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

In [2]:
```
df=pd.read_csv("OnlineRetail.csv",encoding='latin-1')
```

In [3]:
```
df
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

541909 rows × 8 columns

In [4]:
```
print(df.isnull().sum())
```

```
InvoiceNo          0
StockCode          0
Description     1454
Quantity           0
InvoiceDate        0
UnitPrice          0
CustomerID     135080
Country            0
dtype: int64
```

In [5]:
```
df.shape
```

Out[5]:
```
(541909, 8)
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  object
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [7]:
```
df=df.dropna()
```

In [8]:
```
df
```

Out[8]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

406829 rows × 8 columns

In [9]:
```
df1=df.duplicated().sum()
```

In [10]:
```
df1
```

Out[10]:
```
5225
```

In [11]:
```
df=df.drop_duplicates()
```

In [12]:
```
df
```

Out[12]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

401604 rows × 8 columns

In [13]:
```
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
total_revenue = df['TotalPrice'].sum()
print("Total sales revenue:", total_revenue)
```

```
Total sales revenue: 8278519.4240000015
C:\Users\DELL\AppData\Local\Temp\ipykernel_6072\174943563.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
C:\Users\DELL\AppData\Local\Temp\ipykernel_6072\174943563.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
```

In [14]:
```
unique_customers = df['CustomerID'].nunique()
print("Number of unique customers:", unique_customers)
```

```
Number of unique customers: 4372
```

In [15]:
```
num_orders = df['InvoiceNo'].nunique()
```

```
print("Number of orders:", num_orders)
```

```
Number of orders: 22190
```

In [16]:
```
avg_order_value = total_revenue / num_orders
print("Average order value:", avg_order_value)
```

```
Average order value: 373.0743318611988
```

In [17]:
```
popular_products = df['Description'].value_counts().head(5)
print("Most popular products:\n", popular_products)
```

```
Most popular products:
 WHITE HANGING HEART T-LIGHT HOLDER    2058
REGENCY CAKESTAND 3 TIER              1894
JUMBO BAG RED RETROSPOT               1659
PARTY BUNTING                         1409
ASSORTED COLOUR BIRD ORNAMENT         1405
Name: Description, dtype: int64
```

In [18]:
```
customer_with_most_purchases = df['CustomerID'].value_counts().idxmax()
print("Customer with the highest number of purchases:", customer_with_most_purchases)
```

```
Customer with the highest number of purchases: 17841.0
```

In [19]:
```
monthly_sales = df.groupby(df['InvoiceDate'].dt.to_period('M')).sum()['TotalPrice']
print("Monthly sales trends:\n", monthly_sales)
```

```
Monthly sales trends:
 InvoiceDate
2010-12     552372.860
2011-01     473731.900
2011-02     435534.070
2011-03     578576.210
2011-04     425222.671
2011-05     647011.670
2011-06     606862.520
2011-07     573112.321
2011-08     615078.090
2011-09     929356.232
2011-10     973306.380
2011-11    1126815.070
2011-12     341539.430
Freq: M, Name: TotalPrice, dtype: float64
C:\Users\DELL\AppData\Local\Temp\ipykernel_6072\3226995229.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, nu
meric_only will default to False. Either specify numeric_only or select only columns whi
ch should be valid for the function.
  monthly_sales = df.groupby(df['InvoiceDate'].dt.to_period('M')).sum()['TotalPrice']
```

In [20]:
```
sales_distribution = df.groupby('Country').sum()['TotalPrice'].sort_values(ascending=Fal
print("Country-wise sales distribution:\n", sales_distribution)
```

```
Country-wise sales distribution:
 Country
United Kingdom          6747156.154
Netherlands              284661.540
EIRE                     250001.780
Germany                  221509.470
France                   196626.050
Australia                137009.770
Switzerland               55739.400
Spain                     54756.030
Belgium                   40910.960
Sweden                    36585.410
Japan                     35340.620
Norway                    35163.460
```

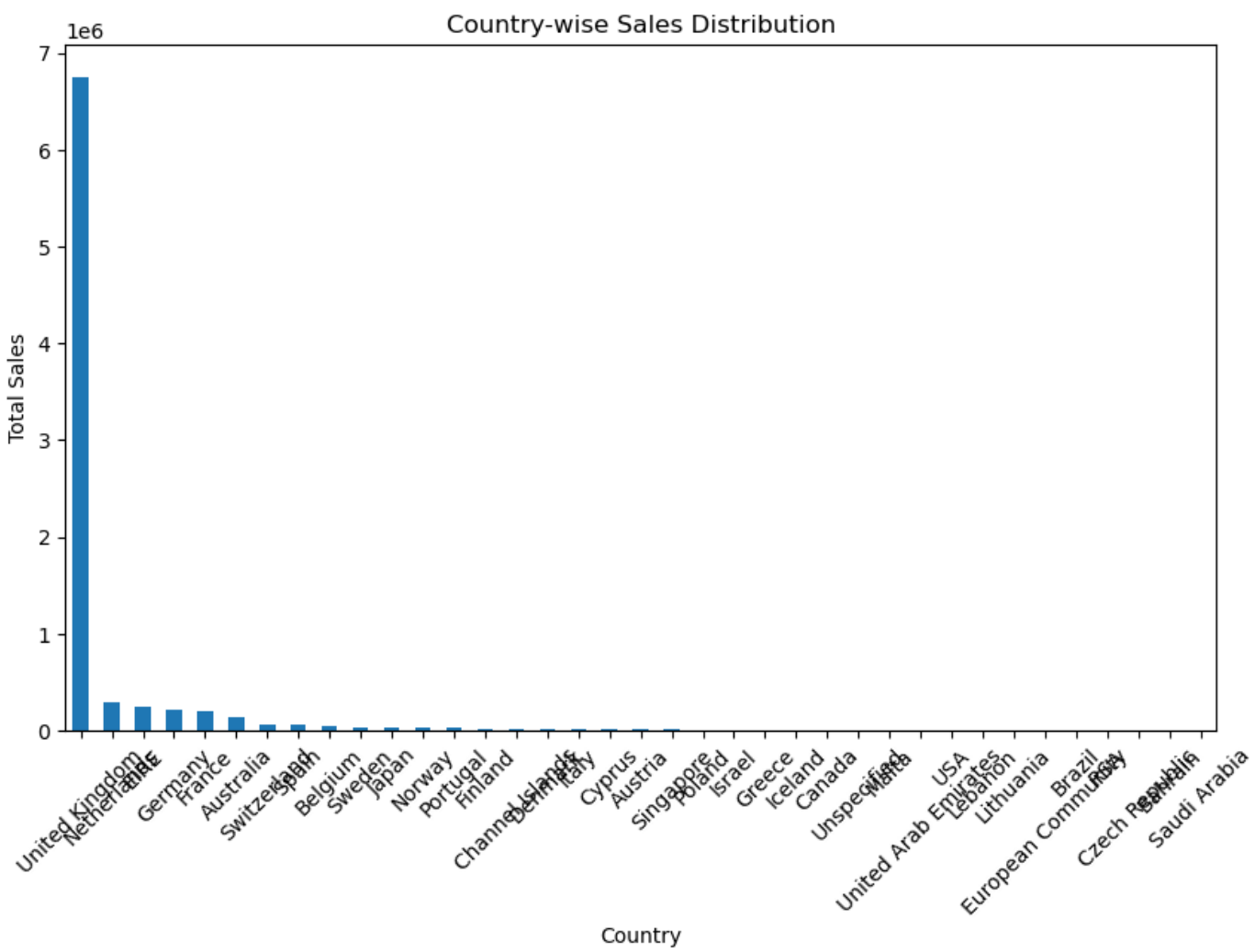
```
Portugal                 28995.760
Finland                  22326.740
Channel Islands          20076.390
Denmark                  18768.140
Italy                    16890.510
Cyprus                   12858.760
Austria                  10154.320
Singapore                 9120.390
Poland                    7213.140
Israel                    6988.400
Greece                    4710.520
Iceland                   4310.000
Canada                    3666.380
Unspecified               2660.770
Malta                     2505.470
United Arab Emirates      1902.280
USA                       1730.920
Lebanon                   1693.880
Lithuania                 1661.060
European Community        1291.750
Brazil                    1143.600
RSA                       1002.310
Czech Republic             707.720
Bahrain                    548.400
Saudi Arabia               131.170
Name: TotalPrice, dtype: float64
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_6072\1016603934.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, nu
meric_only will default to False. Either specify numeric_only or select only columns whi
ch should be valid for the function.
  sales_distribution = df.groupby('Country').sum()['TotalPrice'].sort_values(ascending=F
alse)
```

In [21]:
```
df['PurchaseDate'] = df['InvoiceDate'].dt.date
average_purchase_frequency = df.groupby('CustomerID')['PurchaseDate'].nunique().mean()
print("Average purchase frequency per customer:", average_purchase_frequency)
```

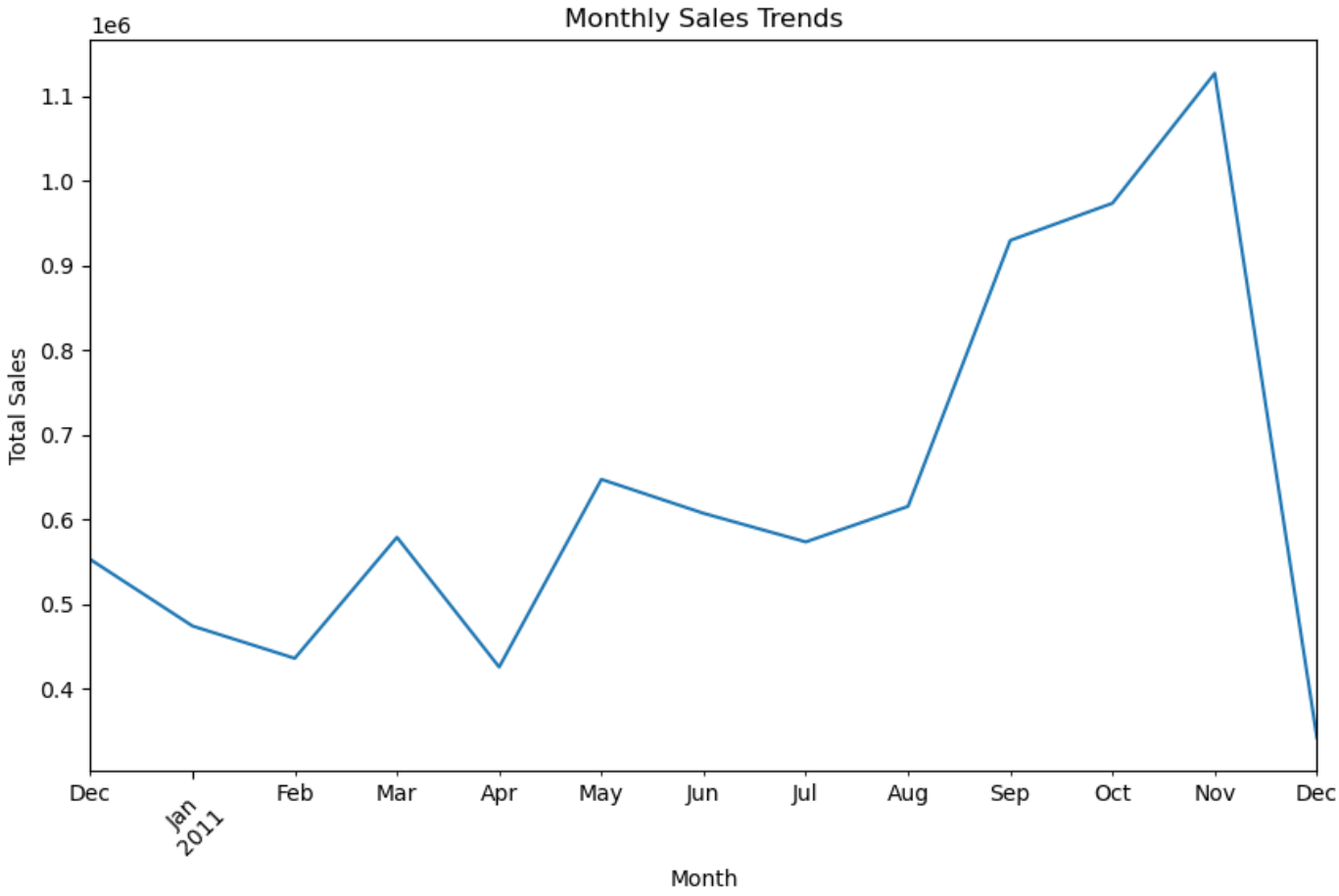
```
C:\Users\DELL\AppData\Local\Temp\ipykernel_6072\21020352.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['PurchaseDate'] = df['InvoiceDate'].dt.date
Average purchase frequency per customer: 4.413540713632205
```

In [22]:
```
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 6))
sales_distribution.plot(kind='bar')
plt.title('Country-wise Sales Distribution')
plt.xlabel('Country')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show()
```



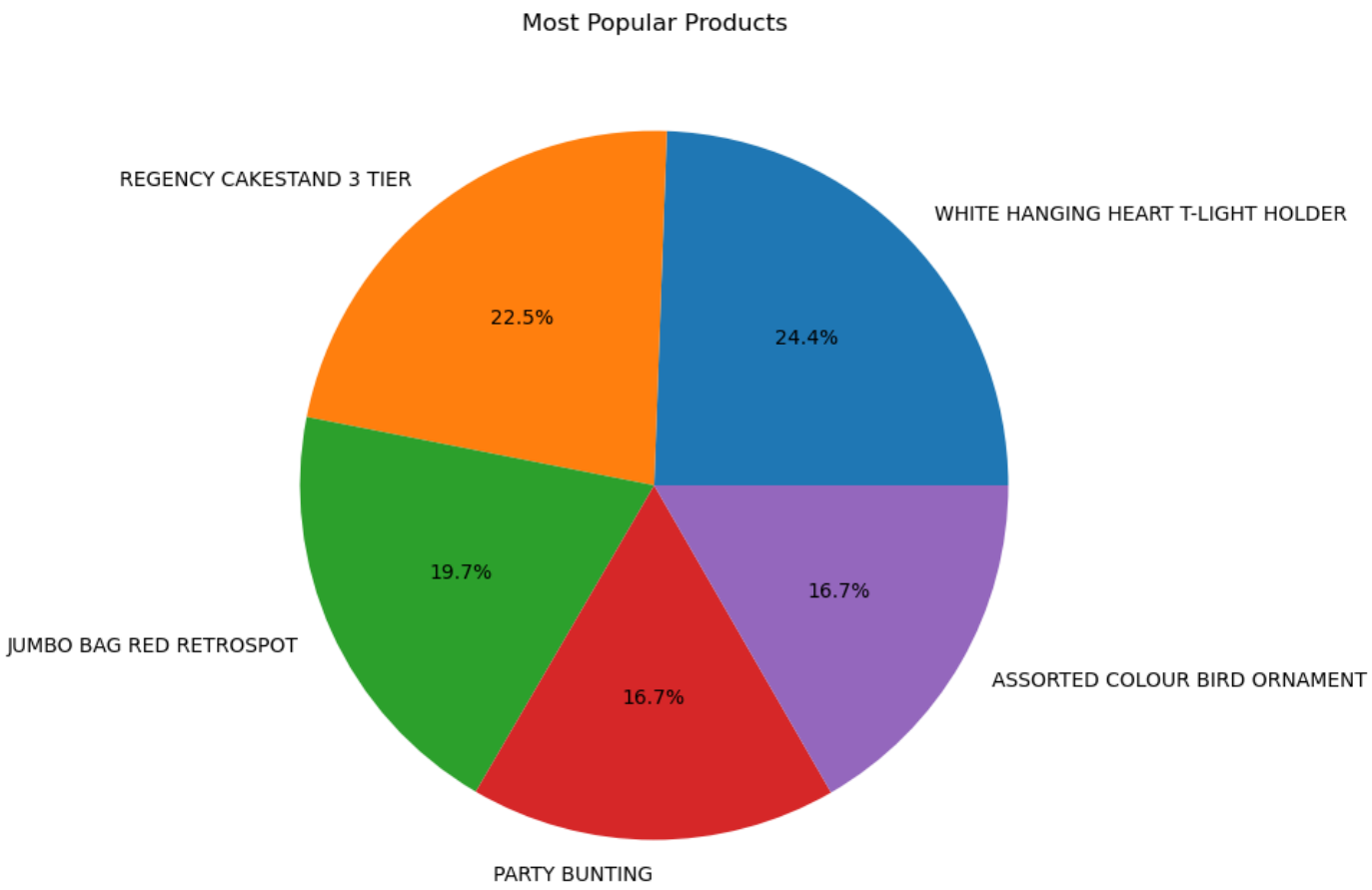
In [23]:

```
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 6))
monthly_sales.plot(kind='line')
plt.title('Monthly Sales Trends')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show()
```

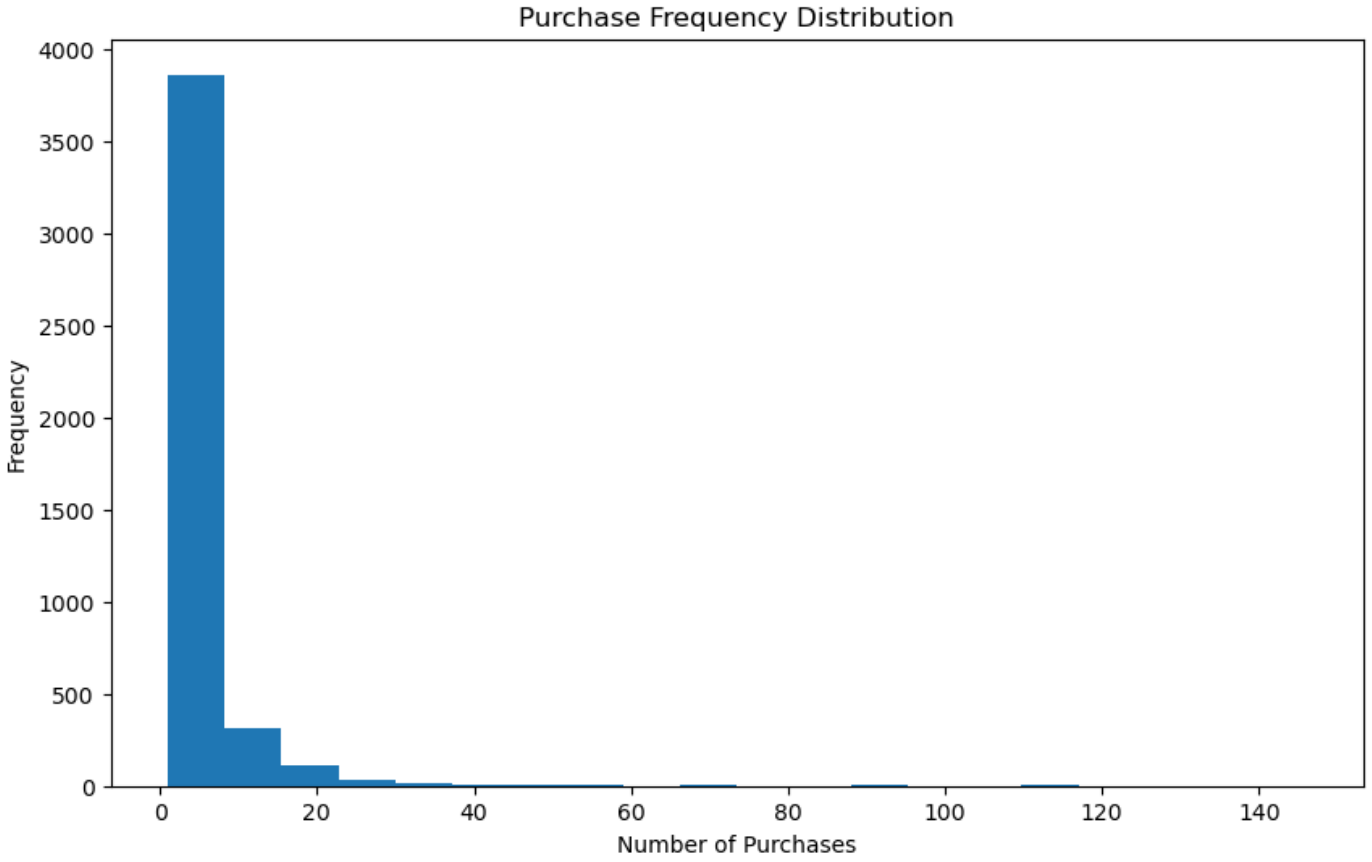


In [24]:

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 8))
popular_products.plot(kind='pie', autopct='%1.1f%%')
plt.title('Most Popular Products')
plt.ylabel('')
plt.show()
```



In [25]:
```python
import matplotlib.pyplot as plt


purchase_frequency = df.groupby('CustomerID')['PurchaseDate'].nunique()


plt.figure(figsize=(10, 6))
plt.hist(purchase_frequency, bins=20)
plt.title('Purchase Frequency Distribution')
plt.xlabel('Number of Purchases')
plt.ylabel('Frequency')
plt.show()
```


Purchase Frequency Distribution
Frequency
Number of Purchases
0
500
1000
1500
2000
2500
3000
3500
4000
0
20
40
60
80
100
120
140

In [ ]:

In [ ]: