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).

Dataset

The company collected the transactional data of customers who purchased products from the Walmart Stores during Black Friday. The dataset has the following features: Dataset link: Walmart_data.csv

- User_ID: User ID
- Product_ID: Product ID
- Gender: Sex of User
- Age: Age in bins
- Occupation: Occupation(Masked)
- City_Category: Category of the City (A,B,C)
- StayInCurrentCityYears: Number of years stay in current city
- Marital_Status: Marital Status
- ProductCategory: Product Category (Masked)
- Purchase: Purchase Amount

In [36]:

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

In [4]:

```
df = pd.read_csv('walmart_data.txt')
```

In [5]:

```
df
```

Out[5]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Ye
0	1000001	P00069042	F	0-17	10	A	
1	1000001	P00248942	F	0-17	10	A	
2	1000001	P00087842	F	0-17	10	A	
3	1000001	P00085442	F	0-17	10	A	
4	1000002	P00285442	M	55+	16	C	
...	...	...	...	...	...	...	
550063	1006033	P00372445	M	51-55	13	B	
550064	1006035	P00375436	F	26-35	1	C	
550065	1006036	P00375436	F	26-35	15	B	
550066	1006038	P00375436	F	55+	1	C	
550067	1006039	P00371644	F	46-50	0	B	

550068 rows × 10 columns

In [14]:

```
# Checking structure & D-types of our columns in data

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 [15]:

```
df.shape
```

Out[15]:

```
(550068, 10)
```

Our data contains 550k records & 10 columns

In [13]:

```
# Null-values check point

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

Out[13]:

```
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 are no null values in our data

In [10]:

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

Out[10]:

	Product_ID	Gender	Age	City_Category	Stay_In_Current_City_Years
count	550068	550068	550068	550068	550068
unique	3631	2	7	3	5
top	P00265242	M	26-35	B	1
freq	1880	414259	219587	231173	193821

In [11]:

```
df.describe()
```

Out[11]:

	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 [18]:

```
print(f'Number of Users in the data : {df.User_ID.nunique()}')
print(f'Number of Products in the data : {df.Product_ID.nunique()}')
```

```
Number of Users in the data : 5891
Number of Products in the data : 3631
```

Distribution & Unique values in categorical columns of our data

In [32]:

```
df['Gender'].value_counts()/len(df)
```

Out[32]:

```
M    0.753105
F    0.246895
Name: Gender, dtype: float64
```

Our data has 75% Male & 25% Female

In [34]:

```
df['Age'].value_counts()/len(df)*100
```

Out[34]:

```
26-35    39.919974
36-45    19.999891
18-25    18.117760
46-50     8.308246
51-55     6.999316
55+       3.909335
0-17      2.745479
Name: Age, dtype: float64
```

Majority of our data are from 26-35 Age.

Approx. 80% our data are >18 age.

In [30]:

```
round(df['City_Category'].value_counts()/len(df)*100, 2)
```

Out[30]:

```
B    42.03
C    31.12
A    26.85
Name: City_Category, dtype: float64
```

In [31]:

```
round(df['Stay_In_Current_City_Years'].value_counts()/len(df)*100, 2)
```

Out[31]:

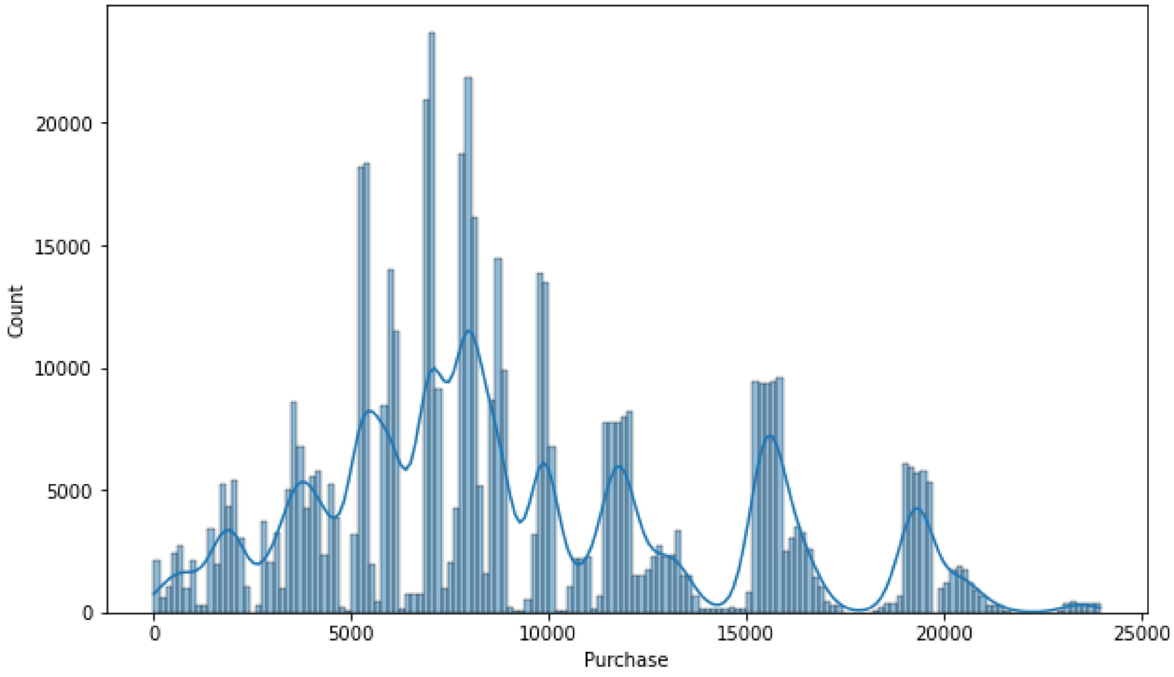
```
1     35.24
2     18.51
3     17.32
4+    15.40
0     13.53
Name: Stay_In_Current_City_Years, dtype: float64
```

Uni-variate Analysis

Understanding the distribution of data and detecting abornmal data points for continuous variables¶

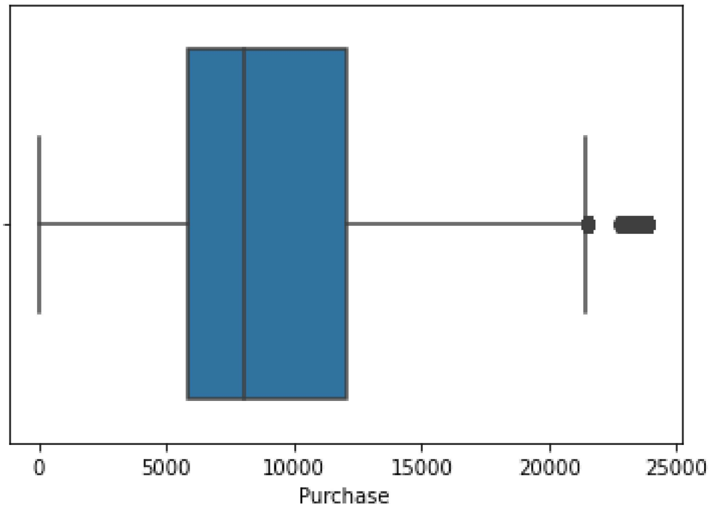
In [37]:

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



In [38]:

```
sns.boxplot(data=df, x='Purchase', orient='h')
plt.show()
```



As we can see here from the plot, Purchase column is having outliers above 20k

As we can see here from the plot, Purchase column is having outliers above 20k

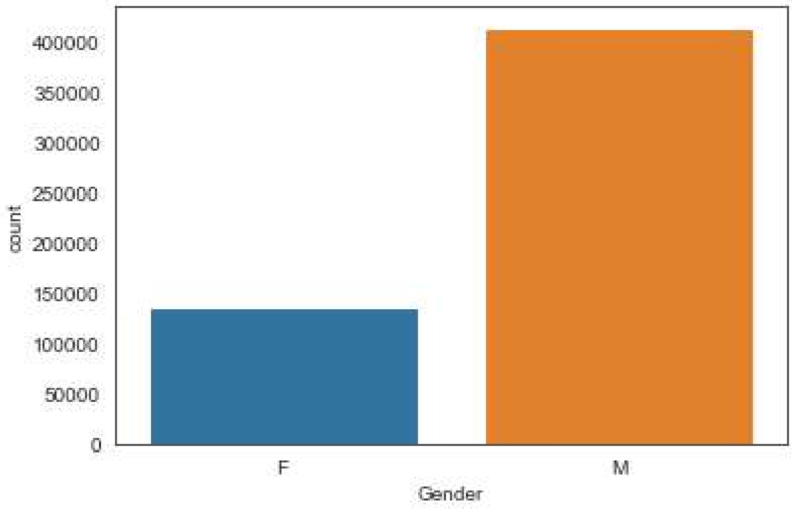
Understanding Distribution for Categorical columns

In [60]:

```
sns.countplot(data=df, x='Gender')
```

Out[60]:

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



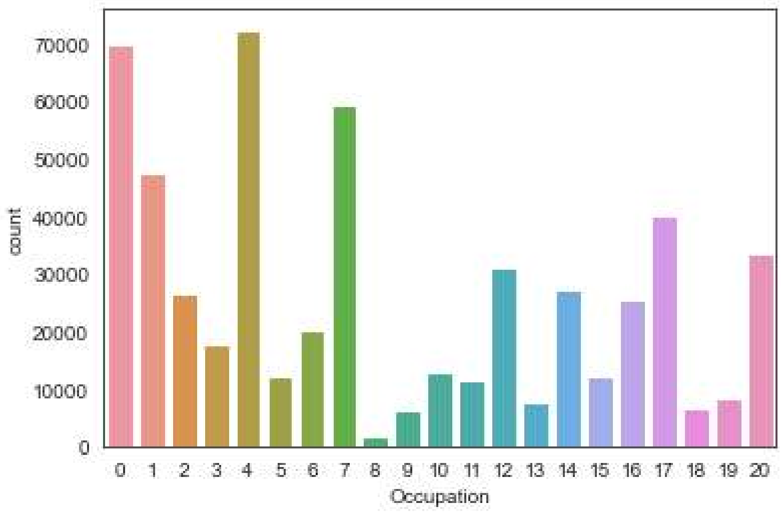
Our data has most of the users as Male.

In [61]:

```
sns.countplot(data=df, x='Occupation')
```

Out[61]:

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



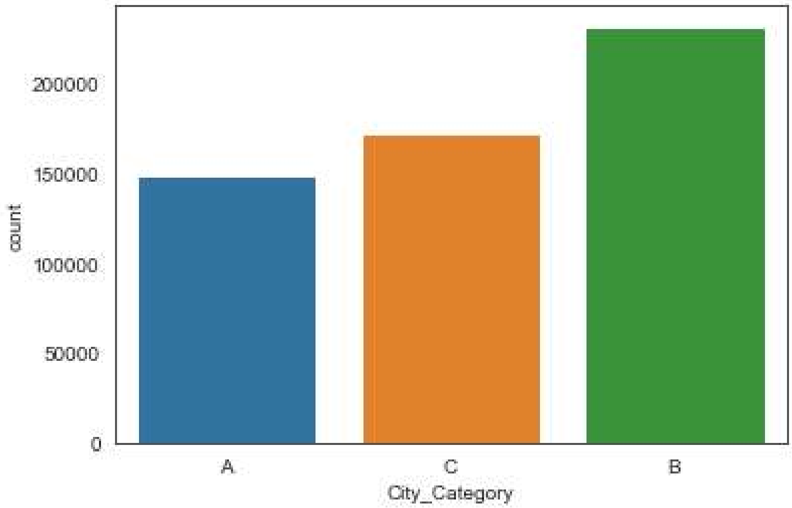
There are 20 different types of Occupation and Product_Category.

In [62]:

```
sns.countplot(data=df, x='City_Category')
```

Out[62]:

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



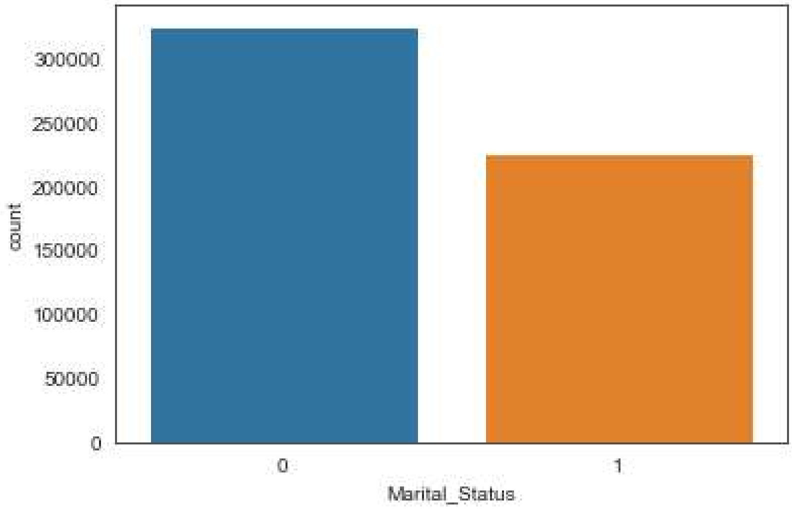
More users belong to B City_Category

In [63]:

```
sns.countplot(data=df, x='Marital_Status')
```

Out[63]:

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



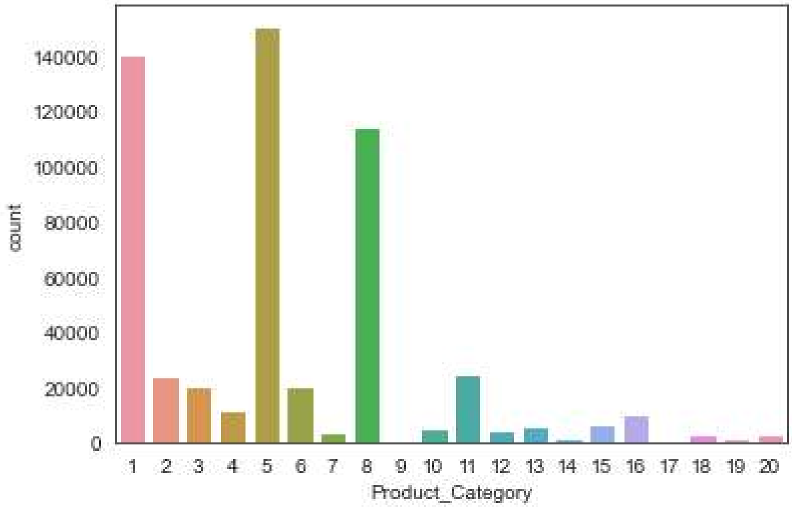
More users are Single as compared to Married

In [64]:

