Analysis of sales of an international company

Target: understand which regions, products, and channels generate the most profit and provide business recommendations.

In [16]:
```
#We import the libraries necessary for work
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#Setting up graph display
plt.style.use('ggplot')
plt.rcParams['figure.figsize'] = (10,5)

#Connecting the dataset
df = pd.read_csv('sales.csv')
#Checking if the dataset has loaded
df.head()
```

Out[16]:

	region	country	item_type	sales_channel	order_priority	order_date	order_id
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	2010-05-28	669165933
1	Central America and the Caribbean	Grenada	Cereal	Online	C	2012-08-22	963881480
2	Europe	Russia	Office Supplies	Offline	L	2014-05-02	341417157
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	2014-06-20	514321792
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2013-02-01	115456712

In [17]:
```
#Initial data review
print(df.shape)
print(df.columns)
df.info()
print(df.describe())
```

```
(100, 17)
Index(['region', 'country', 'item_type', 'sales_channel', 'order_priority',
       'order_date', 'order_id', 'ship_date', 'units_sold', 'unit_price',
       'unit_cost', 'total_revenue', 'total_cost', 'total_profit', 'month',
       'year', 'shipping_days'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 17 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   region          100 non-null    object 
 1   country         100 non-null    object 
 2   item_type       100 non-null    object 
 3   sales_channel   100 non-null    object 
 4   order_priority  100 non-null    object 
 5   order_date      100 non-null    object 
 6   order_id        100 non-null    int64  
 7   ship_date       100 non-null    object 
 8   units_sold      100 non-null    int64  
 9   unit_price      100 non-null    float64
 10  unit_cost       100 non-null    float64
 11  total_revenue   100 non-null    float64
 12  total_cost      100 non-null    float64
 13  total_profit    100 non-null    float64
 14  month           100 non-null    object 
 15  year            100 non-null    int64  
 16  shipping_days   100 non-null    int64  
dtypes: float64(5), int64(4), object(8)
memory usage: 13.4+ KB
           order_id   units_sold  unit_price   unit_cost  total_revenue  \
count  1.000000e+02   100.000000  100.000000  100.000000   1.000000e+02   
mean   5.550204e+08  5128.710000  276.761300  191.048000   1.373488e+06   
std    2.606153e+08  2794.484562  235.592241  188.208181   1.460029e+06   
min    1.146066e+08   124.000000    9.330000    6.920000   4.870260e+03   
25%    3.389225e+08  2836.250000   81.730000   35.840000   2.687212e+05   
50%    5.577086e+08  5382.500000  179.880000  107.275000   7.523144e+05   
75%    7.907551e+08  7369.000000  437.200000  263.330000   2.212045e+06   
max    9.940222e+08  9925.000000  668.270000  524.960000   5.997055e+06   

         total_cost  total_profit         year  shipping_days  
count  1.000000e+02  1.000000e+02   100.000000     100.000000  
mean   9.318057e+05  4.416820e+05  2013.230000      23.360000  
std    1.083938e+06  4.385379e+05     2.088231      14.742586  
min    3.612240e+03  1.258020e+03  2010.000000       0.000000  
25%    1.688680e+05  1.214436e+05  2012.000000       9.750000  
50%    3.635664e+05  2.907680e+05  2013.000000      23.500000  
75%    1.613870e+06  6.358288e+05  2015.000000      36.250000  
max    4.509794e+06  1.719922e+06  2017.000000      50.000000  
```

In [18]:
```
#We remove spaces and convert everything to lowercase for easier work.
df.columns = df.columns.str.strip().str.replace(' ', '_').str.lower()

#Removing complete duplicates
df = df.drop_duplicates()
```

In [19]:
```
#Converting date column types to datetime, and incorrect dates in NaT
df['order_date'] = pd.to_datetime(df['order_date'], errors='coerce', dayfirst=Fa
df['ship_date'] = pd.to_datetime(df['ship_date'], errors='coerce', dayfirst=Fals
#Checking the number of empty dates
df['order_date'].isnull().sum(), df['ship_date'].isnull().sum()
```

Out[19]:
```
(np.int64(0), np.int64(0))
```

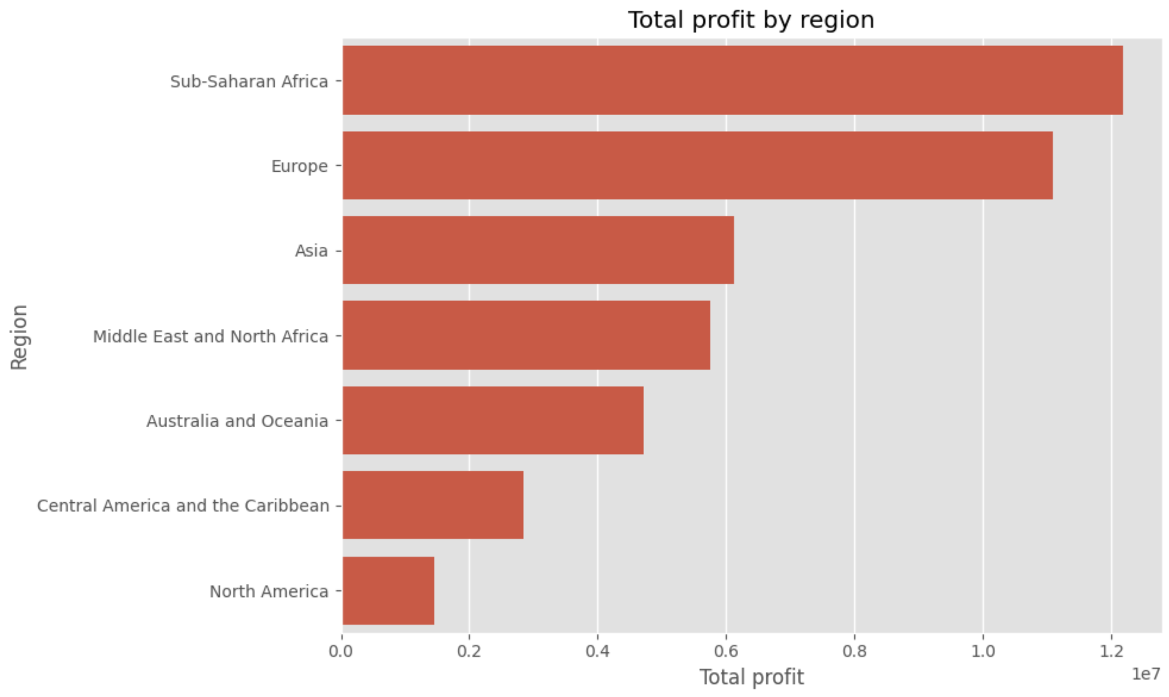
In [30]:
```
#Adding additional columns in the form of month and year
df['month'] = df['order_date'].dt.to_period('M')
df['year'] = df['order_date'].dt.year

#We calculate delivery time in days
df['shipping_days'] = (df['ship_date'] - df['order_date']).dt.days

#We calculate how many days it takes for delivery on average
average_shipping_days = round(df['shipping_days'].mean())
print(f"Average delivery time: {average_shipping_days} days")
```

```
Average delivery time: 23 days
```

