WALMART

IMPORT NECESSARY LIBRARIES

In [2]:

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

In [3]:

```
df = pd.read_csv(r"C:\Users\DEEPAK\Data science\Case study\walmart_data.csv")
```

About the dataset

In [4]:

```
df.sample(6)
```

Out[4]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Ye
89323	1001747	P00040042	M	26-35	0	C	
462587	1005241	P00216742	F	26-35	4	A	
459534	1004715	P00270942	M	26-35	2	B	
516106	1001483	P00147142	M	18-25	4	B	
275457	1000454	P00193242	M	26-35	20	C	
237031	1000560	P00193542	M	46-50	15	C	

We can see that there are columns with User ID and product ID available in the data. We need to find out the gender wise purchase one would choose on the basis of his/her age group, gender, city category, marital status etc.

Data type,shape,count

In [5]:

```
df.shape
```

Out[5]:

```
(550068, 10)
```

Data set has 550068 rows with 10 columns

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype 
---  ------                      --------------   ----- 
 0   User_ID                     550068 non-null  int64 
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64 
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64 
 8   Product_Category            550068 non-null  int64 
 9   Purchase                    550068 non-null  int64 
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

In th data set 5 columns have integer type and 5 columns have object type

Data Preprocessing

In [75]:

```
df.Product_ID=df["Product_ID"].astype("category")
df.Gender=df["Gender"].astype("category")
df.City_Category=df['City_Category'].astype("category")
df.Stay_In_Current_City_Years=df['Stay_In_Current_City_Years'].astype('category')
df.Age=df['Age'].astype('category')
```

Data preprocessing reduce the memory of the data with missing any values

Statistical Summery

In [8]:

```
df[df['Gender']=='F'].describe()
```

Out[8]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	1.358090e+05	135809.000000	135809.000000	135809.000000	135809.000000
mean	1.003130e+06	6.740540	0.419619	5.717714	8734.565765
std	1.786631e+03	6.239639	0.493498	3.696752	4767.233289
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001569e+06	1.000000	0.000000	3.000000	5433.000000
50%	1.003159e+06	4.000000	0.000000	5.000000	7914.000000
75%	1.004765e+06	11.000000	1.000000	8.000000	11400.000000
max	1.006039e+06	20.000000	1.000000	20.000000	23959.000000

OBSERVATIONS

Some of the important conclusions that can be drawn by looking at our results are:

1.Exactly Female customers purchased 135809 product in walmart.

2.Where as the min and max purchase are 12 and 23959 nos.

3.Product Category 75% are in below 8.

4.Female Customers occupation are in dataset have 75% below with value 11.

In [9]:

```
df[df['Gender']=='M'].describe()
```

Out[9]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	4.142590e+05	414259.00000	414259.000000	414259.000000	414259.00000
mean	1.002996e+06	8.51475	0.406386	5.301512	9437.52604
std	1.706494e+03	6.55379	0.491159	4.006275	5092.18621
min	1.000002e+06	0.00000	0.000000	1.000000	12.00000
25%	1.001505e+06	3.00000	0.000000	1.000000	5863.00000
50%	1.003041e+06	7.00000	0.000000	5.000000	8098.00000
75%	1.004411e+06	15.00000	1.000000	8.000000	12454.00000
max	1.006040e+06	20.00000	1.000000	20.000000	23961.00000

OBSERVATIONS

Some of the important conclusions that can be drawn by looking at our results are:

1.Exactly Male customers purchased 414259 product in walmart.

2.Where as the min and max purchase are 12 and 23961 nos.

3.Product Category 75% are in below 8.

4.Male Customers occupation are in dataset have 75% below with value 15.

Non-graphical Analysis

In [10]:

```
#Unique values in our Data Frame
df.nunique()
```

Out[10]:

```
User_ID                        5891
Product_ID                     3631
Gender                            2
Age                               7
Occupation                       21
City_Category                     3
Stay_In_Current_City_Years        5
Marital_Status                    2
Product_Category                 20
Purchase                      18105
dtype: int64
```

Observation:

- There are 3631 different products.
- There are both Marital and single customers
- Product category are 20 different types
- Age are classified in 7 different groups

In [11]:

```
#Are there any duplicate values?
df.duplicated().sum()
```

Out[11]:

```
0
```

There are no duplicate values in the data set.

Unique Values

In [12]:

```
list_col=['Gender','Age','Occupation','City_Category','Stay_In_Current_City_Years','Marital
for col in list_col:
    print('{} :{} ' . format(col.upper(),df[col].unique()))
```

```
GENDER :['F', 'M']
Categories (2, object): ['F', 'M']
AGE :['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25']
Categories (7, object): ['0-17', '18-25', '26-35', '36-45', '46-50', '51-5
5', '55+']
OCCUPATION :[10 16 15  7 20  9  1 12 17  0  3  4 11  8 19  2 18  5 14 13  6]
CITY_CATEGORY :['A', 'C', 'B']
Categories (3, object): ['A', 'B', 'C']
STAY_IN_CURRENT_CITY_YEARS :['2', '4+', '3', '1', '0']
Categories (5, object): ['0', '1', '2', '3', '4+']
MARITAL_STATUS :[0 1]
PRODUCT_CATEGORY :[ 3  1 12  8  5  4  2  6 14 11 13 15  7 16 18 10 17  9 20
19]
```

observation

-All the unique values in the Dataset

Value count

In [13]:

```
list_col=['Gender','Age','Occupation','City_Category','Stay_In_Current_City_Years','Marital
for col in list_col:
    print('{} :{} ' . format(col.upper(), df[col].value_counts()))
```

```
GENDER :M    414259
F    135809
Name: Gender, dtype: int64
AGE :26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
Name: Age, dtype: int64
OCCUPATION :4     72308
0     69638
7     59133
1     47426
17    40043
20    33562
12    31179
14    27309
2     26588
16    25371
6     20355
3     17650
10    12930
5     12177
15    12165
11    11586
19     8461
13     7728
18     6622
9      6291
8      1546
Name: Occupation, dtype: int64
CITY_CATEGORY :B    231173
C    171175
A    147720
Name: City_Category, dtype: int64
STAY_IN_CURRENT_CITY_YEARS :1     193821
2     101838
3      95285
4+     84726
0      74398
Name: Stay_In_Current_City_Years, dtype: int64
MARITAL_STATUS :0    324731
1    225337
Name: Marital_Status, dtype: int64
PRODUCT_CATEGORY :5     150933
1     140378
8     113925
11     24287
2      23864
6      20466
3      20213
4      11753
16      9828
```

```
15      6290
13      5549
10      5125
12      3947
7       3721
18      3125
20      2550
19      1603
14      1523
17       578
9        410
Name: Product_Category, dtype: int64
```

observation

-All the value count based on the Dataset

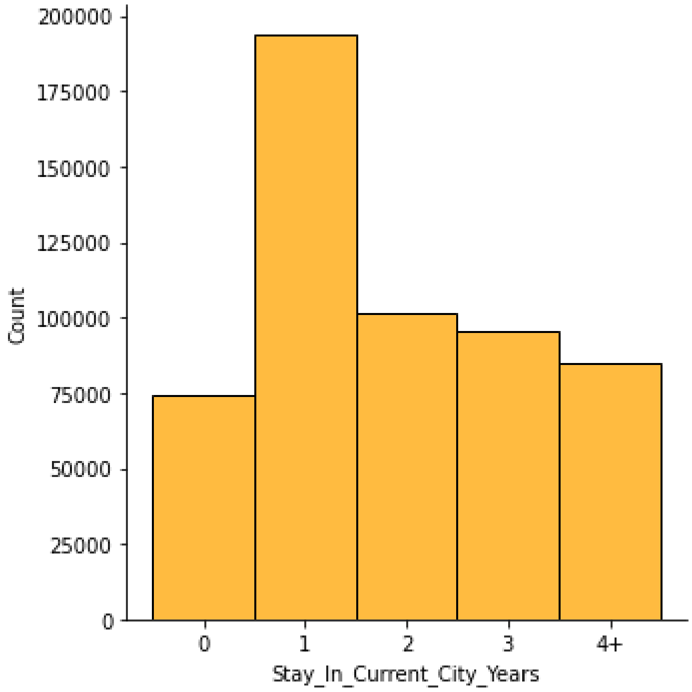
Visual Analysis

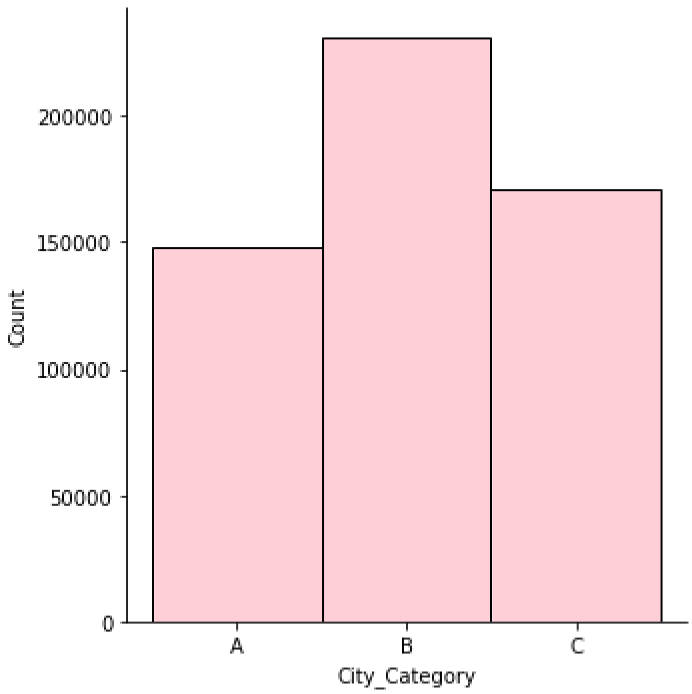
In [14]:

```
plt.figure(figsize=(15,7))
sns.displot(x=df.Stay_In_Current_City_Years,color='orange')

sns.displot(x=df.City_Category,color='pink')
plt.show()
```

<Figure size 1080x504 with 0 Axes>





Observation

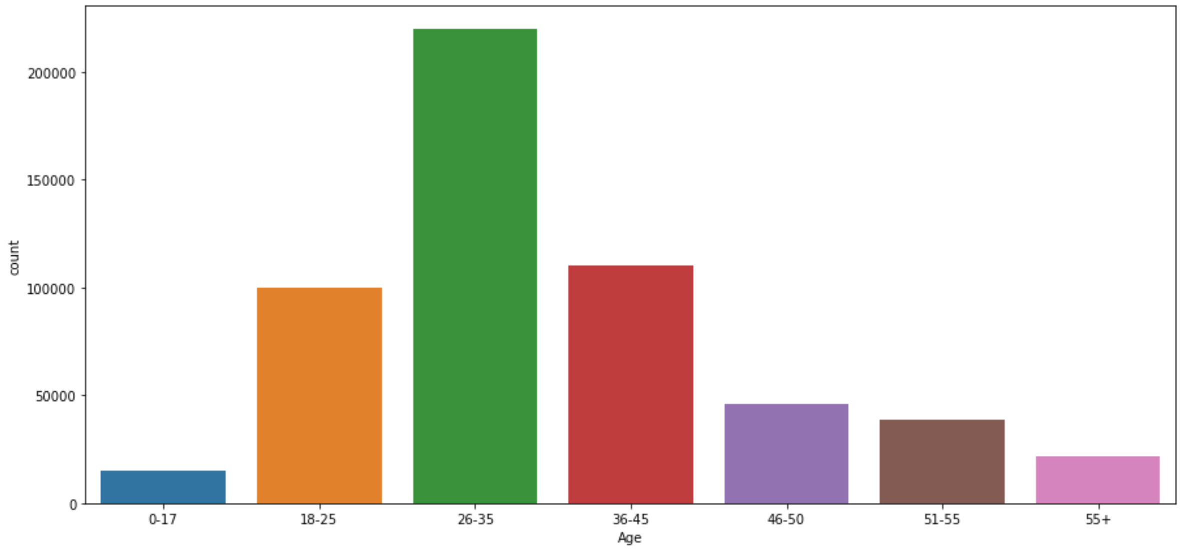
- The customers Stay in city years by one years are high compared to all
- The city category type of B is More

In [15]:

```
plt.figure(figsize=(15,7))
sns.countplot(x=df.Age)
```

Out[15]:

```
<AxesSubplot:xlabel='Age', ylabel='count'>
```



From the above Graph

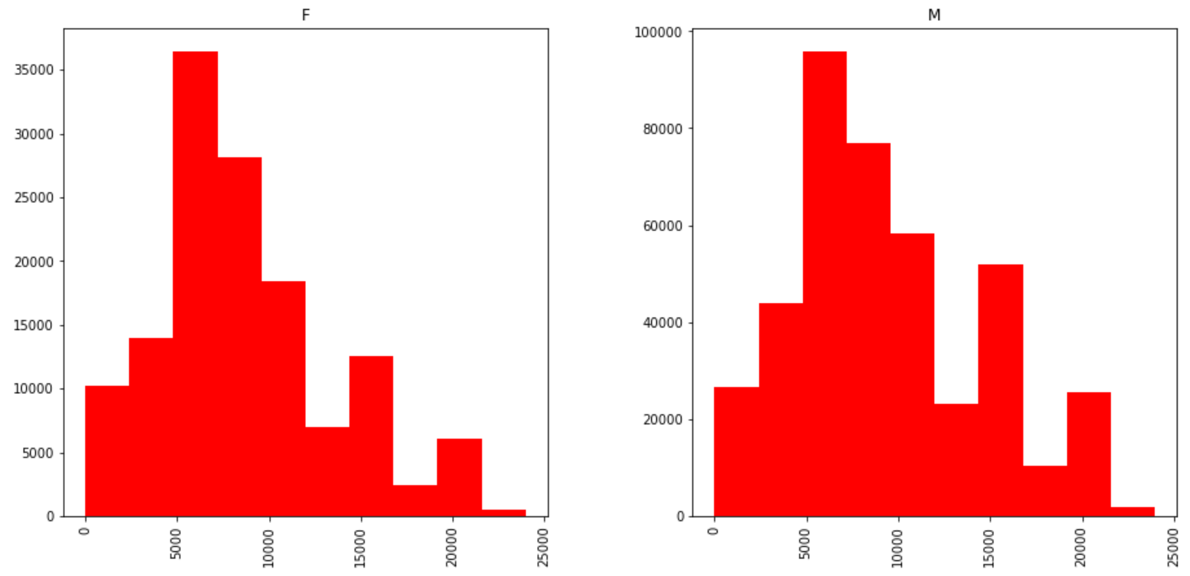
- The age between 26-35 are purchaseing more comapared to other age group of people

In [16]:

```
df.hist(by='Gender',column='Purchase',figsize=(15,7),color='red')
```

Out[16]:

```
array([<AxesSubplot:title={'center':'F'}>,
       <AxesSubplot:title={'center':'M'}>], dtype=object)
```



Observation

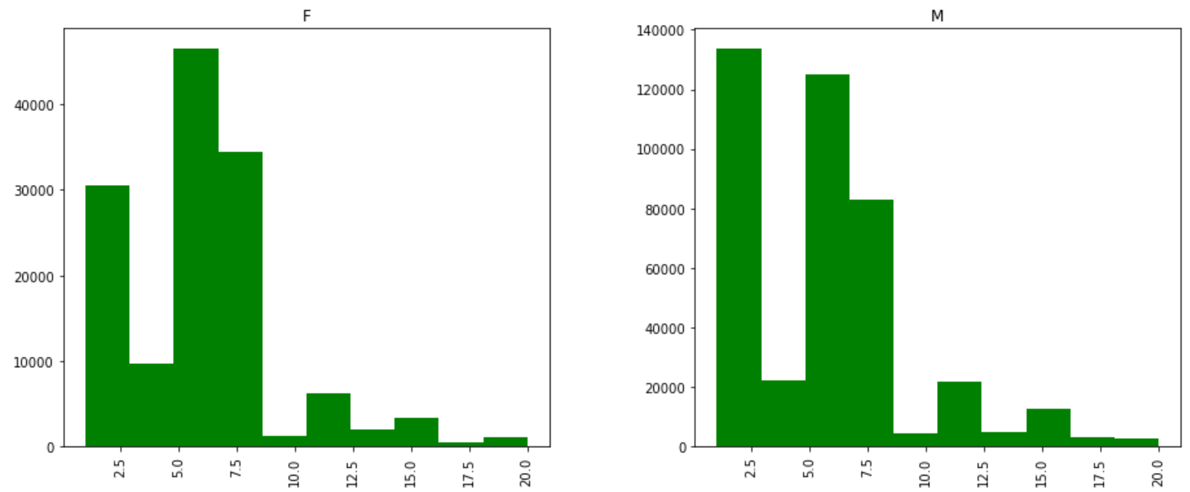
- The product Purchase rate of Male is high compared to female
- The Female and Male customers Purchase between 5000 to 7000 are high
- Male Customers spending more money

In [17]:

```
df.hist(by='Gender',column='Product_Category',figsize=(15,6),color='green')
```

Out[17]:

```
array([<AxesSubplot:title={'center':'F'}>,
       <AxesSubplot:title={'center':'M'}>], dtype=object)
```



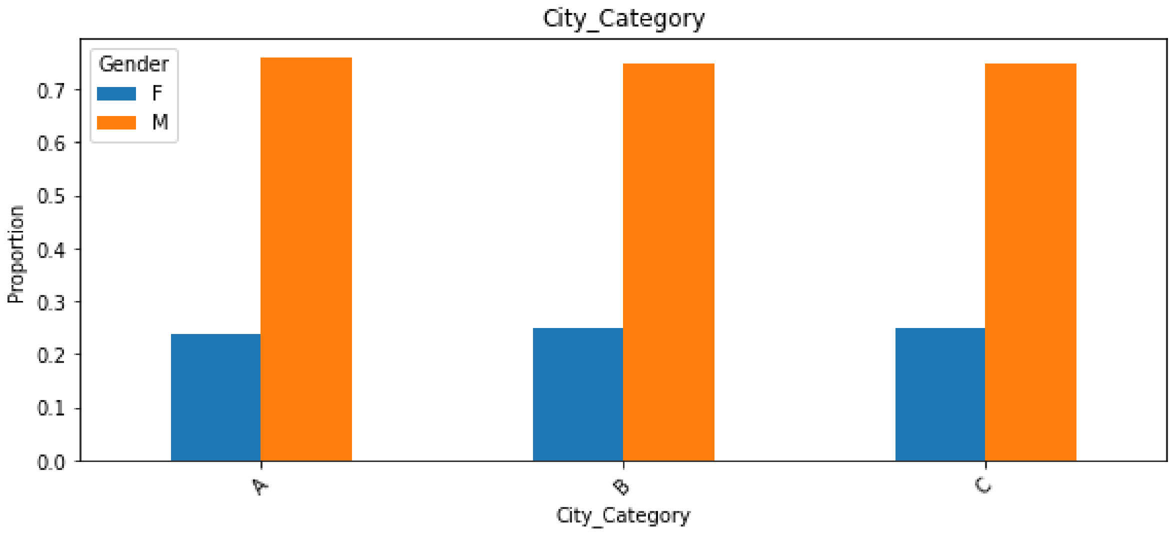
From the above graph

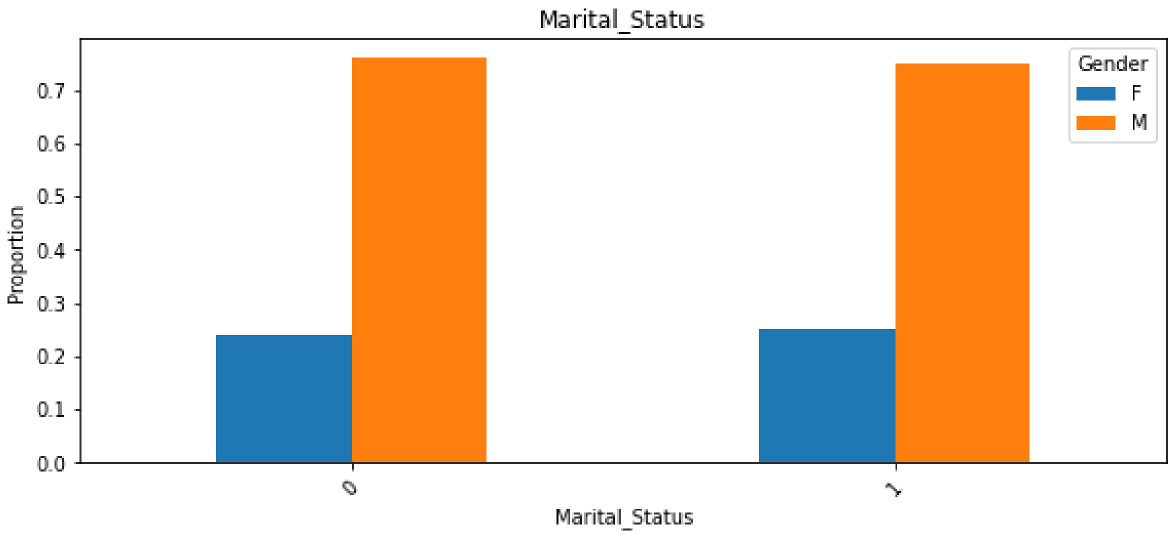
- Female customers buys more product in the Category of 5-6

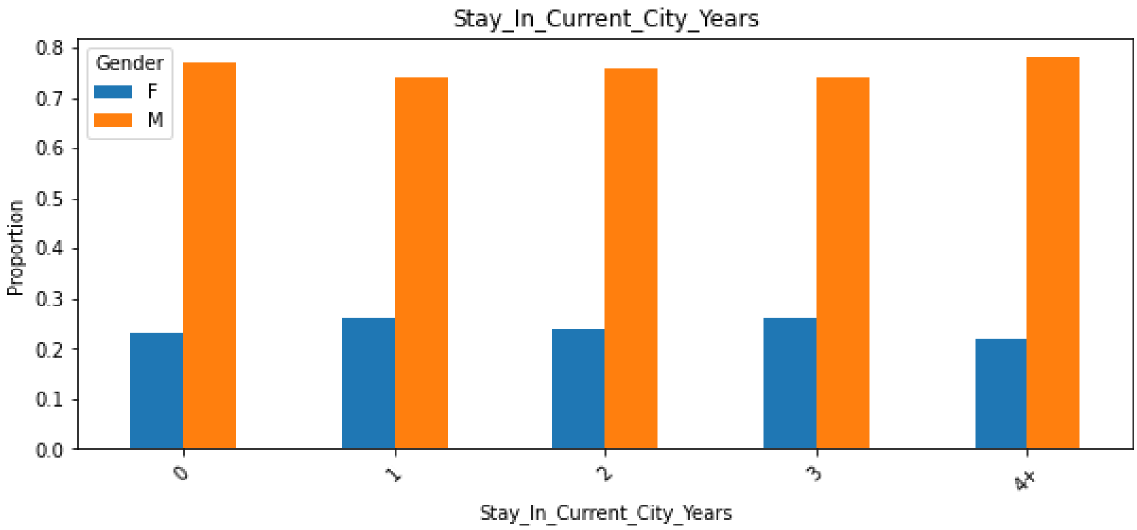
- Male customers buys more product in the category of 1-2.5

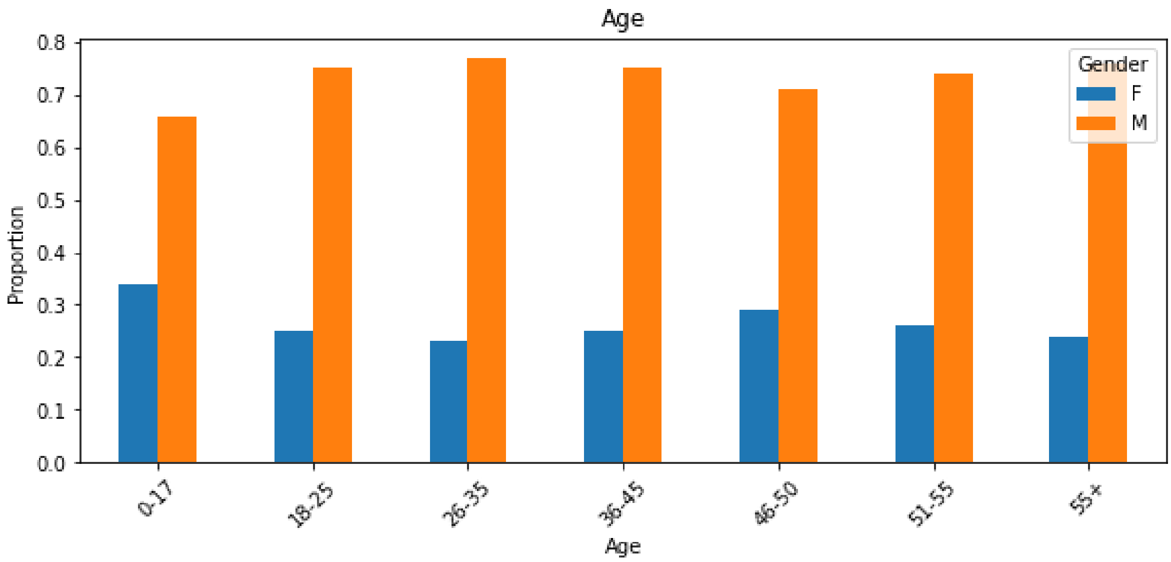
In [18]:

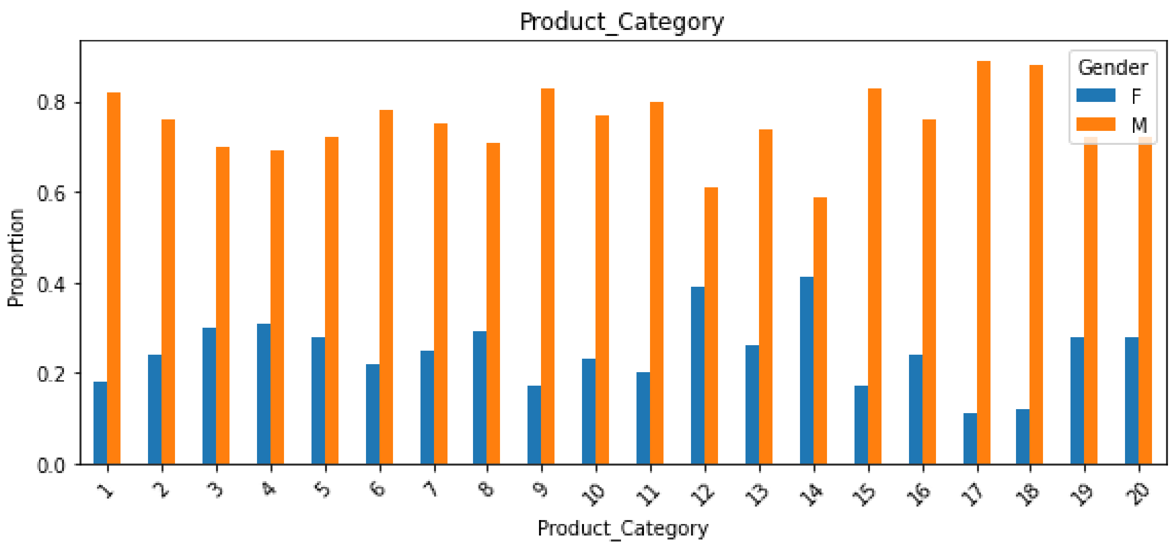
```
cat_cols=['City_Category', 'Marital_Status', 'Stay_In_Current_City_Years','Age','Product_Ca
for i in cat_cols:
    other= round(pd.crosstab(df[df[i].notnull()][i], df['Gender']).\
                 div(pd.crosstab(df[df[i].notnull()][i],df['Gender']).apply(sum,1),0),2)
    ax = other.plot(kind ='bar', title = i, figsize = (10,4))
    ax.set_xlabel(i)
    ax.set_ylabel('Proportion')
    plt.xticks(rotation=45)
    plt.show()
```











Observation

- The Male customers in the City category of A,B,C are more compared to Female
- The Maritual status of Female are same and Male
- The Age group of Male customers are equally spreaded and female customers are low compared to Male
- The Male customers choose high Product Category of

Heat Map

In [19]:

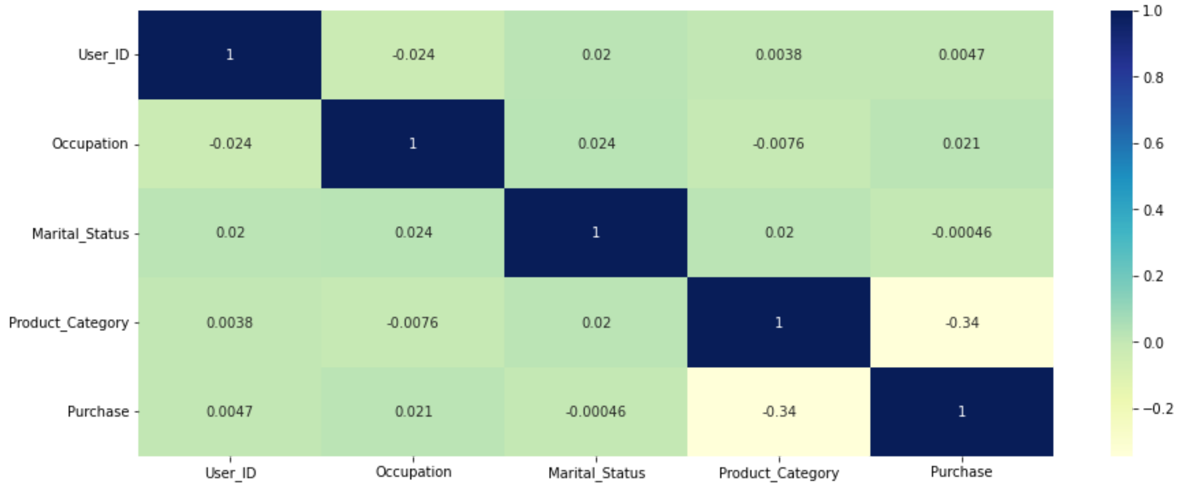
```
# A broader look at correlation between the columns of dataframe

# Creating a copy of the dataframe -
df_copy=df.copy()

df_copy['Gender'].replace(['Male', 'Female'], [1, 0], inplace=True)
```

In [20]:

```
plt.figure(figsize=(15,6))
sns.heatmap(df_copy.corr(), cmap="YlGnBu", annot=True)
plt.show()
```



Noteworthy Points

1) Product category are low correlated with purchase,which means if a customers's purchase level is high some product category

2) Marital status are low correlation with occupation and Product Category

3) From the heat map the correlation based on purchase didn't affect any other category

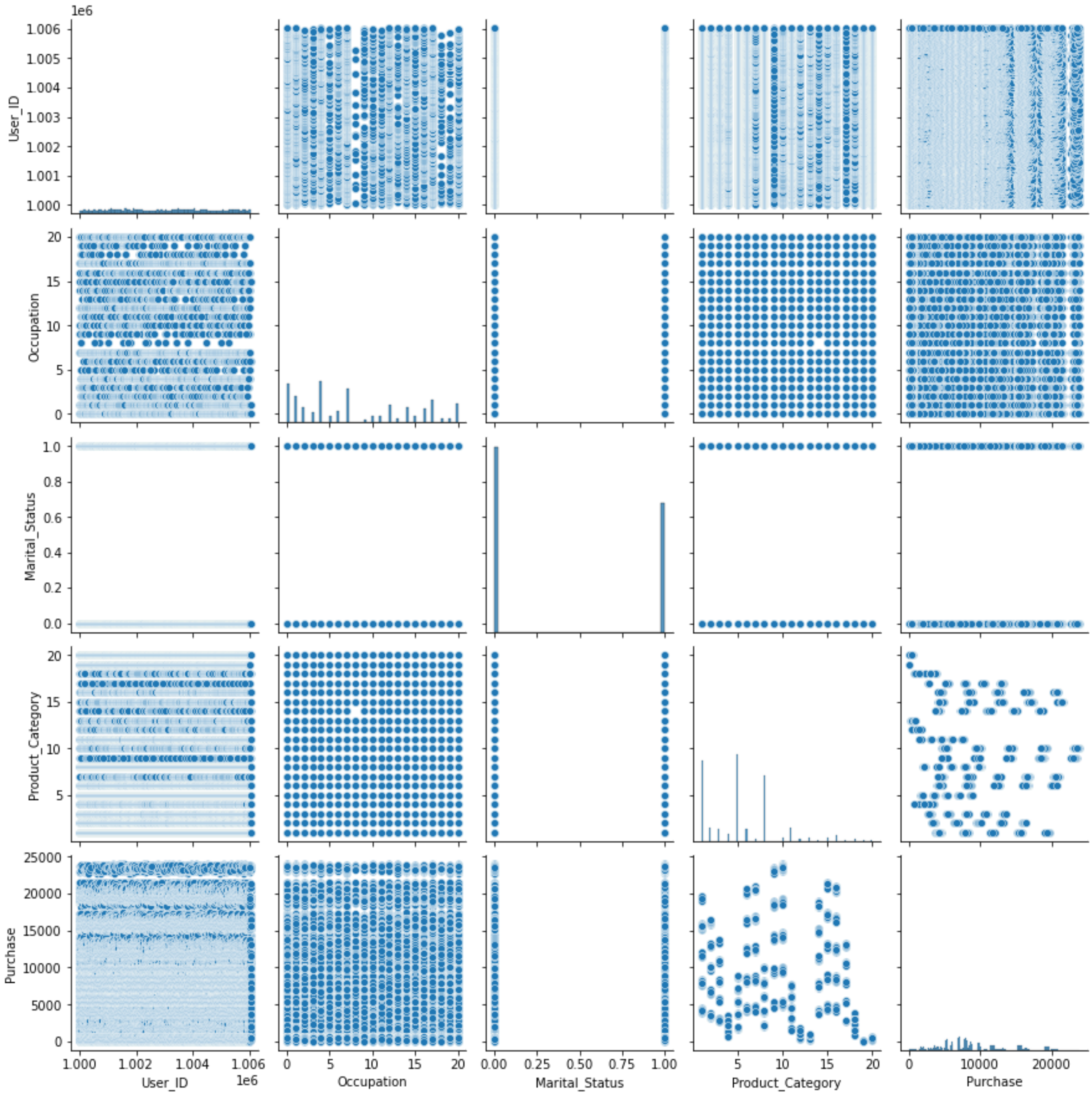
Pair plot

In [21]:

```
sns.pairplot(data=df,corner=False)
```

Out[21]:

<seaborn.axisgrid.PairGrid at 0x181eb920340>



Observation

- There is some corelation between Purchase and Occupation
- User Id correlation with Purchase

Cheacking Null values

In [22]:

```
df.isnull().sum()
```

Out[22]:

```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

We do not have any null values in our data set which makes it easier for us to conduct our data analysis.

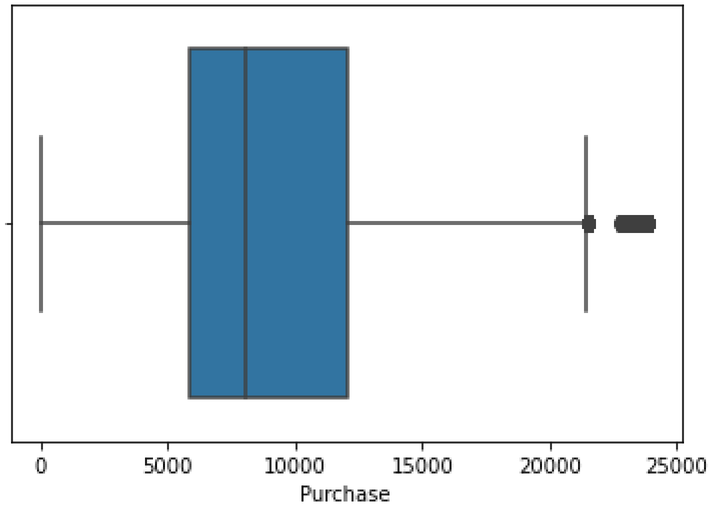
Box_plot - outlier Detection

In [23]:

```
sns.boxplot(data= df,x=df.Purchase)
```

Out[23]:

```
<AxesSubplot:xlabel='Purchase'>
```



In [24]:

```
Q3 = df['Purchase'].quantile(0.75)
Q1 = df['Purchase'].quantile(0.25)
IQR = Q3-Q1
upper = Q3+(1.5*IQR).astype('int')
lower = Q1-(1.5*IQR).astype('int')
print('Upper: '+str(upper))
print('Lower: '+str(lower))
```

```
Upper: 21400.0
Lower: -3523.0
```

Are women spending more money per transaction than men? Why or Why not?

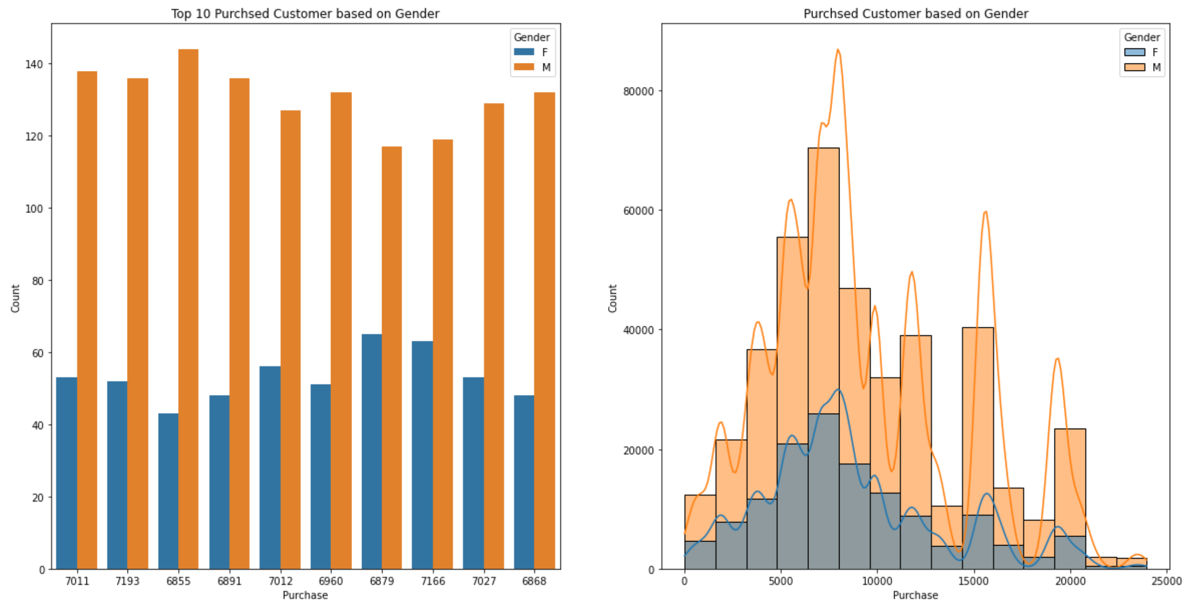
In [25]:

```
plt.figure(figsize=(20,10))
plt.subplot(1,2,1)
g = sns.countplot(x = df.Purchase, order=df.Purchase.value_counts().index[:10],hue=df.Gende
plt.title('Top 10 Purchsed Customer based on Gender')
plt.xlabel('Purchase')
plt.ylabel('Count')

plt.subplot(1,2,2)
sns.histplot(data=df,x=df.Purchase,hue=df.Gender,kde=True,bins=15)
plt.title('Purchsed Customer based on Gender')
```

Out[25]:

```
Text(0.5, 1.0, 'Purchsed Customer based on Gender')
```



observation

1) The Male customers are highly purchased when compared to Female

2) Female and Male Customers are overlaping in the purchase of 23,000.

3) Customers who buying in the top order also dominated by Male

Conclusion

Female customers didn't spend more money per transaction Based on in our Dataset

Married Vs Unmarried

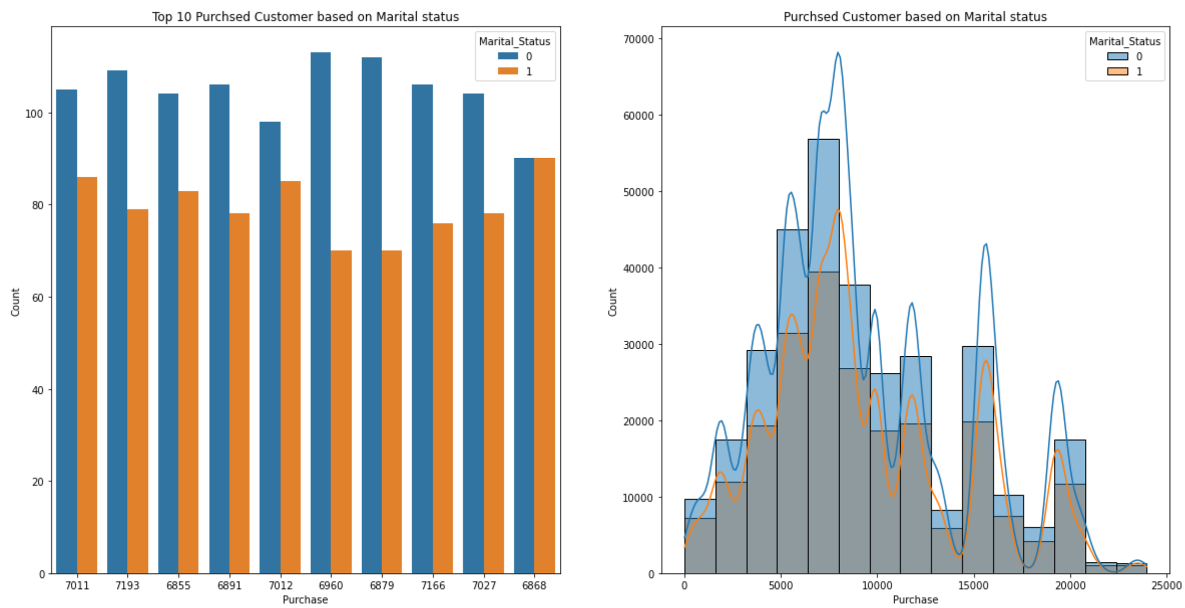
In [26]:

```
plt.figure(figsize=(20,10))
plt.subplot(1,2,1)
g = sns.countplot(x = df.Purchase, order=df.Purchase.value_counts().index[:10],hue=df.Marit
plt.title('Top 10 Purchsed Customer based on Marital status')
plt.xlabel('Purchase')
plt.ylabel('Count')

plt.subplot(1,2,2)
sns.histplot(data=df,x=df.Purchase,hue=df.Marital_Status,kde=True,bins=15)
plt.title('Purchsed Customer based on Marital status')
```

Out[26]:

```
Text(0.5, 1.0, 'Purchsed Customer based on Marital status')
```



observation

1) The UnMarried customers purchase is high when compared to Married

2) Unmarried and Married Customers are overlaping in the purchase of 23,000.

3) Customers who buying in the top order also dominated by Unmarried

Conclusion

Unmarried customers spend more money per transaction Based on in our Dataset

Customers Based on Age

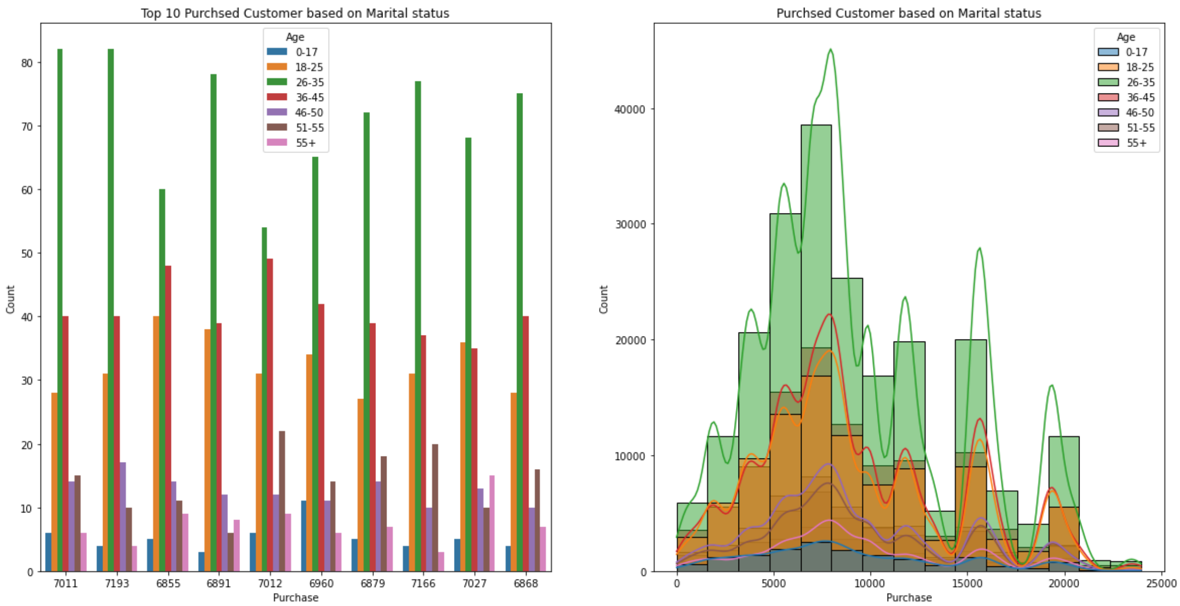
In [27]:

```
plt.figure(figsize=(20,10))
plt.subplot(1,2,1)
g = sns.countplot(x = df.Purchase, order=df.Purchase.value_counts().index[:10],hue=df.Age)
plt.title('Top 10 Purchsed Customer based on Marital status')
plt.xlabel('Purchase')
plt.ylabel('Count')

plt.subplot(1,2,2)
sns.histplot(data=df,x=df.Purchase,hue=df.Age,kde=True,bins=15)
plt.title('Purchsed Customer based on Marital status')
```

Out[27]:

```
Text(0.5, 1.0, 'Purchsed Customer based on Marital status')
```



observation

1) The Customers in Age of between 26-35 are purchase is high when compared to All age bins

2) The Age bins of all are overlaping in the purchase of 23,000.

3) Top customers are in Age of b/t 26-35

4) The age limlit of Male and Female Customers between 18 to 55 are purchased more

Conclusion

Customers in the age between 26 to 35 spend more money per transaction Based on in our Dataset

Confidence intervals and distribution of the mean of the expenses by female and male customers

In [202]:

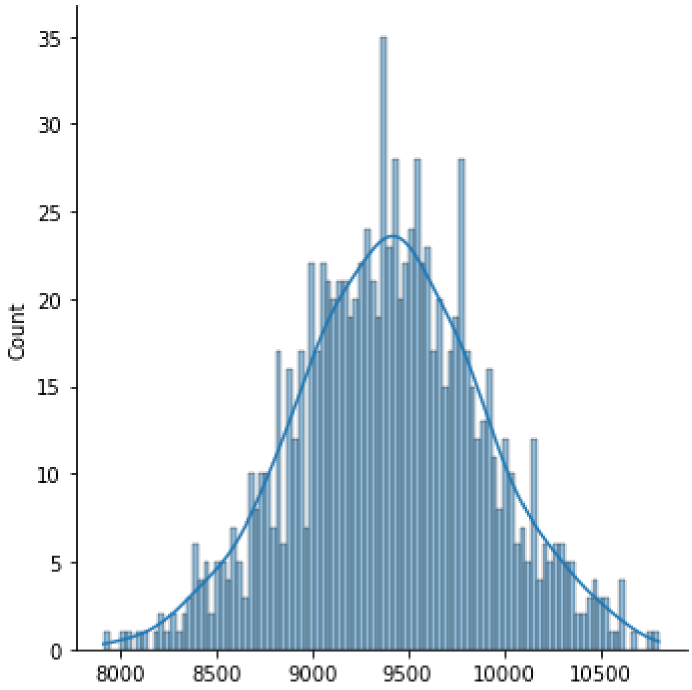
```
mal_cus=[df[df['Gender']=='M']['Purchase'].sample(100,replace=True).mean() for i in range(1
fema_cus=[df[df['Gender']=='F']['Purchase'].sample(100,replace=True).mean() for i in range(
```

In [29]:

```
sns.displot(mal_cus,bins=100,kde=True)
```

Out[29]:

<seaborn.axisgrid.FacetGrid at 0x181811e6490>

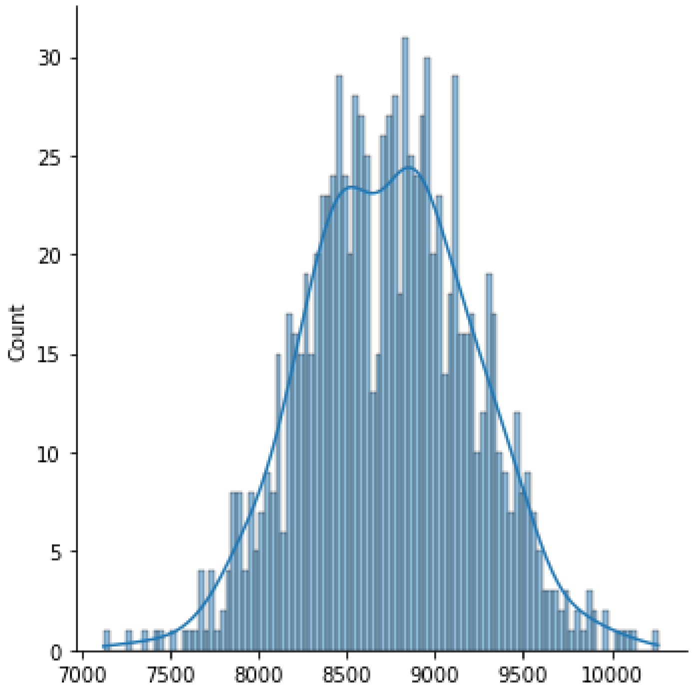


In [30]:

```
sns.displot(fema_cus,bins=100,kde=True)
```

Out[30]:

```
<seaborn.axisgrid.FacetGrid at 0x181815bcfd0>
```



Marital Status based on Gender

In [190]:

```
#UnMarried
age_Male_un = df[df['Gender']=='M'][['Marital_Status','Purchase']].groupby('Marital_Status'
#Married
age_Male_ma = df[df['Gender']=='M'][['Marital_Status','Purchase']].groupby('Marital_Status'
```

In [191]:

```
ma_un = [age_Male_un['Purchase'].sample(100,replace=True).mean() for i in range(1000)]
ma_Ma = [age_Male_ma['Purchase'].sample(100,replace=True).mean() for i in range(1000)]
```

In [196]:

```
#UnMarried
age_fema_un = df[df['Gender']=='F'][['Marital_Status','Purchase']].groupby('Marital_Status'
#Married
age_fema_ma = df[df['Gender']=='F'][['Marital_Status','Purchase']].groupby('Marital_Status'
```

In [197]:

```
fm_un = [age_fema_un['Purchase'].sample(100,replace=True).mean() for i in range(1000)]
fm_Ma = [age_fema_ma['Purchase'].sample(100,replace=True).mean() for i in range(1000)]
```

C.I on 99th percentile value for Purchase via bootsrapping

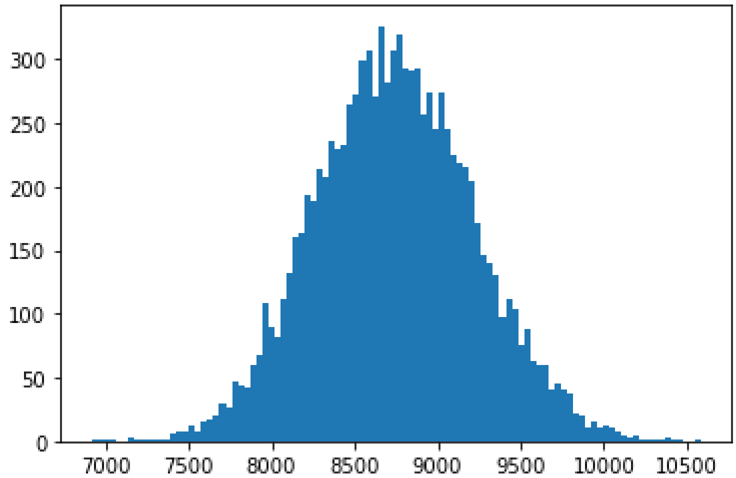
In [33]:

```
# Let's create r=10000 bootstrap samples, and let each bootstrap sample be of size=50
# bs_means is a list of 'r' bootstrap sample means
r = 10000
data = df[df['Gender']=='F']['Purchase']
size = 100
bs_means = np.empty(r)

for i in range(r):
    bs_sample = np.random.choice(data, size=size)
    bs_means[i] = np.mean(bs_sample)
```

In [34]:

```
plt.hist(bs_means, bins=100)
plt.show()
```



In [35]:

```
type(np.mean(bs_means))
```

Out[35]:

```
numpy.float64
```

In [36]:

```
#compute C.I on the mean given that bs_means follows Gaussian distribution: CLT
print('Mean = '+str(np.mean(bs_means).round(2)))
print('Standard Devaition = '+str(np.std(bs_means).round(2)))
```

```
Mean =8742.36
Standard Devaition=479.56
```

95% C.I on 99th percentile value for Purchase

Male confidence Intravel based on Gender

In [37]:

```
print('Lower Mean ='+str((np.mean(mal_cus)-1.96*np.std(mal_cus)).round(2)))
print('Upper Mean ='+str((np.mean(mal_cus)+1.96*np.std(mal_cus)).round(2)))
```

```
Lower Mean =8451.33
Upper Mean =10366.0
```

Confidence Intravel of Age

In [38]:

```
# could we just use the 2.5th percentile and 97.5th percentile value
print(np.percentile(mal_cus,2.5))
print(np.percentile(mal_cus,97.5))
```

```
8427.42
10373.724499999998
```

Male confidence Intravel of Marital status

In [193]:

```
#Married male
print('Lower Mean ='+str((np.mean(ma_Ma)-1.96*np.std(ma_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(ma_Ma)+1.96*np.std(ma_Ma)).round(2)))
```

```
Lower Mean =8424.18
Upper Mean =10435.73
```

In [195]:

```
#Unmarried male
print('Lower Mean ='+str((np.mean(ma_un)-1.96*np.std(ma_un)).round(2)))
print('Upper Mean ='+str((np.mean(ma_un)+1.96*np.std(ma_un)).round(2)))
```

```
Lower Mean =8443.31
Upper Mean =10427.44
```

Female Confidence Intravel based on Gender

In [198]:

```
#Female CI based on gender
print('Lower Mean ='+str((np.mean(fema_cus)-1.96*np.std(fema_cus)).round(2)))
print('Upper Mean ='+str((np.mean(fema_cus)+1.96*np.std(fema_cus)).round(2)))
```

```
Lower Mean =7805.02
Upper Mean =9671.81
```

Female confdence intravel of Age

In [204]:

```
# could we just use the 2.5th percentile and 97.5th percentile value
print(np.percentile(fema_cus,2.5))
print(np.percentile(fema_cus,97.5))
```

```
7827.403
9665.50725
```

Female confidence Intravel of Marital status

In [199]:

```
#Married female
print('Lower Mean ='+str((np.mean(fm_Ma)-1.96*np.std(fm_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(fm_Ma)+1.96*np.std(fm_Ma)).round(2)))
```

```
Lower Mean =7854.45
Upper Mean =9717.25
```

In [200]:

```
#Unmarried female
print('Lower Mean ='+str((np.mean(fm_un)-1.96*np.std(fm_un)).round(2)))
print('Upper Mean ='+str((np.mean(fm_un)+1.96*np.std(fm_un)).round(2)))
```

```
Lower Mean =7744.64
Upper Mean =9574.36
```

99% C.I on 99th percentile value for Purchase

Male Confidence Intravel

In [41]:

```
print('Lower Mean ='+str((np.mean(mal_cus)-2.57*np.std(mal_cus)).round(2)))
print('Upper Mean ='+str((np.mean(mal_cus)+2.57*np.std(mal_cus)).round(2)))
```

```
Lower Mean =8153.38
Upper Mean =10663.95
```

In [42]:

```
# could we just use the 0.05th percentile and 99.05th percentile value
print(np.percentile(mal_cus,0.005))
print(np.percentile(mal_cus,99.005))
```

```
7922.263861
10531.6800495
```

In [205]:

```
#Married male
print('Lower Mean ='+str((np.mean(ma_Ma)-2.57*np.std(ma_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(ma_Ma)+2.57*np.std(ma_Ma)).round(2)))
```

```
Lower Mean =8111.16
Upper Mean =10748.75
```

In [206]:

```
#Unmarried male
print('Lower Mean ='+str((np.mean(ma_un)-2.57*np.std(ma_un)).round(2)))
print('Upper Mean ='+str((np.mean(ma_un)+2.57*np.std(ma_un)).round(2)))
```

```
Lower Mean =8134.55
Upper Mean =10736.19
```

Female Confidence intravel

In [43]:

```
print('Lower Mean ='+str((np.mean(fema_cus)-2.57*np.std(fema_cus)).round(2)))
print('Upper Mean ='+str((np.mean(fema_cus)+2.57*np.std(fema_cus)).round(2)))
```

```
Lower Mean =7514.53
Upper Mean =9962.3
```

In [44]:

```
# could we just use the 0.005th percentile and 99.05th percentile value
print(np.percentile(fema_cus,0.005))
print(np.percentile(fema_cus,99.005))
```

```
7126.4061575
9864.3671515
```

In [208]:

```
#Married female
print('Lower Mean ='+str((np.mean(fm_Ma)-2.57*np.std(fm_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(fm_Ma)+2.57*np.std(fm_Ma)).round(2)))
```

```
Lower Mean =7564.58
Upper Mean =10007.12
```

In [207]:

```
#Unmarried female
print('Lower Mean ='+str((np.mean(fm_un)-2.57*np.std(fm_un)).round(2)))
print('Upper Mean ='+str((np.mean(fm_un)+2.57*np.std(fm_un)).round(2)))
```

```
Lower Mean =7459.91
Upper Mean =9859.08
```

90% C.I on 99th percentile value for Purchase via bootsrapping

Male Confidence Intravel

In [45]:

```
print('Lower Mean ='+str((np.mean(mal_cus)-1.64*np.std(mal_cus)).round(2)))
print('Upper Mean ='+str((np.mean(mal_cus)+1.64*np.std(mal_cus)).round(2)))
```

```
Lower Mean =8607.63
Upper Mean =10209.7
```

In [46]:

```
# could we just use the 5th percentile and 95th percentile value
print(np.percentile(mal_cus,5))
print(np.percentile(mal_cus,95))
```

```
8592.591
10254.0315
```

In [209]:

```
#Married male
print('Lower Mean ='+str((np.mean(ma_Ma)-1.64*np.std(ma_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(ma_Ma)+1.64*np.std(ma_Ma)).round(2)))
```

```
Lower Mean =8588.39
Upper Mean =10271.52
```

In [210]:

```
#Unmarried male
print('Lower Mean ='+str((np.mean(ma_un)-1.64*np.std(ma_un)).round(2)))
print('Upper Mean ='+str((np.mean(ma_un)+1.64*np.std(ma_un)).round(2)))
```

```
Lower Mean =8605.28
Upper Mean =10265.47
```

Female Confidence Intravel

In [47]:

```
print('Lower Mean ='+str((np.mean(fema_cus)-1.64*np.std(fema_cus)).round(2)))
print('Upper Mean ='+str((np.mean(fema_cus)+1.64*np.std(fema_cus)).round(2)))
```

```
Lower Mean =7957.41
Upper Mean =9519.41
```

In [48]:

```
# could we just use the 5th percentile and 95th percentile value
print(np.percentile(fema_cus,5))
print(np.percentile(fema_cus,95))
```

```
7965.2615000000005
9509.3735
```

In [212]:

```
#Married female
print('Lower Mean ='+str((np.mean(fm_Ma)-1.64*np.std(fm_Ma)).round(2)))
print('Upper Mean ='+str((np.mean(fm_Ma)+1.64*np.std(fm_Ma)).round(2)))
```

```
Lower Mean =8006.52
Upper Mean =9565.18
```

In [211]:

```
#Unmarried female
print('Lower Mean ='+str((np.mean(fm_un)-1.64*np.std(fm_un)).round(2)))
print('Upper Mean ='+str((np.mean(fm_un)+1.64*np.std(fm_un)).round(2)))
```

```
Lower Mean =7894.0
Upper Mean =9424.99
```

Observation / Interpretation of above Confidence Interval

By definition we know the interpretation of a 90%,95%,99% confidence interval for the population mean as - If repeated random samples were taken and the 90%,95%,99% confidence interval was computed for each sample, 95% of the intervals would contain the population mean.

So in this case

- There is a 90% chance that the confidence interval of (8446.17,10094.61) contains the true population Mean.
- There is a 95% chance that the confidence intrave of(8285.34,10255.43,) contains the true population Mean
- There is a 99% chance that the confidence interval of (7978.77,10562) contains the true population Mean.

Recommondation

- Male and Female custumers buying are Overlapping in Purchase
- City category sales of Male and Female customers are high in A
- The customers in the Age of 26-35 are purchasing more comapred to all category
- The product category of 5-7.5 are the most selled one in Product
- Occupation are 21 different category which will not affect the purchase
- Stay in city year of 1+ are puchasing more in our data set

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: