SQL AND PYTHON

Name: Jaavanaika L

Step 1: Clean Data in SQL (SQLite)

1. Remove Null or Missing Records

In DB Browser > Execute SQL, run:

```
DELETE FROM superstore
WHERE "Order ID" IS NULL
  OR "Category" IS NULL
  OR "Sales" IS NULL
  OR "Profit" IS NULL;
```

2. Remove Duplicates (Based on Order ID)

Since SQLite doesn’t have ROW_NUMBER(), use this workaround:

```
DELETE FROM superstore
WHERE rowid NOT IN (
   SELECT MIN(rowid)
   FROM superstore
   GROUP BY "Order ID"
);
```

3. Verify Cleaned Table

Check the number of rows after cleaning:

```
SELECT COUNT(*) FROM superstore;
```

4. Export Cleaned Table (Optional)

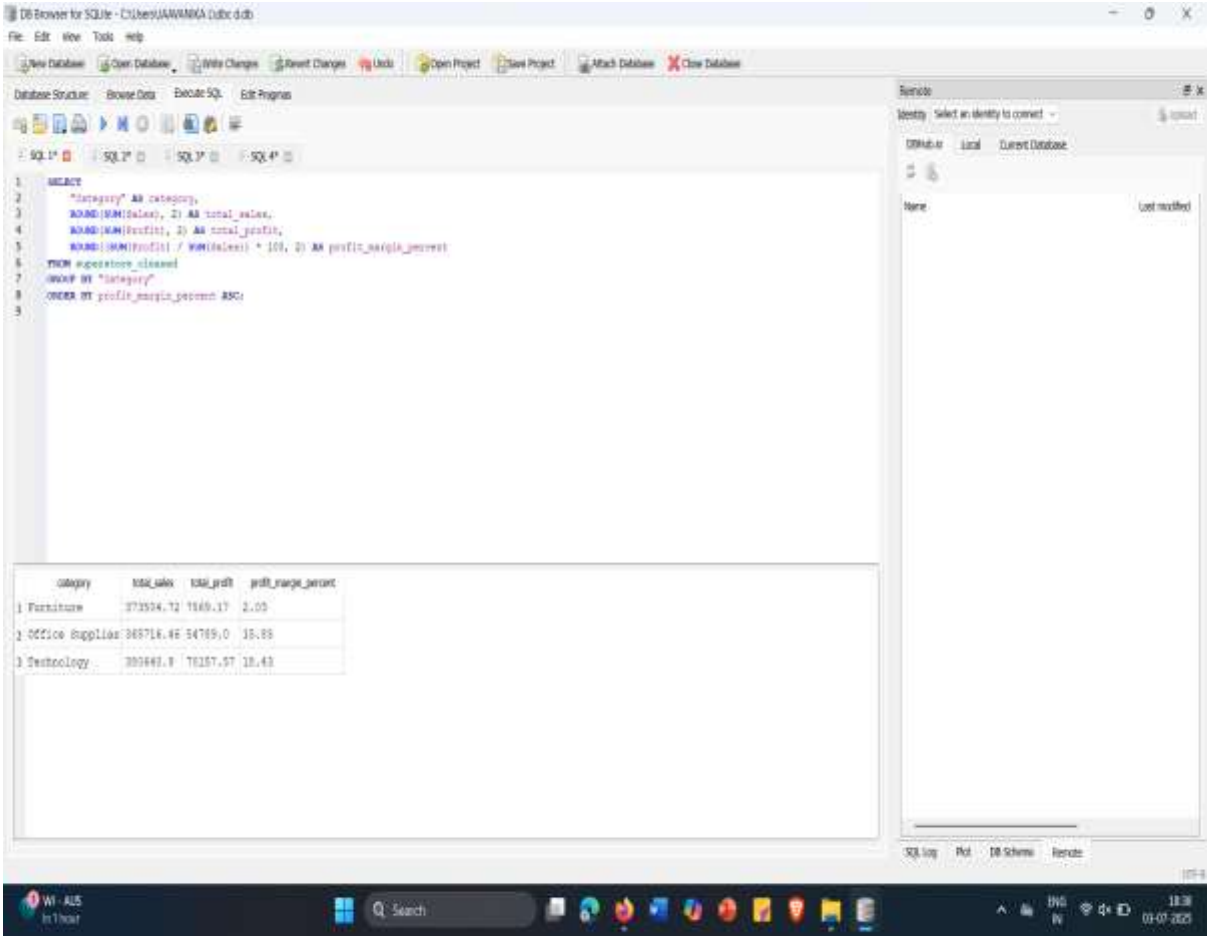
You can now:

