E-Commerce Sales EDA by Blessing Osuagwu (GH1038684)

Business Context: The E-commerce industry has seen exponential growth over the past decade, driven by advancements in technology and changing consumer behavior. With the rise of online shopping, businesses are increasingly relying on data analytics to help them understand customer preferences, optimize inventory, and improve customer satisfaction.

Our client, a diversified e-commerce retailer, operates in a highly competitive market where understanding customer demographics, product performance, and regional sales trends is crucial for maintaining a competitive edge. By leveraging transactional data, the company aims to identify key areas for improvement, such as targeted marketing campaigns, inventory management, and shipping efficiency.

They have provided a transactional dataset in CSV format. It contains comprehensive transactional details related to e-commerce sales. Understanding its structure and content is fundamental to extracting meaningful insights. The dataset is composed of various key features, offering a view of customer purchases and their associated attributes. While a full statistical summary is presented in later sections, here's a preliminary look at its structure:

Customer ID: A unique identifier for each customer.

Gender: The self-identified gender of the customer.

Age: The age of the customer at the time of purchase.

Region: The geographical region where the customer is located.

Product Name: The specific name of the product purchased.

Category: The product category to which the item belongs (e.g., Electronics).

Unit Price: The price of a single unit of the product.

Quantity: The number of units of the product purchased in a single transaction.

Total Price: The total cost of the transaction, typically Unit Price multiplied by Quantity plus Shipping Fee.

Shipping Fee: The cost associated with shipping the order.

Shipping Status: The current status of the product's delivery (e.g. Delivered).

Order Date: The date when the order was placed.

This dataset provides the foundation for our exploratory data analysis, allowing us to delve into customer demographics, product performance, and sales trends.The dataset is accompanied by eight (8) business questions that require analysis and insights.

My expertise is required to perform a thorough analysis of their sales data in order to: identify peak sales periods and seasonal trends, segment their customers for targeted marketing, evaluate the performance of their product categories, optimize their shipping and delivery processes, and ultimately, provide data-driven recommendations that will help them achieve their strategic business goals.

Data Source:

This dataset is derived from Kaggle (https://www.kaggle.com/datasets/brsahan/e-commerce-dataset/data?select=realistic_e_commerce_sales_data.csv)

A Quick Overview of the Business Questions to be answered after EDA

Q1. Which age group generates the most revenue?

Q2. Which product categories have highest average basket size?

Q3. Do higher priced categories sell fewer total units?

Q4. How does customer location impact revenue?

Q5. What are the peak sales periods?

Q6. What are shipping trends and their impact on avg order value?

Q7. What are the most common returns (Top 8)? And how many returned?

Q8. Which regions have the highest return rates?

Importing Libraries

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading Dataset

In [2]:
```
# Load dataset derived from Kaggle
df_org = pd.read_csv('/content/realistic_e_commerce_sales_data.csv')
```

Dataset Overview

In [3]:
```
# Check dataset shape
df_org.shape
```

Out[3]:
```
(1000, 12)
```

In [4]:
```
# Display first five rows
df_org.head(5)
```

Out[4]:

	Customer ID	Gender	Region	Age	Product Name	Category	Unit Price	Quantity	Total Price	Shi
0	CUST0268	Male	North	NaN	Monitor	Electronics	300.0	5	1500	
1	CUST0046	Male	West	22.0	Headphones	Accessories	100.0	2	200	
2	CUST0169	Female	South	54.0	Monitor	Electronics	300.0	1	300	
3	CUST0002	Male	North	23.0	Headphones	Accessories	100.0	5	500	
4	CUST0173	Female	South	NaN	Laptop	Electronics	1500.0	3	4500	

In [5]:
```
# Get dataset information
df_org.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Customer ID      1000 non-null   object 
 1   Gender           1000 non-null   object 
 2   Region           950 non-null    object 
 3   Age              900 non-null    float64
 4   Product Name     1000 non-null   object 
 5   Category         1000 non-null   object 
 6   Unit Price       1000 non-null   float64
 7   Quantity         1000 non-null   int64  
 8   Total Price      1000 non-null   int64  
 9   Shipping Fee     1000 non-null   float64
 10  Shipping Status  950 non-null    object 
 11  Order Date       1000 non-null   object 
dtypes: float64(3), int64(2), object(7)
memory usage: 93.9+ KB
```

The dataset has 1000 rows and 12 columns. There are missing values in the Region, Age, and Shipping This dataset contains a combination of numerical and categorical data.

In [6]:
```
#Check the data types.
df_org.dtypes
```

Out[6]:

	0
Customer ID	object
Gender	object
Region	object
Age	float64
Product Name	object
Category	object
Unit Price	float64
Quantity	int64
Total Price	int64
Shipping Fee	float64
Shipping Status	object
Order Date	object

dtype: object

In [7]:
```
#Check for missing (empty) values in the dataset. As seen below,there are visible m
df_org.isnull()
```

Out[7]:

	Customer ID	Gender	Region	Age	Product Name	Category	Unit Price	Quantity	Total Price	Shippin Fe
0	False	False	False	True	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	True	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	
995	False	False	False	False	False	False	False	False	False	Fals
996	False	False	False	False	False	False	False	False	False	Fals
997	False	False	False	True	False	False	False	False	False	Fals
998	False	False	False	False	False	False	False	False	False	Fals
999	False	False	True	False	False	False	False	False	False	Fals

1000 rows × 12 columns

In [8]:
```
# Find out the sum of missing values in each column.
df_org.isna().sum()
```

Out[8]:

	0
Customer ID	0
Gender	0
Region	50
Age	100
Product Name	0
Category	0
Unit Price	0
Quantity	0
Total Price	0
Shipping Fee	0
Shipping Status	50
Order Date	0

dtype: int64

In [9]:
```
#Check for duplicates
df_org.duplicated().values.any()
```

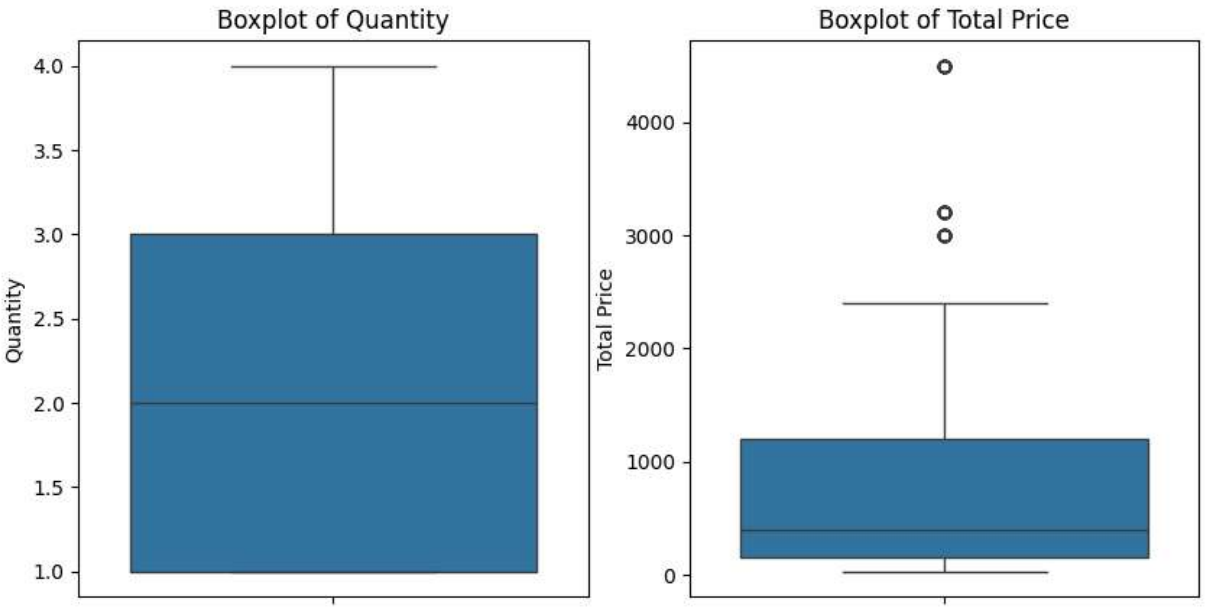
Out[9]: np.False_

Data Cleaning

In [10]:
```
# Detect outliers in Quantity and Total Price by calculating the 99th and 1st perce

upper_limit = df_org["Quantity"].quantile(0.99)
lower_limit = df_org["Quantity"].quantile(0.01)
print(upper_limit)
print(lower_limit)

upper_limit2 = df_org["Total Price"].quantile(0.99)
lower_limit2 = df_org["Total Price"].quantile(0.01)
print(upper_limit2)
print(lower_limit2)

# Remove outliers in Quantity and Total Price and filter the dataset to exclude val
df_org = df_org[df_org["Quantity"] < df_org["Quantity"].quantile(0.99)]
df_org = df_org[df_org["Total Price"] < df_org["Total Price"].quantile(0.99)]
```

```
5.0
1.0
7500.0
30.0
```

In [11]:
```
#Boxplot visualization for outlier
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
sns.boxplot(y=df_org["Quantity"])
plt.title("Boxplot of Quantity")

plt.subplot(1, 2, 2)
sns.boxplot(y=df_org["Total Price"])
plt.title("Boxplot of Total Price")

plt.show()

print(df_org["Quantity"].describe())
print(df_org["Total Price"].describe())
```




```
count    772.000000
mean       2.459845
std        1.095774
min        1.000000
25%        1.000000
50%        2.000000
75%        3.000000
max        4.000000
Name: Quantity, dtype: float64
count     772.000000
mean      903.432642
std      1120.592651
min        30.000000
25%       150.000000
50%       400.000000
75%      1200.000000
max      4500.000000
Name: Total Price, dtype: float64
```

In [12]:
```
# Fill missing Age values with the median age and convert to int64
df_org["Age"] = df_org["Age"].fillna(df_org["Age"].median()).astype("int64")
```

In [13]:
```
# Fill missing Region values with the most common region (mode)
df_org["Region"] = df_org["Region"].fillna(df_org["Region"].mode()[0])
```

In [14]:
```
# Fill missing Shipping Status with "Unknown" (or use the mode)
df_org["Shipping Status"] = df_org["Shipping Status"].fillna("Unknown")
```

In [15]:
```
# Convert Order Date to datetime format
df_org["Order Date"] = pd.to_datetime(df_org["Order Date"], errors="coerce")

# Remove future dates
df_org = df_org[df_org["Order Date"] <= pd.Timestamp.today()]
```

In [16]:
```
# Convert Shipping Status to consistent capitalization
df_org["Shipping Status"] = df_org["Shipping Status"].str.title()
```

In [17]:
```
# Standardize product names (remove extra spaces and make lowercase)
df_org["Product Name"] = df_org["Product Name"].str.strip().str.lower()
```

In [18]:
```
print(df_org.shape) # Check if changes were applied correctly
```

```
(772, 12)
```

In [19]:
```
print(df_org.info())  # Check if changes were applied correctly
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 772 entries, 1 to 999
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype         
---  ------           --------------  -----         
 0   Customer ID      772 non-null    object        
 1   Gender           772 non-null    object        
 2   Region           772 non-null    object        
 3   Age              772 non-null    int64         
 4   Product Name     772 non-null    object        
 5   Category         772 non-null    object        
 6   Unit Price       772 non-null    float64       
 7   Quantity         772 non-null    int64         
 8   Total Price      772 non-null    int64         
 9   Shipping Fee     772 non-null    float64       
 10  Shipping Status  772 non-null    object        
 11  Order Date       772 non-null    datetime64[ns]
dtypes: datetime64[ns](1), float64(2), int64(3), object(6)
memory usage: 78.4+ KB
None
```

Data Cleaning The raw dataset often contains inconsistencies, missing values, and outliers that can skew analysis and lead to inaccurate insights. Therefore, a crucial step in our Exploratory Data Analysis was to perform robust data cleaning. This phase ensured the dataset was reliable, consistent, and ready for meaningful analysis.

The cleaning process involved several key steps:

Initial Data Assessment: We began by examining the dataset's overall shape (number of rows and columns) and reviewing the first few rows to get a preliminary understanding of its structure and content. This was followed by a detailed inspection of data types for each column using df.info(). We confirmed the presence of 1000 entries and 12 columns, with a mix of numerical (float64, int64) and object (string) data types. Missing Value Identification and Imputation: We systematically checked for missing values across all columns using df.isnull().sum(). We identified 100 missing values in the 'Age' column, 50 in 'Region', and 50 in 'Shipping Status'to handle these: Missing 'Age' values were filled with the median age of the dataset to maintain the central tendency and then converted to an integer data type. Missing 'Region' values were imputed with the most frequently occurring region (mode) to preserve the dominant regional distribution. Missing 'Shipping Status' values were addressed by filling them with 'Unknown', providing a clear category for unrecorded statuses. Duplicate Check: A check for duplicate rows was performed using df.duplicated().values.any() to ensure uniqueness of records. No duplicates were found in the dataset. Outlier Detection and Removal: Outliers in 'Quantity' and 'Total Price' can significantly distort statistical measures and visualizations. We detected outliers by calculating the 1st and 99th percentiles for both columns. Values exceeding the 99th percentile for 'Quantity' (5.0) and 'Total Price' (7500.0) were removed to create a more representative dataset for analysis. Boxplots were generated to visually confirm the distribution and the impact of outlier removal on these numerical features.

Data Type Conversion and Standardization: The 'Order Date' column was converted from an object data type to a datetime format. This is essential for time-series analysis. Future dates (if any) were also removed to ensure data validity.

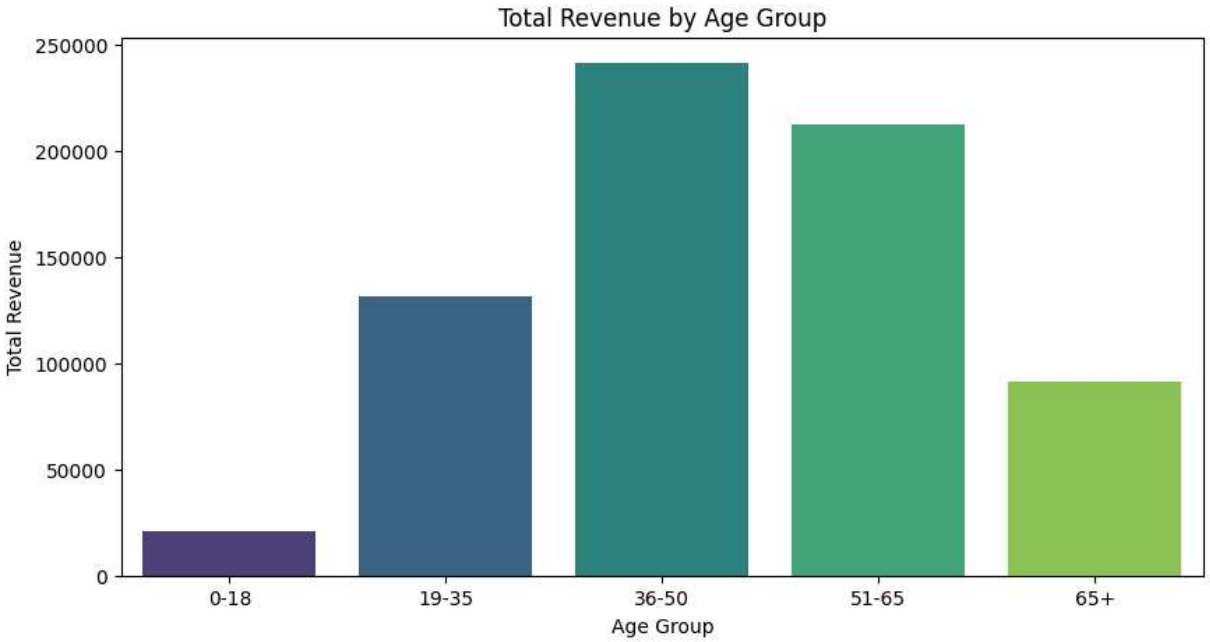
'Shipping Status' values were standardized to consistent capitalization (e.g., "delivered" to "Delivered") using .str.title() to ensure uniform categorization. 'Product Name' entries were cleaned by removing leading/trailing spaces and converted to lowercase using .str.strip().str.lower() to standardize product identification.

After these comprehensive cleaning steps, the dataset was refined to 772 rows and 12 columns, ready for in-depth analysis.

Business Q1: Which Age Group Generates the Most Revenue?

In [79]:
```
df_org["Age Group"] = pd.cut(df_org["Age"], bins=[0, 18, 35, 50, 65, 100], labels=[

age_revenue = df_org.groupby("Age Group", observed=False)["Total Price"].sum().sort

plt.figure(figsize=(10, 5))
sns.barplot(x=age_revenue.index, y=age_revenue.values, hue=age_revenue.index, palet
plt.title("Total Revenue by Age Group")
plt.xlabel("Age Group")
plt.ylabel("Total Revenue")
```

```
plt.show()
print(age_revenue)
```




```
Age Group
36-50    241290
51-65    212340
19-35    131310
65+       91460
0-18      21050
Name: Total Price, dtype: int64
```

Does average order value increase with age?

In [37]:
```
age_avg_order = df_org.groupby('Age').agg({'Revenue':'mean'}).reset_index()
plt.figure(figsize=(8,5))
sns.lineplot(x='Age', y='Revenue', data=age_avg_order, marker='o', color='darkblue'
plt.title("Avg Order Value vs Age")
plt.ylabel("Avg Order Value ($)")
plt.show()
```

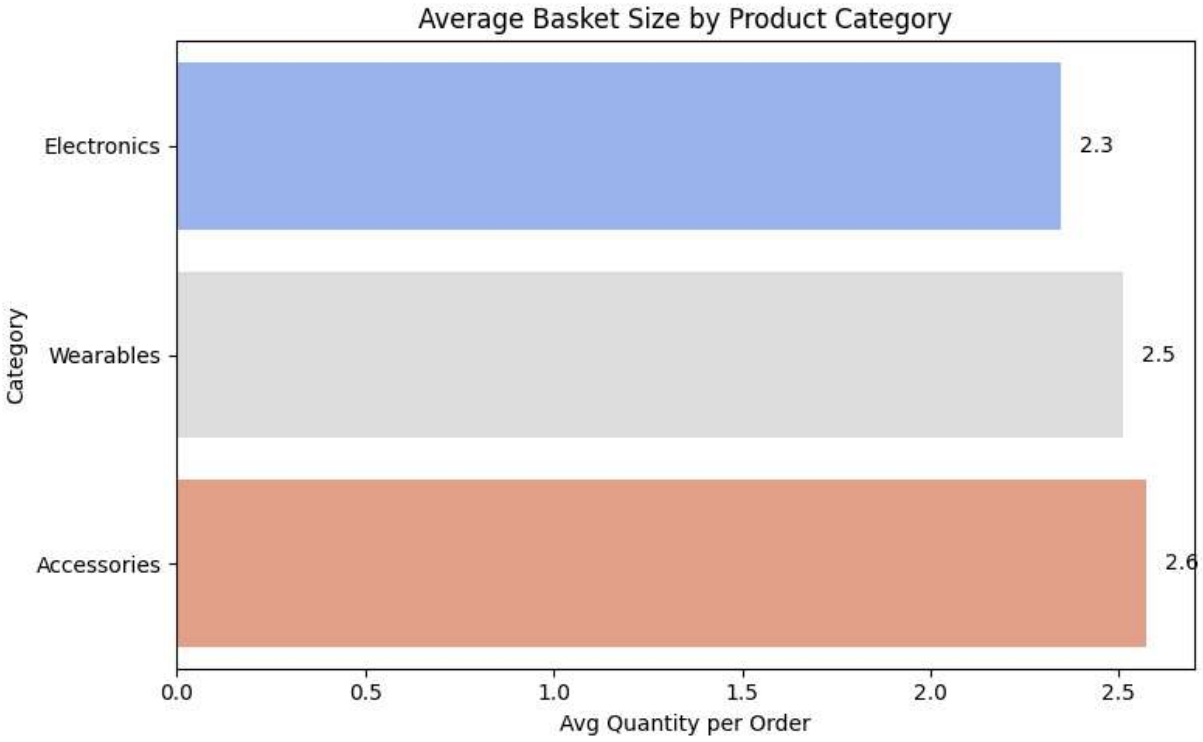
Summary of Analysis: We first calculated a 'Revenue' column for each transaction by summing 'Total Price' and 'Shipping Fee'. Then, customers were segmented into predefined age groups. The total revenue for each of these age groups was then calculated and visualized using a horizontal bar chart.

Answer: Based on the analysis (as shown in the "Total Revenue by Age Group" chart), the 36-50 age group consistently generates the highest revenue for the e-commerce retailer and Older customers (50+) have higher average order values.This indicates that this demographic is the most valuable and should be a primary focus for marketing efforts and product development.

Business Q2: Which product categories have highest average basket size?

In [80]:
```
category_basket = df_org.groupby('Category')['Quantity'].mean().sort_values()

plt.figure(figsize=(8,5))
sns.barplot(x=category_basket.values,
            y=category_basket.index,
            hue=category_basket.index,
            palette='coolwarm',
            dodge=False,
            legend=False)
plt.title("Average Basket Size by Product Category")
plt.xlabel("Avg Quantity per Order")
```

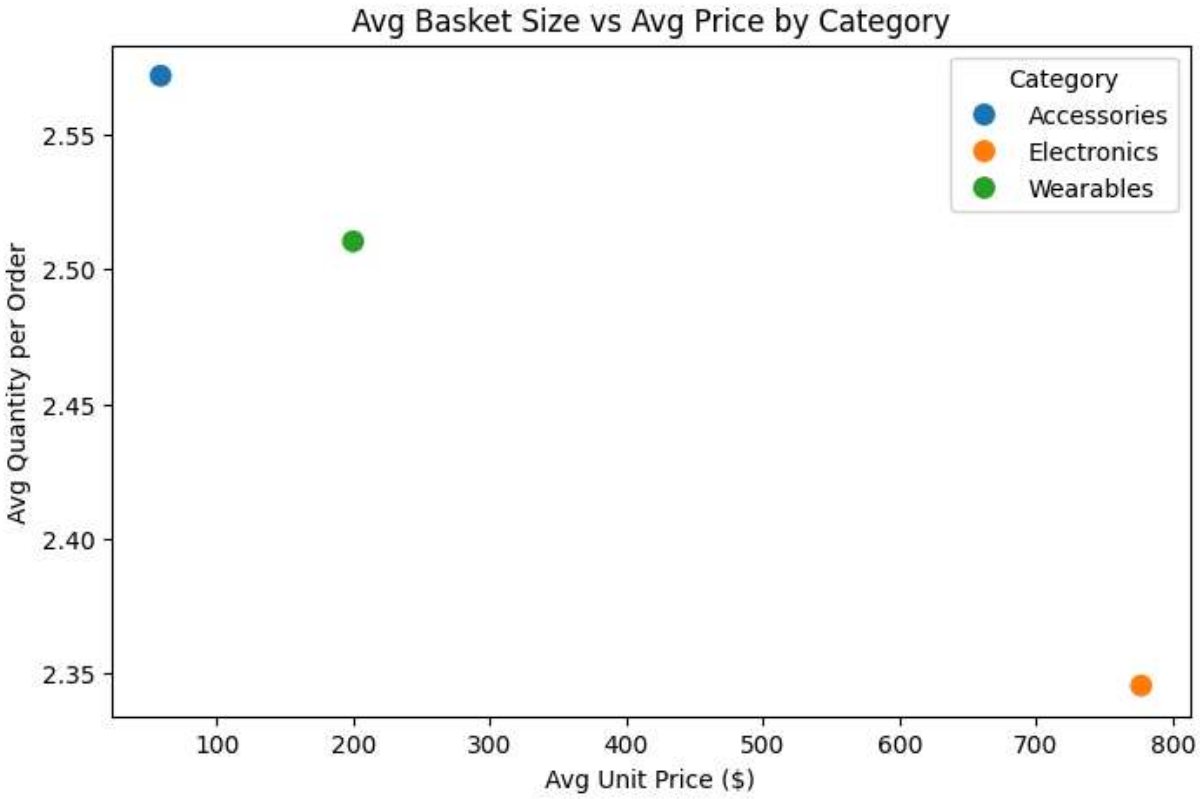
```
for i, v in enumerate(category_basket.values):
    plt.text(v + 0.05, i, f"{v:.1f}", va='center')
plt.tight_layout()
plt.show()

print(category_basket)
```




```
Category
Electronics    2.345506
Wearables      2.510417
Accessories    2.571875
Name: Quantity, dtype: float64
```

Visualize if there is a relationship between avg price and basket size

In [42]:
```
category_stats = df_org.groupby('Category').agg({'Quantity':'mean', 'Unit Price':'m
plt.figure(figsize=(8,5))
sns.scatterplot(x='Unit Price', y='Quantity', data=category_stats, hue='Category',
plt.title("Avg Basket Size vs Avg Price by Category")
plt.xlabel("Avg Unit Price ($)")
plt.ylabel("Avg Quantity per Order")
plt.show()
```



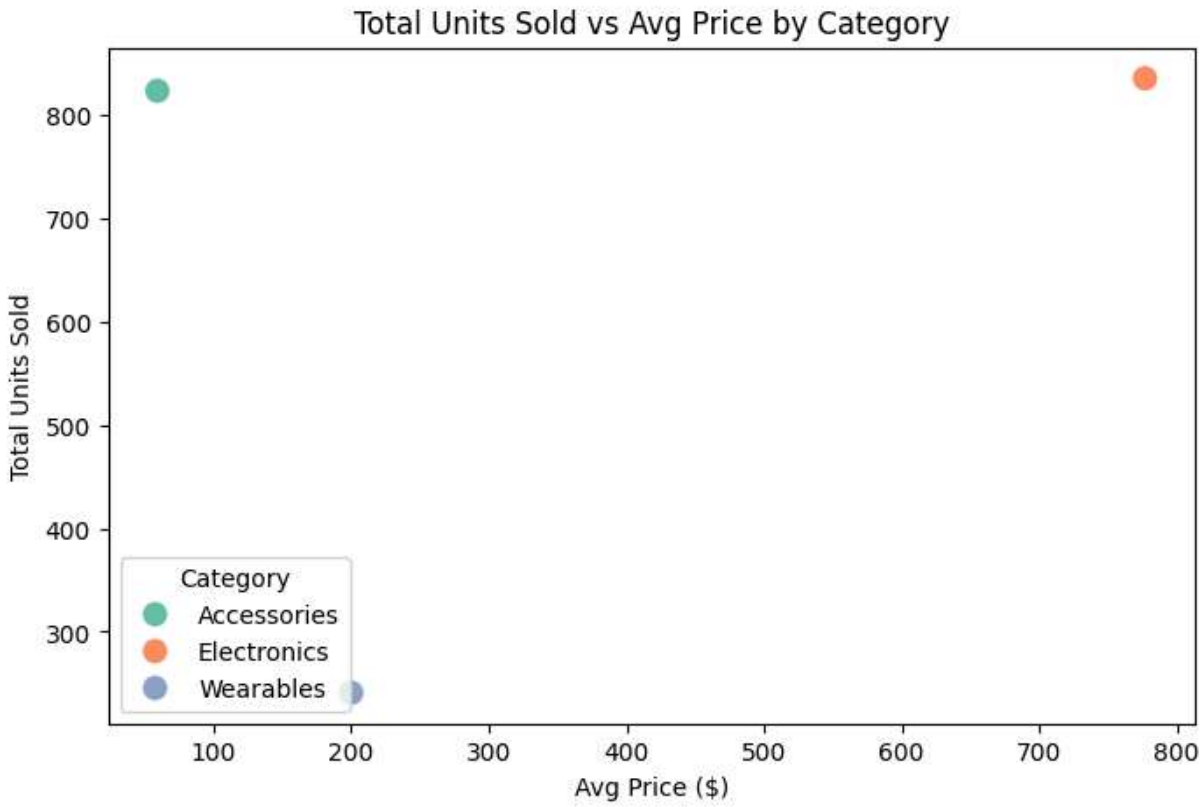
Summary of Analysis: To understand purchasing habits, we computed the average 'Quantity' of items per order for each 'Product Category'. The results were then sorted and presented via a horizontal bar chart to highlight categories where customers tend to buy more items in a single transaction.

Answer: From the "Average Basket Size by Product Category" chart, Accessories show the highest average basket size (approximately 2.6 units per order), followed closely by Wearables (2.5 units) and Electronics (2.3 units). This suggests that customers are more likely to purchase multiple units of accessories in one go, as higher prices slightly reduce basket size. Initiating bundled offers could help as a strategy.

Business Q3: Do higher priced categories sell fewer total units?

In [117…

```
category_units = df_org.groupby('Category').agg({'Quantity':'sum', 'Unit Price':'me
plt.figure(figsize=(8,5))
sns.scatterplot(x='Unit Price', y='Quantity', data=category_units, hue='Category',
plt.title("Total Units Sold vs Avg Price by Category")
plt.xlabel("Avg Price ($)")
plt.ylabel("Total Units Sold")
plt.show()

print(category_units)
```

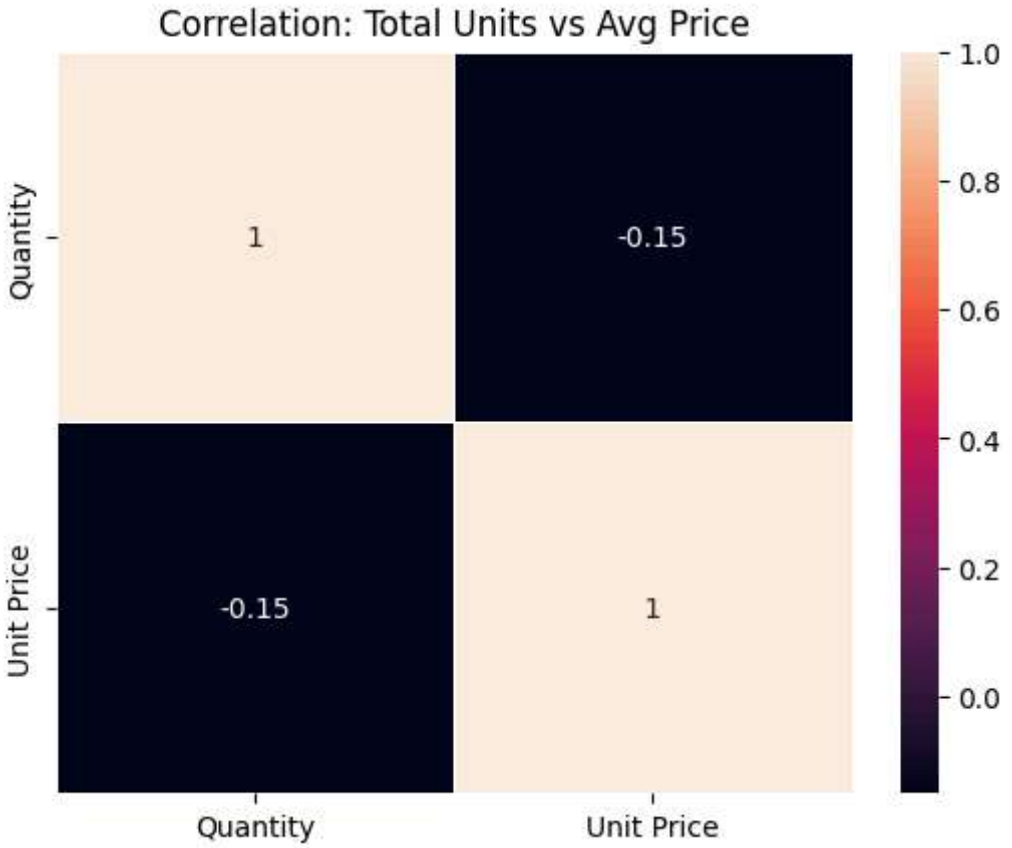


	Category	Quantity	Unit Price
0	Accessories	823	59.120268
1	Electronics	835	776.854367
2	Wearables	241	200.000000

Visualize the correlation of category-level data

In [106…

```
category_corr = df_org[['Quantity', 'Unit Price']].corr()
sns.heatmap(category_corr, annot=True, cmap='rocket', linewidths=0.5)
plt.title("Correlation: Total Units vs Avg Price")
plt.show()
```



Summary of Analysis: To investigate the relationship between a product category's average price and its overall sales volume, we first aggregated our dataset by 'Category'. For each category, we calculated the total 'Quantity' sold (representing total units) and the mean 'Unit Price'. This aggregated data was then used to create a scatter plot visualizing 'Total Units Sold' against 'Avg Price' for each category, with different categories distinguished by color. This plot allows for a direct visual assessment of how sales volume varies with price across product categories.

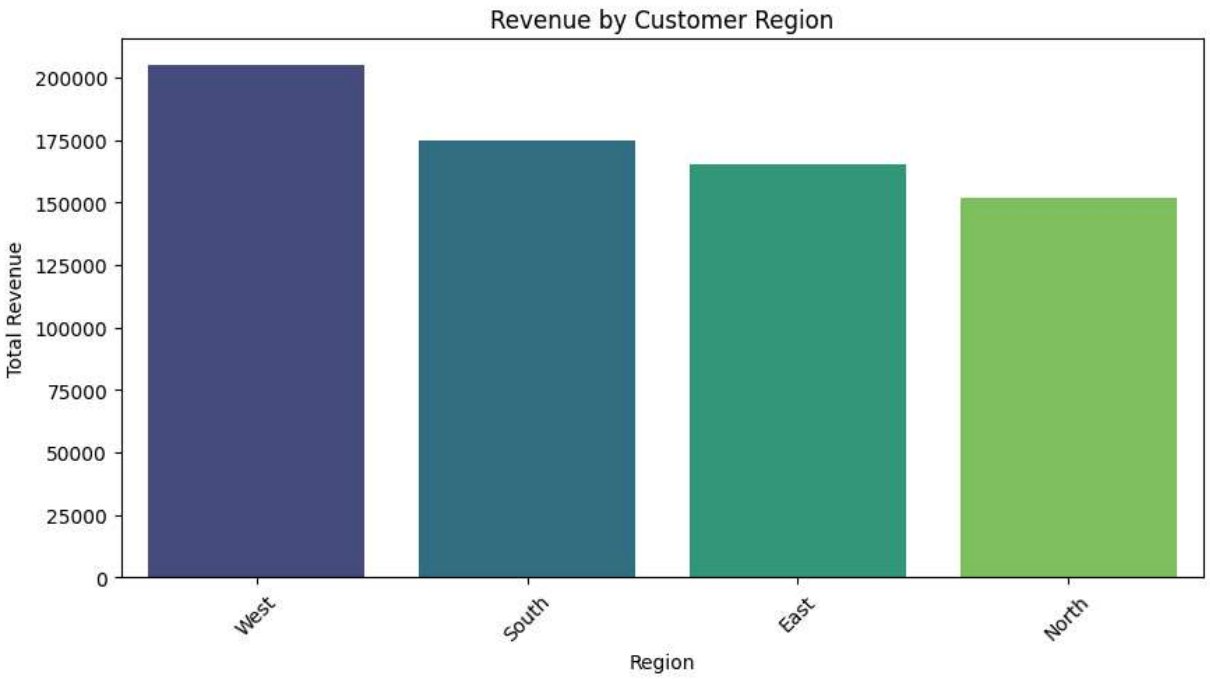
Answer: Based on the category_units data and the "Total Units Sold vs Avg Price by Category" scatter plot:

No, higher-priced categories do not sell fewer total units. In fact, the opposite trend is observed among these categories:

Electronics, which has the highest average unit price (776.85), also boasts the highest total units sold (835). Wearables have a moderate average price (200.00) but the lowest total units sold (241). Accessories have the lowest average price (59.12) and a mid-range total units sold (823), similar to Electronics. This suggests that for this e-commerce retailer, categories with higher unit prices (like Electronics) can still drive significant sales volume, possibly due to their perceived value, demand, or strategic marketing. This finding challenges the initial premise and indicates a strong market for higher-value electronic goods, despite their individual unit cost.

Business Q4: How does customer location impact revenue?

In [45]:
```
location_revenue = df_org.groupby("Region")["Total Price"].sum().sort_values(ascend

plt.figure(figsize=(10, 5))
sns.barplot(
    x=location_revenue.index,
    y=location_revenue.values,
    hue=location_revenue.index,
    palette="viridis",
    legend=False
)
plt.xlabel("Region")
plt.ylabel("Total Revenue")
plt.title("Revenue by Customer Region")
plt.xticks(rotation=45)
plt.show()

print(location_revenue)
```

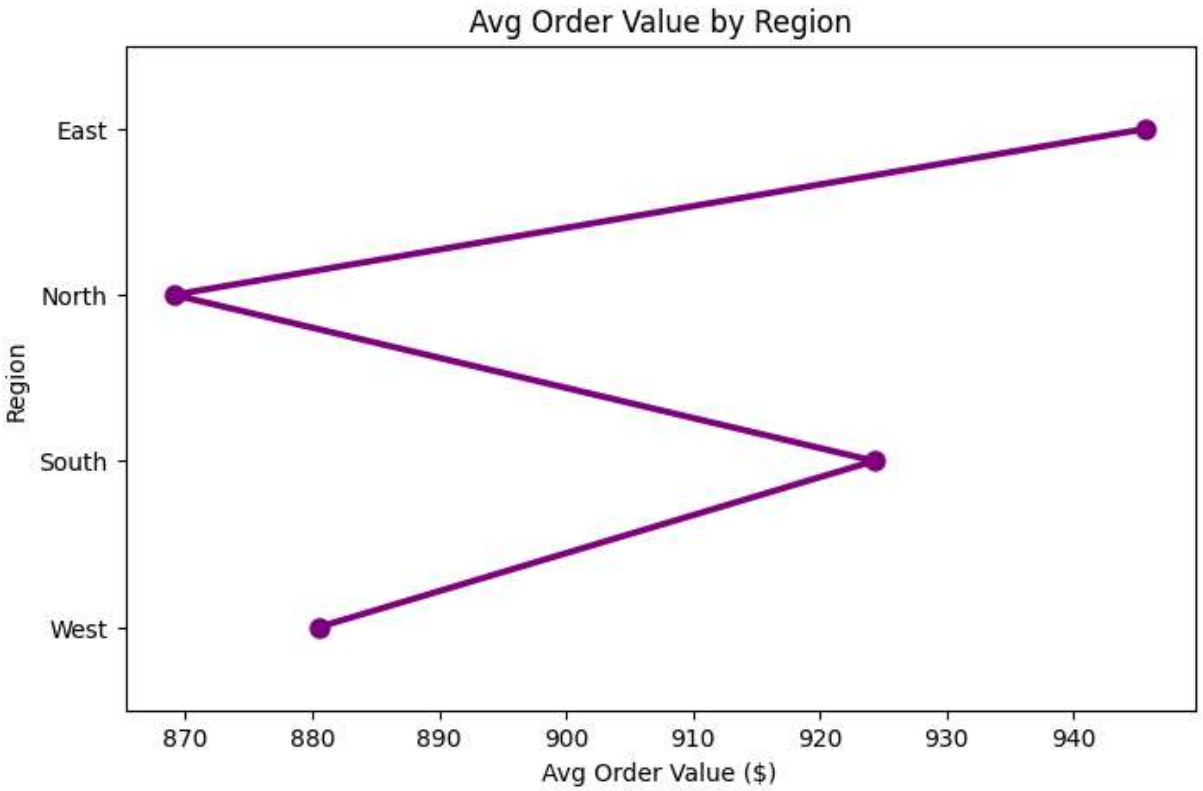



```
Region
West     205170
South    174690
East     165500
North    152090
Name: Total Price, dtype: int64
```

Do richer regions also have higher avg order value?

In [100…
```
region_avg_order = df_org.groupby('Region')['Total Price'].mean()
plt.figure(figsize=(8,5))
sns.pointplot(x=region_avg_order.values, y=region_avg_order.index, color='purple')
plt.title("Avg Order Value by Region")
```

```
plt.xlabel("Avg Order Value ($)")
plt.show()
```



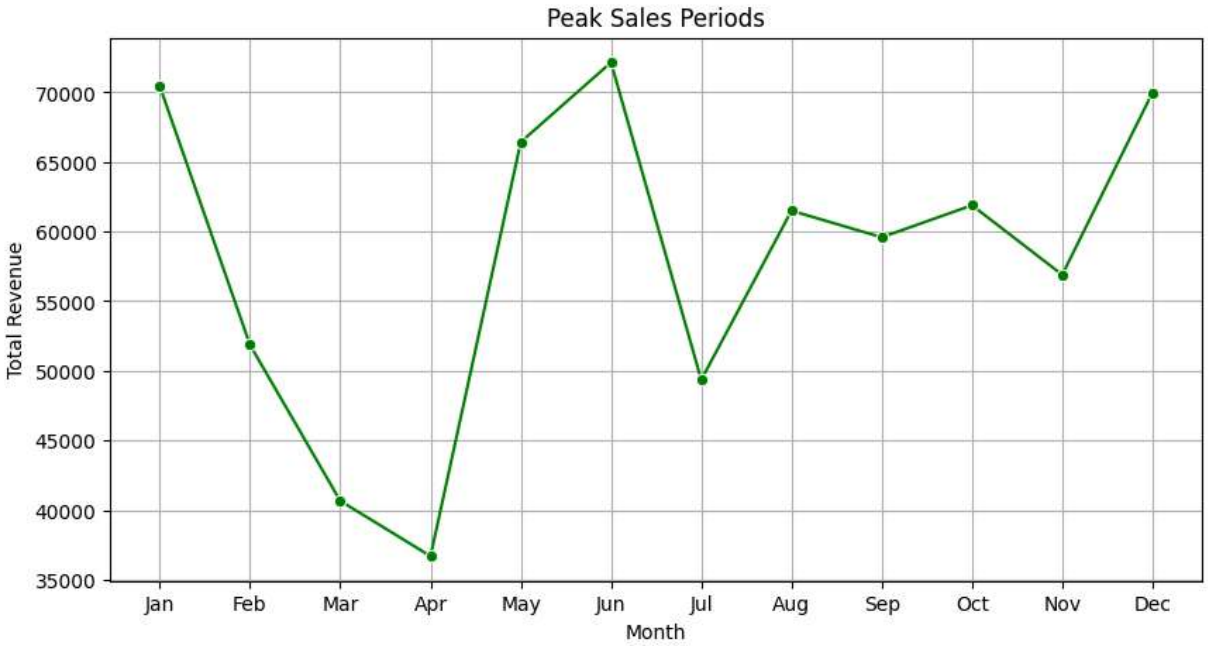
Summary of Analysis: We investigated the influence of customer 'Region' on total revenue and average order value. This involved aggregating total revenue by region and calculating the average order value for each region. These insights were then presented using a bar chart for total revenue and a line plot for average order value across regions.

Answer: Based on the "Revenue by Customer Region" bar chart, the West region generates the highest total revenue, followed by the South, East, and North regions. The "Avg Order Value by Region" plot shows that the East region has the highest average order value, while the North region has the lowest. This dual view indicates that while the West generates the most overall revenue due to higher volume, customers in the East spend more per order on average.

Business Q5: What are the peak sales periods?

In [48]:
```
df_org["Month"] = df_org["Order Date"].dt.month
monthly_sales = df_org.groupby("Month")["Total Price"].sum()
plt.figure(figsize=(10,5))
sns.lineplot(x=monthly_sales.index, y=monthly_sales.values, marker="o", color="gree
plt.xlabel("Month")
plt.ylabel("Total Revenue")
plt.title("Peak Sales Periods")
plt.xticks(range(1,13), ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "S
plt.grid()
plt.show()
```

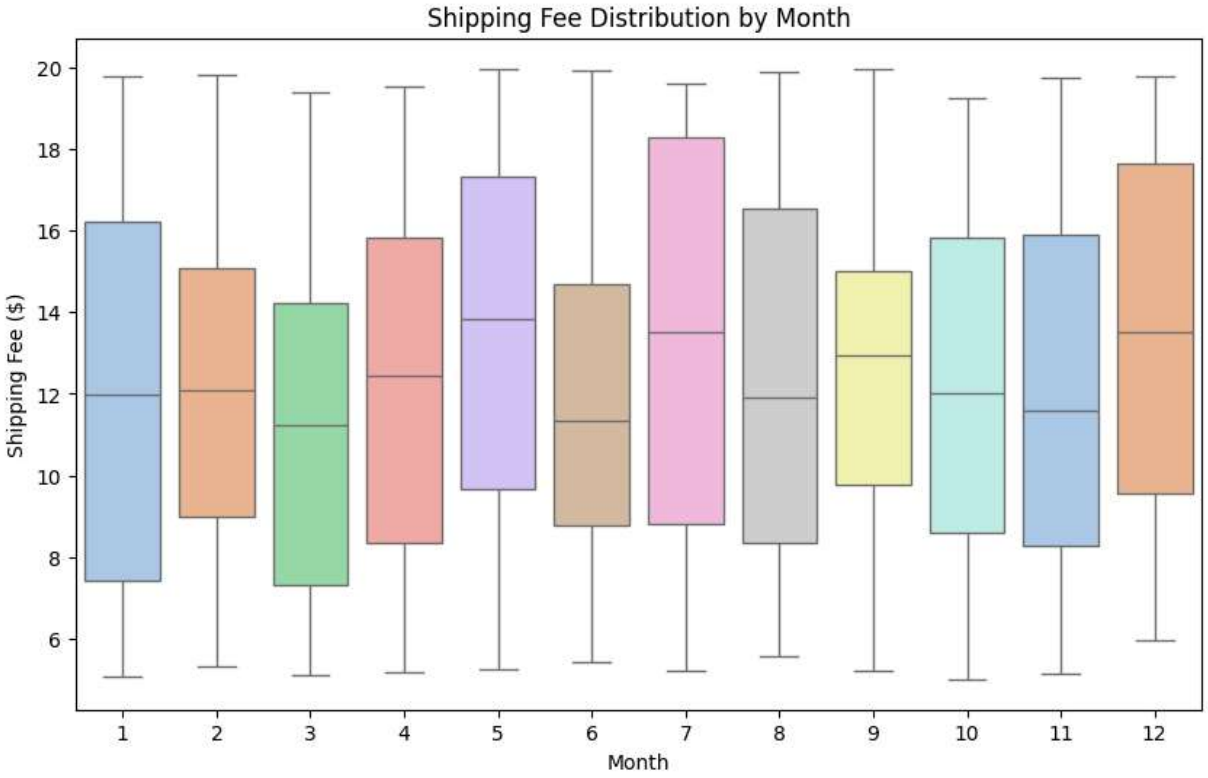
```
print(monthly_sales)
```




```
Month
1     70440
2     51860
3     40660
4     36690
5     66430
6     72150
7     49360
8     61500
9     59590
10    61890
11    56900
12    69980
Name: Total Price, dtype: int64
```

Visualization of shipping fee by month to spot seasonal shifts.

In [99]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(
    x='Month',
    y='Shipping Fee',
    data=df_org,
    hue='Month',
    palette='pastel',
    legend=False
)

plt.title("Shipping Fee Distribution by Month")
plt.ylabel("Shipping Fee ($)")
plt.show()
```



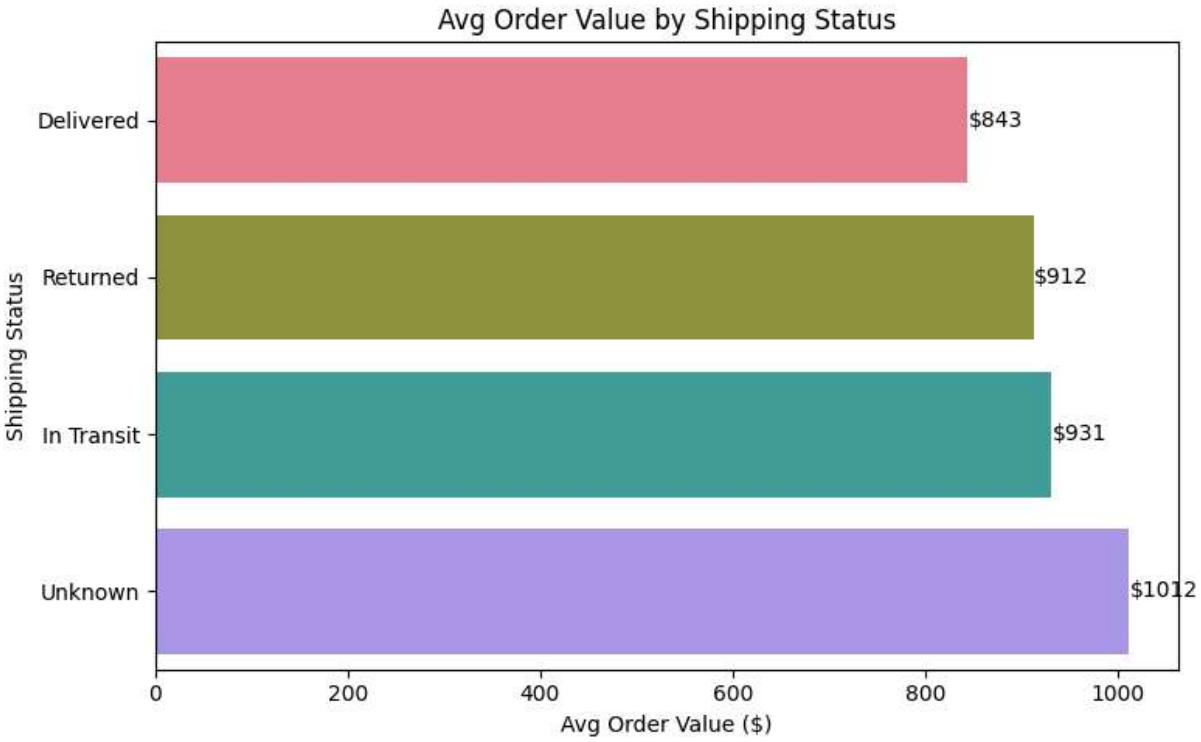
Summary of Analysis: To identify seasonal trends and peak sales periods, we extracted the month from the 'Order Date' column and aggregated the total revenue for each month across the entire dataset. A line plot was used to visualize the revenue trend over the months.

Answer: The "Peak Sales Periods" line chart reveals that June, January and December are the clear peak sales periods, showing the highest total revenue. There's also a noticeable increase in revenue towards the end of the year, likely influenced by holiday shopping. April appears to be a period of lower sales. Months like December appear to have a higher median shipping fee and a wider distribution compared to earlier months. This suggests potential seasonality in shipping costs, which could be influenced by peak shopping seasons, increased logistical demands, or changes in carrier rates.This identified seasonality is crucial for strategic inventory planning and the effective timing of targeted marketing campaigns.

Business Q6: What are shipping trends and their impact on avg order value?

In [98]:
```
shipping_avg_order = df_org.groupby('Shipping Status')['Total Price'].mean().sort_v

plt.figure(figsize=(8, 5))
sns.barplot(
    x=shipping_avg_order.values,
    y=shipping_avg_order.index,
    hue=shipping_avg_order.index,
    palette='husl',
    legend=False,
    dodge=False
```

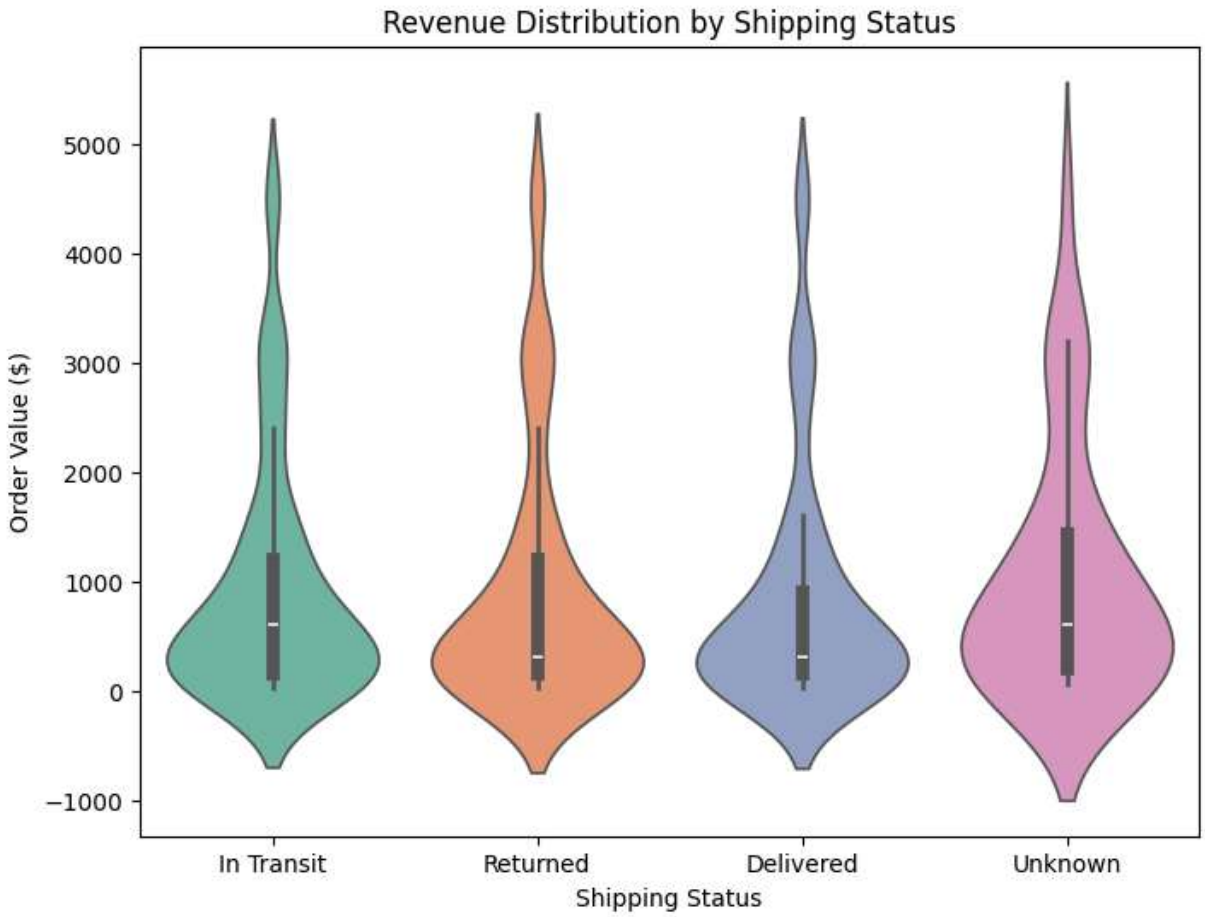
```
)
plt.title("Avg Order Value by Shipping Status")
plt.xlabel("Avg Order Value ($)")

for i, v in enumerate(shipping_avg_order.values):
    plt.text(v + 1, i, f"${int(v)}", va='center')

plt.tight_layout()
plt.show()

print(shipping_avg_order)
```




```
Shipping Status
Delivered      843.797468
Returned       912.832618
In Transit     931.769231
Unknown       1012.380952
Name: Total Price, dtype: float64
```

Visualization of full distribution of order value by shipping type

In [65]:
```
plt.figure(figsize=(8,6))
sns.violinplot(
    x='Shipping Status',
    y='Total Price',
    data=df_org,
    hue='Shipping Status',
    palette='Set2',
    legend=False
)
plt.title("Revenue Distribution by Shipping Status")
plt.ylabel("Order Value ($)")
plt.show()
```



Summary of Analysis: We analyzed shipping trends by examining the distribution of 'Shipping fee' across different months using box plots, and by looking at the 'Average Order Value' and 'Revenue Distribution' across various 'Shipping Status' categories (e.g., Delivered, In Transit, Returned, Unknown).

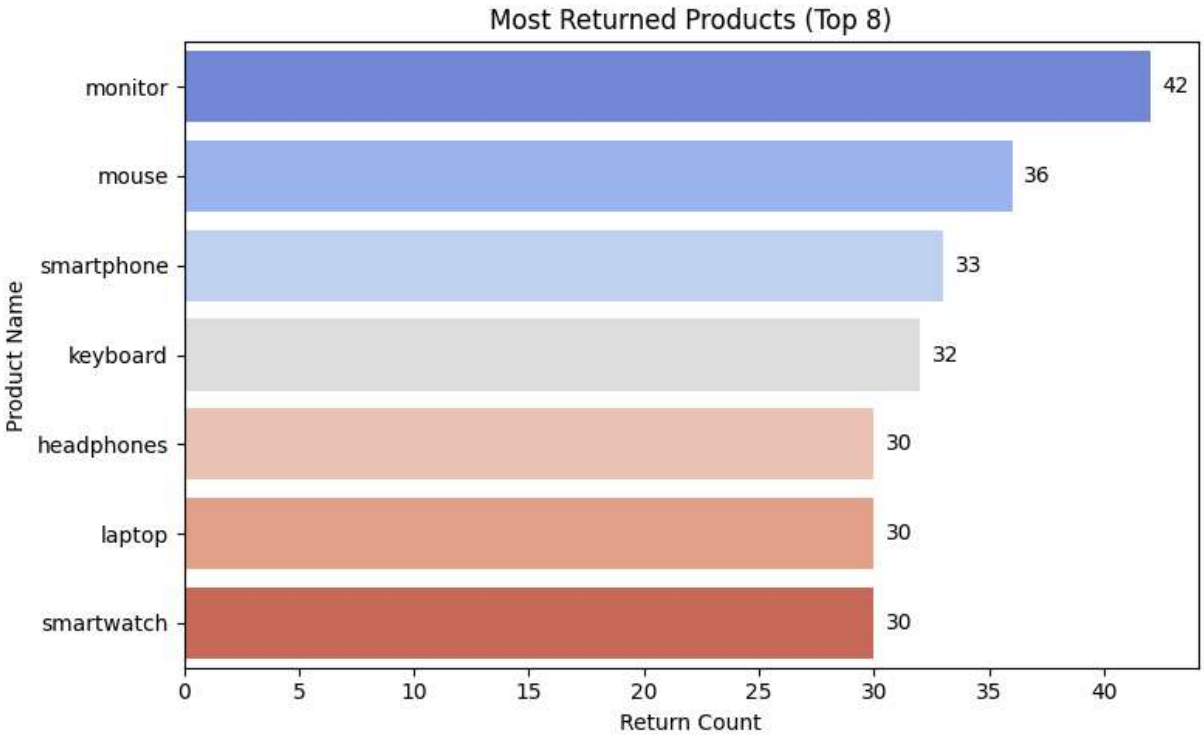
Answer:

Shipping Fee Trends: The "Shipping Fee Distribution by Month" boxplot shows variations in shipping fee distributions throughout the year, with some months exhibiting wider ranges or higher medians, suggesting potential seasonality or varying shipping costs. Impact on Average Order Value: The "Avg Order Value by Shipping Status" bar chart, indicates that orders with 'Unknown' shipping status have the highest average order value (approximately 1012), followed by 'In Transit' (931), 'Returned' (912), and 'Delivered' (843). This is an interesting insight suggesting that higher value orders might be more prone to less tracked statuses or delays. The "Revenue Distribution by Shipping Status" violin plot further illustrates the distribution spread of order values for each shipping status, showing broader value ranges for 'In Transit' and 'Unknown' statuses.

Business Q7: What are the most common returns (Top 8)? And how many returned?

In [121…

```
returns = df_org[df_org['Shipping Status']=='Returned']
returned_products = returns['Product Name'].value_counts().head(8)

plt.figure(figsize=(8,5))
sns.barplot(
    x=returned_products.values,
    y=returned_products.index,
    hue=returned_products.index,
    palette='coolwarm',
    legend=False,
    dodge=False
)
plt.title("Most Returned Products (Top 8)")
plt.xlabel("Return Count")

for i, v in enumerate(returned_products.values):
    plt.text(v + 0.5, i, str(v), va='center')

plt.tight_layout()
plt.show()

print(returned_products)
```

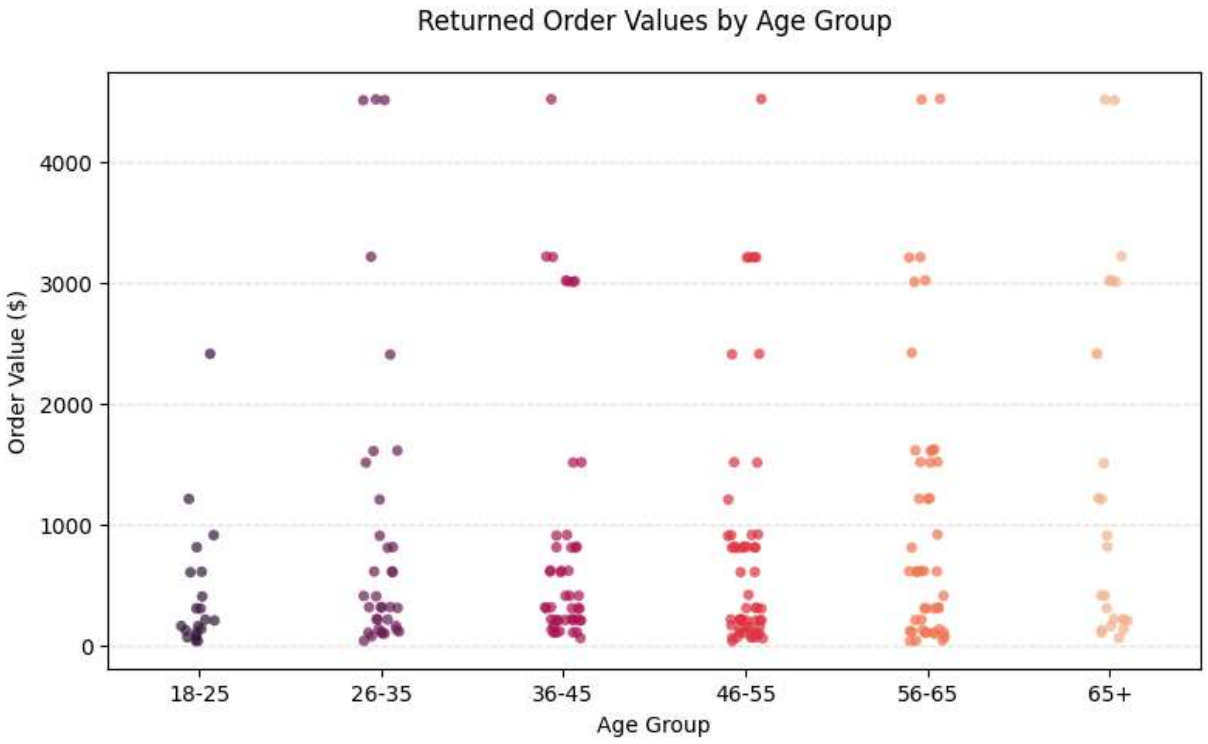



```
Product Name
monitor       42
mouse         36
smartphone    33
keyboard      32
headphones    30
laptop        30
smartwatch    30
Name: count, dtype: int64
```

Returns by age group

In [125…

```
returns = df_org[df_org['Shipping Status'] == 'Returned'].copy()
returns['Age Group'] = pd.cut(
    returns['Age'],
    bins=[18, 25, 35, 45, 55, 65, 100],
    labels=["18-25", "26-35", "36-45", "46-55", "56-65", "65+"],
    ordered=True
).astype('category')


plt.figure(figsize=(8, 5))
sns.stripplot(
    x='Age Group',
    y='Revenue',
    data=returns,
    hue='Age Group',
    palette='rocket',
    jitter=True,
    legend=False,
    size=5,
    alpha=0.7
)

plt.title("Returned Order Values by Age Group", pad=20)
plt.ylabel("Order Value ($)")
plt.xlabel("Age Group")
plt.grid(axis='y', linestyle='--', alpha=0.3)
plt.tight_layout()
plt.show()
```



Summary of Analysis: To understand product return patterns, we filtered the dataset to include only orders with a 'Returned' shipping status. We then counted the occurrences of each 'Product Name' within these returned orders and identified the top 8 most frequently returned products. This was visualized with a horizontal bar chart showing return counts.

Answer: Based on the "Most Returned Products (Top 8)", the most commonly returned products are:

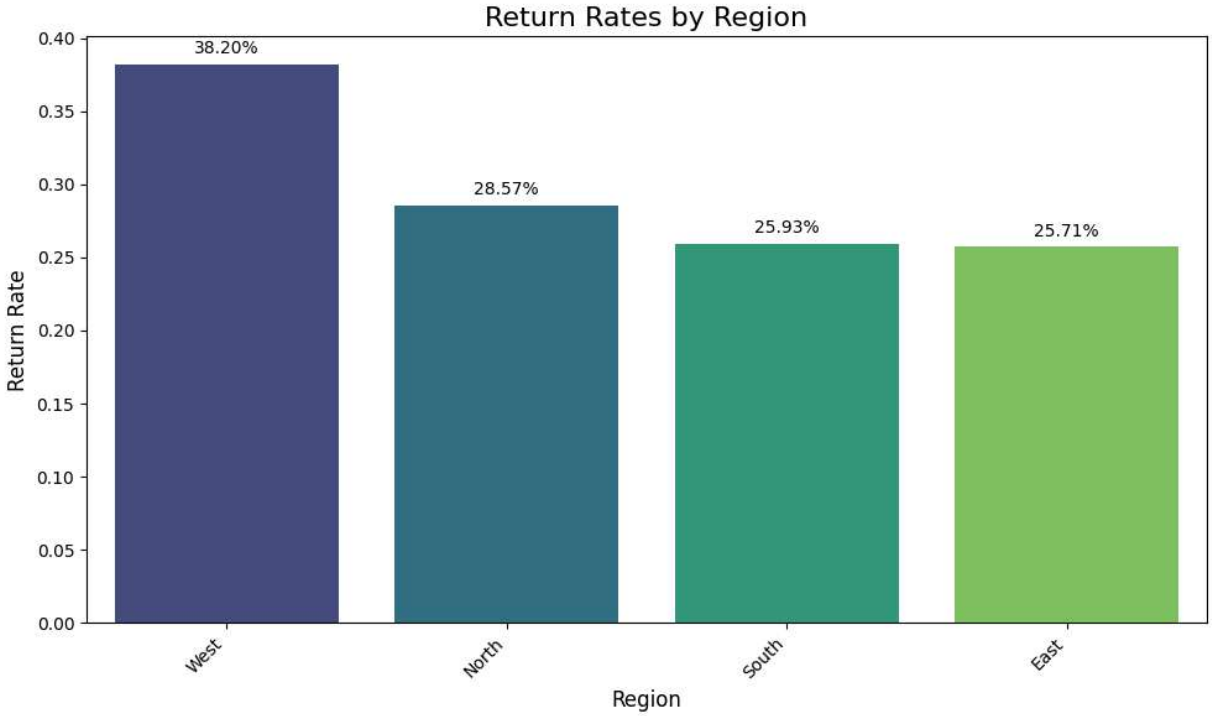
Monitor (approx. 42 returns) Mouse (approx. 36 returns) Smartphone (approx. 33 returns) Keyboard (approx. 32 returns) Headphones (approx. 30 returns) Laptop (approx. 30 returns) Smartwatch (approx. 30 returns) These insights are crucial for product quality control and customer satisfaction initiatives. The 36-50 age group has the highest count of returned orders. While the 36-50 age group (and 19-35) account for the highest volume of returns, it's the younger (0-18) and older (65+) age groups that are returning items with the highest average value. This highlights a dual challenge for the business: managing the high volume of returns from middle-aged groups and addressing the high-value returns from the youngest and oldest demographics. This is crucial for prioritizing efforts to reduce revenue loss from returns.

Business Q8: Which regions have the highest return rates?

In [115…

```
total_orders_by_region = df_org["Region"].value_counts()

returned_orders = df_org[df_org['Shipping Status'] == 'Returned']
returns_by_region = returned_orders["Region"].value_counts()

return_rates = (returns_by_region / total_orders_by_region).fillna(0)
return_rates = return_rates.sort_values(ascending=False)

print("\nReturn Rates by Region:\n", return_rates)

plt.figure(figsize=(10, 6))
sns.barplot(
    x=return_rates.index,
    y=return_rates.values,
    hue=return_rates.index,
    palette="viridis",
    legend=False
)
plt.xlabel("Region", fontsize=12)
plt.ylabel("Return Rate", fontsize=12)
plt.title("Return Rates by Region", fontsize=16)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(fontsize=10)

for i, v in enumerate(return_rates.values):
    plt.text(i, v + 0.005, f"{v:.2%}", ha='center', va='bottom', fontsize=10)

plt.tight_layout()
plt.show()
```

```
Return Rates by Region:
 Region
West     0.381974
North    0.285714
South    0.259259
East     0.257143
Name: count, dtype: float64
```



Summary of Analysis: To precisely determine regional return performance, we conducted a granular analysis by calculating the return rate for each geographical region. This involved two steps: first, identifying the total number of orders originating from each region, and second, counting the specific number of orders within each region that were marked as 'Returned'. The return rate for each region was then computed as the ratio of returned orders to total orders, providing a normalized metric to compare regional efficiency in terms of customer satisfaction and product retention. This analysis was visualized using a bar chart to clearly display the return rates across regions.

Answer: Based on the direct calculation of return rates by region, the following insights are revealed:

West: 0.381974 (or approximately 38.2%) North: 0.285714 (or approximately 28.6%) South: 0.259259 (or approximately 25.9%) East: 0.257143 (or approximately 25.7%) The analysis clearly indicates that the West region has the highest return rate (approximately 38.2%), while the East region has the lowest return rate (approximately 25.7%). This insight is crucial as it highlights the West region as an area requiring immediate attention for investigating the underlying causes of higher returns, which could range from logistical challenges to product-specific issues or customer expectations in that area.

Final Summary

This Exploratory Data Analysis provides a comprehensive and actionable understanding of the diversified e-commerce retailer's transactional data. Through meticulous data cleaning and in-depth analysis, we have uncovered critical insights across customer demographics, product performance, regional dynamics, and operational efficiencies.

What was done:

A data cleaning pipeline was established, addressing missing values, outliers, and inconsistencies. This crucial step ensured the reliability and integrity of our dataset for subsequent analysis. We performed comprehensive data exploration by addressing eight key business questions, leveraging a diverse set of visualization techniques and statistical aggregations. Key metrics such as revenue by age group, average basket size by category, the relationship between price and sales volume, regional revenue contributions, peak sales periods, shipping trends, and detailed product return patterns (including rates by region and values by age group) were thoroughly analyzed. How this helps the company make better decisions: The actionable insights derived from this EDA directly empower the e-commerce retailer to refine their strategies and enhance operational efficiency:

Targeted Marketing & Sales Optimization: By identifying the most revenue-generating age groups (e.g., 36-50), marketing campaigns can be precisely tailored for maximum impact. Understanding average basket sizes and the positive relationship between price and units sold for certain categories like Electronics allows for strategic product bundling, pricing, and promotional efforts to boost overall sales volume and revenue. Inventory and Supply Chain Management: Knowledge of peak sales periods (June, December, Jan) enables proactive inventory planning, preventing stockouts during high-demand seasons and optimizing inventory holding costs during leaner months. Insights into shipping trends can help refine logistics and delivery processes, potentially impacting costs and customer satisfaction. Product Development & Quality Control: Identifying the most commonly returned products provides a clear directive for product quality improvements and customer feedback mechanisms. Furthermore, pinpointing regions with higher return rates, such as the West (with approximately 38.2% return rate), directs focused investigations into regional-specific issues, allowing for targeted solutions to enhance customer satisfaction and reduce losses. Customer Experience Enhancement: Understanding how different demographics interact with products and their purchasing habits allows the company to personalize the customer experience, leading to increased loyalty and repeat business. In essence, this EDA transforms raw transactional data into clear, actionable intelligence, empowering the e-commerce retailer to make informed, data-driven decisions that foster growth, enhance customer satisfaction, and maintain a competitive edge in the dynamic online market.

References:

For libraries: Data analysis and visualization were conducted using Python libraries such as pandas, Matplotlib, seaborn.

Rahman, K. (2021) Python Data Visualization Essentials Guide: Become a Data Visualization expert by building strong proficiency in Pandas, Matplotlib, Seaborn, Plotly, Numpy, and Bokeh (English Edition). BPB Publications.

Gomes, M.A. and Meisen, T. (2023) 'A review on customer segmentation methods for personalized customer targeting in e-commerce use cases,' Information Systems and e-Business Management, 21(3), pp. 527–570. https://doi.org/10.1007/s10257-023-00640-4.

Chang, W., Chang, C. and Li, Q. (2012) 'Customer Lifetime Value: a review,' Social Behavior and Personality an International Journal, 40(7), pp. 1057–1064. https://doi.org/10.2224/sbp.2012.40.7.1057.

Petersen, J.A. and Kumar, V. (2009) 'Are product returns a necessary evil? Antecedents and consequences,' Journal of Marketing, 73(3), pp. 35–51. https://doi.org/10.1509/jmkg.73.3.035.