TASK_3 - PAYTM E_PURCHASE DATA ANALYSIS

September 30, 2024

1 TASK 3 : PAYTM DATA ANALYTICS

2 1. Connect Notebook with MySQL Database

[2]:
```
# Import required libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mysql.connector
```

[3]:
```
# MySQL database connection details
host = 'YOUR_HOST_NAME'
user = 'YOUR_USER_NAME'
password = 'YOUR_PASSWORD'
```

[4]:
```
# CSV file path
csv_file = 'paytm_data.csv'  # Replace 'your_file.csv' with your CSV file path

# Read the first row of the CSV to extract column names
with open(csv_file, 'r') as file:
    first_line = file.readline().strip()  # Read the first line
    column_names = first_line.split(',')  # Assuming columns are comma-separated
    file.close()
```

[5]:
```
# Create MySQL connection
conn = mysql.connector.connect(host=host, user=user, password=password)
cursor = conn.cursor()
```

[6]:
```
# Name of database
database = str(input("Enter database name: "))

# Create the database if it doesn't exist
create_db_query = f"CREATE DATABASE IF NOT EXISTS {database}"
cursor.execute(create_db_query)

# Close the connection as the database is created
conn.close()
```

```
Enter database name: paytmanalytics
```

[7]:
```
# Reconnect to the newly created or existing database
conn = mysql.connector.connect(host=host, user=user, password=password,␣
 ↪database=database)
cursor = conn.cursor()
```

[8]:
```
column_names
```

[8]:
```
['S.no',
 'Name',
 'Shipping_city',
 'Category_Grouped',
 'Category',
 'Sub_category',
 'Product_Gender',
 'Segment',
 'Class',
 'Family',
 'Brand',
 'Brick',
 'Item_NM',
 'Color',
 'Size',
 'Sale_Flag',
 'Payment_Method',
 'coupon_money_effective',
 'Coupon_Percentage',
 'Quantity',
 'Cost_Price',
 'Item_Price',
 'Special_Price_effective',
 'paid_pr_effective',
 'Value_CM1',
 'Value_CM2',
 'Special_price',
 'Paid_pr']
```

[9]:
```
# Ask for table name
table_name = input("Enter table name: ")

# Create table with extracted column names
create_table_query = f"CREATE TABLE IF NOT EXISTS `{table_name}` ({', '.
 ↪join([f'`{col}` TEXT' for col in column_names])})"
cursor.execute(create_table_query)
```

```
Enter table name: paytm_table
```

[10]:
```
# Read the CSV file into a pandas DataFrame
df = pd.read_csv(csv_file)
df.head()
```

[10]:
```
   S.no             Name Shipping_city Category_Grouped          Category  \
0     1  ABHINAV CHATTER      Jabalpur           Others        SUNGLASSES
1     2    AMIT GALPHADE     Ahmedabad         Apparels  Sports Equipment
2     3  PRABHU NAMBIAPP       Chennai           Others              Bags
3     4   MALLIKARJUNA H     Bangalore         Apparels  Sports Equipment
4     5  ANUPAM UPADHYAY       Gurgaon              NaN      Men Footwear

     Sub_category Product_Gender        Segment    Class             Family  \
0      SUNGLASSES         UNISEX     SUNGLASSES  AVIATOR             UNISEX
1  Sports Apparel            MEN      MENS WEAR     TOPS  SPORT & ADVENTURE
2           Bags         UNISEX         UNISEX      NaN                NaN
3  Sports Apparel            MEN      MENS WEAR     TOPS  SPORT & ADVENTURE
4   Mens Footwear            MEN  MENS FOOTWEAR      NaN             SPORTS

   … Coupon_Percentage Quantity Cost_Price Item_Price  \
0  …               0.0        1    2294.54       4999
1  …               NaN        1    2919.33       4999
2  …               NaN        1    2186.66       4095
3  …               NaN        1    2919.33       4999
4  …              25.0        1    5167.83       7495

  Special_Price_effective paid_pr_effective Value_CM1  Value_CM2  \
0                  4999.0           4544.38   1722.77    1134.77
1                  4999.0           4999.00   1499.87     876.87
2                  4095.0           4095.00   1433.07     955.07
3                  4999.0           4999.00   1499.87     876.87
4                  7495.0           5621.25   -198.99    -357.99

   Special_price  Paid_pr
0           4999     4544
1           4999     4999
2           4095     4095
3           4999     4999
4           7495     5621

[5 rows x 28 columns]
```

[11]:
```
# Checking shape of dataset
df.shape
```

[11]:
```
(50846, 28)
```

[12]:
```
# Checking column names
df.columns
```

[12]:
```
Index(['S.no', 'Name', 'Shipping_city', 'Category_Grouped', 'Category',
       'Sub_category', 'Product_Gender', 'Segment', 'Class', 'Family', 'Brand',
       'Brick', 'Item_NM', 'Color', 'Size', 'Sale_Flag', 'Payment_Method',
       'coupon_money_effective', 'Coupon_Percentage', 'Quantity', 'Cost_Price',
       'Item_Price', 'Special_Price_effective', 'paid_pr_effective',
       'Value_CM1', 'Value_CM2', 'Special_price', 'Paid_pr'],
      dtype='object')
```

[13]:
```
# Insert data into the created table
for _, row in df.iterrows():
    # Handle NaN values by converting them to None
    row = [None if pd.isna(value) else value for value in row]

    # Prepare and execute the insert query
    insert_query = f"INSERT INTO `{table_name}` ({', '.join([f'`{col}`' for col␣
 ↪in column_names])}) VALUES ({', '.join(['%s'] * len(column_names))})"
    cursor.execute(insert_query, tuple(row))

# Commit changes and close the connection
conn.commit()
conn.close()

print(f"Table '{table_name}' created and data loaded successfully.")
```

```
Table 'paytm_table' created and data loaded successfully.
```

3 2. Excess Database Information as per your Requirement

[14]:
```
# Pass the query through this function ant get your results
from IPython.display import display

def execute_query(query):

    # MySQL database connection details
    host = 'YOUR_HOST_NAME'
    user = 'YOUR_USER_NAME'
    password = 'YOUR_PASSWORD'
    database = 'paytmanalytics'  # Replace with your database name

    # Connect to MySQL
    conn = mysql.connector.connect(host=host, user=user, password=password,␣
 ↪database=database)
```

```
    # Create a cursor
    cursor = conn.cursor()

    result_df = None  # Initialize result_df to None

    try:
        # Execute the query
        cursor.execute(query)

        # Fetch all results
        results = cursor.fetchall()

        # Display results in a table format
        if results:
            header = [desc[0] for desc in cursor.description]
            data = list(results)
            result_df = pd.DataFrame(data, columns=header)
            # display(result_df)
        else:
            print("No results found.")

    except mysql.connector.Error as error:
        print(f"Error: {error.msg}")
        # If an error occurs, result_df will remain None

    finally:
        # Close the cursor and connection
        cursor.close()
        conn.close()

    return result_df  # Return the DataFrame result_df
```

3.1 1. What does the "Category_Grouped" column represent, and how many unique categories are there?

[15]:
```
# Define the query
query = "SELECT COUNT(DISTINCT Category_Grouped) AS Unique_Categories FROM␣
 ↪paytm_table;"
# Call the function with the query
execute_query(query)
```

[15]:

	Unique_Categories
0	4

3.2 2. Can you list the top 5 shipping cities in terms of the number of orders?

[16]:
```
# Define the query
query = "SELECT Shipping_city AS Shipping_City, COUNT(*) AS Order_Count FROM␣
 ↪paytm_table GROUP BY Shipping_city ORDER BY Order_Count DESC LIMIT 5;"
# Call the function with the query
execute_query(query)
```

[16]:

	Shipping_City	Order_Count
0	New Delhi	4560
1	Chennai	4254
2	Bangalore	3974
3	Mumbai	3159
4	Hyderabad	2849

3.3 3. Show me a table with all the data for products that belong to the "Electronics" category.

[17]:
```
# Define the query
query = "SELECT * FROM paytm_table WHERE Category = 'Electronics';"
# Call the function with the query
execute_query(query)
```

```
No results found.
```

3.4 4. Filter the data to show only rows with a "Sale_Flag" of 'Yes'.

[18]:
```
df['Sale_Flag'].value_counts()
```

[18]:
```
Not on Sale    34675
On Sale        16171
Name: Sale_Flag, dtype: int64
```

[19]:
```
# Define the query
query = "SELECT * FROM paytm_table WHERE Sale_Flag = 'On Sale';"
# Call the function with the query
execute_query(query)
```

[19]:

	S.no	Name	Shipping_city	Category_Grouped	\
0	2	AMIT GALPHADE	Ahmedabad	Apparels	
1	4	MALLIKARJUNA H	Bangalore	Apparels	
2	10	ASHWIN GIDWANI	Pune	Apparels	
3	16	Rompelli GopalK	Salem	Shoes	
4	20	prabhakar reddy	Jhansi	None	
…	…	…	…	…	
16166	50833	Chandrashekhar	Chennai	Others	
16167	50837	Madhusudhanan M	Madurai	Apparels	

```
16168  50839     partha padhi  Bhubaneshwar           Others
16169  50840  ALAGARASAN CHIN     Hyderabad           Others
16170  50843    ROBIN TRAKROO        Kanpur         Apparels

               Category    Sub_category Product_Gender             Segment  \
0      Sports Equipment  Sports Apparel            MEN           MENS WEAR
1      Sports Equipment  Sports Apparel            MEN           MENS WEAR
2      Sports Equipment  Sports Apparel            MEN           MENS WEAR
3          Men Footwear   Mens Footwear            MEN       MENS FOOTWEAR
4               WATCHES         WATCHES            MEN  WOMENS ACCESSORIES
…                 …             …             …                 …
16166           WATCHES         WATCHES          WOMEN             WOMEN
16167     Women Apparel          Ethnic          WOMEN         WOMENS WEAR
16168           WATCHES         WATCHES         UNISEX             UNISEX
16169           WATCHES         WATCHES          WOMEN             WOMEN
16170     Women Apparel          Ethnic          WOMEN         WOMENS WEAR

         Class             Family  … Coupon_Percentage Quantity Cost_Price  \
0         TOPS  SPORT & ADVENTURE  …              None        1    2919.33
1         TOPS  SPORT & ADVENTURE  …              None        1    2919.33
2         TOPS  SPORT & ADVENTURE  …              None        1     2412.0
3         None             SPORTS  …               0.0        1    3211.31
4      WATCHES               None  …              None        1     2938.8
…        …                 …  …  …               …      …        …
16166  WATCHES               None  …              None        1     2978.4
16167     None             ETHNIC  …              None        1    6386.45
16168  WATCHES               None  …              25.0        1    3356.68
16169  WATCHES               None  …              None        1    2610.13
16170     None             ETHNIC  …              20.0        1    3628.41

      Item_Price Special_Price_effective paid_pr_effective Value_CM1  \
0           4999                  4999.0            4999.0   1499.87
1           4999                  4999.0            4999.0   1499.87
2           4020                  4020.0            4020.0    1608.0
3           5499                  5499.0            4499.0    765.88
4           4799                  4799.0            4799.0    1860.2
…          …                     …                 …       …
16166       8250                  4125.0            4125.0    668.17
16167       8995                  4497.5            4497.5  -1888.95
16168       5395                  5395.0           4046.25    219.96
16169       4195                  4195.0            4195.0   1098.32
16170       6795                  6795.0            5436.0    1176.9

      Value_CM2 Special_price Paid_pr
0        876.87          4999    4999
1        876.87          4999    4999
2        1430.0          4020    4020
```

```
3         349.88          5499    4499
4         1581.2          4799    4799
…           …           …       …
16166     281.17          4125    4125
16167   -2151.95          4498    4498
16168      -8.04          5395    4046
16169     677.32          4195    4195
16170      791.9          6795    5436

[16171 rows x 28 columns]
```

3.5 5. Sort the data by "Item_Price" in descending order. What is the most expensive item ?

[20]:
```
# Define the query
query = "SELECT * FROM paytm_table ORDER BY Item_Price DESC LIMIT 1;"
# Call the function with the query
execute_query(query)
```

[20]:
```
     S.no         Name Shipping_city Category_Grouped   Category Sub_category  \
0   50841   Rajib Ghosh       Kolkata           Others  Jewellery    Jewellery

  Product_Gender           Segment Class Family  … Coupon_Percentage  \
0          WOMEN   WOMENS JEWELLERY  None   None  …               None

  Quantity Cost_Price Item_Price Special_Price_effective paid_pr_effective  \
0        1    6386.45       8995                  4497.5            4497.5

  Value_CM1 Value_CM2 Special_price Paid_pr
0  -1888.95  -2151.95          4498    4498

[1 rows x 28 columns]
```

3.6 6. Apply conditional formatting to highlight all products with a"Special_Price_effective" value below $50 in red.

[21]:
```
df['Special_Price_effective'].value_counts().sum() < 50
```

[21]:
```
False
```

[22]:
```
# Define the query
query = "SELECT * FROM paytm_table WHERE Special_Price_effective < 50;"
# Call the function with the query
execute_query(query)
```

```
No results found.
```

3.7 7. Create a pivot table to find the total sales value for each category.

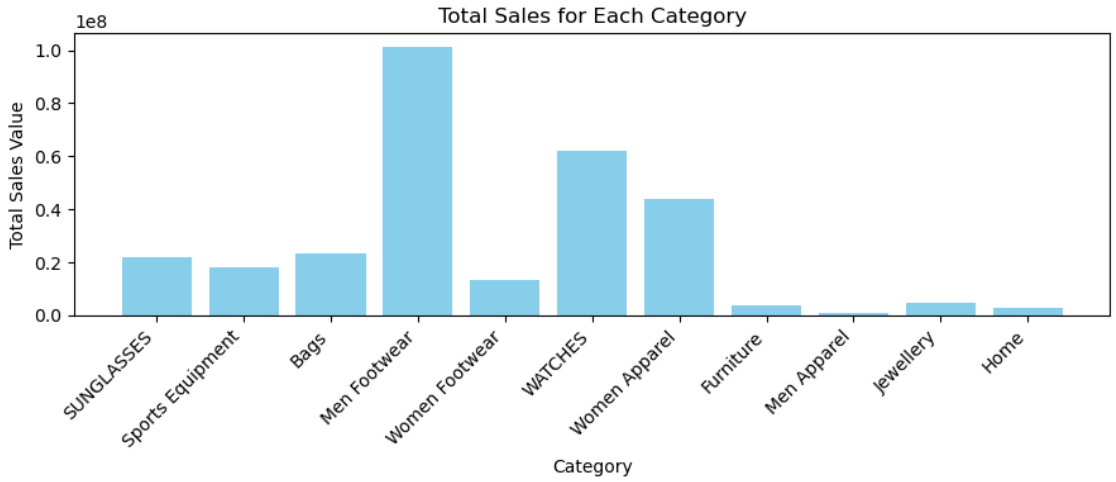
[23]:
```
# Define the query
query = "SELECT Category, SUM(Item_Price) AS Total_Sales_Value FROM paytm_table␣
 ↪GROUP BY Category;"
# Call the function with the query
execute_query(query)
```

[23]:

	Category	Total_Sales_Value
0	SUNGLASSES	21935695.0
1	Sports Equipment	18085020.0
2	Bags	23272288.0
3	Men Footwear	101245089.0
4	Women Footwear	13408398.0
5	WATCHES	62213793.0
6	Women Apparel	44010575.0
7	Furniture	3961755.0
8	Men Apparel	723305.0
9	Jewellery	4549307.0
10	Home	3051213.0

3.8 8. Create a bar chart to visualize the total sales for each category.

[24]:
```
# From Question_7:
df_cat_total_sales = execute_query(query)

# Plotting the bar chart
plt.figure(figsize=(9, 4))
plt.bar(df_cat_total_sales['Category'],␣
 ↪df_cat_total_sales['Total_Sales_Value'], color='skyblue')
plt.xlabel('Category')
plt.ylabel('Total Sales Value')
plt.title('Total Sales for Each Category')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



3.9 9. Create a pie chart to show the distribution of products in the "Family" category.

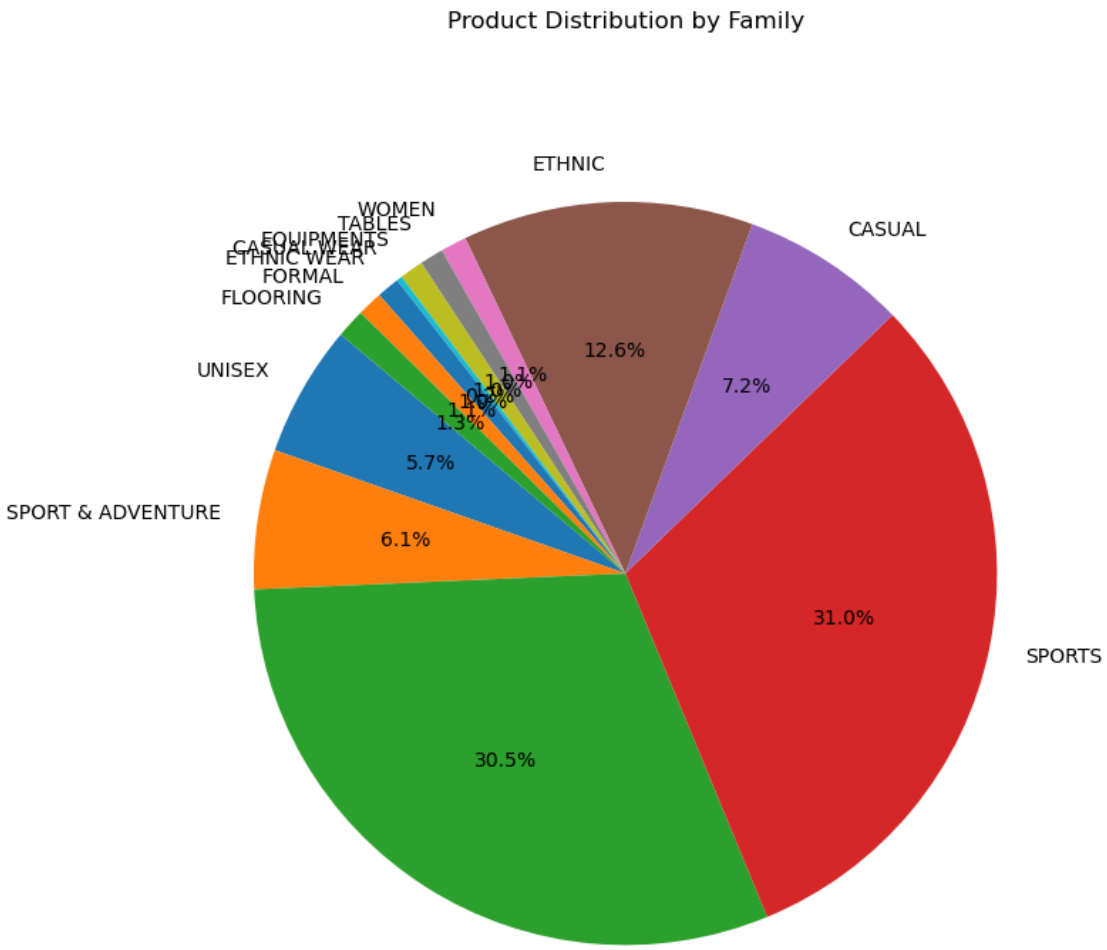
[25]:
```
# Define the query
query = "SELECT Family, COUNT(*) AS Product_Count FROM paytm_table GROUP BY␣
 ↪Family;"
# Call the function with the query
execute_query(query)
```

[25]:

	Family	Product_Count
0	UNISEX	2912
1	SPORT & ADVENTURE	3087
2	None	15533
3	SPORTS	15762
4	CASUAL	3670
5	ETHNIC	6426
6	WOMEN	576
7	TABLES	524
8	EQUIPMENTS	527
9	CASUAL WEAR	121
10	ETHNIC WEAR	505
11	FORMAL	562
12	FLOORING	641

[26]:
```
df_prod_distr = execute_query(query)

# Plotting the pie chart
plt.figure(figsize=(8, 8))  # Increase the figure size for better visualization
plt.pie(df_prod_distr['Product_Count'], labels=df_prod_distr['Family'],␣
 ↪autopct='%1.1f%%', startangle=140)
```

```
plt.title('Product Distribution by Family')
plt.axis('equal')
plt.tight_layout()  # Use tight layout for better spacing
plt.show()
```



3.10 10. Ensure that the "Payment_Method" column only contains valid [online] payment methods (e.g., Visa, MasterCard).

[27]:
```
df['Payment_Method'].value_counts()
```

[27]:
```
COD        38447
Prepaid    12399
```

```
Name: Payment_Method, dtype: int64
```

[28]:
```
# Define the query
query = "SELECT * FROM paytm_table WHERE Payment_Method = 'Prepaid';"
# Call the function with the query
execute_query(query)
```

[28]:
```
        S.no             Name Shipping_city Category_Grouped      Category  \
0          3  PRABHU NAMBIAPP       Chennai            Others          Bags
1          5  ANUPAM UPADHYAY       Gurgaon              None  Men Footwear
2          7  Abdul Qadir Sha        Kalyan              None  Men Footwear
3         16  Rompelli GopalK         Salem             Shoes  Men Footwear
4         24      kamla singh       Lucknow             Shoes  Men Footwear
…        …               …           …                …           …
12394  50819    deepak khanna        Kanpur              None     Furniture
12395  50832    rupesh pandey       Lucknow            Others       WATCHES
12396  50836   ABHINAV SANGAL        Mumbai              None          Bags
12397  50841      Rajib Ghosh       Kolkata            Others     Jewellery
12398  50846  MAHESH KULKARNI     Bangalore            Others    SUNGLASSES

        Sub_category Product_Gender           Segment    Class  Family  …  \
0               Bags         UNISEX            UNISEX     None    None  …
1      Mens Footwear            MEN     MENS FOOTWEAR     None  SPORTS  …
2      MENS FOOTWEAR            MEN     MENS FOOTWEAR     None  CASUAL  …
3      Mens Footwear            MEN     MENS FOOTWEAR     None  SPORTS  …
4      Mens Footwear            MEN     MENS FOOTWEAR     None  SPORTS  …
…              …            …                …      …      …  …
12394         LIVING         UNISEX            LIVING     None  TABLES  …
12395        WATCHES            MEN               MEN  WATCHES    None  …
12396          Bags          WOMEN             WOMEN     None    None  …
12397      Jewellery          WOMEN  WOMENS JEWELLERY     None    None  …
12398     SUNGLASSES         UNISEX        SUNGLASSES  AVIATOR  UNISEX  …

      Coupon_Percentage Quantity Cost_Price Item_Price  \
0                  None        1    2186.66       4095
1                  25.0        1    5167.83       7495
2                  None        1    2589.62       4560
3                   0.0        1    3211.31       5499
4                  None        1     3244.0       4990
…                 …      …         …         …
12394              None        1    3945.85       5900
12395              None        1    3686.03       6950
12396              None        1     3244.0       4990
12397              None        1    6386.45       8995
12398              20.0        1     3711.7       5690

      Special_Price_effective paid_pr_effective Value_CM1 Value_CM2  \
```

```
0                      4095.0            4095.0   1433.07    955.07
1                      7495.0            5621.25  -198.99   -357.99
2                      4560.0            4560.0    1441.5    1248.5
3                      5499.0            4499.0    765.88    349.88
4                      4990.0            4990.0   1166.85    932.85
…                       …                …        …         …
12394                  4130.0            4130.0   -294.86   -490.86
12395                  5560.0            5560.0    1229.1     901.1
12396                  4990.0            4990.0   1166.85    932.85
12397                  4497.5            4497.5  -1888.95  -2151.95
12398                  5690.0            4552.0    312.35    312.35

      Special_price Paid_pr
0              4095    4095
1              7495    5621
2              4560    4560
3              5499    4499
4              4990    4990
…               …       …
12394          4130    4130
12395          5560    5560
12396          4990    4990
12397          4498    4498
12398          5690    4552

[12399 rows x 28 columns]
```

3.11 11. Calculate the average "Quantity" sold for products in the "Clothing" category, grouped by "Product_Gender.

[29]:
```
df['Category'].value_counts()
```

[29]:
```
Men Footwear        17647
WATCHES             10440
Women Apparel        6931
SUNGLASSES           4030
Bags                 3949
Sports Equipment     3614
Women Footwear       2347
Home                  641
Jewellery             602
Furniture             524
Men Apparel           121
Name: Category, dtype: int64
```

[30]:
```
# Define the query
query = "SELECT Product_Gender, AVG(Quantity) AS Avg_Quantity_Sold FROM␣
 ↪paytm_table WHERE Category IN ('Women Apparel', 'Men Apparel') GROUP BY␣
 ↪Product_Gender;"
# Call the function with the query
execute_query(query)
```

[30]:

	Product_Gender	Avg_Quantity_Sold
0	WOMEN	1.0
1	MEN	1.0

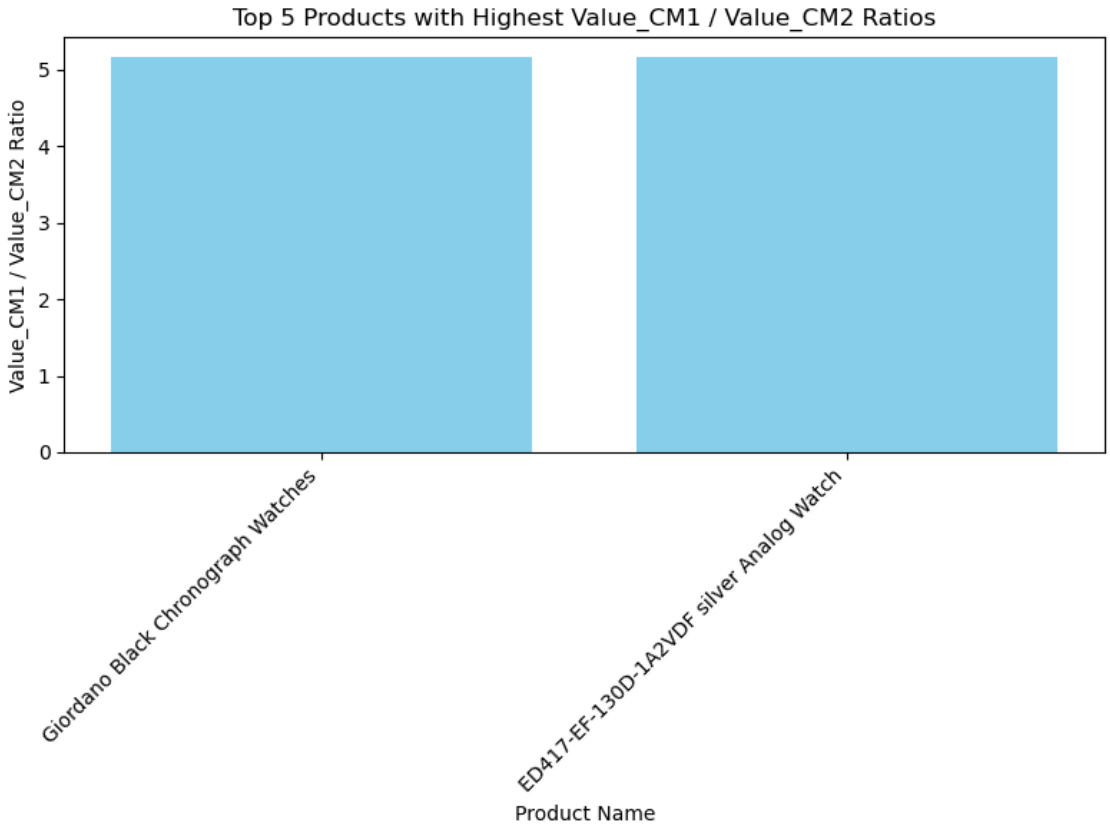
3.12 12. Find the top 5 products with the highest "Value_CM1" and "Value_CM2" ratios. Create a chart to visualize this data.

[31]:
```
# Define the query
query = "SELECT Item_NM, Value_CM1, Value_CM2, Value_CM1 / Value_CM2 AS Ratio␣
 ↪FROM paytm_table WHERE Value_CM2 != 0 ORDER BY (Value_CM1 / Value_CM2) DESC␣
 ↪LIMIT 5;"
# Call the function with the query
execute_query(query)
```

[31]:

	Item_NM	Value_CM1	Value_CM2	Ratio
0	Giordano Black Chronograph Watches	473.71	91.71	5.165304
1	Giordano Black Chronograph Watches	473.71	91.71	5.165304
2	ED417-EF-130D-1A2VDF silver Analog Watch	473.71	91.71	5.165304
3	Giordano Black Chronograph Watches	473.71	91.71	5.165304
4	Giordano Black Chronograph Watches	473.71	91.71	5.165304

[32]:
```
df_top_ratios = execute_query(query)
# Plotting the data
plt.figure(figsize=(8, 6))
plt.bar(df_top_ratios['Item_NM'], df_top_ratios['Ratio'], color='skyblue')
plt.xlabel('Product Name')
plt.ylabel('Value_CM1 / Value_CM2 Ratio')
plt.title('Top 5 Products with Highest Value_CM1 / Value_CM2 Ratios')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



3.13 13. Identify the top 3 "Class" categories with the highest total sales. Create a stacked bar chart to represent this data.

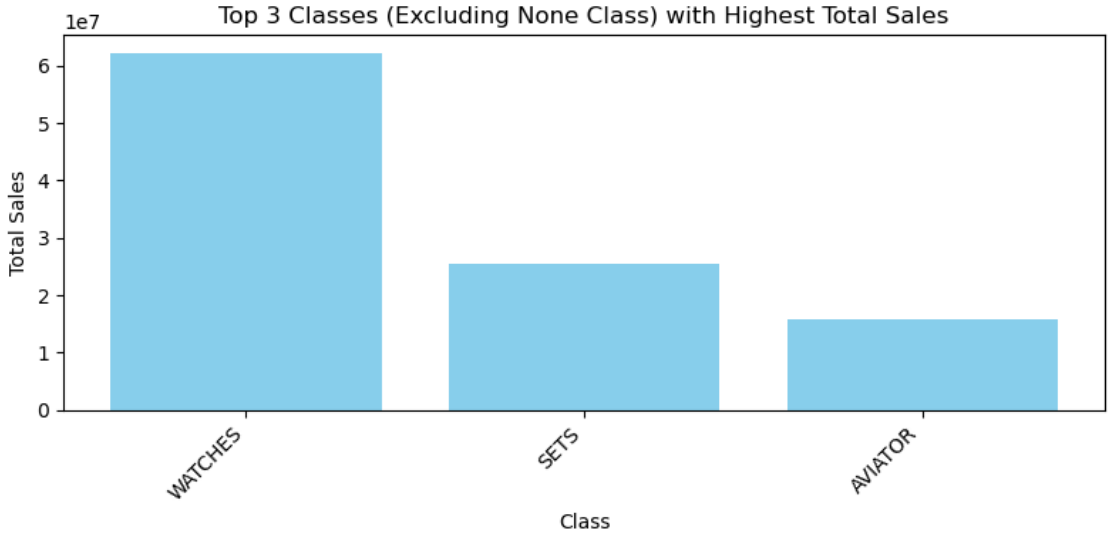
[33]:
```
# Define the query
query = "SELECT Class, SUM(Item_Price) AS Total_Sales FROM paytm_table GROUP BY␣
 ↪Class ORDER BY Total_Sales DESC LIMIT 4;"
# Call the function with the query
execute_query(query)
```

[33]:

	Class	Total_Sales
0	None	170968090.0
1	WATCHES	62213793.0
2	SETS	25529167.0
3	AVIATOR	15863415.0

[34]:
```
df_top_3_classes = execute_query(query)

# Filter the DataFrame to include only WATCHES, SETS, and AVIATOR classes
filtered_classes = ['WATCHES', 'SETS', 'AVIATOR']
filtered_df = df_top_3_classes[df_top_3_classes['Class'].isin(filtered_classes)]
```

```
# Plotting the filtered classes
plt.figure(figsize=(8, 4))

# Extracting data for plotting
classes = filtered_df['Class']
total_sales = filtered_df['Total_Sales']

# Stacked bar chart
plt.bar(classes, total_sales, color='skyblue')
plt.xlabel('Class')
plt.ylabel('Total Sales')
plt.title('Top 3 Classes (Excluding None Class) with Highest Total Sales')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



3.14 Use VLOOKUP or INDEX-MATCH to retrieve the "Color" of a product with a specific "Item_NM."

[35]:
```
df['Item_NM'].value_counts()
```

[35]:
```
Navy Blue Georgette Brocade Neck & Dupatta Suit Set    3257
SKINS Navy Blue Tights                                 3087
Reebok Sports Shoes Black                              1367
Street Tuneo Mid Black Sneakers                        1169
Tan Boots                                               828
                                                       ...
```

```
Leather Red Handbag                                    478
Rb3025 004 Green Sunglasses                            475
Bpb-0016-C Rose Gold/White Analog Watch                463
Knee-Length Brown Boots                                170
Erik Original Twill Jos Beige Casual Trousers          121
Name: Item_NM, Length: 76, dtype: int64
```

[36]:
```
# Define the query
query = "SELECT Color FROM paytm_table WHERE Item_NM = 'Navy Blue Georgette␣
 ↪Brocade Neck & Dupatta Suit Set' GROUP BY Color;"
# Call the function with the query
execute_query(query)
```

[36]:
```
       Color
0  NAVY BLUE
```

[37]:
```
# Define the query
query = "SELECT Color FROM paytm_table WHERE Item_NM = 'Street Tuneo Mid Black␣
 ↪Sneakers' GROUP BY Color;"
# Call the function with the query
execute_query(query)
```

[37]:
```
    Color
0  BLACK
```

[38]:
```
# Define the query
query = "SELECT Color FROM paytm_table WHERE Item_NM = 'Erik Original Twill Jos␣
 ↪Beige Casual Trousers' GROUP BY Color;"
# Call the function with the query
execute_query(query)
```

[38]:
```
    Color
0  BEIGE
```

3.15 15. Calculate the total "coupon_money_effective" and "Coupon_Percentage" for products in the "Electronics" category.

[39]:
```
# Define the query
query = "SELECT SUM(coupon_money_effective) AS Total_Coupon_Money_Effective,␣
 ↪SUM(Coupon_Percentage) AS Total_Coupon_Percentage FROM paytm_table WHERE␣
 ↪Category = 'Electronics';"
# Call the function with the query
execute_query(query)
```

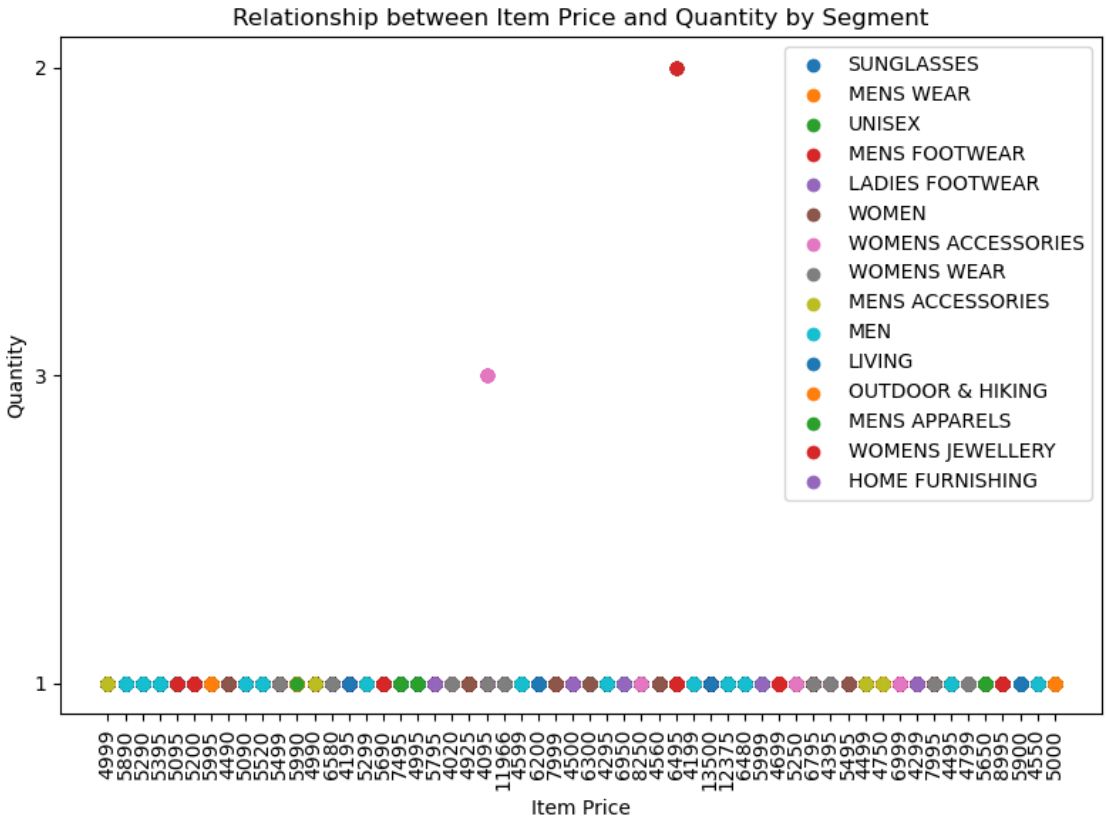
[39]:
```
   Total_Coupon_Money_Effective Total_Coupon_Percentage
0                          None                    None
```

3.16 16. Create a summary report that includes the total sales, average price, and total quantity sold for each product class.

[40]:
```
##### REQUIRED COLUMNS ARE NOT PROVIDED IN THE DATASET.
```

3.17 17. Calculate the total sales for each "Segment" and create a scatter plot to visualize the relationship between "Item_Price" and "Quantity" in this data.

[41]:
```
# Define the SQL query to calculate total sales for each segment
query = "SELECT Segment, SUM(Item_Price * Quantity) AS TotalSales FROM␣
 ↪paytm_table GROUP BY Segment;"

# Execute the query to get total sales for each segment
df_total_sales = execute_query(query)

# Fetch Item_Price and Quantity data for scatter plot
query_scatter = "SELECT Segment, Item_Price, Quantity FROM paytm_table;"
df_scatter = execute_query(query_scatter)

# Plotting scatter plot for each segment
plt.figure(figsize=(8, 6))

for segment in df_total_sales['Segment']:
    segment_data = df_scatter[df_scatter['Segment'] == segment]
    plt.scatter(segment_data['Item_Price'], segment_data['Quantity'],␣
 ↪label=segment)

plt.xlabel('Item Price')
plt.ylabel('Quantity')
plt.title('Relationship between Item Price and Quantity by Segment')
plt.legend()
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



3.18 18. Use the AVERAGEIFS function to find the average "Item_Price" for products that have a "Sale_Flag" of 'Yes.

[42]:
```
# Define the query
query = "SELECT AVG(Item_Price) AS Average_Item_Price FROM paytm_table WHERE␣
 ↪Sale_Flag = 'On Sale';"
# Call the function with the query
execute_query(query)
```

[42]:
```
   Average_Item_Price
0         5803.229609
```

3.19 19. Identify products with a "Paid_pr" higher than the average in their respective "Family" and "Brand" groups.

[43]:
```
# Define the query
```

```
query = "SELECT Category, Sub_category, Product_Gender, Segment, Class FROM␣
 ↪paytm_table p1 INNER JOIN (SELECT Family, Brand, AVG(Paid_pr) AS AvgPaidPr␣
 ↪FROM paytm_table GROUP BY Family, Brand ) AS avg_prices ON p1.Family =␣
 ↪avg_prices.Family AND p1.Brand = avg_prices.Brand WHERE p1.Paid_pr >␣
 ↪avg_prices.AvgPaidPr;"
# Call the function with the query
execute_query(query)
```

[43]:

	Category	Sub_category	Product_Gender	Segment	Class
0	SUNGLASSES	SUNGLASSES	UNISEX	SUNGLASSES	AVIATOR
1	SUNGLASSES	SUNGLASSES	UNISEX	SUNGLASSES	AVIATOR
2	SUNGLASSES	SUNGLASSES	UNISEX	SUNGLASSES	AVIATOR
3	SUNGLASSES	SUNGLASSES	UNISEX	SUNGLASSES	AVIATOR
4	SUNGLASSES	SUNGLASSES	UNISEX	SUNGLASSES	AVIATOR
…	…	…	…	…	…
16930	Sports Equipment	Sports Apparel	MEN	MENS WEAR	TOPS
16931	Sports Equipment	Sports Apparel	MEN	MENS WEAR	TOPS
16932	Sports Equipment	Sports Apparel	MEN	MENS WEAR	TOPS
16933	Sports Equipment	Sports Apparel	MEN	MENS WEAR	TOPS
16934	Sports Equipment	Sports Apparel	MEN	MENS WEAR	TOPS

[16935 rows x 5 columns]

3.20 20. Create a pivot table to show the total sales for each "Color" within the "Clothing" category and use conditional formatting to highlight the highest sales.

[ ]:
```
# Define the query
query = "SELECT Color, SUM(Item_Price) AS Total_Sales FROM paytm_table WHERE␣
 ↪Category IN ('Women Apparel', 'Men Apparel') GROUP BY Color;"
# Call the function with the query
execute_query(query)
```

[ ]:
```

```