In [1]:

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [10]:

```
#Loading the dataset and Displaying the first few rows of the datafram
df=pd.read_csv(r"C:\Users\madin\Downloads\Global-Superstore.csv")

df.head()
```

Out[10]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	
0	32298	CA-2012-124891	7/31/2012	7/31/2012	Same Day	RH-19495	Rick Hansen	Consumer	Ne
1	26341	IN-2013-77878	2/5/2013	2/7/2013	Second Class	JR-16210	Justin Ritter	Corporate	Wollo
2	25330	IN-2013-71249	10/17/2013	10/18/2013	First Class	CR-12730	Craig Reiter	Consumer	Br
3	13524	ES-2013-1579342	1/28/2013	1/30/2013	First Class	KM-16375	Katherine Murray	Home Office	
4	47221	SG-2013-4320	11/5/2013	11/6/2013	Same Day	RH-9495	Rick Hansen	Consumer	

5 rows × 24 columns

In [11]:

```
#Displaying the basic information about the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Row ID          51290 non-null  int64  
 1   Order ID        51290 non-null  object 
 2   Order Date      51290 non-null  object 
 3   Ship Date       51290 non-null  object 
 4   Ship Mode       51290 non-null  object 
 5   Customer ID     51290 non-null  object 
 6   Customer Name   51290 non-null  object 
 7   Segment         51290 non-null  object 
 8   City            51290 non-null  object 
 9   State           51290 non-null  object 
 10  Country         51290 non-null  object 
 11  Postal Code     9994 non-null   float64
 12  Market          51290 non-null  object 
 13  Region          51290 non-null  object 
 14  Product ID      51290 non-null  object 
 15  Category        51290 non-null  object 
 16  Sub-Category    51290 non-null  object 
 17  Product Name    51290 non-null  object 
 18  Sales           51290 non-null  float64
 19  Quantity        51290 non-null  int64  
 20  Discount        51290 non-null  float64
 21  Profit          51290 non-null  float64
 22  Shipping Cost   51290 non-null  float64
 23  Order Priority  51290 non-null  object 
dtypes: float64(5), int64(2), object(17)
memory usage: 9.4+ MB
```

In [26]:

```python
# Displaying the basic information about the dataset
df.describe()
```

Out[26]:

	Row ID	Order Date	Postal Code	Sales	Quantity	Disc
count	51290.00000	51290	9994.000000	51290.000000	51290.000000	51290.00
mean	25645.50000	2013-05-11 21:26:49.155781120	55190.379428	246.490581	3.476545	0.14
min	1.00000	2011-01-01 00:00:00	1040.000000	0.444000	1.000000	0.00
25%	12823.25000	2012-06-19 00:00:00	23223.000000	30.758625	2.000000	0.00
50%	25645.50000	2013-07-08 00:00:00	56430.500000	85.053000	3.000000	0.00
75%	38467.75000	2014-05-22 00:00:00	90008.000000	251.053200	5.000000	0.20
max	51290.00000	2014-12-31 00:00:00	99301.000000	22638.480000	14.000000	0.85
std	14806.29199	NaN	32063.693350	487.565361	2.278766	0.21

In [13]:

```
# Checking for missing values
df.isnull().sum()
```

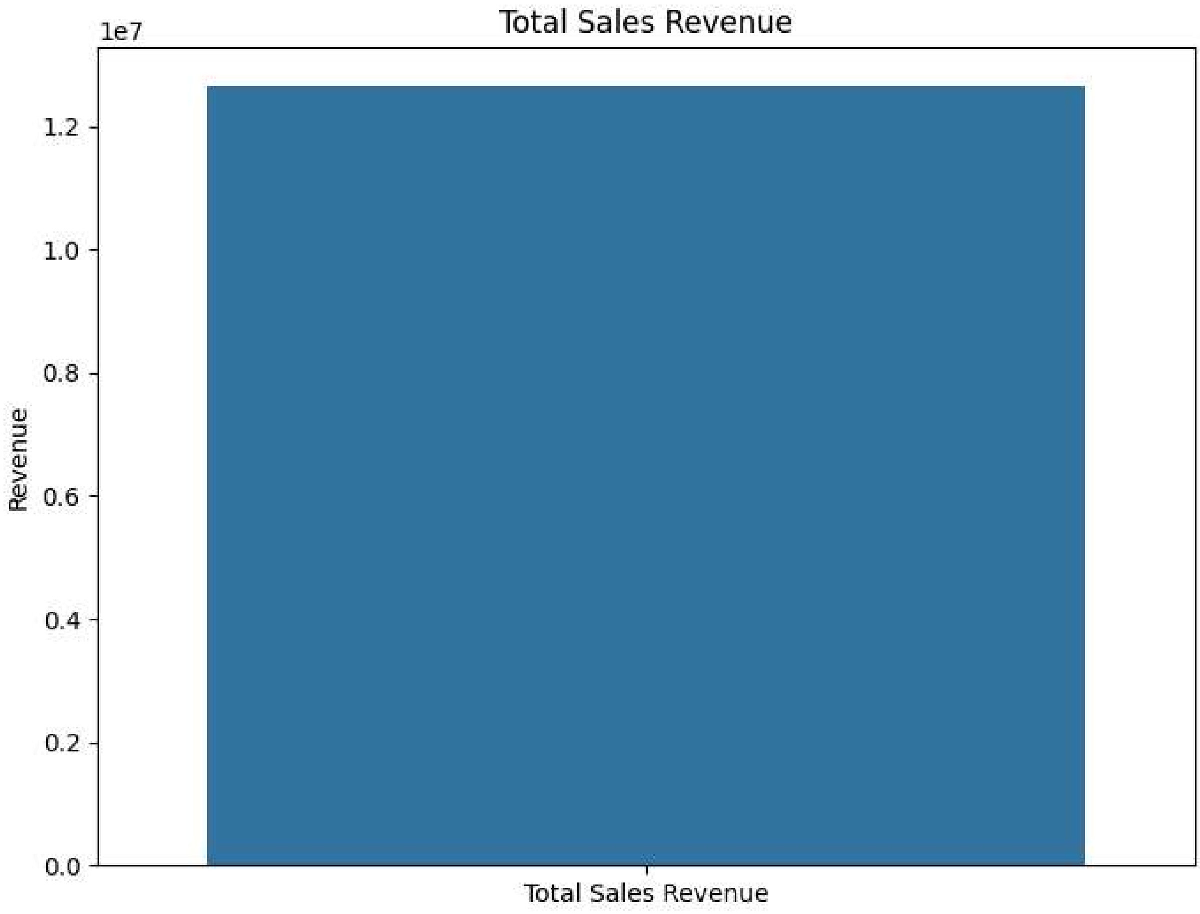
Out[13]:

```
Row ID                0
Order ID              0
Order Date            0
Ship Date             0
Ship Mode             0
Customer ID           0
Customer Name         0
Segment               0
City                  0
State                 0
Country               0
Postal Code       41296
Market                0
Region                0
Product ID            0
Category              0
Sub-Category          0
Product Name          0
Sales                 0
Quantity              0
Discount              0
Profit                0
Shipping Cost         0
Order Priority        0
dtype: int64
```

In [14]:
```python
# Exploring unique values in categorical columns
df['Category'].unique()
df['Sub-Category'].unique()
df['Region'].unique()
```

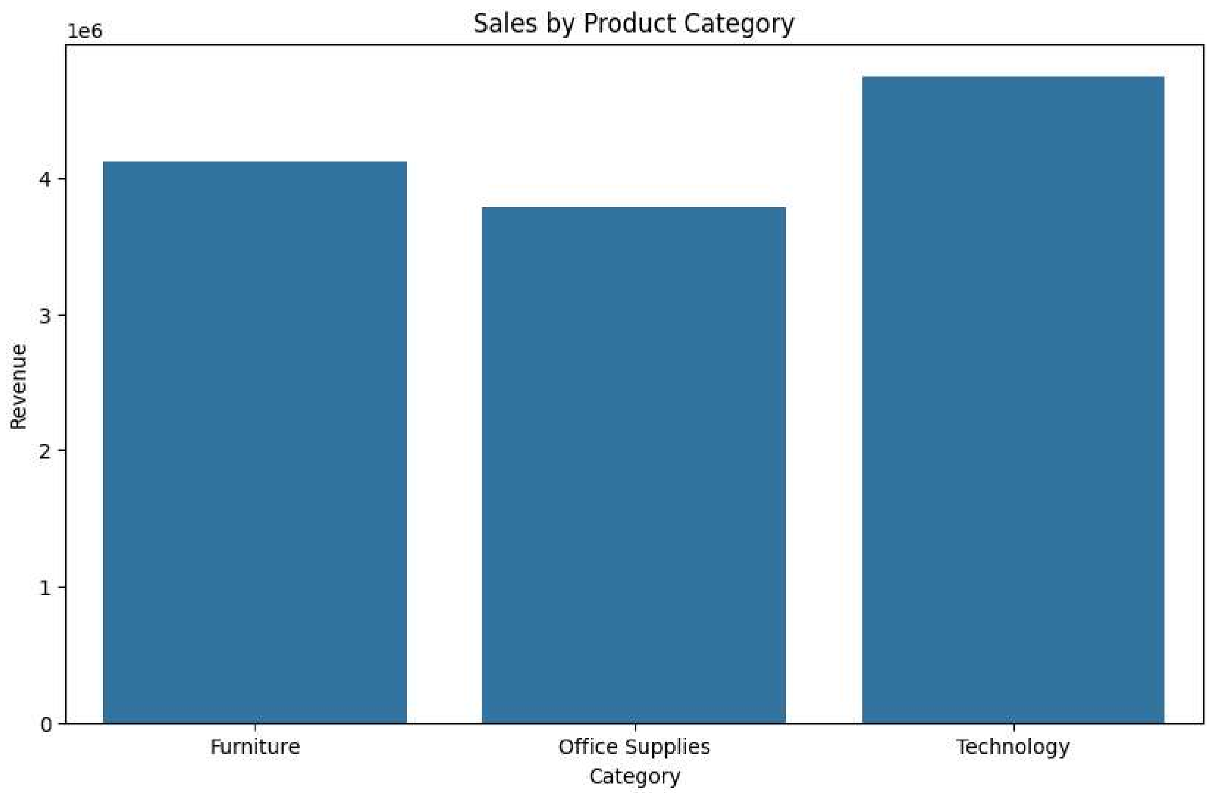
Out[14]:
```
array(['East', 'Oceania', 'Central', 'Africa', 'West', 'South',
       'Central Asia', 'EMEA', 'North Asia', 'North', 'Caribbean',
       'Southeast Asia', 'Canada'], dtype=object)
```

In [27]:
```python
# Total Sales Revenue
total_sales = df['Sales'].sum()

plt.figure(figsize=(8, 6))
sns.barplot(x=['Total Sales Revenue'], y=[total_sales])
plt.title('Total Sales Revenue')
plt.ylabel('Revenue')
plt.show()
```



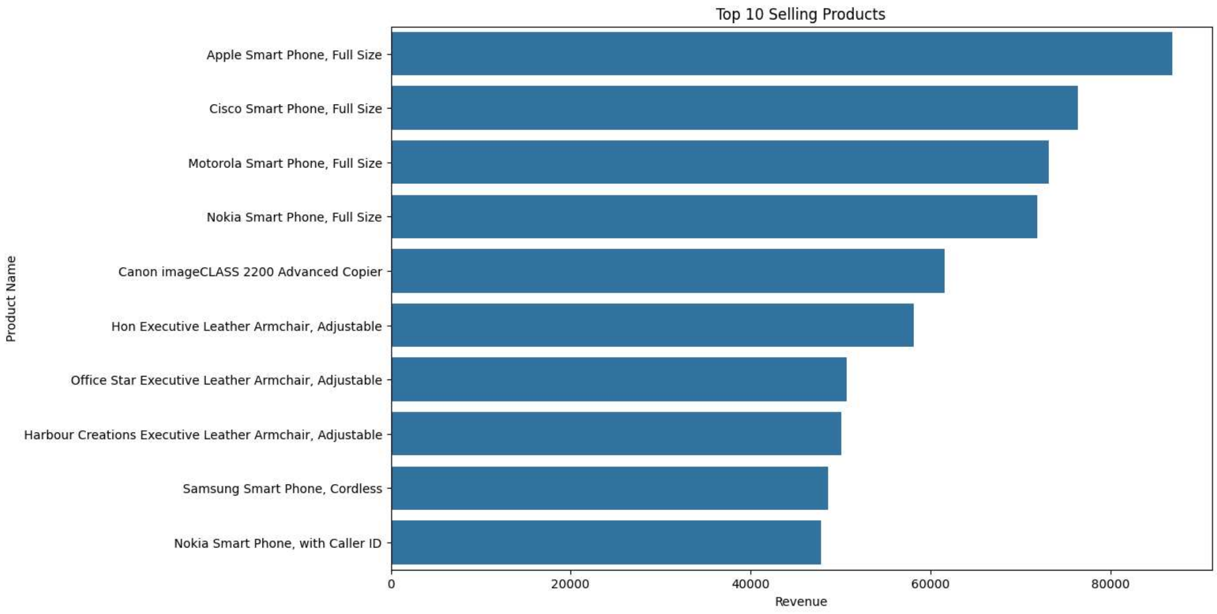
In [28]:
```python
# Converting Order Date to datetime
df['Order Date'] = pd.to_datetime(df['Order Date'])
```

In [18]:
```python
# Sales by Product Category

category_sales = df.groupby('Category')['Sales'].sum().reset_index()
plt.figure(figsize=(10, 6))
sns.barplot(x='Category', y='Sales', data=category_sales)
plt.title('Sales by Product Category')
plt.ylabel('Revenue')
plt.show()
```



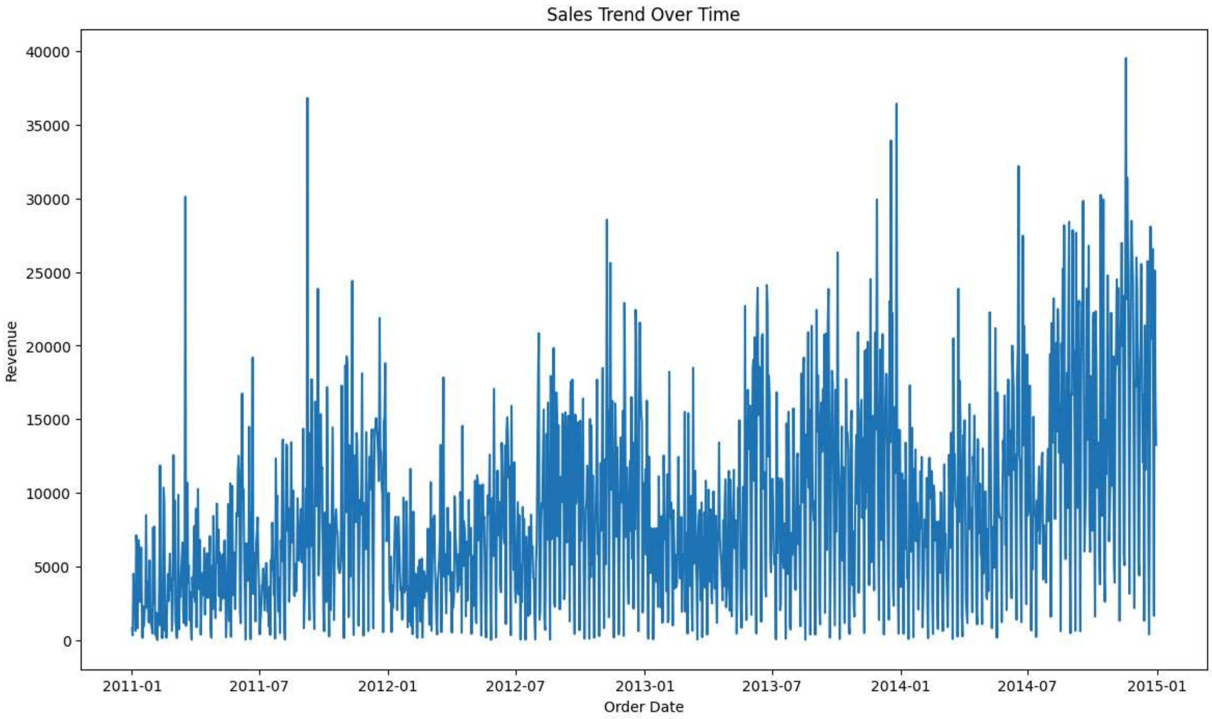
In [19]:

```
#Top Selling Products
product_sales = df.groupby('Product Name')['Sales'].sum().reset_index()
top_products = product_sales.sort_values(by='Sales', ascending=False).head(10)

plt.figure(figsize=(12, 8))
sns.barplot(x='Sales', y='Product Name', data=top_products)
plt.title('Top 10 Selling Products')
plt.xlabel('Revenue')
plt.show()
```



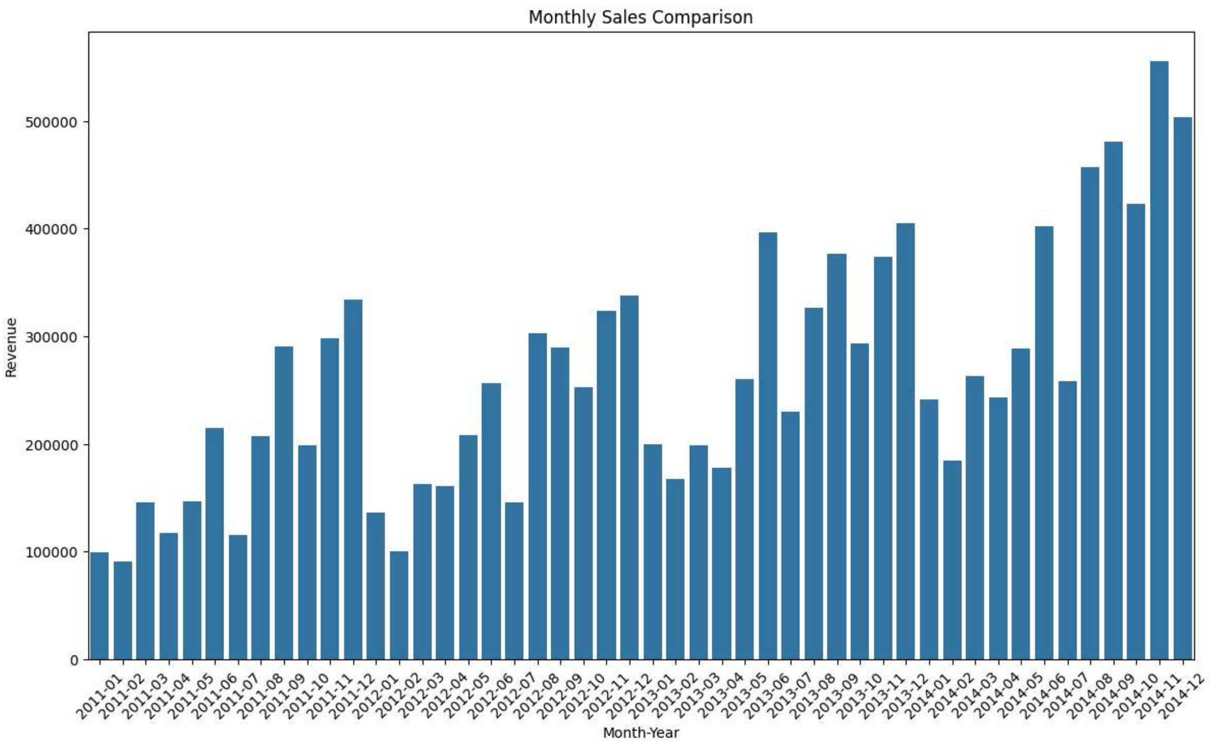
In [20]:

```python
# Sales Trend Over Time
sales_trend = df.groupby('Order Date')['Sales'].sum().reset_index()

plt.figure(figsize=(14, 8))
sns.lineplot(x='Order Date', y='Sales', data=sales_trend)
plt.title('Sales Trend Over Time')
plt.ylabel('Revenue')
plt.show()
```



