Business Case: Walmart - Confidence Interval and CLT

About Walmart :

Walmart is an American multinational retail corporation that operates a chain of supercenters, discount departmental stores, and grocery stores from the United States. Walmart has more than 100 million customers worldwide.

Business Problem

The Management team at Walmart Inc. wants to analyze the customer purchase behavior (specifically, purchase amount) against the customer's gender and the various other factors to help the business make better decisions. They want to understand if the spending habits differ between male and female customers: Do women spend more on Black Friday than men? (Assume 50 million customers are male and 50 million are female).

Import the libraries:

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

```
from google.colab import files
uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving walmart.csv to walmart.csv
```

```
df = pd.read_csv('walmart.csv')
df.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
0	1000001	P00069042	F	0-17	10	A	2	0	3	83
1	1000001	P00248942	F	0-17	10	A	2	0	1	152
2	1000001	P00087842	F	0-17	10	A	2	0	12	14

```
df.isnull().head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purch
0	False	False	False	False	False	False	False	False	False	F
1	False	False	False	False	False	False	False	False	False	F
2	False	False	False	False	False	False	False	False	False	F
3	False	False	False	False	False	False	False	False	False	F
4	False	False	False	False	False	False	False	False	False	F

```
df.isnull().sum()/len(df)*100
```

```
User_ID                       0.0
Product_ID                    0.0
Gender                        0.0
Age                           0.0
Occupation                    0.0
City_Category                 0.0
Stay_In_Current_City_Years    0.0
Marital_Status                0.0
Product_Category              0.0
Purchase                      0.0
dtype: float64
```

```
df.describe(include='all').head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_(
count	5.500680e+05	550068	550068	550068	550068.000000	550068	550068	550068.000000	5500
unique	NaN	3631	2	7	NaN	3	5	NaN	
top	NaN	P00265242	M	26-35	NaN	B	1	NaN	

```
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
```

Initial Observations:

1. There are no missing values in the data.
2. There are 3631 unique product IDs in the dataset. P00265242 is the most sold Product ID.
3. There are 7 unique age groups and most of the purchase belongs to age 26-35 group.
4. There are 3 unique citi categories with category B being the highest.
5. 5 unique values for Stay_in_current_citi_years with 1 being the highest.
6. The difference between mean and median seems to be significant for purchase that suggests outliers in the data.
7. Minimum & Maximum purchase is 12 and 23961 suggests the purchasing behaviour is quite spread over a aignificant range of values. Mean is 9264 and 75% of purchase is of less than or equal to 12054. It suggest most of the purchase is not more than 12k.
8. Few categorical variable are of integer data type. It can be converted to character type.
9. Out of 550068 data points, 414259's gender is Male and rest are the female. Male purchase count is much higher than female.
10. Standard deviation for purchase have significant value which suggests data is more spread out for this attribute.

```
columns=['User_ID','Occupation', 'Marital_Status', 'Product_Category']
df[columns]=df[columns].astype('object')
```

```
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  object
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  object
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  object
 8   Product_Category            550068 non-null  object
 9   Purchase                    550068 non-null  int64
dtypes: int64(1), object(9)
memory usage: 42.0+ MB
```

```
df.describe(include='all')
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Catego
count	550068.0	550068	550068	550068	550068.0	550068	550068	550068.0	550068
unique	5891.0	3631	2	7	21.0	3	5	2.0	20
top	1001680.0	P00265242	M	26-35	4.0	B	1	0.0	5
freq	1026.0	1880	414259	219587	72308.0	231173	193821	324731.0	150933

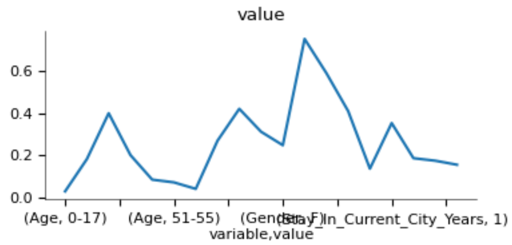
Observation post modifying the categorical variable's data type:

1. There are 5891 unique users, and userid 1001680 being with the highest count.
2. The customers belongs to 21 distinct occupation for the purchases being made with Occupation 4 being the highest.
3. Marital status unmarried contribute more in terms of the count for the purchase.
4. There are 20 unique product categories with 5 being the highest.

```
categ_cols = ['Gender', 'Age', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status']
df[categ_cols].melt().groupby(['variable', 'value'])[['value']].count()/len(df)
```

variable	value	value
Age	**0-17**	0.027455
	18-25	0.181178
	26-35	0.399200
	36-45	0.199999
	46-50	0.083082
	51-55	0.069993
	55+	0.039093
City_Category	**A**	0.268549
	B	0.420263
	C	0.311189
Gender	**F**	0.246895
	M	0.753105
Marital_Status	**0**	0.590347
	1	0.409653
Stay_In_Current_City_Years	**0**	0.135252
	1	0.352358
	2	0.185137
	3	0.173224
	4+	0.154028

Values



Distributions

1. 40% of the purchase done by aged 26-35 and 78% purchase are done by the customers aged between the age 18-45 (40%: 26-35, 18%: 18-25, 20%: 36-4
2. 75% of the purchase count are done by Male and 25% by Female
3. 60% Single, 40% Married contributes to the purchase count.
4. 35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years
5. There are 20 product categories in total.
6. There are 20 different types of occupations in the city.

Checking how the data is spread basis distinct users

```
df2=df.groupby(['User_ID'])['Age'].unique()
df2.value_counts()/len(df2)
```

```
[26-35]    0.348498
[36-45]    0.198099
[18-25]    0.181463
[46-50]    0.090137
[51-55]    0.081650
[55+]      0.063147
[0-17]     0.037006
Name: Age, dtype: float64
```

1. We can see 35% of the users are aged 26-35. 73% of users are aged between 18-45.
2. From the previous observation we saw 40% of the purchase are done by users aged 26-35. And, we have 35% of users aged between 26-35

```
df2=df.groupby(['User_ID'])['Gender'].unique()
df2.value_counts()/len(df2)
```

```
[M]    0.717196
[F]    0.282804
Name: Gender, dtype: float64
```

We have 72% male users and 28% female users. Combining with previous observations we can see 72% of male users contributing to 75% of the purchases

```
df2=df.groupby(['User_ID'])['Marital_Status'].unique()
df2.value_counts()/len(df2)
```

```
[0]    0.580037
[1]    0.419963
Name: Marital_Status, dtype: float64
```

We have 58% of the single users and 42% of married users. Combining with previous observation, single users contributes more as 58% of the single contributes to the 60% of the purchase count.

```
df2=df.groupby(['User_ID'])['City_Category'].unique()
df2.value_counts()/len(df2)
```

```
[C]    0.532847
[B]    0.289764
[A]    0.177389
Name: City_Category, dtype: float64
```

53% of the users belong to city category C whereas 29% to category B and 18% belong to category A

Checking the age group distribution in different city categories

```
pd.crosstab(index=df["City_Category"],columns=df["Age"],margins=True,normalize="index")
```

Age	0-17	18-25	26-35	36-45	46-50	51-55	55+
City_Category							
C	0.041612	0.168705	0.316974	0.209131	0.103333	0.085640	0.074506

We have seen earlier that city category B and A constitutes less percentage of total population, but they contribute more towards purchase count

Checking how genders are contributing towards toatl purchase amount

```
df2=pd.DataFrame(df.groupby(['Gender'])[['Purchase']].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

	Purchase	percent
Gender		
F	1186232642	23.278576
M	3909580100	76.721424

We can see male(72% of the population) contributes to more than 76% of the total purchase amount whereas female(28% of the population) contributes 23% of the total purchase amount.

Checking how purchase value are spread among differnt age categories

```
df2=pd.DataFrame(df.groupby(['Age'])[['Purchase']].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

	Purchase	percent
Age		
0-17	134913183	2.647530
18-25	913848675	17.933325
26-35	2031770578	39.871374
36-45	1026569884	20.145361
46-50	420843403	8.258612
51-55	367099644	7.203947
55+	200767375	3.939850

We can see the net purchase amount spread is similar to the purchase count spread among the different age groups.

```
df2=pd.DataFrame(df.groupby(['Marital_Status'])['Purchase'].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

	Purchase	percent
Marital_Status		
0	3008927447	59.047057
1	2086885295	40.952943

Single users are contributing 59% towards the total purchase amount in comparison to 41% by married users.

```
df2=pd.DataFrame(df.groupby(['City_Category'])['Purchase'].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

	Purchase	percent
City_Category		

City_category contribution to the total purchase amount is also similar to their contribution towards Purchase count.

Users with highest number of purchases

```
df.groupby(['User_ID'])['Purchase'].count().nlargest(10)
```

```
User_ID
1001680    1026
1004277     979
1001941     898
1001181     862
1000889     823
1003618     767
1001150     752
1001015     740
1005795     729
1005831     727
Name: Purchase, dtype: int64
```

Users with highest purchases amount

```
df.groupby(['User_ID'])['Purchase'].sum().nlargest(10)
```

```
User_ID
1004277    10536909
1001680     8699596
1002909     7577756
1001941     6817493
1000424     6573609
1004448     6566245
1005831     6512433
1001015     6511314
1003391     6477160
1001181     6387961
Name: Purchase, dtype: int64
```

The users with high number of purchases contribute more to the purchase amount. Still, we can see there are few users not in the list of top 10

```
df2=pd.DataFrame(df.groupby(['Occupation'])[['Purchase']].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

Occupation	Purchase	percent
0	635406958	12.469198
1	424614144	8.332609

Some of the Occupation like 0, 4, 7 has contributed more towards total purchase amount

3	162002168	3.179123

```
df2=pd.DataFrame(df.groupby(['Product_Category'])[['Purchase']].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

Product_Category	Purchase	percent
1	1910013754	37.482024
2	268516186	5.269350
3	204084713	4.004949
4	27380488	0.537313
5	941835229	18.482532
6	324150302	6.361111
7	60896731	1.195035
8	854318799	16.765114
9	6370324	0.125011
10	100837301	1.978827
11	113791115	2.233032
12	5331844	0.104632
13	4008601	0.078665
14	20014696	0.392767
15	92969042	1.824420
16	145120612	2.847840
17	5878699	0.115363
18	9290201	0.182310
19	59378	0.001165
20	944727	0.018539

1. 1, 8, 5 are among the highest yielding product categories and 19, 20, 13 are among the lowest in terms of their contribution to total amount.

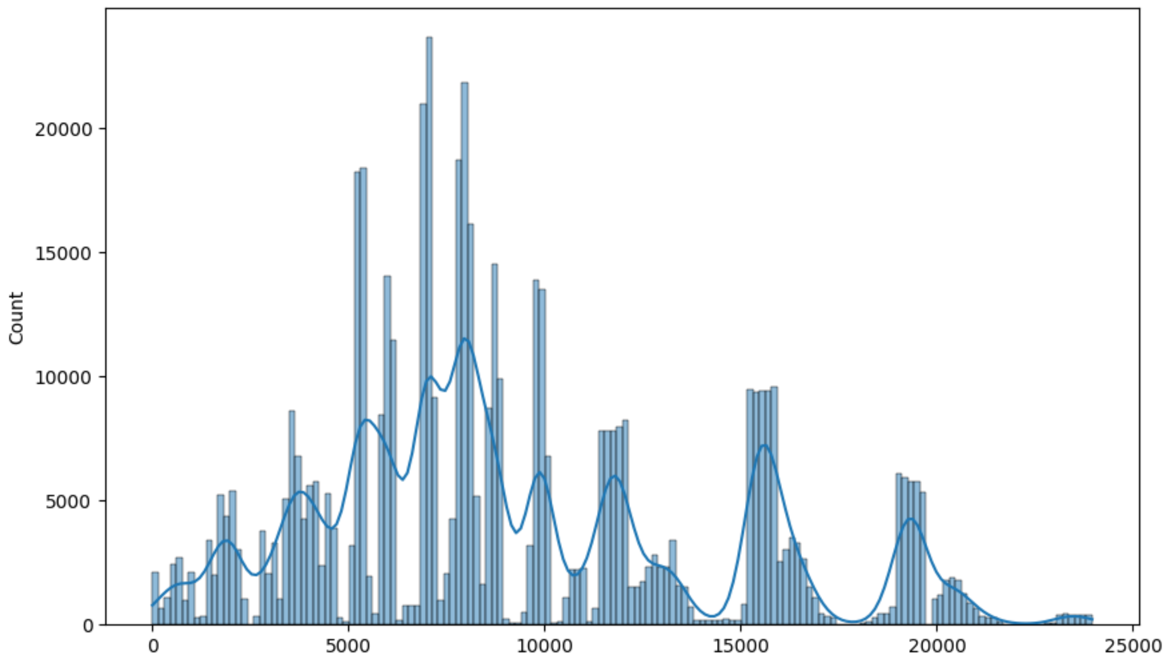
```
df2=pd.DataFrame(df.groupby(['Stay_In_Current_City_Years'])[['Purchase']].sum())
df2['percent'] = (df2['Purchase'] /
df2['Purchase'].sum()) * 100
df2
```

Stay_In_Current_City_Years	Purchase	percent
0	682979229	13.402754
1	1792872533	35.183250
2	949173931	18.626547
3	884902659	17.365290
4+	785884390	15.422160

Analysis: We can explore the distribution of the data for the quantitative attributes using histplot

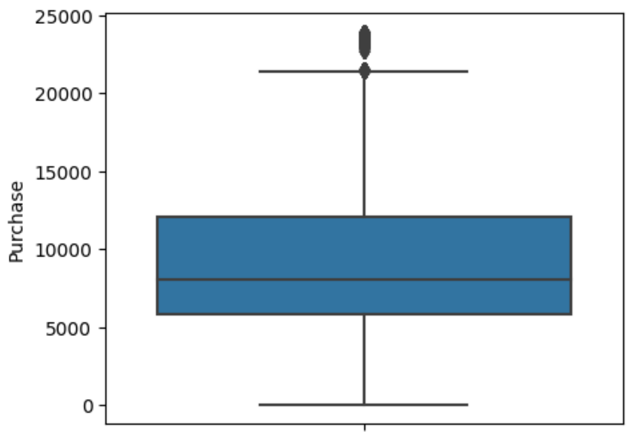
```
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x="Purchase", kde=True)
```

```
plt.show()
```



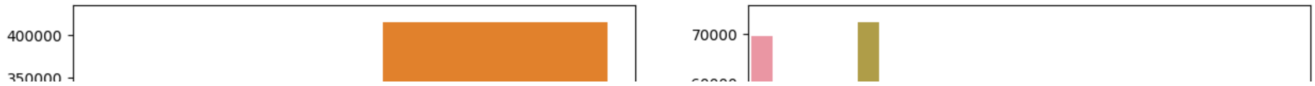
We can see purchase value between 5000 and 10000 have higher count. From the initial observation we have already seen the mean and median is 92

```
plt.figure(figsize=(5, 4))
sns.boxplot(data=df, y='Purchase')
plt.show()
```



We can see there are outliers in the data for purchase.

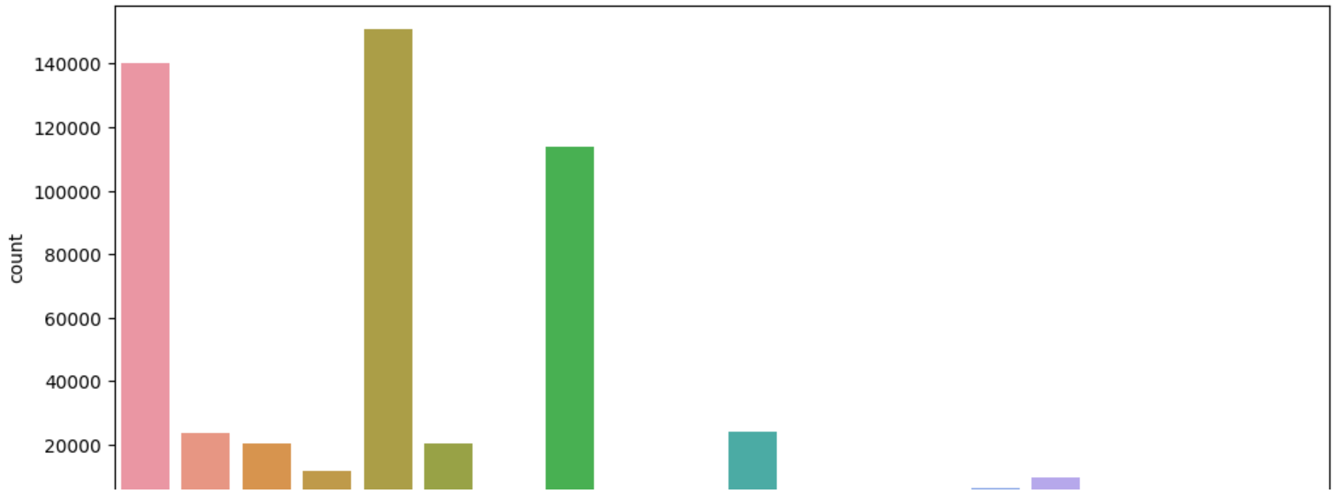
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(15, 10))
sns.countplot(data=df, x='Gender', ax=axs[0,0])
sns.countplot(data=df, x='Occupation', ax=axs[0,1])
sns.countplot(data=df, x='City_Category', ax=axs[1,0])
sns.countplot(data=df, x='Marital_Status', ax=axs[1,1])
plt.show()
```



Observations:

1. We can clearly see from the graphs above the purchases done by males are much higher than females.
2. We have 21 occupations categories. Occupation category 4, 0, and 7 are with higher number of purchases and category 8 with the lowest number of
3. The purchases are highest from City category B.
4. Single customer purchases are higher than married users.

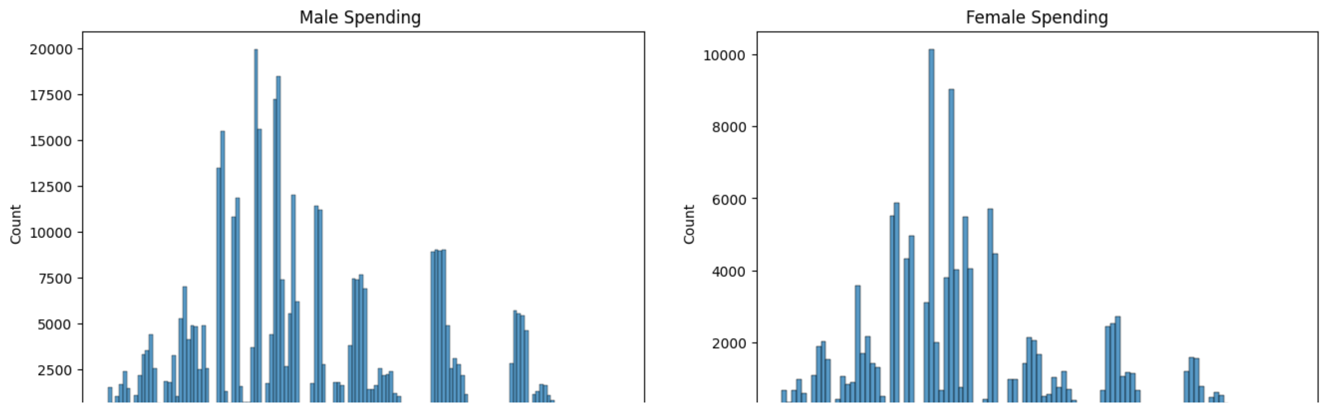
```
plt.figure(figsize=(12, 5))
sns.countplot(data=df, x='Product_Category')
plt.show()
```



There are 20 product categories with product category 1, 5 and 8 having higher purchasing frequency.

Bivariate Analysis:

```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(16,5))
sns.histplot(data=df[df['Gender']=='M']['Purchase'], ax=axs[0]).set_title("Male Spending ")
sns.histplot(data=df[df['Gender']=='F']['Purchase'], ax=axs[1]).set_title("Female Spending")
plt.show()
```

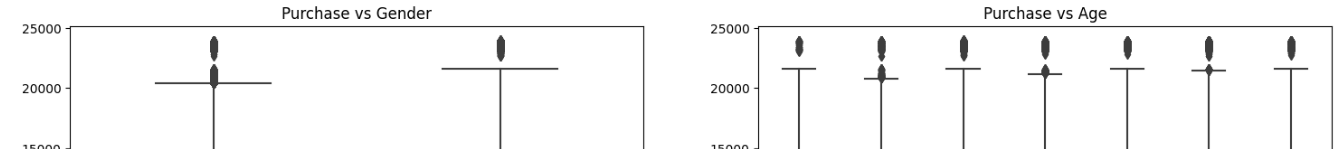


From the above histplot, we can clearly see spending behaviour is very much similar in nature for both males and females

```
attr = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status']
fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(18, 10))
fig.subplots_adjust(top=1.3)
count = 0
for row in range(3):
  for col in range(2):
    sns.boxplot(data=df, y='Purchase', x=attr[count], ax=axs[row, col],)
    axs[row,col].set_title(f"Purchase vs {attr[count]}")
    count += 1
plt.show()

plt.figure(figsize=(8, 5))
```

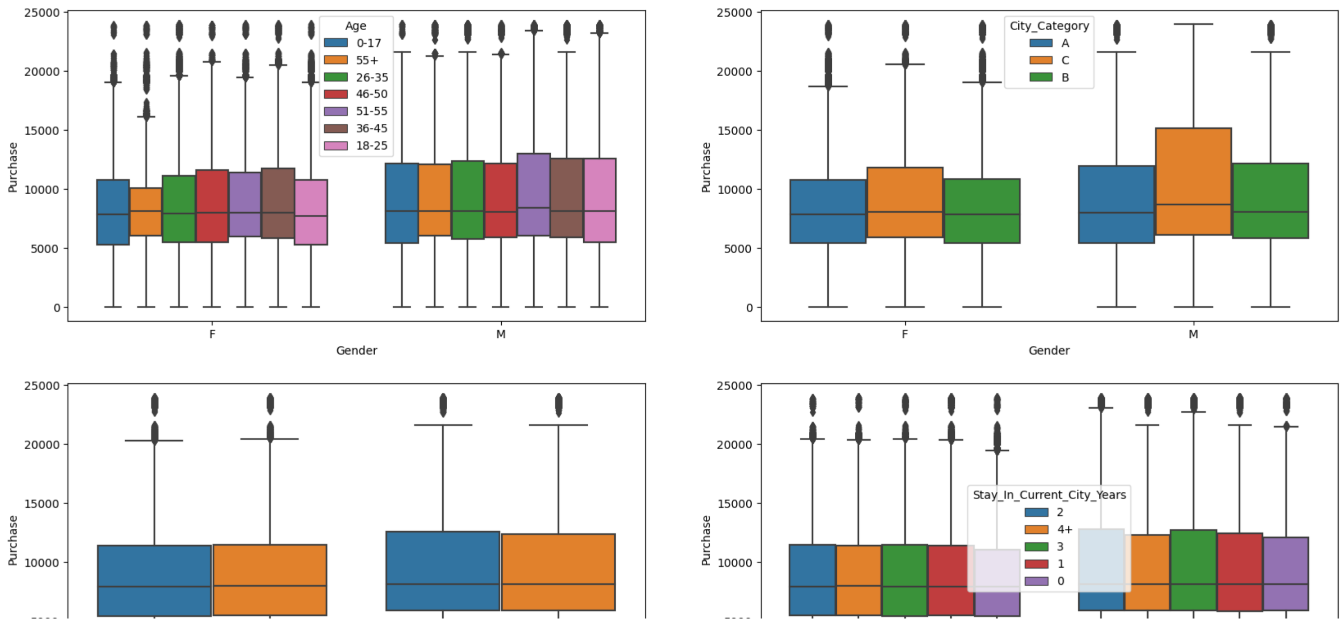
```
sns.boxplot(data=df, y='Purchase', x='Product_Category')
plt.show()
```



1. The spending behaviour for males and females are similar as we had seen from the above histplot. Males purchasing value are in the little highe
2. Among differnt age categories, we see similar purchase behaviour. For all age groups, most of the purchases are of the values between 5k to 12k
3. Among different occupation as well, we see similar purchasing behaviour in terms of the purchase values.
4. Similarly for City category, stay in current city years, marital status - we see the users spends mostly in the range of 5k to 12k.

Multivariate analysis

```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(20, 6))
fig.subplots_adjust(top=1.5)
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age', ax=axs[0,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category', ax=axs[0,1])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status', ax=axs[1,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Stay_In_Current_City_Years', ax=axs[1,1])
plt.show()
```



Observations:

1. The purchasing pattern is very much similar for males and females even among differnt age groups.
2. The purchasing behaviour of males and females basis different citi categories is also similar in nature. Still, males from city category B ten
3. Males and females spending behaviour remains similar even when take into account their marital status.
4. Purchase values are similar for males and females basis Stay_in_current_city_years. Although, Males buy slightly high value products.

Correlation between categorical variables:

```
avgamt_gender = df.groupby(['User_ID', 'Gender'])[['Purchase']].sum()
avgamt_gender = avgamt_gender.reset_index()
avgamt_gender
```

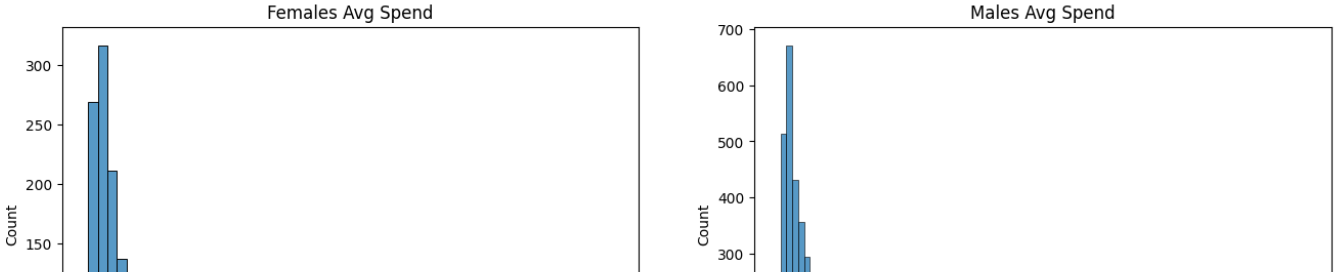
	User_ID	Gender	Purchase
0	1000001	F	334093
1	1000002	M	810472
2	1000003	M	341635
3	1000004	M	206468
4	1000005	M	821001

```
avgamt_gender['Gender'].value_counts()
```

```
M    4225
F    1666
Name: Gender, dtype: int64
```

```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(16,5))
sns.histplot(data=avgamt_gender[avgamt_gender['Gender']=='F']['Purchase'], ax=axs[0]).set_title("Females Avg Spend")
sns.histplot(data=avgamt_gender[avgamt_gender['Gender']=='M']['Purchase'], ax=axs[1]).set_title("Males Avg Spend")
```

```
Text(0.5, 1.0, 'Males Avg Spend')
```



Average amount spend by males are higher than females.

```
avgamt_gender.groupby(['Gender'])[['Purchase']].mean()
```

	Purchase
Gender	
F	712024.394958
M	925344.402367

