## Importing Data & Libraries

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
data = pd.read_csv('Ecom_CRM_analysis.csv', encoding='unicode_escape')
```

In [4]:
```
data.head()
```

Out[4]:

| | 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 |

## Basic Metrics

In [6]:
```
data.shape
```

Out[6]:
```
(541909, 8)
```

In [7]:
```
data.size
```

Out[7]:
```
4335272
```

In [8]:
```
data.describe()
```

Out[8]:

| | Quantity | UnitPrice | CustomerID |
|---|---|---|---|
| **count** | 541909.000000 | 541909.000000 | 406829.000000 |
| **mean** | 9.552250 | 4.611114 | 15287.690570 |
| **std** | 218.081158 | 96.759853 | 1713.600303 |
| **min** | -80995.000000 | -11062.060000 | 12346.000000 |
| **25%** | 1.000000 | 1.250000 | 13953.000000 |
| **50%** | 3.000000 | 2.080000 | 15152.000000 |
| **75%** | 10.000000 | 4.130000 | 16791.000000 |
| **max** | 80995.000000 | 38970.000000 | 18287.000000 |

In [9]:
```
data.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 [10]:
```
data.isna().sum()
```

Out[10]:
```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

In [11]:
```
data.duplicated().value_counts()
```

Out[11]:
```
False    536641
True       5268
dtype: int64
```

## Handling Null Values

In [13]:
```
data.dropna(subset=['CustomerID'], inplace=True)
```

In [14]:
```
data.isna().sum()
```

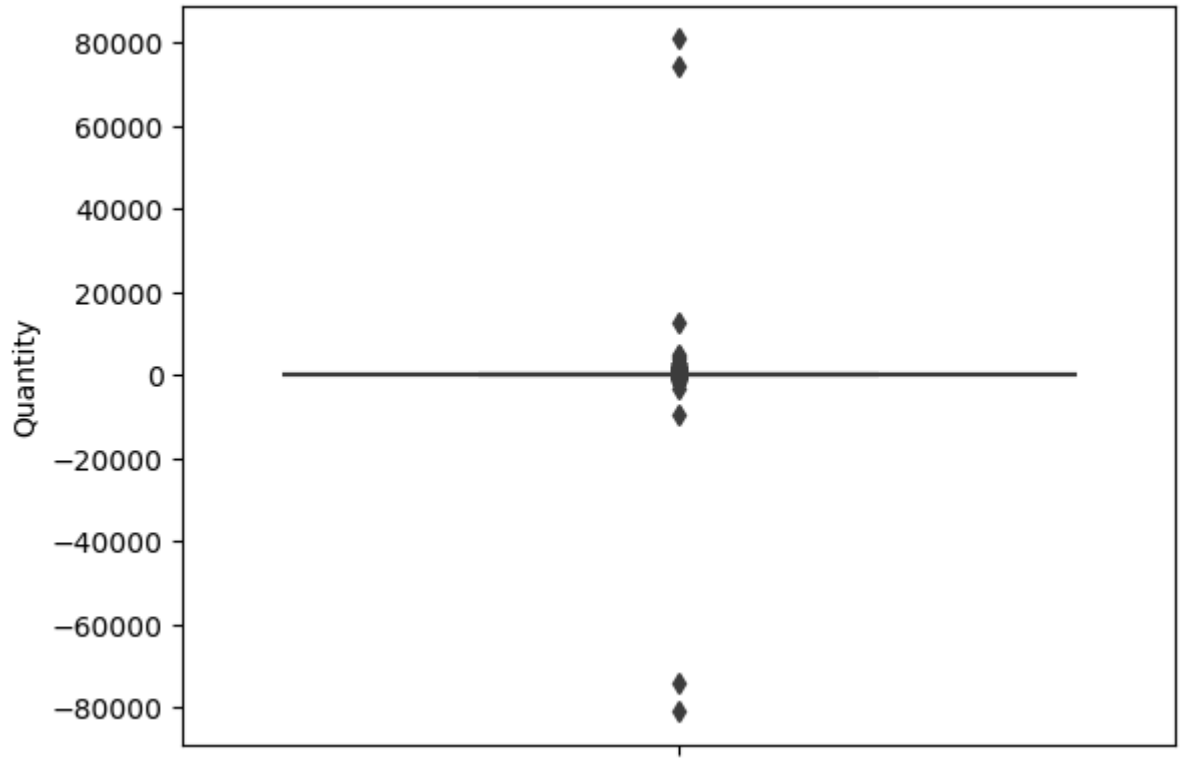
Out[14]:
```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

In [15]:
```
data.shape
```

Out[15]:
```
(406829, 8)
```

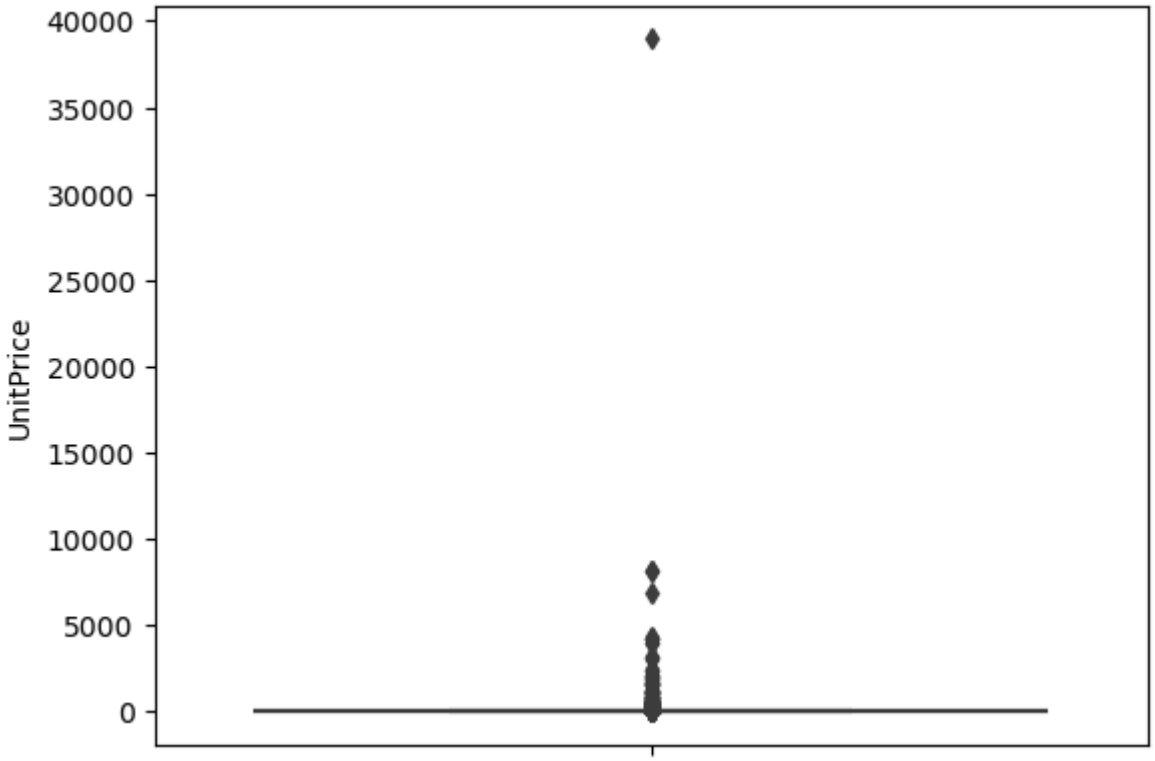
## Outlier Detection & Removal of Outliers

In [17]:
```
sns.boxplot(data = data, y = 'Quantity')
plt.show()
```




In [18]:
```
data = data[~((data['Quantity'] > 6000) | (data['Quantity'] < -6000))]
```

In [19]:
```
sns.boxplot(data = data, y = 'UnitPrice')
plt.show()
```




In [20]:
```
data = data[~(data['UnitPrice'] > 35000)]
```

## Changing the Data type of Date column

In [22]:
```
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
```

In [23]:
```
data['InvoiceDate'] = data['InvoiceDate'].dt.date
```

In [24]:
```
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
```

## Creating Some feature Columns

In [26]:
```
data['Month'] = data['InvoiceDate'].dt.month
```

In [27]:
```
data['days'] = data['InvoiceDate'].dt.day
```

In [28]:
```
data['day_of_week'] = data['InvoiceDate'].dt.dayofweek
```

In [29]:
```
data['Cancellation'] = np.where(data['Quantity'] < 0, 1, 0)
```

In [30]:
```
data["week"] = pd.to_datetime(data["InvoiceDate"]).dt.isocalendar().week
data["year"]= pd.to_datetime(data["InvoiceDate"]).dt.year
data['month_year'] = data['InvoiceDate'].dt.strftime('%b_%Y')
```

In [31]:
```
data['Total_Amount'] = data['Quantity'] * data['UnitPrice']
```

In [32]:
```
data.head()
```

Out[32]:

| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | Month | days | day_of_week | Cancellation | week | year | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 | 2.55 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **1** | 536365 | 71053 | WHITE METAL LANTERN | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **2** | 536365 | 84406B | CREAM CUPID HEARTS COAT HANGER | 8 | 2010-12-01 | 2.75 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **3** | 536365 | 84029G | KNITTED UNION FLAG HOT WATER BOTTLE | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **4** | 536365 | 84029E | RED WOOLLY HOTTIE WHITE HEART. | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |

## Non Grpahical Analysis

In [34]:

```
column = ['Description','Country','day_of_week','Month','Cancellation','week','year','month_year','days']
for i in column:
  print(data.value_counts(i))
  print('\n')
```

```
Description
WHITE HANGING HEART T-LIGHT HOLDER    2070
REGENCY CAKESTAND 3 TIER              1905
JUMBO BAG RED RETROSPOT               1662
ASSORTED COLOUR BIRD ORNAMENT         1418
PARTY BUNTING                         1416
                                      ...
BLUE NEW BAROQUE FLOCK CANDLESTICK       1
IVORY SHELL HEART EARRINGS               1
IVORY PANTRY HANGING LAMP                1
BLUE PAINTED KASHMIRI CHAIR              1
MINT DINER CLOCK                         1
Length: 3895, dtype: int64


Country
United Kingdom          361871
Germany                   9495
France                    8491
EIRE                      7485
Spain                     2533
Netherlands               2371
Belgium                   2069
Switzerland               1877
Portugal                  1480
Australia                 1259
Norway                    1086
Italy                      803
Channel Islands            758
Finland                    695
Cyprus                     622
Sweden                     462
Austria                    401
Denmark                    389
Japan                      358
Poland                     341
USA                        291
Israel                     250
Unspecified                244
Singapore                  229
Iceland                    182
Canada                     151
Greece                     146
Malta                      127
```

```
United Arab Emirates        68
European Community          61
RSA                         58
Lebanon                     45
Lithuania                   35
Brazil                      32
Czech Republic              30
Bahrain                     17
Saudi Arabia                10
dtype: int64


day_of_week
3    82373
2    70599
1    68108
0    66382
6    63237
4    56123
dtype: int64


Month
11    65597
10    50695
12    44508
9     40822
5     28908
6     27835
3     27822
8     27662
7     27502
4     23198
1     21910
2     20363
dtype: int64


Cancellation
0    397921
1      8901
dtype: int64
```

| week | |
|---|---|
| 48 | 21762 |
| 49 | 21316 |
| 46 | 16320 |
| 47 | 15244 |
| 45 | 14758 |
| 44 | 13393 |
| 43 | 12441 |
| 40 | 12233 |
| 42 | 11790 |
| 38 | 11079 |
| 41 | 10970 |
| 39 | 10316 |
| 37 | 8744 |
| 36 | 8240 |
| 20 | 7816 |
| 19 | 7580 |
| 34 | 7512 |
| 23 | 7268 |
| 29 | 7263 |
| 50 | 7245 |
| 24 | 6863 |
| 33 | 6730 |
| 13 | 6669 |
| 31 | 6626 |
| 15 | 6624 |
| 12 | 6587 |
| 28 | 6460 |
| 30 | 6345 |
| 14 | 6337 |
| 25 | 6262 |
| 18 | 6250 |
| 11 | 6178 |
| 27 | 5956 |
| 21 | 5887 |
| 8 | 5868 |
| 4 | 5845 |
| 9 | 5703 |
| 10 | 5533 |
| 32 | 5518 |
| 1 | 5420 |
| 7 | 5341 |
| 5 | 5183 |
| 35 | 5142 |

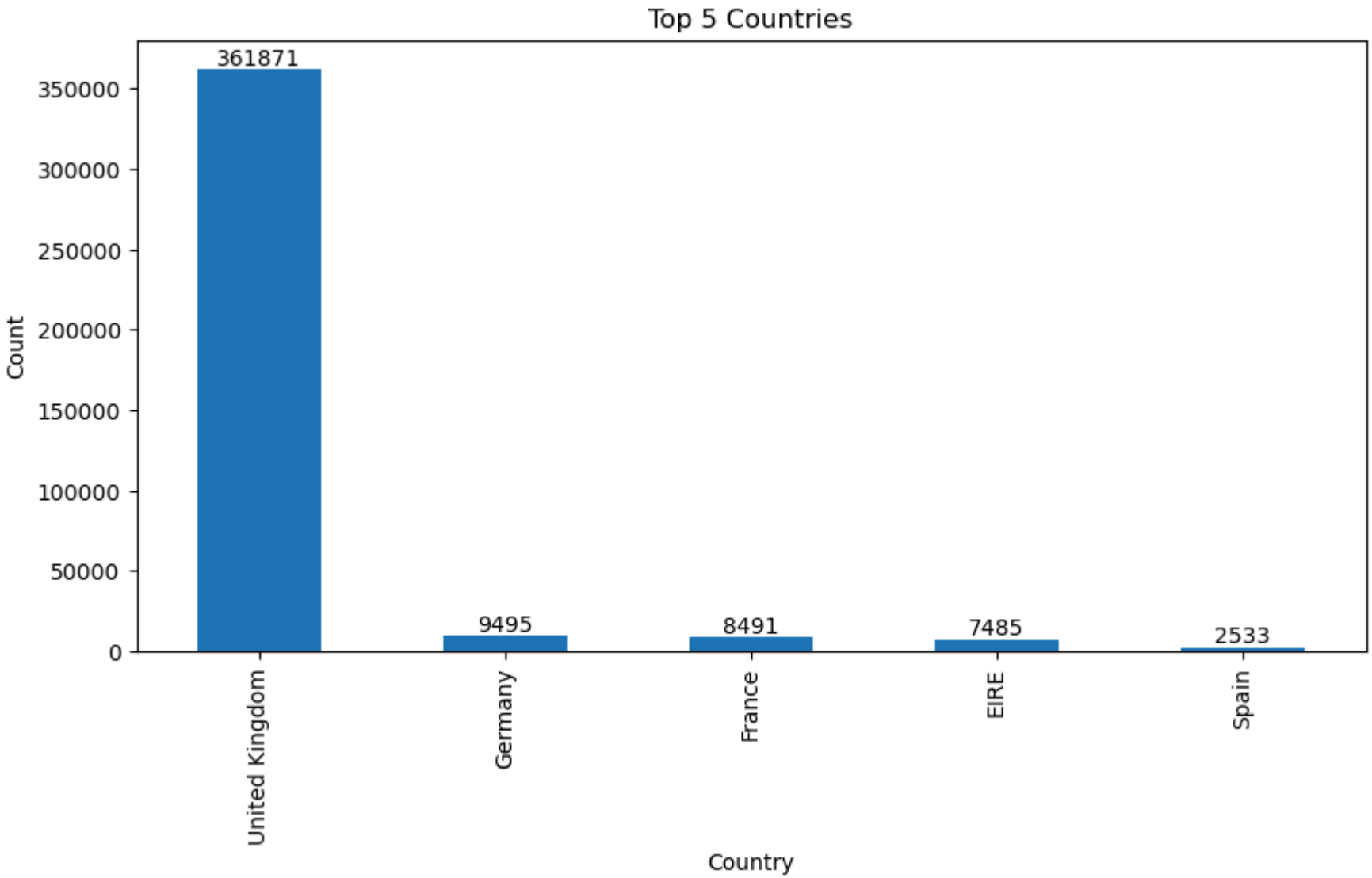
```
16     5028
22     4951
26     4892
2      4813
3      4796
6      4025
17     3795
51     1905
dtype: int64


year
2011    379973
2010     26849
dtype: int64


month_year
Nov_2011    65597
Oct_2011    50695
Sep_2011    40822
May_2011    28908
Jun_2011    27835
Mar_2011    27822
Aug_2011    27662
Jul_2011    27502
Dec_2010    26849
Apr_2011    23198
Jan_2011    21910
Feb_2011    20363
Dec_2011    17659
dtype: int64


days
6     18915
5     16723
8     16254
7     16011
4     15165
17    15139
20    14940
14    14549
23    14545
```

```
10    14472
13    14401
1     13937
28    13778
11    13707
21    13509
18    13232
9     13223
16    12910
27    12708
22    12583
24    12394
2     12379
25    12199
15    11611
19    11106
3     11069
12    10923
30    10221
26     8913
29     8263
31     7043
dtype: int64
```

## Graphical Analysis

## Top 5 Countries

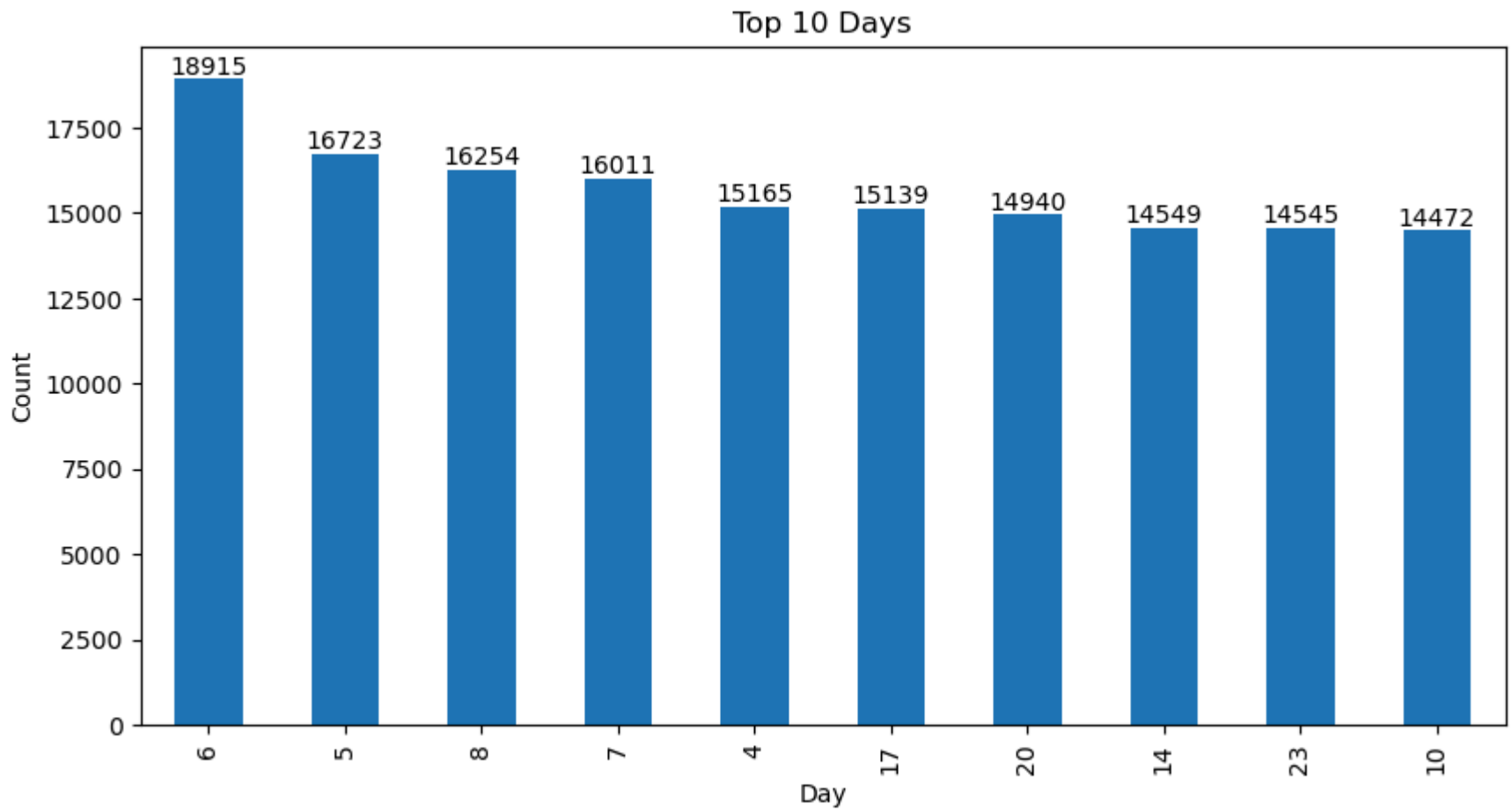
In [37]:
```
data['Country'].value_counts().head(5).plot(kind='bar',figsize=(10,5))
plt.title('Top 5 Countries')
plt.xlabel('Country')
plt.ylabel('Count')
ax=plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```




## Top 10 Days

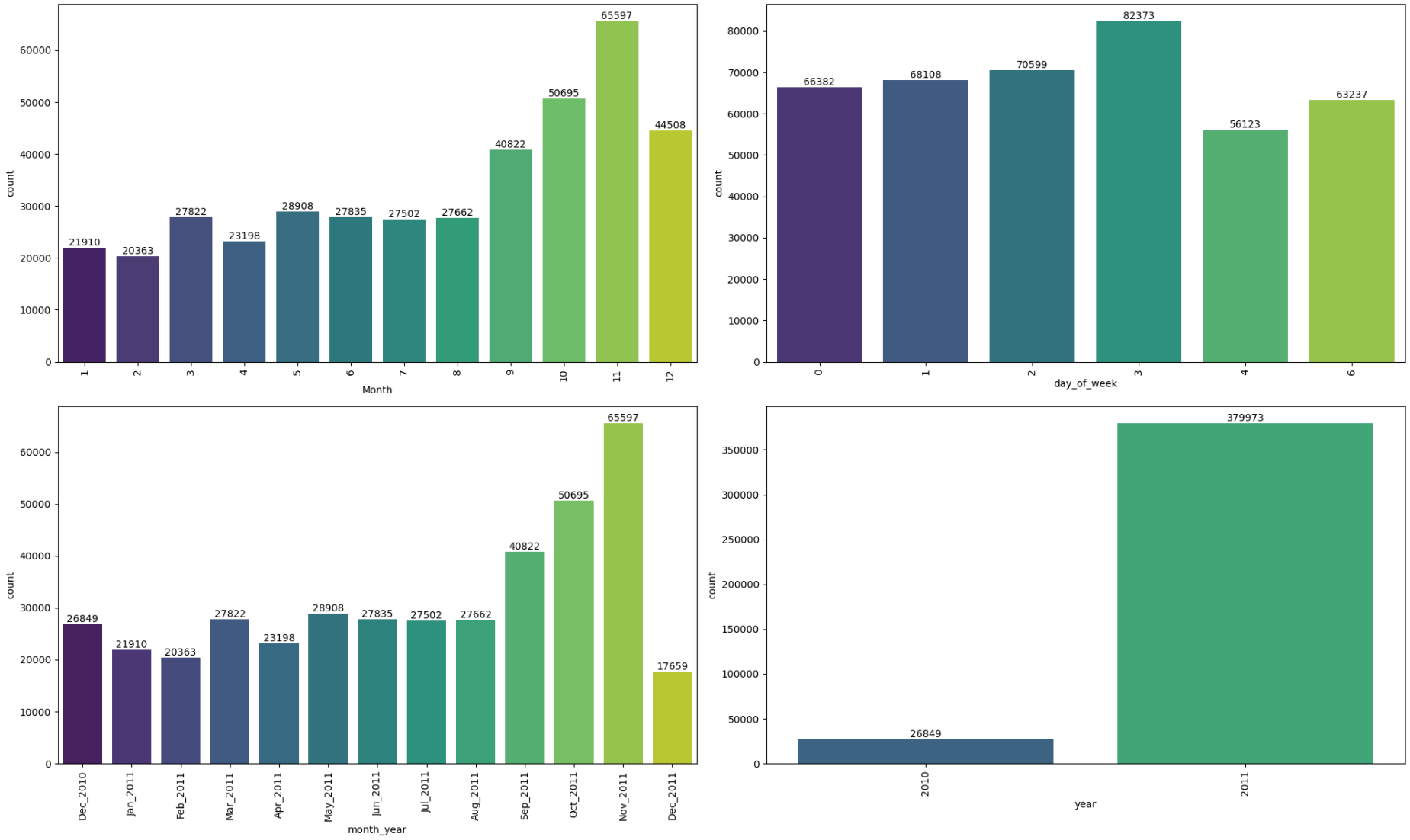
In [39]:
```
data['days'].value_counts().head(10).plot(kind='bar',figsize=(10,5))
plt.title('Top 10 Days')
plt.xlabel('Day')
plt.ylabel('Count')
ax=plt.gca()
```

```
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```



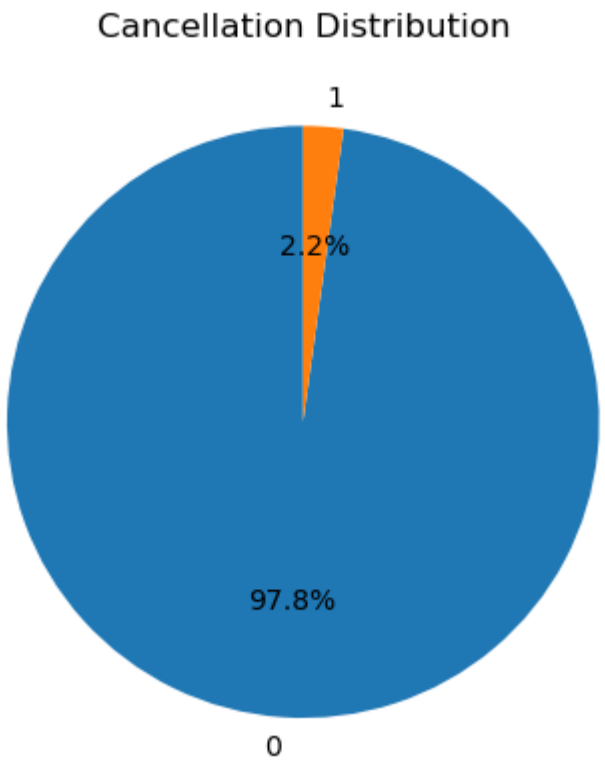

In [41]:
```
fig, axs = plt.subplots(nrows = 2, ncols = 2, figsize = (20,12))
sns.countplot(data = data, x = 'Month', ax = axs[0,0], palette = 'viridis')
sns.countplot(data = data, x = 'day_of_week', ax = axs[0,1], palette = 'viridis')
sns.countplot(data = data, x = 'month_year', ax = axs[1,0], palette = 'viridis')
sns.countplot(data = data, x = 'year', ax = axs[1,1], palette = 'viridis')
for ax in axs.flat:
    ax.tick_params(axis='x', rotation=90)
    for bar in ax.containers:
        ax.bar_label(bar)
```

```
plt.tight_layout()
plt.show()
```




In [42]:
```
cancellation_counts = data['Cancellation'].value_counts()
plt.pie(
    x=cancellation_counts,
    labels=cancellation_counts.index,
    autopct='%1.1f%%',
    startangle=90
```

```
)
plt.title('Cancellation Distribution')
plt.show()
```




## Top 5 most spending customers

In [44]:
```
data.groupby(['CustomerID','Country'])['Total_Amount'].sum().sort_values(ascending = False).head()
```

Out[44]:
```
CustomerID  Country
14646.0     Netherlands       279489.02
18102.0     United Kingdom    256438.49
17450.0     United Kingdom    187482.17
14911.0     EIRE              132572.62
12415.0     Australia         123725.45
Name: Total_Amount, dtype: float64
```

In [45]:
```
data.head(2)
```

Out[45]:

| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | Month | days | day_of_week | Cancellation | week | year | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 | 2.55 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **1** | 536365 | 71053 | WHITE METAL LANTERN | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |

## Top 5 product code which are highest selling

In [47]:
```
data.groupby(['StockCode','Country'])['Quantity'].sum().sort_values(ascending = False).head()
```

Out[47]:
```
StockCode  Country
84077      United Kingdom    47982
22197      United Kingdom    45217
85099B     United Kingdom    40880
84879      United Kingdom    32679
85123A     United Kingdom    32154
Name: Quantity, dtype: int64
```

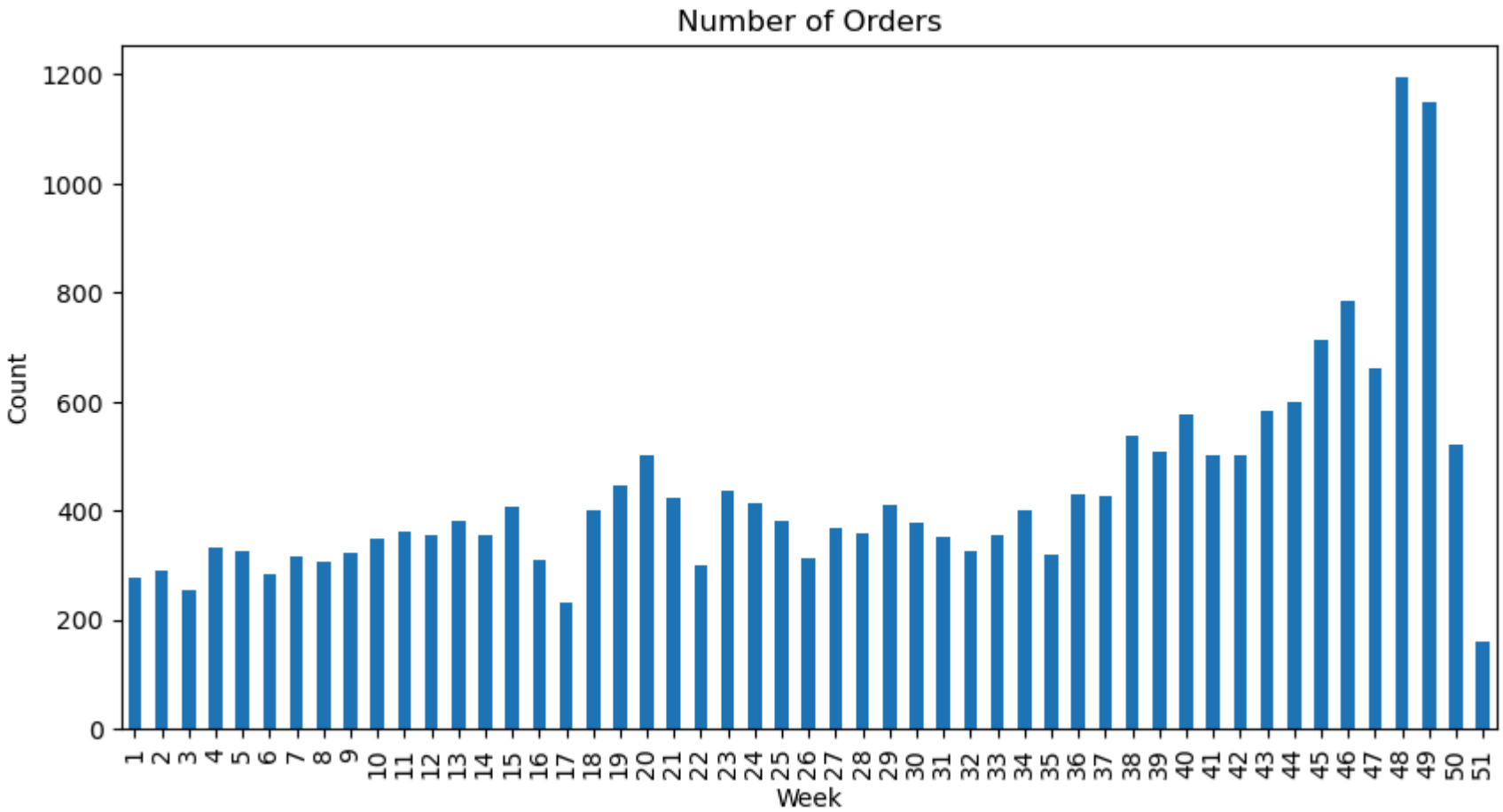
## Seasonality Trends

## Number of Order by Month

In [50]:
```
data.groupby('Month')['InvoiceNo'].nunique().plot(kind='bar',figsize=(10,5))
plt.title('Number of Orders')
plt.xlabel('Month')
plt.ylabel('Count')
ax=plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```

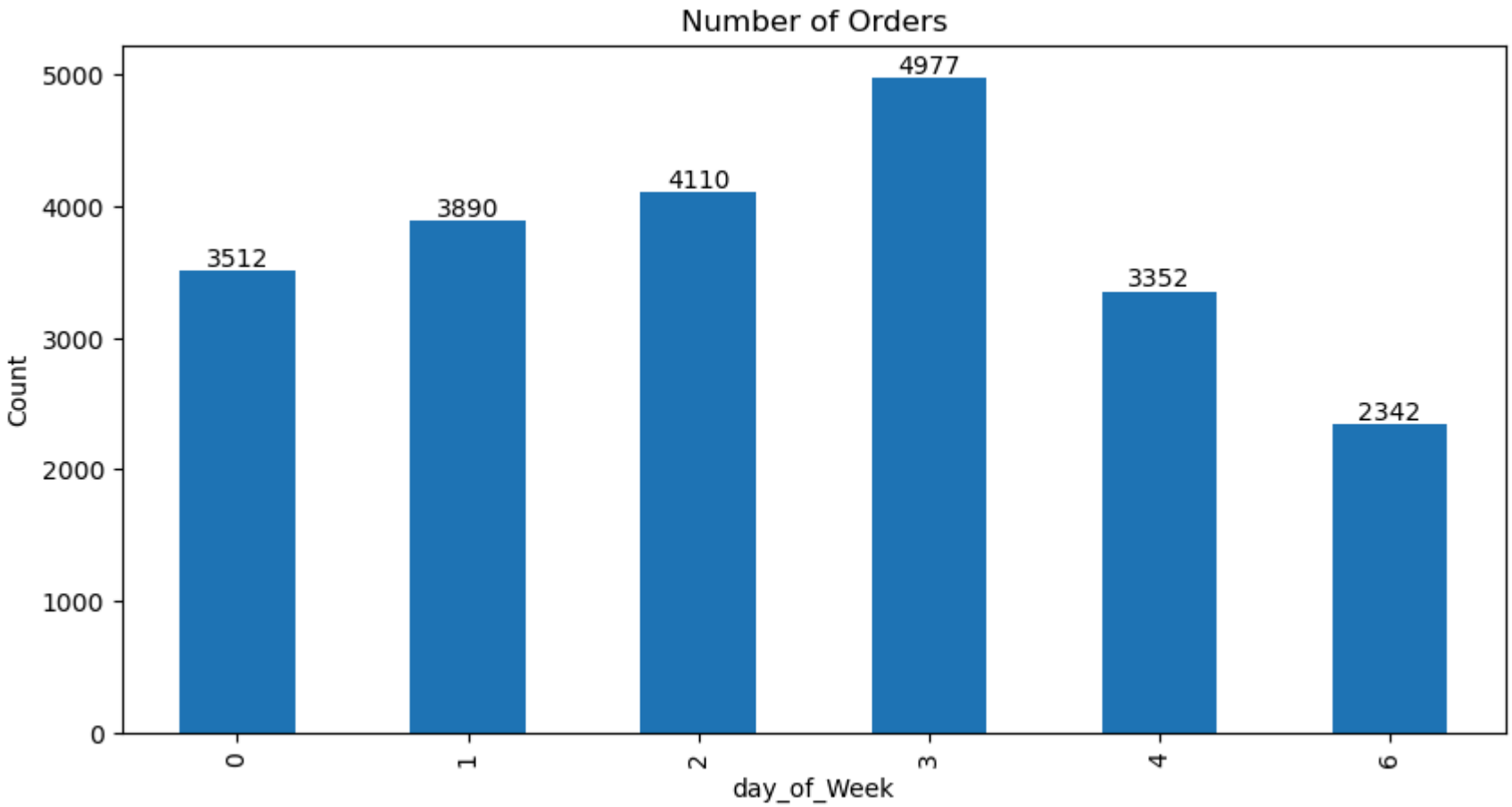

## Number of Order by Week

In [52]:
```
data.groupby('week')['InvoiceNo'].nunique().plot(kind='bar',figsize=(10,5))
plt.title('Number of Orders')
plt.xlabel('Week')
plt.ylabel('Count')
plt.show()
```




## Number of order by Day of Week

In [54]:
```
data.groupby('day_of_week')['InvoiceNo'].nunique().plot(kind='bar',figsize=(10,5))
plt.title('Number of Orders')
plt.xlabel('day_of_Week')
plt.ylabel('Count')
ax = plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```



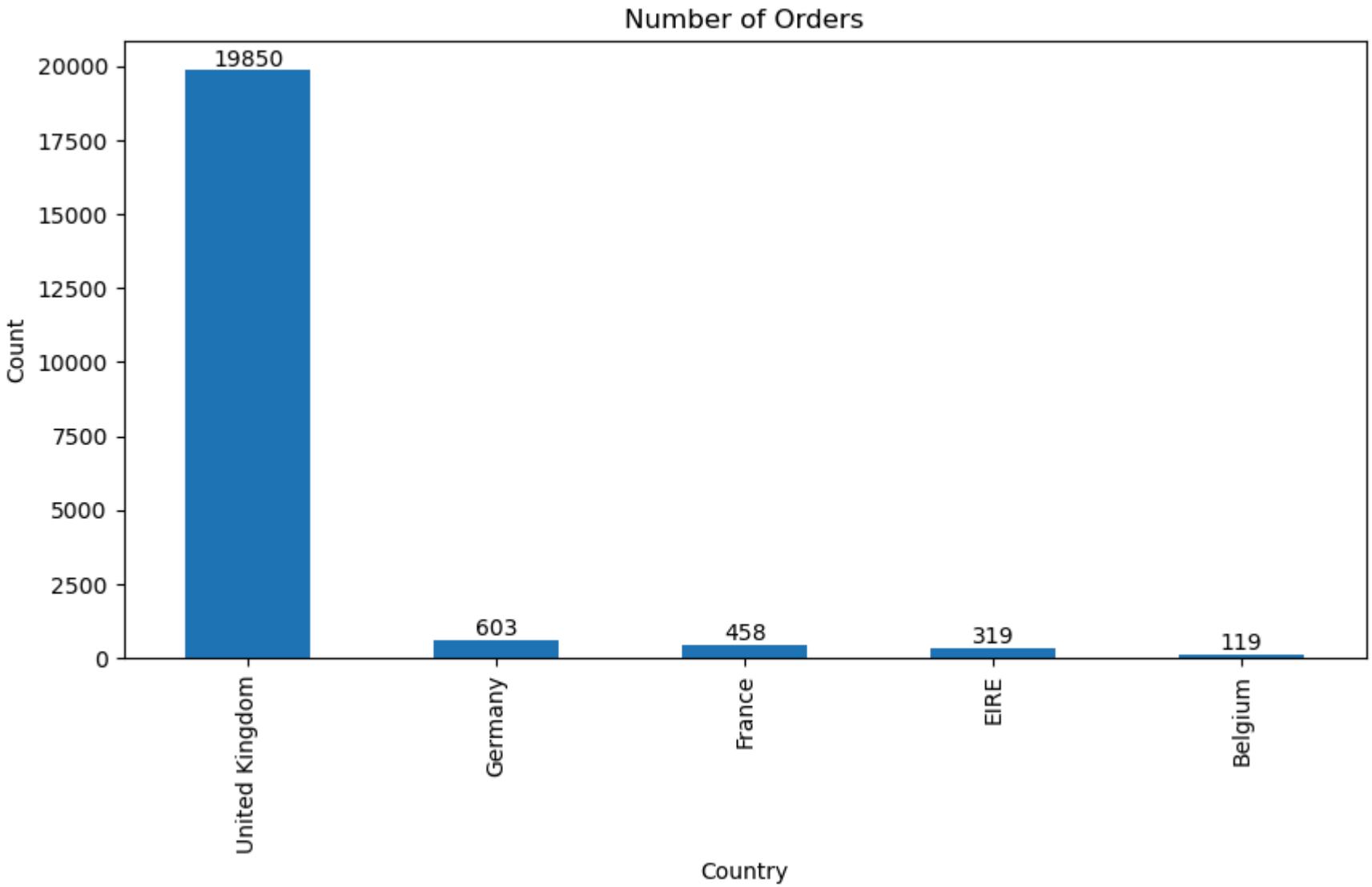

## Numbers of orders by Country

In [56]:
```
data.groupby('Country')['InvoiceNo'].nunique().sort_values(ascending = False).head(5)
```

Out[56]:
```
Country
United Kingdom    19850
Germany             603
France              458
EIRE                319
Belgium             119
Name: InvoiceNo, dtype: int64
```

In [57]:

```
data.groupby('Country')['InvoiceNo'].nunique().sort_values(ascending = False).head(5).plot(kind='bar',figsize=(10,5))
plt.title('Number of Orders')
plt.xlabel('Country')
plt.ylabel('Count')
ax=plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```




## Top 10 order Products

In [59]:
```
orders=data.groupby("Description",as_index=False)['InvoiceNo'].nunique()
orders.sort_values(by="InvoiceNo",ascending=False).head(10)
```

Out[59]:

| | Description | InvoiceNo |
|---|---|---|
| **3715** | WHITE HANGING HEART T-LIGHT HOLDER | 2013 |
| **2779** | REGENCY CAKESTAND 3 TIER | 1884 |
| **1771** | JUMBO BAG RED RETROSPOT | 1643 |
| **2354** | PARTY BUNTING | 1399 |
| **217** | ASSORTED COLOUR BIRD ORNAMENT | 1385 |
| **1953** | LUNCH BAG RED RETROSPOT | 1329 |
| **3013** | SET OF 3 CAKE TINS PANTRY DESIGN | 1218 |
| **2623** | POSTAGE | 1194 |
| **1945** | LUNCH BAG BLACK SKULL. | 1073 |
| **2279** | PACK OF 72 RETROSPOT CAKE CASES | 1041 |

## Average Order Value by Month

In [61]:
```
data.groupby('month_year')['Total_Amount'].mean().round(2)
```

Out[61]:
```
month_year
Apr_2011    18.37
Aug_2011    22.28
Dec_2010    20.67
Dec_2011    19.40
Feb_2011    21.44
Jan_2011    21.68
Jul_2011    20.88
Jun_2011    23.24
Mar_2011    20.85
May_2011    22.42
Nov_2011    17.26
Oct_2011    19.22
Sep_2011    22.82
Name: Total_Amount, dtype: float64
```

## Customer Acquisition Month on Month

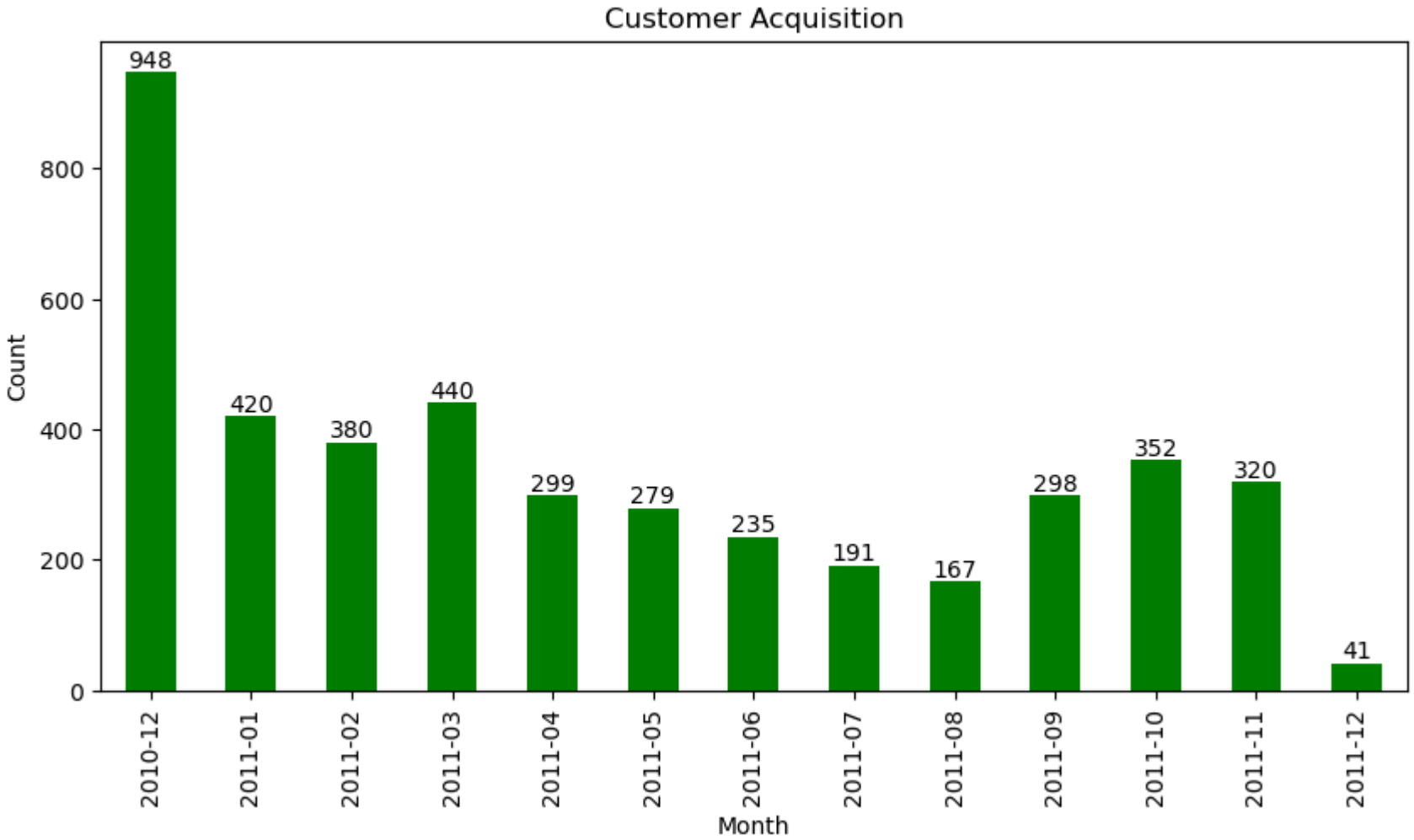
In [63]:
```
data['First_transaction_Date'] = data.groupby('CustomerID')['InvoiceDate'].transform('min')
```

In [64]:
```
data['first_Month'] = data['First_transaction_Date'].dt.to_period('M')
```

In [65]:
```
data.groupby('first_Month')['CustomerID'].nunique()
```

Out[65]:
```
first_Month
2010-12    948
2011-01    420
2011-02    380
2011-03    440
2011-04    299
2011-05    279
2011-06    235
2011-07    191
2011-08    167
2011-09    298
2011-10    352
2011-11    320
2011-12     41
Freq: M, Name: CustomerID, dtype: int64
```

In [66]:
```
data.groupby('first_Month')['CustomerID'].nunique().plot(kind='bar',color='green',figsize=(10,5))
plt.title('Customer Acquisition')
plt.xlabel('Month')
plt.ylabel('Count')
ax=plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```

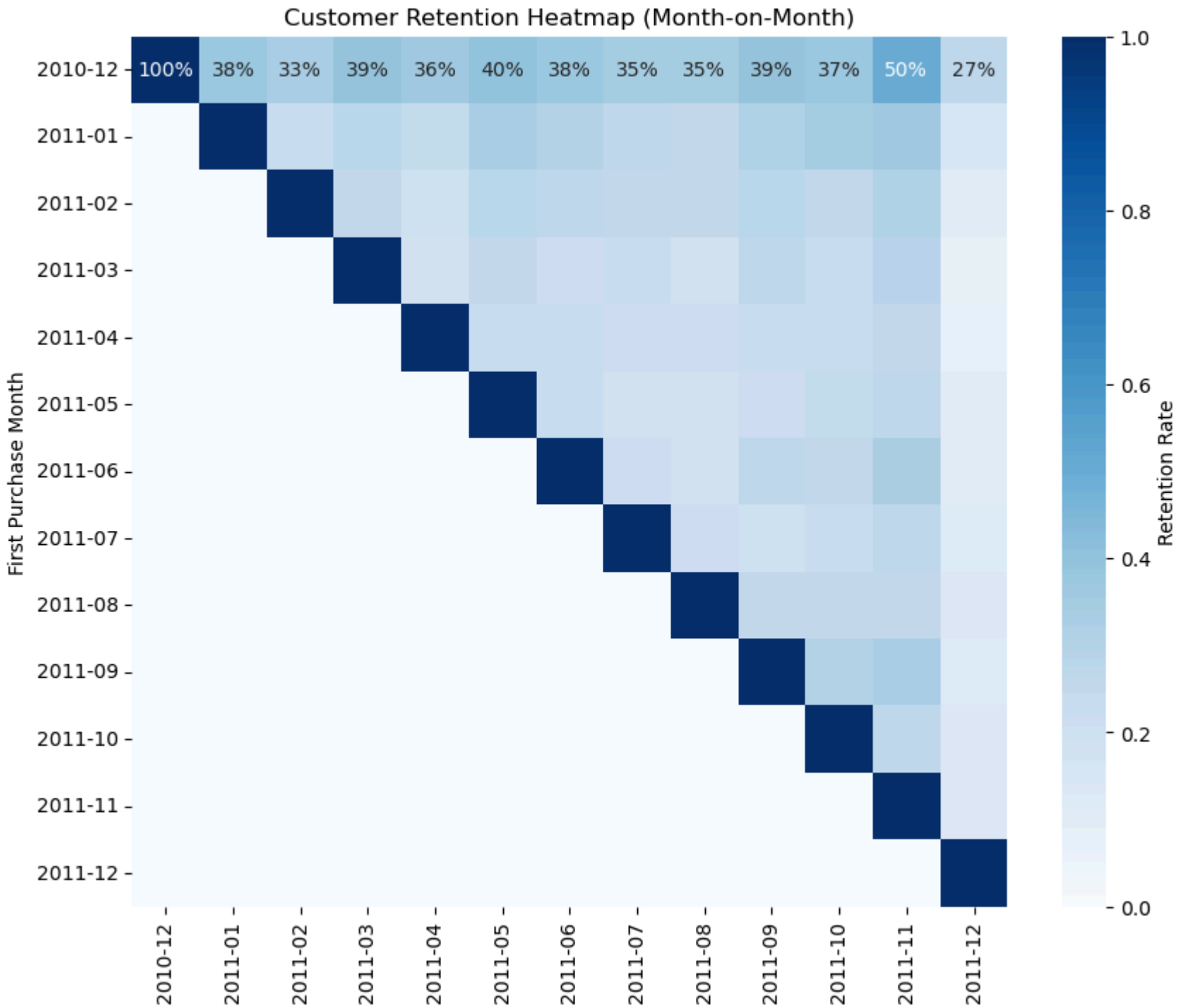



## Customer Retention Month on Month

In [68]:
```
data['Invoicemonthyear'] = data['InvoiceDate'].dt.to_period('M')
```

In [69]:
```
retention = data.pivot_table(
    index='first_Month',
    columns='Invoicemonthyear',
    values='CustomerID',
    aggfunc='nunique',
    fill_value=0
```

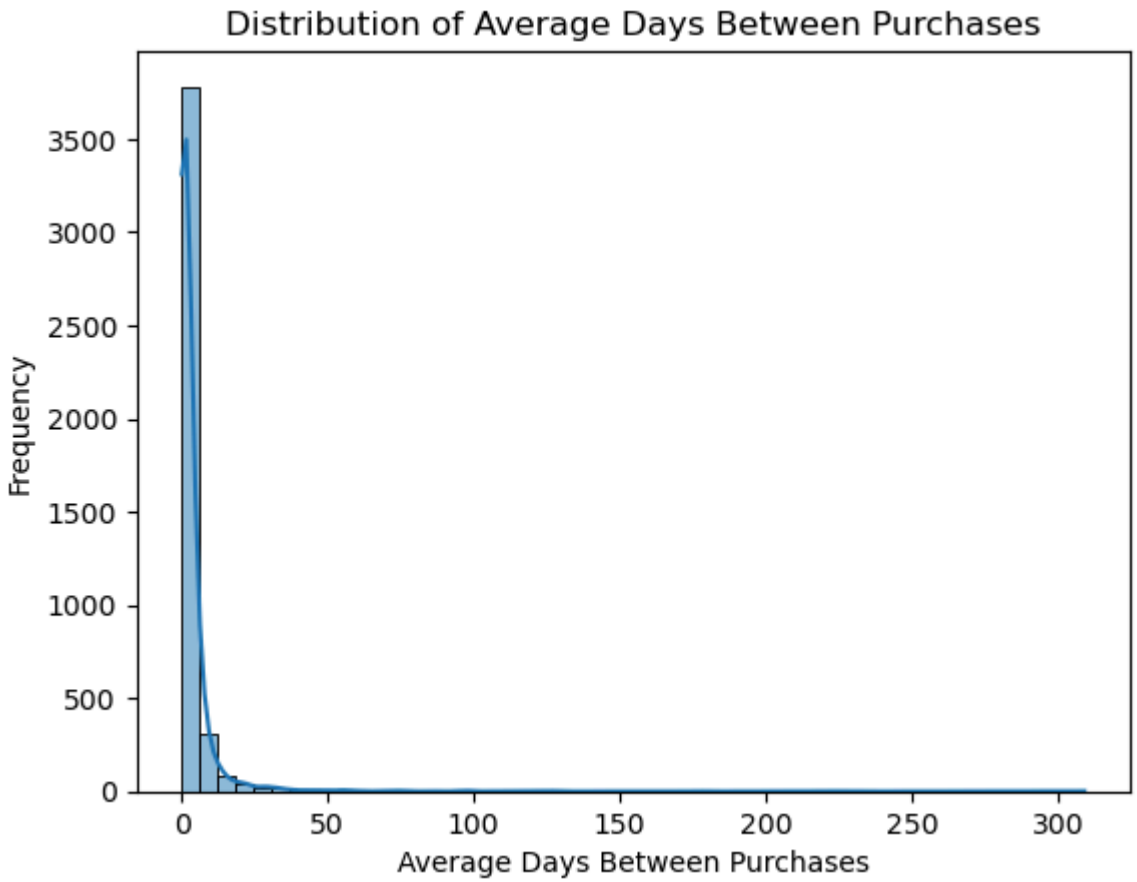
```
)


retention_rate = retention.div(retention.max(axis=1), axis=0)


plt.figure(figsize=(10, 8))
sns.heatmap(retention_rate, annot=True, fmt=".0%", cmap="Blues", cbar_kws={'label': 'Retention Rate'})
plt.title("Customer Retention Heatmap (Month-on-Month)")
plt.ylabel("First Purchase Month")
plt.xlabel("Invoice Month")
plt.show()
```


Customer Retention Heatmap (Month-on-Month)
First Purchase Month
2010-12
2011-01
2011-02
2011-03
2011-04
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2011-12
100% 38% 33% 39% 36% 40% 38% 35% 35% 39% 37% 50% 27%
Retention Rate
1.0
0.8
0.6
0.4
0.2
0.0

## Average Days between Purchases

In [71]:
```
df = data.sort_values(by=['CustomerID', 'InvoiceDate'])

df['DaysBetweenPurchases'] = df.groupby('CustomerID')['InvoiceDate'].diff().dt.days

customer_metrics = df.groupby('CustomerID')['DaysBetweenPurchases'].mean().reset_index()
customer_metrics.rename(columns={'DaysBetweenPurchases': 'AvgDaysBetweenPurchases'}, inplace=True)
```

In [153…
```
sns.histplot(customer_metrics['AvgDaysBetweenPurchases'],bins = 50, kde=True)
plt.xlabel('Average Days Between Purchases')
plt.ylabel('Frequency')
plt.title('Distribution of Average Days Between Purchases')
plt.show()
```




## RFM Analysis

In [74]:
```
today = data['InvoiceDate'].max()
```

In [75]:
```
segment = data.groupby('CustomerID').agg({'InvoiceDate': lambda x: (today - x.max()).days,
                                          'InvoiceNo' : 'count',
                                          'Total_Amount' : 'sum',
                                          'Quantity': 'sum'})
segment.rename(columns = {'InvoiceDate':'Recency', 'InvoiceNo':'Frequency', 'Total_Amount':'Monetory'}, inplace = True)
segment.reset_index(inplace = True)
```

In [76]:
```
labels = [1,2,3,4,5]
segment['recency_bin'] = pd.qcut(segment['Recency'], q=5, labels = labels)
```

```
labels = [5,4,3,2,1]
segment['frequency_bin'] = pd.qcut(segment['Frequency'], q = 5, labels = labels)

labels = [5,4,3,2,1]
segment['monetory_bin'] = pd.qcut(segment['Monetory'], q = 5, labels = labels)
```

In [77]:
```
segment['recency_bin'] = segment['recency_bin'].astype('int')
segment['frequency_bin'] = segment['frequency_bin'].astype('int')
segment['monetory_bin'] = segment['monetory_bin'].astype('int')
```

In [78]:
```
segment['RFM'] = segment['recency_bin'] + segment['frequency_bin'] + segment['monetory_bin']
```

In [79]:
```
def rfm_analysis(rfm):
    if rfm >= 13:
        return 'Premium'
    elif rfm > 7 and rfm < 13:
        return 'Gold'
    else:
        return 'Silver'

segment['Customer_segmentation'] = segment['RFM'].apply(rfm_analysis)
segment.head()
```

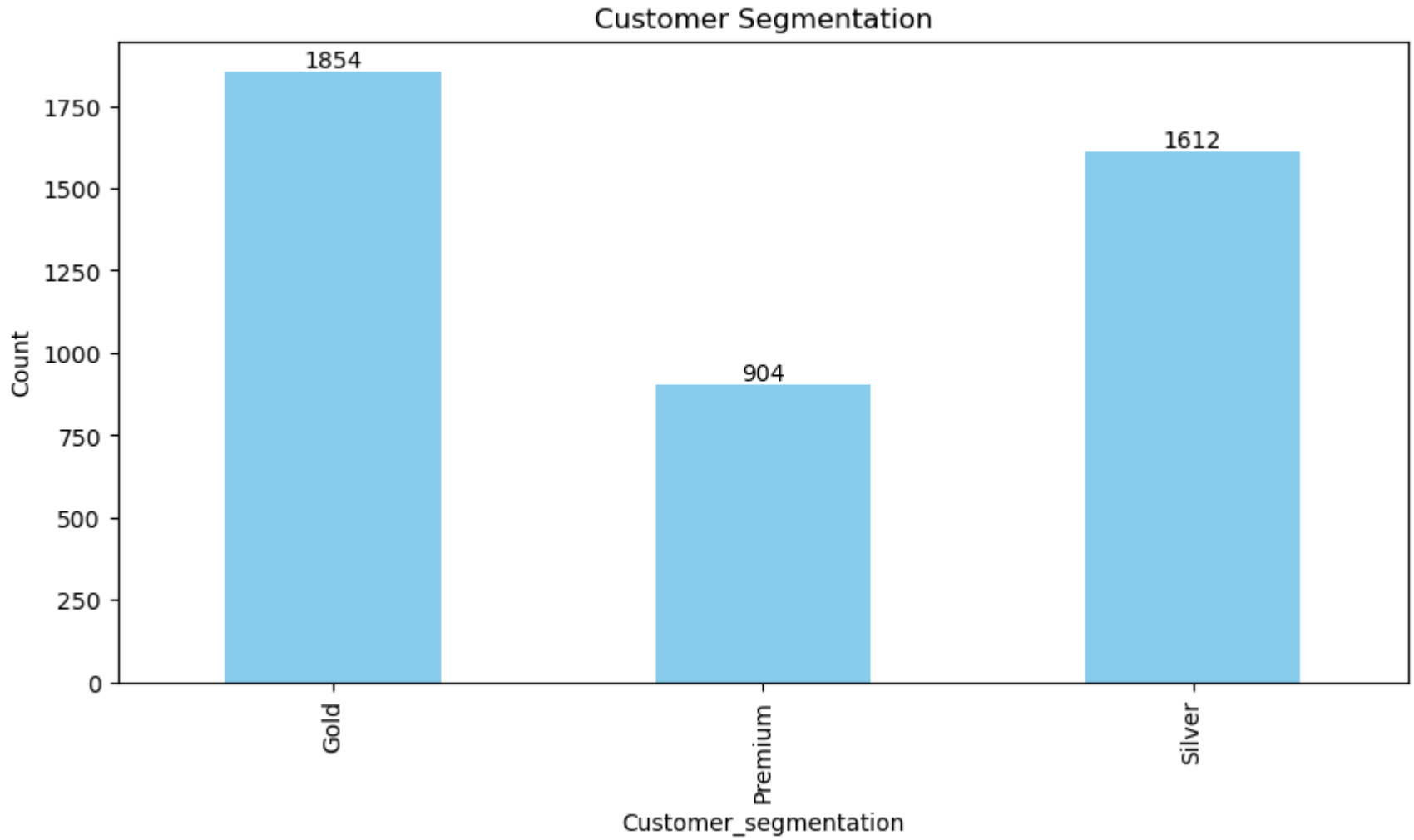
Out[79]:

| | CustomerID | Recency | Frequency | Monetory | Quantity | recency_bin | frequency_bin | monetory_bin | RFM | Customer_segmentation |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 12347.0 | 2 | 182 | 4310.00 | 2458 | 1 | 1 | 1 | 3 | Silver |
| **1** | 12348.0 | 75 | 31 | 1797.24 | 2341 | 4 | 3 | 2 | 9 | Gold |
| **2** | 12349.0 | 18 | 73 | 1757.55 | 631 | 2 | 2 | 2 | 6 | Silver |
| **3** | 12350.0 | 310 | 17 | 334.40 | 197 | 5 | 4 | 4 | 13 | Premium |
| **4** | 12352.0 | 36 | 95 | 1545.41 | 470 | 3 | 2 | 2 | 7 | Silver |

In [80]:
```
segment.groupby('Customer_segmentation')['CustomerID'].count()
```

Out[80]:
```
Customer_segmentation
Gold       1854
Premium     904
Silver     1612
Name: CustomerID, dtype: int64
```

In [81]:
```
segment.groupby('Customer_segmentation')['CustomerID'].count().plot(kind='bar', color = 'skyblue',figsize=(10,5))
plt.title('Customer Segmentation')
plt.xlabel('Customer_segmentation')
plt.ylabel('Count')
ax=plt.gca()
for bar in ax.containers:
    ax.bar_label(bar)
plt.show()
```




In [82]:
```
data.head()
```

Out[82]:

| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | Month | days | day_of_week | Cancellation | week | year | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 | 2.55 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **1** | 536365 | 71053 | WHITE METAL LANTERN | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **2** | 536365 | 84406B | CREAM CUPID HEARTS COAT HANGER | 8 | 2010-12-01 | 2.75 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **3** | 536365 | 84029G | KNITTED UNION FLAG HOT WATER BOTTLE | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |
| **4** | 536365 | 84029E | RED WOOLLY HOTTIE WHITE HEART. | 6 | 2010-12-01 | 3.39 | 17850.0 | United Kingdom | 12 | 1 | 2 | 0 | 48 | 2010 | |

## Market Basket Analysis

In [84]:
```
data = data[data['Quantity'] > 0]
```

In [85]:
```
basket = (data[data['Country'] =="France"].groupby(['InvoiceNo', 'Description'])['Quantity'].sum().unstack().reset_index().fillna
basket[basket > 0] = 1
basket.head()
```

Out[85]:

| Description | 50'S CHRISTMAS GIFT BAG LARGE | DOLLY GIRL BEAKER | I LOVE LONDON MINI BACKPACK | NINE DRAWER OFFICE TIDY | SET 2 TEA TOWELS I LOVE LONDON | SPACEBOY BABY GIFT SET | TRELLIS COAT RACK | 10 COLOUR SPACEBOY PEN | 12 COLOURED PARTY BALLOONS | 12 EGG HOUSE PAINTED WOOD | ... | WRAP SUKI AND FRIENDS | WRAP VINTAGE PETALS DESIGN | YEL ( I F FASI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **InvoiceNo** | | | | | | | | | | | | | | |
| **536370** | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | |
| **536852** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | |
| **536974** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | |
| **537065** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | |
| **537463** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | |

5 rows × 1544 columns

In [86]:
```
frequent_basket = apriori(basket, min_support = 0.07, use_colnames=True)
association_df = association_rules(frequent_basket, metric = 'lift', min_threshold = 0.5)
association_df
```

Out[86]:

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction | zhangs_metric |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | (ALARM CLOCK BAKELIKE GREEN) | (ALARM CLOCK BAKELIKE PINK) | 0.097686 | 0.102828 | 0.074550 | 0.763158 | 7.421711 | 0.064505 | 3.788061 | 0.958935 |
| **1** | (ALARM CLOCK BAKELIKE PINK) | (ALARM CLOCK BAKELIKE GREEN) | 0.102828 | 0.097686 | 0.074550 | 0.725000 | 7.421711 | 0.064505 | 3.281140 | 0.964430 |
| **2** | (ALARM CLOCK BAKELIKE RED ) | (ALARM CLOCK BAKELIKE GREEN) | 0.095116 | 0.097686 | 0.079692 | 0.837838 | 8.576814 | 0.070400 | 5.564267 | 0.976265 |
| **3** | (ALARM CLOCK BAKELIKE GREEN) | (ALARM CLOCK BAKELIKE RED ) | 0.097686 | 0.095116 | 0.079692 | 0.815789 | 8.576814 | 0.070400 | 4.912229 | 0.979046 |
| **4** | (POSTAGE) | (ALARM CLOCK BAKELIKE GREEN) | 0.771208 | 0.097686 | 0.084833 | 0.110000 | 1.126053 | 0.009496 | 1.013836 | 0.489275 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **137** | (SET/6 RED SPOTTY PAPER CUPS, SET/6 RED SPOTTY... | (POSTAGE, SET/20 RED RETROSPOT PAPER NAPKINS ) | 0.123393 | 0.110540 | 0.082262 | 0.666667 | 6.031008 | 0.068622 | 2.668380 | 0.951613 |
| **138** | (POSTAGE) | (SET/6 RED SPOTTY PAPER PLATES, SET/6 RED SPOT... | 0.771208 | 0.100257 | 0.082262 | 0.106667 | 1.063932 | 0.004943 | 1.007175 | 0.262640 |
| **139** | (SET/20 RED RETROSPOT PAPER NAPKINS ) | (POSTAGE, SET/6 RED SPOTTY PAPER CUPS, SET/6 R... | 0.133676 | 0.102828 | 0.082262 | 0.615385 | 5.984615 | 0.068517 | 2.332648 | 0.961424 |
| **140** | (SET/6 RED SPOTTY PAPER CUPS) | (POSTAGE, SET/6 RED SPOTTY PAPER PLATES, SET/2... | 0.138817 | 0.084833 | 0.082262 | 0.592593 | 6.985410 | 0.070486 | 2.246319 | 0.994963 |
| **141** | (SET/6 RED SPOTTY PAPER PLATES) | (POSTAGE, SET/6 RED SPOTTY PAPER CUPS, SET/20 ... | 0.128535 | 0.084833 | 0.082262 | 0.640000 | 7.544242 | 0.071358 | 2.542131 | 0.995391 |

142 rows × 10 columns

In [87]:
```
Ans=association_df[(association_df['lift'] >= 6) &(association_df['confidence'] >= 0.8)]
Ans[['antecedents','consequents']]
```

Out[87]:

| | antecedents | consequents |
|---|---|---|
| **2** | (ALARM CLOCK BAKELIKE RED ) | (ALARM CLOCK BAKELIKE GREEN) |
| **3** | (ALARM CLOCK BAKELIKE GREEN) | (ALARM CLOCK BAKELIKE RED ) |
| **78** | (SET/6 RED SPOTTY PAPER CUPS) | (SET/6 RED SPOTTY PAPER PLATES) |
| **79** | (SET/6 RED SPOTTY PAPER PLATES) | (SET/6 RED SPOTTY PAPER CUPS) |
| **80** | (ALARM CLOCK BAKELIKE RED , POSTAGE) | (ALARM CLOCK BAKELIKE GREEN) |
| **82** | (POSTAGE, ALARM CLOCK BAKELIKE GREEN) | (ALARM CLOCK BAKELIKE RED ) |
| **116** | (POSTAGE, SET/6 RED SPOTTY PAPER CUPS) | (SET/6 RED SPOTTY PAPER PLATES) |
| **117** | (POSTAGE, SET/6 RED SPOTTY PAPER PLATES) | (SET/6 RED SPOTTY PAPER CUPS) |
| **121** | (SET/6 RED SPOTTY PAPER PLATES) | (POSTAGE, SET/6 RED SPOTTY PAPER CUPS) |
| **122** | (SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO... | (SET/6 RED SPOTTY PAPER PLATES) |
| **123** | (SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET... | (SET/6 RED SPOTTY PAPER CUPS) |
| **124** | (SET/6 RED SPOTTY PAPER CUPS, SET/6 RED SPOTTY... | (SET/20 RED RETROSPOT PAPER NAPKINS ) |
| **128** | (POSTAGE, SET/6 RED SPOTTY PAPER CUPS, SET/20 ... | (SET/6 RED SPOTTY PAPER PLATES) |
| **129** | (POSTAGE, SET/6 RED SPOTTY PAPER PLATES, SET/2... | (SET/6 RED SPOTTY PAPER CUPS) |

## Insights

### Top Products & Customer Preferences

- Top-Selling Products:
  - The WHITE HANGING HEART T-LIGHT HOLDER and REGENCY CAKESTAND 3 TIER are the top-selling products across multiple countries, indicating high customer preference and demand for these items.

### Geographical Distribution

- Sales by Region:
  - Approximately 91% of the sales data is from the United Kingdom, showcasing it as the primary market for the business.

- Top Customers by Spending:
    - The highest spender is from the Netherlands, followed by two top spenders from the United Kingdom.

**Sales Trends**

- Weekly Sales Patterns:
    - Majority of sales occur on the 3rd day of the week (Tuesday), providing an opportunity to focus marketing efforts on this day to maximize revenue.
- Monthly Sales Patterns:
    - November 2011 recorded the highest sales, making it the peak month for customer transactions.
- Order Volumes:
    - The most orders were placed in Week 48 (late November/early December), emphasizing the importance of this period for strategic sales and inventory planning.

**Customer Acquisition & Retention**

- New Customer Acquisition:
    - The largest number of new customers were acquired in December 2010, followed by March 2011, suggesting effective onboarding strategies during these months.

**RFM Analysis Insights:**

- Based on RFM (Recency, Frequency, Monetary) analysis, customers are segmented into:
    - Gold: Majority of the customer base falls under this high-value category.
    - Silver: The second-largest segment.
    - Premium: A smaller but significant portion.

**Order Cancellations**

- Cancellation Insights:
    - Out of approximately 4 lakh orders, there have been around 9,000 cancellations, equating to a cancellation rate of ~2.25%. This low rate reflects strong customer satisfaction or operational efficiency.

## Recommendations

**Enhance Focus on Best-Selling Products**

- Increase Stock Availability: Ensure sufficient inventory for WHITE HANGING HEART T-LIGHT HOLDER and REGENCY CAKESTAND 3 TIER, especially during high-sales periods like November and Week 48.
- Promotional Campaigns: Design targeted campaigns featuring these products to further boost sales.

**Optimize Sales in the UK and Beyond**

- Tailored Marketing for the UK: Since 91% of sales are from the UK, create localized marketing strategies, such as region-specific discounts or promotions.
- Expand in the Netherlands: Given the high spending customer from the Netherlands, consider running exclusive offers or loyalty programs to strengthen the presence in this market.

**Leverage Weekly and Monthly Sales Trends**

- Tuesday Campaigns: Utilize the insight that most sales occur on Tuesdays to launch midweek flash sales or promotions.
- Focus on November and Week 48: Plan ahead for these periods by introducing attractive deals, festive bundles, and targeted marketing efforts to maximize revenue.

**Improve Customer Acquisition Strategies**

- Seasonal Onboarding: Analyze successful strategies from December 2010 and March 2011 to replicate them during other months, focusing on acquiring new customers during these peak onboarding periods.
- Retention Programs: Develop tailored loyalty programs for Gold, Silver, and Premium customers to enhance retention and increase customer lifetime value.

**Manage and Minimize Cancellations**

- Cancellation Insights: Investigate the reasons behind 2.25% cancellations and address them by improving customer service, optimizing return policies, and ensuring better product descriptions to manage expectations.

**Nurture High-Value Customers**

- Exclusive Benefits for Top Spenders: Offer personalized discounts or VIP perks to the top 3 customers to maintain their loyalty.
- Segment-Specific Offers: Develop unique benefits for Gold, Silver, and Premium customers, encouraging them to move to higher-value segments.

**Expand Data-Driven Marketing Efforts**

- RFM-Based Campaigns: Use the segmentation from RFM analysis to design personalized campaigns, focusing on the preferences of Gold customers while re-engaging lower-tier segments.
- Cross-Selling and Upselling: Recommend complementary products during peak sales periods to boost average transaction value.

In [ ]: