```
import pandas as pd
```

```
# Load dataset
df = pd.read_csv("customer_shopping_behavior.csv")
```

```
# Preview data
df.head()
```

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

Next steps: Generate code with df | New interactive sheet

```
# Dataset structure
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Customer ID             3900 non-null   int64
 1   Age                     3900 non-null   int64
 2   Gender                  3900 non-null   object
 3   Item Purchased          3900 non-null   object
 4   Category                3900 non-null   object
 5   Purchase Amount (USD)   3900 non-null   int64
 6   Location                3900 non-null   object
 7   Size                    3900 non-null   object
 8   Color                   3900 non-null   object
 9   Season                  3900 non-null   object
 10  Review Rating           3863 non-null   float64
 11  Subscription Status     3900 non-null   object
 12  Shipping Type           3900 non-null   object
 13  Discount Applied        3900 non-null   object
 14  Promo Code Used         3900 non-null   object
 15  Previous Purchases      3900 non-null   int64
 16  Payment Method          3900 non-null   object
 17  Frequency of Purchases  3900 non-null   object
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```

```
# Summary statistics
df.describe(include="all")
```

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscr
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN	
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN	
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN	
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065	
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983	
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000	
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000	
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000	
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000	
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000	

```
# Check missing values
df.isnull().sum()
```

	0
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

```
# Impute missing review_rating with median per category
df["Review Rating"] = (
    df.groupby("Category")["Review Rating"]
      .transform(lambda x: x.fillna(x.median()))
)
```

```
# Verify missing values resolved
df.isnull().sum()
```

	0
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

```
# Convert column names to snake_case
df.columns = df.columns.str.lower().str.replace(" ", "_")
```

```
# Rename purchase amount column
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')
```

```
labels = ["Young Adult", "Adult", "Middle-aged", "Senior"]
df["age_group"] = pd.qcut(df["age"], q=4, labels=labels)
```

```
df[["age", "age_group"]].head(10)
```

	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	Young Adult
8	26	Young Adult
9	57	Middle-aged

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

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

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

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

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

	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

```
# Check if both columns are identical
(df["discount_applied"] == df["promo_code_used"]).all()
```

```
np.True_
```

```
# Drop redundant column
df = df.drop(columns=["promo_code_used"])
```

```
df.columns
```

```
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')
```

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

```
Collecting psycopg2-binary
  Downloading psycopg2_binary-2.9.11-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86_64.whl.metadata (4.9 kB)
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.12/dist-packages (2.0.45)
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (4.15.0
Downloading psycopg2_binary-2.9.11-cp312-cp312-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (4.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 35.8 MB/s eta 0:00:00
Installing collected packages: psycopg2-binary
Successfully installed psycopg2-binary-2.9.11
```

```
from sqlalchemy import create_engine
```

```
!pip install pymysql sqlalchemy
```

```
Collecting pymysql
  Downloading pymysql-1.1.2-py3-none-any.whl.metadata (4.3 kB)
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.12/dist-packages (2.0.45)
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (4.15.0
Downloading pymysql-1.1.2-py3-none-any.whl (45 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.3/45.3 kB 1.6 MB/s eta 0:00:00
Installing collected packages: pymysql
Successfully installed pymysql-1.1.2
```

```
import pandas as pd
from sqlalchemy import create_engine
```

```
!pip install pymysql sqlalchemy
```

```
Requirement already satisfied: pymysql in /usr/local/lib/python3.12/dist-packages (1.1.2)
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.12/dist-packages (2.0.45)
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (4.15.0
```

```
from sqlalchemy import create_engine
import pandas as pd

engine = create_engine(
    "mysql+pymysql://root:ishu2004@ballast.proxy.rlwy.net:57063/railway"
)
```

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

```
Requirement already satisfied: psycopg2-binary in /usr/local/lib/python3.12/dist-packages (2.9.11)
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.12/dist-packages (2.0.45)
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (4.15.0
```

```
import pandas as pd
from sqlalchemy import create_engine
```

```
from urllib.parse import quote_plus

raw_password = "YOUR_REAL_PASSWORD"
encoded_password = quote_plus(raw_password)
encoded_password
```

```
'YOUR_REAL_PASSWORD'
```

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

```
Requirement already satisfied: psycopg2-binary in /usr/local/lib/python3.12/dist-packages (2.9.11)
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.12/dist-packages (2.0.45)
Requirement already satisfied: greenlet>=1 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.12/dist-packages (from sqlalchemy) (4.15.0
```

```
from sqlalchemy import create_engine
from urllib.parse import quote_plus
```

```
username = "postgres"
password = quote_plus("dfZBWJxhAMqdppkHPJkYZOCvDrewTxHn")  # VERY IMPORTANT
host = "trolley.proxy.rlwy.net"
port = "21517"
database = "railway"   # Railway default DB name
```

```
engine = create_engine(
    f"postgresql+psycopg2://{username}:{password}@{host}:{port}/{database}"
)
```

```
with engine.connect() as conn:
    print(" PostgreSQL connected successfully!")
```

```
 PostgreSQL connected successfully!
```

```
df.to_sql("customer", engine, if_exists="replace", index=False)
print(" Data loaded into Railway PostgreSQL")
```

```
 Data loaded into Railway PostgreSQL
```

```
import pandas as pd

pd.read_sql("SELECT COUNT(*) FROM customer;", engine)
```

	count
0	3900

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

	customer_id	age	gender	item_purchased	category	purchase_amount	location	size	color	season	review_rating
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1
1	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7

```
pd.read_sql("""
SELECT season,
       COUNT(*) AS orders,
       ROUND(AVG(purchase_amount), 2) AS avg_spend
FROM customer
GROUP BY season
ORDER BY orders DESC;
""", engine)
```

	season	orders	avg_spend
0	Spring	999	58.74
1	Fall	975	61.56
2	Winter	971	60.36
3	Summer	955	58.41

Q1 Total revenue by Male vs Female

```
pd.read_sql("""
SELECT gender,
       SUM(purchase_amount) AS total_revenue
FROM customer
GROUP BY gender;
""", engine)
```

	gender	total_revenue
0	Female	75191.0
1	Male	157890.0

Q2. Customers who used a discount and spent more than average

```
pd.read_sql("""
SELECT *
FROM customer
WHERE discount_applied = 'Yes'
  AND purchase_amount >
      (SELECT AVG(purchase_amount) FROM customer);
""", engine)
```

	customer_id	age	gender	item_purchased	category	purchase_amount	location	size	color	season	review_rating
0	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1
1	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1
2	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5
3	7	63	Male	Shirt	Clothing	85	Montana	M	Gray	Fall	3.2
4	9	26	Male	Coat	Outerwear	97	West Virginia	L	Silver	Summer	2.6
...	...	...	...	...	...	...	...	...	...	...	...
834	1667	51	Male	Skirt	Clothing	64	Arkansas	M	Blue	Summer	3.1
835	1671	22	Male	Pants	Clothing	73	Utah	L	Cyan	Fall	3.6
836	1673	18	Male	Boots	Footwear	73	South Carolina	L	Gold	Fall	3.8
837	1674	21	Male	Blouse	Clothing	62	Hawaii	M	Violet	Fall	3.4
838	1676	35	Male	Pants	Clothing	90	Colorado	M	Beige	Spring	4.7

839 rows × 19 columns

Q3. Top 5 products with highest average review rating

```
pd.read_sql("""
SELECT item_purchased,
       ROUND(AVG(review_rating)::numeric, 2) AS avg_rating
FROM customer
GROUP BY item_purchased
ORDER BY avg_rating DESC
LIMIT 5;
""", engine)
```

	item_purchased	avg_rating
0	Gloves	3.86
1	Sandals	3.84
2	Boots	3.82
3	Hat	3.80
4	T-shirt	3.78

Q4 Avg purchase: Standard vs Express

```
pd.read_sql("""
SELECT shipping_type,
       ROUND(AVG(purchase_amount)::numeric, 2) AS avg_purchase_amount
FROM customer
WHERE shipping_type IN ('Standard', 'Express')
GROUP BY shipping_type;
""", engine)
```

	shipping_type	avg_purchase_amount
0	Express	60.48
1	Standard	58.46

Q5 Subscription vs Non-subscription 🔹 Avg spend

```
pd.read_sql("""
SELECT subscription_status,
       ROUND(AVG(purchase_amount)::numeric, 2) AS avg_spend
FROM customer
GROUP BY subscription_status;
""", engine)
```

	subscription_status	avg_spend
0	No	59.87
1	Yes	59.49

Total revenue

```
pd.read_sql("""
SELECT subscription_status,
       SUM(purchase_amount) AS total_revenue
FROM customer
GROUP BY subscription_status;
""", engine)
```

	subscription_status	total_revenue
0	No	170436.0
1	Yes	62645.0

Q6 Products with highest % discounted purchases

```
pd.read_sql("""
SELECT item_purchased,
       ROUND(
           100.0 * SUM(CASE WHEN discount_applied = 'Yes' THEN 1 ELSE 0 END)
           / COUNT(*),
           2
       ) AS discount_percentage
FROM customer
GROUP BY item_purchased
ORDER BY discount_percentage DESC
LIMIT 5;
""", engine)
```

	item_purchased	discount_percentage
0	Hat	50.00
1	Sneakers	49.66
2	Coat	49.07
3	Sweater	48.17
4	Pants	47.37

Q7 Customer segmentation (New / Returning / Loyal)

```
pd.read_sql("""
SELECT
    CASE
        WHEN previous_purchases <= 5 THEN 'New'
        WHEN previous_purchases BETWEEN 6 AND 20 THEN 'Returning'
        ELSE 'Loyal'
    END AS customer_segment,
    COUNT(*) AS customer_count
FROM customer
GROUP BY customer_segment;
""", engine)
```

	customer_segment	customer_count
0	New	424
1	Returning	1137
2	Loyal	2339

Q8 Top 3 products per category (window function)

```
pd.read_sql("""
SELECT category,
       item_purchased,
       purchase_count
FROM (
    SELECT category,
           item_purchased,
           COUNT(*) AS purchase_count,
           RANK() OVER (PARTITION BY category ORDER BY COUNT(*) DESC) AS rnk
    FROM customer
    GROUP BY category, item_purchased
) ranked
WHERE rnk <= 3;
""", engine)
```

	category	item_purchased	purchase_count
0	Accessories	Jewelry	171
1	Accessories	Sunglasses	161
2	Accessories	Belt	161
3	Clothing	Pants	171
4	Clothing	Blouse	171
5	Clothing	Shirt	169
6	Footwear	Sandals	160
7	Footwear	Shoes	150
8	Footwear	Sneakers	145
9	Outerwear	Jacket	163
10	Outerwear	Coat	161

Q9 Repeat buyers vs subscription

```
pd.read_sql("""
SELECT subscription_status,
       COUNT(*) AS repeat_buyers
```

```
FROM customer
WHERE previous_purchases > 5
GROUP BY subscription_status;
""", engine)
```

	subscription_status	repeat_buyers
0	No	2518
1	Yes	958

Q🔟 Revenue by age group

```
pd.read_sql("""
SELECT age_group,
       SUM(purchase_amount) AS total_revenue
FROM customer
GROUP BY age_group
ORDER BY total_revenue DESC;
""", engine)
```

	age_group	total_revenue
0	Young Adult	62143.0
1	Middle-aged	59197.0
2	Adult	55978.0
3	Senior	55763.0

```
import pandas as pd

query = """
SELECT age_group,
       SUM(purchase_amount) AS total_revenue
FROM customer
GROUP BY age_group
ORDER BY total_revenue DESC;
"""

df_result = pd.read_sql(query, engine)

df_result
```

	age_group	total_revenue
0	Young Adult	62143.0
1	Middle-aged	59197.0
2	Adult	55978.0