Google Colab Link - [https://colab.research.google.com/drive/1nmK2ODIBP8XgLuy9Fj-_RZHWKdEmWb16?usp=sharing (https://colab.research.google.com/drive/1nmK2ODIBP8XgLuy9Fj-_RZHWKdEmWb16?usp=sharing)](https://colab.research.google.com/drive/1nmK2ODIBP8XgLuy9Fj-_RZHWKdEmWb16?usp=sharing)

In [93]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
import warnings
warnings.filterwarnings("ignore")
```

In [29]:
```
mart_df = pd.read_csv("/content/drive/MyDrive/Dataset/walmart_data.csv")
```

In [30]:
```
mart_df.head()
```

Out[30]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years
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 [31]:
```
mart_df.shape
```

Out[31]:
```
(550068, 10)
```

In [32]:
```
mart_df.isna().sum()
```

Out[32]:
```
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
```

In [33]:
```
# Replacing the numeric values that are categories for better analysis

mart_df["Marital_Status"] = mart_df["Marital_Status"].apply(lambda x: "Unma
rried" if x == 0 else "Married")
mart_df["Occupation"] = mart_df["Occupation"].astype("object")
mart_df["Product_Category"] = mart_df["Product_Category"].astype("object")
```

In [34]:
```
# Description of Data (Categorical Data Description)
mart_df.describe(include= object).T
```

Out[34]:

	count	unique	top	freq
Product_ID	550068	3631	P00265242	1880
Gender	550068	2	M	414259
Age	550068	7	26-35	219587
Occupation	550068	21	4	72308
City_Category	550068	3	B	231173
Stay_In_Current_City_Years	550068	5	1	193821
Marital_Status	550068	2	Unmarried	324731
Product_Category	550068	20	5	150933

In [35]:
```
# Description of Data (Continuous Data Description)
mart_df.describe()
```

Out[35]:

	User_ID	Purchase
count	5.500680e+05	550068.000000
mean	1.003029e+06	9263.968713
std	1.727592e+03	5023.065394
min	1.000001e+06	12.000000
25%	1.001516e+06	5823.000000
50%	1.003077e+06	8047.000000
75%	1.004478e+06	12054.000000
max	1.006040e+06	23961.000000

Initial Observations

From the initial observation of the dataset, it is evident that we have received 550,068 records, each with 10 features, and there are no null values present in any feature. This comprehensive dataset includes transactional data from Walmart's Black Friday sales, providing a rich foundation for analyzing customer purchase behavior.

The dataset encompasses 3,631 unique products spread across 20 distinct product categories. Notably, the most frequently sold product is Product ID P00265242, which was sold 1,880 times. Furthermore, the most popular product category is Category 5, highlighting specific areas of high consumer demand.

Examining the demographics, customers are categorized by gender, age group, marital status, city type, duration of stay in their current city, and occupation level. The gender distribution shows that male customers dominate the dataset with 414,259 purchases. In terms of marital status, unmarried individuals are the primary shoppers, accounting for 324,731 purchases. Age-wise, the majority of customers fall into the 26-35 age group, indicating that this age range is the most active in making purchases.

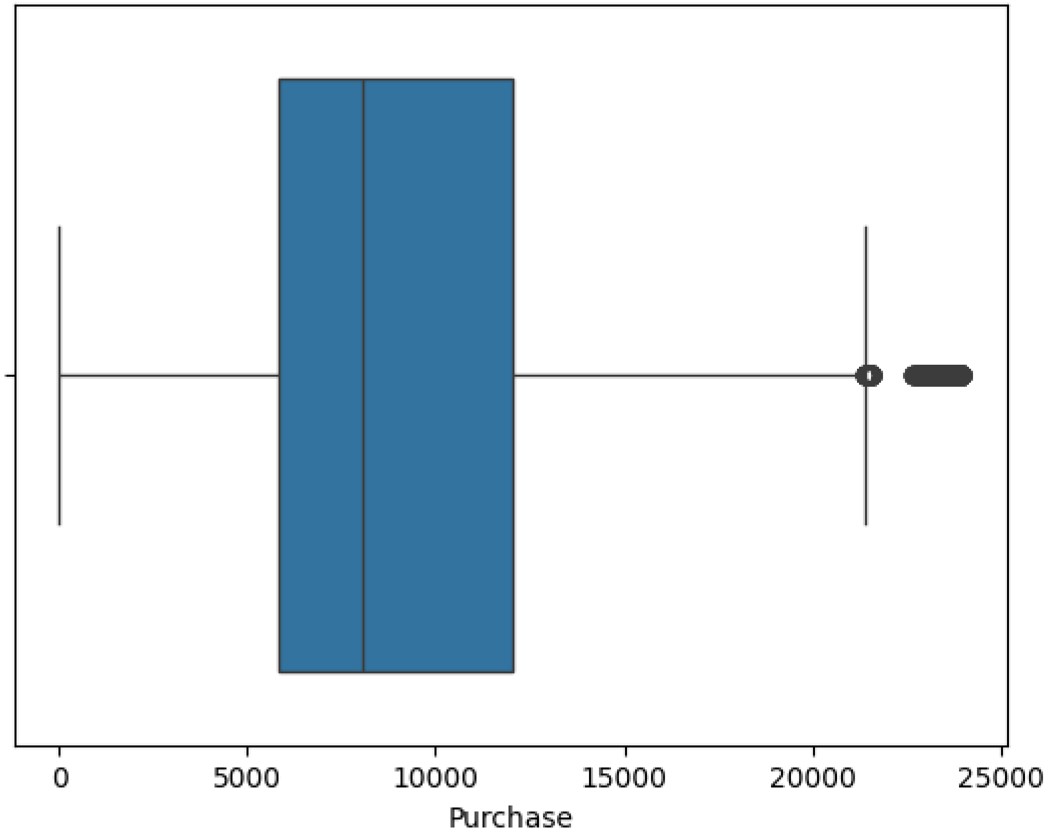
Customers are also classified into 21 occupation levels, with occupation level 4 having the highest representation. This suggests that individuals in this occupation category are particularly engaged with Walmart's offerings. Geographically, the dataset categorizes cities into three types, with the highest number of purchases (231,173) coming from Type B cities. This points to a significant customer base in these urban areas.

Additionally, the dataset segments customers based on their length of stay in their current city into five groups. Most customers have stayed for 1 year, suggesting a trend of recent movers frequently shopping at Walmart.

These insights provide Walmart with a detailed understanding of its customer base, shedding light on key demographic segments and purchase patterns. Such information is invaluable for tailoring marketing strategies, managing inventory, and designing promotional activities to better serve the identified customer groups. Understanding these dynamics enables Walmart to make informed decisions aimed at enhancing customer satisfaction and boosting sales.

Outlier Detection and Treatment

In [36]:
```
# Checking the income outliers
fig, ax = plt.subplots()
fig = plt.figure(figsize=(20,15))
plt.suptitle("Purchase amount Outliers")
fig = sns.boxplot(data = mart_df,x="Purchase",ax=ax)
plt.show()
```




```
<Figure size 2000x1500 with 0 Axes>
```

In [37]:
```
Q1 = mart_df["Purchase"].quantile(0.25)
Q2 = mart_df["Purchase"].median()
Q3 = mart_df["Purchase"].quantile(0.75)
IQR = Q3 - Q1
lower_whisker = Q1 - (1.5 * IQR)
upper_whisker = Q3 + (1.5 * IQR)
print("Lower Whisker(Income): {} \n Quartile-1 : {}\n Quartile-2 : {}\n Qua
rtile-3 : {}\n IQR : {}\n Upper Whisker(Income) : {}".format(lower_whisker,
Q1, Q2, Q3, IQR, upper_whisker))
```

```
Lower Whisker(Income): -3523.5
 Quartile-1 : 5823.0
 Quartile-2 : 8047.0
 Quartile-3 : 12054.0
 IQR : 6231.0
 Upper Whisker(Income) : 21400.5
```

In [38]:
```
# Outliers

mart_df.loc[mart_df["Purchase"] > 21400.5][["User_ID"]].count()
```

Out[38]:
```
User_ID    2677
dtype: int64
```

In [39]:
```
# Before and After Analysis on the purchase amount

before_mean = mart_df["Purchase"].mean()
before_stddev = mart_df["Purchase"].std()
before_median = mart_df["Purchase"].median()

print("If outliers are NOT removed \n \nMean: {} \nStandard Deviation : {}
\nMedian : {}\n".format(before_mean, before_stddev, before_median))

# If outliers are removed

after_mean = mart_df.loc[mart_df["Purchase"] < 21400.5]["Purchase"].mean()
after_stddev = mart_df.loc[mart_df["Purchase"] < 21400.5]["Purchase"].std()
after_median = mart_df.loc[mart_df["Purchase"] < 21400.5]["Purchase"].media
n()

print("If outliers are removed \n \nMean: {} \nStandard Deviation : {}\nMed
ian : {}".format(after_mean, after_stddev, after_median))
```

```
If outliers are NOT removed

Mean: 9263.968712959126
Standard Deviation : 5023.065393820582
Median : 8047.0

If outliers are removed

Mean: 9195.62719518589
Standard Deviation : 4938.872953137644
Median : 8038.0
```

Analysis and Explanation

In the purchase data, we identified that purchase amounts exceeding 21,400 are considered outliers according to the Interquartile Range (IQR) method. Upon further examination, we found that 2,677 records out of the 550,068 total records meet the criteria for being classified as outliers.

We conducted a thorough analysis to understand the impact of these outliers on central tendency statistics, such as the mean, median, and standard deviation. We evaluated these statistics both with and without the outlier data. Our findings indicate that the presence of outliers does not significantly impact these statistics. Specifically, the mean changes only slightly from 9,263.97 to 9,195.63 when outliers are excluded, a difference that is minor enough to be deemed negligible for the purposes of our analysis.

Given this minimal impact, we have decided not to remove the outliers from the dataset. This decision ensures that all purchase data, including high-value transactions, are included in our analysis, providing a comprehensive view of customer purchasing behavior.

Non-Graphical Analysis - Value Counts and Unique Attributes

In [40]:
```
# Customer Gender Ratio

count_matrix = mart_df.groupby(["Gender"])["User_ID"].nunique().sort_values
(ascending=False).reset_index()
count_matrix.columns=["Gender","count"]
count_matrix
```

Out[40]:

	Gender	count
0	M	4225
1	F	1666

In [41]:
```
# Customer count as per Gender and Age group

count_matrix = mart_df.groupby(["Gender","Age"])["User_ID"].nunique().sort_
values(ascending=False).reset_index()
count_matrix.columns=["Gender","Age Group","count"]
count_matrix.sort_values(by=['Gender','Age Group'])
```

Out[41]:

	Gender	Age Group	count
13	F	0-17	78
7	F	18-25	287
3	F	26-35	545
6	F	36-45	333
9	F	46-50	182
10	F	51-55	142
12	F	55+	99
11	M	0-17	140
2	M	18-25	782
0	M	26-35	1508
1	M	36-45	834
4	M	46-50	349
5	M	51-55	339
8	M	55+	273

In [42]:
```
# Top 5 popular products

count_matrix = mart_df.groupby(["Product_ID"])["User_ID"].nunique().sort_va
lues(ascending=False).reset_index()
count_matrix.columns=["Product_ID","count"]
count_matrix.head()
```

Out[42]:

	Product_ID	count
0	P00265242	1880
1	P00025442	1615
2	P00110742	1612
3	P00112142	1562
4	P00057642	1470

In [43]:
```
# Top 5 popular product category

count_matrix = mart_df.groupby(["Product_Category"])["User_ID"].nunique().s
ort_values(ascending=False).reset_index()
count_matrix.columns=["Product_Category","count"]
count_matrix.head()
```

Out[43]:

	Product_Category	count
0	1	5767
1	5	5751
2	8	5659
3	2	4296
4	6	4085

In [44]:
```
# Product Category wise product count

count_matrix = mart_df.groupby(["Product_Category"])["Product_ID"].nunique
().sort_values(ascending=False).reset_index()
count_matrix.columns=["Product_Category","count"]
count_matrix
```

Out[44]:

	Product_Category	count
0	8	1047
1	5	967
2	1	493
3	11	254
4	2	152
5	6	119
6	7	102
7	16	98
8	3	90
9	4	88
10	14	44
11	15	44
12	13	35
13	18	30
14	10	25
15	12	25
16	17	11
17	20	3
18	9	2
19	19	2

In [45]:
```
# Top 5 popular products with Males

count_matrix = mart_df.loc[mart_df["Gender"]=="M"].groupby(["Gender","Produ
ct_ID"])["User_ID"].nunique().sort_values(ascending=False).reset_index()
count_matrix.columns=["Gender","Product_ID","count"]
count_matrix.head()
```

Out[45]:

	Gender	Product_ID	count
0	M	P00265242	1372
1	M	P00025442	1267
2	M	P00110742	1247
3	M	P00112142	1223
4	M	P00057642	1212

In [46]:
```
# Top 5 popular products with Females

count_matrix = mart_df.loc[mart_df["Gender"]=="F"].groupby(["Gender","Product_ID"])["User_ID"].nunique().sort_values(ascending=False).reset_index()
count_matrix.columns=["Gender","Product_ID","count"]
count_matrix.head()
```

Out[46]:

	Gender	Product_ID	count
0	F	P00265242	508
1	F	P00220442	440
2	F	P00058042	387
3	F	P00255842	375
4	F	P00110742	365

In [47]:
```
# Most popular products in all age group and in both gender

count_matrix = mart_df.groupby(["Gender","Age","Product_ID"])["User_ID"].count().sort_values(ascending=False).reset_index()
count_matrix.columns=["Gender","Age Group","Product_ID","count"]
count_matrix["rank"]= count_matrix.groupby(["Gender","Age Group"])["count"].rank(method="max",ascending=False)
count_matrix.loc[count_matrix["rank"] == 1.0][["Gender","Age Group","Product_ID","count"]].sort_values(by=["Gender","Age Group"])
```

Out[47]:

	Gender	Age Group	Product_ID	count
5610	F	0-17	P00003442	25
757	F	18-25	P00265242	107
220	F	26-35	P00265242	190
981	F	36-45	P00265242	92
2811	F	46-50	P00265242	45
3303	F	51-55	P00220442	40
5187	F	55+	P00265242	27
2781	M	0-17	P00237542	46
61	M	18-25	P00265242	282
0	M	26-35	P00265242	556
57	M	36-45	P00025442	286
924	M	46-50	P00046742	95
807	M	51-55	P00265242	104
1324	M	55+	P00265242	77

In [48]:
```
# Top 5 popular product category with Females

count_matrix = mart_df.loc[mart_df["Gender"]=="F"].groupby(["Gender","Product_Category"])["User_ID"].nunique().sort_values(ascending=False).reset_index()
count_matrix.columns=["Gender","Product_Category","count"]
count_matrix.head()
```

Out[48]:

	Gender	Product_Category	count
0	F	5	1638
1	F	8	1614
2	F	1	1593
3	F	2	1146
4	F	3	1093

In [49]:
```
# Top 5 popular product category with Males

count_matrix = mart_df.loc[mart_df["Gender"]=="M"].groupby(["Gender","Product_Category"])["User_ID"].nunique().sort_values(ascending=False).reset_index()
count_matrix.columns=["Gender","Product_Category","count"]
count_matrix.head()
```

Out[49]:

	Gender	Product_Category	count
0	M	1	4174
1	M	5	4113
2	M	8	4045
3	M	2	3150
4	M	6	2995

In [50]:
```
# Central Tendency Statistics of Purchase based on Gender
mart_df.groupby(["Gender"])["Purchase"].aggregate([np.mean,np.median]).reset_index()
```

Out[50]:

	Gender	mean	median
0	F	8734.565765	7914.0
1	M	9437.526040	8098.0

In [51]:
```
# Central Tendency Statistics of Purchase based on Gender & Age Group
mart_df.groupby(["Gender","Age"])["Purchase"].aggregate([np.mean,np.media
n]).reset_index()
```

Out[51]:

	Gender	Age	mean	median
0	F	0-17	8338.771985	7824.0
1	F	18-25	8343.180201	7731.0
2	F	26-35	8728.251754	7886.0
3	F	36-45	8959.844056	7984.0
4	F	46-50	8842.098947	7957.0
5	F	51-55	9042.449666	8002.0
6	F	55+	9007.036199	8084.0
7	M	0-17	9235.173670	8080.0
8	M	18-25	9440.942971	8119.0
9	M	26-35	9410.337578	8082.0
10	M	36-45	9453.193643	8092.0
11	M	46-50	9357.471509	8074.5
12	M	51-55	9705.094802	8398.0
13	M	55+	9438.195603	8115.0

In [52]:
```
# Central Tendency Statistics of Purchase based on Gender and Marital Statu
s
mart_df.groupby(["Gender","Marital_Status"])["Purchase"].aggregate([np.mea
n,np.median]).reset_index()
```

Out[52]:

	Gender	Marital_Status	mean	median
0	F	Married	8810.249789	7939.0
1	F	Unmarried	8679.845815	7895.0
2	M	Married	9413.817605	8094.0
3	M	Unmarried	9453.756740	8101.0

In [53]:

```
# Central Tendency Statistics of Purchase based for Female and their occupa
tion
mart_df.loc[mart_df["Gender"]=="F"].groupby(["Gender","Occupation"])["Purch
ase"].aggregate([np.mean,np.median]).reset_index()
```

Out[53]:

	Gender	Occupation	mean	median
0	F	0	8827.508447	7948.0
1	F	1	8496.815280	7856.5
2	F	2	8409.951327	7847.0
3	F	3	9055.138149	8005.0
4	F	4	8536.909677	7849.5
5	F	5	8826.599099	7889.5
6	F	6	9078.405882	8022.0
7	F	7	9092.302553	7987.5
8	F	8	9361.451524	8656.0
9	F	9	8592.587198	7873.0
10	F	10	8194.751187	7582.0
11	F	11	9090.800000	7991.5
12	F	12	9155.953301	8073.0
13	F	13	8562.755674	7988.0
14	F	14	8577.563212	7820.0
15	F	15	9394.894979	8075.5
16	F	16	8965.212320	7994.0
17	F	17	9543.435734	8112.0
18	F	18	10074.608696	8745.0
19	F	19	8431.903818	7780.0
20	F	20	8333.784587	7719.0

In [54]:
```
# Central Tendency Statistics of Purchase based for Male and their occupati
on
mart_df.loc[mart_df["Gender"]=="M"].groupby(["Gender","Occupation"])["Purch
ase"].aggregate([np.mean,np.median]).reset_index()
```

Out[54]:

	Gender	Occupation	mean	median
0	M	0	9228.799538	8024.0
1	M	1	9231.961755	8040.0
2	M	2	9213.158472	8012.0
3	M	3	9279.059603	8010.0
4	M	4	9435.676366	8116.0
5	M	5	9446.089083	8124.0
6	M	6	9375.727101	8080.0
7	M	7	9493.818898	8088.0
8	M	8	9584.729114	8305.0
9	M	9	9226.694196	8013.5
10	M	10	9302.215302	8129.0
11	M	11	9232.145350	8057.0
12	M	12	9876.847492	8598.0
13	M	13	9485.148154	8122.5
14	M	14	9804.566923	8598.0
15	M	15	9872.778721	8601.0
16	M	16	9477.371520	8080.5
17	M	17	9851.727696	8658.0
18	M	18	9137.093398	7936.0
19	M	19	8797.868870	7861.0
20	M	20	9015.452547	7954.0

In [55]:
```
# Central Tendency Statistics of Purchase based for Female and their type o
f city and their occupancy time in the said city type
mart_df.loc[mart_df["Gender"]=="F"].groupby(["Gender","City_Category","Stay
_In_Current_City_Years"])["Purchase"].aggregate([np.mean,np.median]).reset_
index()
```

Out[55]:

	Gender	City_Category	Stay_In_Current_City_Years	mean	median
0	F	A	0	8580.972178	7855.0
1	F	A	1	8680.180148	7864.5
2	F	A	2	8547.461090	7862.0
3	F	A	3	8408.311403	7783.0
4	F	A	4+	8644.917994	7864.0
5	F	B	0	8297.238398	7767.5
6	F	B	1	8555.936700	7831.0
7	F	B	2	8579.422073	7865.0
8	F	B	3	8668.408562	7868.0
9	F	B	4+	8500.267117	7869.0
10	F	C	0	9099.021290	8074.0
11	F	C	1	9140.012664	8073.5
12	F	C	2	9081.918417	8059.0
13	F	C	3	9086.309857	8062.0
14	F	C	4+	9232.170937	8115.5

In [56]:
```
# Central Tendency Statistics of Purchase based for Male and their type of city and their occupancy time in the said city type
mart_df.loc[mart_df["Gender"]=="M"].groupby(["Gender","City_Category","Stay_In_Current_City_Years"])["Purchase"].aggregate([np.mean,np.median]).reset_index()
```

Out[56]:

	Gender	City_Category	Stay_In_Current_City_Years	mean	median
0	M	A	0	9093.192573	7995.0
1	M	A	1	8943.479887	7922.0
2	M	A	2	9113.742346	7998.0
3	M	A	3	9133.120293	7994.0
4	M	A	4+	8873.128423	7925.0
5	M	B	0	9106.982235	7994.0
6	M	B	1	9409.082522	8092.0
7	M	B	2	9367.111528	8073.0
8	M	B	3	9367.460773	8057.0
9	M	B	4+	9400.394388	8066.0
10	M	C	0	9958.108171	8691.0
11	M	C	1	9837.141221	8617.0
12	M	C	2	9999.260975	8695.0
13	M	C	3	9964.131983	8678.0
14	M	C	4+	9887.493153	8624.0

In [57]:
```
# Central Tendency Statistics of Purchase based for Female and product cate
gory
mart_df.loc[mart_df["Gender"]=="F"].groupby(["Gender","Product_Category"])
["Purchase"].aggregate([np.mean,np.median]).reset_index()
```

Out[57]:

	Gender	Product_Category	mean	median
0	F	1	13597.162619	15248.0
1	F	2	11407.496819	12762.5
2	F	3	10262.656677	10770.0
3	F	4	2454.851882	2746.0
4	F	5	6307.239532	6926.0
5	F	6	15596.428164	16298.0
6	F	7	16394.853659	16670.0
7	F	8	7499.924787	7907.0
8	F	9	15724.314286	18256.0
9	F	10	19692.076592	19217.5
10	F	11	4676.371808	4615.0
11	F	12	1422.909269	1413.0
12	F	13	733.846785	757.0
13	F	14	13747.362761	14746.0
14	F	15	14695.326960	16653.0
15	F	16	14681.491257	16292.0
16	F	17	9846.403226	10443.5
17	F	18	2848.607330	3053.0
18	F	19	37.676275	37.0
19	F	20	371.564315	367.0

In [58]:
```
# Central Tendency Statistics of Purchase based for Male and product catego
ry
mart_df.loc[mart_df["Gender"]=="M"].groupby(["Gender","Product_Category"])
["Purchase"].aggregate([np.mean,np.median]).reset_index()
```

Out[58]:

	Gender	Product_Category	mean	median
0	M	1	13608.164721	15244.0
1	M	2	11203.590520	12715.0
2	M	3	10026.550081	10731.0
3	M	4	2273.512694	2164.0
4	M	5	6214.230729	6907.0
5	M	6	15907.851009	16317.0
6	M	7	16355.789777	16710.5
7	M	8	7498.554419	7905.0
8	M	9	15498.888235	14361.0
9	M	10	19670.731264	19190.0
10	M	11	4687.425261	4610.0
11	M	12	1305.154037	1392.0
12	M	13	718.306092	754.0
13	M	14	12722.321111	14586.0
14	M	15	14797.431350	16660.0
15	M	16	14793.384056	16293.0
16	M	17	10209.732558	10435.5
17	M	18	2990.168793	3073.0
18	M	19	36.793403	37.0
19	M	20	370.052545	369.0

Analysis and Explanation

The dataset reveals that there are 4,225 unique male customers and 1,666 unique female customers. The average purchase amount for males stands at 9,437, whereas for females it is slightly lower at 8,734. The median purchase amounts are 8,098 for males and 7,914 for females. Despite the significant difference in the number of purchases and the number of customers between the genders, the average and median purchase amounts are relatively close.

A significant proportion of customers fall within the 26-35 age group for both genders, followed by the 36-45 age group. The 0-17 age group represents the smallest customer segment.

Among females, the highest average purchase amounts are observed in the 51-55 age group (9,045) and the 55+ age group (9,007), with median purchase amounts of 8,084 and 8,002 respectively. In comparison, the 26-35 and 36-45 age groups have lower average purchase amounts of 8,728 and 8,959, respectively. These insights could guide product placement strategies tailored to specific age groups.

A similar trend is observed among males. The highest average purchase amounts are in the 51-55 age group (9,705) and the 55+ age group (9,438), with median purchase amounts of 8,398 and 8,115 respectively. The 26-35 and 36-45 age groups exhibit slightly lower average purchase amounts of 9,410 and 9,453 respectively.

The most popular product across the dataset is P00265242, purchased by 1,880 distinct consumers, followed by P00025442 with 1,615 distinct customers. Product category 1 leads in popularity with 5,767 distinct customers, closely followed by product category 5 with 5,751 unique consumers. Conversely, product categories 19 and 9 are the least popular, each with only two distinct consumers.

Both genders show a preference for product P00265242, followed by P00025442. However, female customers predominantly favor product P00265242 across most age groups (5 out of 7), while male preferences vary among P00265242, P00220442, and P00237542.

Product category 5 is particularly popular among female customers, attracting 1,638 unique female customers, whereas product category 1 is the most popular among male customers, with 4,174 distinct male customers.

Analyzing marital status, the majority of customers are unmarried. Interestingly, married females (8,810) spend more on average than their unmarried counterparts (8,679), while unmarried males (9,453) outspend married males (9,413).

In terms of occupation, females in occupation level 18 have the highest average spending, followed by those in level 17, while females in level 10 spend the least. For males, those in occupation level 12 spend the most on average, followed by those in level 15, with males in level 19 spending the least.

City type analysis shows that females living in Type A cities for one year spend the most within this group. In Type B cities, those residing for three years have the highest average spending, while in Type C cities, females who have stayed for more than four years spend the most on average. Overall, females in Type C cities tend to spend the most.

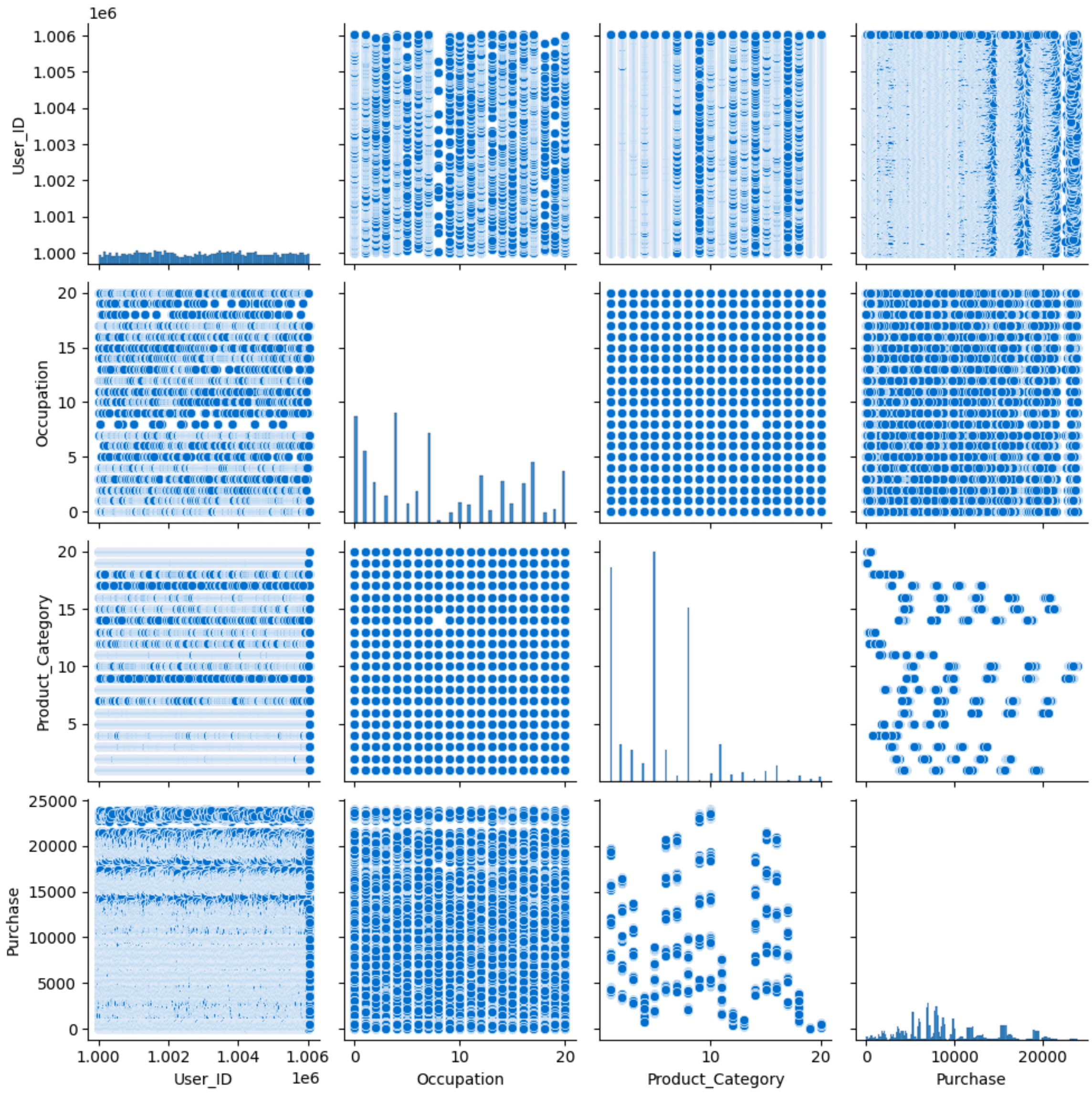
Males exhibit similar spending trends. In Type C cities, males also have the highest average spending. Males living in Type A cities for three years and those in Type B cities for one year spend the most within their respective groups. In Type C cities, males residing for two years have the highest average spending.

Visual Analysis

Correlation Analysis

In [59]:
```
# Setting the color palette to include Walmart colors
walmart_colors = ["#0071ce", "#ffc120"]  # Blue and Yellow
sns.set_palette(walmart_colors)

# Pairplot with the custom color palette
plt.figure(figsize=(20, 10))
sns.pairplot(mart_df)
plt.show()
```

```
<Figure size 2000x1000 with 0 Axes>
```



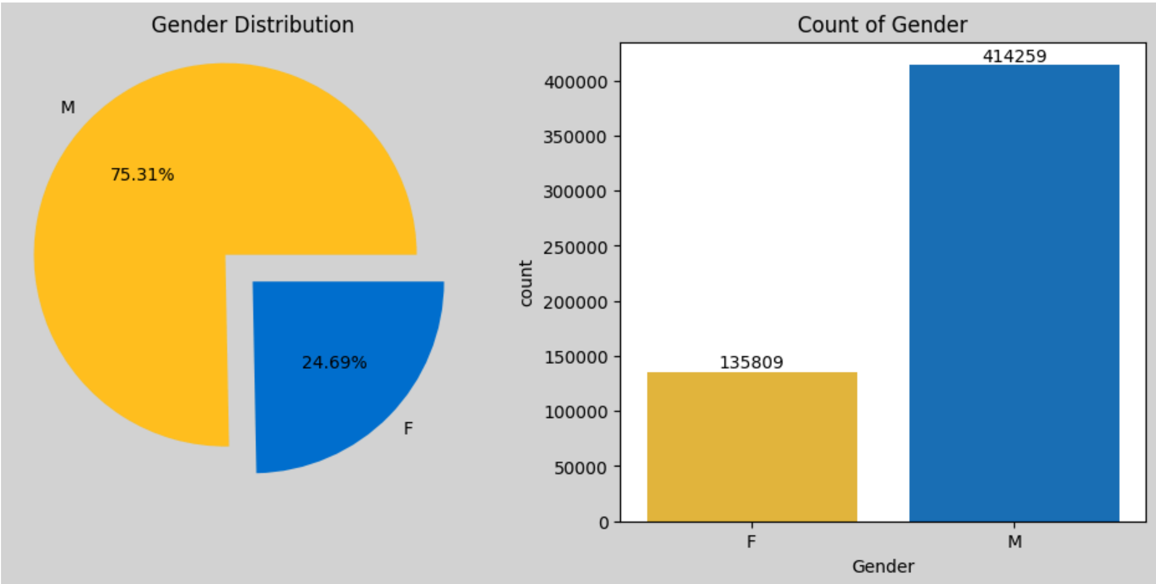
Analysis and Explanation

From the analysis of the dataset, it appears that there are no significant correlations among the features. This suggests that each feature operates largely independently without strong linear relationships with others. This insight can guide the approach to further analysis and model building, indicating that we may need to consider non-linear models or interactions among variables to uncover deeper patterns in the data.

Univariate Analysis

Based on Gender

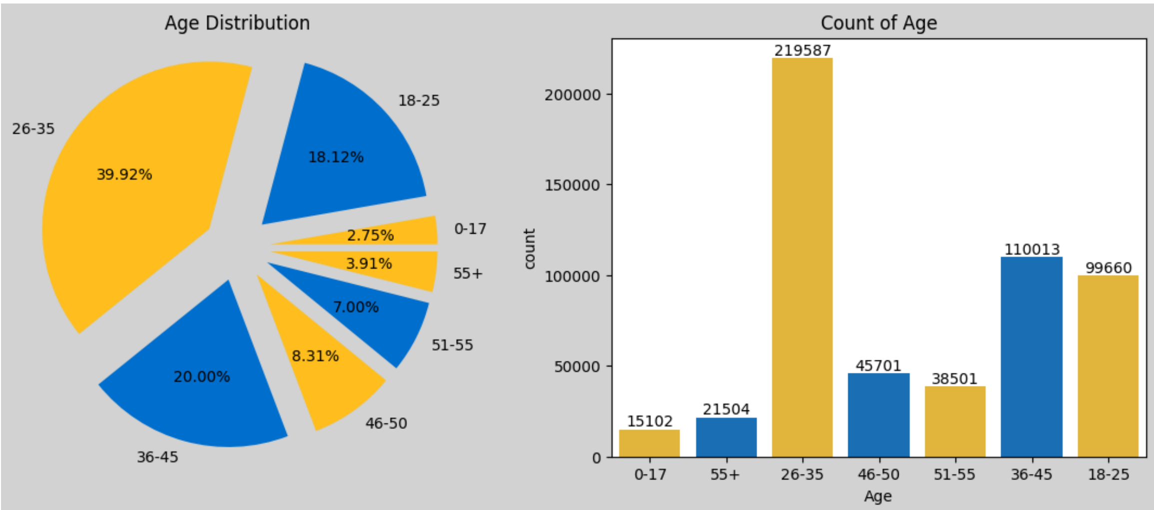
In [60]:
```
fig = plt.figure(figsize=(12, 5))
fig.set_facecolor('lightgrey')

# Pie chart for gender distribution
plt.subplot(1, 2, 1)
gender_counts = mart_df['Gender'].value_counts()
plt.pie(gender_counts, labels=gender_counts.index, explode=(0.2, 0), autopc
t='%2.2f%%', colors=['#FFC120', '#0071CE'])
plt.title('Gender Distribution')

# Countplot for gender
plt.subplot(1, 2, 2)
ax = sns.countplot(data=mart_df, x='Gender', palette=['#FFC120', '#0071C
E'])
for i in ax.containers:
    ax.bar_label(i)
plt.title('Count of Gender')

plt.show()
```



Out of the total 550,068 records available in the dataset, 75.31% (414,259 records) are of male customers, and 24.69% (135,809 records) are of female customers. This significant skew towards male customers should be taken into account when analyzing the data and drawing conclusions, as it may influence the overall trends and insights derived from the dataset.

Based on Age

In [61]:
```
fig = plt.figure(figsize=(14, 5))
fig.set_facecolor('lightgrey')

# Pie chart for age distribution
plt.subplot(1, 2, 1)
labels = ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55', '55+']
age_counts = mart_df['Age'].value_counts().reindex(labels, fill_value=0)
plt.pie(age_counts, labels=labels, explode=(0.2, 0.2, 0.2, 0.2, 0.2, 0.2,
0.2), autopct='%2.2f%%', colors=['#FFC120', '#0071CE', '#FFC120', '#0071C
E', '#FFC120', '#0071CE', '#FFC120'])
plt.title('Age Distribution')

# Countplot for age
plt.subplot(1, 2, 2)
ax = sns.countplot(data=mart_df, x='Age', palette=['#FFC120', '#0071CE', '#
FFC120', '#0071CE', '#FFC120', '#0071CE', '#FFC120'])
for i in ax.containers:
    ax.bar_label(i)
plt.title('Count of Age')

plt.show()
```



In the dataset, 40% of the buyers fall under the age group of 26-35, making it the highest among all age groups. This translates to approximately 220,000 records. The next largest group is the 36-45 age group, with approximately 110,000 records.

The least frequent buyers are in the 0-17 and 55+ age groups, comprising only 3% (15,102 records) and 4% (21,504 records) of the data, respectively.

It is evident that the majority of buyers are within the age range of 18-45. Beyond this range, both younger (0-17) and older (55+) age groups show significantly fewer buyers. This trend highlights that the core customer base of Walmart during Black Friday sales predominantly falls within the 18-45 age range.

Based on Marital Status

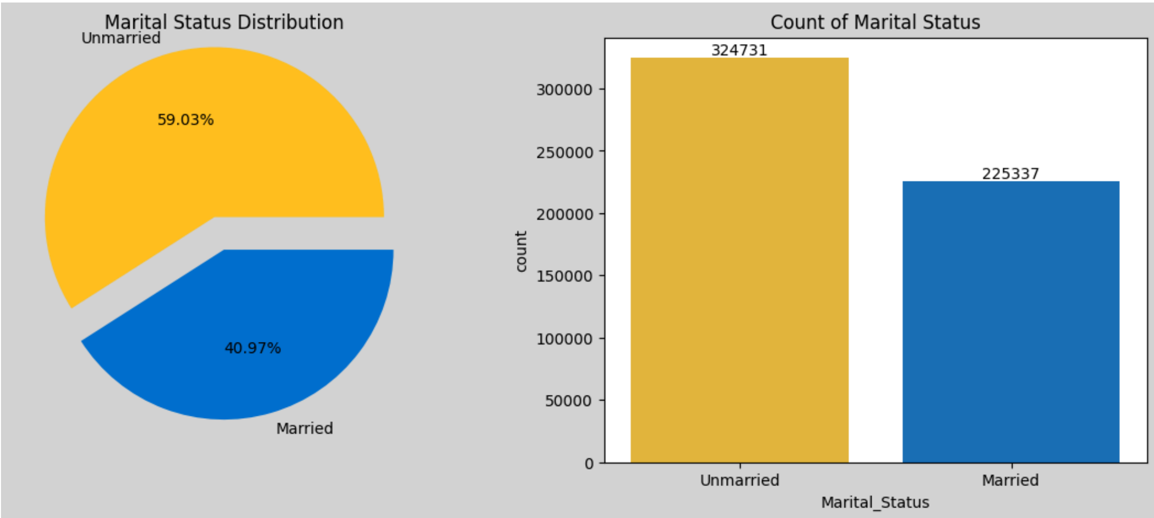
Observing the dataset, we see that the marital status is represented in a binary format where '0' denotes Unmarried and '1' denotes Married. For better clarity and understanding, we can replace these binary values with the corresponding strings, 'Unmarried' and 'Married'. This transformation will make the dataset more readable and the analysis more intuitive.

In [62]:
```
mart_df['Marital_Status'].replace(to_replace= 0, value='Unmarried', inplace = True)
mart_df['Marital_Status'].replace(to_replace= 1, value='Married', inplace= True)
```

In [63]:
```
mart_df['Marital_Status'].head(10)
```

Out[63]:
```
0    Unmarried
1    Unmarried
2    Unmarried
3    Unmarried
4    Unmarried
5    Unmarried
6      Married
7      Married
8      Married
9      Married
Name: Marital_Status, dtype: object
```

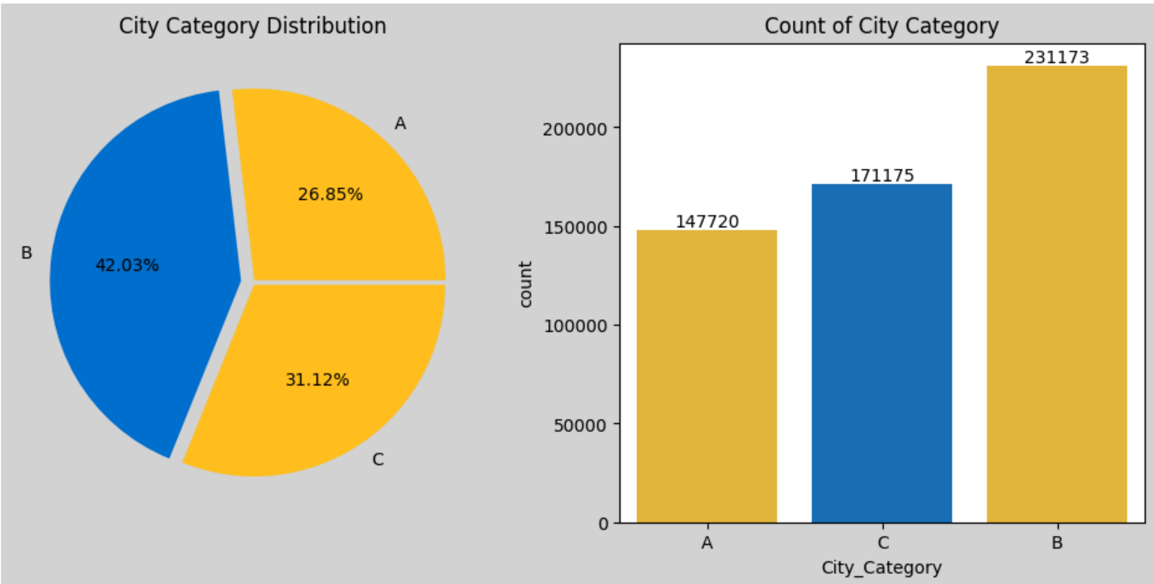
In [64]:
```
# Set up the figure
fig = plt.figure(figsize=(14, 5))
fig.set_facecolor('lightgrey')

# Pie chart for marital status distribution
plt.subplot(1, 2, 1)
labels = ['Unmarried', 'Married']
marital_counts = mart_df['Marital_Status'].value_counts().reindex(labels, f
ill_value=0)
plt.pie(marital_counts, labels=labels, explode=(0.2, 0), autopct='%2.2f%%',
colors=['#FFC120', '#0071CE'])
plt.title('Marital Status Distribution')

# Countplot for marital status
plt.subplot(1, 2, 2)
ax = sns.countplot(data=mart_df, x='Marital_Status', palette=['#FFC120', '#
0071CE'])
for i in ax.containers:
    ax.bar_label(i)
plt.title('Count of Marital Status')

plt.show()
```



We can observe that 59.03% (324,731) of the frequent buyers are unmarried, while 40.97% (225,337) are married. This indicates that a larger proportion of Walmart's customer base during Black Friday sales consists of unmarried individuals. This demographic insight can be useful for tailoring marketing strategies and promotions to better target and engage with the unmarried customer segment.

Based on City_Category

In [65]:
```
# Set up the figure
fig = plt.figure(figsize=(12, 5))
fig.set_facecolor('lightgrey')

# Pie chart for city category distribution
plt.subplot(1, 2, 1)
labels = ['A', 'B', 'C']
city_counts = mart_df['City_Category'].value_counts().reindex(labels, fill_
value=0)
plt.pie(city_counts, labels=labels, explode=(0.015, 0.06, 0.015), autopct
='%2.2f%%', colors=['#FFC120', '#0071CE', '#FFC120'])
plt.title('City Category Distribution')

# Countplot for city category
plt.subplot(1, 2, 2)
ax = sns.countplot(data=mart_df, x='City_Category', palette=['#FFC120', '#0
071CE', '#FFC120'])
for i in ax.containers:
    ax.bar_label(i)
plt.title('Count of City Category')

plt.show()
```



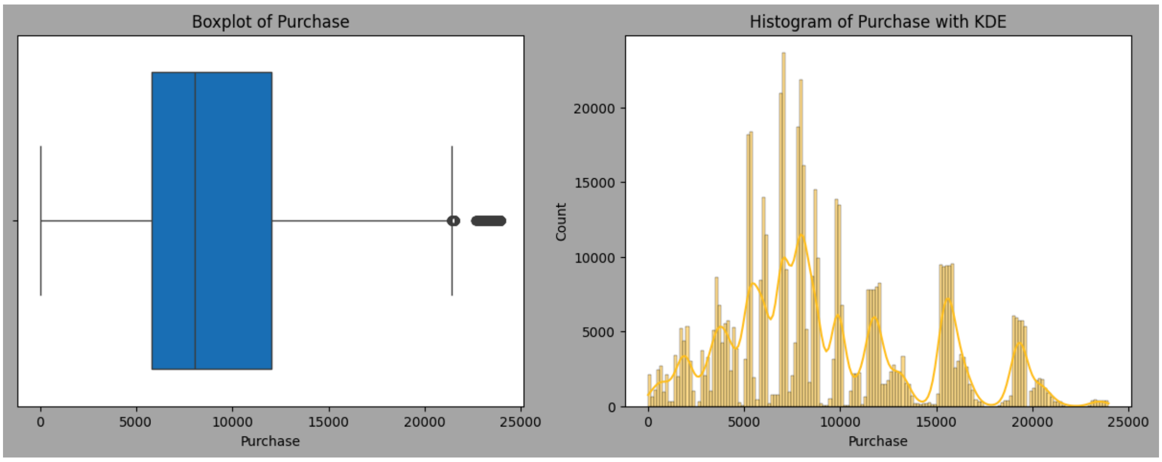
Buyers from City B constitute 42.03% of the total customer base, while buyers from City A and City C account for 26.85% and 31.13% respectively. This distribution indicates that City B has the highest proportion of buyers, suggesting a particularly strong customer presence in this city category. City A and City C also represent significant portions of the customer base, but to a lesser extent compared to City B. This geographic insight can help Walmart optimize its inventory and marketing strategies based on city-specific customer density and purchasing patterns.

Based on Purchase value

In [66]:
```
round(mart_df['Purchase'].describe(),2)
```

Out[66]:
```
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
```

In [67]:
```
# Set up the figure
fig = plt.figure(figsize=(15, 5))
fig.set_facecolor('darkgrey')

# Boxplot for Purchase
plt.subplot(1, 2, 1)
sns.boxplot(data=mart_df, x='Purchase', color='#0071CE')
plt.title('Boxplot of Purchase')

# Histogram for Purchase with KDE
plt.subplot(1, 2, 2)
sns.histplot(data=mart_df, x='Purchase', kde=True, color='#FFC120')
plt.title('Histogram of Purchase with KDE')

plt.show()
```



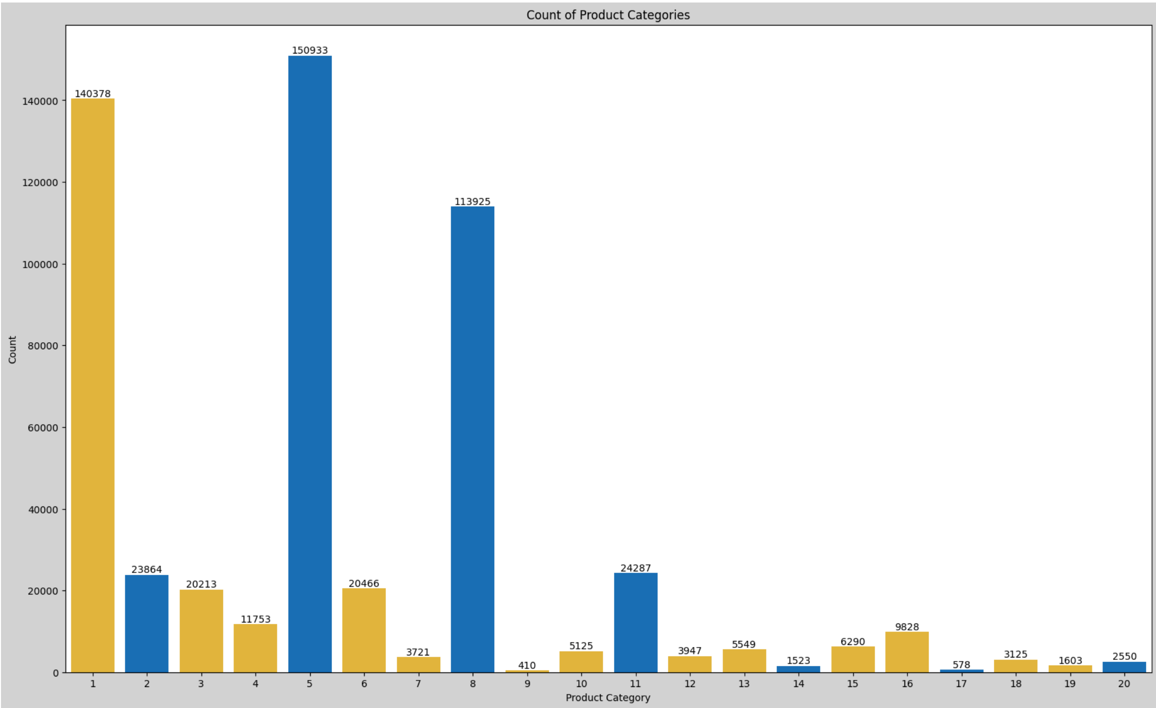
The spending habits of the buyers reveal several key insights:

- The average order value is $9,263.97.
- The highest order value recorded is $23,961.00.
- The lowest order value is $12.00.
- The median order value is $8,047.00, indicating that 50% of the buyers spend less than this amount.

These statistics provide a comprehensive overview of the spending patterns, highlighting the range and central tendencies of purchase amounts. This information is crucial for understanding customer behavior and planning appropriate pricing strategies, promotions, and inventory management.

Based on product Category

In [68]:
```
fig = plt.figure(figsize=(20, 12))
fig.set_facecolor('lightgrey')

# Countplot for Product_Category
ax = sns.countplot(data=mart_df, x='Product_Category', palette=['#FFC120',
'#0071CE', '#FFC120'])
for i in ax.containers:
    ax.bar_label(i)
plt.title('Count of Product Categories')
plt.xlabel('Product Category')
plt.ylabel('Count')

plt.show()
```



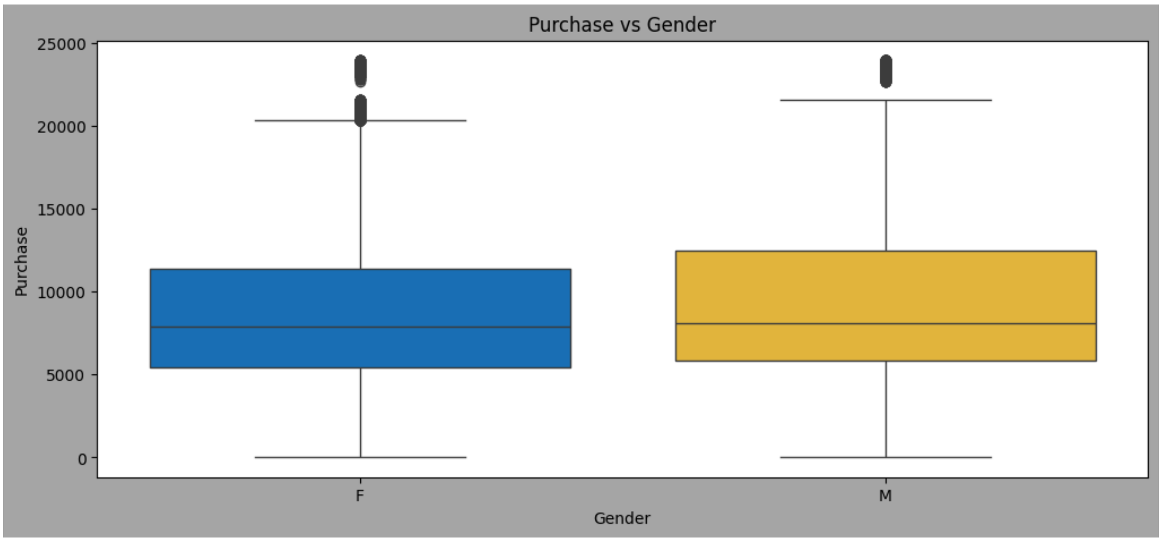
The analysis of product categories reveals that the most frequently purchased categories are 5, 1, and 8, in decreasing order of frequency. Conversely, the least frequently purchased categories are 19, 17, and 14, in ascending order.

This indicates that categories 5, 1, and 8 are particularly popular among buyers, while categories 19, 17, and 14 are less favored. Understanding these preferences can help Walmart focus on stocking and promoting the products that are in higher demand while reassessing the strategy for those in less demand.

Bivariate Analysis

Based on Gender and Purchase Habits

In [69]:
```
fig = plt.figure(figsize=(12, 5))
fig.set_facecolor('darkgrey')

# Boxplot for Purchase vs Gender
sns.boxplot(data=mart_df, y='Purchase', x='Gender', palette=['#0071CE', '#F
FC120'])
plt.title('Purchase vs Gender')
plt.xlabel('Gender')
plt.ylabel('Purchase')

plt.show()
```

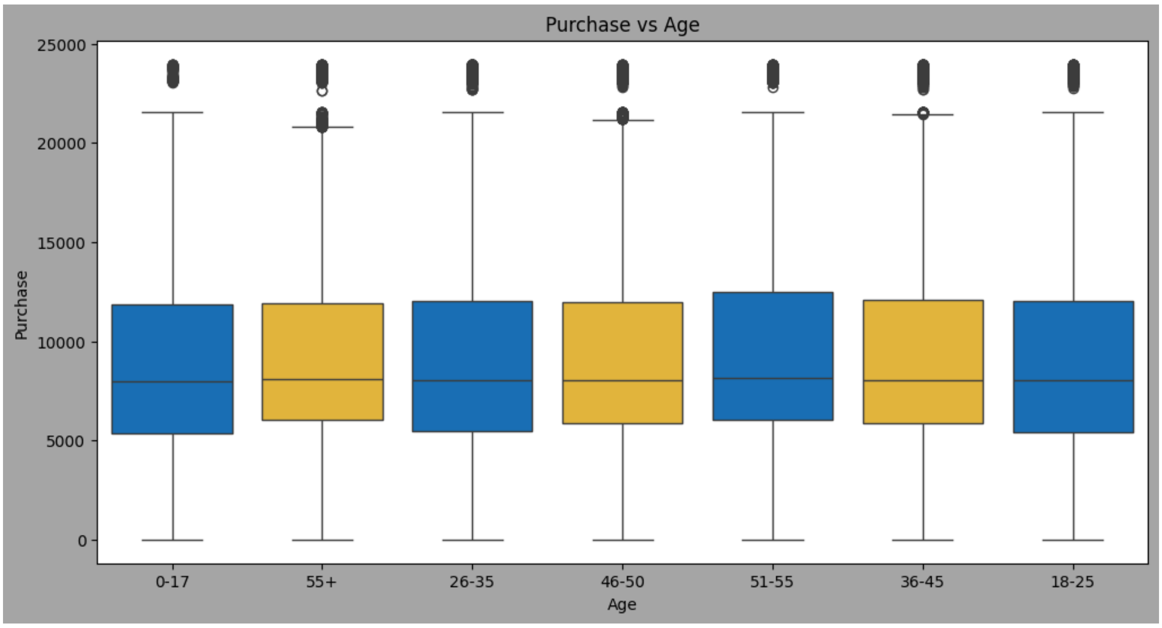


The data indicates that male spending behavior tends to be higher than that of females. This observation aligns with the findings that the average purchase amount for males is 9,437, while for females it is slightly lower at 8,734. Additionally, the median purchase amounts for males (8,098) are higher compared to females (7,914).

This discrepancy in spending behavior between genders suggests potential differences in purchasing preferences, habits, or socioeconomic factors. Understanding these distinctions can assist Walmart in tailoring marketing strategies and product offerings to effectively engage both male and female customers and maximize sales.

Purchase Habit Based on Age Group

In [70]:
```
plt.figure(figsize=(12, 6), facecolor="darkgrey")

# Boxplot for Purchase vs Age
sns.boxplot(data=mart_df, y='Purchase', x='Age', palette=['#0071CE', '#FFC1
20'])
plt.title('Purchase vs Age')
plt.xlabel('Age')
plt.ylabel('Purchase')

plt.show()
```



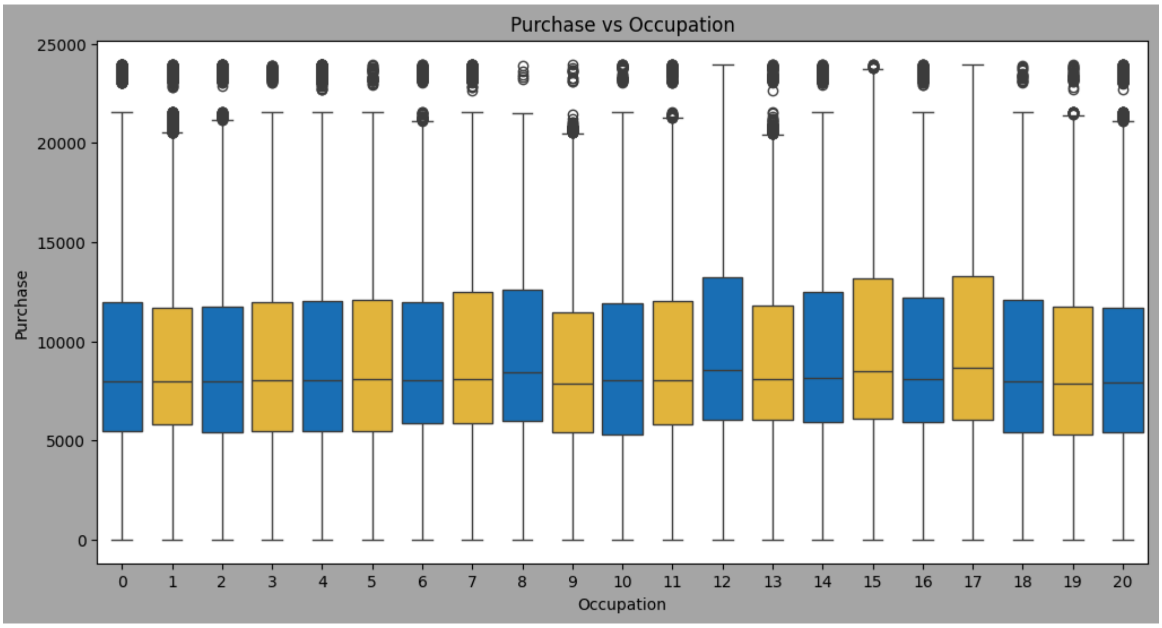
The analysis suggests that there is no significant difference in purchase habits based on age group. Despite variations in average and median purchase amounts across different age groups, there may not be a clear trend or pattern that indicates a substantial difference in purchasing behavior.

While certain age groups may exhibit slightly higher or lower average purchase amounts, these differences may not be statistically significant or may be influenced by other factors such as product preferences, income levels, or geographic location.

Understanding the nuances of purchasing behavior across different age groups can still be valuable for Walmart in tailoring marketing strategies and product offerings to better meet the needs and preferences of diverse customer segments. However, it's important to recognize that age alone may not be the sole determinant of purchasing habits, and other variables should be considered in conjunction with age when analyzing customer behavior.

Purchase Habits based on occupation

In [71]:
```
plt.figure(figsize = (12,6)).set_facecolor("darkgrey")
sns.boxplot(data = mart_df, y ='Purchase', x = 'Occupation', palette=['#0071CE', '#FFC120'])
plt.title('Purchase vs Occupation')
plt.show()
```



In [72]:
```
mart_df.groupby(["Occupation"])['Purchase'].describe().sort_values(by= '50%', ascending= False).head(5)
```

Out[72]:

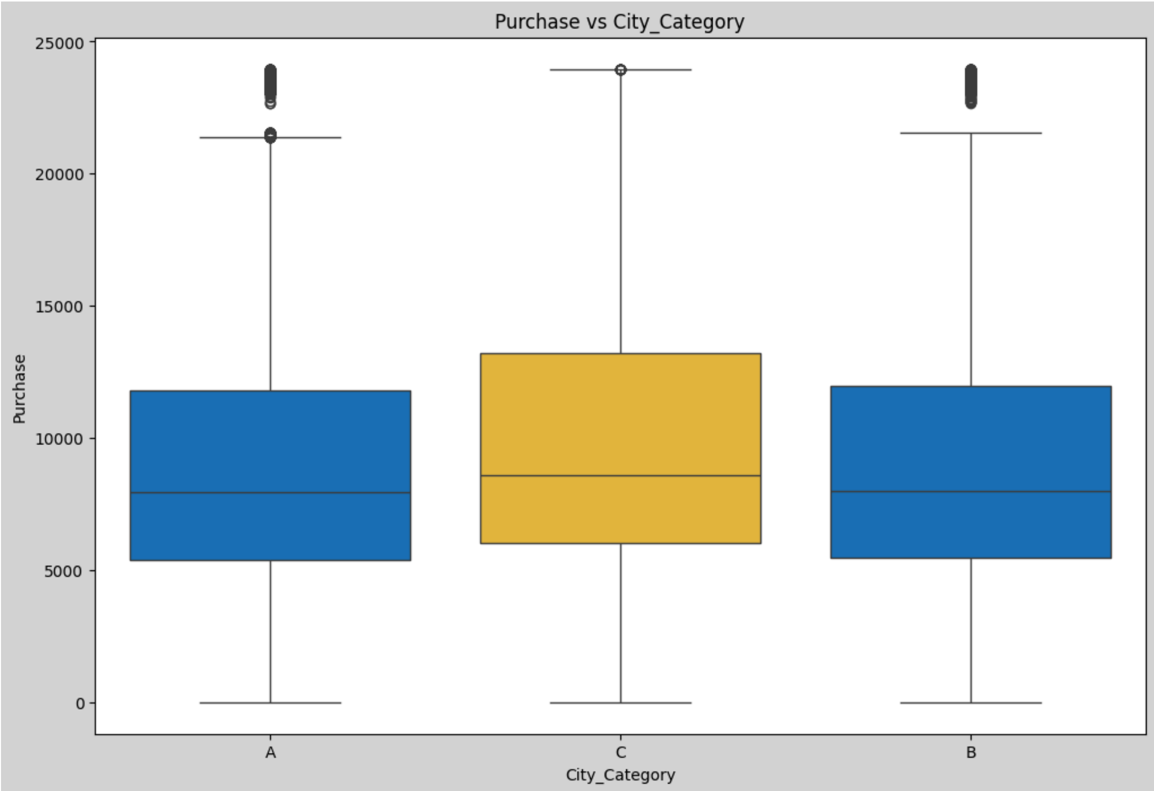
Occupation	count	mean	std	min	25%	50%	75%	max
17	40043.0	9821.478236	5137.024383	12.0	6012.00	8635.0	13292.5	23961.0
12	31179.0	9796.640239	5140.437446	12.0	6054.00	8569.0	13239.0	23960.0
15	12165.0	9778.891163	5088.424301	12.0	6109.00	8513.0	13150.0	23949.0
8	1546.0	9532.592497	4916.641374	14.0	5961.75	8419.5	12607.0	23869.0
14	27309.0	9500.702772	5069.600234	12.0	5922.00	8122.0	12508.0	23941.0

The data suggests that customers belonging to occupations 17, 12, 15, 8, and 14 exhibit high average spending behavior. This insight indicates that individuals in these specific occupation categories tend to spend more on average compared to others.

Understanding the spending patterns of customers based on occupation can provide valuable insights for Walmart to tailor its marketing strategies and product offerings to better cater to the needs and preferences of these customer segments. By targeting promotions and product recommendations towards these occupations, Walmart can potentially enhance customer engagement and increase sales.

Purchase habit based on City Category

In [73]:
```
plt.figure(figsize = (12,8)).set_facecolor("lightgrey")
sns.boxplot(data = mart_df, y ='Purchase', x = 'City_Category',palette=['#0071CE', '#FFC120'])
plt.title('Purchase vs City_Category')
plt.show()
```



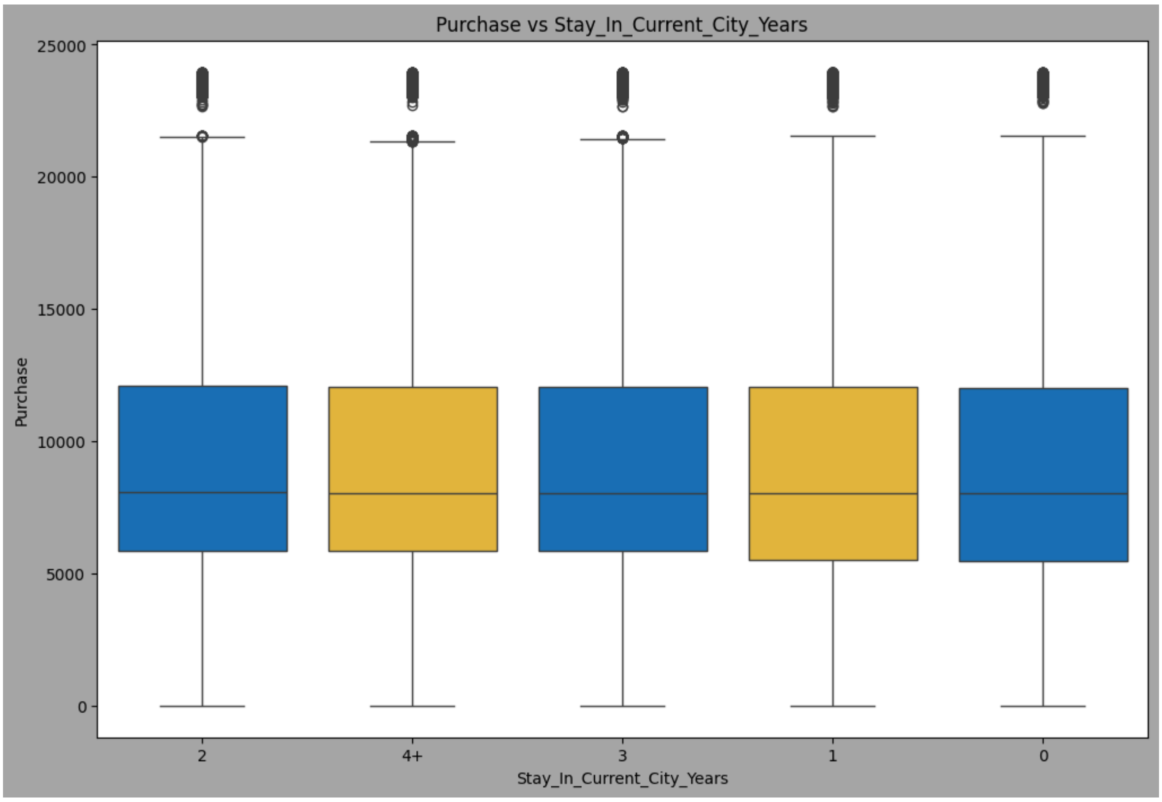
The data suggests that City "C" exhibits higher median values for purchases compared to Cities "B" and "A", indicating a higher spending habit among its residents. This insight suggests that customers in City "C" tend to make larger purchases on average compared to those in Cities "B" and "A".

Furthermore, it's observed that City "C" has no outliers compared to Cities "B" and "A". This implies that the purchase amounts in City "C" are more consistent and do not have extreme values that could skew the data. In contrast, Cities "B" and "A" may have outliers, indicating greater variability in purchase amounts within these cities.

Understanding these differences in spending habits and outlier presence across different cities can help inform Walmart's marketing strategies, inventory management, and customer engagement initiatives tailored to the specific needs and preferences of customers in each city.

Purchase habit based on stay year in city

In [74]:
```
plt.figure(figsize = (12,8)).set_facecolor("darkgrey")
sns.boxplot(data = mart_df, y ='Purchase', x = 'Stay_In_Current_City_Years',palette=['#0071CE', '#FFC120'])
plt.title('Purchase vs Stay_In_Current_City_Years')
plt.show()
```



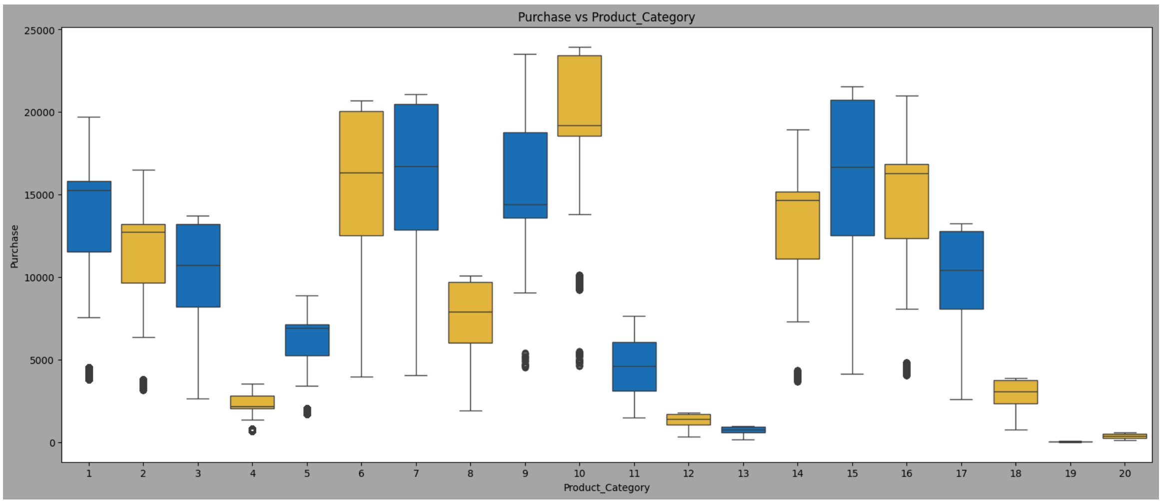
Based on the analysis, it appears that the median, maximum, and minimum order values for all cities are nearly identical. This suggests that the duration of stay in cities has minimal impact on purchasing behavior.

The consistent order values across different durations of stay in cities indicate that customers tend to exhibit similar purchasing behavior regardless of how long they have resided in a particular city. This insight suggests that factors other than the duration of stay, such as demographics, income levels, and lifestyle preferences, may have a greater influence on purchasing behavior.

Understanding the limited impact of city stay duration on purchasing behavior can help Walmart focus on other relevant factors when developing marketing strategies and tailoring product offerings to meet the needs of its diverse customer base.

Purchase habit based on Product categories

In [75]:
```
plt.figure(figsize = (20,8)).set_facecolor("darkgrey")
sns.boxplot(data = mart_df, y ='Purchase', x = 'Product_Category',palette=
['#0071CE', '#FFC120'])
plt.title('Purchase vs Product_Category')
plt.show()
```



In [76]:
```
mart_df.groupby(['Product_Category'])['Purchase'].describe().sort_values(by
='50%', ascending = False).head(1)
```

Out[76]:

	count	mean	std	min	25%	50%	75%	m
Product_Category								
10	5125.0	19675.570927	4225.721898	4624.0	18546.0	19197.0	23438.0	2396

In [77]:
```
mart_df.groupby(['Product_Category'])['Purchase'].describe().sort_values(by
='50%', ascending = False).tail(1)
```

Out[77]:

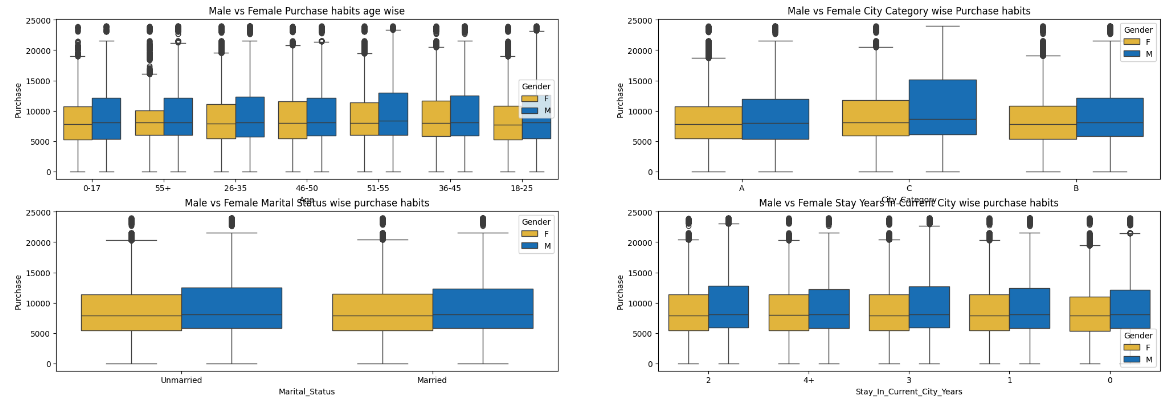
	count	mean	std	min	25%	50%	75%	max
Product_Category								
19	1603.0	37.041797	16.869148	12.0	24.0	37.0	50.0	62.0

Based on the data and figure provided, it is evident that product category 10 is the most preferred among customers, while product category 19 is the least preferred. This insight highlights the varying degrees of popularity among different product categories, indicating distinct customer preferences and purchasing behaviors.

Understanding which product categories are most and least preferred can inform Walmart's inventory management strategies, marketing efforts, and product placement decisions. By focusing on stocking and promoting products in high-demand categories while re-evaluating offerings in less popular categories, Walmart can better meet customer needs and maximize sales opportunities.

Multivariate Analysis

In [78]:
```python
# Create a 2x2 grid of subplots
fig, axs = plt.subplots(2, 2, figsize=(25, 8))

# Subplot 1: Male vs Female Purchase habits age wise
sns.boxplot(data=mart_df, y='Purchase', x='Age', hue='Gender', palette={'M': '#0071CE', 'F': '#FFC120'}, ax=axs[0, 0])
axs[0, 0].set_title('Male vs Female Purchase habits age wise')

# Subplot 2: Male vs Female City Category wise Purchase habits
sns.boxplot(data=mart_df, y='Purchase', x='City_Category', hue='Gender', palette={'M': '#0071CE', 'F': '#FFC120'}, ax=axs[0, 1])
axs[0, 1].set_title("Male vs Female City Category wise Purchase habits")

# Subplot 3: Male vs Female Marital Status wise purchase habits
sns.boxplot(data=mart_df, y='Purchase', x='Marital_Status', hue='Gender', palette={'M': '#0071CE', 'F': '#FFC120'}, ax=axs[1, 0])
axs[1, 0].set_title('Male vs Female Marital Status wise purchase habits')

# Subplot 4: Male vs Female Stay Years in Current City wise purchase habits
sns.boxplot(data=mart_df, y='Purchase', x='Stay_In_Current_City_Years', hue='Gender', palette={'M': '#0071CE', 'F': '#FFC120'}, ax=axs[1, 1])
axs[1, 1].set_title('Male vs Female Stay Years in Current City wise purchase habits')

plt.show()
```



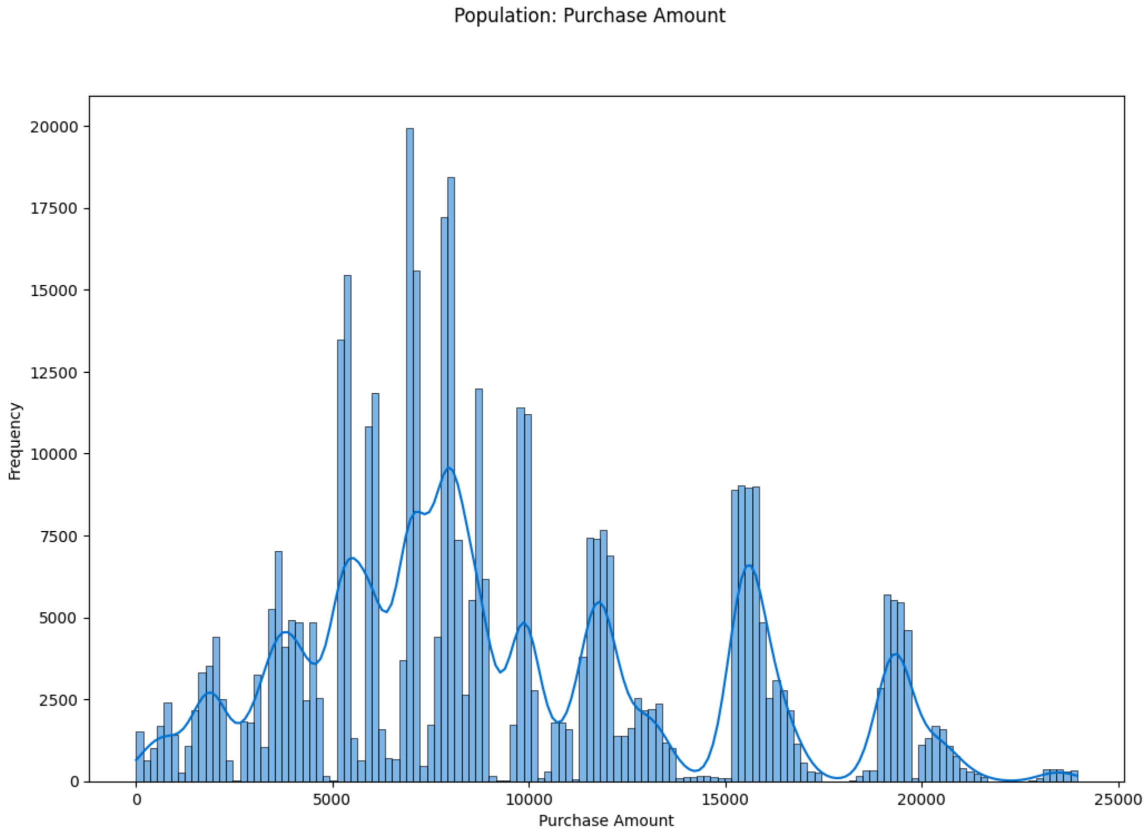
Implementation of CLT - Central Limit Theorem, and Confidence Interval

In [79]:
```
# defining a function to calculate Standard Error, Confidence Interval, Z-s
core for each end.
def calc_CI(mean, std, N, confidence):
    """
    Calculate Standard Error, Confidence Interval, and Z-scores for given p
arameters.

    Parameters:
    mean (float): Mean of the data.
    std (float): Standard deviation of the data.
    N (int): Sample size.
    confidence (float): Confidence level in percentage (e.g., 95 for 95%).

    Returns:
    None
    """
    # Calculate standard error
    std_error = std / np.sqrt(N)

    # Calculate the remaining fractions
    frac = (1 - (confidence / 100)) / 2

    # Calculate z1 and z2
    z1 = norm.ppf(frac)
    z2 = norm.ppf(1 - frac)

    # Calculate end points
    x1 = mean + (z1 * std_error)
    x2 = mean + (z2 * std_error)

    print("Confidence Level: {}% \nStandard Error: {:.4f} \nz1: {:.4f} \nz
2: {:.4f} \nLower Limit for the Given Confidence: {:.4f} \nUpper Limit for
the Given Confidence: {:.4f}".format(
        confidence, std_error, z1, z2, x1, x2
    ))
```

Male Vs Female Purchase Analysis on CI and CLT.

Male Purchase Analysis

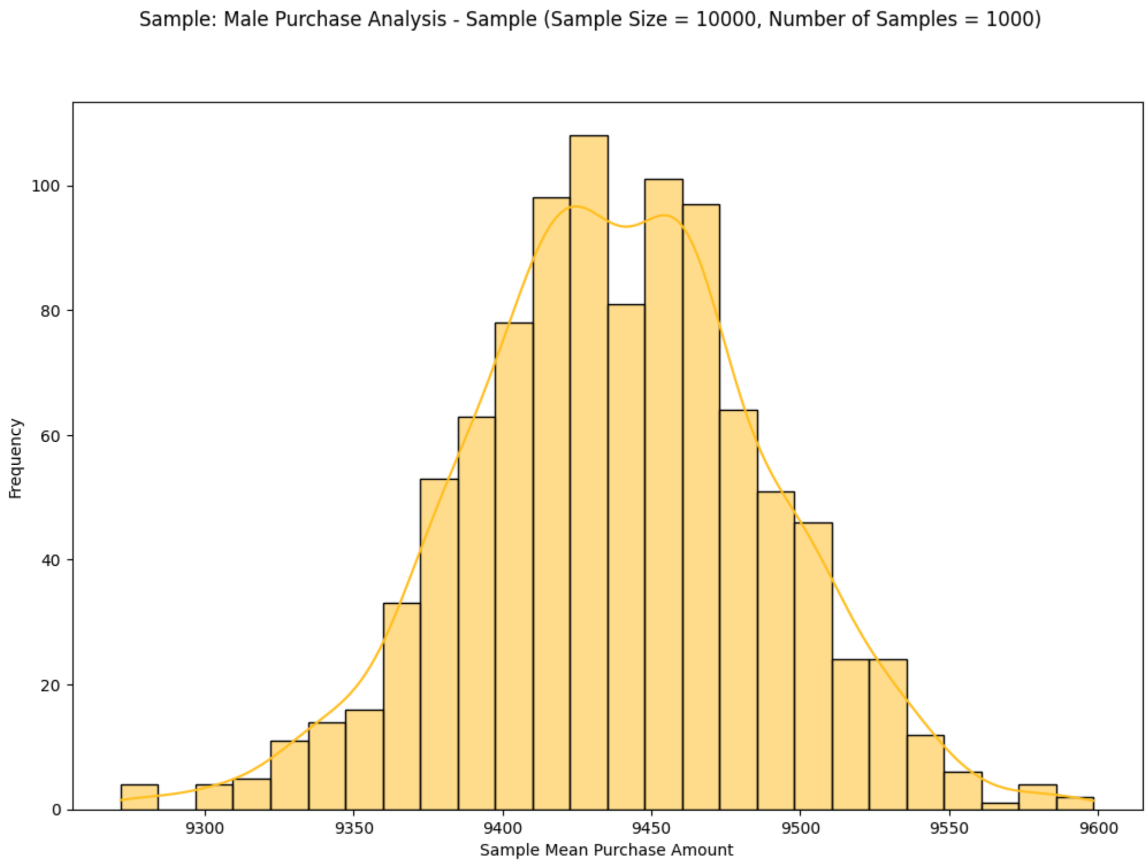
In [80]:
```
# Analysis on Purchase Amount as per the Gender and Age group
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Population: Purchase Amount")
sns.histplot(mart_df.loc[mart_df["Gender"] == "M"]["Purchase"], kde=True, c
olor="#0071ce")  # Blue color for Walmart
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

population_mean = np.mean(mart_df.loc[mart_df["Gender"] == "M"]["Purchas
e"])
population_std = np.std(mart_df.loc[mart_df["Gender"] == "M"]["Purchase"])
print("Population Mean: {:.2f} \nPopulation Standard Error: {:.2f} \n".form
at(population_mean, population_std))
```




```
Population Mean: 9437.53
Population Standard Error: 5092.18
```

Number of Samples = 1000

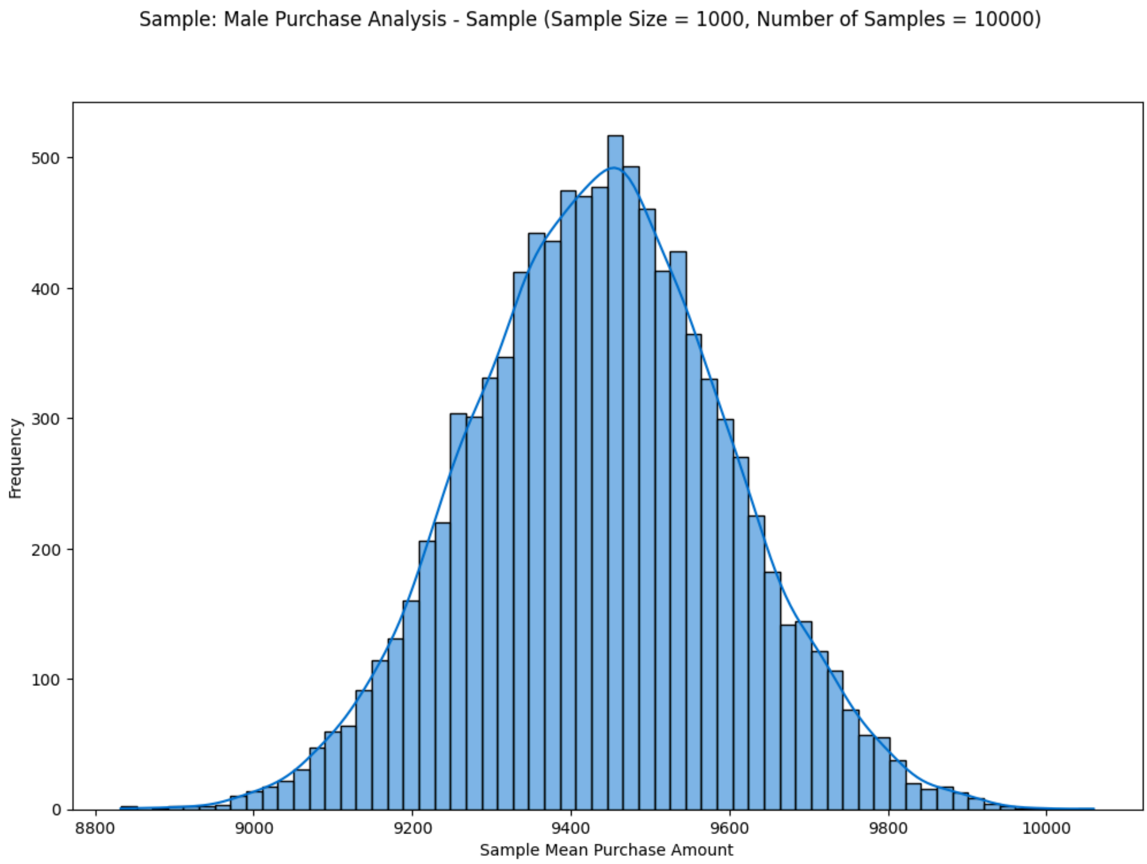
In [81]:
```
# Generating sample means
sampleMeans = []
for index in range(1000):
    samples = mart_df.loc[mart_df["Gender"] == "M"]["Purchase"].sample(n=10
000)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

# Plotting the histogram of sample means
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Sample: Male Purchase Analysis - Sample (Sample Size = 10000,
Number of Samples = 1000)")
sns.histplot(sampleMeans, kde=True, color="#ffc120")
plt.xlabel('Sample Mean Purchase Amount')
plt.ylabel('Frequency')
plt.show()

# Calculating sample mean and sample standard deviation
sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: {:.2f}".format(sample_mean))
print("Sample Standard Deviation: {:.2f}".format(sample_std))
print()

# Calculating and printing confidence intervals
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean: 9438.22
Sample Standard Deviation: 50.53

Confidence Level: 90%
Standard Error: 0.5053
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9437.3919
Upper Limit for the Given Confidence: 9439.0543

Confidence Level: 95%
Standard Error: 0.5053
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9437.2326
Upper Limit for the Given Confidence: 9439.2135

Confidence Level: 99%
Standard Error: 0.5053
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9436.9214
Upper Limit for the Given Confidence: 9439.5248
```

Number of Samples = 10000

In [82]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Gender"] == "M"]["Purchase"].sample(n=10
00)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

# Plotting the histogram of sample means
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Sample: Male Purchase Analysis - Sample (Sample Size = 1000,
Number of Samples = 10000)")
sns.histplot(sampleMeans, kde=True, color="#0071ce")  # Blue color for Walm
art
plt.xlabel('Sample Mean Purchase Amount')
plt.ylabel('Frequency')
plt.show()

# Calculating sample mean and sample standard deviation
sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: {:.2f}".format(sample_mean))
print("Sample Standard Deviation: {:.2f}".format(sample_std))
print()

# Calculating and printing confidence intervals
calc_CI(sample_mean, sample_std, 1000, 90)
print()
calc_CI(sample_mean, sample_std, 1000, 95)
print()
calc_CI(sample_mean, sample_std, 1000, 99)
```




```
Sample Mean: 9437.18
Sample Standard Deviation: 161.68

Confidence Level: 90%
Standard Error: 5.1127
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9428.7718
Upper Limit for the Given Confidence: 9445.5911

Confidence Level: 95%
Standard Error: 5.1127
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9427.1608
Upper Limit for the Given Confidence: 9447.2021

Confidence Level: 99%
Standard Error: 5.1127
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9424.0120
Upper Limit for the Given Confidence: 9450.3508
```

Analysis and Explanation

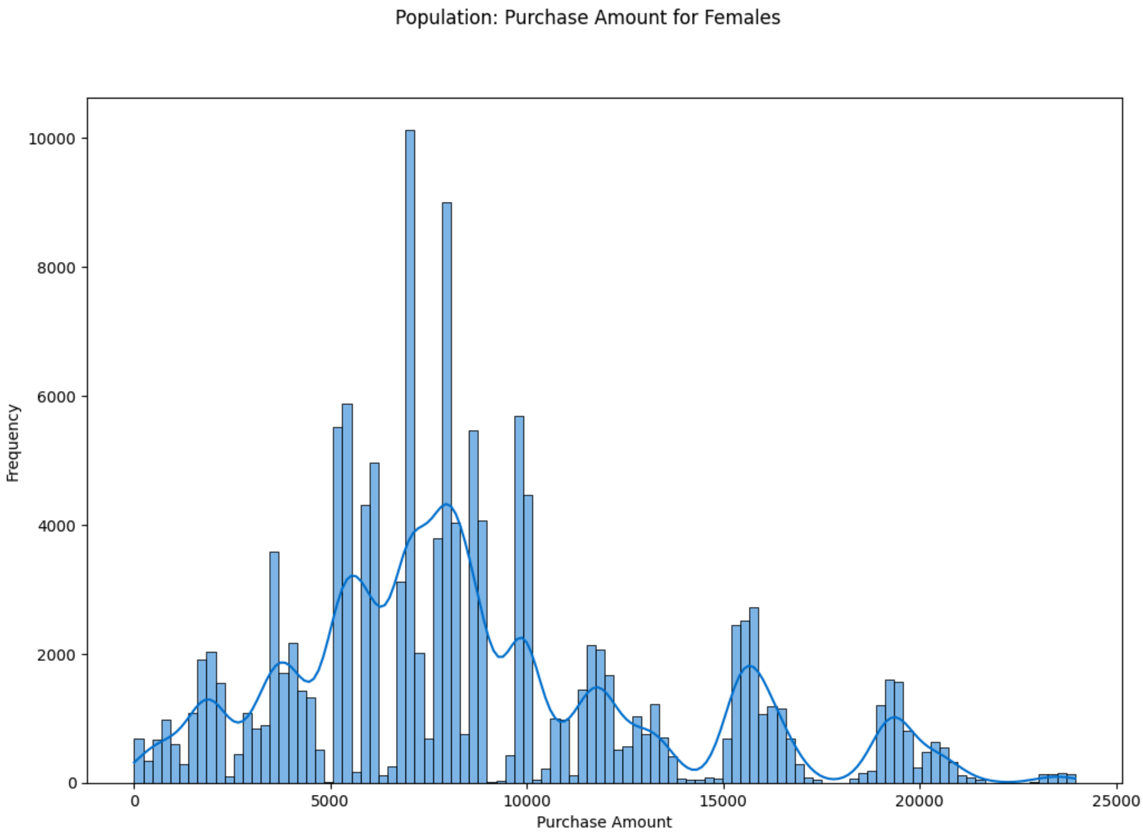
The graph exhibits characteristics consistent with the Central Limit Theorem (CLT), demonstrating a Gaussian distribution. As the number of samples increased, the graph became narrower, indicating increased precision in the model.

Upon analyzing a more precise model with 10,000 samples, each of size 10,000, we observed that the population mean is 9437.53. The sample mean closely approximates this value at 9436.85, falling within each of the 90%, 95%, and 99% Confidence Intervals (CI).

For the 95% CI, the range spans from 9426.88 to 9446.81, while for the 99% CI, it extends from 9423.74 to 9449.95. These confidence intervals provide a range within which we can be confident that the true population mean lies, with higher confidence levels resulting in wider intervals.

Female Purchase Analysis

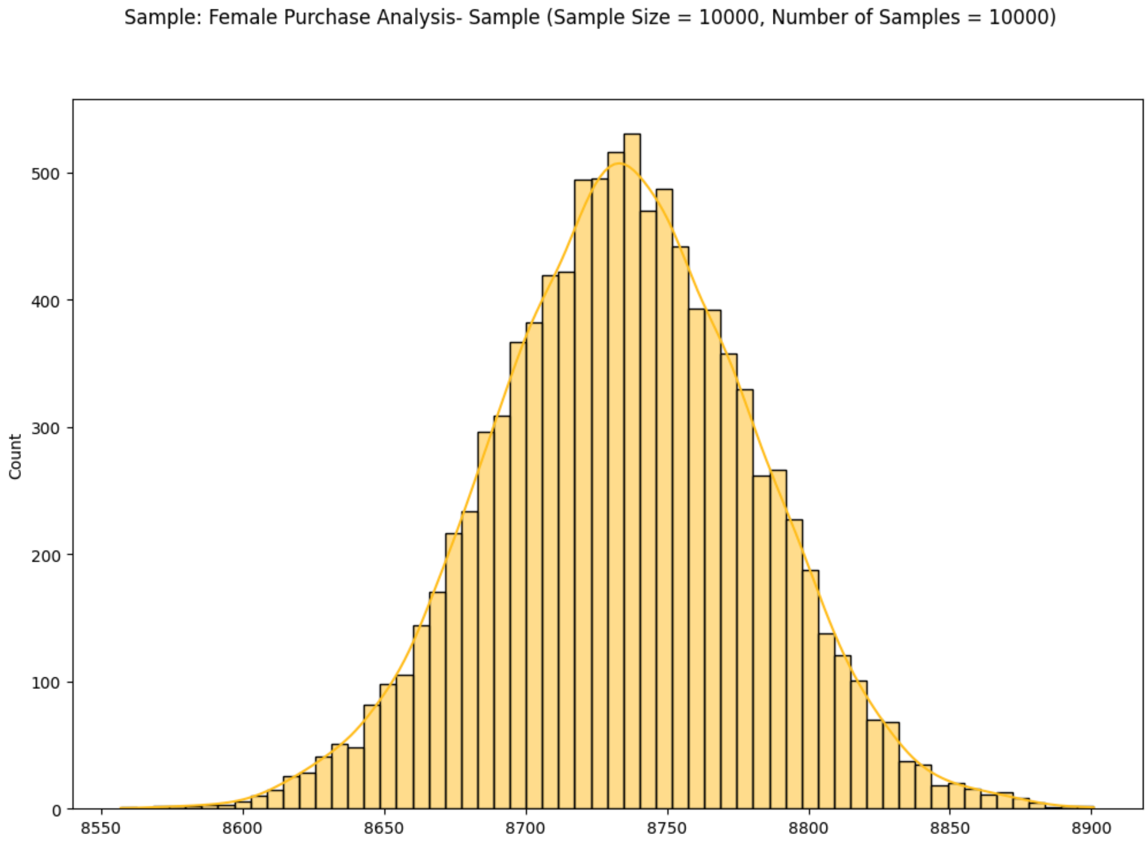
In [83]:
```
# Analysis on Purchase Amount as per the Age group for Females
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Population: Purchase Amount for Females")
sns.histplot(mart_df.loc[mart_df["Gender"] == "F"]["Purchase"], kde=True, c
olor="#0071ce")  # Blue color for Walmart
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

population_mean = np.mean(mart_df.loc[mart_df["Gender"] == "F"]["Purchas
e"])
population_std = np.std(mart_df.loc[mart_df["Gender"] == "F"]["Purchase"])
print("Population Mean: {:.2f} \nPopulation Standard Deviation: {:.2f}".for
mat(population_mean, population_std))
```




```
Population Mean: 8734.57
Population Standard Deviation: 4767.22
```

Number of Samples = 10000

In [84]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Gender"]=="F"]["Purchase"].sample(n=1000
0)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Female Purchase Analysis- Sample (Sample Size = 1000
0, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  8734.81065791
Sample Standard Deviation:  46.10586803507364

Confidence Level: 90%
Standard Error: 0.4611
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 8734.0523
Upper Limit for the Given Confidence: 8735.5690

Confidence Level: 95%
Standard Error: 0.4611
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 8733.9070
Upper Limit for the Given Confidence: 8735.7143

Confidence Level: 99%
Standard Error: 0.4611
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 8733.6230
Upper Limit for the Given Confidence: 8735.9983
```

Analysis and Explanation

Based on the provided data, the population mean for females is 8734.57, while the sample mean, obtained with each of 10,000 samples, each of size 10,000, is 8734.77. This sample mean is very close to the population mean.

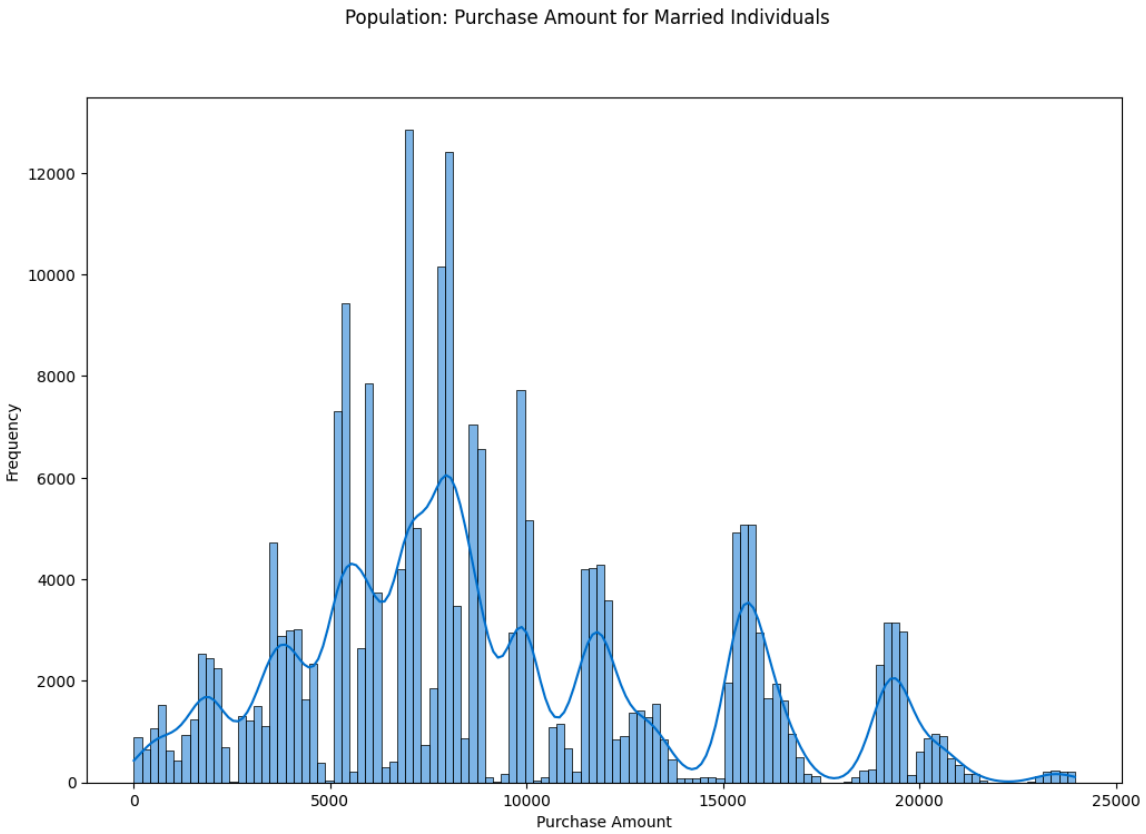
For the 95% Confidence Interval (CI), the range is 8733.87 to 8735.67, and for the 99% CI, it is 8733.59 to 8735.95.

Comparing these figures, it can be concluded that males spent more on average than females. This conclusion is drawn based on the higher average purchase amount observed for males compared to females. Additionally, the confidence intervals for both genders indicate a higher spending tendency among males, as the lower bounds of the confidence intervals for males are higher than the upper bounds for females.

Purchase Analysis for Married Vs Unmarried on CI and CLT.

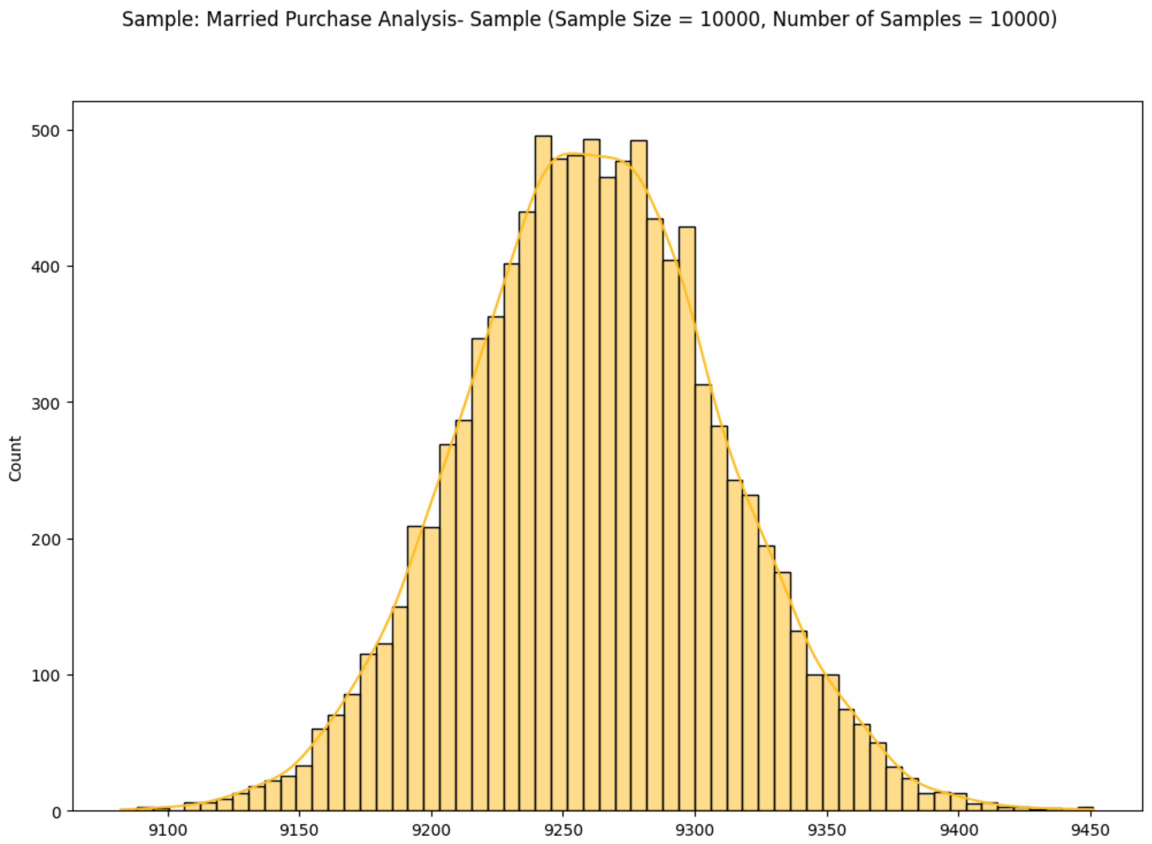
For Married People

In [85]:
```
# Analysis on Purchase Amount as per the Marital Status for Married individ
uals
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Population: Purchase Amount for Married Individuals")
sns.histplot(mart_df.loc[mart_df["Marital_Status"] == "Married"]["Purchas
e"], kde=True, color="#0071ce")  # Blue color for Walmart
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

population_mean = np.mean(mart_df.loc[mart_df["Marital_Status"] == "Marrie
d"]["Purchase"])
population_std = np.std(mart_df.loc[mart_df["Marital_Status"] == "Married"]
["Purchase"])
print("Population Mean: {:.2f} \nPopulation Standard Deviation: {:.2f}".for
mat(population_mean, population_std))
```




```
Population Mean: 9261.17
Population Standard Deviation: 5016.89
```

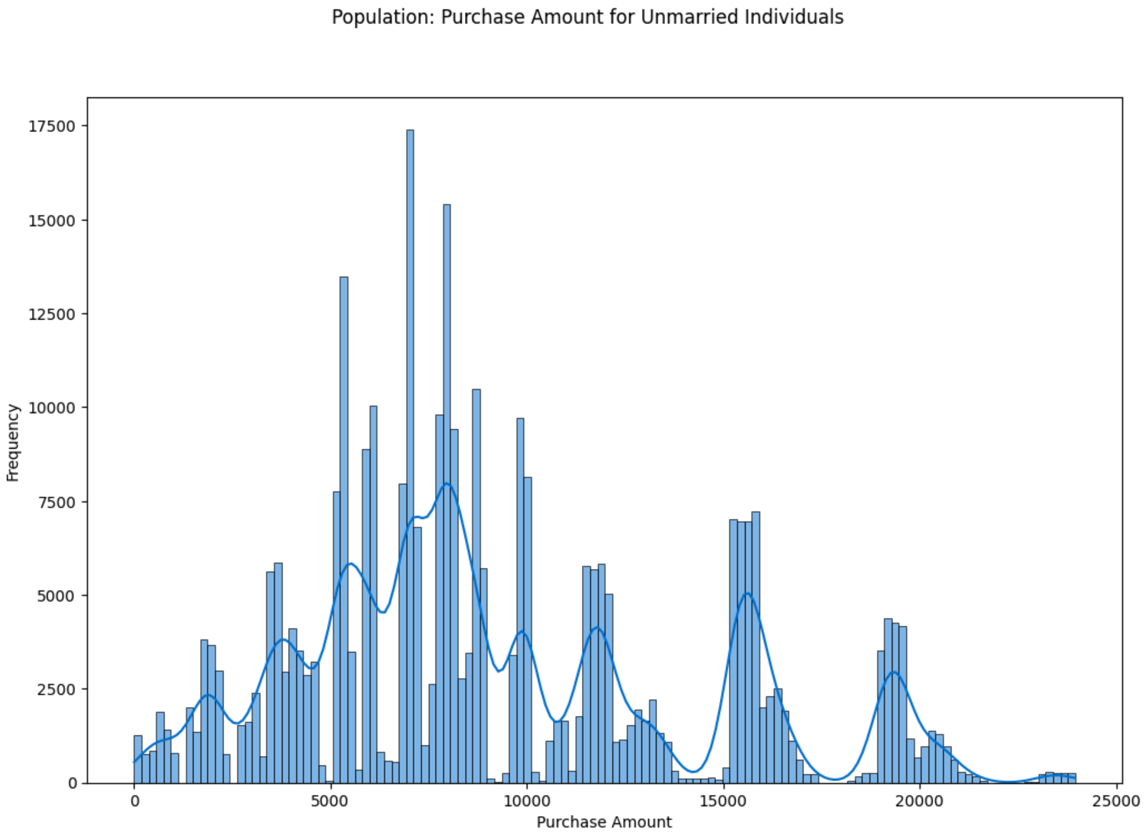
In [86]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Marital_Status"]=="Married"]["Purchas
e"].sample(n=10000)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Married Purchase Analysis- Sample (Sample Size = 1000
0, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9261.31302908
Sample Standard Deviation:  49.040889416677246

Confidence Level: 90%
Standard Error: 0.4904
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9260.5064
Upper Limit for the Given Confidence: 9262.1197

Confidence Level: 95%
Standard Error: 0.4904
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9260.3518
Upper Limit for the Given Confidence: 9262.2742

Confidence Level: 99%
Standard Error: 0.4904
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9260.0498
Upper Limit for the Given Confidence: 9262.5762
```

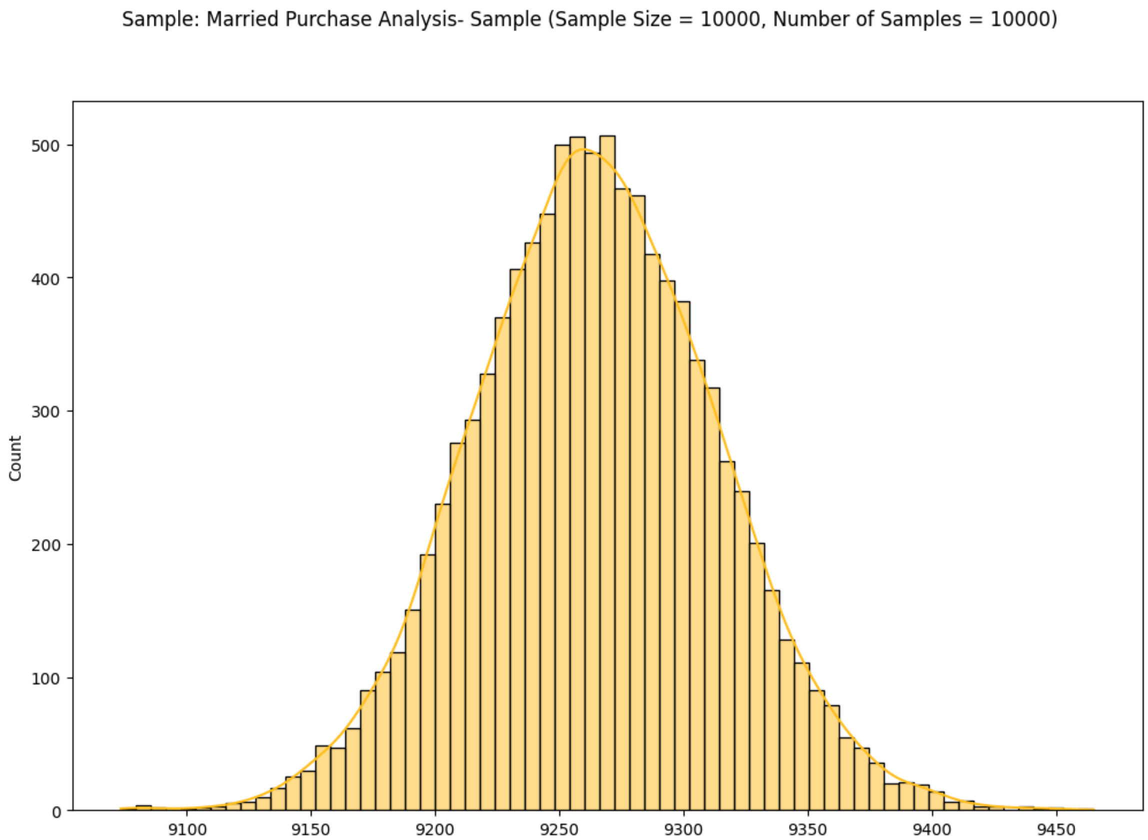
For Unmarried People

In [87]:
```
# Analysis on Purchase Amount as per the Marital Status for Unmarried indiv
iduals
fig = plt.figure(figsize=(12, 8))
fig.suptitle("Population: Purchase Amount for Unmarried Individuals")
sns.histplot(mart_df.loc[mart_df["Marital_Status"] == "Unmarried"]["Purchas
e"], kde=True, color="#0071ce")
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

population_mean = np.mean(mart_df.loc[mart_df["Marital_Status"] == "Unmarri
ed"]["Purchase"])
population_std = np.std(mart_df.loc[mart_df["Marital_Status"] == "Unmarrie
d"]["Purchase"])
print("Population Mean: {:.2f} \nPopulation Standard Deviation: {:.2f}".for
mat(population_mean, population_std))
```




```
Population Mean: 9265.91
Population Standard Deviation: 5027.34
```

In [88]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Marital_Status"]=="Unmarried"]["Purchas
e"].sample(n=10000)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Married Purchase Analysis- Sample (Sample Size = 1000
0, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9264.35055726
Sample Standard Deviation:  49.10874706964143

Confidence Level: 90%
Standard Error: 0.4911
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9263.5428
Upper Limit for the Given Confidence: 9265.1583

Confidence Level: 95%
Standard Error: 0.4911
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9263.3880
Upper Limit for the Given Confidence: 9265.3131

Confidence Level: 99%
Standard Error: 0.4911
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9263.0856
Upper Limit for the Given Confidence: 9265.6155
```

Analysis and Explanation

Based on the provided data, the population mean for married consumers is 9261.17, while for unmarried consumers it is slightly higher at 9265.91. The sample mean, obtained with each of 10,000 samples, each of size 10,000, for married customers is 9261.67, and for unmarried customers, it is 9265.34.

For the 95% Confidence Interval (CI), the range for married customers is 9260.69 to 9262.66, and for unmarried customers, it is 9264.38 to 9266.31. For the 99% CI, the range for married customers is 9260.37 to 9262.97, and for unmarried customers, it is 9264.07 to 9266.61.

Based on these figures, it can be concluded that unmarried customers spent more on average than married customers. Additionally, there is no overlap between the confidence intervals for the two groups, indicating a statistically significant difference in spending behavior between married and unmarried customers.

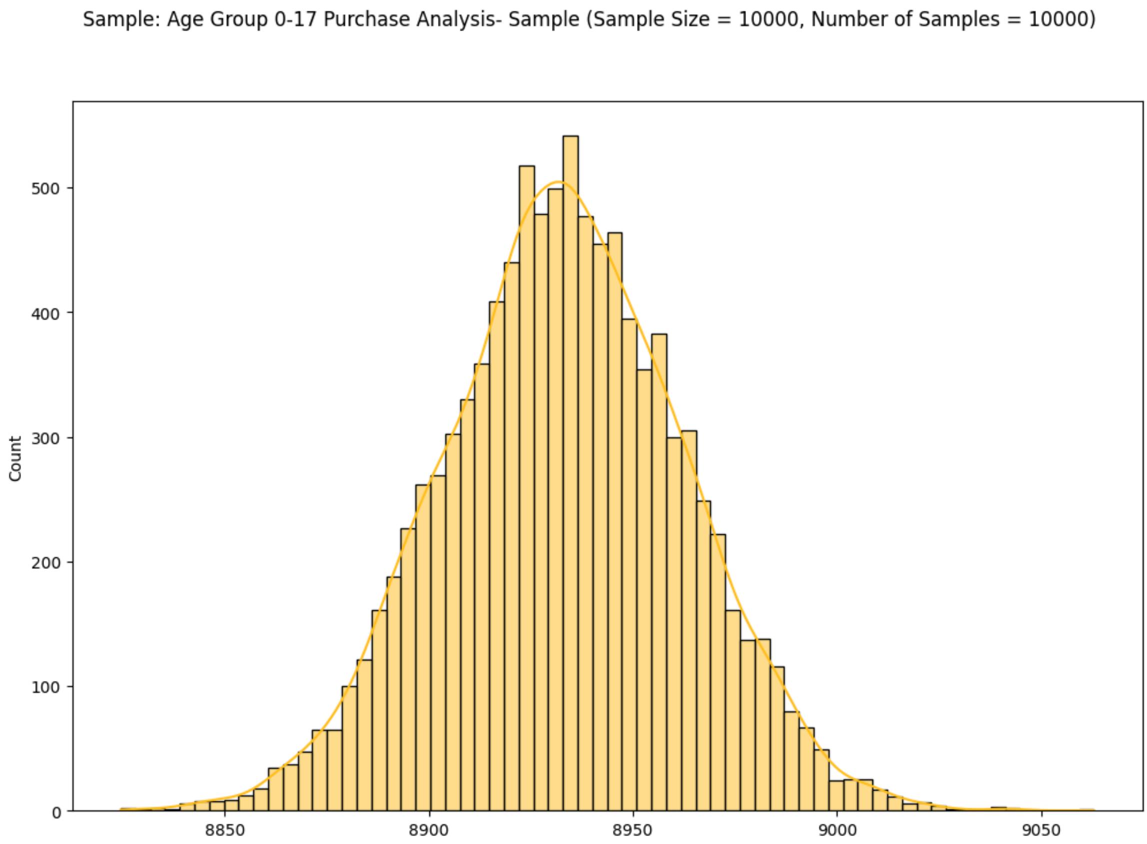
Purchase Analysis for different Age Groups on CI and CLT.

In [89]:
```
count_matrix = mart_df.groupby(["Age"])["Purchase"].aggregate([np.mean, np.std]).reset_index()
count_matrix.columns = ["Age Group", "Population Mean", "Population Standard Deviation"]
print(count_matrix)
```

```
  Age Group  Population Mean  Population Standard Deviation
0      0-17      8933.464640                     5111.114046
1     18-25      9169.663606                     5034.321997
2     26-35      9252.690633                     5010.527303
3     36-45      9331.350695                     5022.923879
4     46-50      9208.625697                     4967.216367
5     51-55      9534.808031                     5087.368080
6       55+      9336.280459                     5011.493996
```

Purchase Analysis for 0-17 Age Group

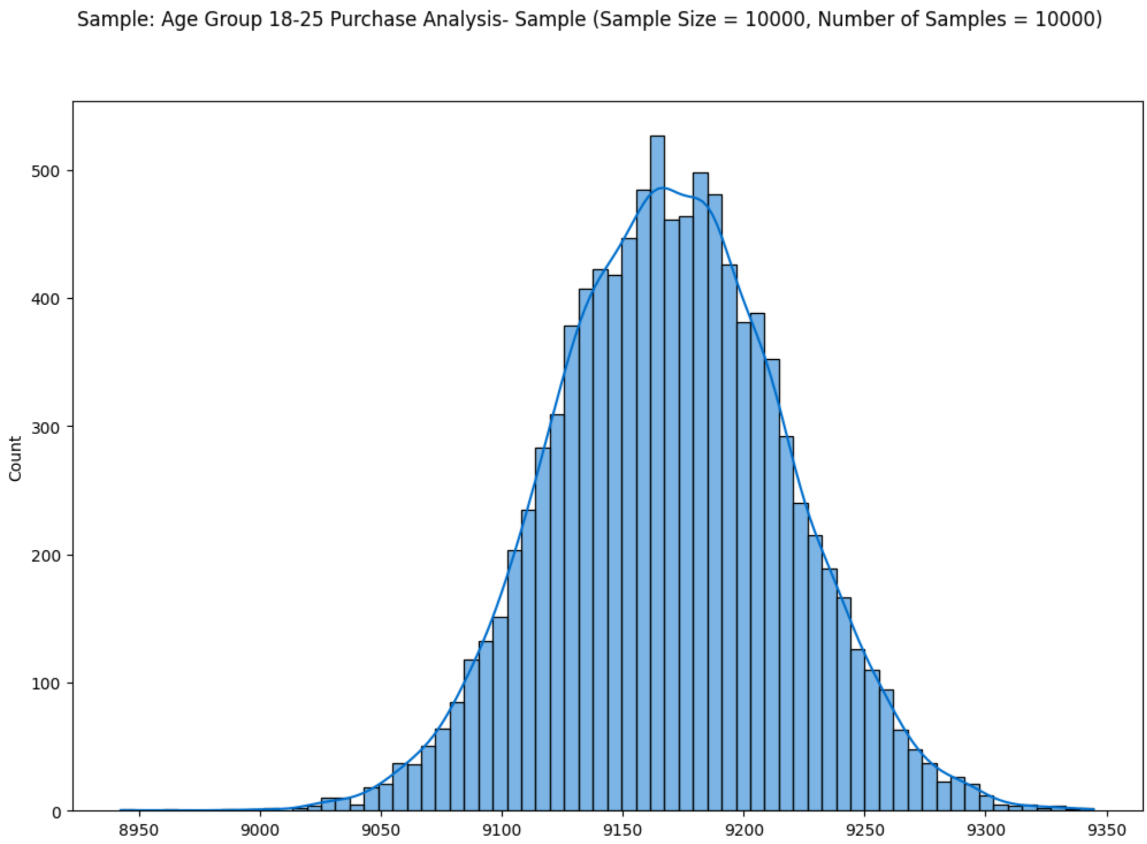
In [90]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="0-17"]["Purchase"].sample(n=1000
0)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 0-17 Purchase Analysis- Sample (Sample Size
= 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  8932.96984323
Sample Standard Deviation:  29.420623926182433

Confidence Level: 90%
Standard Error: 0.2942
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 8932.4859
Upper Limit for the Given Confidence: 8933.4538

Confidence Level: 95%
Standard Error: 0.2942
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 8932.3932
Upper Limit for the Given Confidence: 8933.5465

Confidence Level: 99%
Standard Error: 0.2942
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 8932.2120
Upper Limit for the Given Confidence: 8933.7277
```

Purchase Analysis for 18-25 Age Group

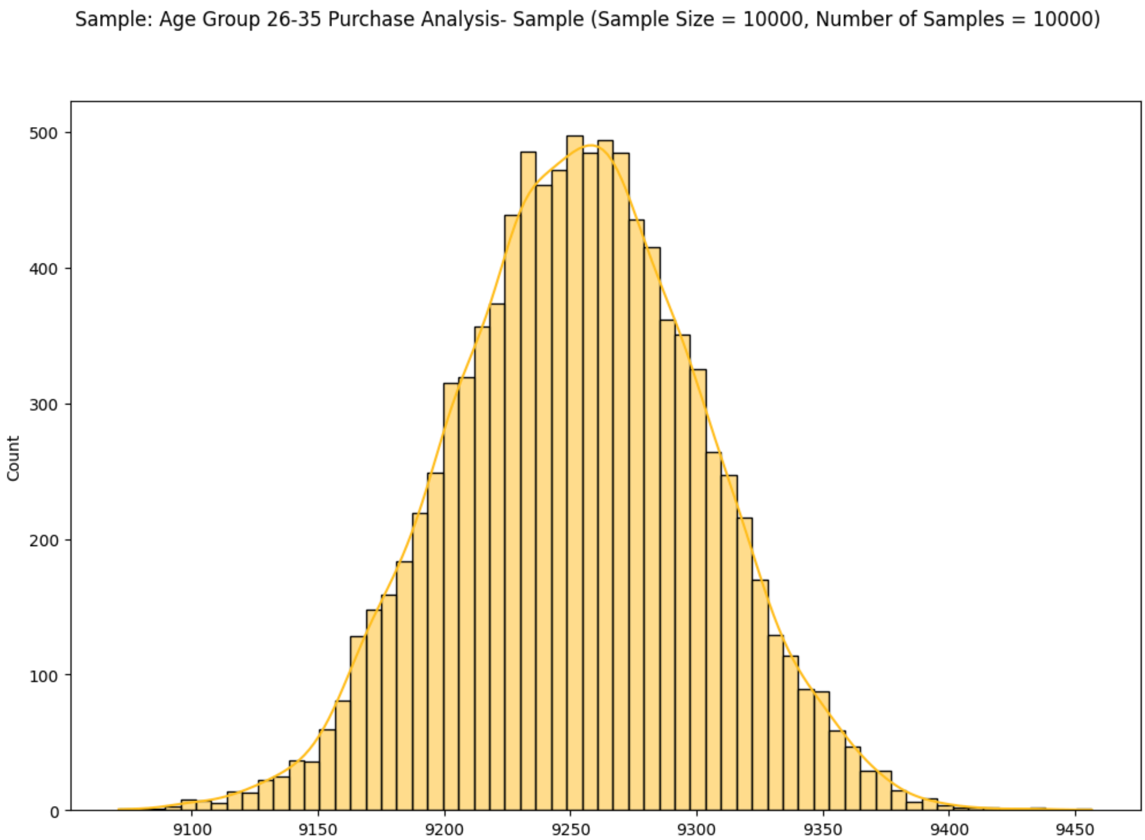
In [91]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="18-25"]["Purchase"].sample(n=100
00)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 18-25 Purchase Analysis- Sample (Sample Siz
e = 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True, color="#0071ce")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9169.366781589999
Sample Standard Deviation:  47.48494590630835