```
sns.countplot(data=df, x='Product_Category')
```

Out[64]:

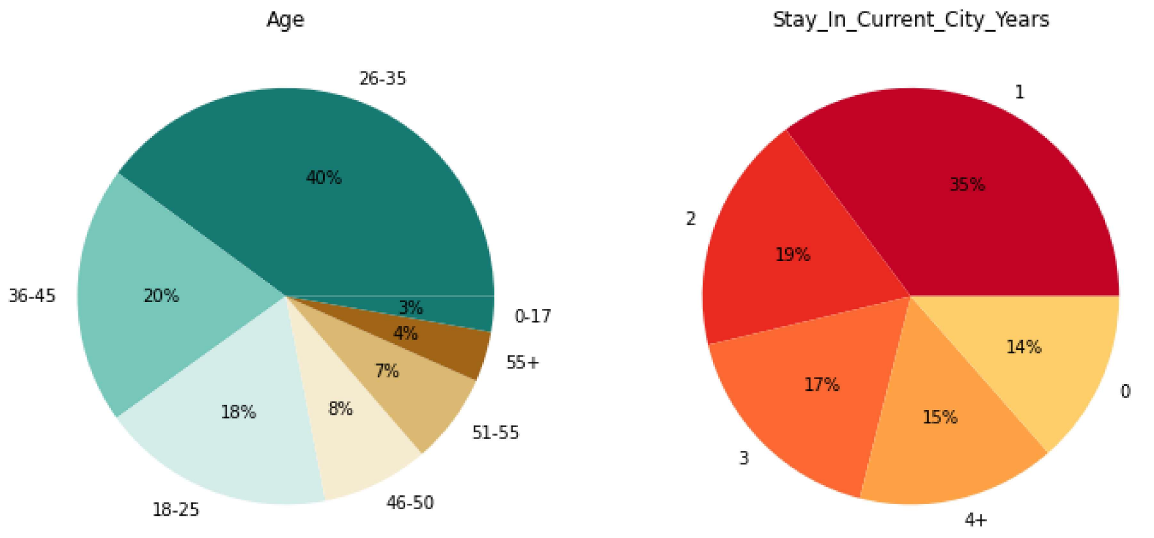
```
<AxesSubplot:xlabel='Product_Category', ylabel='count'>
```



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

In [40]:

```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(12, 8))

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()
```



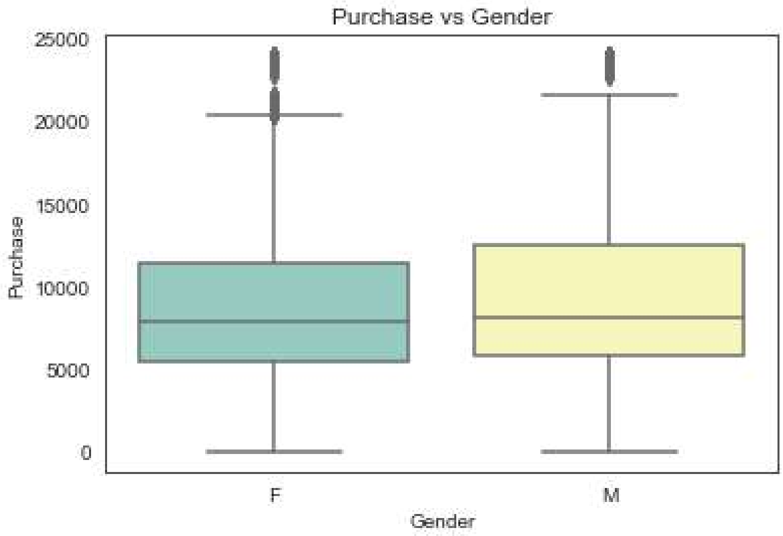
Bi-variate Analysis

In [75]:

```
def box_plot_(col):
    sns.set_style("white")
    sns.boxplot(data=df, y='Purchase', x=col, palette='Set3').set_title(f"Purchase vs {col}
```

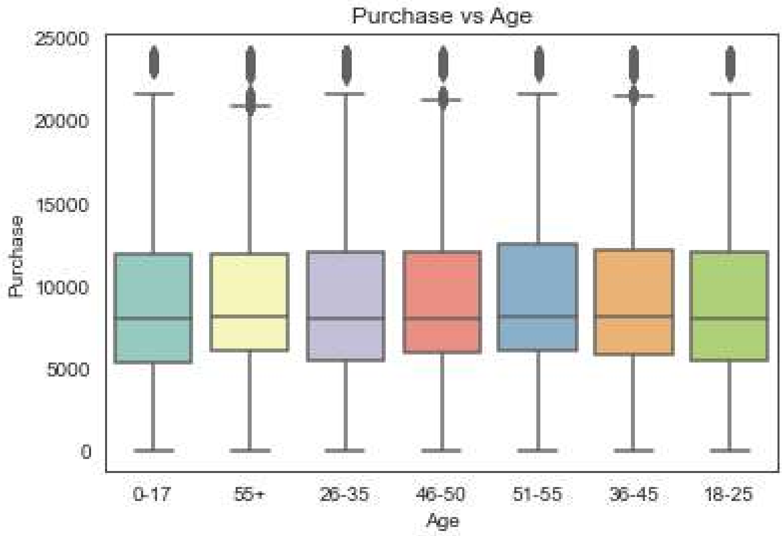
In [76]:

```
box_plot_('Gender')
```



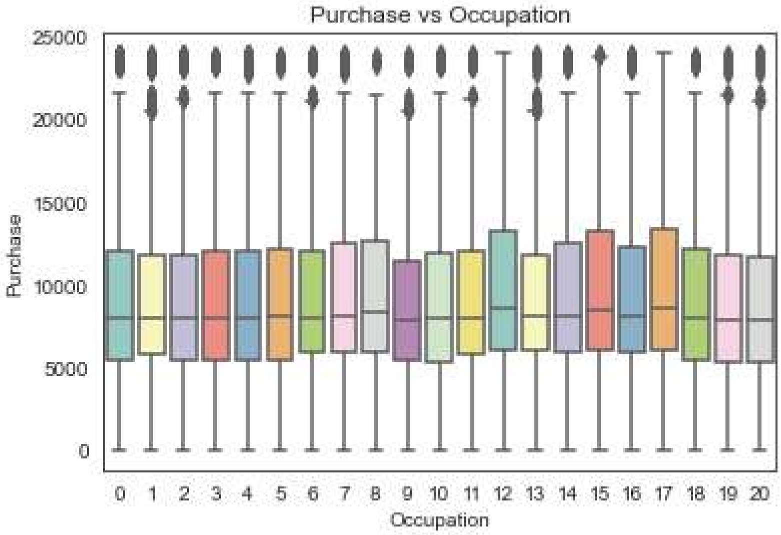
In [77]:

```
box_plot_('Age')
```



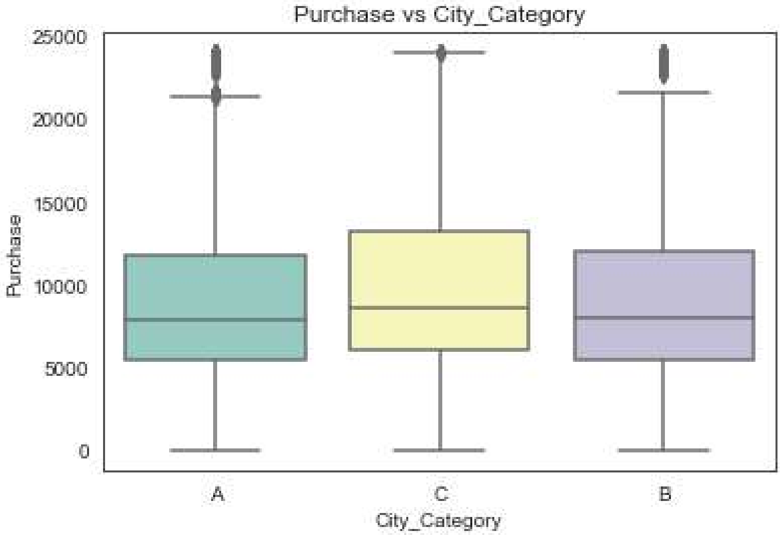
In [78]:

```
box_plot_('Occupation')
```



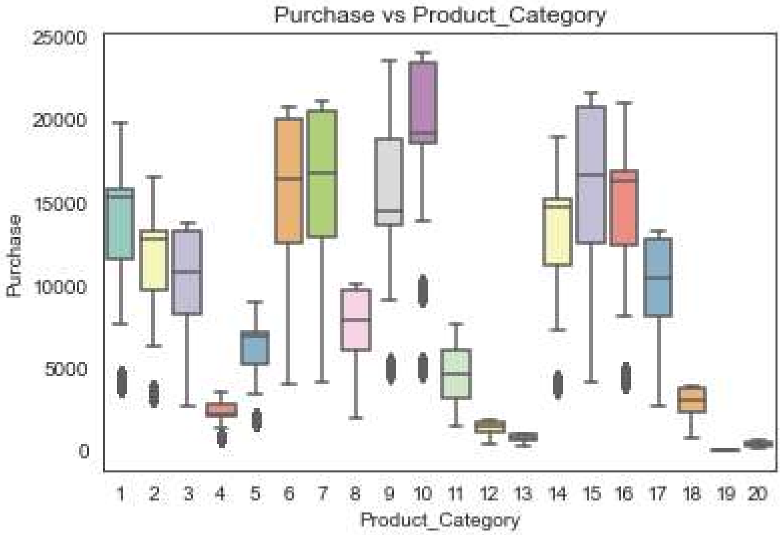
In [79]:

```
box_plot_('City_Category')
```



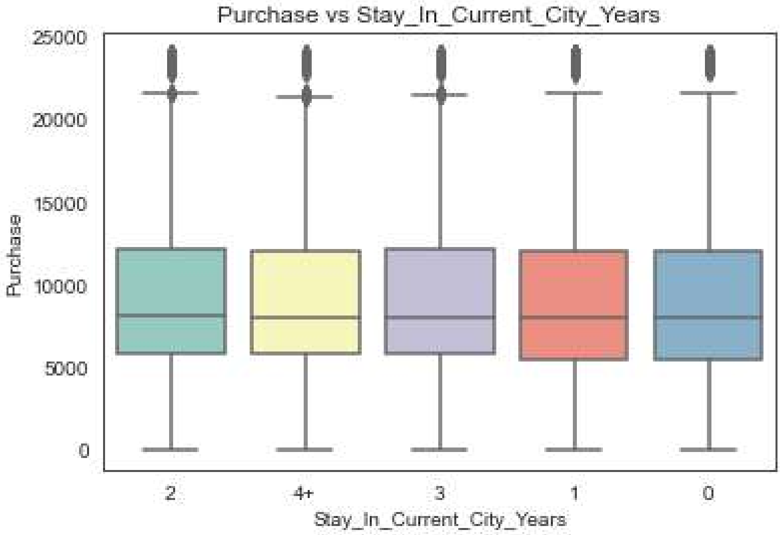
In [80]:

```
box_plot_('Product_Category')
```



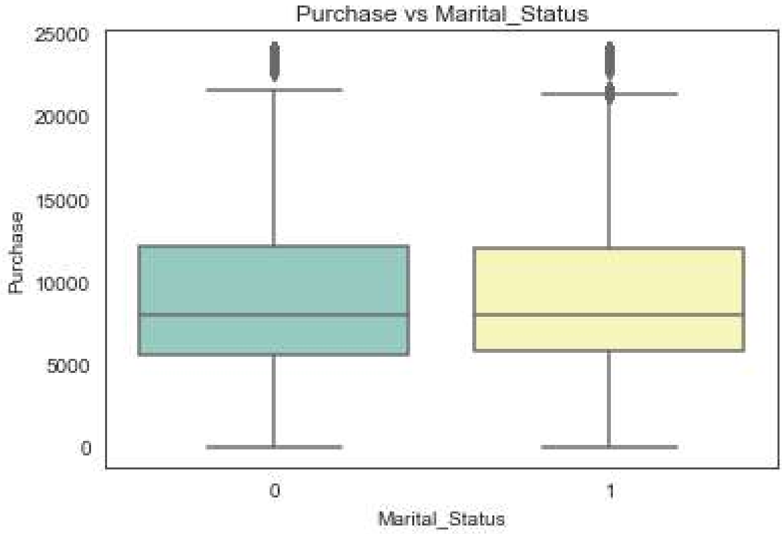
In [81]:

```
box_plot_('Stay_In_Current_City_Years')
```



In [82]:

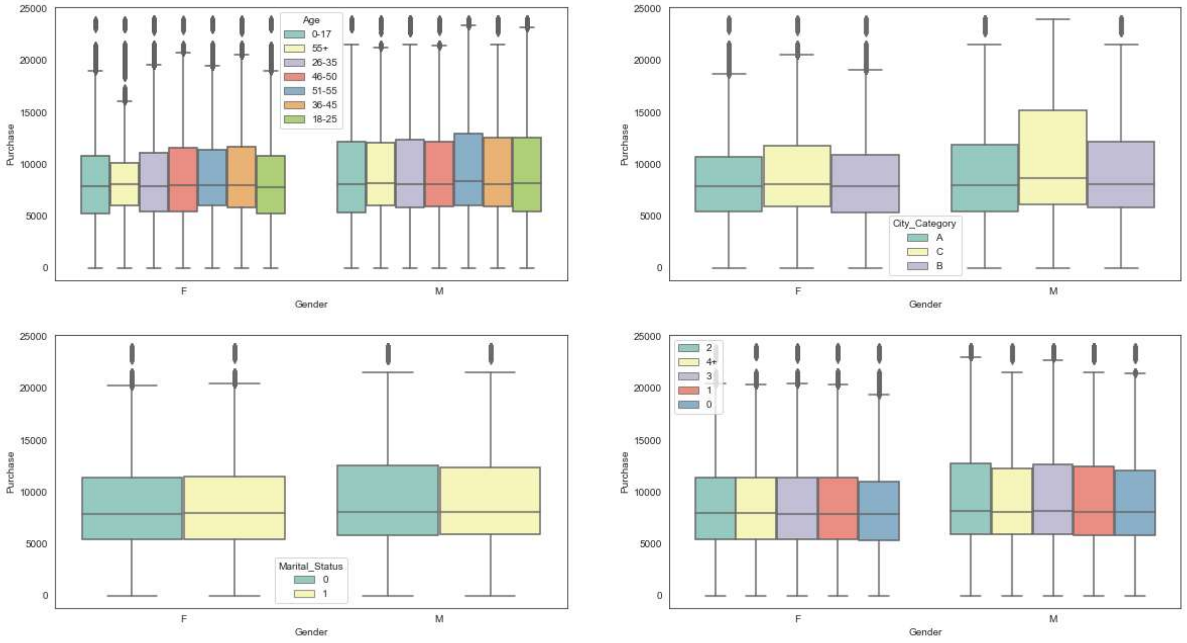
```
box_plot_('Marital_Status')
```



Multivariate Analysis

In [44]:

```
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[

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

plt.show()
```



Average amount spend per customer for Male and Female

In [47]:

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

Out[47]:

	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 [48]:

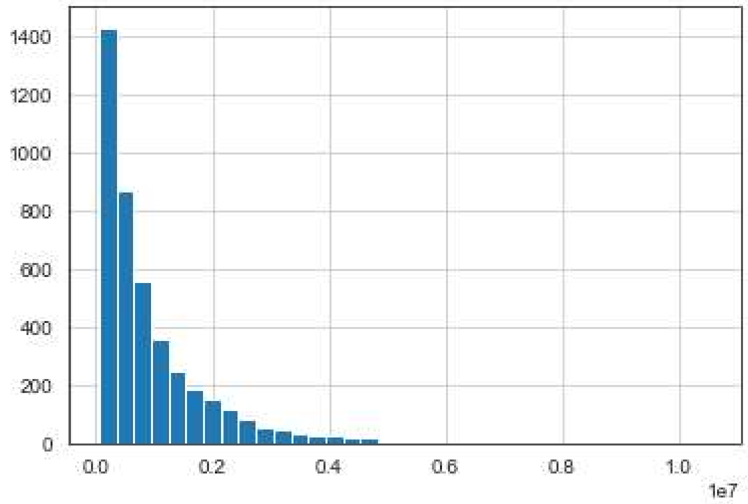
```
# Gender wise value counts in pur_df
pur_df['Gender'].value_counts()
```

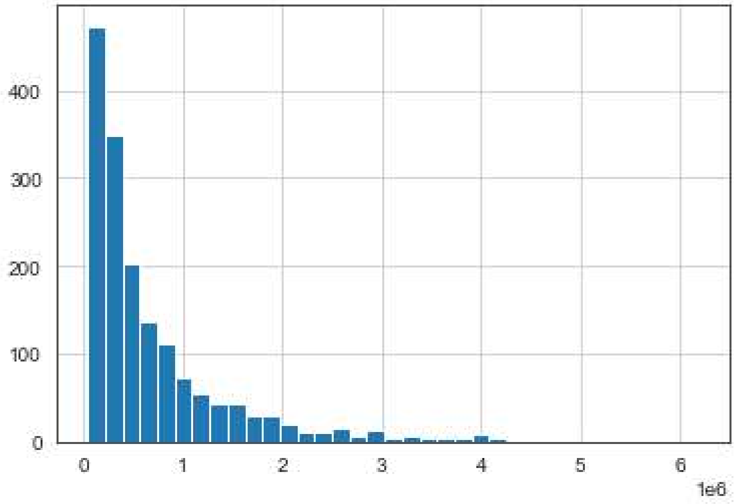
Out[48]:

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

In [50]:

```
# histogram of average amount spend for each customer - Male & Female
pur_df[pur_df['Gender']=='M']['Purchase'].hist(bins=35)
plt.show()

pur_df[pur_df['Gender']=='F']['Purchase'].hist(bins=35)
plt.show()
```





In [51]:

```
male_avg = pur_df[pur_df['Gender']=='M']['Purchase'].mean()
female_avg = pur_df[pur_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
```

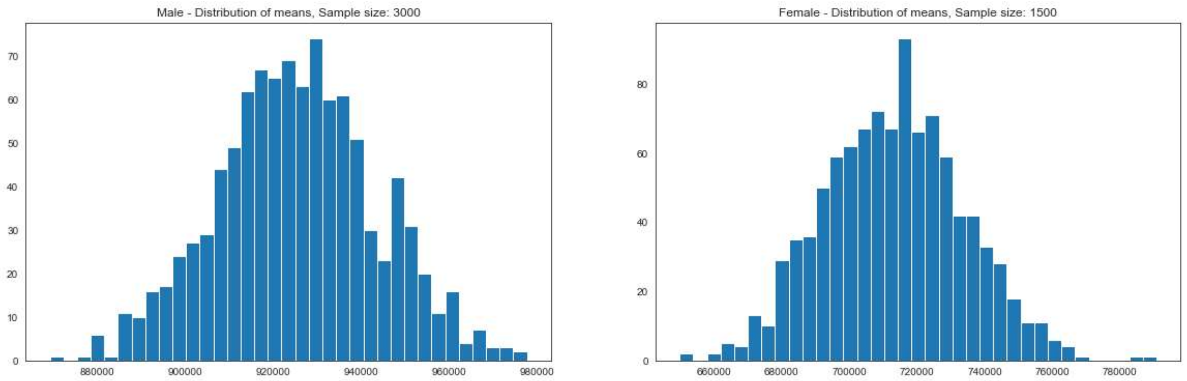
Male customers spend more money than female customers

In [53]:

```
male_df = pur_df[pur_df['Gender']=='M']
female_df = pur_df[pur_df['Gender']=='F']


genders = ["M", "F"]

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

for _ 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 [54]:

```
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 [55]:

```
print("Population mean - Mean of sample means of amount spend for Male: {:.2f}".format(np.m
print("Population mean - Mean of sample means of amount spend for Female: {:.2f}".format(np

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

```
Population mean - Mean of sample means of amount spend for Male: 925262.26
Population mean - Mean of sample means of amount spend for Female: 713078.93

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

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

- Average amount spend by male customers is 925262.26
- Average amount spend by female customers is 713078.93

In [56]:

```
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(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_upp
print("Female confidence interval of means: ({:.2f}, {:.2f})".format(female_lower_lim, fema
```

```
Male confidence interval of means: (895617.83, 955070.97)
Female confidence interval of means: (673254.77, 750794.02)
```

Insights from married vs unmarried

In [57]:

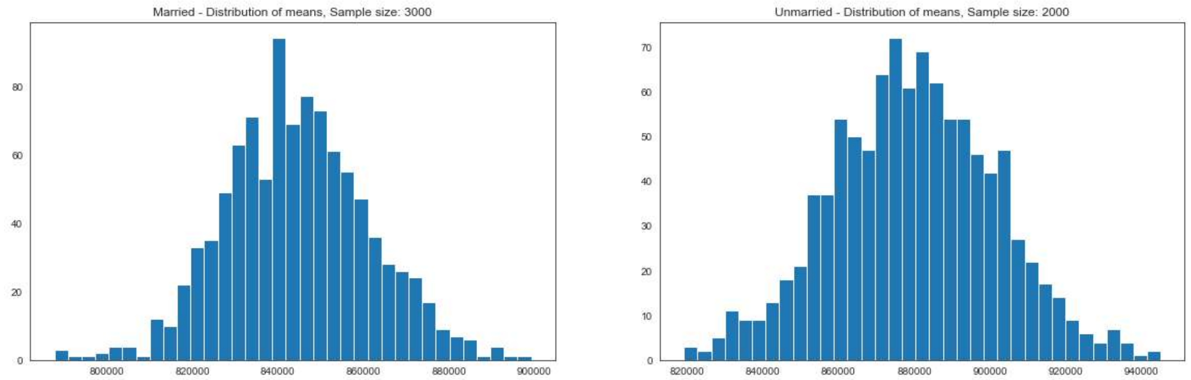
```
amt_df = df.groupby(['User_ID', 'Marital_Status'])[['Purchase']].sum()
amt_df = amt_df.reset_index()

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

Out[57]:

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

In [58]:

```
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)[
    unmarid_mean = amt_df[amt_df['Marital_Status']==0].sample(unmarid_sample_size, replace=

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


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()

print("Population mean - Mean of sample means of amount spend for Married: {:.2f}".format(n
print("Population mean - Mean of sample means of amount spend for Unmarried: {:.2f}".format

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



Married - Distribution of means, Sample size: 3000

Unmarried - Distribution of means, Sample size: 2000

```
Population mean - Mean of sample means of amount spend for Married: 844300.6
2
Population mean - Mean of sample means of amount spend for Unmarried: 87976
2.59

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

In [59]:

```
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_
```

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

Calculating the average amount spent by Age

In [85]:

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

df_age['Age'].value_counts()
```

Out[85]:

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

In [90]:

```
sample_size = 200
num_repitions = 1000

all_means = {}

age_intervals = df_age['Age'].unique()
for age_interval in age_intervals:
    all_means[age_interval] = []

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

In [91]:

```
for val in df_age['Age'].unique():

    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
```

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

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.

In [ ]: