SOURCE CODE

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
diwaliData = pd.read_csv(r"D:\FIFTHFORCE\DiwaliSalesData.csv", encoding = 'unicode_escape')
```

```
diwaliData.head(10) # Gives the top 10 elements...
```

User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed1
1002903.0	Sanskriti	P00125942	F	26-35	28	0.0	Maharashtra	Western	Healthcare	Auto	1.0	23952	NaN	NaN
1000732.0	Kartik	P00110942	F	26-35	35	1.0	Andhra Pradesh	Southern	Govt	Auto	3.0	23934	NaN	NaN
1001990.0	Bindu	P00118542	F	26-35	35	1.0	Uttar Pradesh	Central	Automobile	Auto	3.0	23924	NaN	NaN
1001425.0	Sudevi	P00237842	M	0-17	16	0.0	Karnataka	Southern	Construction	Auto	2.0	23912	NaN	NaN
1000588.0	Joni	P00057942	M	26-35	28	1.0	Gujarat	Western	Food Processing	Auto	2.0	23877	NaN	NaN
1000588.0	Joni	P00057942	M	26-35	28	1.0	Himachal Pradesh	Northern	Food Processing	Auto	1.0	23877	NaN	NaN
1001132.0	Balk	P00018042	F	18-25	25	1.0	Uttar Pradesh	Central	Lawyer	Auto	4.0	23841	NaN	NaN
1002092.0	Shivangi	P00273442	F	55+	61	0.0	Maharashtra	Western	IT Sector	Auto	1.0	NaN	NaN	NaN
1003224.0	Kushal	P00205642	M	26-35	NaN	0.0	Uttar Pradesh	Zentral	Govt	Auto	2.0	23809	NaN	NaN
1003650.0	Ginny	P00031142	F	26-35	NaN	1.0	Andhra Pradesh	Southern	Media	Auto	4.0	23799.99	NaN	NaN

```
# Creating a copt so that data should be safe...
dataCopy = diwaliData
dataCopy.head(10)
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnai
0	1002903.0	Sanskriti	P00125942	F	26-35	28	0.0	Maharashtra	Western	Healthcare	Auto	1.0	23952	NaN	
1	1000732.0	Kartik	P00110942	F	26-35	35	1.0	Andhra Pradesh	Southern	Govt	Auto	3.0	23934	NaN	
2	1001990.0	Bindu	P00118542	F	26-35	35	1.0	Uttar Pradesh	Central	Automobile	Auto	3.0	23924	NaN	
3	1001425.0	Sudevi	P00237842	M	0-17	16	0.0	Karnataka	Southern	Construction	Auto	2.0	23912	NaN	
4	1000588.0	Joni	P00057942	M	26-35	28	1.0	Gujarat	Western	Food Processing	Auto	2.0	23877	NaN	
5	1000588.0	Joni	P00057942	M	26-35	28	1.0	Himachal Pradesh	Northern	Food Processing	Auto	1.0	23877	NaN	
6	1001132.0	Balk	P00018042	F	18-25	25	1.0	Uttar Pradesh	Central	Lawyer	Auto	4.0	23841	NaN	
7	1002092.0	Shivangi	P00273442	F	55+	61	0.0	Maharashtra	Western	IT Sector	Auto	1.0	NaN	NaN	
8	1003224.0	Kushal	P00205642	M	26-35	NaN	0.0	Uttar Pradesh	Zentral	Govt	Auto	2.0	23809	NaN	
9	1003650.0	Ginny	P00031142	F	26-35	NaN	1.0	Andhra Pradesh	Southern	Media	Auto	4.0	23799.99	NaN	

[16]:
```
# Data Pre-processing...
# -- Data Pre profiling.....
# -- Data Cleaning....
# -- Data Consistancy i.e. Typing error or formating error....
# Outlier Detection and Removal...
dataCopy.shape
```

[16]:
```
(11345, 15)
```

[17]:
```
percentageOfNull = ((dataCopy.isna().sum()) / (dataCopy.shape[0])) * 100
percentageOfNull
```

[17]:
```
User_ID              0.211547
Cust_name            0.008814
Product_ID           0.035258
Gender               0.035258
Age Group            0.052887
Age                  0.096959
Marital_Status       0.052887
State                0.114588
Zone                 0.096959
Occupation           0.052887
Product_Category     0.096959
Orders               0.026443
Amount               0.123402
Status             100.000000
unnamed1           100.000000
dtype: float64
```

```
# Creating a copt so that data should be safe...
dataCopy = diwaliData
dataCopy.head(10)
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unna
0	1002903.0	Sanskriti	P00125942	F	26-35	28	0.0	Maharashtra	Western	Healthcare	Auto	1.0	23952	NaN	
1	1000732.0	Kartik	P00110942	F	26-35	35	1.0	Andhra Pradesh	Southern	Govt	Auto	3.0	23934	NaN	
2	1001990.0	Bindu	P00118542	F	26-35	35	1.0	Uttar Pradesh	Central	Automobile	Auto	3.0	23924	NaN	
3	1001425.0	Sudevi	P00237842	M	0-17	16	0.0	Karnataka	Southern	Construction	Auto	2.0	23912	NaN	
4	1000588.0	Joni	P00057942	M	26-35	28	1.0	Gujarat	Western	Food Processing	Auto	2.0	23877	NaN	
5	1000588.0	Joni	P00057942	M	26-35	28	1.0	Himachal Pradesh	Northern	Food Processing	Auto	1.0	23877	NaN	
6	1001132.0	Balk	P00018042	F	18-25	25	1.0	Uttar Pradesh	Central	Lawyer	Auto	4.0	23841	NaN	
7	1002092.0	Shivangi	P00273442	F	55+	61	0.0	Maharashtra	Western	IT Sector	Auto	1.0	NaN	NaN	
8	1003224.0	Kushal	P00205642	M	26-35	NaN	0.0	Uttar Pradesh	Zentral	Govt	Auto	2.0	23809	NaN	
9	1003650.0	Ginny	P00031142	F	26-35	NaN	1.0	Andhra Pradesh	Southern	Media	Auto	4.0	23799.99	NaN	

[20]:
```
# To see all columns....
allColumns = dataCopy.columns
allColumns
```

[20]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount', 'Status', 'unnamed1'],
      dtype='object')
```

[21]:
```
# Numeric columns....
numeric_Columns = dataCopy.select_dtypes(include = np.number)
numeric_Columns
```

[21]:

	User_ID	Marital_Status	Orders	Status	unnamed1
0	1002903.0	0.0	1.0	NaN	NaN
1	1000732.0	1.0	3.0	NaN	NaN
2	1001990.0	1.0	3.0	NaN	NaN
3	1001425.0	0.0	2.0	NaN	NaN
4	1000588.0	1.0	2.0	NaN	NaN
...	...	...	...	...	...
11340	11001657.0	0.0	24.0	NaN	NaN
11341	11003022.0	0.0	4.0	NaN	NaN
11342	11001657.0	0.0	2.0	NaN	NaN
11343	11003022.0	0.0	4.0	NaN	NaN
11344	11001657.0	0.0	4.0	NaN	NaN

