Introduction

This project performs Exploratory Data Analysis (EDA) on Customers and Products datasets to understand customer behavior, product trends, discounts, and purchasing patterns using Python.

In [3]:
```
#import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

load Dataset

In [4]:
```
customers=pd.read_csv("Customers.csv") ##customer dataset
products=pd.read_csv("Products.csv")#products dataset
```

Dataset Overview

In [8]:
```
customers.head() ###Displays the first 5 rows of the customers DataFrame
```

Out[8]:

	Customer_ID	Customer_Name	Contact_no	Email	Address	
0	C001	Joan Johnson	825-522-8959x935	schultzjohn@hotmail.com	1639 Pamela Streets, New Emily, IN 04537	
1	C002	Daniel Welch	762-020-3805x90140	kcarter@williams.com	11086 Vasquez Cape Apt. 122, Joelmouth, IA 76054	Quir
2	C003	Kristin Wolf	001-253-585-5846	julie22@yahoo.com	4404 Morgan Wall, Darrellburgh, NC 21390	Wate
3	C004	Kimberly Johnson	779-888-4859x165	andrewlee@gmail.com	2412 Michael Estate, Veronicafort, NC 16962	J
4	C005	Juan Woodward	(832)416-1791	gabrielmills@olson.net	872 Joseph Lakes, Rebeccahaven, TN 76169	

In [9]:
```
products.head()  #Displays the first 5 rows of the products DataFrame
```

Out[9]:

	Product_ID	Product_Name	Quantity	Price	Discount
0	P001	Realize	878	249.81	15.0
1	P002	Process	471	263.44	NaN
2	P003	Nature	329	157.08	10.0
3	P004	Space	521	385.53	0.0
4	P005	Art	73	383.83	0.0

In [10]:
```
customers.info() ##Shows a summary of the customers DataFrame, including column nam
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Customer_ID    100 non-null    object
 1   Customer_Name  100 non-null    object
 2   Contact_no     90 non-null     object
 3   Email          85 non-null     object
 4   Address        100 non-null    object
 5   City           93 non-null     object
 6   Country        100 non-null    object
 7   Product_ID     100 non-null    object
dtypes: object(8)
memory usage: 6.4+ KB
```

In [11]:
```
products.info() ##Shows a summary of the products DataFrame, including column names
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Product_ID    100 non-null    object 
 1   Product_Name  100 non-null    object 
 2   Quantity      100 non-null    int64  
 3   Price         100 non-null    float64
 4   Discount      88 non-null     float64
dtypes: float64(2), int64(1), object(2)
memory usage: 4.0+ KB
```

** TASKS IMPLEMENTATION **

Task 1: Total Inventory Value

In [16]:
```
## Calculate the total value of all products in inventory
total_inventory_value= (products["Price"]*products["Quantity"]).sum()
print(total_inventory_value)
```

```
13133737.88
```

Task 2: Discounted Price

In [26]:
```
products['Final_Price'] = products['Price'] - (products['Price'] * products['Discou
products[['Product_Name','Price','Discount','Final_Price']]
```

Out[26]:

	Product_Name	Price	Discount	Final_Price
0	Realize	249.81	15.0	212.3385
1	Process	263.44	NaN	263.4400
2	Nature	157.08	10.0	141.3720
3	Space	385.53	0.0	385.5300
4	Art	383.83	0.0	383.8300
...	...	...	...	...
95	Positive	154.61	0.0	154.6100
96	Region	422.02	5.0	400.9190
97	Man	9.31	10.0	8.3790
98	Religious	392.70	5.0	373.0650
99	Natural	74.72	20.0	59.7760

100 rows × 4 columns

Task3: How many products are available in the dataset

In [29]:
```
total_products=products.shape[0]   ####total number of products in the DataFrame
total_products
```

Out[29]: 100

*Task4: List customers from a specific city or country. *

In [40]:
```
#pd.set_option('display.max_columns', None); pd.set_option('display.width', 1000);

specific_country = "French Guiana"
customers_from_country = customers[customers['Country'] == specific_country]   ####
print(customers_from_country)
```

```
   Customer_ID    Customer_Name              Contact_no                      Email
Address         City        Country Product_ID
36        C037   William Henson       207.335.9920x0268           mdavis@yahoo.com
04024 Smith Flats Apt. 372, New Joshua, ME 68833   Brownmouth  French Guiana       P
059
52        C053      David Tyler      (816)659-6803x2817        kayla47@hotmail.com
3533 Cheryl Track, South David, GA 60944   Adamsville  French Guiana       P037
69        C070   Michelle Ortiz      913.342.5200x29400  davidmartinez@fischer.com
33497 April Meadows, North Michael, WI 95919  Collinsport  French Guiana       P004
86        C087  Amanda Phillips  001-590-719-0766x58412          betty57@yahoo.com
8132 Lee Crescent, Joyceshire, WY 66868  Reevesmouth  French Guiana       P021
```

*Task5: Identify which product category is most purchased. *

In [46]:
```
customers['Product_ID'].value_counts().idxmax() #Display the most popular product I
```

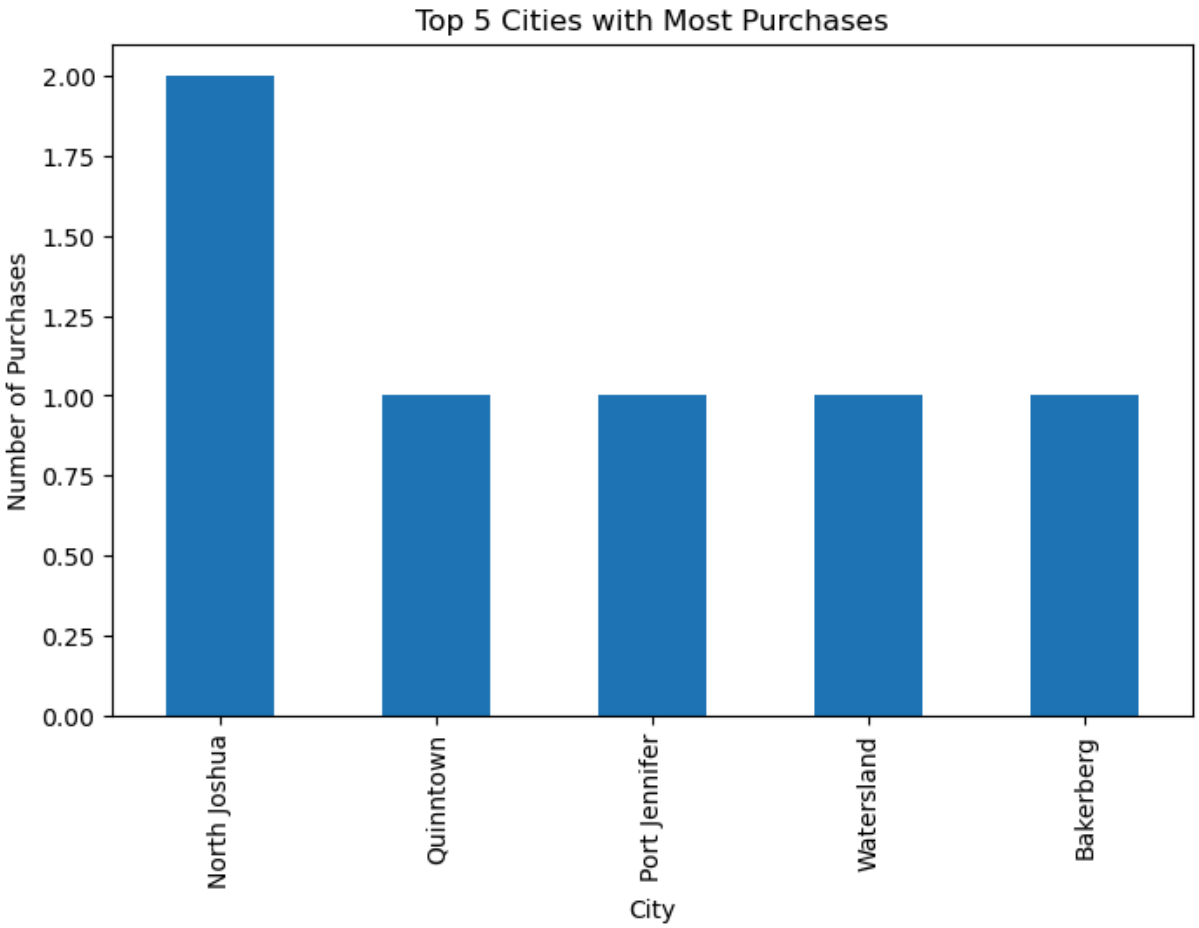
Out[46]:
```
'P047'
```

*Task6: Transform Age into groups (Teen, Young, Adult, Senior). *

age column not present

*Task7: Visualize top 5 cities with the most purchases. *

In [54]:
```
top_cities = customers['City'].value_counts().head(5)
plt.figure(figsize=(8,5))
top_cities.plot(kind='bar')
plt.title("Top 5 Cities with Most Purchases")
plt.xlabel("City")
plt.ylabel("Number of Purchases")
plt.show()
```



*Task8: How many customers purchased each product? (Use Product_ID to link with Products dataset.) *

In [55]:
```
# Merge the customers and products DataFrames based on Product_ID
merged = pd.merge(customers, products, on='Product_ID')
## Count how many customers purchased each product
merged.groupby('Product_Name')['Customer_ID'].count()
```

Out[55]:
```
Product_Name
Activity      1
Always        1
Around        3
Art           1
Artist        1
             ..
Together      1
Understand    3
Whether       1
White         2
Why           1
Name: Customer_ID, Length: 63, dtype: int64
```

*Task9: Create a report showing: Customer_Name — Product_Name — Quantity — Price — Final_Amount *

In [56]:
```
## Create a report with customer and product details
report = merged[['Customer_Name','Product_Name','Quantity','Price','Final_Price']]
## Calculate the total amount each customer paid for each product
report['Final_Amount'] = report['Quantity'] * report['Final_Price']
report
```

```
C:\Users\PMYLS\AppData\Local\Temp\ipykernel_17160\2005742380.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  report['Final_Amount'] = report['Quantity'] * report['Final_Price']
```

Out[56]:

	Customer_Name	Product_Name	Quantity	Price	Final_Price	Final_Amount
0	Joan Johnson	Half	181	367.88	349.4860	63256.9660
1	Daniel Welch	Position	675	279.93	265.9335	179505.1125
2	Kristin Wolf	Suffer	639	376.83	376.8300	240794.3700
3	Kimberly Johnson	Government	657	362.28	344.1660	226117.0620
4	Juan Woodward	Major	23	251.94	239.3430	5504.8890
...	...	...	...	...	...	...
95	Courtney Watson	New	143	377.19	377.1900	53938.1700
96	Cameron Ewing	Suffer	639	376.83	376.8300	240794.3700
97	Gary Gonzalez II	Republican	102	217.29	195.5610	19947.2220
98	Lauren Fields	Person	48	266.26	226.3210	10863.4080
99	Zachary Murphy	Realize	878	249.81	212.3385	186433.2030

100 rows × 6 columns

*Task10: Which customers purchased products with discounts? *

In [57]:
```
# Select customers who received a discount
discount_customers = merged[merged['Discount'] > 0]
discount_customers[['Customer_Name','Product_Name','Discount']]## Display the names
```

Out[57]:

	Customer_Name	Product_Name	Discount
0	Joan Johnson	Half	5.0
1	Daniel Welch	Position	5.0
3	Kimberly Johnson	Government	5.0
4	Juan Woodward	Major	5.0
6	Dr. Peggy Hanson	Own	5.0
...	...	...	...
92	Lauren Hernandez	Artist	15.0
94	Austin Mejia	Sit	10.0
97	Gary Gonzalez II	Republican	10.0
98	Lauren Fields	Person	15.0
99	Zachary Murphy	Realize	15.0

67 rows × 3 columns

In [ ]: