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

df = pd.read_csv("data.csv", encoding='latin1')

df.head()
```

```
  InvoiceNo StockCode                          Description
Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6

1    536365     71053                  WHITE METAL LANTERN         6

2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8

3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6

4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6


      InvoiceDate  UnitPrice  CustomerID         Country
0  12/1/2010 8:26       2.55     17850.0  United Kingdom
1  12/1/2010 8:26       3.39     17850.0  United Kingdom
2  12/1/2010 8:26       2.75     17850.0  United Kingdom
3  12/1/2010 8:26       3.39     17850.0  United Kingdom
4  12/1/2010 8:26       3.39     17850.0  United Kingdom
```

```
print(df.head())
```

```
  InvoiceNo StockCode                          Description
Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6

1    536365     71053                  WHITE METAL LANTERN         6

2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8

3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6

4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6


      InvoiceDate  UnitPrice  CustomerID         Country
0  12/1/2010 8:26       2.55     17850.0  United Kingdom
1  12/1/2010 8:26       3.39     17850.0  United Kingdom
2  12/1/2010 8:26       2.75     17850.0  United Kingdom
3  12/1/2010 8:26       3.39     17850.0  United Kingdom
4  12/1/2010 8:26       3.39     17850.0  United Kingdom
```

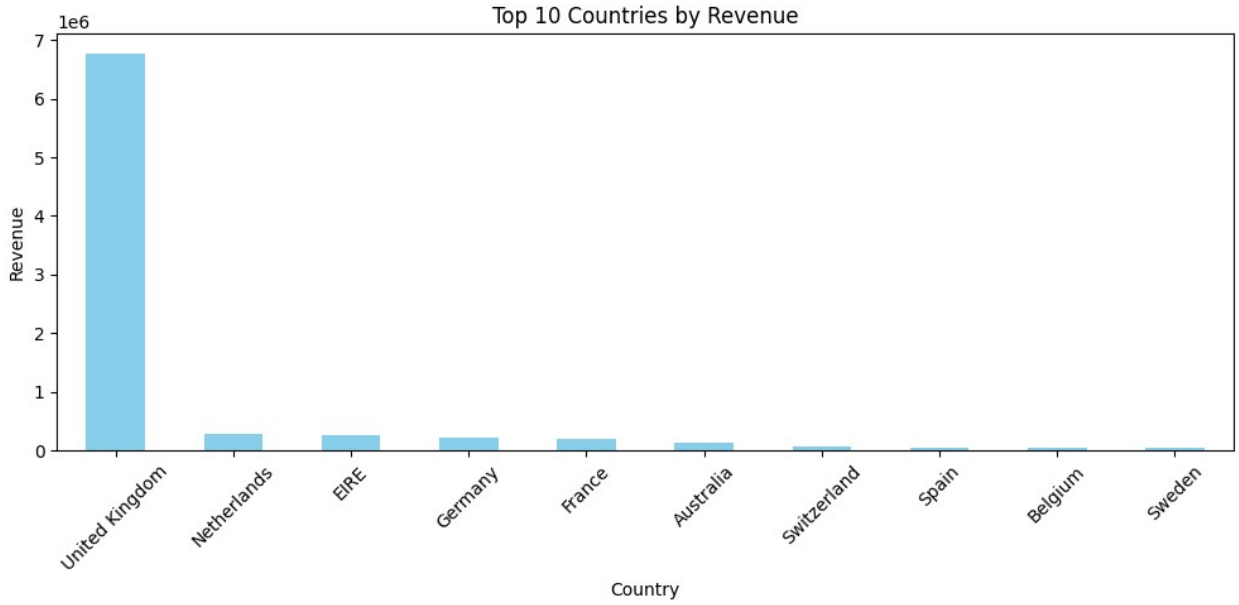
```
# Data Cleaning
df.dropna(subset=['CustomerID'], inplace=True)
```

```python
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']

# Basic info
print(df.info())
print(df.describe())
```

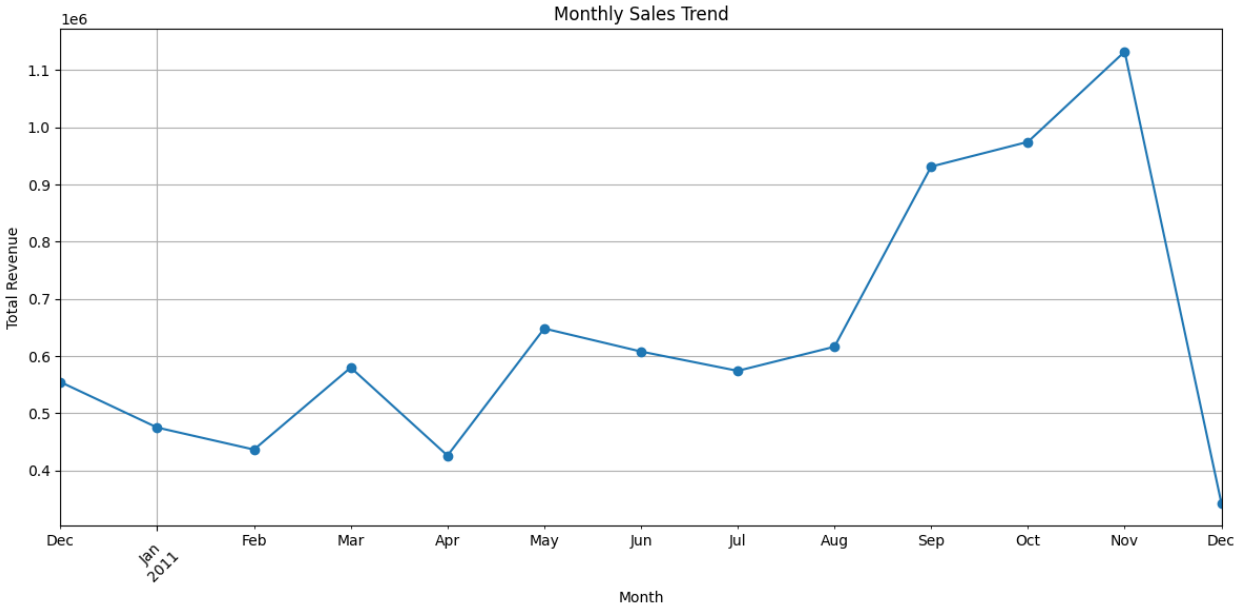
```
<class 'pandas.core.frame.DataFrame'>
Index: 406829 entries, 0 to 541908
Data columns (total 9 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    406829 non-null  object
 1   StockCode    406829 non-null  object
 2   Description  406829 non-null  object
 3   Quantity     406829 non-null  int64
 4   InvoiceDate  406829 non-null  datetime64[ns]
 5   UnitPrice    406829 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      406829 non-null  object
 8   TotalPrice   406829 non-null  float64
dtypes: datetime64[ns](1), float64(3), int64(1), object(4)
memory usage: 31.0+ MB
None
            Quantity                    InvoiceDate      UnitPrice  \
count  406829.000000                         406829  406829.000000
mean       12.061303  2011-07-10 16:30:57.879207424       3.460471
min    -80995.000000            2010-12-01 08:26:00       0.000000
25%         2.000000            2011-04-06 15:02:00       1.250000
50%         5.000000            2011-07-31 11:48:00       1.950000
75%        12.000000            2011-10-20 13:06:00       3.750000
max     80995.000000            2011-12-09 12:50:00   38970.000000
std       248.693370                            NaN      69.315162

          CustomerID     TotalPrice
count  406829.000000  406829.000000
mean    15287.690570      20.401854
min     12346.000000 -168469.600000
25%     13953.000000       4.200000
50%     15152.000000      11.100000
75%     16791.000000      19.500000
max     18287.000000  168469.600000
std      1713.600303     427.591718
```

```python
# Top 10 Countries by Revenue
country_revenue = df.groupby('Country')
['TotalPrice'].sum().sort_values(ascending=False).head(10)

plt.figure(figsize=(10,5))
country_revenue.plot(kind='bar', color='skyblue')
```

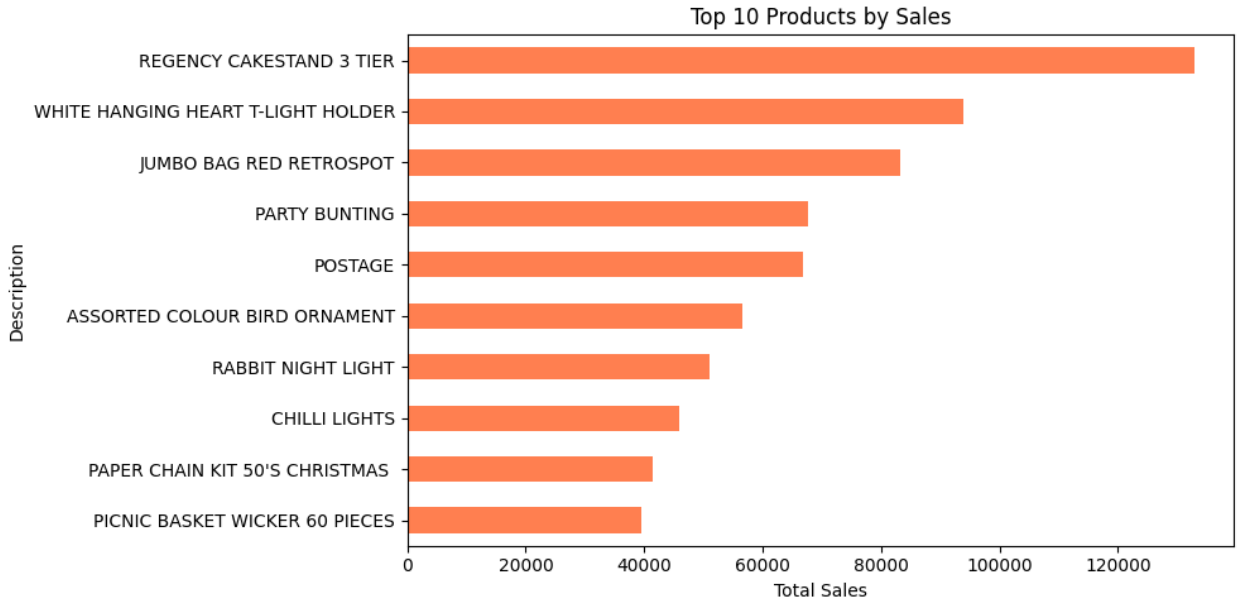
```
plt.title('Top 10 Countries by Revenue')
plt.ylabel('Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
# Monthly Sales Trend
df['Month'] = df['InvoiceDate'].dt.to_period('M')
monthly_sales = df.groupby('Month')['TotalPrice'].sum()

# check sale
plt.figure(figsize=(12,6))
monthly_sales.plot(marker='o')
plt.title('Monthly Sales Trend')
plt.ylabel('Total Revenue')
plt.xlabel('Month')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
# Top 10 Products by Sales
product_sales = df.groupby('Description')
['TotalPrice'].sum().sort_values(ascending=False).head(10)
plt.figure(figsize=(10,5))
product_sales.plot(kind='barh', color='coral')
plt.title('Top 10 Products by Sales')
plt.xlabel('Total Sales')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```



```
# RFM Analysis Preparation
rfm = df.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (df['InvoiceDate'].max() - x.max()).days,
    'InvoiceNo': 'count',
    'TotalPrice': 'sum'
})
rfm.columns = ['Recency', 'Frequency', 'Monetary']
print(rfm.head())
```

CustomerID	Recency	Frequency	Monetary
12346.0	325	2	0.00
12347.0	1	182	4310.00
12348.0	74	31	1797.24
12349.0	18	73	1757.55
12350.0	309	17	334.40

```
# Correlation Heatmap
plt.figure(figsize=(6,4))
sns.heatmap(rfm.corr(), annot=True, cmap='coolwarm')
plt.title('RFM Correlation')
plt.tight_layout()
plt.show()
```