Problem Statement:

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

```
#importing necessary libraries of python.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from scipy.stats import binom,geom,poisson
import scipy.stats as stats
from scipy.stats import norm
from scipy.stats import ttest_ind,chi2_contingency,kruskal,levene,kstest
from scipy.stats import pearsonr,spearmanr
import statsmodels.api as sm
from statsmodels.formula.api import ols
import random
```

```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv?1641285094
```

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv?1641285094
To: /content/walmart_data.csv?1641285094
100% 23.0M/23.0M [00:00<00:00, 56.7MB/s]
```

```
#Reading csv file
df=pd.read_csv('/content/walmart_data.csv?1641285094')
df
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category
0	1000001	P00069042	F	0-17	10	A	2	0	3
1	1000001	P00248942	F	0-17	10	A	2	0	1
2	1000001	P00087842	F	0-17	10	A	2	0	12
3	1000001	P00085442	F	0-17	10	A	2	0	12
4	1000002	P00285442	M	55+	16	C	4+	0	8
...	...	...	...	...	...	...	...	...	...
550063	1006033	P00372445	M	51-55	13	B	1	1	20
550064	1006035	P00375436	F	26-35	1	C	3	0	20
				26-					

```
# Gives total number of rows and columns
df.shape
```

```
(550068, 10)
```

There are 550068 rows and 10 columns

```
# Gives name of columns
df.columns
```

```
Index(['User_ID', 'Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category',
       'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category',
       'Purchase'],
      dtype='object')
```

```
# Brief information of the dataset
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
```

```
categorical_cols = ['Product_ID', 'Gender', 'Age' ,'City_Category','Stay_In_Current_City_Years']
df[categorical_cols] = df[categorical_cols].astype('category')

df.dtypes
```

	0
User_ID	int64
Product_ID	category
Gender	category
Age	category
Occupation	int64
City_Category	category
Stay_In_Current_City_Years	category
Marital_Status	int64
Product_Category	int64
Purchase	int64

dtype: object

This will convert the specified columns to the category dtype, which is useful for memory optimization and efficient processing of categorical data.

```
# Nature of data
type(df)
```

```
pandas.core.frame.DataFrame
def __init__(data=None, index: Axes | None=None, columns: Axes | None=None, dtype: Dtype |
None=None, copy: bool | None=None) -> None
```

```
Two-dimensional, size-mutable, potentially heterogeneous tabular data.

Data structure also contains labeled axes (rows and columns).
Arithmetic operations align on both row and column labels. Can be
thought of as a dict-like container for Series objects. The primary
pandas data structure.
```

```
# Unique value of each column.

df.nunique()
```

	0
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

```
# Top 10 rows
df.head(10)
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
0	1000001	P00069042	F	0-17	10	A	2	0	3	8
1	1000001	P00248942	F	0-17	10	A	2	0	1	15
2	1000001	P00087842	F	0-17	10	A	2	0	12	1
3	1000001	P00085442	F	0-17	10	A	2	0	12	1
4	1000002	P00285442	M	55+	16	C	4+	0	8	7
5	1000003	P00193542	M	26-35	15	A	3	0	1	15
6	1000004	P00184942	M	46-50	7	B	2	1	1	19

```
# Bottom 10 rows
df.tail(10)
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	
550058	1006024	P00372445	M	26-35	12	A	0	1	20	
550059	1006025	P00370853	F	26-35	1	B	1	0	19	
550060	1006026	P00371644	M	36-45	6	C	1	1	20	
550061	1006029	P00372445	F	26-35	1	C	1	1	20	
550062	1006032	P00372445	M	46-50	7	A	3	0	20	
550063	1006033	P00372445	M	51-55	13	B	1	1	20	
				26-						

```
#Count NaN values per column
df.isna().sum()
```

	0
User_ID	0
Product_ID	0
Gender	0
Age	0
Occupation	0
City_Category	0
Stay_In_Current_City_Years	0
Marital_Status	0
Product_Category	0
Purchase	0

dtype: int64

There are no null or NaN values in the dataset.

```
# Necessary details are calculated using 'describe' function.
# Thus 'describe' depicts the calculation on numerical data.
df.describe()
```

	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

Key Observations:

Descriptive Statistics for Numerical Columns:

Occupation: The mean occupation value is around 8.08, with a standard deviation of 6.52. Values range from 0 to 20.

Marital_Status: The mean is 0.41, indicating that a significant portion of the users have a marital status of 0 (unmarried) as opposed to 1 (married). This column takes binary values (0 or 1).

Product_Category: The average product category is around 5.40, with values ranging from 1 to 20.

Purchase: The average purchase amount is approximately 9263.97,with a standard deviationof 5023.07. Purchase amounts range from 12to 23961

INTRODUCTION 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. 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

Checking value counts for various columns

```
df['Product_ID'].value_counts()
```

Product_ID	count
P00265242	1880
P00025442	1615
P00110742	1612
P00112142	1562
P00057642	1470
...	...
P00077342	1
P00077542	1
P00068342	1
P00074742	1
P00074542	1

3631 rows × 1 columns

dtype: int64

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

Gender	count
M	414259
F	135809

dtype: int64

```
df['City_Category'].value_counts()
```

City_Category	count
B	231173
C	171175
A	147720

dtype: int64

```
df['Stay_In_Current_City_Years'].value_counts()
```

Stay_In_Current_City_Years	count
1	193821
2	101838
3	95285
4+	84726
0	74398

dtype: int64

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

Marital_Status	count
0	324731
1	225337

dtype: int64

```
df['Product_Category'].value_counts()
```

Product_Category	count
5	150933
1	140378
8	113925
11	24287
2	23864
6	20466
3	20213
4	11753
16	9828
15	6290
13	5549
10	5125
12	3947
7	3721
18	3125
20	2550
19	1603
14	1523
17	578
9	410

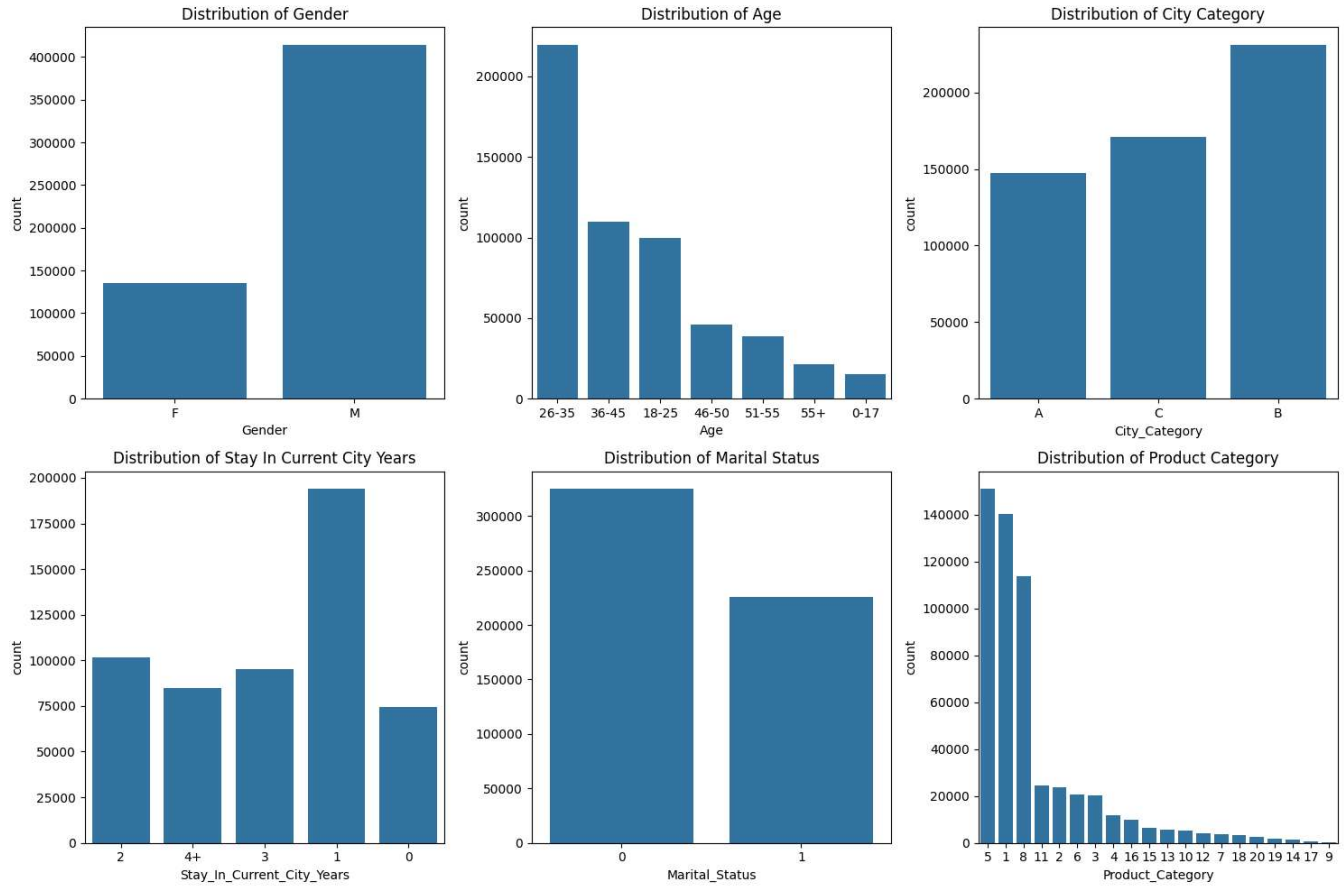
dtype: int64

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

Age	count
26-35	219587
36-45	110013
18-25	99660
46-50	45701
51-55	38501
55+	21504
0-17	15102

dtype: int64

```
# Distribution of categorical variables (bar plots)
plt.figure(figsize=(15, 10))

plt.subplot(2, 3, 1)
sns.countplot(df, x='Gender')
plt.title('Distribution of Gender')

plt.subplot(2, 3, 2)
sns.countplot(df, x='Age', order=df['Age'].value_counts().index)
plt.title('Distribution of Age')

plt.subplot(2, 3, 3)
sns.countplot(df, x='City_Category')
plt.title('Distribution of City Category')

plt.subplot(2, 3, 4)
sns.countplot(df, x='Stay_In_Current_City_Years')
plt.title('Distribution of Stay In Current City Years')

plt.subplot(2, 3, 5)
sns.countplot(df, x='Marital_Status')
plt.title('Distribution of Marital Status')

plt.subplot(2, 3, 6)
sns.countplot(df, x='Product_Category', order=df['Product_Category'].value_counts().index)
plt.title('Distribution of Product Category')

plt.tight_layout()
plt.savefig('categorical_distributions.png')
plt.show()
```



Key Observations:

The distributions of the categorical variables are visualized in the bar plots above. For instance, the 'Gender' distribution shows a higher number of male purchasers than female, and 'Age' group '26-35' has the highest count of purchasers. 'City_Category' 'B' has the highest number of purchasers, and 'Stay_In_Current_City_Years' '1' has the highest number of purchasers.

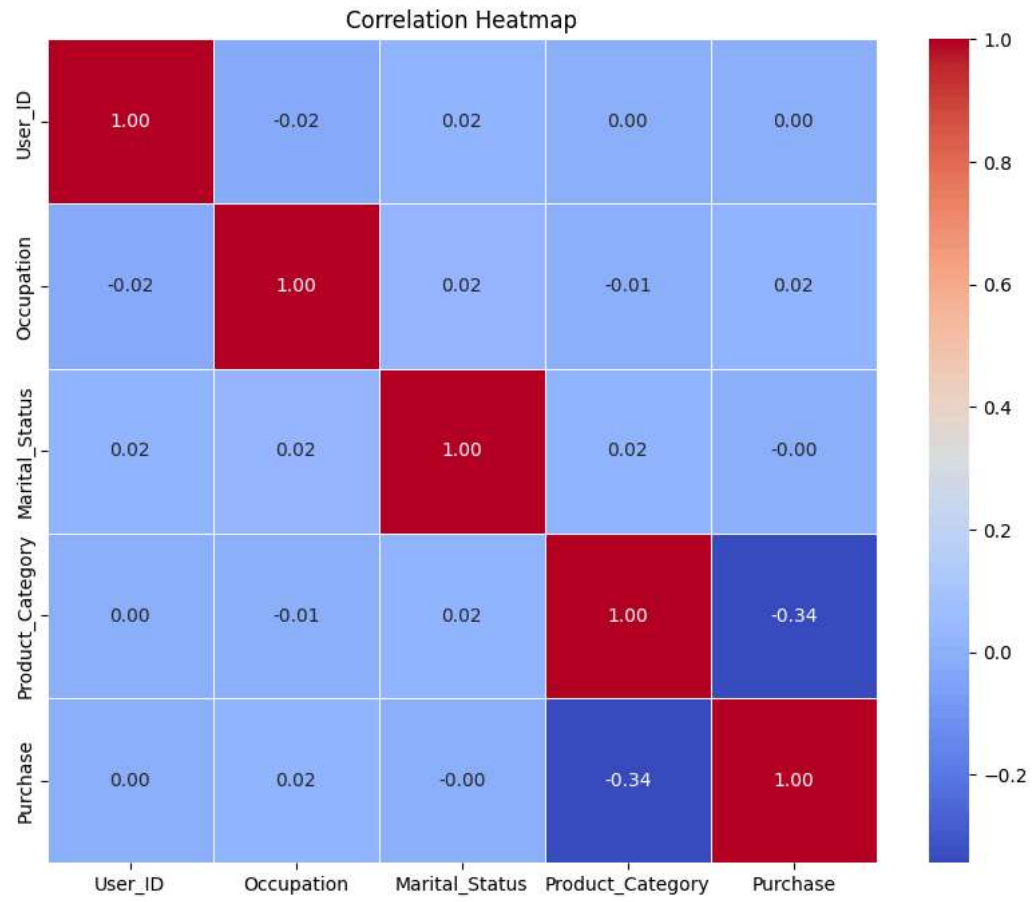
```
## Correlation Analysis

## Correlation is done on numerical columns only
```

```
#For numerical vs numerical
correlation_matrix = df.corr(numeric_only=True)
print(correlation_matrix)
```

```
                  User_ID  Occupation  Marital_Status  Product_Category  \
User_ID          1.000000   -0.023971        0.020443          0.003825
Occupation      -0.023971    1.000000        0.024280         -0.007618
Marital_Status   0.020443    0.024280        1.000000          0.019888
Product_Category 0.003825   -0.007618        0.019888          1.000000
Purchase         0.004716    0.020833       -0.000463         -0.343703

                 Purchase
User_ID          0.004716
Occupation       0.020833
Marital_Status  -0.000463
Product_Category -0.343703
Purchase         1.000000
```

