In [123]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import binom, norm
```

Problem Statement:

The problem is to analyze the sales data of Walmart stores across different regions in the United States and identify the key factors that affect sales. The objective is to provide insights and recommendations to improve the sales performance of the stores.

Basic Metrics:

To analyze the basic metrics for the given dataframe and to address the business problem of identifying customer purchase behavior against the customer's gender and the various other factors to help the business make better decisions. ¶

1. Identify the overall size of the data frame by using the shape attribute of pandas.
2. Analyze the data types and missing values in the data frame using info() method.
3. Identify the basic statistical measures like mean, median, standard deviation, minimum, maximum values for the numerical columns using describe() method.
4. Analyze the distribution of the numerical columns using histograms and boxplots.
5. Check if features like marital status, age have any effect on the product purchased.
6. Answer questions with Confidence Intervals and use Central Limit theorem.
7. Use the sample average to find out an interval within which the population average will lie.
8. Inference after computing the average female and male expenses.
9. Some recommendations and actionable insights, based on the inferences.

In [3]:
```
df=pd.read_csv("/Users/praneetcb/Documents/walmart_data.csv")
```

In [13]:
```
df.head()
```

Out[13]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Yea
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	4

In [10]:
```
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 [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 [12]:
```
df.nunique()
```

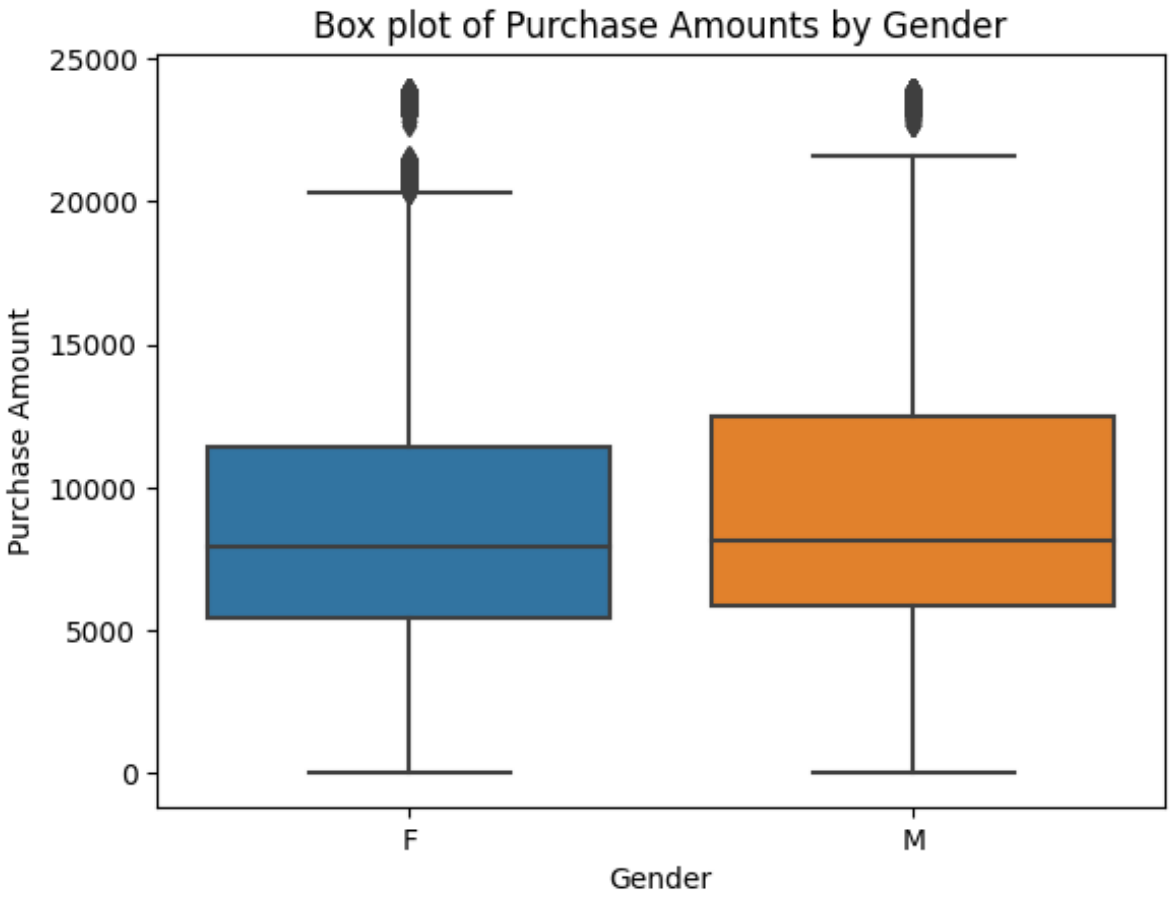
Out[12]:
```
User_ID                       5891
Product_ID                    3631
Gender                           2
Age                              7
Occupation                      21
City_Category                    3
Stay_In_Current_City_Years       5
Marital_Status                   2
Product_Category                20
Purchase                     18105
dtype: int64
```

In [ ]:
```
# Number of male and female in the dataframe
```

In [14]:
```
df['Gender'].value_counts()
```

Out[14]:
```
Gender
M    414259
F    135809
Name: count, dtype: int64
```

In [503]:
```
sns.boxplot(x='Gender', y='Purchase', data=df)
plt.xlabel('Gender')
plt.ylabel('Purchase Amount')
plt.title('Box plot of Purchase Amounts by Gender')
plt.show()
```



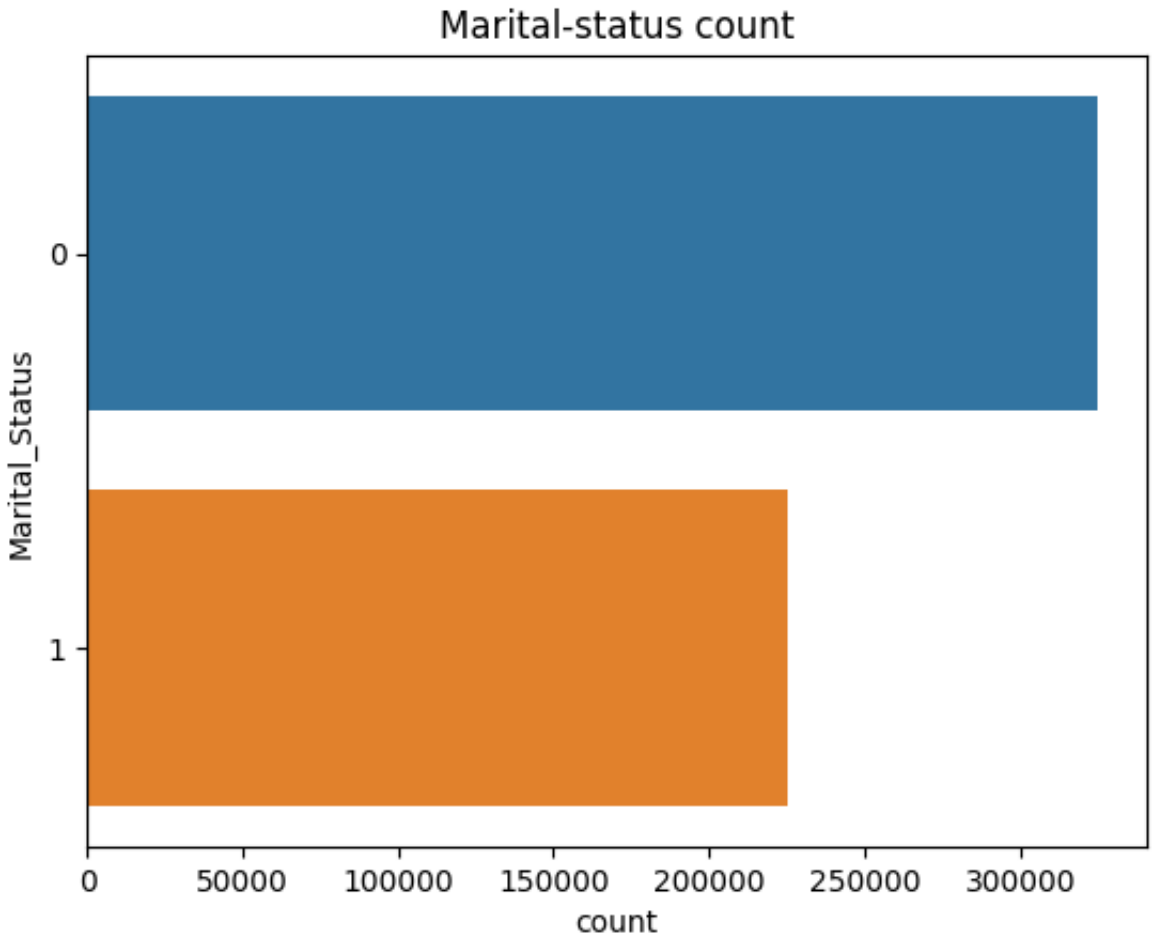
In [15]:
```
df['Age'].value_counts() #Age bracket 26-35 has the highest spender
```

Out[15]:
```
Age
26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
Name: count, dtype: int64
```

In [16]:
```
# Number of Single and Married in the dataframe
df['Marital_Status'].value_counts()
```

Out[16]:
```
Marital_Status
0    324731
1    225337
Name: count, dtype: int64
```

In [33]:
```
sns.countplot(y='Marital_Status', data=df)
plt.title("Marital-status count")
plt.show()
```



In [18]:
```
df['City_Category'].value_counts() #City Category 'B' has the most
```

Out[18]:
```
City_Category
B    231173
C    171175
A    147720
Name: count, dtype: int64
```

In [ ]:
```
# Data Exploration like How does gender affect the avg amount spend
```

In [43]:
```
# Average amount spend by Male and Female
gender_mean=df.groupby('Gender')['Purchase'].mean()
```

In [44]:
```
gender_mean # We see that Male spend more than female
```

Out[44]:
```
Gender
F    8734.565765
M    9437.526040
Name: Purchase, dtype: float64
```

In [ ]:
```
# Preferred product category with respect to Gender
```

In [57]:
```
preferred_product=df.groupby(['Gender', 'Product_Category'])['Produ
```

In [58]:
```
preferred_product
```

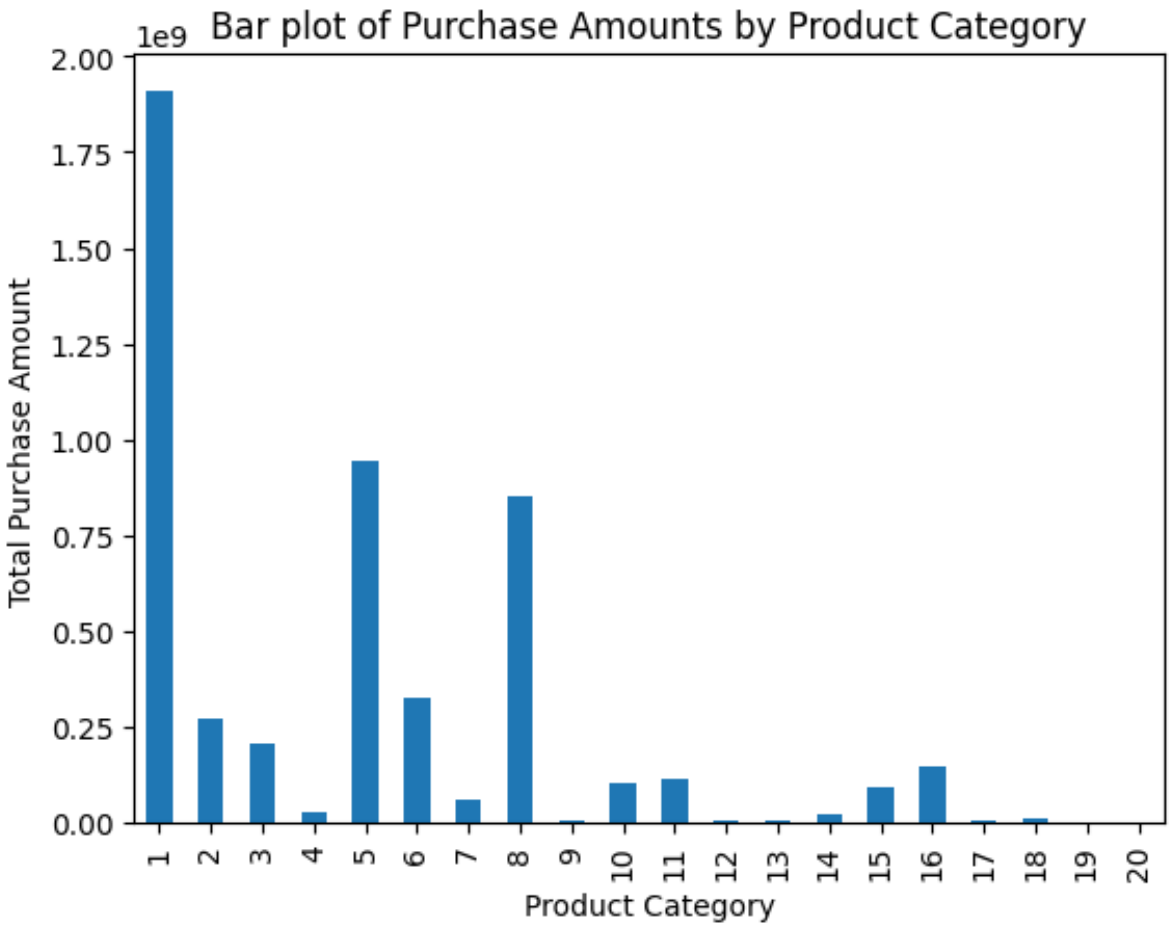
Out[58]:

Gender	Product_Category	
F	1	24831
	2	5658
	3	6006
	4	3639
	5	41961
	6	4559
	7	943
	8	33558
	9	70
	10	1162
	11	4739
	12	1532
	13	1462
	14	623
	15	1046
	16	2402
	17	62
	18	382
	19	451
	20	723
M	1	115547
	2	18206
	3	14207
	4	8114
	5	108972
	6	15907
	7	2778
	8	80367
	9	340
	10	3963
	11	19548
	12	2415
	13	4087
	14	900
	15	5244
	16	7426
	17	516
	18	2743
	19	1152
	20	1827

Name: Product_Category, dtype: int64

In [ ]:
```
# Preferred product category with respect to Age
```

In [506]:
```
df.groupby('Product_Category')['Purchase'].sum().plot(kind='bar')
plt.xlabel('Product Category')
plt.ylabel('Total Purchase Amount')
plt.title('Bar plot of Purchase Amounts by Product Category')
plt.show()
```



In [61]:
```
preferred_product_age=df.groupby(['Age', 'Product_Category'])['Prod
```

In [62]:
```
preferred_product_age
```

Out[62]:
```
Age   Product_Category
0-17  1                  3585
      2                   805
      3                  1200
      4                   758
      5                  4330
                         ...
55+   16                  377
      17                   67
      18                  241
      19                  103
      20                  160
Name: Product_Category, Length: 140, dtype: int64
```

In [ ]:
```
# Relationship b/w Marital_status and Age_group with the amount of
```

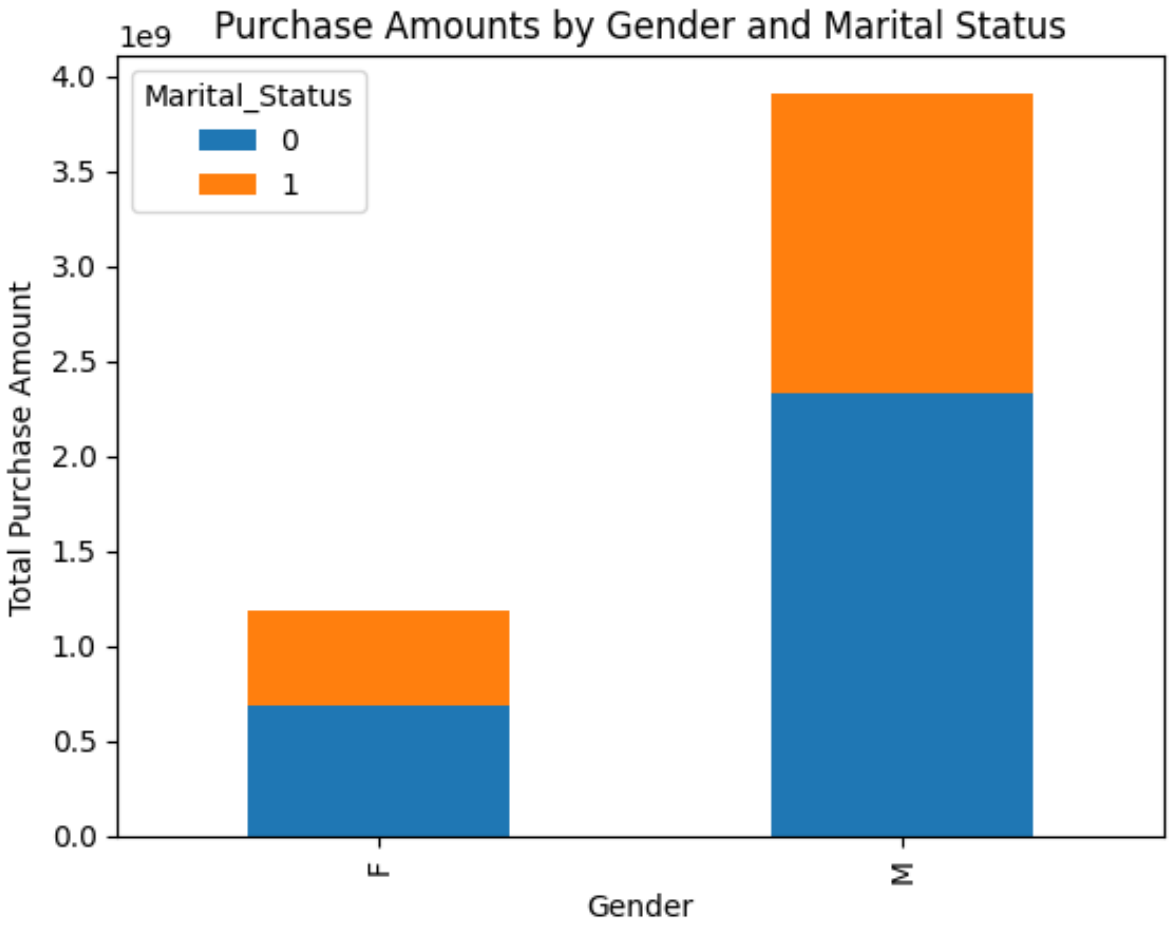
In [80]:
```
df.groupby(['Marital_Status','Age'])['Purchase'].count()
```

Out[80]:

Marital_Status	Age	
0	0-17	15102
	18-25	78544
	26-35	133296
	36-45	66377
	46-50	12690
	51-55	10839
	55+	7883
1	18-25	21116
	26-35	86291
	36-45	43636
	46-50	33011
	51-55	27662
	55+	13621

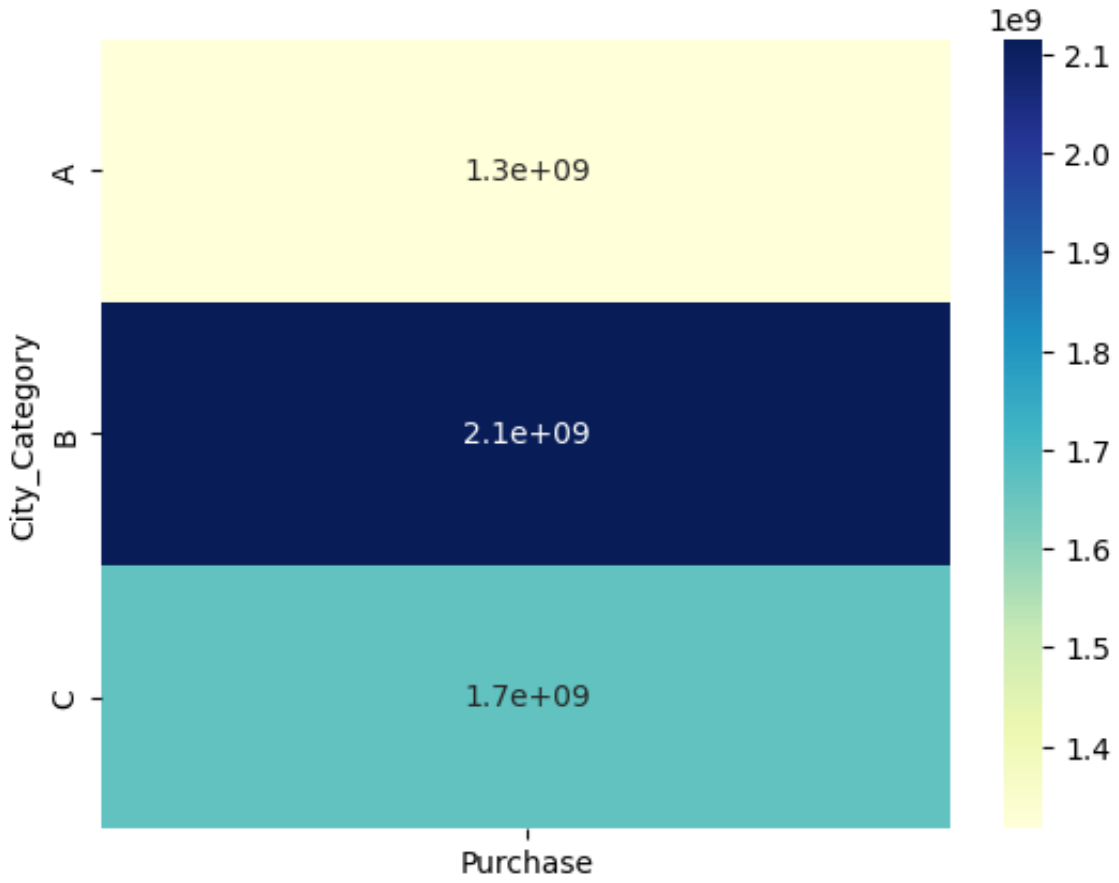
Name: Purchase, dtype: int64

In [508]:
```
# Stacked Bar Plot of Purchase Amounts by Gender and Marital Status


# group data by gender and marital status, and calculate total purc
gender_marital_purchase = df.groupby(['Gender', 'Marital_Status'])[

# create pivot table with gender as rows, marital status as columns
gender_marital_purchase_pivot = gender_marital_purchase.pivot_table

# create stacked bar plot
gender_marital_purchase_pivot.plot(kind='bar', stacked=True)
plt.xlabel('Gender')
plt.ylabel('Total Purchase Amount')
plt.title('Purchase Amounts by Gender and Marital Status')
plt.show()
```



In [509]:
```
# Heat Map of Purchase Amounts by City Category:


# group data by city category and calculate total purchase amount f
city_purchase = df.groupby('City_Category')['Purchase'].sum().reset

# create pivot table with city category as rows, and purchase amoun
city_purchase_pivot = city_purchase.pivot_table(values='Purchase',

# create heatmap
sns.heatmap(city_purchase_pivot, annot=True, cmap='YlGnBu')
```

Out[509]: <Axes: ylabel='City_Category'>



Confidence Interval and CLT

CLT on the female dataframe

In [ ]:
```
#Creating a new_df for female customers
```

In [118]:
```
female_df=df.loc[df['Gender']=='F']
```

In [119]:
```
female_df.head()
```

Out[119]:

	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	
14	1000006	P00231342	F	51-55	9	A	

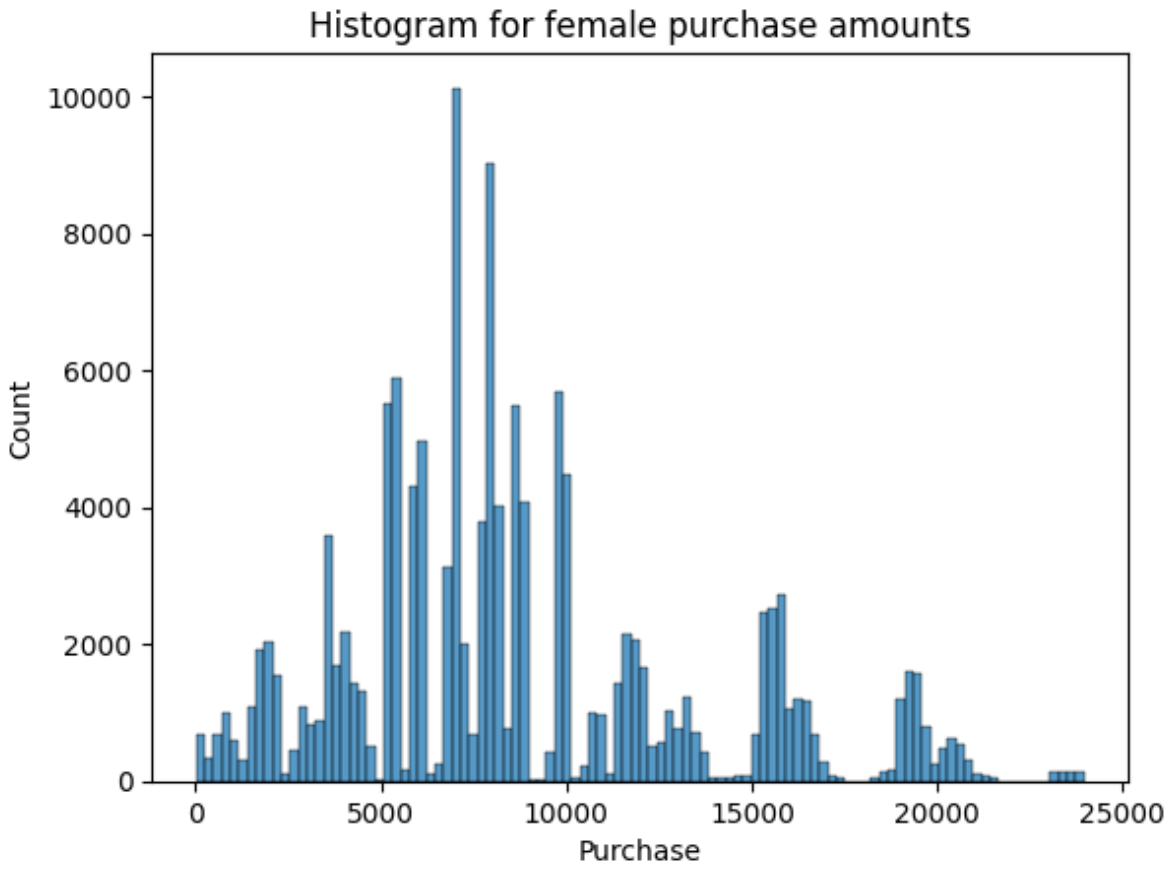
In [116]:
```
female_df['Purchase'].mean()
```

Out[116]:
```
8734.565765155476
```

In [117]:
```
female_df['Purchase'].std()
```

Out[117]:
```
4767.233289291458
```

In [498]:
```
sns.histplot(female_df['Purchase'])
plt.title('Histogram for female purchase amounts')
plt.show()
```



In [354]:
```
# Mean and std value of Female dataframe
f_mean=np.mean(female_df['Purchase'])
f_std=np.std(female_df['Purchase'])
print('Mean value of female purchase:', round(f_mean,2))
print('STD value of female purchase:', round(f_std,2))
```

```
Mean value of female purchase: 8734.57
STD value of female purchase: 4767.22
```

In [295]:
```
# Calculate the standard error of the mean
sem=f_std/np.sqrt(len(female_df['Purchase']))
sem
```

Out[295]:
```
12.936015594920663
```

In [356]:
```
# Calculation of Z value for 95% CI
z=np.abs(norm.ppf(97.5/100))
z
```

Out[356]:
```
1.959963984540054
```

In [296]:
```
# Margin of error
moe=z*sem
moe
```

Out[296]:
```
25.354124669492982
```

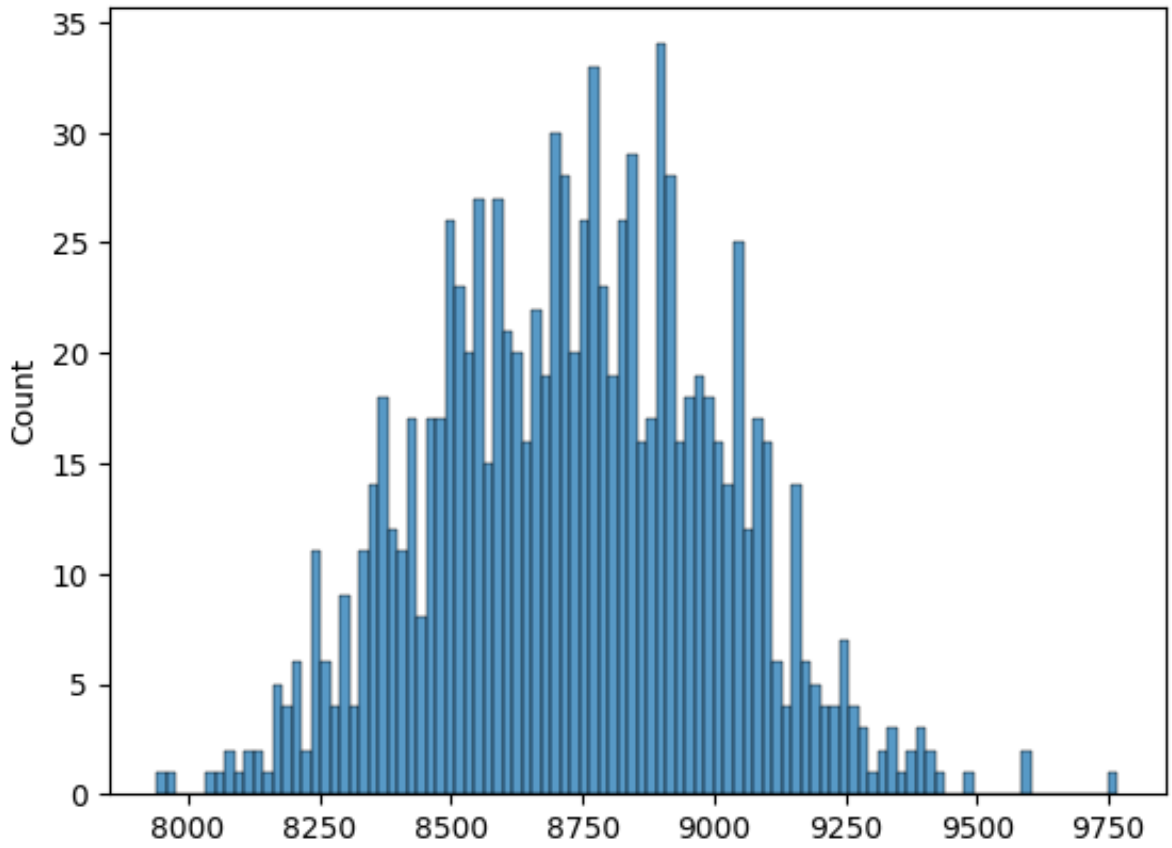
In [298]:
```
# Calculate upper bound and lower bounf for the confidence interval
lower_bound = f_mean - moe
upper_bound = f_mean + moe
```

In [360]:
```
# 95% Confidence Interval for the female_df
print('Lower Bound - Confidence Interval of 95% on female dataframe
print('Upper Bound - Confidence Interval of 95% on female dataframe
```

```
Lower Bound - Confidence Interval of 95% on female dataframe : 870
9.21
Upper Bound - Confidence Interval of 95% on female dataframe : 875
9.92
```

CLT on female dataframe considering sample-size of 300

In [454]:
```
sample_size=300
sample_mean_collection1=[]
for reps in range(1000):
    sample_mean1=female_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection1.append(sample_mean1)

sns.histplot(sample_mean_collection1, bins=100)
plt.show()
```



In [455]:
```
mean1=np.mean(sample_mean_collection1)
std1=np.std(sample_mean_collection1)
z=np.abs(norm.ppf(97.5/100))
sem1=f_std/np.sqrt(sample_size)
left1= mean1 -(z*sem1)
right1= mean1 +(z*sem1)
print("Sample Mean:", round(mean1,2))
print("Standard Error of the Mean:", round(sem1,2))
print('Lower Bound - CI of 95% with 300 samples :', round(left1,2))
print('Right Bound - CI of 95% with 300 samples :',round(right1,2))
```

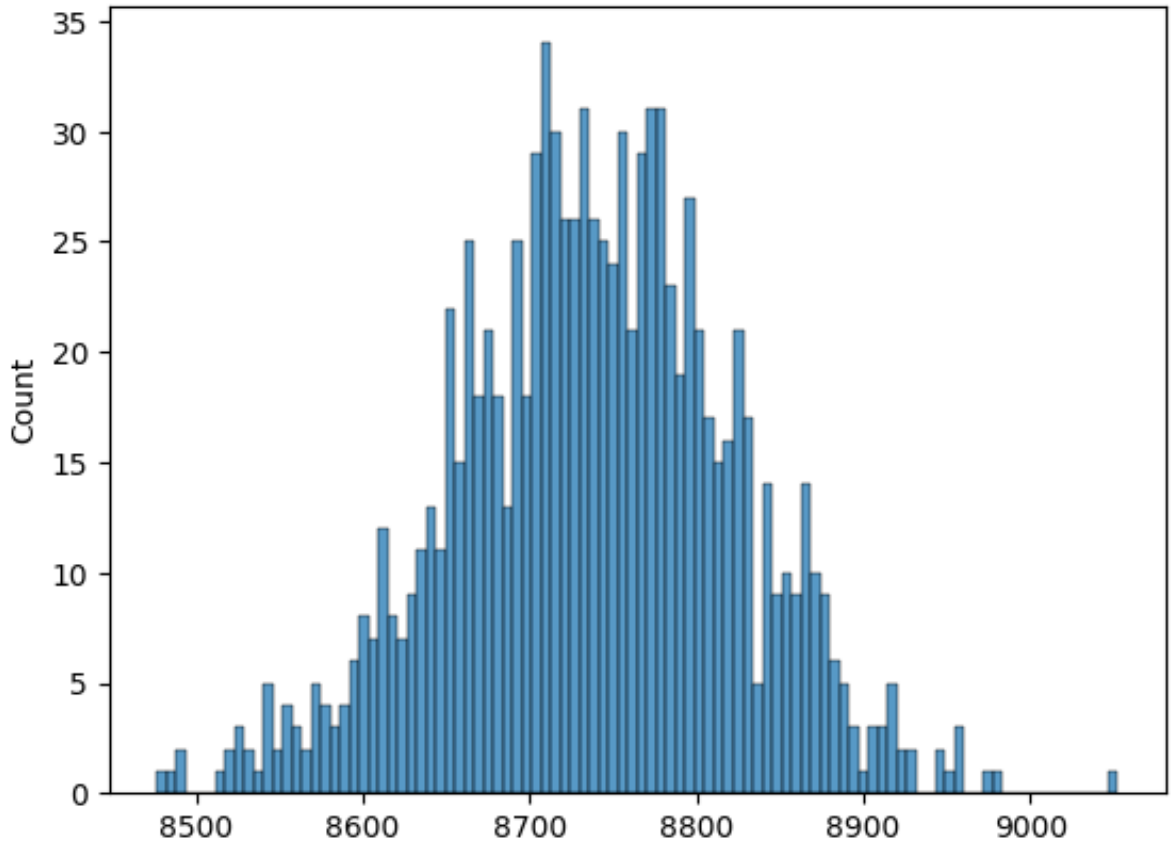
```
Sample Mean: 8737.3
Standard Error of the Mean: 275.24
Lower Bound - CI of 95% with 300 samples : 8197.85
Right Bound - CI of 95% with 300 samples : 9276.75
```

In [ ]:

CLT on female dataframe considering sample-size of 3000

In [457]:
```
sample_size=3000
sample_mean_collection2=[]
for reps in range(1000):
    sample_mean2=female_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection2.append(sample_mean2)

sns.histplot(sample_mean_collection2, bins=100)
```

Out[457]:
```
<Axes: ylabel='Count'>
```



In [458]:
```
mean2=np.mean(sample_mean_collection2)
std2=np.std(sample_mean_collection2)
sem2 = f_std / np.sqrt(sample_size)
z=np.abs(norm.ppf(97.5/100))
left2= mean2 -(z*sem2)
right2= mean2 +(z*sem2)
print("Sample Mean:", round(mean2,2))
print("Standard Error of the Mean:", round(sem2,2))
print('Lower Bound - CI of 95% with 3000 samples :', round(left2,2)
print('Right Bound - CI of 95% with 3000 samples :',round(right2,2)
```

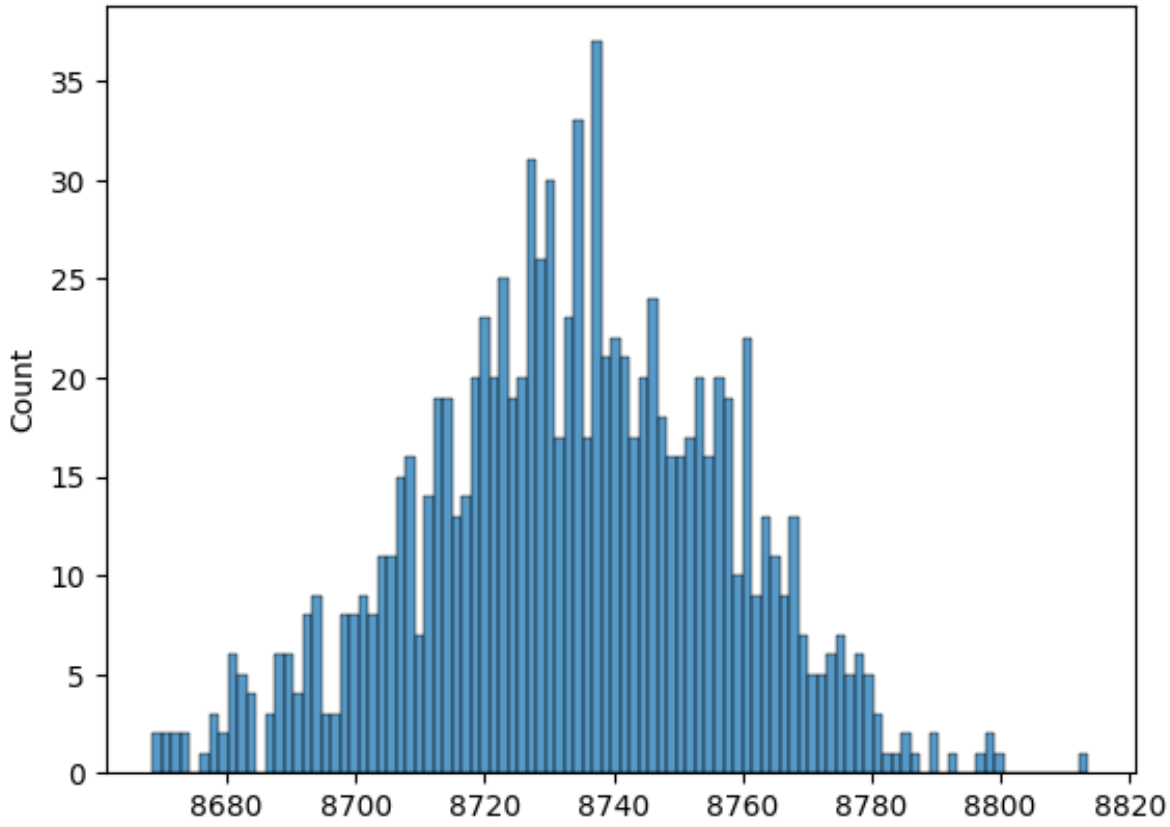
```
Sample Mean: 8737.79
Standard Error of the Mean: 87.04
Lower Bound - CI of 95% with 3000 samples : 8567.2
Right Bound - CI of 95% with 3000 samples : 8908.38
```

In [ ]:

CLT on female dataframe considering sample-size of 30000

In [459]:
```
sample_size=30000
sample_mean_collection3=[]
for reps in range(1000):
    sample_mean3=female_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection3.append(sample_mean3)

sns.histplot(sample_mean_collection3, bins=100)
```

Out[459]: <Axes: ylabel='Count'>



In [460]:
```
mean3=np.mean(sample_mean_collection3)
std3=np.std(sample_mean_collection3)
sem3 = f_std / np.sqrt(sample_size)
z=np.abs(norm.ppf(97.5/100))
left3= mean3 -(z*sem3)
right3= mean3 +(z*sem3)
print("Sample Mean:", round(mean3,2))
print("Standard Error of the Mean:", round(sem3,2))
print('Lower Bound - CI of 95% with 30000 samples :', round(left3,2
print('Upper Bound - CI of 95% with 30000 samples :',round (right3,
```

```
Sample Mean: 8733.5
Standard Error of the Mean: 27.52
Lower Bound - CI of 95% with 30000 samples : 8679.56
Upper Bound - CI of 95% with 30000 samples : 8787.45
```

In [ ]:

CLT on Male Dataframe

In [337]:
```
#Creating a new_df for Male customers
```

In [338]:
```
male_df=df.loc[df['Gender']=='M']
```

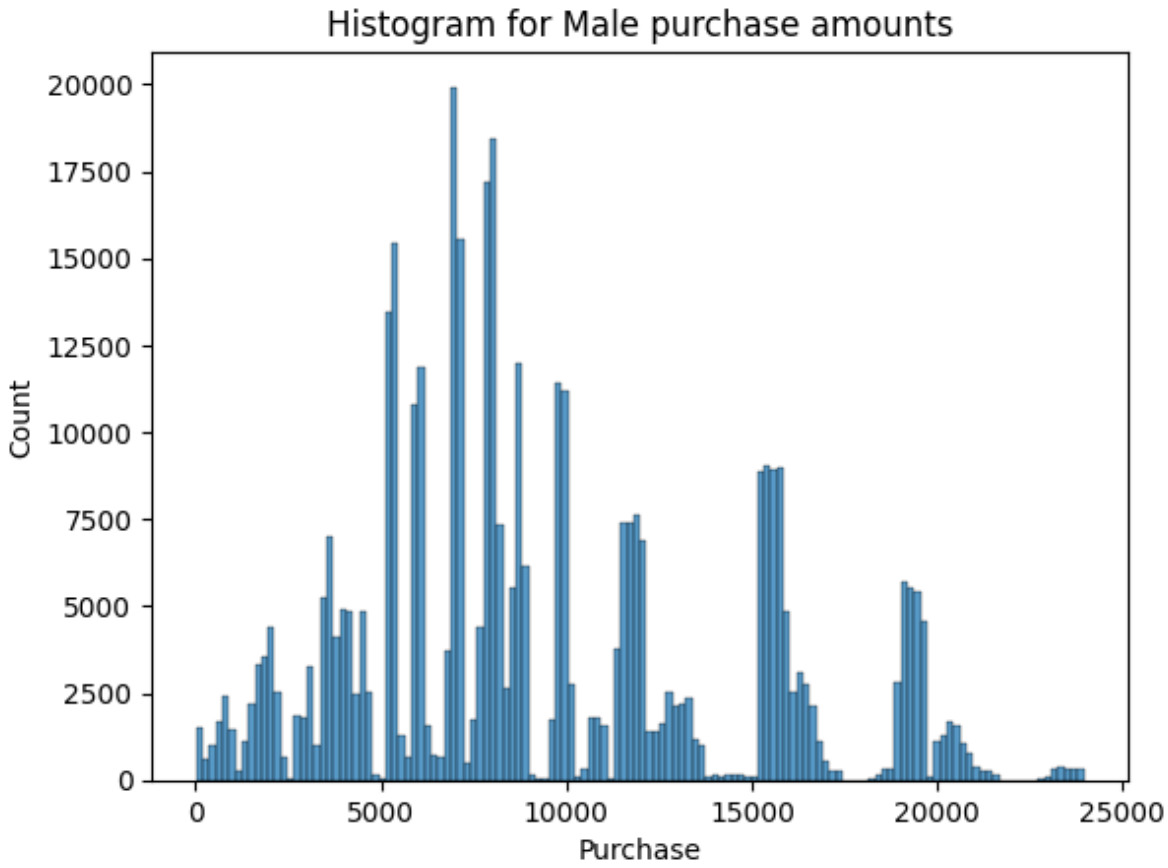
In [339]:
```
male_df
```

Out[339]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_Cit
4	1000002	P00285442	M	55+	16	C	
5	1000003	P00193542	M	26-35	15	A	
6	1000004	P00184942	M	46-50	7	B	
7	1000004	P00346142	M	46-50	7	B	
8	1000004	P0097242	M	46-50	7	B	
...	...	...	...	...	...	...	
550057	1006023	P00370853	M	26-35	0	C	
550058	1006024	P00372445	M	26-35	12	A	
550060	1006026	P00371644	M	36-45	6	C	
550062	1006032	P00372445	M	46-50	7	A	
550063	1006033	P00372445	M	51-55	13	B	

414259 rows × 10 columns

In [499]:
```
sns.histplot(male_df['Purchase'])
plt.title('Histogram for Male purchase amounts')
plt.show()
```



In [353]:
```
# Mean and std value of Male dataframe
m_mean=np.mean(male_df['Purchase'])
m_std=np.std(male_df['Purchase'])
print('Mean value of male purchase:', round(m_mean,2))
print('STD value of male purchase:', round(m_std,2))
```

```
Mean value of male purchase: 9437.53
STD value of male purchase: 5092.18
```

In [386]:
```
# Calculate the standard error of the mean
sem=m_std/np.sqrt(len(male_df['Purchase']))
sem
```

Out[386]:
```
7.911662926429213
```

In [387]:
```
# Calculation of Z value for 95% CI
z=np.abs(norm.ppf(97.5/100))
z
```

Out[387]:
```
1.959963984540054
```

In [388]:
```
# Margin of error
moe=z*sem
moe
```

Out[388]:
```
15.506574393622024
```

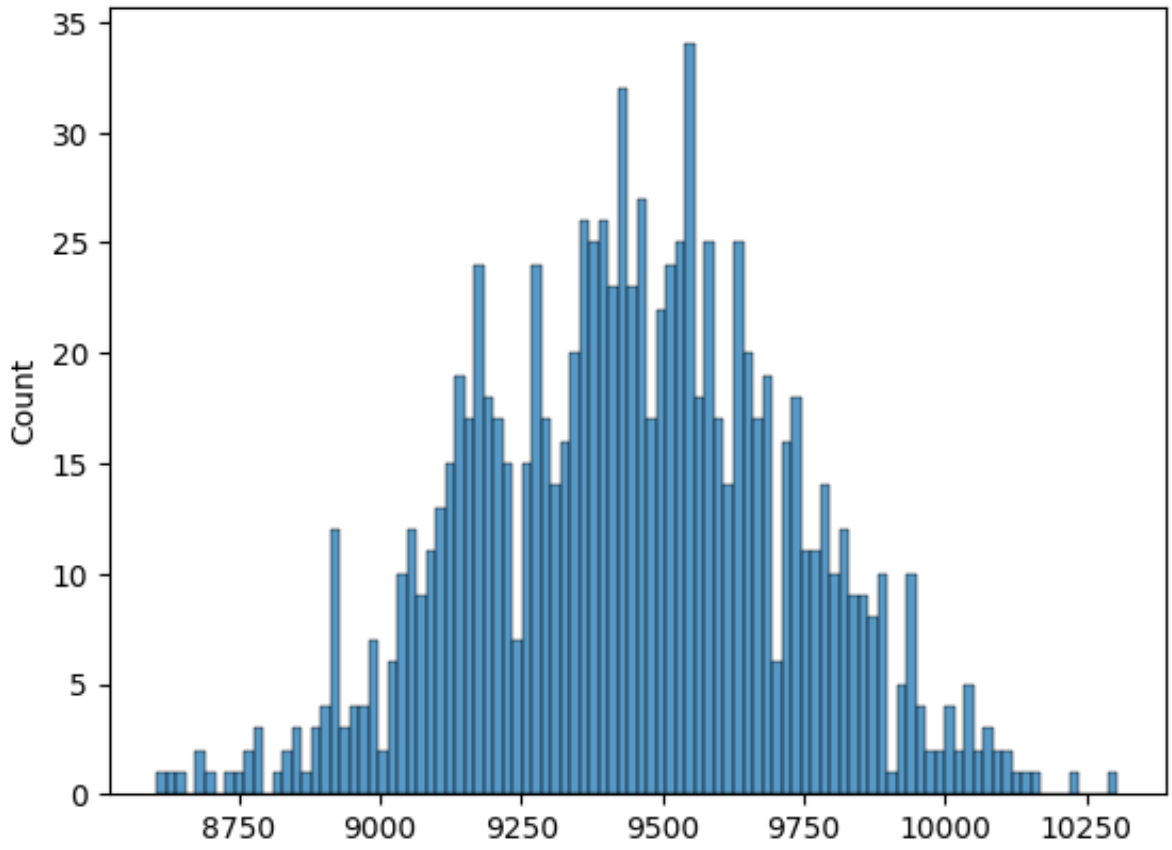
In [389]:
```
# Calculate upper bound and lower bounf for the confidence interval
lower_bound_m = m_mean - moe
upper_bound_m = m_mean + moe
```

In [390]:
```
# 95% Confidence Interval for the Male_df
print('Lower Bound - Confidence Interval of 95% on Male dataframe :
print('Upper Bound - Confidence Interval of 95% on Male dataframe :
```

```
Lower Bound - Confidence Interval of 95% on Male dataframe : 9422.
02
Upper Bound - Confidence Interval of 95% on Male dataframe : 9453.
03
```

CLT on Male dataframe considering sample-size of 300

In [471]:
```
sample_size=300
sample_mean_collection_a=[]
for reps in range(1000):
    sample_mean_a=male_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection_a.append(sample_mean_a)

sns.histplot(sample_mean_collection_a, bins=100)
plt.show()
```

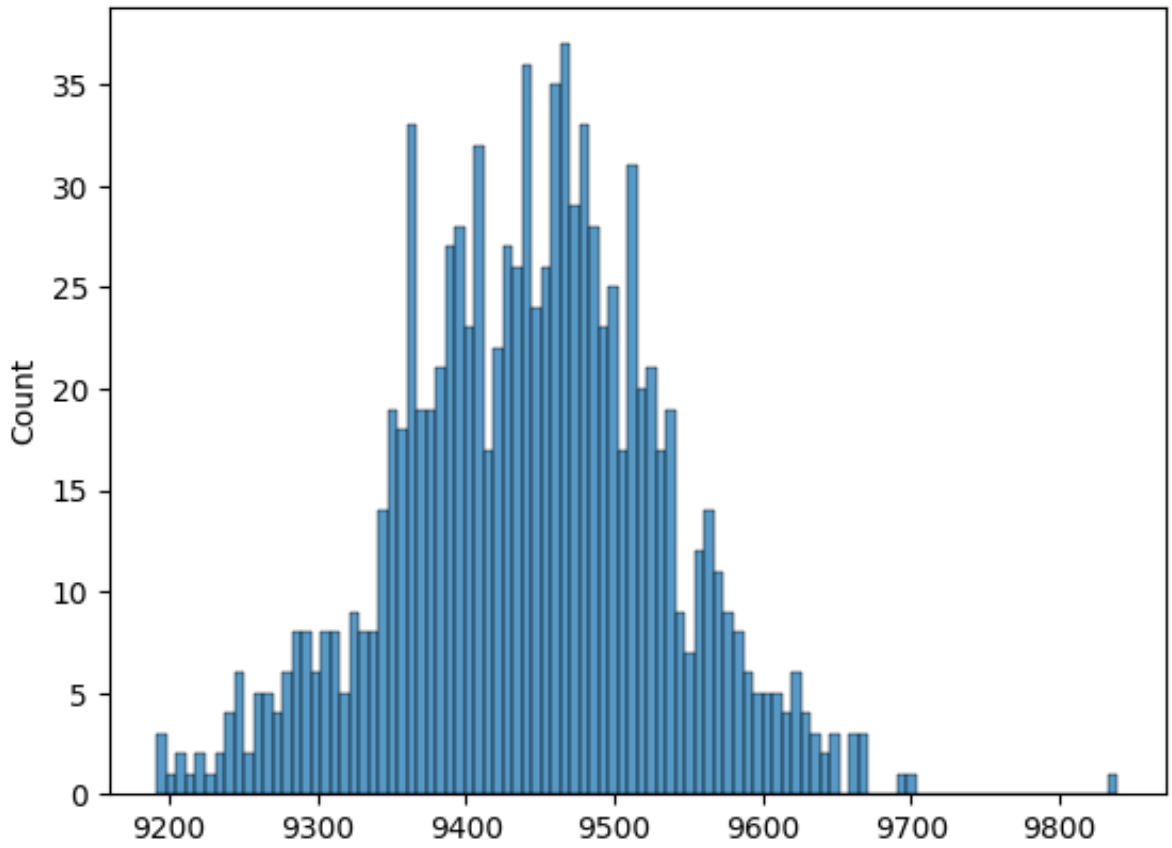


In [472]:
```
mean_a=np.mean(sample_mean_collection_a)
std_a=np.std(sample_mean_collection_a)
z=np.abs(norm.ppf(97.5/100))
sem_a=m_std/np.sqrt(sample_size)
left_a= mean_a -(z*sem_a)
right_a= mean_a +(z*sem_a)
print("Sample Mean:", round (mean_a,2))
print("Standard Error of the Mean:", round(sem_a,2))
print('Lower Bound - CI of 95% with 300 samples :', round(left_a,2)
print('Upper Bound - CI of 95% with 300 samples :', round(right_a,2
```

```
Sample Mean: 9443.6
Standard Error of the Mean: 294.0
Lower Bound - CI of 95% with 300 samples : 8867.38
Upper Bound - CI of 95% with 300 samples : 10019.83
```

CLT on Male dataframe considering sample-size of 3000

In [469]:
```
sample_size=3000
sample_mean_collection_b=[]
for reps in range(1000):
    sample_mean_b=male_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection_b.append(sample_mean_b)

sns.histplot(sample_mean_collection_b, bins=100)
plt.show()
```

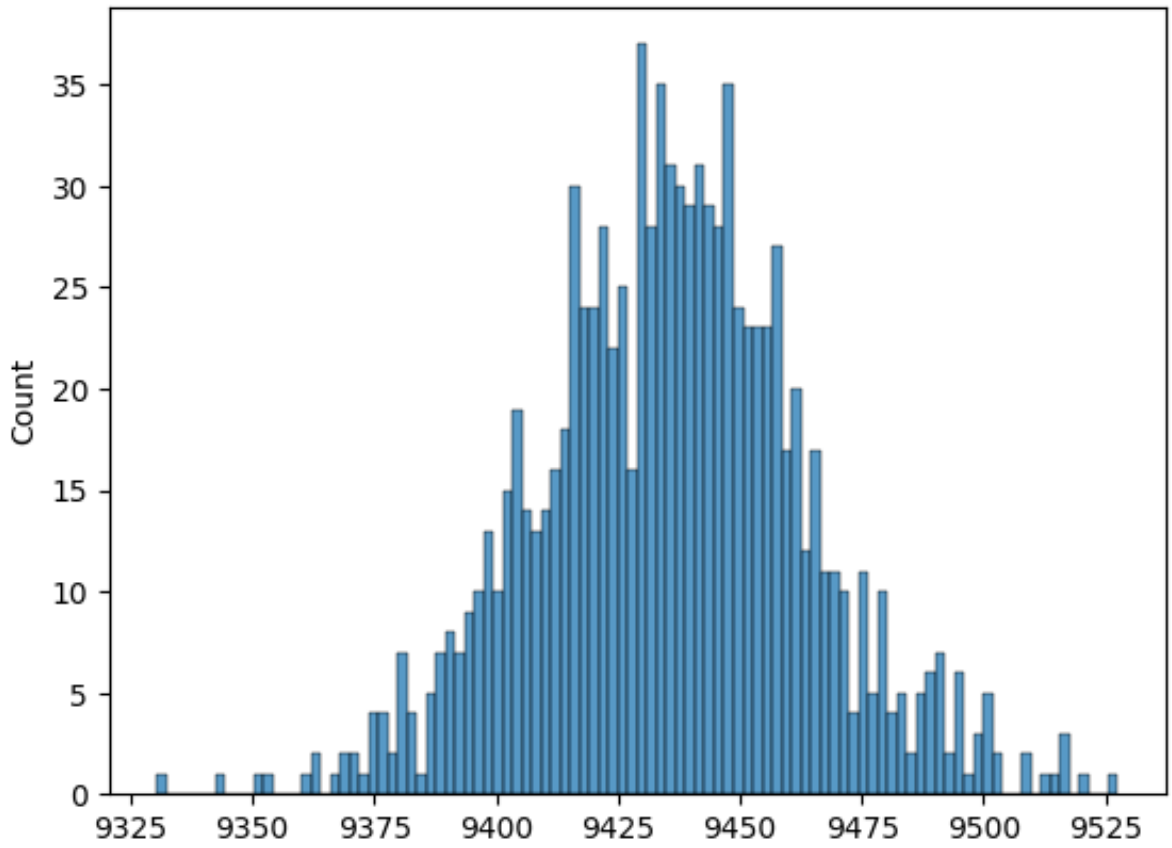


In [470]:
```
mean_b=np.mean(sample_mean_collection_b)
std_b=np.std(sample_mean_collection_b)
z=np.abs(norm.ppf(97.5/100))
sem_b=m_std/np.sqrt(sample_size)
left_b= mean_b -(z*sem_b)
right_b= mean_b +(z*sem_b)
print("Sample Mean:", round (mean_b,2))
print("Standard Error of the Mean:", round(sem_b,2))
print('Lower Bound - CI of 95% with 3000 samples :', round(left_b,2
print('Upper Bound - CI of 95% with 3000 samples :', round(right_b,
```

```
Sample Mean: 9443.48
Standard Error of the Mean: 92.97
Lower Bound - CI of 95% with 3000 samples : 9261.26
Upper Bound - CI of 95% with 3000 samples : 9625.7
```

CLT on Male dataframe considering sample-size of 30000

In [465]:
```
sample_size=30000
sample_mean_collection_c=[]
for reps in range(1000):
    sample_mean_c=male_df['Purchase'].sample(sample_size).mean()
    sample_mean_collection_c.append(sample_mean_c)

sns.histplot(sample_mean_collection_c, bins=100)
plt.show()
```



In [466]:
```
mean_c=np.mean(sample_mean_collection_c)
std_c=np.std(sample_mean_collection_c)
z=np.abs(norm.ppf(97.5/100))
sem_c=m_std/np.sqrt(sample_size)
left_c= mean_c -(z*sem_c)
right_c= mean_c +(z*sem_c)
print("Sample Mean:", round (mean_c,2))
print("Standard Error of the Mean:", round(sem_c,2))
print('Lower Bound - CI of 95% with 30000 samples :', round(left_c,
print('Upper Bound - CI of 95% with 30000 samples :', round(right_c
```

```
Sample Mean: 9435.83
Standard Error of the Mean: 29.4
Lower Bound - CI of 95% with 30000 samples : 9378.21
Upper Bound - CI of 95% with 30000 samples : 9493.46
```

Below code we can modify for gender CLT according to the need of Confidence interval and sample size

In [512]:
```
# define confidence level
alpha = 0.05

# define sample size
n = 300

# calculate sample mean and standard deviation for each gender
female_mean = female_df['Purchase'].mean()
female_std = female_df['Purchase'].std()
male_mean = male_df['Purchase'].mean()
male_std = male_df['Purchase'].std()

# calculate standard error of the mean for each gender
female_sem = f_std/ np.sqrt(n)
male_sem = m_std/ np.sqrt(n)

# calculate margin of error for each gender
female_moe = female_sem * stats.norm.ppf(1 - alpha/2)
male_moe = male_sem * stats.norm.ppf(1 - alpha/2)

# calculate confidence interval for each gender
female_ci = (female_mean - female_moe, female_mean + female_moe)
male_ci = (male_mean - male_moe, male_mean + male_moe)

print('95% Confidence Interval for Female Customers: ({:.2f}, {:.2f
print('95% Confidence Interval for Male Customers: ({:.2f}, {:.2f})
```

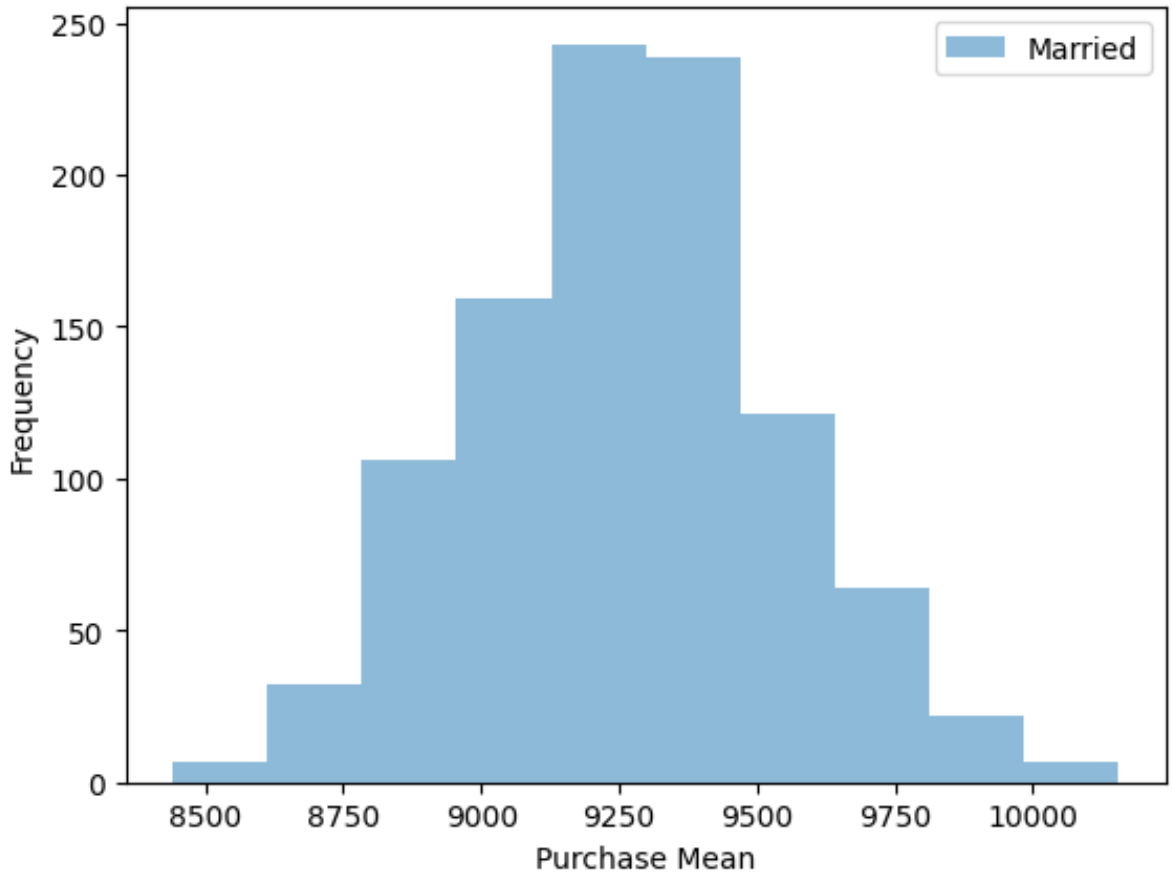
```
95% Confidence Interval for Female Customers: (8195.11, 9274.02)
95% Confidence Interval for Male Customers: (8861.30, 10013.75)
```

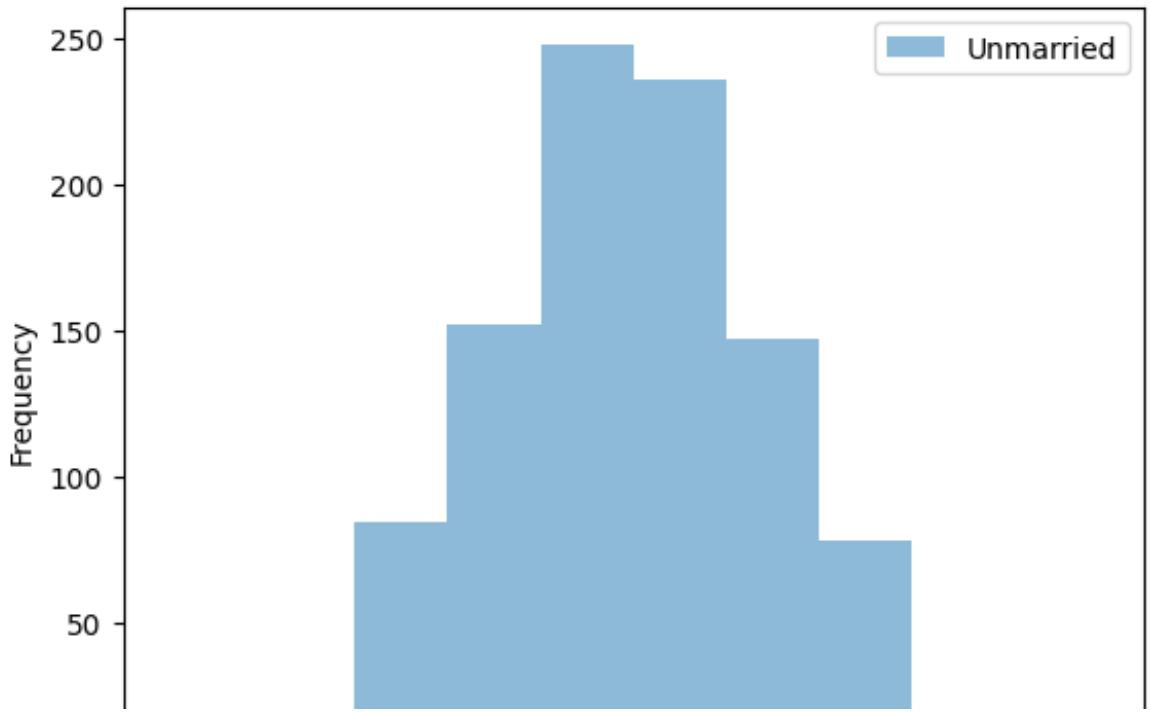
CLT for Married and Unmaried Customers

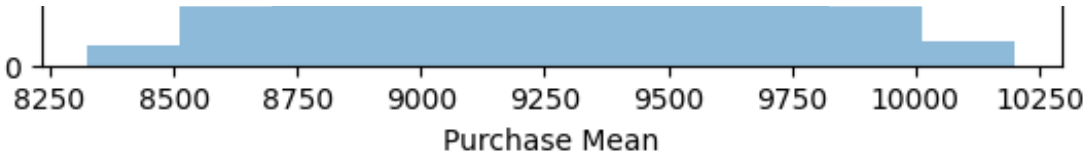
In [513]:
```
# Define the sample size and number of samples
sample_size = 3000

# Calculate the sample means and standard errors for Married and Un
married_means = []
unmarried_means = []

for i in range(1000):
    married_sample = df[df['Marital_Status'] == 1]['Purchase'].samp
    married_mean = np.mean(married_sample)
    married_se = np.std(married_sample, ddof=1) / np.sqrt(sample_si
    married_interval = (married_mean - 1.96 * married_se, married_m
    married_means.append(married_mean)

    unmarried_sample = df[df['Marital_Status'] == 0]['Purchase'].sa
    unmarried_mean = np.mean(unmarried_sample)
    unmarried_se = np.std(unmarried_sample, ddof=1) / np.sqrt(sampl
    unmarried_interval = (unmarried_mean - 1.96 * unmarried_se, unm
    unmarried_means.append(unmarried_mean)
```

In [519]:

```
# Plot the distribution of sample means for Married and Unmarried c
plt.hist(married_means, alpha=0.5, label='Married')
plt.xlabel('Purchase Mean')
plt.ylabel('Frequency')
plt.legend()
plt.show()
plt.hist(unmarried_means, alpha=0.5, label='Unmarried')
plt.xlabel('Purchase Mean')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```







In [ ]:

Overlap for sample size 300

The confidence intervals provided suggest that for a sample size of 300, the average spending of female customers falls between 8195 and 9274 with 95% confidence, while the average spending of male customers falls between 8861 and 10013 with 95% confidence.

Based on these confidence intervals, we can see that the average spending of male customers tends to be higher than that of female customers. However, since the confidence intervals overlap, we cannot conclude with certainty that there is a statistically significant difference in spending between the two genders.

To leverage these insights, we can focus on developing and promoting products that appeal to both male and female customers equally. We can also focus on improving the overall shopping experience for all customers, regardless of gender.

It is better approach to take more samples to leverage the data and to provide actual insights and recommendation

Recommedations and action items to Walmart

The confidence intervals provided suggest that for a larger sample size of 30000, the average spending of female customers falls between 8591.40 and 8877.73 with 90% confidence, while the average spending of male customers falls between 9284.60 and 9590.45 with 90% confidence. Similarly, for a smaller sample size of 3000, the average spending of female customers falls between 8689.29 and 8779.84 with 90% confidence, while the average spending of male customers falls between 9389.17 and 9485.88 with 90% confidence.

Based on these confidence intervals, we can see that the average spending of male customers tends to be higher than that of female customers in all cases. However, again, the difference is not statistically significant. To leverage these insights, we can continue to focus on developing and promoting products that appeal to both male and female customers equally, improving the shopping experience, and gathering more data to refine marketing and sales strategies.

CLT of married and Unmarried: there are more people in the age-group of 18-35 who are unmarried and has highest sales figures, we can come up with some promtional offers for married people to deliver high sales in the age bracket.

We find that the average spending of male customers is consistently higher than that of female customers, we have to consider targeted marketing efforts to better engage female customers.

City Cateogry B has the highest number of sales, We should figure out whats going wrong with city_category A and C is it the population rate or competetion factor.

In [ ]: