In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Setup visualization styles
sns.set(style="whitegrid")
```

Transcation dataset

In [3]:

```
# Load the transaction dataset & print 1st 10 rows
transaction = pd.read_csv('Transactions.csv')
transaction.head(10)
```

Out[3]:

	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	TotalValue	Price
0	T00001	C0199	P067	2024-08-25 12:38:23	1	300.68	300.68
1	T00112	C0146	P067	2024-05-27 22:23:54	1	300.68	300.68
2	T00166	C0127	P067	2024-04-25 07:38:55	1	300.68	300.68
3	T00272	C0087	P067	2024-03-26 22:55:37	2	601.36	300.68
4	T00363	C0070	P067	2024-03-21 15:10:10	3	902.04	300.68
5	T00442	C0188	P067	2024-12-26 14:40:03	1	300.68	300.68
6	T00490	C0195	P067	2024-11-24 11:49:48	3	902.04	300.68
7	T00536	C0008	P067	2024-09-22 06:13:59	1	300.68	300.68
8	T00564	C0157	P067	2024-12-07 17:57:40	3	902.04	300.68
9	T00631	C0130	P067	2024-05-14 23:14:59	2	601.36	300.68

In [4]:

```
# check the no.of missing value
transaction.isnull().sum()
```

Out[4]:

```
TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64
```

In [5]:

```
# check the datatype of the each data column
transaction.dtypes
```

Out[5]:

```
TransactionID       object
CustomerID          object
ProductID           object
TransactionDate     object
Quantity             int64
TotalValue         float64
Price              float64
dtype: object
```

In [6]:

```
# describe the data
transaction.describe()
```

Out[6]:

	Quantity	TotalValue	Price
count	1000.000000	1000.000000	1000.00000
mean	2.537000	689.995560	272.55407
std	1.117981	493.144478	140.73639
min	1.000000	16.080000	16.08000
25%	2.000000	295.295000	147.95000
50%	3.000000	588.880000	299.93000
75%	4.000000	1011.660000	404.40000
max	4.000000	1991.040000	497.76000

In [7]:

```
transaction.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   TransactionID    1000 non-null   object 
 1   CustomerID       1000 non-null   object 
 2   ProductID        1000 non-null   object 
 3   TransactionDate  1000 non-null   object 
 4   Quantity         1000 non-null   int64  
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
```

product dataset

In [8]:

```
# Load the transaction dataset & print 1st 10 rows
product = pd.read_csv('Products.csv')
product.head(10)
```

Out[8]:

	ProductID	ProductName	Category	Price
0	P001	ActiveWear Biography	Books	169.30
1	P002	ActiveWear Smartwatch	Electronics	346.30
2	P003	ComfortLiving Biography	Books	44.12
3	P004	BookWorld Rug	Home Decor	95.69
4	P005	TechPro T-Shirt	Clothing	429.31
5	P006	ActiveWear Rug	Home Decor	121.32
6	P007	SoundWave Cookbook	Books	420.15
7	P008	BookWorld Bluetooth Speaker	Electronics	146.85
8	P009	BookWorld Wall Art	Home Decor	325.01
9	P010	ComfortLiving Smartwatch	Electronics	350.13

In [9]:

```
# check null values
product.isnull().sum()
```

Out[9]:

```
ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64
```

In [10]:

```
# check datatypes
product.dtypes
```

Out[10]:

```
ProductID       object
ProductName     object
Category        object
Price          float64
dtype: object
```

In [11]:

```
product.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   ProductID    100 non-null    object 
 1   ProductName  100 non-null    object 
 2   Category     100 non-null    object 
 3   Price        100 non-null    float64
dtypes: float64(1), object(3)
memory usage: 3.3+ KB
```

In [12]:

```
product.describe()
```

Out[12]:

	Price
count	100.000000
mean	267.551700
std	143.219383
min	16.080000
25%	147.767500
50%	292.875000
75%	397.090000
max	497.760000

Customer Dataset

In [13]:

```
customer = pd.read_csv('Customers.csv')
customer.head(10)
```

Out[13]:

	CustomerID	CustomerName	Region	SignupDate
0	C0001	Lawrence Carroll	South America	2022-07-10
1	C0002	Elizabeth Lutz	Asia	2022-02-13
2	C0003	Michael Rivera	South America	2024-03-07
3	C0004	Kathleen Rodriguez	South America	2022-10-09
4	C0005	Laura Weber	Asia	2022-08-15
5	C0006	Brittany Palmer	South America	2024-01-07
6	C0007	Paul Graves	Asia	2022-06-18
7	C0008	David Li	North America	2024-01-13
8	C0009	Joy Clark	Europe	2023-08-14
9	C0010	Aaron Cox	Europe	2022-12-15

In [14]:

```
customer.isnull().sum()
```

Out[14]:

```
CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64
```

In [15]:

```
customer.dtypes
```

Out[15]:

```
CustomerID      object
CustomerName    object
Region          object
SignupDate      object
dtype: object
```

In [16]:

```
customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   CustomerID    200 non-null    object
 1   CustomerName  200 non-null    object
 2   Region        200 non-null    object
 3   SignupDate    200 non-null    object
dtypes: object(4)
memory usage: 6.4+ KB
```

In [17]:

```
customer.describe()
```

Out[17]:

	CustomerID	CustomerName	Region	SignupDate
count	200	200	200	200
unique	200	200	4	179
top	C0001	Lawrence Carroll	South America	2024-11-11
freq	1	1	59	3

Merge The Dataframes

In [18]:

```
# Merge the dataframes
# Merge transactions with customers on 'CustomerID'
merged_data = pd.merge(transaction, customer, on='CustomerID', how='left')
```

In [19]:

```
# Merge the above result with products on 'ProductID'
merged_data = pd.merge(merged_data, product, on='ProductID', how='left')
```

In [20]:

```
# Check the first 10 rows of the merged data
print(merged_data.head(10))
```

```
  TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0        T00001      C0199      P067  2024-08-25 12:38:23         1   
1        T00112      C0146      P067  2024-05-27 22:23:54         1   
2        T00166      C0127      P067  2024-04-25 07:38:55         1   
3        T00272      C0087      P067  2024-03-26 22:55:37         2   
4        T00363      C0070      P067  2024-03-21 15:10:10         3   
5        T00442      C0188      P067  2024-12-26 14:40:03         1   
6        T00490      C0195      P067  2024-11-24 11:49:48         3   
7        T00536      C0008      P067  2024-09-22 06:13:59         1   
8        T00564      C0157      P067  2024-12-07 17:57:40         3   
9        T00631      C0130      P067  2024-05-14 23:14:59         2   

   TotalValue  Price_x       CustomerName         Region  SignupDate  \
0      300.68   300.68     Andrea Jenkins         Europe  2022-12-03   
1      300.68   300.68    Brittany Harvey           Asia  2024-09-04   
2      300.68   300.68    Kathryn Stevens         Europe  2024-04-04   
3      601.36   300.68    Travis Campbell  South America  2024-04-11   
4      902.04   300.68      Timothy Perez         Europe  2022-03-15   
5      300.68   300.68          Anna Ball  South America  2022-05-17   
6      902.04   300.68  Jeremy Mclaughlin  South America  2024-09-17   
7      300.68   300.68           David Li  North America  2024-01-13   
8      902.04   300.68        Miguel Wong  North America  2024-01-30   
9      601.36   300.68      Robert Jones  South America  2023-04-19   

                       ProductName     Category  Price_y  
0  ComfortLiving Bluetooth Speaker  Electronics   300.68  
1  ComfortLiving Bluetooth Speaker  Electronics   300.68  
2  ComfortLiving Bluetooth Speaker  Electronics   300.68  
3  ComfortLiving Bluetooth Speaker  Electronics   300.68  
4  ComfortLiving Bluetooth Speaker  Electronics   300.68  
5  ComfortLiving Bluetooth Speaker  Electronics   300.68  
6  ComfortLiving Bluetooth Speaker  Electronics   300.68  
7  ComfortLiving Bluetooth Speaker  Electronics   300.68  
8  ComfortLiving Bluetooth Speaker  Electronics   300.68  
9  ComfortLiving Bluetooth Speaker  Electronics   300.68  
```

In [21]:

```
# Save the merged dataset for further analysis
merged_data.to_csv('Consolidated_eCommerce_Data.csv', index=False)
```

Loading Saved Dataset (Merged Data Set)

In [22]:

```
# Load the consolidated dataset (assuming it's already merged)
merged_data = pd.read_csv('Consolidated_eCommerce_Data.csv')
```

In [23]:

```
# Print 1st 10 rows
merged_data.head(10)
```

Out[23]:

	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	TotalValue	Price_x	CustomerName	Region	SignupDate	ProductName
0	T00001	C0199	P067	2024-08-25 12:38:23	1	300.68	300.68	Andrea Jenkins	Europe	2022-12-03	ComfortLiving Bluetooth Speaker
1	T00112	C0146	P067	2024-05-27 22:23:54	1	300.68	300.68	Brittany Harvey	Asia	2024-09-04	ComfortLiving Bluetooth Speaker
2	T00166	C0127	P067	2024-04-25 07:38:55	1	300.68	300.68	Kathryn Stevens	Europe	2024-04-04	ComfortLiving Bluetooth Speaker
3	T00272	C0087	P067	2024-03-26 22:55:37	2	601.36	300.68	Travis Campbell	South America	2024-04-11	ComfortLiving Bluetooth Speaker
4	T00363	C0070	P067	2024-03-21 15:10:10	3	902.04	300.68	Timothy Perez	Europe	2022-03-15	ComfortLiving Bluetooth Speaker
5	T00442	C0188	P067	2024-12-26 14:40:03	1	300.68	300.68	Anna Ball	South America	2022-05-17	ComfortLiving Bluetooth Speaker
6	T00490	C0195	P067	2024-11-24 11:49:48	3	902.04	300.68	Jeremy Mclaughlin	South America	2024-09-17	ComfortLiving Bluetooth Speaker
7	T00536	C0008	P067	2024-09-22 06:13:59	1	300.68	300.68	David Li	North America	2024-01-13	ComfortLiving Bluetooth Speaker
8	T00564	C0157	P067	2024-12-07 17:57:40	3	902.04	300.68	Miguel Wong	North America	2024-01-30	ComfortLiving Bluetooth Speaker
9	T00631	C0130	P067	2024-05-14 23:14:59	2	601.36	300.68	Robert Jones	South America	2023-04-19	ComfortLiving Bluetooth Speaker

Descriptive Statistics

In [24]:

```python
# 1. Total Sales (Total Value of All Transactions)
total_sales = merged_data['TotalValue'].sum()
print(f"Total Sales: ${total_sales:,.2f}")
```

```
Total Sales: $689,995.56
```

In [25]:

```python
# 2. Average Transaction Value (Average of the 'TotalValue' column)
avg_transaction_value = merged_data['TotalValue'].mean()
print(f"Average Transaction Value: ${avg_transaction_value:,.2f}")
```

```
Average Transaction Value: $690.00
```

In [26]:

```python
# 3. Most Sold Products (By Quantity)
most_sold_products = merged_data.groupby('ProductName')
['Quantity'].sum().sort_values(ascending=False).head(10)
print("\nMost Sold Products by Quantity:")
print(most_sold_products)
```

```
Most Sold Products by Quantity:
ProductName
ActiveWear Smartwatch    100
SoundWave Headphones      97
HomeSense Desk Lamp       81
ActiveWear Rug            79
SoundWave Cookbook        78
ActiveWear Jacket         76
BookWorld Biography       71
TechPro T-Shirt           66
SoundWave Desk Lamp       64
TechPro Textbook          62
Name: Quantity, dtype: int64
```

In [27]:

```python
# 4. Highest Revenue-Generating Products (By Total Sales)
highest_revenue_products = merged_data.groupby('ProductName')
['TotalValue'].sum().sort_values(ascending=False).head(10)
print("\nHighest Revenue-Generating Products:")
print(highest_revenue_products)
```

```
Highest Revenue-Generating Products:
ProductName
ActiveWear Smartwatch      39096.97
SoundWave Headphones       25211.64
SoundWave Novel            24507.90
ActiveWear Jacket          22712.56
ActiveWear Rug             22314.43
TechPro Headphones         19513.80
BookWorld Cookbook         19221.99
BookWorld Sweater          18743.79
TechPro Textbook           18267.96
ActiveWear Cookware Set    18083.73
Name: TotalValue, dtype: float64
```

In [28]:

```python
# 5. Total Number of Unique Products Sold
unique_products_sold = merged_data['ProductID'].nunique()
print(f"\nTotal Unique Products Sold: {unique_products_sold}")
```

```
Total Unique Products Sold: 100
```

In [29]:

```python
# 6. Total Number of Unique Customers
unique_customers = merged_data['CustomerID'].nunique()
print(f"Total Unique Customers: {unique_customers}")
```

```
Total Unique Customers: 199
```

In [30]:

```python
# 7. Total Quantity Sold Across All Transactions
total_quantity_sold = merged_data['Quantity'].sum()
print(f"\nTotal Quantity Sold: {total_quantity_sold}")
```

```
Total Quantity Sold: 2537
```

In [31]:

```python
# 8. Total Revenue by Region
total_revenue_by_region = merged_data.groupby('Region')
['TotalValue'].sum().sort_values(ascending=False)
print("\nTotal Revenue by Region:")
print(total_revenue_by_region)
```

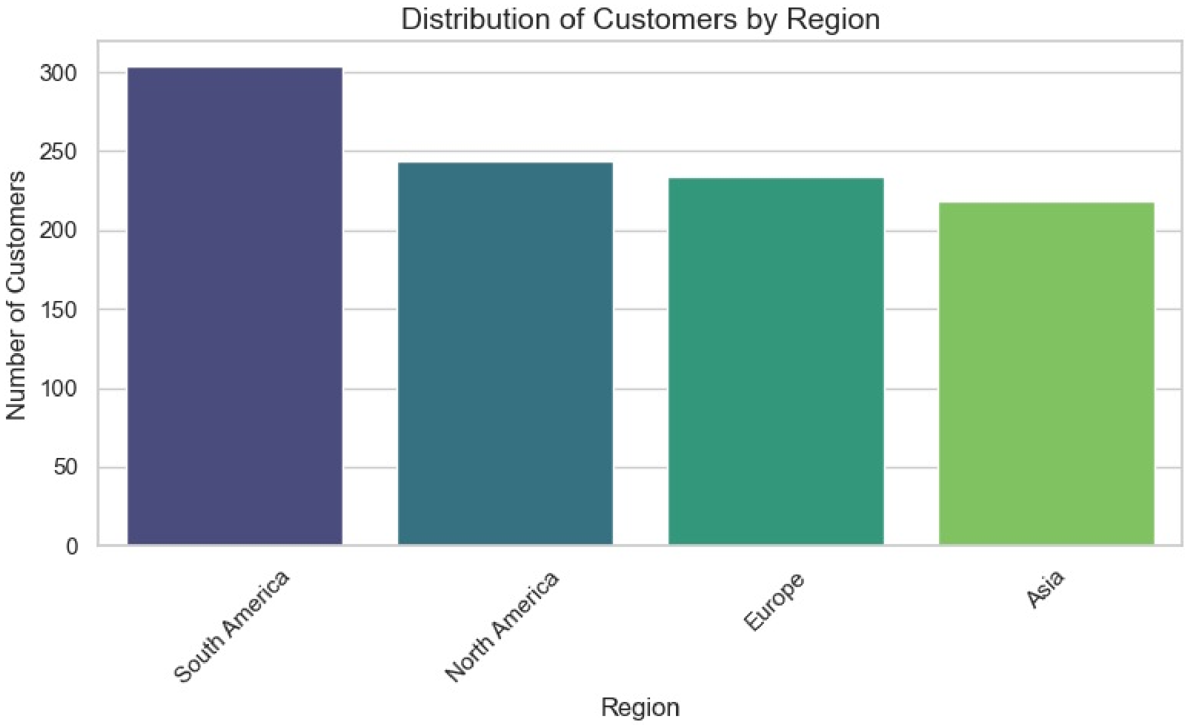
```
Total Revenue by Region:
Region
South America    219352.56
Europe           166254.63
North America    152313.40
Asia             152074.97
Name: TotalValue, dtype: float64
```

Data Visualization

1. Distribution of Customers by Region

In [32]:

```python
region_counts = merged_data['Region'].value_counts()
plt.figure(figsize=(8, 5))
sns.barplot(x=region_counts.index, y=region_counts.values, palette='viridis')
plt.title('Distribution of Customers by Region', fontsize=14)
plt.xlabel('Region', fontsize=12)
plt.ylabel('Number of Customers', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Business Insights

Customer Distribution by Region:

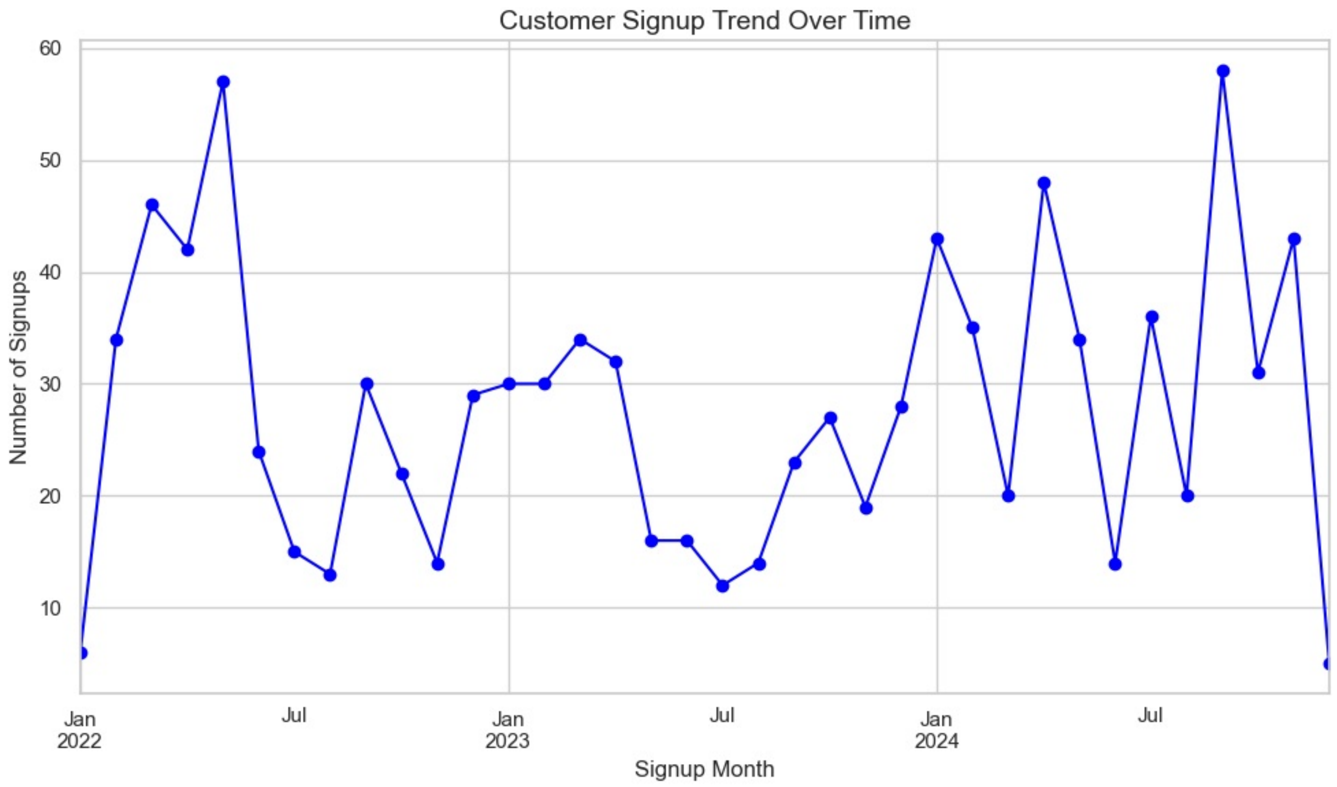
The majority of customers come from south America and North America, while customers from Asia and Europe are less represented.

This insight could suggest that marketing efforts should focus more on the underrepresented regions to expand the customer base.

2. Signup Date Distribution

In [33]:

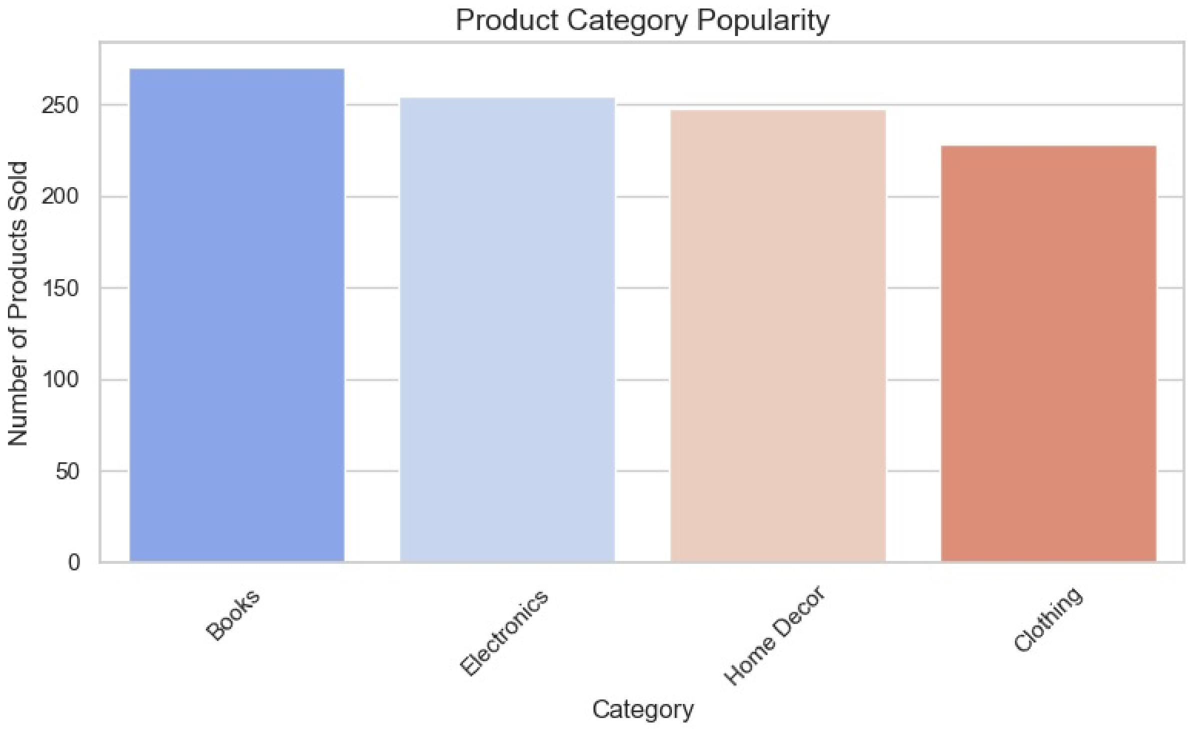
```
merged_data['SignupDate'] = pd.to_datetime(merged_data['SignupDate'])
signup_counts = merged_data['SignupDate'].dt.to_period('M').value_counts().sort_index()
plt.figure(figsize=(10, 6))
signup_counts.plot(kind='line', marker='o', color='blue')
plt.title('Customer Signup Trend Over Time', fontsize=14)
plt.xlabel('Signup Month', fontsize=12)
plt.ylabel('Number of Signups', fontsize=12)
plt.tight_layout()
plt.show()
```



3. Product Category Popularity

In [34]:

```
category_counts = merged_data['Category'].value_counts()
plt.figure(figsize=(8, 5))
sns.barplot(x=category_counts.index, y=category_counts.values, palette='coolwarm')
plt.title('Product Category Popularity', fontsize=14)
plt.xlabel('Category', fontsize=12)
plt.ylabel('Number of Products Sold', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Business insight

Product category popularity

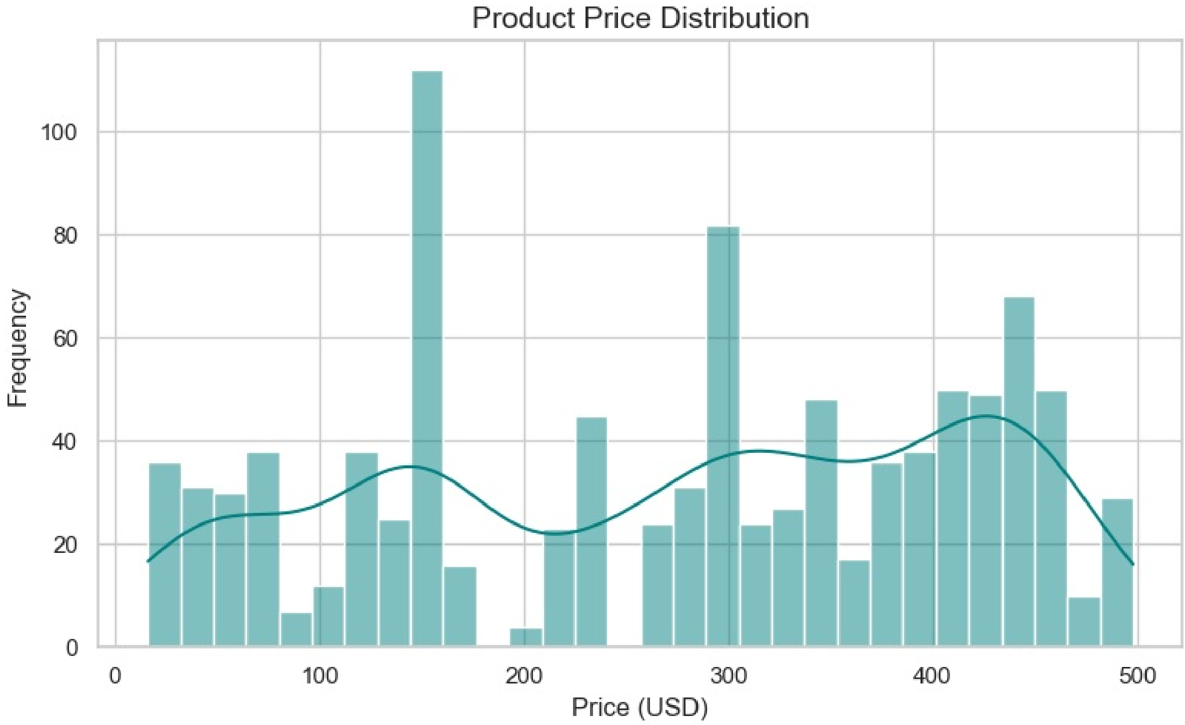
the top products are soldout is books, sold more than 250 products and 2nd top is electronics which sold out 250 while home decor and clothing are less popular

4. Product Price Distribution

In [36]:

```
plt.figure(figsize=(8, 5))
sns.histplot(merged_data['Price_x'], bins=30, kde=True, color='teal')
plt.title('Product Price Distribution', fontsize=14)
plt.xlabel('Price (USD)', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.tight_layout()
plt.show()
```

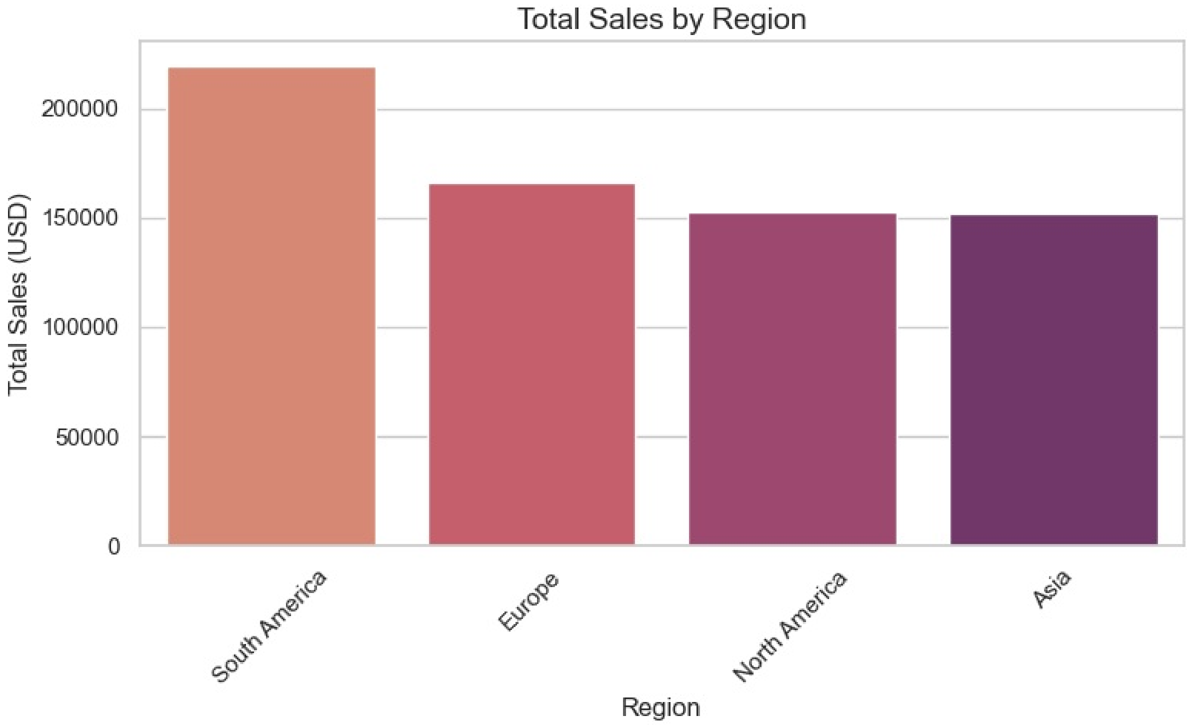
```
C:\Users\kotesh\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is de
precated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



5. Total Sales by Region

In [37]:

```
sales_by_region = merged_data.groupby('Region')
['TotalValue'].sum().sort_values(ascending=False)
plt.figure(figsize=(8, 5))
sns.barplot(x=sales_by_region.index, y=sales_by_region.values, palette='flare')
plt.title('Total Sales by Region', fontsize=14)
plt.xlabel('Region', fontsize=12)
plt.ylabel('Total Sales (USD)', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Business Insight

Total sales by Region:

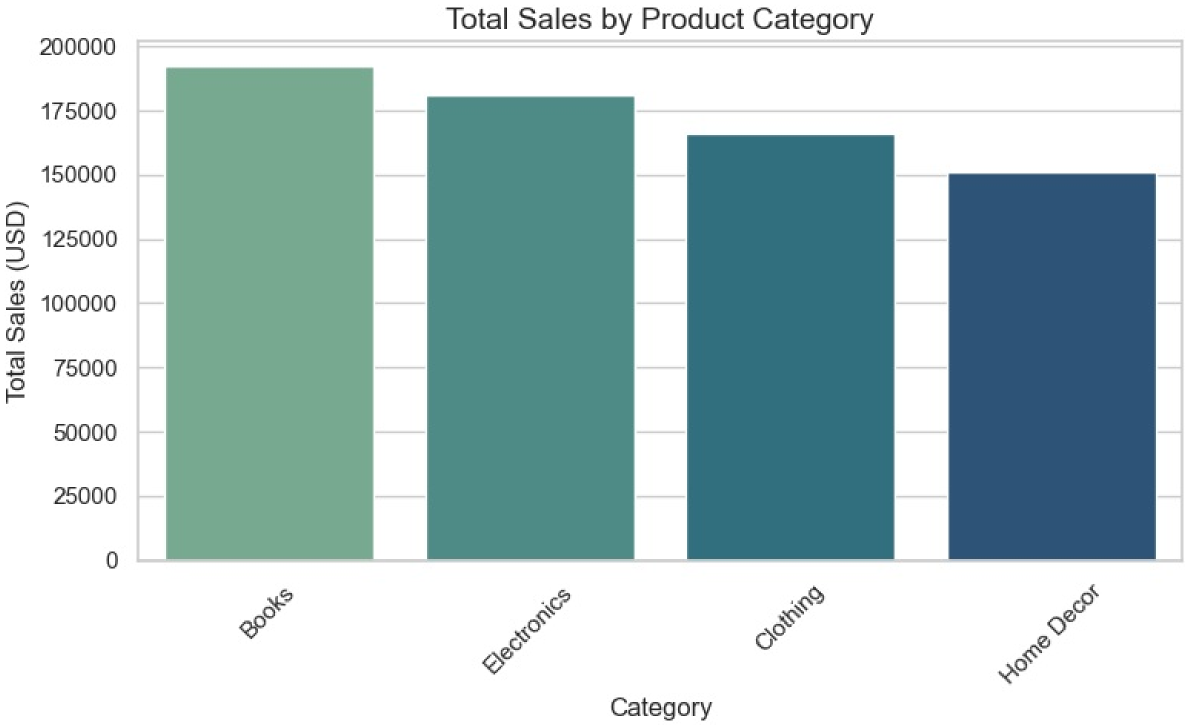
The majority of sales from south America and Europe, while sales from Asia and North America are less represented.

This insight could suggest that marketing efforts should focus more on the underrepresented regions to increase our sales.

6. Total Sales by Product Category

In [38]:

```
sales_by_category = merged_data.groupby('Category')
['TotalValue'].sum().sort_values(ascending=False)
plt.figure(figsize=(8, 5))
sns.barplot(x=sales_by_category.index, y=sales_by_category.values, palette='crest')
plt.title('Total Sales by Product Category', fontsize=14)
plt.xlabel('Category', fontsize=12)
plt.ylabel('Total Sales (USD)', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Business Insight

Top 4 Product Categories by Revenue:

Books and Electronics are the top-performing categories, contributing to highest of total sales. This indicates that the business should invest more in marketing for these categories, while other categories like Home Decor & clothing also performing better.

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js