Business Case: Walmart - Confidence Interval and CLT

1. Defining Problem Statement and Analyzing basic metrics

Bussiness 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.

1.1 Observations on shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), statistical summary

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [46]:
```
from scipy.stats import norm, binom
```

In [2]:
```
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

In [3]:
```
df=pd.read_csv("walmart.txt")
df.head()
```

Out[3]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

In [4]:
```
df.columns
```

Out[4]:
```
Index(['User_ID', 'Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category',
       'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category',
       'Purchase'],
      dtype='object')
```

In [5]:
```
df.shape
```

Out[5]:
```
(550068, 10)
```

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 [7]:
```
cols = ['Occupation', 'Marital_Status', 'Product_Category']
df[cols] = df[cols].astype('object')
```

In [8]:
```
df.dtypes
```

Out[8]:
```
User_ID                      int64
Product_ID                  object
Gender                      object
Age                         object
Occupation                  object
City_Category               object
Stay_In_Current_City_Years  object
Marital_Status              object
Product_Category            object
Purchase                     int64
dtype: object
```

In [9]:
```
df.describe(include="all")
```

Out[9]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068	550068	550068	550068.0	550068	550068	550068.0	550068.0	550068.000000
unique	NaN	3631	2	7	21.0	3	5	2.0	20.0	NaN
top	NaN	P00265242	M	26-35	4.0	B	1	0.0	5.0	NaN
freq	NaN	1880	414259	219587	72308.0	231173	193821	324731.0	150933.0	NaN
mean	1.003029e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	9263.968713
std	1.727592e+03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5023.065394
min	1.000001e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	12.000000
25%	1.001516e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5823.000000
50%	1.003077e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8047.000000
75%	1.004478e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	12054.000000
max	1.006040e+06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	23961.000000

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
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
```

```
The given data consists of 550068 data points across 10 fields. Its a sample data.
There are no missing/null values.
Usre_id and Purchase are int64 type rest are converted to object type
```

In [ ]:
```

```

1.2: Non-Graphical Analysis: Value counts and unique attributes

In [12]:
```
df['User_ID'].nunique()
```

Out[12]:
```
5891
```

In [13]:
```
df['Product_ID'].nunique()
```

Out[13]:
```
3631
```

In [14]:
```
col_unique=['User_ID', 'Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category',
       'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category',
       'Purchase']
df[col_unique].nunique()
```

Out[14]:
```
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 [15]:

```
categorical_cols = ['Gender', 'Age', 'Occupation', 'City_Category',
                    'Stay_In_Current_City_Years','Marital_Status', 'Product_Category']
df[categorical_cols].melt().groupby(['variable', 'value'])[['value']].count()/len(df)*100
```

Out[15]:

		value
variable	value	
Age	0-17	2.745479
	18-25	18.117760
	26-35	39.919974
	36-45	19.999891
	46-50	8.308246
	51-55	6.999316
	55+	3.909335
City_Category	A	26.854862
	B	42.026259
	C	31.118880
Gender	F	24.689493
	M	75.310507
Marital_Status	0	59.034701
	1	40.965299
Occupation	0	12.659889
	1	8.621843
	2	4.833584
	3	3.208694
	4	13.145284
	5	2.213726
	6	3.700452
	7	10.750125
	8	0.281056
	9	1.143677
	10	2.350618
	11	2.106285
	12	5.668208
	13	1.404917
	14	4.964659
	15	2.211545
	16	4.612339
	17	7.279645
	18	1.203851
	19	1.538173
	20	6.101427

		value
variable	value	
Product_Category	1	25.520118
	2	4.338373
	3	3.674637
	4	2.136645
	5	27.438971
	6	3.720631
	7	0.676462
	8	20.711076
	9	0.074536
	10	0.931703
	11	4.415272
	12	0.717548
	13	1.008784
	14	0.276875
	15	1.143495
	16	1.786688
	17	0.105078
	18	0.568112
	19	0.291419
	20	0.463579
Stay_In_Current_City_Years	0	13.525237
	1	35.235825
	2	18.513711
	3	17.322404
	4+	15.402823

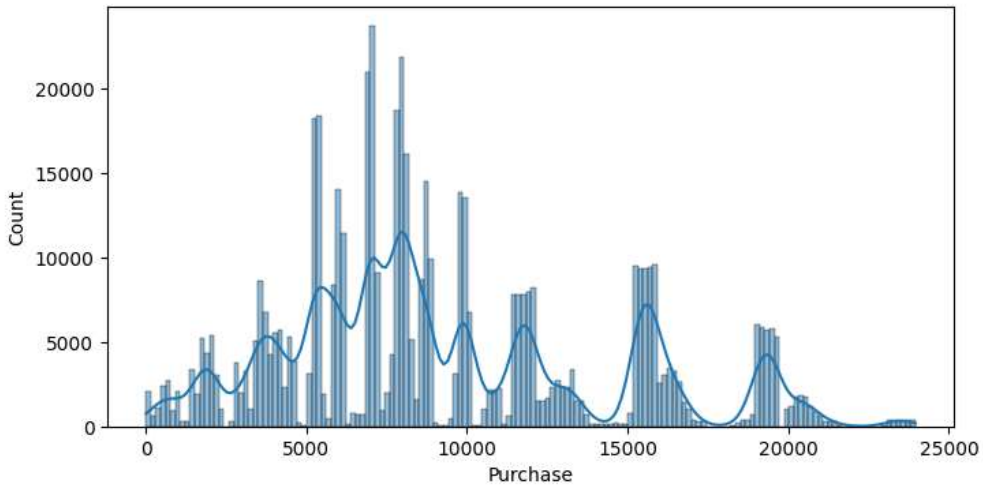
```
Observations:
~80% of the users are between the age 18-50 (40%: 26-35, 18%: 18-25, 20%: 36-45).
75.31% of the users are Male and 24.69% are Female.
59.03% Single, 40.97% Married.
35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years.
Total of 20 product categories are there.
There are 20 differnent types of occupations in the city
```

In [ ]:
```
```

1.3: Visual Analysis - Univariate & Bivariate

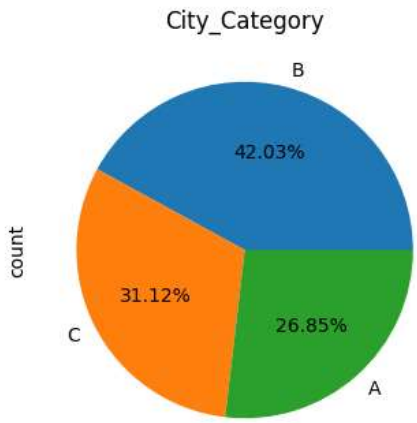
~ For continuous variable(s): Distplot, countplot, histogram for univariate analysis

In [16]:
```
plt.figure(figsize=(8, 4))
sns.histplot(data=df, x='Purchase', kde=True)
plt.show()
```

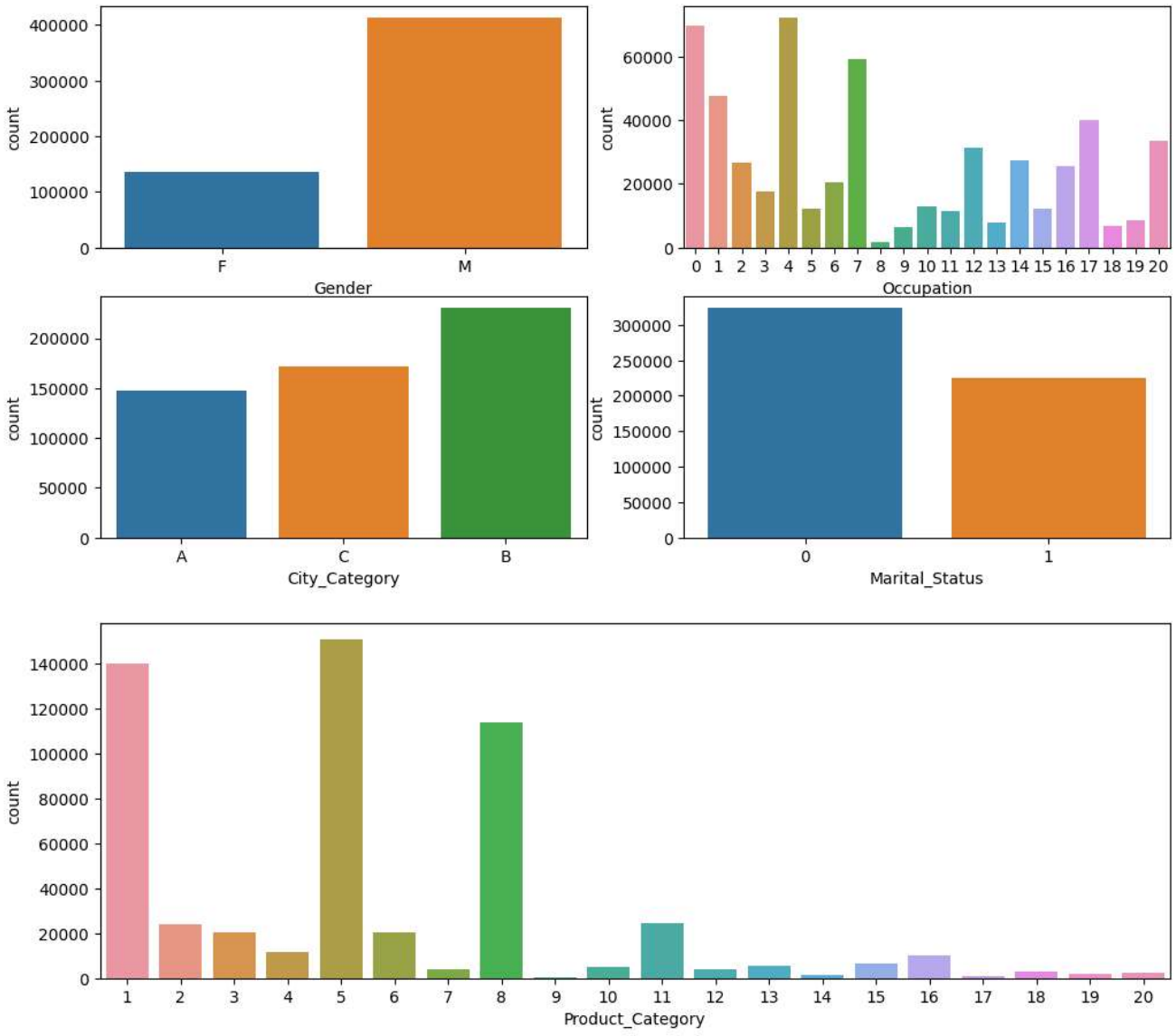


In [17]:
```
plt.figure(figsize=(6, 4))
df['City_Category'].value_counts().plot(kind='pie',autopct="%.2f%%")
plt.title("City_Category")
```

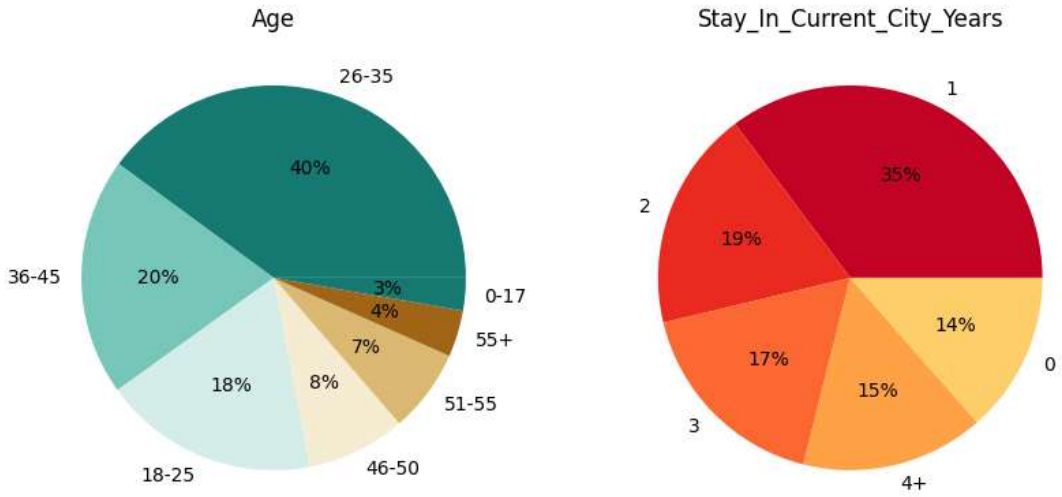
Out[17]: Text(0.5, 1.0, 'City_Category')



In [18]:
```
categorical_cols = ['Gender', 'Occupation','City_Category','Marital_Status','Product_Category']

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(12, 6))
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()

plt.figure(figsize=(12, 4))
sns.countplot(data=df, x='Product_Category')
plt.show()
```



In [23]:
```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 6))

data = df['Age'].value_counts(normalize=True)*100
palette_color = sns.color_palette('BrBG_r')
axs[0].pie(x=data.values, labels=data.index, autopct='%.0f%%', colors=palette_color)
axs[0].set_title("Age")

data = df['Stay_In_Current_City_Years'].value_counts(normalize=True)*100
palette_color = sns.color_palette('YlOrRd_r')
axs[1].pie(x=data.values, labels=data.index, autopct='%.0f%%', colors=palette_color)
axs[1].set_title("Stay_In_Current_City_Years")


plt.show()
```

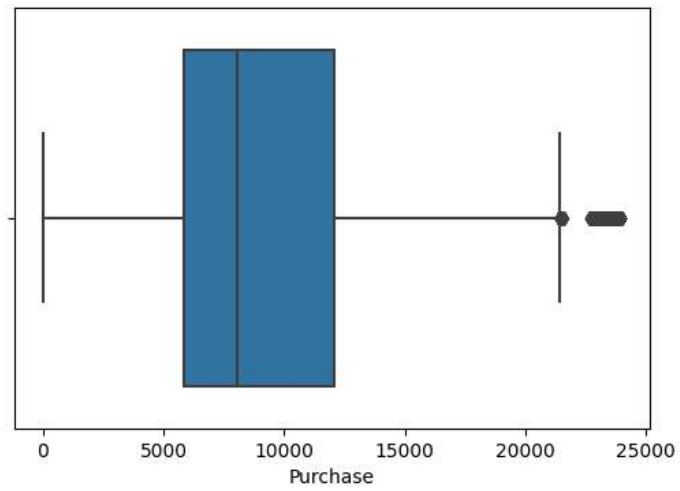



```
Observations:
Most of the users are Male
There are 20 different types of Occupation and Product_Category
More users belong to B City_Category
More users are Single as compare to Married
Product_Category - 1, 5, 8, & 11 have highest purchasing frequency.
Age group 26-35 are 40% which is highest min % of buyers are of age 55+.
35%(max customers) are living in city since 1 year.
```

In [ ]:
```

```

~ For categorical variable(s): Boxplot

In [19]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(data=df, x='Purchase')
plt.show()
```

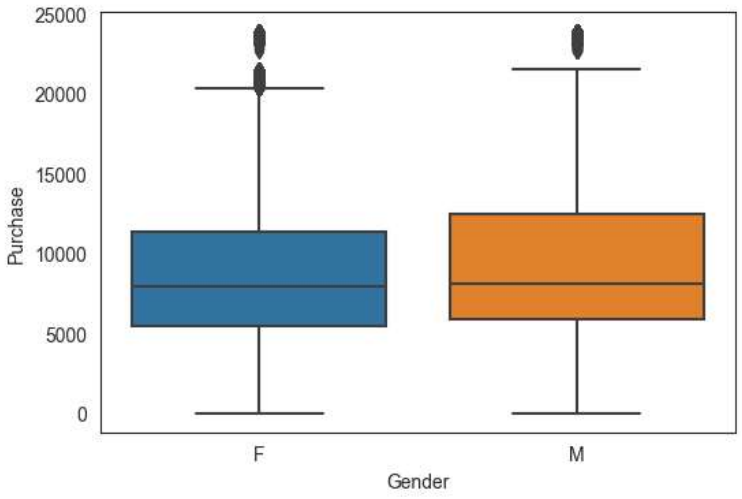



```
Observation:
```

```
2 Purchase is having outliers
```

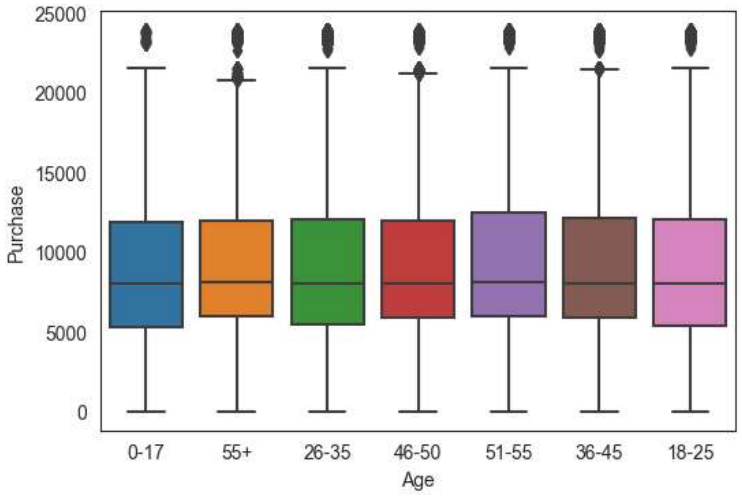
In [34]:
```
1 plt.figure(figsize=(6, 4))
2 sns.boxplot(data=df, y='Purchase', x='Gender')
```

Out[34]: <Axes: xlabel='Gender', ylabel='Purchase'>



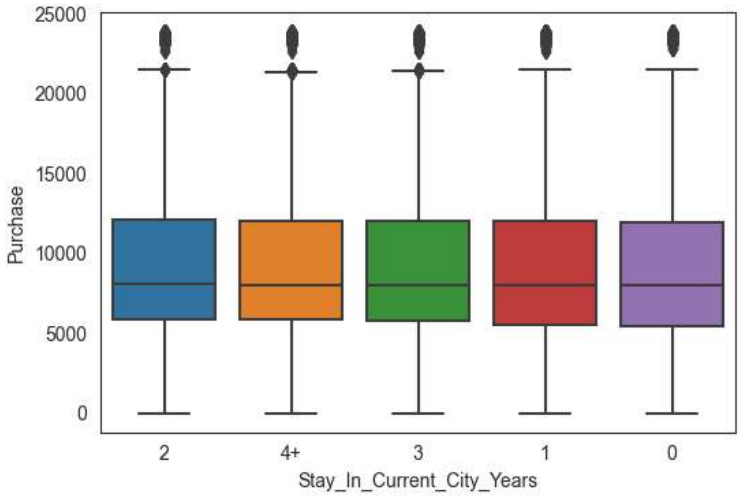
In [33]:
```
1 plt.figure(figsize=(6, 4))
2 sns.boxplot(data=df, y='Purchase', x='Age')
```

Out[33]: <Axes: xlabel='Age', ylabel='Purchase'>



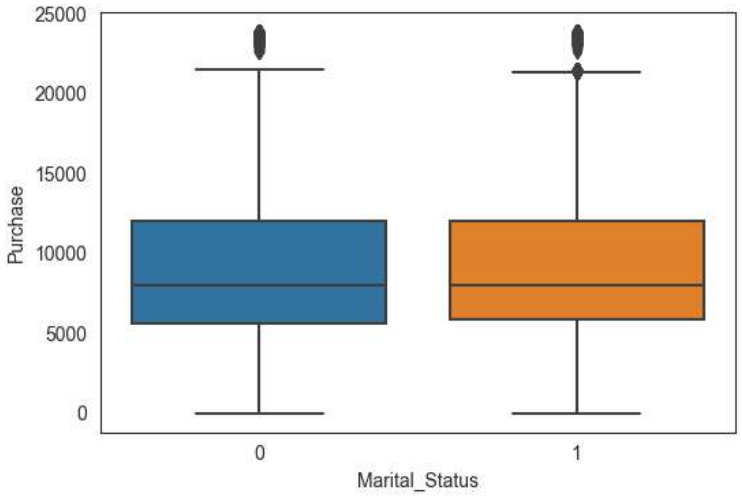
In [29]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(data=df, y='Purchase', x='Stay_In_Current_City_Years')
```

Out[29]: <Axes: xlabel='Stay_In_Current_City_Years', ylabel='Purchase'>



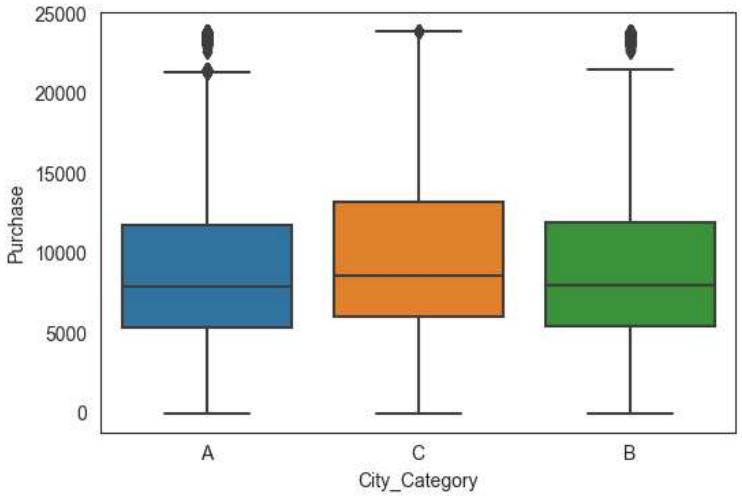
In [30]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(data=df, y='Purchase', x='Marital_Status')
```

Out[30]: <Axes: xlabel='Marital_Status', ylabel='Purchase'>



In [31]:
```
plt.figure(figsize=(6, 4))
sns.boxplot(data=df, y='Purchase', x='City_Category')
```

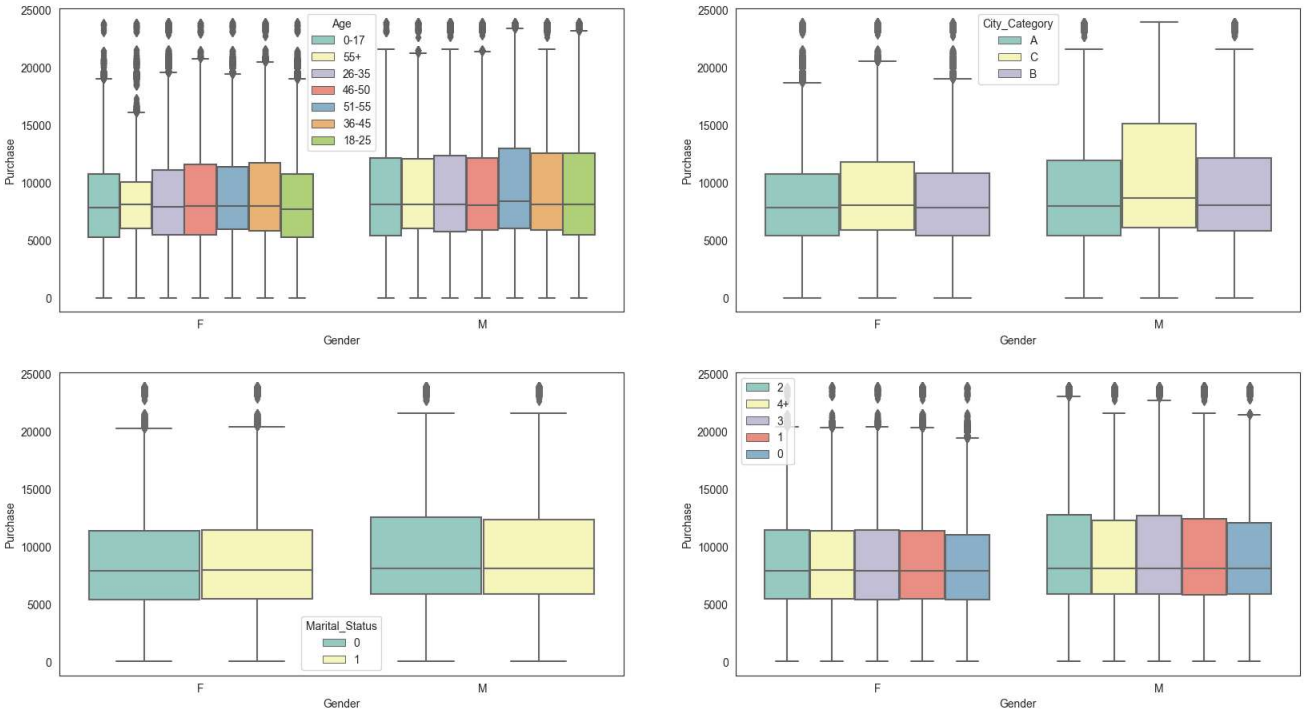
Out[31]: <Axes: xlabel='City_Category', ylabel='Purchase'>



In [ ]:
```

```

```

```

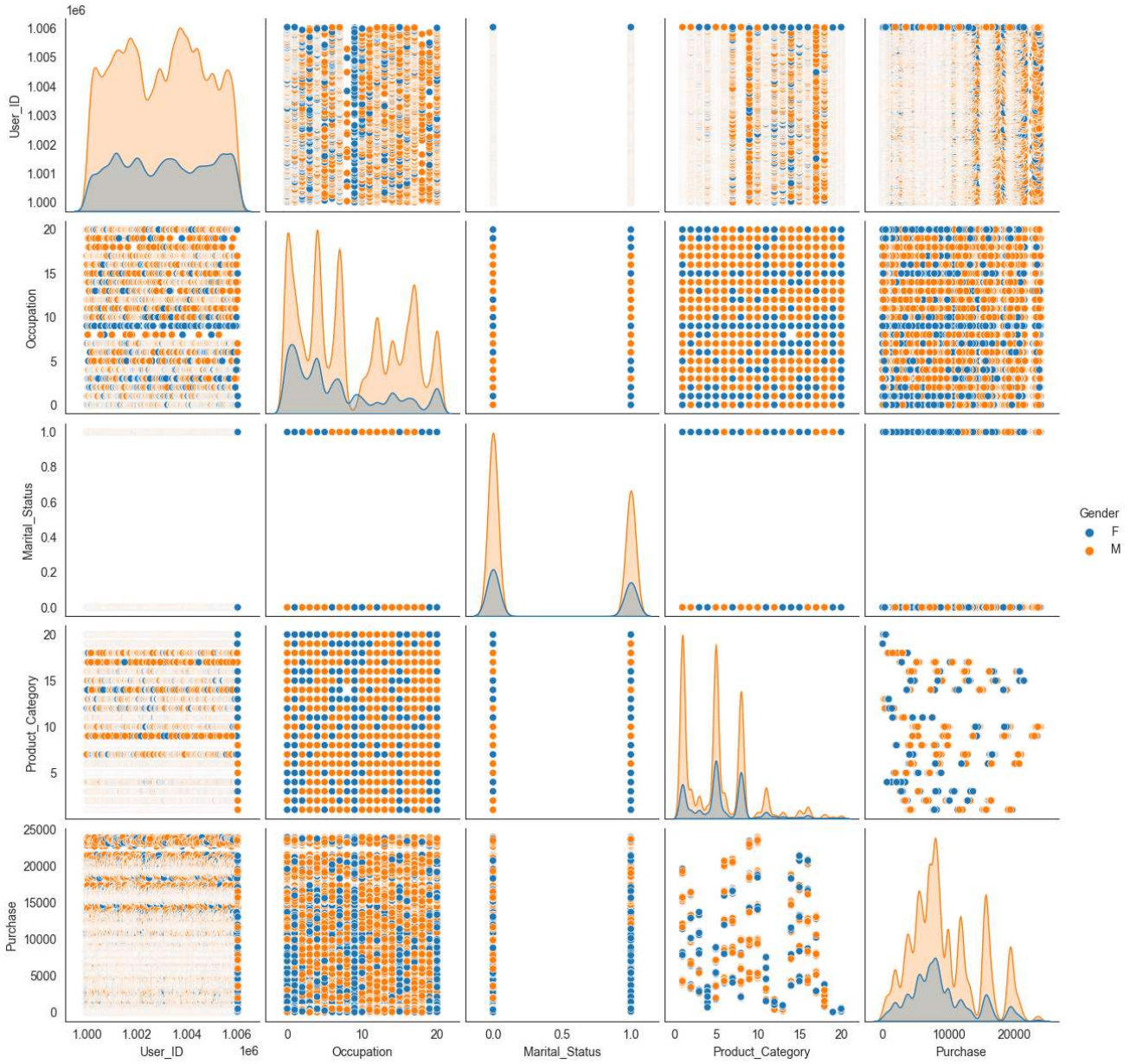
In [35]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(20, 6))
fig.subplots_adjust(top=1.5)
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age', palette='Set3', ax=axs[0,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category', palette='Set3', ax=axs[0,1])

sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status', palette='Set3', ax=axs[1,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Stay_In_Current_City_Years', palette='Set3', ax=axs[1,1])
axs[1,1].legend(loc='upper left')

plt.show()
```



~For correlation: Heatmaps, Pairplots

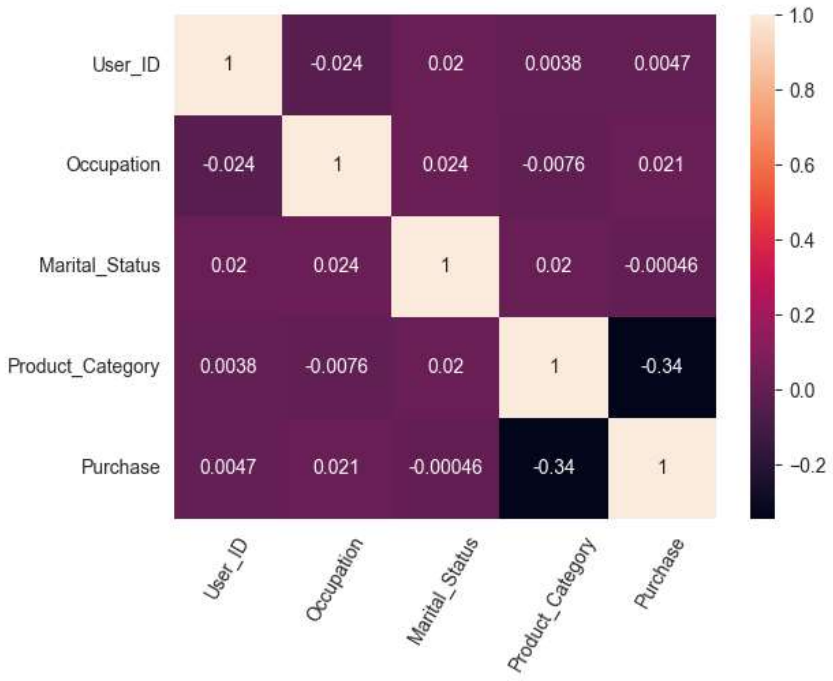
In [47]:
```
1 sns.pairplot(data=df,hue="Gender")
```

Out[47]:
```
<seaborn.axisgrid.PairGrid at 0x21b8173e7d0>
```



In [40]:
```
data=pd.read_csv("walmart.txt")
data.drop(["Stay_In_Current_City_Years","City_Category","Age","Gender","Product_ID"],axis=1,inplace=True)
plt.xticks(rotation=60)
sns.heatmap(data.corr(), annot=True)
```

Out[40]: <Axes: >



In [ ]:
```
1
```

2. Missing Value & Outlier Detection

In [41]:
```
df.isnull().sum().sort_values(ascending=False)        # No. of null values in each series
```

Out[41]:
```
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
```

```
    There is no missing value prestent in the given dataset
```

```
By analyzing the plots in part 1.3 we can infer that there are outlierts present in the given data base on purchasing
habits of the customers.
```

In [ ]:
```
1
```

3. Business Insights based on Non- Graphical and Visual Analysis

```
The given data consists of 550068 data points across 10 fields. Its a sample data.
There are no missing/null values.
Usre_id and Purchase are int64 type rest are converted to object type.
~80% of the users are between the age 18-50 (40%: 26-35, 18%: 18-25, 20%: 36-45) 75.
31% of the users are Male and 24.69% are Female
59.03% are Single, 40.97% are Married
35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years
Total of 20 product categories are there.
There are 20 differnent types of occupations in the city

Most of the users are Male
```

```
There are 20 different types of Occupation and 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.
Age group 26-35 are 40% which is highest min % of buyers are of age 55+.
35%(max customers) are living in city since 1 year.

There is no missing value prestent in the given dataset
By analyzing the plots in part 1.3 we can infer that there are outlierts present in the given data base on purchasing habits of the customers.
```

In [ ]:
```
```

4. Answering questions

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

In [42]:
```
amt_df = df.groupby(['User_ID', 'Gender'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

Out[42]:

	User_ID	Gender	Purchase
0	1000001	F	334093
1	1000002	M	810472
2	1000003	M	341635
3	1000004	M	206468
4	1000005	M	821001
...	...	...	...
5886	1006036	F	4116058
5887	1006037	F	1119538
5888	1006038	F	90034
5889	1006039	F	590319
5890	1006040	M	1653299

5891 rows × 3 columns

In [44]:
```
# Gender wise value counts in amt_df
amt_df['Gender'].value_counts()
```

Out[44]:
```
Gender
M    4225
F    1666
Name: count, dtype: int64
```

In [45]:
```
male_avg = amt_df[amt_df['Gender']=='M']['Purchase'].mean()
female_avg = amt_df[amt_df['Gender']=='F']['Purchase'].mean()

print("Average amount spend by Male customers: {:.2f}".format(male_avg))
print("Average amount spend by Female customers: {:.2f}".format(female_avg))
```

```
Average amount spend by Male customers: 925344.40
Average amount spend by Female customers: 712024.39
```

```
Observation:
Male customers spend more money than female customers
```

In [ ]:
```
```

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

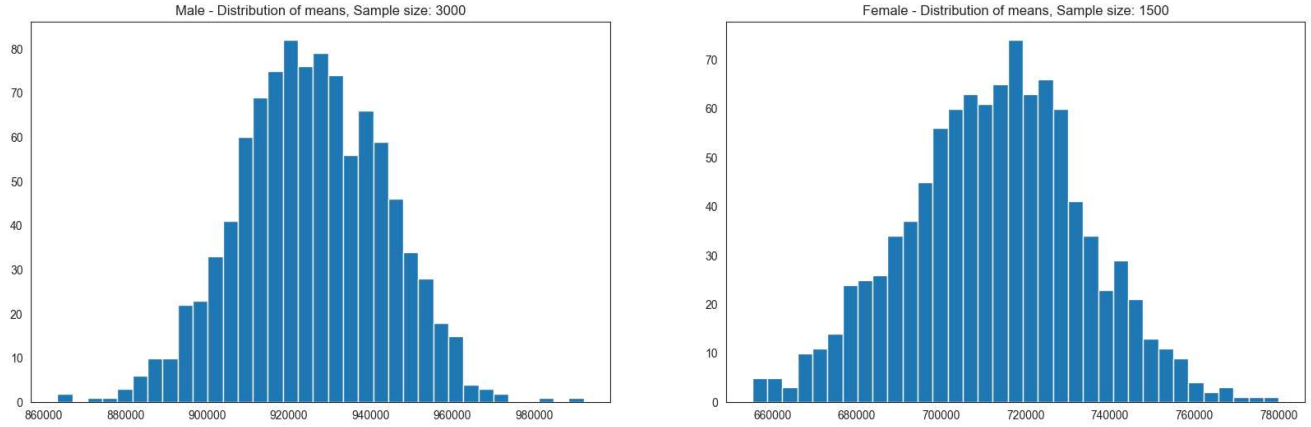
In [49]:
```
alpha=0.05                  # for 95% confidence interval
cl=1-alpha/2
z=norm.ppf(cl)
print(z)
```

```
1.959963984540054
```

In [51]:
```
male_df = amt_df[amt_df['Gender']=='M']
female_df = amt_df[amt_df['Gender']=='F']
```

In [55]:
```
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'].mean()
    female_mean = female_df.sample(female_sample_size, replace=True)['Purchase'].mean()
    male_means.append(male_mean)
    female_means.append(female_mean)
```

In [65]:
```
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: 3000")
axis[1].set_title("Female - Distribution of means, Sample size: 1500")

plt.show()
```



In [56]:
```
print("Population mean - Mean of sample means of amount spend for Male: {:.2f}".format(np.mean(male_means)))
print("Population mean - Mean of sample means of amount spend for Female: {:.2f}".format(np.mean(female_means)))

print("\nMale - Sample mean: {:.2f} Sample std: {:.2f}".format(male_df['Purchase'].mean(), male_df['Purchase'].std()))
print("Female - Sample mean: {:.2f} Sample std: {:.2f}".format(female_df['Purchase'].mean(), female_df['Purchase'].std()))
```

```
Population mean - Mean of sample means of amount spend for Male: 925398.21
Population mean - Mean of sample means of amount spend for Female: 712567.89

Male - Sample mean: 925344.40 Sample std: 985830.10
Female - Sample mean: 712024.39 Sample std: 807370.73
```

Observation

Now using the Central Limit Theorem for the population we can say that:

Average amount spend by male customers is 9,26,341.86 Average amount spend by female customers is 7,11,704.09

In [57]:
```
male_margin_of_error_clt = z*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 = z*female_df['Purchase'].std()/np.sqrt(len(female_df))
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

print("Male confidence interval of means: ({:.2f}, {:.2f})".format(male_lower_lim, male_upper_lim))
print("Female confidence interval of means: ({:.2f}, {:.2f})".format(female_lower_lim, female_upper_lim))
```

```
Male confidence interval of means: (895618.38, 955070.43)
Female confidence interval of means: (673255.48, 750793.30)
```

```
    Now we can infer about the population that, 95% of the times:

    Average amount spend by male customer will lie in between: (895617.83, 955070.97)
    Average amount spend by female customer will lie in between: (673254.77, 750794.02)
```

In [ ]:
```
```

4.3: Are confidence intervals of average male and female spending overlapping? How can Walmart leverage this conclusion to make changes or improvements?

```
Confidence interval of male & female spendings are not overlapping. Males spend more than females.
In light of the fact that females spend less than males on average, management needs to focus on their specific needs
differently. Adding some additional offers for women can increase their spending on Black Friday.
```