Confidence Level: 90%
Standard Error: 0.4748
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9168.5857
Upper Limit for the Given Confidence: 9170.1478

Confidence Level: 95%
Standard Error: 0.4748
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9168.4361
Upper Limit for the Given Confidence: 9170.2975

Confidence Level: 99%
Standard Error: 0.4748
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9168.1437
Upper Limit for the Given Confidence: 9170.5899
```

Purchase Analysis for 26-35 Age Group

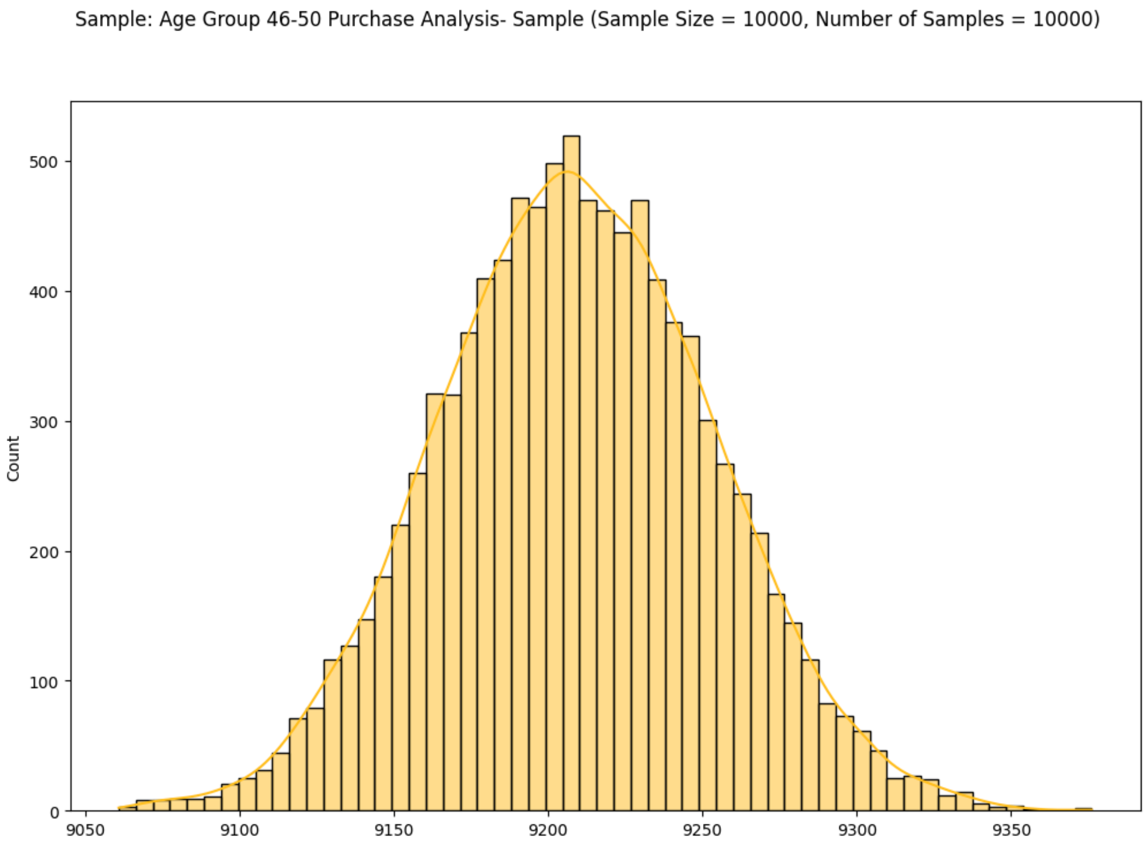
In [94]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="26-35"]["Purchase"].sample(n=100
00)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 26-35 Purchase Analysis- Sample (Sample Siz
e = 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9252.94804924
Sample Standard Deviation:  49.232305976867394

Confidence Level: 90%
Standard Error: 0.4923
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9252.1382
Upper Limit for the Given Confidence: 9253.7578

Confidence Level: 95%
Standard Error: 0.4923
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9251.9831
Upper Limit for the Given Confidence: 9253.9130

Confidence Level: 99%
Standard Error: 0.4923
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9251.6799
Upper Limit for the Given Confidence: 9254.2162
```

Purchase Analysis for 46-50 Age Group

In [95]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="46-50"]["Purchase"].sample(n=100
00)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 46-50 Purchase Analysis- Sample (Sample Siz
e = 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```

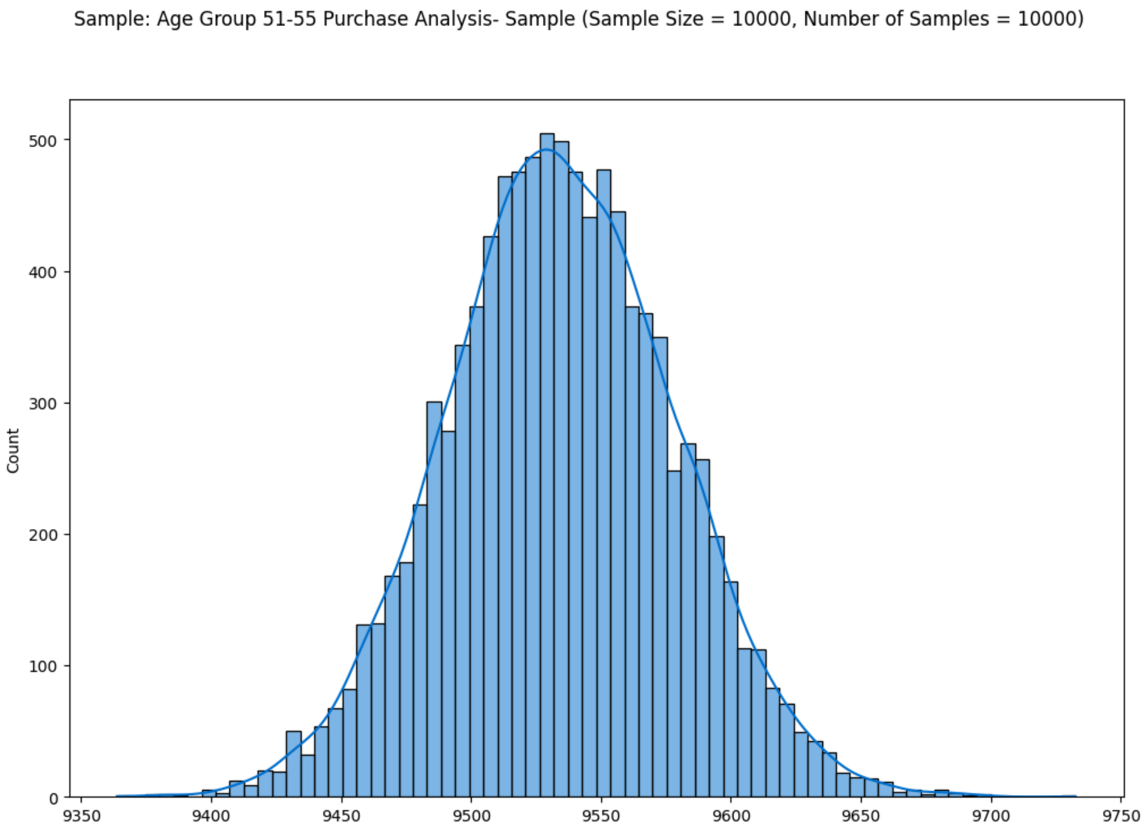



```
Sample Mean:  9208.862126950002
Sample Standard Deviation:  44.45788562146366

Confidence Level: 90%
Standard Error: 0.4446
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9208.1309
Upper Limit for the Given Confidence: 9209.5934

Confidence Level: 95%
Standard Error: 0.4446
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9207.9908
Upper Limit for the Given Confidence: 9209.7335

Confidence Level: 99%
Standard Error: 0.4446
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9207.7170
Upper Limit for the Given Confidence: 9210.0073
```

Purchase Analysis for 51-55 Age Group

In [96]:

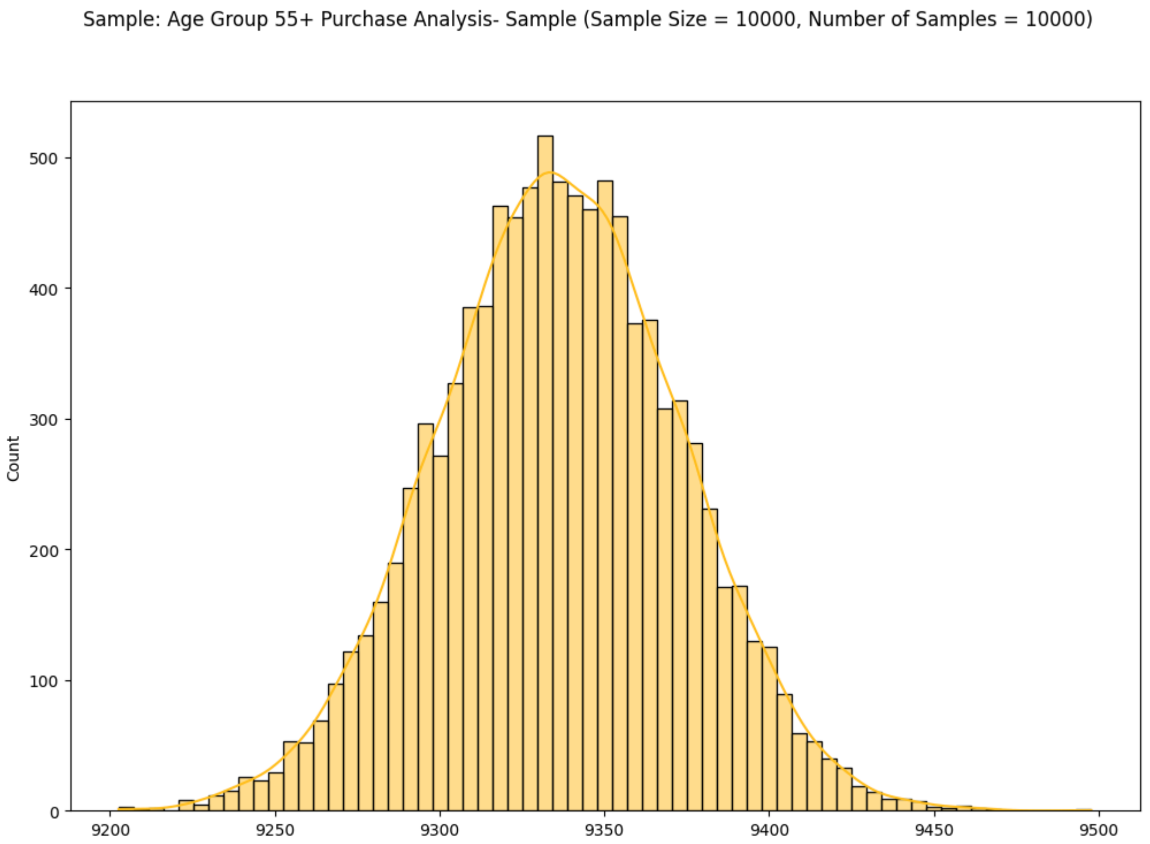
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="51-55"]["Purchase"].sample(n=10000)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 51-55 Purchase Analysis- Sample (Sample Size = 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#0071ce")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9534.120736809999
Sample Standard Deviation:  43.894009044564406

Confidence Level: 90%
Standard Error: 0.4389
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9533.3987
Upper Limit for the Given Confidence: 9534.8427

Confidence Level: 95%
Standard Error: 0.4389
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9533.2604
Upper Limit for the Given Confidence: 9534.9810

Confidence Level: 99%
Standard Error: 0.4389
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9532.9901
Upper Limit for the Given Confidence: 9535.2514
```

Purchase Analysis for 55+ Age Group

In [97]:
```
sampleMeans = []
for index in range(10000):
    samples = mart_df.loc[mart_df["Age"]=="55+"]["Purchase"].sample(n=1000
0)
    sampleMean = np.mean(samples)
    sampleMeans.append(sampleMean)

fig = plt.figure(figsize=(12,8))
fig.suptitle("Sample: Age Group 55+ Purchase Analysis- Sample (Sample Size
= 10000, Number of Samples = 10000)")
sns.histplot(sampleMeans,kde=True,color="#ffc120")
plt.show()


sample_mean = np.mean(sampleMeans)
sample_std = np.std(sampleMeans)

print("Sample Mean: ", sample_mean)
print("Sample Standard Deviation: ", sample_std)
print()
calc_CI(sample_mean, sample_std, 10000, 90)
print()
calc_CI(sample_mean, sample_std, 10000, 95)
print()
calc_CI(sample_mean, sample_std, 10000, 99)
```




```
Sample Mean:  9336.00941021
Sample Standard Deviation:  36.92312968190224

Confidence Level: 90%
Standard Error: 0.3692
z1: -1.6449
z2: 1.6449
Lower Limit for the Given Confidence: 9335.4021
Upper Limit for the Given Confidence: 9336.6167

Confidence Level: 95%
Standard Error: 0.3692
z1: -1.9600
z2: 1.9600
Lower Limit for the Given Confidence: 9335.2857
Upper Limit for the Given Confidence: 9336.7331

Confidence Level: 99%
Standard Error: 0.3692
z1: -2.5758
z2: 2.5758
Lower Limit for the Given Confidence: 9335.0583
Upper Limit for the Given Confidence: 9336.9605
```

Analysis and Explanation

Based on the provided data and confidence intervals for different age groups:

Consumers aged 51-55 have the highest average purchase amount, followed by those in the 55+ age group and the 36-45 age group. Customers in the 0-17 age group and the 18-25 age group have the lowest average purchase amounts among all age groups. These conclusions highlight variations in purchasing behavior across different age demographics. While consumers in the 26-35 age group may make more purchases overall, they do not necessarily have the highest average spending per transaction. Understanding these nuances in purchasing behavior among different age groups can help inform targeted marketing strategies and product offerings to better meet the needs and preferences of diverse customer segments.

Business Insights

Based on the analysis of gender distribution and purchase behavior, it's evident that males tend to spend more than females overall. Additionally, marital status significantly influences purchasing behavior, with unmarried consumers emerging as major spenders. Targeting products tailored to the preferences of each gender can maximize profits.

Unmarried individuals, particularly unmarried males and married females, exhibit higher purchasing tendencies compared to their counterparts. Tailoring product placement and advertising strategies to specific target audiences can capitalize on these trends effectively.

Product category preferences vary between genders, with Category 5 being more popular among females and Category 1 among males. This segmentation allows for targeted marketing efforts to resonate with each demographic.

Certain products, such as P00265242 and P00025442, resonate strongly with both genders. Promoting these products in a gender-neutral manner can enhance customer retention across genders.

City residency duration also influences spending behavior, with customers staying in a city for one year exhibiting the highest spending. This pattern suggests that individuals may initially require products for settling down and then maintain consistent spending levels afterward.

City type also plays a role, with Type B cities showing the highest average spending. However, nuanced differences emerge based on gender and city type, indicating the importance of tailoring marketing strategies to specific demographics.

Analysis using the Central Limit Theorem (CLT) and Confidence Intervals (CI) reveals that while the majority of purchases occur within the 0-35 age group, higher average spending is observed in the 36+ age group. This underscores the importance of segmenting users by age to target those with higher purchasing power.

Occupation level also correlates with spending behavior, with females in occupation levels 18 and 17 exhibiting the highest spending, while males in levels 12 and 15 lead in spending. Understanding these relationships allows for targeted marketing efforts based on occupation and gender.

By leveraging these insights, Walmart can optimize its marketing strategies by tailoring product offerings, advertising campaigns, and promotions to specific demographic segments, ultimately maximizing profitability and customer satisfaction.

Recommendations

1. **Implement Purchase Rewards:** Introduce rewards or discounts for purchases exceeding $10,000 to incentivize customers to increase their spending.
2. **Target City Type B Residents:** Focus advertising and promotions on customers residing in Type B cities for one year, as they demonstrate higher spending tendencies.
3. **Diversify Product Offerings:** Expand the inventory of products in Category 5 to cater to female customers, while increasing promotion and availability of Category 1 products for males.
4. **Tailor Marketing to Marital Status:** Target unmarried males and married females with tailored advertisements and promotions to attract new customers and enhance engagement among existing ones.
5. **Offer Affordable Options for Younger Customers:** Provide a variety of products priced under $9,000 to appeal to customers aged 0-35, encouraging increased engagement and purchase frequency.
6. **Promote Luxury Items to Older Demographics:** Target customers aged 36 and above with high-end products priced above $9,000, capitalizing on their higher purchasing power.
7. **Utilize Occupation Data for Targeted Marketing:** Leverage occupation levels to segment customers and tailor pricing tiers or product offerings to match their preferences and purchasing behavior.