11345 rows × 5 columns

[22]:
```
numeric_Columns_name = dataCopy.select_dtypes(include = np.number).columns
numeric_Columns_name
```

[22]:
```
Index(['User_ID', 'Marital_Status', 'Orders', 'Status', 'unnamed1'], dtype='object')
```

[23]:
```
non_numeric = dataCopy.select_dtypes(include=object).columns
non_numeric
```

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

[24]:
```
# Duplicate values...
dataCopy.duplicated()
```

[24]:
```
0        False
1        False
2        False
3        False
4        False
         ...
11340    False
11341    False
11342    False
11343     True
11344    False
Length: 11345, dtype: bool
```

[25]:
```
# Filtering out the duplicate enetities...
dataCopy = dataCopy[~ dataCopy.duplicated()] # It will insert the non duplicate values...
# To check is duplicate is present or not...
dataCopy.duplicated() # All the values are false, Since
# dataCopy[dataCopy.duplicated()]
```

[25]:
```
0        False
1        False
2        False
3        False
4        False
```

[27]: 6

Data Profiling...

1. 'Amount' -- Need to change the data type from object to integer... -- The rows correspoing to null values should be droped... -- Need to drop the rows correspoinding to the values 'abcde'
2. "Status and unnamed" -- No data, should be dropped...
3. "Orders" -- Null values should be dropped... -- There are outliers which can be removed...
4. "State " -- Have to change the value with Andhra Pradesh... -- Drop the rows corresponding the null values...
5. 'Zone' -- Drop the zones that are null values.. -- Zentral should be replaced with central....
6. 'Product_ID' -- Drop the null values..
7. 'Martial_Status..' -- Null values can be replaced with mode -- 1 can be replaced with 'Married' and 0 will be unmaried -- Will create a new column...
8. 'Age Column' -- Age can we replaced with mode -- DataType needs to be changed from object to numeric...
9. 'Age Group' -- low-high should be replaced with mode..
10. "product Category" -- Null values can be dropped..
11. 'Occupation' -- Null values are to mode..
12. 'Gender' -- Null values will be replaced by Mode

[28]:
```
# Data Cleaning....
# dataCopy.drop(["Status", "unnamed1"], inplace = True, axis = 1)
dataCopy.head(10)
```

[29]:
```
# Cleanig in AMOUNT column...
# Changing the datatype...

dataCopy['Amount'] = pd.to_numeric(dataCopy['Amount'], errors = 'coerce') # erroes are conved to null values...
dataCopy.head(10)
```

```
C:\Users\ashmi\AppData\Local\Temp\ipykernel_11244\1145617877.py:4: 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
  dataCopy['Amount'] = pd.to_numeric(dataCopy['Amount'], errors = 'coerce') # erroes are conved to null values...
```

[29]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnam
0	1002903.0	Sanskriti	P00125942	F	26-35	28	0.0	Maharashtra	Western	Healthcare	Auto	1.0	23952.00	NaN	N
1	1000732.0	Kartik	P00110942	F	26-35	35	1.0	Andhra Pradesh	Southern	Govt	Auto	3.0	23934.00	NaN	N
2	1001990.0	Bindu	P00118542	F	26-35	35	1.0	Uttar Pradesh	Central	Automobile	Auto	3.0	23924.00	NaN	N
3	1001425.0	Sudevi	P00237842	M	0-17	16	0.0	Karnataka	Southern	Construction	Auto	2.0	23912.00	NaN	N
4	1000588.0	Joni	P00057942	M	26-35	28	1.0	Gujarat	Western	Food Processing	Auto	2.0	23877.00	NaN	N
5	1000588.0	Joni	P00057942	M	26-35	28	1.0	Himachal Pradesh	Northern	Food Processing	Auto	1.0	23877.00	NaN	N
6	1001132.0	Balk	P00018042	F	18-25	25	1.0	Uttar Pradesh	Central	Lawyer	Auto	4.0	23841.00	NaN	N
7	1002092.0	Shivangi	P00273442	F	55+	61	0.0	Maharashtra	Western	IT Sector	Auto	1.0	NaN	NaN	N
8	1003224.0	Kushal	P00205642	M	26-35	NaN	0.0	Uttar Pradesh	Zentral	Govt	Auto	2.0	23809.00	NaN	N
9	1003650.0	Ginny	P00031142	F	26-35	NaN	1.0	Andhra Pradesh	Southern	Media	Auto	4.0	23799.99	NaN	N

[30]:
```
# To check amount have null values and drop from all columns....
# OR .dropnull().....
dataCopy.dropna(subset = ['Amount', 'Orders', 'State', 'Zone', 'Product_ID', 'Product_Category'], inplace = True)
```

```
C:\Users\ashmi\AppData\Local\Temp\ipykernel_11244\1850091685.py:3: 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
  dataCopy.dropna(subset = ['Amount', 'Orders', 'State', 'Zone', 'Product_ID', 'Product_Category'], inplace = True)
```

[31]:
```
dataCopy.isnull().sum()
```

[31]:
```
User_ID               23
Cust_name              0
Product_ID             0
Gender                 2
Age Group              5
Age                    9
Marital_Status         5
State                  0
Zone                   0
Occupation             5
Product_Category       0
Orders                 0
Amount                 0
Status             11254
unnamed1           11254
dtype: int64
```

[32]:
```
# State Column and Zone column...
# Replacing the type
# 'Andhra\xa0Pradesh' - 'Nndhra\xa0Pradesh'
dataCopy['State'].unique()
dataCopy['State'] = dataCopy['State'].str.replace('Nndhra\xa0Pradesh', "Andhra Pradesh")
dataCopy['State'] = dataCopy['State'].str.replace('Andhra\xa0Pradesh', "Andhra Pradesh")
```

[32]:
```
# State Column and Zone column...
# Replacing the type
# 'Andhra\xa0Pradesh' - 'Nndhra\xa0Pradesh'
dataCopy['State'].unique()
dataCopy['State'] = dataCopy['State'].str.replace('Nndhra\xa0Pradesh', "Andhra Pradesh")
dataCopy['State'] = dataCopy['State'].str.replace('Andhra\xa0Pradesh', "Andhra Pradesh")
```

```
C:\Users\ashmi\AppData\Local\Temp\ipykernel_11244\3446481324.py:5: 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
  dataCopy['State'] = dataCopy['State'].str.replace('Nndhra\xa0Pradesh', "Andhra Pradesh")
C:\Users\ashmi\AppData\Local\Temp\ipykernel_11244\3446481324.py:6: 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
  dataCopy['State'] = dataCopy['State'].str.replace('Andhra\xa0Pradesh', "Andhra Pradesh")
```

[145]:
```
# Replacing the null values with mode..
martial_status_mode = dataCopy['Marital_Status'].mode()[0] # [0] is the index value we need...
martial_status_mode

dataCopy["Marital_Status"] = dataCopy['Marital_Status'].fillna(martial_status_mode)
dataCopy["Marital_Status"]
```

[145]:
```
0        0.0
1        1.0
2        1.0
3        0.0
4        1.0
        ...
11336    0.0
11337    0.0
11338    1.0
11339    0.0
11340    0.0
Name: Marital_Status, Length: 11254, dtype: float64
```

[147]:
```
# Creating a new Column with Martial_status...
dataCopy['marrage_status'] = dataCopy['Marital_Status'].apply(lambda x: 'Married' if (x == 1.) else 'Unmarried')
dataCopy.head(10)
```

```
C:\Users\ashmi\AppData\Local\Temp\ipykernel_10452\2259515924.py:2: 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
  dataCopy['marrage_status'] = dataCopy['Marital_Status'].apply(lambda x: 'Married' if (x == 1.) else 'Unmarried')
```

[147]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
0	1002903.0	Sanskriti	P00125942	F	26-35	28	0.0	Maharashtra	Western	Healthcare	Auto	1.0	23952.00	Unmarried
1	1000732.0	Kartik	P00110942	F	26-35	35	1.0	Andhra Pradesh	Southern	Govt	Auto	3.0	23934.00	Married
2	1001990.0	Bindu	P00118542	F	26-35	35	1.0	Uttar Pradesh	Central	Automobile	Auto	3.0	23924.00	Married
3	1001425.0	Sudevi	P00237842	M	0-17	16	0.0	Karnataka	Southern	Construction	Auto	2.0	23912.00	Unmarried
4	1000588.0	Joni	P00057942	M	26-35	28	1.0	Gujarat	Western	Food Processing	Auto	2.0	23877.00	Married
5	1000588.0	Joni	P00057942	M	26-35	28	1.0	Himachal Pradesh	Northern	Food Processing	Auto	1.0	23877.00	Married
6	1001132.0	Balk	P00018042	F	18-25	25	1.0	Uttar Pradesh	Central	Lawyer	Auto	4.0	23841.00	Married
8	1003224.0	Kushal	P00205642	M	26-35	NaN	0.0	Uttar Pradesh	Zentral	Govt	Auto	2.0	23809.00	Unmarried
9	1003650.0	Ginny	P00031142	F	26-35	NaN	1.0	Andhra Pradesh	Southern	Media	Auto	4.0	23799.99	Married
10	1003829.0	Harshita	P00200842	M	26-35	NaN	0.0	Delhi	Central	Banking	Auto	1.0	23770.00	Unmarried

[152]:
```
# Changing the dataType to numeric...
dataCopy['Age'] = pd.to_numeric(dataCopy['Age'], errors = 'coerce')
```

[170]:
```
dataCopy.info()
# null to mode value...
age_mode = dataCopy['Age'].mode()[0] # index is important...
age_mode

# Fill value...
dataCopy['Age'] = dataCopy['Age'].fillna(age_mode)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11254 entries, 0 to 11340
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11231 non-null  float64
 1   Cust_name         11254 non-null  object
 2   Product_ID        11254 non-null  object
 3   Gender            11252 non-null  object
 4   Age Group         11249 non-null  object
 5   Age               11254 non-null  float64
 6   Marital_Status    11254 non-null  float64
 7   State             11254 non-null  object
 8   Zone              11254 non-null  object
 9   Occupation        11249 non-null  object
 10  Product_Category  11254 non-null  object
 11  Orders            11254 non-null  float64
 12  Amount            11254 non-null  float64
 13  marrage_status    11254 non-null  object
dtypes: float64(5), object(9)
memory usage: 1.3+ MB
```

[170]:
```
30.0
```

[176]:
```
# Age group cleaning....
age_group_mode = dataCopy['Age Group'].mode()[0]
age_group_mode

dataCopy['Age Group'] = dataCopy['Age Group'].str.replace("low-high", age_group_mode)
dataCopy['Age Group'] = dataCopy['Age Group'].fillna(age_group_mode)
```

[176]:
```
'26-35'
```

[182]:
```
dataCopy['Age Group'].dtype
dataCopy['Age Group'].unique()
```

[182]:
```
array(['26-35', '0-17', '18-25', '51-55', '55+', '36-45', '46-50'],
      dtype=object)
```

[185]:
```
# Product Category already done...
# Gender and Occupation....
# Replacing null values with mode....

Gendermode = dataCopy['Gender'].mode()[0]
Gendermode
dataCopy['Gender'] = dataCopy['Gender'].fillna(Gendermode)
```

[187]:
```
Occupation_mode = dataCopy['Occupation'].mode()[0]
Occupation_mode
dataCopy['Occupation'] = dataCopy['Occupation'].fillna(Occupation_mode)
```

[189]:
```
# User ID
user_id_mode = dataCopy['User_ID'].mode()[0]
dataCopy['User_ID'] = dataCopy['User_ID'].fillna(user_id_mode)
```

[190]:
```
dataCopy['User_ID'].isnull().sum()
```

[190]:
```
0
```

[191]:
```
dataCopy.to_csv("D:\FIFTHFORCE\Filter_DiwaliSalesData.csv")
```

[1]:
```
# Importing the libraries...
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
```

[2]:
```
# importing cleaned data....
data_path = r"D:\FIFTHFORCE\Filter_DiwaliSalesData.csv"
filtered_Data = pd.read_csv(data_path)
```

[3]:
```
# Coping the data....
dataCopy = filtered_Data
dataCopy.head(10)
```

[3]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.00	Unmarried
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.00	Married
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.00	Married
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.00	Unmarried
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.00	Married
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh	Northern	Food Processing	Auto	1	23877.00	Married
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh	Central	Lawyer	Auto	4	23841.00	Married
7	1003224	Kushal	P00205642	M	26-35	30	0	Uttar Pradesh	Zentral	Govt	Auto	2	23809.00	Unmarried
8	1003650	Ginny	P00031142	F	26-35	30	1	Andhra Pradesh	Southern	Media	Auto	4	23799.99	Married
9	1003829	Harshita	P00200842	M	26-35	30	0	Delhi	Central	Banking	Auto	1	23770.00	Unmarried

Outlier Detection & Removal

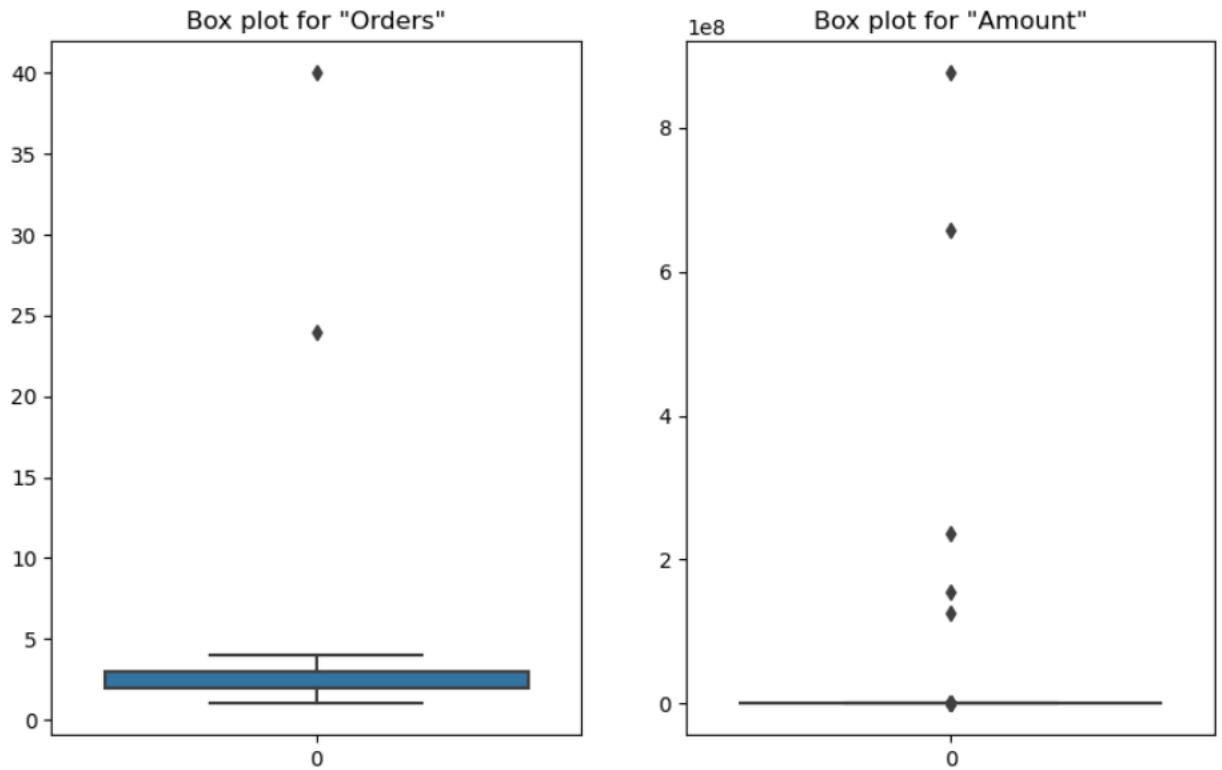
[4]:
```
dataCopy.describe() # Mainly used for only describing the numeric values....
```

[4]:

	User_ID	Age	Marital_Status	Orders	Amount
count	1.125400e+04	11254.000000	11254.000000	11254.000000	1.125400e+04
mean	1.013664e+06	35.424382	0.419762	2.494135	1.916697e+05
std	3.263686e+05	12.753259	0.493542	1.186925	1.072906e+07
min	1.000001e+06	12.000000	0.000000	1.000000	1.880000e+02
25%	1.001496e+06	27.000000	0.000000	2.000000	5.445000e+03
50%	1.003064e+06	33.000000	0.000000	2.000000	8.113000e+03
75%	1.004430e+06	43.000000	1.000000	3.000000	1.269900e+04
max	1.100505e+07	92.000000	1.000000	40.000000	8.760212e+08

[5]:
```
# Visulaization of outliers....
fig, ax = plt.subplots(1, 2, figsize = (10, 6))

sb.boxplot(data = dataCopy['Orders'], ax = ax[0])
ax[0].set_title('Box plot for "Orders"')

sb.boxplot(data = dataCopy['Amount'], ax = ax[1])
ax[1].set_title('Box plot for "Amount"')
```

[5]:
```
Text(0.5, 1.0, 'Box plot for "Amount"')
```



Removing the Outlier

[6]:
```
# Orders...
```

[7]:
```
Q1_order = np.quantile(dataCopy['Orders'], 0.25)
Q3_order = np.quantile(dataCopy['Orders'], 0.75)
IQR_order = Q3_order - Q1_order
IQR_order
lower_limit_order = Q1_order - 1.5*IQR_order
upper_limit_order = Q3_order + 1.5*IQR_order
```

[8]:
```
print(lower_limit_order)
print(upper_limit_order)
```

```
0.5
4.5
```

[9]:
```
# Removing...
dataCopy = dataCopy[(dataCopy['Orders'] > lower_limit_order) & (dataCopy['Orders'] < upper_limit_order)]
dataCopy.head(10)
```

[9]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.00	Unmarried
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.00	Married
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.00	Married
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.00	Unmarried
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.00	Married
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh	Northern	Food Processing	Auto	1	23877.00	Married
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh	Central	Lawyer	Auto	4	23841.00	Married

[10]:
```
Q1_order1 = np.quantile(dataCopy['Amount'], 0.25)
Q3_order1 = np.quantile(dataCopy['Amount'], 0.75)
IQR_order1 = Q3_order1 - Q1_order1
IQR_order1
lower_limit_order1 = Q1_order1 - 1.5*IQR_order1
upper_limit_order1 = Q3_order1 + 1.5*IQR_order1

print(lower_limit_order1)
print(upper_limit_order1)
```

```
-5436.625
23580.375
```

[11]:
```
dataCopy = dataCopy[(dataCopy['Amount'] > lower_limit_order1) & (dataCopy['Amount'] < upper_limit_order1)]
dataCopy.head(10)
```

[11]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
13	1001883	Praneet	P00029842	M	26-35	54	1	Uttar Pradesh	Central	Hospitality	Auto	1	23568.0	Married
14	1001883	Praneet	P00029842	M	26-35	54	1	Uttar Pradesh	Zentral	Hospitality	Auto	1	23568.0	Married
15	1000113	Ellis	P00180642	F	26-35	19	1	Andhra Pradesh	Southern	Govt	Auto	4	23546.0	Married
16	1000416	Hrisheekesh	P00181842	F	26-35	46	1	Uttar Pradesh	Central	Banking	Auto	2	23525.0	Married
17	1005256	Grant	P00101742	F	26-35	30	0	Andhra Pradesh	Southern	IT Sector	Auto	1	23518.0	Unmarried
18	1001505	Gilcrest	P00271842	F	51-55	53	0	Uttar Pradesh	Central	Automobile	Auto	2	23515.0	Unmarried
19	1000900	Skaria	P00317842	M	55+	83	0	Karnataka	Southern	Automobile	Auto	3	23513.0	Unmarried
20	1005908	Eric	P00282642	F	26-35	33	0	Andhra Pradesh	Southern	IT Sector	Auto	3	23462.0	Unmarried
21	1001101	Gibson	P00234742	F	36-45	40	0	Uttar Pradesh	Central	Banking	Auto	3	23456.0	Unmarried

[12]:
```
# Saved the file....
dataCopy.to_csv("D:\FIFTHFORCE\Filtered_DiwaliSalesData1-0.csv")
```

Exploratory Data Analysis

Statistical Data Analysis

[13]:
```
dataCopy.head(10)
```

[13]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
13	1001883	Praneet	P00029842	M	26-35	54	1	Uttar Pradesh	Central	Hospitality	Auto	1	23568.0	Married
14	1001883	Praneet	P00029842	M	26-35	54	1	Uttar Pradesh	Zentral	Hospitality	Auto	1	23568.0	Married
15	1000113	Ellis	P00180642	F	26-35	19	1	Andhra Pradesh	Southern	Govt	Auto	4	23546.0	Married
16	1000416	Hrisheekesh	P00181842	F	26-35	46	1	Uttar Pradesh	Central	Banking	Auto	2	23525.0	Married
17	1005256	Grant	P00101742	F	26-35	30	0	Andhra Pradesh	Southern	IT Sector	Auto	1	23518.0	Unmarried
18	1001505	Gilcrest	P00271842	F	51-55	53	0	Uttar Pradesh	Central	Automobile	Auto	2	23515.0	Unmarried
19	1000900	Skaria	P00317842	M	55+	83	0	Karnataka	Southern	Automobile	Auto	3	23513.0	Unmarried
20	1005908	Eric	P00282642	F	26-35	33	0	Andhra Pradesh	Southern	IT Sector	Auto	3	23462.0	Unmarried
21	1001101	Gibson	P00234742	F	36-45	40	0	Uttar Pradesh	Central	Banking	Auto	3	23456.0	Unmarried
22	1004736	Mahima	P00058042	F	18-25	25	1	Andhra Pradesh	Southern	Banking	Auto	4	23451.0	Married

[14]:
```
dataCopy.describe()
```

[14]:

	User_ID	Age	Marital_Status	Orders	Amount
count	1.123400e+04	11234.000000	11234.000000	11234.000000	11234.000000
mean	1.007453e+06	35.433772	0.419708	2.489496	9452.764638
std	2.109026e+05	12.759432	0.493533	1.114845	5213.784648
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000
25%	1.001496e+06	27.000000	0.000000	2.000000	5443.000000
50%	1.003064e+06	33.000000	0.000000	2.000000	8109.500000
75%	1.004429e+06	43.000000	1.000000	3.000000	12681.500000
max	1.100173e+07	92.000000	1.000000	4.000000	23568.000000

Data Vizualization

Analysis Parameter

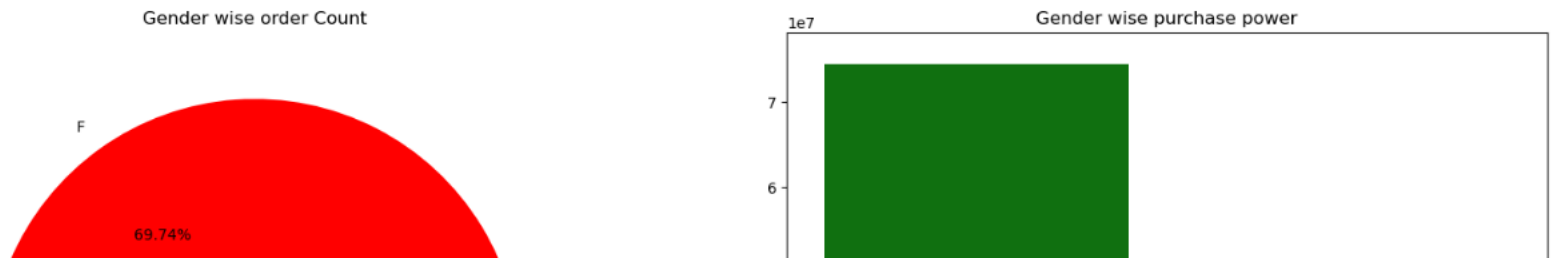
- Gender
- Marriage_Status
- Age group
- Product_Category
- State
- Product ID
- Occupation

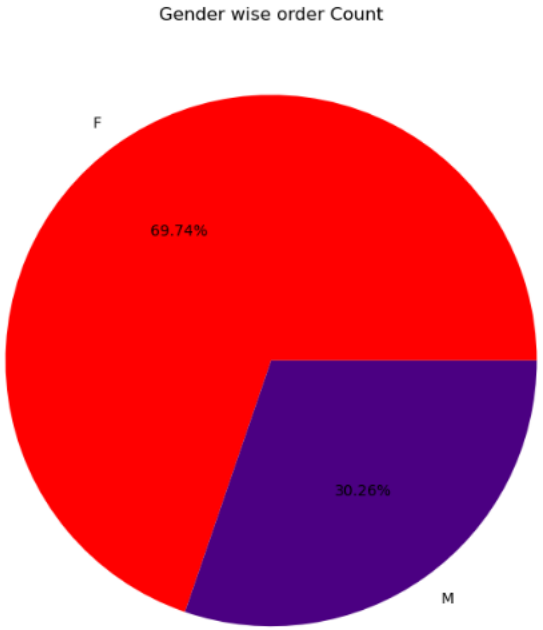
Gender Wise Analysis

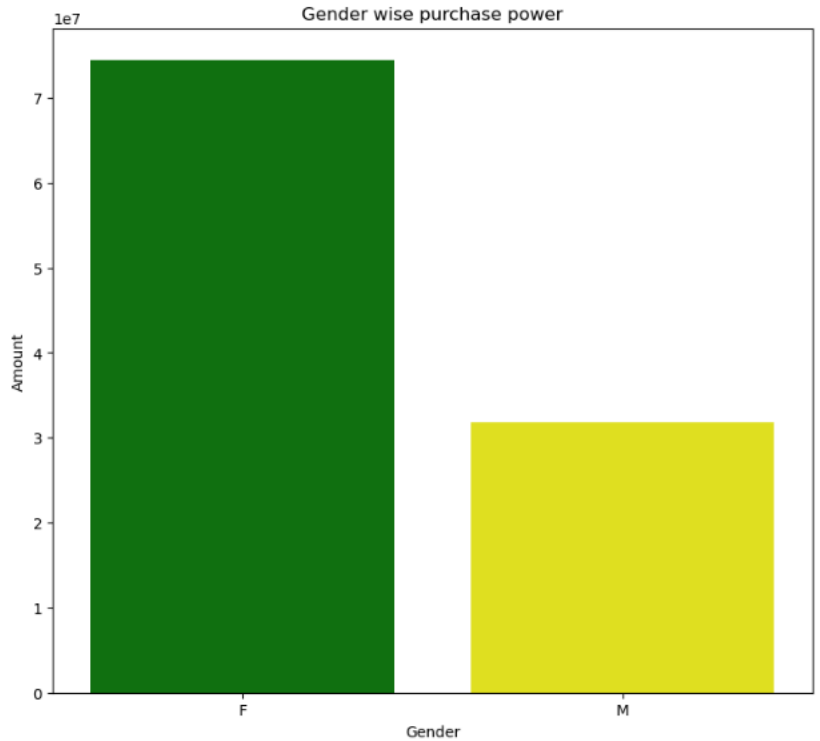
```
data_gender_count = dataCopy.groupby(['Gender'], as_index = False)['Orders'].count()
print("Gender wise order count: \n", data_gender_count)
print()
data_gender_amount = dataCopy.groupby(['Gender'], as_index = False)['Amount'].sum()
print("Gender wise order Amount: \n", data_gender_amount)
```

```
Gender wise order count:
   Gender  Orders
0      F    7835
1      M    3399

Gender wise order Amount:
   Gender       Amount
0      F  74422989.94
1      M  31769368.00
```

```
fig, ax = plt.subplots(1, 2, figsize = (20, 8))

ax[0].pie(x = data_gender_count['Orders'],
          labels = data_gender_count['Gender'], autopct = "%1.2f%%", colors = ['Red', 'Indigo']) # autopct - used for percentage analysis..
# plt.show() # X = dataValue, labels = lebels of the value...
ax[0].set_title("Gender wise order Count")

# Bar Chart...
sb.barplot(x = "Gender", y = "Amount", data = data_gender_amount, ax = ax[1], palette = ['green', 'yellow']) # For Seaborn we need to write index inside
# If we write 'tab:blue' it means that it is blue colour defined in Tableau palette
ax[1].set_title("Gender wise purchase power")
plt.show()
```







Martial Status wise analysis

[20]:
```
# Plot Data
data_marriage_status_amount = dataCopy.groupby(['marrage_status'], as_index = False)['Amount'].sum()
data_marriage_status_amount
```

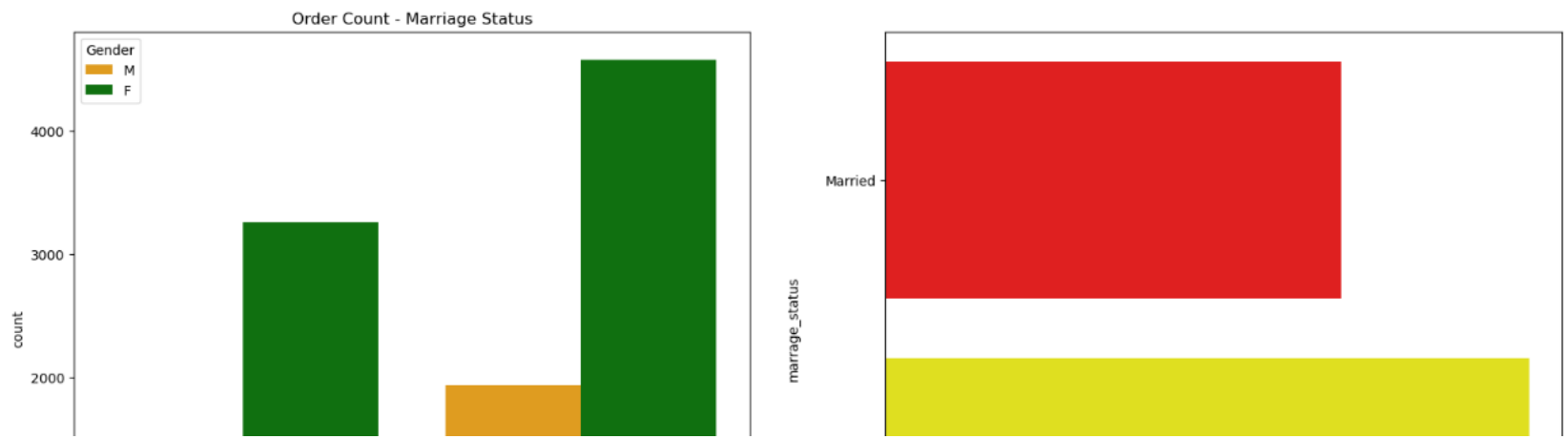
Martial Status wise analysis

[20]:
```
# Plot Data
data_marriage_status_amount = dataCopy.groupby(['marrage_status'], as_index = False)['Amount'].sum()
data_marriage_status_amount
```

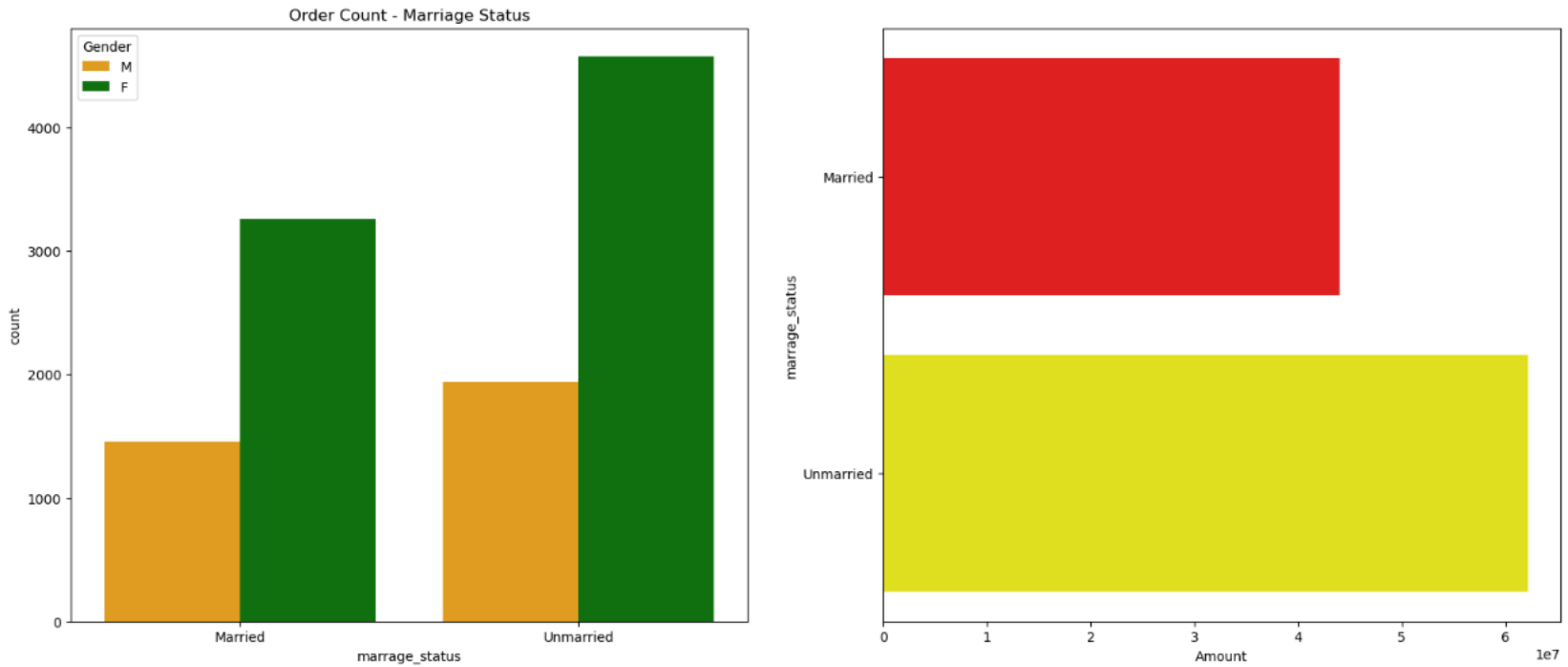
[20]:

	marrage_status	Amount
0	Married	44003938.00
1	Unmarried	62188419.94

[27]:
```
# Plots...
fig, ax = plt.subplots(1, 2, figsize = (20, 8))

# CountPlot...
sb.countplot(x = "marrage_status",hue = 'Gender', data = dataCopy, ax = ax[0], palette = ['Orange', 'Green'])
ax[0].set_title("Order Count - Marriage Status")

sb.barplot(x = 'Amount', y = 'marrage_status' , orient = 'h', data = data_marriage_status_amount, ax = ax[1], palette = ['red', 'Yellow'])
# Orient makes horizontal and vertical coulmns.....
plt.show()
```



```
# CountPlot...
sb.countplot(x = "marrage_status",hue = 'Gender', data = dataCopy, ax = ax[0], palette = ['Orange', 'Green'])
ax[0].set_title("Order Count - Marriage Status")

sb.barplot(x = 'Amount', y = 'marrage_status' , orient = 'h', data = data_marriage_status_amount, ax = ax[1], palette = ['red', 'Yellow'])
# Orient makes horizontal and vertical coulmns.....
plt.show()
```



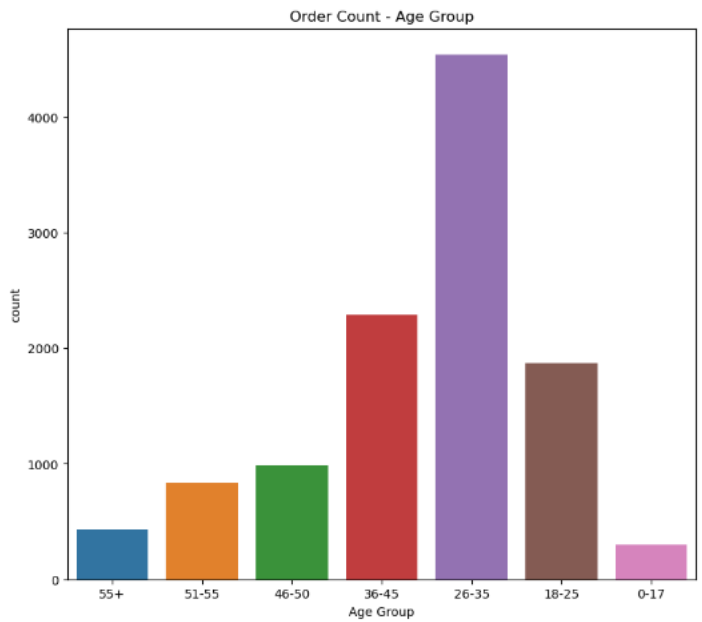
Conclusion

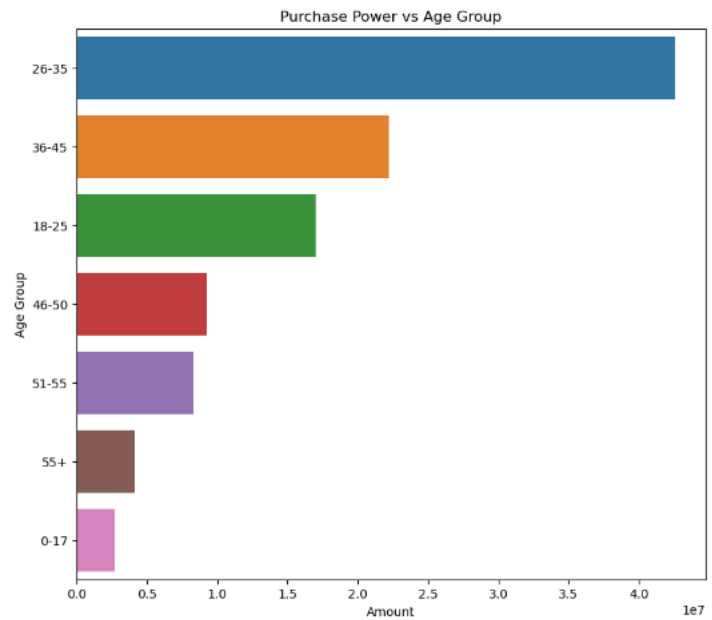
Unmarried female gives more order in products...

Age Group wise Analysis

[28]:
```
# Plot Data
data_age_group_amount = dataCopy.groupby(['Age Group'], as_index = False)['Amount'].sum()
data_age_group_amount.sort_values(by = 'Amount', ascending = False, inplace = True)
```

[30]:
```
# Plot Graph..
fig, ax = plt.subplots(1, 2, figsize = (20, 8))

# CountPlot...
use_dataCopy = dataCopy.sort_values(by = 'Age Group', ascending = False, inplace = True)
sb.countplot(x = "Age Group", data = dataCopy, ax = ax[0])
ax[0].set_title("Order Count - Age Group")

sb.barplot(x = 'Amount', y = 'Age Group' , orient = 'h', data = data_age_group_amount, ax = ax[1])
ax[1].set_title("Purchase Power vs Age Group")
# Orient makes horizontal and vertical coulmns.....
plt.show()
```





[32]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	marrage_status
1685	1003754	Katz	P00121642	F	55+	87	0	Maharashtra	Western	Chemical	Food	4	15836.0	Unmarried
10935	1001016	Applegate	P00024742	F	55+	89	0	Andhra Pradesh	Southern	Media	Beauty	1	1626.0	Unmarried
5316	1004084	Tillman	P00330242	F	55+	68	0	Gujarat	Western	IT Sector	Sports Products	2	8410.0	Unmarried
7352	1004083	Randy	P00338442	F	55+	87	0	Gujarat	Western	Retail	Clothing & Apparel	2	6960.0	Unmarried
9145	1002570	Herbert	P00034142	M	55+	58	1	Gujarat	Western	Retail	Beauty	1	4517.0	Married
10941	1002676	Jitesh	P00102442	F	55+	62	1	Madhya Pradesh	Central	Aviation	Beauty	2	1600.0	Married
1272	1005097	Krohn	P00315742	F	55+	76	0	Gujarat	Western	Retail	Furniture	1	16512.0	Unmarried
8053	1001252	McMahon	P00120942	F	55+	76	0	Kerala	Southern	Construction	Electronics & Gadgets	1	5975.0	Unmarried
5331	1005196	Derr	P00033642	M	55+	76	1	Madhya Pradesh	Central	Aviation	Footwear & Shoes	2	8378.0	Married
8049	1002656	Valerie	P00057442	F	55+	61	0	Kerala	Southern	Banking	Electronics & Gadgets	4	5978.0	Unmarried

Conclusion

- Age group of 26-35 have power purchase high or places more order

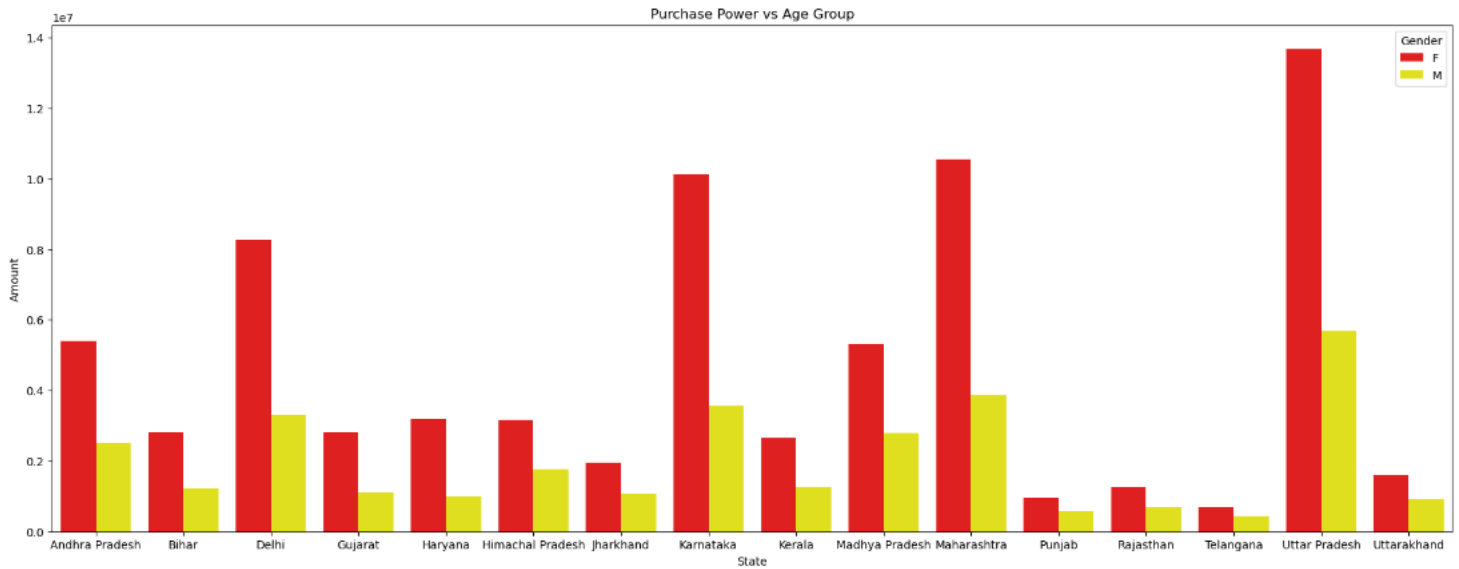
State wise sales

[33]:

```
data_state_amount = dataCopy.groupby(['State', 'Gender'], as_index = False)['Amount'].sum()
data_state_amount
```

22	Punjab	F	950267.00
23	Punjab	M	575533.00
24	Rajasthan	F	1243444.00
25	Rajasthan	M	686761.00
26	Telangana	F	703814.00

[34]:
```
# Plot state data....
fig, ax = plt.subplots(figsize = (22, 8))

sb.barplot(x = 'State', y = 'Amount', data = data_state_amount, hue = data_state_amount['Gender'], palette = ['red', 'yellow'], ax = ax)
ax.set_title("Purchase Power vs Age Group")

# Orient makes horizontal and vertical coulmns.....
plt.show()
```

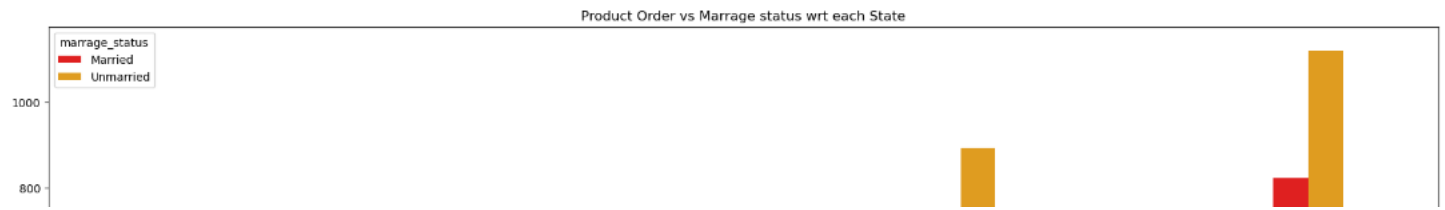


[35]:
```
data_state_count_marriage = dataCopy.groupby(['State', 'marriage_status'], as_index = False)['Product_ID'].count()
```

[35]:
```
data_state_count_marriage = dataCopy.groupby(['State', 'marrage_status'], as_index = False)['Product_ID'].count()
data_state_count_marriage
```

22	Punjab	Married	84
23	Punjab	Unmarried	116
24	Rajasthan	Married	98
25	Rajasthan	Unmarried	134
26	Telangana	Married	51
27	Telangana	Unmarried	74
28	Uttar Pradesh	Married	823
29	Uttar Pradesh	Unmarried	1120
30	Uttarakhand	Married	119
31	Uttarakhand	Unmarried	202

[36]:
```
# Plot state data....
fig, ax = plt.subplots(figsize = (22, 8))

sb.barplot(x = 'State', y = 'Product_ID', data = data_state_count_marriage, hue = data_state_count_marriage['marrage_status'], palette = ['red', 'orange
ax.set_title("Product Order vs Marrage status wrt each State")

# Orient makes horizontal and vertical coulmns.....
plt.show()
```



Order Count for TOP 5 States

[38]:
```
dataCopy.shape
```

[38]:
```
(11234, 14)
```

[39]:
```
# Orders Count....
order_count = dataCopy.groupby(['State'], as_index = False)['Orders'].sum()
order_count.sort_values(by = 'Orders', ascending = False, inplace = True)

# For top 5 values I used head()
order_count_top5 = order_count.head(5)
order_count_top5
```

[39]:

	State	Orders
14	Uttar Pradesh	4805
10	Maharashtra	3799
7	Karnataka	3266
2	Delhi	2728
9	Madhya Pradesh	2253

[40]:
```
purchase_power = dataCopy.groupby(['State'], as_index = False)['Amount'].sum()
purchase_power.sort_values(by = 'Amount', ascending = False, inplace = True)

# Top 5 State for Purchase power...
purchase_power_top5 = purchase_power.head(5)
purchase_power_top5
```

[40]:

	State	Amount
14	Uttar Pradesh	19360148.00
10	Maharashtra	14400762.00
7	Karnataka	13705829.00
2	Delhi	11554343.95
9	Madhya Pradesh	8100944.00

[42]:
```
fig, ax = plt.subplots(1, 2, figsize = (14, 4))
# For Beautification....
fig.tight_layout(w_pad = 12)

# order_count vs State...
sb.barplot(x = 'State', y = 'Orders', data = order_count_top5, ax = ax[0], palette = ['Red', 'Orange', 'Yellow', 'Green', 'Blue'])
ax[0].set_title('Order_count vs State')

# purchase_power vs State
sb.barplot(x = 'Amount', y = 'State', data = purchase_power_top5, orient = 'h', ax = ax[1], palette = ['Violet', 'Indigo', 'Blue', 'Green', 'Yellow'])
ax[1].set_title('Order_count vs State')

plt.show()
```