Complete Guide: Resolving the Retail Sales Dataset Issue Using Pandas

# Applying the 5 Why’s Methodology:

# Why has the total revenue decreased? → The number of products sold has declined in the past month.

# Why has the number of products sold declined? → The demand for certain product categories (e.g., Electronics and Home Decor) has dropped significantly.

# Why has the demand for these product categories dropped? → Customers have been purchasing from competitors who offer discounts and promotions.

# Why are competitors offering better discounts and promotions? → They are leveraging seasonal sales and better supply chain management, reducing their costs.

# Why is our store unable to match the competitors' offers? → Higher procurement costs and lack of promotional campaigns have impacted pricing competitiveness.

# Step 1: Load the Dataset

import pandas as pd  
df = pd.read\_csv("retail\_sales\_data.csv")  
df.head()

# Step 2: Perform Initial Data Exploration

df.info()  
df.isnull().sum()  
df.describe()

# Step 3: Analyze Sales Trends

import matplotlib.pyplot as plt  
df["Date"] = pd.to\_datetime(df["Date"])  
monthly\_sales = df.groupby(df["Date"].dt.to\_period("M"))["Total\_Revenue"].sum()  
monthly\_sales.plot(marker='o')  
plt.title("Monthly Sales Revenue Trend")  
plt.xlabel("Month")  
plt.ylabel("Total Revenue")  
plt.show()

# Step 4: Identify the Decline in Product Sales

category\_sales = df.groupby("Product\_Category")["Total\_Revenue"].sum().sort\_values(ascending=False)  
category\_sales.plot(kind="bar", title="Total Revenue by Product Category")  
plt.xlabel("Product Category")  
plt.ylabel("Total Revenue")  
plt.show()

# Step 5: Compare Sales Across Different Regions

region\_sales = df.groupby("Region")["Total\_Revenue"].sum()  
region\_sales.plot(kind="bar", title="Total Revenue by Region")  
plt.xlabel("Region")  
plt.ylabel("Total Revenue")  
plt.show()

# Step 6: Investigate Competitor Impact (Discounted Prices)

avg\_price\_per\_category = df.groupby("Product\_Category")["Unit\_Price"].mean()  
avg\_price\_per\_category.plot(kind="bar", title="Average Price per Product Category")  
plt.xlabel("Product Category")  
plt.ylabel("Average Unit Price")  
plt.show()

# Step 7: Check the Effect of Promotions

df["Month"] = df["Date"].dt.month  
sales\_before\_promo = df[df["Month"] < 3].groupby("Product\_Category")["Total\_Revenue"].sum()  
sales\_after\_promo = df[df["Month"] >= 3].groupby("Product\_Category")["Total\_Revenue"].sum()  
promo\_comparison = pd.DataFrame({"Before Promo": sales\_before\_promo, "After Promo": sales\_after\_promo})  
promo\_comparison.plot(kind="bar", title="Sales Before and After Promotions")  
plt.xlabel("Product Category")  
plt.ylabel("Total Revenue")  
plt.show()

# Step 8: Optimize Pricing Strategy

df["Suggested\_Price"] = df["Unit\_Price"] \* 0.9  
df[["Product\_Name", "Unit\_Price", "Suggested\_Price"]].head()

# Step 9: Implement Customer Loyalty Programs

top\_customers = df.groupby("Customer\_ID")["Total\_Revenue"].sum().sort\_values(ascending=False).head(10)  
print(top\_customers)

# Step 10: Final Recommendations

1. Adjust Pricing: Reduce unit prices by 10% on affected products.  
2. Promotions & Discounts: Offer targeted promotions in underperforming categories.  
3. Regional Targeting: Increase marketing in regions with declining sales.  
4. Customer Engagement: Implement a loyalty program for frequent customers.  
5. Competitive Monitoring: Regularly check competitor pricing and discounts.