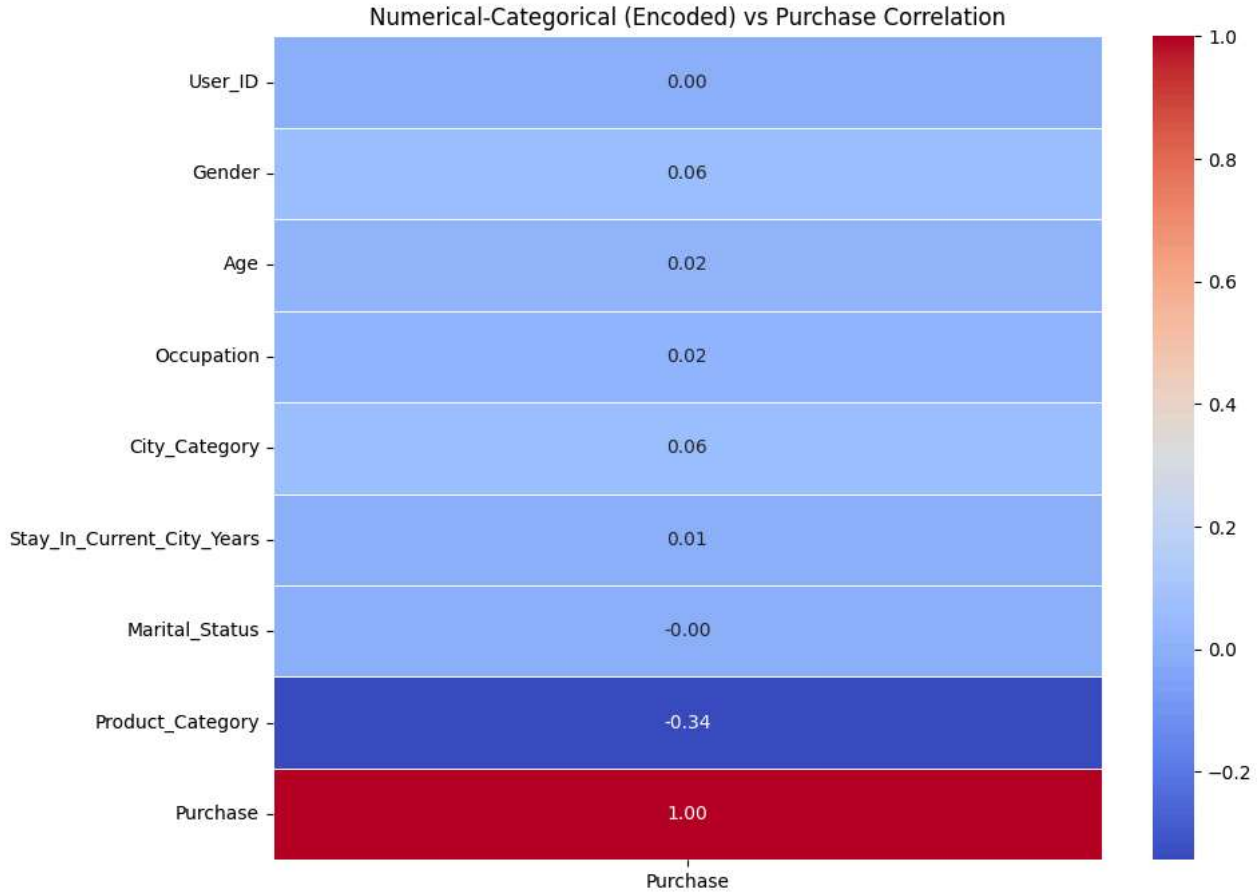
```
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
```

```
plt.title('Correlation Heatmap')
plt.show()
```



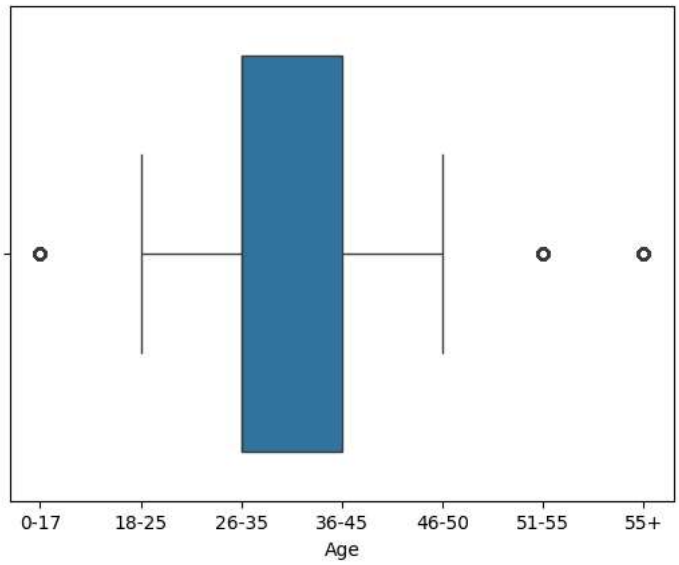

```
#For numerical vs categorical

# Encode categorical features
cat_cols = ['Gender', 'Age', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category']
df_encoded = df.copy()
for col in cat_cols:
    df_encoded[col] = df_encoded[col].astype('category').cat.codes

# Compute correlation matrix
mixed_corr = df_encoded.corr(numeric_only=True)
print(mixed_corr['Purchase'].sort_values(ascending=False))

# Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(mixed_corr[['Purchase']], annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Numerical-Categorical (Encoded) vs Purchase Correlation')
plt.show()
```

```
Purchase                    1.000000
City_Category               0.061914
Gender                      0.060346
Occupation                  0.020833
Age                         0.015839
Stay_In_Current_City_Years  0.005422
User_ID                     0.004716
Marital_Status             -0.000463
Product_Category           -0.343703
Name: Purchase, dtype: float64
```

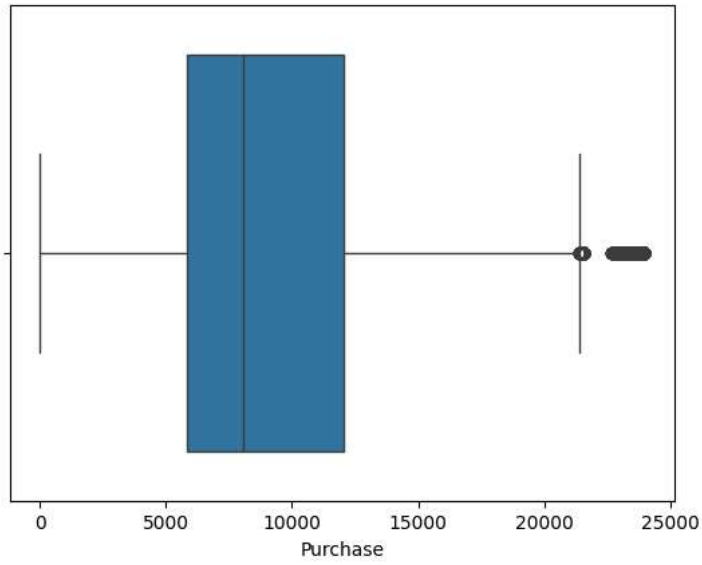


Observing the outliers:

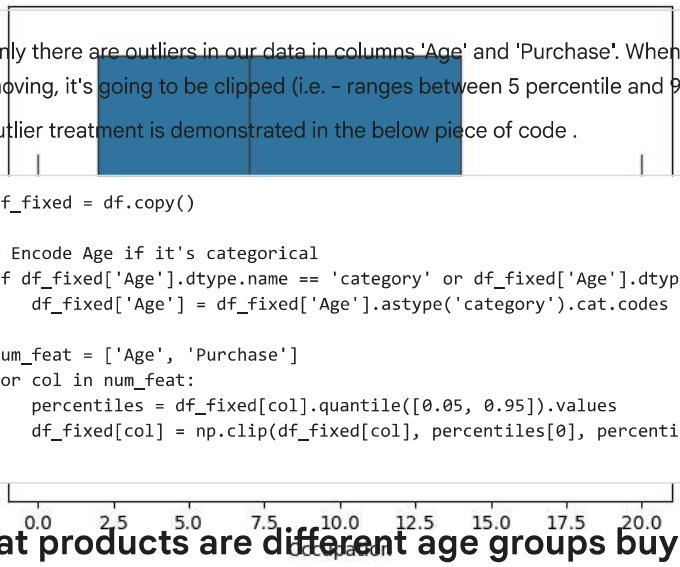
```
ax = sns.boxplot(x=df["Age"])
plt.show()
```




```
ax = sns.boxplot(x=df["Purchase"])
plt.show()
```




```
ax = sns.boxplot(x=df["Occupation"])
plt.show()
```



Certainly there are outliers in our data in columns 'Age' and 'Purchase'. When we remove them, this leads to loss of information. So instead of removing, it's going to be clipped (i.e. - ranges between 5 percentile and 95 percentile).

The outlier treatment is demonstrated in the below piece of code .

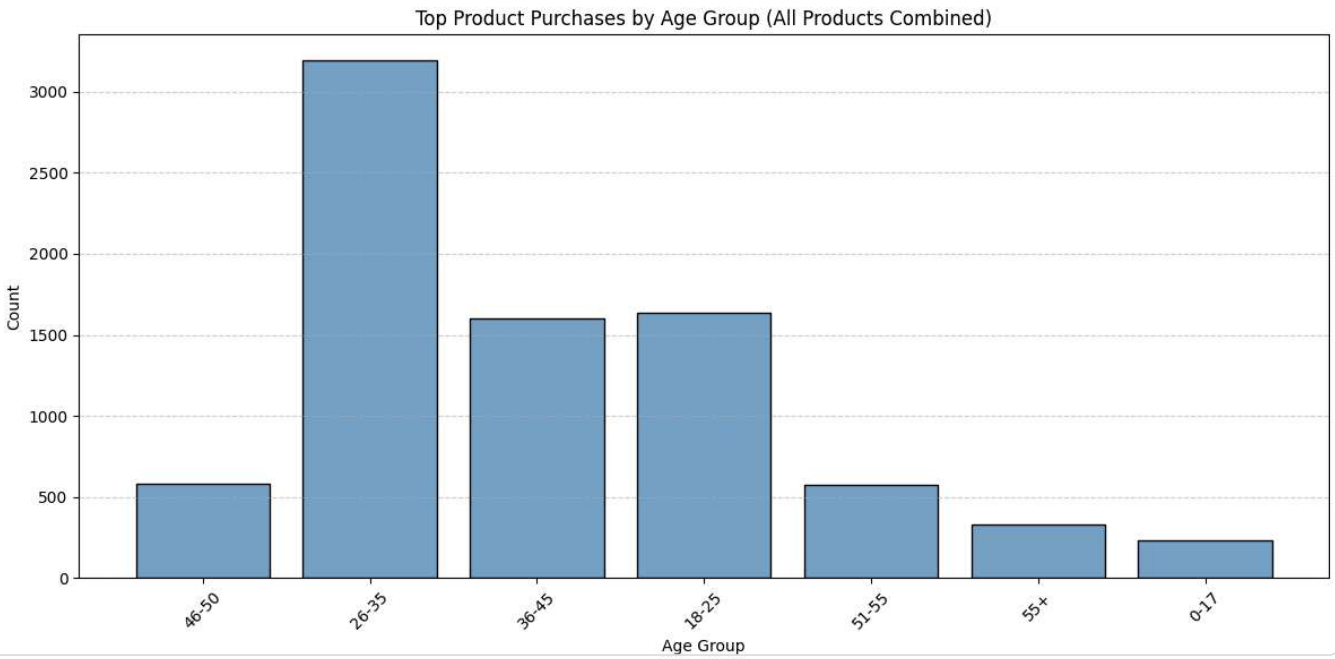
```
df_fixed = df.copy()

# Encode Age if it's categorical
if df_fixed['Age'].dtype.name == 'category' or df_fixed['Age'].dtype == 'object':
    df_fixed['Age'] = df_fixed['Age'].astype('category').cat.codes

num_feat = ['Age', 'Purchase']
for col in num_feat:
    percentiles = df_fixed[col].quantile([0.05, 0.95]).values
    df_fixed[col] = np.clip(df_fixed[col], percentiles[0], percentiles[1])
```

What products are different age groups buying?

```
df.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
0	1000001	P00069042	F	0-17	10	A	2	0	3	8
1	1000001	P00248942	F	0-17	10	A	2	0	1	15
2	1000001	P00087842	F	0-17	10	A	2	0	12	1

```
#Histplot
# Step 1: Filter top products
top_products = df['Product_ID'].value_counts().nlargest(5).index
filtered_df = df[df['Product_ID'].isin(top_products)]

# Step 2: Plot histplot
plt.figure(figsize=(12, 6))
sns.histplot(
    data=filtered_df,
    x='Age',
    bins=len(filtered_df['Age'].unique()),
    shrink=0.8,
    color='steelblue',
    discrete=True  # since Age is categorical
)
plt.title('Top Product Purchases by Age Group (All Products Combined)')
plt.xlabel('Age Group')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```

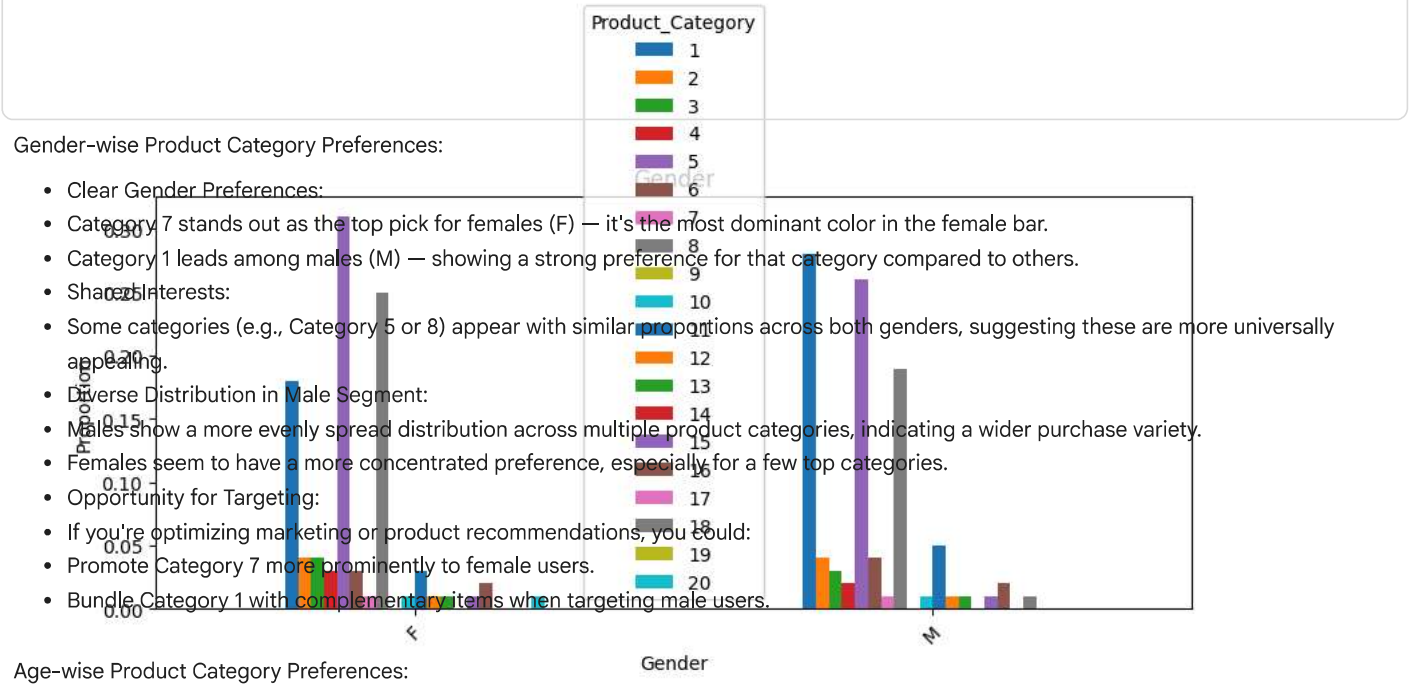


Key Observations:

- Dominant Buyer Group: The age group 26–35 clearly leads in total product purchases. This suggests they're either more engaged, economically active, or more responsive to the products being offered.
- Youth Power: The 18–25 segment comes next, showing solid engagement, possibly due to digital affinity or targeted marketing.
- Decline with Age: As age increases beyond 36–45, purchase volume drops significantly. This could reflect changing needs, purchasing habits, or digital access in older demographics.
- Product Popularity Spread: A handful of specific Product_IDs (from your top product list) are consistently popular across multiple age groups. However, some products are more favored in certain age brackets—worth digging into for targeted marketing.
- Low Response from <18 and 55+: These extremes represent niche or under-engaged segments.

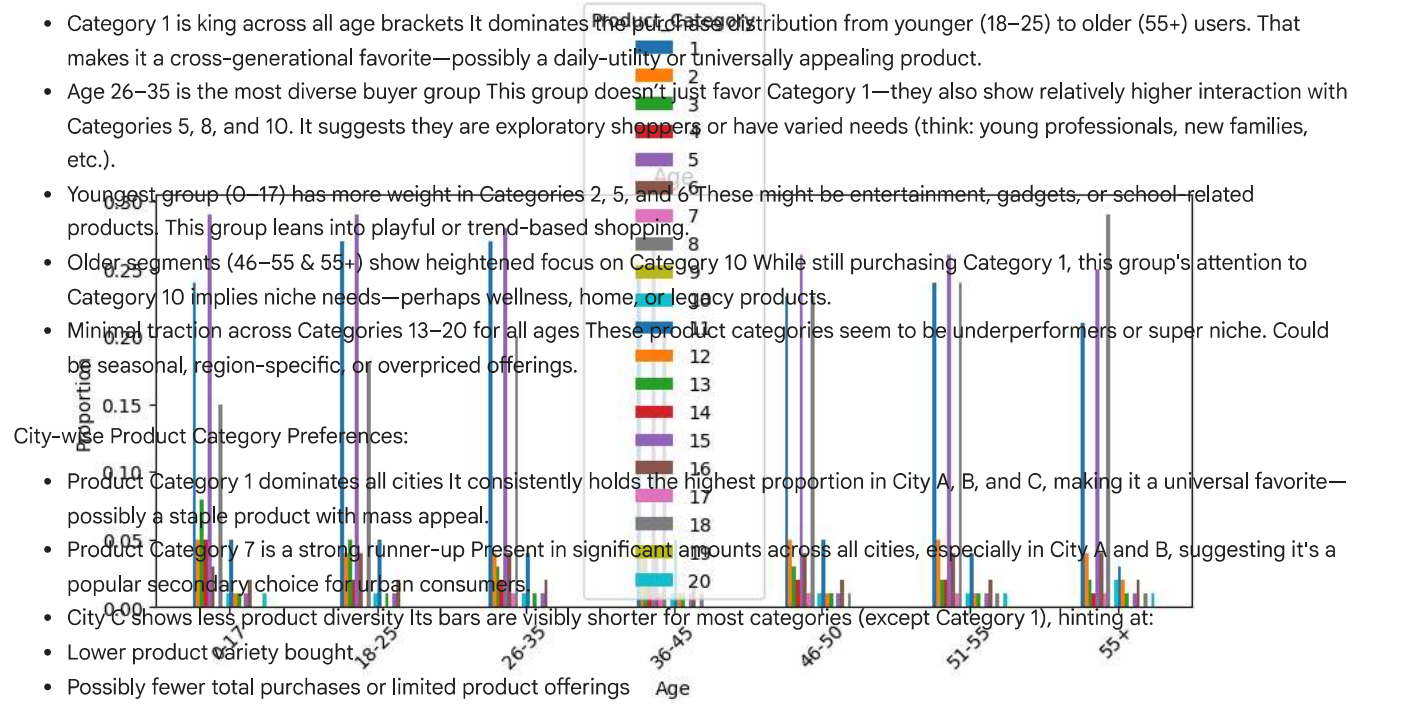
Are there preferred product categories for different genders?

```
#CrossTabs

cat_cols=['Gender','Age', 'City_Category','Marital_Status']
for i in cat_cols:
    other= round(pd.crosstab(df[df[i].notnull()][i], df['Product_Category']).\
                 div(pd.crosstab(df[df[i].notnull()][i],df['Product_Category']).apply(sum,1),0),2)
    ax = other.plot(kind ='bar', title = i, figsize = (10,4))
    ax.set_xlabel(i)
    ax.set_ylabel('Proportion')
    plt.xticks(rotation=45)
    plt.show()
```



Gender-wise Product Category Preferences:

- Clear Gender Preferences:
- Category 7 stands out as the top pick for females (F) — it's the most dominant color in the female bar.
- Category 1 leads among males (M) — showing a strong preference for that category compared to others.
- Shared Interests:
- Some categories (e.g., Category 5 or 8) appear with similar proportions across both genders, suggesting these are more universally appealing.
- Diverse Distribution in Male Segment:
- Males show a more evenly spread distribution across multiple product categories, indicating a wider purchase variety.
- Females seem to have a more concentrated preference, especially for a few top categories.
- Opportunity for Targeting:
- If you're optimizing marketing or product recommendations, you could:
- Promote Category 7 more prominently to female users.
- Bundle Category 1 with complementary items when targeting male users.

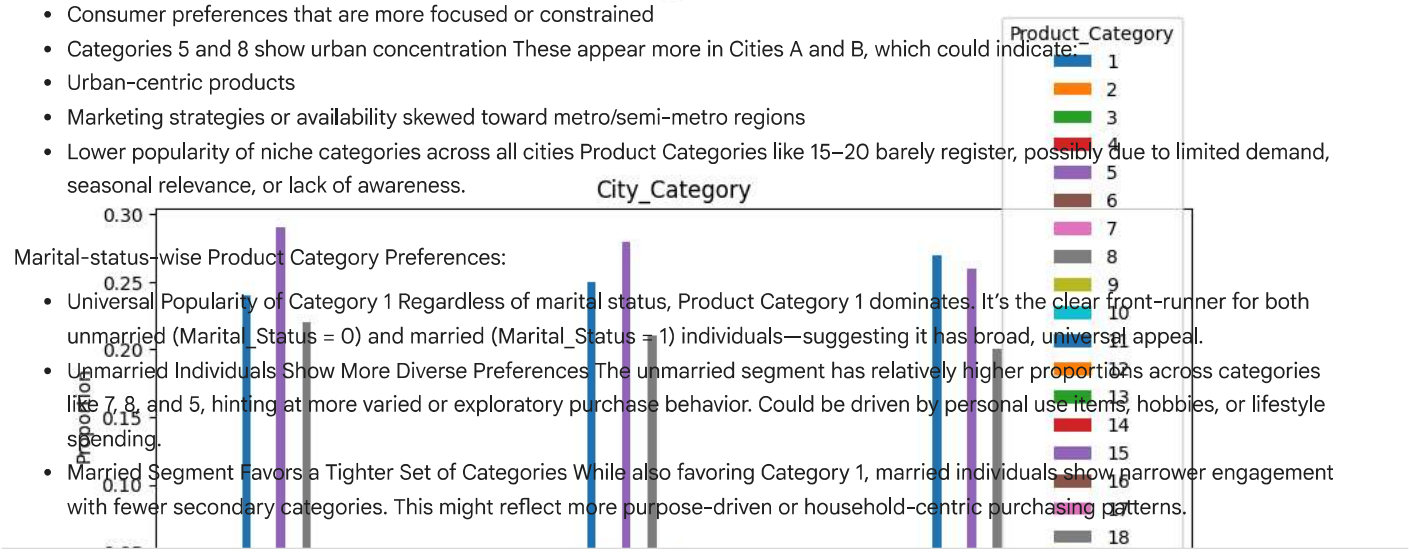


Age-wise Product Category Preferences:

- Category 1 is king across all age brackets It dominates the purchase distribution from younger (18–25) to older (55+) users. That makes it a cross-generational favorite—possibly a daily-utility or universally appealing product.
- Age 26–35 is the most diverse buyer group This group doesn't just favor Category 1—they also show relatively higher interaction with Categories 5, 8, and 10. It suggests they are exploratory shoppers or have varied needs (think: young professionals, new families, etc.).
- Youngest ~~group (0–17) has more weight in Categories~~ 2, 5, and 6 These ~~might be entertainment, gadgets, or school~~ related products. This group leans into playful or trend-based shopping.
- Older segments (46–55 & 55+) show heightened focus on Category 10 While still purchasing Category 1, this group's attention to Category 10 implies niche needs—perhaps wellness, home, or legacy products.
- Minimal traction across Categories 13–20 for all ages These product categories seem to be underperformers or super niche. Could be seasonal, region-specific, or overpriced offerings.

City-wise Product Category Preferences:

- Product Category 1 dominates all cities It consistently holds the highest proportion in City A, B, and C, making it a universal favorite—possibly a staple product with mass appeal.
- Product Category 7 is a strong runner-up Present in significant amounts across all cities, especially in City A and B, suggesting it's a popular secondary choice for urban consumers.
- City C shows less product diversity Its bars are visibly shorter for most categories (except Category 1), hinting at:
- Lower product variety bought
- Possibly fewer total purchases or limited product offerings
- Consumer preferences that are more focused or constrained
- Categories 5 and 8 show urban concentration These appear more in Cities A and B, which could indicate:
- Urban-centric products
- Marketing strategies or availability skewed toward metro/semi-metro regions
- Lower popularity of niche categories across all cities Product Categories like 15–20 barely register, possibly due to limited demand, seasonal relevance, or lack of awareness.



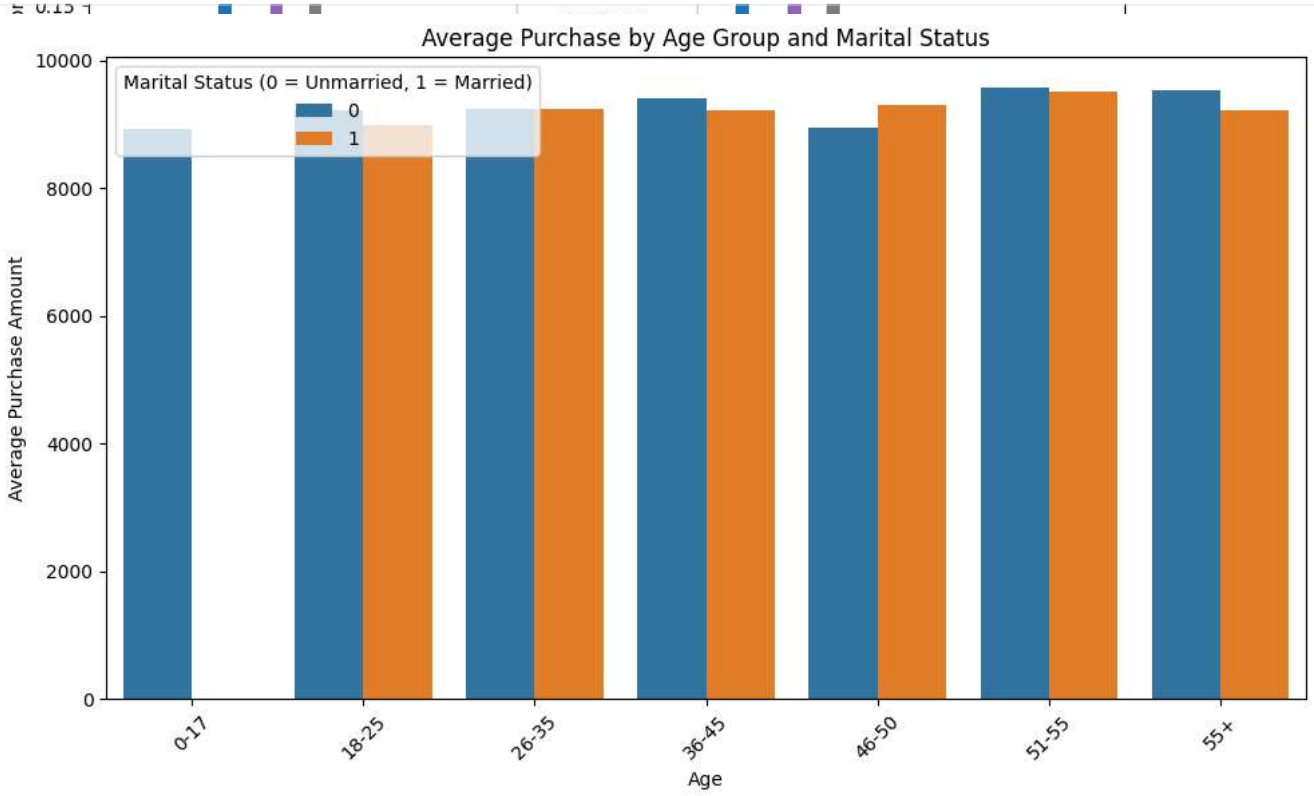
Marital-status-wise Product Category Preferences:

- Universal Popularity of Category 1 Regardless of marital status, Product Category 1 dominates. It's the clear front-runner for both unmarried (Marital_Status = 0) and married (Marital_Status = 1) individuals—suggesting it has broad, universal appeal.
- Unmarried Individuals Show More Diverse Preferences The unmarried segment has relatively higher proportions across categories like 7, 8, and 5, hinting at more varied or exploratory purchase behavior. Could be driven by personal use items, hobbies, or lifestyle spending.
- Married Segment Favors a Tighter Set of Categories While also favoring Category 1, married individuals show narrower engagement with fewer secondary categories. This might reflect more purpose-driven or household-centric purchasing patterns.

- Low Interaction with Higher Product Categories Product Categories 10 through 20 have consistently low proportions in both groups. These could be under-marketed, niche, or irrelevant to the majority of buyers.
- Potential Marketing Strategies:
- Promote Category 8 or 7 more heavily to unmarried buyers via digital platforms.
- Bundle Category 1 with others like 5 or 7 for married shoppers to drive cross-category interest.

Is there a relationship between age, marital status, and the amount spent?

```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Age', y='Purchase', hue='Marital_Status', ci=None)
plt.title('Average Purchase by Age Group and Marital Status')
plt.ylabel('Average Purchase Amount')
plt.xticks(rotation=45)
plt.legend(title='Marital Status (0 = Unmarried, 1 = Married)')
plt.tight_layout()
plt.show()
```



This shows how marital status affects spending across different age groups

2-way Anova Test:

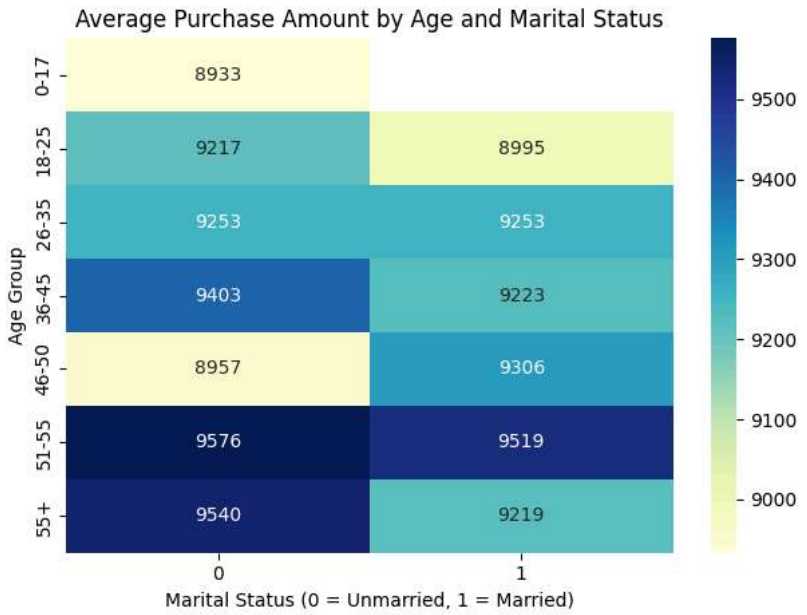
```
#Ho(Null Hypothesis): There's no effect
#Ha(Alternate Hypothesis): There's significant effect
#let's take significance level(alpha) as 0.05
df['Age'] = df['Age'].astype('category')
model = ols('Purchase ~ C(Age) + C(Marital_Status) + C(Age):C(Marital_Status)', data=df).fit()
anova_result = sm.stats.anova_lm(model, typ=2)
print(anova_result)
```

```
                              sum_sq        df             F        PR(>F)
C(Age)                  6.140003e+09       6.0  4.058509e+01  1.025085e-49
C(Marital_Status)      -1.959014e-04       1.0 -7.769388e-12  1.000000e+00
C(Age):C(Marital_Status)  3.326735e+09       6.0  2.198954e+01  5.060990e-26
Residual                1.386938e+13  550055.0           NaN           NaN
```

Key Observations:

- Age: Strongly influences the amount spent. Purchasing behavior varies significantly across age groups.(Reject Ho)
- Marital Status: Has no significant individual effect on purchase amount.(since PR value=1) Being married or unmarried alone doesn't change spending behavior noticeably.(Fail to reject Ho)
- Interaction (Age × Marital Status): The combined effect is significant, meaning:(Reject Ho)
- The impact of age on purchase behavior depends on marital status. For example, married individuals aged 26–35 might behave differently than unmarried peers of the same age.

```
# Heat Map
pivot = df.pivot_table(index='Age', columns='Marital_Status', values='Purchase', aggfunc='mean')

sns.heatmap(pivot, annot=True, fmt=".0f", cmap='YlGnBu')
plt.title('Average Purchase Amount by Age and Marital Status')
plt.xlabel('Marital Status (0 = Unmarried, 1 = Married)')
plt.ylabel('Age Group')
plt.tight_layout()
plt.show()
```



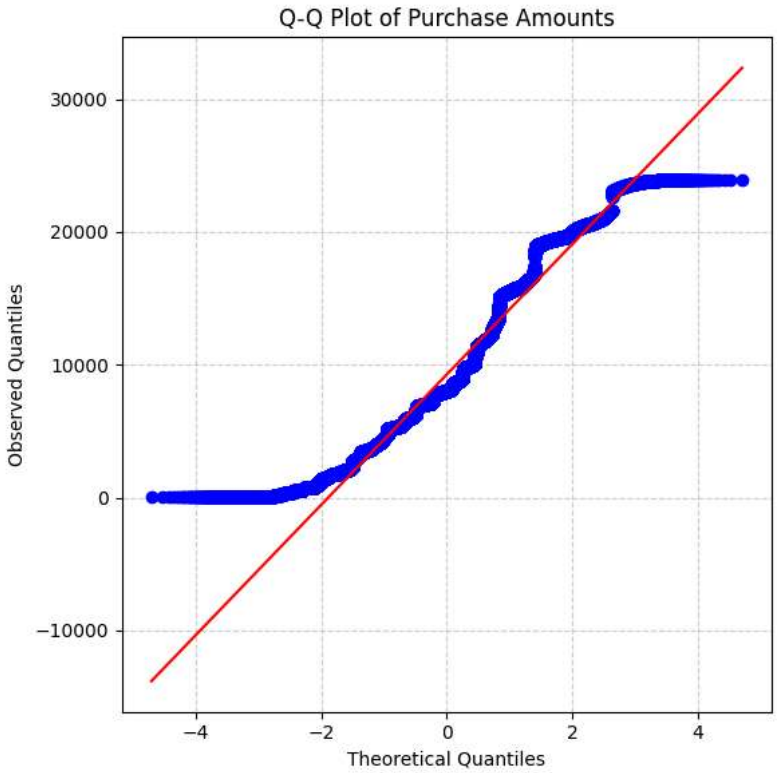
Key Observations:

- Age is a dominant factor in purchase behavior
- Average spending increases with age up to the 51–55 bracket, peaking for unmarried individuals at ₹9576.
- The 0–17 group shows the lowest average purchase (₹8933), which aligns with their likely limited purchasing power.
- Marital status has limited effect—except in a few age bands
- In younger (18–35) and older (51–55) groups, unmarried individuals tend to spend more than married individuals.
- An exception occurs in the 46–50 group, where married individuals outspend unmarried ones, possibly reflecting stable income or family-driven purchases.
- Strong interaction between Age × Marital Status
- The difference in spending between marital statuses isn't consistent across ages. This reinforces the significant interaction effect we found in the ANOVA test.
- For instance:
- Age 36–45 ⟶ Unmarried: ₹9403 vs Married: ₹9223
- Age 55+ ⟶ Unmarried: ₹9540 vs Married: ₹9219

Generate Q-Q plot:

```
# Generate Q-Q plot for the 'Purchase' column
plt.figure(figsize=(6, 6))
stats.probplot(df['Purchase'], dist="norm", plot=plt)
plt.title('Q-Q Plot of Purchase Amounts')
plt.xlabel('Theoretical Quantiles')
```

```
plt.ylabel('Observed Quantiles')
plt.grid(True, linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```



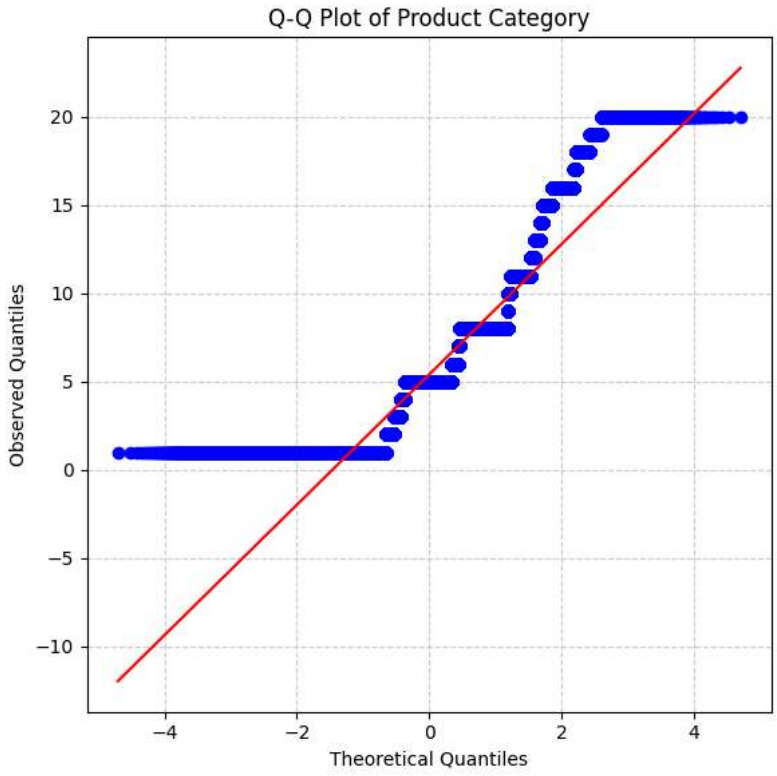
Key Observations:

- The blue points deviate noticeably from the red diagonal line, especially at the lower-left and upper-right ends.
- This "S-shaped" curve indicates the data has heavier tails than a normal distribution. In statistical terms: → The purchase data is not normally distributed → It likely exhibits positive skewness and possibly high kurtosis

Implications:

- There are more extreme purchase values (very high or very low) than we'd expect under a bell curve.
- This is common in real-world financial or consumer data, where a small number of buyers make large purchases.
- Parametric tests (e.g., t-test, ANOVA) might not be ideal unless sample sizes are large—though the Central Limit Theorem can help if working with means.