- Go to File > Export > Table as CSV
- Save as superstore_cleaned.csv
  (You'll use this cleaned CSV in Python and Tableau)

Step 2: SQL Profitability Analysis (Clean Data)

A. Profit by Category

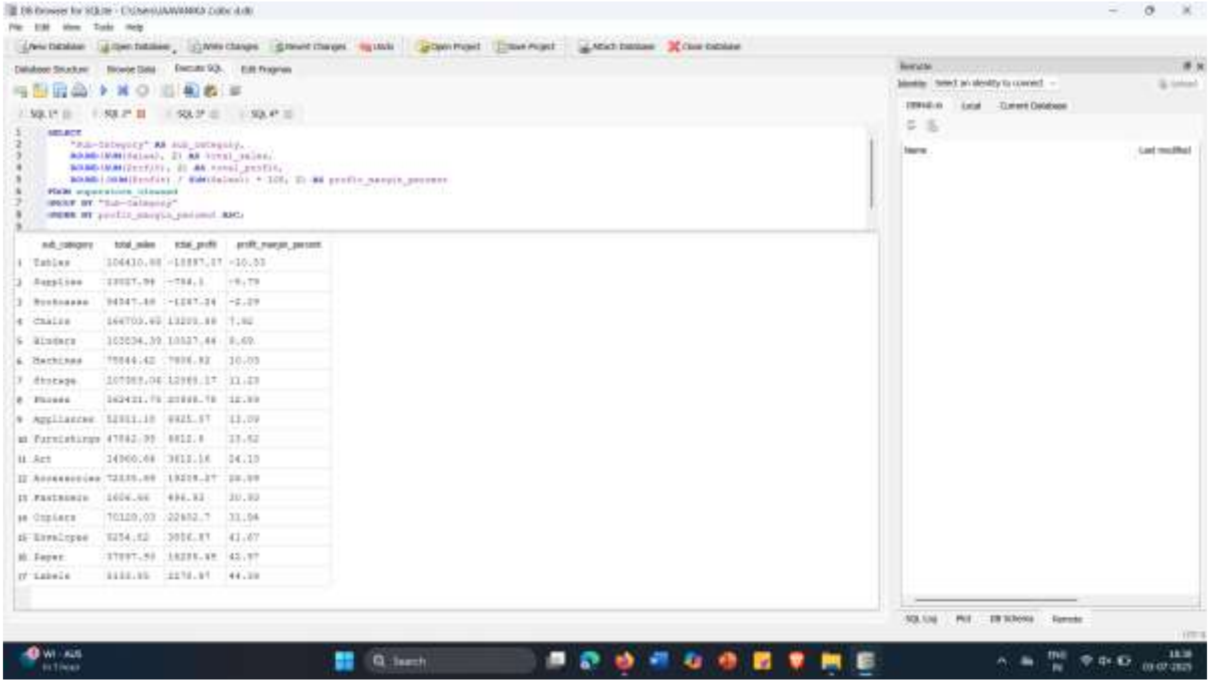
```
SELECT
  "Category" AS category,
  ROUND(SUM(Sales), 2) AS total_sales,
  ROUND(SUM(Profit), 2) AS total_profit,
  ROUND((SUM(Profit) / SUM(Sales)) * 100, 2) AS profit_margin_percent
FROM superstore_cleaned
GROUP BY "Category"
ORDER BY profit_margin_percent ASC;
```



B. Profit by Sub-Category

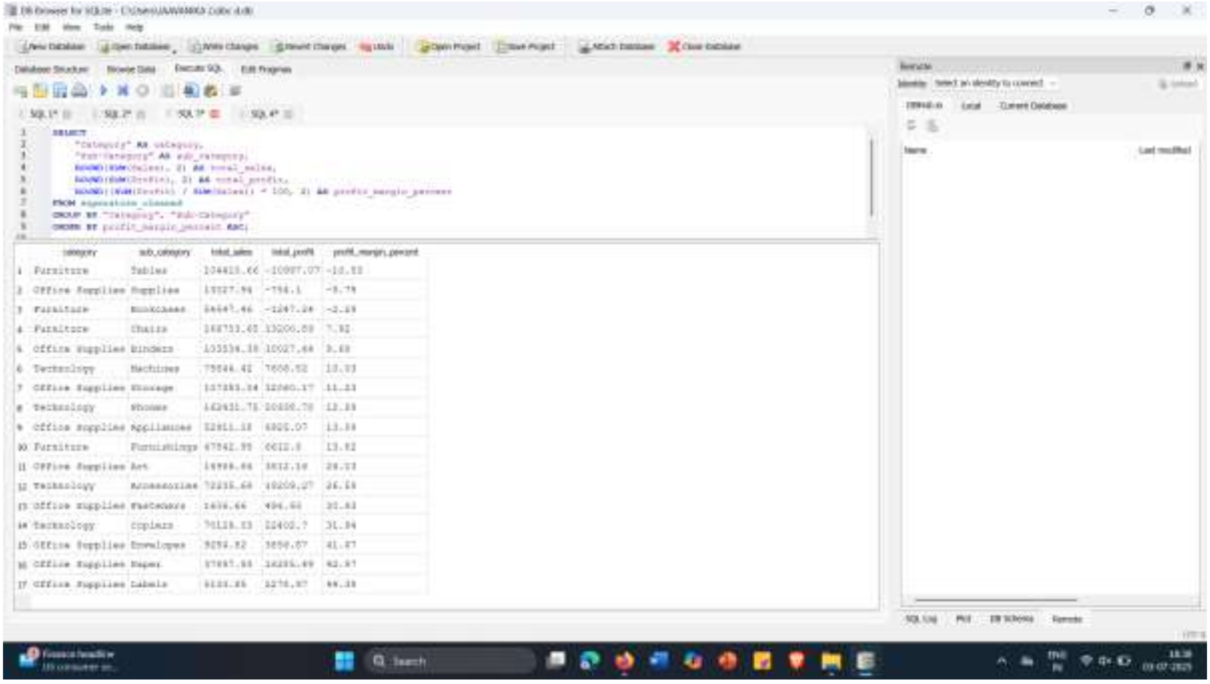
```
SELECT
  "Sub-Category" AS sub_category,
  ROUND(SUM(Sales), 2) AS total_sales,
  ROUND(SUM(Profit), 2) AS total_profit,
  ROUND((SUM(Profit) / SUM(Sales)) * 100, 2) AS profit_margin_percent
FROM superstore_cleaned
GROUP BY "Sub-Category"
```

```
ORDER BY profit_margin_percent ASC;
```



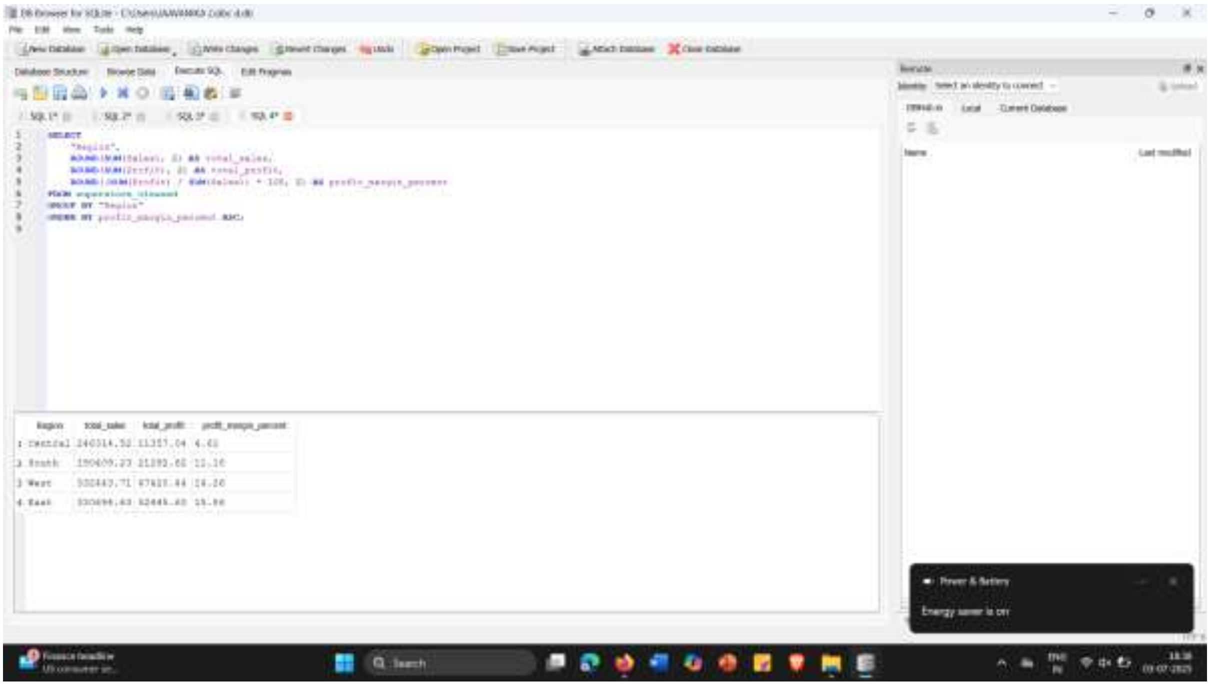
C. Profit by Category + Sub-Category

```
SELECT
  "Category" AS category,
  "Sub-Category" AS sub_category,
  ROUND(SUM(Sales), 2) AS total_sales,
  ROUND(SUM(Profit), 2) AS total_profit,
  ROUND((SUM(Profit) / SUM(Sales)) * 100, 2) AS profit_margin_percent
FROM superstore
GROUP BY "Category", "Sub-Category"
ORDER BY profit_margin_percent ASC;
```



D. Profit by Region

```
SELECT
  "Region",
  ROUND(SUM(Sales), 2) AS total_sales,
  ROUND(SUM(Profit), 2) AS total_profit,
  ROUND((SUM(Profit) / SUM(Sales)) * 100, 2) AS profit_margin_percent
FROM superstore
GROUP BY "Region"
ORDER BY profit_margin_percent ASC;
```



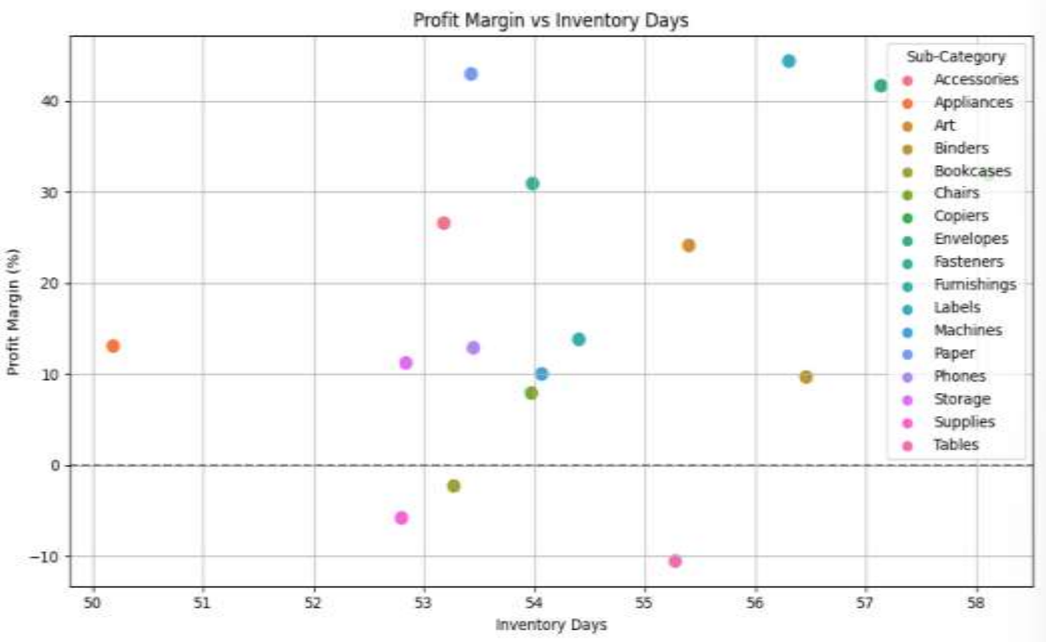
Step 3: Python – Correlation Between Inventory Days & Profitability Visualizations (Python/Seaborn/Matplotlib)

```
import pandas as pd
import numpy as np

# Load cleaned data
df = pd.read_csv("superstore_cleaned.csv")

# Simulate Inventory Days (since not in original dataset)
np.random.seed(42)
df["Inventory Days"] = np.random.randint(10, 101, size=len(df))

# Convert date columns to datetime
df["Order Date"] = pd.to_datetime(df["Order Date"])
df["Month"] = df["Order Date"].dt.month
df["Season"] = df["Month"].map({
  12: "Winter", 1: "Winter", 2: "Winter",
  3: "Spring", 4: "Spring", 5: "Spring",
  6: "Summer", 7: "Summer", 8: "Summer",
  9: "Fall", 10: "Fall", 11: "Fall"
})
```

1. 📈 **Scatter Plot: Inventory Days vs Profit Margin**

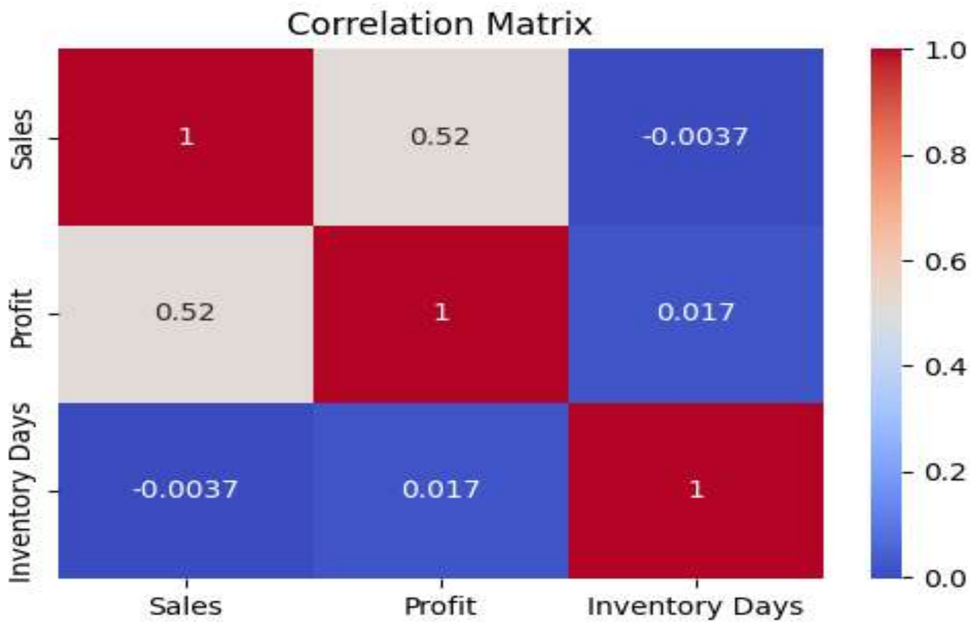
```
grouped = df.groupby("Sub-Category").agg({
  "Sales": "sum",
  "Profit": "sum",
```

```
    "Inventory Days": "mean"
}).reset_index()
grouped["Profit Margin (%)"] = (grouped["Profit"] / grouped["Sales"]) * 100

import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10,6))
sns.scatterplot(data=grouped, x="Inventory Days", y="Profit Margin (%)",
hue="Sub-Category", s=100)
plt.title("Profit Margin vs Inventory Days")
plt.axhline(0, linestyle='--', color='gray')
plt.grid(True)
plt.tight_layout()
plt.show()
```



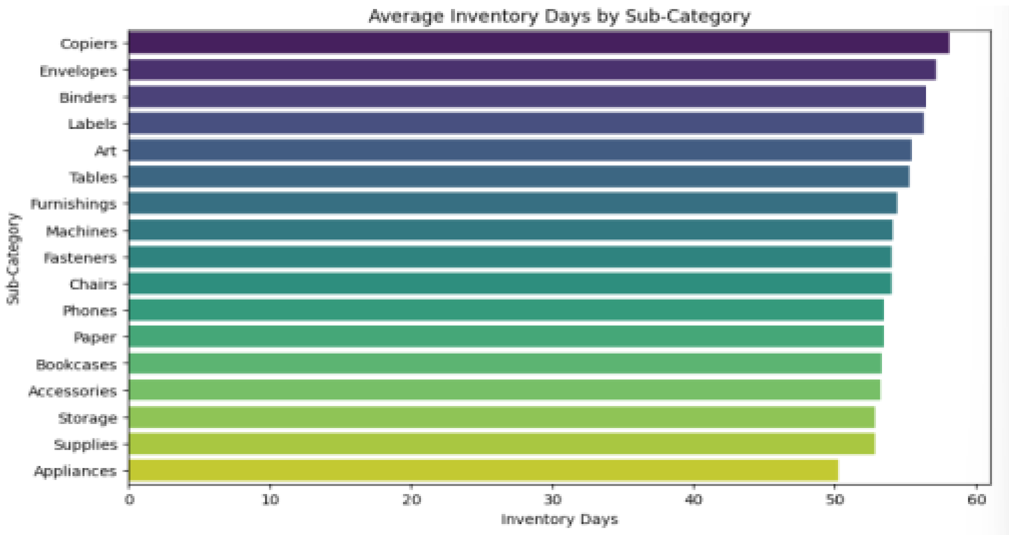
2. 🔥 **Heatmap: Inventory Days vs Profit Margin (Correlation Matrix)**

```
# Correlation between numeric columns
plt.figure(figsize=(6,4))
sns.heatmap(df[["Sales", "Profit", "Inventory Days"]].corr(), annot=True,
cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```



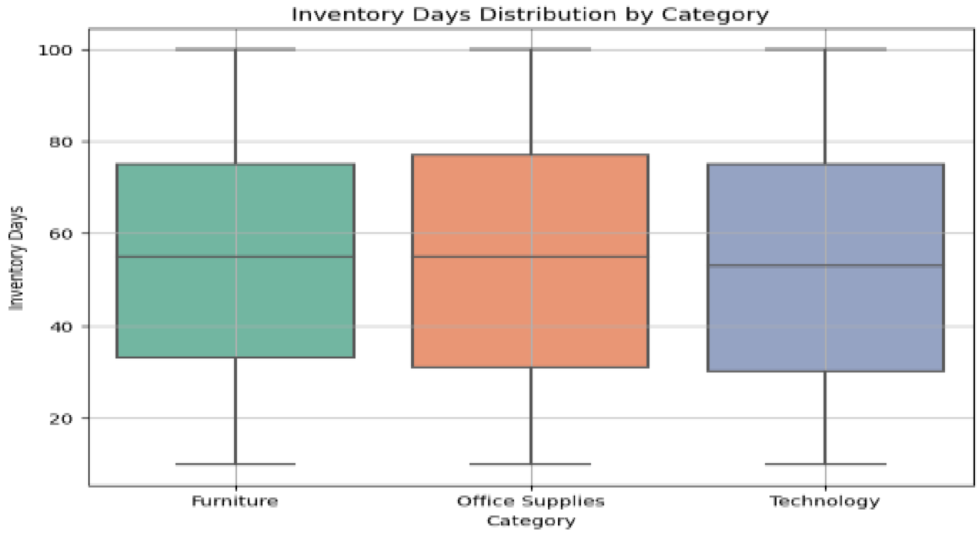
3. 📉 **Bar Chart: Sub-Categories with Highest Inventory Days**

```
inv_days = df.groupby("Sub-Category")["Inventory
Days"].mean().sort_values(ascending=False)

plt.figure(figsize=(10,6))
sns.barplot(x=inv_days.values, y=inv_days.index, palette="viridis")
plt.title("Average Inventory Days by Sub-Category")
plt.xlabel("Inventory Days")
plt.ylabel("Sub-Category")
plt.show()
```



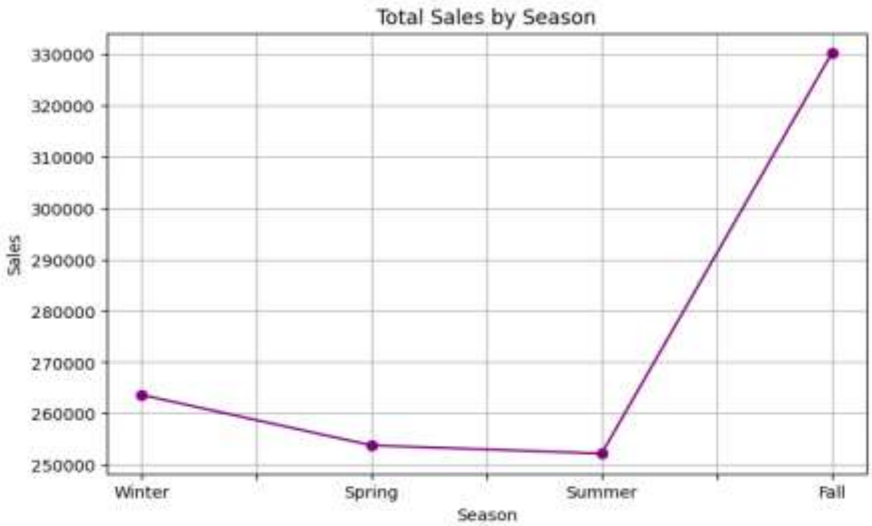
4. 📦 **Box Plot: Inventory Days Distribution by Category**

```
plt.figure(figsize=(8,6))
sns.boxplot(data=df, x="Category", y="Inventory Days", palette="Set2")
plt.title("Inventory Days Distribution by Category")
plt.grid(True)
plt.show()
```



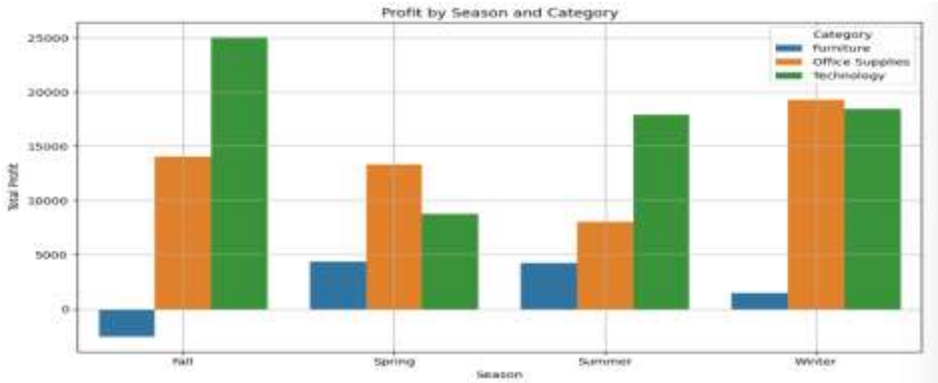
5. 🌧️ **Seasonal Sales Trend (Line Plot by Season)**

```
season_sales = df.groupby("Season")["Sales"].sum().reindex(["Winter", "Spring", "Summer", "Fall"])
plt.figure(figsize=(8,5))
season_sales.plot(kind="line", marker='o', color="purple")
plt.title("Total Sales by Season")
plt.xlabel("Season")
plt.ylabel("Sales")
plt.grid(True)
plt.show()
```



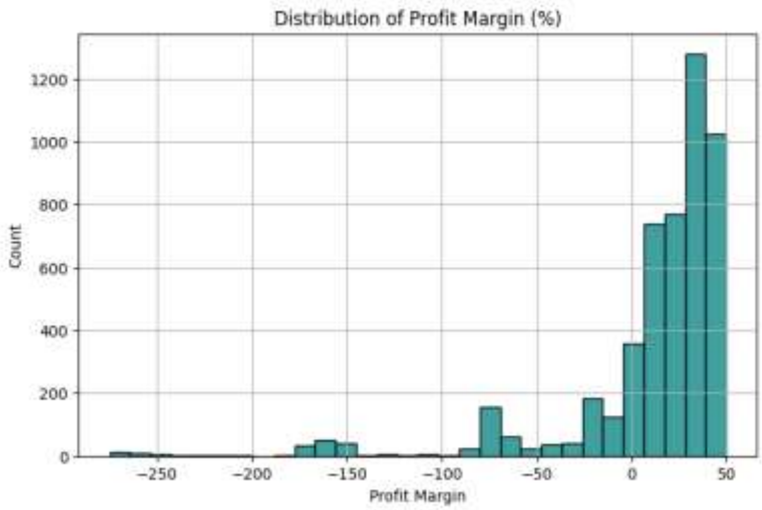
6. 💸 **Profitability by Season & Category (Grouped Bar Chart)**

```
season_cat = df.groupby(["Season",
"Category"])["Profit"].sum().reset_index()
plt.figure(figsize=(10,6))
sns.barplot(data=season_cat, x="Season", y="Profit", hue="Category")
plt.title("Profit by Season and Category")
plt.ylabel("Total Profit")
plt.grid(True)
plt.tight_layout()
plt.show()
```



7. 🧮 **Histogram: Distribution of Profit Margins**

```
df["Profit Margin (%)"] = (df["Profit"] / df["Sales"]) * 100
plt.figure(figsize=(8,5))
sns.histplot(df["Profit Margin (%)"], bins=30, color="teal")
plt.title("Distribution of Profit Margin (%)")
plt.xlabel("Profit Margin")
plt.grid(True)
plt.show()
```



8. 🔍 **Sub-Category Level Comparison (Bar Chart)**

```
sub_profit = df.groupby("Sub-Category")["Profit"].sum().sort_values()
```

```
plt.figure(figsize=(10,6))
sns.barplot(x=sub_profit.values, y=sub_profit.index, palette="coolwarm")
plt.title("Total Profit by Sub-Category")
plt.xlabel("Profit")
plt.ylabel("Sub-Category")
plt.axvline(0, color="black", linestyle="--")
plt.show()
```