Importing Libraries

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv(r'C:\Users\LENOVO\Downloads\Online Retail Sample.csv')
print("Initial shape:", df.shape)
```

```
Initial shape: (3066, 8)
```

In [3]:
```
df.head()
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	1/12/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	1/12/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	1/12/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	1/12/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	1/12/2010 8:26	3.39	17850.0	United Kingdom

Data Cleaning

In [4]:
```
# Data types and missing values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3066 entries, 0 to 3065
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   InvoiceNo    3066 non-null   object 
 1   StockCode    3066 non-null   object 
 2   Description  3056 non-null   object 
 3   Quantity     3066 non-null   int64  
 4   InvoiceDate  3066 non-null   object 
 5   UnitPrice    3066 non-null   float64
 6   CustomerID   1932 non-null   float64
 7   Country      3066 non-null   object 
dtypes: float64(2), int64(1), object(5)
memory usage: 191.8+ KB
```

In [5]:
```
# Count missing values
print("\nMissing values:\n", df.isnull().sum())
```

```
Missing values:
 InvoiceNo         0
StockCode          0
Description       10
Quantity           0
InvoiceDate        0
UnitPrice          0
CustomerID      1134
Country            0
dtype: int64
```

In [6]:
```
# Drop rows with missing CustomerID or Description
df_cleaned = df.dropna(subset=['CustomerID', 'Description'])
print("After dropping missing values:", df_cleaned.shape)
```

```
After dropping missing values: (1932, 8)
```

In [7]:
```
# Cancelled orders have InvoiceNo starting with 'C'
df_cleaned = df_cleaned[~df_cleaned['InvoiceNo'].astype(str).str.startswith('C')]
print("After removing cancelled invoices:", df_cleaned.shape)
```

```
After removing cancelled invoices: (1906, 8)
```

In [8]:
```
#Remove Negative or Zero Quantities and Unit Prices
df_cleaned = df_cleaned[(df_cleaned['Quantity'] > 0) & (df_cleaned['UnitPrice'] > 0)]
print("After removing negative/zero quantity & price:", df_cleaned.shape)
```

```
After removing negative/zero quantity & price: (1906, 8)
```

In [9]:
```
# Convert InvoiceDate to Datetime...
df_cleaned['InvoiceDate'] = pd.to_datetime(df_cleaned['InvoiceDate'])
df_cleaned['InvoiceDate'].dtype
```

Out[9]:
```
dtype('<M8[ns]')
```

In [10]:
```
#Add Total Amount Column
df_cleaned['TotalAmount'] = df_cleaned['Quantity'] * df_cleaned['UnitPrice']
df_cleaned['TotalAmount'].describe()
```

Out[10]:
```
count    1906.000000
mean       24.234208
std        66.378259
min         0.290000
25%         4.200000
50%        11.800000
75%        20.280000
max      1627.200000
Name: TotalAmount, dtype: float64
```

In [11]:

```
# Final check for nulls or duplicates
print("Final missing values:\n", df_cleaned.isnull().sum())
print("Number of duplicate rows:", df_cleaned.duplicated().sum())
```

```
Final missing values:
 InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
TotalAmount    0
dtype: int64
Number of duplicate rows: 44
```

In [12]:

```
# Drop exact duplicate rows
df_cleaned = df_cleaned.drop_duplicates()
print("Duplicates after dropping:", df_cleaned.duplicated().sum())
```

```
Duplicates after dropping: 0
```

Exploratory Data Analysis (EDA)

In [13]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Set plot style
plt.style.use('seaborn-v0_8-whitegrid')
sns.set_palette('muted')
```

In [14]:
```
# Check basic stats
df_cleaned.describe()

# Number of unique products, customers, and countries
print("Unique Products:", df_cleaned['Description'].nunique())
print("Unique Customers:", df_cleaned['CustomerID'].nunique())
print("Countries:", df_cleaned['Country'].nunique())
```

```
Unique Products: 927
Unique Customers: 94
Countries: 7
```

In [15]:
```
print("Min date:", df_cleaned['InvoiceDate'].min())
print("Max date:", df_cleaned['InvoiceDate'].max())

```

```
Min date: 2010-01-12 08:26:00
Max date: 2010-01-12 17:24:00
```

jupyter OnlineRetail_EDA Last Checkpoint: Yesterday at 7:39 PM (autosaved)

Logout

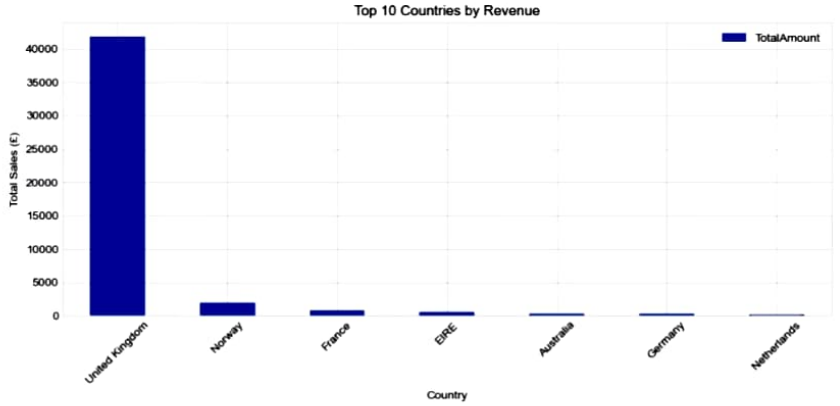
File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Run Markdown

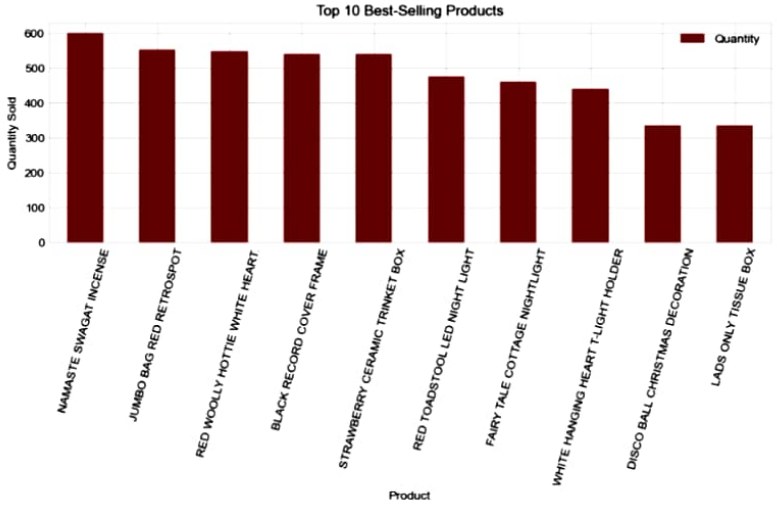
In [30]:

```
# Top 10 Countries by Revenue
country_sales = df_cleaned.groupby('Country')['TotalAmount'].sum().sort_values(ascending=False).head(10)

# Plot
plt.figure(figsize=(10,5))
country_sales.plot(kind='bar', color='darkblue')
plt.title('Top 10 Countries by Revenue')
plt.ylabel('Total Sales (£)')
plt.xlabel('Country')
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.savefig('Top_10_Countries_by_Revenue.png',dpi=300,bbox_inches='tight')
plt.show()
```



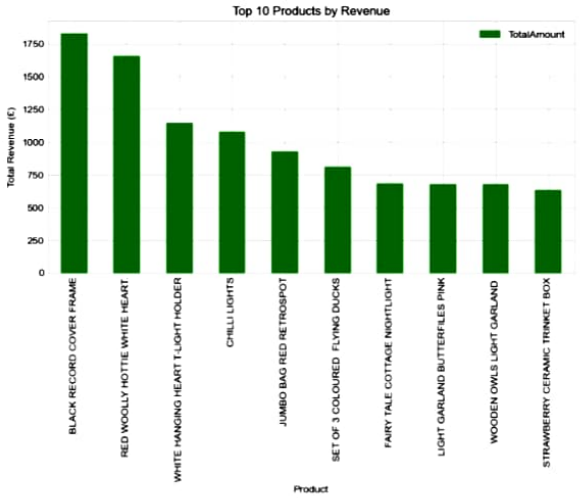
In [31]:

```
#Best-Selling Products
product_sales = df_cleaned.groupby('Description')['Quantity'].sum().sort_values(ascending=False).head(10)

# Plot
plt.figure(figsize=(9,6))
product_sales.plot(kind='bar', color='darkred')
plt.title('Top 10 Best-Selling Products')
plt.ylabel('Quantity Sold')
plt.xlabel('Product')
plt.xticks(rotation=75)
plt.legend()
plt.tight_layout()
plt.savefig('Top_10_Best-Selling_Products',dpi=300,bbox_inches='tight')
plt.show()
```



In [33]:

```
#High Revenue Products
top_revenue_products = df_cleaned.groupby('Description')['TotalAmount'].sum().sort_values(ascending=False).head(10)

# Plot
plt.figure(figsize=(8,7))
top_revenue_products.plot(kind='bar', color='green')
plt.title('Top 10 Products by Revenue')
plt.ylabel('Total Revenue (£)')
plt.xlabel('Product')
plt.xticks(rotation=90)
plt.legend()
plt.tight_layout()
plt.savefig('Top_10_Products_by_Revenue',dpi=300,bbox_inches='tight')
plt.show()

```



jupyter OnlineRetail_EDA Last Checkpoint: Yesterday at 7:39 PM (autosaved)

Logout

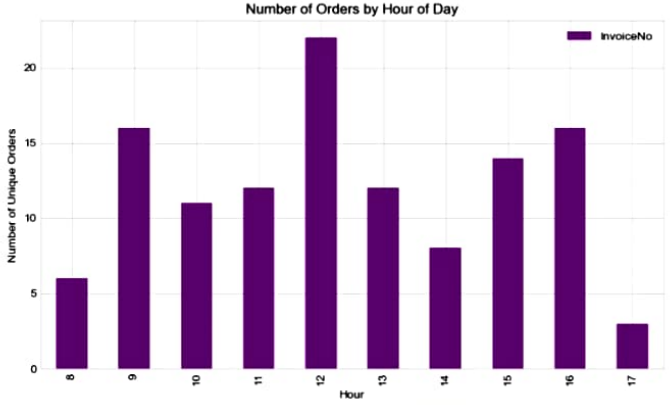
File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3 (ipykernel)

Run Markdown

In [34]:

```
#Number of Orders per Hour
df_cleaned['Hour'] = df_cleaned['InvoiceDate'].dt.hour
hourly_orders = df_cleaned.groupby('Hour')['InvoiceNo'].nunique()

# Plot
plt.figure(figsize=(8,5))
hourly_orders.plot(kind='bar', color='purple')
plt.title('Number of Orders by Hour of Day')
plt.xlabel('Hour')
plt.ylabel('Number of Unique Orders')
plt.legend()
plt.tight_layout()
plt.savefig('Number_of_Orders_by_Hour_of_Day',dpi=300,bbox_inches='tight')
plt.show()

```



In [42]:

```
# Returns Analysis
# Look at negative quantities
if not most_returned.empty:
    most_returned.plot(kind='bar', color='red')
    plt.title('Most Frequently Returned Products')
    plt.ylabel('Number of Returns')
    plt.xlabel('Product')
    plt.xticks(rotation=75)
    plt.tight_layout()
    plt.show()
else:
    print("No returned products found in the dataset.")

```

```
No returned products found in the dataset.
```

In [35]:

```
#Top Customers by Spend
top_customers = df_cleaned.groupby('CustomerID')['TotalAmount'].sum().sort_values(ascending=False).head(10)

# Plot
plt.figure(figsize=(8,6))
top_customers.plot(kind='bar', color='brown')
plt.title('Top 10 Customers by Revenue')
plt.ylabel('Total Spend (£)')
plt.xlabel('Customer ID')
plt.legend()
plt.tight_layout()
plt.savefig('Top_10_Customers_by_Revenue',dpi=300,bbox_inches='tight')
plt.show()

```