```
#Importing all libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

Customers_data=pd.read_csv(r"C:\Users\A.Rohith Venkatesh\Downloads\
Customers.csv")
Products_data=pd.read_csv(r"C:\Users\A.Rohith Venkatesh\Downloads\
Products.csv")
Transactions_data=pd.read_csv(r"C:\Users\A.Rohith Venkatesh\Downloads\
Transactions.csv")

Customers_data

    CustomerID        CustomerName         Region  SignupDate
0        C0001    Lawrence Carroll  South America  2022-07-10
1        C0002      Elizabeth Lutz           Asia  2022-02-13
2        C0003      Michael Rivera  South America  2024-03-07
3        C0004  Kathleen Rodriguez  South America  2022-10-09
4        C0005         Laura Weber           Asia  2022-08-15
..         ...                 ...            ...         ...
195      C0196         Laura Watts         Europe  2022-06-07
196      C0197    Christina Harvey         Europe  2023-03-21
197      C0198         Rebecca Ray         Europe  2022-02-27
198      C0199      Andrea Jenkins         Europe  2022-12-03
199      C0200         Kelly Cross           Asia  2023-06-11

[200 rows x 4 columns]

Customers_data.shape #shape():this function it defines to calculate
rows and columns of a dataset

(200, 4)

Products_data

   ProductID              ProductName     Category   Price
0       P001     ActiveWear Biography        Books  169.30
1       P002    ActiveWear Smartwatch  Electronics  346.30
2       P003  ComfortLiving Biography        Books   44.12
3       P004            BookWorld Rug   Home Decor   95.69
4       P005        TechPro T-Shirt     Clothing  429.31
..       ...                      ...          ...     ...
95      P096      SoundWave Headphones  Electronics  307.47
96      P097       BookWorld Cookbook        Books  319.34
97      P098         SoundWave Laptop  Electronics  299.93
98      P099     SoundWave Mystery Book        Books  354.29
99      P100        HomeSense Sweater     Clothing  126.34

[100 rows x 4 columns]
```

```
Products_data.shape

(100, 4)

Transactions_data

    TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0          T00001      C0199      P067  2024-08-25 12:38:23         1

1          T00112      C0146      P067  2024-05-27 22:23:54         1

2          T00166      C0127      P067  2024-04-25 07:38:55         1

3          T00272      C0087      P067  2024-03-26 22:55:37         2

4          T00363      C0070      P067  2024-03-21 15:10:10         3

..            ...        ...       ...                  ...       ...

995        T00496      C0118      P037  2024-10-24 08:30:27         1

996        T00759      C0059      P037  2024-06-04 02:15:24         3

997        T00922      C0018      P037  2024-04-05 13:05:32         4

998        T00959      C0115      P037  2024-09-29 10:16:02         2

999        T00992      C0024      P037  2024-04-21 10:52:24         1


     TotalValue   Price
0        300.68  300.68
1        300.68  300.68
2        300.68  300.68
3        601.36  300.68
4        902.04  300.68
..          ...     ...
995      459.86  459.86
996     1379.58  459.86
997     1839.44  459.86
998      919.72  459.86
999      459.86  459.86

[1000 rows x 7 columns]

Transactions_data.shape

(1000, 7)

Customers_data.head() #head()-->function defines to display top 5 rows of the dataset
```

```
  CustomerID        CustomerName         Region  SignupDate
0      C0001    Lawrence Carroll  South America  2022-07-10
1      C0002      Elizabeth Lutz           Asia  2022-02-13
2      C0003      Michael Rivera  South America  2024-03-07
3      C0004  Kathleen Rodriguez  South America  2022-10-09
4      C0005         Laura Weber           Asia  2022-08-15
```

```
Customers_data.info() #info()--> this function defines to display
which  datatypes are present on the dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   CustomerID    200 non-null    object
 1   CustomerName  200 non-null    object
 2   Region        200 non-null    object
 3   SignupDate    200 non-null    object
dtypes: object(4)
memory usage: 6.4+ KB
```

```
Customers_data.isna()#isna()--> isna() defines the dataset contain any
null values if we get #True:The dataset contains null vales #False:The
dataset contains no null values
```

```
     CustomerID  CustomerName  Region  SignupDate
0         False         False   False       False
1         False         False   False       False
2         False         False   False       False
3         False         False   False       False
4         False         False   False       False
..          ...           ...     ...         ...
195       False         False   False       False
196       False         False   False       False
197       False         False   False       False
198       False         False   False       False
199       False         False   False       False

[200 rows x 4 columns]
```

```
Customers_data.isna().sum() #isna().sum()--> describes that dataset
contains any nullvalues and calculate the nullvalues
```

```
CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64
```

```
Products_data
```

```
   ProductID                ProductName     Category   Price
0       P001       ActiveWear Biography        Books  169.30
1       P002      ActiveWear Smartwatch  Electronics  346.30
2       P003    ComfortLiving Biography        Books   44.12
3       P004               BookWorld Rug   Home Decor   95.69
4       P005            TechPro T-Shirt     Clothing  429.31
..       ...                        ...          ...     ...
95      P096       SoundWave Headphones  Electronics  307.47
96      P097         BookWorld Cookbook        Books  319.34
97      P098           SoundWave Laptop  Electronics  299.93
98      P099       SoundWave Mystery Book        Books  354.29
99      P100          HomeSense Sweater     Clothing  126.34

[100 rows x 4 columns]
```

```
Products_data.head()
```

```
  ProductID                ProductName     Category   Price
0      P001       ActiveWear Biography        Books  169.30
1      P002      ActiveWear Smartwatch  Electronics  346.30
2      P003    ComfortLiving Biography        Books   44.12
3      P004               BookWorld Rug   Home Decor   95.69
4      P005            TechPro T-Shirt     Clothing  429.31
```

```
Products_data.tail() #tail()-->this function defines it displays last
five rows of the dataset
```

```
   ProductID                ProductName     Category   Price
95      P096       SoundWave Headphones  Electronics  307.47
96      P097         BookWorld Cookbook        Books  319.34
97      P098           SoundWave Laptop  Electronics  299.93
98      P099     SoundWave Mystery Book        Books  354.29
99      P100          HomeSense Sweater     Clothing  126.34
```

```
Products_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   ProductID    100 non-null    object
 1   ProductName  100 non-null    object
 2   Category     100 non-null    object
 3   Price        100 non-null    float64
dtypes: float64(1), object(3)
memory usage: 3.3+ KB
```

```
Products_data.isna()
```

```
    ProductID  ProductName  Category  Price
0       False        False     False  False
1       False        False     False  False
2       False        False     False  False
3       False        False     False  False
4       False        False     False  False
..        ...          ...       ...    ...
95      False        False     False  False
96      False        False     False  False
97      False        False     False  False
98      False        False     False  False
99      False        False     False  False

[100 rows x 4 columns]
```

```
Products_data.isna().sum()
```

```
ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64
```

```
Transactions_data
```

```
    TransactionID CustomerID ProductID      TransactionDate  Quantity
\
0          T00001      C0199      P067  2024-08-25 12:38:23         1

1          T00112      C0146      P067  2024-05-27 22:23:54         1

2          T00166      C0127      P067  2024-04-25 07:38:55         1

3          T00272      C0087      P067  2024-03-26 22:55:37         2

4          T00363      C0070      P067  2024-03-21 15:10:10         3

..            ...        ...       ...                  ...       ...

995        T00496      C0118      P037  2024-10-24 08:30:27         1

996        T00759      C0059      P037  2024-06-04 02:15:24         3

997        T00922      C0018      P037  2024-04-05 13:05:32         4

998        T00959      C0115      P037  2024-09-29 10:16:02         2

999        T00992      C0024      P037  2024-04-21 10:52:24         1


     TotalValue   Price
```

```
0        300.68  300.68
1        300.68  300.68
2        300.68  300.68
3        601.36  300.68
4        902.04  300.68
..          ...     ...
995      459.86  459.86
996     1379.58  459.86
997     1839.44  459.86
998      919.72  459.86
999      459.86  459.86

[1000 rows x 7 columns]
```

```
Transactions_data.head()
```

```
  TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0        T00001      C0199      P067  2024-08-25 12:38:23         1
1        T00112      C0146      P067  2024-05-27 22:23:54         1
2        T00166      C0127      P067  2024-04-25 07:38:55         1
3        T00272      C0087      P067  2024-03-26 22:55:37         2
4        T00363      C0070      P067  2024-03-21 15:10:10         3

   TotalValue   Price
0      300.68  300.68
1      300.68  300.68
2      300.68  300.68
3      601.36  300.68
4      902.04  300.68
```

```
Transactions_data.tail()
```

```
    TransactionID CustomerID ProductID      TransactionDate  Quantity
\
995        T00496      C0118      P037  2024-10-24 08:30:27         1

996        T00759      C0059      P037  2024-06-04 02:15:24         3

997        T00922      C0018      P037  2024-04-05 13:05:32         4

998        T00959      C0115      P037  2024-09-29 10:16:02         2

999        T00992      C0024      P037  2024-04-21 10:52:24         1


     TotalValue   Price
995      459.86  459.86
996     1379.58  459.86
997     1839.44  459.86
998      919.72  459.86
999      459.86  459.86
```

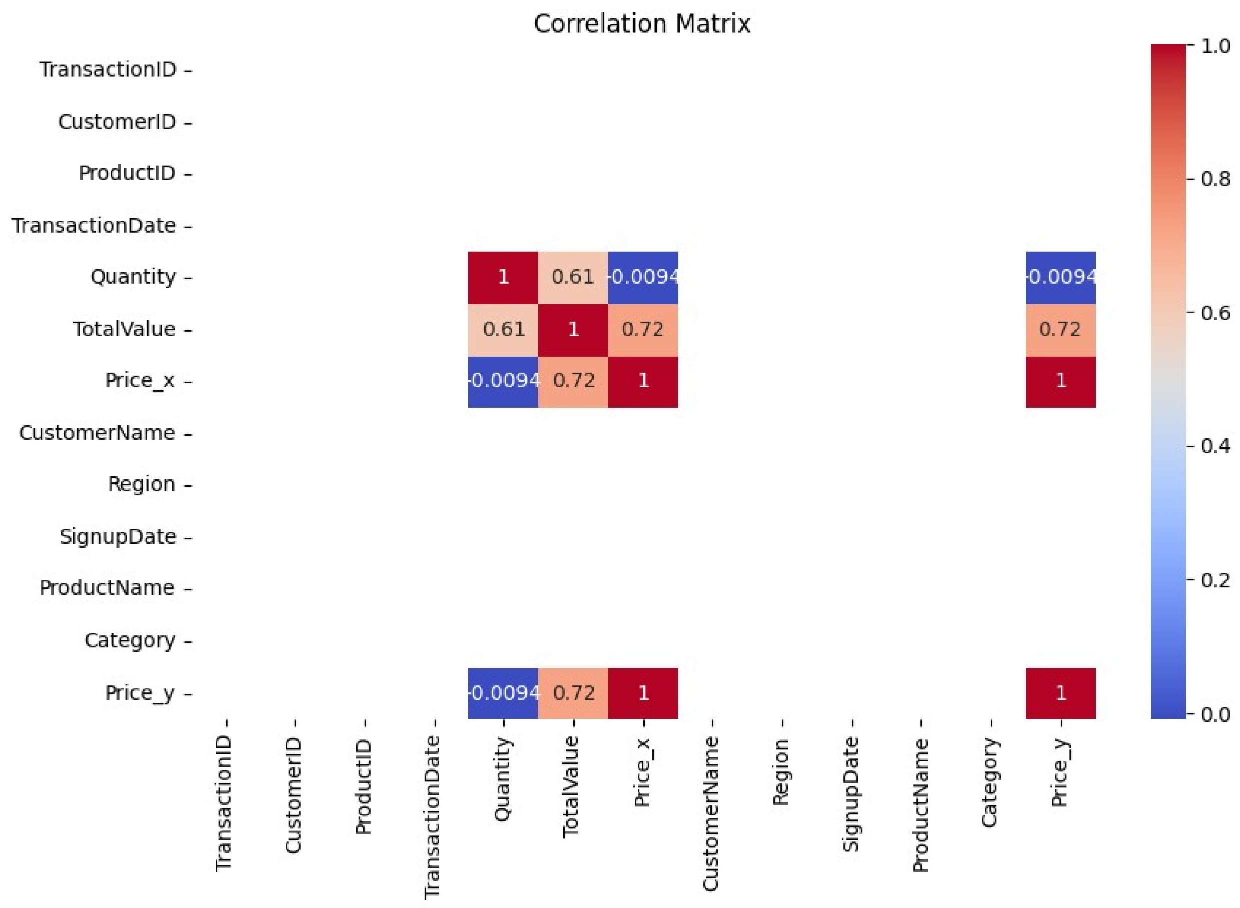
```
Transactions_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   TransactionID    1000 non-null   object 
 1   CustomerID       1000 non-null   object 
 2   ProductID        1000 non-null   object 
 3   TransactionDate  1000 non-null   object 
 4   Quantity         1000 non-null   int64  
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
```

```
Transactions_data.isna()
```

```
     TransactionID  CustomerID  ProductID  TransactionDate  
Quantity  \
0            False       False      False            False     False   

1            False       False      False            False     False   

2            False       False      False            False     False   

3            False       False      False            False     False   

4            False       False      False            False     False   

..             ...         ...        ...              ...       ...   

995          False       False      False            False     False   

996          False       False      False            False     False   

997          False       False      False            False     False   

998          False       False      False            False     False   

999          False       False      False            False     False   


     TotalValue  Price  
0         False  False  
1         False  False  
2         False  False  
3         False  False  
4         False  False  
..          ...    ...  
```

```
995        False  False
996        False  False
997        False  False
998        False  False
999        False  False

[1000 rows x 7 columns]

Transactions_data.isna().sum()

TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64

#merge datasets for EDA
merged_data = Transactions_data.merge(Customers_data, 
on='CustomerID').merge(Products_data, on='ProductID')

# General statistics
print("Merged Dataset Overview:")
print(merged_data.describe()) #describe()--> this function is define 
to calculate mean,mode and median 25%,50%,75% in the dataset

Merged Dataset Overview:
          Quantity   TotalValue     Price_x     Price_y
count  1000.000000  1000.000000  1000.00000  1000.00000
mean      2.537000   689.995560   272.55407   272.55407
std       1.117981   493.144478   140.73639   140.73639
min       1.000000    16.080000    16.08000    16.08000
25%       2.000000   295.295000   147.95000   147.95000
50%       3.000000   588.880000   299.93000   299.93000
75%       4.000000  1011.660000   404.40000   404.40000
max       4.000000  1991.040000   497.76000   497.76000

numeric_data = merged_data.select_dtypes(include=['float64', 'int64'])
correlation = numeric_data.corr()

numeric_data = numeric_data.dropna()  # Drop rows with missing values
# Alternatively, fill missing values with a default value:
# numeric_data = numeric_data.fillna(0)
correlation = numeric_data.corr()

# Convert possible numeric strings to numeric
for col in merged_data.columns:
    try:
        merged_data[col] = pd.to_numeric(merged_data[col], 
errors='coerce')  # Convert, setting invalid parsing as NaN
    except Exception as e:
        print(f"Error converting column {col}: {e}")

numeric_data = numeric_data.dropna()  # Drop rows with NaN values
# Alternatively, you can fill NaNs with a default value
# numeric_data = numeric_data.fillna(0)

# Correlation Analysis
correlation = merged_data.corr()
plt.figure(figsize=(10, 6))
sns.heatmap(correlation, annot=True, cmap="coolwarm")
plt.title("Correlation Matrix")
plt.show()
```



Business Insights for EDA(Exploratory Data Analysis)

```
#1.Top 5 purchased products
most_purchased_products = merged_data.groupby('ProductName')
['Quantity'].sum().sort_values(ascending=False).head(5)
print("Top 5 Most Purchased Products:")
print(most_purchased_products)
```

```
Top 5 Most Purchased Products:
Series([], Name: Quantity, dtype: int64)
```

```python
# 2. Regions with the highest revenue
region_revenue = merged_data.groupby('Region')
['TotalValue'].sum().sort_values(ascending=False)
print("Revenue by Region:")
print(region_revenue)
```

```
Revenue by Region:
Series([], Name: TotalValue, dtype: float64)
```

```python
# 3. Average transaction value by product category
category_avg_value = merged_data.groupby('Category')
['TotalValue'].mean().sort_values(ascending=False)
print("Average Transaction Value by Category:")
print(category_avg_value)
```

```
Average Transaction Value by Category:
Series([], Name: TotalValue, dtype: float64)
```

```python
# 4. Most active customers
active_customers = merged_data.groupby('CustomerID')
['TransactionID'].count().sort_values(ascending=False).head(5)
print("Top 5 Most Active Customers:")
print(active_customers)
```

```
Top 5 Most Active Customers:
Series([], Name: TransactionID, dtype: int64)
```

```python
print(merged_data[['TransactionDate', 'TotalValue']].isnull().sum())
```

```
TransactionDate    1000
TotalValue            0
dtype: int64
```

```python
merged_data = merged_data.dropna(subset=['TransactionDate',
'TotalValue'])
```

```python
merged_data['TransactionDate'] =
pd.to_datetime(merged_data['TransactionDate'], errors='coerce')
print(merged_data['TransactionDate'].head())
```

```
Series([], Name: TransactionDate, dtype: datetime64[ns])
```

```python
merged_data = merged_data.dropna(subset=['TransactionDate'])
```

```python
# Ensure 'TransactionDate' is correctly parsed as a datetime
merged_data['TransactionDate'] =
pd.to_datetime(merged_data['TransactionDate'], errors='coerce')

# Drop rows with missing 'TransactionDate' or 'TotalValue'
```

```
merged_data = merged_data.dropna(subset=['TransactionDate',
'TotalValue'])

# Aggregate the daily revenue
daily_revenue = merged_data.groupby('TransactionDate')
['TotalValue'].sum()

# Check the summary statistics of the daily revenue
print(daily_revenue.describe())

# Plot the revenue trend
plt.figure(figsize=(12, 6))
daily_revenue.plot(title="Revenue Trend Over Time", xlabel="Date",
ylabel="Revenue")
plt.show()
```

```
count    0.0
mean     NaN
std      NaN
min      NaN
25%      NaN
50%      NaN
75%      NaN
max      NaN
Name: TotalValue, dtype: float64
```