```
avgamt_gender.groupby(['Gender'])['Purchase'].sum()
```

```
Gender
F    1186232642
M    3909580100
Name: Purchase, dtype: int64
```

Observations:

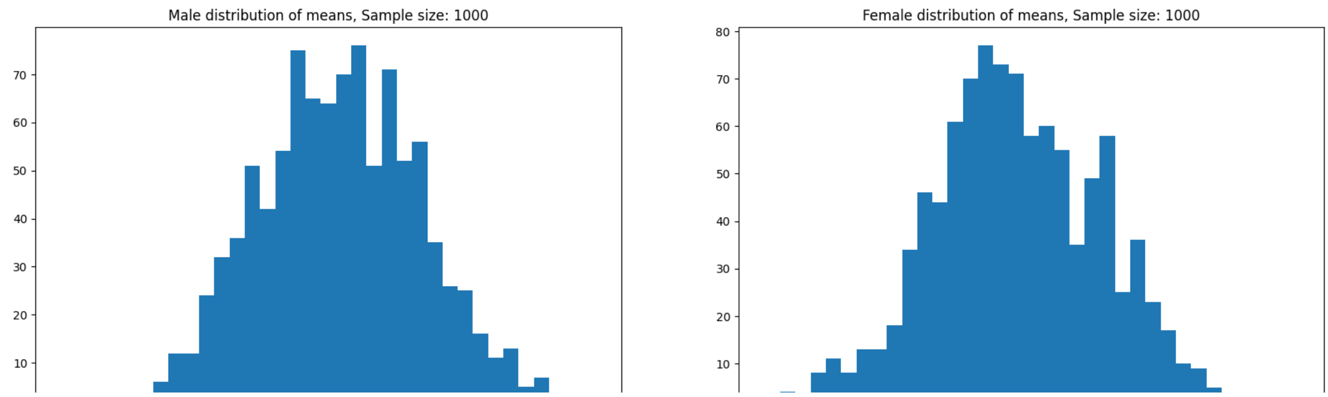
1. Average amount for the males is 925344 for the entire population whereas it's much lesser for females(712024).
2. Total amount spend by males is around 4 billion whereas for females it's 1.2 billion.

```
avgamt_male = avgamt_gender[avgamt_gender['Gender']=='M']
avgamt_female = avgamt_gender[avgamt_gender['Gender']=='F']
```

Finding the sample(sample size=1000) for avg purchase amount for males and females

```
sample_size = 1000
num_repitions = 1000
male_means = []
female_means = []
for i in range(num_repitions):
  male_mean = avgamt_male.sample(sample_size, replace=True)['Purchase'].mean()
  female_mean = avgamt_female.sample(sample_size, replace=True)['Purchase'].mean()
  male_means.append(male_mean)
  female_means.append(female_mean)
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)
axis[0].set_title("Male distribution of means, Sample size: 1000")
```

```
axis[1].set_title("Female distribution of means, Sample size: 1000")
plt.show()
```



Observations:

1. The means sample seems to be normally distributed for both males and females

Calculating 90% confidence interval for sample size 1000:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1000)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1000)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1000)
sample_std_error_female=sample_std_female/np.sqrt(1000)
Upper_Limit_male=z90*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z90*sample_std_error_male
Upper_Limit_female=z90*sample_std_error_female + sample_mean_female
Lower_Limit_female=sample_mean_female - z90*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 925558.21
Sample avg spend amount for Female: 712007.93

Sample std for Male: 31016.56
Sample std for Female: 24948.65

Sample std error for Male: 980.83
Sample std error for Female: 788.95

Male_CI:  [923944.7442159649, 927171.6744800351]
Female_CI:  [710710.1096089971, 713305.7404150029]
```

Now using the Confidence interval at 90%, we can say that: Average amount spend by male customers lie in the range 9,22,940.71 - 9,26,225.18 Average amount spend by female customers lie in range 7,10,425.64 - 7,13,064.55

Calculating 95% confidence interval for sample size 1000:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
```

```
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1000)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1000)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1000)
sample_std_error_female=sample_std_female/np.sqrt(1000)
Upper_Limit_male=z95*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z95*sample_std_error_male
Upper_Limit_female=z95*sample_std_error_female + sample_mean_female
Lower_Limit_female=sample_mean_female - z95*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 925558.21
Sample avg spend amount for Female: 712007.93

Sample std for Male: 31016.56
Sample std for Female: 24948.65

Sample std error for Male: 980.83
Sample std error for Female: 788.95

Male_CI:  [923635.7828077029, 927480.6358882971]
Female_CI:  [710461.591765869, 713554.2582581311]
```

Observation: Now using the Confidence interval at 95%, we can say that: Average amount spend by male customers lie in the range 9,22,626.24 - 9,26,539. Average amount spend by female customers lie in range 7,10,172.98 - 7,13,317.21 65

Calculating 99% confidence interval for sample size 1000:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1000)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1000)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1000)
sample_std_error_female=sample_std_female/np.sqrt(1000)
Upper_Limit_male=z99*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z99*sample_std_error_male
Upper_Limit_female=z99*sample_std_error_female + sample_mean_female
Lower_Limit_female=sample_mean_female - z99*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 925558.21
Sample avg spend amount for Female: 712007.93

Sample std for Male: 31016.56
Sample std for Female: 24948.65

Sample std error for Male: 980.83
Sample std error for Female: 788.95

Male_CI:  [923031.5916093237, 928084.8270866763]
Female_CI:  [709975.6013170849, 714040.2487069152]
```

Observation: Now using the Confidence interval at 99%, we can say that:

Average amount spend by male customers lie in the range 9,22,011.28 - 9,27,154.61

Average amount spend by female customers lie in range 7,09,678.88 - 7,13,811.31

Calculating 90% confidence interval for sample size 1500:

Finding the sample(sample size=1000) avg purchase amount for males and females

```
genders = ["M", "F"]
sample_size = 1500
num_repitions = 1000
male_means = []
female_means = []
for i in range(num_repitions):
  male_mean = avgamt_male.sample(sample_size, replace=True)['Purchase'].mean()
  female_mean = avgamt_female.sample(sample_size, replace=True)['Purchase'].mean()
male_means.append(male_mean)
female_means.append(female_mean)
```

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1500)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1500)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1500)
sample_std_error_female=sample_std_female/np.sqrt(1500)
Upper_Limit_male=z90*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z90*sample_std_error_male
Upper_Limit_female=z90*sample_std_error_female + sample_mean_female
Lower_Limit_female=sample_mean_female - z90*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 904322.16
Sample avg spend amount for Female: 712835.77

Sample std for Male: nan
Sample std for Female: nan

Sample std error for Male: nan
Sample std error for Female: nan

Male_CI:  [nan, nan]
Female_CI:  [nan, nan]
```

Observation: Now using the Confidence interval at 90%, we can say that: Average amount spend by male customers lie in the range 9,24,177.41 - 9,26,318.90 Average amount spend by female customers lie in range 7,11,187.27 - 7,12,971.67 By increasing the sample size we can see confidence interval is more closer to the population mean.

Calculating 95% confidence interval for sample size 1500:

```
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1500)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1500)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1500)
sample_std_error_female=sample_std_female/np.sqrt(1500)
Upper_Limit_male=z95*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z95*sample_std_error_male
Upper_Limit_female=z95*sample_std_error_female + sample_mean_female
```

```
Lower_Limit_female=sample_mean_female - z95*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 904322.16
Sample avg spend amount for Female: 712835.77

Sample std for Male: nan
Sample std for Female: nan

Sample std error for Male: nan
Sample std error for Female: nan

Male_CI:  [nan, nan]
Female_CI:  [nan, nan]
```

Observation: Now using the Confidence interval at 95%, we can say that: Average amount spend by male customers lie in the range 9,23,972.41 - 9,26,523.93 Average amount spend by female customers lie in range 7,11,016.42 - 7,13,142.51 By increasing the sample size we can see confidence interval is more closer to the population mean.

Calculating 99% confidence interval for sample size 1500

