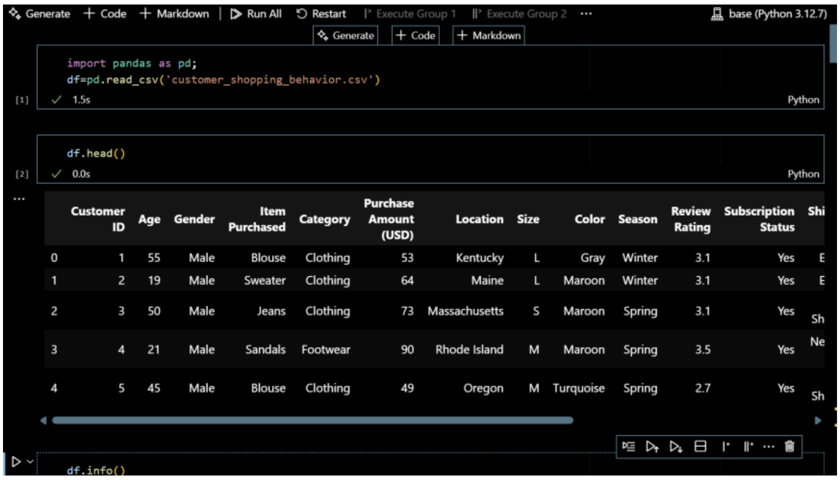
Customer Shopping Behaviour – Data Cleaning Summary

1. Data Loading

The dataset was imported using the **Pandas** library and explored with **head(), info(),** and **describe()** to understand structure, data types, and missing values.



```
import pandas as pd;
df=pd.read_csv('customer_shopping_behavior.csv')
```

```
df.head()
```

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscription Status	Shi
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1	Yes	E
1	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1	Yes	E
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1	Yes	Sh
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes	Ne
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7	Yes	Sh

```
df.info()
```

2. Column Standardization:

All column names were standardized to lowercase and converted to **snake_case** by replacing spaces with underscores.
This would improve consistency and ensure compatibility with SQL databases and visualization tools such as Power BI.

```
df.columns=df.columns.str.lower()
df.columns=df.columns.str.replace(' ','_')
```

```
df=df.rename(columns={'purchase_amount_(usd)':'purchase_amount'})
```

```
df.columns
```

```
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')
```

3. Handling Missing Values:

Missing values in the Review Rating column were filled using the category-wise **median**.

The median was preferred over the mean since it reduces the influence of **outliers** and better represents the central tendency of skewed data.

```
[5]
df.isnull().sum()
✓ 0.0s

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
```

```
[6]
df['Review Rating']=df.groupby('Category')['Review Rating'].transform(lambda x:x.fillna(x.median())
✓ 0.0s

[7]
df.isnull().sum()
✓ 0.0s

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
```

4. Age Group Segmentation

A new column **age_group** was created using **pd.qcut()** to divide customers into four quantile-based categories — Young Adult, Adult, Middle Age, and Senior.

```
[11]
labels=['Young Adult','Adult','Middle Age','Senior']
✓ 0.0s

[12]
df['age_group']=pd.qcut(df['age'],q=4,labels=labels)
✓ 0.0s

[13]
df[['age','age_group']].head(10)
✓ 0.0s
```

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

5. Purchase Frequency Mapping

The categorical column **frequency_of_purchases** was mapped to numeric values to represent time intervals in days.
Examples include: Weekly = 7, Monthly = 30, Annually = 365.
This would enable better comparison and calculation of purchase patterns over time.

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

df['purchase_frequency_days']=df['frequency_of_purchases'].map(frequency_mapping)
```

✓ 0.0s

```python
df[['purchase_frequency_days','frequency_of_purchases']].head(10)
```

✓ 0.0s

	purchase_frequency_days	frequency_of_purchases
0	14.0	Fortnightly
1	14.0	Fortnightly
2	7.0	Weekly
3	7.0	Weekly
4	365.0	Annually
5	7.0	Weekly
6	90.0	Quarterly
7	7.0	Weekly
8	365.0	Annually
9	90.0	Quarterly

6. Duplicate Column Removal

The columns **discount_applied** and **promo_code_used** contained identical data.
After verification using equality checks, **promo_code_used** was dropped to maintain schema clarity and prevent redundancy.

```python
df[['discount_applied','promo_code_used']].head(10)
```

✓ 0.0s

	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

```python
(df['discount_applied']==df['promo_code_used']).all()
```

✓ 0.0s

```
True
```

```python
df=df.drop('promo_code_used',axis=1)
```

✓ 0.0s

```python
df.columns
```

✓ 0.0s

```
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')
```

7. Safe Database Export Setup

A commented SQL export block was included using **SQLAlchemy.**
This would allow future database integration while preventing any accidental modifications to the MySQL database during notebook execution.

```
#username = "root"
#password = "myPassword"
#host = "127.0.0.1"
#port = "3311"
#database = "customer_behavior"

#engine = create_engine(f"mysql+pymysql://{username}:{password}@{host}:{port}/{database}")
#table_name = "customer"
#df.to_sql(table_name, engine, if_exists="replace", index=False)

#pd.read_sql("SELECT * FROM customer LIMIT 5;", engine)
```
✓ 0.0s

8. Final Outcome

- Dataset standardized using snake_case naming.
- Missing values treated with median-based imputation to minimize outlier effects.
- New engineered features (age_group, purchase_frequency_days) added for deeper analysis.
- Redundant columns removed for clarity.
- Ready-to-export dataset prepared for SQL integration and Power BI visualization.

```
df.sample(10)
df.tail(10)
```
✓ 0.1s Python

	customer_id	age	gender	item_purchased	category	purchase_amount	location	size	color	season	review_rating	subscription_status	shippin
3890	3891	35	Female	Shirt	Clothing	81	Nebraska	XL	Green	Winter	2.6	No	St
3891	3892	36	Female	Dress	Clothing	30	Colorado	L	Peach	Winter	4.7	No	Free Sl
3892	3893	35	Female	Jewelry	Accessories	86	Michigan	L	Indigo	Summer	3.5	No	St
3893	3894	21	Female	Hat	Accessories	64	Massachusetts	L	White	Fall	3.3	No	Store
3894	3895	66	Female	Skirt	Clothing	78	Connecticut	L	White	Spring	3.9	No	Sl
3895	3896	40	Female	Hoodie	Clothing	28	Virginia	L	Turquoise	Summer	4.2	No	Sl
3896	3897	52	Female	Backpack	Accessories	49	Iowa	L	White	Spring	4.5	No	Store
3897	3898	46	Female	Belt	Accessories	33	New Jersey	L	Green	Spring	2.9	No	St
3898	3899	44	Female	Shoes	Footwear	77	Minnesota	S	Brown	Summer	3.8	No	
3899	3900	52	Female	Handbag	Accessories	81	California	M	Beige	Spring	3.1	No	Store