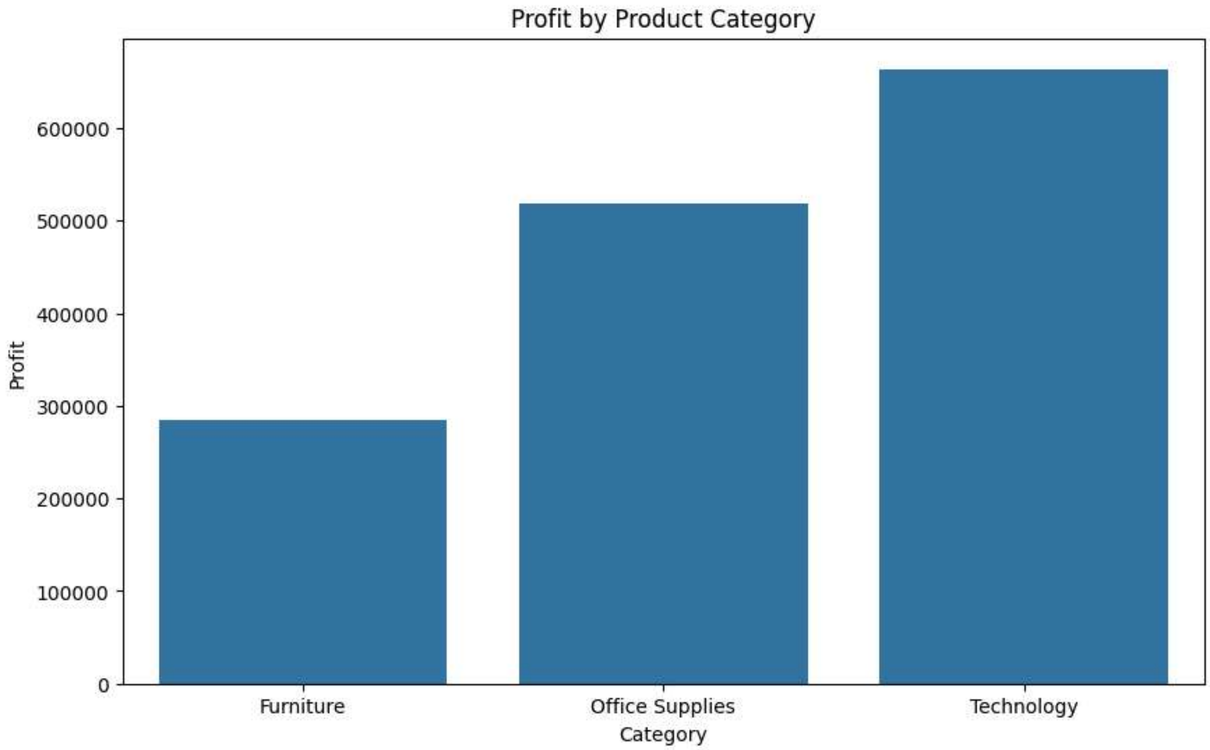
In [25]:

```python
## Monthly Sales Comparison
df['Month-Year'] = df['Order Date'].dt.to_period('M')
monthly_sales = df.groupby('Month-Year')['Sales'].sum().reset_index()

plt.figure(figsize=(14, 8))
sns.barplot(x='Month-Year', y='Sales', data=monthly_sales)
plt.title('Monthly Sales Comparison')
plt.ylabel('Revenue')
plt.xticks(rotation=45)
plt.show()
```



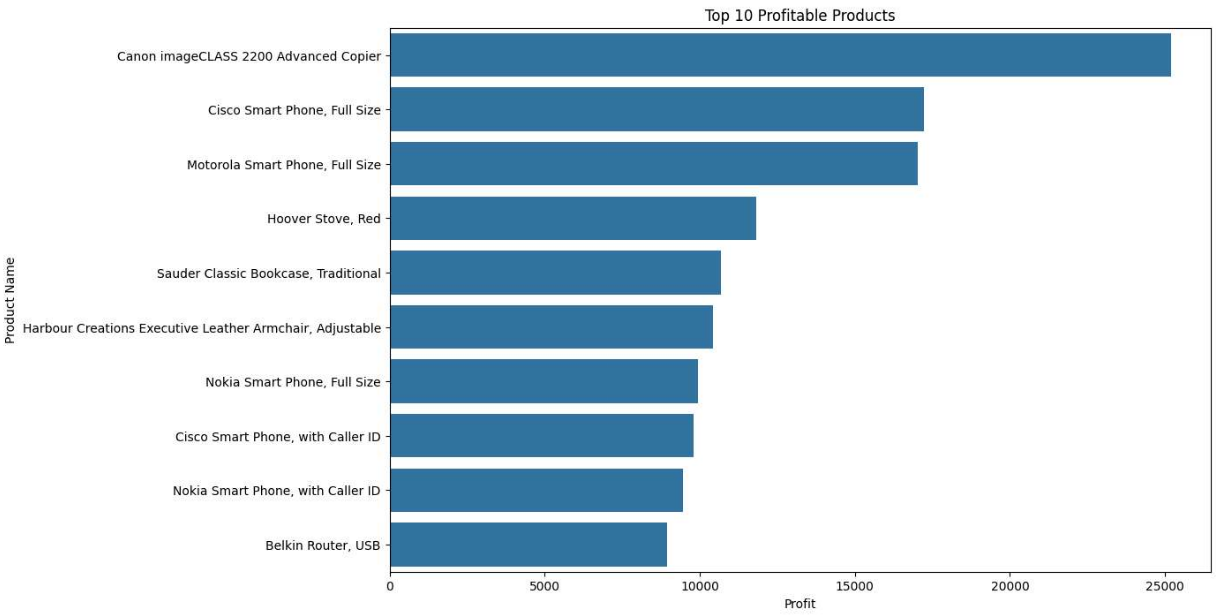
```
In [29]: # Profit by Product Category
         category_profit = df.groupby('Category')['Profit'].sum().reset_index()
         print(category_profit)

         plt.figure(figsize=(10, 6))
         sns.barplot(x='Category', y='Profit', data=category_profit)
         plt.title('Profit by Product Category')
         plt.ylabel('Profit')
         plt.show()
```

```
          Category         Profit
0        Furniture  285204.72380
1  Office Supplies  518473.83430
2       Technology  663778.73318
```



```
In [30]: # Top Profitable Products
         product_profit = df.groupby('Product Name')['Profit'].sum().reset_index()
         top_profitable_products = product_profit.sort_values(by='Profit', ascending=False).
         print(top_profitable_products)

         plt.figure(figsize=(12, 8))
         sns.barplot(x='Profit', y='Product Name', data=top_profitable_products)
         plt.title('Top 10 Profitable Products')
         plt.xlabel('Profit')
         plt.show()
```

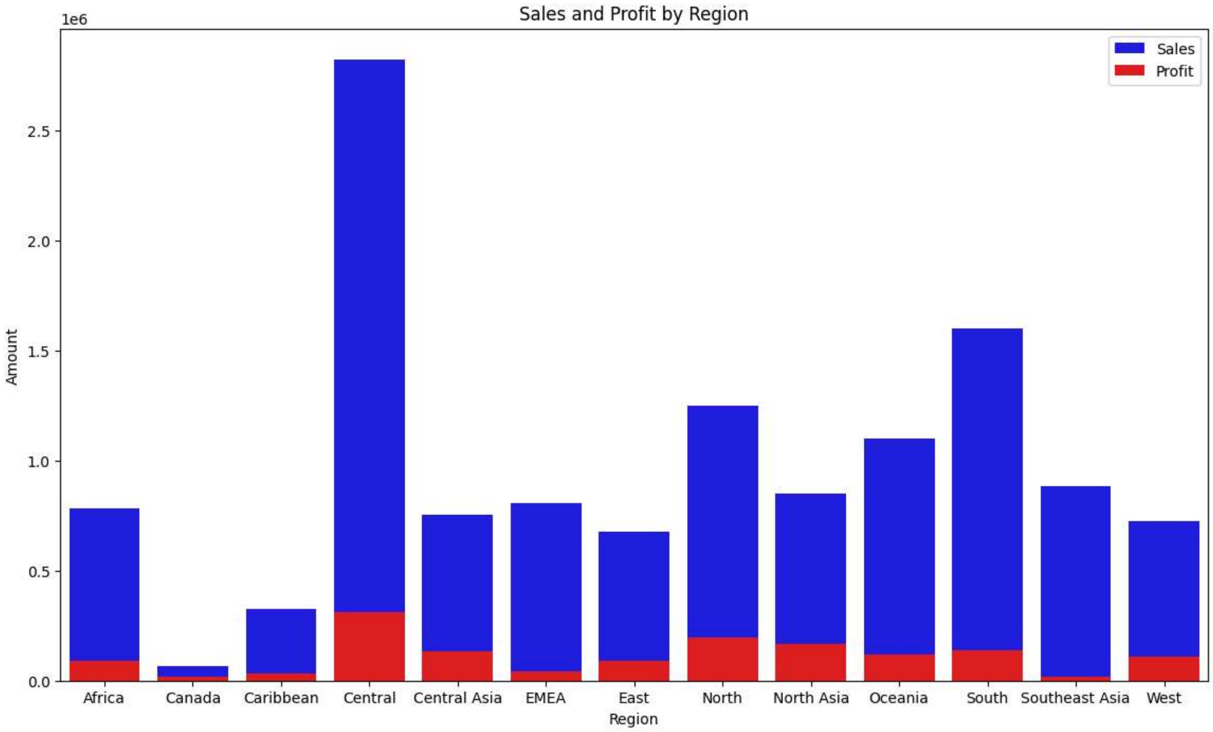
```
                                   Product Name      Profit
866          Canon imageCLASS 2200 Advanced Copier  25199.9280
970                Cisco Smart Phone, Full Size  17238.5206
2415             Motorola Smart Phone, Full Size  17027.1130
1926                         Hoover Stove, Red  11807.9690
3098            Sauder Classic Bookcase, Traditional  10672.0730
1714  Harbour Creations Executive Leather Armchair, ...  10427.3260
2501                Nokia Smart Phone, Full Size   9938.1955
971             Cisco Smart Phone, with Caller ID   9786.6408
2502            Nokia Smart Phone, with Caller ID   9465.3257
589                          Belkin Router, USB   8955.0180
```



In [31]:

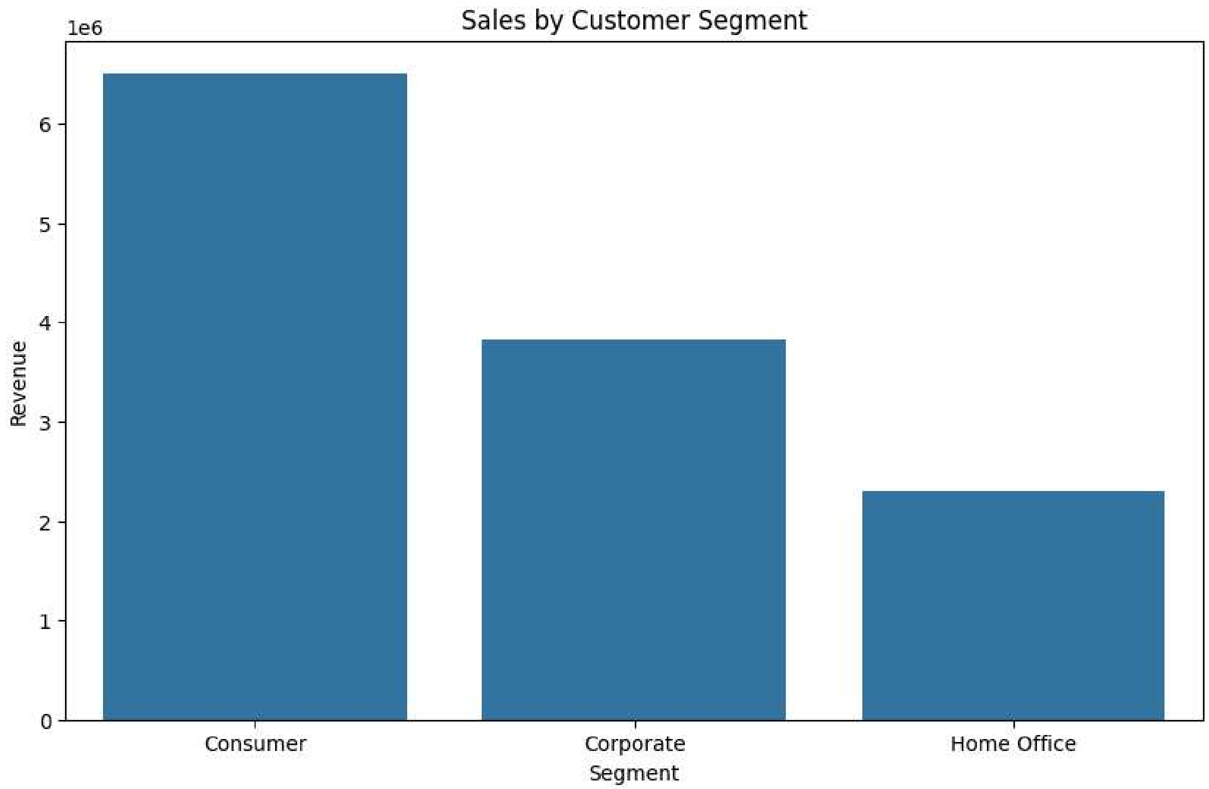
```
# Sales and Profit by Region
region_sales_profit = df.groupby('Region')[['Sales', 'Profit']].sum().reset_index()
print(region_sales_profit)

plt.figure(figsize=(14, 8))
sns.barplot(x='Region', y='Sales', data=region_sales_profit, label='Sales', color='
sns.barplot(x='Region', y='Profit', data=region_sales_profit, label='Profit', color
plt.title('Sales and Profit by Region')
plt.ylabel('Amount')
plt.legend()
plt.show()
```

```
            Region         Sales         Profit
0           Africa  7.837732e+05   88871.63100
1           Canada  6.692817e+04   17817.39000
2        Caribbean  3.242809e+05   34571.32104
3          Central  2.822303e+06  311403.98164
4     Central Asia  7.528266e+05  132480.18700
5             EMEA  8.061613e+05   43897.97100
6             East  6.787812e+05   91522.78000
7            North  1.248166e+06  194597.95252
8       North Asia  8.483098e+05  165578.42100
9          Oceania  1.100185e+06  120089.11200
10           South  1.600907e+06  140355.76618
11  Southeast Asia  8.844232e+05   17852.32900
12            West  7.254578e+05  108418.44890
```



```
In [32]: # Sales by Customer Segment
segment_sales = df.groupby('Segment')['Sales'].sum().reset_index()
print(segment_sales)

plt.figure(figsize=(10, 6))
sns.barplot(x='Segment', y='Sales', data=segment_sales)
plt.title('Sales by Customer Segment')
plt.ylabel('Revenue')
plt.show()
```

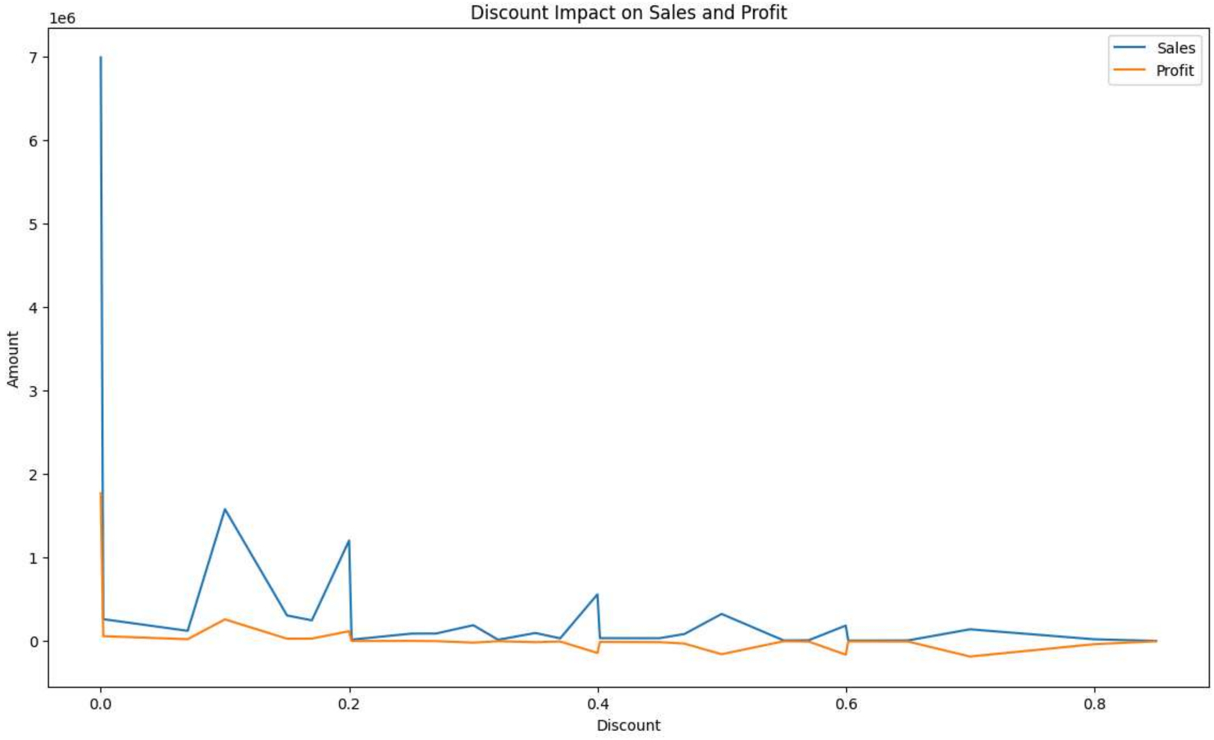
```
       Segment         Sales
0     Consumer  6.507949e+06
1    Corporate  3.824698e+06
2  Home Office  2.309855e+06
```



In [33]:

```
# Discount Impact on Sales and Profit
discount_sales_profit = df.groupby('Discount')[['Sales', 'Profit']].sum().reset_ind
print(discount_sales_profit)

plt.figure(figsize=(14, 8))
sns.lineplot(x='Discount', y='Sales', data=discount_sales_profit, label='Sales')
sns.lineplot(x='Discount', y='Profit', data=discount_sales_profit, label='Profit')
plt.title('Discount Impact on Sales and Profit')
plt.ylabel('Amount')
plt.legend()
plt.show()
```

```
    Discount         Sales        Profit
0      0.000  6.992411e+06  1.770695e+06
1      0.002  2.613956e+05  5.797658e+04
2      0.070  1.216169e+05  2.114850e+04
3      0.100  1.579606e+06  2.590642e+05
4      0.150  3.060298e+05  2.737590e+04
5      0.170  2.473992e+05  2.816307e+04
6      0.200  1.203832e+06  1.177159e+05
7      0.202  1.621565e+04 -5.952727e+02
8      0.250  8.805535e+04  8.005875e+02
9      0.270  8.935167e+04 -1.675079e+03
10     0.300  1.889320e+05 -1.968585e+04
11     0.320  1.449346e+04 -2.391138e+03
12     0.350  9.570649e+04 -1.416965e+04
13     0.370  3.161802e+04 -5.806202e+03
14     0.400  5.595252e+05 -1.437485e+05
15     0.402  3.373387e+04 -1.143045e+04
16     0.450  3.339113e+04 -1.360679e+04
17     0.470  8.343217e+04 -3.116225e+04
18     0.500  3.241311e+05 -1.586299e+05
19     0.550  6.308658e+03 -3.150672e+03
20     0.570  8.144802e+03 -6.313548e+03
21     0.600  1.845819e+05 -1.639547e+05
22     0.602  4.977181e+03 -4.905419e+03
23     0.650  6.579815e+03 -6.221966e+03
24     0.700  1.397722e+05 -1.863505e+05
25     0.800  2.046333e+04 -3.861623e+04
26     0.850  7.968420e+02 -3.068658e+03
```



```
In [ ]:
```