Library Imports

```
# @title #Library Imports

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import plotly.express as px
import requests

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)
```

Import Data From DataSource

```
# @title Import Data From DataSource

# URL
dataset_url = "https://community.tableau.com/sfc/servlet.shepherd/document/download/0694T000001GnpUQAS?operationContext=S1"

# Send a GET request to download the file
response = requests.get(dataset_url, allow_redirects=True)

# Save in Colab
file_path = "sample-superstore-sales.xls"
with open(file_path, "wb") as file:
    file.write(response.content)
```

```
df = pd.read_excel('/content/sample-superstore-sales.xls')
```

Descriptive Statical Analysis

```
df.head()
```

	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	Produ Na
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bu Somer Collect Bookca
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Delu Fab Upholster Stack Cha Round Ba
2	3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	Se Adhes Addre Labels Typewrit

```
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.describe(include='all')
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State	Postal Code	Region
count	9994.000000	9994	9994	9994	9994	9994	9994	9994	9994	9994	9994	9994.000000	9994
unique	NaN	5009	NaN	NaN	4	793	793	3	1	531	49	NaN	4
top	NaN	CA-2017-100111	NaN	NaN	Standard Class	WB-21850	William Brown	Consumer	United States	New York City	California	NaN	West
freq	NaN	14	NaN	NaN	5968	37	37	5191	9994	915	2001	NaN	3203
mean	4997.500000	NaN	2016-04-30 00:07:12.259355648	2016-05-03 23:06:58.571142912	NaN	NaN	NaN	NaN	NaN	NaN	NaN	55190.379428	NaN
min	1.000000	NaN	2014-01-03 00:00:00	2014-01-07 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1040.000000	NaN
25%	2499.250000	NaN	2015-05-23 00:00:00	2015-05-27 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	23223.000000	NaN
50%	4997.500000	NaN	2016-06-26 00:00:00	2016-06-29 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	56430.500000	NaN
75%	7495.750000	NaN	2017-05-14 00:00:00	2017-05-18 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	90008.000000	NaN
max	9994.000000	NaN	2017-12-30 00:00:00	2018-01-05 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	99301.000000	NaN
std	2885.163629	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	32063.693350	NaN

```python
df.isnull().sum()
```

	0
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

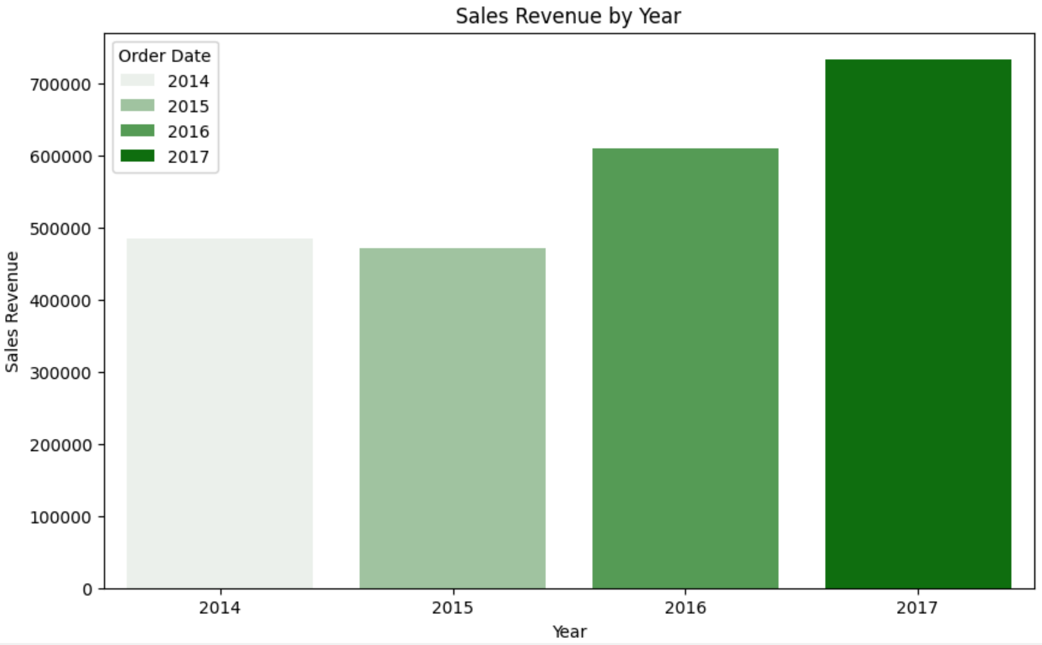
dtype: int64

Data Analysis

1. What is the total sales revenue?

```python
# @title 1. What is the total sales revenue?

# total sales revenue from 2014 to 2017
total_sales_revenue = df['Sales'].sum()
print(f"Total Sales Revenue: ${total_sales_revenue:.2f}\n")

# sales revenue- year wise
sales_revenue_by_year = df.groupby(df['Order Date'].dt.year)['Sales'].sum()
print(sales_revenue_by_year)

# visualization
# fig = px.bar(x=sales_revenue_by_year.index, y=sales_revenue_by_year.values, labels={'x': 'Year', 'y': 'Sales Revenue'})
# fig.update_layout(title='Sales Revenue by Year')
# fig.show()

plt.figure(figsize=(10,6))
sns.barplot(
    data=pd.DataFrame(sales_revenue_by_year),
    x=sales_revenue_by_year.index,
    y=sales_revenue_by_year.values,
    palette='light:g',
    hue=sales_revenue_by_year.index
)
plt.title('Sales Revenue by Year')
plt.xlabel('Year')
plt.ylabel('Sales Revenue')
plt.show()
```

```
Total Sales Revenue: $2297200.86

Order Date
2014    484247.4981
2015    470532.5090
2016    609205.5980
2017    733215.2552
Name: Sales, dtype: float64
```



Insight

- The dataset from 2014 to 2017 shows that the total sales revenue amounts to **$2,297,200.86**.
- This value reflects the cumulative gross revenue across all regions, product categories, and sales from *United States*
- The sales revenue per year shows that the **least sales are from year 2015** while **2017 has the most sales**:

```
2014:  $484,247.49
2015:  $470,532.50
2016:  $609,205.59
2017:  $733,215.25
```

- We notice most sales in year 2017 with **$733,215.25** in sales, indicating a growth trend over the years.
- The least sales were noted in year 2015 with with **$470,532.50** in sales, suggesting a potential dip in business activity or market demand during that year.

Conclusion

- The **Increase** in sales over the years, culminating in 2017, may indicate a successful business strategy, market expansion, or product-specific factors contributing to higher revenue.

Recommendation

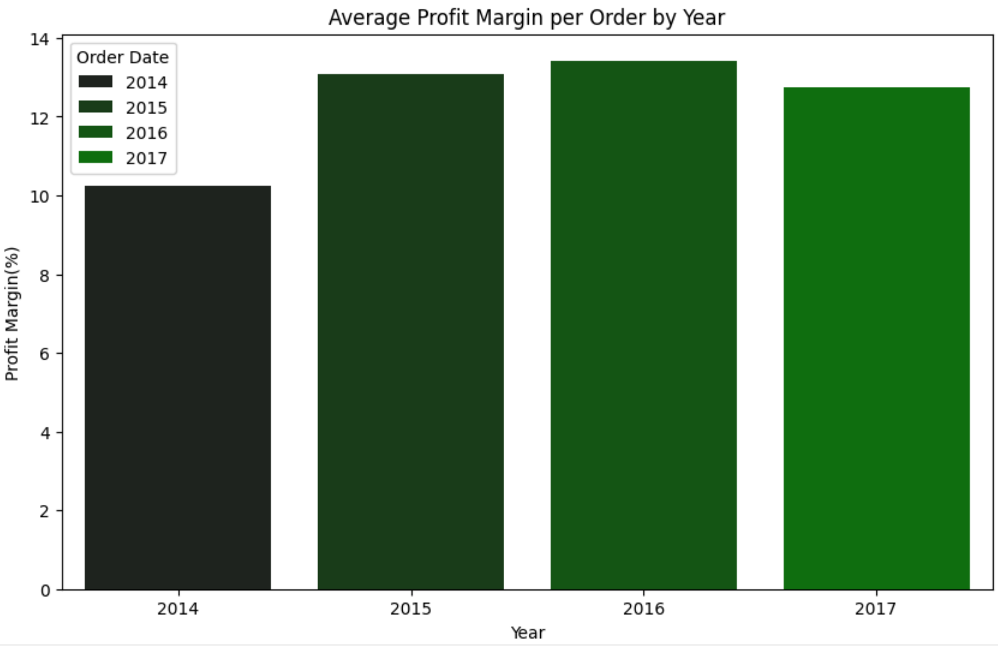
1. **Investigate 2015 Decline**:
   - Perform a deeper analysis to identify contributing factors (e.g., regions, product categories, or external economic influences).
2. **Leverage Insights from 2017**:
   - Identify the drivers of high sales in 2017 (e.g., successful campaigns, high-performing products) and replicate those strategies.
3. **Seasonality Analysis**:
   - Explore monthly or quarterly sales patterns to determine if there are seasonal trends impacting yearly totals.
4. **Regional and Category Breakdown**:
   - Evaluate the contribution of different regions and product categories to yearly sales to identify consistent high performers.

2. What is the average profit margin per order?

```
# @title 2. What is the average profit margin per order?

# profit margin from 2014 to 2017
profit_margin = pd.DataFrame(
    data=df['Profit'] / df['Sales'],
    columns=['Profit Margin']).mean().iloc[0]
# print(f"The average overall profit margin per order is: {profit_margin:.4f *}")
print(f"The average overall profit margin per order is: {(profit_margin*100):.2f}%")

# profit margin per year
profit_revenue_per_year = df.groupby(df['Order Date'].dt.year)['Profit'].sum()

profit_margin_per_year = pd.DataFrame(
    data=profit_revenue_per_year / sales_revenue_by_year,
    columns=['Profit Margin'])

for i in range(len(profit_margin_per_year)):
  print(f"The average profit margin per order in {profit_margin_per_year.index[i]} is: {profit_margin_per_year.iloc[i, 0]*100:.2f}%")
print("")

# visualization
profit_margin_per_year['Profit'] = profit_revenue_per_year
profit_margin_per_year['Sales'] = sales_revenue_by_year
```

```python
profit_margin_per_year.reset_index(inplace=True)

plt.figure(figsize=(10,6))
sns.barplot(
    data=profit_margin_per_year,
    x='Order Date',
    y=profit_margin_per_year['Profit Margin'] * 100,
    palette='dark:g',
    hue='Order Date'
)
plt.title('Average Profit Margin per Order by Year')
plt.xlabel('Year')
plt.ylabel('Profit Margin(%)')
plt.show()
```

```
The average overall profit margin per order is: 12.03%
The average profit margin per order in 2014 is: 10.23%
The average profit margin per order in 2015 is: 13.10%
The average profit margin per order in 2016 is: 13.43%
The average profit margin per order in 2017 is: 12.74%
```



Insight

- The dataset reveals the average overall profit margin per order to be **12.03%**, reflecting the profitability of individual transactions across all regions, product categories, and time periods.
- Breaking it down by year, we observe variations in profit margins:

```
2014:  10.23%
2015:  13.10%
2016:  13.43%
2017:  12.74%
```

- The highest average profit margin was recorded in 2016 (13.43%), while the lowest was in 2014 (10.23%).
- A steady increase in profit margin from 2014 to 2016 indicates improving profitability strategies.
- However, a slight decline in 2017 suggests potential challenges or cost increases impacting profitability.

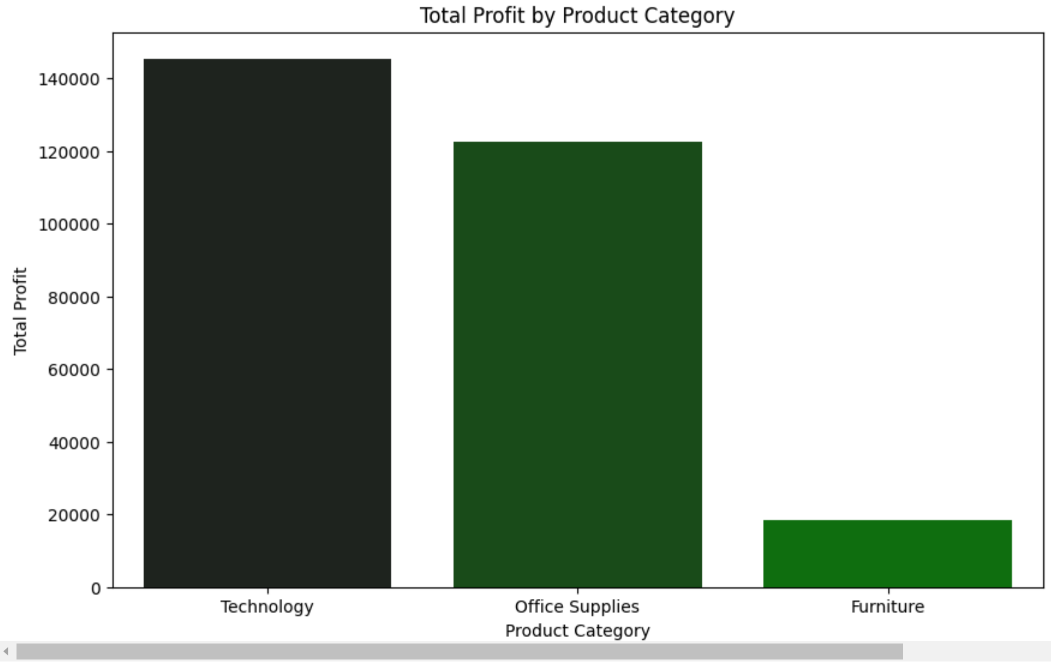
Conclusion

- The gradual improvement from 2014 to 2016 may indicate *enhanced cost management*, *product pricing strategies*, or a *shift toward more profitable product categories*.
- The dip in 2017, despite being a high-sales year, could suggest:
  - Increased costs (e.g., shipping, production, or marketing expenses).
  - A focus on high-volume, low-margin products or regions.

3. Which product category generates the highest total profit?

```python
# @title 3. Which product category generates the highest total profit?

# group sumation by product category
product_based_profit = df.groupby('Category')['Profit'].sum().sort_values(ascending=False)
print(product_based_profit)
print("")
print(f"The product category with the highest total profit is: {product_based_profit.index[0]}")

# vizualtion
plt.figure(figsize=(10,6))
sns.barplot(
    # index is category names
    x=product_based_profit.index,
    # values is total profit for each category
    y=product_based_profit.values,
    palette='dark:g',
    hue=product_based_profit.index
)
plt.title('Total Profit by Product Category')
plt.xlabel('Product Category')
plt.ylabel('Total Profit')
plt.show()
```

```
Category
Technology         145454.9481
Office Supplies    122490.8008
Furniture           18451.2728
Name: Profit, dtype: float64

The product category with the highest total profit is: Technology
```



Insight

- **Technology:** Generates the highest total profit at $145,454.95, representing a considerable lead over other categories.
- **Office Supplies:** Shows a substantial profit of $122,490.80, indicating a strong performance but lagging behind the Technology sector.
- **Furniture:** Reports the lowest total profit at $18,451.27, suggesting potential challenges in this category.
- The dataset reveals the total profit generated by each product category as follows:

```
Technology         $145,454.95
Office Supplies    $122,490.80
Furniture           $18,451.27
```

- The product category with the highest total profit is Technology, generating $145,454.95, which is 18.8% higher than the profit from Office Supplies, the second most profitable category.
- The Furniture category has significantly lower profitability compared to the other two categories, contributing only $18,451.27 to total profit.

Conclusion

- The dominance of the Technology category in terms of profitability can be attributed to higher margins on electronic products, premium pricing, or better alignment with today's customer demand.
- The Office Supplies category, despite contributing significantly to total sales, may have lower margins or higher cost structures compared to Technology.
- The Furniture category's low profit could result from:
  - Higher production and shipping costs.
  - Lower sales volume compared to the other categories.
  - Potentially lower demand for furniture items in the dataset's timeframe.

Recommendations

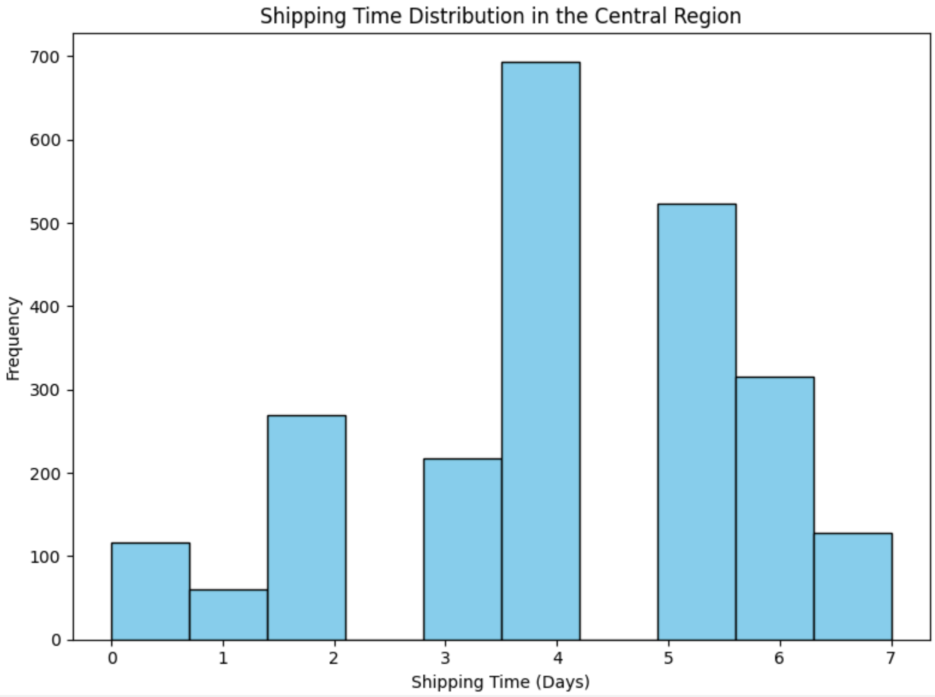
1. Focus on Technology:
   - Continue prioritizing and expanding the product range within the Technology category to sustain its profitability.
2. Improve Furniture Profitability:
   - Evaluate cost structures for furniture and explore strategies to reduce expenses (e.g., shipping costs or sourcing materials more efficiently).
   - Assess demand and consider repositioning furniture products to attract more buyers.
3. Enhance Office Supplies Margins:
   - Identify low-margin products within Office Supplies and explore pricing adjustments or alternative suppliers.
4. Customer Segmentation:
   - Segment customers by purchase behavior and preferences across different product categories to tailor marketing strategies and product offerings more effectively.

4. What is the average shipping time for orders in the Central region?

```python
# @title 4. What is the average shipping time for orders in the Central region?

shipping_time = df.copy(deep=True)
shipping_time['shipping_time'] = (df['Ship Date'] - df['Order Date']).dt.days
c_region = shipping_time[shipping_time['Region'] == 'Central']['shipping_time']
c_region_mean = c_region.mean()
print(f"The average shipping time for orders in the Central region is: {c_region_mean:.2f} days")
```

```
The average shipping time for orders in the Central region is: 4.06 days
```

```python
plt.figure(figsize=(8, 6))
plt.hist(c_region, bins=10, color='skyblue', edgecolor='black')
plt.title('Shipping Time Distribution in the Central Region')
plt.xlabel('Shipping Time (Days)')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



Insight

- The dataset reveals that the average shipping time for orders in the Central region is 4.06 days.

Explanation

- The average shipping time reflects the typical duration from order placement to delivery for customers in the Central region.
- Factors influencing shipping time may include:
    - Warehouse proximity to the delivery address.
    - Efficiency of logistics and shipping carriers.
    - Product availability and processing time before dispatch.

Recommendations

1. Improve Shipping Efficiency:
    - Investigate orders with shipping times significantly longer than the average to identify bottlenecks in the logistics process.
2. Set Delivery Expectations:
    - Clearly communicate expected delivery times to customers in the Central region, emphasizing the average shipping time of 4 days.
3. Analyze Other Regions:
    - Compare shipping times across regions to identify and replicate best practices from faster-performing areas.
4. Seasonal Adjustments:
    - Consider seasonal trends that may impact shipping times and plan for increased capacity during peak demand periods.

5. Which customer segment has the highest average order value?

```python
# @title 5. Which customer segment has the highest average order value?

# finding which segment is highest for sales alone
avg_count = df.groupby('Segment')['Sales'].sum().sort_values(ascending=False)
print(f"The customer segment with the highest average order value is: {avg_count.index[0]}\n")

# finding which segment is highest- region wise
avg_count_region = df.groupby(['Segment', 'Region'])['Sales'].sum()
avg_count_region = avg_count_region.unstack()
print(f"The customer segment with the highest average order value in each region is:")
print(avg_count_region)
print("")
print(f"The customer segment with the highest average order value in each region is:")
print(avg_count_region.idxmax())
print("")
```

```
The customer segment with the highest average order value is: Consumer

The customer segment with the highest average order value in each region is:
Region           Central        East        South         West
Segment
Consumer     252031.4340  350908.167  195580.9710  362880.7730
Corporate    157995.8128  200409.347  121885.9325  225855.2745
Home Office   91212.6440  127463.726   74255.0015  136721.7770

The customer segment with the highest average order value in each region is:
Region
Central    Consumer
East       Consumer
South      Consumer
```

```
West       Consumer
dtype: object
```

```python
# finding which segment is highest- state wise
avg_count_state = df.groupby(['Segment', 'State'])['Sales'].sum()
avg_count_state = avg_count_state.unstack()
print(f"The customer segment with the highest average order value in each state is:")
print(avg_count_state)
print("")
```

```
The customer segment with the highest average order value in each state is:
State         Alabama    Arizona  Arkansas   California   Colorado  Connecticut   Delaware  \
Segment
Consumer      7537.54  16424.422   8802.01  229636.0800  15794.492     5933.477  16961.763
Corporate    10969.38  11736.322   2463.78  147174.7265   9945.912     5715.690   8311.656
Home Office   1003.72   7121.257    412.34   80876.8250   6367.714     1735.190   2177.650

State        District of Columbia     Florida   Georgia     Idaho   Illinois   Indiana     Iowa  \
Segment
Consumer                  2753.34  32701.1960  24447.12  1444.496  45182.195  14986.96  2100.07
Corporate                     NaN  22477.5915  15982.25  2630.250  15984.280  31788.74   911.45
Home Office                111.68  34294.9205   8666.47   307.740  18999.626   6779.66  1568.24

State         Kansas  Kentucky  Louisiana    Maine   Maryland  Massachusetts   Michigan  \
Segment
Consumer      697.18  20430.72    6174.26      NaN  10054.013      11151.540  36709.911
Corporate     898.18   7927.83    1882.35  1164.45  11386.130       9639.594  23391.553
Home Office  1318.95   8233.20    1160.42   106.08   2265.380       7843.300  16168.150

State        Minnesota  Mississippi  Missouri   Montana  Nebraska    Nevada  New Hampshire  \
Segment
Consumer      19235.18      7688.58   5150.92   898.088   5261.25  6584.414        908.640
Corporate      3111.11      1362.72  10500.43    48.188   1266.74  4802.406        968.900
Home Office    7516.86      1720.04   6553.80  4643.076    936.94  5342.282       5414.984

State        New Jersey  New Mexico    New York  North Carolina  North Dakota       Ohio  \
Segment
Consumer      13333.982    2186.324  175209.035       29997.226        891.53  43194.024
Corporate     18268.190    1269.776   77951.313       18656.746           NaN  24209.973
Home Office    4162.140    1327.422   57715.923        6949.192         28.38  10854.139

State        Oklahoma    Oregon  Pennsylvania  Rhode Island  South Carolina  South Dakota  \
Segment
Consumer     11561.77  8893.933     66899.293      2483.336         5539.75         45.73
Corporate     2569.75  6563.970     31130.061      5381.150         2916.04       1269.83
Home Office   5551.87  1973.247     18482.560     14763.470           25.92           NaN

State        Tennessee       Texas      Utah  Vermont  Virginia  Washington  West Virginia  \
Segment
Consumer     16578.939  95976.3780  7152.004  1352.38  35683.63    73866.52        673.344
Corporate     9745.765  53908.1198  1956.614  6282.24  27501.48    39727.11            NaN
Home Office   4337.169  20303.5480  2111.438  1294.75   7451.61    25047.64        536.480

State        Wisconsin   Wyoming
Segment
Consumer      14232.36       NaN
Corporate     12395.63       NaN
Home Office    5486.62  1603.136
```

```python
# state wise customer segment with the highest average order value

print(f"The customer segment with the highest average order value in each state is:")
print(avg_count_state.idxmax())
print("")
```

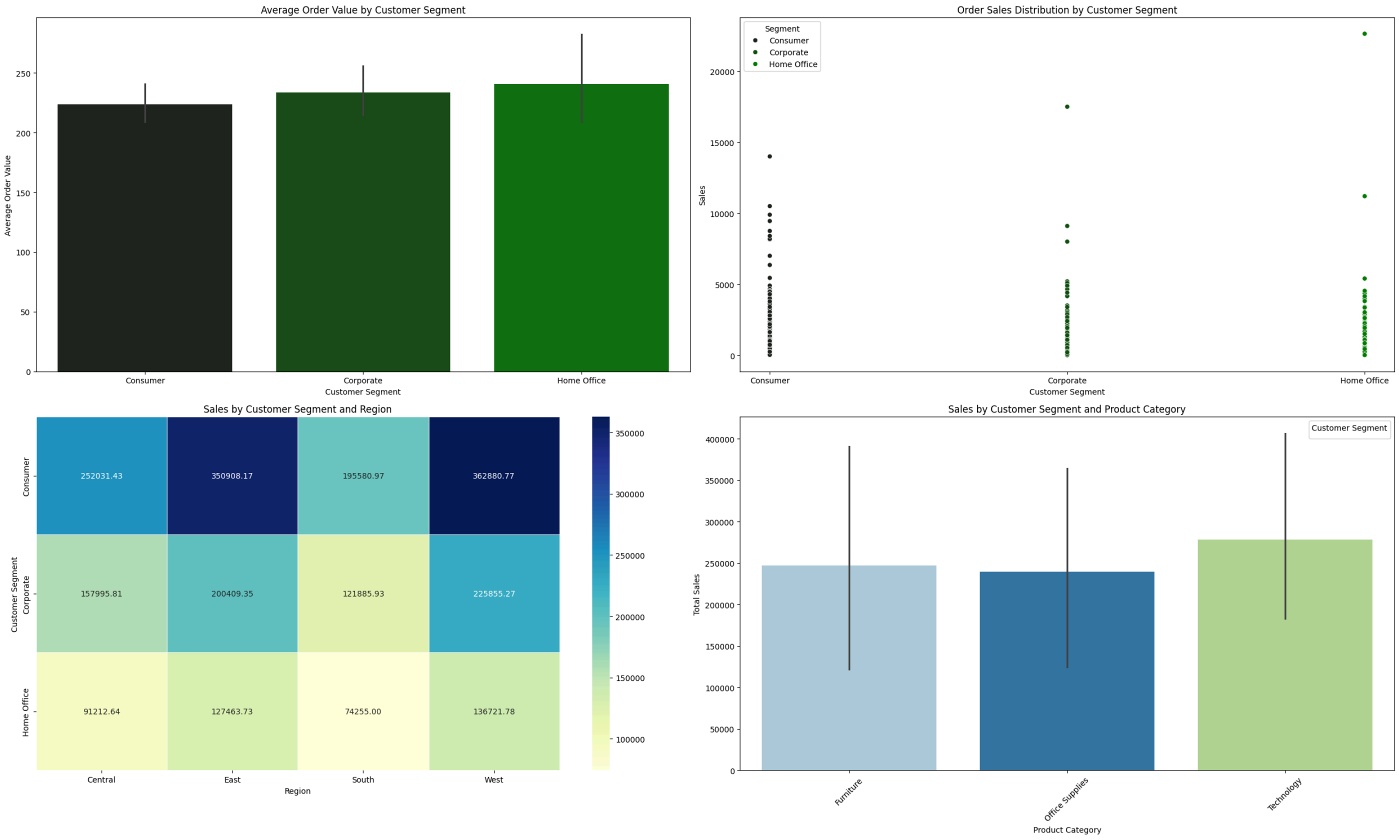
```
The customer segment with the highest average order value in each state is:
State
Alabama                    Corporate
Arizona                     Consumer
Arkansas                    Consumer
California                  Consumer
Colorado                    Consumer
Connecticut                 Consumer
Delaware                    Consumer
District of Columbia        Consumer
Florida                  Home Office
Georgia                     Consumer
Idaho                      Corporate
Illinois                    Consumer
Indiana                    Corporate
Iowa                        Consumer
Kansas                   Home Office
Kentucky                    Consumer
Louisiana                   Consumer
Maine                      Corporate
Maryland                   Corporate
Massachusetts               Consumer
Michigan                    Consumer
Minnesota                   Consumer
Mississippi                 Consumer
Missouri                   Corporate
Montana                  Home Office
Nebraska                    Consumer
Nevada                      Consumer
New Hampshire            Home Office
New Jersey                 Corporate
New Mexico                  Consumer
New York                    Consumer
North Carolina              Consumer
North Dakota                Consumer
Ohio                        Consumer
Oklahoma                    Consumer
Oregon                      Consumer
Pennsylvania                Consumer
Rhode Island             Home Office
South Carolina              Consumer
South Dakota               Corporate
Tennessee                   Consumer
Texas                       Consumer
Utah                        Consumer
Vermont                    Corporate
Virginia                    Consumer
Washington                  Consumer
West Virginia               Consumer
Wisconsin                   Consumer
Wyoming                  Home Office
dtype: object
```

```python
# finding which segment is highest- product and profit wise
avg_count_product_sales = df.groupby(['Segment', 'Category'])['Sales'].sum()
avg_count_product_sales = avg_count_product_sales.unstack()
print(f"The customer segment with the highest average order value in each product category is:")
print(avg_count_product_sales)
print("")
print(f"The customer segment with the highest average order value in each product category is:")
print(avg_count_product_sales.idxmax())
print("")

avg_count_product_profit = df.groupby(['Segment', 'Category'])['Profit'].sum()
avg_count_product_profit = avg_count_product_profit.unstack()
print(f"The customer segment with the highest average order value in each product category is:")
print(avg_count_product_profit)
print("")
print(f"The customer segment with the highest average order value in each product category is:")
print(avg_count_product_profit.idxmax())
print("")
```

```
The customer segment with the highest average order value in each product category is:
Category       Furniture  Office Supplies  Technology
Segment
Consumer     391049.3120       363952.136  406399.897
Corporate    229019.7858       230676.462  246450.119
Home Office  121930.6975       124418.434  183304.017

The customer segment with the highest average order value in each product category is:
Category
Furniture          Consumer
Office Supplies    Consumer
Technology         Consumer
dtype: object

The customer segment with the highest average order value in each product category is:
Category     Furniture  Office Supplies  Technology
Segment
Consumer     6991.0786       56330.3210  70797.8096
Corporate    7584.8158       40227.3202  44166.9980
Home Office  3875.3784       25933.1596  30490.1405

The customer segment with the highest average order value in each product category is:
Category
Furniture          Corporate
Office Supplies     Consumer
Technology          Consumer
dtype: object
```

```python
# visualization
# show catplot and scatterplot side by side

# Create subplots
fig, axs = plt.subplots(2, 2, figsize=(25, 15))

# Bar plot for average sales by customer segment
sns.barplot(
    data=df,
    x='Segment',
    y='Sales',
    ax=axs[0,0],
    palette='dark:g',
    hue='Segment'
)
axs[0,0].set_title('Average Order Value by Customer Segment')
axs[0,0].set_xlabel('Customer Segment')
axs[0,0].set_ylabel('Average Order Value')

# Scatter plot for sales by customer segment
sns.scatterplot(
    data=df,
    x='Segment',
    y='Sales',
    hue='Segment',
    palette='dark:g',
    ax=axs[0,1],
    hue_order=['Consumer', 'Corporate', 'Home Office']
)
axs[0,1].set_title('Order Sales Distribution by Customer Segment')
axs[0,1].set_xlabel('Customer Segment')
axs[0,1].set_ylabel('Sales')

# Heatmap of sales by region and segment
sns.heatmap(
    data=avg_count_region,
    annot=True,
    fmt=".2f",
    cmap="YlGnBu",
    linewidths=0.5,
    ax=axs[1,0]
)
axs[1,0].set_title("Sales by Customer Segment and Region")
axs[1,0].set_xlabel("Region")
axs[1,0].set_ylabel("Customer Segment")

# Bar plot for sales by product category
sns.barplot(
    data=avg_count_product_sales,
    ax=axs[1,1],
    palette="Paired"
)
axs[1,1].set_title("Sales by Customer Segment and Product Category")
axs[1,1].set_xlabel("Product Category")
axs[1,1].set_ylabel("Total Sales")
axs[1,1].set_xticklabels(axs[1,1].get_xticklabels(), rotation=45)
axs[1,1].legend(title="Customer Segment")

# Adjust layout
plt.tight_layout()
plt.show()
```

```
<ipython-input-17-3fc64558b929>:56: UserWarning: set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks() or using a Fi
  axs[1,1].set_xticklabels(axs[1,1].get_xticklabels(), rotation=45)
<ipython-input-17-3fc64558b929>:57: UserWarning: No artists with labels found to put in legend.  Note that artists whose label start with an underscore
  axs[1,1].legend(title="Customer Segment")
```



Insights

1. Overall Sales Performance by Segment:
    - The customer segment with the highest total sales is Consumer, outperforming Corporate and Home Office segments.
2. Region-Wise Performance:
    - Consumer consistently has the highest average order value across all

```
regions:
Central :  $ 252,031.43
  East  :  $ 350,908.17
 South  :  $ 195,580.97
  West  :  $ 362,880.77
```

    - Home Office lags behind in all regions with the lowest average sales.
3. State-Wise Performance:
    - Consumer dominates in most states for the highest average order value.
    - Notable deviations:
        - Corporate leads in states such as Alabama, Missouri, and Vermont.
    - Home Office outperforms in Montana, New Hampshire, Rhode Island, and Wyoming.
    - Specific state-level outliers show potential niche opportunities for Corporate and Home Office segments.
4. Product Category Performance: Sales Perspective:
    - Consumer segment performs best in all three categories:

```
   Furniture     :  $ 391,049.31
Office Supplies  :  $ 363,952.13
   Technology    :  $ 406,399.89
```

    - Profit Perspective:
        - Corporate overtakes Consumer in Furniture, showcasing higher profitability potential.
    - Consumer remains dominant in Office Supplies and Technology.

Explaination

- Overall Segment Performance:

    - Consumer leads in total sales across regions, states, and product categories, indicating strong market presence and higher transaction volume. Corporate shows profitability in Furniture, and Home Office underperforms overall but has niche dominance in specific states.
- Region-Wise Insights:
    - Consumer dominates all regions, with the West region contributing the highest sales.
    - Home Office struggles, requiring targeted strategies for growth.
- State-Specific Trends:
    - Consumer leads in most states, while Corporate and Home Office perform well in select states like Alabama and Rhode Island, respectively.
- Category Performance:
    - Consumer excels in Technology and Office Supplies, while Corporate outperforms in Furniture profitability.

Recommendation

- Regional Strategy:
    - Focus on Consumer in regions where it already dominates to maximize revenue.
    - Prioritize resources to sustain Consume's lead in high-performing regions like the West.
    - Leverage Corporate's potential in underperforming states.
    - Invest in Technology and Office Supplies categories where Consumer already has a stronghold. [change]
- Product Strategy:
    - Prioritize Consumer for technology-related sales due to its substantial lead.
    - Increase Corporate's penetration in Furniture for profitability.
- Segment-Specific Campaigns:
    - Invest in tailored promotions to boost Home Office sales in smaller or niche markets.