```
print("Population avg spend amount for Male: {:.2f}".format(avgamt_male['Purchase'].mean()))
print("Population avg spend amount for Female: {:.2f}\n".format(avgamt_female['Purchase'].mean()))
print("Sample avg spend amount for Male: {:.2f}".format(np.mean(male_means)))
print("Sample avg spend amount for Female: {:.2f}\n".format(np.mean(female_means)))
print("Sample std for Male: {:.2f}".format(pd.Series(male_means).std()))
print("Sample std for Female: {:.2f}\n".format(pd.Series(female_means).std()))
print("Sample std error for Male: {:.2f}".format(pd.Series(male_means).std()/np.sqrt(1500)))
print("Sample std error for Female: {:.2f}\n".format(pd.Series(female_means).std()/np.sqrt(1500)))
sample_mean_male=np.mean(male_means)
sample_mean_female=np.mean(female_means)
sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()
sample_std_error_male=sample_std_male/np.sqrt(1500)
sample_std_error_female=sample_std_female/np.sqrt(1500)
Upper_Limit_male=z99*sample_std_error_male + sample_mean_male
Lower_Limit_male=sample_mean_male - z99*sample_std_error_male
Upper_Limit_female=z99*sample_std_error_female + sample_mean_female
Lower_Limit_female=sample_mean_female - z99*sample_std_error_female
print("Male_CI: ",[Lower_Limit_male,Upper_Limit_male])
print("Female_CI: ",[Lower_Limit_female,Upper_Limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Sample avg spend amount for Male: 904322.16
Sample avg spend amount for Female: 712835.77

Sample std for Male: nan
Sample std for Female: nan

Sample std error for Male: nan
Sample std error for Female: nan

Male_CI:  [nan, nan]
Female_CI:  [nan, nan]
```

Observation: Now using the Confidence interval at 99%, we can say that: Average amount spend by male customers lie in the range 923571.42 - 926924.89 Average amount spend by female customers lie in range 710682.32 - 713476.61 By increasing the sample size we can see confidence interval is more closer to the population mean.

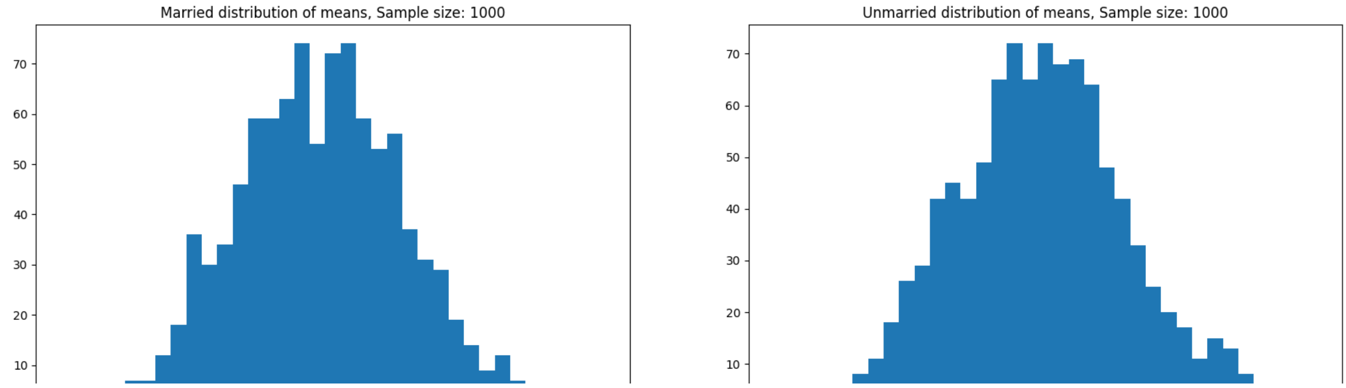
CLT and Confidence interval considering marital status:

```
avg_Marital = df.groupby(['User_ID', 'Marital_Status'])[['Purchase']].sum()
avg_Marital = avg_Marital.reset_index()
avgamt_married = avg_Marital[avg_Marital['Marital_Status']==1]
avgamt_single = avg_Marital[avg_Marital['Marital_Status']==0]
sample_size = 1000
num_repitions = 1000
married_means = []
single_means = []

for i in range(num_repitions):
  avg_married = avg_Marital[avg_Marital['Marital_Status']==1].sample(sample_size, replace=True)['Purchase'].mean()
```

```
    avg_single = avg_Marital[avg_Marital['Marital_Status']==0].sample(sample_size, replace=True)['Purchase'].mean()
    married_means.append(avg_married)
    single_means.append(avg_single)

fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))
axis[0].hist(married_means, bins=35)
axis[1].hist(single_means, bins=35)
axis[0].set_title("Married distribution of means, Sample size: 1000")
axis[1].set_title("Unmarried distribution of means, Sample size: 1000")
plt.show()
```



Observations:

1. The means sample seems to be normally distributed for both married and singles. Also, we can see the mean of the sample means are closer to the population mean as per central limit theorem.

```
avg_Marital['Marital_Status'].value_counts()
```

```
0    3417
1    2474
Name: Marital_Status, dtype: int64
```

Calculating 90% confidence interval for avg expenses for married/single for sample size 1000:

Taking the values for z at 90%, 95% and 99% confidence interval as:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Married: {:.2f}".format(avgamt_married['Purchase'].mean()))
print("Population avg spend amount for Single: {:.2f}\n".format(avgamt_single['Purchase'].mean()))
print("Sample avg spend amount for Married: {:.2f}".format(np.mean(married_means)))
print("Sample avg spend amount for Single: {:.2f}\n".format(np.mean(single_means)))
print("Sample std for Married: {:.2f}".format(pd.Series(married_means).std()))
print("Sample std for Single: {:.2f}\n".format(pd.Series(single_means).std()))
print("Sample std error for Married: {:.2f}".format(pd.Series(married_means).std()/np.sqrt(1000)))
print("Sample std error for Single: {:.2f}\n".format(pd.Series(single_means).std()/np.sqrt(1000)))
sample_mean_married=np.mean(married_means)
sample_mean_single=np.mean(single_means)
sample_std_married=pd.Series(married_means).std()
sample_std_single=pd.Series(single_means).std()
sample_std_error_married=sample_std_married/np.sqrt(1000)
sample_std_error_single=sample_std_single/np.sqrt(1000)
Upper_Limit_married=z90*sample_std_error_male + sample_mean_married
Lower_Limit_married=sample_mean_married - z90*sample_std_error_married
Upper_Limit_single=z90*sample_std_error_single + sample_mean_single
Lower_Limit_single=sample_mean_single - z90*sample_std_error_single
print("Married_CI: ",[Lower_Limit_married,Upper_Limit_married])
print("Single_CI: ",[Lower_Limit_single,Upper_Limit_single])
```

```
Population avg spend amount for Married: 843526.80
Population avg spend amount for Single: 880575.78

Sample avg spend amount for Married: 844254.71
Sample avg spend amount for Single: 882729.74

Sample std for Married: 29241.27
Sample std for Single: 30800.17

Sample std error for Married: 924.69
Sample std error for Single: 973.99

Married_CI:  [842733.5934802435, nan]
Single_CI:  [881127.5290807011, 884331.9456912987]
```

Calculating 95% confidence interval for avg expenses for married/single for sample size 1000:

Taking the values for z at 90%, 95% and 99% confidence interval as:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Married: {:.2f}".format(avgamt_married['Purchase'].mean()))
print("Population avg spend amount for Single: {:.2f}\n".format(avgamt_single['Purchase'].mean()))
print("Sample avg spend amount for Married: {:.2f}".format(np.mean(married_means)))
print("Sample avg spend amount for Single: {:.2f}\n".format(np.mean(single_means)))
print("Sample std for Married: {:.2f}".format(pd.Series(married_means).std()))
print("Sample std for Single: {:.2f}\n".format(pd.Series(single_means).std()))
print("Sample std error for Married: {:.2f}".format(pd.Series(married_means).std()/np.sqrt(1000)))
print("Sample std error for Single: {:.2f}\n".format(pd.Series(single_means).std()/np.sqrt(1000)))
sample_mean_married=np.mean(married_means)
sample_mean_single=np.mean(single_means)
sample_std_married=pd.Series(married_means).std()
sample_std_single=pd.Series(single_means).std()
sample_std_error_married=sample_std_married/np.sqrt(1000)
sample_std_error_single=sample_std_single/np.sqrt(1000)
Upper_Limit_married=z95*sample_std_error_male + sample_mean_married
Lower_Limit_married=sample_mean_married - z95*sample_std_error_married
Upper_Limit_single=z95*sample_std_error_single + sample_mean_single
Lower_Limit_single=sample_mean_single - z95*sample_std_error_single
print("Married_CI: ",[Lower_Limit_married,Upper_Limit_married])
print("Single_CI: ",[Lower_Limit_single,Upper_Limit_single])
```

```
Population avg spend amount for Married: 843526.80
Population avg spend amount for Single: 880575.78

Sample avg spend amount for Married: 844254.71
Sample avg spend amount for Single: 882729.74

Sample std for Married: 29241.27
Sample std for Single: 30800.17

Sample std error for Married: 924.69
Sample std error for Single: 973.99

Married_CI:  [842442.31612129, nan]
Single_CI:  [880820.7232350056, 884638.7515369941]
```

Calculating 99% confidence interval for avg expenses for married/single for sample size 1000:

Taking the values for z at 90%, 95% and 99% confidence interval as:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
print("Population avg spend amount for Married: {:.2f}".format(avgamt_married['Purchase'].mean()))
print("Population avg spend amount for Single: {:.2f}\n".format(avgamt_single['Purchase'].mean()))
print("Sample avg spend amount for Married: {:.2f}".format(np.mean(married_means)))
print("Sample avg spend amount for Single: {:.2f}\n".format(np.mean(single_means)))
print("Sample std for Married: {:.2f}".format(pd.Series(married_means).std()))
print("Sample std for Single: {:.2f}\n".format(pd.Series(single_means).std()))
print("Sample std error for Married: {:.2f}".format(pd.Series(married_means).std()/np.sqrt(1000)))
print("Sample std error for Single: {:.2f}\n".format(pd.Series(single_means).std()/np.sqrt(1000)))
sample_mean_married=np.mean(married_means)
sample_mean_single=np.mean(single_means)
sample_std_married=pd.Series(married_means).std()
sample_std_single=pd.Series(single_means).std()
sample_std_error_married=sample_std_married/np.sqrt(1000)
sample_std_error_single=sample_std_single/np.sqrt(1000)
Upper_Limit_married=z99*sample_std_error_male + sample_mean_married
Lower_Limit_married=sample_mean_married - z99*sample_std_error_married
Upper_Limit_single=z99*sample_std_error_single + sample_mean_single
Lower_Limit_single=sample_mean_single - z99*sample_std_error_single
print("Married_CI: ",[Lower_Limit_married,Upper_Limit_married])
print("Single_CI: ",[Lower_Limit_single,Upper_Limit_single])
```

```
Population avg spend amount for Married: 843526.80
Population avg spend amount for Single: 880575.78

Sample avg spend amount for Married: 844254.71
Sample avg spend amount for Single: 882729.74
```

```
Sample std for Married: 29241.27
Sample std for Single: 30800.17

Sample std error for Married: 924.69
Sample std error for Single: 973.99

Married_CI:  [841872.7070637811, nan]
Single_CI:  [880220.7473589788, 885238.7274130209]
```

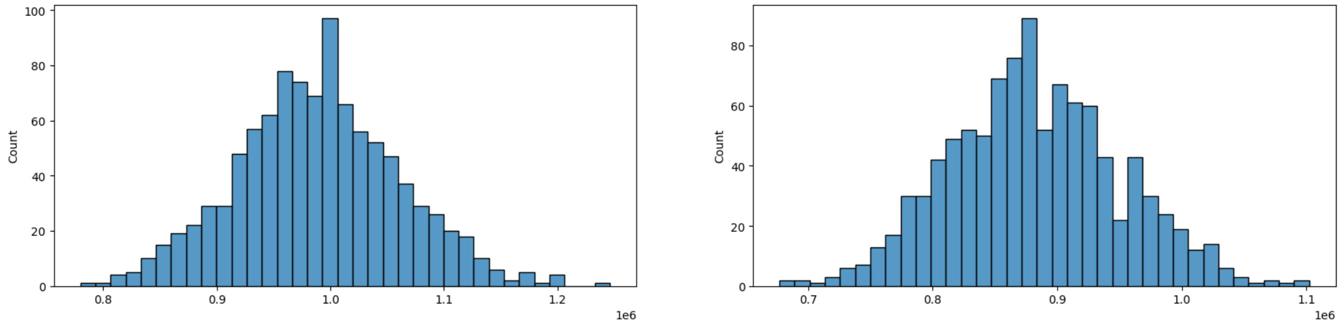
Observation: For married and singles, it can be seen with larger sample size the sample mean gets closer to tthe population mean.

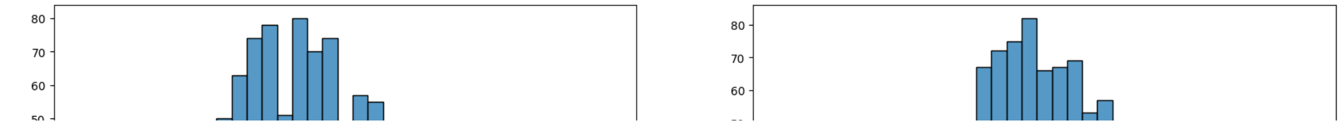
```
avgamt_age = df.groupby(['User_ID', 'Age'])[['Purchase']].sum()
avgamt_age = avgamt_age.reset_index()
avgamt_age['Age'].value_counts()
```

```
26-35    2053
36-45    1167
18-25    1069
46-50     531
51-55     481
55+       372
0-17      218
Name: Age, dtype: int64
```

```
sample_size = 200
num_repitions = 1000
all_sample_means = {}
age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
  all_sample_means[i] = []
for i in age_intervals:
  for j in range(num_repitions):
    mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
    all_sample_means[i].append(mean)

fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(20, 15))
sns.histplot(all_sample_means['26-35'],bins=35,ax=axis[0,0])
sns.histplot(all_sample_means['36-45'],bins=35,ax=axis[0,1])
sns.histplot(all_sample_means['18-25'],bins=35,ax=axis[1,0])
sns.histplot(all_sample_means['46-50'],bins=35,ax=axis[1,1])
sns.histplot(all_sample_means['51-55'],bins=35,ax=axis[2,0])
sns.histplot(all_sample_means['55+'],bins=35,ax=axis[2,1])
plt.show()
plt.figure(figsize=(10, 5))
sns.histplot(all_sample_means['0-17'],bins=35)
plt.show()
```





Observations:

1. The means sample seems to be normally distributed for all age groups. Also, we can see the mean of the sample means are closer to the population mean as per central limit theorem.

Calculating 90% confidence interval for avg expenses for different age groups for sample size 200:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
sample_size = 200
num_repitions = 1000
all_population_means={}
all_sample_means = {}
age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
  all_sample_means[i] = []
  all_population_means[i]=[]
  population_mean=avgamt_age[avgamt_age['Age']==i]['Purchase'].mean()
  all_population_means[i].append(population_mean)
print("All age group population mean: \n", all_population_means)
print("\n")

for i in age_intervals:
  for j in range(num_repitions):
    mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
    all_sample_means[i].append(mean)

for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:
  new_df = avgamt_age[avgamt_age['Age']==val]
  std_error = z90*new_df['Purchase'].std()/np.sqrt(len(new_df))
  sample_mean = new_df['Purchase'].mean()
  lower_lim = sample_mean - std_error
  upper_lim = sample_mean + std_error
  print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
All age group population mean: 
 {'26-35': [989659.3170969313], '36-45': [879665.7103684661], '18-25': [854863.119738073], '46-50': [792548.7815442561], '51-55': [


For age 26-35 confidence interval of means: (952206.28, 1027112.35)
For age 36-45 confidence interval of means: (832398.89, 926932.53)
For age 18-25 confidence interval of means: (810187.65, 899538.59)
For age 46-50 confidence interval of means: (726209.00, 858888.57)
For age 51-55 confidence interval of means: (703772.36, 822629.48)
For age 55+ confidence interval of means: (487032.92, 592361.57)
For age 0-17 confidence interval of means: (542320.46, 695415.16)
```

Calculating 95% confidence interval for avg expenses for different age groups for sample size 200:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
sample_size = 200
num_repitions = 1000
all_means = {}
age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
  all_means[i] = []
```

```
for i in age_intervals:
  for j in range(num_repitions):
    mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
    all_means[i].append(mean)

for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:
  new_df = avgamt_age[avgamt_age['Age']==val]
  std_error = z95*new_df['Purchase'].std()/np.sqrt(len(new_df))
  sample_mean = new_df['Purchase'].mean()
  lower_lim = sample_mean - std_error
  upper_lim = sample_mean + std_error
  print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
For age 26-35 confidence interval of means: (945034.42, 1034284.21)
For age 36-45 confidence interval of means: (823347.80, 935983.62)
For age 18-25 confidence interval of means: (801632.78, 908093.46)
For age 46-50 confidence interval of means: (713505.63, 871591.93)
For age 51-55 confidence interval of means: (692392.43, 834009.42)
For age 55+ confidence interval of means: (476948.26, 602446.23)
For age 0-17 confidence interval of means: (527662.46, 710073.17)
```

Calculating 99% confidence interval for avg expenses for different age groups for sample size 200:

```
z90=1.645 #90% Confidence Interval
z95=1.960 #95% Confidence Interval
z99=2.576 #99% Confidence Interval
sample_size = 200
num_repitions = 1000
all_means = {}
age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
  all_means[i] = []
for i in age_intervals:
  for j in range(num_repitions):
    mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
    all_means[i].append(mean)
for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:
  new_df = avgamt_age[avgamt_age['Age']==val]
  std_error = z99*new_df['Purchase'].std()/np.sqrt(len(new_df))
  sample_mean = new_df['Purchase'].mean()
  lower_lim = sample_mean - std_error
  upper_lim = sample_mean + std_error
  print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
For age 26-35 confidence interval of means: (931009.46, 1048309.18)
For age 36-45 confidence interval of means: (805647.89, 953683.53)
For age 18-25 confidence interval of means: (784903.24, 924823.00)
For age 46-50 confidence interval of means: (688663.50, 896434.06)
For age 51-55 confidence interval of means: (670138.33, 856263.52)
For age 55+ confidence interval of means: (457227.15, 622167.34)
For age 0-17 confidence interval of means: (498997.92, 738737.71)
```

Answering The Questions

1.Are women spending more money per transaction than men? Why or Why not? Ans: No. CI's of male and female do not overlap and upper limits of female purchase CI are lesser than lower limits of male purchase CI. This prov The reason for less purchase by women could have several factors: Males might be doing the purchase for females. Salary can be a factor in less purchase. We also need to see whether male-based products were sold more than women-based products to clearly identify difference in spending pattern. If the female based products quality/quantity needs to be improved for women purchasing.

2. Confidence intervals and distribution of the mean of the expenses by female and male customers. At 99% Confidence Interval with sample size 1000 Average amount spend by male customers lie in the range 9,22,011.28 - 9,27,154.61 Average amount spend by female customers lie in range 7,09,678.88 - 7,13,811.31

3. Are confidence intervals of average male and female spending overlapping? How can Walmart leverage this conclusion to make changes or improveme **Ans**: No. Confidence intervals of average male and female spending are not overlapping

4. Results when the same activity is performed for Married vs Unmarried At 99% Confidence Interval with sample size 1000 Average amount spend by married customers lie in the range: [841059.6309378392, 845078.140167503] Average amount spend by unmarried customers lie in the range: [879093.3492016713, 884078.6782803286]

5. Results when the same activity is performed for Age

At 99% Confidence Interval with sample size 200 For age 26-35 confidence interval of means: (931009.46,1048309.18) For age 36-45 confidence interval of means: (805647.89, 953683.53) For age 18-25 confidence interval of means: (784903.24, 924823.00) For age 46-50 confidence interval of means: (688663.50, 896434.06) For age 51-55 confidence interval of means: (670138.33, 856263.52) For age 55+ confidence interval of means: (457227.15, 622167.34) For age 0-17 confidence interval of means: (498997.92, 738737.71)

Recommendations:

1. Men spent more money than women, company can focus on retaining the male customers and getting more male customers.
2. Product_Category - 1, 5, 8 have highest purchasing frequency. it means these are the products in these categories are in more demand. Company c
3. Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
4. Customers in the age 26-35 spend more money than the others, So company should focus on acquisition of customers who are in the age 26-35.
5. We have more customers aged 26-35 in the city category B and A, company can focus more on these customers for these cities to increase the busi
6. Male customers living in City_Category C spend more money than other male customers living in B or C, Selling more products in the City_Categor
7. Some of the Product category like 19,20,13 have very less purchase. Company can think of dropping it.
8. The top 10 users who have purchased more company should give more offers and discounts so that they can be retained and can be helpful for comp
9. The occupation which are contributing more company can think of offering credit cards or other benefits to those customers by liasing with some
10. The top products should be given focus in order to maintain the quality in order to further increase the sales of those products.
11. People who are staying in city for an year have contributed to 35% of the total purchase amount. Company can focus on such customer base who a
12. We have highest frequency of purchase order between 5k and 10k, company can focus more on these mid range products to increase the sales.