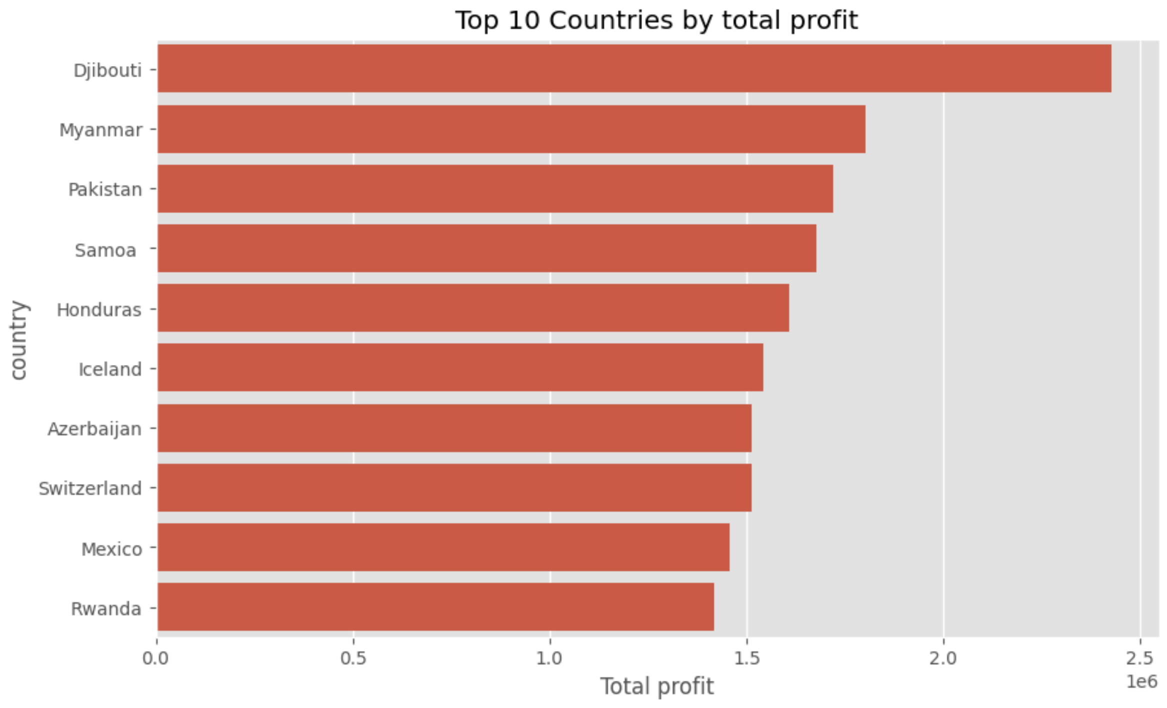
In [21]:
```
#Basic business metrics
total_revenue = df['total_revenue'].sum()
total_profit = df['total_profit'].sum()
unique_orders = df['order_id'].nunique()

#Calculate the average bill
order_revenue = df.groupby('order_id')['total_revenue'].sum()
average_order_value = order_revenue.mean()

#Information output
print(f"Total revenue: {total_revenue:,.2f}")
print(f"Total profit: {total_profit:,.2f}")
print(f"Number of unique orders: {unique_orders:,}")
print(f"Average order value: {average_order_value:,.2f}")
```

```
Total revenue: 137,348,768.31
Total profit: 44,168,198.40
Number of unique orders: 100
Average order value: 1,373,487.68
```

In [22]:
```
#We calculate profit by region
region_profit = df.groupby('region')['total_profit'].sum().sort_values(ascending
region_profit

#Creating a graph
plt.figure(figsize=(10, 6))
sns.barplot(x=region_profit.values, y=region_profit.index)
plt.title('Total profit by region')   # исправлено: title, не titel
plt.xlabel('Total profit')
plt.ylabel('Region')
plt.tight_layout()
plt.show()
```

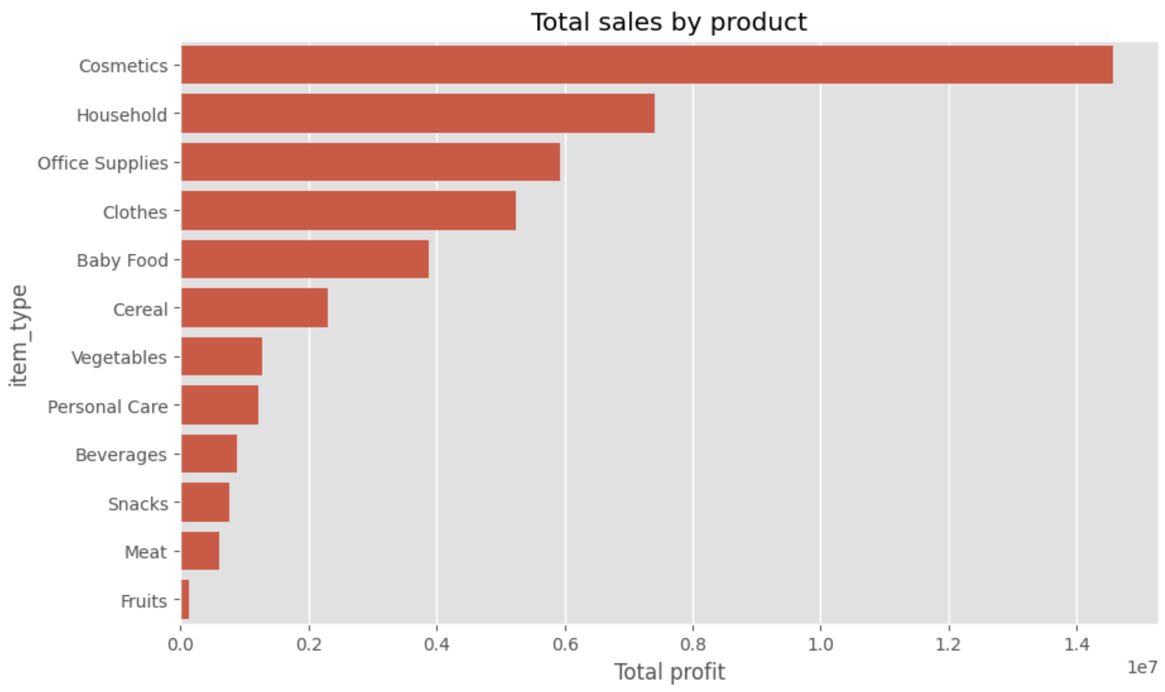


In [23]:
```
#We count the top 10 countries by income
top_countries = df.groupby('country')['total_profit'].sum().nlargest(10)
top_countries

#Creating a graph
plt.figure(figsize=(10,6))
sns.barplot(x=top_countries.values, y=top_countries.index)
plt.title('Top 10 Countries by total profit')
plt.xlabel('Total profit')
plt.show()
```

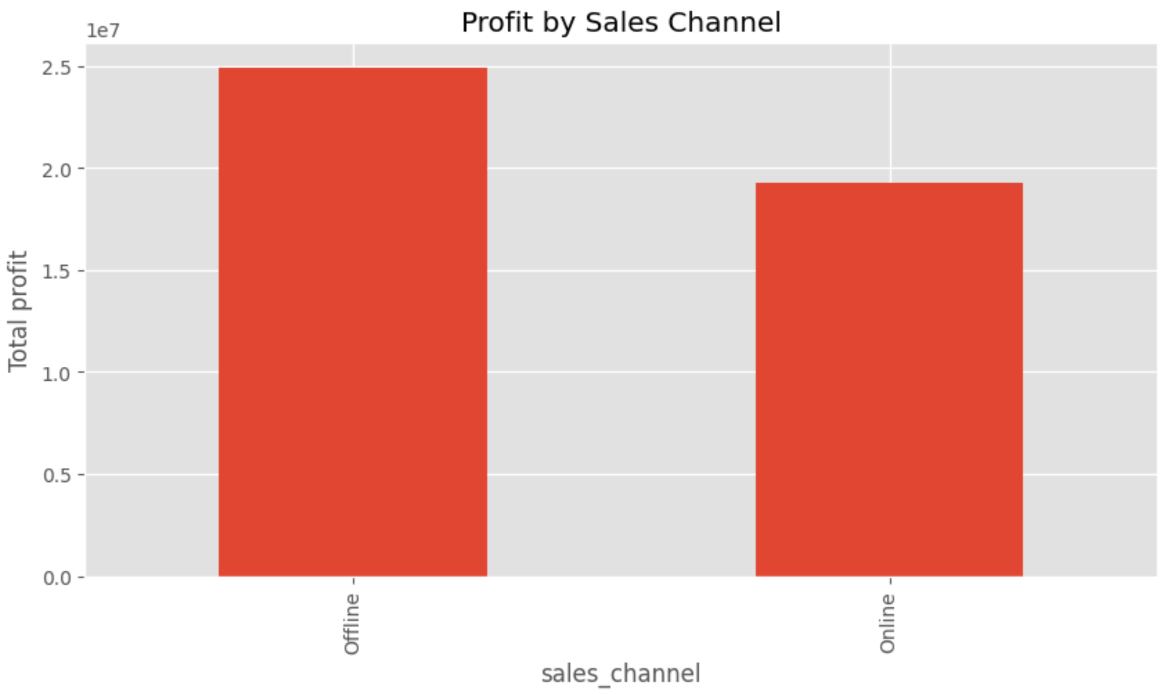


In [24]:
```
#Analysis of product categories by profit
item_profit = df.groupby('item_type')['total_profit'].sum().sort_values(ascendin
item_profit

#Creating a graph
```

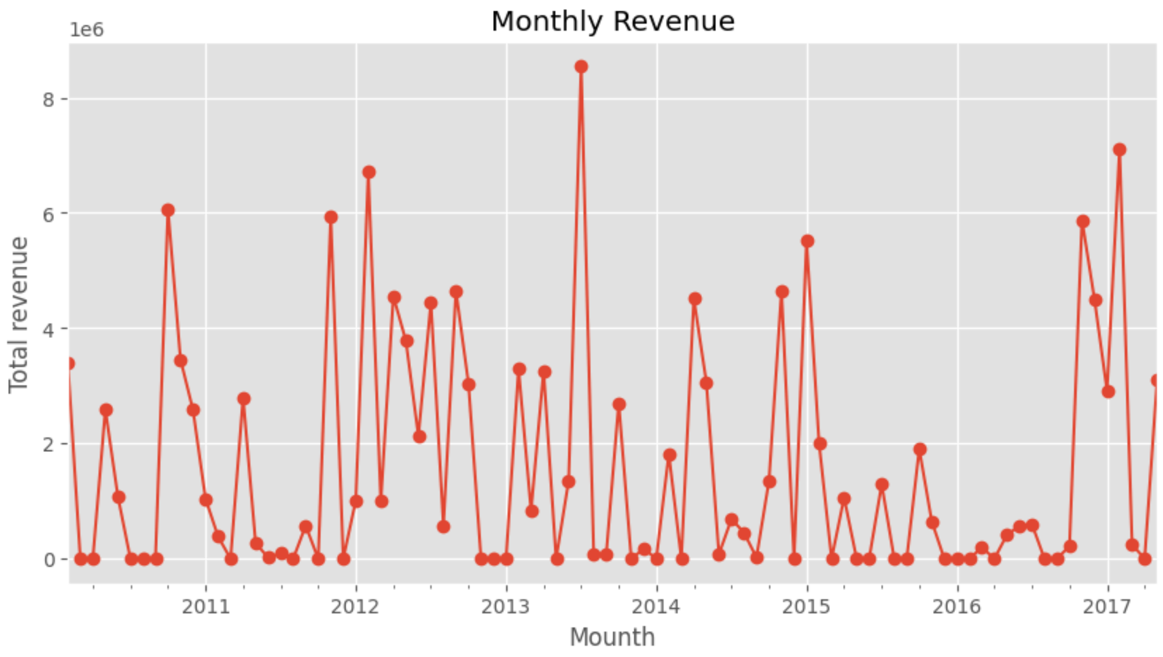
```
plt.figure(figsize=(10,6))
sns.barplot(x=item_profit.values, y=item_profit.index)
plt.title('Total sales by product')
plt.xlabel('Total profit')
plt.show()
```



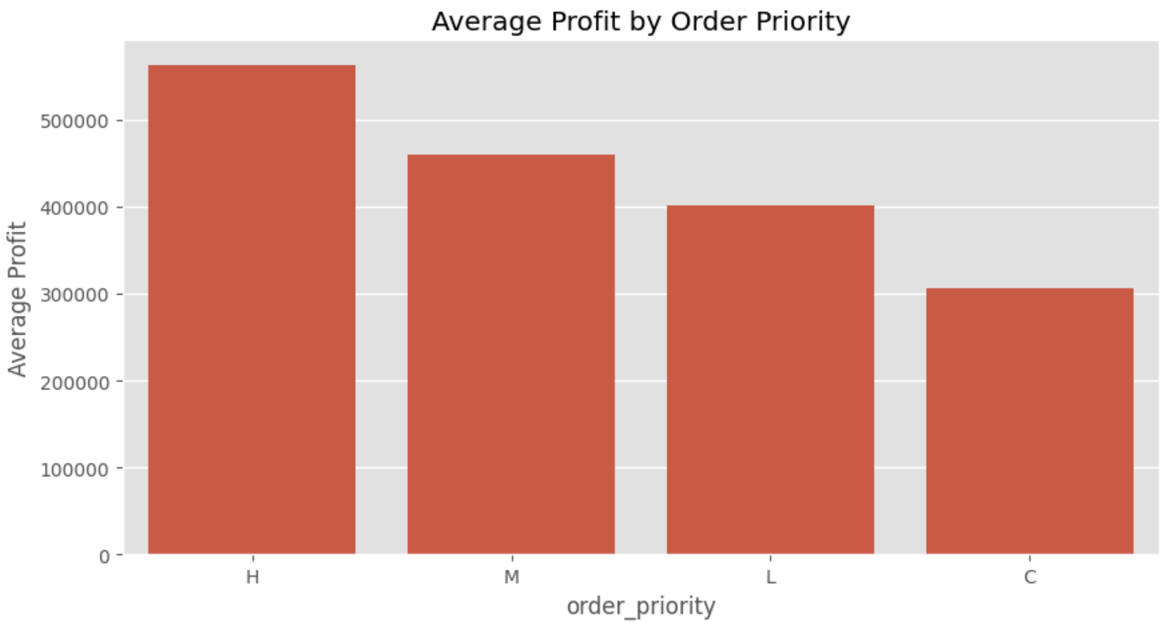
In [25]:
```
#Comparison of offline and online sales
channel_profit = df.groupby('sales_channel')['total_profit'].sum()
channel_profit

#Visualization
channel_profit.plot(kind='bar')
plt.title('Profit by Sales Channel')
plt.ylabel('Total profit')
plt.show()
```



In [26]:
```
#Comparison of profits by month
monthly_revenue = df.set_index('order_date').resample('ME')['total_revenue'].sum
monthly_revenue.plot(marker='o')

#Visualization
plt.title('Monthly Revenue')
plt.xlabel('Mounth')
plt.ylabel('Total revenue')
plt.show()
```



In [27]:
```
#Impact of order category on profit
priority_profit = df.groupby('order_priority')['total_profit'].mean().sort_value
priority_profit

#Visualization
sns.barplot(x=priority_profit.index, y=priority_profit.values)
plt.title('Average Profit by Order Priority')
plt.ylabel('Average Profit')
plt.show()
```



In [28]:
```
#Saving the modified dataset
df.to_csv('sales.csv', index=False)
```

📊 Sales Analysis Summary

Based on the performed data analysis, the following key insights were identified:

1. **Regional Performance**
    - The most profitable region is **Sub-Saharan Africa**, followed by **Europe**, which generates almost twice as much profit as third-place Asia.
    - **North America** is the least profitable region, showing significantly weaker results compared to others.
2. **Country-Level Insights**
    - **Djibouti** stands out as the top-performing country with a noticeable lead in total profit.
    - Other countries show more balanced and stable performance, without large deviations.
3. **Product Categories**
    - **Cosmetics** is the best-performing category, generating **over 14 million in profit**, more than double the next category - Household products.
    - **Household products** take second place , while **Meat** and **Fruits** show very low profitability and could be reconsidered in the product lineup.
4. **Sales Channels**
    - **Offline sales outperform online**, but online channels still contribute solid revenue and growth potential.
5. **Overall Financials**
    - **Total Revenue:** 137,348,768.31
    - **Total Profit:** 44,168,198.40
    - **Average Order Value:** 1,373,487.68

💡 Business Recommendations

1. **Expand operations in Sub-Saharan Africa and Europe**, as they deliver the highest profit margins.
   Consider investing more in logistics and advertising in these regions to strengthen market dominance.

2. **Reassess North American strategy**, as this region underperforms significantly.
   Investigate pricing, competition, or supply chain inefficiencies.

3. **Increase focus on high-profit product categories**, especially **Cosmetics** and **Household goods**.

Expand inventory and marketing campaigns in these segments.

4. **Review low-performing categories** such as **Meat** and **Fruits** — either improve their margins (e.g., through cost optimization) or reduce their presence in the catalog.

5. **Support both sales channels**, but prioritize **offline**, which currently delivers higher profit.
   At the same time, develop **online sales** as a long-term growth area.

6. **Evaluate and optimize delivery performance.**
   The current **average shipping time is 23 days**, which is relatively long for most business contexts.
   It is recommended to **analyze the supply chain** and identify opportunities to reduce shipping delays,
   especially if faster delivery could lead to higher customer satisfaction and repeat purchases.

✅ Summary

The company demonstrates strong sales and profit performance in key regions and categories.
Focusing on the most profitable markets and optimizing low-performing areas could significantly increase overall profitability.