In [1]:
```
# Loading the dataset using pandas

import pandas as pd
d = pd.read_csv('Customer_shopping_data.csv')
```

In [2]:
```
# Display the first 5 rows
d.head()
```

Out[2]:

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size
0	1	55	Male	Blouse	Clothing	53	Kentucky	L
1	2	19	Male	Sweater	Clothing	64	Maine	L
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M
4	5	45	Male	Blouse	Clothing	49	Oregon	M

In [3]:
```
# Display the last 5 rows
d.tail()
```

Out[3]:

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size
3495	3496	24	Female	Shorts	Clothing	33	Georgia	XL
3496	3497	39	Female	Shoes	Footwear	38	Idaho	M
3497	3498	35	Female	Handbag	Accessories	48	Colorado	M
3498	3499	47	Female	Sandals	Footwear	35	Kentucky	M
3499	3500	22	Female	Socks	Clothing	29	New Jersey	L

In [4]:
```
d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3500 entries, 0 to 3499
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Customer ID             3500 non-null   int64  
 1   Age                     3500 non-null   int64  
 2   Gender                  3500 non-null   object 
 3   Item Purchased          3500 non-null   object 
 4   Category                3500 non-null   object 
 5   Purchase Amount (USD)   3500 non-null   int64  
 6   Location                3500 non-null   object 
 7   Size                    3500 non-null   object 
 8   Color                   3500 non-null   object 
 9   Season                  3500 non-null   object 
 10  Review Rating           3463 non-null   float64
 11  Subscription Status     3500 non-null   object 
 12  Shipping Type           3500 non-null   object 
 13  Discount Applied        3500 non-null   object 
 14  Promo Code Used         3500 non-null   object 
 15  Previous Purchases      3500 non-null   int64  
 16  Payment Method          3500 non-null   object 
 17  Frequency of Purchases  3500 non-null   object 
dtypes: float64(1), int64(4), object(13)
memory usage: 492.3+ KB
```

In [5]:
```
# Summerize statistics for numerical columns
d.describe()
```

Out[5]:

	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
count	3500.000000	3500.000000	3500.000000	3463.000000	3500.000000
mean	1750.500000	44.032571	59.712857	3.748859	25.373143
std	1010.507298	15.233519	23.713364	0.716415	14.441421
min	1.000000	18.000000	20.000000	2.500000	1.000000
25%	875.750000	31.000000	38.000000	3.100000	13.000000
50%	1750.500000	44.000000	60.000000	3.800000	25.000000
75%	2625.250000	57.000000	80.000000	4.400000	38.000000
max	3500.000000	70.000000	100.000000	5.000000	50.000000

In [6]:
```
# Descriptive statistics for all columns
d.describe(include='all')
```

Out[6]:

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	L
count	3500.000000	3500.000000	3500	3500	3500	3500.000000	
unique	NaN	NaN	2	25	4	NaN	
top	NaN	NaN	Male	Pants	Clothing	NaN	I
freq	NaN	NaN	2652	156	1560	NaN	
mean	1750.500000	44.032571	NaN	NaN	NaN	59.712857	
std	1010.507298	15.233519	NaN	NaN	NaN	23.713364	
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	
25%	875.750000	31.000000	NaN	NaN	NaN	38.000000	
50%	1750.500000	44.000000	NaN	NaN	NaN	60.000000	
75%	2625.250000	57.000000	NaN	NaN	NaN	80.000000	
max	3500.000000	70.000000	NaN	NaN	NaN	100.000000	

In [7]:
```
# Count the number of missing (null or NaN) values in each column
d.isnull().sum()
```

Out[7]:
```
Customer ID                0
Age                        0
Gender                     0
Item Purchased             0
Category                   0
Purchase Amount (USD)      0
Location                   0
Size                       0
Color                      0
Season                     0
Review Rating             37
Subscription Status        0
Shipping Type              0
Discount Applied           0
Promo Code Used            0
Previous Purchases         0
Payment Method             0
Frequency of Purchases     0
dtype: int64
```

In [8]:
```
# Filling missing values in 'Review Rating' by the median value of each 'Categ

d['Review Rating'] = d['Review Rating'].fillna(d.groupby('Category')['Review R
```

In [9]:
```
d.isnull().sum()
```

Out[9]:
```
Customer ID               0
Age                       0
Gender                    0
Item Purchased            0
Category                  0
Purchase Amount (USD)     0
Location                  0
Size                      0
Color                     0
Season                    0
Review Rating             0
Subscription Status       0
Shipping Type             0
Discount Applied          0
Promo Code Used           0
Previous Purchases        0
Payment Method            0
Frequency of Purchases    0
dtype: int64
```

In [14]:
```
# Renaming columns according to snake casing for better readability and docume

# Converting all column names to lowercase
d.columns = d.columns.str.lower()

# Replacing spaces with underscores
d.columns = d.columns.str.replace(' ', '_')

# Renaming a 'purchase_amount_(usd)' column manually
d = d.rename( columns = {'purchase_amount_(usd)':'purchase_amount'} )
```

In [16]:
```
d.columns
```

Out[16]:
```
Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
       'purchase_amount', 'location', 'size', 'color', 'season',
       'review_rating', 'subscription_status', 'shipping_type',
       'discount_applied', 'promo_code_used', 'previous_purchases',
       'payment_method', 'frequency_of_purchases'],
      dtype='object')
```

In [23]:
```
# Creating New Column age_group
# Manually define age boundaries (bins) and labels for analysis using pd.cut.
# Young Adult: 18–25  |  Adult: 26–35  |  Middle-aged: 36–60   |   Senior: 61-

bins = [17, 25, 35, 60, 100]
labels = ['Young Adult', 'Adult', 'Middle-aged', 'Senior']

d['age_group'] = pd.cut(
    d['age'],
    bins = bins,
    labels = labels,
    right = True,                # Ensures the right edge of the bin is inclu
    include_lowest = True        # Ensures the lowest value (18) is included
```

```
)
```

In [24]:
```
d[['age','age_group']].head(10)
```

Out[24]:

	age	age_group
0	55	Middle-aged
1	19	Young Adult
2	50	Middle-aged
3	21	Young Adult
4	45	Middle-aged
5	46	Middle-aged
6	63	Senior
7	27	Adult
8	26	Adult
9	57	Middle-aged

In [29]:
```
# Creating New Column purchase_frequency_days by mapping text labels to numeri

frequency_mapping = {
    'Fortnightly' : 14,
    'Weekly' : 7,
    'Monthly' : 30,
    'Quarterly' : 90,
    'Bi-Weekly' : 14,
    'Annually' : 365,
    'Every 3 Months' : 90
}

d['purchase_frequency_days'] = d['frequency_of_purchases'].map(frequency_mappi
```

In [31]:
```
d[['purchase_frequency_days', 'frequency_of_purchases']].head(10)
```

Out[31]:

	purchase_frequency_days	frequency_of_purchases
0	14	Fortnightly
1	14	Fortnightly
2	7	Weekly
3	7	Weekly
4	365	Annually
5	7	Weekly
6	90	Quarterly
7	7	Weekly
8	365	Annually
9	90	Quarterly

In [33]:
```
d[['discount_applied', 'promo_code_used']].head(10)
```

Out[33]:

	discount_applied	promo_code_used
0	Yes	Yes
1	Yes	Yes
2	Yes	Yes
3	Yes	Yes
4	Yes	Yes
5	Yes	Yes
6	Yes	Yes
7	Yes	Yes
8	Yes	Yes
9	Yes	Yes

In [37]:
```
(d['discount_applied'] == d['promo_code_used']).all()
```

Out[37]:
```
np.True_
```

In [38]:
```
# Dropping promo_code_used column

d = d.drop('promo_code_used', axis = 1)
```

In [39]:
```
d.columns
```

Out[39]:
```
Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
       'purchase_amount', 'location', 'size', 'color', 'season',
       'review_rating', 'subscription_status', 'shipping_type',
       'discount_applied', 'previous_purchases', 'payment_method',
       'frequency_of_purchases', 'age_group', 'purchase_frequency_days'],
      dtype='object')
```

Connecting Python script to PostgreSQL

In [42]:
```
!pip install psycopg2-binary sqlalchemy
```

```
Collecting psycopg2-binary
  Downloading psycopg2_binary-2.9.11-cp310-cp310-win_amd64.whl.metadata (5.1 k
B)
Collecting sqlalchemy
  Downloading sqlalchemy-2.0.44-cp310-cp310-win_amd64.whl.metadata (9.8 kB)
Collecting greenlet>=1 (from sqlalchemy)
  Downloading greenlet-3.2.4-cp310-cp310-win_amd64.whl.metadata (4.2 kB)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\soleman\app
data\local\programs\python\python310\lib\site-packages (from sqlalchemy) (4.1
2.2)
Downloading psycopg2_binary-2.9.11-cp310-cp310-win_amd64.whl (2.7 MB)
   ---------------------------------------- 0.0/2.7 MB ? eta -:--:--
   --- ------------------------------------ 0.3/2.7 MB ? eta -:--:--
   ----------- ---------------------------- 0.8/2.7 MB 3.4 MB/s eta 0:00:01
   ----------- ---------------------------- 0.8/2.7 MB 3.4 MB/s eta 0:00:01
   -------------- ------------------------- 1.0/2.7 MB 1.5 MB/s eta 0:00:02
   -------------- ------------------------- 1.0/2.7 MB 1.5 MB/s eta 0:00:02
   -------------- ------------------------- 1.0/2.7 MB 1.5 MB/s eta 0:00:02
   ------------------ --------------------- 1.3/2.7 MB 895.5 kB/s eta 0:00:02
   ---------------------- ----------------- 1.6/2.7 MB 976.0 kB/s eta 0:00:02
   ---------------------------- ------------ 1.8/2.7 MB 1.0 MB/s eta 0:00:01
   ------------------------------ --------- 2.1/2.7 MB 1.0 MB/s eta 0:00:01
   ---------------------------------- ----- 2.4/2.7 MB 1.1 MB/s eta 0:00:01
   ---------------------------------------- 2.7/2.7 MB 1.1 MB/s  0:00:02
Downloading sqlalchemy-2.0.44-cp310-cp310-win_amd64.whl (2.1 MB)
   ---------------------------------------- 0.0/2.1 MB ? eta -:--:--
   ---- ----------------------------------- 0.3/2.1 MB ? eta -:--:--
   --------- ------------------------------ 0.5/2.1 MB 1.3 MB/s eta 0:00:02
   ------------------ --------------------- 1.0/2.1 MB 1.6 MB/s eta 0:00:01
   ------------------------ --------------- 1.3/2.1 MB 1.7 MB/s eta 0:00:01
   ------------------------------ ---------- 1.6/2.1 MB 1.7 MB/s eta 0:00:01
   ---------------------------------- ----- 1.8/2.1 MB 1.6 MB/s eta 0:00:01
   ---------------------------------------- 2.1/2.1 MB 1.7 MB/s  0:00:01
Downloading greenlet-3.2.4-cp310-cp310-win_amd64.whl (298 kB)
Installing collected packages: psycopg2-binary, greenlet, sqlalchemy

   ---------------------------------------- 0/3 [psycopg2-binary]
   ------------ --------------------------- 1/3 [greenlet]
   ------------ --------------------------- 1/3 [greenlet]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
   ------------------------- -------------- 2/3 [sqlalchemy]
```

```
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ------------------------- ------------- 2/3 [sqlalchemy]
   ---------------------------------------- 3/3 [sqlalchemy]

Successfully installed greenlet-3.2.4 psycopg2-binary-2.9.11 sqlalchemy-2.0.44
```

In [43]:

```
!pip install psycopg2-binary sqlalchemy
```

```
Requirement already satisfied: psycopg2-binary in c:\users\soleman\appdata\loca
l\programs\python\python310\lib\site-packages (2.9.11)
Requirement already satisfied: sqlalchemy in c:\users\soleman\appdata\local\pro
grams\python\python310\lib\site-packages (2.0.44)
Requirement already satisfied: greenlet>=1 in c:\users\soleman\appdata\local\pr
ograms\python\python310\lib\site-packages (from sqlalchemy) (3.2.4)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\soleman\app
data\local\programs\python\python310\lib\site-packages (from sqlalchemy) (4.1
2.2)
```

In [45]:

```
from sqlalchemy import create_engine

# Step 1: Connect to PostgreSQL
# Replace placeholders with your actual details
username = "postgres"      # default user
password = "123456789" # the password you set during installation
host = "localhost"         # if running locally
port = "5432"              # default PostgreSQL port
database = "customer_behavior"    # the database you created in pgAdmin

engine = create_engine(f"postgresql+psycopg2://{username}:{password}@{host}:{p

# Step 2: Load DataFrame into PostgreSQL
table_name = "customer"   # choose any table name
d.to_sql(table_name, engine, if_exists="replace", index=False)
print(f"Data successfully loaded into table '{table_name}' in database '{datab
```

```
Data successfully loaded into table 'customer' in database 'customer_behavior'.
```

In [ ]: