Business Case: Walmart - Confidence Interval and CLT

About Walmart

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

Business Problem

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

Dataset

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

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

In [ ]:

In [ ]:

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')
```

In [2]:
```
df=pd.read_csv("walmart_data.csv")
```

In [3]:
```
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
```

Converting datatypes for the columns in object type

In [4]:
```
df = df.convert_dtypes()
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  string
 2   Gender                      550068 non-null  string
 3   Age                         550068 non-null  string
 4   Occupation                  550068 non-null  Int64
 5   City_Category               550068 non-null  string
 6   Stay_In_Current_City_Years  550068 non-null  string
 7   Marital_Status              550068 non-null  Int64
 8   Product_Category            550068 non-null  Int64
 9   Purchase                    550068 non-null  Int64
dtypes: Int64(5), string(5)
memory usage: 44.6 MB
```

Finding number of unique values in each column:

In [5]:
```
df.nunique()
```

Out[5]:
```
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
```

A quick description of data:

In [6]:
```
df.describe()
```

Out[6]:

	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

There are 550068 rows and 10 columns in the data

In [7]:
```
df.shape
```

Out[7]:
```
(550068, 10)
```

Checking Null values: There are no Null values in the data.

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
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
```

Checking NA values: No NA values in the data

In [9]:
```
df.isna().sum()
```

Out[9]:
```
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
```

Checking Duplicates: No duplicates in the data

In [10]:
```
df.duplicated().sum()
```

Out[10]:
```
0
```

Quick sample of data:

In [11]:
```
df.head(10)
```

Out[11]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969
5	1000003	P00193542	M	26-35	15	A	3	0	1	15227
6	1000004	P00184942	M	46-50	7	B	2	1	1	19215
7	1000004	P00346142	M	46-50	7	B	2	1	1	15854
8	1000004	P0097242	M	46-50	7	B	2	1	1	15686
9	1000005	P00274942	M	26-35	20	A	1	1	8	7871

There are 3 tier cities:

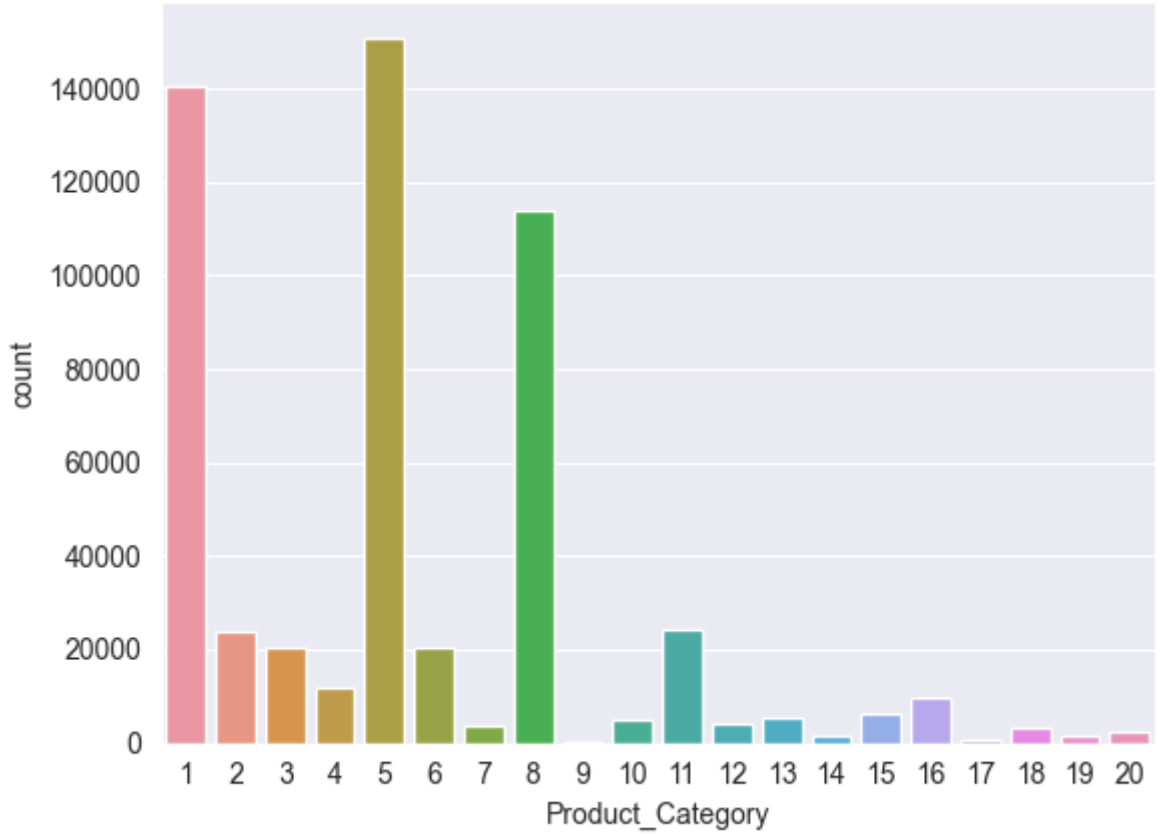
In [12]:
```
df["City_Category"].value_counts()
```

Out[12]:
```
B    231173
C    171175
A    147720
Name: City_Category, dtype: Int64
```

Purchase distribution by Product Category:

In [59]:
```
sns.countplot(data=df,x=df['Product_Category'])
```

Out[59]:
```
<Axes: xlabel='Product_Category', ylabel='count'>
```



Product Category 1, 5, 8 are more popular

In [14]:
```
df["Stay_In_Current_City_Years"].value_counts()
```

Out[14]:
```
1     193821
2     101838
3      95285
4+     84726
0      74398
Name: Stay_In_Current_City_Years, dtype: Int64
```

In [15]:
```
df["Age"].value_counts()
```

Out[15]:
```
26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
Name: Age, dtype: Int64
```

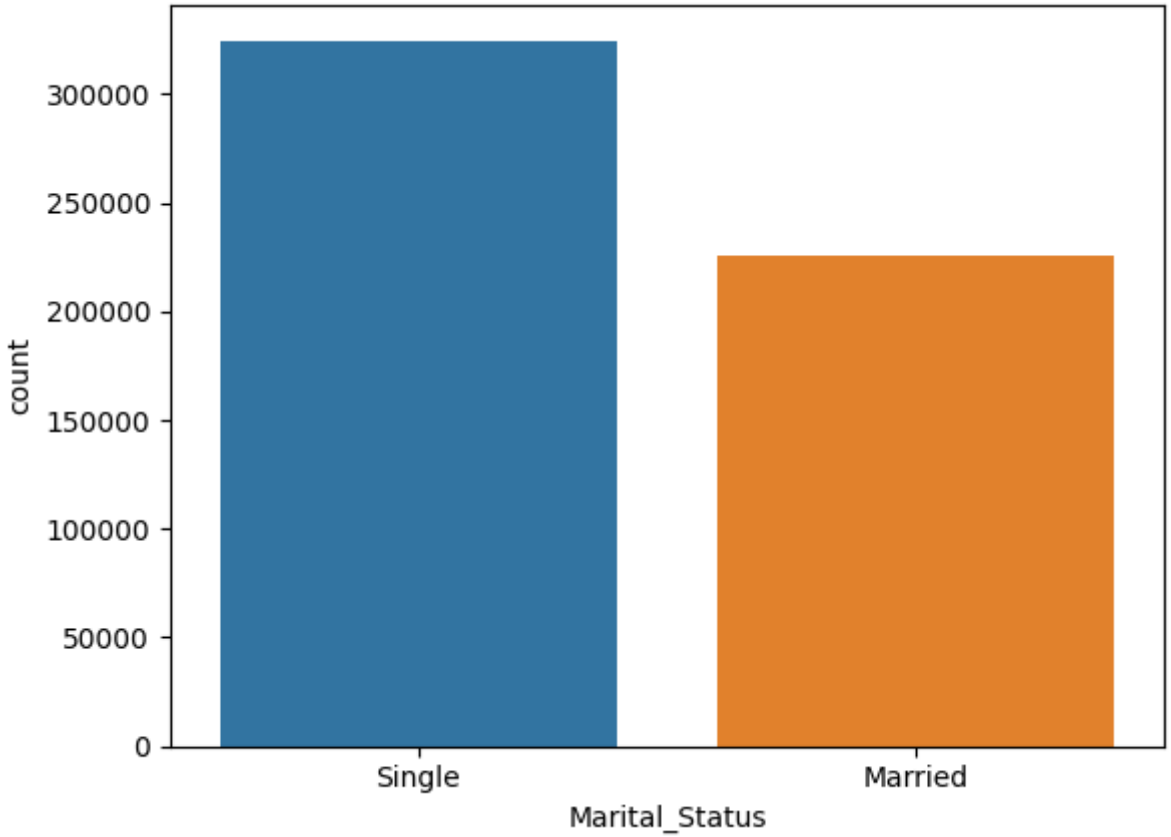
In [16]:
```
df["Marital_Status"]=df["Marital_Status"].astype('string')
```

Convert Marital Status data and get value counts of each values:

In [17]:
```
df["Marital_Status"].replace({"0":"Single",
                              "1":"Married"},inplace=True)
```

In [18]:
```
sns.countplot(data=df,x=df['Marital_Status'])
```

Out[18]:
```
<Axes: xlabel='Marital_Status', ylabel='count'>
```

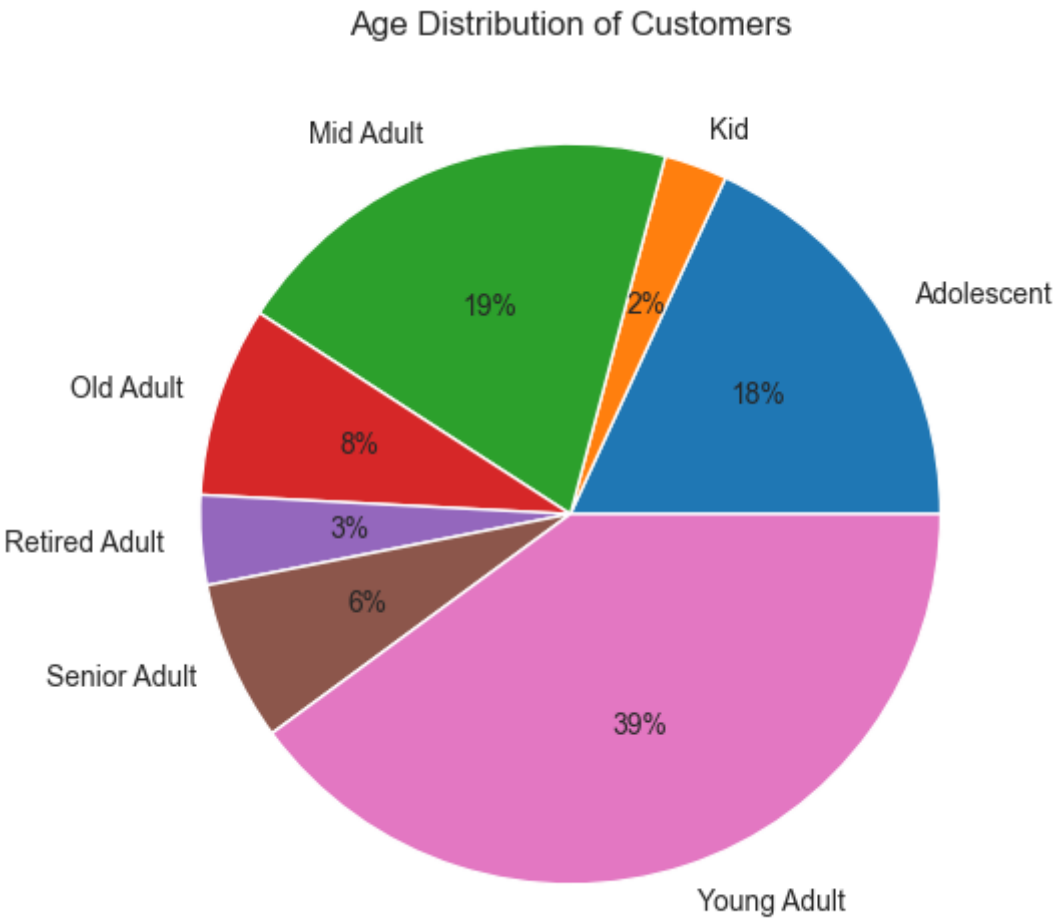


Give a name for each age group:

In [19]:
```
df["Age"].replace({"0-17":"Kid",
"18-25":"Adolescent",
"26-35":"Young Adult",
"36-45":"Mid Adult",
"46-50":"Old Adult",
"51-55":"Senior Adult",
"55+":"Retired Adult"},inplace=True)
```

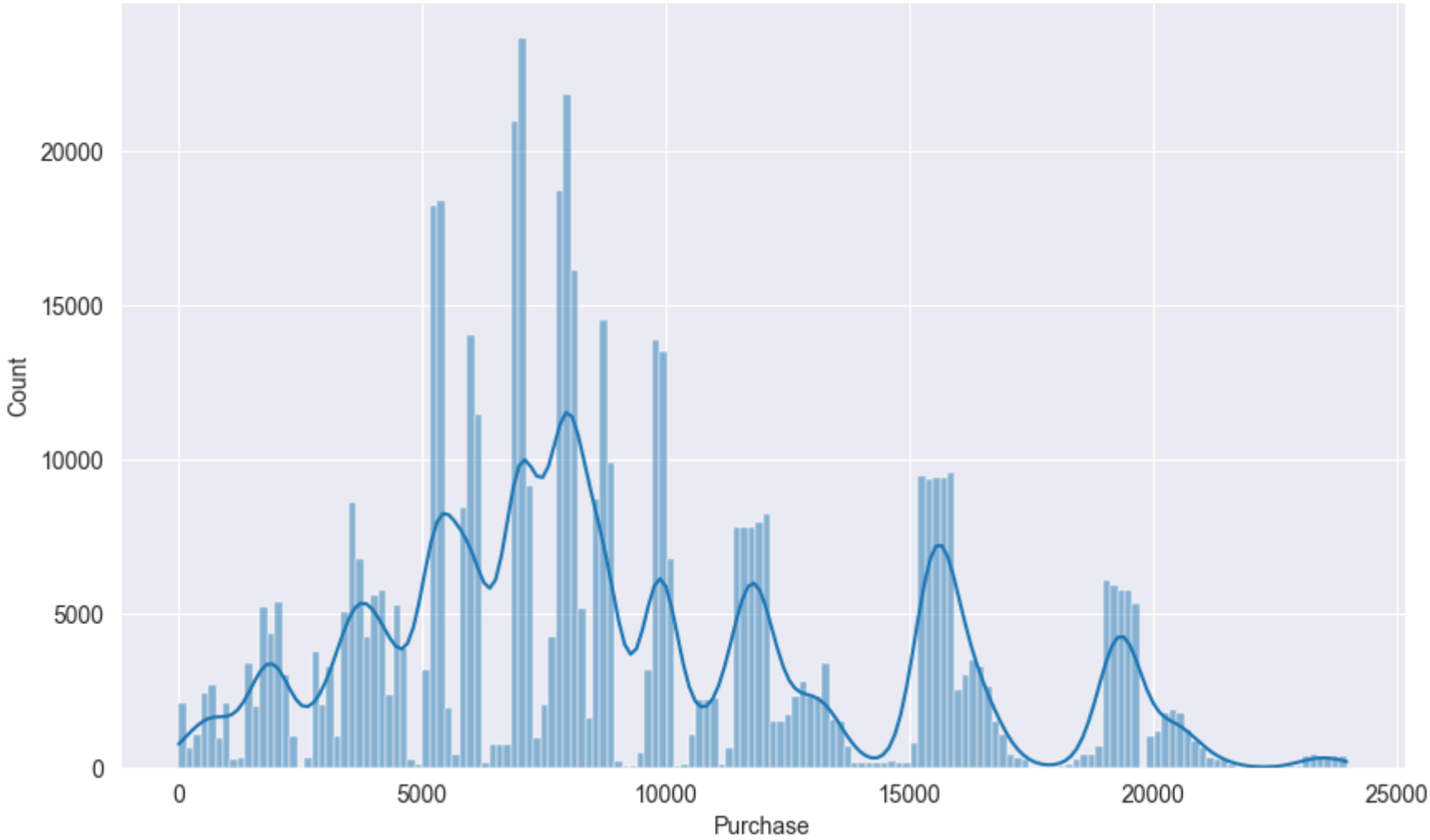
Age group distribution:

In [60]:
```
# Comparison of Age of customers
x = df.groupby(['Age'])['Age'].count()
y = len(df)
r = ((x/y) * 100).round(2)
mf_ratio = pd.DataFrame(r)
mf_ratio.rename({'Age': '%'}, axis=1, inplace=True)
plt.figure(figsize=(12, 6))
plt.pie(mf_ratio['%'], labels=mf_ratio.index,autopct='%i%%')
plt.title('Age Distribution of Customers')
plt.show()
```



Young Adults 26-35 age group contribute mostly to the sales on Black Friday

In [61]:
```
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='Purchase', kde=True)
plt.show()
```



Most of the sales are between 5000-10000

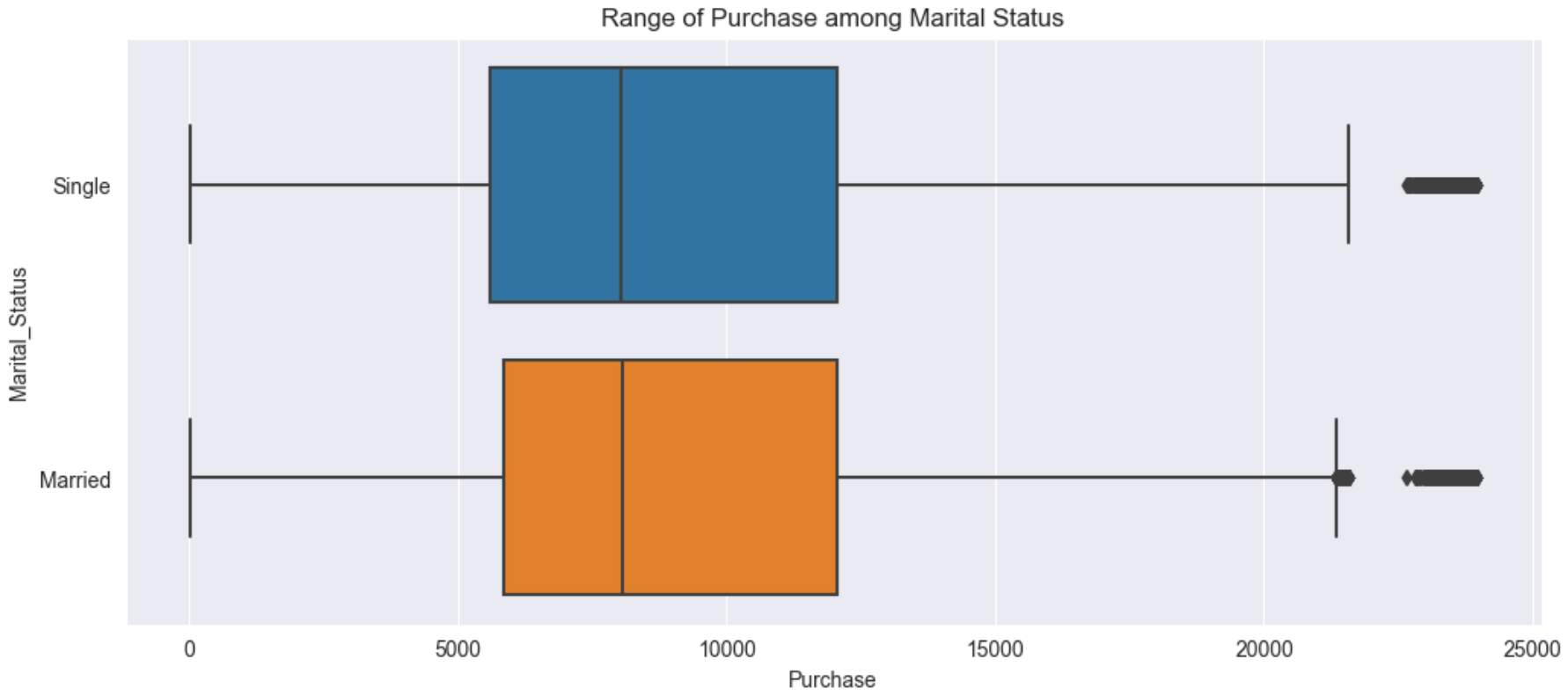
Genderwise purchase distribution:

In [62]:
```
plt.figure(figsize=(12,5))
sns.boxplot(x='Purchase',y='Gender',data=df)
plt.title('Range of Purchase among Gender')
plt.show()
```

Avg. Sales by Male are sligthly more than Female

Genderwise purchase distribution:

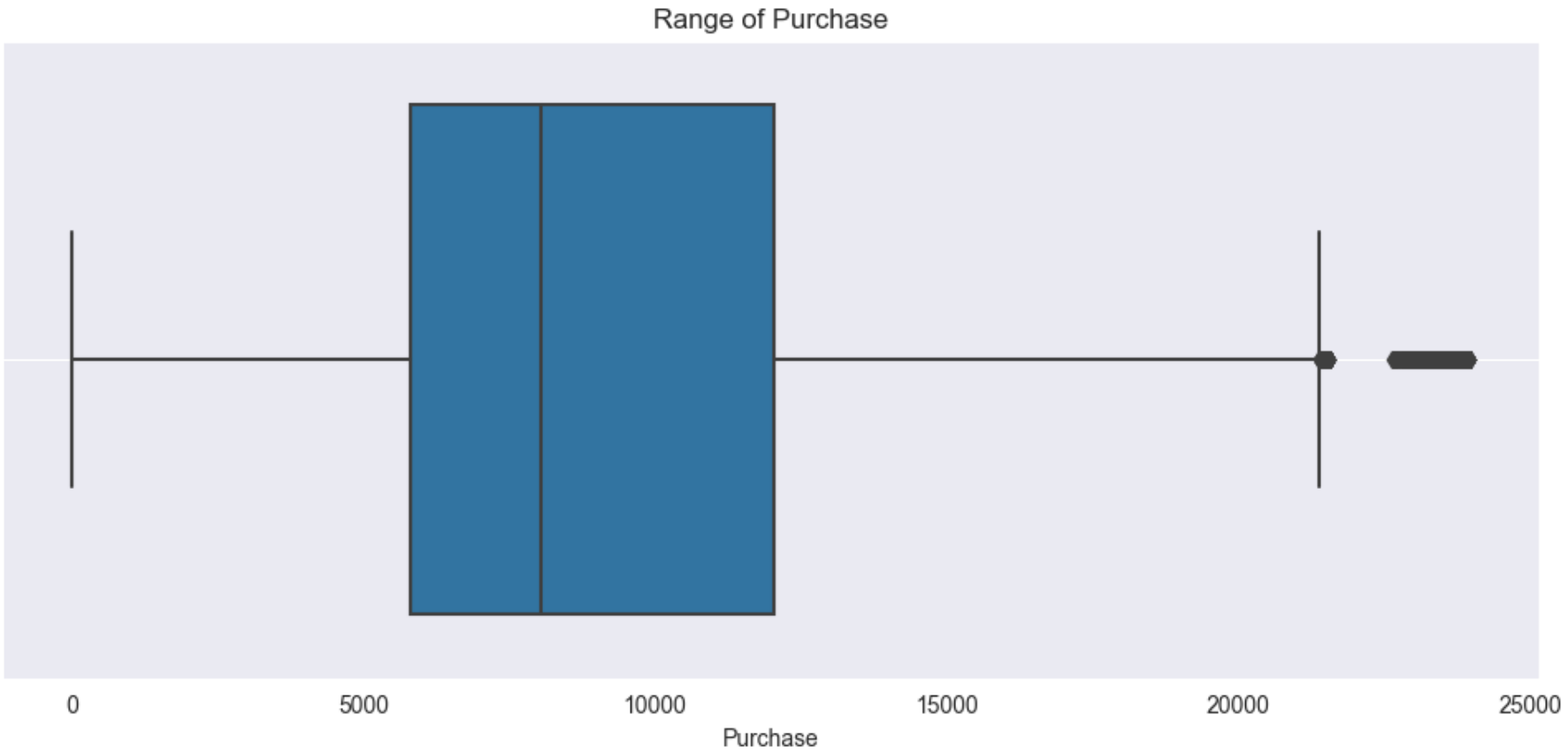
In [63]:
```
plt.figure(figsize=(12,5))
sns.boxplot(x='Purchase',y='Marital_Status',data=df)
plt.title('Range of Purchase among Marital Status')
plt.show()
```



Marital Status has no major impact on Avg. Purchases.

Overall purchase distribution:

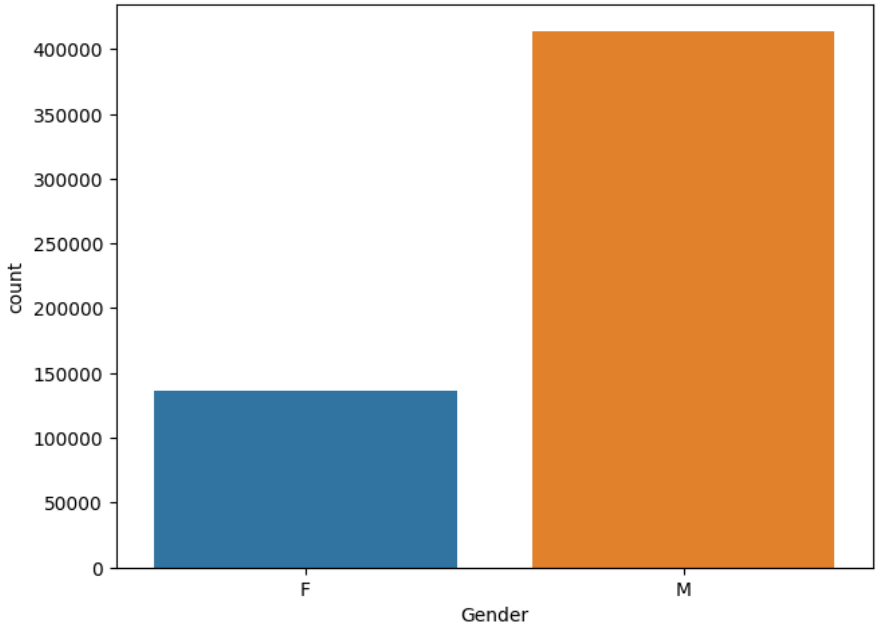
In [65]:
```
plt.figure(figsize=(12,5))
sns.boxplot(x='Purchase',data=df)
plt.grid()
plt.title('Range of Purchase')
plt.show()
```

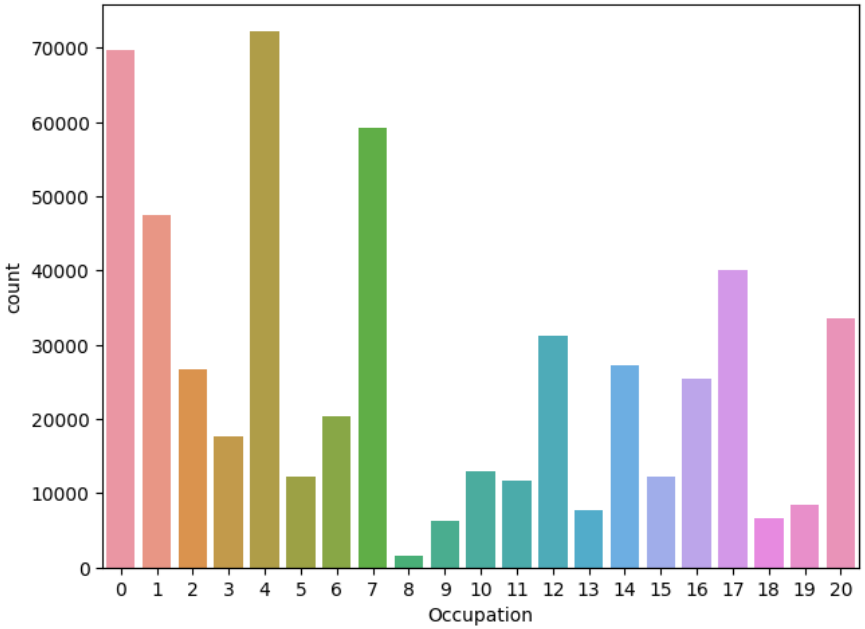


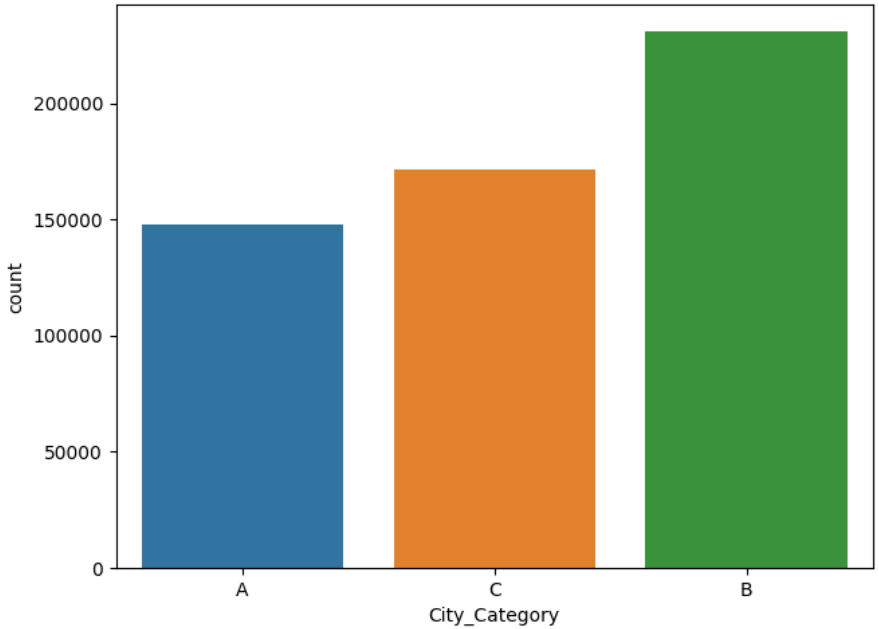
Overall average purchase is about 8000

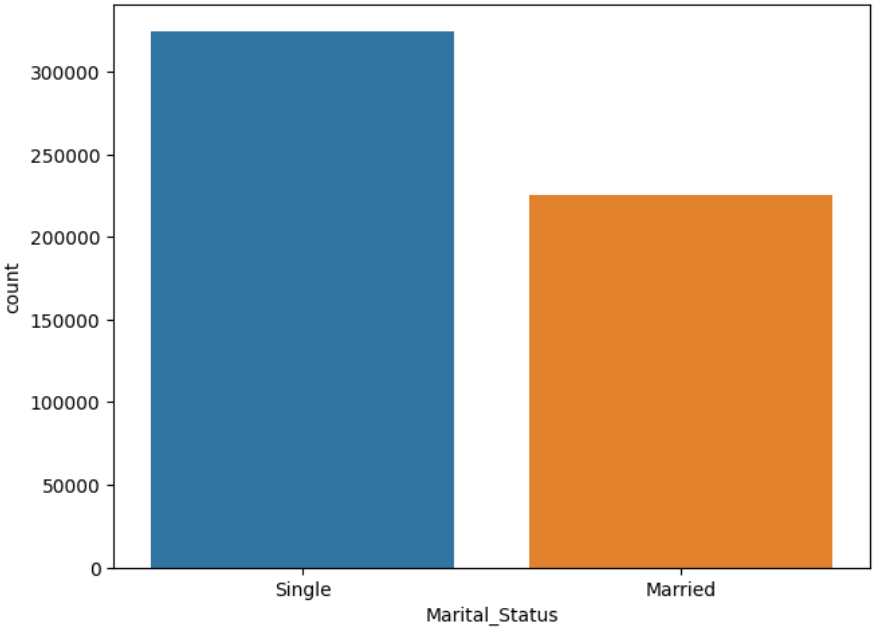
Univariate Analysis

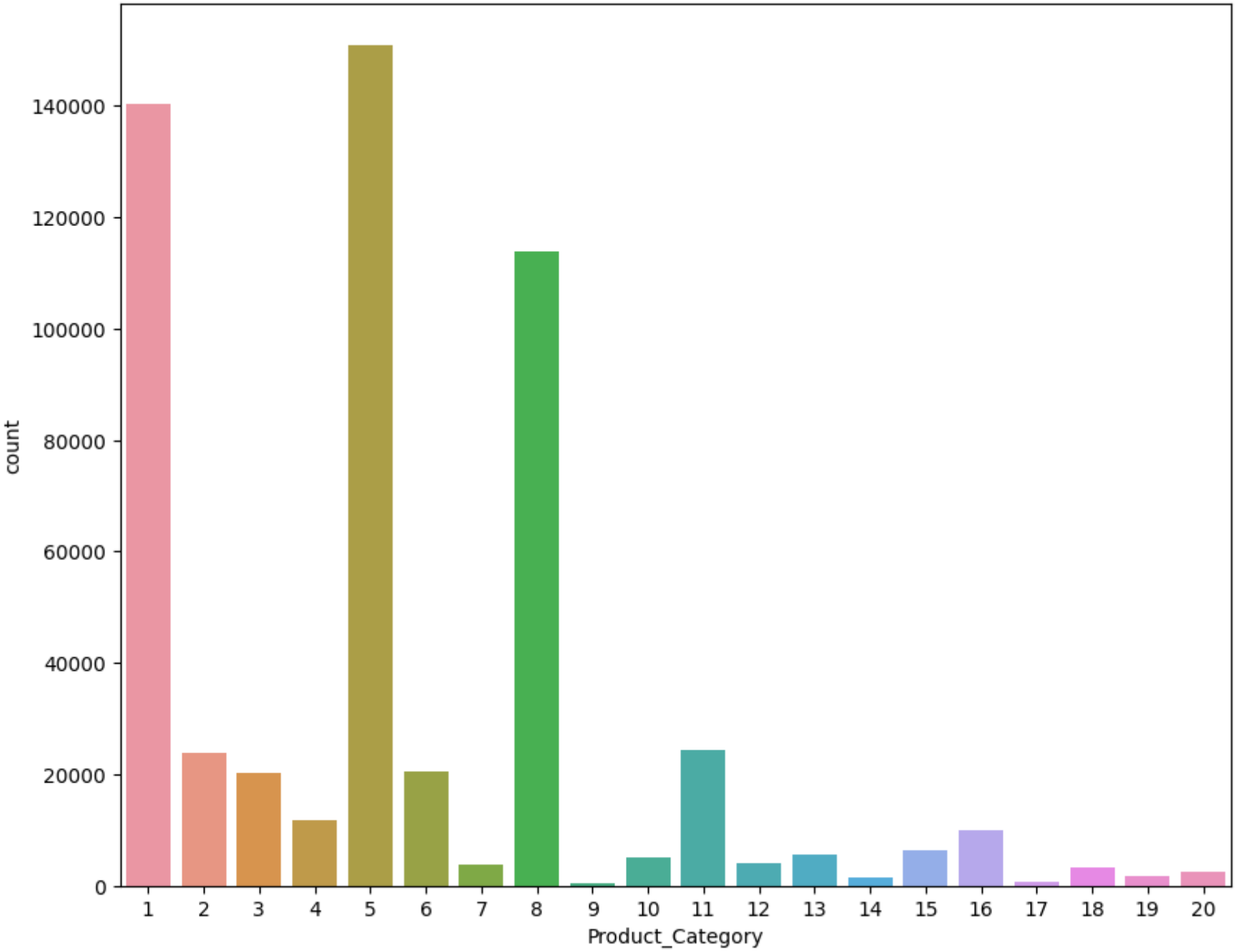
In [25]:
```
categorical_cols = ['Gender', 'Occupation','City_Category','Marital_Status','Product_Category']

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16, 12))
sns.countplot(data=df, x='Gender', ax=axs[0,0])
sns.countplot(data=df, x='Occupation', ax=axs[0,1])
sns.countplot(data=df, x='City_Category', ax=axs[1,0])
sns.countplot(data=df, x='Marital_Status', ax=axs[1,1])
plt.show()

plt.figure(figsize=(10, 8))
sns.countplot(data=df, x='Product_Category')
plt.show()
```


count
400000
350000
300000
250000
200000
150000
100000
50000
0
F
M
Gender


count
70000
60000
50000
40000
30000
20000
10000
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Occupation


count
200000
150000
100000
50000
0
A
C
B
City_Category


count
300000
250000
200000
150000
100000
50000
0
Single
Married
Marital_Status


count
0
20000
40000
60000
80000
100000
120000
140000
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Product_Category

Observations:

Most of the users are Male

There are 20 different types of Occupation and Product_Category

More users belong to B City_Category

More users are Single as compare to Married

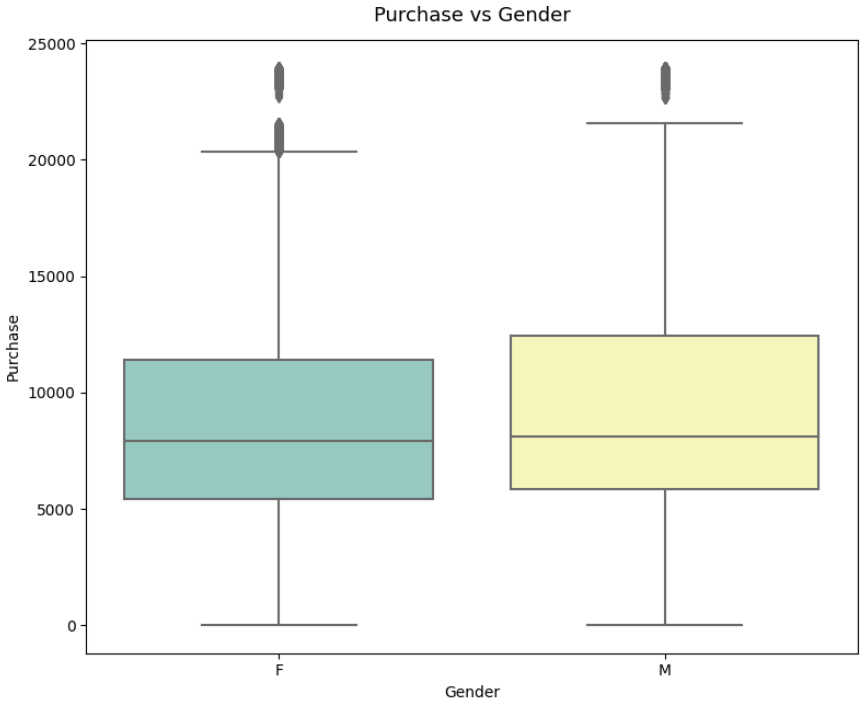
Product_Category - 1, 5, 8, & 11 have highest purchasing frequency.

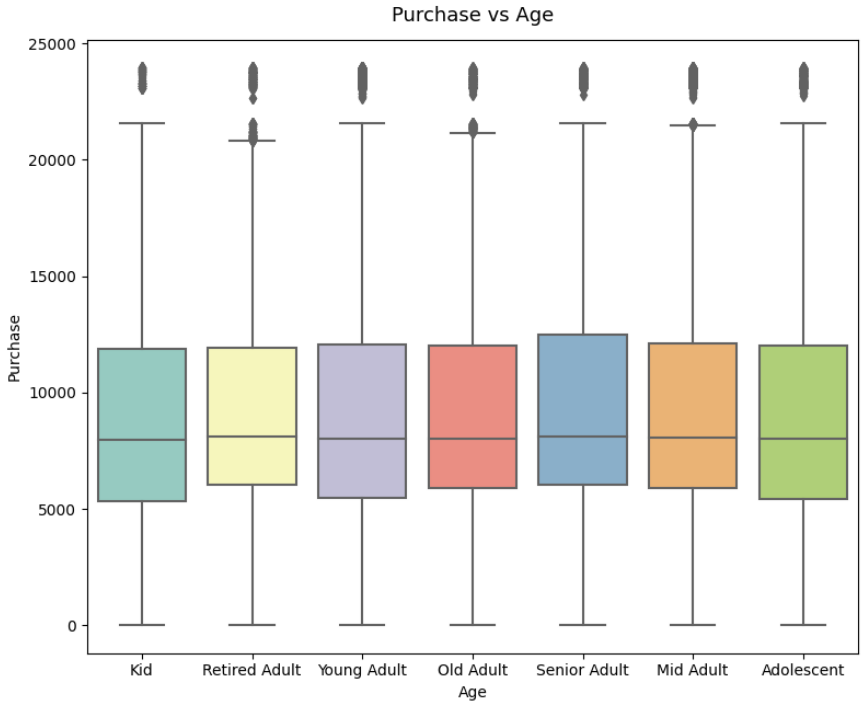
People with occupations 0, 4 and 7 make more purchase than other occupations. There are 21 different occupations.

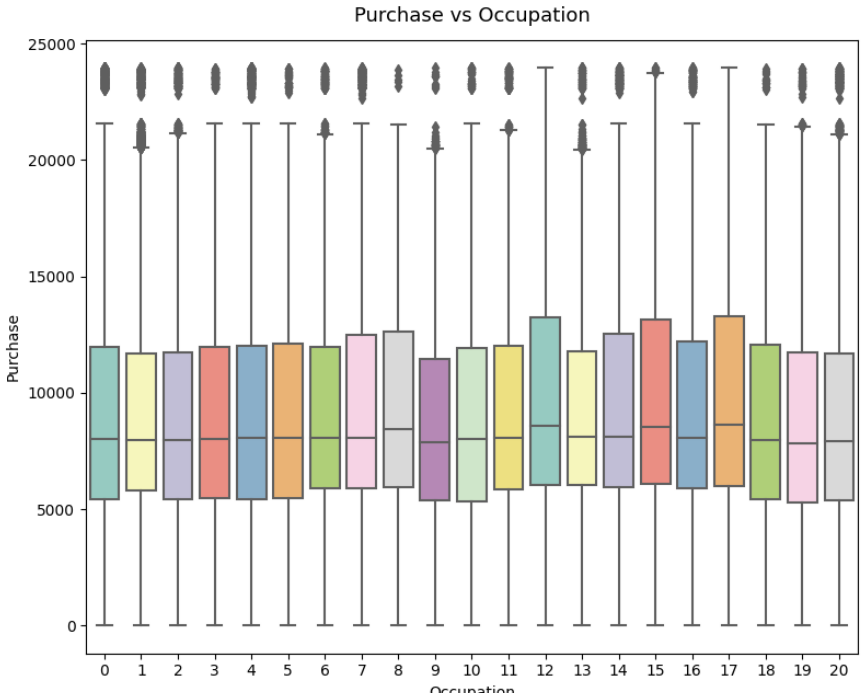
There are many outliers and median purchase is around 8000

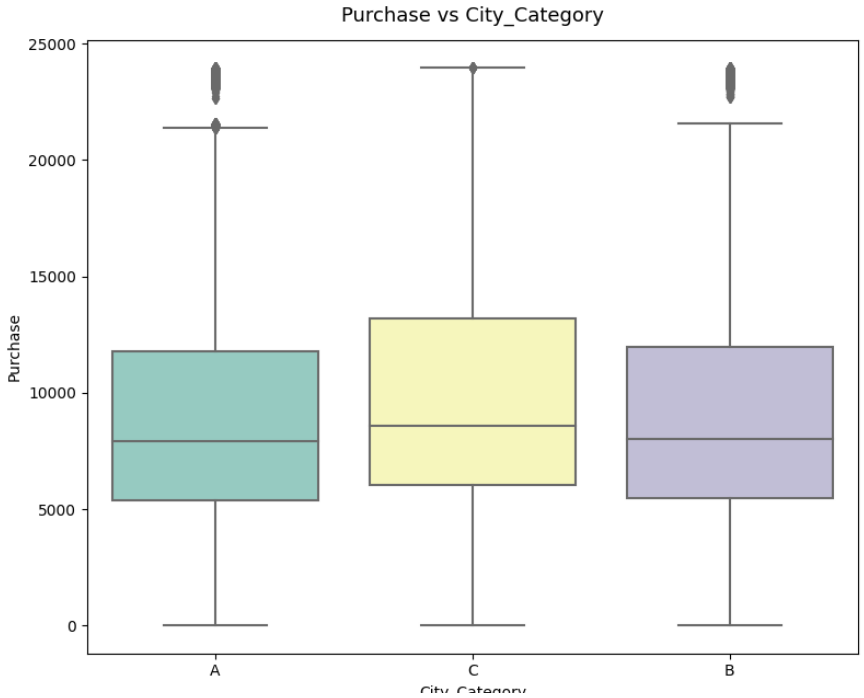
Bivariate Analysis

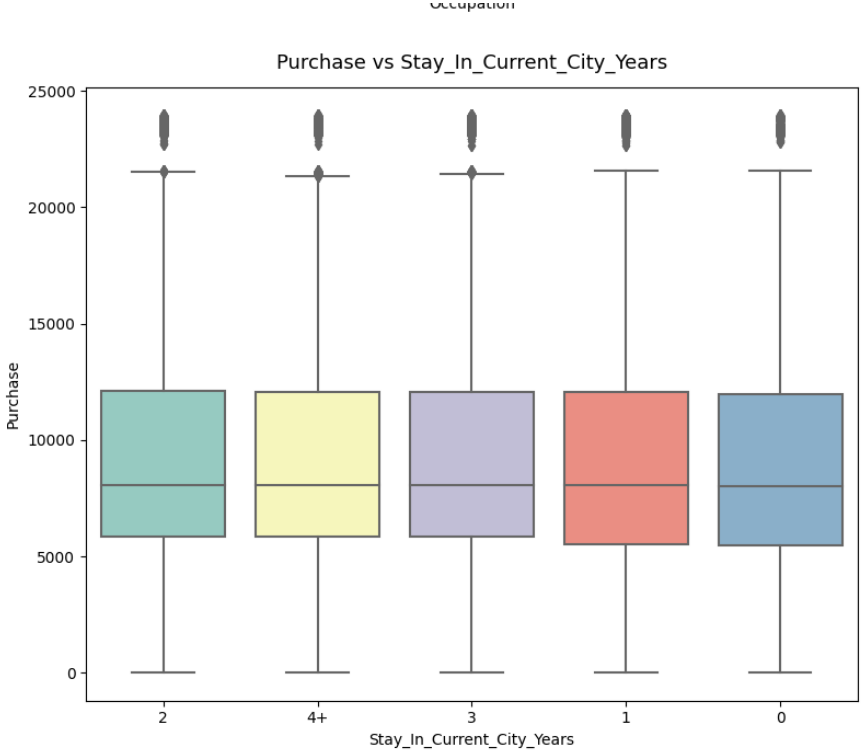
In [26]:
```
attrs = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Cate

fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(20, 16))
fig.subplots_adjust(top=1.3)
count = 0
for row in range(3):
    for col in range(2):
        sns.boxplot(data=df, y='Purchase', x=attrs[count], ax=axs[row, col], palette='Set3')
        axs[row,col].set_title(f"Purchase vs {attrs[count]}", pad=12, fontsize=13)
        count += 1
plt.show()
```

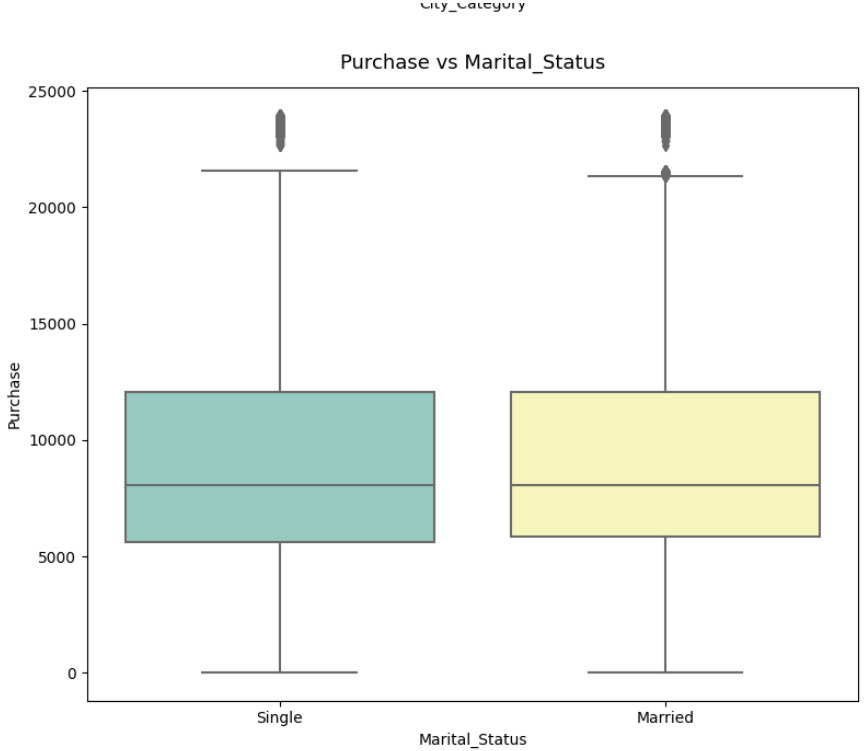

Purchase vs Gender
Purchase
Gender
F
M
0
5000
10000
15000
20000
25000


Purchase vs Age
Purchase
Age
Kid
Retired Adult
Young Adult
Old Adult
Senior Adult
Mid Adult
Adolescent
0
5000
10000
15000
20000
25000


Purchase vs Occupation
Purchase
Occupation
0
5000
10000
15000
20000
25000
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20


Purchase vs City_Category
Purchase
City_Category
A
C
B
0
5000
10000
15000
20000
25000





Observations:

Avg purchase across all age groups are almost same.

Avg. purchase by Male are slighlty higher than Female

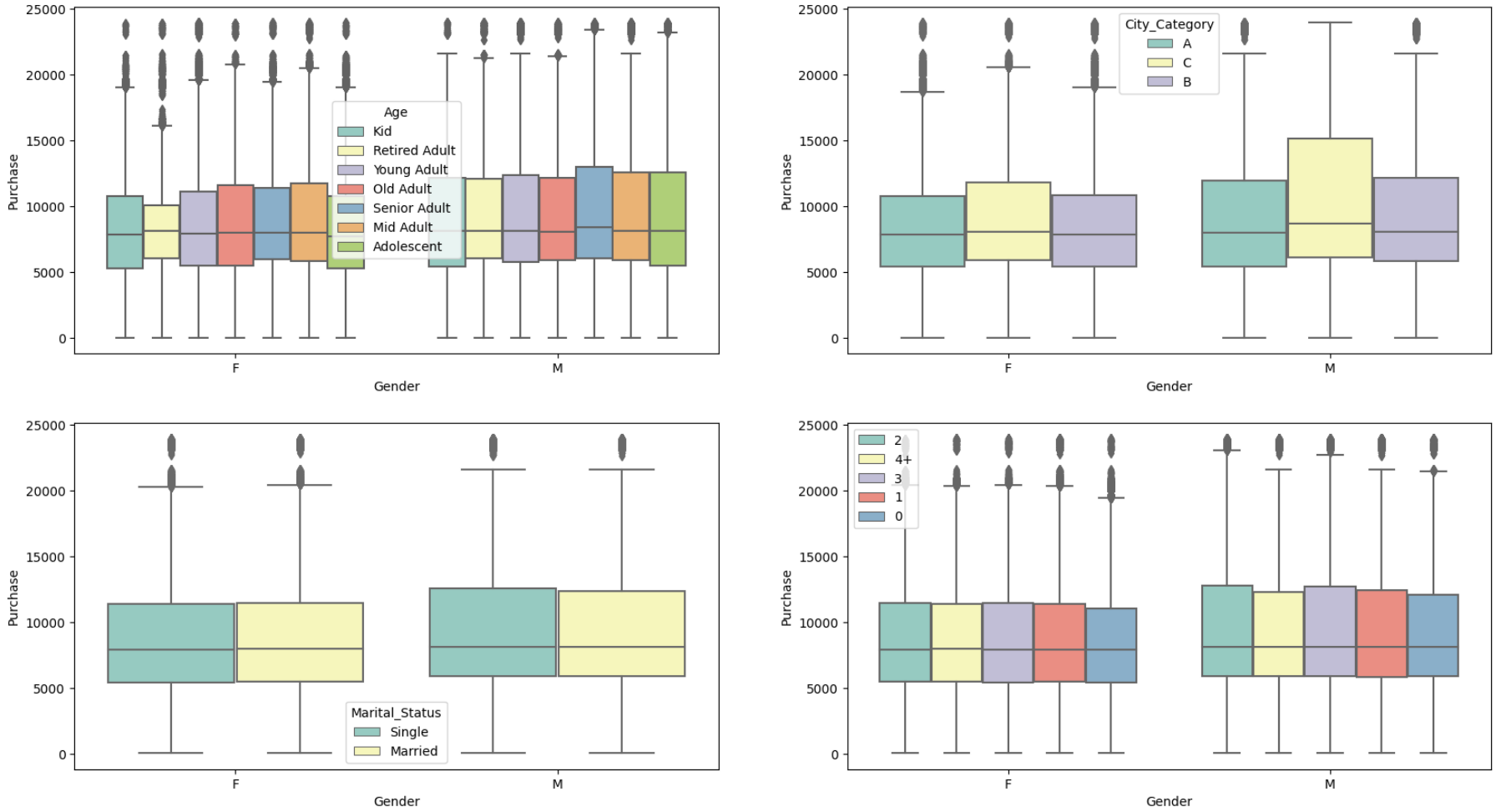
Avg. purchase across both Marital Status are same.

Avg. purchase across all Stay_in_current_city_years are almost.

Avg. puchase in city_category 'C' is slightly higher than A & B.

Multivariate Analysis:

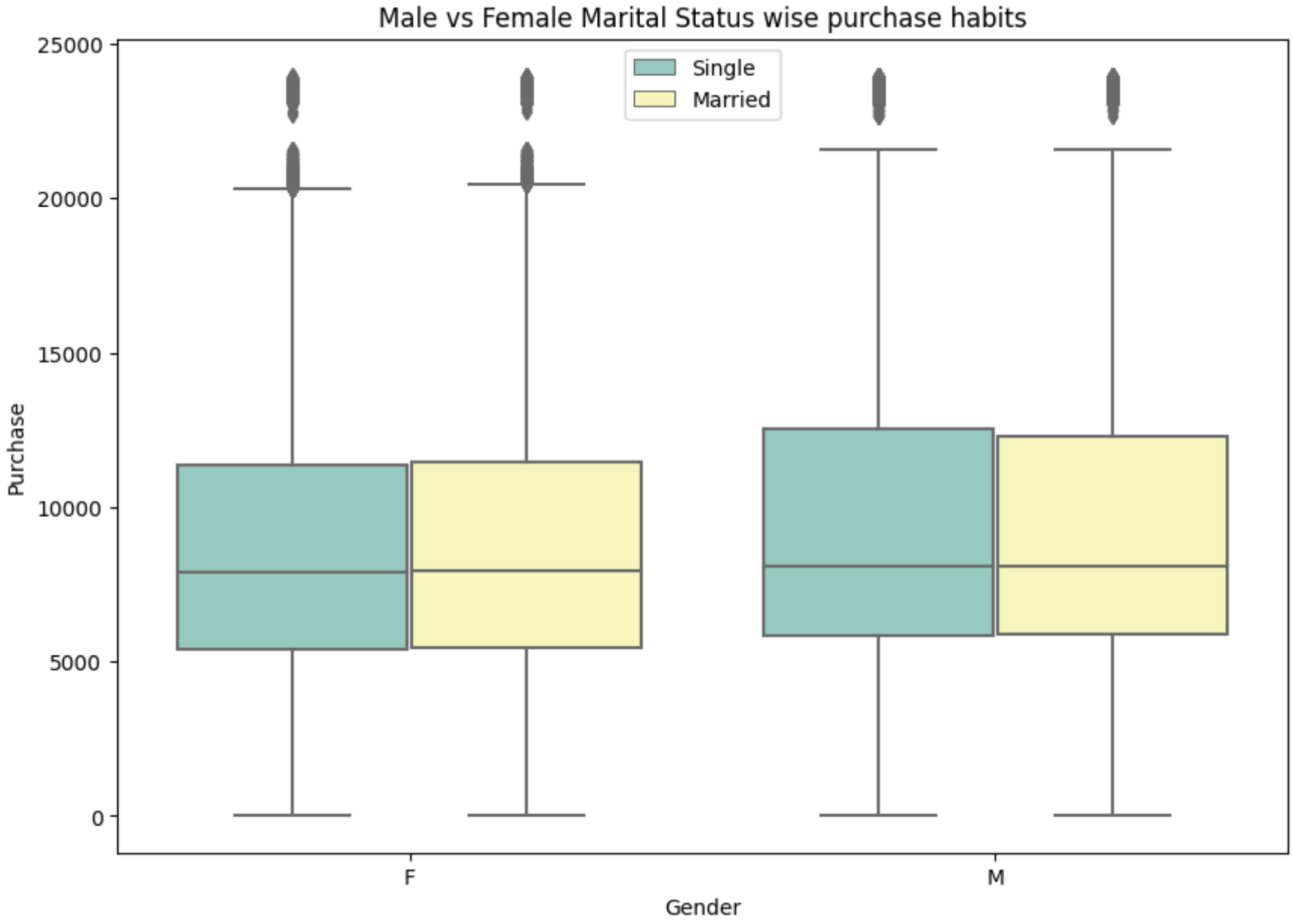
In [27]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(20, 6))
fig.subplots_adjust(top=1.5)
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age', palette='Set3', ax=axs[0,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category', palette='Set3', ax=axs[0,1])

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

plt.show()
```



The median value for females in city category C is highest compared to city A and B.

The median value for males in city category C is also highest compared to city A and B.

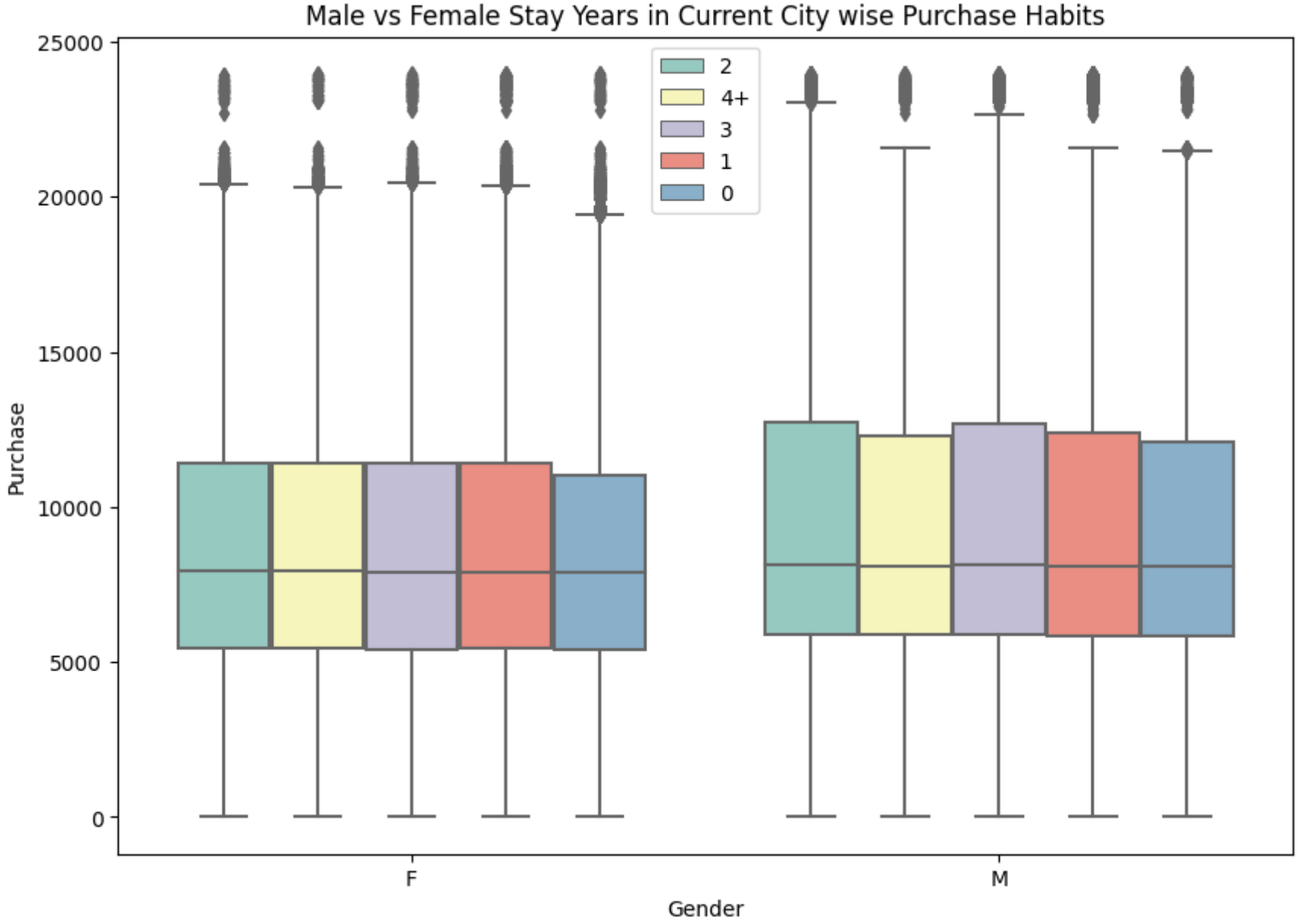
In [28]:
```
plt.figure(figsize = (10,7))
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status', palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Marital Status wise purchase habits')
plt.show()
```


Male vs Female Marital Status wise purchase habits
Single
Married
25000
20000
15000
10000
5000
0
Purchase
F
M
Gender

There is no effect of marital status on the spending habits of both the genders.

While we can observe that the median values for Male is higher comapred to Females.

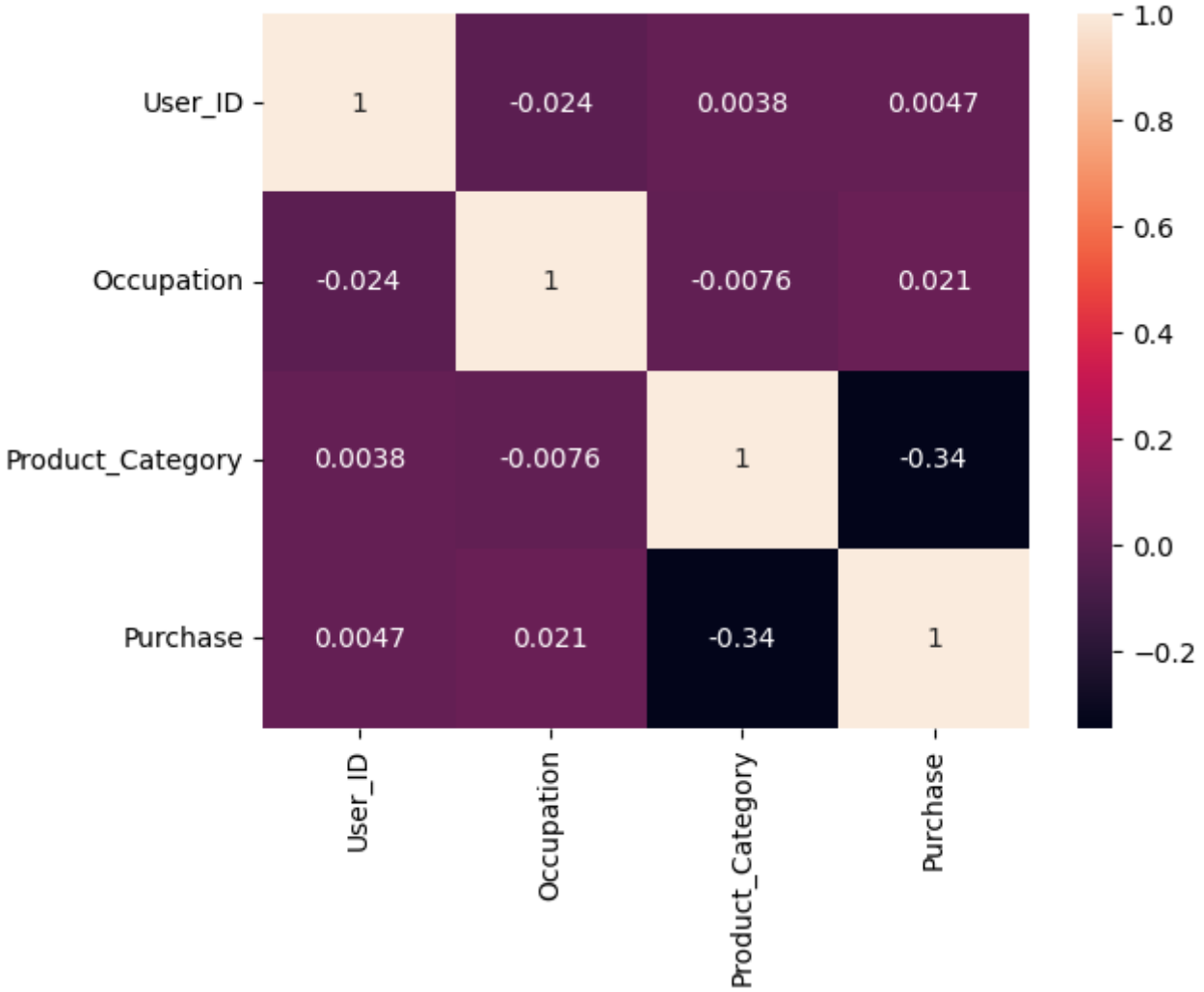
In [29]:
```
plt.figure(figsize = (10,7))
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Stay_In_Current_City_Years', palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Stay Years in Current City wise Purchase Habits')
plt.show()
```


Male vs Female Stay Years in Current City wise Purchase Habits
25000
20000
15000
10000
5000
0
Purchase
2
4+
3
1
0
F
M
Gender

We can observe for females the median values for purchase amount is a little lower for women staying for 3 and 0 years as compared to others.

For men, there is no much difference.

In [30]:
```
sns.heatmap(df.corr(), annot = True)
plt.show()
```



It is seen, High Negative Correlation(-0.0076) between Product Category and Occupation.

Then observed Slight Positive Correlation(0.021) between Purchase and Occupation.

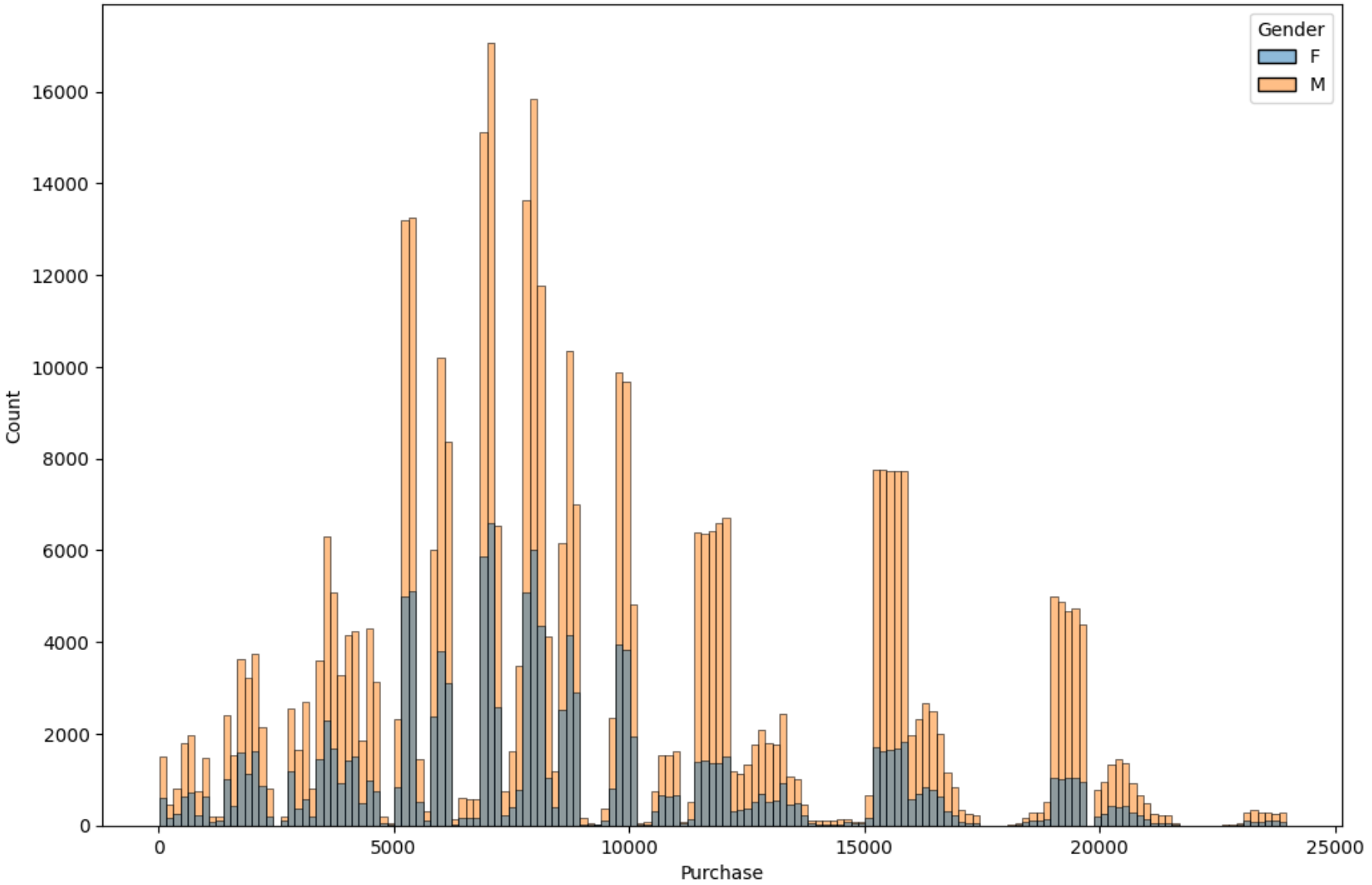
Also, there is Negative Correlation(-0.34) between Product Category and Purchase.

In [31]:
```
def getCLT(sampleA,sampleB,sampleSize,itrerationSize=1000,ci=90):
    ci = ci/100

    plt.figure(figsize=(16,8))
    sampleA_n = [np.mean(sampleA.sample(sampleSize)) for i in range(itrerationSize)]
    sampleB_n = [np.mean(sampleB.sample(sampleSize)) for i in range(itrerationSize)]

    # For sampleA's means
    meanA = np.mean(sampleA_n)
    sigmaA = np.std(sampleA_n)
    semA = stats.sem(sampleA_n)

    lowLimitA = norm.ppf((1-ci)/2) * sigmaA + meanA
    upLimitA = norm.ppf(ci+(1-ci)/2) * sigmaA + meanA

    # For sampleB's means
    meanB = np.mean(sampleB_n)
    sigmaB = np.std(sampleB_n)
    semB = stats.sem(sampleB_n)

    lowLimitB = norm.ppf((1-ci)/2) * sigmaB + meanB
    upLimitB = norm.ppf(ci + (1-ci)/2) * sigmaB + meanB

    sns.kdeplot(data = sampleA_n, color="blue", fill = True, linewidth = 2)
    label_meanA=("μ (Males) :  {:.2f}".format(meanA))
    plt.axvline(meanA, color = 'blue', linestyle = 'solid', linewidth = 2, label=label_meanA)
    label_limitsA=("Lower Limit(M):  {:.2f}\nUpper Limit(M):   {:.2f}".format(lowLimitA,upLimitA))
    plt.axvline(lowLimitA, color = 'blue', linestyle = 'dashdot', linewidth = 2, label=label_limitsA)
    plt.axvline(upLimitA, color = 'blue', linestyle = 'dashdot', linewidth = 2)

    sns.kdeplot(data = sampleB_n,color='pink', fill = True, linewidth = 2)
    label_meanB=("μ (Females):  {:.2f}".format(meanB))
    plt.axvline(meanB, color = 'pink', linestyle = 'solid', linewidth = 2, label=label_meanB)
    label_limitsB=("Lower Limit(F):  {:.2f}\nUpper Limit(F):   {:.2f}".format(lowLimitB,upLimitB))
    plt.axvline(lowLimitB, color = 'pink', linestyle = 'dashdot', linewidth = 2, label=label_limitsB)
    plt.axvline(upLimitB, color = 'pink', linestyle = 'dashdot', linewidth = 2)

    plt.title(f"Sample Size: {sampleSize}, Male Avg: {np.round(meanA, 2)}, Male SME: {np.round(semA,2)}, Female Avg:{r
    plt.legend(loc = 'upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')
```

```
    return round(meanA,2), round(meanB,2), round(lowLimitA,2), round(upLimitA,2), round(lowLimitB,2), round(upLimitB,2
```

In [32]:
```
df_male = df[df['Gender']=='M']
df_female = df[df['Gender']=='F']
```

In [33]:
```
plt.figure(figsize=(12,8))
sns.histplot(x= 'Purchase',data=df,hue='Gender')
plt.show()
```



We can observe that Male spend more than Female.

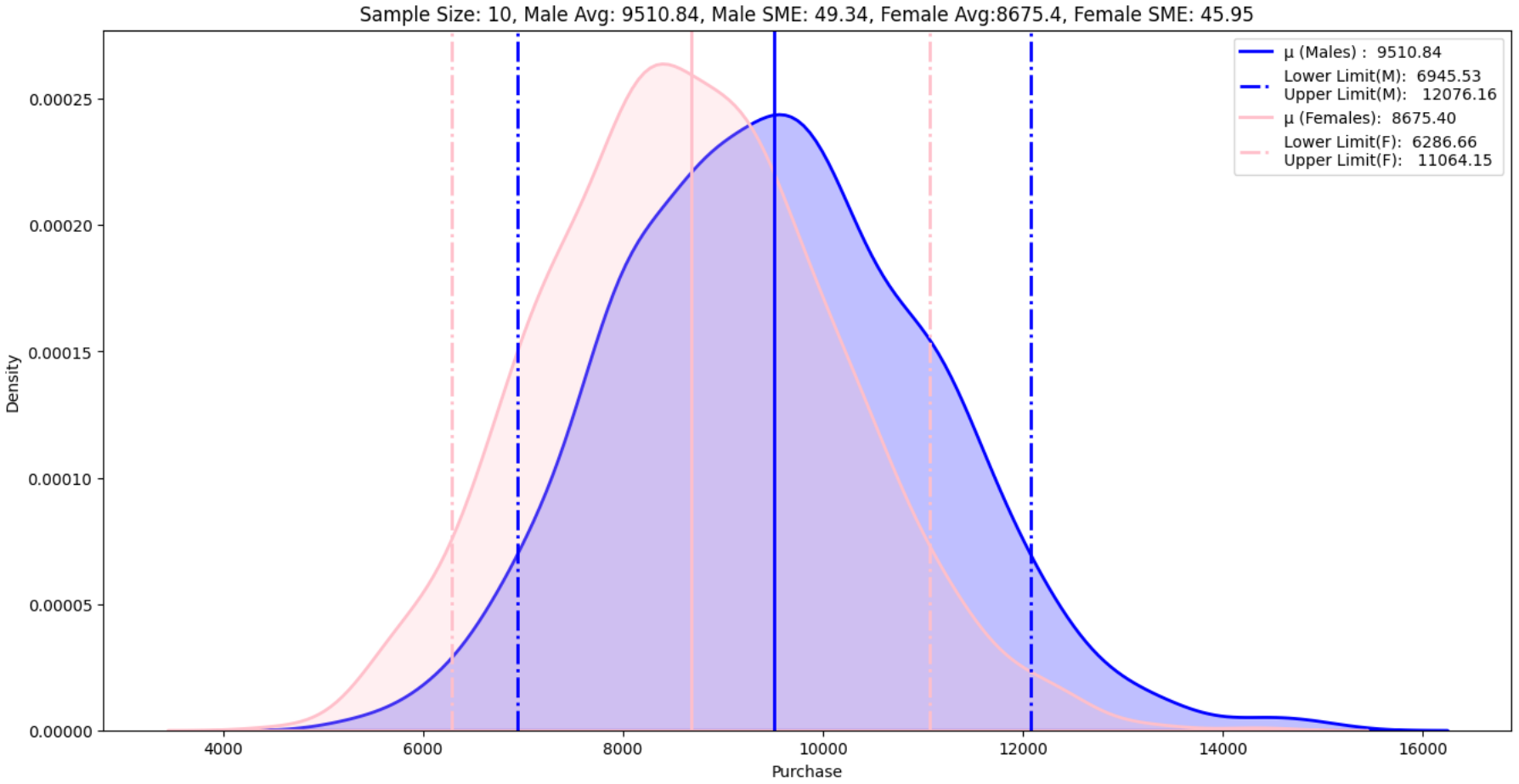
In [34]:
```
df.groupby(['Gender'])['Purchase'].describe()
```

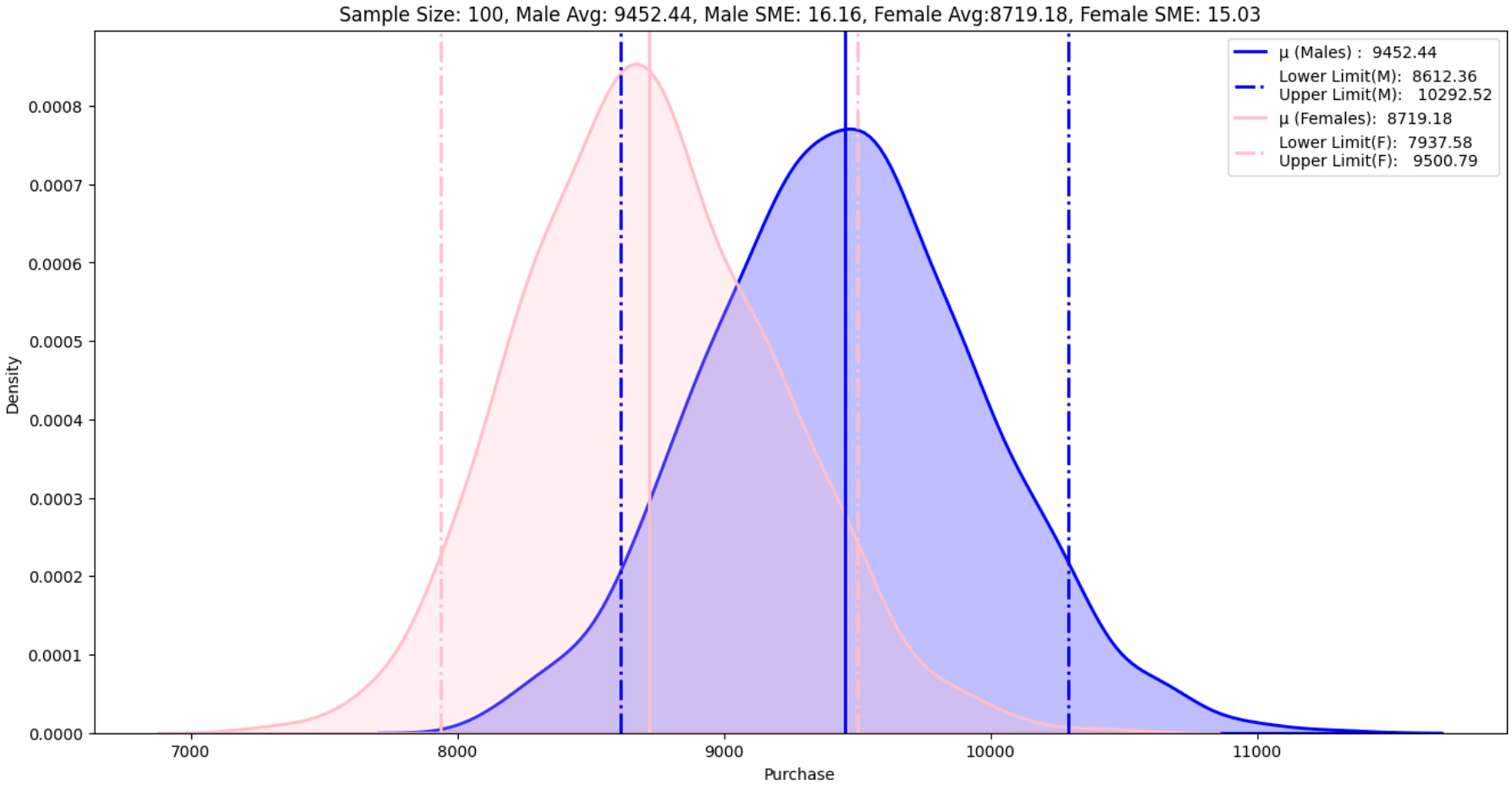
Out[34]:

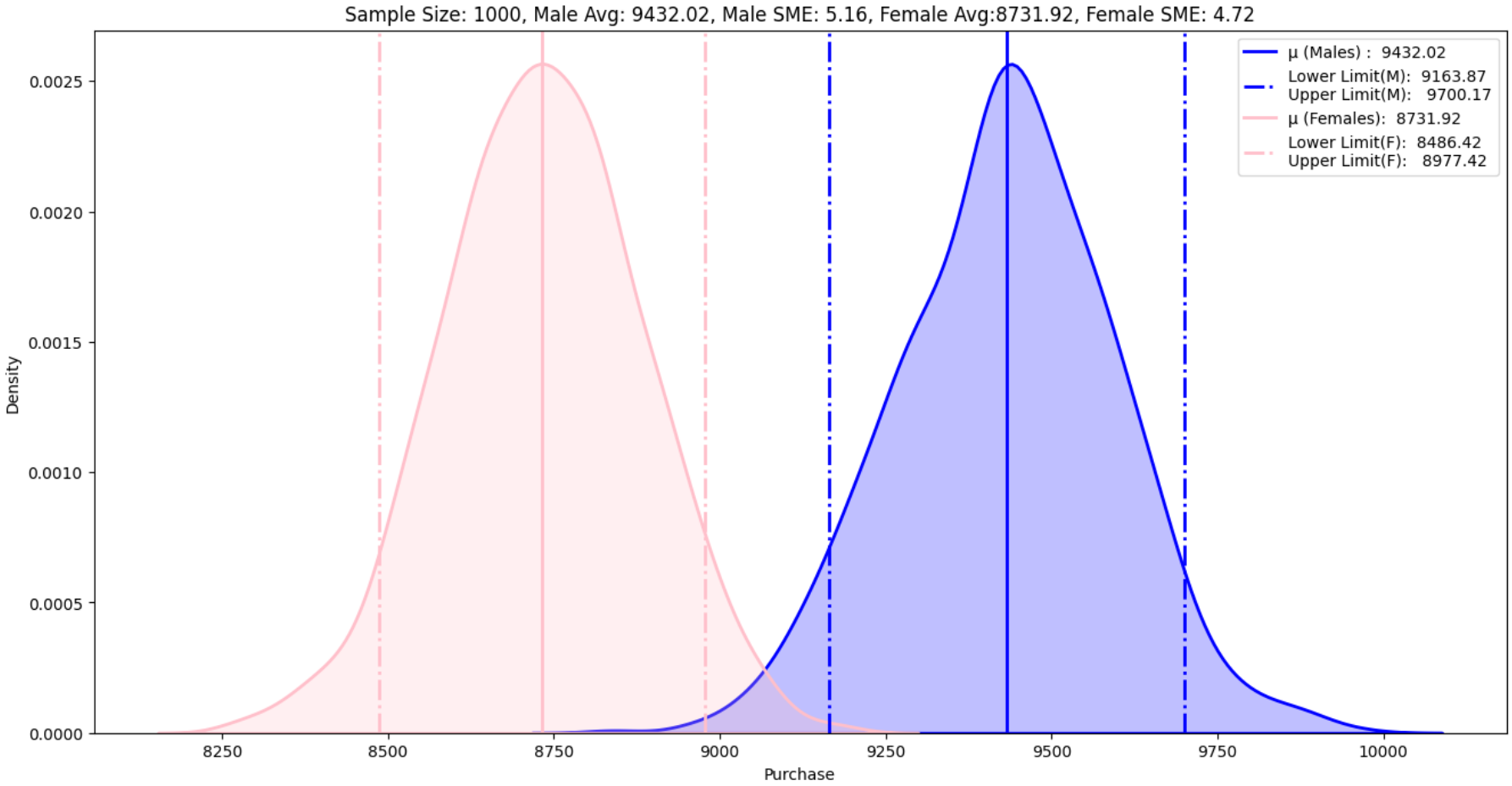
Gender	count	mean	std	min	25%	50%	75%	max
F	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
M	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

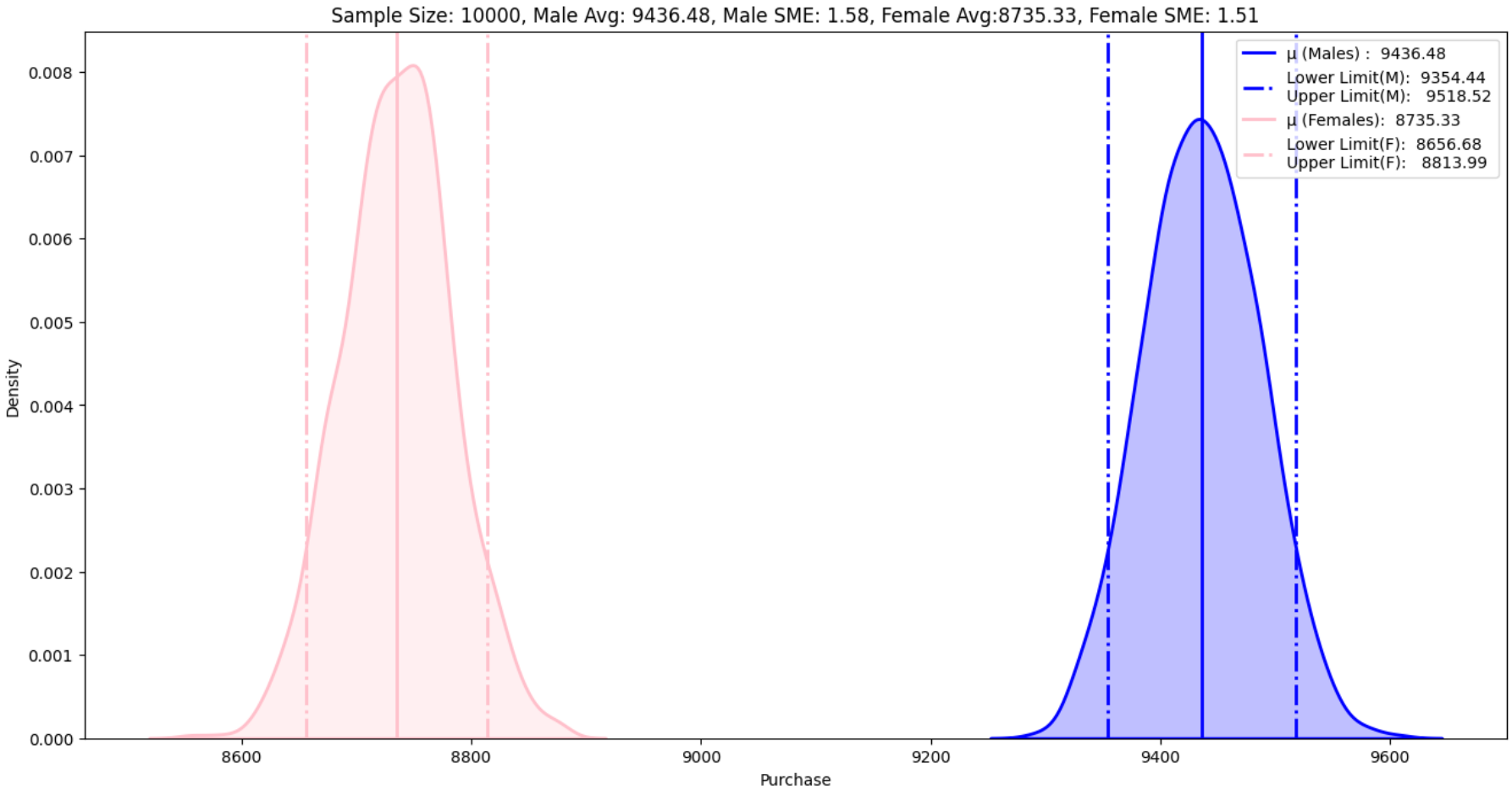
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 90% Confidence Interval

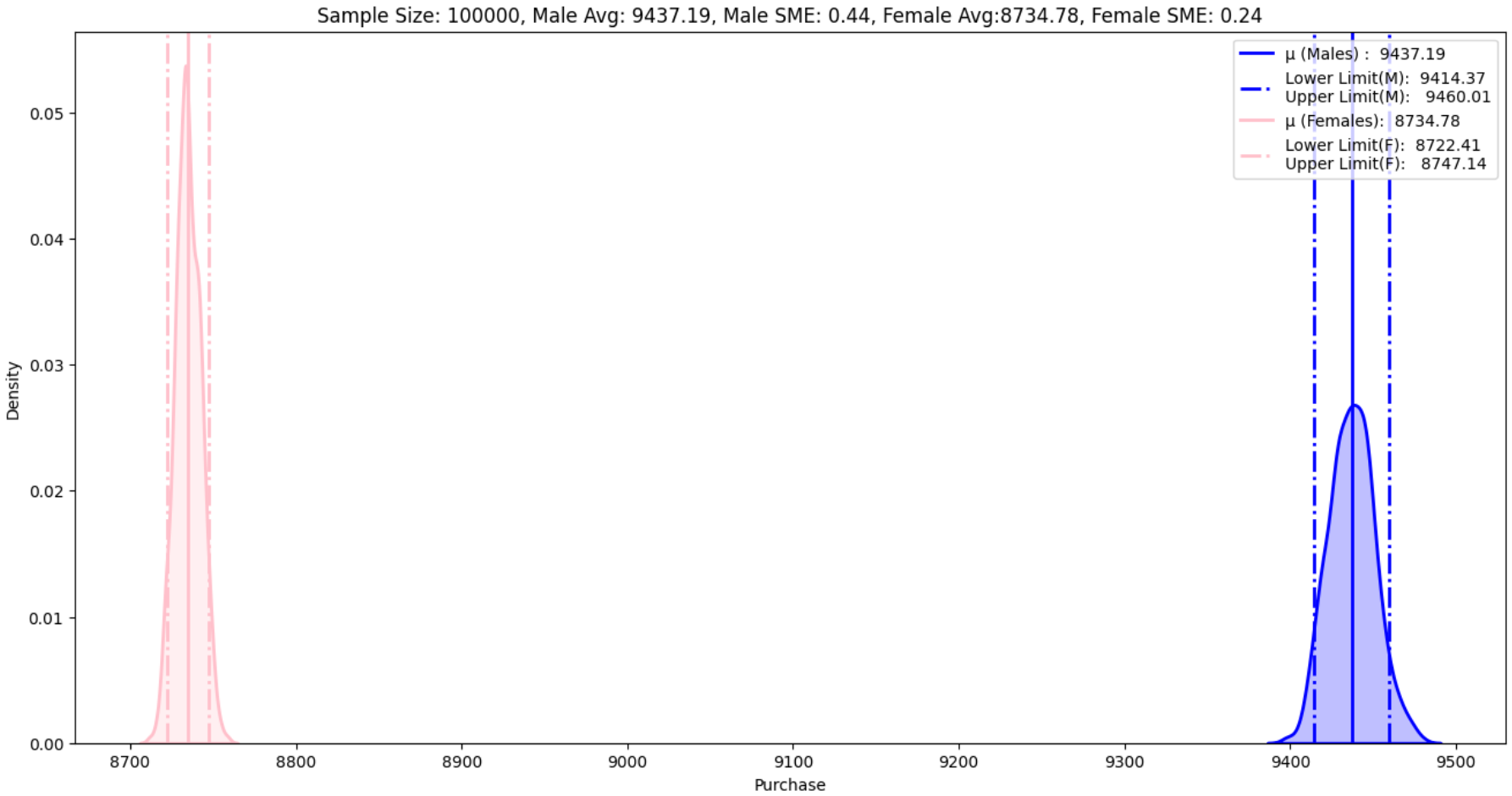
In [35]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 90
itr_size = 1000

res = pd.DataFrame(columns = ['Gender','Sample Size','Lower Limit','Upper Limit','Sample Mean','Confidence Interval','

for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f = getCLT(df_male['Purchase'],df_female['Purchase'],i,itr_size,ci)

    res = res.append({'Gender':'M','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m_avg,'Confider
    res = res.append({'Gender':'F','Sample Size':i,'Lower Limit':ll_f,'Upper Limit':ul_f,'Sample Mean':f_avg,'Confider
```




Sample Size: 100, Male Avg: 9452.44, Male SME: 16.16, Female Avg:8719.18, Female SME: 15.03
μ (Males) : 9452.44
Lower Limit(M): 8612.36
Upper Limit(M): 10292.52
μ (Females): 8719.18
Lower Limit(F): 7937.58
Upper Limit(F): 9500.79
Density
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0.0007
0.0008
7000
8000
9000
10000
11000
Purchase


Sample Size: 1000, Male Avg: 9432.02, Male SME: 5.16, Female Avg:8731.92, Female SME: 4.72
μ (Males) : 9432.02
Lower Limit(M): 9163.87
Upper Limit(M): 9700.17
μ (Females): 8731.92
Lower Limit(F): 8486.42
Upper Limit(F): 8977.42
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: 9436.48, Male SME: 1.58, Female Avg:8735.33, Female SME: 1.51
μ (Males) : 9436.48
Lower Limit(M): 9354.44
Upper Limit(M): 9518.52
μ (Females): 8735.33
Lower Limit(F): 8656.68
Upper Limit(F): 8813.99
Density
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
8600
8800
9000
9200
9400
9600
Purchase



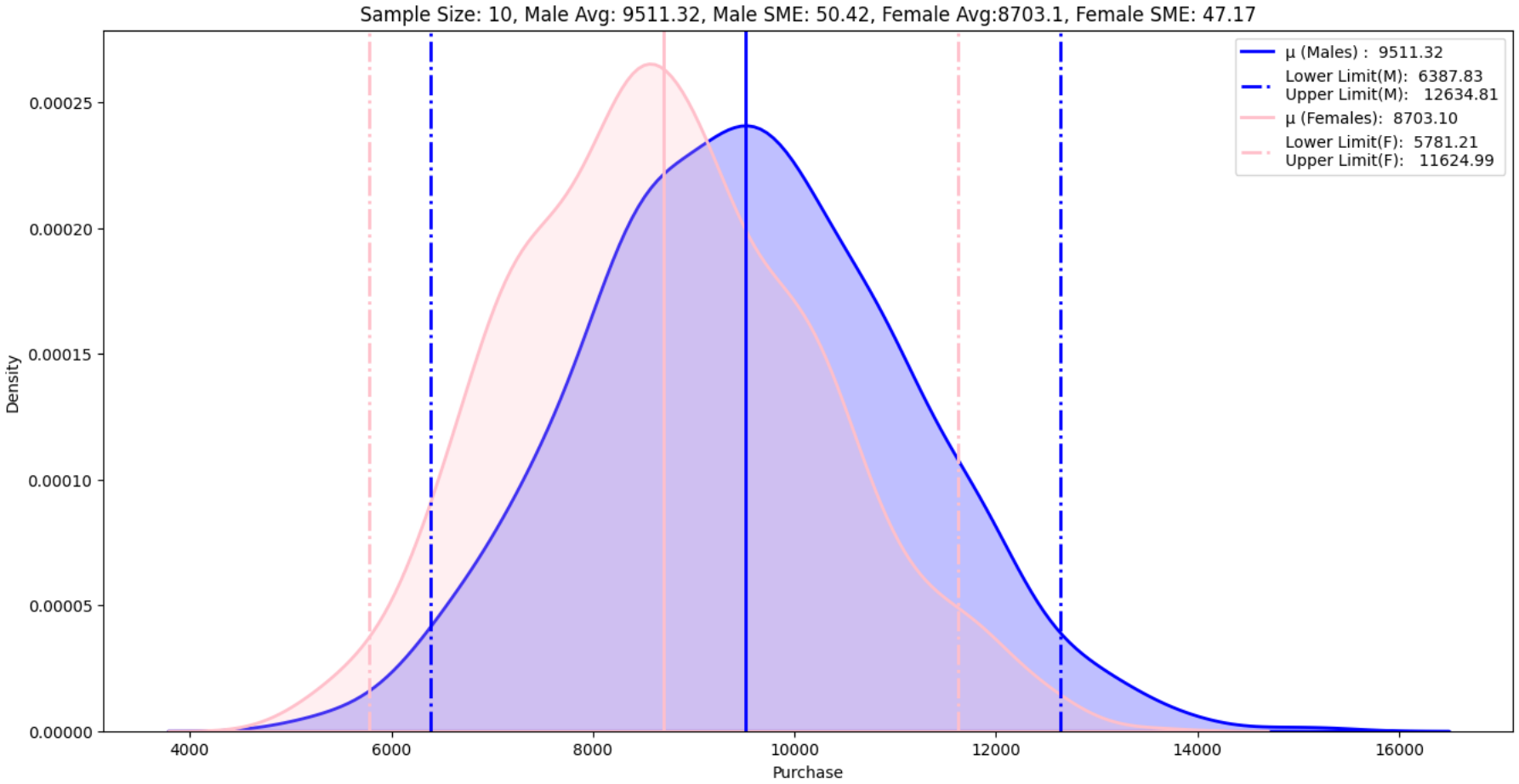
We can observe that as the sample size increases,

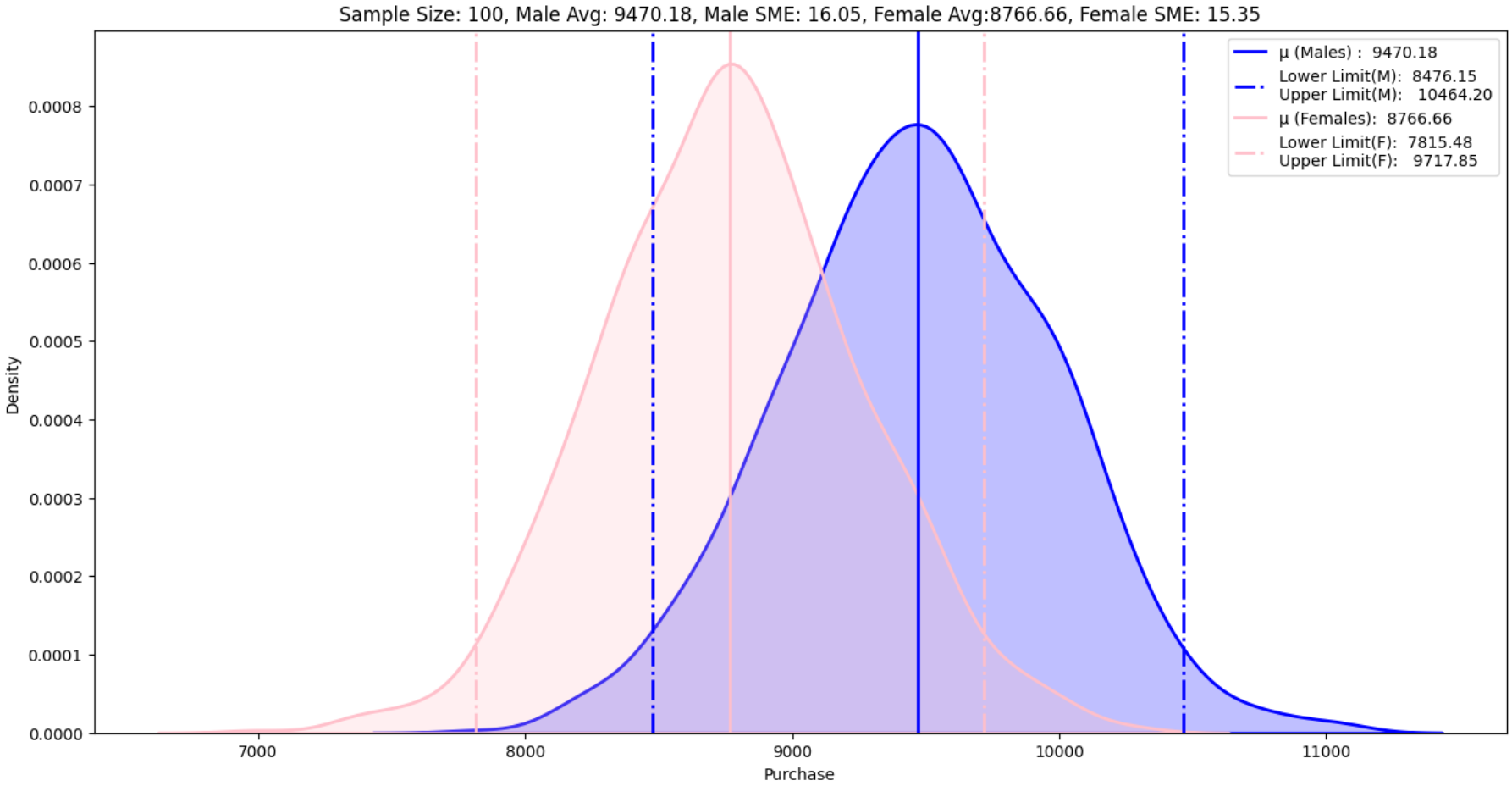
`The average for both of them change significantly.`

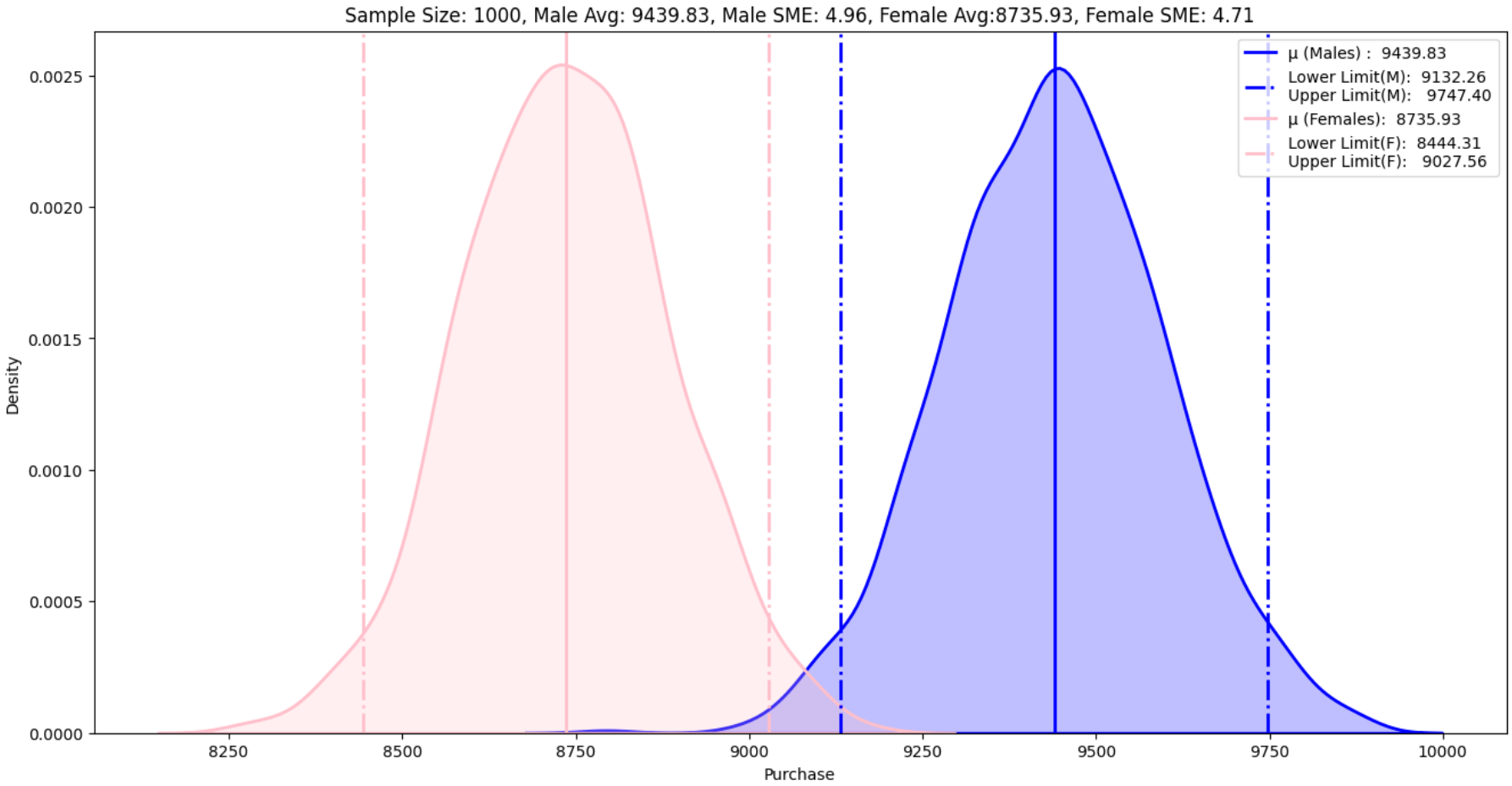
`Both the plots start to seperate and become distinct.`

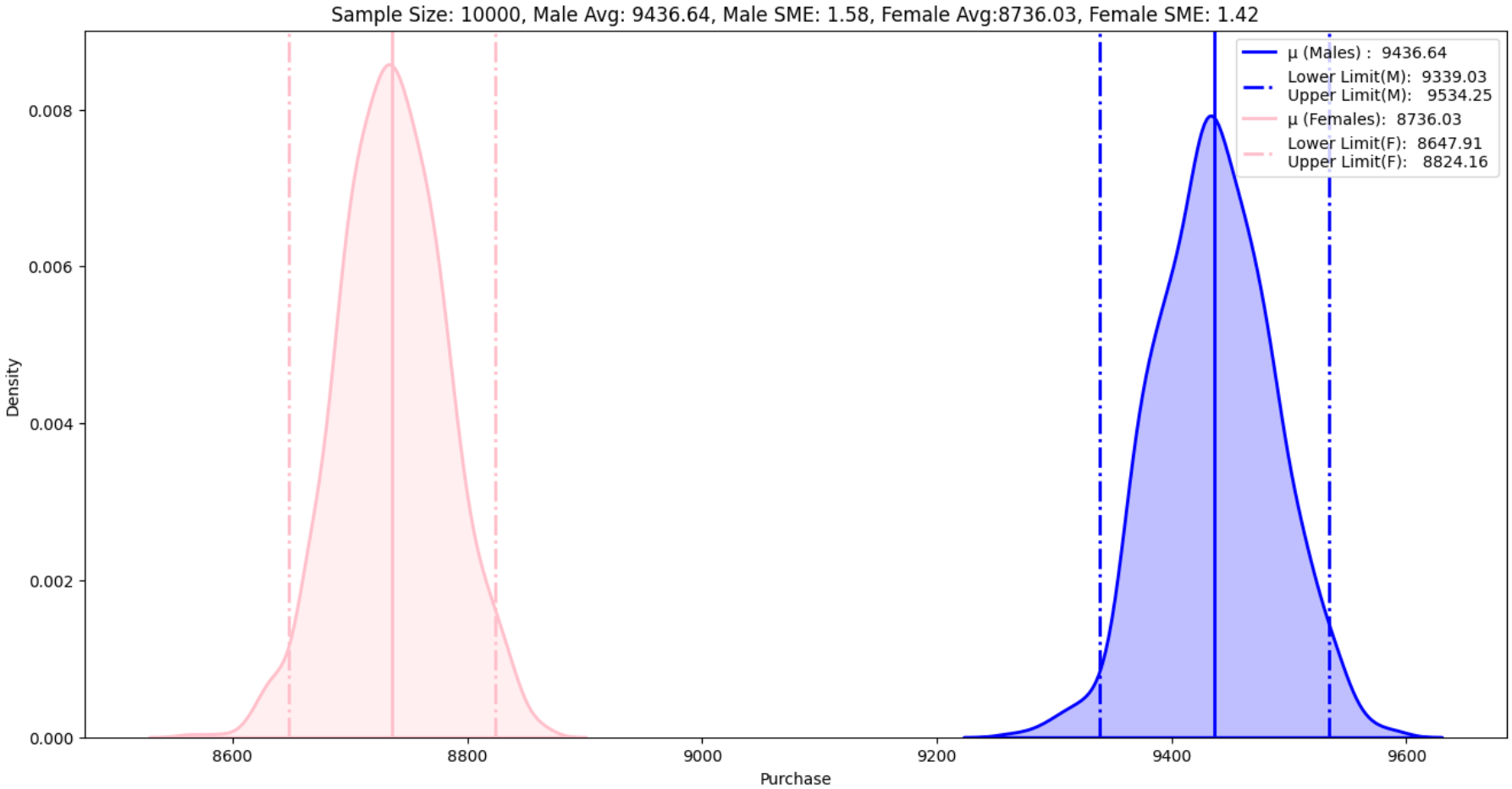
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 95% Confidence Interval

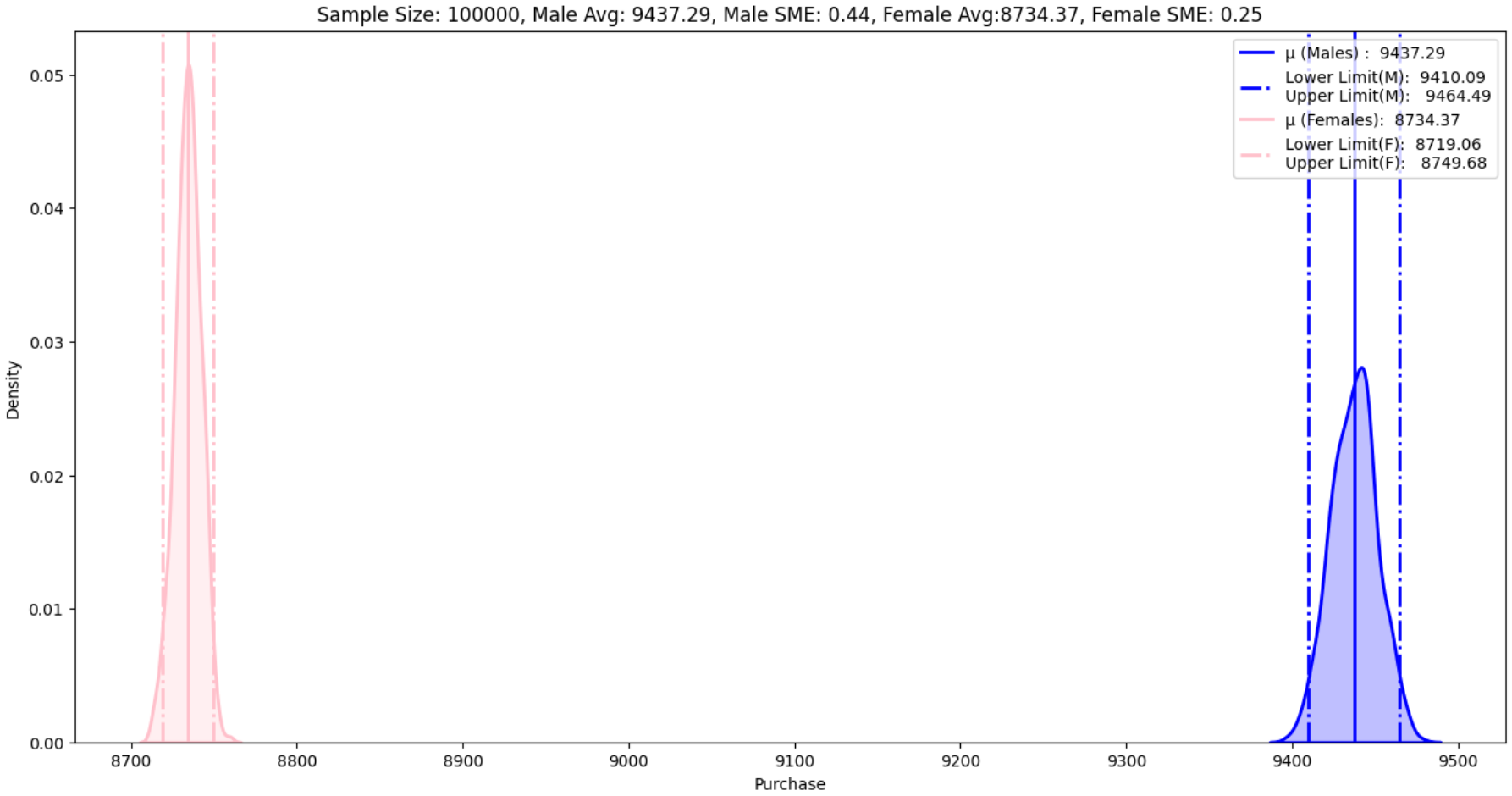
In [36]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 95
itr_size = 1000

for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f = getCLT(df_male['Purchase'],df_female['Purchase'],i,itr_size,ci)

    res = res.append({'Gender':'M','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m_avg,'Confider
    res = res.append({'Gender':'F','Sample Size':i,'Lower Limit':ll_f,'Upper Limit':ul_f,'Sample Mean':f_avg,'Confider
```




Sample Size: 100, Male Avg: 9470.18, Male SME: 16.05, Female Avg:8766.66, Female SME: 15.35
μ (Males) : 9470.18
Lower Limit(M): 8476.15
Upper Limit(M): 10464.20
μ (Females): 8766.66
Lower Limit(F): 7815.48
Upper Limit(F): 9717.85
Density
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0.0007
0.0008
7000
8000
9000
10000
11000
Purchase


Sample Size: 1000, Male Avg: 9439.83, Male SME: 4.96, Female Avg:8735.93, Female SME: 4.71
μ (Males) : 9439.83
Lower Limit(M): 9132.26
Upper Limit(M): 9747.40
μ (Females): 8735.93
Lower Limit(F): 8444.31
Upper Limit(F): 9027.56
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: 9436.64, Male SME: 1.58, Female Avg:8736.03, Female SME: 1.42
μ (Males) : 9436.64
Lower Limit(M): 9339.03
Upper Limit(M): 9534.25
μ (Females): 8736.03
Lower Limit(F): 8647.91
Upper Limit(F): 8824.16
Density
0.000
0.002
0.004
0.006
0.008
8600
8800
9000
9200
9400
9600
Purchase


Sample Size: 100000, Male Avg: 9437.29, Male SME: 0.44, Female Avg:8734.37, Female SME: 0.25
μ (Males) : 9437.29
Lower Limit(M): 9410.09
Upper Limit(M): 9464.49
μ (Females): 8734.37
Lower Limit(F): 8719.06
Upper Limit(F): 8749.68
Density
0.00
0.01
0.02
0.03
0.04
0.05
8700
8800
8900
9000
9100
9200
9300
9400
9500
Purchase

In [37]:
```
#Cheking the values
res
```

Out[37]:

	Gender	Sample Size	Lower Limit	Upper Limit	Sample Mean	Confidence Interval	Interval Range	Range
0	M	10	6945.53	12076.16	9510.84	90	[6945.53, 12076.16]	5130.63
1	F	10	6286.66	11064.15	8675.40	90	[6286.66, 11064.15]	4777.49
2	M	100	8612.36	10292.52	9452.44	90	[8612.36, 10292.52]	1680.16
3	F	100	7937.58	9500.79	8719.18	90	[7937.58, 9500.79]	1563.21
4	M	1000	9163.87	9700.17	9432.02	90	[9163.87, 9700.17]	536.30
5	F	1000	8486.42	8977.42	8731.92	90	[8486.42, 8977.42]	491.00
6	M	10000	9354.44	9518.52	9436.48	90	[9354.44, 9518.52]	164.08
7	F	10000	8656.68	8813.99	8735.33	90	[8656.68, 8813.99]	157.31
8	M	100000	9414.37	9460.01	9437.19	90	[9414.37, 9460.01]	45.64
9	F	100000	8722.41	8747.14	8734.78	90	[8722.41, 8747.14]	24.73
10	M	10	6387.83	12634.81	9511.32	95	[6387.83, 12634.81]	6246.98
11	F	10	5781.21	11624.99	8703.10	95	[5781.21, 11624.99]	5843.78
12	M	100	8476.15	10464.20	9470.18	95	[8476.15, 10464.2]	1988.05
13	F	100	7815.48	9717.85	8766.66	95	[7815.48, 9717.85]	1902.37
14	M	1000	9132.26	9747.40	9439.83	95	[9132.26, 9747.4]	615.14
15	F	1000	8444.31	9027.56	8735.93	95	[8444.31, 9027.56]	583.25
16	M	10000	9339.03	9534.25	9436.64	95	[9339.03, 9534.25]	195.22
17	F	10000	8647.91	8824.16	8736.03	95	[8647.91, 8824.16]	176.25
18	M	100000	9410.09	9464.49	9437.29	95	[9410.09, 9464.49]	54.40
19	F	100000	8719.06	8749.68	8734.37	95	[8719.06, 8749.68]	30.62

We can observe that for 90% CI:

For Sample size 10 The confidence interval for both Male and Female is overlapping

And as the sample size increases, we can see the interval ranges seperating and then finally they both dont overalap.

For Sample size 100000 The confidence interval for both Male and Female is now not overlapping.

We can also observe the same with 95% CI:

For Sample size 10 The confidence interval for both Male and Female is overlapping

And as the sample size increases, we can see the interval ranges seperating and then finally they both dont overalap.

For Sample size 100000 The confidence interval for both Male and Female is now not overlapping.

Married vs Unmarried Purchases:

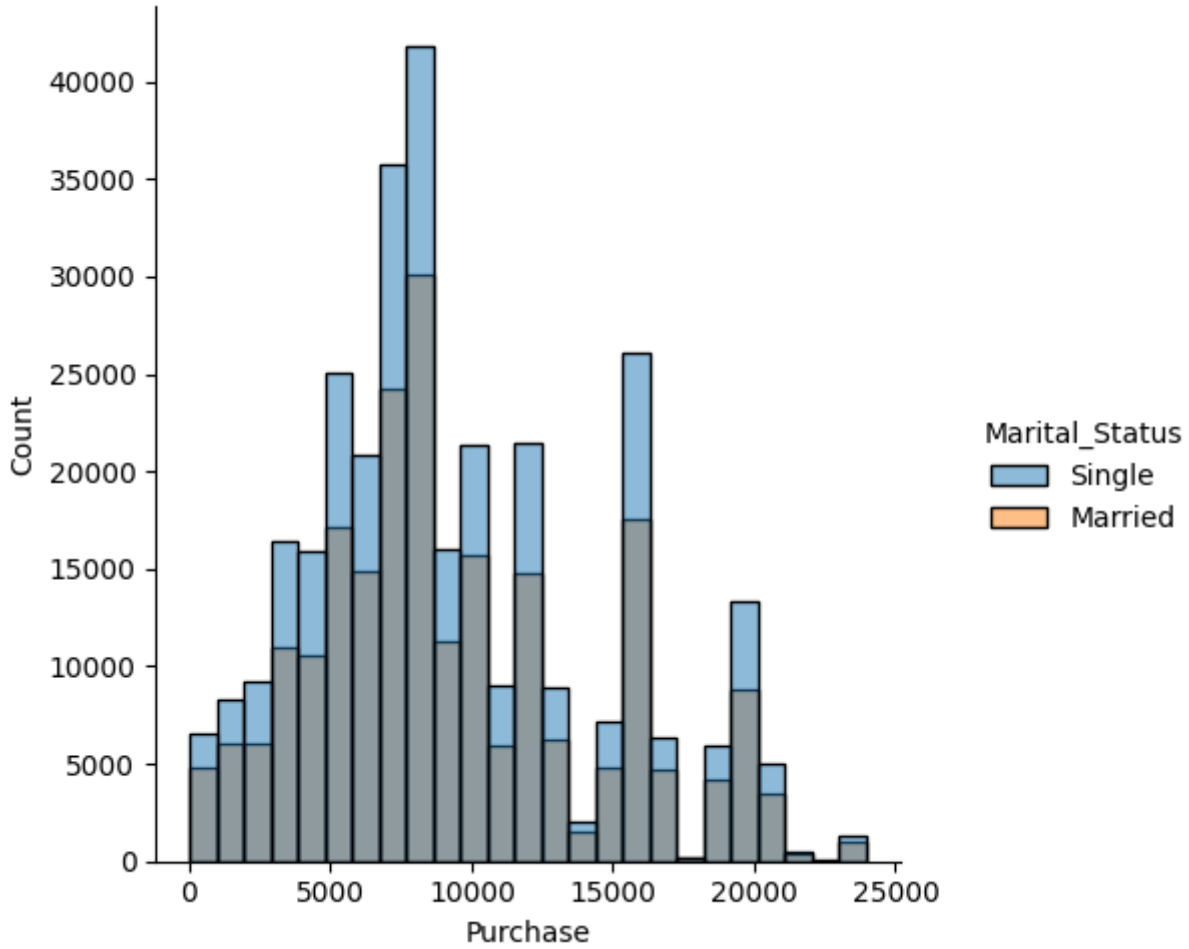
In [38]:
```
def getCLTMarital(sampleA,sampleB,sampleSize,itrerationSize=1000,ci=90):
    ci = ci/100

    plt.figure(figsize=(16,8))
    sampleA_n = [np.mean(sampleA.sample(sampleSize)) for i in range(itrerationSize)]
    sampleB_n = [np.mean(sampleB.sample(sampleSize)) for i in range(itrerationSize)]

    # For sampleA's means
    meanA = np.mean(sampleA_n)
    sigmaA = np.std(sampleA_n)
    semA = stats.sem(sampleA_n)

    lowLimitA = norm.ppf((1-ci)/2) * sigmaA + meanA
    upLimitA = norm.ppf(ci+(1-ci)/2) * sigmaA + meanA

    # For sampleB's means
    meanB = np.mean(sampleB_n)
    sigmaB = np.std(sampleB_n)
    semB = stats.sem(sampleB_n)

    lowLimitB = norm.ppf((1-ci)/2) * sigmaB + meanB
    upLimitB = norm.ppf(ci + (1-ci)/2) * sigmaB + meanB

    sns.kdeplot(data = sampleA_n, color="green", fill = True, linewidth = 2)
    label_meanA=("μ (Marrieds) :  {:.2f}".format(meanA))
    plt.axvline(meanA, color = 'green', linestyle = 'solid', linewidth = 2, label=label_meanA)
    label_limitsA=("Lower Limit(M):  {:.2f}\nUpper Limit(M):   {:.2f}".format(lowLimitA,upLimitA))
    plt.axvline(lowLimitA, color = 'green', linestyle = 'dashdot', linewidth = 2, label=label_limitsA)
    plt.axvline(upLimitA, color = 'green', linestyle = 'dashdot', linewidth = 2)

    sns.kdeplot(data = sampleB_n,color='orange', fill = True, linewidth = 2)
    label_meanB=("μ (Unmarrieds):  {:.2f}".format(meanB))
    plt.axvline(meanB, color = 'orange', linestyle = 'solid', linewidth = 2, label=label_meanB)
    label_limitsB=("Lower Limit(F):  {:.2f}\nUpper Limit(F):   {:.2f}".format(lowLimitB,upLimitB))
    plt.axvline(lowLimitB, color = 'orange', linestyle = 'dashdot', linewidth = 2, label=label_limitsB)
    plt.axvline(upLimitB, color = 'orange', linestyle = 'dashdot', linewidth = 2)

    plt.title(f"Sample Size: {sampleSize}, Married Avg: {np.round(meanA, 2)}, Married SME: {np.round(semA,2)}, Unmarri
    plt.legend(loc = 'upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')
```

```
    return round(meanA,2), round(meanB,2), round(lowLimitA,2), round(upLimitA,2), round(lowLimitB,2), round(upLimitB,2
```

In [44]:
```
df_married = df[df['Marital_Status'] == 'Married']
df_unmarried = df[df['Marital_Status'] == 'Single']
```

In [45]:
```
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 count of orders of unmarried customers is more than Married customers.

In [46]:
```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

Out[46]:

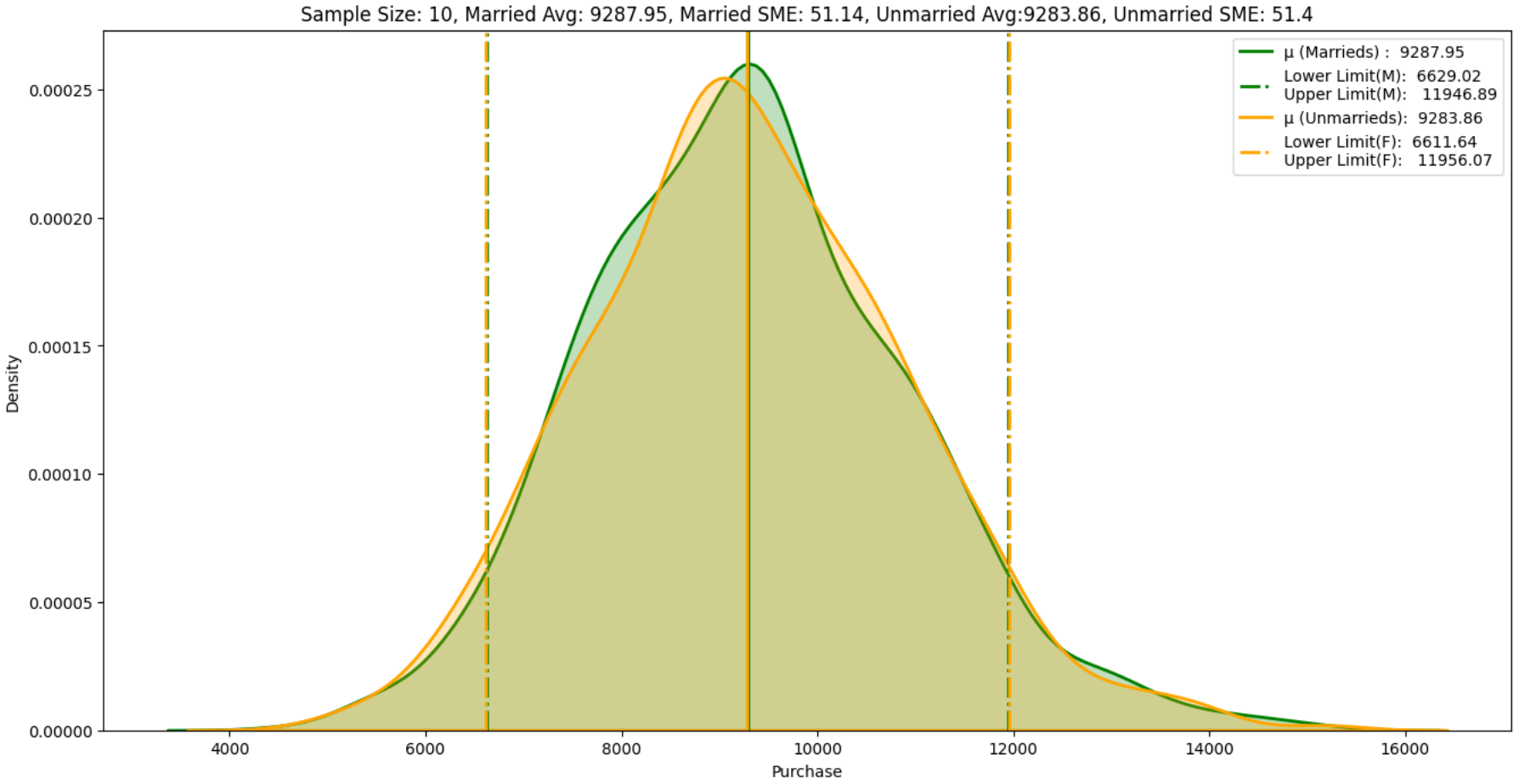
Marital_Status	count	mean	std	min	25%	50%	75%	max
Married	225337.0	9261.174574	5016.897378	12.0	5843.0	8051.0	12042.0	23961.0
Single	324731.0	9265.907619	5027.347859	12.0	5605.0	8044.0	12061.0	23961.0

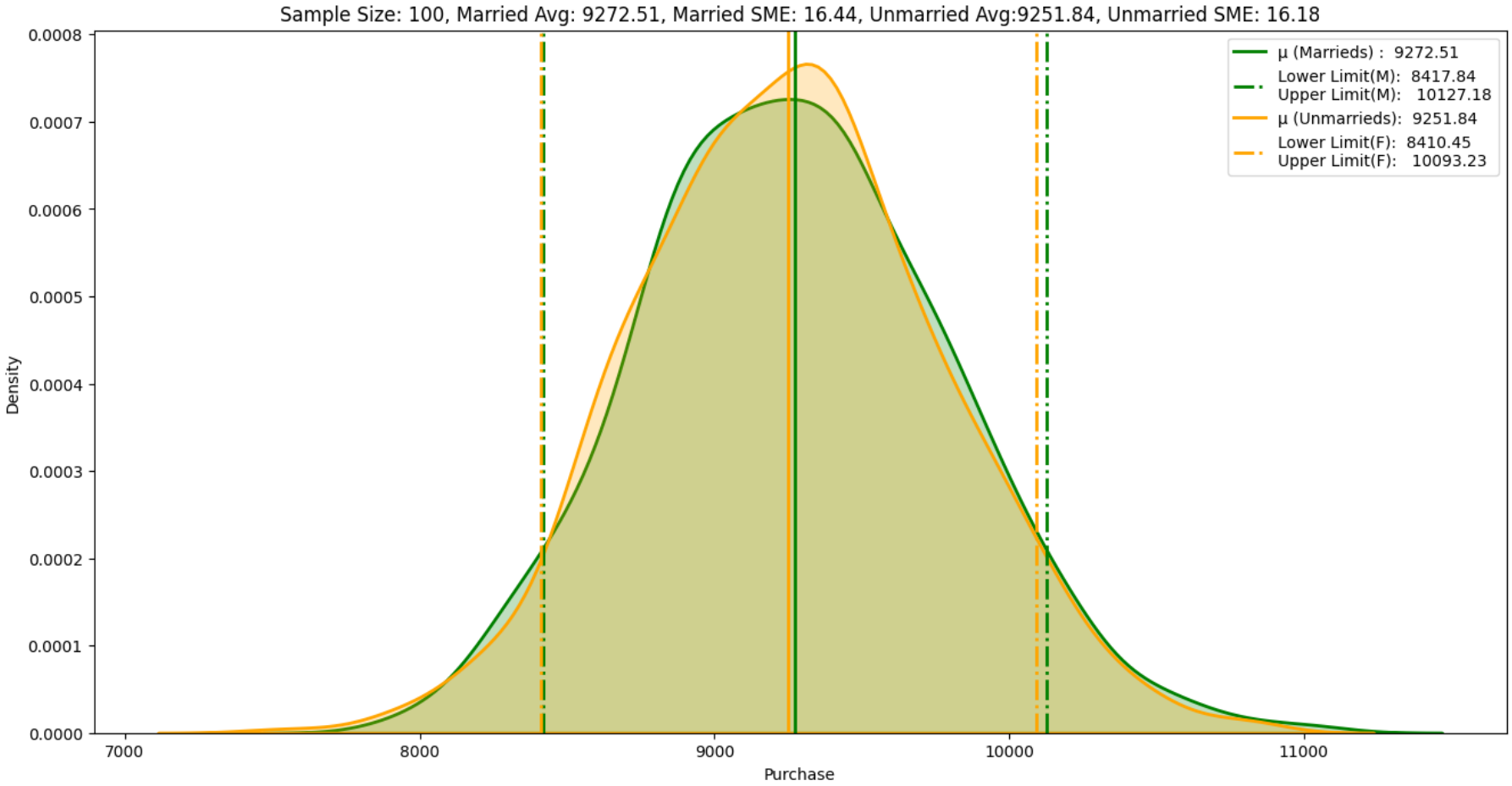
There is no difference in the mean or median values for both of them.

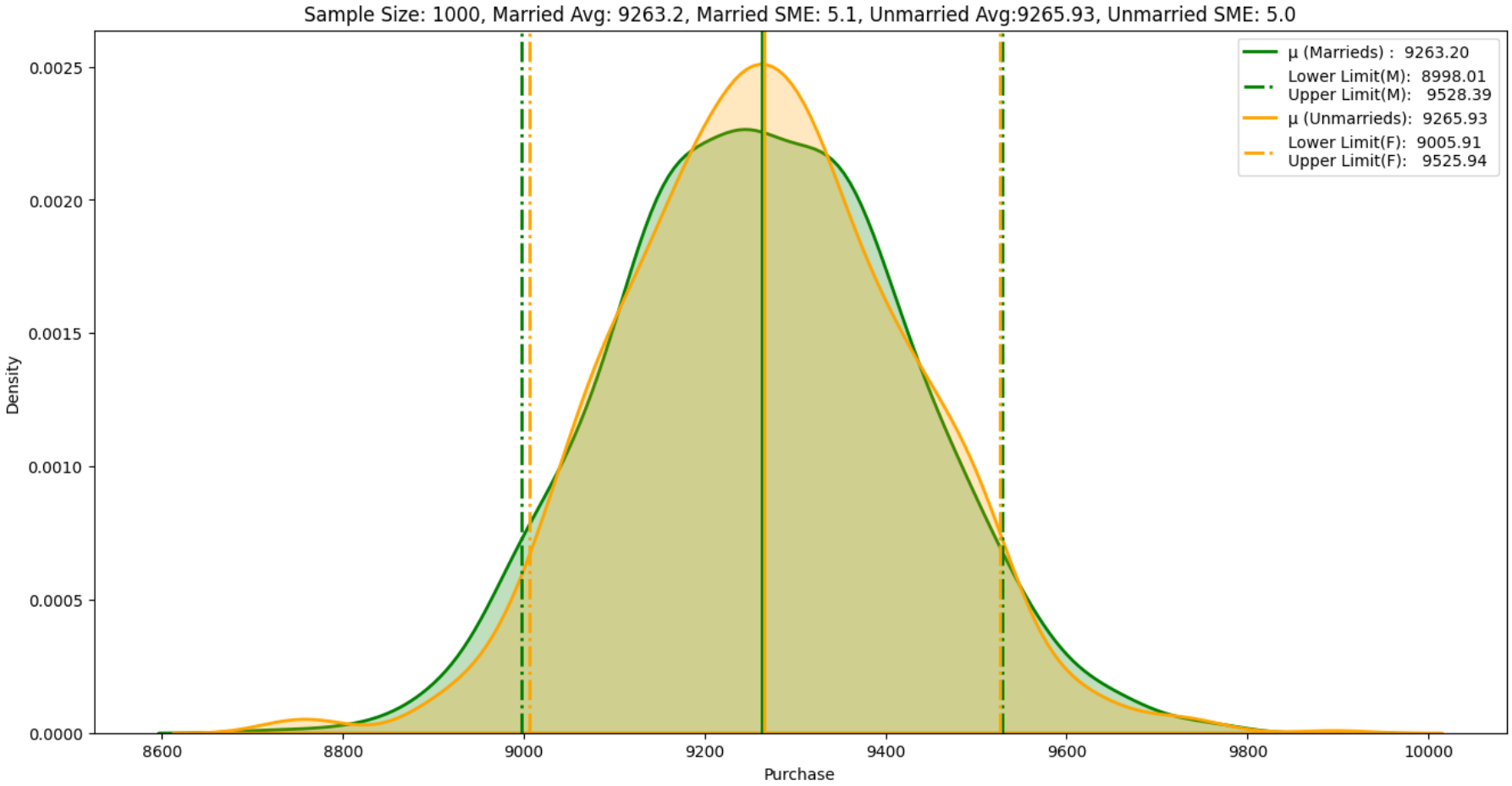
Lets dive deeper using bootstrapping and verify.

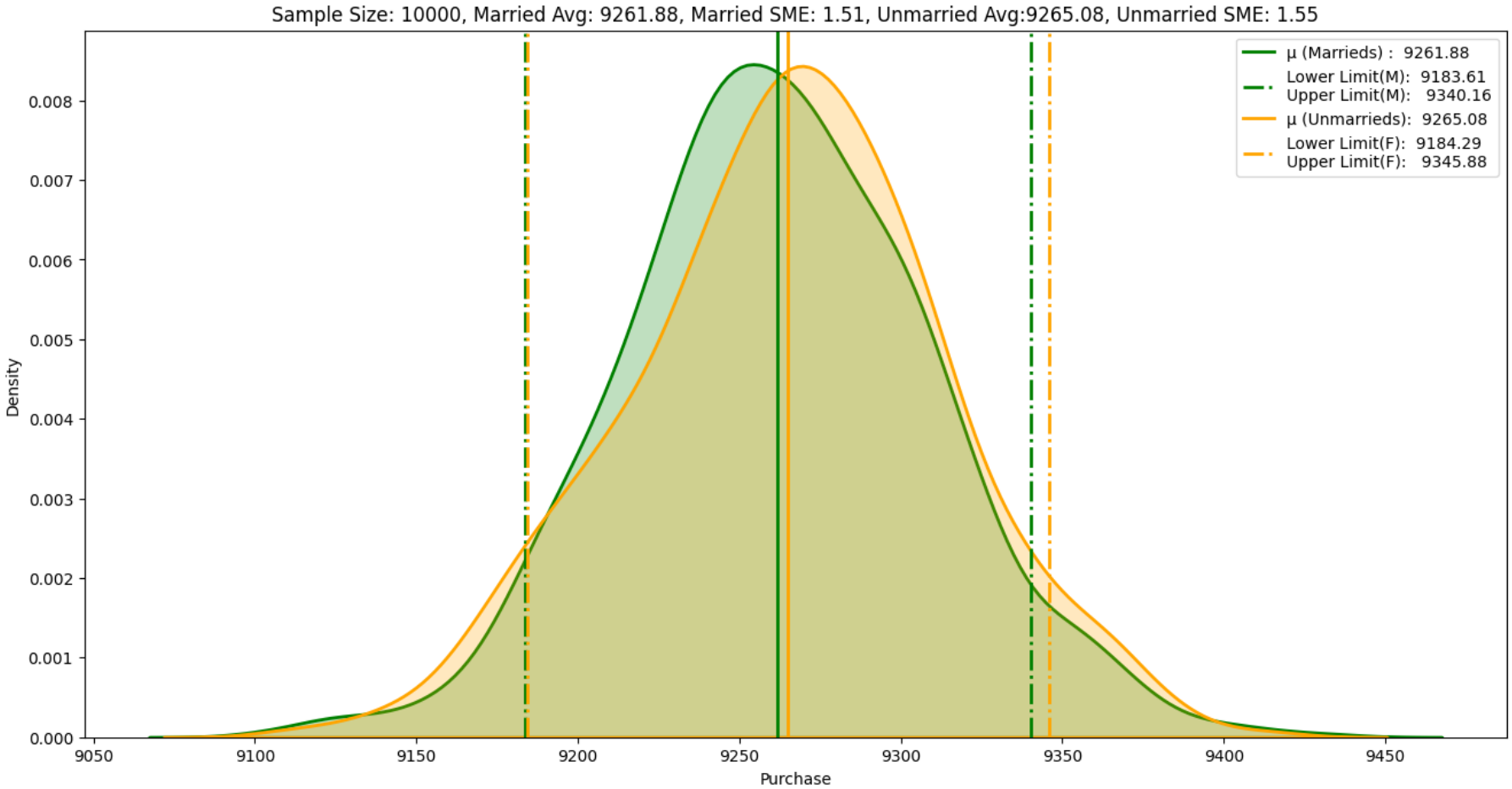
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 90% Confidence Interval

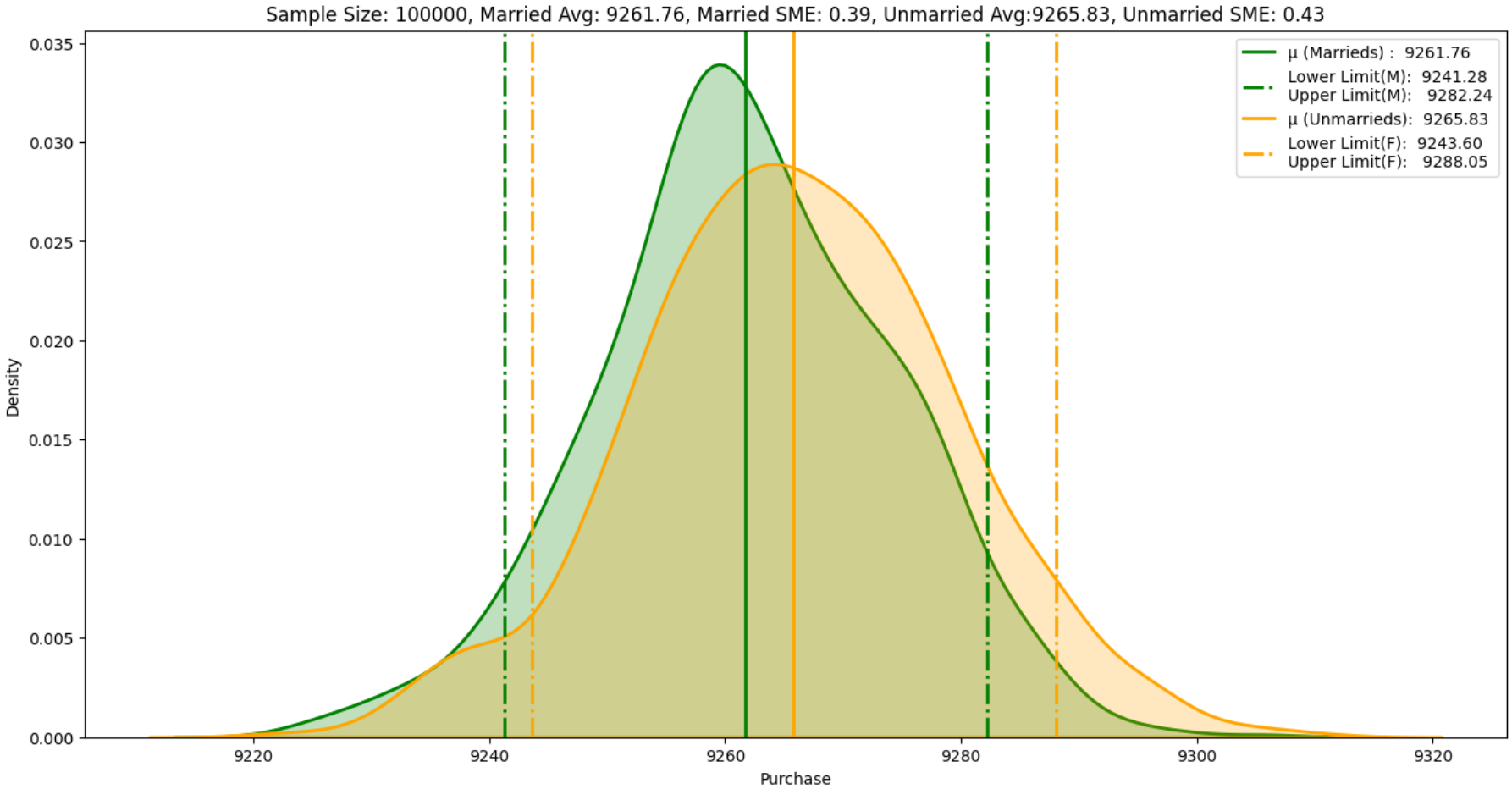
In [47]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 90
itr_size = 1000

res = pd.DataFrame(columns = ['Marital_Status','Sample Size','Lower Limit','Upper Limit','Sample Mean','Confidence Int

for i in sample_sizes:
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un = getCLTMarital(df_married['Purchase'],df_unmarried['Purchase'],i,itr_size

    res = res.append({'Marital_Status':'Married','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m
    res = res.append({'Marital_Status':'Unmarried','Sample Size':i,'Lower Limit':ll_un,'Upper Limit':ul_un,'Sample Mea
```



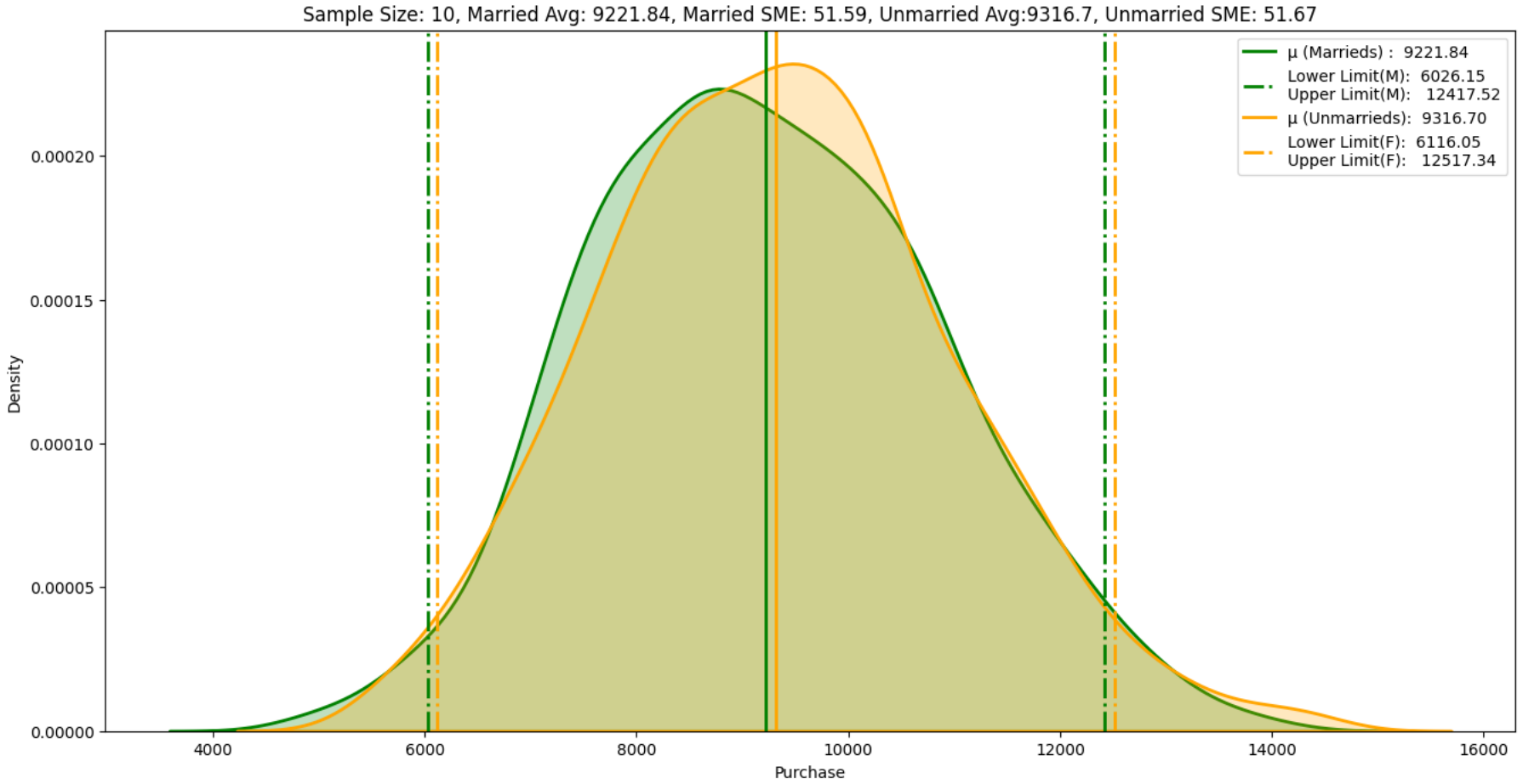

Sample Size: 100, Married Avg: 9272.51, Married SME: 16.44, Unmarried Avg:9251.84, Unmarried SME: 16.18
μ (Marrieds) : 9272.51
Lower Limit(M): 8417.84
Upper Limit(M): 10127.18
μ (Unmarrieds): 9251.84
Lower Limit(F): 8410.45
Upper Limit(F): 10093.23
Density
Purchase
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0.0007
0.0008
7000
8000
9000
10000
11000

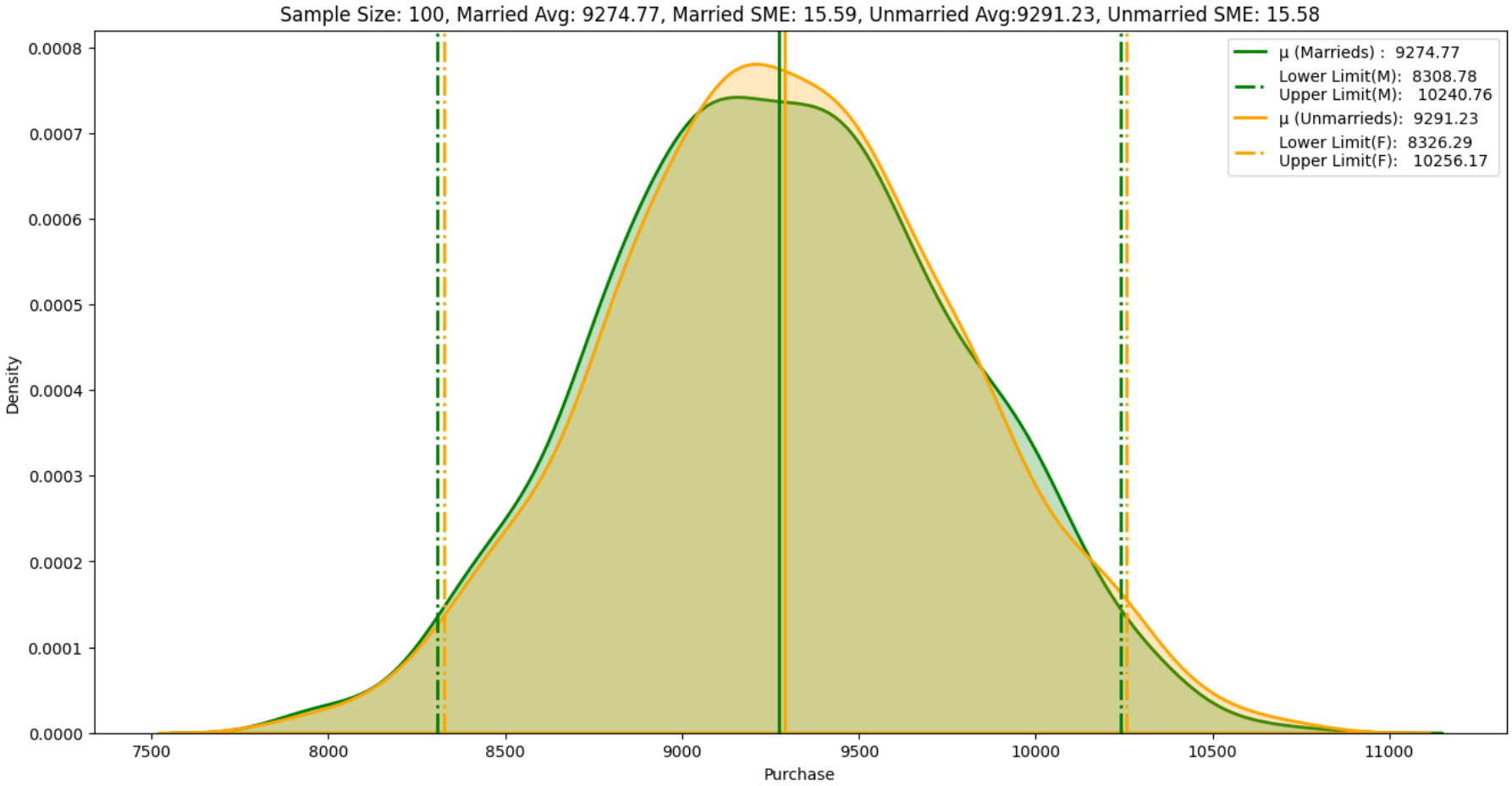

Sample Size: 1000, Married Avg: 9263.2, Married SME: 5.1, Unmarried Avg:9265.93, Unmarried SME: 5.0
μ (Marrieds) : 9263.20
Lower Limit(M): 8998.01
Upper Limit(M): 9528.39
μ (Unmarrieds): 9265.93
Lower Limit(F): 9005.91
Upper Limit(F): 9525.94
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
10000
Purchase

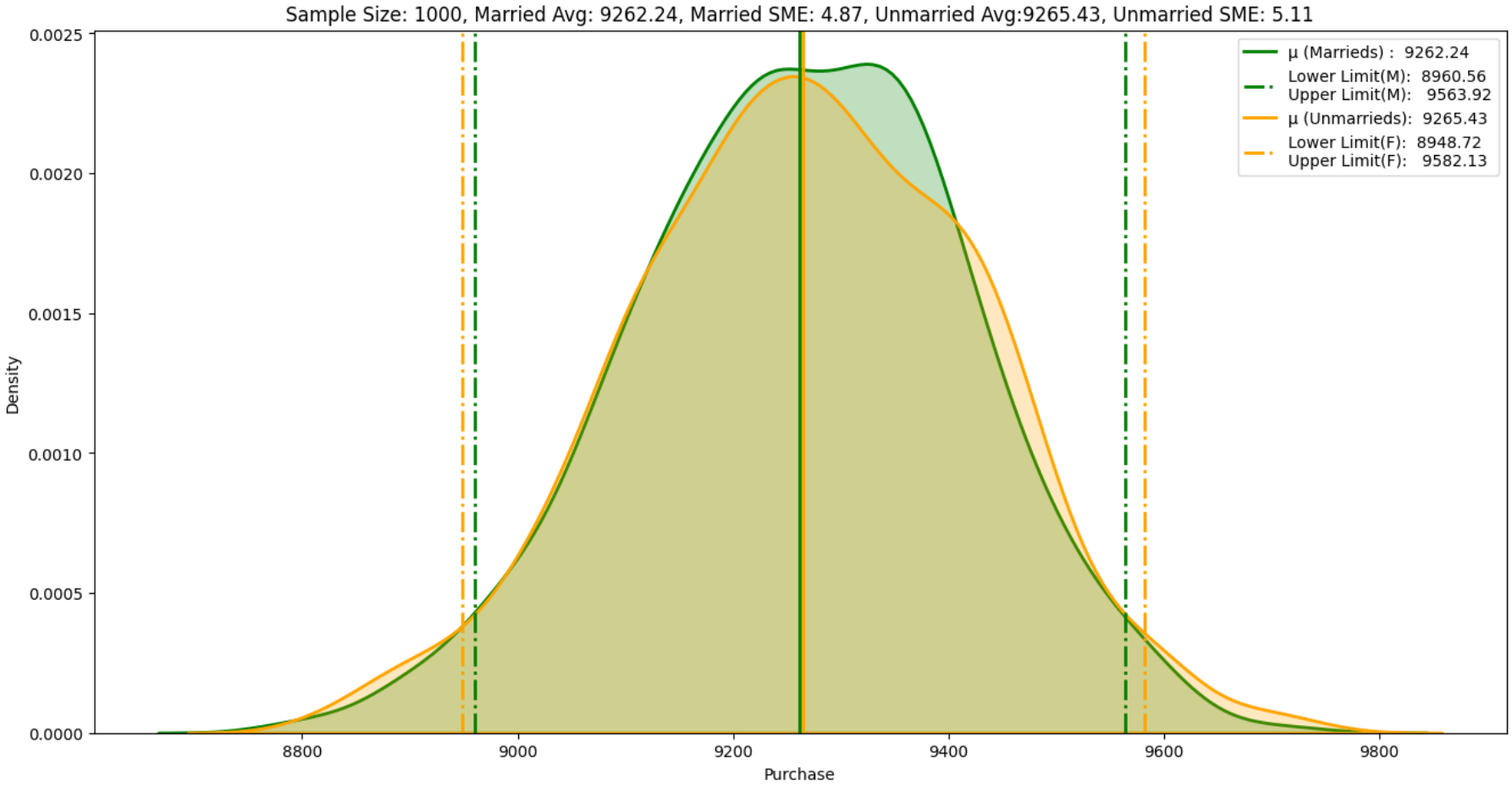

Sample Size: 10000, Married Avg: 9261.88, Married SME: 1.51, Unmarried Avg:9265.08, Unmarried SME: 1.55
μ (Marrieds) : 9261.88
Lower Limit(M): 9183.61
Upper Limit(M): 9340.16
μ (Unmarrieds): 9265.08
Lower Limit(F): 9184.29
Upper Limit(F): 9345.88
Density
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
9050
9100
9150
9200
9250
9300
9350
9400
9450
Purchase

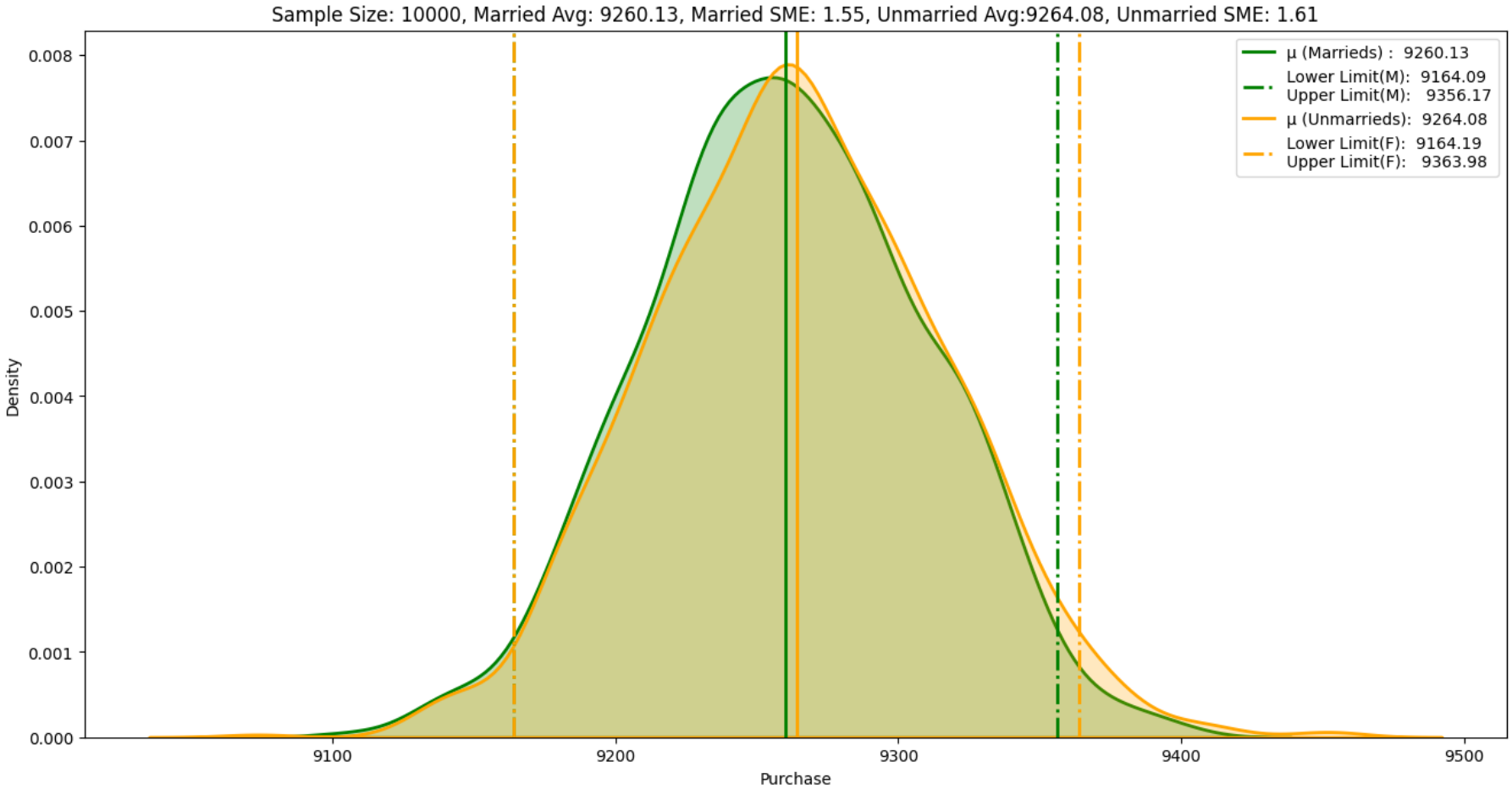

Sample Size: 100000, Married Avg: 9261.76, Married SME: 0.39, Unmarried Avg:9265.83, Unmarried SME: 0.43
μ (Marrieds) : 9261.76
Lower Limit(M): 9241.28
Upper Limit(M): 9282.24
μ (Unmarrieds): 9265.83
Lower Limit(F): 9243.60
Upper Limit(F): 9288.05
Density
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
9220
9240
9260
9280
9300
9320
Purchase

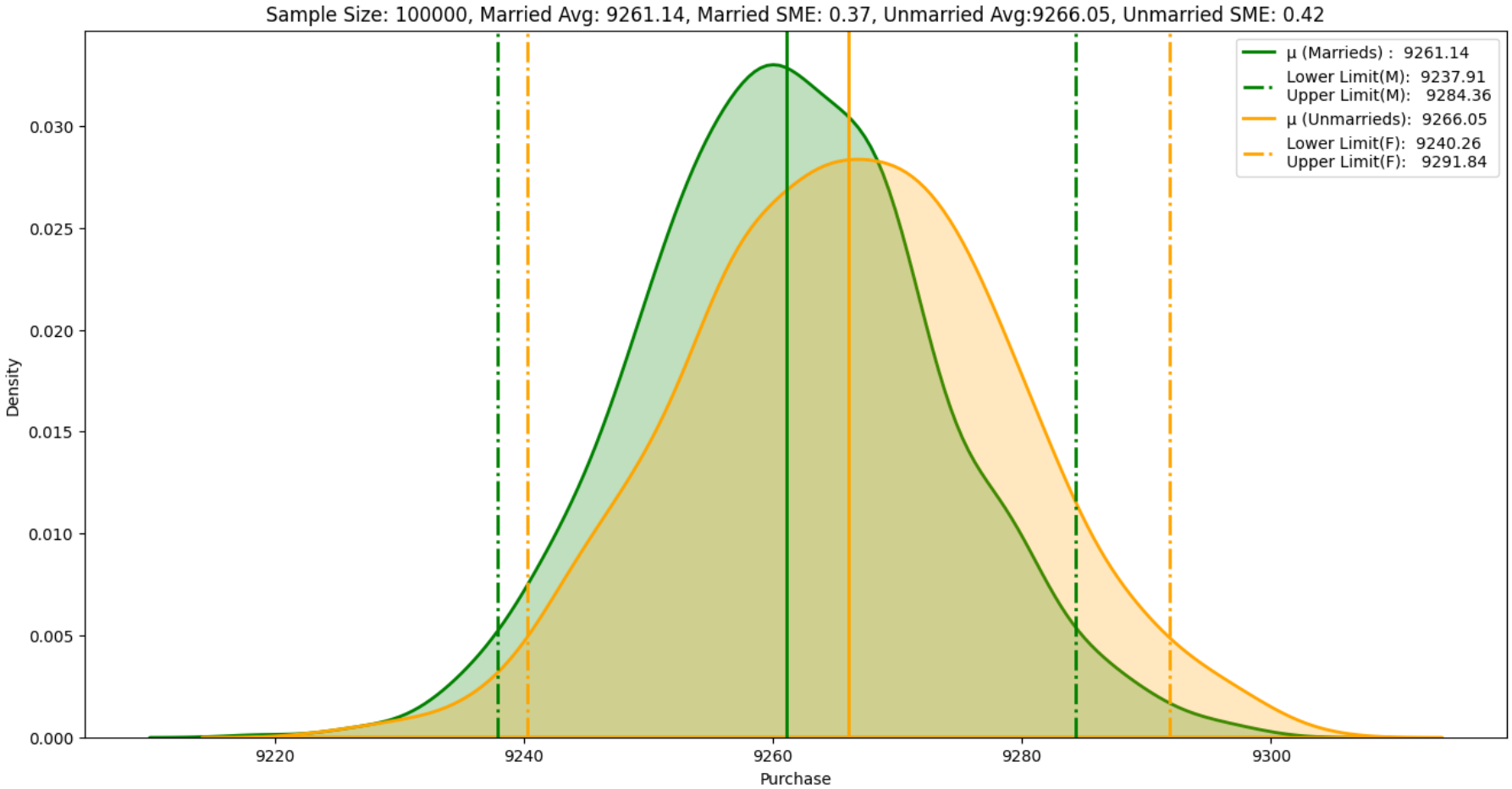
In [48]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 95
itr_size = 1000

for i in sample_sizes:
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un = getCLTMarital(df_married['Purchase'],df_unmarried['Purchase'],i,itr_size

    res = res.append({'Marital_Status':'Married','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m
    res = res.append({'Marital_Status':'Unmarried','Sample Size':i,'Lower Limit':ll_un,'Upper Limit':ul_un,'Sample Mea
```




Sample Size: 100, Married Avg: 9274.77, Married SME: 15.59, Unmarried Avg:9291.23, Unmarried SME: 15.58
μ (Marrieds) : 9274.77
Lower Limit(M): 8308.78
Upper Limit(M): 10240.76
μ (Unmarrieds): 9291.23
Lower Limit(F): 8326.29
Upper Limit(F): 10256.17
Density
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0.0007
0.0008
7500
8000
8500
9000
9500
10000
10500
11000
Purchase


Sample Size: 1000, Married Avg: 9262.24, Married SME: 4.87, Unmarried Avg:9265.43, Unmarried SME: 5.11
μ (Marrieds) : 9262.24
Lower Limit(M): 8960.56
Upper Limit(M): 9563.92
μ (Unmarrieds): 9265.43
Lower Limit(F): 8948.72
Upper Limit(F): 9582.13
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
Purchase


Sample Size: 10000, Married Avg: 9260.13, Married SME: 1.55, Unmarried Avg:9264.08, Unmarried SME: 1.61
μ (Marrieds) : 9260.13
Lower Limit(M): 9164.09
Upper Limit(M): 9356.17
μ (Unmarrieds): 9264.08
Lower Limit(F): 9164.19
Upper Limit(F): 9363.98
Density
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
9100
9200
9300
9400
9500
Purchase


Sample Size: 100000, Married Avg: 9261.14, Married SME: 0.37, Unmarried Avg:9266.05, Unmarried SME: 0.42
μ (Marrieds) : 9261.14
Lower Limit(M): 9237.91
Upper Limit(M): 9284.36
μ (Unmarrieds): 9266.05
Lower Limit(F): 9240.26
Upper Limit(F): 9291.84
Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
9220
9240
9260
9280
9300
Purchase

In [49]:
```
res
```

Out[49]:

	Marital_Status	Sample Size	Lower Limit	Upper Limit	Sample Mean	Confidence Interval	Interval Range	Range
0	Married	10	6629.02	11946.89	9287.95	90	[6629.02, 11946.89]	5317.87
1	Unmarried	10	6611.64	11956.07	9283.86	90	[6611.64, 11956.07]	5344.43
2	Married	100	8417.84	10127.18	9272.51	90	[8417.84, 10127.18]	1709.34
3	Unmarried	100	8410.45	10093.23	9251.84	90	[8410.45, 10093.23]	1682.78
4	Married	1000	8998.01	9528.39	9263.20	90	[8998.01, 9528.39]	530.38
5	Unmarried	1000	9005.91	9525.94	9265.93	90	[9005.91, 9525.94]	520.03
6	Married	10000	9183.61	9340.16	9261.88	90	[9183.61, 9340.16]	156.55
7	Unmarried	10000	9184.29	9345.88	9265.08	90	[9184.29, 9345.88]	161.59
8	Married	100000	9241.28	9282.24	9261.76	90	[9241.28, 9282.24]	40.96
9	Unmarried	100000	9243.60	9288.05	9265.83	90	[9243.6, 9288.05]	44.45
10	Married	10	6026.15	12417.52	9221.84	95	[6026.15, 12417.52]	6391.37
11	Unmarried	10	6116.05	12517.34	9316.70	95	[6116.05, 12517.34]	6401.29
12	Married	100	8308.78	10240.76	9274.77	95	[8308.78, 10240.76]	1931.98
13	Unmarried	100	8326.29	10256.17	9291.23	95	[8326.29, 10256.17]	1929.88
14	Married	1000	8960.56	9563.92	9262.24	95	[8960.56, 9563.92]	603.36
15	Unmarried	1000	8948.72	9582.13	9265.43	95	[8948.72, 9582.13]	633.41
16	Married	10000	9164.09	9356.17	9260.13	95	[9164.09, 9356.17]	192.08
17	Unmarried	10000	9164.19	9363.98	9264.08	95	[9164.19, 9363.98]	199.79
18	Married	100000	9237.91	9284.36	9261.14	95	[9237.91, 9284.36]	46.45
19	Unmarried	100000	9240.26	9291.84	9266.05	95	[9240.26, 9291.84]	51.58

For married and unmarried customers, sample size 10, confidence interval 90 we can observe that the interval range is overlapping

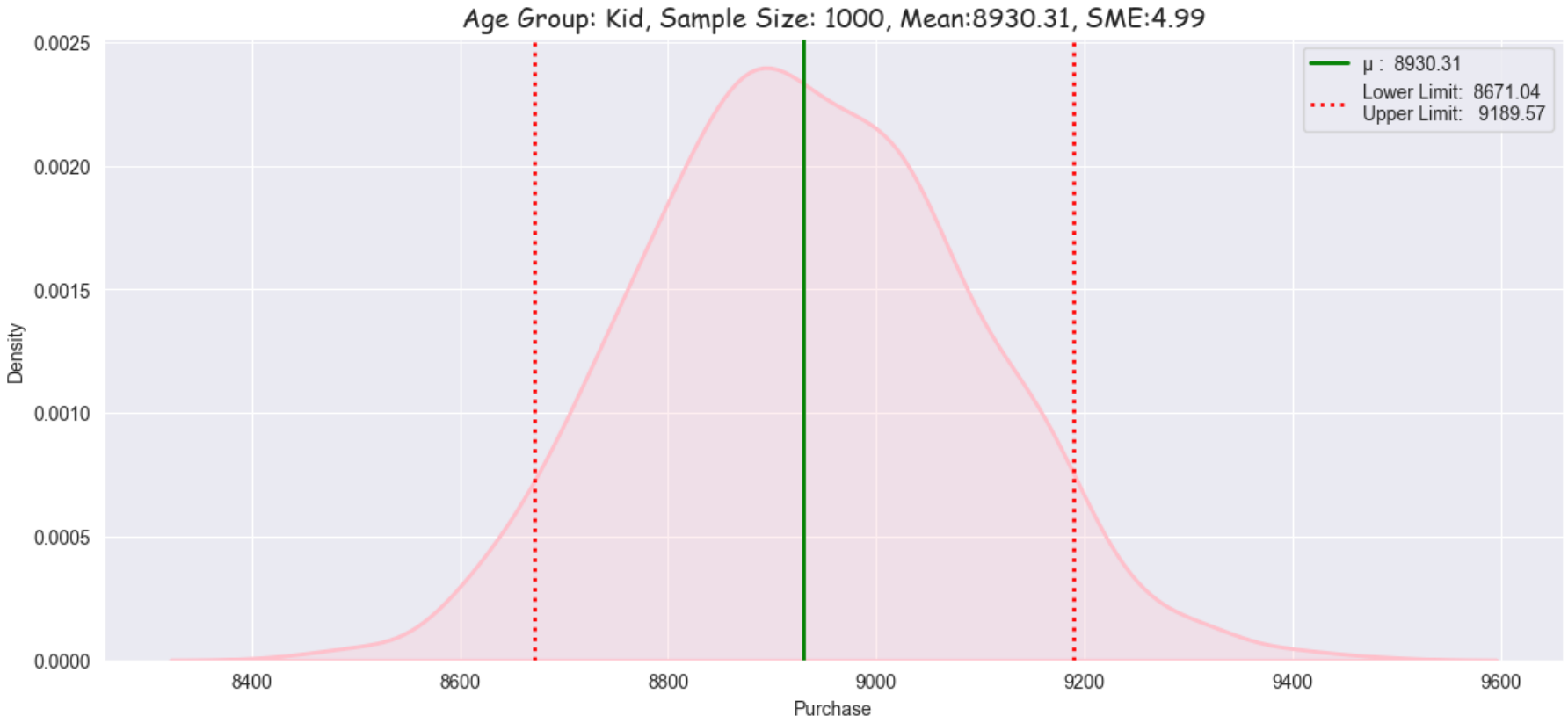
For married and unmarried customers, sample size 100000, confidence interval 90 we can observe that the interval range is still overlapping

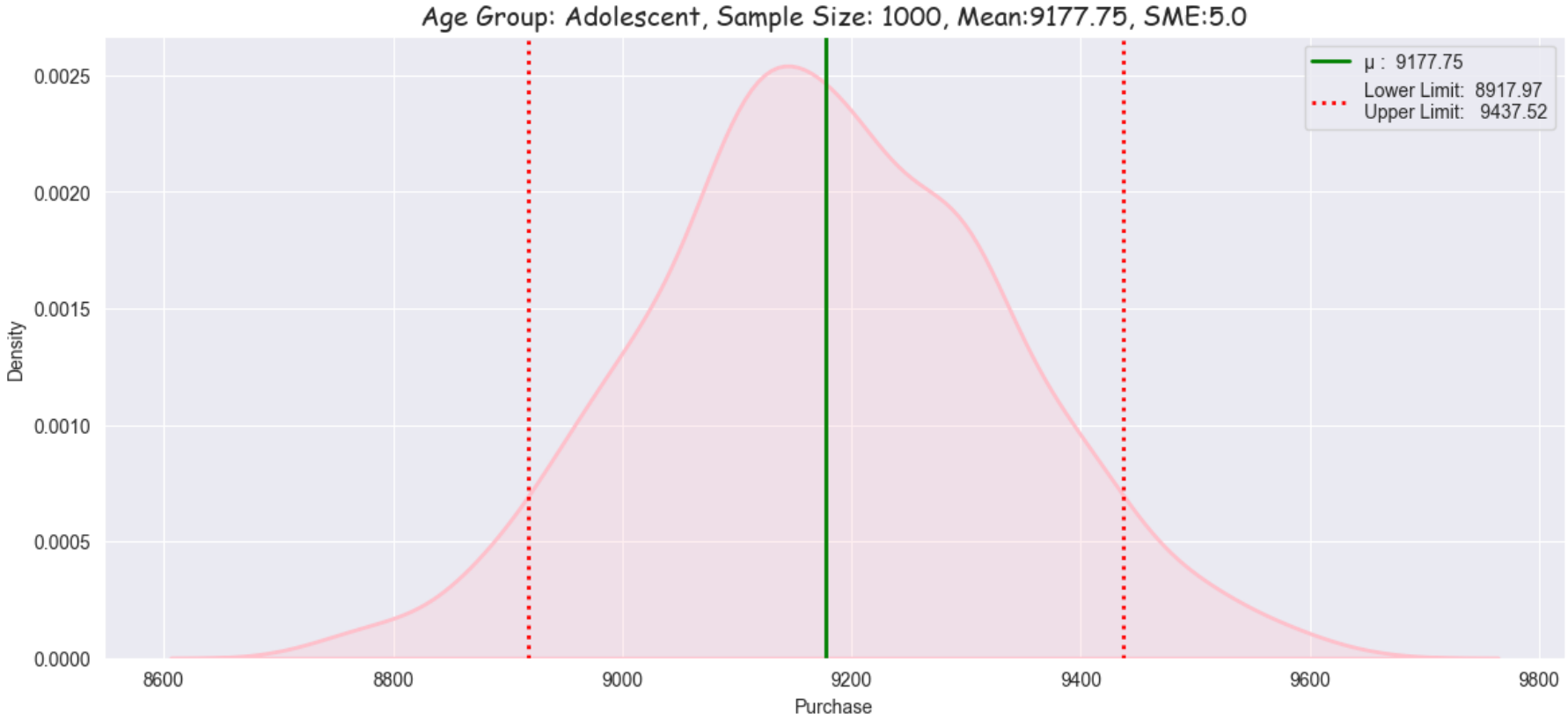
This means there is no effect of marital status on purchase habits of customers

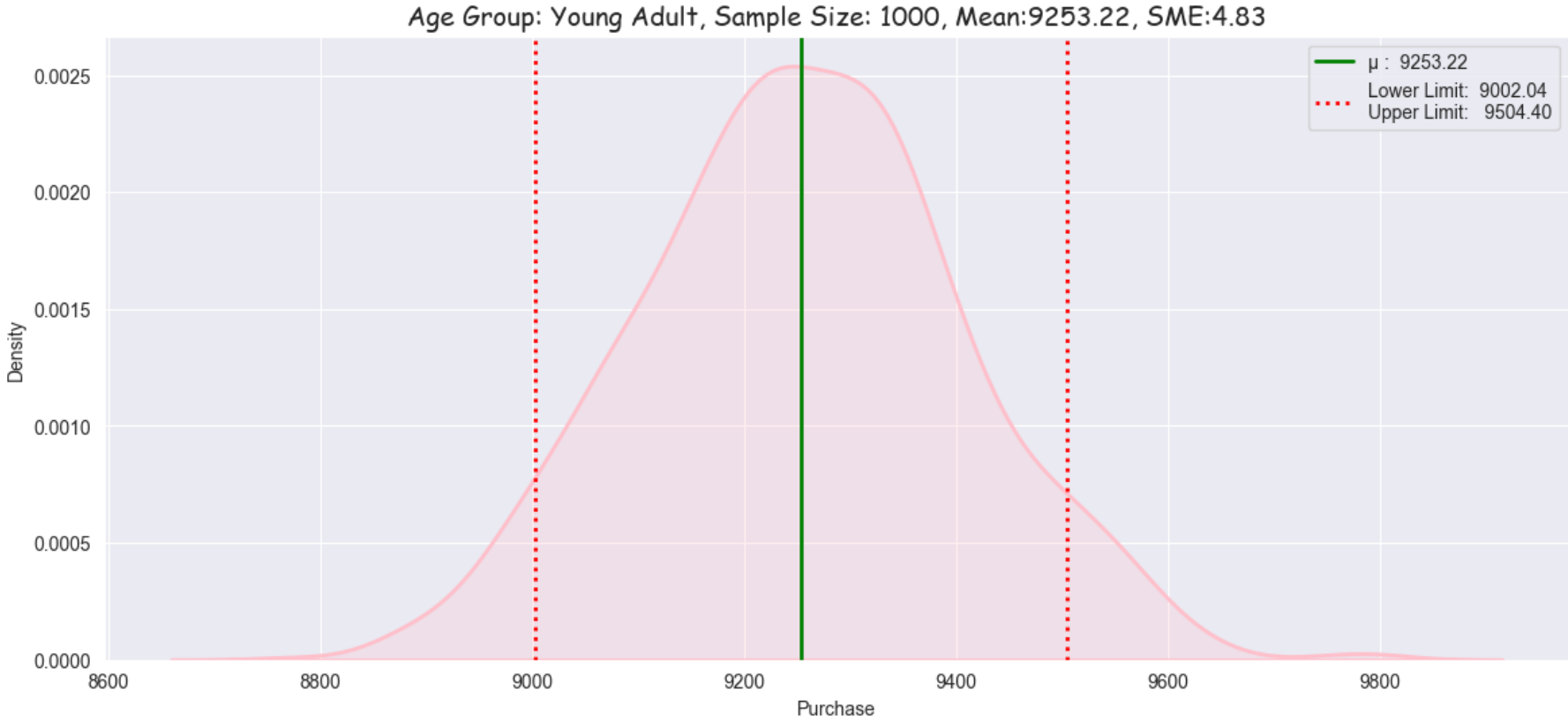
In [53]:
```
def getCLT_age(sample, sample_size, itr_size=1000, ci = 90):
    ci = ci/100

    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="pink",fill=True,linewidth=2)

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

    plt.title(f"Age Group: {age_group[flag]}, Sample Size: {sample_size}, Mean:{np.round(mean,2)}, SME:{np.round(sem,2
    plt.xlabel('Purchase')
    plt.axvline(mean, color = 'g', linestyle = 'solid', linewidth = 2,label=label_mean)
    plt.axvline(upper_limit, color = 'r', linestyle = 'dotted', linewidth = 2,label=label_ult)
    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)
```

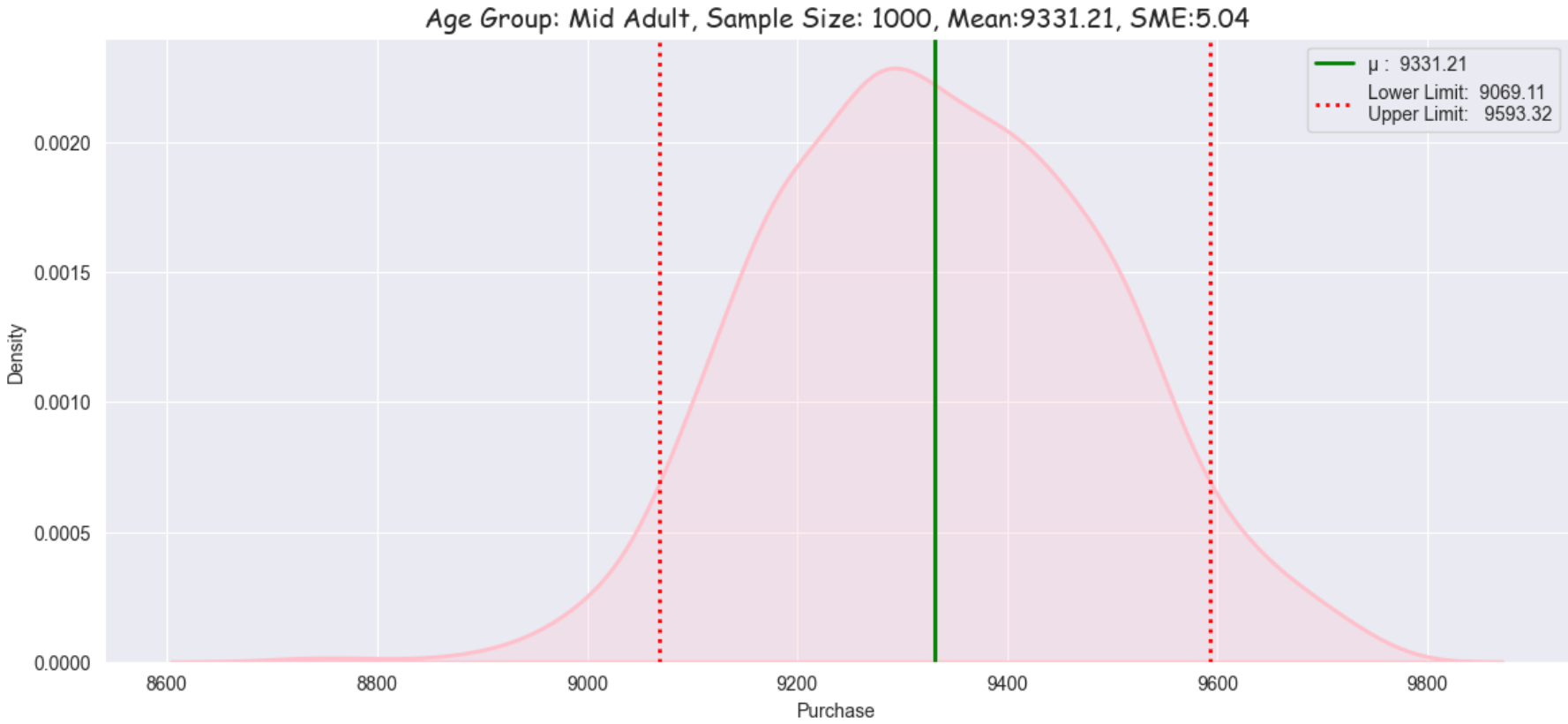
Lets visualise the graphs of 1000 mean values of purchase samples for sample size of 1000 for all the age groups with 90% confidence interval.

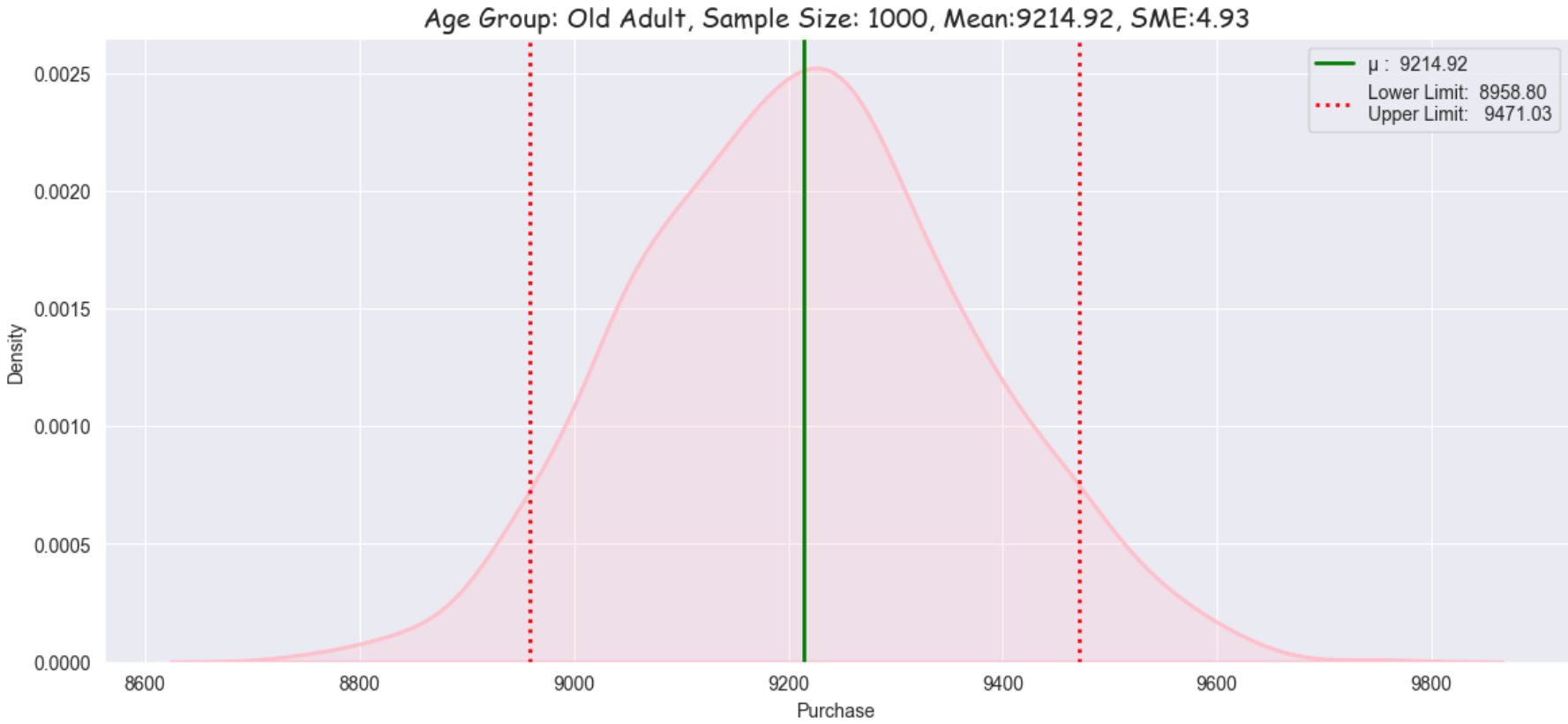
In [54]:
```
ci = 90
itr_size = 1000
sample_size = 1000
flag = 0
global age_group
age_group = ["Kid","Adolescent","Young Adult","Mid Adult","Old Adult","Senior Adult","Retired Adult"]

res = pd.DataFrame(columns = ['Age_Group','Sample Size','Lower Limit','Upper Limit','Sample Mean','Confidence Interval

for i in age_group:
    m_avg, ll, ul, mean = getCLT_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

    res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':ll,'Upper Limit':ul,'Sample Mean':mean,'Co
```



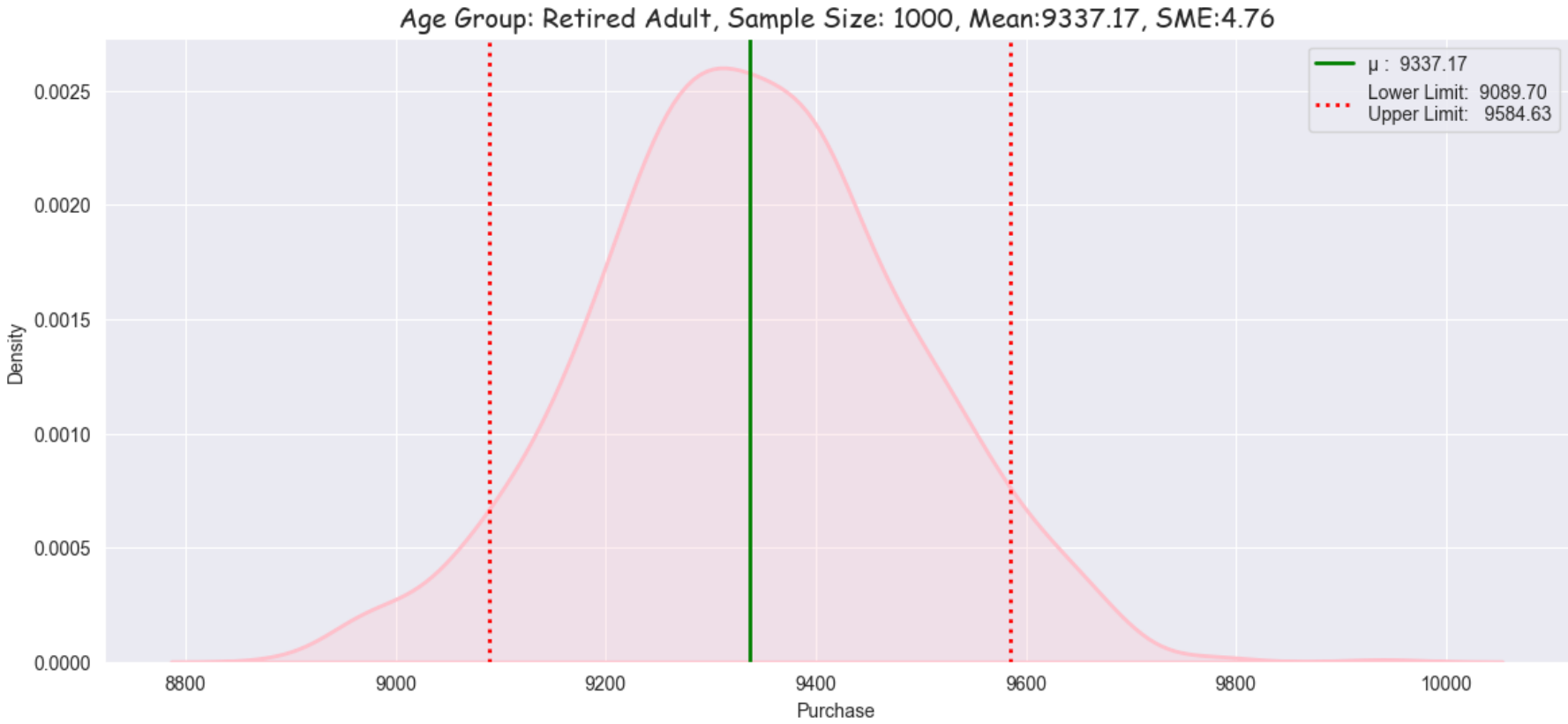

Age Group: Adolescent, Sample Size: 1000, Mean:9177.75, SME:5.0
μ : 9177.75
Lower Limit: 8917.97
Upper Limit: 9437.52
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: Young Adult, Sample Size: 1000, Mean:9253.22, SME:4.83
μ : 9253.22
Lower Limit: 9002.04
Upper Limit: 9504.40
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase

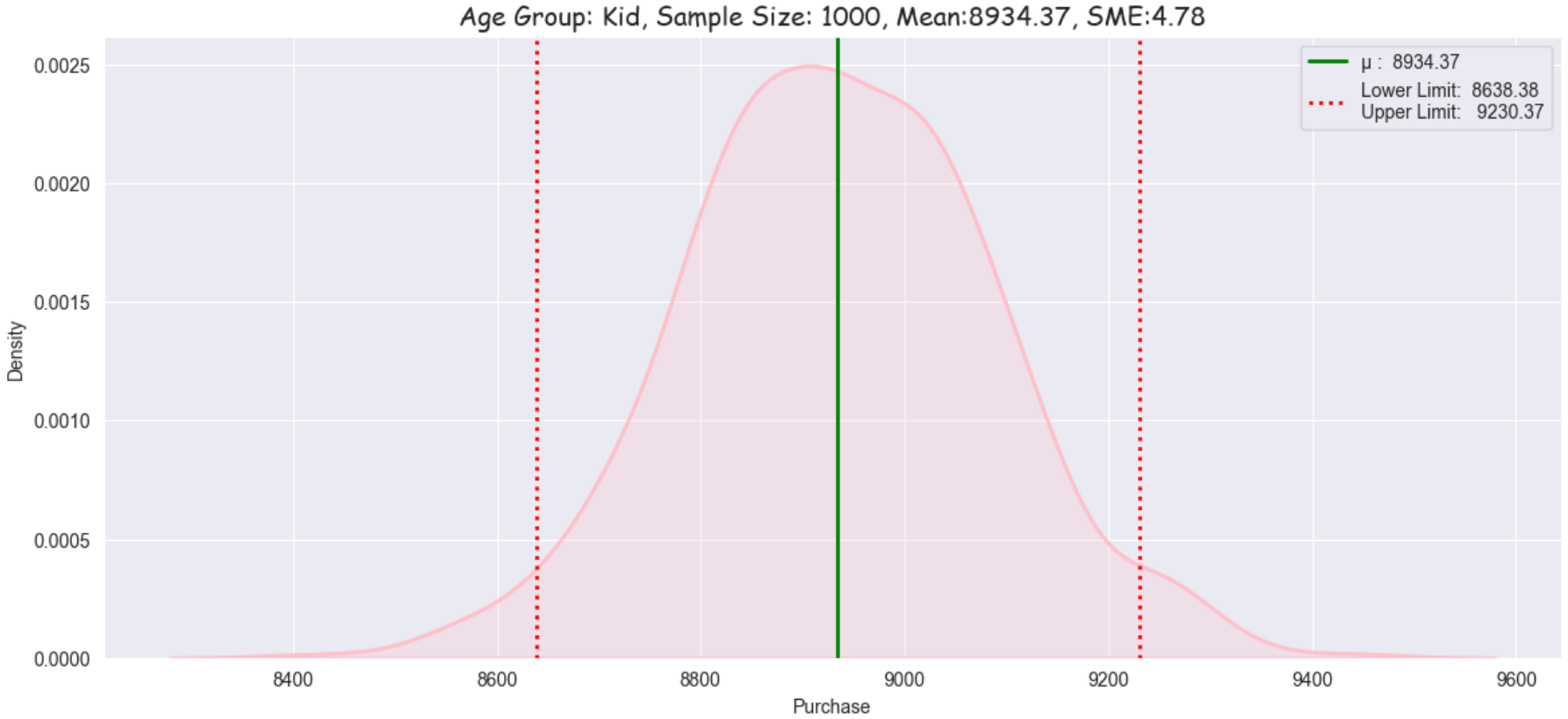

Age Group: Mid Adult, Sample Size: 1000, Mean:9331.21, SME:5.04
μ : 9331.21
Lower Limit: 9069.11
Upper Limit: 9593.32
Density
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase

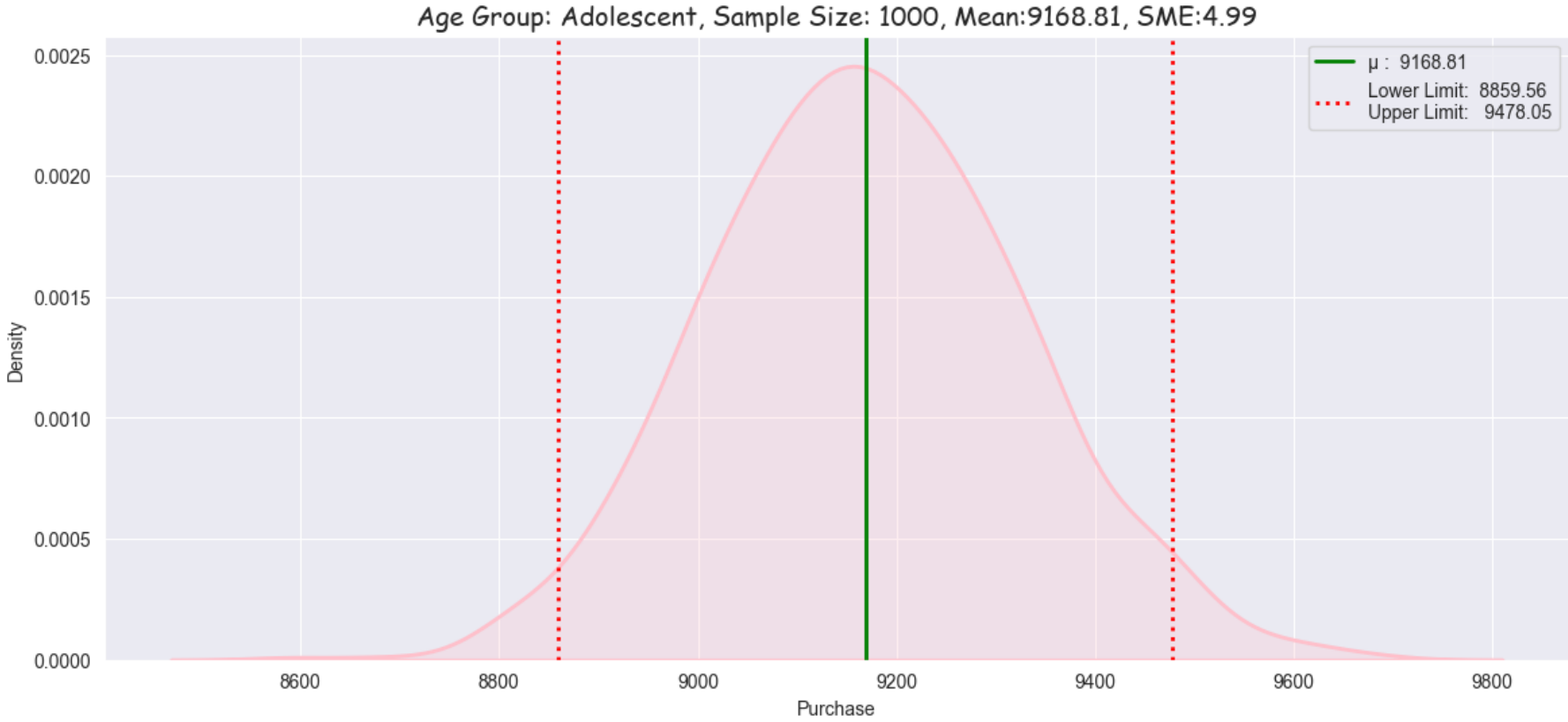

Age Group: Old Adult, Sample Size: 1000, Mean:9214.92, SME:4.93
μ : 9214.92
Lower Limit: 8958.80
Upper Limit: 9471.03
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase

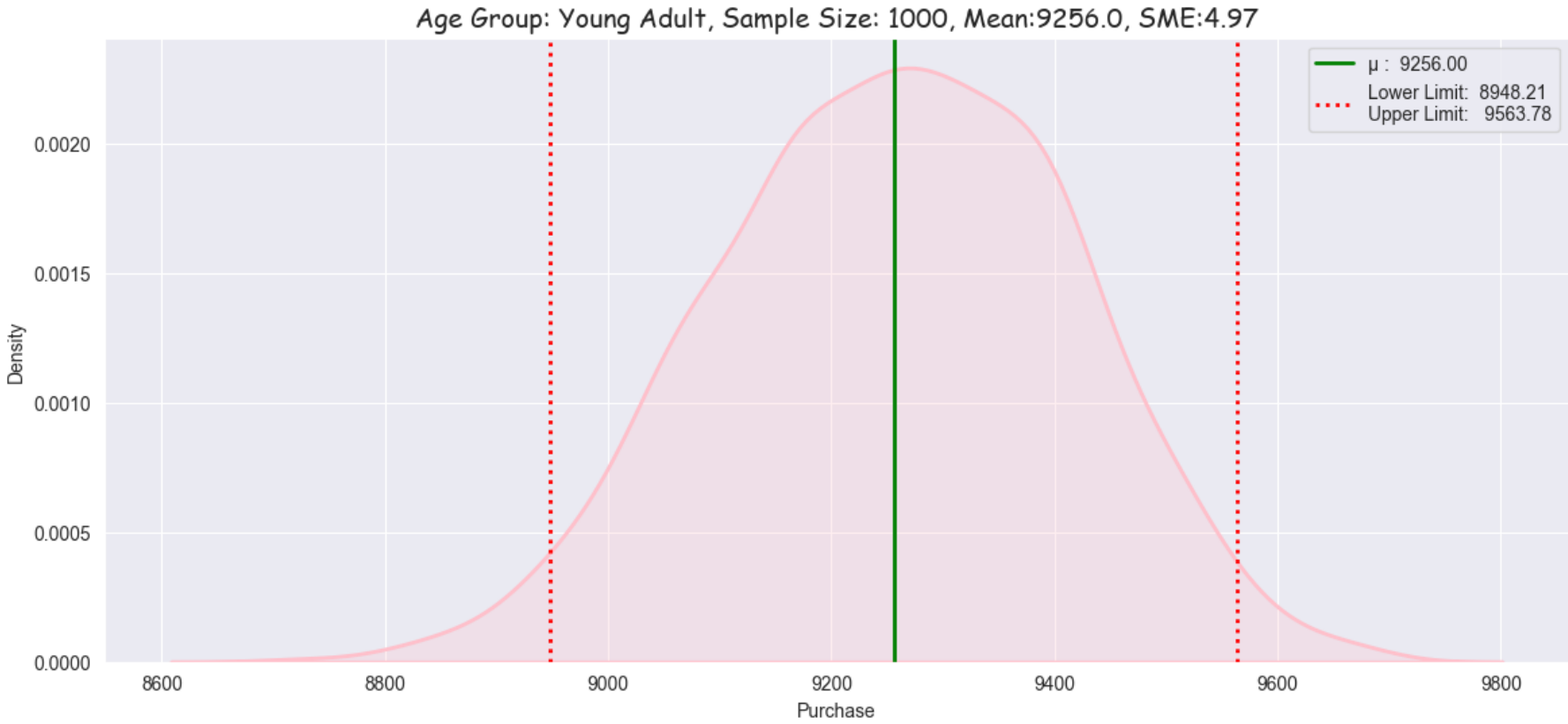

Age Group: Senior Adult, Sample Size: 1000, Mean:9538.23, SME:4.9
μ : 9538.23
Lower Limit: 9283.66
Upper Limit: 9792.80
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
9000
9200
9400
9600
9800
10000
Purchase

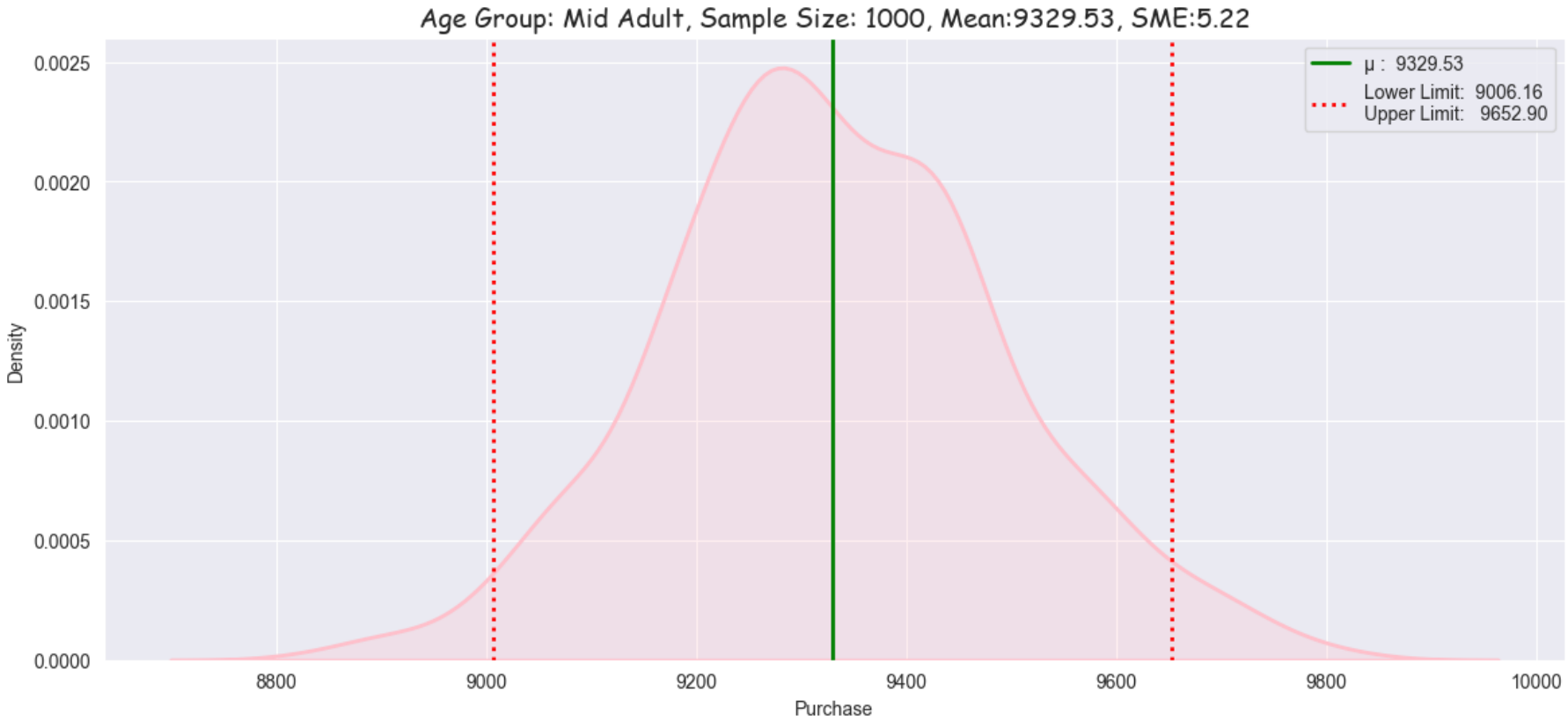

Age Group: Retired Adult, Sample Size: 1000, Mean:9337.17, SME:4.76
μ : 9337.17
Lower Limit: 9089.70
Upper Limit: 9584.63
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000
Purchase

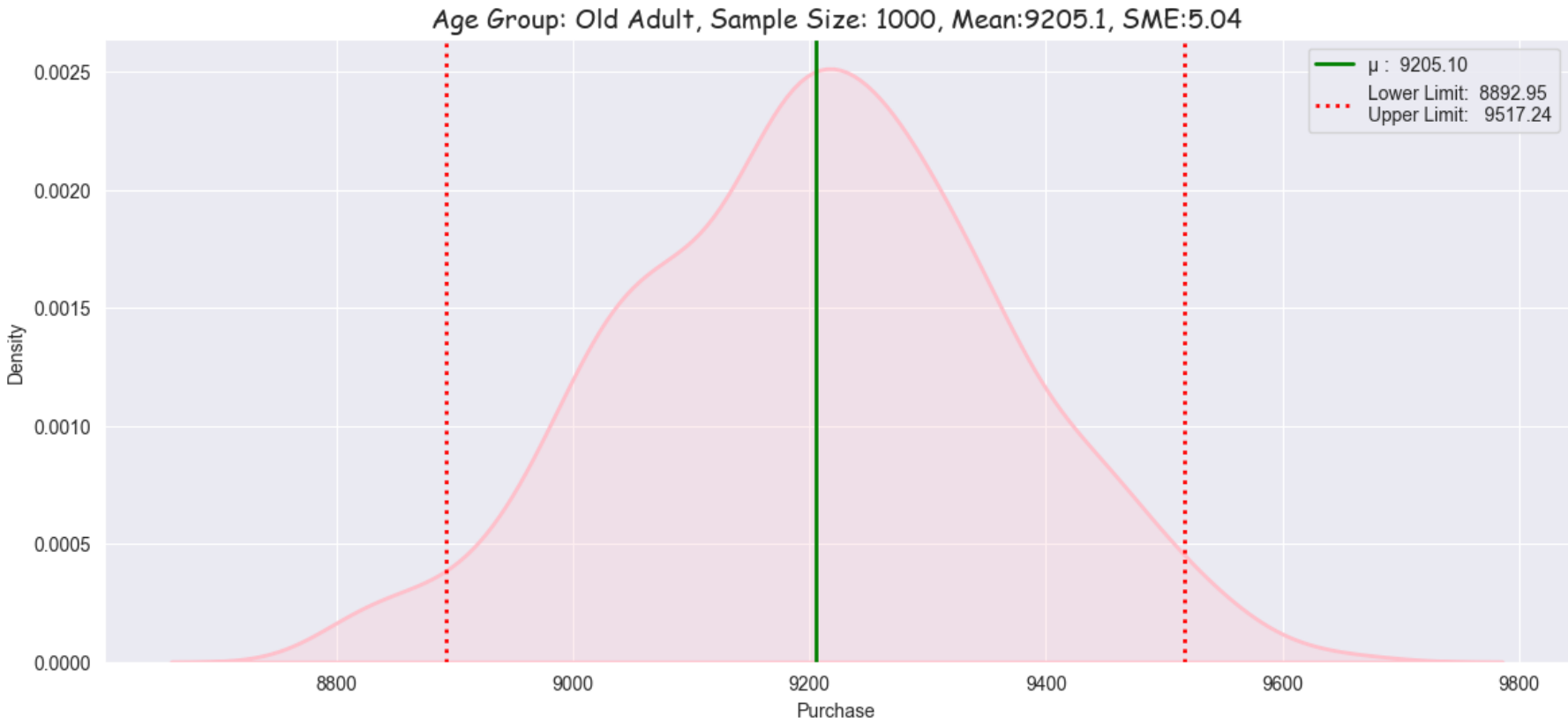
In [55]:
```
ci = 95
itr_size = 1000
sample_size = 1000
flag = 0

for i in age_group:
     m_avg, ll, ul, mean = getCLT_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

