PROJECT - 3

E-commerce Customer & Sales Analysis

Importing all the required packages

import pandas as pd

import numpy as np

```
import pandas as pd
import numpy as np
```

[52] ✓ 0.0s

 # 1. Load both CSV files into separate Pandas DataFrames

customers_df = pd.read_csv("customers.csv")

sales_df = pd.read_csv("sales.csv")

```
# 1. Load both CSV files into separate Pandas DataFrames
customers_df = pd.read_csv("customers.csv")
sales_df = pd.read_csv("sales.csv")
```

[53] ✓ 0.0s

#2. Display the first 5 and last 5 rows of each DataFrame

print("Customers - First 5 rows:")

print(customers_df.head())

print("\nCustomers - Last 5 rows:")

print(customers_df.tail())

print("\nSales - First 5 rows:")

print(sales_df.head())

print("\nSales - Last 5 rows:")

print(sales_df.tail())

```python
#2. Display the first 5 and last 5 rows of each DataFrame
print("Customers - First 5 rows:")
print(customers_df.head())

print("\nCustomers - Last 5 rows:")
print(customers_df.tail())

print("\nSales - First 5 rows:")
print(sales_df.head())

print("\nSales - Last 5 rows:")
print(sales_df.tail())
```

[55] ✓ 0.0s

```
Customers - First 5 rows:
   customer_id                  name                       email  \
0         1001           Norma Fisher          ysullivan@yahoo.com
1         1002           Susan Wagner  katelynmontgomery@yahoo.com
2         1003  Dr. Stephanie Collins  thomas15@stewart-bowman.com
3         1004           Joseph Brown    cortezraymond@garrett.com
4         1005              Amy Stark         lindathomas@west.net

                    country signup_date
0                   Lesotho  2023-12-20
1  United States of America  2024-09-16
2                    Mexico  2024-06-22
3                   Ecuador  2023-10-30
4                 Venezuela  2024-07-11

Customers - Last 5 rows:
     customer_id             name                             email  \
195         1196  Robin Schroeder  robersonjulie@phillips-daniel.biz
196         1197    Madison Hicks         williamsalexis@beasley.biz
197         1198      Emily Weiss            vschneider@williams.com
198         1199     Brandi Simon               isullivan@gmail.com
199         1200     Brianna Pugh             briannajackson@ray.com

         country signup_date
195      Estonia  2023-08-14
...
196           49.99
197          789.99
198           19.99
199           25.50
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

#3. Show column names, data types, and check for null values

print("Customers Info:")

print(customers_df.info())

print("\nMissing values:\n", customers_df.isnull().sum())

print("\nSales Info:")

print(sales_df.info())

print("\nMissing values:\n", sales_df.isnull().sum())

```
#3. Show column names, data types, and check for null values
print("Customers Info:")
print(customers_df.info())
print("\nMissing values:\n", customers_df.isnull().sum())

print("\nSales Info:")
print(sales_df.info())
print("\nMissing values:\n", sales_df.isnull().sum())
```

[56] ✓ 0.0s

```
Customers Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   customer_id  200 non-null    int64
 1   name         200 non-null    object
 2   email        200 non-null    object
 3   country      200 non-null    object
 4   signup_date  200 non-null    object
dtypes: int64(1), object(4)
memory usage: 7.9+ KB
None

Missing values:
 customer_id    0
name           0
email          0
country        0
signup_date    0
dtype: int64

Sales Info:
<class 'pandas.core.frame.DataFrame'>
...
category          0
quantity          0
price_per_unit    0
dtype: int64
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

 # 4. Convert the date columns to datetime

customers_df['signup_date'] = pd.to_datetime(customers_df['signup_date'])

sales_df['order_date'] = pd.to_datetime(sales_df['order_date'])

print("Converted 'signup_date' and 'order_date' to datetime.")

```
# 4. Convert the date columns to datetime
customers_df['signup_date'] = pd.to_datetime(customers_df['signup_date'])
sales_df['order_date'] = pd.to_datetime(sales_df['order_date'])
print("Converted 'signup_date' and 'order_date' to datetime.")
```

[57] ✓ 0.0s

```
Converted 'signup_date' and 'order_date' to datetime.
```

#5. Calculate total revenue and create 'total_amount'

sales_df['total_amount'] = sales_df['quantity'] * sales_df['price_per_unit']

print("Sales with total_amount column:")

print(sales_df.head())

```
#5. Calculate total revenue and create 'total_amount'
sales_df['total_amount'] = sales_df['quantity'] * sales_df['price_per_unit']
print("Sales with total_amount column:")
print(sales_df.head())
```

[58] ✓ 0.0s

```
Sales with total_amount column:
   order_id  customer_id order_date     product     category  quantity  \
0      5001         1071 2023-09-19      Tablet  Electronics         4
1      5002         1035 2022-10-01  Headphones  Accessories         1
2      5003         1093 2023-04-01      Webcam  Accessories         1
3      5004         1057 2023-07-12  Smartphone  Electronics         1
4      5005         1100 2023-03-13      Laptop  Electronics         2

   price_per_unit  total_amount
0          399.00       1596.00
1           89.99         89.99
2           59.00         59.00
3          599.00        599.00
4          789.99       1579.98
```

#6. Merge datasets on customer_id

merged_df = pd.merge(sales_df, customers_df, *on*='customer_id', *how*='inner')

print("Merged DataFrame:")

print(merged_df.head())

```python
#6. Merge datasets on customer_id
merged_df = pd.merge(sales_df, customers_df, on='customer_id', how='inner')
print("Merged DataFrame:")
print(merged_df.head())
```

[59] ✓ 0.0s

```
Merged DataFrame:
   order_id  customer_id order_date  product     category  quantity  \
0      5001         1071 2023-09-19   Tablet  Electronics         4
1      5021         1071 2022-11-30   Laptop  Electronics         4
2      5023         1071 2022-04-22  Charger  Accessories         5
3      5084         1071 2023-12-17  Monitor  Electronics         5
4      5153         1071 2022-09-30   Tablet  Electronics         4

   price_per_unit  total_amount           name                          email  \
0          399.00       1596.00  Gerald Garcia  marcus75@johnson-williams.com
1          789.99       3159.96  Gerald Garcia  marcus75@johnson-williams.com
2           25.50        127.50  Gerald Garcia  marcus75@johnson-williams.com
3          299.49       1497.45  Gerald Garcia  marcus75@johnson-williams.com
4          399.00       1596.00  Gerald Garcia  marcus75@johnson-williams.com

   country signup_date
0  Belgium  2023-05-03
1  Belgium  2023-05-03
2  Belgium  2023-05-03
3  Belgium  2023-05-03
4  Belgium  2023-05-03
```

#7. Top 5 customers by total spending

top5 = merged_df.groupby(['customer_id', 'name'])['total_amount'].sum().sort_values(*ascending*=False).head(5)

print("Top 5 Customers by Spending:")

print(top5)

```python
#7. Top 5 customers by total spending
top5 = merged_df.groupby(['customer_id', 'name'])['total_amount'].sum().sort_values(ascending=False).head(5)
print("Top 5 Customers by Spending:")
print(top5)
```

[60] ✓ 0.0s

```
Top 5 Customers by Spending:
customer_id  name
1100         Nicholas Wright PhD    8003.79
1071         Gerald Garcia          7976.91
1081         Kevin Fuller           7442.95
1009         Joanne Keller          7379.88
1052         Michael Anderson       5644.95
Name: total_amount, dtype: float64
```

#8. Count of customers by country

print("Customer Count by Country:")

customers_df['country'].value_counts()

```
#8. Count of customers by country
print("Customer Count by Country:")
customers_df['country'].value_counts()
```

[61] ✓ 0.0s

```
Customer Count by Country:

Burkina Faso    4
Hungary         4
Zambia          4
Slovenia        4
Congo           3
                ..
Lithuania       1
Micronesia      1
Cuba            1
Albania         1
El Salvador     1
Name: country, Length: 132, dtype: int64
```

#9. Average order value per customer

avg_order_value = merged_df.groupby('customer_id')['total_amount'].mean()

print("Average Order Value per Customer:")

avg_order_value.head()

```
#9. Average order value per customer
avg_order_value = merged_df.groupby('customer_id')['total_amount'].mean()
print("Average Order Value per Customer:")
avg_order_value.head()
```

[62] ✓ 0.0s

```
Average Order Value per Customer:

customer_id
1001    3159.960000
1002    1788.310000
1004     109.490000
1005     635.316667
1006     636.750000
Name: total_amount, dtype: float64
```

#10. Remove duplicates from both datasets

customers_df.drop_duplicates(*inplace*=True)

sales_df.drop_duplicates(*inplace*=True)

print("Removed duplicates. Current shape:")

print("Customers:", customers_df.shape)

print("Sales:", sales_df.shape)

```
#10. Remove duplicates from both datasets
customers_df.drop_duplicates(inplace=True)
sales_df.drop_duplicates(inplace=True)
print("Removed duplicates. Current shape:")
print("Customers:", customers_df.shape)
print("Sales:", sales_df.shape)
```

[63] ✓ 0.0s

```
Removed duplicates. Current shape:
Customers: (200, 5)
Sales: (200, 8)
```

 # 11. Handle missing/invalid data

sales_df = sales_df[(sales_df['quantity'] > 0) & (sales_df['price_per_unit'] > 0)]

print("Sales after removing invalid rows:")

print(sales_df.head())

```
# 11. Handle missing/invalid data
sales_df = sales_df[(sales_df['quantity'] > 0) & (sales_df['price_per_unit'] > 0)]
print("Sales after removing invalid rows:")
print(sales_df.head())
```

[64] ✓ 0.0s

```
Sales after removing invalid rows:
   order_id  customer_id order_date     product     category  quantity  \
0      5001         1071 2023-09-19      Tablet  Electronics         4
1      5002         1035 2022-10-01  Headphones  Accessories         1
2      5003         1093 2023-04-01      Webcam  Accessories         1
3      5004         1057 2023-07-12  Smartphone  Electronics         1
4      5005         1100 2023-03-13      Laptop  Electronics         2

   price_per_unit  total_amount
0          399.00       1596.00
1           89.99         89.99
2           59.00         59.00
3          599.00        599.00
4          789.99       1579.98
```

#12. Group by category: quantity sold and revenue

```
category_summary = merged_df.groupby('category').agg(
    total_quantity_sold=('quantity', 'sum'),
    total_revenue=('total_amount', 'sum')
)
print("Category Summary:")
print(category_summary)
```

```
#12. Group by category: quantity sold and revenue
category_summary = merged_df.groupby('category').agg(
    total_quantity_sold=('quantity', 'sum'),
    total_revenue=('total_amount', 'sum')
)
print("Category Summary:")
print(category_summary)
```

[65] ✓ 0.0s

```
Category Summary:
             total_quantity_sold  total_revenue
category
Accessories                  325       14257.55
Electronics                  281      128645.31
```

#13. Extract year and month, analyze monthly sales

```
merged_df['order_year_month'] = merged_df['order_date'].dt.to_period('M')
monthly_sales = merged_df.groupby('order_year_month')['total_amount'].sum()
print("Monthly Sales:")
print(monthly_sales)
```

```python
#13. Extract year and month, analyze monthly sales
merged_df['order_year_month'] = merged_df['order_date'].dt.to_period('M')
monthly_sales = merged_df.groupby('order_year_month')['total_amount'].sum()
print("Monthly Sales:")
print(monthly_sales)
```

[66] ✓ 0.0s

```
Monthly Sales:
order_year_month
2022-01     4180.77
2022-02     8713.20
2022-03     5867.79
2022-04     4738.39
2022-05     7510.29
2022-06     5971.85
2022-07     7814.84
2022-08    11263.29
2022-09     5995.33
2022-10     5614.86
2022-11     5696.31
2022-12     6313.38
2023-01     7512.38
2023-02     3972.42
2023-03     5026.39
2023-04     5367.36
2023-05     4687.38
2023-06     3586.39
2023-07     6722.90
2023-08     3810.35
2023-09     5275.88
2023-10     8844.39
2023-11     4165.90
2023-12     4250.82
Freq: M, Name: total_amount, dtype: float64
```

#14. Customers who signed up in the last 6 months but didn't buy

```
from datetime import datetime
cutoff_date = pd.Timestamp.now() - pd.DateOffset(months=6)

recent_signups = customers_df[customers_df['signup_date'] >= cutoff_date]
purchased_customers = merged_df['customer_id'].unique()

inactive_recent = recent_signups[~recent_signups['customer_id'].isin(purchased_customers)]
inactive_recent
```

```python
#14. Customers who signed up in the last 6 months but didn't buy
from datetime import datetime
cutoff_date = pd.Timestamp.now() - pd.DateOffset(months=6)

recent_signups = customers_df[customers_df['signup_date'] >= cutoff_date]
purchased_customers = merged_df['customer_id'].unique()

inactive_recent = recent_signups[~recent_signups['customer_id'].isin(purchased_customers)]
inactive_recent
```

[67] ✓ 0.0s

	customer_id	name	email	country	signup_date
31	1032	Bradley Robinson	ryanhoward@gmail.com	Swaziland	2025-06-06
91	1092	Michael Bryant	dreed@nelson.com	Pitcairn Islands	2025-05-03
93	1094	Melanie Gomez	meghan09@cunningham.com	French Guiana	2025-02-08
105	1106	Carla Orozco	blairapril@hotmail.com	Portugal	2025-05-30
129	1130	Frances Wilson	rachel58@yahoo.com	Marshall Islands	2025-05-26
131	1132	Traci Forbes	dylansilva@bush.com	Nauru	2025-03-08
151	1152	Michelle Nelson	nicholasberry@flores.net	Botswana	2025-05-23
155	1156	Kelly Miller	shaheric@gmail.com	Grenada	2025-02-01
156	1157	Pamela Mcdonald	kcrawford@williams-vaughan.com	Tuvalu	2025-05-06
158	1159	Amanda Freeman	aaronbrown@gmail.com	Kenya	2025-04-24
170	1171	William Bailey	stevenedwards@hill.com	Belize	2025-04-02
174	1175	Margaret Adams	tiffany40@hotmail.com	Egypt	2025-05-22
181	1182	Joseph Garcia	nolanpatricia@nixon.biz	Korea	2025-04-24
193	1194	Gary Hendricks	morgandevon@burgess.com	Faroe Islands	2025-03-11
197	1198	Emily Weiss	vschneider@williams.com	Australia	2024-12-28

#15. Products sold less than 10 times (low performers)

total_sales = merged_df.groupby('product')['quantity'].sum()

low_performers = total_sales[total_sales < 10]

print("Low Performing Products (<10 sales):")

print(low_performers)

```python
#15. Products sold less than 10 times (low performers)
total_sales = merged_df.groupby('product')['quantity'].sum()
low_performers = total_sales[total_sales < 10]
print("Low Performing Products (<10 sales):")
print(low_performers)
```

[68] ✓ 0.0s

```
Low Performing Products (<10 sales):
Series([], Name: quantity, dtype: int64)
```

```
# 16. Summary report per customer
summary = merged_df.groupby(['customer_id', 'name']).agg(
    total_orders=('order_id', 'nunique'),
    total_items=('quantity', 'sum'),
    total_spent=('total_amount', 'sum')
)
summary['avg_order_value'] = summary['total_spent'] / summary['total_orders']
summary
```

```
# 16. Summary report per customer
summary = merged_df.groupby(['customer_id', 'name']).agg(
    total_orders=('order_id', 'nunique'),
    total_items=('quantity', 'sum'),
    total_spent=('total_amount', 'sum')
)
summary['avg_order_value'] = summary['total_spent'] / summary['total_orders']
summary
```

69] ✓ 0.0s

customer_id	name	total_orders	total_items	total_spent	avg_order_value
1001	Norma Fisher	1	4	3159.96	3159.960000
1002	Susan Wagner	3	10	5364.93	1788.310000
1004	Joseph Brown	3	10	328.47	109.490000
1005	Amy Stark	3	9	1905.95	635.316667
1006	Juan Mann	2	6	1273.50	636.750000
...	...	...	...	...	...
1095	Johnny Powell	1	5	1497.45	1497.450000
1096	Adam Griffith	3	6	1325.48	441.826667
1097	Amy Fowler	2	4	118.97	59.485000
1099	Andrew Gaines	1	1	49.99	49.990000
1100	Nicholas Wright PhD	8	25	8003.79	1000.473750

86 rows × 4 columns

#17. NumPy discount: Apply 10% discount to Electronics

```
merged_df['discounted_amount'] = np.where(
    merged_df['category'] == 'Electronics',
    merged_df['total_amount'] * 0.90,
    merged_df['total_amount']
)
print("\nApplied discount to Electronics category:")
print(merged_df[['product', 'category', 'total_amount', 'discounted_amount']].head())
```

```
#17. NumPy discount: Apply 10% discount to Electronics
merged_df['discounted_amount'] = np.where(
    merged_df['category'] == 'Electronics',
    merged_df['total_amount'] * 0.90,
    merged_df['total_amount']
)
print("\nApplied discount to Electronics category:")
print(merged_df[['product', 'category', 'total_amount', 'discounted_amount']].head())
```

[70] ✓ 0.0s

```
Applied discount to Electronics category:
   product     category  total_amount  discounted_amount
0   Tablet  Electronics       1596.00           1436.400
1   Laptop  Electronics       3159.96           2843.964
2  Charger  Accessories        127.50            127.500
3  Monitor  Electronics       1497.45           1347.705
4   Tablet  Electronics       1596.00           1436.400
```