In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
data=pd.read_csv("walmart_data.csv")
```

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

Out[2]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Ma
0	1000001	P00069042	F	0-17	10	A	2	
1	1000001	P00248942	F	0-17	10	A	2	
2	1000001	P00087842	F	0-17	10	A	2	
3	1000001	P00085442	F	0-17	10	A	2	
4	1000002	P00285442	M	55+	16	C	4+	

In [3]:
```
data.shape
```

Out[3]:
```
(550068, 10)
```

In [4]:
```
data.dtypes
```

Out[4]:
```
User_ID                       int64
Product_ID                   object
Gender                       object
Age                          object
Occupation                    int64
City_Category                object
Stay_In_Current_City_Years   object
Marital_Status                int64
Product_Category              int64
Purchase                      int64
dtype: object
```

In [5]:
```
data.describe()
```

Out[5]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068.000000	550068.000000	550068.000000	550068.000000
mean	1.003029e+06	8.076707	0.409653	5.404270	9263.968713
std	1.727592e+03	6.522660	0.491770	3.936211	5023.065394
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001516e+06	2.000000	0.000000	1.000000	5823.000000
50%	1.003077e+06	7.000000	0.000000	5.000000	8047.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	12054.000000
max	1.006040e+06	20.000000	1.000000	20.000000	23961.000000

In [6]:
```
data.value_counts()
```

Out[6]:
```
User_ID  Product_ID  Gender  Age    Occupation  City_Category  Stay_In_Curren
t_City_Years  Marital_Status  Product_Category  Purchase
1000001  P00000142   F       0-17   10          A              2
0               3                 13650       1
1004007  P00105342   M       36-45  12          A              1
1               1                 11668       1
         P00115942   M       36-45  12          A              1
1               8                 9800        1
         P00115142   M       36-45  12          A              1
1               1                 11633       1
         P00114942   M       36-45  12          A              1
1               1                 19148       1

..
1001973  P00265242   M       26-35  1           A              0
0               5                 8659        1
         P00226342   M       26-35  1           A              0
0               11                6112        1
         P00198042   M       26-35  1           A              0
0               11                5915        1
         P00129842   M       26-35  1           A              0
0               6                 16101       1
1006040  P00349442   M       26-35  6           B              2
0               6                 16389       1
Length: 550068, dtype: int64
```

In [7]:
```
data.isnull().sum()
```

Out[7]:
```
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
```

In [8]:
```
data.nunique()
```

Out[8]:
```
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
```

In [9]:
```
data['Product_Category'].nunique()
```

Out[9]:
```
20
```

In [10]:
```
data['Occupation'].nunique()
```

Out[10]:
```
21
```

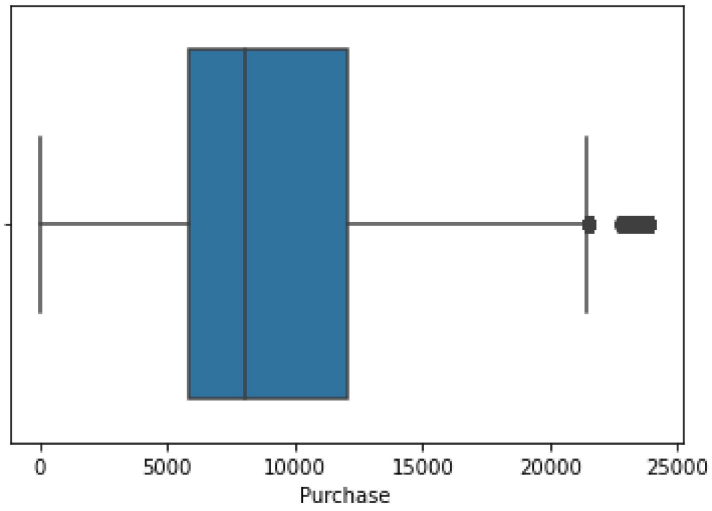
In [11]:
```
data['Stay_In_Current_City_Years'].nunique()
```

Out[11]:
```
5
```

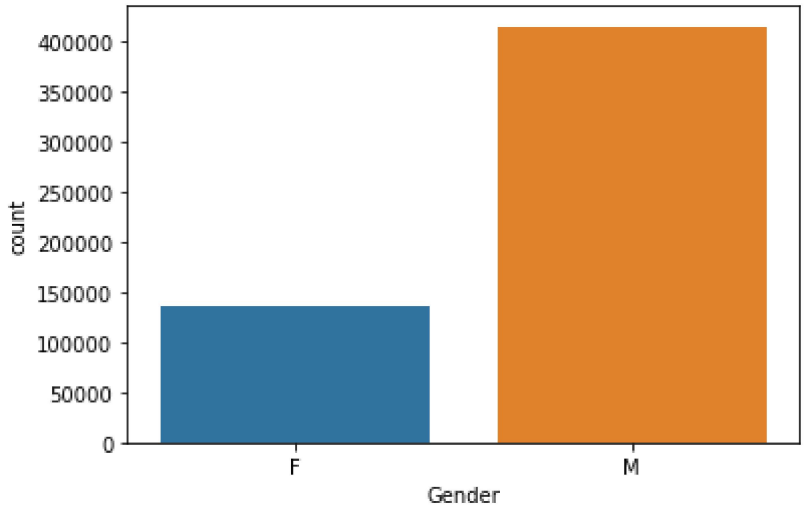
In [12]:
```
data['City_Category'].value_counts()
```

Out[12]:
```
B    231173
C    171175
A    147720
Name: City_Category, dtype: int64
```

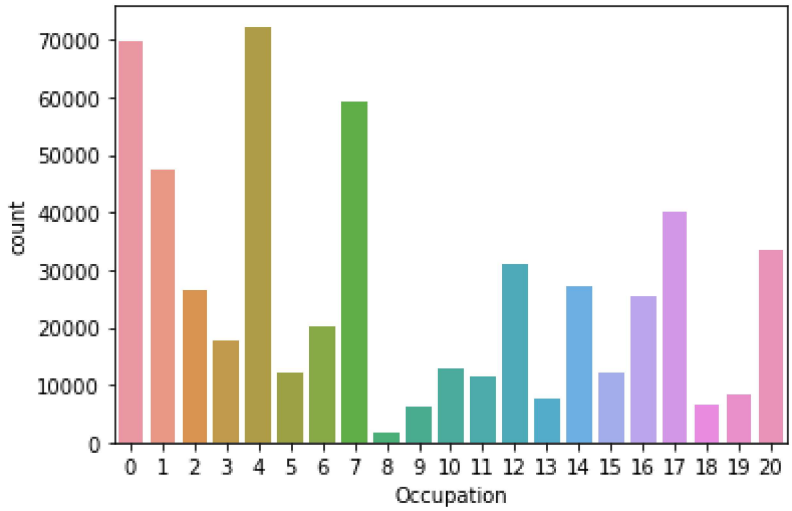
In [13]:
```
sns.boxplot(data=data, x='Purchase')
plt.show()
```



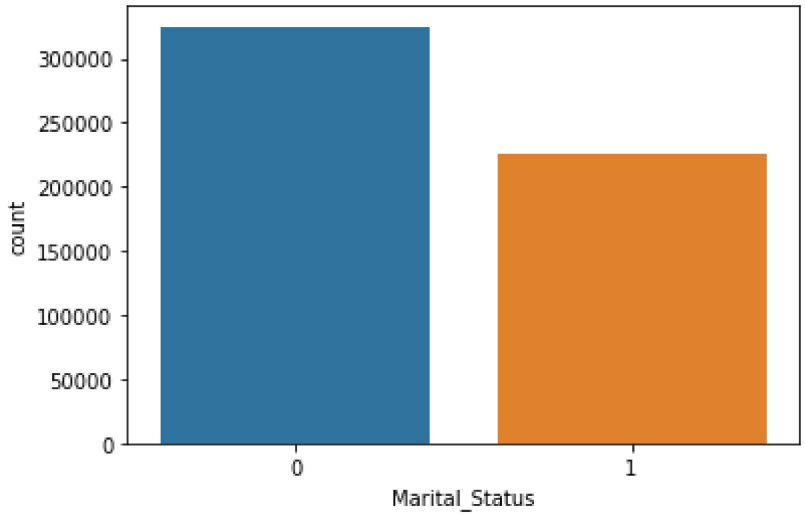
In [14]:
```
sns.countplot(data=data, x='Gender')
plt.show()
```



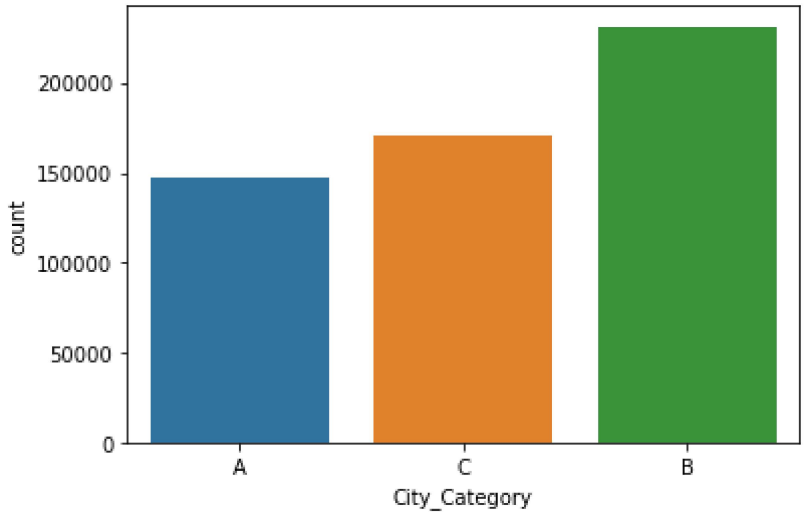
In [15]:
```
sns.countplot(data=data, x='Occupation')
plt.show()
```



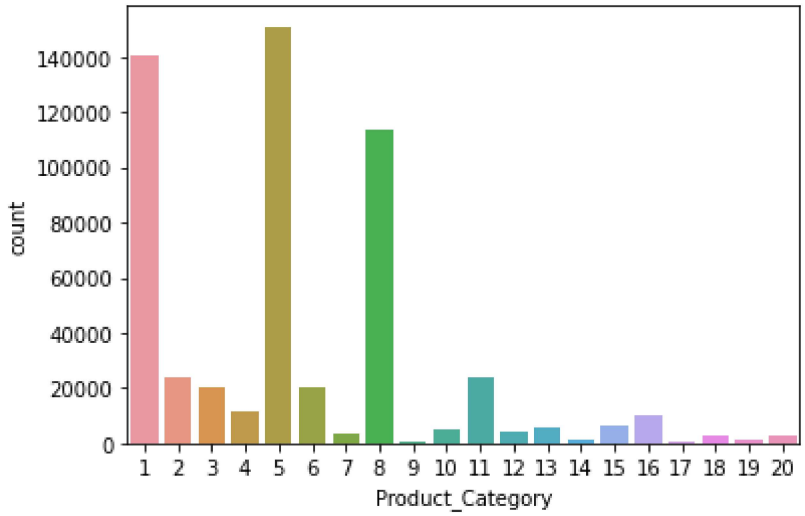
In [16]:
```
sns.countplot(data=data, x='Marital_Status')
plt.show()
```



In [17]:
```
sns.countplot(data=data, x='City_Category')
plt.show()
```



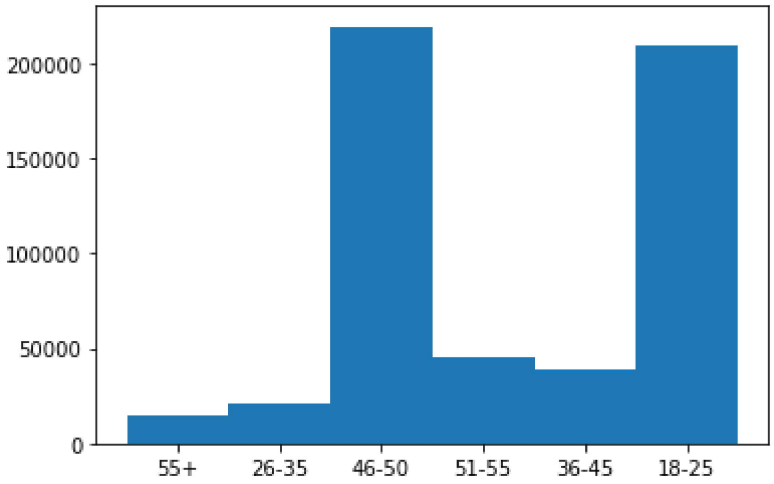
In [18]:
```
sns.countplot(data=data, x='Product_Category')
plt.show()
```



In [ ]:

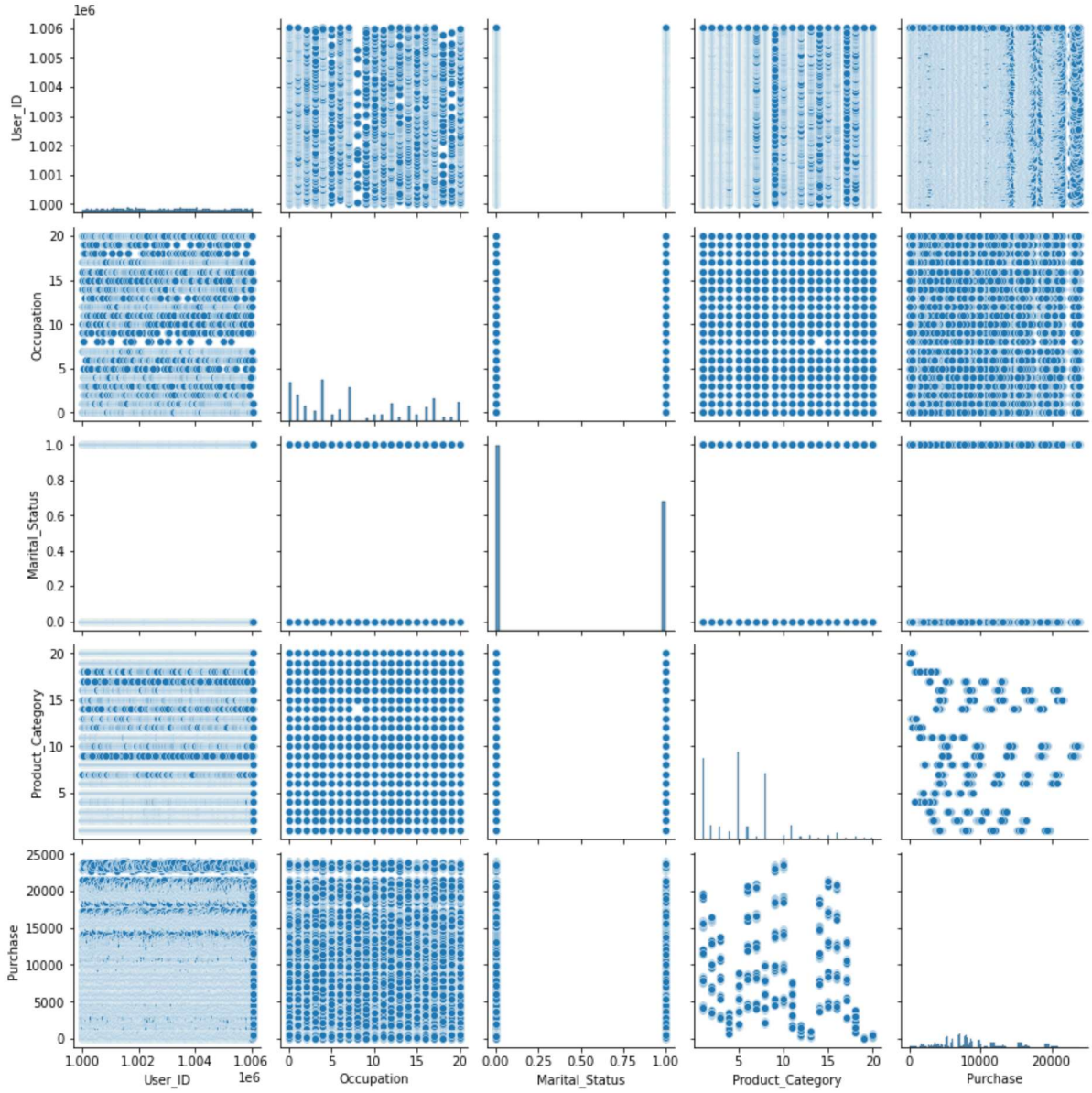
In [19]:
```
plt.hist(data["Age"],bins=6,align='right')
```

Out[19]:
```
(array([ 15102.,  21504., 219587.,  45701.,  38501., 209673.]),
 array([0., 1., 2., 3., 4., 5., 6.]),
 <BarContainer object of 6 artists>)
```



In [20]:
```
sns.pairplot(data)
```

Out[20]:
```
<seaborn.axisgrid.PairGrid at 0xc6e7910>
```



Observations

Most of the users are Male.

There are 21 different types of Occupation and 20 types Product_Category.

More users belong to B City_Category.

More users are Single as compare to Married.

Product_Category - 1, 5, 8, & 11 have highest purchasing frequency.

Purchase amount is having outliers.

In [21]:
```
amt = data.groupby(['User_ID', 'Gender'])[['Purchase']].sum()
amt = amt.reset_index()
amt['Gender'].value_counts()
```

Out[21]:
```
M    4225
F    1666
Name: Gender, dtype: int64
```

In [22]:
```
women= amt[amt["Gender"]=='F']['Purchase'].mean()
women
```

Out[22]:
```
712024.3949579832
```

In [23]:
```
men= amt[amt["Gender"]=='M']['Purchase'].mean()
men
```

Out[23]:
```
925344.4023668639
```

Obsevation

Average amount spent by male customers is 925344.40.

Average amount spent by female customers is 712024.39.

Male customers spending more money compared to female.

In [24]:
```
male_df = amt[amt['Gender']=='M']
female_df = amt[amt['Gender']=='F']
male_df
female_df
```

Out[24]:

	User_ID	Gender	Purchase
0	1000001	F	334093
5	1000006	F	379930
9	1000010	F	2169510
10	1000011	F	557023
15	1000016	F	150490
...	...	...	...
5885	1006035	F	956645
5886	1006036	F	4116058
5887	1006037	F	1119538
5888	1006038	F	90034
5889	1006039	F	590319

1666 rows × 3 columns

In [25]:
```
genders = ["M", "F"]

male_sample_size = 3000
female_sample_size = 1500
num_repitions = 1000
male_means = []
female_means = []

for i in range(num_repitions):
    male_mean = male_df.sample(male_sample_size, replace=True)['Purchase'].mear
    female_mean = female_df.sample(female_sample_size, replace=True)['Purchase'

    male_means.append(male_mean)
    female_means.append(female_mean)
```

In [ ]:
```
```

In [26]:
```
np.mean(male_means)
```

Out[26]: 925294.1850553334

In [27]:
```
np.mean(female_means)
```

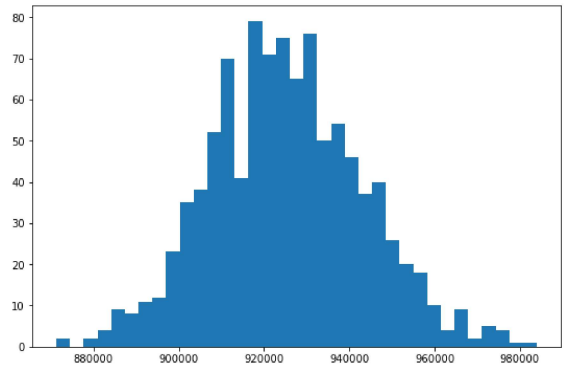
Out[27]: 712053.819548

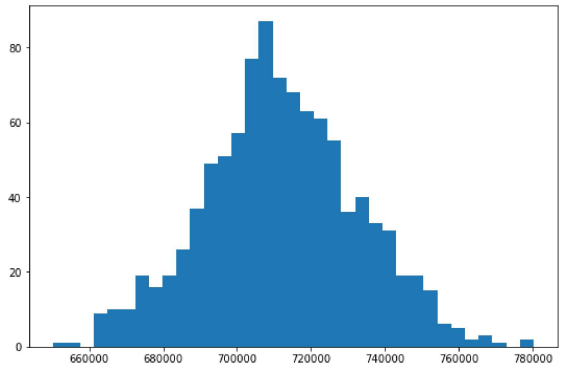
In [28]:
```
male_margin_of_error_clt = 1.96*male_df['Purchase'].std()/np.sqrt(len(male_df))
male_sample_mean = male_df['Purchase'].mean()
male_lower_lim = male_sample_mean - male_margin_of_error_clt
male_upper_lim = male_sample_mean + male_margin_of_error_clt

female_margin_of_error_clt = 1.96*female_df['Purchase'].std()/np.sqrt(len(femal
female_sample_mean = female_df['Purchase'].mean()
female_lower_lim = female_sample_mean - female_margin_of_error_clt
female_upper_lim = female_sample_mean + female_margin_of_error_clt
male_CI=(male_lower_lim, male_upper_lim)
female_CI=(female_lower_lim, female_upper_lim)
```

In [29]:
```
fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

axis[0].hist(male_means, bins=35)
axis[1].hist(female_means, bins=35)
plt.show()
```





In [30]:
```
male_CI
```

Out[30]:
```
(895617.8331736492, 955070.9715600787)
```

In [31]:
```
female_CI
```

Out[31]:
```
(673254.7725364959, 750794.0173794704)
```

In [32]:
```
mdata = data.groupby(['User_ID', 'Marital_Status'])[['Purchase']].sum()
mdata = mdata.reset_index()
mdata['Marital_Status'].value_counts()
```

Out[32]:
```
0    3417
1    2474
Name: Marital_Status, dtype: int64
```

In [33]:
```
marid_df = mdata[mdata['Marital_Status']==0]
unmarid_df = mdata[mdata['Marital_Status']==0]
marid_df
unmarid_df
```

Out[33]:

	User_ID	Marital_Status	Purchase
0	1000001	0	334093
1	1000002	0	810472
2	1000003	0	341635
5	1000006	0	379930
8	1000009	0	594099
...	...	...	...
5884	1006034	0	197086
5885	1006035	0	956645
5887	1006037	0	1119538
5888	1006038	0	90034
5890	1006040	0	1653299

3417 rows × 3 columns

In [34]:
```
marid_samp_size = 3000
unmarid_sample_size = 2000
num_repitions = 1000
marid_means = []
unmarid_means = []

for i in range(num_repitions):
    marid_mean = mdata[mdata['Marital_Status']==1].sample(marid_samp_size, rep
    unmarid_mean = mdata[mdata['Marital_Status']==0].sample(unmarid_sample_size

    marid_means.append(marid_mean)
    unmarid_means.append(unmarid_mean)
```

In [35]:
```
np.mean(marid_means)
```

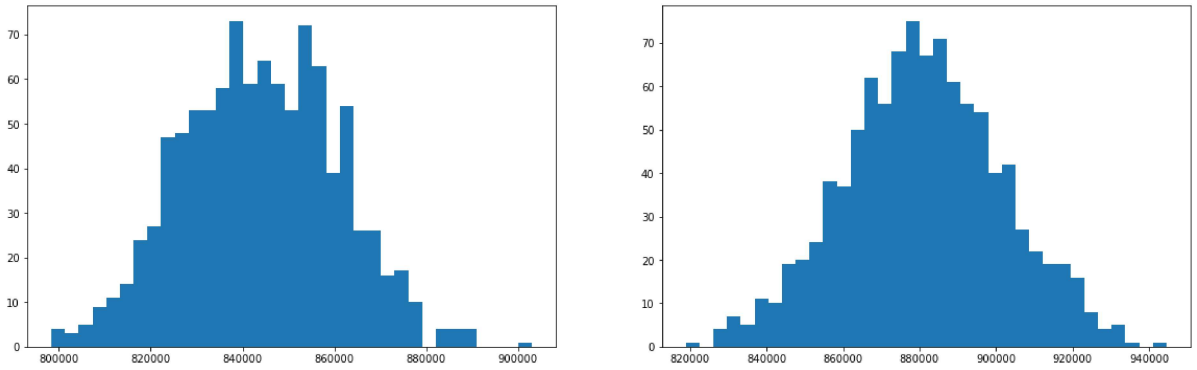
Out[35]: 843796.009888

In [36]:
```
np.mean(unmarid_means)
```

Out[36]: 880916.7558345

In [37]:
```
fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

axis[0].hist(marid_means, bins=35)
axis[1].hist(unmarid_means, bins=35)
plt.show()
```



In [38]:
```
marid_margin_of_error_clt = 1.96*marid_df['Purchase'].std()/np.sqrt(len(marid_d
marid_sample_mean = marid_df['Purchase'].mean()
marid_lower_lim = marid_sample_mean - marid_margin_of_error_clt
marid_upper_lim = marid_sample_mean + marid_margin_of_error_clt

unmarid_margin_of_error_clt = 1.96*unmarid_df['Purchase'].std()/np.sqrt(len(unm
unmarid_sample_mean = unmarid_df['Purchase'].mean()
unmarid_lower_lim = unmarid_sample_mean - unmarid_margin_of_error_clt
unmarid_upper_lim = unmarid_sample_mean + unmarid_margin_of_error_clt
marid_CI=(marid_lower_lim, marid_upper_lim)
unmarid_CI=(unmarid_lower_lim, unmarid_upper_lim)
```

In [39]:
```
marid_CI
```

Out[39]:
```
(848741.1824337274, 912410.3815112535)
```

In [40]:
```
unmarid_CI
```

Out[40]:
```
(848741.1824337274, 912410.3815112535)
```

In [41]:
```
age_df = data.groupby(['User_ID', 'Age'])[['Purchase']].sum()
age_df = age_df.reset_index()
age_df
```

Out[41]:

	User_ID	Age	Purchase
0	1000001	0-17	334093
1	1000002	55+	810472
2	1000003	26-35	341635
3	1000004	46-50	206468
4	1000005	26-35	821001
...	...	...	...
5886	1006036	26-35	4116058
5887	1006037	46-50	1119538
5888	1006038	55+	90034
5889	1006039	46-50	590319
5890	1006040	26-35	1653299

5891 rows × 3 columns

In [42]:
```
sample_size = 200
num_repitions = 1000

all_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for age_interval in age_intervals:
    all_means[age_interval] = []

for age_interval in age_intervals:
    for _ in range(num_repitions):
        mean = age_df[age_df['Age']==age_interval].sample(sample_size, replace=
        all_means[age_interval].append(mean)
```

In [43]:
```
for val in ['26-35','36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = age_df[age_df['Age']==val]

    margin_of_error_clt = 1.96*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - margin_of_error_clt
    upper_lim = sample_mean + margin_of_error_clt

age_CI=(lower_lim, upper_lim)
age_CI
```

Out[43]: (527662.4567141125, 710073.1671390985)

Observations

Male confidence interval of means (895617.83, 955070.97).

Female confidence interval od means (673254.77, 750794.02).

Married confidence interval of means: (806668.83, 880384.76).

Unmarried confidence interval of means: (848741.18, 912410.38).

For age 0-17 confidence interval of means: (527662.46, 710073.17).

For age 18-25 confidence interval of means: (801632.78, 908093.46).

For age 26-35 confidence interval of means: (945034.42, 1034284.21).

For age 36-45 confidence interval of means: (823347.80, 935983.62).

Insights

Male customers are more than the female customers.

Unmarried/single customers are more than married customers.

Most of the customers are from city category B.

There are 21 types of occupations and 21 different types of product categories.

Product categories 1,5,8,11 are having highest frequency of purchasing.

Recommendations

Male customers are purchasing more so wallmart should focus on retaining male customers and also to increase female customers make some promotions and advertisements.

Unmarried customers are spending more money, so company should focus on retaining unmarried customers.

Product categories 1,5,8 and 11 are most selling products so wallmart should increase tha availability of the prodcuts are under these categories and give discounts on other category products to increase the sale of those.

In [ ]: