In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\User\anaconda3\Lib\site-packages\pandas\core\arrays\masked.p
y:60: UserWarning: Pandas requires version '1.3.6' or newer of 'bottle
neck' (version '1.3.5' currently installed).
  from pandas.core import (
```

In [4]:
```
df = pd.read_csv('Apriori Customer_Online_Transaction_data.csv')
df.head()
```

Out[4]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01/12/2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01/12/2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01/12/2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01/12/2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01/12/2010 08:26	3.39	17850.0	United Kingdom

In [5]:
```
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
```

In [6]:
```
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'], format='%d/%m/%Y %
```

In [7]:
```
df['InvoiceDate'].fillna(method='ffill', inplace=True)
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_6328\3208964462.py:1: Futur
eWarning: A value is trying to be set on a copy of a DataFrame or Seri
es through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values al
ways behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try usi
ng 'df.method({col: value}, inplace=True)' or df[col] = df[col].method
(value) instead, to perform the operation inplace on the original obje
ct.


  df['InvoiceDate'].fillna(method='ffill', inplace=True)
C:\Users\User\AppData\Local\Temp\ipykernel_6328\3208964462.py:1: Futur
eWarning: Series.fillna with 'method' is deprecated and will raise in
a future version. Use obj.ffill() or obj.bfill() instead.
  df['InvoiceDate'].fillna(method='ffill', inplace=True)
```

In [8]:
```
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  TotalPri
ce
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom       15.
30
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom       22.
00
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34
```

In [9]:
```
df['Recency'] = (df['InvoiceDate'].max() - df['InvoiceDate']).dt.days
```

In [10]:
```
frequency_df = df.groupby('CustomerID')['InvoiceNo'].count().reset_inde
frequency_df.rename(columns={'InvoiceNo': 'Frequency'}, inplace=True)
df = df.merge(frequency_df, on='CustomerID', how='left')
```

In [11]:
```
monetary_df = df.groupby('CustomerID')['TotalPrice'].sum().reset_index(
monetary_df.rename(columns={'TotalPrice': 'MonetaryValue'}, inplace=Tru
df = df.merge(monetary_df, on='CustomerID', how='left')
```

In [12]:
```
rfm_df = df[['CustomerID', 'Recency', 'Frequency', 'MonetaryValue']]
print(rfm_df.head())
print(df.head())
```

```
   CustomerID  Recency  Frequency  MonetaryValue
0     17850.0      373      312.0        5288.63
1     17850.0      373      312.0        5288.63
2     17850.0      373      312.0        5288.63
3     17850.0      373      312.0        5288.63
4     17850.0      373      312.0        5288.63
  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  TotalPri
ce  \
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom       15.
30
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom       22.
00
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom       20.
34

   Recency  Frequency  MonetaryValue
0      373      312.0        5288.63
1      373      312.0        5288.63
2      373      312.0        5288.63
3      373      312.0        5288.63
4      373      312.0        5288.63
```

In [13]:
```
france_df = df[df['Country'] == 'France']
```

In [14]:
```
print(france_df.head())
```

```
   InvoiceNo StockCode                      Description  Quantity  \
26    536370     22728       ALARM CLOCK BAKELIKE PINK        24
27    536370     22727       ALARM CLOCK BAKELIKE RED         24
28    536370     22726      ALARM CLOCK BAKELIKE GREEN        12
29    536370     21724  PANDA AND BUNNIES STICKER SHEET        12
30    536370     21883                STARS GIFT TAPE         24

           InvoiceDate  UnitPrice  CustomerID Country  TotalPrice  Rec
ency  \
26 2010-12-01 08:45:00       3.75     12583.0  France        90.0
373
27 2010-12-01 08:45:00       3.75     12583.0  France        90.0
373
28 2010-12-01 08:45:00       3.75     12583.0  France        45.0
373
29 2010-12-01 08:45:00       0.85     12583.0  France        10.2
373
30 2010-12-01 08:45:00       0.65     12583.0  France        15.6
373

    Frequency  MonetaryValue
26      251.0        7187.34
27      251.0        7187.34
28      251.0        7187.34
29      251.0        7187.34
30      251.0        7187.34
```

In [16]:
```
top_100_france = france_df.sort_values(by='MonetaryValue', ascending=Fa
top_100_france['Rank'] = range(1, len(top_100_france) + 1)
print(top_100_france)
```

```
       InvoiceNo StockCode                         Description  Quantity
\
186113    552851     22554  PLASTERS IN TIN WOODLAND ANIMALS        36
268297    560398     20725            LUNCH BAG RED RETROSPOT        30
186128    552851     21535       RED RETROSPOT SMALL MILK JUG        12
140554    548410     22991                GIRAFFE WOODEN RULER        60
140553    548410     22992             REVOLVER WOODEN RULER        60
...          ...       ...                                ...       ...
23280     538196     21915             RED  HARMONICA IN BOX        24
407340    571864     22988                  SOLDIERS EGG CUP        24
268267    560398     22139  RETROSPOT TEA SET CERAMIC 11 PC         24
268268    560398     22561       WOODEN SCHOOL COLOURING SET        24
268269    560398     22560        TRADITIONAL MODELLING CLAY        48

                InvoiceDate  UnitPrice  CustomerID Country  TotalPrice
\
186113 2011-05-11 15:07:00       1.65     12731.0  France       59.40
268297 2011-07-18 14:10:00       1.65     12731.0  France       49.50
186128 2011-05-11 15:07:00       2.55     12731.0  France       30.60
140554 2011-03-31 10:29:00       1.95     12731.0  France      117.00
140553 2011-03-31 10:29:00       1.95     12731.0  France      117.00
...                    ...        ...         ...     ...         ...
23280  2010-12-10 10:56:00       1.25     12731.0  France       30.00
407340 2011-10-19 13:49:00       1.25     12731.0  France       30.00
268267 2011-07-18 14:10:00       4.25     12731.0  France      102.00
268268 2011-07-18 14:10:00       1.65     12731.0  France       39.60
268269 2011-07-18 14:10:00       1.06     12731.0  France       50.88

        Recency  Frequency  MonetaryValue  Rank
186113      211      277.0       18793.41     1
268297      143      277.0       18793.41     2
186128      211      277.0       18793.41     3
140554      253      277.0       18793.41     4
140553      253      277.0       18793.41     5
...         ...        ...            ...   ...
23280       364      277.0       18793.41    96
407340       50      277.0       18793.41    97
268267      143      277.0       18793.41    98
268268      143      277.0       18793.41    99
268269      143      277.0       18793.41   100

[100 rows x 13 columns]
```

In [ ]:
```

```