In [ ]:
```
```

4.4: Results when the same activity is performed for Married vs Unmarried

In [58]:
```
amt_df
amt_df = df.groupby(['User_ID', 'Marital_Status'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

Out[58]:

	User_ID	Marital_Status	Purchase
0	1000001	0	334093
1	1000002	0	810472
2	1000003	0	341635
3	1000004	1	206468
4	1000005	1	821001
...	...	...	...
5886	1006036	1	4116058
5887	1006037	0	1119538
5888	1006038	0	90034
5889	1006039	1	590319
5890	1006040	0	1653299

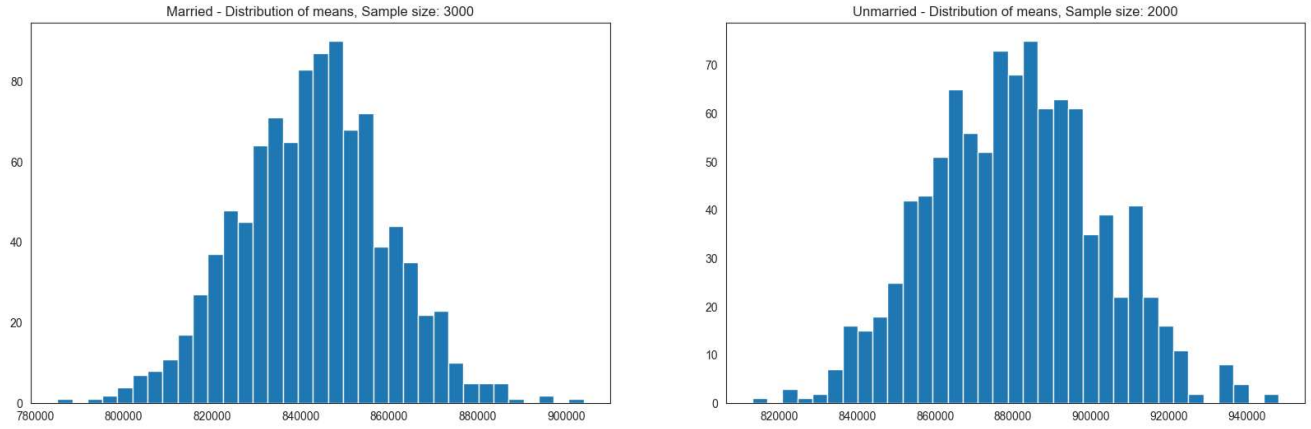
5891 rows × 3 columns

In [59]:
```
amt_df['Marital_Status'].value_counts()
```

Out[59]:
```
Marital_Status
0    3417
1    2474
Name: count, dtype: int64
```

In [60]:
```
marid_samp_size = 3000
unmarid_sample_size = 2000
num_repitions = 1000
marid_means = []
unmarid_means = []

for _ in range(num_repitions):
    marid_mean = amt_df[amt_df['Marital_Status']==1].sample(marid_samp_size, replace=True)['Purchase'].mean()
    unmarid_mean = amt_df[amt_df['Marital_Status']==0].sample(unmarid_sample_size, replace=True)['Purchase'].mean()

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

In [61]:
```
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)
axis[0].set_title("Married - Distribution of means, Sample size: 3000")
axis[1].set_title("Unmarried - Distribution of means, Sample size: 2000")

plt.show()
```



In [68]:
```
print("Population mean - Mean of sample means of amount spend for Married: {:.2f}".format(np.mean(marid_means)))
print("Population mean - Mean of sample means of amount spend for Unmarried: {:.2f}".format(np.mean(unmarid_means)))

print("\nMarried - Sample mean: {:.2f} Sample std: {:.2f}".format(amt_df[amt_df['Marital_Status']==1]['Purchase'].mean(),
                                                                  amt_df[amt_df['Marital_Status']==1]['Purchase'].std()))
print("Unmarried - Sample mean: {:.2f} Sample std: {:.2f}".format(amt_df[amt_df['Marital_Status']==0]['Purchase'].mean(),
                                                                  amt_df[amt_df['Marital_Status']==0]['Purchase'].std()))
```

```
Population mean - Mean of sample means of amount spend for Married: 842656.14
Population mean - Mean of sample means of amount spend for Unmarried: 880374.27

Married - Sample mean: 843526.80 Sample std: 935352.12
Unmarried - Sample mean: 880575.78 Sample std: 949436.25
```

```
Observation

Now using the Central Limit Theorem for the population we can say that:

Average amount spend by male customers is 9,26,341.86
Average amount spend by female customers is 7,11,704.09
```

In [69]:
```
for val in ["Married", "Unmarried"]:

    new_val = 1 if val == "Married" else 0

    new_df = amt_df[amt_df['Marital_Status']==new_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

    print("{} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
Married confidence interval of means: (806668.83, 880384.76)
Unmarried confidence interval of means: (848741.18, 912410.38)
```

In [ ]:
```

```

4.5: Results when the same activity is performed for Age

In [70]:
```
amt_df = df.groupby(['User_ID', 'Age'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

Out[70]:

	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 [71]:
```
amt_df['Age'].value_counts()
```

Out[71]:
```
Age
26-35    2053
36-45    1167
18-25    1069
46-50     531
51-55     481
55+       372
0-17      218
Name: count, dtype: int64
```

In [72]:
```
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 = amt_df[amt_df['Age']==age_interval].sample(sample_size, replace=True)['Purchase'].mean()
        all_means[age_interval].append(mean)
```

In [73]:
```
for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = amt_df[amt_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

    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)
```

In [ ]:
```

```

5. Final Insights - Illustrate the insights based on exploration and CLT

```
Average amount spend by Male customers: 925344.40
Average amount spend by Female customers: 712024.39
```

```
Confidence Interval by Gender
Now using the Central Limit Theorem for the population:

Average amount spend by male customers is 9,26,341.86
Average amount spend by female customers is 7,11,704.09
Now we can infer about the population that, 95% of the times:

Average amount spend by male customer will lie in between: (895617.83, 955070.97)
Average amount spend by female customer will lie in between: (673254.77, 750794.02)
Confidence Interval by Marital_Status
Married confidence interval of means: (806668.83, 880384.76)
Unmarried confidence interval of means: (848741.18, 912410.38)
Confidence Interval by Age
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)
```

In [ ]:

```
```

6. Recommendations

```
Men spent more money than women, So company should focus on retaining the male customers and getting more male customers.
Product_Category - 1, 5, 8, & 11 have highest purchasing frequency. it means these are the products in these categories are liked more by customers. Company can focus on selling more of these products or selling more of the products which are purchased less.
Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
Customers in the age 18-45 spend more money than the others, So company should focus on acquisition of customers who are in the age 18-45
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_Category C will help the company increase the revenue.
```