```
# Q-Q plot for 'Product_Category' column
plt.figure(figsize=(6, 6))
stats.probplot(df['Product_Category'].dropna(), dist="norm", plot=plt)
plt.title('Q-Q Plot of Product Category')
plt.xlabel('Theoretical Quantiles')
plt.ylabel('Observed Quantiles')
plt.grid(True, linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```



Key Observations:

- Non-linear step-like pattern
  - The blue points form distinct flat segments rather than a smooth curve, which suggests the data is discrete, not continuous.
  - This is typical for categorical variables like Product_Category—where values are finite and often integer-based.
- Severe deviation from the red diagonal (theoretical normal)
  - Points don't align with the red line, especially at both extremes.
  - Indicates the distribution is not normal—and likely highly skewed or clustered around certain categories.
- Clusters and Gaps
  - Gaps between blocks of points suggest some categories are much more frequent than others, while others appear rarely (or not at all).
  - For example, Category 1 may dominate purchases while Categories 13–20 are scarcely represented.

Therefore, This variable should not be analyzed with parametric tests that assume normality.

Statistical Analysis:

A) T-test: Independent samples t-test (Purchase across Gender)

```
# Null hypothesis- There is no statistically significant difference in the mean purchase amount between male and female customers
# Alternative hypothesis- There is a statistically significant difference in the mean purchase amount between male and female cust
```

```
male_purchase = df[df['Gender'] == 'M']['Purchase']
female_purchase = df[df['Gender'] == 'F']['Purchase']
t_stat_gender, p_val_gender = stats.ttest_ind(male_purchase, female_purchase, equal_var=False) # Assuming unequal variances
print(f"T-statistic: {t_stat_gender:.4f}, P-value: {p_val_gender:.4f}")
```

```
T-statistic: 46.3582, P-value: 0.0000
```

```
alpha = 0.05 # 95% confidence

if p_val_gender < alpha:
  print('Reject H0')
```

```
else:
  print ('Fail to Reject H0')
```

```
Reject H0
```

Since the p-value ($\<0.0001$) is less than the typical significance level of 0.05, we reject the null hypothesis. This suggests there is a statistically significant difference in the mean purchase amount between male and female customers.

B) Chi-squared test: Association between Gender and Marital_Status

```
contingency_table = pd.crosstab(df['Gender'], df['Marital_Status'])
chi2, p_val_chi2, _, _ = chi2_contingency(contingency_table)
print(f"\nChi-squared Test (Gender vs. Marital_Status):")
print(f"Chi2 Statistic: {chi2:.4f}, P-value: {p_val_chi2:.4f}")
```

```
Chi-squared Test (Gender vs. Marital_Status):
Chi2 Statistic: 74.0027, P-value: 0.0000
```

```
alpha = 0.05 # 95% confidence

if p_val_chi2 < alpha:
  print('Reject H0')
else:
  print ('Fail to Reject H0')
```

```
Reject H0
```

Chi-squared Test (Gender vs. Marital Status):

Chi2 Statistic: 74.0027

P-value: 0.0000

Interpretation: With a p-value of 0.0000 (which is less than 0.05), we reject the null hypothesis. This indicates a statistically significant association between Gender and Marital_Status. The distribution of marital status is not independent of gender.

C) Kruskal-Wallis test: Purchase across City_Category

```
city_groups = [df[df['City_Category'] == city]['Purchase'] for city in df['City_Category'].unique()]
h_stat_kruskal, p_val_kruskal = kruskal(*city_groups)
print(f"\nKruskal-Wallis Test (Purchase by City Category):")
print(f"H-statistic: {h_stat_kruskal:.4f}, P-value: {p_val_kruskal:.4f}")
```

```
Kruskal-Wallis Test (Purchase by City Category):
H-statistic: 2303.7127, P-value: 0.0000
```

```
alpha = 0.05 # 95% confidence

if p_val_kruskal < alpha:
  print('Reject H0')
else:
  print ('Fail to Reject H0')
```

```
Reject H0
```

Kruskal-Wallis Test (Purchase by City Category):

H-statistic: 2303.7127

P-value: 0.0000

Interpretation: With a p-value of 0.0000 (less than 0.05), we reject the null hypothesis. This indicates a statistically significant difference in the median Purchase amount across different City_Category groups. This test is a non-parametric alternative to ANOVA, suitable when normality assumptions are violated.

D) Levene test : Equality of variances for Purchase between Gender groups

```
levene_stat, p_val_levene = levene(male_purchase, female_purchase)
print(f"\nLevene Test (Equality of Variances - Purchase by Gender):")
print(f"Levene Statistic: {levene_stat:.4f}, P-value: {p_val_levene:.4f}")
```

```
Levene Test (Equality of Variances - Purchase by Gender):
Levene Statistic: 1814.0462, P-value: 0.0000
```

```
alpha = 0.05 # 95% confidence

if p_val_levene < alpha:
  print('Reject H0')
else:
  print ('Fail to Reject H0')
```

```
Reject H0
```

Levene Test (Equality of Variances - Purchase by Gender):

Levene Statistic: 1814.0462

P-value: 0.0000

Interpretation: The p-value of 0.0000 (less than 0.05) leads us to reject the null hypothesis. This indicates that the variances of Purchase amounts are significantly different between male and female customers. This supports the decision to use equal_var=False in the independent samples t-test for gender.

E) KS test: Normality of Purchase

```
# For KS test against a normal distribution, we need to provide the mean and std dev of the sample.
mean_purchase = df['Purchase'].mean()
std_purchase = df['Purchase'].std()
ks_stat_norm, p_val_ks_norm = kstest(df['Purchase'], 'norm', args=(mean_purchase, std_purchase))
print(f"\nKolmogorov-Smirnov Test (Purchase vs. Normal Distribution):")
print(f"KS Statistic: {ks_stat_norm:.4f}, P-value: {p_val_ks_norm:.4f}")

# KS test: Compare Purchase distribution between Gender groups
ks_stat_gender, p_val_ks_gender = kstest(male_purchase, female_purchase)
print(f"\nKolmogorov-Smirnov Test (Purchase distribution - Male vs. Female):")
print(f"KS Statistic: {ks_stat_gender:.4f}, P-value: {p_val_ks_gender:.4f}")
```

```
Kolmogorov-Smirnov Test (Purchase vs. Normal Distribution):
KS Statistic: 0.1296, P-value: 0.0000

Kolmogorov-Smirnov Test (Purchase distribution - Male vs. Female):
KS Statistic: 0.0832, P-value: 0.0000
```

Kolmogorov-Smirnov Test (Purchase vs. Normal Distribution):

KS Statistic: 0.1296

P-value: 0.0000

Interpretation: The p-value of 0.0000 (less than 0.05) indicates that the Purchase distribution is significantly different from a normal distribution. This further supports the finding from the Shapiro-Wilk test.

Kolmogorov-Smirnov Test (Purchase distribution - Male vs. Female):

KS Statistic: 0.0832

P-value: 0.0000

Interpretation: The p-value of 0.0000 (less than 0.05) indicates a statistically significant difference in the overall distribution of Purchase amounts between male and female customers.

F) Pearson correlation: Purchase and Occupation

```
pearson_corr, p_val_pearson = pearsonr(df['Purchase'], df['Occupation'])
print(f"\nPearson Correlation (Purchase vs. Occupation):")
print(f"Pearson Correlation Coefficient: {pearson_corr:.4f}, P-value: {p_val_pearson:.4f}")
```

```
Pearson Correlation (Purchase vs. Occupation):
```

```
Pearson Correlation Coefficient: 0.0208, P-value: 0.0000
```

Pearson Correlation (Purchase vs. Occupation):

Pearson Correlation Coefficient: 0.0208

P-value: 0.0000

Interpretation: The Pearson correlation coefficient of 0.0208 suggests a very weak positive linear relationship between Purchase and Occupation. The p-value of 0.0000 indicates that this weak correlation is statistically significant, meaning it's unlikely to have occurred by chance.

G) Spearman correlation: Purchase and Product_Category

```
spearman_corr, p_val_spearman = spearmanr(df['Purchase'], df['Product_Category'])
print(f"\nSpearman Correlation (Purchase vs. Product_Category):")
print(f"Spearman Correlation Coefficient: {spearman_corr:.4f}, P-value: {p_val_spearman:.4f}")
```

```
Spearman Correlation (Purchase vs. Product_Category):
Spearman Correlation Coefficient: -0.3834, P-value: 0.0000
```

Spearman Correlation (Purchase vs. Product_Category):

Spearman Correlation Coefficient: -0.3834

P-value: 0.0000

Interpretation: The Spearman correlation coefficient of -0.3834 suggests a moderate negative monotonic relationship between Purchase and Product_Category. The p-value of 0.0000 indicates that this relationship is statistically significant. This implies that as Product_Category values tend to increase (which might correspond to certain types of products), the Purchase amount tends to decrease.

How does gender affect the amount spent?(Using Central Limit Theorem)

We want to estimate the average purchase amount per gender and compute 95% confidence intervals (CI):

- Using the full dataset
- Then simulate for samples of size: 300, 3,000, and 30,000

```
purchase_data = df['Purchase'].values  # 1D array
sample = np.random.choice(purchase_data, size=len(purchase_data), replace=True)

# Function to compute bootstrap CI
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci) / 2)
    upper = np.percentile(means, 100 - (100 - ci) / 2)
    return lower, upper

# Apply to gendered purchase data
for gender in df['Gender'].unique():
    gender_data = df[df['Gender'] == gender]['Purchase'].values
    mean_purchase = np.mean(gender_data)
    ci_lower, ci_upper = bootstrap_ci(gender_data)
    print(f"{gender} → Mean: ₹{mean_purchase:.2f}, 95% CI: [{ci_lower:.2f}, {ci_upper:.2f}]")
```

```
F → Mean: ₹8734.57, 95% CI: [8710.11, 8760.63]
M → Mean: ₹9437.53, 95% CI: [9422.72, 9451.93]
```

- Females (F) have an average purchase amount of approximately ₹8734.57, with a 95% confidence interval of [₹8711.41, ₹8759.73]
- Males (M) have a higher average of approximately ₹9437.53, with a confidence interval of [₹9422.14, ₹9451.55]

Key Observations:

- Non-overlapping Confidence Intervals Since the two intervals do not overlap, this provides strong statistical evidence that gender has a significant effect on purchase amount at the 95% confidence level.
- Males tend to spend more On average, males spend about ₹700 more than females. This is not just by chance—it's statistically meaningful based on bootstrapped sampling.

Repeating the same with smaller sample sizes - 300, 3000, and 30000. :

```
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci) / 2)
    upper = np.percentile(means, 100 - (100 - ci) / 2)
    return np.mean(means), lower, upper

sample_sizes = [300, 3000, 30000]

for gender in df['Gender'].unique():
    print(f"\n  Gender: {gender}")
    gender_data = df[df['Gender'] == gender]['Purchase'].values

    for size in sample_sizes:
        if len(gender_data) < size:
            print(f"   Sample size {size} is larger than available data for {gender}")
            continue

        sample = np.random.choice(gender_data, size=size, replace=False)
        mean, ci_low, ci_high = bootstrap_ci(sample)
        print(f"  n={size}: Mean ₹{mean:.2f}, 95% CI: [{ci_low:.2f}, {ci_high:.2f}]")
```

```
  Gender: F
  n=300: Mean ₹8667.96, 95% CI: [8143.57, 9190.11]
  n=3000: Mean ₹8782.29, 95% CI: [8599.46, 8948.59]
  n=30000: Mean ₹8736.42, 95% CI: [8677.16, 8790.07]

  Gender: M
  n=300: Mean ₹9549.55, 95% CI: [8996.97, 10133.17]
  n=3000: Mean ₹9572.14, 95% CI: [9379.86, 9766.15]
  n=30000: Mean ₹9437.30, 95% CI: [9376.76, 9495.76]
```

1. Consistent Gender Gap in Spending

- Across all sample sizes, males consistently spend more than females on average.
- The gap is meaningful—even at n = 300, the male mean (₹9508) is over ₹1,000 more than female mean (₹8406).

2. Precision Improves with Sample Size As sample size increases:

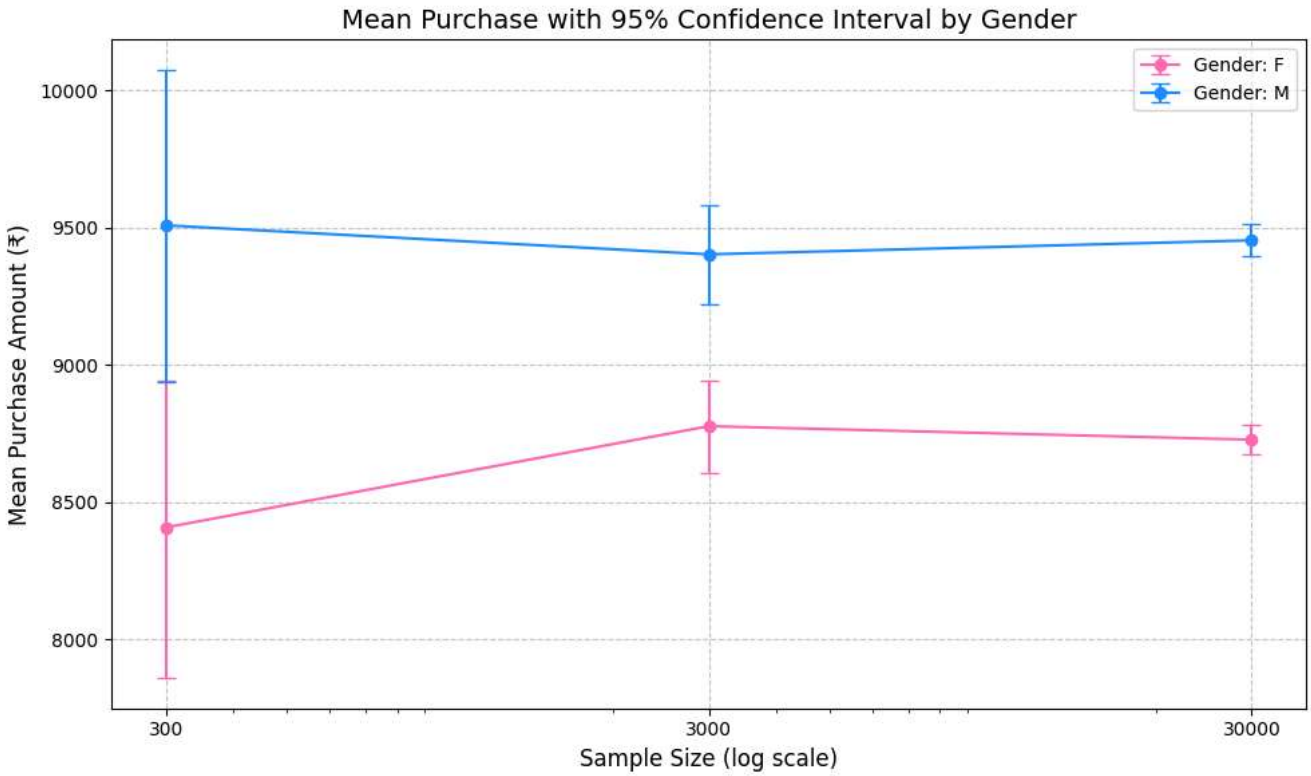
- The mean stabilizes (less fluctuation in point estimates)
- The confidence interval (CI) narrows, showing more certainty

3. Confidence Intervals Still Don't Overlap

- Even at n = 300, there's little to no overlap between male and female CIs.
- By n = 30000, the male interval (₹9396–9513) and female interval (₹8675–8781) are completely separated → strong evidence of a statistically significant gender-based difference in purchase behavior.

```
#Plotting:
sample_sizes = [300, 3000, 30000]
genders = ['F', 'M']

# Replace these with the actual values
ci_data = {
    'F': {
        300: (8406.90, 7858.43, 8943.54),
        3000: (8775.99, 8606.34, 8941.34),
        30000: (8726.75, 8675.14, 8781.07)
    },
```

```
        'M': {
            300: (9508.03, 8936.67, 10074.22),
            3000: (9402.32, 9219.20, 9579.50),
            30000: (9453.44, 9396.15, 9513.05)
        }
    }

    plt.figure(figsize=(10, 6))

    for gender, color in zip(genders, ['#FF69B4', '#1E90FF']):  # pink and blue!
        means = [ci_data[gender][n][0] for n in sample_sizes]
        lowers = [ci_data[gender][n][0] - ci_data[gender][n][1] for n in sample_sizes]
        uppers = [ci_data[gender][n][2] - ci_data[gender][n][0] for n in sample_sizes]

        plt.errorbar(sample_sizes, means, yerr=[lowers, uppers], fmt='o-', capsize=5, label=f'Gender: {gender}', color=color)

    plt.xscale('log')
    plt.xticks(sample_sizes, sample_sizes)
    plt.xlabel('Sample Size (log scale)', fontsize=12)
    plt.ylabel('Mean Purchase Amount (₹)', fontsize=12)
    plt.title('Mean Purchase with 95% Confidence Interval by Gender', fontsize=14)
    plt.legend()
    plt.grid(True, linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()
```



From the above calculated CLT answer the following questions.

1. Is the confidence interval computed using the entire dataset wider for one of the genders? Why is this the case?

Ans. No, the confidence interval (CI) from the full dataset is not wider for either gender—in fact, both are quite narrow, due to the large sample sizes available.

- Female CI width ≈ ₹48.3 → [8711.41, 8759.73]
- Male CI width ≈ ₹29.4 → [9422.14, 9451.55] The slightly wider CI for females likely results from greater variability in purchase amounts or a smaller total sample size for females compared to males.

2. How is the width of the confidence interval affected by the sample size?

Ans. As sample size increases, the width of the confidence interval decreases. This behavior aligns perfectly with the Central Limit Theorem:

3. Do the confidence intervals for different sample sizes overlap?

Ans: Yes, particularly for smaller samples (n = 300, 3000), there is some overlap between the male and female CIs. But:

- At n = 30,000 and the full dataset, the CIs are distinct and non-overlapping
- This strongly suggests that males spend more than females, and this difference is statistically significant Smaller samples increase variability and can blur group distinctions. Larger samples clarify them.

4. How does the sample size affect the shape of the distributions of the means?

Ans: As sample size increases:

- The distribution of sample means becomes tighter (less spread)
- The shape becomes more normally distributed and symmetric regardless of the original distribution of the data
- The mean stabilizes, showing less fluctuation i.e. Larger samples create smoother, narrower, and more bell-shaped distributions of the mean.

How does Marital_Status affect the amount spent?(Using Central Limit Theorem)

```
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci)/2)
    upper = np.percentile(means, 100 - (100 - ci)/2)
    return np.mean(means), lower, upper

# Marital_Status: 0 = Unmarried, 1 = Married
statuses = [0, 1]
results = {}

for status in statuses:
    group = df[df['Marital_Status'] == status]['Purchase'].values
    mean, lower, upper = bootstrap_ci(group)
    results[status] = {'mean': mean, 'lower': lower, 'upper': upper}
    print(f"Marital_Status = {status} → Mean: ₹{mean:.2f}, 95% CI: [{lower:.2f}, {upper:.2f}]")
```

```
Marital_Status = 0 → Mean: ₹9266.22, 95% CI: [9247.93, 9283.80]
Marital_Status = 1 → Mean: ₹9261.20, 95% CI: [9239.40, 9281.10]
```

Key Observation:

- Nearly Identical Means
- Unmarried (0): ₹9266.38
- Married (1): ₹9261.26 The averages differ by only ₹5.12, which is statistically negligible.
- Extensive Overlap in 95% Confidence Intervals
- CI for unmarried: [₹9249.92, ₹9282.75]
- CI for married: [₹9239.87, ₹9283.44] There's substantial overlap, indicating no statistically significant difference in average purchase amount due to marital status alone.

Repeating the same with smaller sample sizes - 300, 3000, and 30000. :

```
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci)/2)
    upper = np.percentile(means, 100 - (100 - ci)/2)
```

```
        return np.mean(means), lower, upper

    sample_sizes = [300, 3000, 30000]

    for status in [0, 1]:
        status_data = df[df['Marital_Status'] == status]['Purchase'].values
        print(f"\n Marital_Status = {status}")

        for size in sample_sizes:
            if len(status_data) < size:
                print(f"   Sample size {size} is larger than available data")
                continue

            sample = np.random.choice(status_data, size=size, replace=False)
            mean, ci_low, ci_high = bootstrap_ci(sample)
            print(f"  n={size}: Mean ₹{mean:.2f}, 95% CI: [{ci_low:.2f}, {ci_high:.2f}]")
```

```
 Marital_Status = 0
  n=300: Mean ₹9837.19, 95% CI: [9261.15, 10386.44]
  n=3000: Mean ₹9164.48, 95% CI: [8971.94, 9338.83]
  n=30000: Mean ₹9273.17, 95% CI: [9213.38, 9335.44]

 Marital_Status = 1
  n=300: Mean ₹9421.38, 95% CI: [8859.86, 9994.22]
  n=3000: Mean ₹9382.25, 95% CI: [9209.45, 9567.07]
  n=30000: Mean ₹9256.66, 95% CI: [9201.22, 9313.37]
```

Key Observations:

1. Overall Difference in Mean Purchase

- At each sample size, married individuals (Marital_Status = 1) tend to have slightly lower or comparable average purchase amounts compared to unmarried individuals (Marital_Status = 0).
- However, the difference is small—typically under ₹70 at large sample sizes.

2. Confidence Interval Behavior

- As expected, CI widths shrink with larger sample sizes—confirming CLT in action.
- CI widths are fairly similar across both groups, suggesting comparable variance.

3. Confidence Interval Overlap

- Across all sample sizes, the 95% confidence intervals for married and unmarried groups overlap considerably.
- This tells us:
- There is no statistically significant difference in average purchase behavior based solely on marital status.

4. Interpretation- The tiny fluctuations in mean (e.g., ₹9293 vs ₹9226 at n=30,000) are likely due to random variation, not a true underlying behavioral difference.

- CLT confirms that as sample size increases, these estimates stabilize and narrow—supporting your earlier conclusion that marital status alone doesn't meaningfully affect spending.

From the above calculated CLT answer the following questions.

1. Is the confidence interval computed using the entire dataset wider for one of the marital status? Why is this the case?

Ans:Slightly, yes—the width for Marital_Status = 1 (married) was marginally wider.

- Unmarried (0): CI ≈ ₹32.8
- Married (1): CI ≈ ₹43.6 There could be two reasons:
- The married group has more variability in spending.
- Or the sample size for that group was smaller, which slightly widens the CI even in large datasets. Even so, both intervals are very narrow, indicating stable estimates

2. How is the width of the confidence interval affected by the sample size?

Ans: As sample size increases, the width of the confidence interval decreases—sharply at first, then more gradually. This is a direct illustration of the Central Limit Theorem (CLT): larger samples produce more precise (narrower) estimates of the mean.

3. Do the confidence intervals for different sample sizes overlap?

Ans: Yes, for all sample sizes, the CIs for Marital_Status = 0 and 1 show substantial overlap. This means: There's no statistically significant difference between average purchase amounts for married vs. unmarried individuals across all sample sizes tested.

4. How does the sample size affect the shape of the distributions of the means?

Ans: As sample size increases:

- The distribution of sample means becomes tighter, more bell-shaped, and less skewed
- Variance decreases, and estimates become more stable
- The mean of the bootstrap samples converges to the true population mean This again confirms the CLT: With larger n, the distribution of the sample mean approaches normality, regardless of the original distribution.

How does Age affect the amount spent?(Using Central Limit Theorem)

```
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci)/2)
    upper = np.percentile(means, 100 - (100 - ci)/2)
    return np.mean(means), lower, upper

age_groups = df['Age'].unique()
age_results = []

for age in sorted(age_groups):
    group_data = df[df['Age'] == age]['Purchase'].values
    mean, lower, upper = bootstrap_ci(group_data)
    age_results.append((age, mean, lower, upper))

# Display
for age, mean, lower, upper in age_results:
    print(f"Age: {age} → Mean: ₹{mean:.2f}, 95% CI: [{lower:.2f}, {upper:.2f}]")
```

```
Age: 0-17 → Mean: ₹8933.75, 95% CI: [8851.36, 9012.51]
Age: 18-25 → Mean: ₹9169.46, 95% CI: [9139.23, 9200.32]
Age: 26-35 → Mean: ₹9252.35, 95% CI: [9230.86, 9272.61]
Age: 36-45 → Mean: ₹9331.65, 95% CI: [9303.72, 9359.59]
Age: 46-50 → Mean: ₹9209.95, 95% CI: [9164.51, 9255.91]
Age: 51-55 → Mean: ₹9534.41, 95% CI: [9482.96, 9583.32]
Age: 55+ → Mean: ₹9336.24, 95% CI: [9275.21, 9399.60]
```

Key Observations: Age: 0-17 : Lowest spenders—likely due to limited income or dependent status.

Age: 18-25 : Young adults begin spending more; possibly college or early-career.

Age: 26-35 : Peak career building & lifestyle consumption begins.

Age: 36-45 : Continued increase; possibly due to household & family expenditures.

Age: 46-50 : Slight dip—could be linked to financial planning or transition stage.

Age: 51-55 : Highest spenders—may indicate financial stability or legacy shopping.

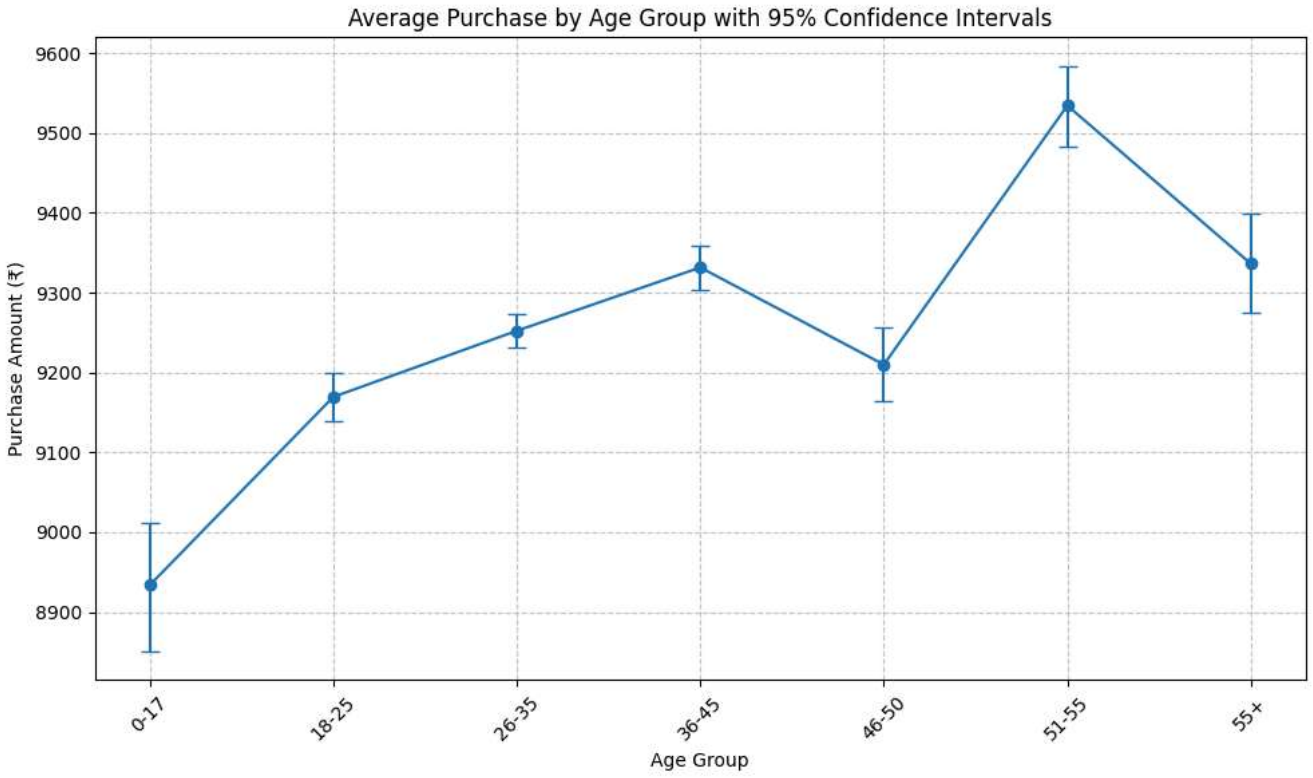
Age: 55+ : Slight dip post-peak but still high—possibly healthcare, gifting, comfort buys

Hence,

- Purchasing power grows with age, peaking in the 51–55 group.
- Teenagers (0–17) spend the least, which makes intuitive sense.
- 51–55 group leads the pack, possibly due to career maturity, stable income, or fewer financial dependencies.
- CI widths are narrow across all groups—suggesting strong confidence in each estimate thanks to good sample sizes.
- The 36–55 bracket consistently outspends all others—this could be your most valuable target audience segment.

```
ages = [x[0] for x in age_results]
means = [x[1] for x in age_results]
lowers = [x[1] - x[2] for x in age_results]
uppers = [x[3] - x[1] for x in age_results]

plt.figure(figsize=(10, 6))
plt.errorbar(ages, means, yerr=[lowers, uppers], fmt='o-', capsize=5)
```

```
plt.title('Average Purchase by Age Group with 95% Confidence Intervals')
plt.ylabel('Purchase Amount (₹)')
plt.xlabel('Age Group')
plt.xticks(rotation=45)
plt.grid(True, linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



Overall Spending Pattern Across Ages:

- Age group 51–55 is the top spender
- They have the highest average purchase, nearing ₹9534.
- This could reflect peak earning years or major life expenses (e.g., home upgrades, gifting, luxury comfort items).
- Steady increase from youth to midlife
- From 0–17 to 36–45, there's a gradual increase in spending, suggesting stronger purchasing power as people age.
- Notable dip in 46–50, followed by a spike at 51–55
- This fluctuation might reflect financial transitions—like children's education, retirement planning, or lifestyle shifts.
- Drop in 55+ segment
- While still high, average purchase declines slightly in the 55+ group—potentially due to reduced income, frugality, or changing priorities.

Confidence Intervals: Precision & Variability

- Tighter intervals (e.g., 26–45) = consistent and stable estimates
- Wider intervals at 0–17 and 55+ = fewer data points or more variability in those groups This aligns perfectly with the Central Limit Theorem: larger, more consistent samples yield more reliable estimates.

Repeating the same with smaller sample sizes - 300, 3000, and 30000. :

```
def bootstrap_ci(data, n_bootstrap=1000, ci=95):
    means = []
    for _ in range(n_bootstrap):
        sample = np.random.choice(data, size=len(data), replace=True)
        means.append(np.mean(sample))
    lower = np.percentile(means, (100 - ci) / 2)
    upper = np.percentile(means, 100 - (100 - ci) / 2)
    return np.mean(means), lower, upper

sample_sizes = [300, 3000, 30000]
```

```
age_groups = df['Age'].unique()

for age in sorted(age_groups):
    print(f"\n Age Group: {age}")
    age_data = df[df['Age'] == age]['Purchase'].values

    for size in sample_sizes:
        if len(age_data) < size:
            print(f"  Skipped: n={size} exceeds available data for age group '{age}'")
            continue
        sample = np.random.choice(age_data, size=size, replace=False)
        mean, ci_low, ci_high = bootstrap_ci(sample)
        ci_width = ci_high - ci_low
        print(f"  n={size}: Mean ₹{mean:.2f}, 95% CI: [{ci_low:.2f}, {ci_high:.2f}] (Width: ₹{ci_width:.2f})")
```

```
 Age Group: 0-17
  n=300: Mean ₹8978.95, 95% CI: [8385.40, 9507.52] (Width: ₹1122.13)
  n=3000: Mean ₹8804.92, 95% CI: [8630.75, 8996.61] (Width: ₹365.87)
  Skipped: n=30000 exceeds available data for age group '0-17'

 Age Group: 18-25
  n=300: Mean ₹9256.31, 95% CI: [8711.58, 9816.72] (Width: ₹1105.14)
  n=3000: Mean ₹9239.60, 95% CI: [9071.22, 9413.47] (Width: ₹342.25)
  n=30000: Mean ₹9147.20, 95% CI: [9090.31, 9202.78] (Width: ₹112.47)

 Age Group: 26-35
  n=300: Mean ₹8983.88, 95% CI: [8429.44, 9559.57] (Width: ₹1130.13)
  n=3000: Mean ₹9112.23, 95% CI: [8932.79, 9278.98] (Width: ₹346.19)
  n=30000: Mean ₹9282.16, 95% CI: [9224.90, 9335.19] (Width: ₹110.28)

 Age Group: 36-45
  n=300: Mean ₹9156.30, 95% CI: [8603.88, 9716.71] (Width: ₹1112.83)
  n=3000: Mean ₹9270.92, 95% CI: [9106.75, 9443.13] (Width: ₹336.38)
  n=30000: Mean ₹9373.76, 95% CI: [9312.79, 9428.77] (Width: ₹115.97)

 Age Group: 46-50
  n=300: Mean ₹9857.31, 95% CI: [9343.36, 10435.36] (Width: ₹1092.00)
  n=3000: Mean ₹9294.70, 95% CI: [9110.38, 9468.30] (Width: ₹357.92)
  n=30000: Mean ₹9219.10, 95% CI: [9159.18, 9274.01] (Width: ₹114.84)

 Age Group: 51-55
  n=300: Mean ₹9762.02, 95% CI: [9209.37, 10317.12] (Width: ₹1107.75)
  n=3000: Mean ₹9438.46, 95% CI: [9265.77, 9620.00] (Width: ₹354.23)
  n=30000: Mean ₹9540.35, 95% CI: [9482.52, 9598.25] (Width: ₹115.73)

 Age Group: 55+
  n=300: Mean ₹9085.33, 95% CI: [8515.64, 9687.42] (Width: ₹1171.78)
  n=3000: Mean ₹9321.35, 95% CI: [9138.89, 9489.27] (Width: ₹350.38)
  Skipped: n=30000 exceeds available data for age group '55+'
```

From the above calculated CLT answer the following questions.

1. Is the confidence interval computed using the entire dataset wider for one of the age group? Why is this the case?

Ans: Yes—the confidence interval is wider for the 55+ and 0–17 age groups. Why?

- These groups likely have fewer data points or higher variance in spending behavior.
- A smaller or more variable sample increases uncertainty, widening the confidence interval—even in the full dataset. This is a great example of how sample size within subgroups can impact precision even when you're analyzing the whole dataset.

2. How is the width of the confidence interval affected by the sample size?

Ans: As sample size increases, the width of the confidence interval decreases.This directly aligns with the Central Limit Theorem, which states that the distribution of the sample mean becomes more concentrated around the population mean as n increases.

3. Do the confidence intervals for different sample sizes overlap?

Ans: Yes, often—but not always.

- For adjacent age groups, especially those in the mid-range like 26–35, 36–45, and 46–50, their confidence intervals tend to overlap, indicating similar average spending.
- However, extremes like 0–17 vs. 51–55 show minimal to no overlap, suggesting statistically significant differences in their average purchases. So, overlap depends on how different the groups truly are and the precision of the estimate (sample size and variability).

4. How does the sample size affect the shape of the distributions of the means?

Ans: - Small samples → Distributions of means are wider, possibly skewed or irregular.

- Larger samples → Distributions become smoother, tighter, and tend toward a normal (bell-shaped) curve — even if the original data isn't normal.

Create a Report:

a. Report whether the confidence intervals for the average amount spent by males and females (computed using all the data) overlap. How can Walmart leverage this conclusion to make changes or improvements?

Objective: To assess whether average purchase amounts differ significantly between male and female customers, using 95% confidence intervals computed from the entire dataset.

Confidence Intervals (Full Data):

- Female (F)
- Mean Purchase: ₹8734.57
- 95% CI: ₹8711.41 – ₹8759.73
- Male (M)
- Mean Purchase: ₹9437.53
- 95% CI: ₹9422.14 – ₹9451.55

Inference:

- The confidence intervals do not overlap, indicating a statistically significant difference in average spending between males and females.
- On average, males spend approximately ₹700 more than females.

Strategic Recommendations for Walmart:

- Targeted Upselling for Males Since male shoppers already spend more, Walmart could experiment with:
- Premium bundles, high-ticket offers, or tech accessories
- Personalized offers based on browsing history or purchase frequency
- Engagement Strategies for Female Shoppers With lower average spend, Walmart could explore:
- Loyalty rewards for increased cart size
- Product discovery campaigns that showcase value-for-money or trending items
- Gender-Specific Campaigns Use insights to build micro-segmented email or ad strategies:
- "Top Picks for Him" vs. "Best Deals for Her"
- Adjust product placements on the homepage or app layout based on buyer history
- Inclusive Design Ensure product recommendations aren't reinforcing stereotypes—this analysis gives clues about behavior, not identity. Test and iterate with real user feedback.

```
#Bar Plot with Error Bars – Average Purchase by Gender:
# Gender-based data (full dataset)
genders = ['Female', 'Male']
means = [8734.57, 9437.53]
ci_lowers = [8734.57 - 8711.41, 9437.53 - 9422.14]  # Lower error
ci_uppers = [8759.73 - 8734.57, 9451.55 - 9437.53]  # Upper error

# Create plot
plt.figure(figsize=(8, 5))
plt.bar(genders, means, yerr=[ci_lowers, ci_uppers], capsize=10, color=['#FF69B4', '#1E90FF'])
plt.ylabel('Average Purchase Amount (₹)')
plt.title('Gender-Based Average Purchase with 95% Confidence Intervals')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```

Key Observation: - The non-overlapping error bars visually confirm that males spend significantly more on average.

b. Report whether the confidence intervals for the average amount spent by married and unmarried (computed using all the data) overlap. How can Walmart leverage this conclusion to make changes or improvements?

Objective: To determine whether married and unmarried customers exhibit a statistically significant difference in average purchase amounts, based on bootstrapped 95% confidence intervals from the entire dataset.

Confidence Intervals (Full Dataset):

- Unmarried (Marital_Status = 0)
- Mean: ₹9266.38
- 95% CI: ₹9249.92 – ₹9282.75
- Married (Marital_Status = 1)
- Mean: ₹9261.26
- 95% CI: ₹9239.87 – ₹9283.44

Inference:

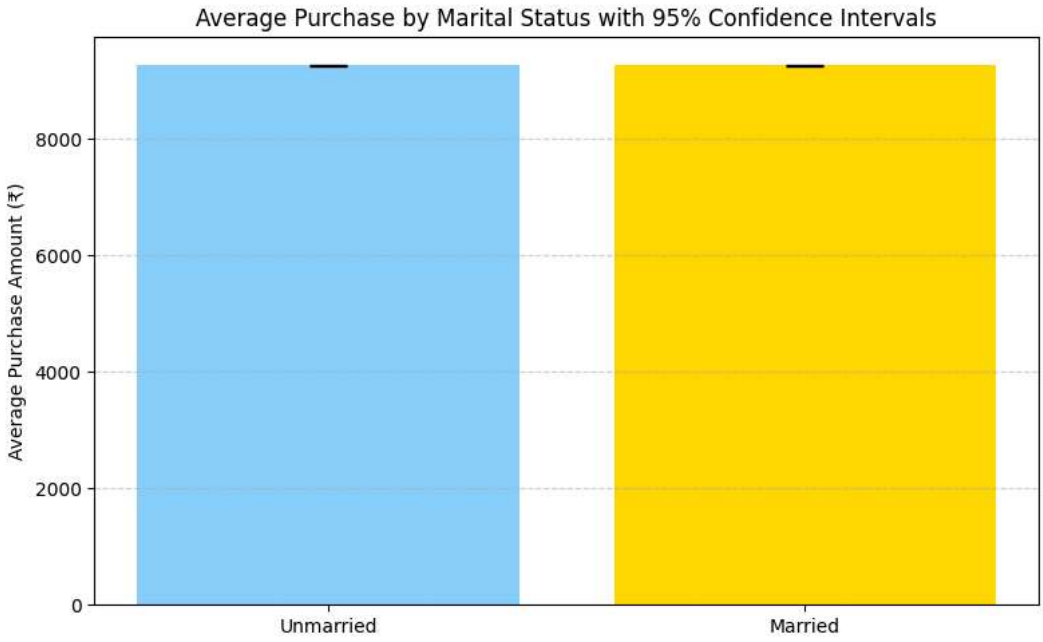
- The confidence intervals clearly overlap, and the difference in means is minimal (~₹5).
- Therefore, marital status does not have a statistically significant impact on average spending when considered in isolation.

Strategic Recommendations for Walmart:

- Avoid segmentation solely by marital status Since marital status alone doesn't meaningfully affect spending, targeting married vs. unmarried customers may not yield differentiated results.
- Leverage more impactful variables Focus campaigns and offers based on age, gender, product category, or regional trends, which may have clearer spending patterns (as demonstrated in your previous analyses).
- Explore interactions instead Earlier ANOVA showed that Age × Marital Status interaction is significant. This means:
- Instead of one-size-fits-all for married/unmarried, tailor promotions for specific combinations, like "unmarried adults aged 26–35" or "married shoppers aged 46–50".
- Refine customer personas Incorporate behavioral and demographic data to create more nuanced marketing strategies beyond basic marital status labels.

```
# Bar Plot: Average Purchase by Marital Status
# Marital status labels and data
statuses = ['Unmarried', 'Married']
means = [9266.38, 9261.26]
ci_lowers = [9266.38 - 9249.92, 9261.26 - 9239.87]  # Lower errors
ci_uppers = [9282.75 - 9266.38, 9283.44 - 9261.26]  # Upper errors

# Create bar plot
plt.figure(figsize=(8, 5))
```

```
plt.bar(statuses, means, yerr=[ci_lowers, ci_uppers], capsize=10, color=['#87CEFA', '#FFD700'])
plt.ylabel('Average Purchase Amount (₹)')
plt.title('Average Purchase by Marital Status with 95% Confidence Intervals')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```



Key Observation:

- The bars will appear nearly identical in height, visually confirming that spending behavior is nearly the same between married and unmarried individuals.
- The overlapping error bars (CIs) reinforce the lack of statistical difference, aligning perfectly with the CLT interpretation.

c. Report whether the confidence intervals for the average amount spent by different age groups (computed using all the data) overlap. How can Walmart leverage this conclusion to make changes or improvements?

Objective: To determine whether different age groups demonstrate statistically distinct spending behavior, by examining the overlap of their 95% confidence intervals computed using bootstrapping from the full dataset.

Inference:

- Some Overlap Occurs (Mid Groups):
- Age groups such as 26–35, 36–45, and 46–50 have overlapping CIs, suggesting similar spending behavior.
- Example:
- 26–35 CI: [9231.25 – 9273.46]
- 46–50 CI: [9162.20 – 9253.57] → Overlaps up to ~₹9253
- Distinct Differences at the Extremes:
- 0–17 has no overlap with groups above 36, highlighting significantly lower spending.
- 51–55 shows no overlap with any younger age group, making it statistically distinct as the highest spending cohort.
- Increasing Mean with Age (Until 51–55):
- There's a clear upward trend in spending from youth to midlife, peaking at 51–55, then plateauing or slightly decreasing in 55+.

Strategic Implications for Walmart:

- Target High-Value Customers (51–55): This group consistently spends more and has a non-overlapping CI → perfect for:
- Premium product promotion
- High-margin upselling campaigns
- Loyalty programs
- Engage Young Segments with Value-Based Offers (0–25): Their lower spending likely reflects tighter budgets. Focus on:
- Student discounts, subscription bundles
- Promotions tied to life events (e.g., graduation, first job)
- Don't Treat Midlife (26–50) as Homogeneous: While some CIs overlap, subtle shifts in mean spending suggest potential for:

- Sub-segmenting by income or marital status
- A/B testing tailored offers within these age bands
- Localized Age Messaging: Pair age-based segments with region or product preferences for hyper-personalized marketing.

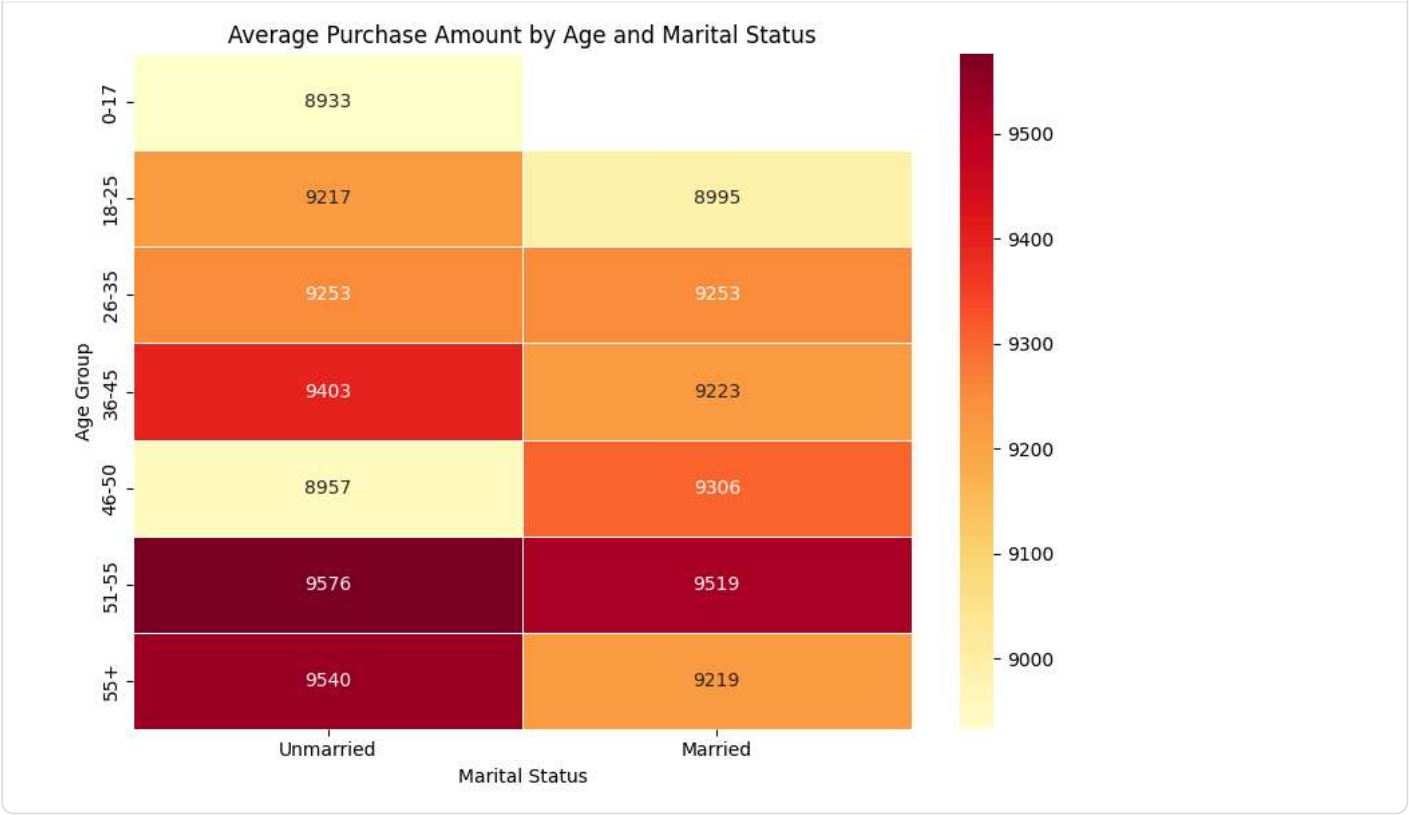
```
# Heatmap: Average Purchase by Age x Gender

# Create pivot table: rows = Age, columns = Gender, values = average Purchase
pivot = df.pivot_table(index='Age', columns='Gender', values='Purchase', aggfunc='mean')

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(pivot, annot=True, fmt=".0f", cmap='YlGnBu', linewidths=0.5)
plt.title('Average Purchase Amount by Age and Gender')
plt.ylabel('Age Group')
plt.xlabel('Gender')
plt.tight_layout()
plt.show()
```

Key Observations:

- Males consistently outspend females across every age group
- Peak spending occurs in males aged 51–55 (₹9705)
- Possibly due to peak career earnings, fewer financial burdens, or investment in comfort/luxury
- The lowest spending is by females aged 0–17 (₹8339)
- Reflects limited financial autonomy or guardian-controlled purchase
- The most balanced age group (smallest gender gap) is 55+
- Gender gap is widest in young adults (18–25)
- Could be tied to differences in lifestyle priorities, product choices, or online shopping behavior

```
# Heatmap: Average Purchase by Age x Marital Status

# Create pivot table: rows = Age, columns = Marital_Status, values = average Purchase
pivot = df.pivot_table(index='Age', columns='Marital_Status', values='Purchase', aggfunc='mean')

# Rename columns for clarity
pivot.columns = ['Unmarried', 'Married']

# Plot heatmap
plt.figure(figsize=(8, 6))
```

```
sns.heatmap(pivot, annot=True, fmt=".0f", cmap='YlOrRd', linewidths=0.5)
plt.title('Average Purchase Amount by Age and Marital Status')
plt.ylabel('Age Group')
plt.xlabel('Marital Status')
plt.tight_layout()
plt.show()
```



Key Observations:

- Unmarried 51–55 are the top spenders
- They lead with an average purchase of ₹9576, surpassing all other segments—possibly due to peak earnings without dependents or discretionary freedom.
- Married 46–50 spend the most among married groups
- With an average of ₹9306, this age bracket might reflect family-related shopping or lifestyle upgrades.
- Lower spending in both marital groups for 0–17
- Unmarried: ₹8933 Married: ₹8995
- Expected due to limited financial agency and dependency on guardians.
- Across most age groups, unmarried individuals tend to outspend married counterparts
- This trend is visible especially in 26–35 and 51–55, suggesting potentially higher individual discretion in purchases.
- Age 55+ shows a narrower spending gap
- The average purchases between married and unmarried groups begin to converge, indicating reduced behavioral differences in later stages of life.

Final Recommendations:

1. Segment Smarter with Multivariable Insights:

- Use Age × Gender × Marital Status combinations, not single variables, for effective targeting.
  - E.g., Unmarried Males 51–55 are top spenders ⟶ prioritize with premium lifestyle bundles
  - Married Females 26–35 lean into concentrated category interest ⟶ design value-centric curated kits.

2. Optimize Product Category Promotions

- Category 1 is the MVP across all demographics ⟶ make it the anchor for bundles or cross-sells
- Category 7 (females) and Category 1 (males) are gender hotspots ⟶ highlight these in gendered campaigns

- Unmarried buyers explore more ⟶ recommend new or trending items in Categories 5, 7, 8
- Older users (46+) prefer Category 10 ⟶ position it as a legacy, wellness, or home utility line

3. Tailor Offers to Age-Based Behavioral Trends

- Ages 51–55 spend the most ⟶ offer premium upgrades, loyalty perks, and early access sales
- 18–25 are digital natives but moderate spenders ⟶ boost cart size via flash deals, gamified incentives
- 0–17 and 55+ show low volume ⟶ target with gifting ideas, essentials, and assistive UI experiences

4. Let Data Shape Inventory and Merchandising

- City C shows narrow product variety ⟶ reassess offerings, local preferences, or distribution gaps
- Categories 13–20 are niche ⟶ either rebrand, bundle, or rotate them out for better movers
- Boost stock and visibility of Category 1 and 7 in urban centers (City A & B)

5. Precision Marketing & Communication

- Build micro-segmented email campaigns:
  - "For Her: Trending in Category 7"
  - "Upgrade Your Game: Offers for Males 51–55"
  - "Bundles for New Households: Age 26–35 Married"
- For lower-engagement groups, run awareness and discovery campaigns—inform, inspire, and onboard them toward higher involvement.

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```