```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

file_path = r"C:\Users\Dell\Desktop\retail_data.xlsx"
# Load the Excel file
df = pd.read_excel(file_path)
# Show the first few rows
print(df.head())
```

```
   Row ID        Order ID Order Date  Ship Date       Ship Mode
Customer ID  \
0       1  CA-2013-152156 2013-11-09 2013-11-12    Second Class
CG-12520
1       2  CA-2013-152156 2013-11-09 2013-11-12    Second Class
CG-12520
2       3  CA-2013-138688 2013-06-13 2013-06-17    Second Class
DV-13045
3       4  US-2012-108966 2012-10-11 2012-10-18  Standard Class
SO-20335
4       5  US-2012-108966 2012-10-11 2012-10-18  Standard Class
SO-20335

     Customer Name    Segment        Country
City  ...  \
0      Claire Gute   Consumer  United States        Henderson  ...

1      Claire Gute   Consumer  United States        Henderson  ...

2  Darrin Van Huff  Corporate  United States      Los Angeles  ...

3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...

4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...


  Postal Code  Region       Product ID         Category Sub-Category
\
0       42420   South  FUR-BO-10001798        Furniture    Bookcases

1       42420   South  FUR-CH-10000454        Furniture       Chairs

2       90036    West  OFF-LA-10000240  Office Supplies       Labels

3       33311   South  FUR-TA-10000577        Furniture       Tables

4       33311   South  OFF-ST-10000760  Office Supplies      Storage


                                        Product Name     Sales
Quantity  \
0                  Bush Somerset Collection Bookcase  261.9600
2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400
3
```

```
2  Self-Adhesive Address Labels for Typewriters b...   14.6200
2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775
5
4                     Eldon Fold 'N Roll Cart System   22.3680
2

   Discount    Profit
0      0.00   41.9136
1      0.00  219.5820
2      0.00    6.8714
3      0.45 -383.0310
4      0.20    2.5164

[5 rows x 21 columns]
```

```
df.head()
```

```
   Row ID        Order ID Order Date  Ship Date       Ship Mode
Customer ID  \
0       1  CA-2013-152156 2013-11-09 2013-11-12    Second Class
CG-12520
1       2  CA-2013-152156 2013-11-09 2013-11-12    Second Class
CG-12520
2       3  CA-2013-138688 2013-06-13 2013-06-17    Second Class
DV-13045
3       4  US-2012-108966 2012-10-11 2012-10-18  Standard Class
SO-20335
4       5  US-2012-108966 2012-10-11 2012-10-18  Standard Class
SO-20335

     Customer Name    Segment        Country
City  ...  \
0      Claire Gute   Consumer  United States        Henderson  ...

1      Claire Gute   Consumer  United States        Henderson  ...

2  Darrin Van Huff  Corporate  United States      Los Angeles  ...

3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...

4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  ...


  Postal Code  Region       Product ID         Category Sub-Category
\
0       42420   South  FUR-BO-10001798        Furniture    Bookcases

1       42420   South  FUR-CH-10000454        Furniture       Chairs

2       90036    West  OFF-LA-10000240  Office Supplies       Labels

3       33311   South  FUR-TA-10000577        Furniture       Tables

4       33311   South  OFF-ST-10000760  Office Supplies      Storage
```

```
                                        Product Name     Sales
Quantity  \
0                  Bush Somerset Collection Bookcase  261.9600
2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400
3
2  Self-Adhesive Address Labels for Typewriters b...   14.6200
2
3     Bretford CR4500 Series Slim Rectangular Table  957.5775
5
4                    Eldon Fold 'N Roll Cart System   22.3680
2

   Discount    Profit
0      0.00   41.9136
1      0.00  219.5820
2      0.00    6.8714
3      0.45 -383.0310
4      0.20    2.5164

[5 rows x 21 columns]

df.tail()

      Row ID        Order ID Order Date  Ship Date       Ship
Mode  \
9989    9990  CA-2011-110422 2011-01-22 2011-01-24    Second Class

9990    9991  CA-2014-121258 2014-02-27 2014-03-04  Standard Class

9991    9992  CA-2014-121258 2014-02-27 2014-03-04  Standard Class

9992    9993  CA-2014-121258 2014-02-27 2014-03-04  Standard Class

9993    9994  CA-2014-119914 2014-05-05 2014-05-10    Second Class


     Customer ID     Customer Name   Segment        Country
City  ...  \
9989    TB-21400  Tom Boeckenhauer  Consumer  United States
Miami  ...
9990    DB-13060       Dave Brooks  Consumer  United States   Costa
Mesa  ...
9991    DB-13060       Dave Brooks  Consumer  United States   Costa
Mesa  ...
9992    DB-13060       Dave Brooks  Consumer  United States   Costa
Mesa  ...
9993    CC-12220      Chris Cortes  Consumer  United States
Westminster  ...

     Postal Code  Region       Product ID         Category Sub-
Category  \
9989       33180   South  FUR-FU-10001889        Furniture
```

```
Furnishings
9990       92627    West  FUR-FU-10000747        Furniture
Furnishings
9991       92627    West  TEC-PH-10003645       Technology
Phones
9992       92627    West  OFF-PA-10004041  Office Supplies
Paper
9993       92683    West  OFF-AP-10002684  Office Supplies
Appliances

                                              Product Name    Sales
Quantity  \
9989                               Ultra Door Pull Handle   25.248
3
9990  Tenex B1-RE Series Chair Mats for Low Pile Car...   91.960
2
9991                                Aastra 57i VoIP phone  258.576
2
9992  It's Hot Message Books with Stickers, 2 3/4" x 5"   29.600
4
9993  Acco 7-Outlet Masterpiece Power Center, Wihtou...  243.160
2

      Discount   Profit
9989       0.2   4.1028
9990       0.0  15.6332
9991       0.2  19.3932
9992       0.0  13.3200
9993       0.0  72.9480

[5 rows x 21 columns]
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
```

```
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: datetime64[ns](2), float64(3), int64(3), object(13)
memory usage: 1.6+ MB
```

```
df.isna().sum()
```

```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
Segment          0
Country          0
City             0
State            0
Postal Code      0
Region           0
Product ID       0
Category         0
Sub-Category     0
Product Name     0
Sales            0
Quantity         0
Discount         0
Profit           0
dtype: int64
```

```
df.describe().round()
```

```
       Row ID                      Order Date
Ship Date  \
count  9994.0                            9994
9994
mean   4998.0  2013-04-30 19:20:02.401441024  2013-05-04
18:20:49.229537792
min       1.0            2011-01-04 00:00:00            2011-01-08
00:00:00
25%    2499.0            2012-05-23 00:00:00            2012-05-27
00:00:00
50%    4998.0            2013-06-27 00:00:00            2013-06-30
00:00:00
75%    7496.0            2014-05-15 00:00:00            2014-05-19
00:00:00
max    9994.0            2014-12-31 00:00:00            2015-01-06
00:00:00
std    2885.0                            NaN
NaN
```

```
       Postal Code    Sales  Quantity  Discount  Profit
count       9994.0   9994.0    9994.0    9994.0  9994.0
mean       55190.0    230.0       4.0       0.0    29.0
min         1040.0      0.0       1.0       0.0 -6600.0
25%        23223.0     17.0       2.0       0.0     2.0
50%        56430.0     54.0       3.0       0.0     9.0
75%        90008.0    210.0       5.0       0.0    29.0
max        99301.0  22638.0      14.0       1.0  8400.0
std        32064.0    623.0       2.0       0.0   234.0
```

```
df.nunique()
```

```
Row ID           9994
Order ID         5009
Order Date       1238
Ship Date        1334
Ship Mode           4
Customer ID       793
Customer Name     793
Segment             3
Country             1
City              531
State              49
Postal Code       631
Region              4
Product ID       1862
Category            3
Sub-Category       17
Product Name     1841
Sales            5825
Quantity           14
Discount           12
Profit           7287
dtype: int64
```

```
# Clean column names
df.columns = df.columns.str.strip().str.lower().str.replace(" ",
"_")
df.columns
```

```
Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'country', 'city',
'state',
       'postal_code', 'region', 'product_id', 'category', 'sub-
category',
       'product_name', 'sales', 'quantity', 'discount', 'profit'],
      dtype='object')
```

```
df['order_date'] = pd.to_datetime(df['order_date'], errors='coerce')
df['order_date']
```

```
0      2013-11-09
1      2013-11-09
2      2013-06-13
```

```
3      2012-10-11
4      2012-10-11
          ...
9989   2011-01-22
9990   2014-02-27
9991   2014-02-27
9992   2014-02-27
9993   2014-05-05
Name: order_date, Length: 9994, dtype: datetime64[ns]
```

```
# Drop rows with missing crucial data
df.dropna(subset=['sales', 'profit', 'order_date'], inplace=True)
```

```
# Feature engineering
df['month'] = df['order_date'].dt.month
df['month']
```

```
0       11
1       11
2        6
3       10
4       10
        ..
9989     1
9990     2
9991     2
9992     2
9993     5
Name: month, Length: 9994, dtype: int32
```

```
df['year'] = df['order_date'].dt.year
df['year']
```

```
0       2013
1       2013
2       2013
3       2012
4       2012
        ...
9989    2011
9990    2014
9991    2014
9992    2014
9993    2014
Name: year, Length: 9994, dtype: int32
```

```
df['quarter'] = df['order_date'].dt.quarter
df['quarter']
```

```
0       4
1       4
2       2
3       4
4       4
       ..
```

```
9989    1
9990    1
9991    1
9992    1
9993    2
Name: quarter, Length: 9994, dtype: int32
```

```
df['profit_margin'] = df['profit'] / df['sales']
df['profit_margin']
```

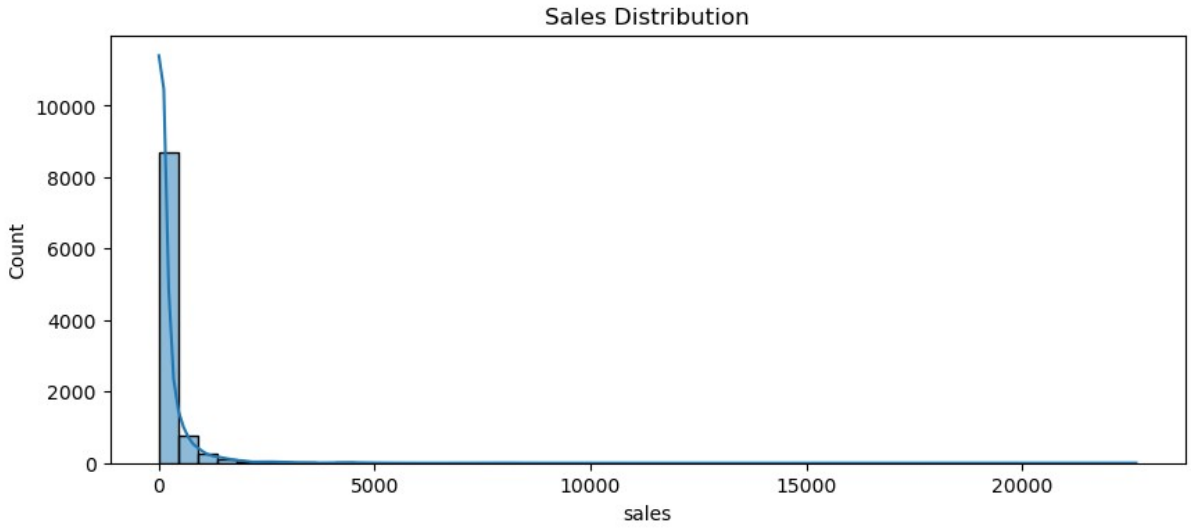
```
0       0.1600
1       0.3000
2       0.4700
3      -0.4000
4       0.1125
         ...
9989    0.1625
9990    0.1700
9991    0.0750
9992    0.4500
9993    0.3000
Name: profit_margin, Length: 9994, dtype: float64
```

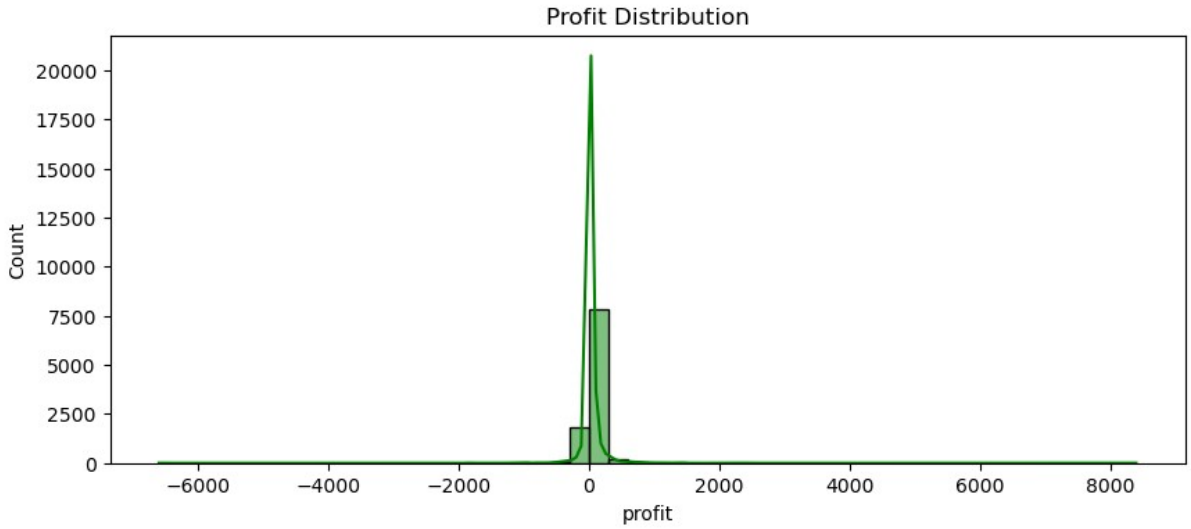
```
#value counts
print(df['category'].value_counts())
print(df['sub-category'].value_counts())
print(df['region'].value_counts())
```

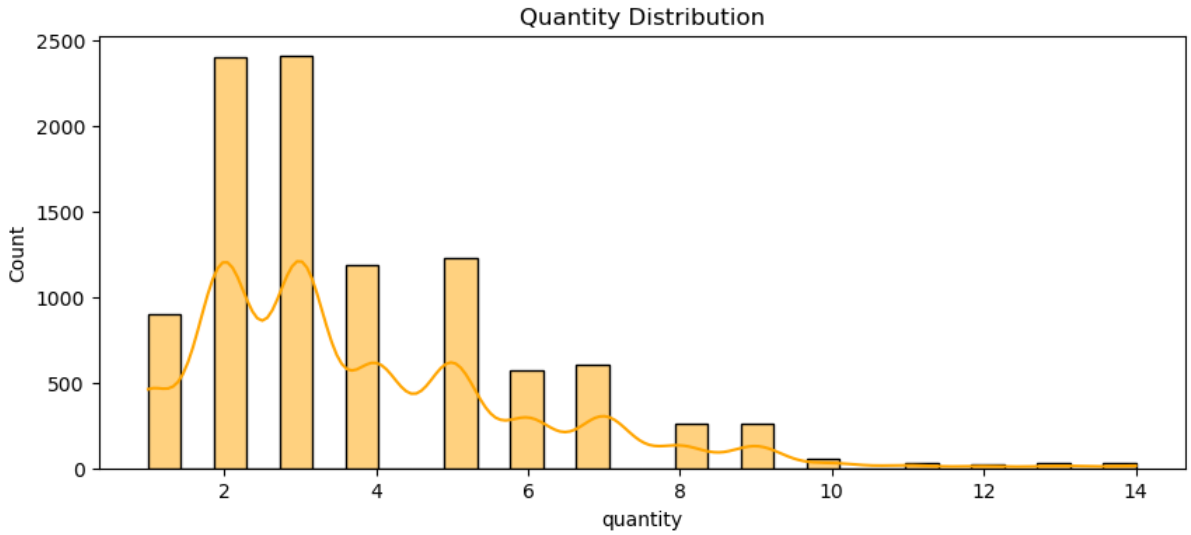
```
category
Office Supplies    6026
Furniture          2121
Technology         1847
Name: count, dtype: int64
sub-category
Binders        1523
Paper          1370
Furnishings     957
Phones          889
Storage         846
Art             796
Accessories     775
Chairs          617
Appliances      466
Labels          364
Tables          319
Envelopes       254
Bookcases       228
Fasteners       217
Supplies        190
Machines        115
Copiers          68
Name: count, dtype: int64
region
West       3203
```

```
East       2848
Central    2323
South      1620
Name: count, dtype: int64
```

```
#UNIVARIATE ANALYSIS (Single Variable)
#Distribution of Sales, Profit, Quantity
plt.figure(figsize=(10,4))
sns.histplot(df['sales'], bins=50, kde=True)
plt.title("Sales Distribution")
plt.show()

plt.figure(figsize=(10,4))
sns.histplot(df['profit'], bins=50, kde=True, color='green')
plt.title("Profit Distribution")
plt.show()

plt.figure(figsize=(10,4))
sns.histplot(df['quantity'], bins=30, kde=True, color='orange')
plt.title("Quantity Distribution")
plt.show()
```



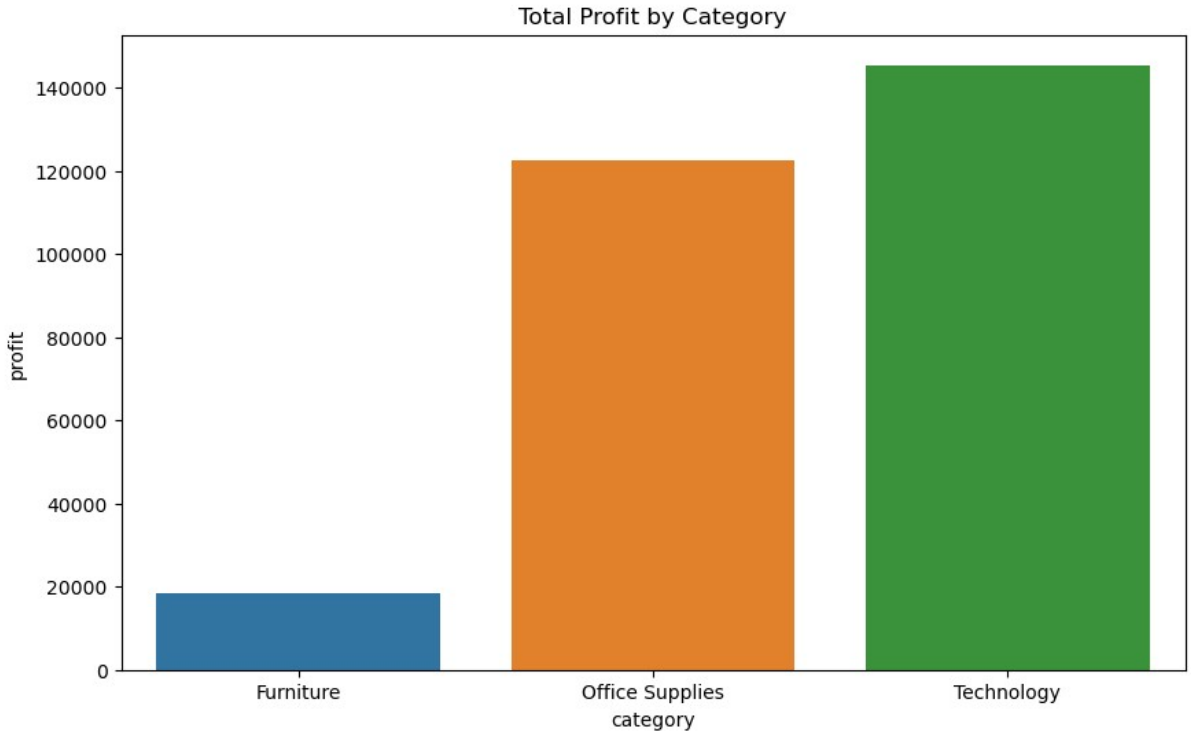




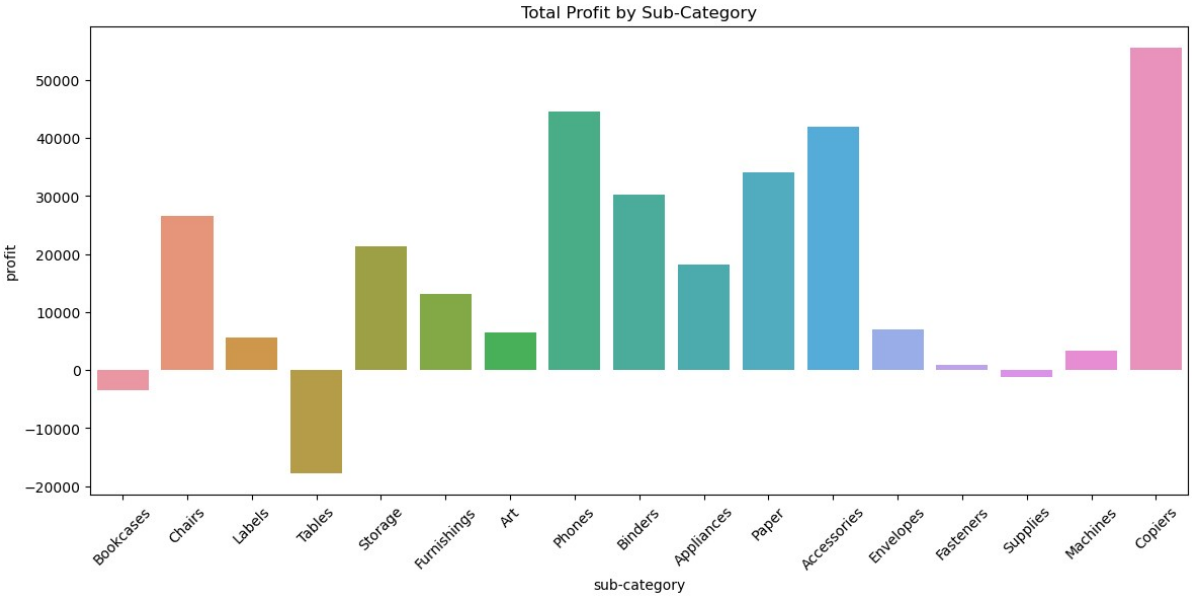

```
#BIVARIATE ANALYSIS (Relationships Between Two Variables)
#Profit vs Sales
plt.figure(figsize=(8,6))
sns.scatterplot(data=df, x='sales', y='profit', alpha=0.6)
plt.title("Profit vs Sales")
plt.show()

#Profit by Category/Sub-category
plt.figure(figsize=(10,6))
sns.barplot(data=df, x='category', y='profit', estimator=sum,
ci=None)
plt.title("Total Profit by Category")
plt.show()

plt.figure(figsize=(14,6))
sns.barplot(data=df, x='sub-category', y='profit', estimator=sum,
ci=None)
plt.title("Total Profit by Sub-Category")
plt.xticks(rotation=45)
```

```
plt.show()
```


```
C:\Users\Dell\AppData\Local\Temp\ipykernel_6996\4033568185.py:10:
FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same
effect.

  sns.barplot(data=df, x='category', y='profit', estimator=sum,
ci=None)
```



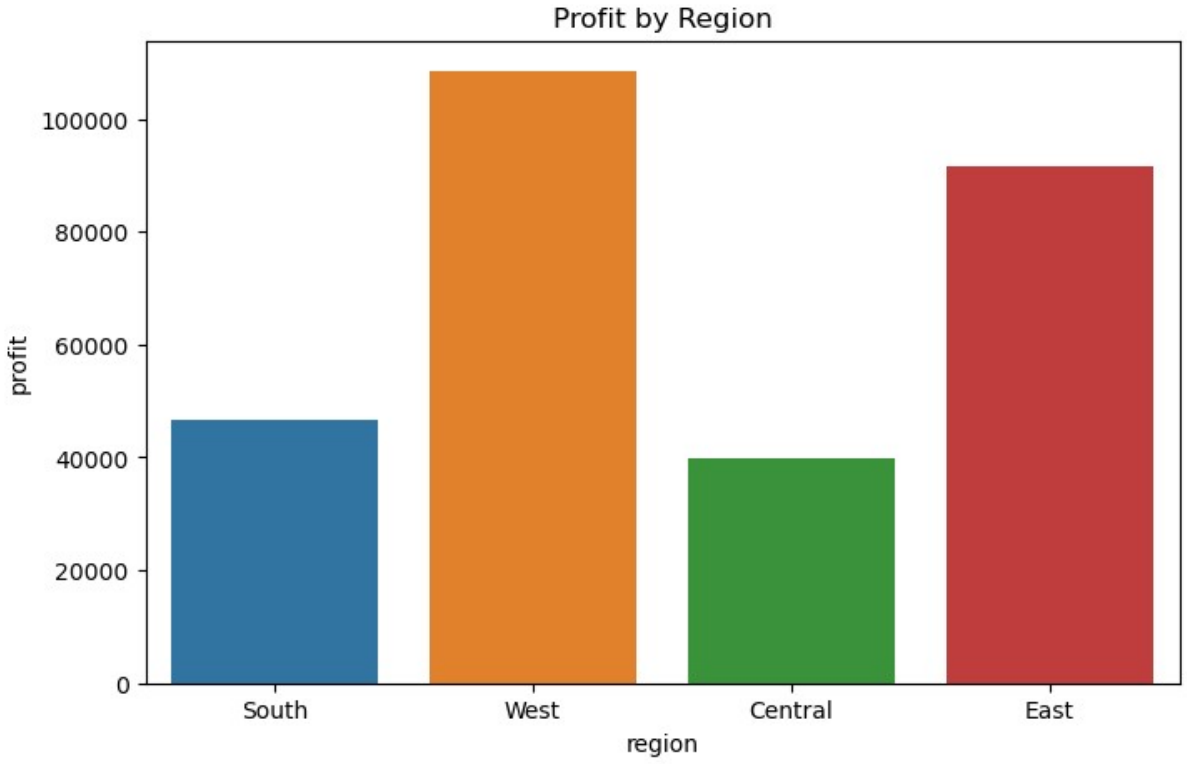

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_6996\4033568185.py:15:
FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same
effect.

  sns.barplot(data=df, x='sub-category', y='profit', estimator=sum,
ci=None)
```

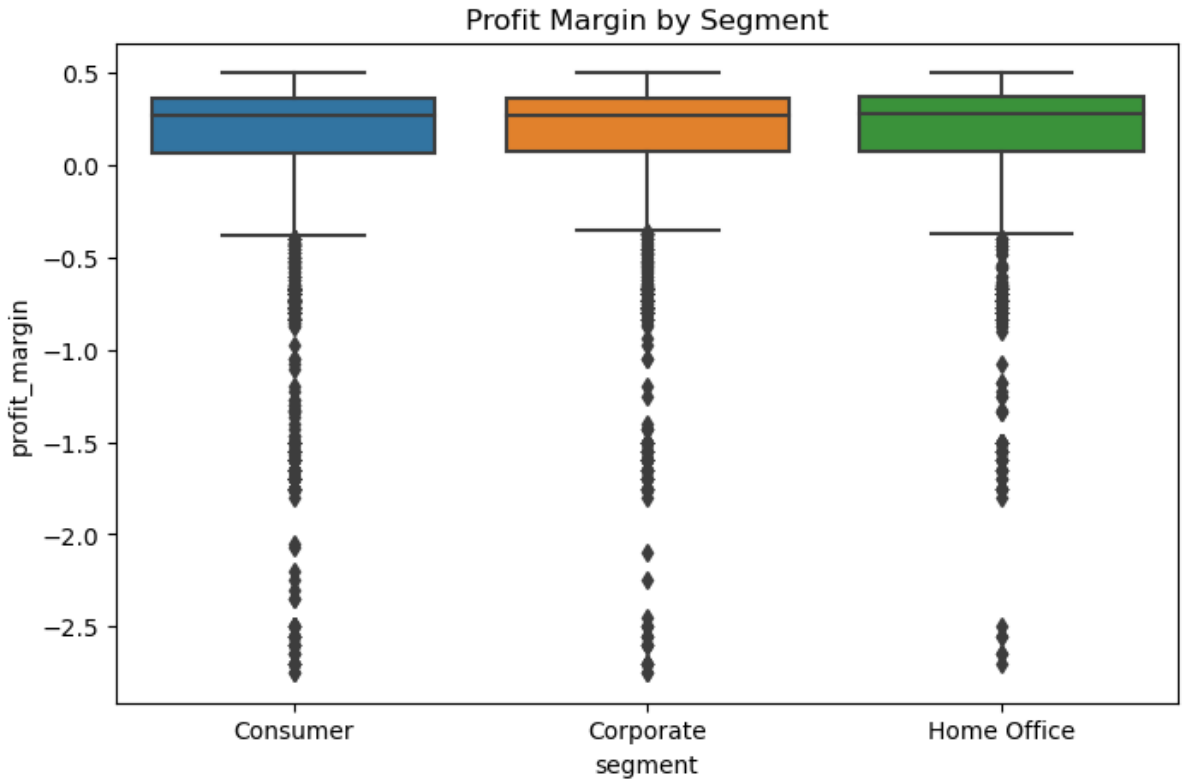



```python
#REGION & SEGMENT ANALYSIS
if 'region' in df.columns:
    plt.figure(figsize=(8,5))
```

```
    sns.barplot(data=df, x='region', y='profit', estimator=sum,
ci=None)
    plt.title("Profit by Region")
    plt.show()

if 'segment' in df.columns:
    plt.figure(figsize=(8,5))
    sns.boxplot(data=df, x='segment', y='profit_margin')
    plt.title("Profit Margin by Segment")
    plt.show()
```

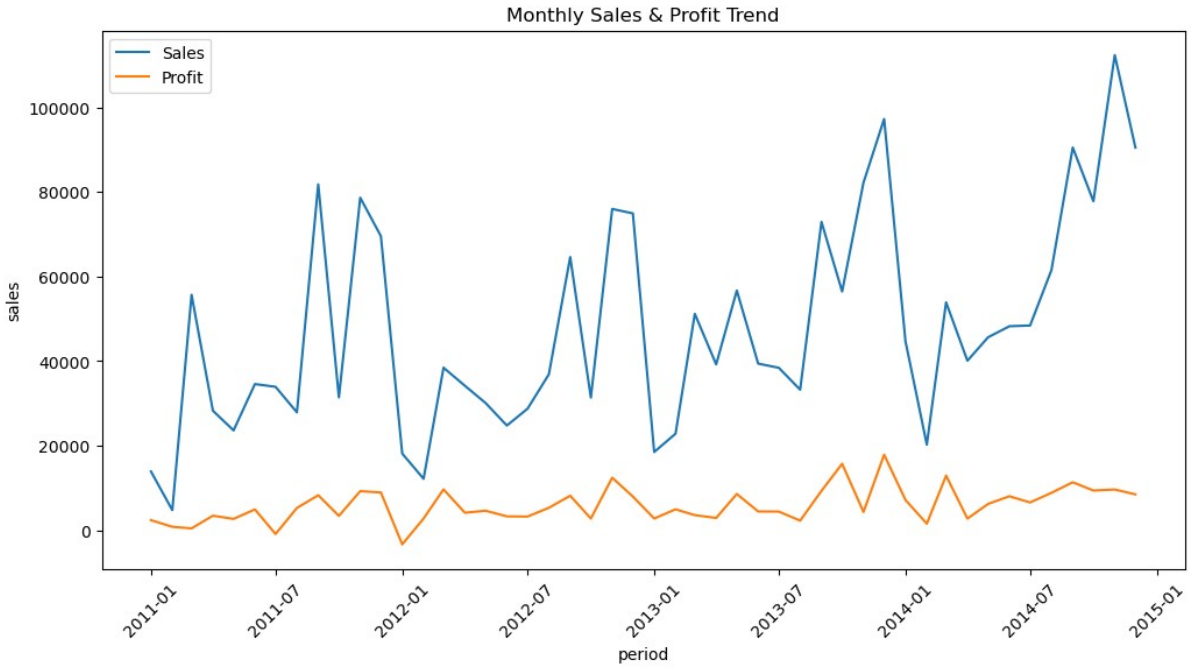
```
C:\Users\Dell\AppData\Local\Temp\ipykernel_6996\2423760571.py:4:
FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same
effect.

  sns.barplot(data=df, x='region', y='profit', estimator=sum,
ci=None)
```

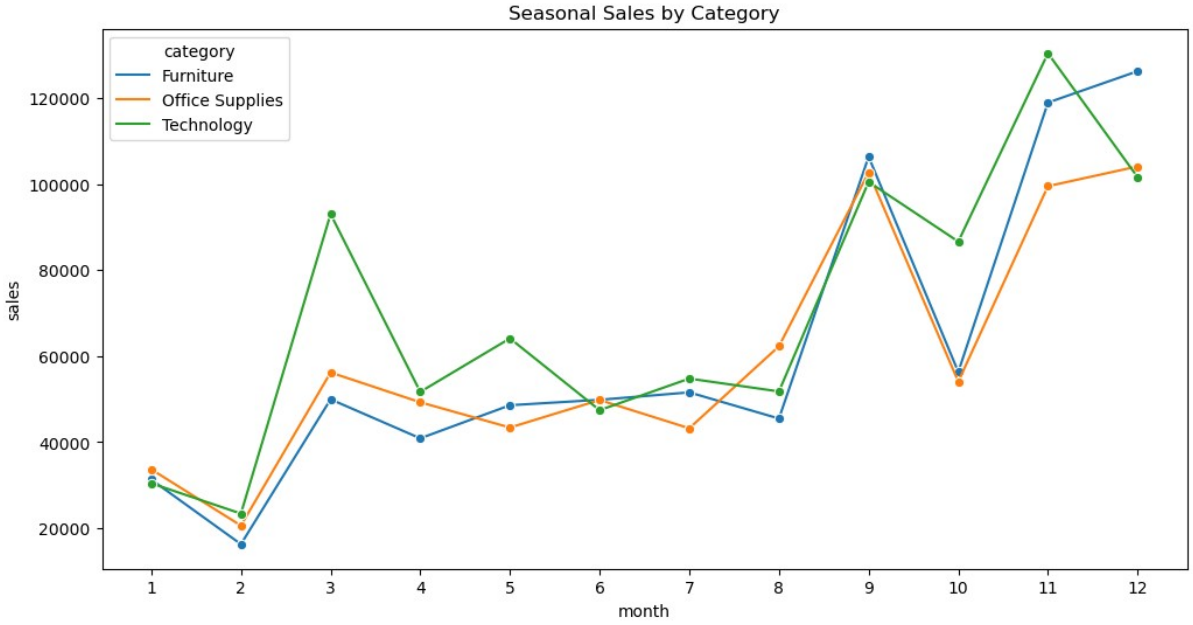


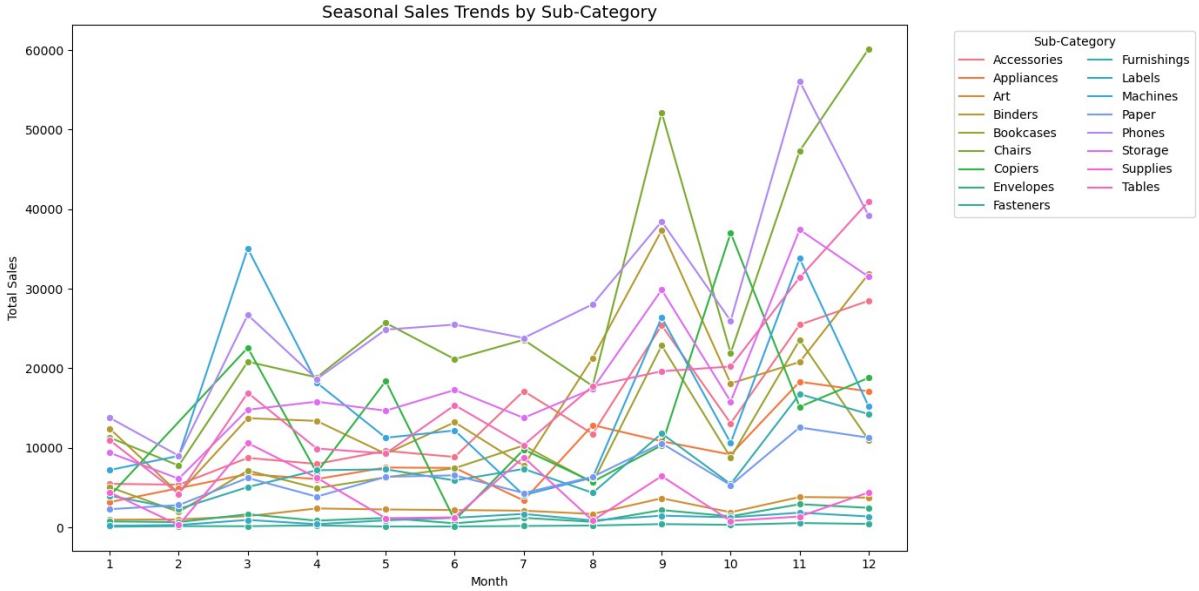



```
#TIME SERIES ANALYSIS
#Monthly Sales & Profit Trends
monthly = df.groupby(['year', 'month'])[['sales',
'profit']].sum().reset_index()
monthly['period'] = pd.to_datetime(monthly[['year',
'month']].assign(day=1))

plt.figure(figsize=(12,6))
sns.lineplot(data=monthly, x='period', y='sales', label='Sales')
sns.lineplot(data=monthly, x='period', y='profit', label='Profit')
plt.title("Monthly Sales & Profit Trend")
plt.xticks(rotation=45)
plt.show()

#Seasonal Trends by Category
seasonal = df.groupby(['category', 'month'])
[['sales']].sum().reset_index()

plt.figure(figsize=(12,6))
sns.lineplot(data=seasonal, x='month', y='sales', hue='category',
marker='o')
plt.title("Seasonal Sales by Category")
plt.xticks(range(1,13))
plt.show()


# Group sales by sub-category and month
seasonal_subcat = df.groupby(['sub-category', 'month'])
['sales'].sum().reset_index()
```

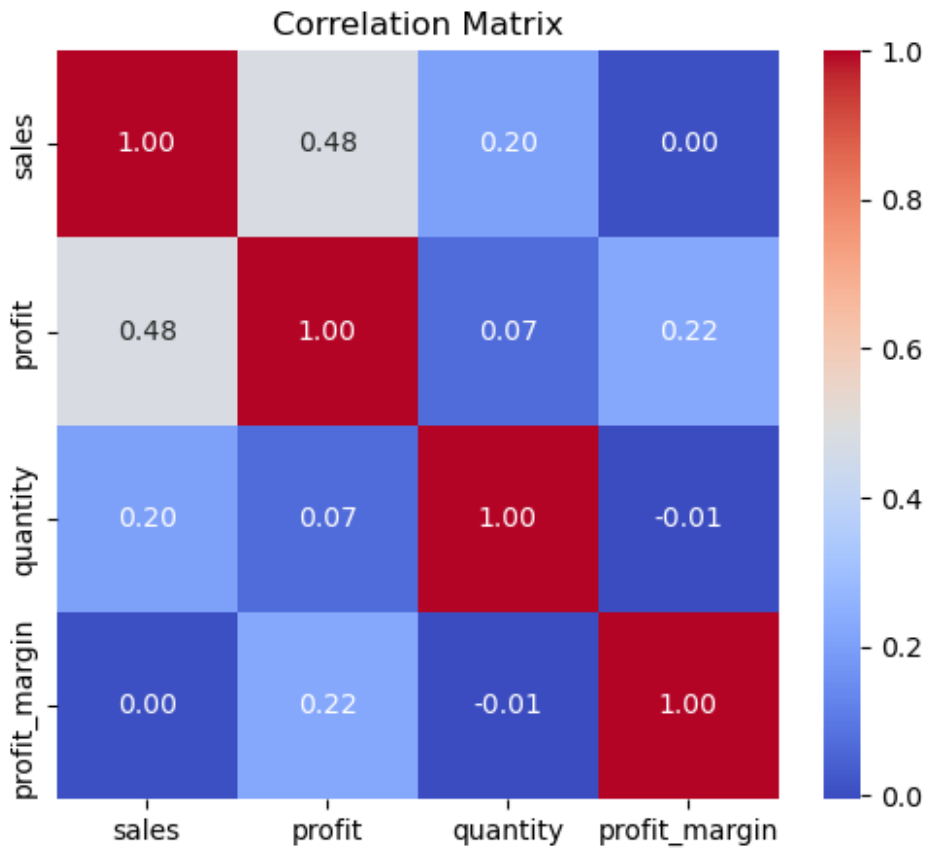
```
# Set plot size and style
plt.figure(figsize=(14, 7))
sns.lineplot(data=seasonal_subcat, x='month', y='sales', hue='sub-
category', marker='o')

# Formatting
plt.title("Seasonal Sales Trends by Sub-Category", fontsize=14)
plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.xticks(range(1, 13))
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', ncol=2,
title="Sub-Category")
plt.tight_layout()
plt.show()
```








```
#CORRELATION ANALYSIS
# Check correlations
numeric_cols = ['sales', 'profit', 'quantity', 'profit_margin']
corr_matrix = df[numeric_cols].corr()

# Plot heatmap
plt.figure(figsize=(6,5))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```




```
#TOP/BOTTOM ANALYSIS
#Most & Least Profitable Products
product_profit = df.groupby('product_name')[['sales',
'profit']].sum().sort_values(by='profit')

print("Top 10 Loss-Making Products:")
print(product_profit.head(10))


print("Top 10 Profitable Products:")
print(product_profit.tail(10))
```

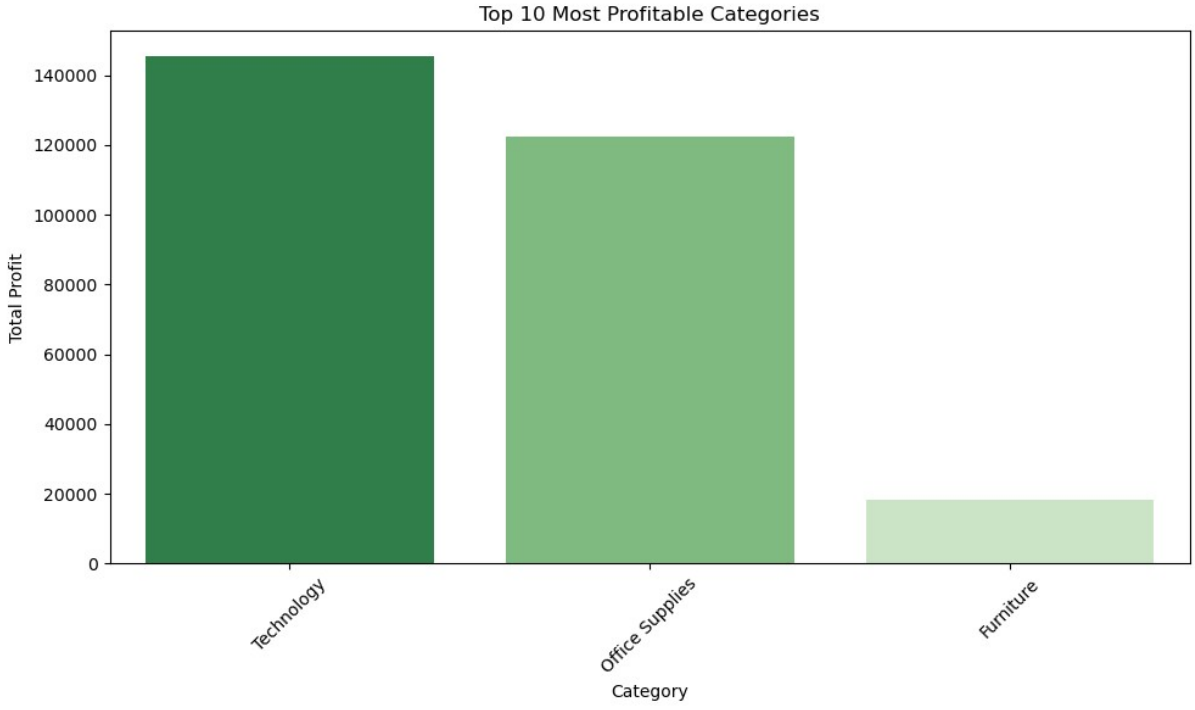
```
Top 10 Loss-Making Products:
                                                      sales
profit
product_name

Cubify CubeX 3D Printer Double Head Print           11099.963 -
8879.9704
Lexmark MX611dhe Monochrome Laser Printer           16829.901 -
4589.9730
Cubify CubeX 3D Printer Triple Head Print             7999.980 -
3839.9904
Chromcraft Bull-Nose Wood Oval Conference Table...   9917.640 -
2876.1156
Bush Advantage Collection Racetrack Conference ...   9544.725 -
1934.3976
```

```
GBC DocuBind P400 Electric Binding System           17965.068 -
1878.1662
Cisco TelePresence System EX90 Videoconferencin...  22638.480 -
1811.0784
Martin Yale Chadless Opener Electric Letter Opener  16656.200 -
1299.1836
Balt Solid Wood Round Tables                         6518.754 -
1201.0581
BoxOffice By Design Rectangular and Half-Moon M...   1706.250 -
1148.4375
Top 10 Profitable Products:
                                                        sales
profit
product_name

Zebra ZM400 Thermal Label Printer                    6965.700
3343.5360
Ibico EPK-21 Electric Binding System                15875.916
3345.2823
Plantronics Savi W720 Multi-Device Wireless Hea...   9367.290
3696.2820
3D Systems Cube Printer, 2nd Generation, Magenta    14299.890
3717.9714
Ativa V4110MDD Micro-Cut Shredder                    7699.890
3772.9461
HP Designjet T520 Inkjet Large Format Printer -...  18374.895
4094.9766
Canon PC1060 Personal Laser Copier                  11619.834
4570.9347
Hewlett Packard LaserJet 3310 Copier                18839.686
6983.8836
Fellowes PB500 Electric Punch Plastic Comb Bind...  27453.384
7753.0390
Canon imageCLASS 2200 Advanced Copier               61599.824
25199.9280
```

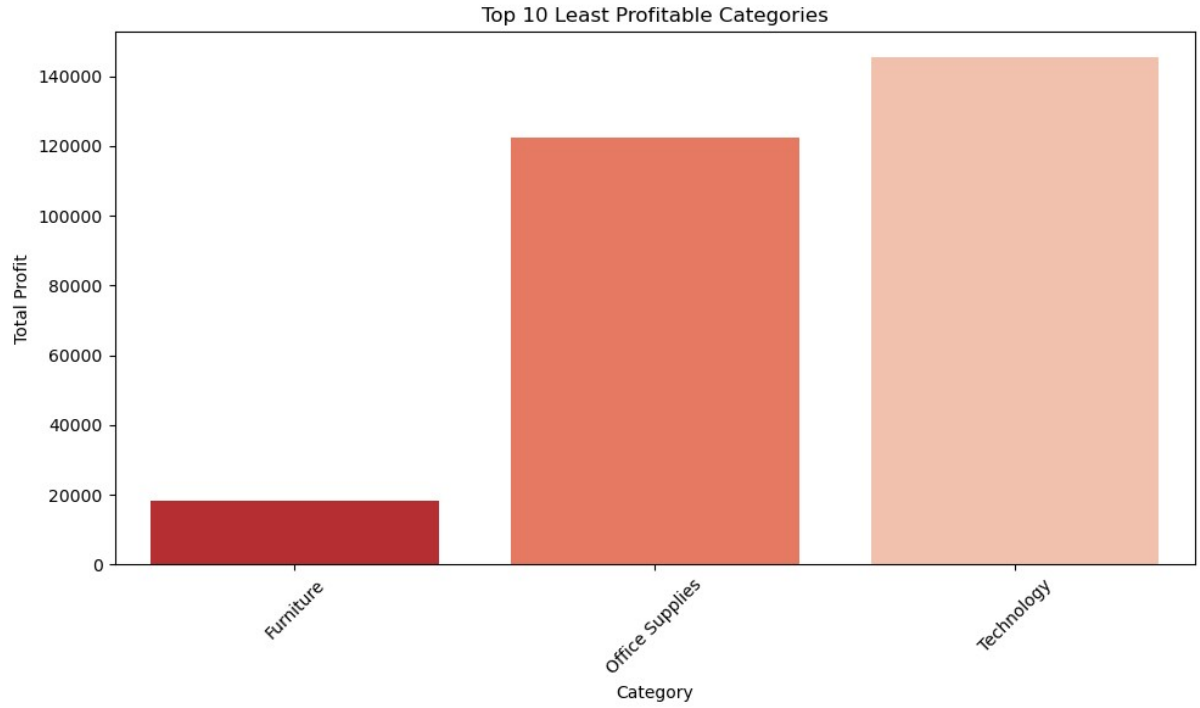
```python
#top 10 most profitable categories
# Group by category and sum the profit
top_categories = df.groupby('category')
['profit'].sum().sort_values(ascending=False).head(10).reset_index()


print("Top 10 Most Profitable Categories:")
print(top_categories)

# Visualization
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
sns.barplot(data=top_categories, x='category', y='profit',
palette='Greens_r')
plt.title("Top 10 Most Profitable Categories")
plt.xlabel("Category")
```

```
plt.ylabel("Total Profit")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

#top 10 least profitbale categories
## Group by category and sort by lowest profit
least_categories = df.groupby('category')
['profit'].sum().sort_values().head(10).reset_index()

# Display the result
print("Top 10 Least Profitable Categories:")
print(least_categories)

# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(data=least_categories, x='category', y='profit',
palette='Reds_r')
plt.title("Top 10 Least Profitable Categories")
plt.xlabel("Category")
plt.ylabel("Total Profit")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

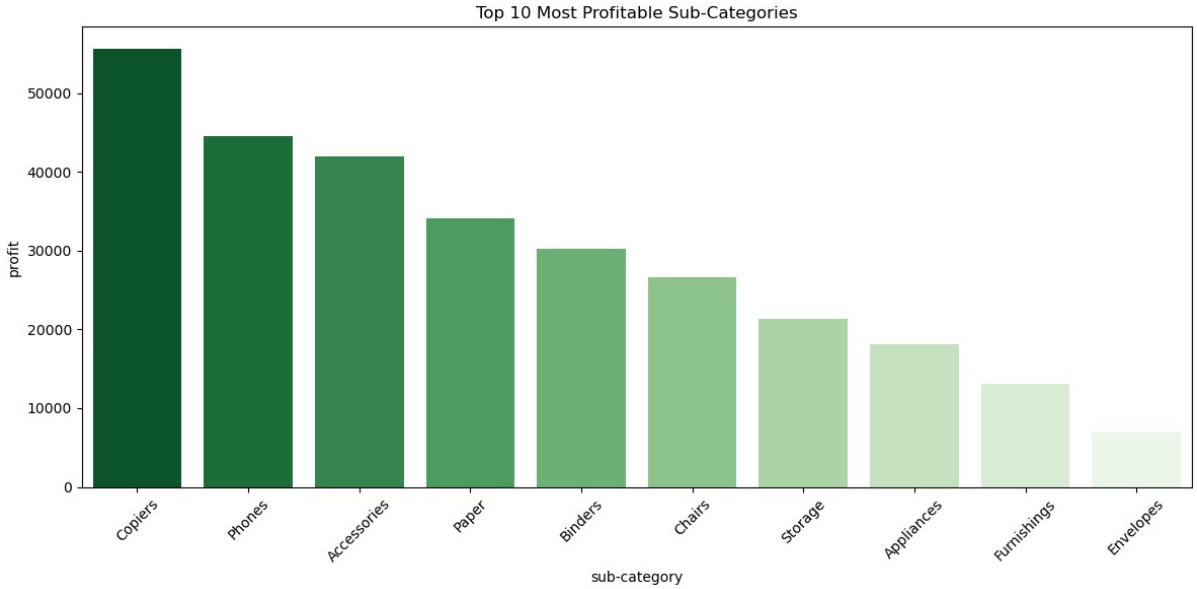
```
Top 10 Most Profitable Categories:
          category       profit
0       Technology  145454.9481
1  Office Supplies  122490.8008
2        Furniture   18451.2728
```

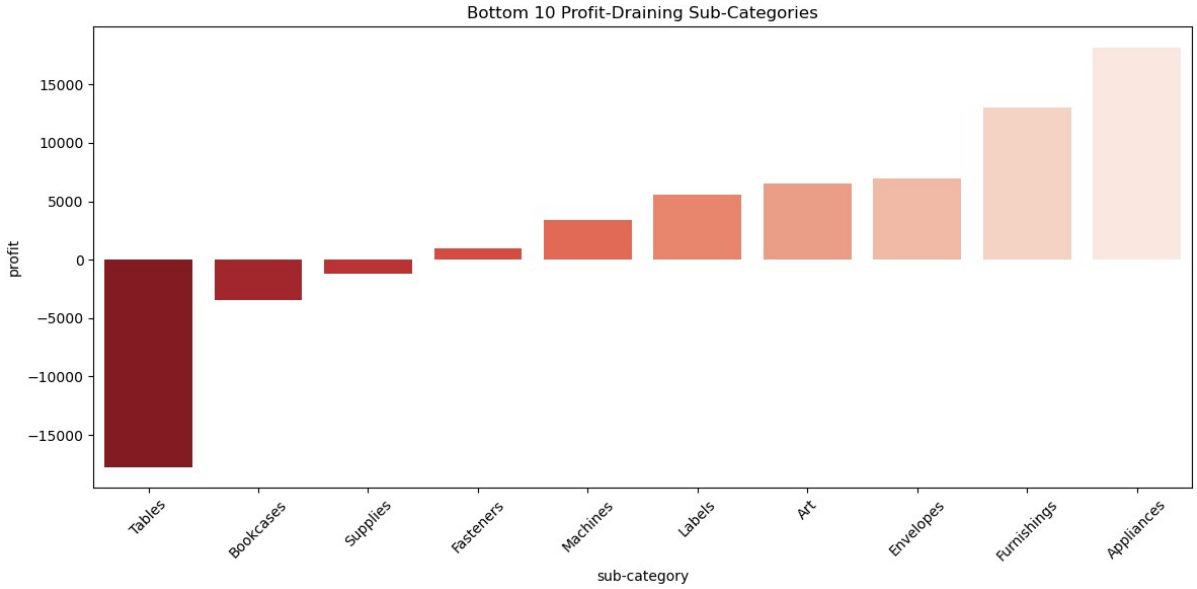


```
Top 10 Least Profitable Categories:
          category       profit
0        Furniture   18451.2728
1  Office Supplies  122490.8008
2       Technology  145454.9481
```




```
# --- . TOP 10 MOST PROFITABLE SUB-CATEGORIES ---
subcat_profit = df.groupby('sub-category')
['profit'].sum().sort_values(ascending=False).head(10).reset_index()

plt.figure(figsize=(12, 6))
sns.barplot(data=subcat_profit, x='sub-category', y='profit',
palette='Greens_r')
plt.title("Top 10 Most Profitable Sub-Categories")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# --- . BOTTOM 10 (LOSS-MAKING) SUB-CATEGORIES ---
subcat_loss = df.groupby('sub-category')
['profit'].sum().sort_values().head(10).reset_index()

plt.figure(figsize=(12, 6))
sns.barplot(data=subcat_loss, x='sub-category', y='profit',
palette='Reds_r')
plt.title("Bottom 10 Profit-Draining Sub-Categories")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



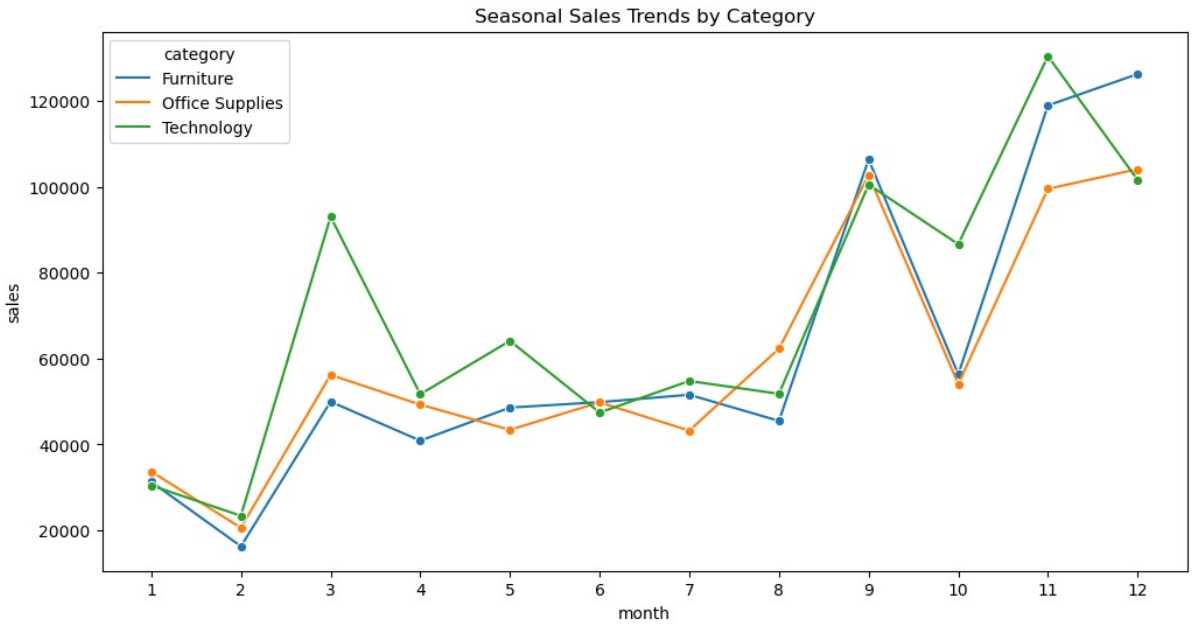



```
#Inventory Turnover Analysis
#Simulating inventory (if no stock/inventory_days column exists):

# Assume average inventory = average of quantity sold per month for
simulation
monthly_quantity = df.groupby(['product_name', 'month'])
['quantity'].sum().groupby('product_name').mean()
avg_inventory = monthly_quantity.to_dict()

# Map to each row (simulated avg inventory)
df['avg_inventory'] = df['product_name'].map(avg_inventory)

# Calculate Inventory Turnover
df['inventory_turnover'] = df['sales'] / df['avg_inventory']


#Correlate turnover vs profit margin
```

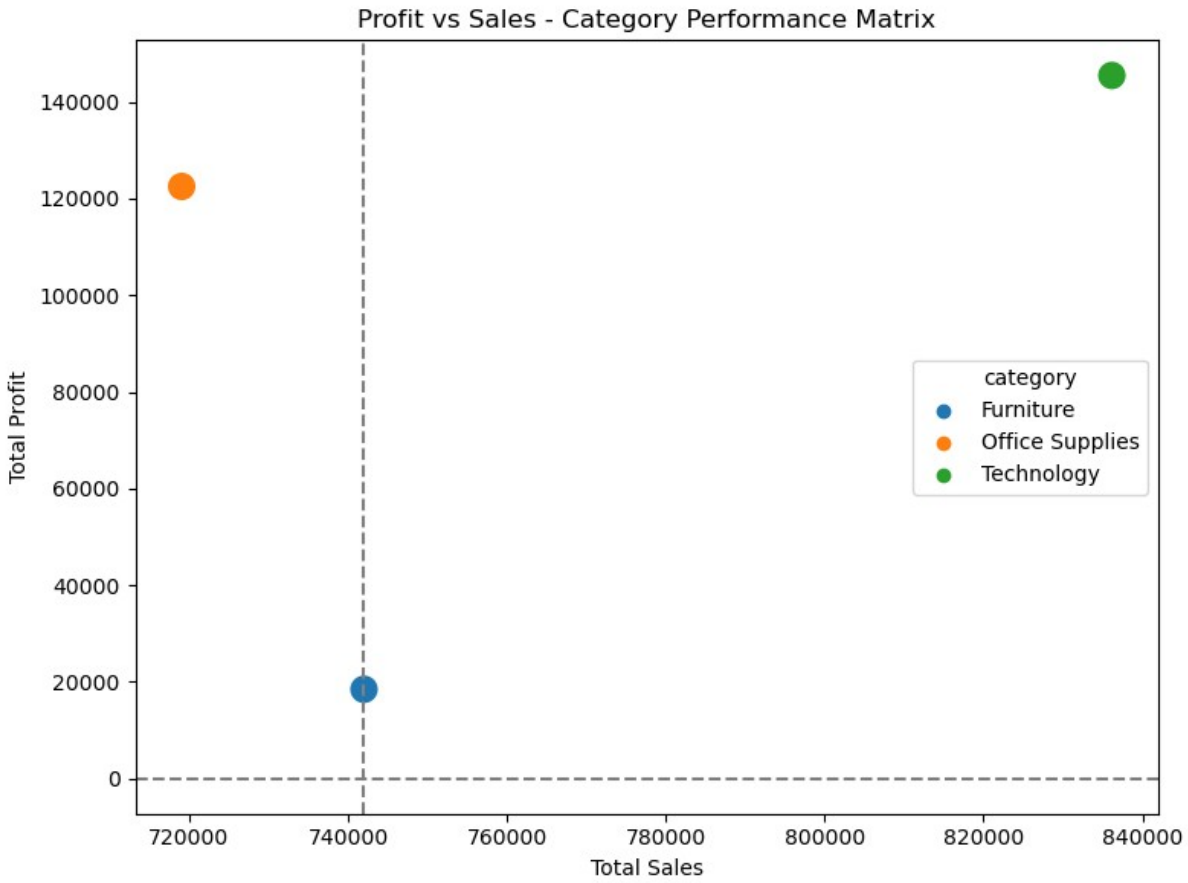
```
# Remove NaNs or Infs
df_filtered = df.dropna(subset=['inventory_turnover',
'profit_margin'])
df_filtered = df_filtered[df_filtered['inventory_turnover'] <
df_filtered['inventory_turnover'].quantile(0.95)]

# Correlation
corr =
df_filtered['inventory_turnover'].corr(df_filtered['profit_margin'])
print(f"Correlation between inventory turnover and profit margin:
{corr:.2f}")


#graph
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_filtered, x='inventory_turnover',
y='profit_margin', hue='category', alpha=0.7)
plt.title("Inventory Turnover vs Profit Margin")
plt.xlabel("Inventory Turnover")
plt.ylabel("Profit Margin")
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()


Correlation between inventory turnover and profit margin: -0.02
```


```
#Seasonal Product Behavior
# Monthly category performance
seasonal = df.groupby(['category', 'month']).agg({
    'sales': 'sum',
```

```
    'profit': 'sum'
}).reset_index()

# Visualize seasonal trends per category
plt.figure(figsize=(12, 6))
sns.lineplot(data=seasonal, x='month', y='sales', hue='category',
marker='o')
plt.title("Seasonal Sales Trends by Category")
plt.xticks(range(1,13))
plt.show()
```




```
#Category Performance Matrix (Quadrant Analysis)
# Aggregate at category level
cat_matrix = df.groupby('category').agg({
    'sales': 'sum',
    'profit': 'sum'
}).reset_index()

# Scatterplot matrix
plt.figure(figsize=(8,6))
sns.scatterplot(data=cat_matrix, x='sales', y='profit', s=200,
hue='category')
plt.axhline(y=0, color='gray', linestyle='--')
plt.axvline(x=cat_matrix['sales'].median(), color='gray',
linestyle='--')
plt.title("Profit vs Sales - Category Performance Matrix")
plt.xlabel("Total Sales")
plt.ylabel("Total Profit")
plt.tight_layout()
plt.show()
```




```
#Product-Level Deep Dive
# Top 10 most profitable products
top_products = df.groupby('product_name').agg({
    'sales': 'sum',
    'profit': 'sum',
    'quantity': 'sum'
}).sort_values(by='profit', ascending=False).head(10)

# Bottom 10 (loss-makers)
loss_products = df.groupby('product_name').agg({
    'sales': 'sum',
    'profit': 'sum',
    'quantity': 'sum'
}).sort_values(by='profit', ascending=True).head(10)

print("Top 10 Products by Profit:")
print(top_products)

print("Bottom 10 Products by Profit:")
print(loss_products)
```

```
Top 10 Products by Profit:
                                                      sales
profit  \
product_name
```

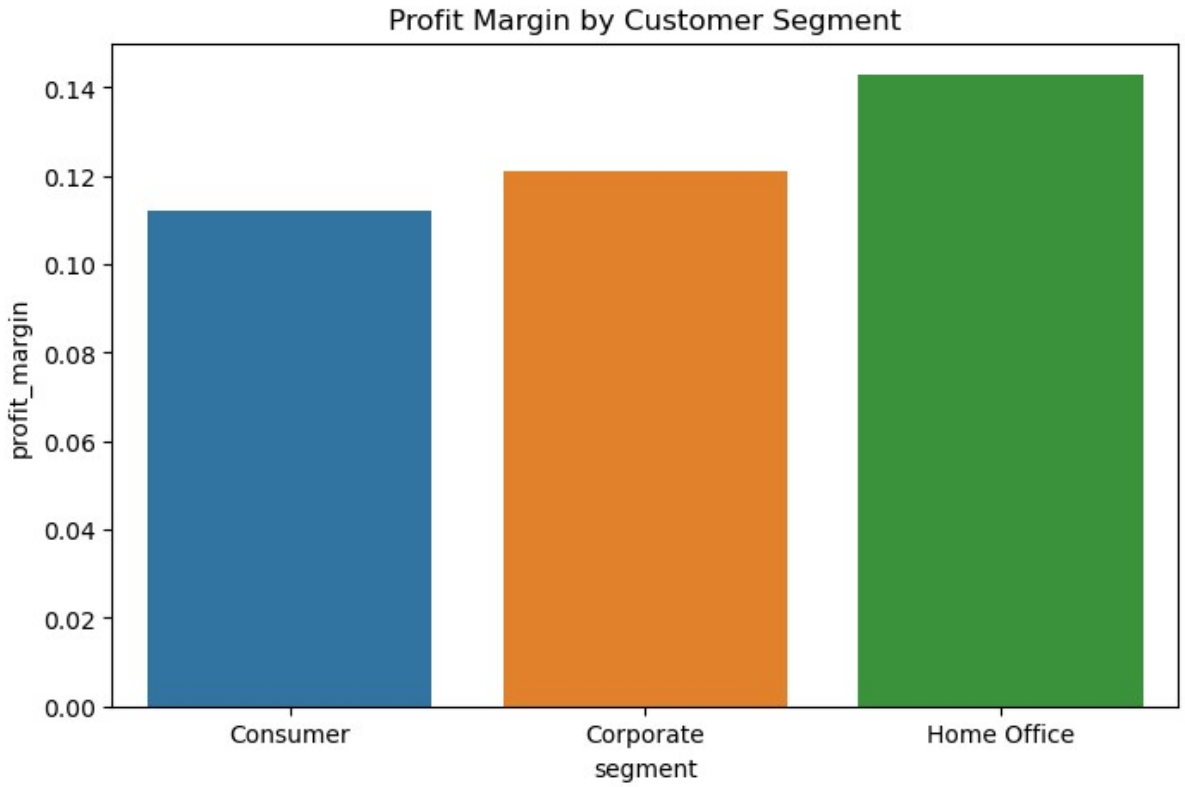
```
Canon imageCLASS 2200 Advanced Copier                61599.824
25199.9280
Fellowes PB500 Electric Punch Plastic Comb Bind...   27453.384
7753.0390
Hewlett Packard LaserJet 3310 Copier                 18839.686
6983.8836
Canon PC1060 Personal Laser Copier                   11619.834
4570.9347
HP Designjet T520 Inkjet Large Format Printer -...   18374.895
4094.9766
Ativa V4110MDD Micro-Cut Shredder                     7699.890
3772.9461
3D Systems Cube Printer, 2nd Generation, Magenta     14299.890
3717.9714
Plantronics Savi W720 Multi-Device Wireless Hea...    9367.290
3696.2820
Ibico EPK-21 Electric Binding System                 15875.916
3345.2823
Zebra ZM400 Thermal Label Printer                     6965.700
3343.5360

                                                     quantity
product_name
Canon imageCLASS 2200 Advanced Copier                      20
Fellowes PB500 Electric Punch Plastic Comb Bind...         31
Hewlett Packard LaserJet 3310 Copier                       38
Canon PC1060 Personal Laser Copier                         19
HP Designjet T520 Inkjet Large Format Printer -...         12
Ativa V4110MDD Micro-Cut Shredder                          11
3D Systems Cube Printer, 2nd Generation, Magenta           11
Plantronics Savi W720 Multi-Device Wireless Hea...         24
Ibico EPK-21 Electric Binding System                       13
Zebra ZM400 Thermal Label Printer                           6
Bottom 10 Products by Profit:
                                                       sales
profit  \
product_name

Cubify CubeX 3D Printer Double Head Print            11099.963 -
8879.9704
Lexmark MX611dhe Monochrome Laser Printer            16829.901 -
4589.9730
Cubify CubeX 3D Printer Triple Head Print             7999.980 -
3839.9904
Chromcraft Bull-Nose Wood Oval Conference Table...    9917.640 -
2876.1156
Bush Advantage Collection Racetrack Conference ...    9544.725 -
1934.3976
GBC DocuBind P400 Electric Binding System            17965.068 -
1878.1662
Cisco TelePresence System EX90 Videoconferencin...   22638.480 -
1811.0784
Martin Yale Chadless Opener Electric Letter Opener   16656.200 -
```

```
1299.1836
Balt Solid Wood Round Tables                          6518.754 -
1201.0581
BoxOffice By Design Rectangular and Half-Moon M...   1706.250 -
1148.4375
```

```
                                                    quantity
product_name
Cubify CubeX 3D Printer Double Head Print                  9
Lexmark MX611dhe Monochrome Laser Printer                 18
Cubify CubeX 3D Printer Triple Head Print                  4
Chromcraft Bull-Nose Wood Oval Conference Table...        27
Bush Advantage Collection Racetrack Conference ...        33
GBC DocuBind P400 Electric Binding System                 27
Cisco TelePresence System EX90 Videoconferencin...         6
Martin Yale Chadless Opener Electric Letter Opener        22
Balt Solid Wood Round Tables                              19
BoxOffice By Design Rectangular and Half-Moon M...        15
```

```
#Customer Segment Performance
if 'segment' in df.columns:
    segment_perf = df.groupby('segment').agg({
        'sales': 'sum',
        'profit': 'sum',
        'profit_margin': 'mean'
    }).reset_index()
    print(segment_perf)

    # Visual
    plt.figure(figsize=(8, 5))
    sns.barplot(data=segment_perf, x='segment', y='profit_margin')
    plt.title("Profit Margin by Customer Segment")
    plt.show()
```

```
       segment         sales       profit  profit_margin
0     Consumer  1.161401e+06  134119.2092       0.112050
1    Corporate  7.061464e+05   91979.1340       0.121203
2  Home Office  4.296531e+05   60298.6785       0.142870
```




```
# --- 8. INVENTORY TURNOVER SIMULATION ---
monthly_quantity = df.groupby(['product_name', 'month'])
['quantity'].sum().groupby('product_name').mean()
df['avg_inventory'] = df['product_name'].map(monthly_quantity)
df['inventory_turnover'] = df['sales'] / df['avg_inventory']
monthly_quantity

product_name
"While you Were Out" Message Book, One Form per Page
2.666667
#10 Gummed Flap White Envelopes, 100/Box
3.666667
#10 Self-Seal White Envelopes
3.333333
#10 White Business Envelopes,4 1/8 x 9 1/2
5.333333
#10- 4 1/8" x 9 1/2" Recycled Envelopes
6.166667

...
iKross Bluetooth Portable Keyboard + Cell Phone Stand Holder + Brush
for Apple iPhone 5S 5C 5, 4S 4    4.800000
iOttie HLCRIO102 Car Mount
3.000000
iOttie XL Car Mount
7.000000
invisibleSHIELD by ZAGG Smudge-Free Screen Protector
5.800000
netTALK DUO VoIP Telephone Service
```

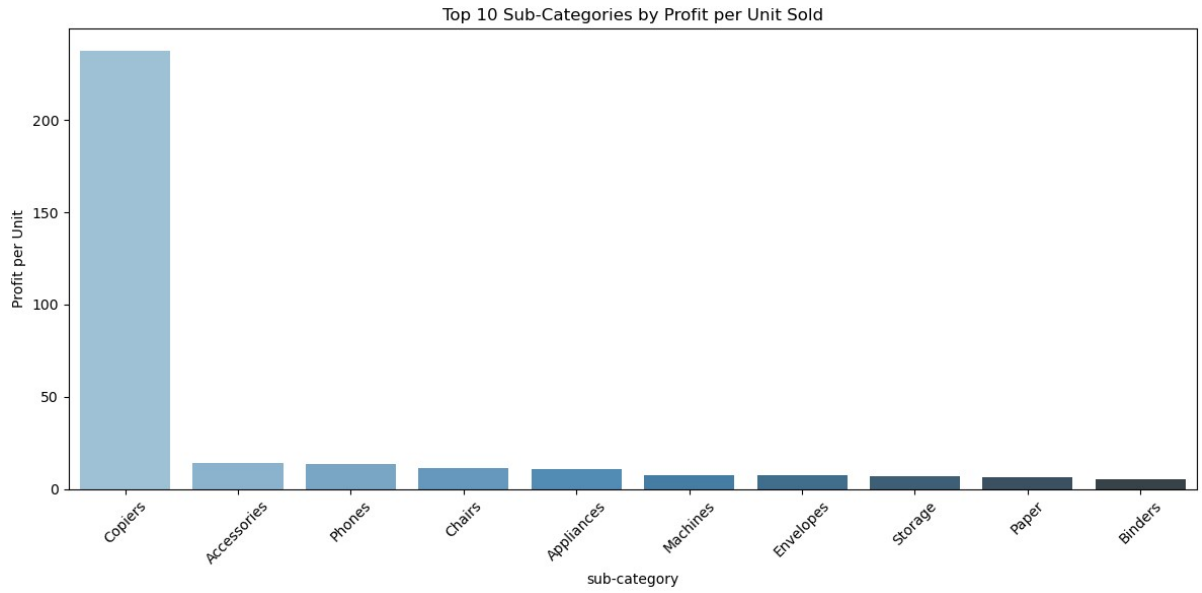
```
4.333333
Name: quantity, Length: 1841, dtype: float64
```

```
df_filtered = df.dropna(subset=['inventory_turnover',
'profit_margin'])
df_filtered = df_filtered[df_filtered['inventory_turnover'] <
df_filtered['inventory_turnover'].quantile(0.95)]

# Correlation between inventory turnover and profit margin
correlation =
df_filtered['inventory_turnover'].corr(df_filtered['profit_margin'])
print(f"Correlation between inventory turnover and profit margin:
{correlation:.2f}")
```

```
Correlation between inventory turnover and profit margin: -0.02
```

```
#Profit per Unit Sold (Efficiency Insight)
profit_per_unit = df.groupby('sub-category').agg({
    'profit': 'sum',
    'quantity': 'sum'
})
profit_per_unit['profit_per_unit'] = profit_per_unit['profit'] /
profit_per_unit['quantity']
profit_per_unit = profit_per_unit.sort_values(by='profit_per_unit',
ascending=False).reset_index()

# top 10 most efficient
print("Top 10 Sub-Categories by Profit per Unit Sold:")
print(profit_per_unit.head(10))

#graph
plt.figure(figsize=(12, 6))
sns.barplot(data=profit_per_unit.head(10), x='sub-category',
y='profit_per_unit', palette='Blues_d')
plt.title("Top 10 Sub-Categories by Profit per Unit Sold")
plt.ylabel("Profit per Unit")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
Top 10 Sub-Categories by Profit per Unit Sold:
  sub-category      profit  quantity  profit_per_unit
0      Copiers  55617.8249       234       237.683012
1  Accessories  41936.6357      2976        14.091611
2       Phones  44515.7306      3289        13.534731
3       Chairs  26590.1663      2356        11.286149
4   Appliances  18138.0054      1729        10.490460
5     Machines   3384.7569       440         7.692629
6    Envelopes   6964.1767       906         7.686729
7      Storage  21278.8264      3158         6.738070
8        Paper  34053.5693      5178         6.576587
9      Binders  30221.7633      5974         5.058882
```

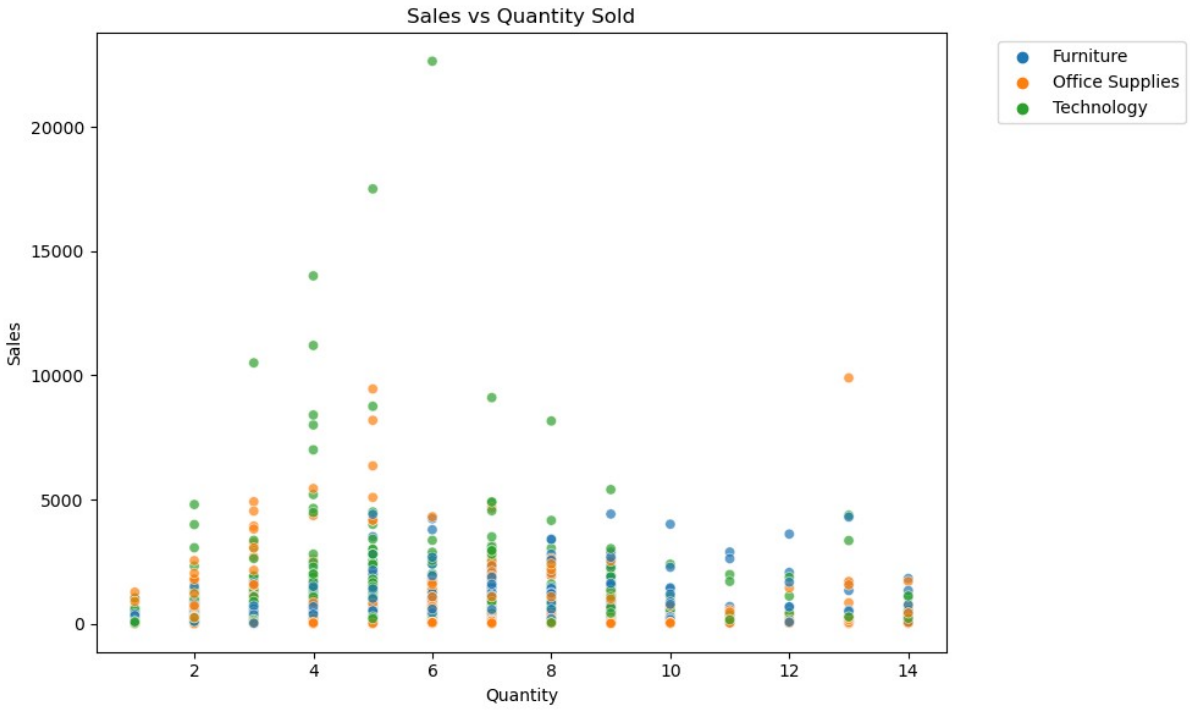



```
#Best & Worst Performing Products
product_perf = df.groupby('product_name').agg({
    'sales': 'sum',
    'profit': 'sum'
}).sort_values(by='profit')

print("Top 5 Worst Performing Products:")
print(product_perf.head(5))

print("\nTop 5 Most Profitable Products:")
print(product_perf.tail(5))
```

```
Top 5 Worst Performing Products:
                                                    sales
profit
product_name

Cubify CubeX 3D Printer Double Head Print          11099.963 -
8879.9704
Lexmark MX611dhe Monochrome Laser Printer          16829.901 -
4589.9730
Cubify CubeX 3D Printer Triple Head Print            7999.980 -
3839.9904
Chromcraft Bull-Nose Wood Oval Conference Table...   9917.640 -
2876.1156
Bush Advantage Collection Racetrack Conference ...   9544.725 -
1934.3976

Top 5 Most Profitable Products:
                                                    sales
profit
product_name

HP Designjet T520 Inkjet Large Format Printer -...  18374.895
4094.9766
Canon PC1060 Personal Laser Copier                  11619.834
```

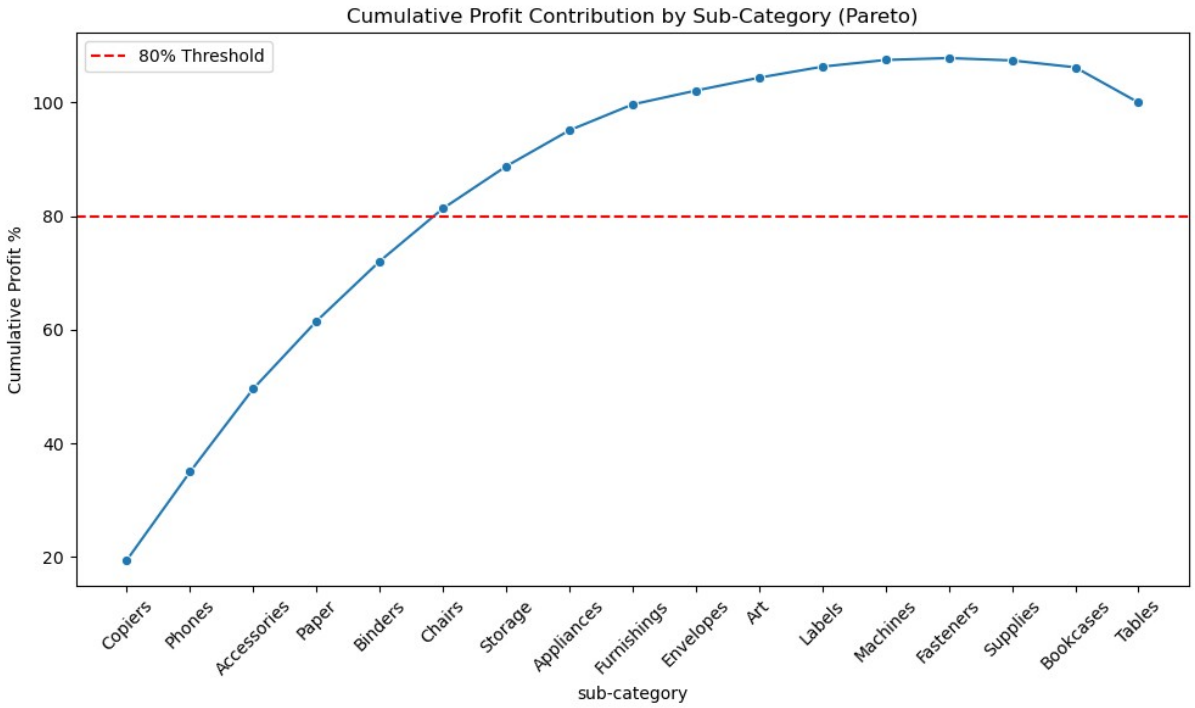
```
4570.9347
Hewlett Packard LaserJet 3310 Copier                 18839.686
6983.8836
Fellowes PB500 Electric Punch Plastic Comb Bind...   27453.384
7753.0390
Canon imageCLASS 2200 Advanced Copier                61599.824
25199.9280
```

```
#Sales vs Quantity Sold (Outliers Detection)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='quantity', y='sales', hue='category',
alpha=0.7)
plt.title("Sales vs Quantity Sold")
plt.xlabel("Quantity")
plt.ylabel("Sales")
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```




```
#Sub-Category Profitability Heatmap by Region
heatmap_data = df.pivot_table(index='sub-category',
columns='region', values='profit', aggfunc='sum')

plt.figure(figsize=(12, 8))
sns.heatmap(heatmap_data, cmap='YlGnBu', annot=True, fmt=".0f")
plt.title("Profit by Sub-Category and Region")
plt.tight_layout()
plt.show()
```

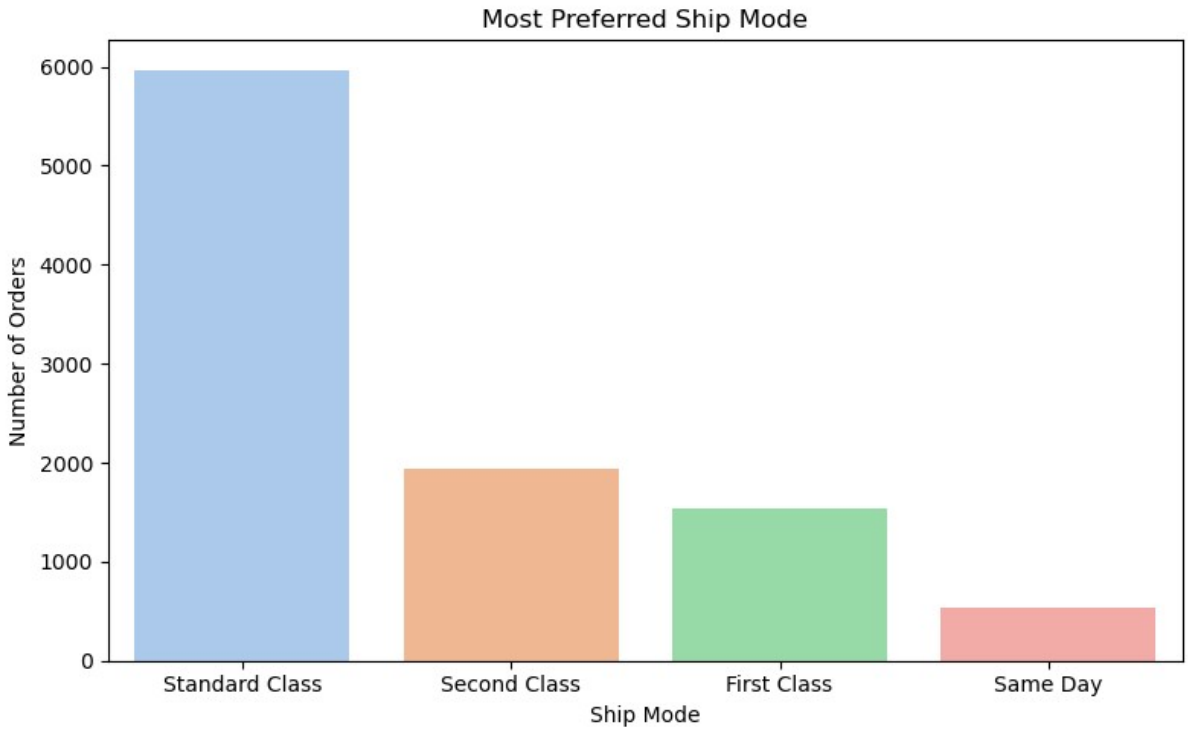

```
#Contribution Analysis (Pareto 80/20 Rule)
contribution = df.groupby('sub-category')
['profit'].sum().sort_values(ascending=False).reset_index()
contribution['cumulative_profit_pct'] =
contribution['profit'].cumsum() / contribution['profit'].sum() * 100

# Plot cumulative contribution
plt.figure(figsize=(10, 6))
sns.lineplot(data=contribution, x='sub-category',
y='cumulative_profit_pct', marker='o')
plt.axhline(80, color='red', linestyle='--', label='80% Threshold')
plt.title("Cumulative Profit Contribution by Sub-Category (Pareto)")
plt.xticks(rotation=45)
plt.ylabel("Cumulative Profit %")
plt.legend()
plt.tight_layout()
plt.show()
```




```
# how many times each ship mode was used
ship_mode_counts = df['ship_mode'].value_counts().reset_index()
ship_mode_counts.columns = ['ship_mode', 'count']

# most preferred ship mode
print("Ship Mode Usage Count:")
print(ship_mode_counts)
```

```
Ship Mode Usage Count:
        ship_mode  count
0  Standard Class   5968
1    Second Class   1945
2     First Class   1538
3        Same Day    543
```

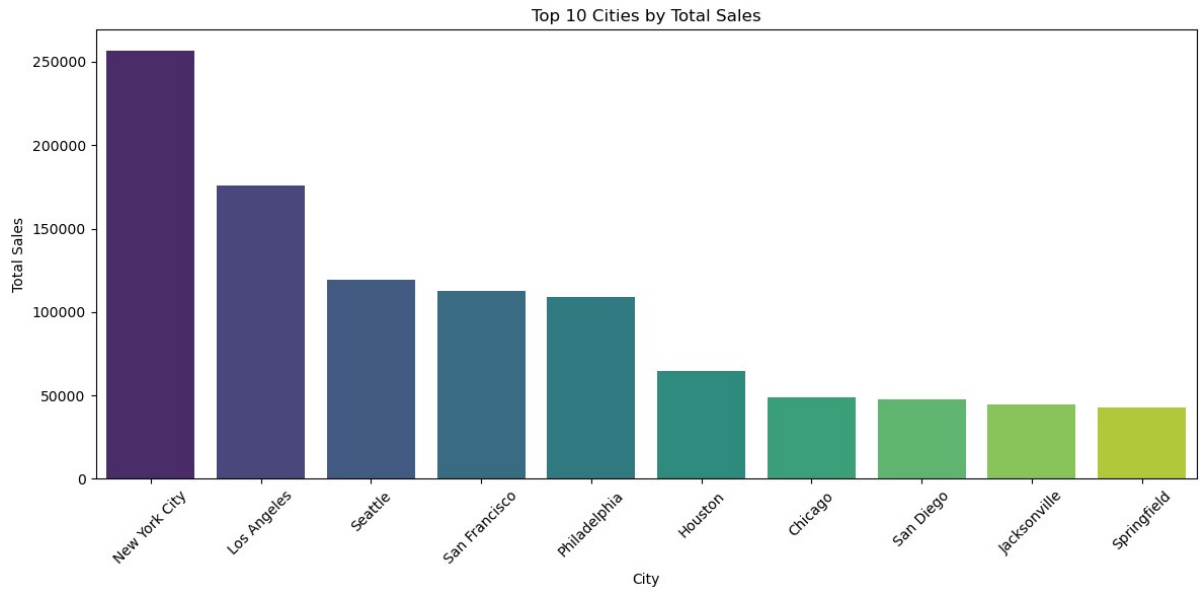
```
plt.figure(figsize=(8, 5))
sns.barplot(data=ship_mode_counts, x='ship_mode', y='count',
palette='pastel')

plt.title("Most Preferred Ship Mode")
plt.xlabel("Ship Mode")
plt.ylabel("Number of Orders")
plt.tight_layout()
plt.show()
```




```
# Group sales by city
city_sales = df.groupby('city')
['sales'].sum().sort_values(ascending=False).reset_index()

# Graph of top cities
print("Top 10 Cities by Total Sales:")
print(city_sales.head(10))



# top 10 cities
top_10_cities = city_sales.head(10)

plt.figure(figsize=(12, 6))
sns.barplot(data=top_10_cities, x='city', y='sales',
palette='viridis')
plt.title("Top 10 Cities by Total Sales")
plt.xlabel("City")
plt.ylabel("Total Sales")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
Top 10 Cities by Total Sales:
            city        sales
0  New York City  256368.1610
1    Los Angeles  175851.3410
2        Seattle  119540.7420
3  San Francisco  112669.0920
4   Philadelphia  109077.0130
5        Houston   64504.7604
```

```
6         Chicago   48539.5410
7       San Diego   47521.0290
8    Jacksonville   44713.1830
9     Springfield   43054.3420
```




```
order_products = df.groupby('order_id')['product_name'].apply(list)


print(order_products.head())

from itertools import combinations
from collections import Counter


pair_counts = Counter()


for products in order_products:
    if len(products) > 1:

        pairs = combinations(sorted(products), 2)
        pair_counts.update(pairs)


most_common_pairs = pd.DataFrame(pair_counts.most_common(10),
columns=['product_pair', 'count'])

print("Top 10 Product Pairs Sold Together:")
print(most_common_pairs)
```

```
order_id
CA-2011-100006                                    [AT&T EL51110 DECT]
CA-2011-100090    [Hon 2111 Invitation Series Corner Table, Wils...
CA-2011-100293                                          [Xerox 1887]
CA-2011-100328    [Pressboard Covers with Storage Hooks, 9 1/2" ...
```

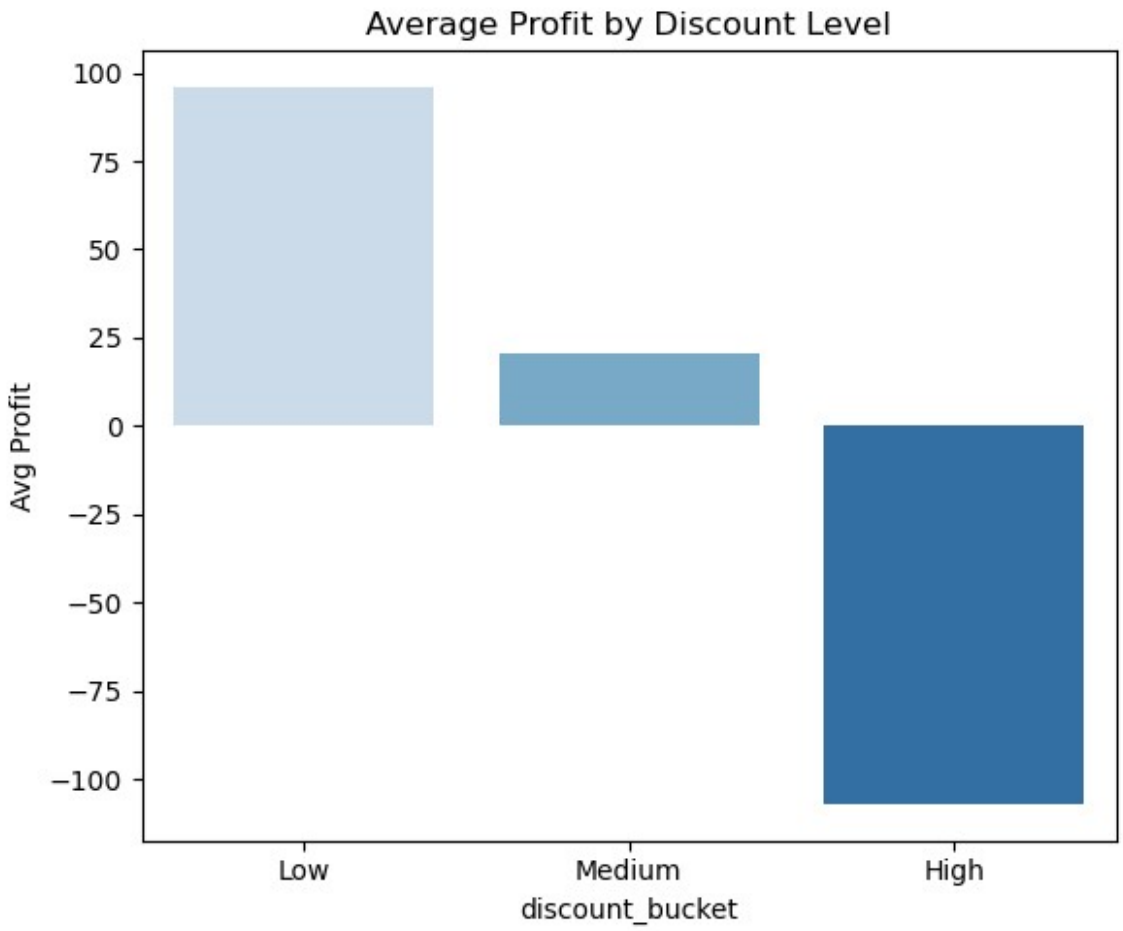
```
CA-2011-100363    [Binder Clips by OIC, Things To Do Today Spira...
Name: product_name, dtype: object
Top 10 Product Pairs Sold Together:
                                          product_pair  count
0                                (Staples, Staples)      5
1                              (Newell 34, Staples)      4
2       (Satellite Sectional Post Binders, Staples)      3
3  (#10 White Business Envelopes,4 1/8 x 9 1/2, S...      3
4             (KI Adjustable-Height Table, Staples)      3
5  (Fellowes Stor/Drawer Steel Plus Storage Drawe...      3
6               (Hon Olson Stacker Chairs, Staples)      3
7  (Hoover Shoulder Vac Commercial Portable Vacuu...      3
8                             (Staples, Xerox 1916)      3
9     (Adjustable Depth Letter/Legal Cart, Staples)      3

# Basic KPIs
total_sales = df['sales'].sum()
total_profit = df['profit'].sum()
total_quantity = df['quantity'].sum()

print(f"Total Sales: ₹{total_sales:,.2f}")
print(f"Total Profit: ₹{total_profit:,.2f}")
print(f"Total Quantity Sold: {total_quantity:,}")

Total Sales: ₹2,297,200.86
Total Profit: ₹286,397.02
Total Quantity Sold: 37,873

profit_margin_cat = df.groupby('category')['profit'].sum() /
df.groupby('category')['sales'].sum()
print("Profit Margin by Category:")
print(profit_margin_cat.round(2))

Profit Margin by Category:
category
Furniture          0.02
Office Supplies    0.17
Technology         0.17
dtype: float64

best_month = df.groupby(df['order_date'].dt.month)
['sales'].sum().reset_index()
best_month.columns = ['month', 'total_sales']
print(best_month.sort_values(by='total_sales',
ascending=False).head(1))

    month  total_sales
10     11   349120.074

top_customers = df.groupby('customer_name')
['sales'].sum().sort_values(ascending=False).head(10).reset_index()

print("Top 10 Customers by Total Sales:")
print(top_customers)
```

```
Top 10 Customers by Total Sales:
        customer_name      sales
0         Sean Miller  25043.050
1        Tamara Chand  19052.218
2        Raymond Buch  15117.339
3        Tom Ashbrook  14595.620
4       Adrian Barton  14473.571
5        Ken Lonsdale  14175.229
6        Sanjit Chand  14142.334
7        Hunter Lopez  12873.298
8        Sanjit Engle  12209.438
9  Christopher Conant  12129.072
```

```
df['discount_bucket'] = pd.cut(df['discount'], bins=[0, 0.1, 0.3,
1], labels=['Low', 'Medium', 'High'])
```

```
discount_analysis = df.groupby('discount_bucket')
['profit'].mean().reset_index()
print(discount_analysis)

plt.figure(figsize=(6,5))
sns.barplot(data=discount_analysis, x='discount_bucket', y='profit',
palette='Blues')
plt.title("Average Profit by Discount Level")
plt.ylabel("Avg Profit")
plt.tight_layout()
plt.show()
```

```
  discount_bucket      profit
0             Low   96.055074
1          Medium   20.677597
2            High -107.209930
```


Average Profit by Discount Level
100
75
50
25
0
−25
−50
−75
−100
Avg Profit
Low
Medium
High
discount_bucket