     res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':ll,'Upper Limit':ul,'Sample Mean':mean,'
```

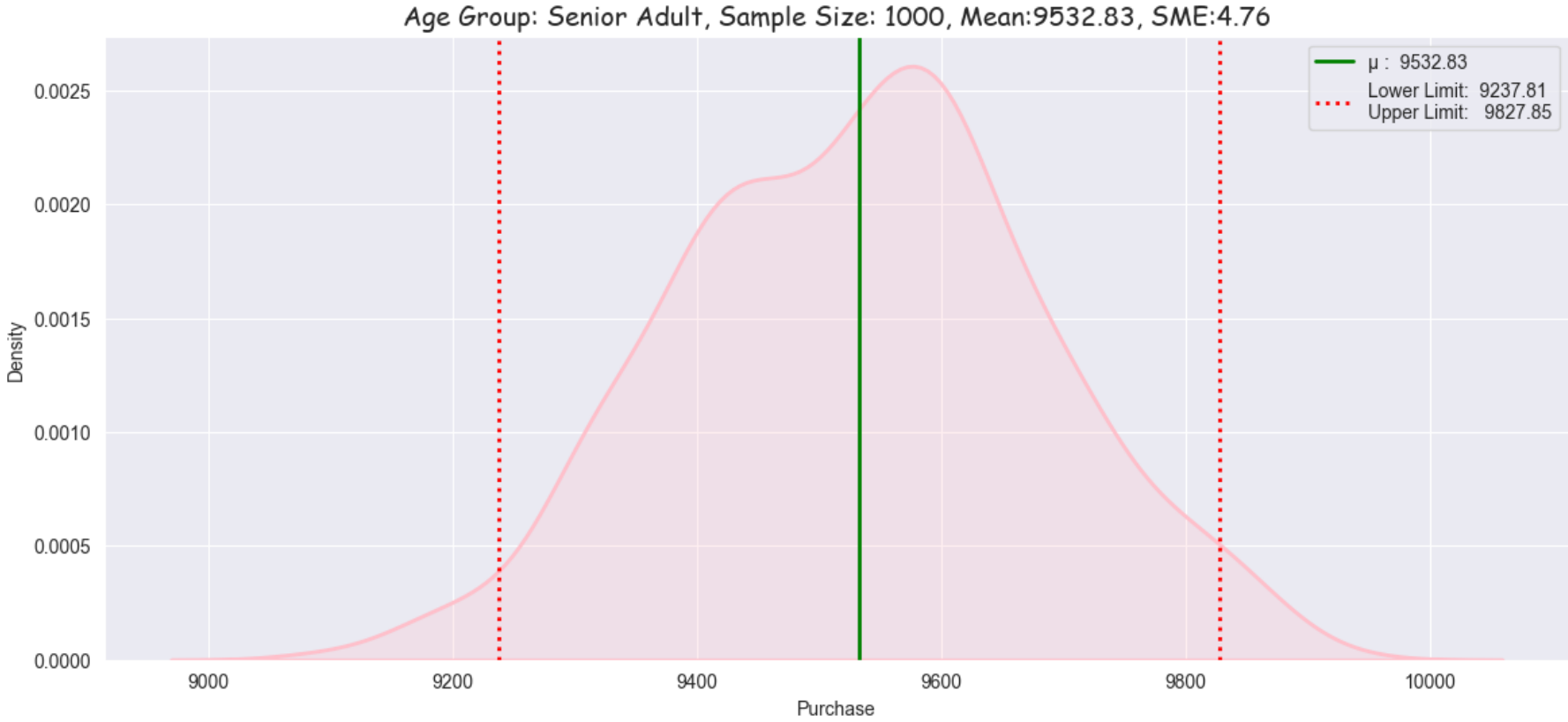


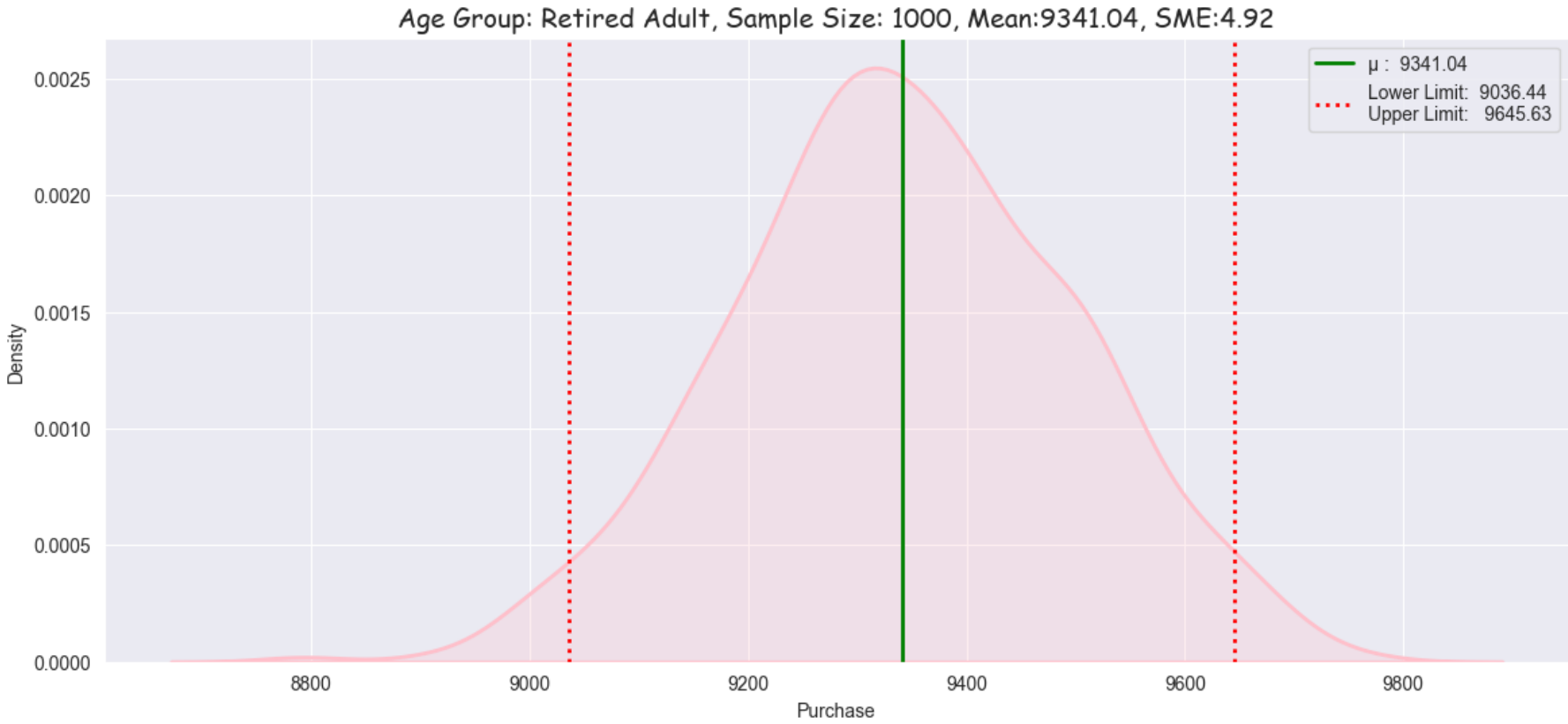

Age Group: Adolescent, Sample Size: 1000, Mean:9168.81, SME:4.99
μ : 9168.81
Lower Limit: 8859.56
Upper Limit: 9478.05
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: Young Adult, Sample Size: 1000, Mean:9256.0, SME:4.97
μ : 9256.00
Lower Limit: 8948.21
Upper Limit: 9563.78
Density
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: Mid Adult, Sample Size: 1000, Mean:9329.53, SME:5.22
μ : 9329.53
Lower Limit: 9006.16
Upper Limit: 9652.90
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000
Purchase


Age Group: Old Adult, Sample Size: 1000, Mean:9205.1, SME:5.04
μ : 9205.10
Lower Limit: 8892.95
Upper Limit: 9517.24
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
Purchase


Age Group: Senior Adult, Sample Size: 1000, Mean:9532.83, SME:4.76
μ : 9532.83
Lower Limit: 9237.81
Upper Limit: 9827.85
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
9000
9200
9400
9600
9800
10000
Purchase


Age Group: Retired Adult, Sample Size: 1000, Mean:9341.04, SME:4.92
μ : 9341.04
Lower Limit: 9036.44
Upper Limit: 9645.63
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
Purchase

In [56]:
```
res
```

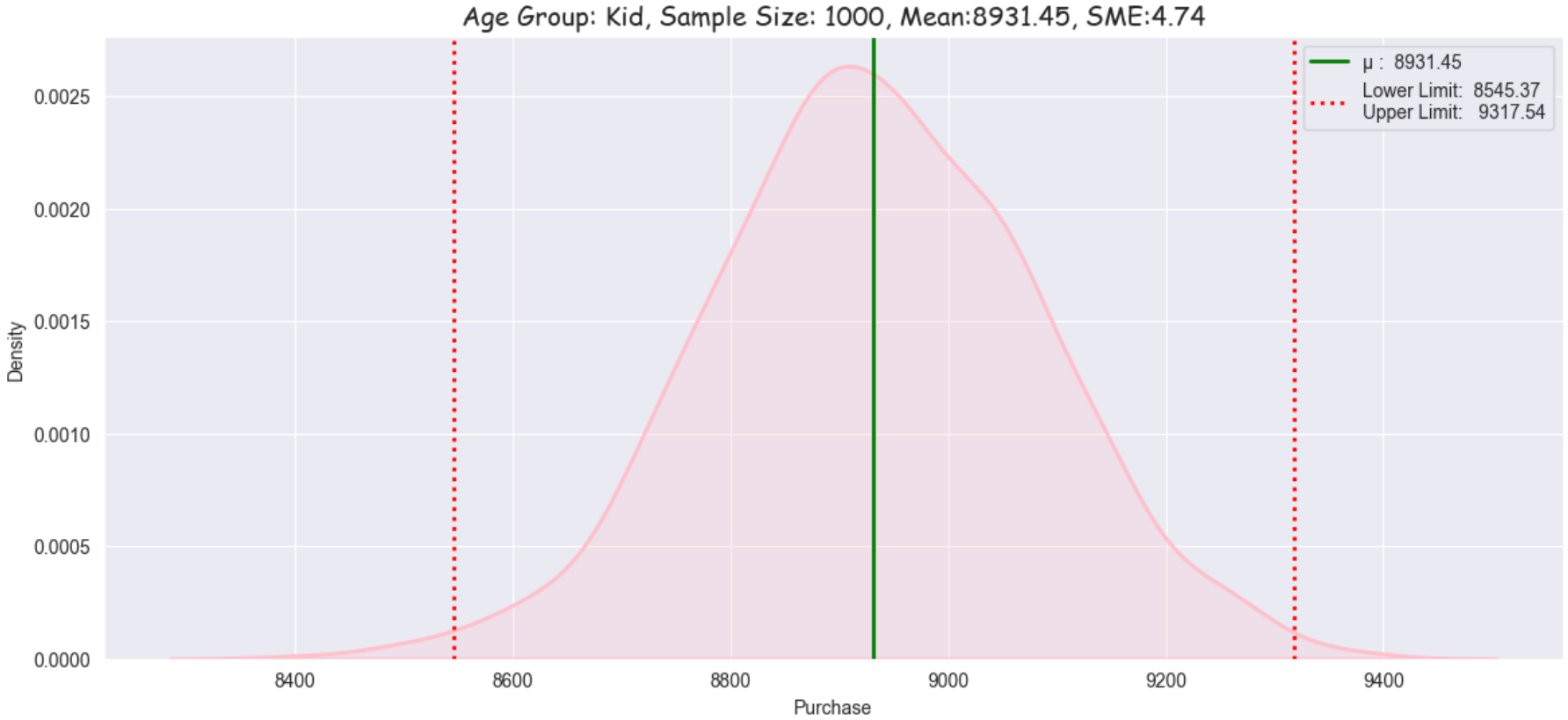
Out[56]:

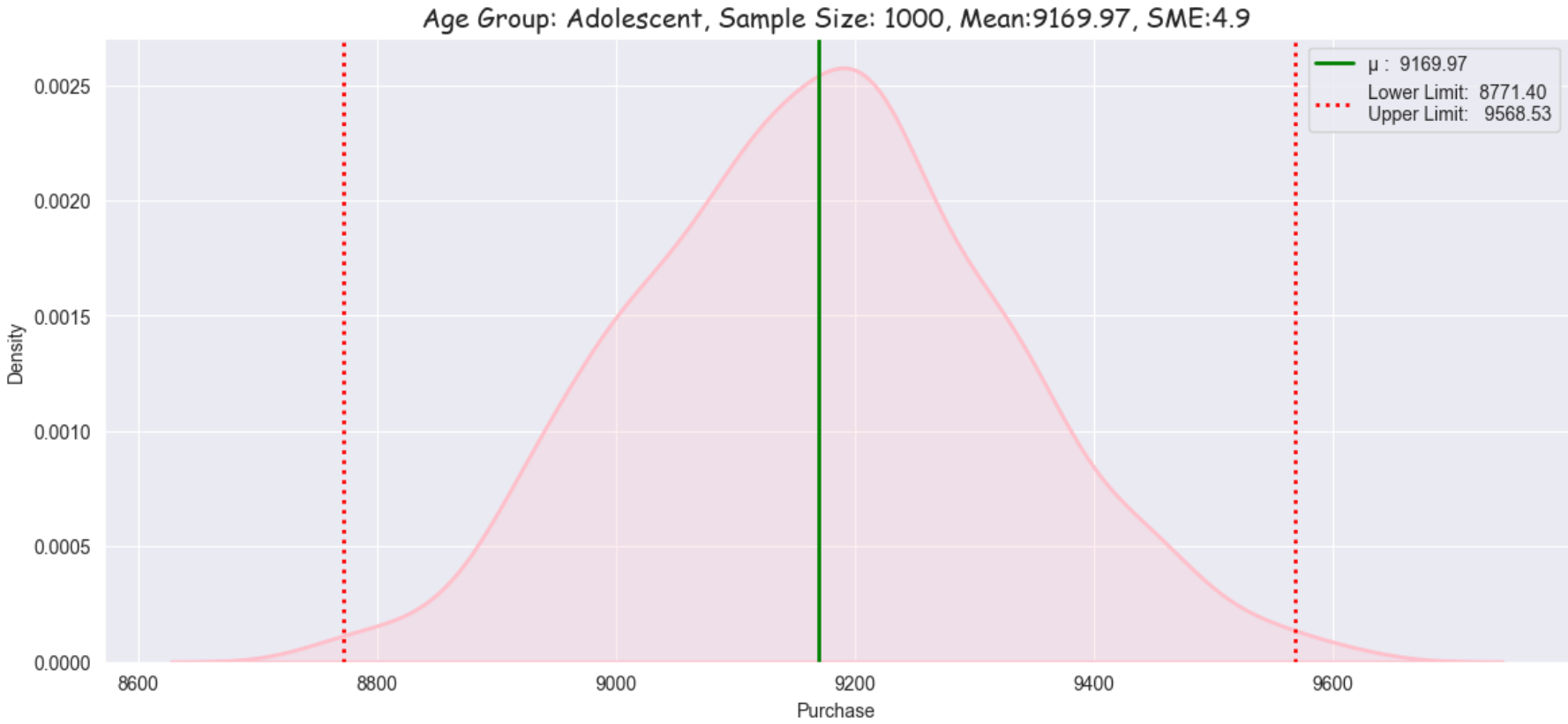
	Age_Group	Sample Size	Lower Limit	Upper Limit	Sample Mean	Confidence Interval	Interval Range	Range
0	Kid	1000	8671.04	9189.57	8930.31	90	[8671.04, 9189.57]	518.53
1	Adolescent	1000	8917.97	9437.52	9177.75	90	[8917.97, 9437.52]	519.55
2	Young Adult	1000	9002.04	9504.40	9253.22	90	[9002.04, 9504.4]	502.36
3	Mid Adult	1000	9069.11	9593.32	9331.21	90	[9069.11, 9593.32]	524.21
4	Old Adult	1000	8958.80	9471.03	9214.92	90	[8958.8, 9471.03]	512.23
5	Senior Adult	1000	9283.66	9792.80	9538.23	90	[9283.66, 9792.8]	509.14
6	Retired Adult	1000	9089.70	9584.63	9337.17	90	[9089.7, 9584.63]	494.93
7	Kid	1000	8638.38	9230.37	8934.37	95	[8638.38, 9230.37]	591.99
8	Adolescent	1000	8859.56	9478.05	9168.81	95	[8859.56, 9478.05]	618.49
9	Young Adult	1000	8948.21	9563.78	9256.00	95	[8948.21, 9563.78]	615.57
10	Mid Adult	1000	9006.16	9652.90	9329.53	95	[9006.16, 9652.9]	646.74
11	Old Adult	1000	8892.95	9517.24	9205.10	95	[8892.95, 9517.24]	624.29
12	Senior Adult	1000	9237.81	9827.85	9532.83	95	[9237.81, 9827.85]	590.04
13	Retired Adult	1000	9036.44	9645.63	9341.04	95	[9036.44, 9645.63]	609.19

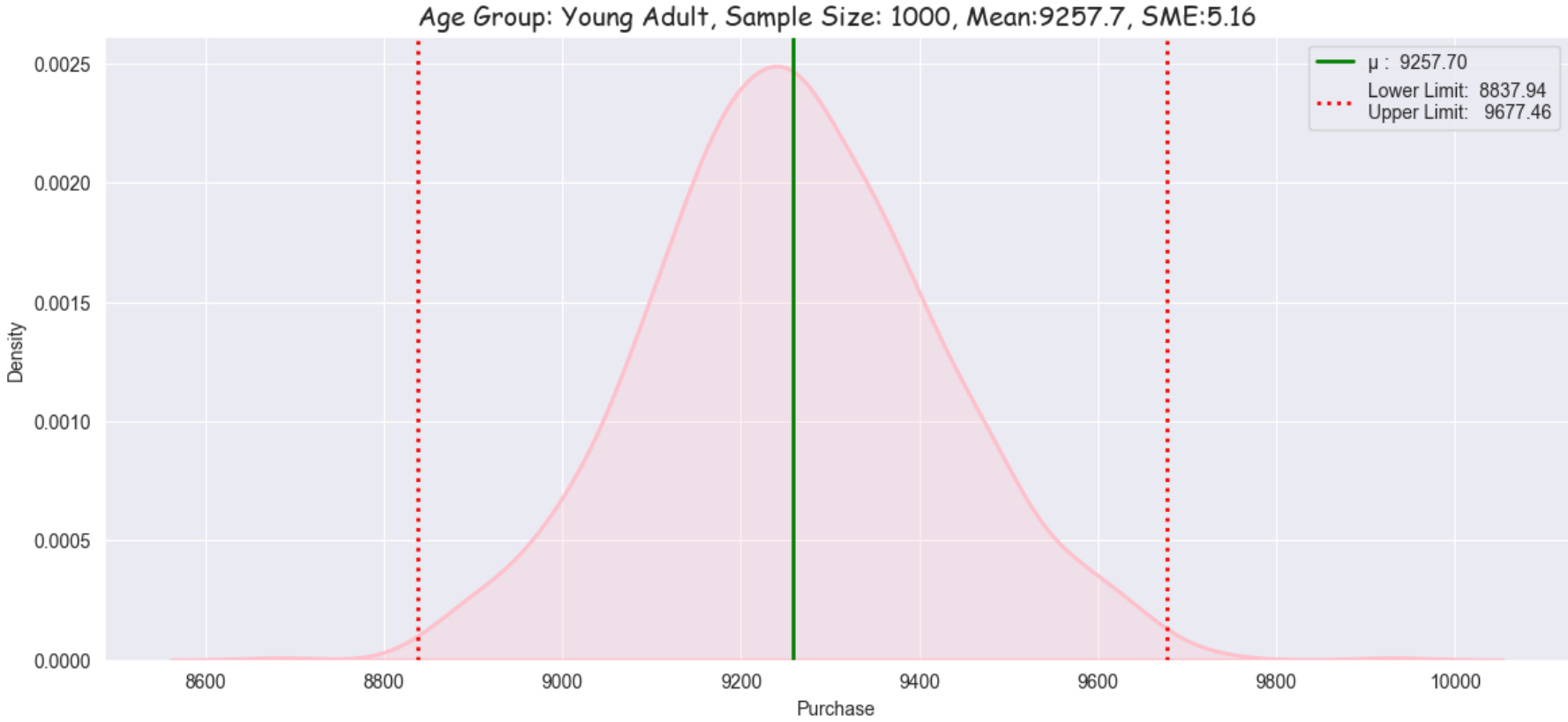
All the age groups still have overlap which makes it difficult to interpret the ranges.

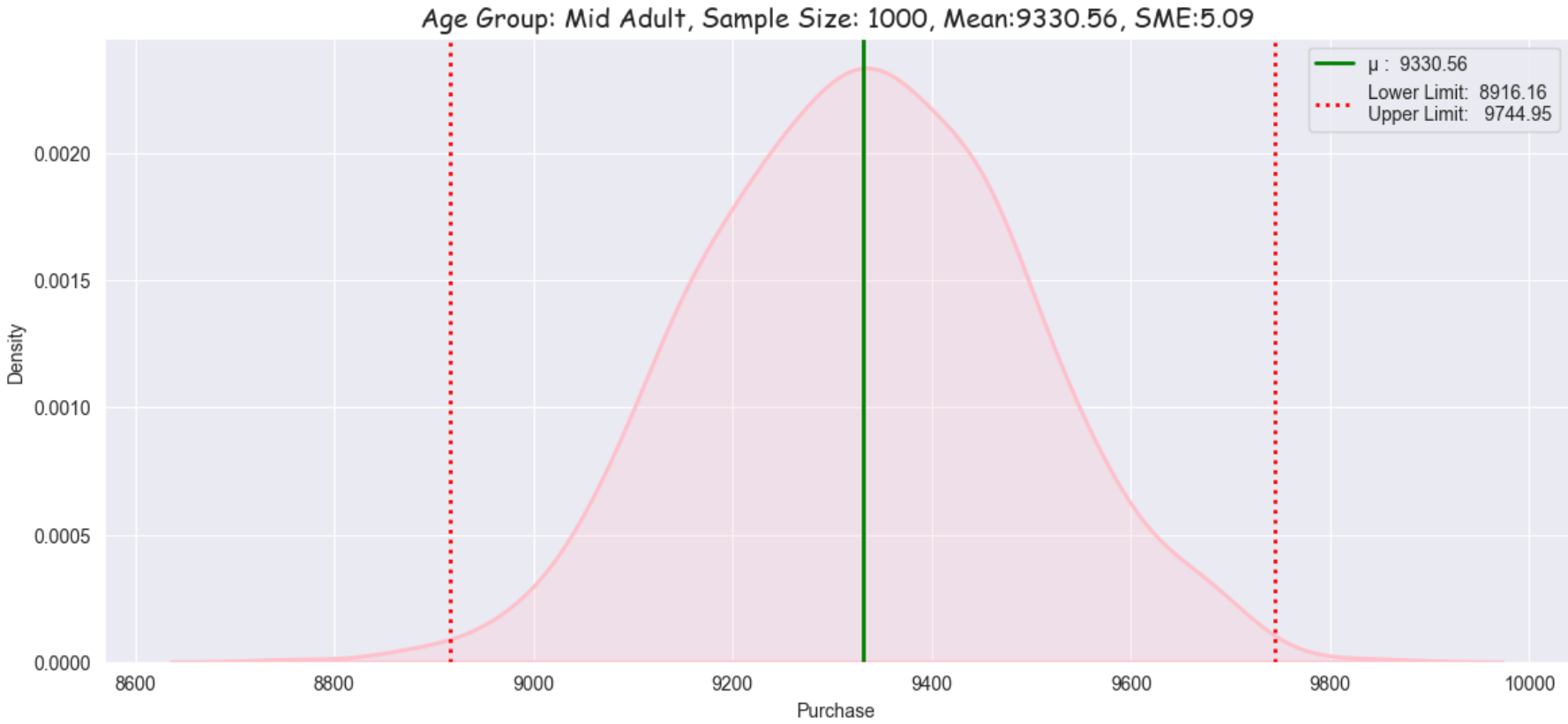
So now, Lets visualise the graphs of 1000 mean values of purchase samples for sample size of 1000 for all the age groups with 99% confidence interval.

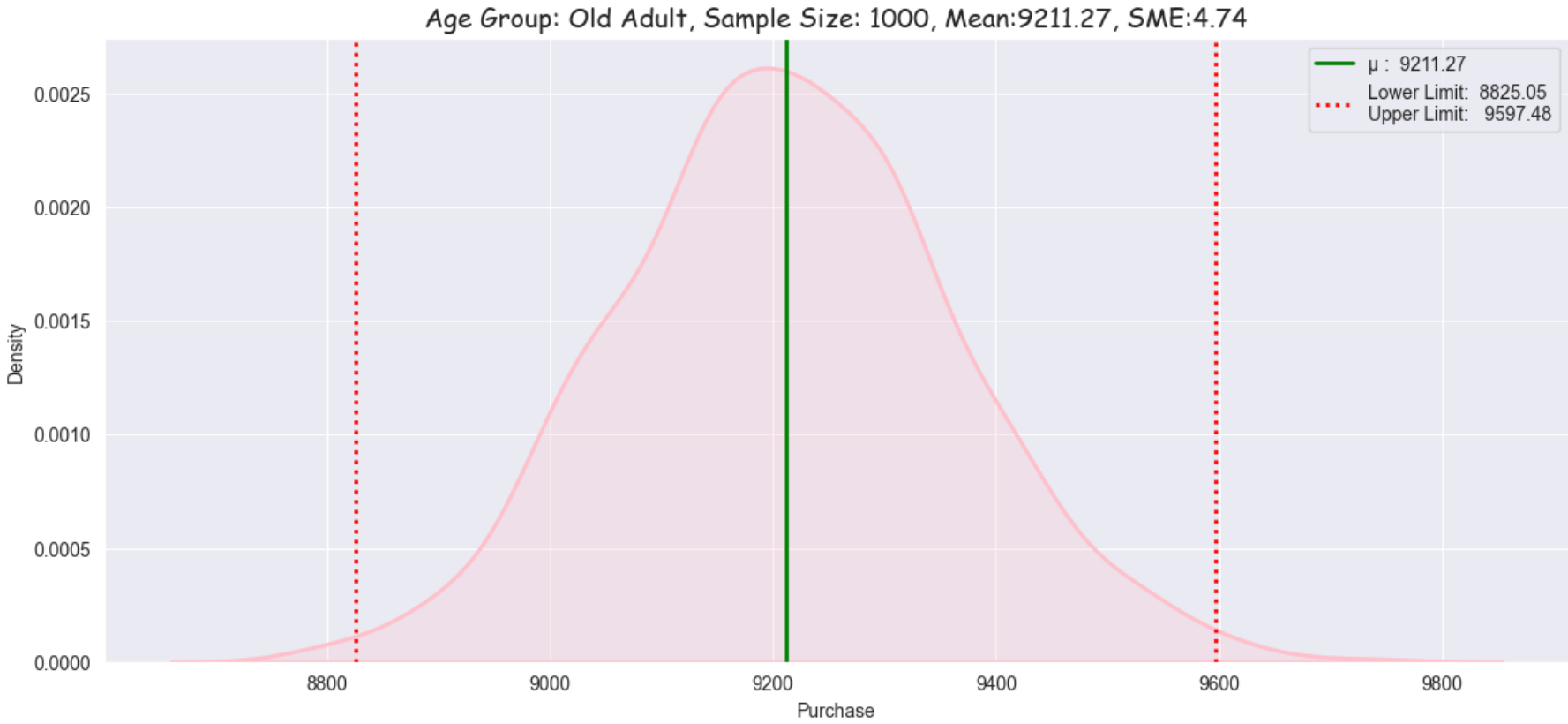
In [57]:
```
ci = 99
itr_size = 1000
sample_size = 1000
flag = 0

for i in age_group:
     m_avg, ll, ul, mean = getCLT_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

     res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':ll,'Upper Limit':ul,'Sample Mean':mean,'(
```

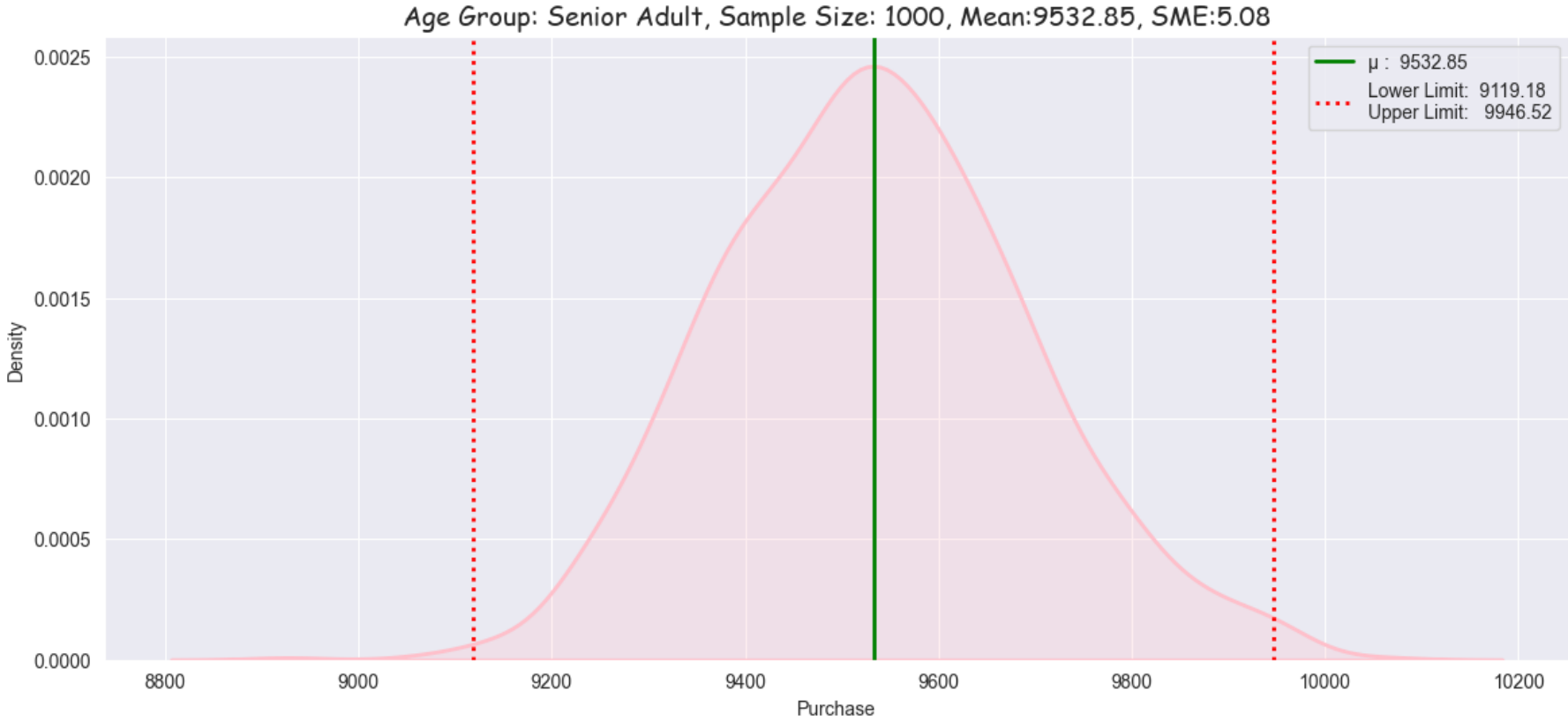



Age Group: Adolescent, Sample Size: 1000, Mean:9169.97, SME:4.9
μ : 9169.97
Lower Limit: 8771.40
Upper Limit: 9568.53
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
Purchase


Age Group: Young Adult, Sample Size: 1000, Mean:9257.7, SME:5.16
μ : 9257.70
Lower Limit: 8837.94
Upper Limit: 9677.46
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
10000
Purchase


Age Group: Mid Adult, Sample Size: 1000, Mean:9330.56, SME:5.09
μ : 9330.56
Lower Limit: 8916.16
Upper Limit: 9744.95
Density
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
10000
Purchase


Age Group: Old Adult, Sample Size: 1000, Mean:9211.27, SME:4.74
μ : 9211.27
Lower Limit: 8825.05
Upper Limit: 9597.48
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
Purchase


Age Group: Senior Adult, Sample Size: 1000, Mean:9532.85, SME:5.08
μ : 9532.85
Lower Limit: 9119.18
Upper Limit: 9946.52
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000
10200
Purchase


Age Group: Retired Adult, Sample Size: 1000, Mean:9341.89, SME:4.77
μ : 9341.89
Lower Limit: 8953.44
Upper Limit: 9730.35
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
Purchase

In [58]:
```
res
```

Out[58]:

	Age_Group	Sample Size	Lower Limit	Upper Limit	Sample Mean	Confidence Interval	Interval Range	Range
0	Kid	1000	8671.04	9189.57	8930.31	90	[8671.04, 9189.57]	518.53
1	Adolescent	1000	8917.97	9437.52	9177.75	90	[8917.97, 9437.52]	519.55
2	Young Adult	1000	9002.04	9504.40	9253.22	90	[9002.04, 9504.4]	502.36
3	Mid Adult	1000	9069.11	9593.32	9331.21	90	[9069.11, 9593.32]	524.21
4	Old Adult	1000	8958.80	9471.03	9214.92	90	[8958.8, 9471.03]	512.23
5	Senior Adult	1000	9283.66	9792.80	9538.23	90	[9283.66, 9792.8]	509.14
6	Retired Adult	1000	9089.70	9584.63	9337.17	90	[9089.7, 9584.63]	494.93
7	Kid	1000	8638.38	9230.37	8934.37	95	[8638.38, 9230.37]	591.99
8	Adolescent	1000	8859.56	9478.05	9168.81	95	[8859.56, 9478.05]	618.49
9	Young Adult	1000	8948.21	9563.78	9256.00	95	[8948.21, 9563.78]	615.57
10	Mid Adult	1000	9006.16	9652.90	9329.53	95	[9006.16, 9652.9]	646.74
11	Old Adult	1000	8892.95	9517.24	9205.10	95	[8892.95, 9517.24]	624.29
12	Senior Adult	1000	9237.81	9827.85	9532.83	95	[9237.81, 9827.85]	590.04
13	Retired Adult	1000	9036.44	9645.63	9341.04	95	[9036.44, 9645.63]	609.19
14	Kid	1000	8545.37	9317.54	8931.45	99	[8545.37, 9317.54]	772.17
15	Adolescent	1000	8771.40	9568.53	9169.97	99	[8771.4, 9568.53]	797.13
16	Young Adult	1000	8837.94	9677.46	9257.70	99	[8837.94, 9677.46]	839.52
17	Mid Adult	1000	8916.16	9744.95	9330.56	99	[8916.16, 9744.95]	828.79
18	Old Adult	1000	8825.05	9597.48	9211.27	99	[8825.05, 9597.48]	772.43
19	Senior Adult	1000	9119.18	9946.52	9532.85	99	[9119.18, 9946.52]	827.34
20	Retired Adult	1000	8953.44	9730.35	9341.89	99	[8953.44, 9730.35]	776.91

We can say that age group does not have much effect on the spending of customers as their interval range is overalpping with 90%, 95% and 99% confidence intervals.

Inferences

- ~80% of the users are between the age 18-50 (40%: 26-35, 18%: 18-25, 20%: 36-45)
- 75% of the users are Male and 25% are Female. Males clearly purchase more than females.
- 59% Single, 41% Married
- 35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years
- The majority of our customers come from city category B but customers come from City category C purchased more.
- Majority of Customers purchase within the 5,000 - 10,000 range.
- Most mall customers are between the ages of 26 and 35, 60% of purchases are made by people between the ages of 26 and 45
- City Category B accounts for 42%, City Category C 31%, and City Category A represents 27% of all customer purchases.Purchases are high in city category C
- Most mall customers are between the ages of 26 and 35. City category C has more customers between the ages of 18 and 45.
- In City Category C, there are slightly more female customers.
- Product 5 and 8 is common among females.

In [ ]:

Recommendations

- Men spent more money than women, So company should focus on retaining the male customers and getting more male customers.
- Product_Category - 1, 5, 8, & 11 have highest purchasing frequency. it means these are the products in these categories are liked more by customers. Company can focus on * selling more of these products or selling more of the products which are purchased less.
- Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
- Customers in the age 18-45 spend more money than the others, So company should focus on acquisition of customers who are in the age 18-45.
- Male customers living in City_Category C spend more money than other male customers living in B or C, Selling more products in the City_Category C will help the company increase the revenue.
- In light of the fact that females spend less than males on average, management needs to focus on their specific needs differently. Adding some additional offers for women can increase their spending on Black Friday.
- Management should come-up with some games in the mall to attract more younger generation will can help them to increase the sale.
- The management should have some offers on kids (0-17 years) in order to increase sales.
- In order to attract more young shoppers, they can offer some games/actvities for the younger generation.

In [ ]: