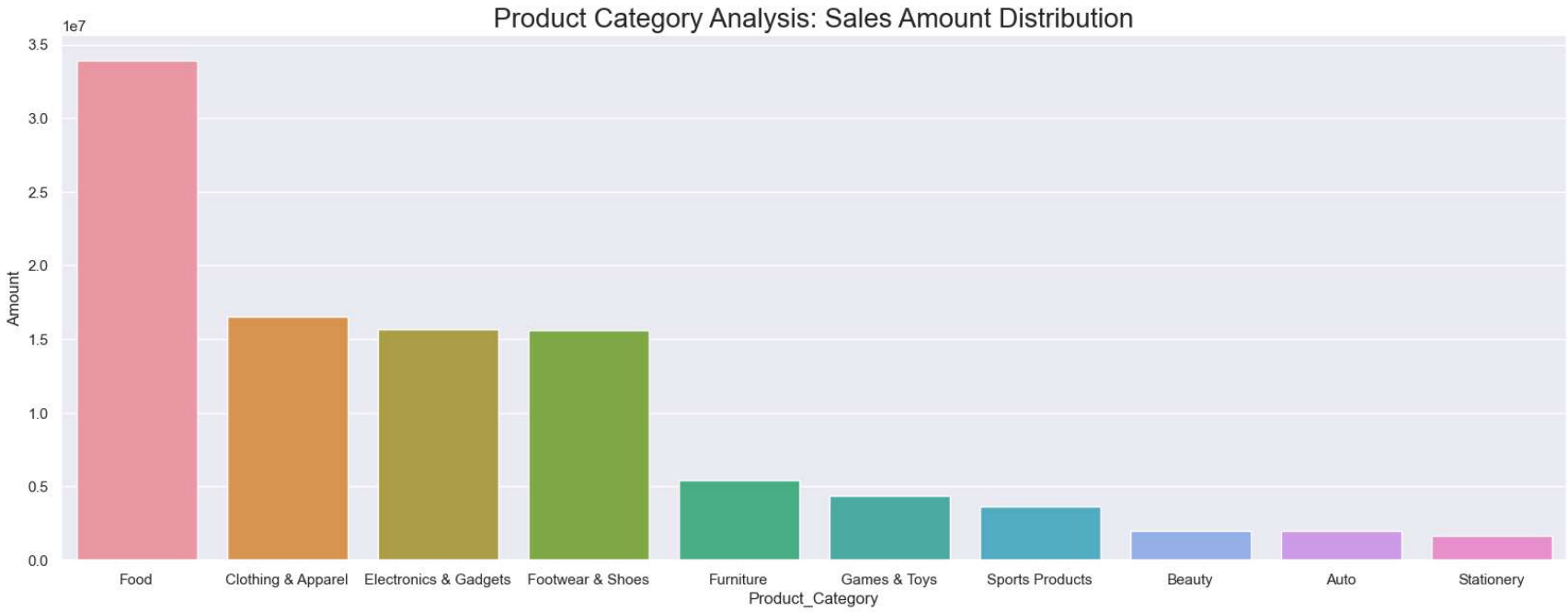
In [129…

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

sns.set(rc={'figure.figsize':(32,7)})
sns.set_palette("Dark2")
plt.title( "Product Category Analysis: Count Distribution", fontsize = 20)
for bars in ax.containers:
    ax.bar_label(bars)
```



In [130…
```
 sales_Product_Category = df.groupby(["Product_Category"], as_index = False
sns.set(rc={'figure.figsize':(20,7)})
sns.set_palette("Dark2")
sns.barplot(data =sales_Product_Category, x = "Product_Category", y = "Amou
plt.title("Product Category Analysis: Sales Amount Distribution", fontsize
```

Out[130]:
```
Text(0.5, 1.0, 'Product Category Analysis: Sales Amount Distribution')
```



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

"Zone: Understanding Relationship Patterns and Implications"

In [131…

```
sales_Zone = df.groupby(["Zone"], as_index = False)['Orders'].sum().sort_va
sns.barplot(data = sales_Zone, x= "Zone" , y = "Orders")
sns.set_palette("Dark2")
plt.title("Zone with the Highest Orders", fontsize = 20)
```

Out[131]: Text(0.5, 1.0, 'Zone with the Highest Orders')

Central zone leads in sales, followed by south and west, emphasizing their significance for business success.

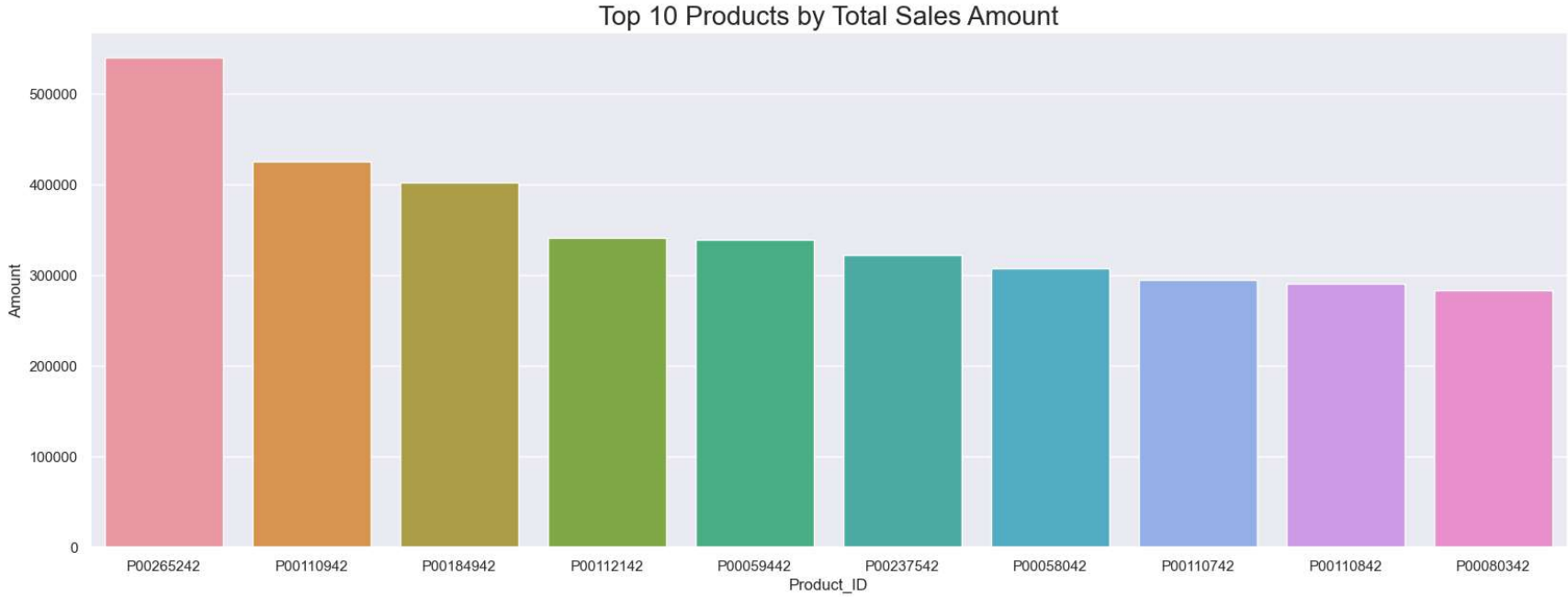
Top 10 Product by Amount

In [132…

```
# Top 10 Product by Amount
sales_Product = df.groupby(["Product_ID"], as_index = False)['Amount'].sum(
sns.set(rc={'figure.figsize':(20,7)})
sns.set_palette("Dark2")
```

```
sns.barplot(data =sales_Product, x = "Product_ID", y = "Amount")
plt.title("Top 10 Products by Total Sales Amount", fontsize = 20)
```

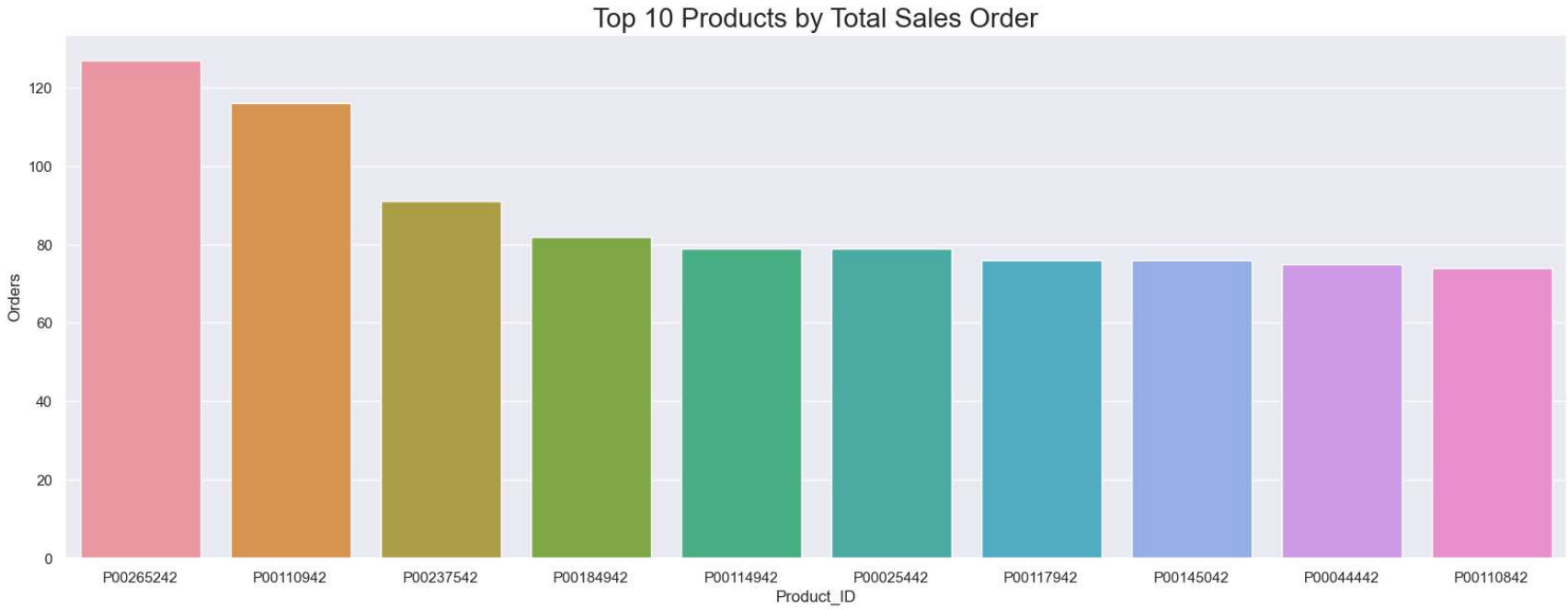
Out[132]: Text(0.5, 1.0, 'Top 10 Products by Total Sales Amount')



In [133…

```
# Top 10 Product by orders
sales_Product = df.groupby(["Product_ID"], as_index = False)['Orders'].sum(
sns.set(rc={'figure.figsize':(20,7)})
sns.set_palette("Dark2")
sns.barplot(data =sales_Product, x = "Product_ID", y = "Orders")
plt.title("Top 10 Products by Total Sales Order", fontsize = 20)
```

Out[133]: Text(0.5, 1.0, 'Top 10 Products by Total Sales Order')



Insights

Targeting married women aged 26-35 years from Uttar Pradesh, Maharashtra, and Karnataka, working in IT, healthcare, and aviation sectors, can lead to higher sales. Focus on offering products in the food, clothing, and electronics categories to meet their preferences and maximize revenue.It is advisable to provide attractive incentives such as discounts, coupons, and special offers.

complete project on GitHub: https://github.com/davidingle188 follow me on linkedin : https://www.linkedin.com/in/davidingle18082000/

Thank you!