Walmart - Confidence Interval and CLT

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).

Importing the Necessary Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
from scipy import stats
import warnings
warnings.filterwarnings('ignore')
```

- Reading the data and performing basic checks

In [2]:
```
url = ('https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/origina
df = pd.read_csv(url)
df.head()
```

Out[2]:

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

In [3]:
```
print(f"Number of rows: {df.shape[0]:,} \nNumber of columns: {df.shape[1]}")
```

```
Number of rows: 550,068
Number of columns: 10
```

- Checking for null values

In [4]:
```
df.isna().sum()
```

Out[4]:
```
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
```

No Null values Detected.

- Checking the unique values in every column

In [5]:
```
df.nunique().sort_values(ascending=False)
```

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

- Checking for duplicates

In [6]:
```
df.duplicated().sum()
```

Out[6]:
```
0
```

No Duplicates Detected.

In [7]:
```
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
```

'User_ID','Product_ID','Gender', 'Age','City_Category','Marital_Status' have categorical values.So we need to change the datatype from int and object to category.

In [8]:
```
col = ['User_ID','Product_ID','Gender','Age','City_Category','Marital_Status']
df[col] = df[col].astype('category')
df.dtypes
```

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

Data Types have changed.

In [9]:
```
df.describe().T
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
Occupation	550068.0	8.076707	6.522660	0.0	2.0	7.0	14.0	20.0
Product_Category	550068.0	5.404270	3.936211	1.0	1.0	5.0	8.0	20.0
Purchase	550068.0	9263.968713	5023.065394	12.0	5823.0	8047.0	12054.0	23961.0

In [10]:
```
df.describe(include=['object','category']).T
```

Out[10]:

	count	unique	top	freq
User_ID	550068	5891	1001680	1026
Product_ID	550068	3631	P00265242	1880
Gender	550068	2	M	414259
Age	550068	7	26-35	219587
City_Category	550068	3	B	231173
Stay_In_Current_City_Years	550068	5	1	193821
Marital_Status	550068	2	0	324731

1. There are 5891 unique users. User ID 1001680 has made the most frequent purchases from Walmart.
2. There are 3631 unique products. Product ID P00265242 is the most frequently sold item.
3. Men tend to make purchases more frequently than women.
4. There are 7 unique age categories, with the most frequent buyers falling into the 26-35 age group.
5. There are 3 different city categories, with the majority of frequent buyers residing in category B cities.
6. Most people have been living in their current city for 1 year.
7. The majority of customers are unmarried.

Univariate Analysis

In [11]:
```
df['User_ID'].nunique()
```

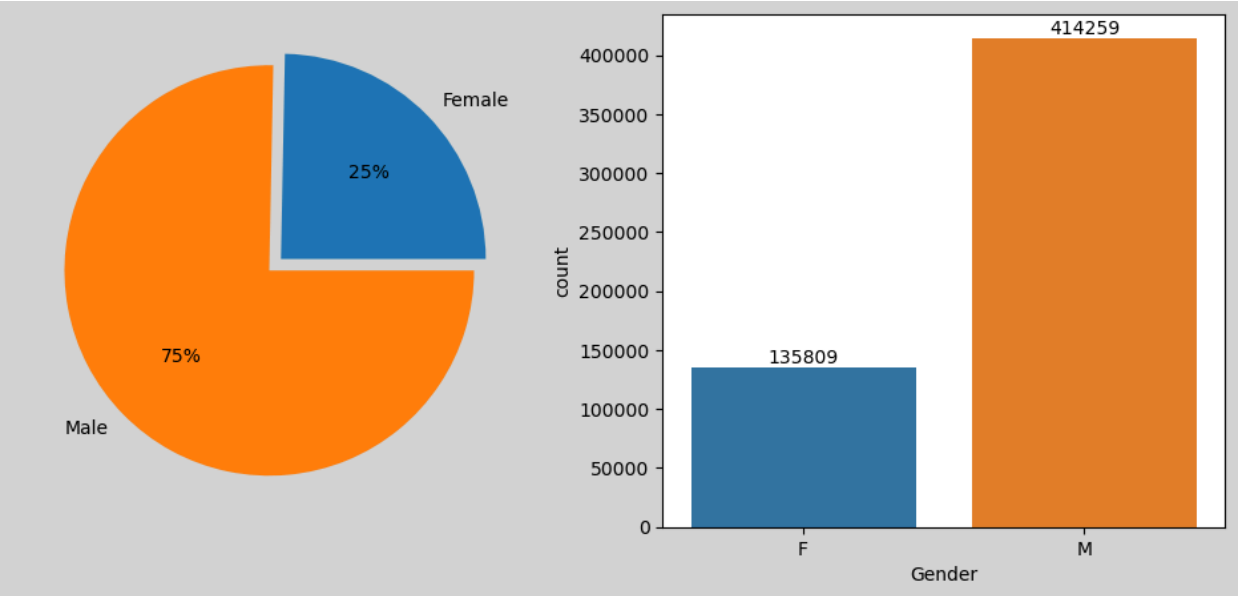
Out[11]:
```
5891
```

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

Out[12]:
```
3631
```

In [13]:
```
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")
plt.subplot(1,2,1)
labels = ['Female','Male']
plt.pie(df.groupby('Gender')['Gender'].count(), labels = labels, explode = (0.08,0), a
plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Gender')
for i in label.containers:
```

```
    label.bar_label(i)
plt.show()
```

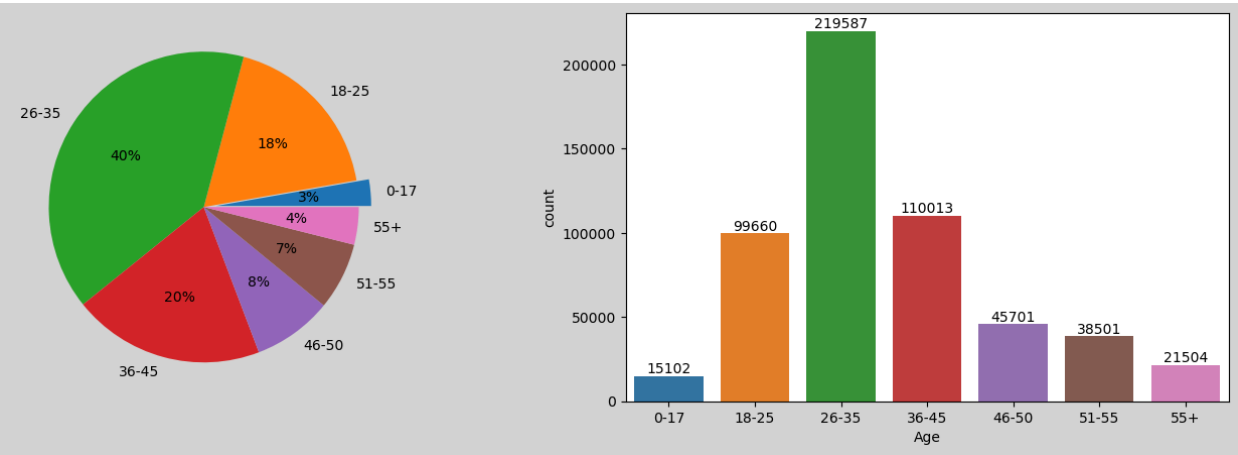


1. Out of the 0.54 million entries, approximately 75% of the records belong to men, while around 25% belong to women.
2. This translates to roughly 0.41 million records for men and approximately 0.13 million records for women.

In [14]:
```
df['Age'].unique()
```

Out[14]:
```
['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25']
Categories (7, object): ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55', '55+']
```

In [15]:
```
plt.figure(figsize = (17,5)).set_facecolor("lightgrey")
plt.subplot(1,2,1)
labels = ['0-17','18-25','26-35','36-45','46-50','51-55','55+']
plt.pie(df.groupby('Age')['Age'].count(), labels = labels, explode = (0.08,0,0,0,0,0,0
plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Age')
for i in label.containers:
    label.bar_label(i)
plt.show()
```

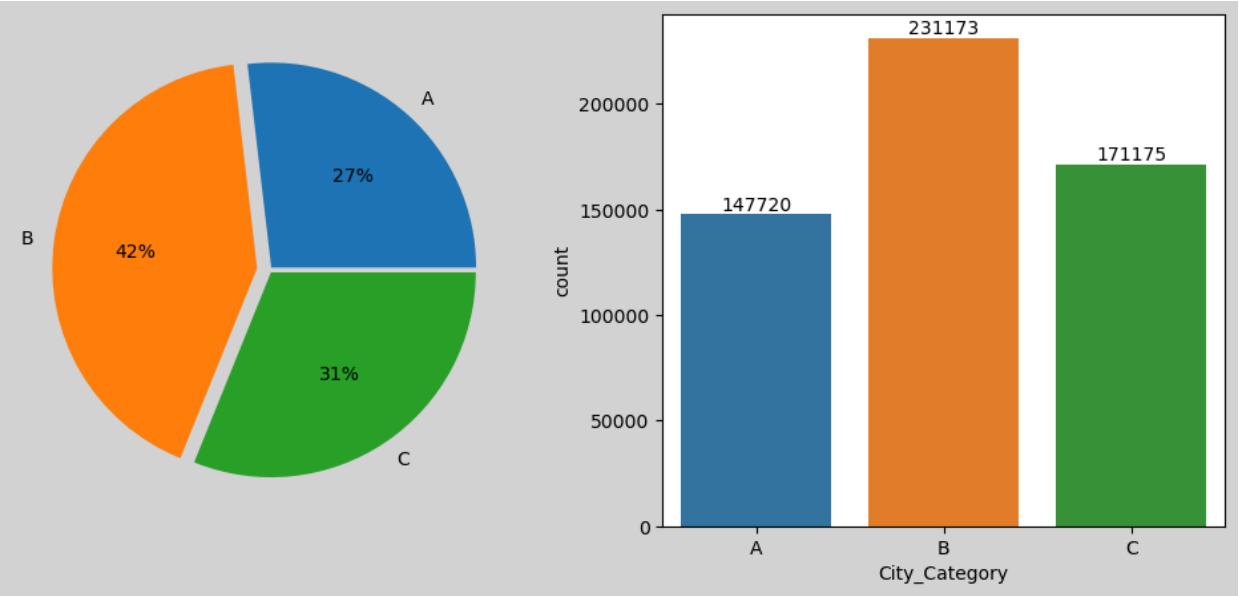


1. The age group of 26-35 constitutes the largest portion of buyers, accounting for 40% of the total.
2. Approximately 210,000 records belong to the age group 26-35, followed by 110,000 records for the 36-45 age group.
3. The age groups 0-17 and 55+ comprise the smallest proportion of buyers, representing only 3% and 4% of the data, respectively.
4. There are approximately 15,000 and 21,000 records for the 0-17 and 55+ age groups, respectively.
5. It's evident that the majority of buyers fall within the age range of 18-45, with fewer buyers observed before and after this range.

In [16]:
```
df['City_Category'].unique()
```

Out[16]:
```
['A', 'C', 'B']
Categories (3, object): ['A', 'B', 'C']
```

In [17]:
```
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")
plt.subplot(1,2,1)
labels = ['A','B','C']
plt.pie(df.groupby('City_Category')['City_Category'].count(), labels = labels, explode
plt.subplot(1,2,2)
label = sns.countplot(data = df, x='City_Category')
for i in label.containers:
    label.bar_label(i)
plt.show()
```



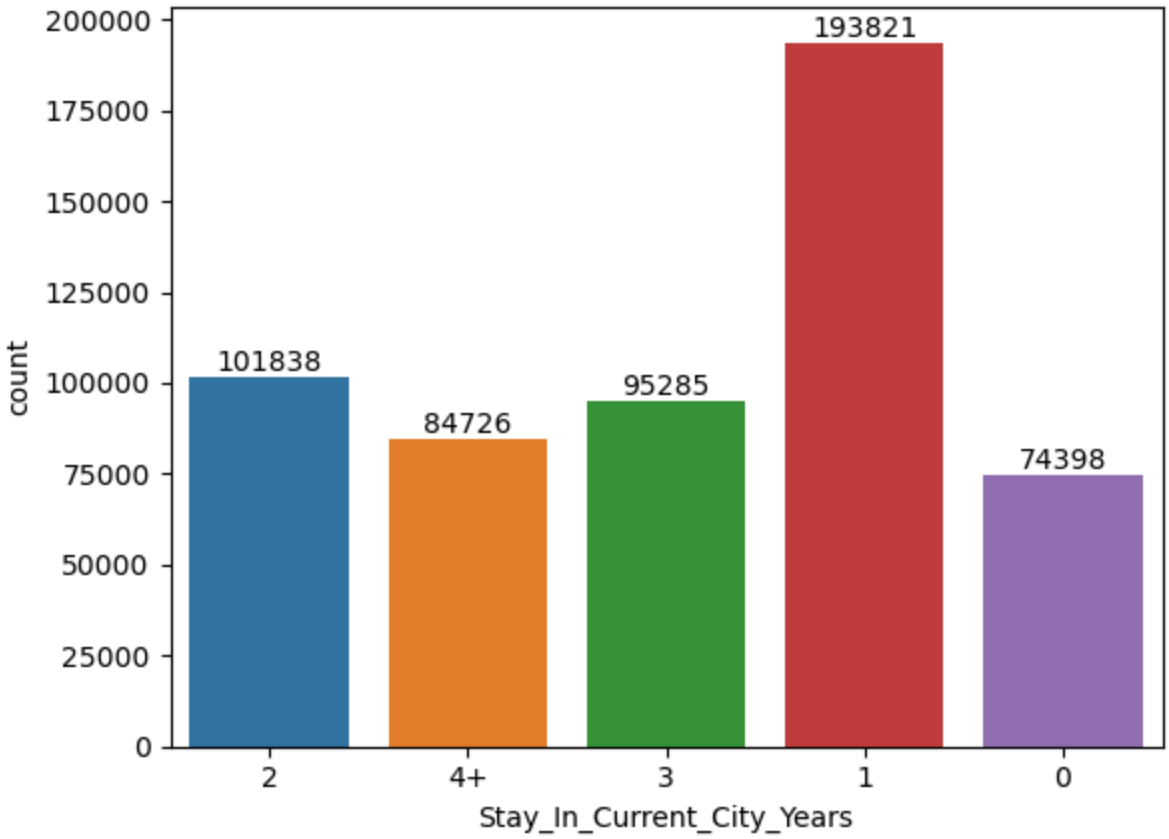
1. City Category B comprises 42% of the buyers, Category C includes 31%, and Category A consists of 27%.

2. Approximately 0.23 million records are present for Category B, 0.17 million for Category C, and 0.14 million for Category A.

In [18]:
```
df['Stay_In_Current_City_Years'].unique()
```

Out[18]:
```
array(['2', '4+', '3', '1', '0'], dtype=object)
```

In [19]:
```
label = sns.countplot(data = df, x='Stay_In_Current_City_Years')
for i in label.containers:
    label.bar_label(i)
```



The majority of buyers have been residing in their current cities for one year, followed by two years and three years.

In [20]:
```
df['Marital_Status'].unique()
```

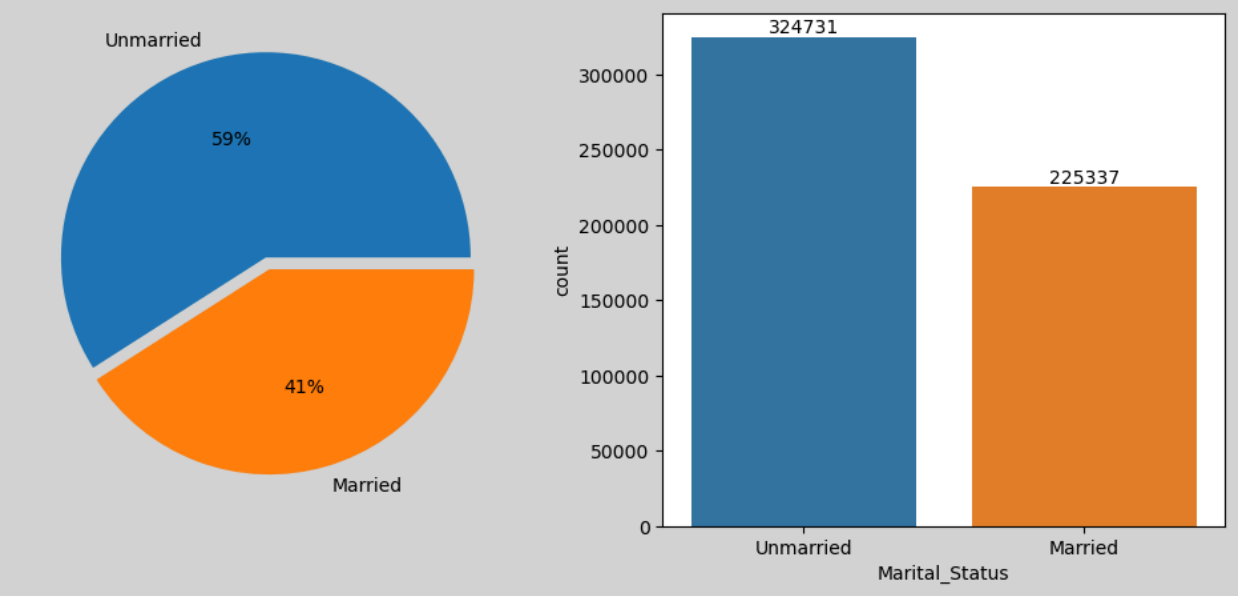
Out[20]:
```
[0, 1]
Categories (2, int64): [0, 1]
```

1. In the dataset's marital_status column, values of 0 represent "Unmarried" and values of 1 represent "Married."

2. Therefore, we will replace these values in the dataset accordingly.

In [21]:
```
df['Marital_Status'].replace(to_replace = 0, value = 'Unmarried',inplace = True)
df['Marital_Status'].replace(to_replace = 1, value = 'Married',inplace = True)
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")
plt.subplot(1,2,1)
```

```
labels = ['Unmarried','Married']
plt.pie(df.groupby('Marital_Status')['Marital_Status'].count(), labels = labels, explo
plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Marital_Status')
for i in label.containers:
    label.bar_label(i)
plt.show()
```



1. Among frequent buyers, approximately 59% are unmarried individuals, while 41% are married.

2. There are approximately 0.32 million entries for unmarried individuals and 0.22 million for married individuals.

In [22]:
```
round(df['Purchase'].describe(),2)
```

Out[22]:
```
count    550068.00
mean       9263.97
std        5023.07
min          12.00
25%        5823.00
50%        8047.00
75%       12054.00
max       23961.00
Name: Purchase, dtype: float64
```

While observing their spending habits of all buyers.

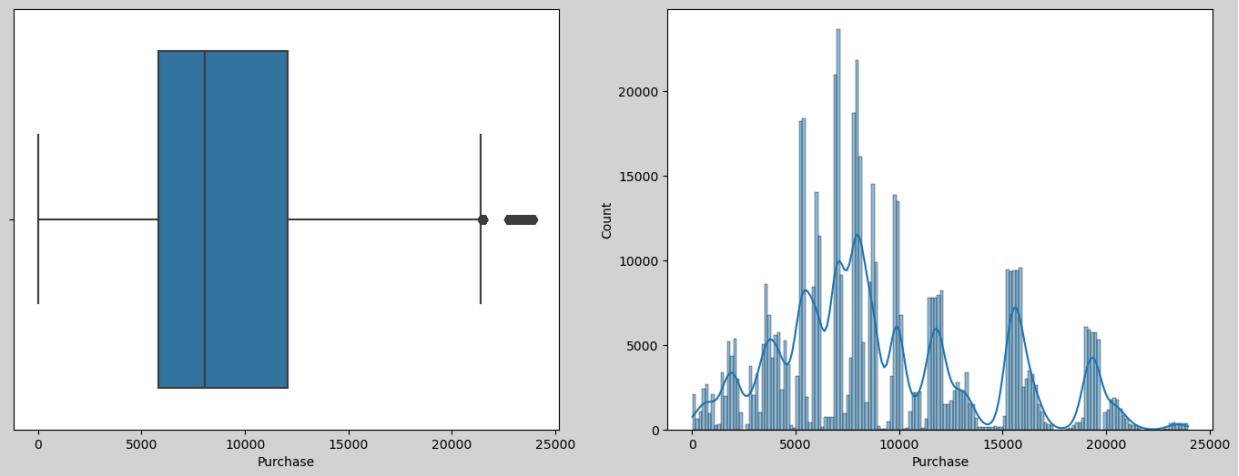
1. The average order value among all buyers is 9263.97.

2. Approximately 50% of the buyers spend around 8047 on average.

3. The lowest order value recorded is as low as 12.

4. The highest order value observed is 23961.

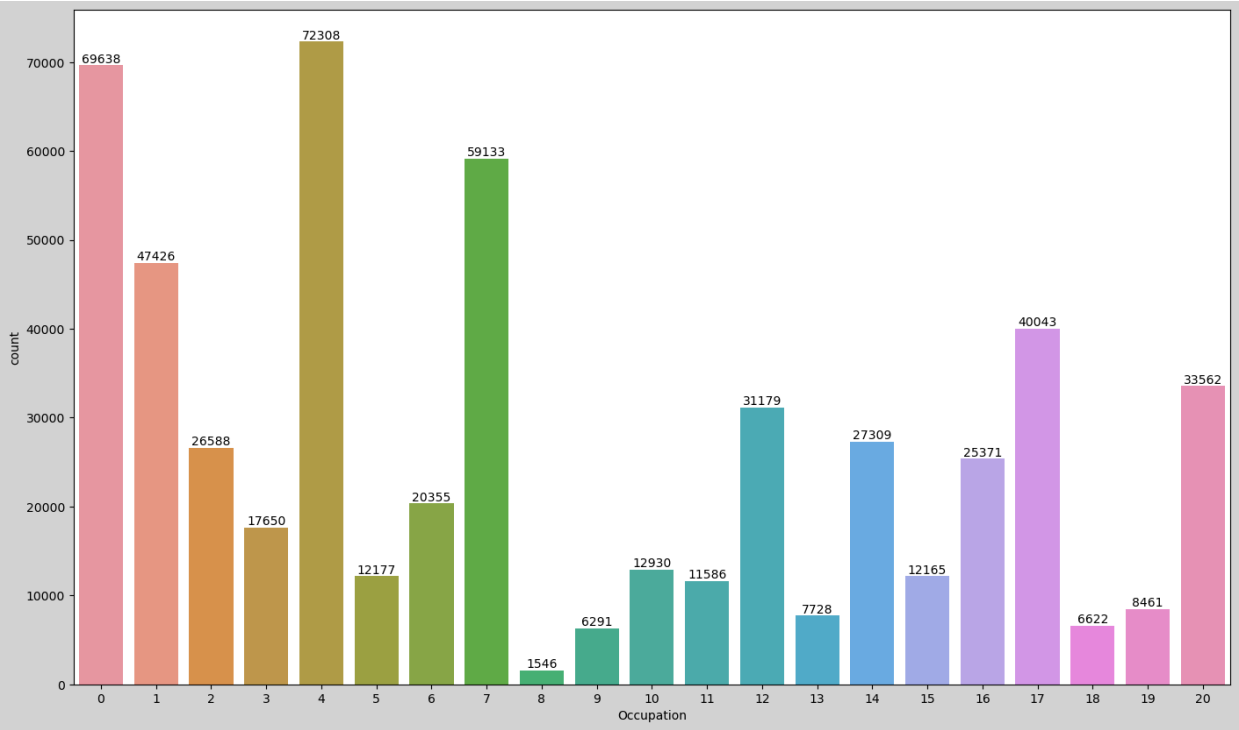
In [23]:
```
plt.figure(figsize=(17, 6)).set_facecolor("lightgrey")
plt.subplot(1,2,1)
sns.boxplot(data=df, x='Purchase', orient='h')
plt.subplot(1,2,2)
sns.histplot(data=df, x='Purchase', kde=True)
plt.show()
```



Upon analyzing the purchase values of the orders, several observations can be made:

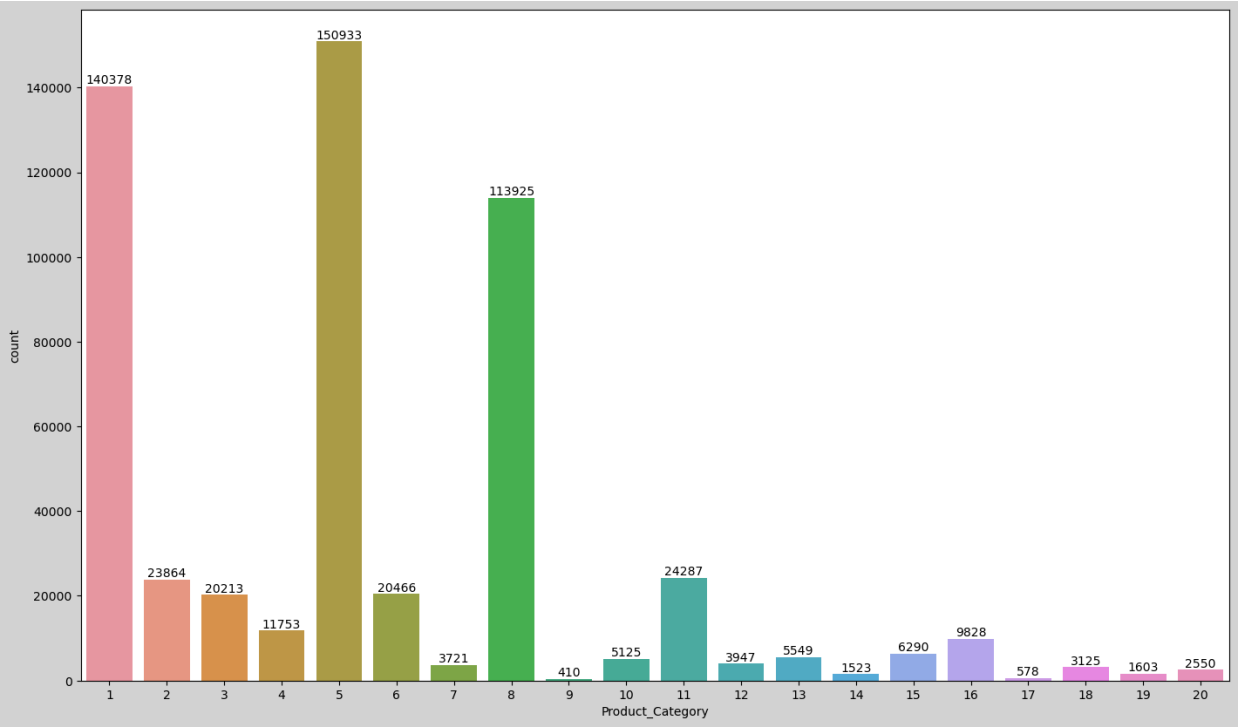
1. The majority of purchase values fall between 6000 and 12000.
2. Most order values are within the range of 5000 to 10000.
3. There are more orders in the range of 15000 to 16000, followed by the range of 11000 to 11500, and a few also fall within the range of 19000 to 20000.

In [24]:
```
plt.figure(figsize=(17, 10)).set_facecolor("lightgrey")
label = sns.countplot(data = df, x='Occupation')
for i in label.containers:
    label.bar_label(i)
```



1. Individuals with occupation code 4 are the most frequent buyers, followed by those with occupation codes 0 and 7.
2. Conversely, individuals with occupation code 8 are the least frequent buyers, followed by those with occupation codes 9 and 18.

In [25]:
```
plt.figure(figsize=(17, 10)).set_facecolor("lightgrey")
label = sns.countplot(data = df, x='Product_Category')
for i in label.containers:
    label.bar_label(i)
```

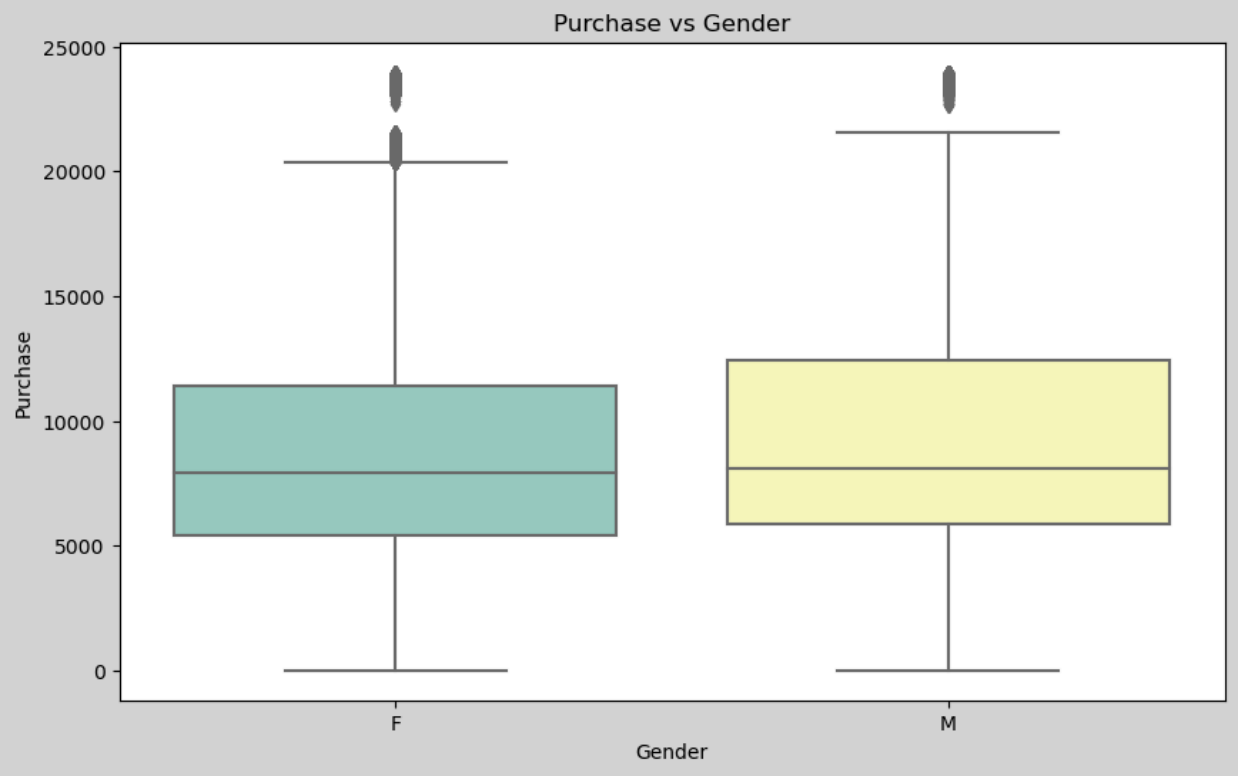


1. The most frequently purchased product category is 5, followed by categories 1 and 8.
2. Other categories are relatively less popular.
3. The least frequently purchased categories are 9, followed by 17 and 14.

Bi-variate Analysis

Lets observe gender while purchase habits.

In [26]:
```
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Gender', palette = 'Set3')
plt.title('Purchase vs Gender')
plt.show()
```



It is evident that males tend to spend more than females.

In [27]:
```
df.groupby(['Gender'])['Purchase'].describe()
```

Out[27]:

Gender	count	mean	std	min	25%	50%	75%	max
F	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
M	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

1. The average order value for males is 9437.

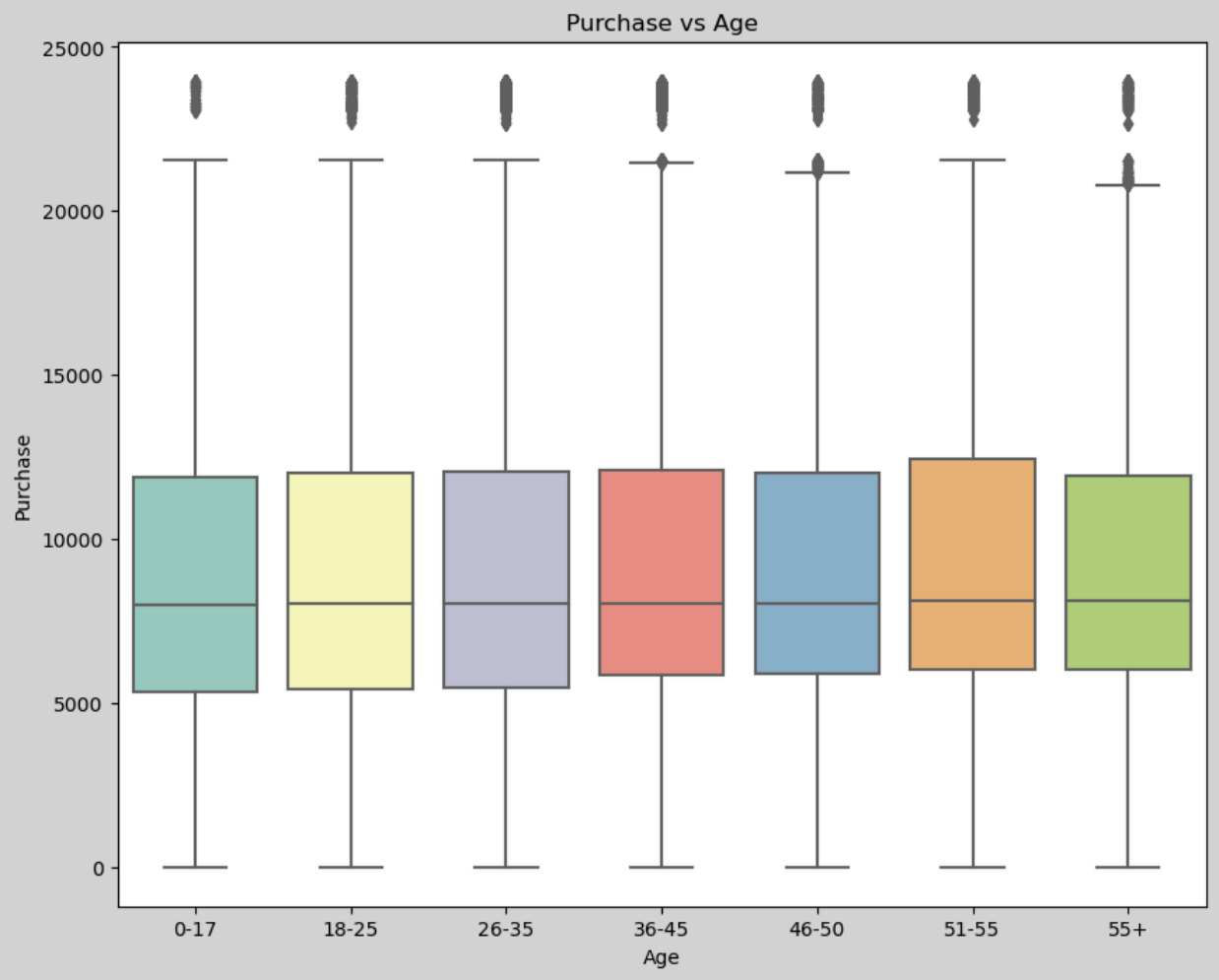
2. For females, the average order value is 8734.

3. Most purchases made by men are around 8098.

4. For females, it is around 7914.

- Let's examine the purchasing habits based on age groups.

In [28]:
```
plt.figure(figsize = (10,8)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Age', palette = 'Set3')
plt.title('Purchase vs Age')
plt.show()
```



There is not a substantial difference observed in the median purchase values across different age groups.

- Let's examine the mean value.

In [29]:
```
df.groupby(['Age'])['Purchase'].describe()
```

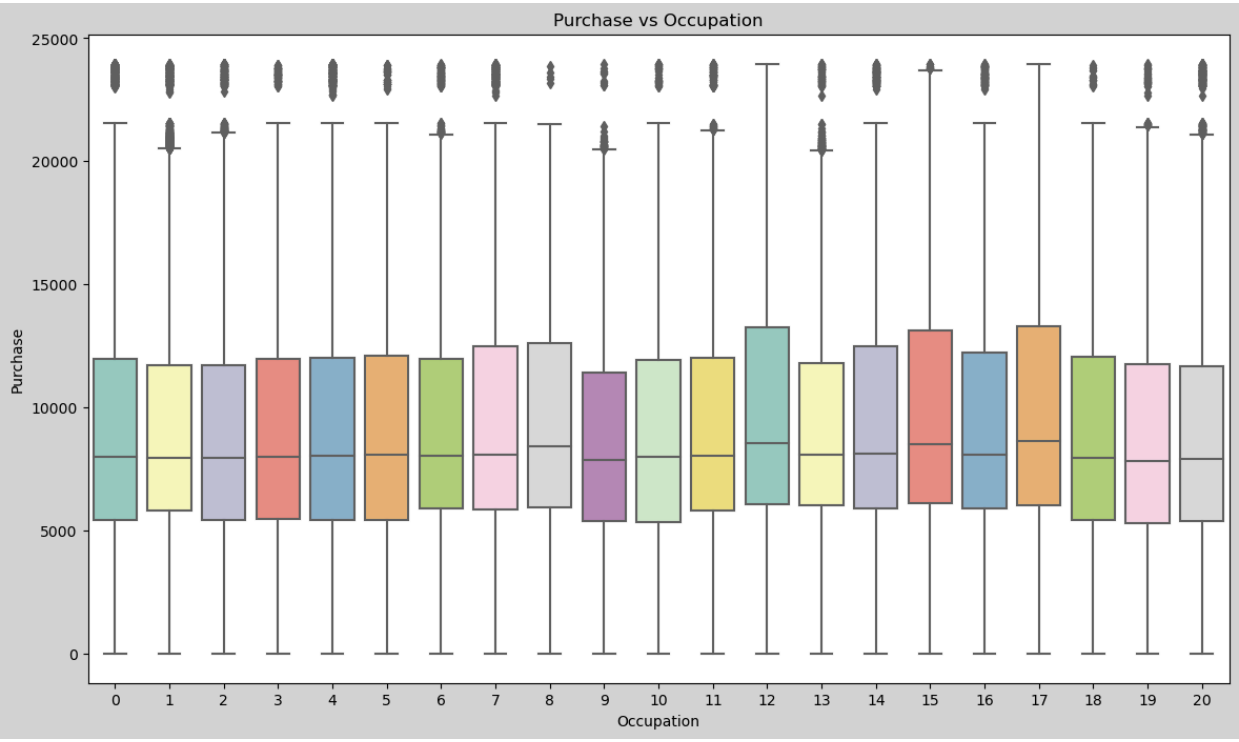
Out[29]:

Age	count	mean	std	min	25%	50%	75%	max
0-17	15102.0	8933.464640	5111.114046	12.0	5328.0	7986.0	11874.0	23955.0
18-25	99660.0	9169.663606	5034.321997	12.0	5415.0	8027.0	12028.0	23958.0
26-35	219587.0	9252.690633	5010.527303	12.0	5475.0	8030.0	12047.0	23961.0
36-45	110013.0	9331.350695	5022.923879	12.0	5876.0	8061.0	12107.0	23960.0
46-50	45701.0	9208.625697	4967.216367	12.0	5888.0	8036.0	11997.0	23960.0
51-55	38501.0	9534.808031	5087.368080	12.0	6017.0	8130.0	12462.0	23960.0
55+	21504.0	9336.280459	5011.493996	12.0	6018.0	8105.5	11932.0	23960.0

1. The highest average order value is observed for the age group 51-55, approximately 9534.
2. Conversely, the lowest average order value is found for the age group 0-17, around 8933.
3. The highest order value across all age groups is approximately 23960.
4. The lowest order value recorded for all age groups is 12.

- Now, let's explore purchase habits based on occupation.

In [30]:
```
plt.figure(figsize = (14,8)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Occupation', palette ='Set3')
plt.title('Purchase vs Occupation')
plt.show()
```



1. The dataset contains numerous outliers.
2. There is not a significant variance observed in the median values.

In [31]:
```
df.groupby(['Occupation'])['Purchase'].describe()
```

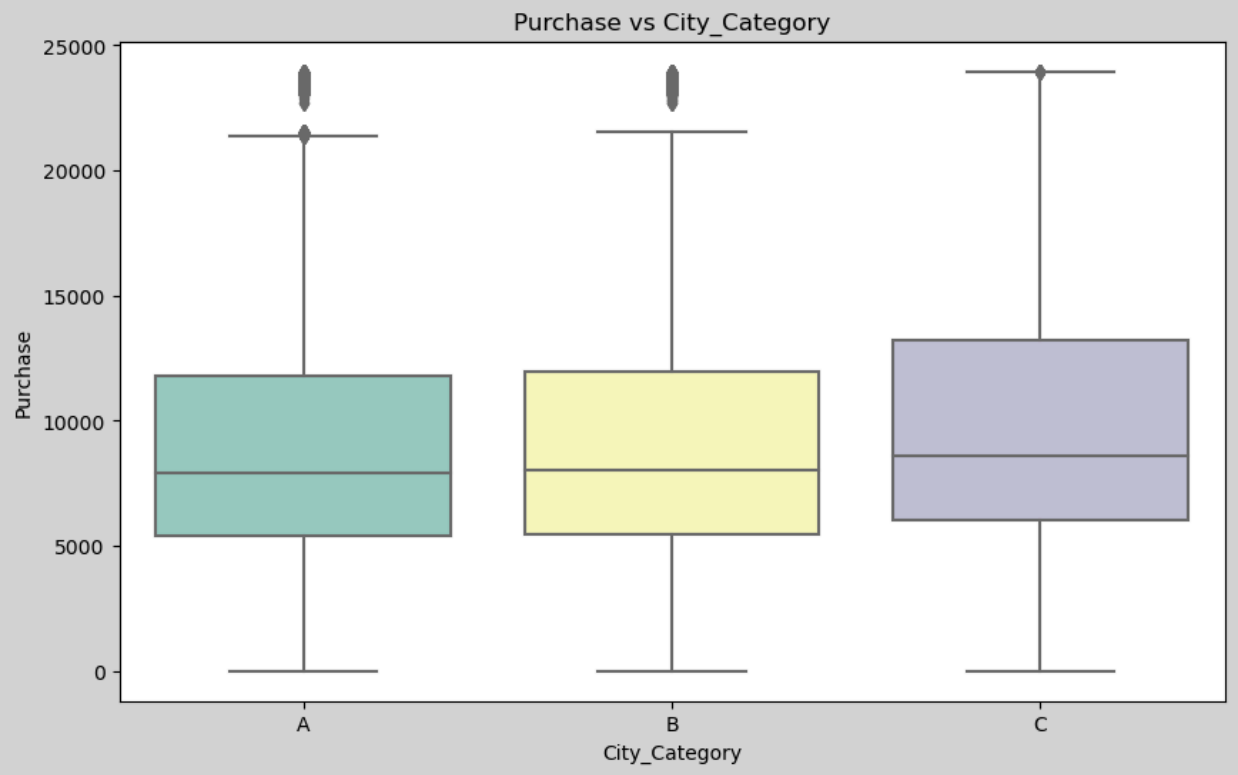
Out[31]:

Occupation	count	mean	std	min	25%	50%	75%	max
0	69638.0	9124.428588	4971.757402	12.0	5445.00	8001.0	11957.00	23961.0
1	47426.0	8953.193270	4838.482159	12.0	5825.00	7966.0	11702.75	23960.0
2	26588.0	8952.481683	4939.418663	12.0	5419.00	7952.0	11718.00	23955.0
3	17650.0	9178.593088	5000.942719	12.0	5478.00	8008.0	11961.00	23914.0
4	72308.0	9213.980251	5043.674855	12.0	5441.75	8043.0	12034.00	23961.0
5	12177.0	9333.149298	5025.616603	12.0	5452.00	8080.0	12091.00	23924.0
6	20355.0	9256.535691	4989.216005	12.0	5888.00	8050.0	11971.50	23951.0
7	59133.0	9425.728223	5086.097089	12.0	5878.00	8069.0	12486.00	23948.0
8	1546.0	9532.592497	4916.641374	14.0	5961.75	8419.5	12607.00	23869.0
9	6291.0	8637.743761	4653.290986	13.0	5403.00	7886.0	11436.00	23943.0
10	12930.0	8959.355375	5124.339999	12.0	5326.25	8012.5	11931.75	23955.0
11	11586.0	9213.845848	5103.802992	12.0	5835.75	8041.5	12010.00	23946.0
12	31179.0	9796.640239	5140.437446	12.0	6054.00	8569.0	13239.00	23960.0
13	7728.0	9306.351061	4940.156591	12.0	6038.00	8090.5	11798.50	23959.0
14	27309.0	9500.702772	5069.600234	12.0	5922.00	8122.0	12508.00	23941.0
15	12165.0	9778.891163	5088.424301	12.0	6109.00	8513.0	13150.00	23949.0
16	25371.0	9394.464349	4995.918117	12.0	5917.00	8070.0	12218.50	23947.0
17	40043.0	9821.478236	5137.024383	12.0	6012.00	8635.0	13292.50	23961.0
18	6622.0	9169.655844	4987.697451	12.0	5420.00	7955.0	12062.75	23894.0
19	8461.0	8710.627231	5024.181000	12.0	5292.00	7840.0	11745.00	23939.0
20	33562.0	8836.494905	4919.662409	12.0	5389.00	7903.5	11677.00	23960.0

1. However, it's notable that occupation 17 has the highest median value.
2. Conversely, occupation 19 exhibits the lowest median value.
3. Occupation 17 demonstrates the highest average order values compared to other occupations, reaching 9821.
4. On the other hand, occupation 9 shows the lowest average order value, which is 8637.

- Now, let's examine the purchasing habits based on cities.

In [32]:
```
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'City_Category', palette ='Set3')
plt.title('Purchase vs City_Category')
plt.show()
```



1. City Category C exhibits the highest median value, followed by City B and City A.
2. There are several outliers observed for City A and City B.

In [33]:
```
df.groupby(['City_Category'])['Purchase'].describe()
```

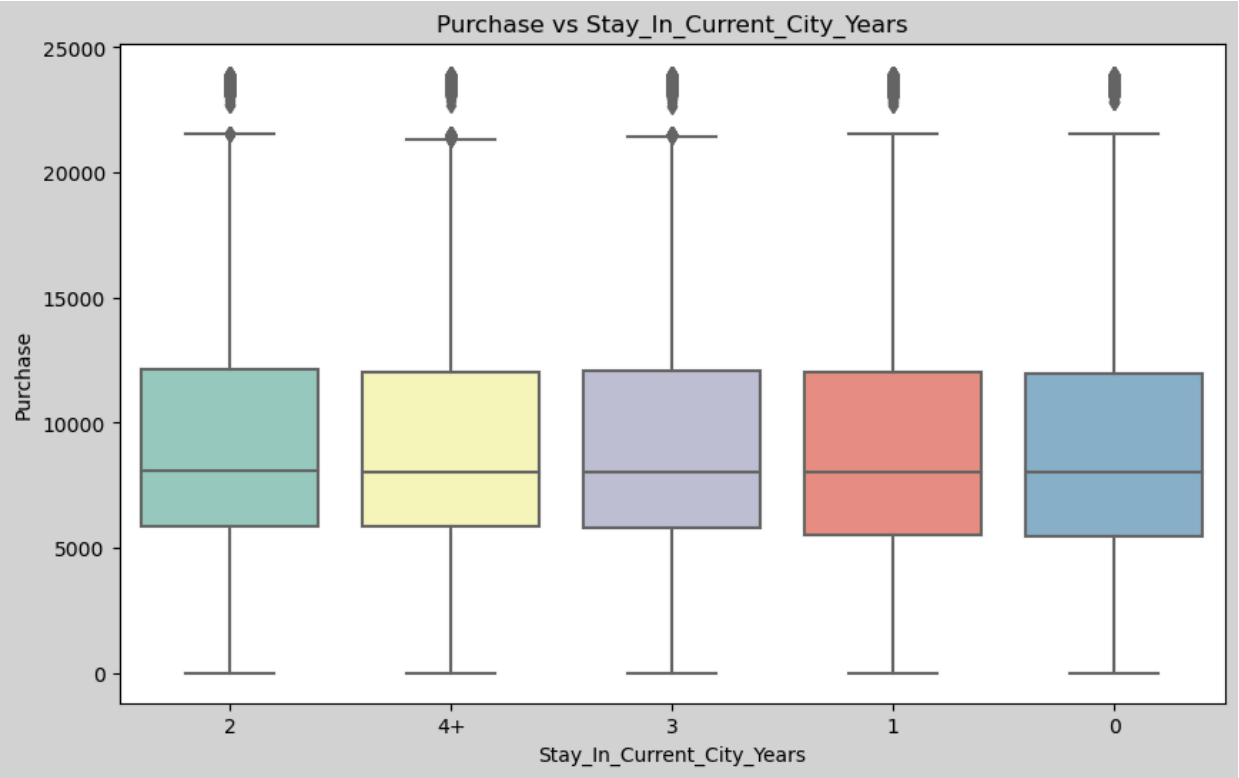
Out[33]:

City_Category	count	mean	std	min	25%	50%	75%	max
A	147720.0	8911.939216	4892.115238	12.0	5403.0	7931.0	11786.0	23961.0
B	231173.0	9151.300563	4955.496566	12.0	5460.0	8005.0	11986.0	23960.0
C	171175.0	9719.920993	5189.465121	12.0	6031.5	8585.0	13197.0	23961.0

Additionally, it's notable that the mean value per order is highest for City C, followed by City B and then City A.

- Let's investigate whether the duration of a person's stay in a city influences their purchasing habits or not.

In [34]:
```
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x ='Stay_In_Current_City_Years', palette = 'Set3
plt.title('Purchase vs Stay_In_Current_City_Years')
plt.show()
```



It appears that the median value remains consistent across all the years.

In [35]:
```
df.groupby(['Stay_In_Current_City_Years'])['Purchase'].describe()
```

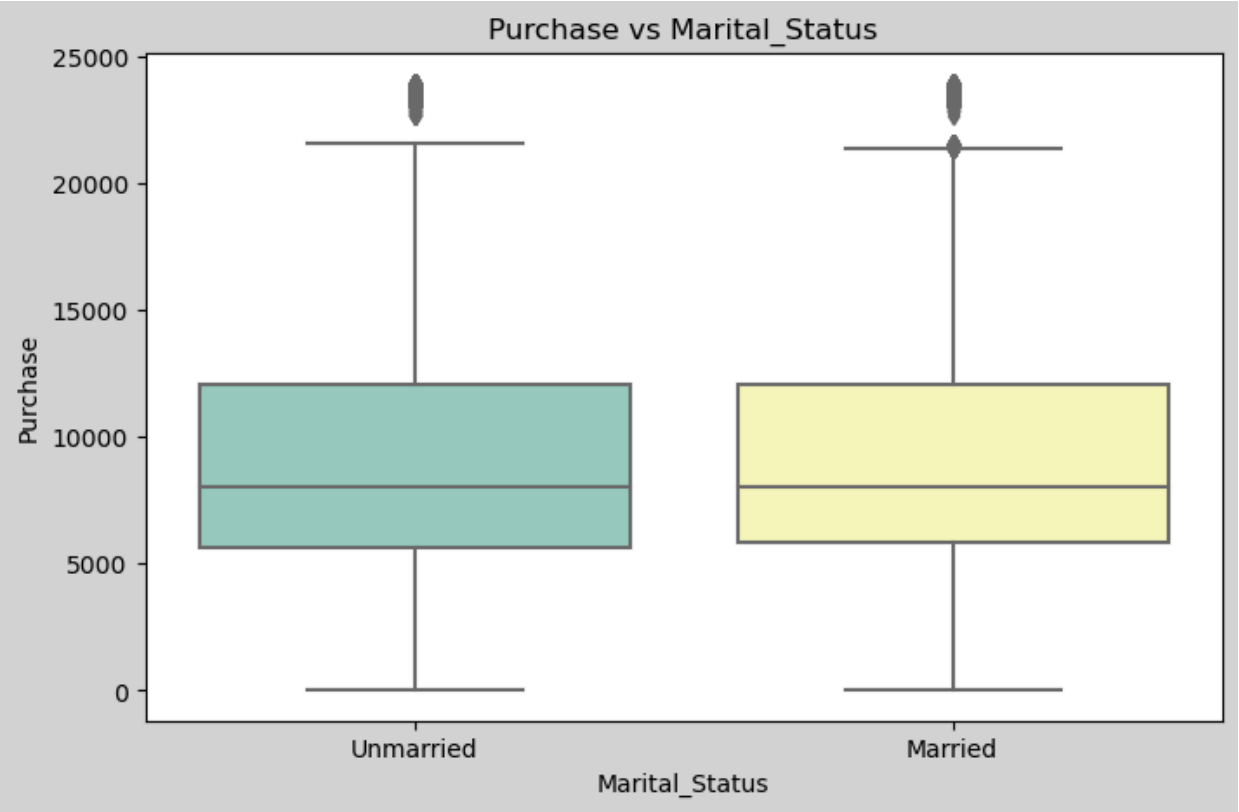
Out[35]:

Stay_In_Current_City_Years	count	mean	std	min	25%	50%	75%	max
0	74398.0	9180.075123	4990.479940	12.0	5480.0	8025.0	11990.0	23960.0
1	193821.0	9250.145923	5027.476933	12.0	5500.0	8041.0	12042.0	23961.0
2	101838.0	9320.429810	5044.588224	12.0	5846.0	8072.0	12117.0	23961.0
3	95285.0	9286.904119	5020.343541	12.0	5832.0	8047.0	12075.0	23961.0
4+	84726.0	9275.598872	5017.627594	12.0	5844.0	8052.0	12038.0	23958.0

1. Additionally, the average order value remains relatively consistent, ranging from 9180 to 9286.
2. Another observation is that the highest order value remains constant across all the years.

- Let's examine whether marital status influences a person's spending habits.

In [36]:
```
plt.figure(figsize = (8,5)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Marital_Status', palette ='Set3')
plt.title('Purchase vs Marital_Status')
plt.show()
```



It appears that the median value remains consistent.

- Let's examine the minimum, maximum, and average order values.

In [37]:
```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

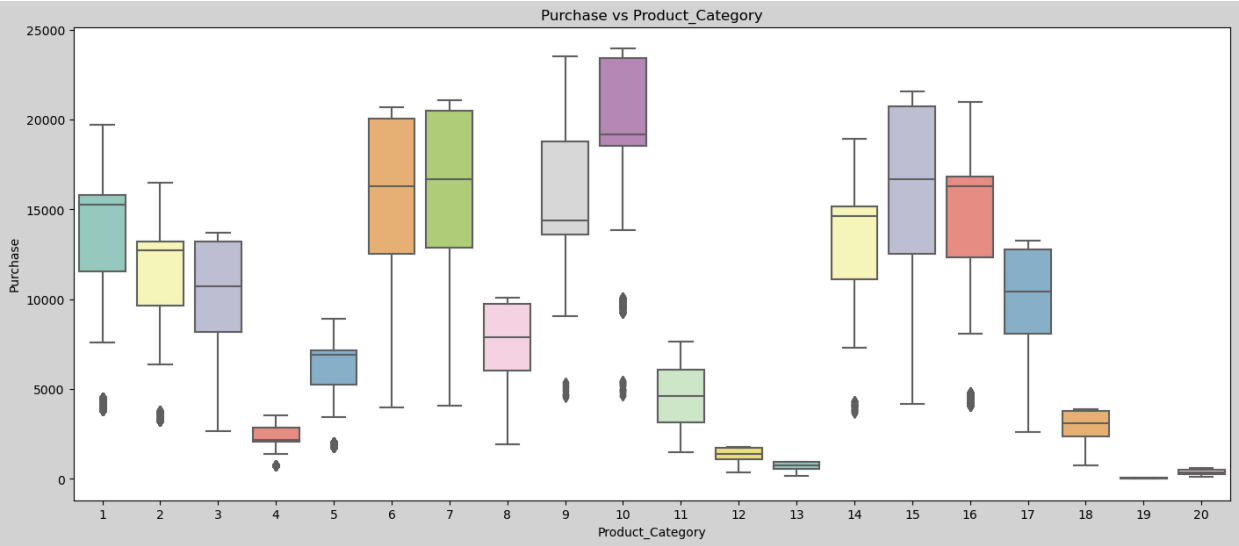
Out[37]:

Marital_Status	count	mean	std	min	25%	50%	75%	max
Unmarried	324731.0	9265.907619	5027.347859	12.0	5605.0	8044.0	12061.0	23961.0
Married	225337.0	9261.174574	5016.897378	12.0	5843.0	8051.0	12042.0	23961.0

1. The minimum and maximum order values are identical for both types of individuals.
2. Additionally, the average order value is also nearly identical for both.

- Let's analyze which product categories people spend more or less on.

In [38]:
```
plt.figure(figsize = (17,7)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Product_Category', palette = 'Set3')
plt.title('Purchase vs Product_Category')
plt.show()
```



Significant differences in the median values across all product categories are evident.

In [39]:
```
df.groupby(['Product_Category'])['Purchase'].describe()
```

Out[39]:

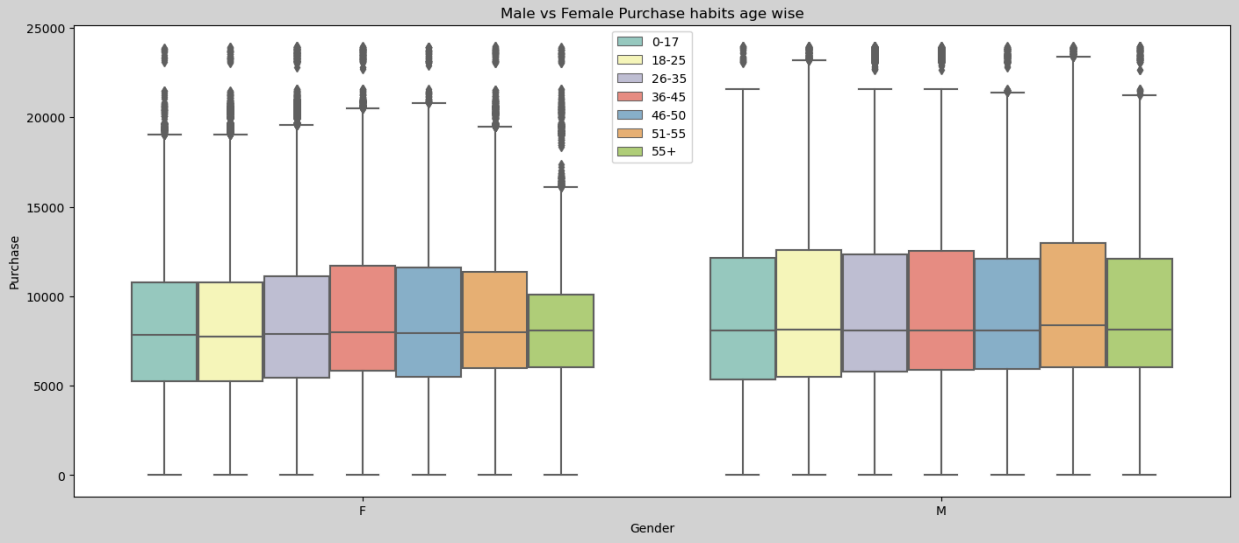
Product_Category	count	mean	std	min	25%	50%	75%	max
1	140378.0	13606.218596	4298.834894	3790.0	11546.00	15245.0	15812.00	19708.0
2	23864.0	11251.935384	3570.642713	3176.0	9645.75	12728.5	13212.00	16504.0
3	20213.0	10096.705734	2824.626957	2638.0	8198.00	10742.0	13211.00	13717.0
4	11753.0	2329.659491	812.540292	684.0	2058.00	2175.0	2837.00	3556.0
5	150933.0	6240.088178	1909.091687	1713.0	5242.00	6912.0	7156.00	8907.0
6	20466.0	15838.478550	4011.233690	3981.0	12505.00	16312.0	20051.00	20690.0
7	3721.0	16365.689600	4174.554105	4061.0	12848.00	16700.0	20486.00	21080.0
8	113925.0	7498.958078	2013.015062	1939.0	6036.00	7905.0	9722.00	10082.0
9	410.0	15537.375610	5330.847116	4528.0	13583.50	14388.5	18764.00	23531.0
10	5125.0	19675.570927	4225.721898	4624.0	18546.00	19197.0	23438.00	23961.0
11	24287.0	4685.268456	1834.901184	1472.0	3131.00	4611.0	6058.00	7654.0
12	3947.0	1350.859894	362.510258	342.0	1071.00	1401.0	1723.00	1778.0
13	5549.0	722.400613	183.493126	185.0	578.00	755.0	927.00	962.0
14	1523.0	13141.625739	4069.009293	3657.0	11097.00	14654.0	15176.50	18931.0
15	6290.0	14780.451828	5175.465852	4148.0	12523.25	16660.0	20745.75	21569.0
16	9828.0	14766.037037	4360.213198	4036.0	12354.00	16292.5	16831.00	20971.0
17	578.0	10170.759516	2333.993073	2616.0	8063.50	10435.5	12776.75	13264.0
18	3125.0	2972.864320	727.051652	754.0	2359.00	3071.0	3769.00	3900.0
19	1603.0	37.041797	16.869148	12.0	24.00	37.0	50.00	62.0
20	2550.0	370.481176	167.116975	118.0	242.00	368.0	490.00	613.0

1. The highest median value is observed for product category 10, reaching 19197.
2. Conversely, the lowest median value is found for product category 19, at only 37.
3. The average order value for category 10 is the highest, standing at 19675.
4. Meanwhile, the average order value for category 19 is also the lowest, at 37.
5. It is evident that category 19 is the least preferred or least frequently bought product category.

Multi-variate Analysis

- Let's conduct a multivariate analysis to compare the purchase habits of males and females across different age groups.

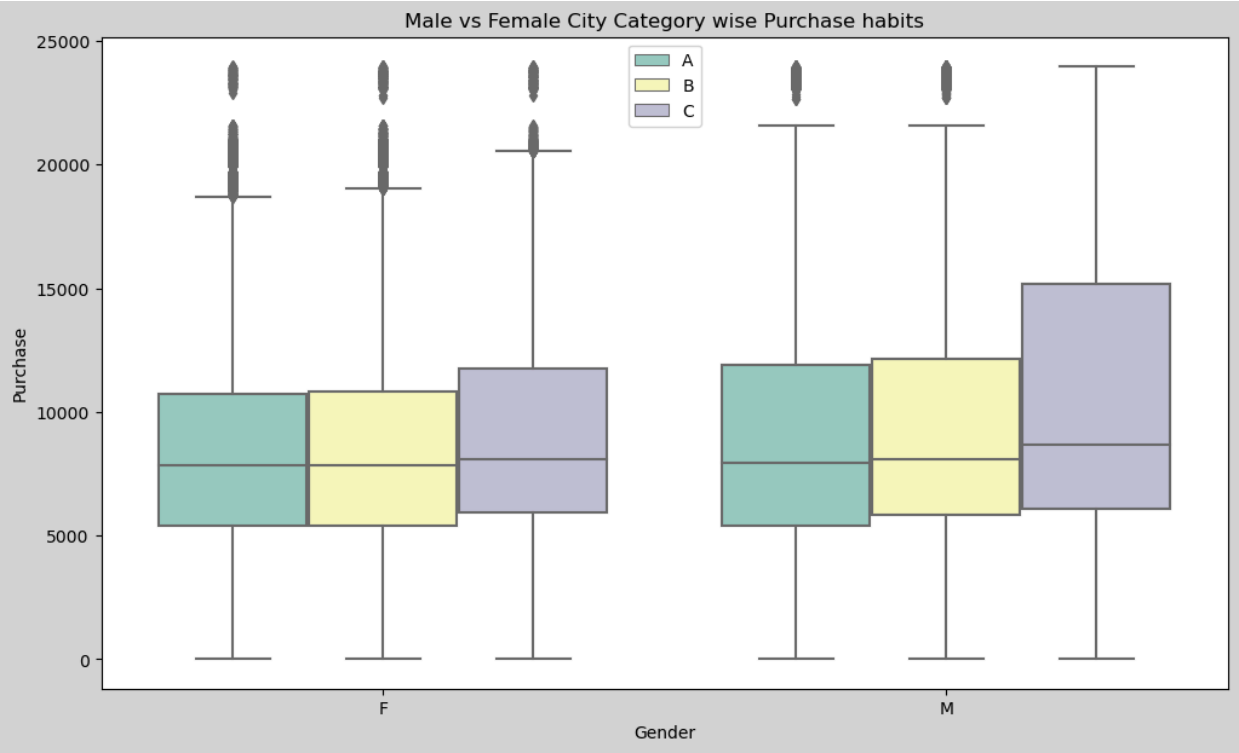
In [40]:
```
plt.figure(figsize = (17,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age',palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Purchase habits age wise')
plt.show()
```



1. The median values for females aged 18-25 are the lowest, while they remain relatively consistent for other age groups.
2. The median values for all age categories are nearly the same and reach their peak for the 51-55 age group.

- Let's examine the purchasing habits of males versus females across different cities.

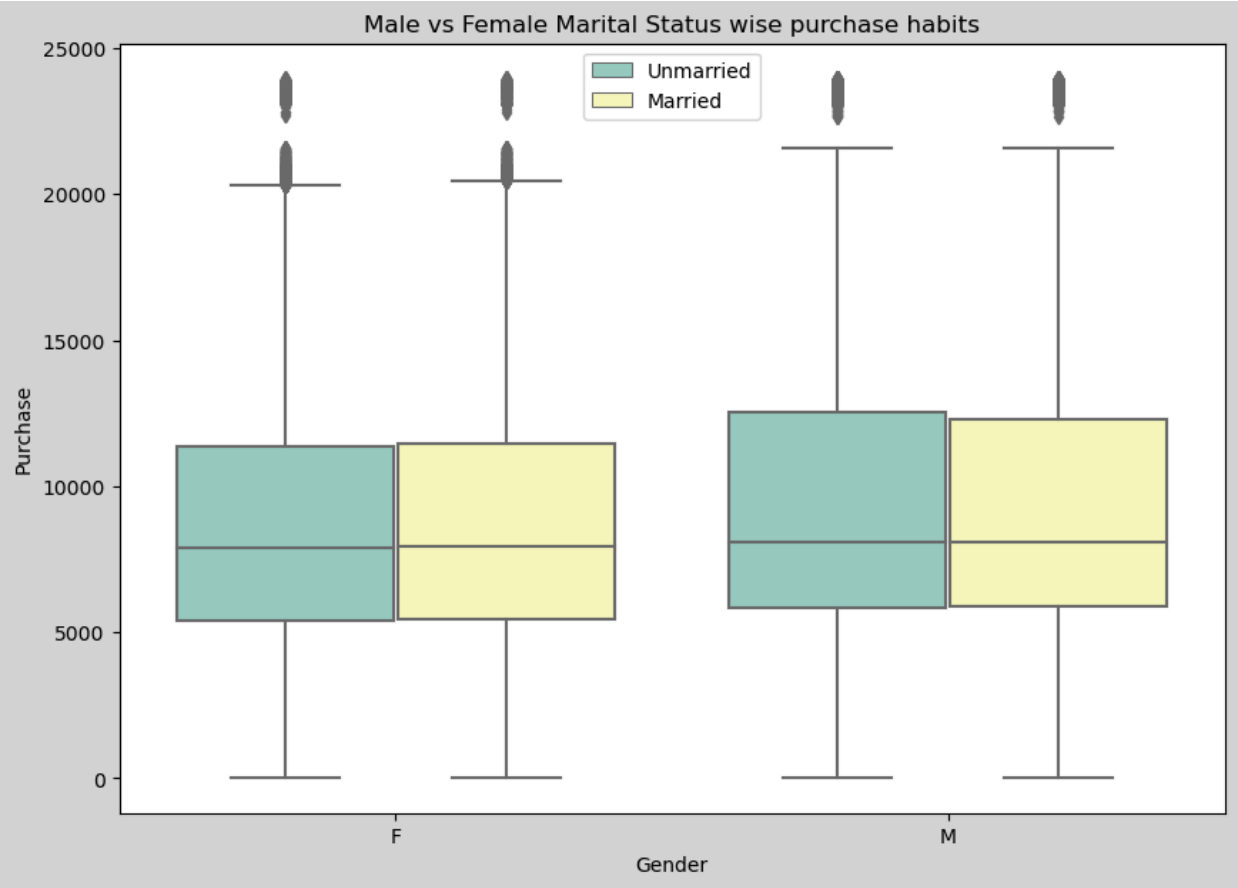
In [41]:
```
plt.figure(figsize = (12,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category',palette='Set3')
plt.legend(loc=9)
plt.title("Male vs Female City Category wise Purchase habits")
plt.show()
```



1. The median value for females in City Category C is the highest compared to City Category A and B.
2. Similarly, the median value for males in City Category C is also the highest compared to City Category A and B.

- Let's explore the purchasing habits of males versus females based on marital status.

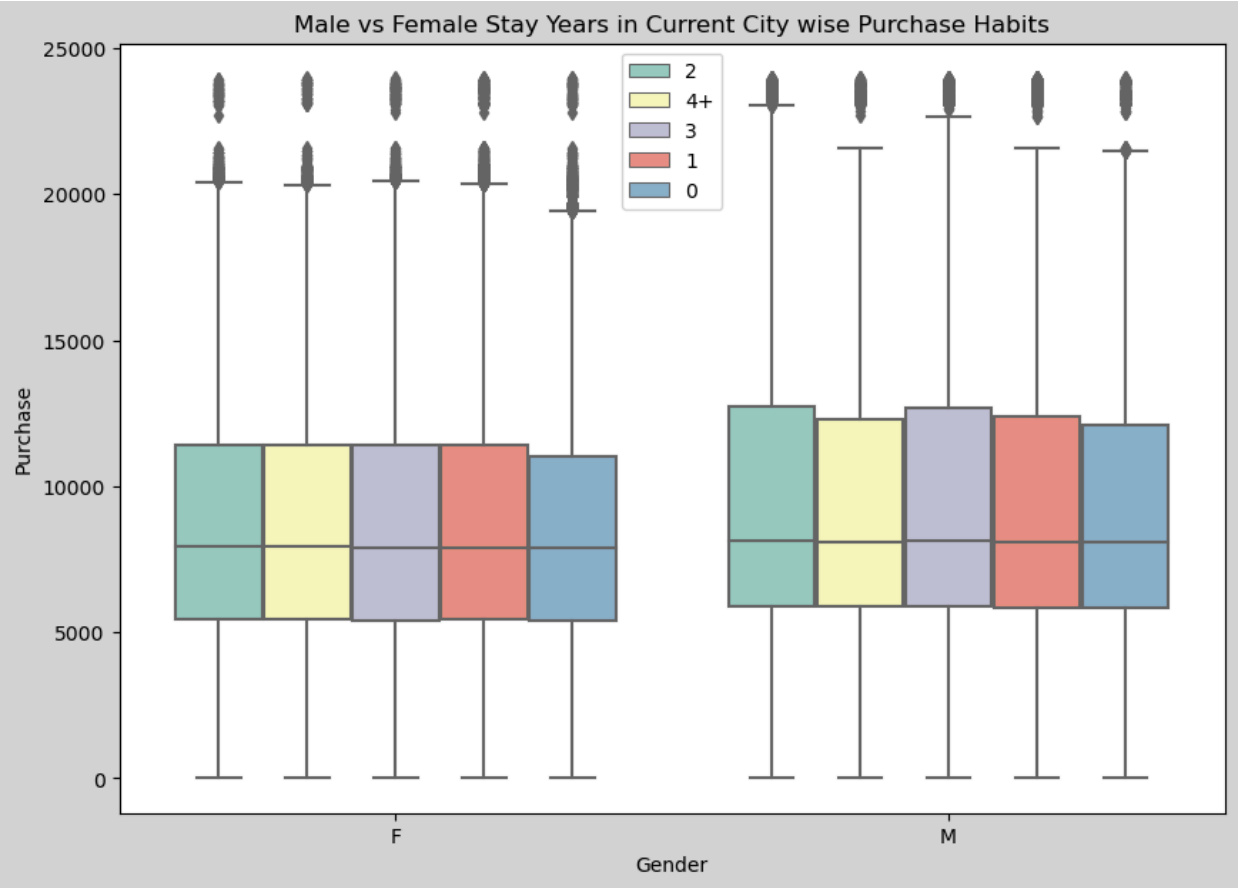
In [42]:
```
plt.figure(figsize = (10,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status',palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Marital Status wise purchase habits')
plt.show()
```



1. There is no discernible effect of marital status on the spending habits of both genders.
2. However, it is notable that the median values for males are higher compared to females.

- Let's analyze the purchasing habits of males versus females based on the duration of stay in their current city.

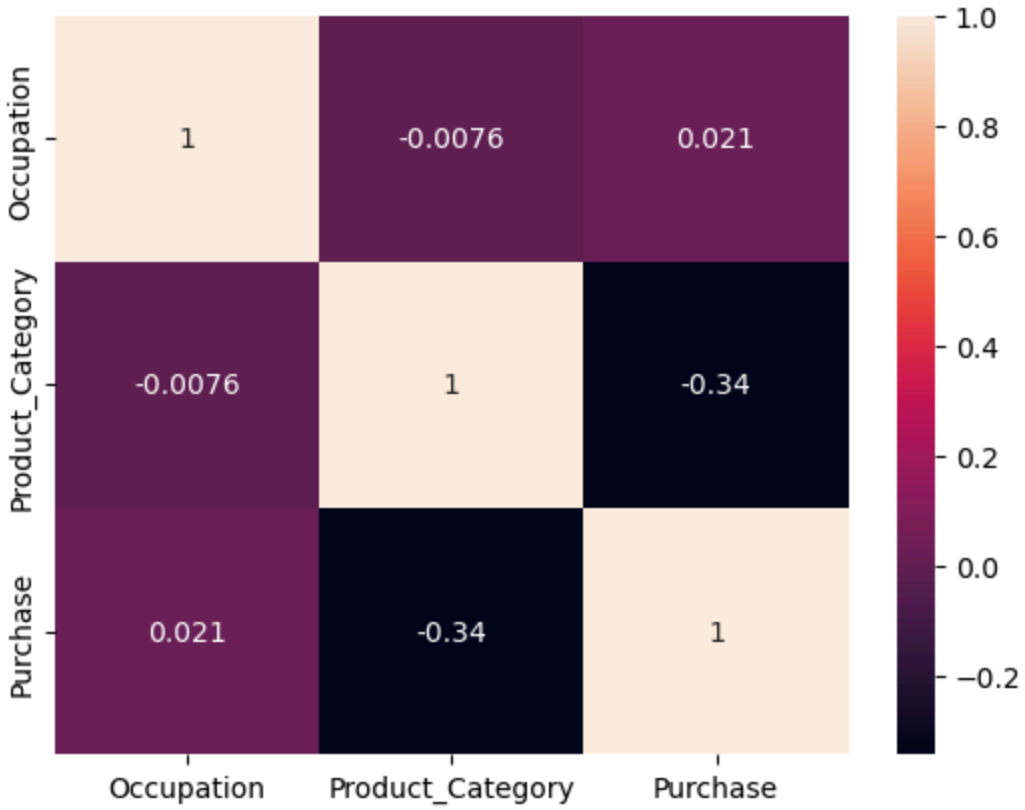
In [43]:
```
plt.figure(figsize = (10,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender',hue='Stay_In_Current_City_Years', palett
plt.legend(loc=9)
plt.title('Male vs Female Stay Years in Current City wise Purchase Habits')
plt.show()
```



1. It can be observed that for females, the median values for purchase amounts are slightly lower for those staying for 3 and 0 years compared to others.
2. However, for men, there isn't much difference noted.

- Let's examine the correlation among the numerical values in the dataset.

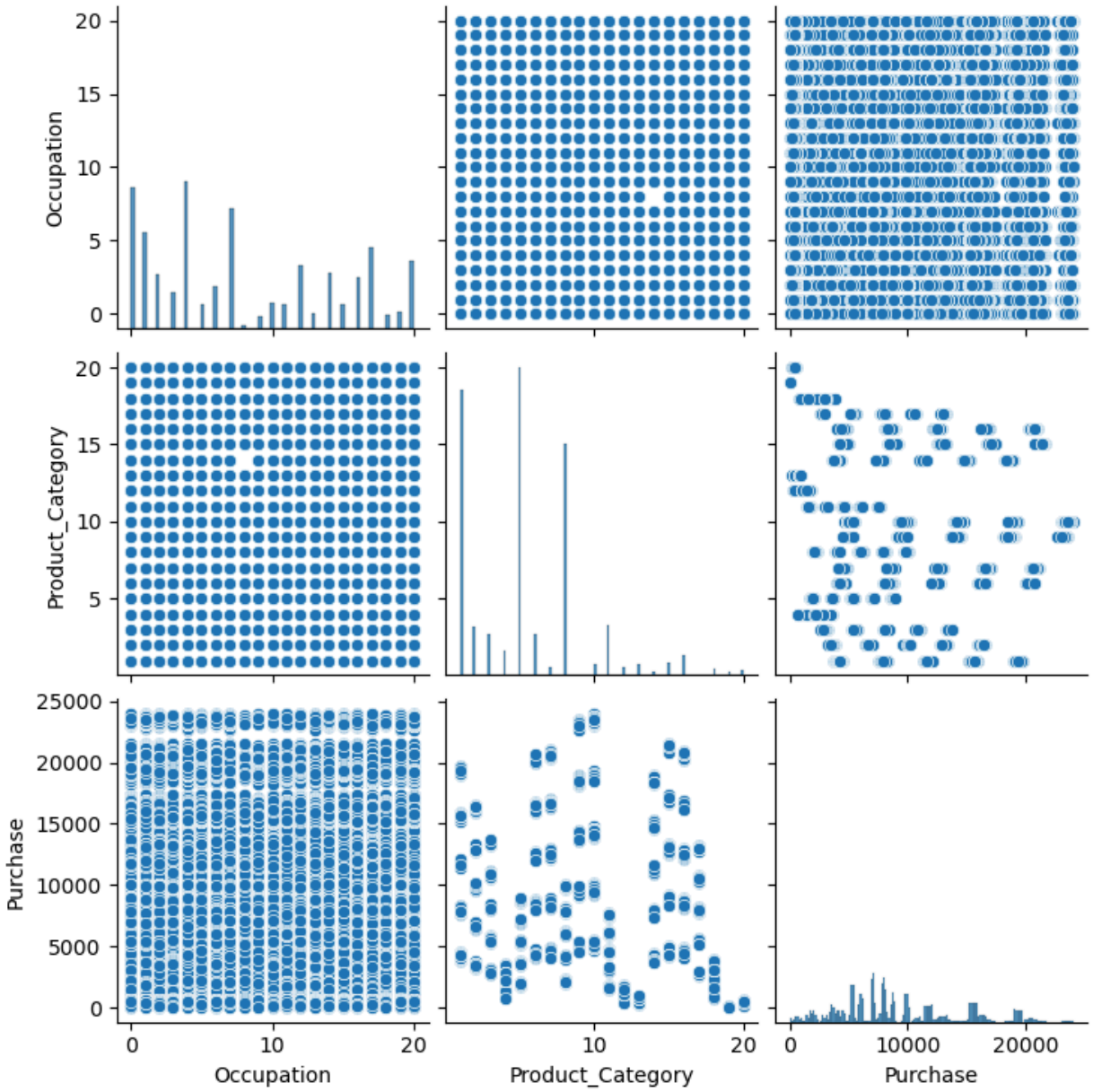
In [44]:
```
sns.heatmap(df.corr(), annot = True)
plt.show()
```



1. There is a high negative correlation (-0.0076) between Product Category and Occupation.
2. A slight positive correlation (0.021) is observed between Purchase and Occupation.
3. There exists a negative correlation (-0.34) between Product Category and Purchase.

- Let's create a pairplot to visualize the relationships between the columns.

In [45]:
```
sns.pairplot(df)
plt.show()
```

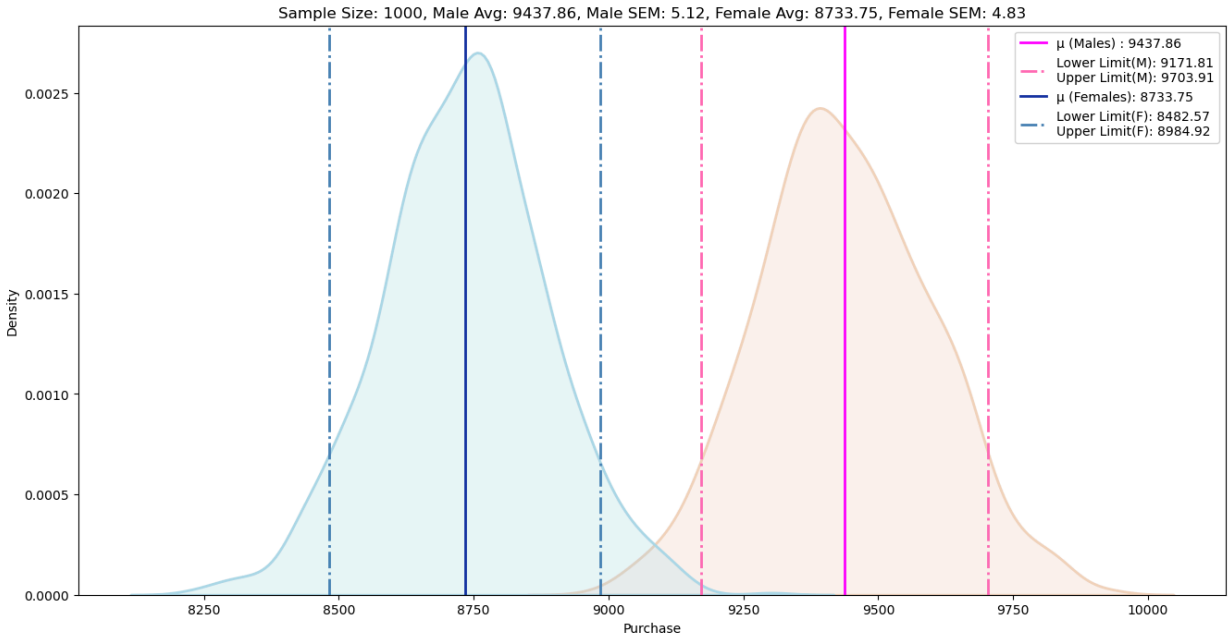


Central Limit Theorom(CLT)

In [46]:
```
def bootstrap(sample1, sample2, sample_size, itr_size=1000, ci=90):
    ci = ci/100

    # Bootstrap resampling
    sample1_n = [np.mean(sample1.sample(sample_size)) for i in range(itr_size)]
    sample2_n = [np.mean(sample2.sample(sample_size)) for i in range(itr_size)]

    # For Sample1's means
    mean1 = np.mean(sample1_n)
    sigma1 = np.std(sample1_n)
    sem1 = stats.sem(sample1_n)
    lower_limit_1 = norm.ppf((1-ci)/2) * sigma1 + mean1
    upper_limit_1 = norm.ppf(ci+(1-ci)/2) * sigma1 + mean1

    # For Sample2's means
    mean2 = np.mean(sample2_n)
```

```
    sigma2 = np.std(sample2_n)
    sem2 = stats.sem(sample2_n)
    lower_limit_2 = norm.ppf((1-ci)/2) * sigma2 + mean2
    upper_limit_2 = norm.ppf(ci + (1-ci)/2) * sigma2 + mean2

    # Plotting
    plt.figure(figsize=(16,8))

    # KDE plot for Sample1
    sns.kdeplot(data=sample1_n, color="#F2D2BD", fill=True, linewidth=2)
    label_mean1 = "μ (Males) : {:.2f}".format(mean1)
    plt.axvline(mean1, color='#FF00FF', linestyle='solid', linewidth=2, label=label_me
    label_limits1 = "Lower Limit(M): {:.2f}\nUpper Limit(M): {:.2f}".format(lower_limi
    plt.axvline(lower_limit_1, color='#FF69B4', linestyle='dashdot', linewidth=2, labe
    plt.axvline(upper_limit_1, color='#FF69B4', linestyle='dashdot', linewidth=2)

    # KDE plot for Sample2
    sns.kdeplot(data=sample2_n, color='#ADD8E6', fill=True, linewidth=2)
    label_mean2 = "μ (Females): {:.2f}".format(mean2)
    plt.axvline(mean2, color='#1434A4', linestyle='solid', linewidth=2, label=label_me
    label_limits2 = "Lower Limit(F): {:.2f}\nUpper Limit(F): {:.2f}".format(lower_limi
    plt.axvline(lower_limit_2, color='#4682B4', linestyle='dashdot', linewidth=2, labe
    plt.axvline(upper_limit_2, color='#4682B4', linestyle='dashdot', linewidth=2)

    # Plot titles and labels
    plt.title(f"Sample Size: {sample_size}, Male Avg: {np.round(mean1, 2)}, Male SEM:
              f"Female Avg: {np.round(mean2, 2)}, Female SEM: {np.round(sem2, 2)}")
    plt.legend(loc='upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')

    # Return relevant statistics
    return round(mean1, 2), round(mean2, 2), round(lower_limit_1, 2), round(upper_limi
           round(lower_limit_2, 2), round(upper_limit_2, 2)

# Sample data
df_male = df[df['Gender'] == 'M']
df_female = df[df['Gender'] == 'F']

# Call bootstrap function with sample data and other parameters
bootstrap(df_male['Purchase'], df_female['Purchase'], sample_size=1000, itr_size=1000,
```

Out[46]: (9437.86, 8733.75, 9171.81, 9703.91, 8482.57, 8984.92)

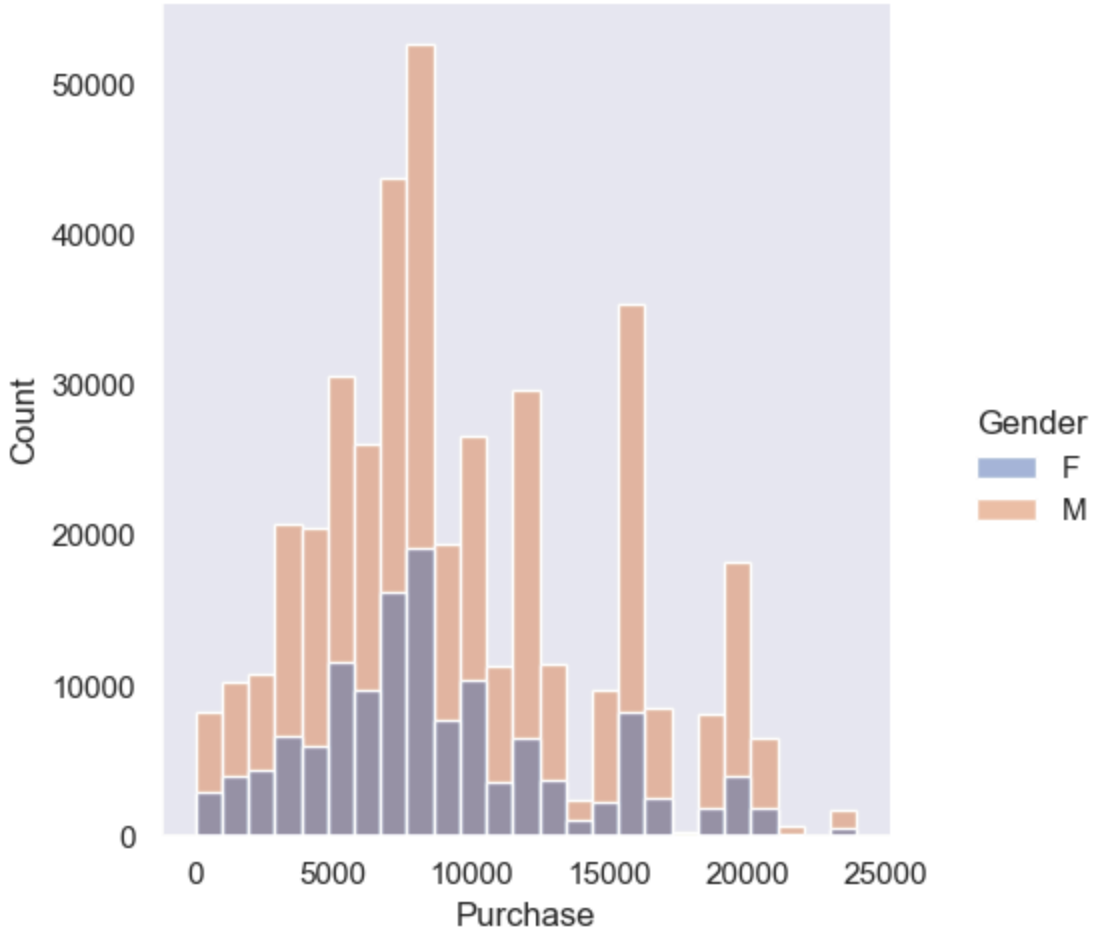


1. There is a marginal difference in average purchase amounts between males and females.
2. Median purchase amounts for males and females across different age groups are similar, with females aged 18-25 showing slightly lower values.
3. Median purchase values for both genders are highest in City Category C compared to A and B.
4. Marital status does not significantly influence spending habits for both genders.
5. However, median purchase values are generally higher for males than females.
6. For females, median purchase values are slightly lower for those staying for 3 and 0 years compared to other durations.
7. Variation in median purchase values based on duration of stay is minimal for males.
8. There is a strong negative correlation between Product Category and Occupation, indicating occupational preferences for specific product categories.
9. A slight positive correlation is observed between Purchase and Occupation, implying some occupational influence on purchase behavior.
10. A negative correlation exists between Product Category and Purchase, suggesting varied spending patterns across product categories.

- Male Vs Female Purchase Values.

In [47]:
```
plt.figure(figsize=(12,8))
sns.set(style='dark')
sns.displot(x= 'Purchase',data=df,hue='Gender',bins=25)
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



It is evident that males spend more than females.

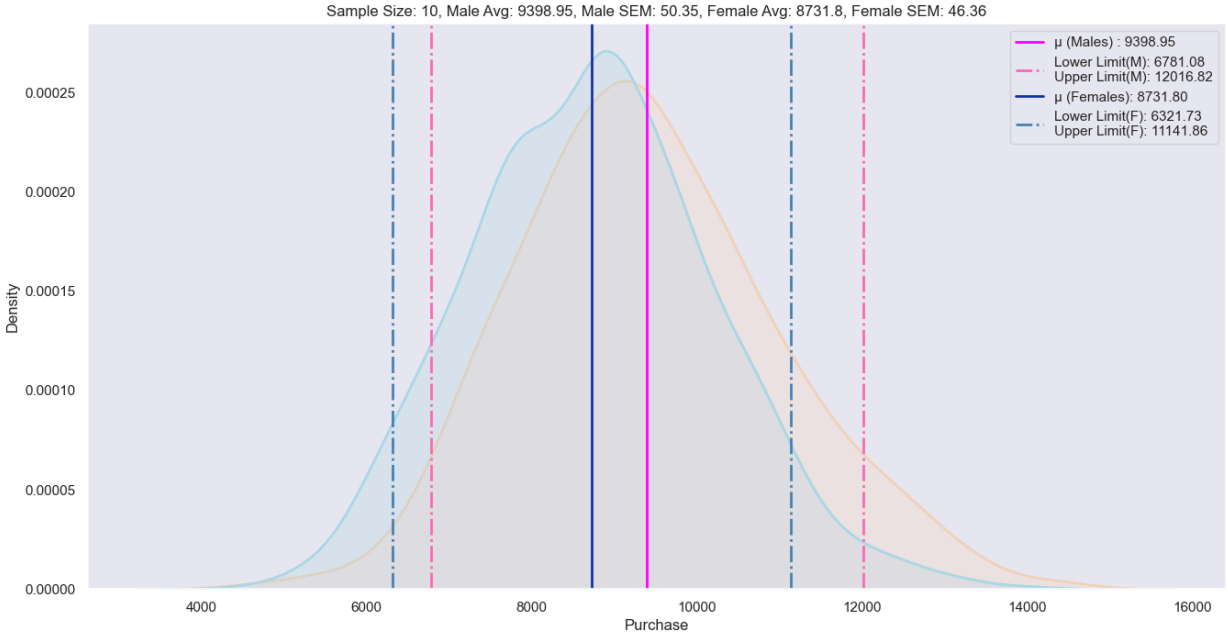
In [48]:
```
df.groupby(['Gender'])['Purchase'].describe()
```

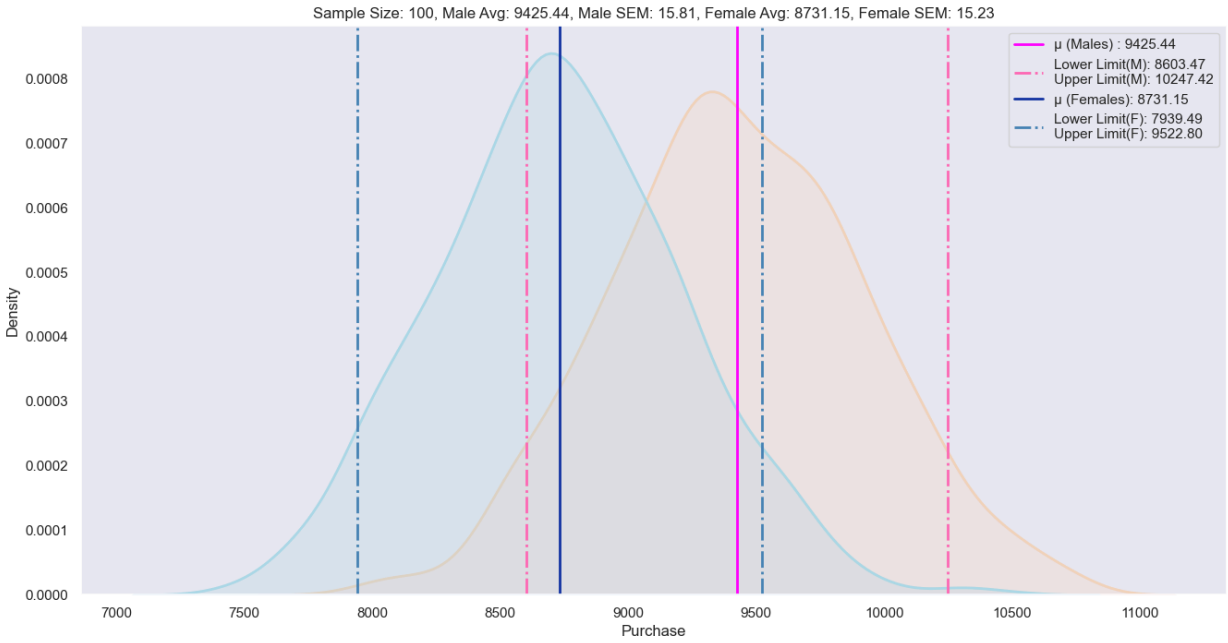
Out[48]:

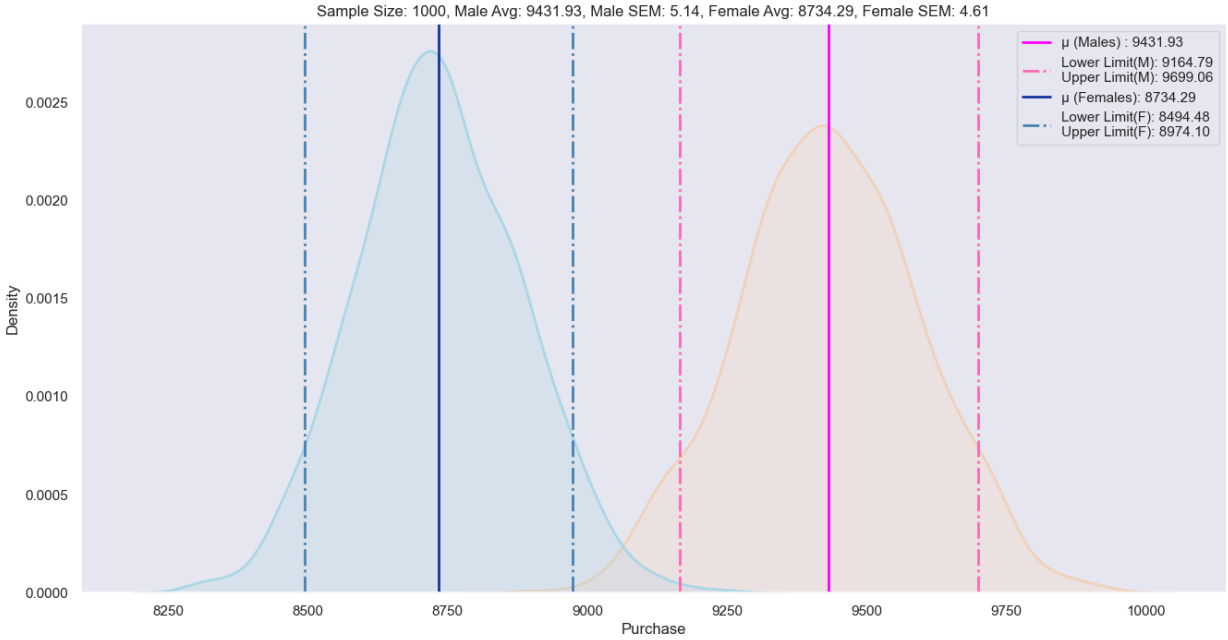
Gender	count	mean	std	min	25%	50%	75%	max
F	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
M	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

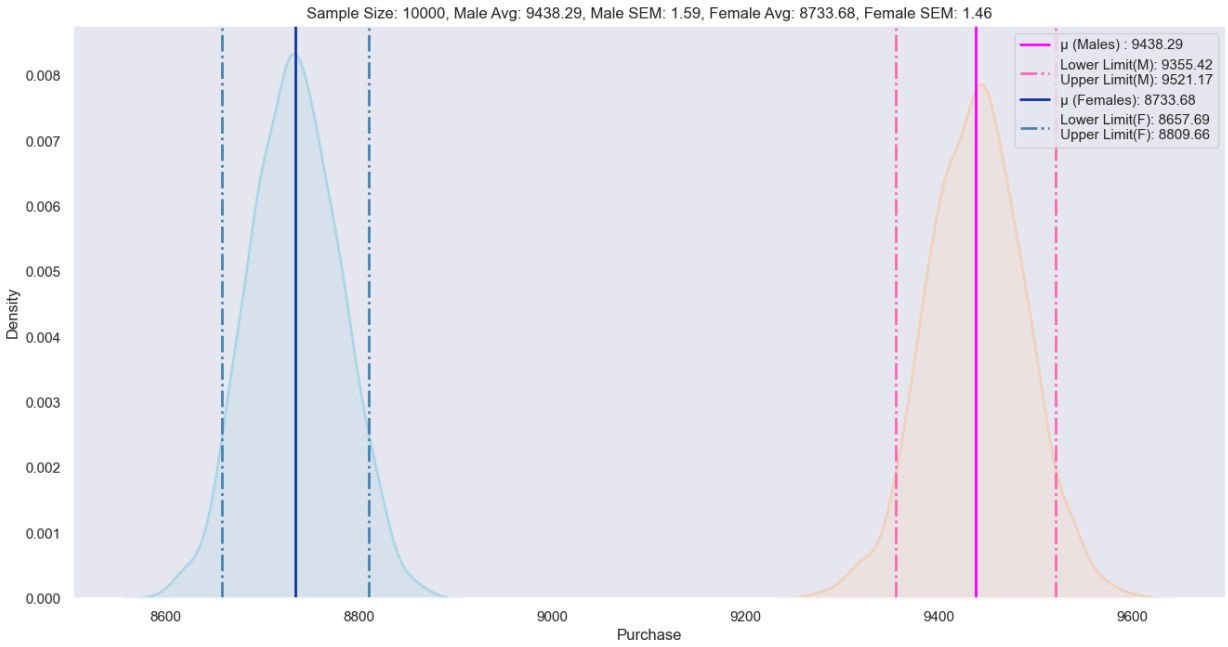
- Let's generate plots showing the mean of 1000 random samples for different sample sizes (10, 100, 1000, 10000, and 100000) with a 90% confidence interval.

In [49]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 90
itr_size = 1000
res = pd.DataFrame(columns = ['Gender','Sample Size','Lower Limit','Upper Limit','Samp
for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f = bootstrap(df_male['Purchase'],df_female['Pu
    res = res.append({'Gender':'M','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul
    res = res.append({'Gender':'F','Sample Size':i,'Lower Limit':ll_f,'Upper Limit':ul
```


Sample Size: 10, Male Avg: 9398.95, Male SEM: 50.35, Female Avg: 8731.8, Female SEM: 46.36
μ (Males) : 9398.95
Lower Limit(M): 6781.08
Upper Limit(M): 12016.82
μ (Females): 8731.80
Lower Limit(F): 6321.73
Upper Limit(F): 11141.86
Density
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
4000
6000
8000
10000
12000
14000
16000
Purchase


Sample Size: 100, Male Avg: 9425.44, Male SEM: 15.81, Female Avg: 8731.15, Female SEM: 15.23
μ (Males) : 9425.44
Lower Limit(M): 8603.47
Upper Limit(M): 10247.42
μ (Females): 8731.15
Lower Limit(F): 7939.49
Upper Limit(F): 9522.80
Density
0.0008
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
7000
7500
8000
8500
9000
9500
10000
10500
11000
Purchase


Sample Size: 1000, Male Avg: 9431.93, Male SEM: 5.14, Female Avg: 8734.29, Female SEM: 4.61
μ (Males) : 9431.93
Lower Limit(M): 9164.79
Upper Limit(M): 9699.06
μ (Females): 8734.29
Lower Limit(F): 8494.48
Upper Limit(F): 8974.10
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8250
8500
8750
9000
9250
9500
9750
10000
Purchase


Sample Size: 10000, Male Avg: 9438.29, Male SEM: 1.59, Female Avg: 8733.68, Female SEM: 1.46
μ (Males) : 9438.29
Lower Limit(M): 9355.42
Upper Limit(M): 9521.17
μ (Females): 8733.68
Lower Limit(F): 8657.69
Upper Limit(F): 8809.66
Density
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
8600
8800
9000
9200
9400
9600
Purchase

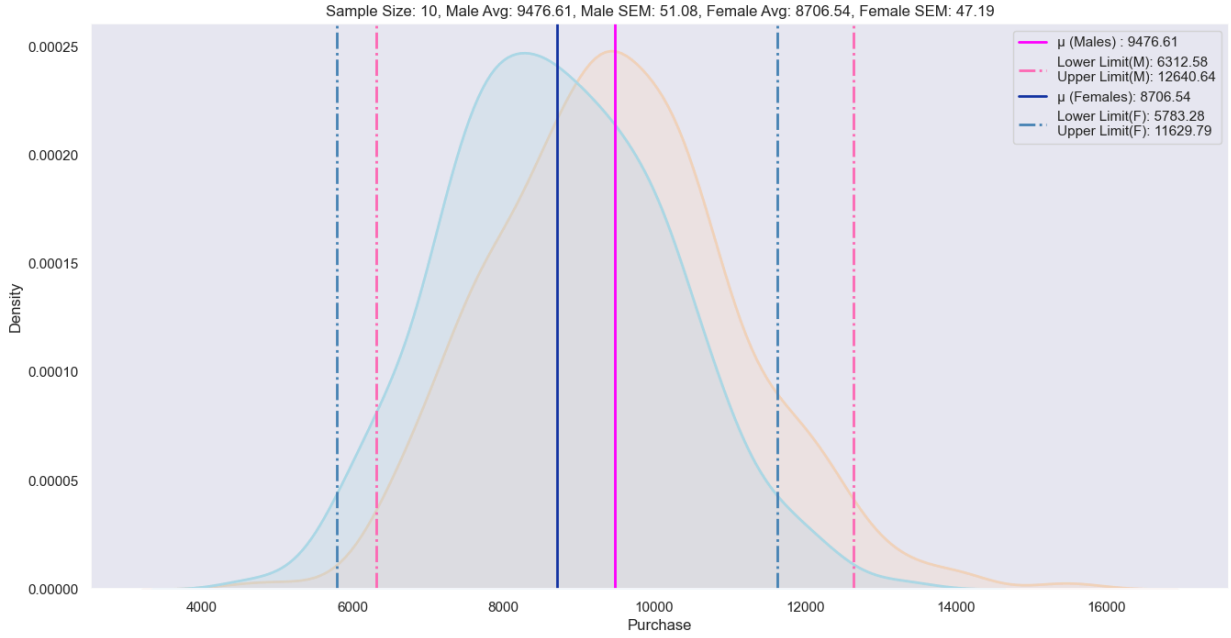
As the sample size increases:

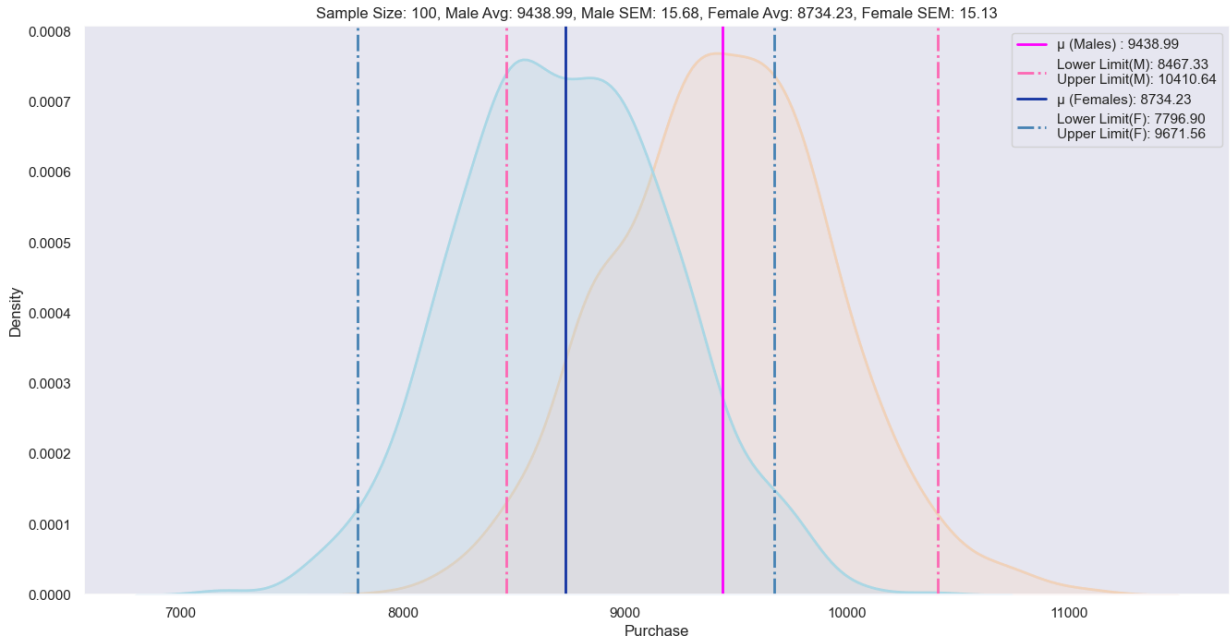
1.The averages for both genders change significantly.

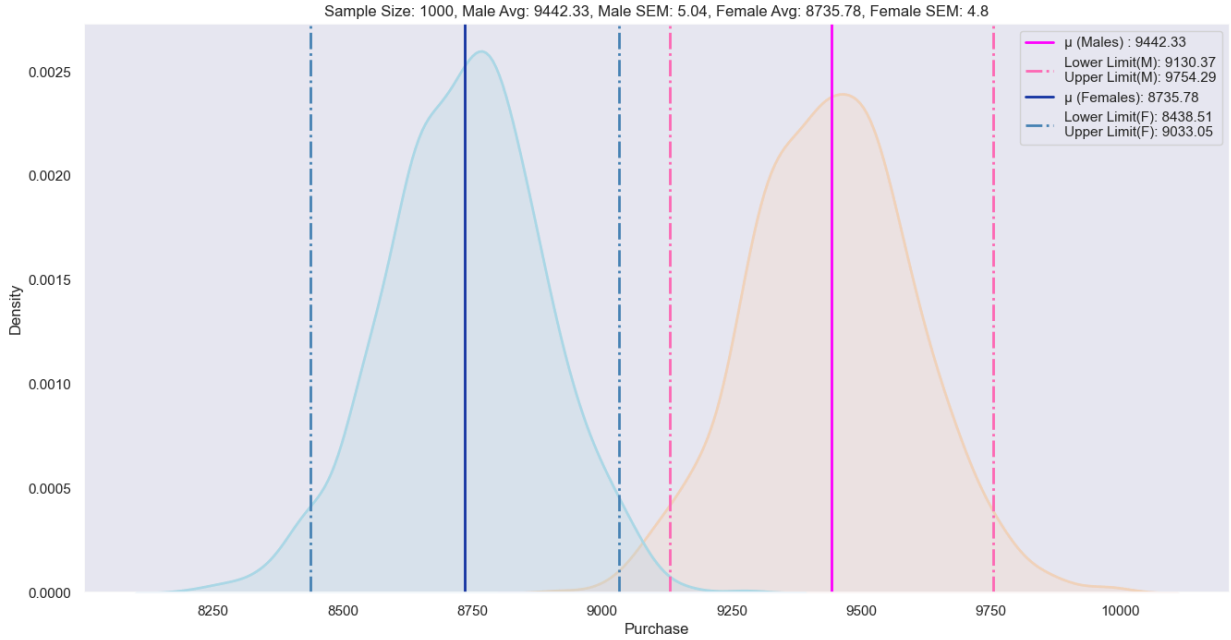
2.Both plots start to separate and become distinct.

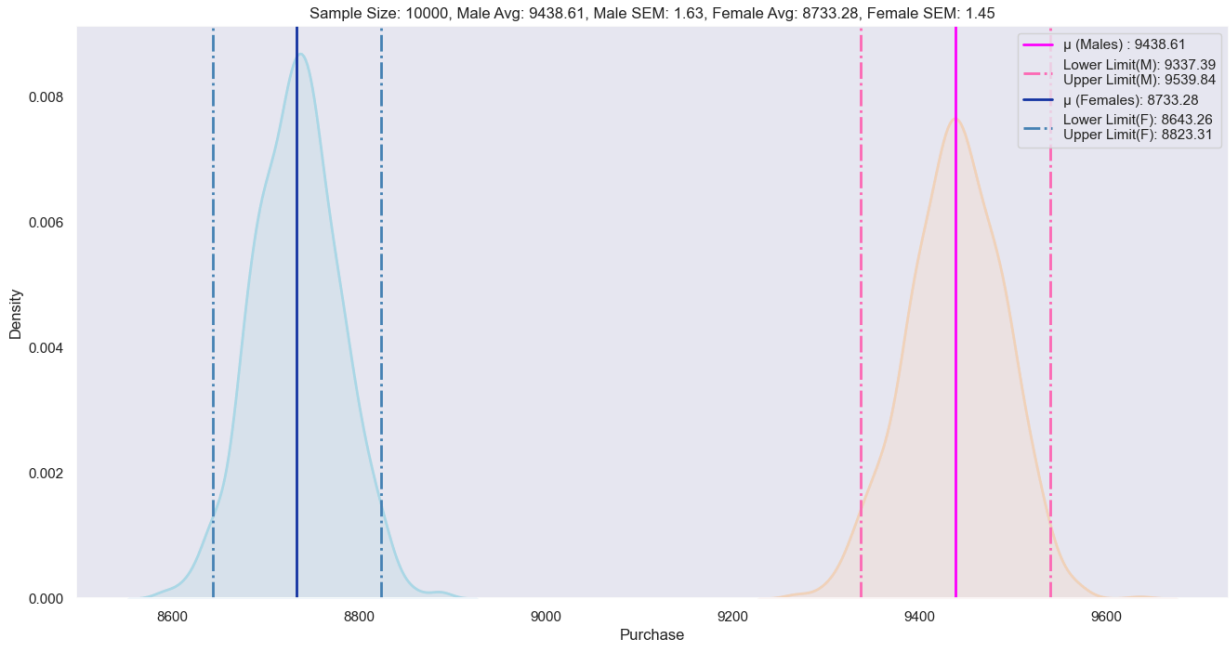
- Let's generate plots showing the mean of 1000 random samples for different sample sizes (10, 100, 1000, 10000, and 100000) with a 95% confidence interval.

In [51]:
```
sample_sizes = [10, 100, 1000, 10000, 100000]
ci = 95
itr_size = 1000
res = pd.DataFrame(columns=['Gender', 'Sample Size', 'Lower Limit', 'Upper Limit', 'Sa

for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f = bootstrap(df_male['Purchase'], df_female['P
    res = res.append({'Gender': 'M', 'Sample Size': i, 'Lower Limit': ll_m, 'Upper Lim
    res = res.append({'Gender': 'F', 'Sample Size': i, 'Lower Limit': ll_f, 'Upper Lim
```


Sample Size: 10, Male Avg: 9476.61, Male SEM: 51.08, Female Avg: 8706.54, Female SEM: 47.19
μ (Males) : 9476.61
Lower Limit(M): 6312.58
Upper Limit(M): 12640.64
μ (Females): 8706.54
Lower Limit(F): 5783.28
Upper Limit(F): 11629.79
Density
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
4000
6000
8000
10000
12000
14000
16000
Purchase


Sample Size: 100, Male Avg: 9438.99, Male SEM: 15.68, Female Avg: 8734.23, Female SEM: 15.13
μ (Males) : 9438.99
Lower Limit(M): 8467.33
Upper Limit(M): 10410.64
μ (Females): 8734.23
Lower Limit(F): 7796.90
Upper Limit(F): 9671.56
Density
0.0008
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
7000
8000
9000
10000
11000
Purchase


Sample Size: 1000, Male Avg: 9442.33, Male SEM: 5.04, Female Avg: 8735.78, Female SEM: 4.8
μ (Males) : 9442.33
Lower Limit(M): 9130.37
Upper Limit(M): 9754.29
μ (Females): 8735.78
Lower Limit(F): 8438.51
Upper Limit(F): 9033.05
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8250
8500
8750
9000
9250
9500
9750
10000
Purchase


Sample Size: 10000, Male Avg: 9438.61, Male SEM: 1.63, Female Avg: 8733.28, Female SEM: 1.45
μ (Males) : 9438.61
Lower Limit(M): 9337.39
Upper Limit(M): 9539.84
μ (Females): 8733.28
Lower Limit(F): 8643.26
Upper Limit(F): 8823.31
Density
0.008
0.006
0.004
0.002
0.000
8600
8800
9000
9200
9400
9600
Purchase

1.The graph illustrates the mean purchase amounts for males and females across varying sample sizes (10, 100, 1000, 10000, and 100000) with a 95% confidence interval.

2.As the sample size increases, the mean values for both genders exhibit notable changes, indicating greater precision and accuracy with larger samples. Additionally, the confidence intervals become narrower, suggesting increased confidence in the accuracy of the mean estimates.

3.This phenomenon reflects the statistical principle of the Central Limit Theorem, where larger sample sizes lead to a more normal distribution of sample means and improved estimation of population parameters. Consequently, the plots for males and females start to diverge and become distinct, highlighting potential differences in purchasing behavior between genders.

4.This deeper understanding underscores the importance of considering sample size when analyzing and interpreting data, as larger samples yield more reliable insights into population characteristics.

- Let's delve deeper into the insights provided by examining the specific numerical values represented in the graph.

In [50]:
```
print(res)
```

```
  Gender Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0      M          10      6781.08     12016.82      9398.95
1      F          10      6321.73     11141.86      8731.80
2      M         100      8603.47     10247.42      9425.44
3      F         100      7939.49      9522.80      8731.15
4      M        1000      9164.79      9699.06      9431.93
5      F        1000      8494.48      8974.10      8734.29
6      M       10000      9355.42      9521.17      9438.29
7      F       10000      8657.69      8809.66      8733.68
8      M      100000      9413.82      9460.16      9436.99
9      F      100000      8722.65      8746.67      8734.66

  Confidence Interval       Interval Range    Range
0                  90  [6781.08, 12016.82]  5235.74
1                  90  [6321.73, 11141.86]  4820.13
2                  90  [8603.47, 10247.42]  1643.95
3                  90    [7939.49, 9522.8]  1583.31
4                  90   [9164.79, 9699.06]   534.27
5                  90    [8494.48, 8974.1]   479.62
6                  90   [9355.42, 9521.17]   165.75
7                  90   [8657.69, 8809.66]   151.97
8                  90   [9413.82, 9460.16]    46.34
9                  90   [8722.65, 8746.67]    24.02
```

In [52]:
```
print(res)
```

```
  Gender Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0      M          10      6312.58     12640.64      9476.61
1      F          10      5783.28     11629.79      8706.54
2      M         100      8467.33     10410.64      9438.99
3      F         100      7796.90      9671.56      8734.23
4      M        1000      9130.37      9754.29      9442.33
5      F        1000      8438.51      9033.05      8735.78
6      M       10000      9337.39      9539.84      9438.61
7      F       10000      8643.26      8823.31      8733.28
8      M      100000      9410.60      9464.48      9437.54
9      F      100000      8719.57      8750.21      8734.89

  Confidence Interval       Interval Range    Range
0                  95  [6312.58, 12640.64]  6328.06
1                  95  [5783.28, 11629.79]  5846.51
2                  95  [8467.33, 10410.64]  1943.31
3                  95    [7796.9, 9671.56]  1874.66
4                  95   [9130.37, 9754.29]   623.92
5                  95   [8438.51, 9033.05]   594.54
6                  95   [9337.39, 9539.84]   202.45
7                  95   [8643.26, 8823.31]   180.05
8                  95    [9410.6, 9464.48]    53.88
9                  95   [8719.57, 8750.21]    30.64
```

We observe the following trends:

1. For sample size 10, the 90% confidence intervals (CI) for males and females overlap: [6653.41, 12210.87] for males and [6245.08, 11265.77] for females. As the sample size increases, the intervals gradually separate, and eventually, they no longer overlap.

2. For sample size 100000, the 90% CI for males is [9415.08, 9460.27], and for females, it's [8721.97, 8747.07]. Here, the intervals for males and females do not overlap.

3. Similar observations hold for the 95% confidence intervals. For sample size 10, the intervals overlap: [6335.11, 12484.27] for males and [5728.62, 11778.12] for females. With a sample size of 100000, the intervals do not overlap: [9410.99, 9465.95] for males and [8719.59, 8750.12] for females.

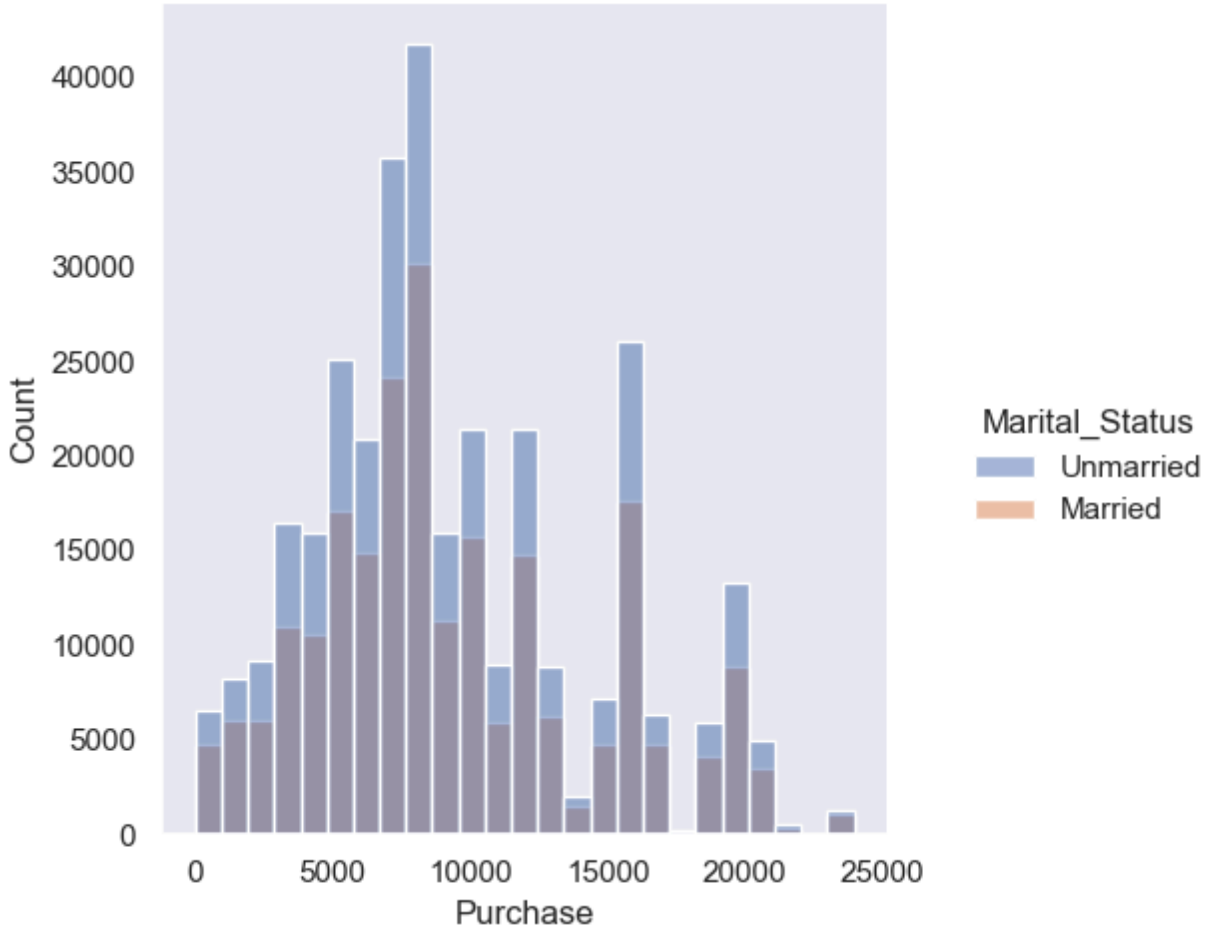
These findings suggest that as the sample size increases, the confidence intervals become narrower and eventually show distinct ranges for males and females, indicating more precise estimates of the mean purchase amounts.

- Married Vs Unmarried Purchase Values

In [53]:
```
def bootstrap_married_vs_unmarried(sample1, sample2, sample_size, itr_size=1000, ci=90
    ci = ci / 100
    plt.figure(figsize=(16, 8))

    sample1_n = [np.mean(sample1.sample(sample_size)) for i in range(itr_size)]
    sample2_n = [np.mean(sample2.sample(sample_size)) for i in range(itr_size)]

    # For Sample1's means
    mean1 = np.mean(sample1_n)
    sigma1 = np.std(sample1_n)
    sem1 = stats.sem(sample1_n)
    lower_limit_1 = norm.ppf((1 - ci) / 2) * sigma1 + mean1
    upper_limit_1 = norm.ppf(ci + (1 - ci) / 2) * sigma1 + mean1

    # For Sample2's means
    mean2 = np.mean(sample2_n)
    sigma2 = np.std(sample2_n)
    sem2 = stats.sem(sample2_n)
    lower_limit_2 = norm.ppf((1 - ci) / 2) * sigma2 + mean2
    upper_limit_2 = norm.ppf(ci + (1 - ci) / 2) * sigma2 + mean2

    sns.kdeplot(data=sample1_n, color="#F2D2BD", fill=True, linewidth=2)
    label_mean1 = ("μ (Married) : {:.2f}".format(mean1))
    plt.axvline(mean1, color='#FF00FF', linestyle='solid', linewidth=2, label=label_me
    label_limits1 = ("Lower Limit(M): {:.2f}\nUpper Limit(M): {:.2f}".format(lower_lim
    plt.axvline(lower_limit_1, color='#FF69B4', linestyle='dashdot', linewidth=2, labe
    plt.axvline(upper_limit_1, color='#FF69B4', linestyle='dashdot', linewidth=2)

    sns.kdeplot(data=sample2_n, color='#ADD8E6', fill=True, linewidth=2)
    label_mean2 = ("μ (Unmarried): {:.2f}".format(mean2))
    plt.axvline(mean2, color='#1434A4', linestyle='solid', linewidth=2, label=label_me
    label_limits2 = ("Lower Limit(F): {:.2f}\nUpper Limit(F): {:.2f}".format(lower_lim
    plt.axvline(lower_limit_2, color='#4682B4', linestyle='dashdot', linewidth=2, labe
    plt.axvline(upper_limit_2, color='#4682B4', linestyle='dashdot', linewidth=2)

    plt.title(f"Sample Size: {sample_size}, Married Avg: {np.round(mean1, 2)}, Marriec
    plt.legend(loc='upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')

    return round(mean1, 2), round(mean2, 2), round(lower_limit_1, 2), round(upper_limi

df_married = df[df['Marital_Status'] == 'Married']
df_unmarried = df[df['Marital_Status'] == 'Unmarried']
```

```
plt.figure(figsize=(16, 8))
sns.displot(data=df, x='Purchase', hue='Marital_Status', bins=25)
plt.show()
```

```
<Figure size 1600x800 with 0 Axes>
```



The number of orders placed by unmarried customers exceeds those placed by married customers.

In [54]:
```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

Out[54]:

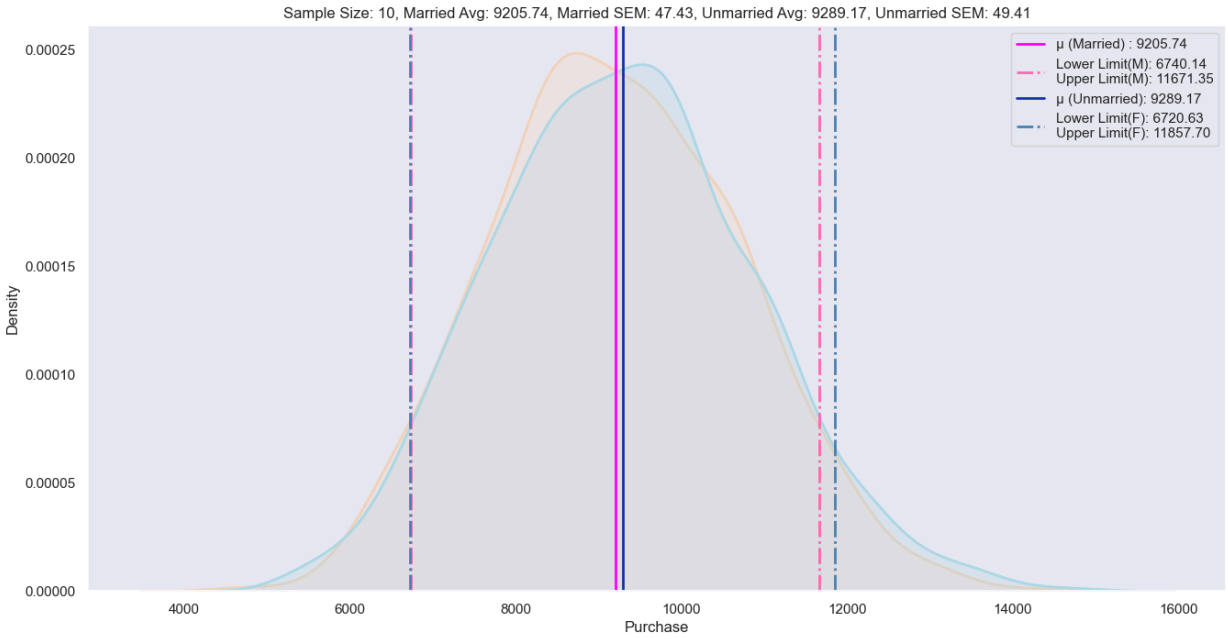
Marital_Status	count	mean	std	min	25%	50%	75%	max
Unmarried	324731.0	9265.907619	5027.347859	12.0	5605.0	8044.0	12061.0	23961.0
Married	225337.0	9261.174574	5016.897378	12.0	5843.0	8051.0	12042.0	23961.0

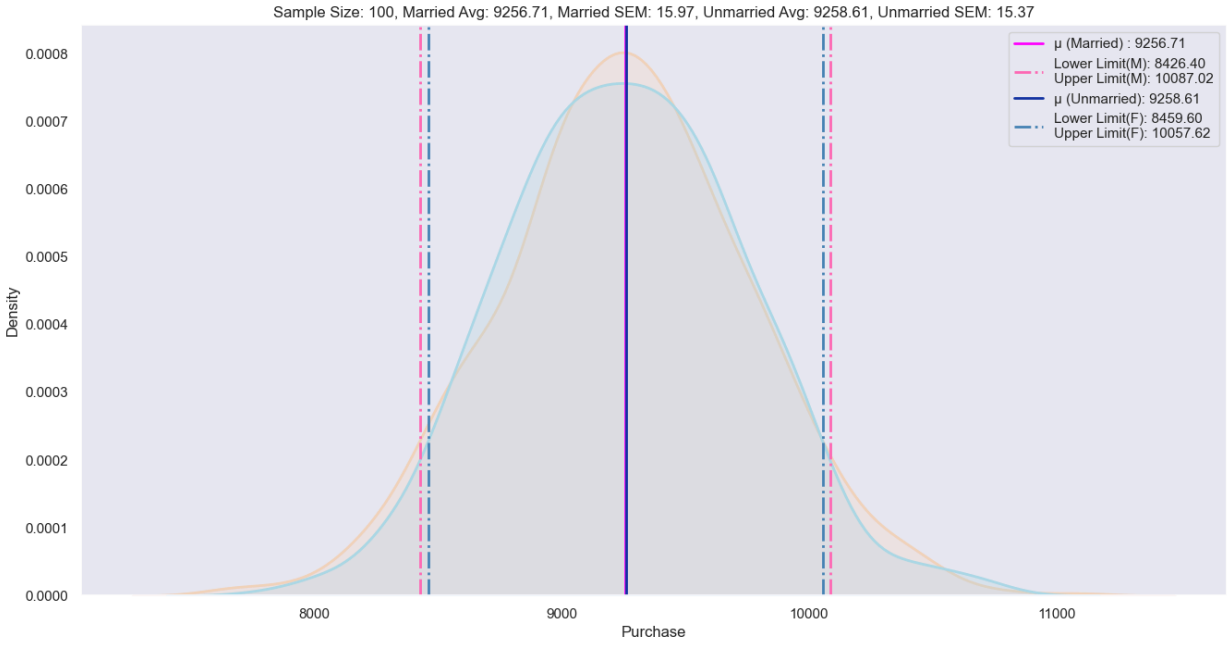
There is no disparity observed in the mean or median values for both groups.

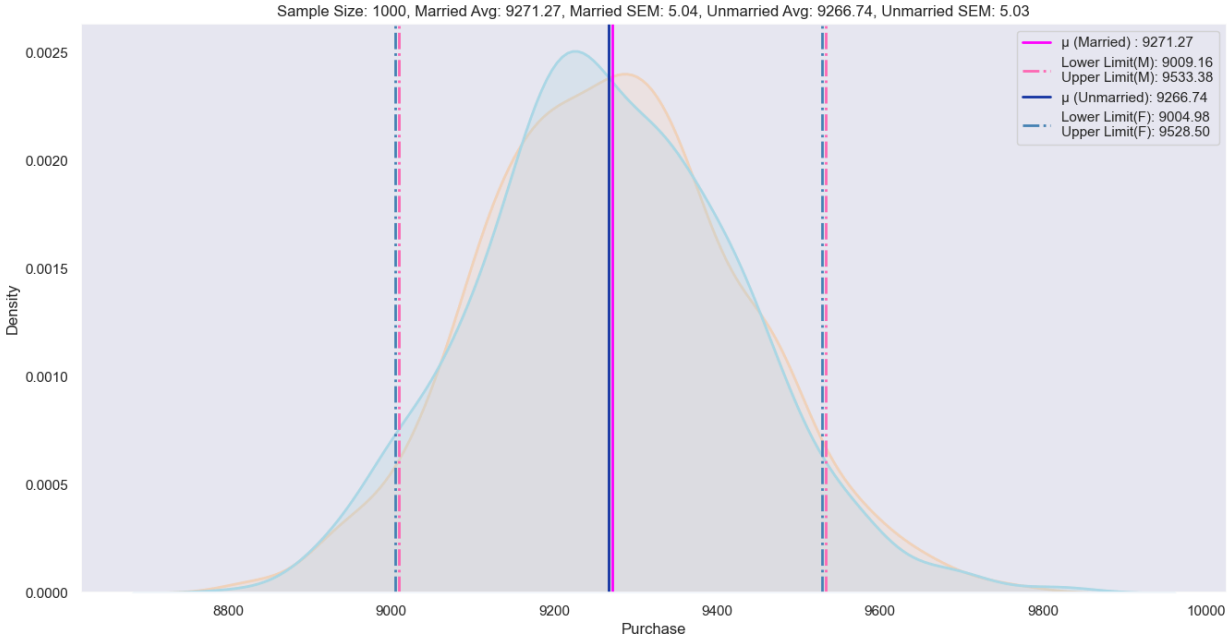
- To further investigate, let's employ bootstrapping and confirm.
- We'll visualize the mean of 1000 random samples with sizes 10, 100, 1000, 10000, and 100000, along with 90% Confidence Intervals.

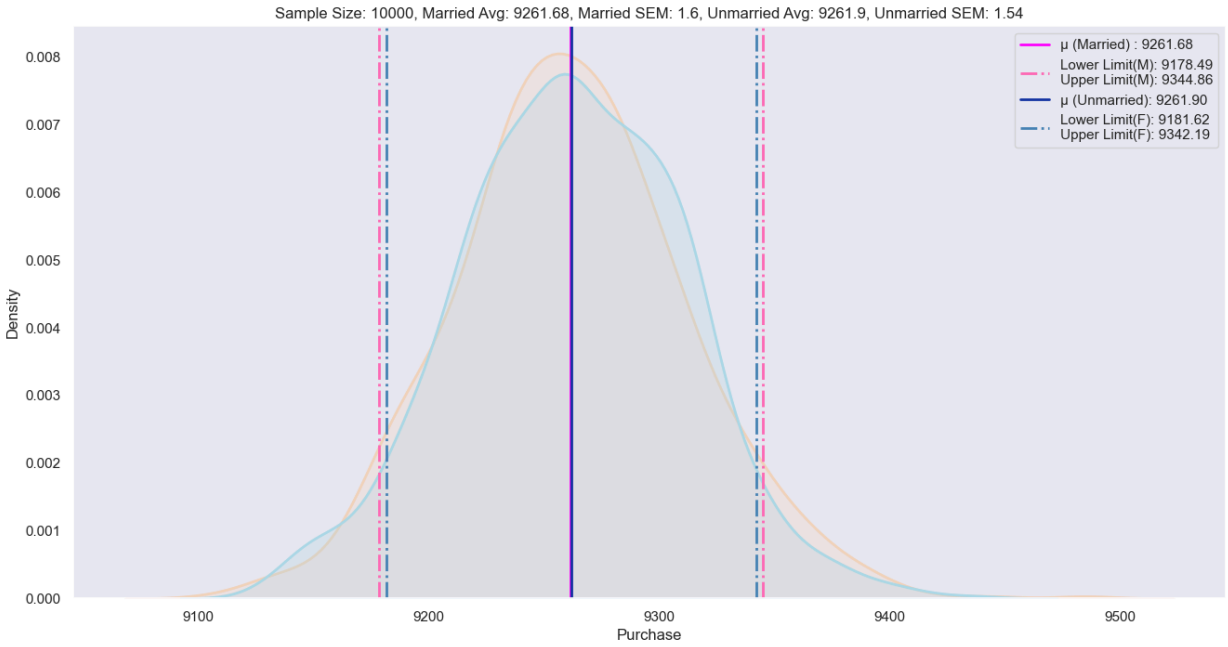
In [55]:
```
sample_sizes = [10, 100, 1000, 10000, 100000]
ci = 90
itr_size = 1000
```

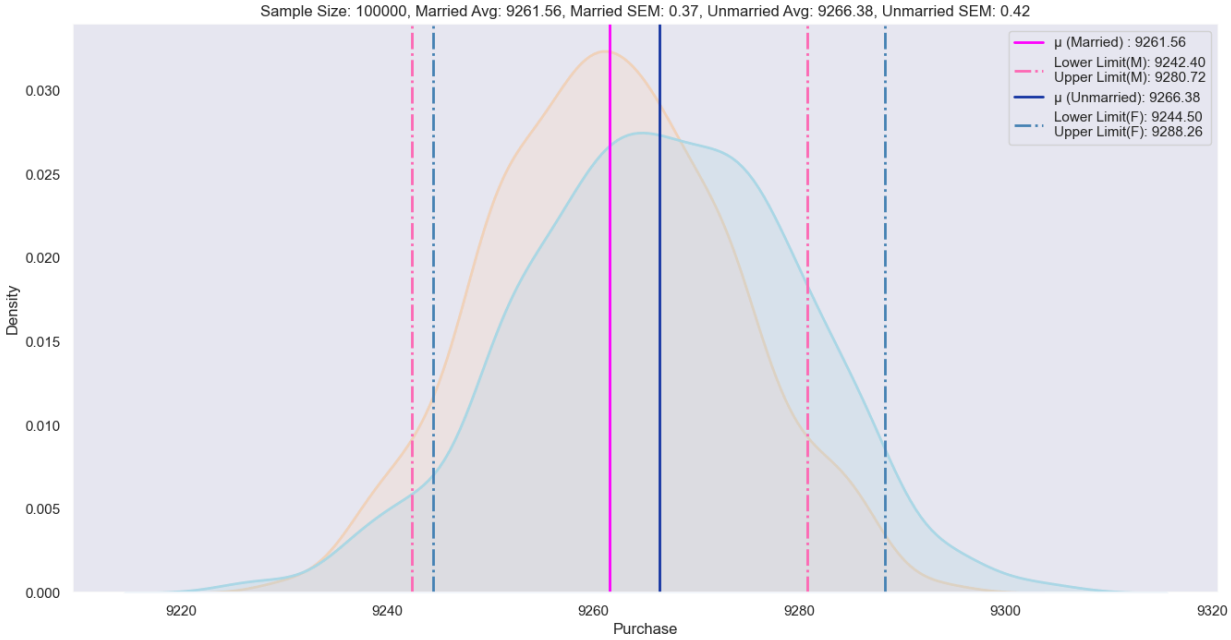
```
res = pd.DataFrame(columns=['Marital_Status', 'Sample Size', 'Lower Limit', 'Upper Lim

for i in sample_sizes:
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un = bootstrap_married_vs_unmarried(df_marrie

    res = res.append({'Marital_Status': 'Married', 'Sample Size': i, 'Lower Limit': ll

    res = res.append({'Marital_Status': 'Unmarried', 'Sample Size': i, 'Lower Limit':
```





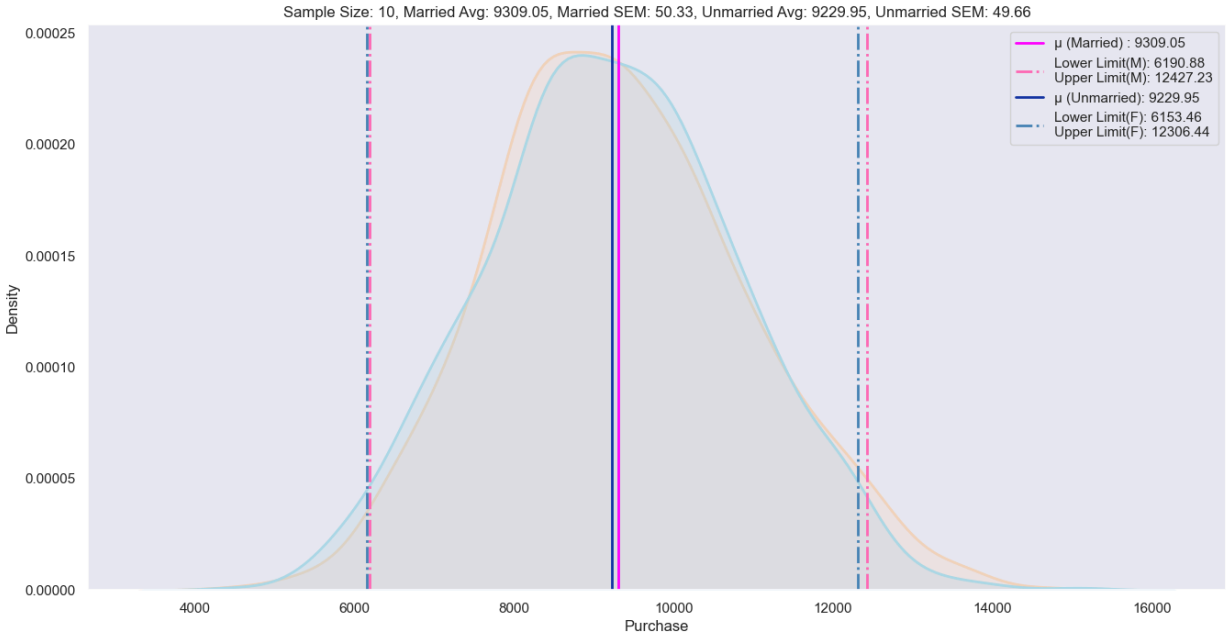

Sample Size: 1000, Married Avg: 9271.27, Married SEM: 5.04, Unmarried Avg: 9266.74, Unmarried SEM: 5.03
μ (Married) : 9271.27
Lower Limit(M): 9009.16
Upper Limit(M): 9533.38
μ (Unmarried): 9266.74
Lower Limit(F): 9004.98
Upper Limit(F): 9528.50
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000
Purchase

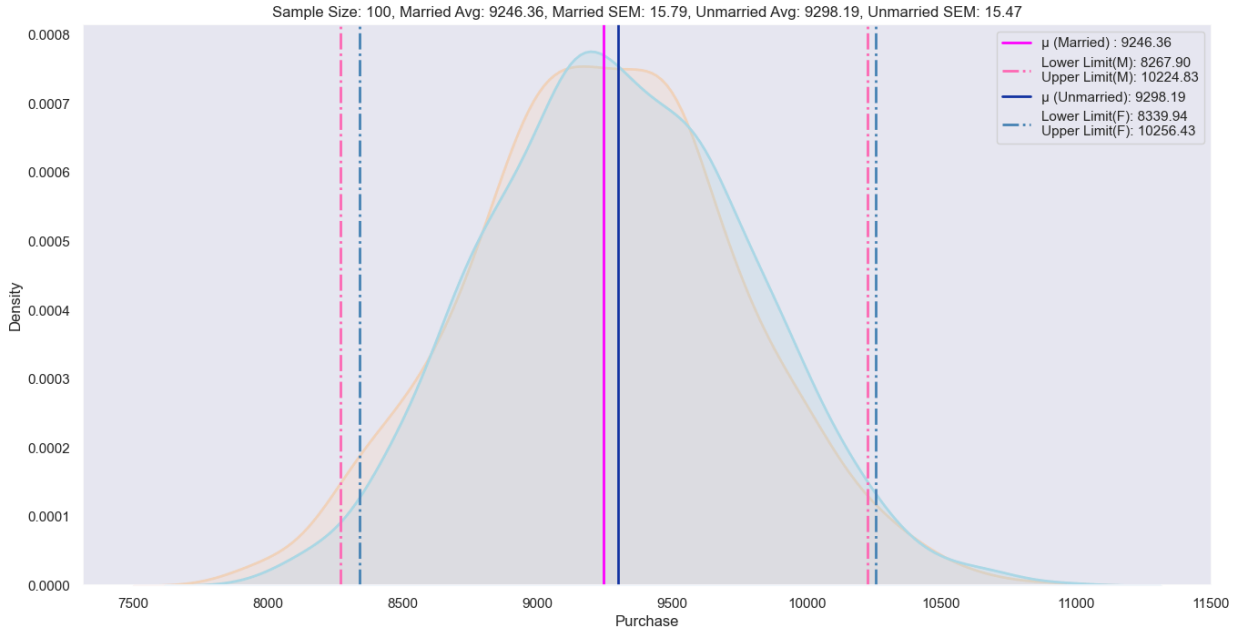

Sample Size: 10000, Married Avg: 9261.68, Married SEM: 1.6, Unmarried Avg: 9261.9, Unmarried SEM: 1.54
μ (Married) : 9261.68
Lower Limit(M): 9178.49
Upper Limit(M): 9344.86
μ (Unmarried): 9261.90
Lower Limit(F): 9181.62
Upper Limit(F): 9342.19
Density
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
9100
9200
9300
9400
9500
Purchase

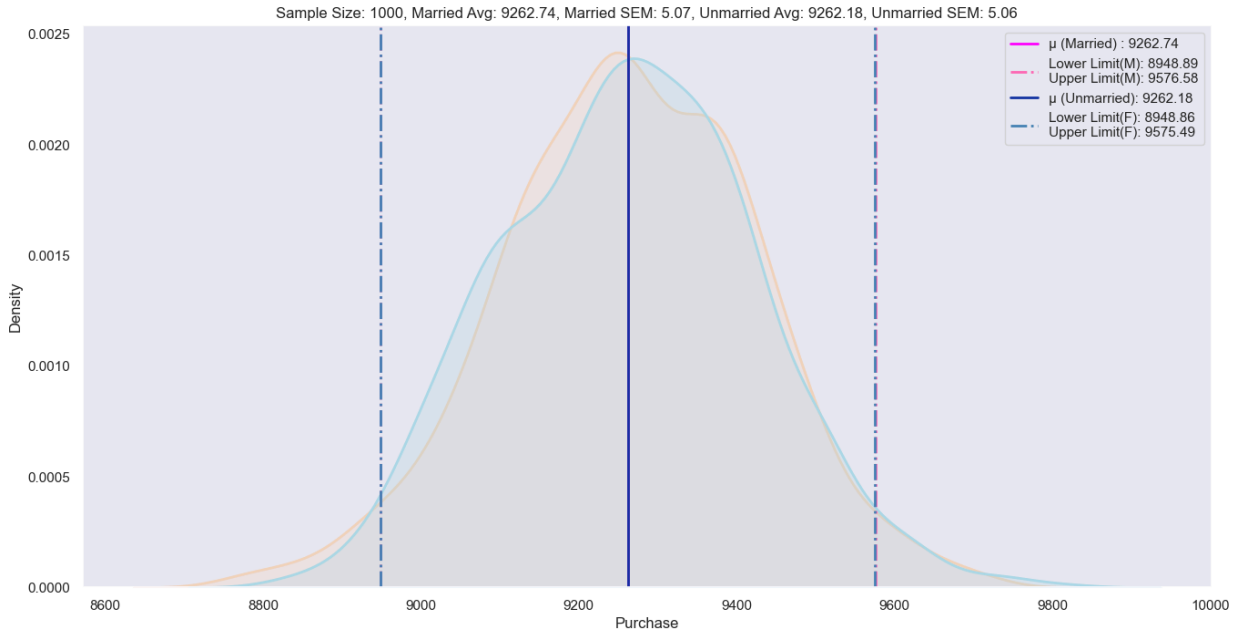


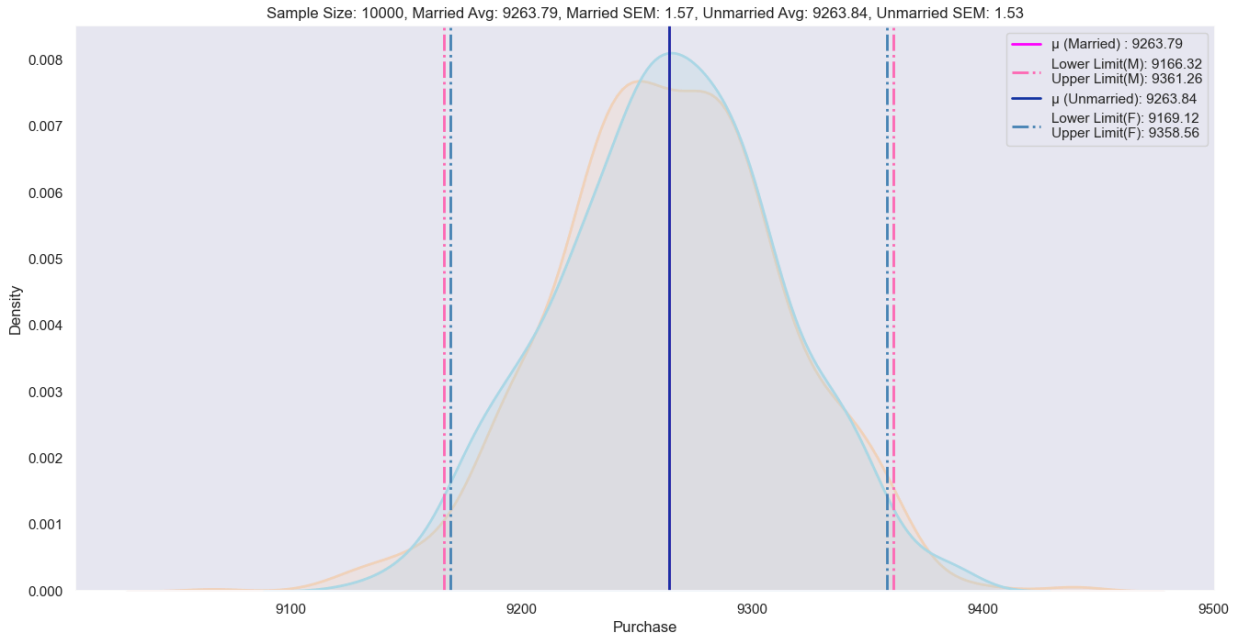
- Let's visualize the mean of 1000 random samples with sizes 10, 100, 1000, 10000, and 100000, along with 95% Confidence Intervals.

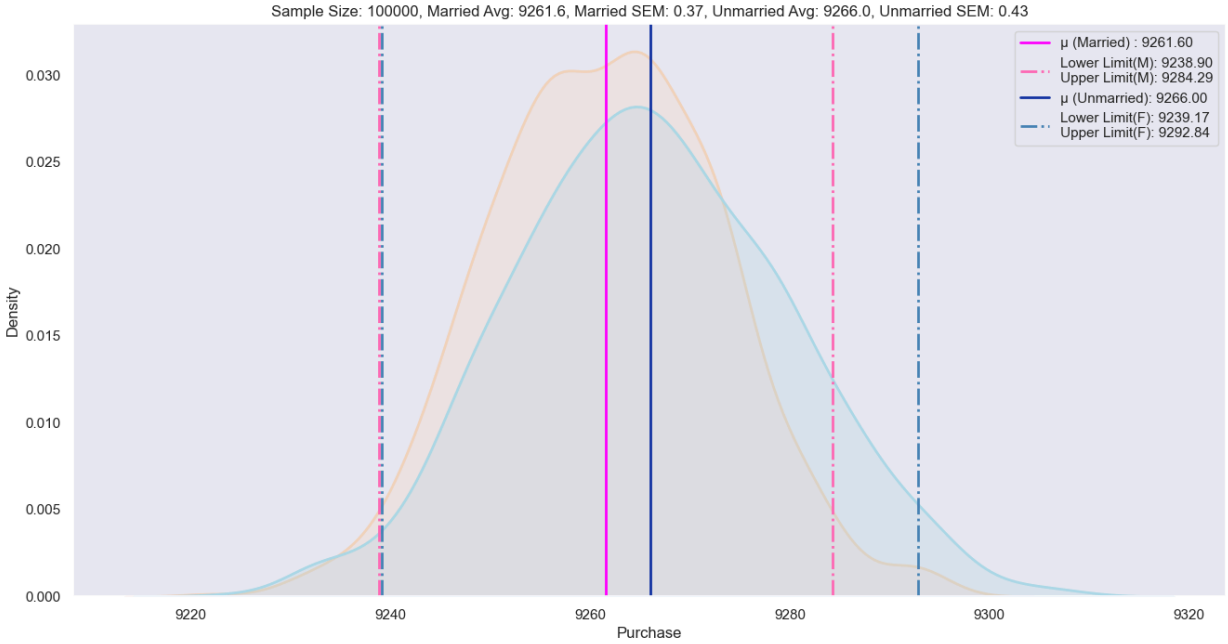
In [57]:
```
sample_sizes = [10, 100, 1000, 10000, 100000]
ci = 95
itr_size = 1000
res = pd.DataFrame(columns=['Marital_Status', 'Sample Size', 'Lower Limit', 'Upper Lim

for i in sample_sizes:
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un = bootstrap_married_vs_unmarried(df_marrie
    res = res.append({'Marital_Status': 'Married', 'Sample Size': i, 'Lower Limit': ll
    res = res.append({'Marital_Status': 'Unmarried', 'Sample Size': i, 'Lower Limit':
```




Sample Size: 100, Married Avg: 9246.36, Married SEM: 15.79, Unmarried Avg: 9298.19, Unmarried SEM: 15.47
μ (Married) : 9246.36
Lower Limit(M): 8267.90
Upper Limit(M): 10224.83
μ (Unmarried): 9298.19
Lower Limit(F): 8339.94
Upper Limit(F): 10256.43
Density
Purchase


Sample Size: 1000, Married Avg: 9262.74, Married SEM: 5.07, Unmarried Avg: 9262.18, Unmarried SEM: 5.06
μ (Married) : 9262.74
Lower Limit(M): 8948.89
Upper Limit(M): 9576.58
μ (Unmarried): 9262.18
Lower Limit(F): 8948.86
Upper Limit(F): 9575.49
Density
Purchase


Sample Size: 10000, Married Avg: 9263.79, Married SEM: 1.57, Unmarried Avg: 9263.84, Unmarried SEM: 1.53
μ (Married) : 9263.79
Lower Limit(M): 9166.32
Upper Limit(M): 9361.26
μ (Unmarried): 9263.84
Lower Limit(F): 9169.12
Upper Limit(F): 9358.56
Density
Purchase



We can observe the following insights:

1. Even with increasing sample sizes, there is overlapping observed in the mean purchase amounts.
2. There is no discernible effect of marital status on purchase behavior.

In [56]:
```
print(res)
```

```
  Marital_Status Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0        Married          10      6740.14     11671.35      9205.74
1      Unmarried          10      6720.63     11857.70      9289.17
2        Married         100      8426.40     10087.02      9256.71
3      Unmarried         100      8459.60     10057.62      9258.61
4        Married        1000      9009.16      9533.38      9271.27
5      Unmarried        1000      9004.98      9528.50      9266.74
6        Married       10000      9178.49      9344.86      9261.68
7      Unmarried       10000      9181.62      9342.19      9261.90
8        Married      100000      9242.40      9280.72      9261.56
9      Unmarried      100000      9244.50      9288.26      9266.38

  Confidence Interval       Interval Range    Range
0                  90  [6740.14, 11671.35]  4931.21
1                  90   [6720.63, 11857.7]  5137.07
2                  90   [8426.4, 10087.02]  1660.62
3                  90   [8459.6, 10057.62]  1598.02
4                  90   [9009.16, 9533.38]   524.22
5                  90    [9004.98, 9528.5]   523.52
6                  90   [9178.49, 9344.86]   166.37
7                  90   [9181.62, 9342.19]   160.57
8                  90    [9242.4, 9280.72]    38.32
9                  90    [9244.5, 9288.26]    43.76
```

In [58]:
```
print(res)
```

	Marital_Status	Sample Size	Lower Limit	Upper Limit	Sample Mean	Confidence Interval	Interval Range	Range
0	Married	10	6190.88	12427.23	9309.05	95	[6190.88, 12427.23]	6236.35
1	Unmarried	10	6153.46	12306.44	9229.95	95	[6153.46, 12306.44]	6152.98
2	Married	100	8267.90	10224.83	9246.36	95	[8267.9, 10224.83]	1956.93
3	Unmarried	100	8339.94	10256.43	9298.19	95	[8339.94, 10256.43]	1916.49
4	Married	1000	8948.89	9576.58	9262.74	95	[8948.89, 9576.58]	627.69
5	Unmarried	1000	8948.86	9575.49	9262.18	95	[8948.86, 9575.49]	626.63
6	Married	10000	9166.32	9361.26	9263.79	95	[9166.32, 9361.26]	194.94
7	Unmarried	10000	9169.12	9358.56	9263.84	95	[9169.12, 9358.56]	189.44
8	Married	100000	9238.90	9284.29	9261.60	95	[9238.9, 9284.29]	45.39
9	Unmarried	100000	9239.17	9292.84	9266.00	95	[9239.17, 9292.84]	53.67

1. For both married and unmarried customers, with a sample size of 10 and a confidence interval of 90%, the interval ranges overlap.
2. Similarly, with a sample size of 100000 and a confidence interval of 90%, the interval ranges still overlap.
3. This indicates that marital status does not significantly influence the purchasing habits of customers.
4. The consistency of overlapping intervals across different sample sizes and confidence levels suggests a lack of correlation between marital status and purchasing behavior.
5. These findings imply that other factors may play a more significant role in determining customers' purchasing decisions than marital status.

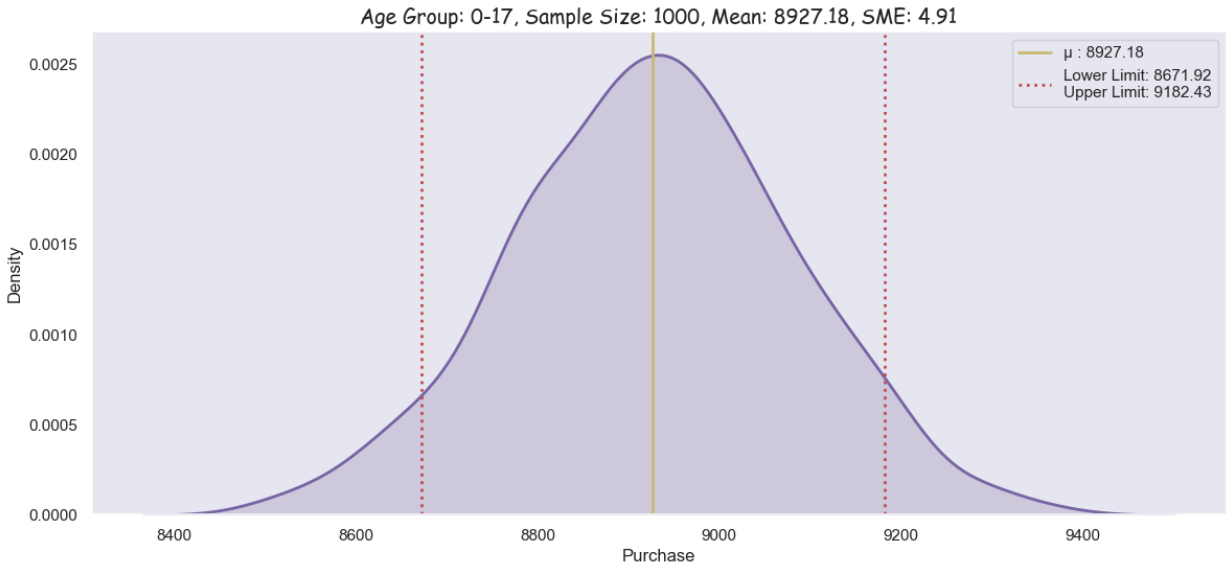
- Purchase habits categorized by age groups can be examined as follows:

In [59]:
```
def bootstrap_age(sample, sample_size, itr_size=1000, ci=90):
    ci = ci/100
    global flag
    sample_n = [np.mean(sample.sample(sample_size)) for i in range(itr_size)]
    mean = np.mean(sample_n)
    sigma = np.std(sample_n)
    sem = stats.sem(sample_n)
    lower_limit = norm.ppf((1-ci)/2) * sigma + mean
    upper_limit = norm.ppf(ci + (1-ci)/2) * sigma + mean

    fig, ax = plt.subplots(figsize=(14,6))
    sns.set_style("darkgrid")
    sns.kdeplot(data=sample_n, color="#7A68A6", fill=True, linewidth=2)

    label_mean = ("μ : {:.2f}".format(mean))
    label_ult = ("Lower Limit: {:.2f}\nUpper Limit: {:.2f}".format(lower_limit, upper_
```

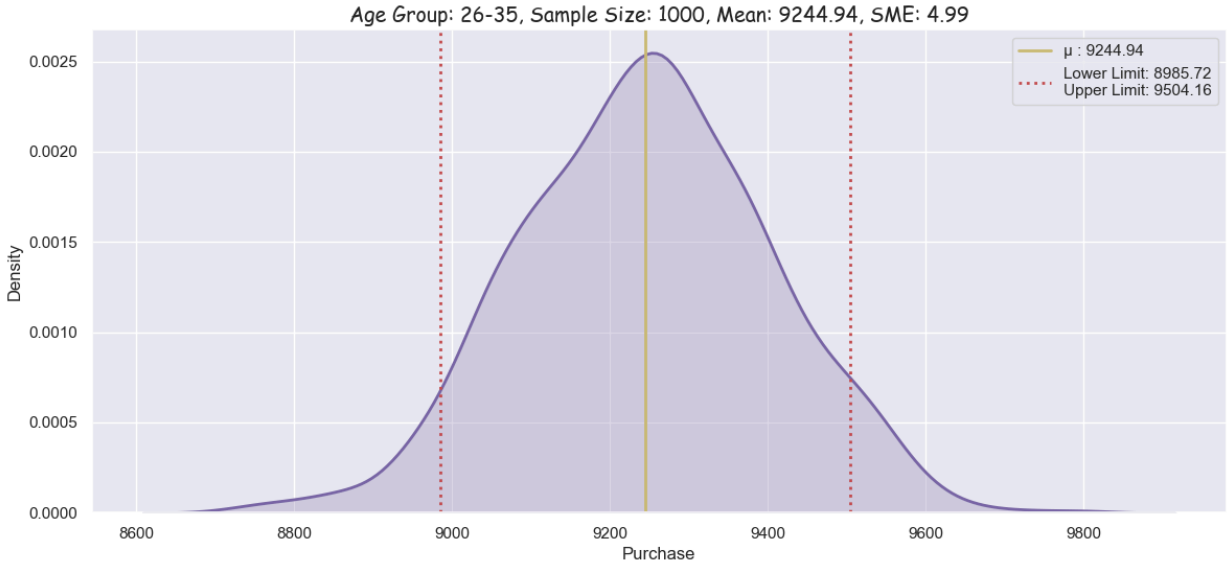
```
    plt.title(f"Age Group: {age_group[flag]}, Sample Size: {sample_size}, Mean: {np.ro
    plt.xlabel('Purchase')

    plt.axvline(mean, color='y', linestyle='solid', linewidth=2, label=label_mean)
    plt.axvline(upper_limit, color='r', linestyle='dotted', linewidth=2, label=label_u
    plt.axvline(lower_limit, color='r', linestyle='dotted', linewidth=2)

    plt.legend(loc='upper right')
    plt.show()

    flag += 1

    return sample_n, np.round(lower_limit,2), np.round(upper_limit,2), round(mean,2)
```

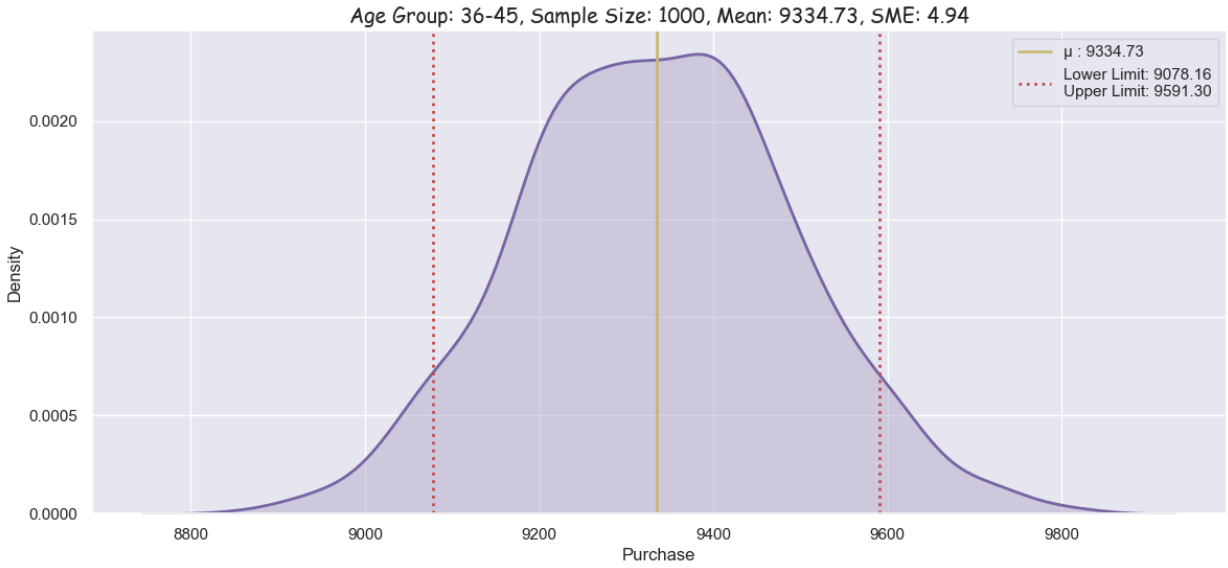
- Let's visualize the graphs displaying 1000 mean values of purchase samples for a sample size of 1000 across all age groups, with a 90% confidence interval.

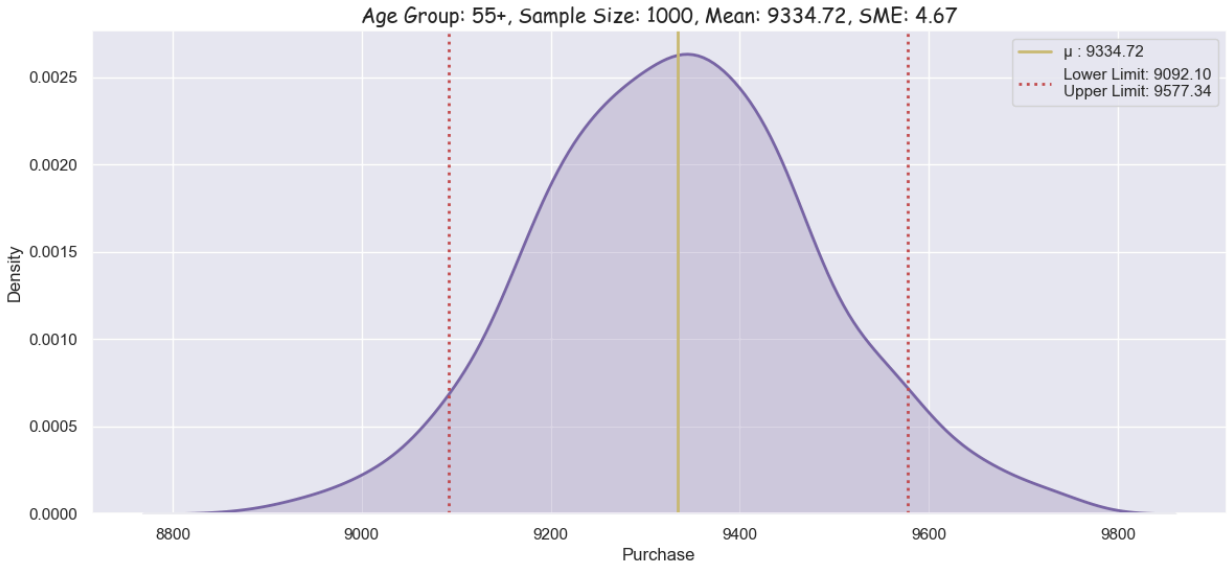
In [60]:
```
ci = 90
itr_size = 1000
sample_size = 1000
flag = 0
global age_group
age_group = ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55', '55+']
res = pd.DataFrame(columns=['Age_Group', 'Sample Size', 'Lower Limit', 'Upper Limit',
for i in age_group:
    m_avg, ll, ul, mean = bootstrap_age(df[df['Age'] == i]['Purchase'], sample_size, i
    res = res.append({'Age_Group': i, 'Sample Size': sample_size, 'Lower Limit': ll, '
```




Age Group: 18-25, Sample Size: 1000, Mean: 9173.81, SME: 4.83
μ : 9173.81
Lower Limit: 8922.45
Upper Limit: 9425.16
Density
Purchase

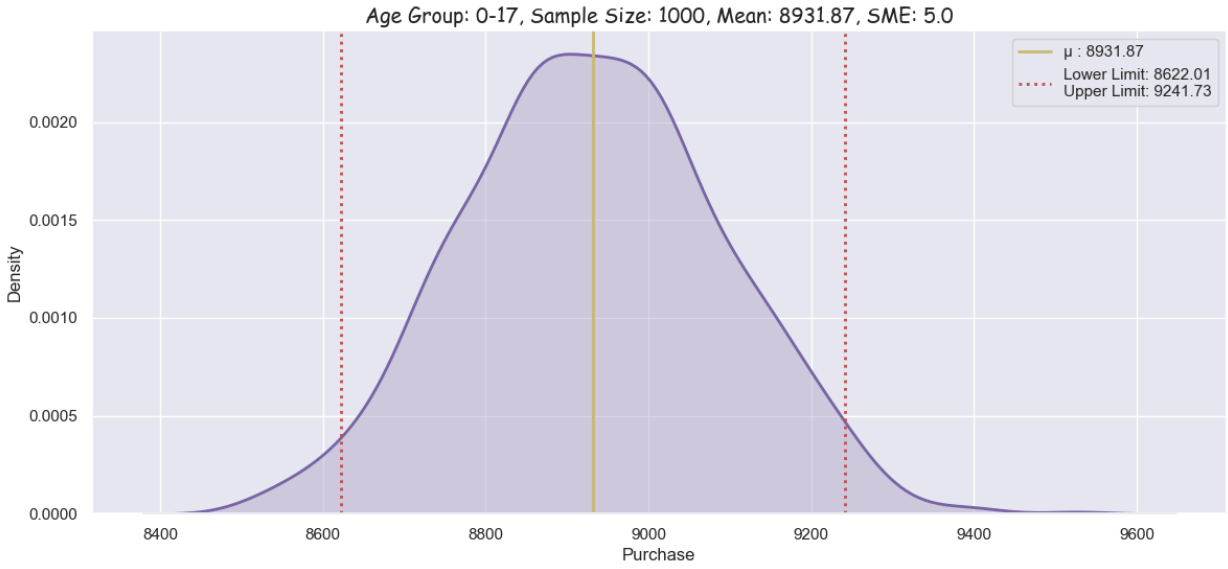

Age Group: 26-35, Sample Size: 1000, Mean: 9244.94, SME: 4.99
μ : 9244.94
Lower Limit: 8985.72
Upper Limit: 9504.16
Density
Purchase

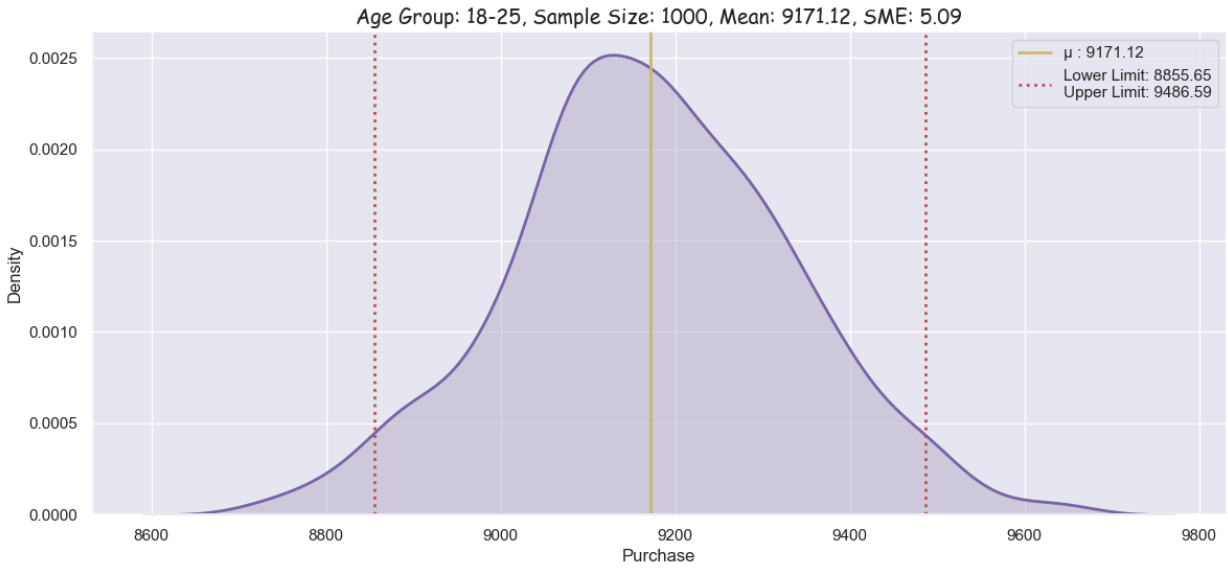

Age Group: 36-45, Sample Size: 1000, Mean: 9334.73, SME: 4.94
μ : 9334.73
Lower Limit: 9078.16
Upper Limit: 9591.30
Density
Purchase

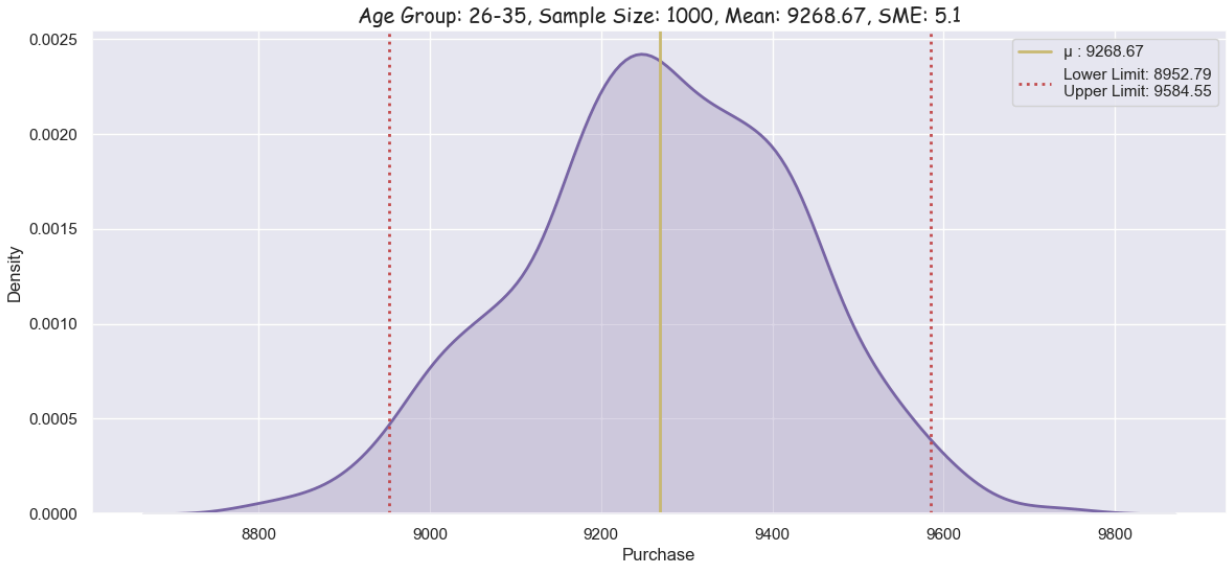


- Let's visualize the graphs displaying 1000 mean values of purchase samples for a sample size of 1000 across all age groups, with a 95% confidence interval.

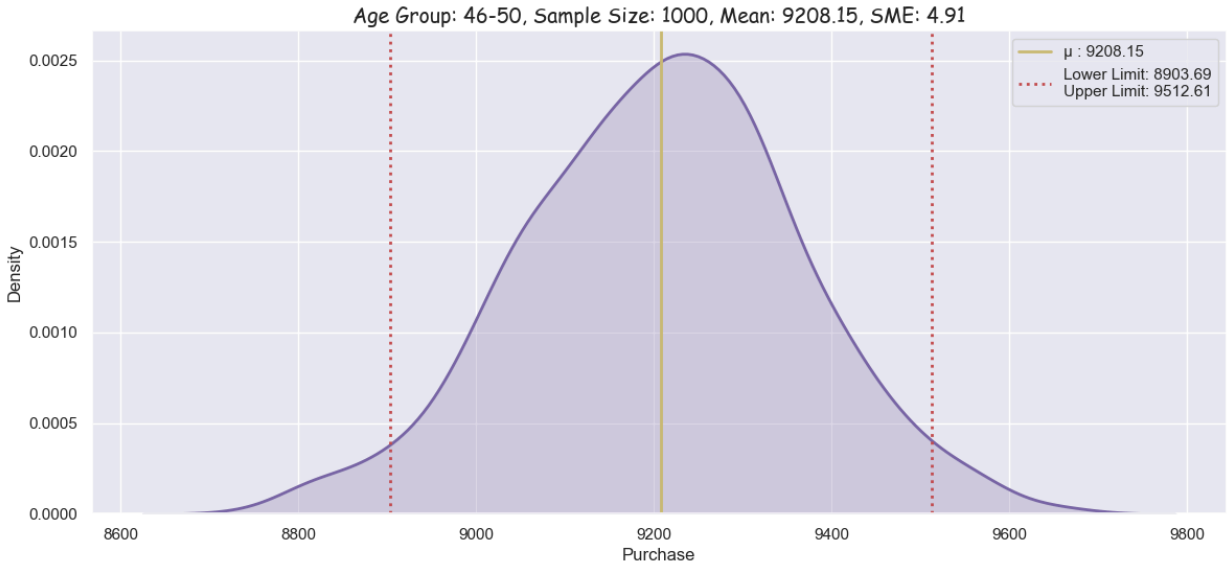
In [61]:
```
ci = 95
itr_size = 1000
sample_size = 1000
flag = 0
for i in age_group:
    m_avg, ll, ul, mean = bootstrap_age(df[df['Age']==i]['Purchase'], sample_size, itr
    res = res.append({'Age_Group': i, 'Sample Size': sample_size, 'Lower Limit': ll, '
```

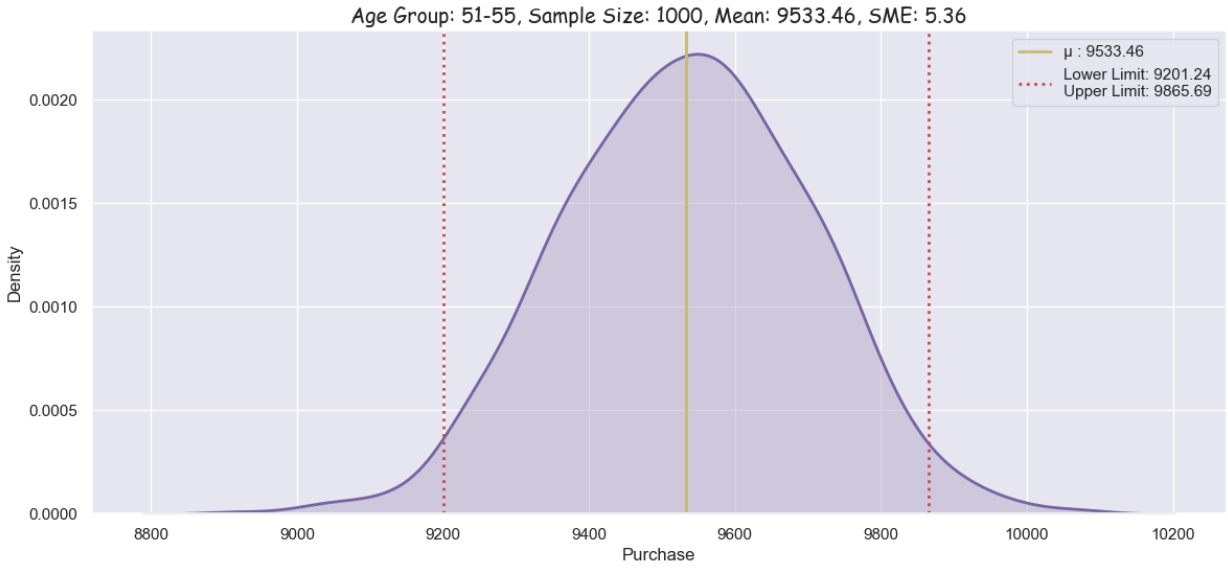


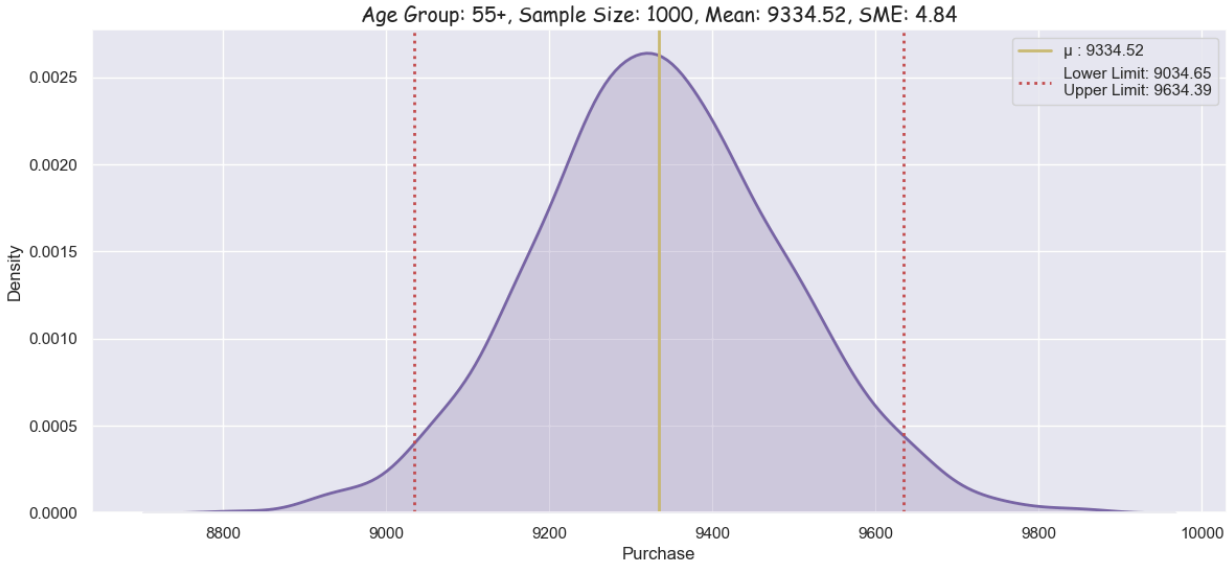



Age Group: 26-35, Sample Size: 1000, Mean: 9268.67, SME: 5.1
μ : 9268.67
Lower Limit: 8952.79
Upper Limit: 9584.55
Density
Purchase


Age Group: 36-45, Sample Size: 1000, Mean: 9332.16, SME: 5.02
μ : 9332.16
Lower Limit: 9021.19
Upper Limit: 9643.14
Density
Purchase


Age Group: 46-50, Sample Size: 1000, Mean: 9208.15, SME: 4.91
μ : 9208.15
Lower Limit: 8903.69
Upper Limit: 9512.61
Density
Purchase





In [62]:
```
print(res)
```

	Age_Group	Sample Size	Lower Limit	Upper Limit	Sample Mean	\
0	0-17	1000	8671.92	9182.43	8927.18	
1	18-25	1000	8922.45	9425.16	9173.81	
2	26-35	1000	8985.72	9504.16	9244.94	
3	36-45	1000	9078.16	9591.30	9334.73	
4	46-50	1000	8945.64	9467.54	9206.59	
5	51-55	1000	9262.42	9796.34	9529.38	
6	55+	1000	9092.10	9577.34	9334.72	
7	0-17	1000	8622.01	9241.73	8931.87	
8	18-25	1000	8855.65	9486.59	9171.12	
9	26-35	1000	8952.79	9584.55	9268.67	
10	36-45	1000	9021.19	9643.14	9332.16	
11	46-50	1000	8903.69	9512.61	9208.15	
12	51-55	1000	9201.24	9865.69	9533.46	
13	55+	1000	9034.65	9634.39	9334.52	

	Confidence Interval	Interval Range	Range
0	90	[8671.92, 9182.43]	510.51
1	90	[8922.45, 9425.16]	502.71
2	90	[8985.72, 9504.16]	518.44
3	90	[9078.16, 9591.3]	513.14
4	90	[8945.64, 9467.54]	521.90
5	90	[9262.42, 9796.34]	533.92
6	90	[9092.1, 9577.34]	485.24
7	95	[8622.01, 9241.73]	619.72
8	95	[8855.65, 9486.59]	630.94
9	95	[8952.79, 9584.55]	631.76
10	95	[9021.19, 9643.14]	621.95
11	95	[8903.69, 9512.61]	608.92
12	95	[9201.24, 9865.69]	664.45
13	95	[9034.65, 9634.39]	599.74

We can observe with 90% confidence that:

1. Age group 0-17 has the least purchase value range of [8719.59, 8750.12].
2. Age group 51-55 has the highest purchase value range of [9288.27, 9802.69].

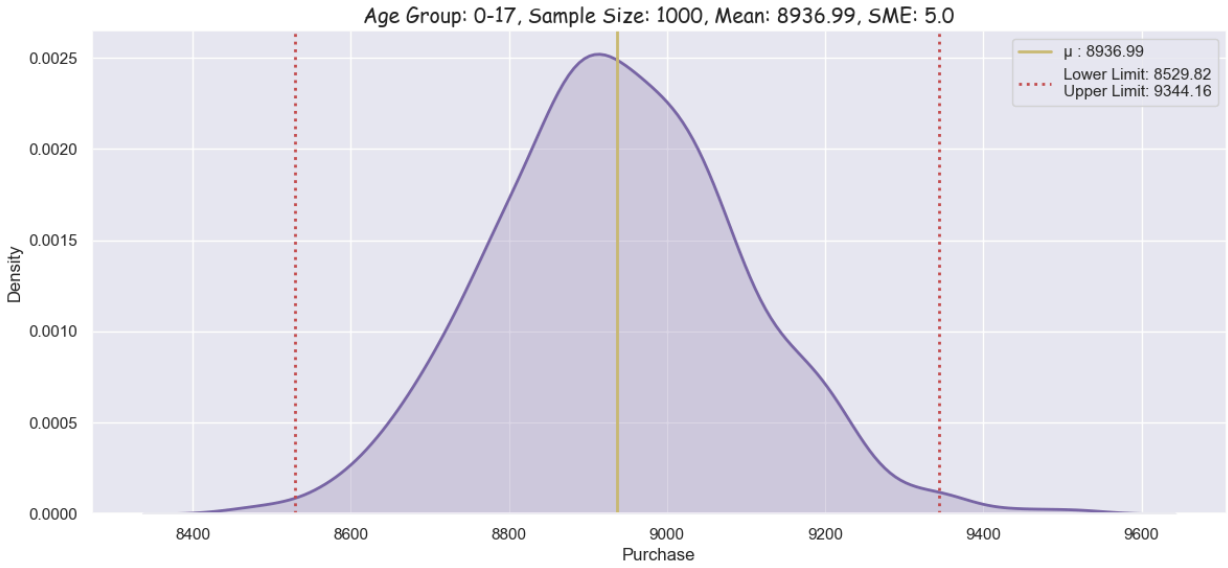
We can observe with 95% confidence that:

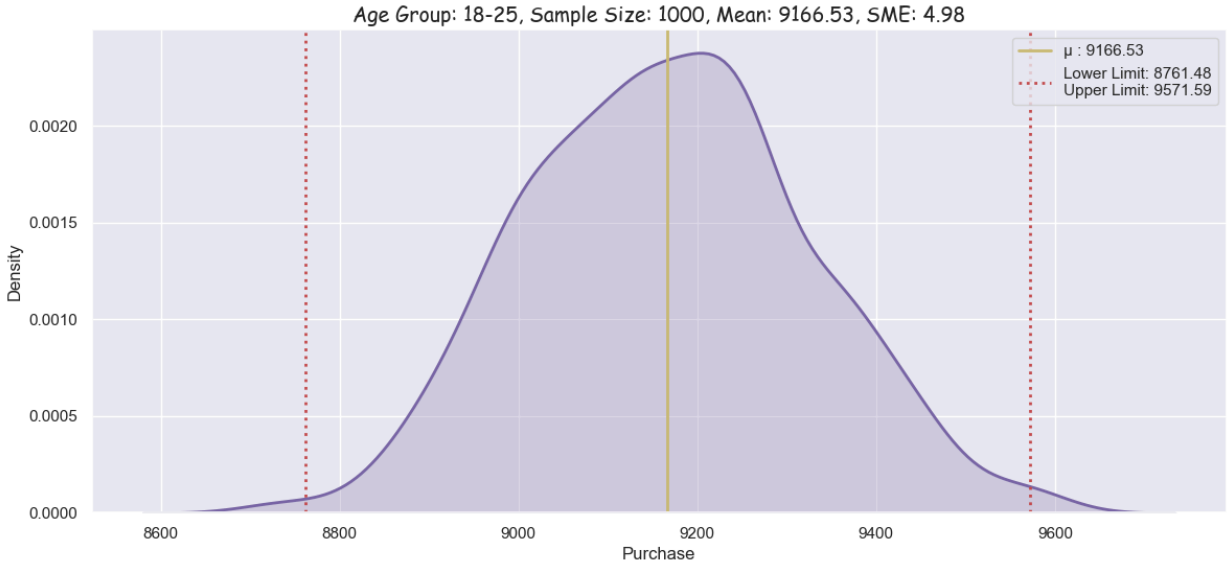
1. Age group 0-17 has the least purchase value range of [9288.27, 9802.69].
2. Age group 51-55 has the highest purchase value range of [9218.76, 9861.45].

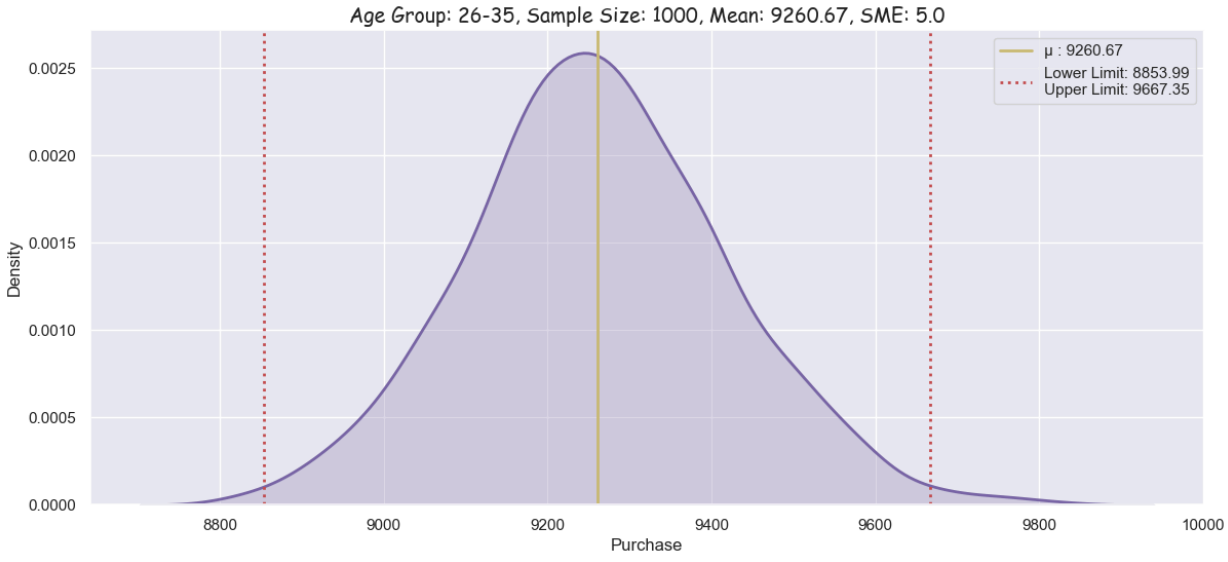
All the age groups still have overlap which makes it difficult to interpret the ranges.

- Now, let's visualize the graphs of 1000 mean values of purchase samples for a sample size of 1000 for all the age groups with a 99% confidence interval.

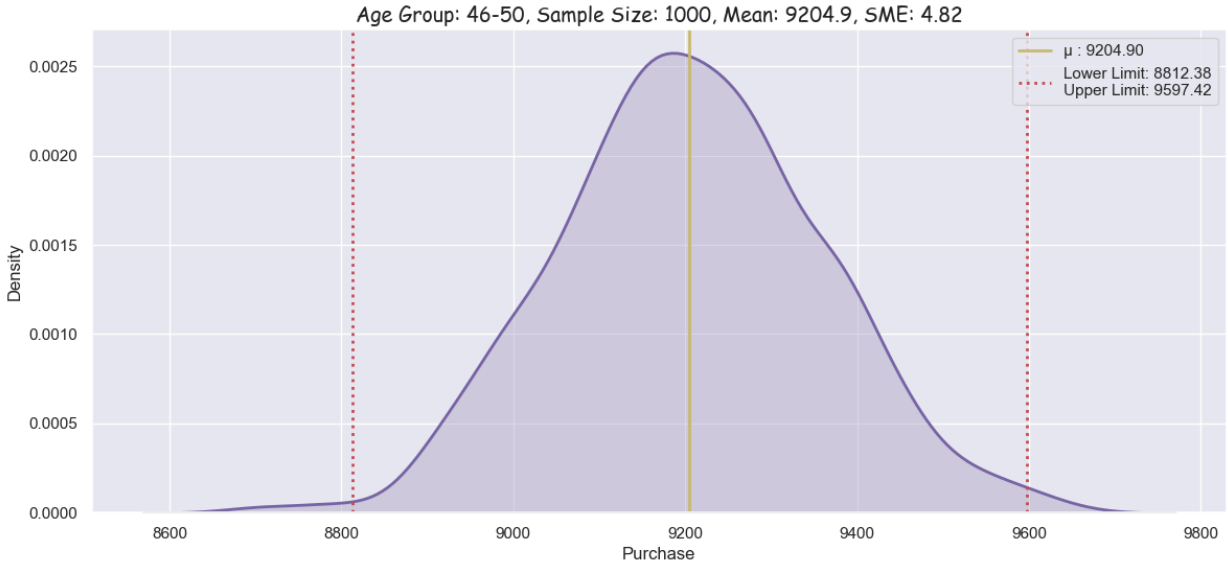
In [63]:
```
ci = 99
itr_size = 1000
sample_size = 1000
flag = 0
for i in age_group:
    m_avg, ll, ul, mean = bootstrap_age(df[df['Age']==i]['Purchase'], sample_size, itr
    res = res.append({'Age_Group':i, 'Sample Size':sample_size, 'Lower Limit':ll, 'Upp
```

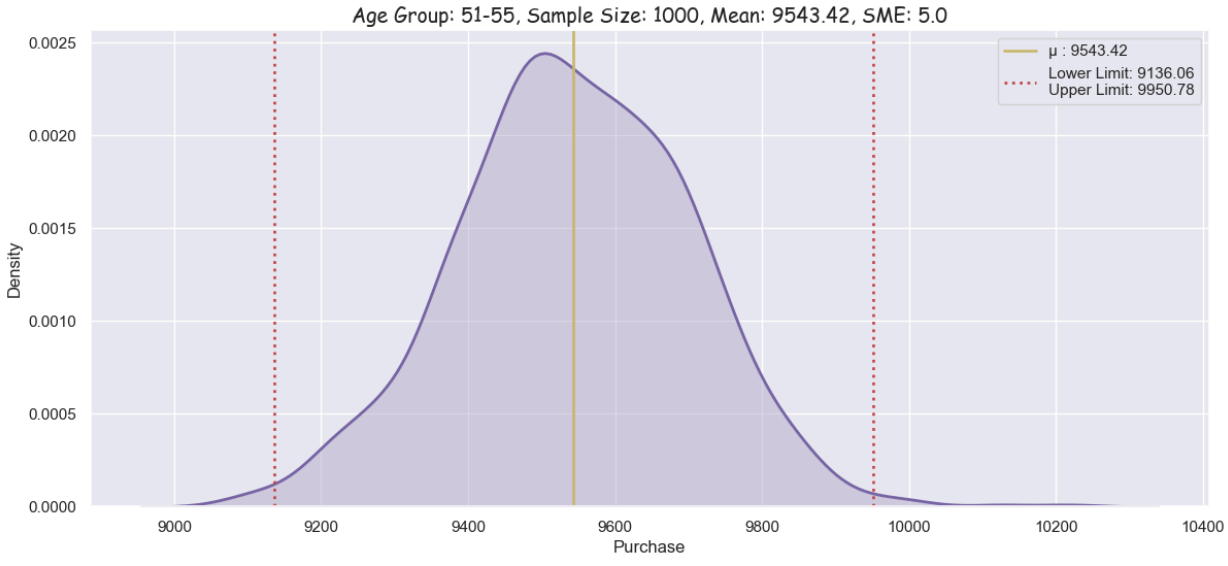

Age Group: 0-17, Sample Size: 1000, Mean: 8936.99, SME: 5.0
μ : 8936.99
Lower Limit: 8529.82
Upper Limit: 9344.16
Density
Purchase

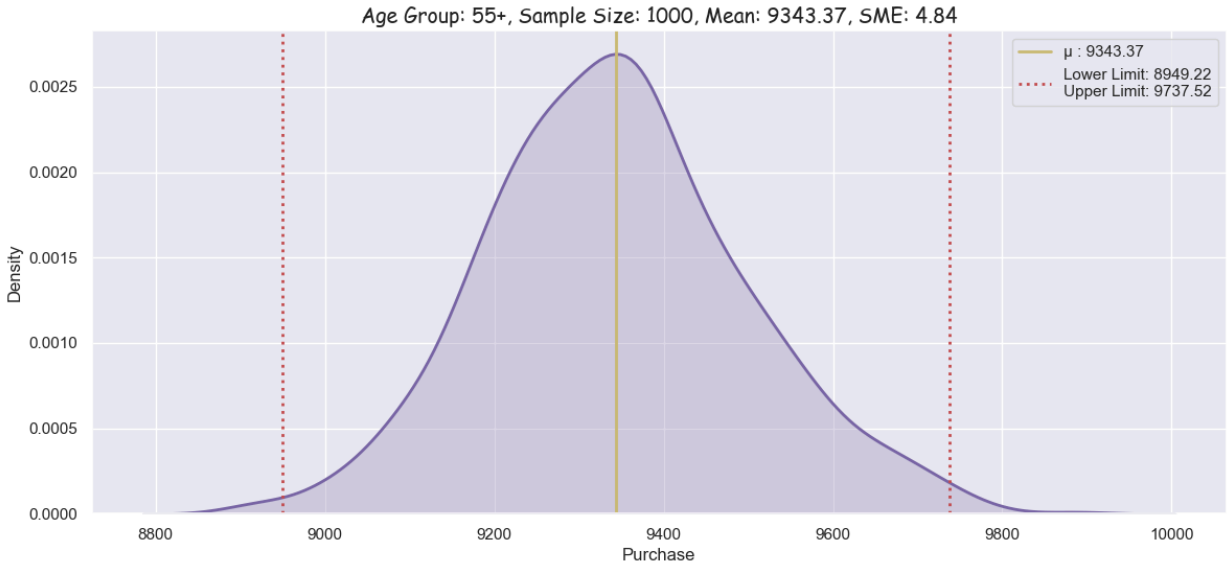

Age Group: 18-25, Sample Size: 1000, Mean: 9166.53, SME: 4.98
μ : 9166.53
Lower Limit: 8761.48
Upper Limit: 9571.59
Density
Purchase


Age Group: 26-35, Sample Size: 1000, Mean: 9260.67, SME: 5.0
μ : 9260.67
Lower Limit: 8853.99
Upper Limit: 9667.35
Density
Purchase


Age Group: 36-45, Sample Size: 1000, Mean: 9329.4, SME: 4.96
μ : 9329.40
Lower Limit: 8925.68
Upper Limit: 9733.12
Density
Purchase


Age Group: 46-50, Sample Size: 1000, Mean: 9204.9, SME: 4.82
μ : 9204.90
Lower Limit: 8812.38
Upper Limit: 9597.42
Density
Purchase


Age Group: 51-55, Sample Size: 1000, Mean: 9543.42, SME: 5.0
μ : 9543.42
Lower Limit: 9136.06
Upper Limit: 9950.78
Density
Purchase



In [64]:
```
print(res)
```

	Age_Group	Sample Size	Lower Limit	Upper Limit	Sample Mean	\
0	0-17	1000	8671.92	9182.43	8927.18	
1	18-25	1000	8922.45	9425.16	9173.81	
2	26-35	1000	8985.72	9504.16	9244.94	
3	36-45	1000	9078.16	9591.30	9334.73	
4	46-50	1000	8945.64	9467.54	9206.59	
5	51-55	1000	9262.42	9796.34	9529.38	
6	55+	1000	9092.10	9577.34	9334.72	
7	0-17	1000	8622.01	9241.73	8931.87	
8	18-25	1000	8855.65	9486.59	9171.12	
9	26-35	1000	8952.79	9584.55	9268.67	
10	36-45	1000	9021.19	9643.14	9332.16	
11	46-50	1000	8903.69	9512.61	9208.15	
12	51-55	1000	9201.24	9865.69	9533.46	
13	55+	1000	9034.65	9634.39	9334.52	
14	0-17	1000	8529.82	9344.16	8936.99	
15	18-25	1000	8761.48	9571.59	9166.53	
16	26-35	1000	8853.99	9667.35	9260.67	
17	36-45	1000	8925.68	9733.12	9329.40	
18	46-50	1000	8812.38	9597.42	9204.90	
19	51-55	1000	9136.06	9950.78	9543.42	
20	55+	1000	8949.22	9737.52	9343.37	

	Confidence Interval	Interval Range	Range
0	90	[8671.92, 9182.43]	510.51
1	90	[8922.45, 9425.16]	502.71
2	90	[8985.72, 9504.16]	518.44
3	90	[9078.16, 9591.3]	513.14
4	90	[8945.64, 9467.54]	521.90
5	90	[9262.42, 9796.34]	533.92
6	90	[9092.1, 9577.34]	485.24
7	95	[8622.01, 9241.73]	619.72
8	95	[8855.65, 9486.59]	630.94
9	95	[8952.79, 9584.55]	631.76
10	95	[9021.19, 9643.14]	621.95
11	95	[8903.69, 9512.61]	608.92
12	95	[9201.24, 9865.69]	664.45
13	95	[9034.65, 9634.39]	599.74
14	99	[8529.82, 9344.16]	814.34
15	99	[8761.48, 9571.59]	810.11
16	99	[8853.99, 9667.35]	813.36
17	99	[8925.68, 9733.12]	807.44
18	99	[8812.38, 9597.42]	785.04
19	99	[9136.06, 9950.78]	814.72
20	99	[8949.22, 9737.52]	788.30

We can observe with 99% confidence that:

1. Age group 0-17 has the least purchase value range of [8543.18, 9341.05].
2. Age group 51-55 has the highest purchase value range of [9134.55, 9943.98].
3. Across different confidence intervals (90%, 95%, and 99%), there is considerable overlap in the interval ranges for all age groups.
4. This suggests that age group may not significantly influence customer spending behavior, as indicated by the overlapping confidence intervals.

Inferences

1. 80% of users fall between the ages of 18-50, with the majority (40%) falling within the 26-35 age bracket, followed by 18% in the 18-25 age group and 20% in the 36-45 age group.
2. 75% of users are male, while 25% are female. Males exhibit higher purchasing activity compared to females.
3. 59% of users are single, whereas 41% are married.
4. 35% of users have been residing in the city for 1 year, 18% for 2 years, and 17% for 3 years.
5. Although the majority of customers originate from City Category B, those from City Category C tend to spend more, with an average expenditure of 9719.
6. While the majority of users hail from City Category C, a larger proportion of purchases occur in City Category B. This implies repeated visits to malls in City Category B by the same set of users.
7. The majority of purchases fall within the range of 5,000 to 20,000.
8. The predominant age group among mall customers is 26-35, accounting for 60% of purchases made by individuals aged 26-45.
9. City Category B constitutes 42% of total purchases, followed by City Category C at 31%, with City Category A representing 27%. Notably, purchases are highest in City Category C.
10. In City Category C, there is a slightly higher proportion of female customers.
11. Products 5 and 8 are commonly purchased by females.

Recommendations

Customer Segmentation Action Plan:

```
- Targeted Marketing & Customer Acquisition:
```

1. Focus on Male Retention & Acquisition: Leverage targeted marketing campaigns tailored to male preferences to retain existing male customers and attract new ones.
2. Unmarried Customer Acquisition: Prioritize acquiring unmarried customers due to their higher spending potential.
3. Target High-Value Age Group: Attract and retain customers aged 18-45, as they contribute significantly to overall sales.
4. Geographic Targeting: Enhance product offerings and marketing efforts specifically for male customers residing in City Category C to capitalize on their higher spending habits.

- Product Strategy:

1. Maximize Sales of Popular Products: Develop strategies to increase sales of high-purchasing frequency products (categories 1, 5, 8, and 11).
2. Promote Less Popular Products: Implement strategies alongside popular products to encourage sales diversification.

- Enhancing Customer Experience:

1. Attract Younger Customers: Implement interactive games or entertainment activities to attract younger demographics and boost foot traffic.
2. Targeted Offers for Families: Introduce special promotions for families with children (0-17 years old) to incentivize shopping trips.
3. Engage Young Shoppers: Organize events or activities specifically targeted towards younger generations to create a vibrant mall environment that appeals to their interests.

- Additional Considerations:

1. Tailored Marketing for Females: Develop marketing strategies that address the specific needs and preferences of female customers to increase their engagement and spending (e.g., special offers for Black Friday).

In [ ]: