Loading necessary library

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

Loading Dataset

In [3]:
```
df = pd.read_excel('Online Retail.xlsx')
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 2010-12-01 08:26:00       2.55     17850.0  United Kingdom
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
```

Data Processing

In [4]:
```
df = df.loc[df['Quantity'] > 0]
```

Identify null components

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 531285 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    531285 non-null  object
 1   StockCode    531285 non-null  object
 2   Description  530693 non-null  object
 3   Quantity     531285 non-null  int64
 4   InvoiceDate  531285 non-null  datetime64[ns]
 5   UnitPrice    531285 non-null  float64
 6   CustomerID   397924 non-null  float64
 7   Country      531285 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 36.5+ MB
```

Handling Nan CustomerID

In [6]:
```
df['CustomerID'].isna().sum()
df = df.dropna(subset=['CustomerID'])
```

Creating the customer-item matrix

In [7]:
```
customer_item_matrix = df.pivot_table(
    index='CustomerID',
    columns='StockCode',
    values='Quantity',
    aggfunc='sum'
```

```
)
customer_item_matrix.loc[12481:].head()
```

Out[7]:

StockCode	10002	10080	10120	10125	10133	10135	11001	15030	15034	15036	...	902
CustomerID												
12481.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	36.0	...	N
12483.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
12484.0	NaN	NaN	NaN	NaN	NaN	NaN	16.0	NaN	NaN	NaN	...	N
12488.0	NaN	NaN	NaN	NaN	NaN	10.0	NaN	NaN	NaN	NaN	...	N
12489.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N

5 rows × 3665 columns

In [8]:
```
print(customer_item_matrix.shape)
customer_item_matrix = customer_item_matrix.applymap(lambda x: 1 if x > 0 else 0)
```

```
(4339, 3665)
```

Collaborative Filtering

In [9]:
```
from sklearn.metrics.pairwise import cosine_similarity
```

User based collaborative filtering

In [10]:
```
user_user_sim_matrix = pd.DataFrame(cosine_similarity(customer_item_matrix))
user_user_sim_matrix.head()
```

Out[10]:

	0	1	2	3	4	5	6	7	8	9	...	43
0	1.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	...	(
1	0.0	1.000000	0.063022	0.046130	0.047795	0.038484	0.0	0.025876	0.136641	0.094742	...	(
2	0.0	0.063022	1.000000	0.024953	0.051709	0.027756	0.0	0.027995	0.118262	0.146427	...	(
3	0.0	0.046130	0.024953	1.000000	0.056773	0.137137	0.0	0.030737	0.032461	0.144692	...	(
4	0.0	0.047795	0.051709	0.056773	1.000000	0.031575	0.0	0.000000	0.000000	0.033315	...	(

5 rows × 4339 columns

In [11]:
```
#Renaming index and column names

user_user_sim_matrix.columns = customer_item_matrix.index

user_user_sim_matrix['CustomerID'] = customer_item_matrix.index
user_user_sim_matrix = user_user_sim_matrix.set_index('CustomerID')
user_user_sim_matrix.head()
```

Out[11]:

CustomerID	12346.0	12347.0	12348.0	12349.0	12350.0	12352.0	12353.0	12354.0	12355
CustomerID									
12346.0	1.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.0000(
12347.0	0.0	1.000000	0.063022	0.046130	0.047795	0.038484	0.0	0.025876	0.1366
12348.0	0.0	0.063022	1.000000	0.024953	0.051709	0.027756	0.0	0.027995	0.1182(
12349.0	0.0	0.046130	0.024953	1.000000	0.056773	0.137137	0.0	0.030737	0.0324(
12350.0	0.0	0.047795	0.051709	0.056773	1.000000	0.031575	0.0	0.000000	0.0000(

5 rows × 4339 columns

In [12]:
```
user_user_sim_matrix.loc[12350.0].sort_values(ascending=False).head(10)
```

Out[12]:
```
CustomerID
12350.0    1.000000
17935.0    0.183340
12414.0    0.181902
12652.0    0.175035
16692.0    0.171499
12791.0    0.171499
16754.0    0.171499
12814.0    0.171499
16426.0    0.166968
16333.0    0.161690
Name: 12350.0, dtype: float64
```

Making Recommendations

In [13]:
```
user_user_sim_matrix.loc[12350.0].sort_values(ascending=False)
items_bought_by_A = customer_item_matrix.loc[12350.0][customer_item_matrix.loc[123
print("Items Bought by A: ")
print(items_bought_by_A)
```

```
Items Bought by A:
StockCode
20615     1
20652     1
21171     1
21832     1
21864     1
21866     1
21908     1
21915     1
22348     1
22412     1
22551     1
22557     1
22620     1
79066K    1
79191C    1
84086C    1
POST      1
Name: 12350.0, dtype: int64
```

In [14]:
```
items_bought_by_B = customer_item_matrix.loc[17935.0][customer_item_matrix.loc[179
print("Items bought by B:")
print(items_bought_by_B)
```

```
print()

items_to_recommend_to_B = set(items_bought_by_A.index) - set(items_bought_by_B.ind
print("Items to Recommend to B ")
print(items_to_recommend_to_B)
df.loc[df['StockCode'].isin(items_to_recommend_to_B),['StockCode', 'Description']]
```

```
Items bought by B:
StockCode
20657     1
20659     1
20828     1
20856     1
21051     1
21866     1
21867     1
22208     1
22209     1
22210     1
22211     1
22449     1
22450     1
22551     1
22553     1
22557     1
22640     1
22659     1
22749     1
22752     1
22753     1
22754     1
22755     1
23290     1
23292     1
23309     1
85099B    1
POST      1
Name: 17935.0, dtype: int64

Items to Recommend to B
{20615, 21832, 21864, 22348, 20652, 22412, '84086C', '79066K', 21171, 21908, '7919
1C', 21915, 22620}
```

Out[14]:

StockCode	Description
21832	CHOCOLATE CALCULATOR
21915	RED HARMONICA IN BOX
22620	4 TRADITIONAL SPINNING TOPS
79066K	RETRO MOD TRAY
21864	UNION JACK FLAG PASSPORT COVER
79191C	RETRO PLASTIC ELEPHANT TRAY
21908	CHOCOLATE THIS WAY METAL SIGN
20615	BLUE POLKADOT PASSPORT COVER
20652	BLUE POLKADOT LUGGAGE TAG
22348	TEA BAG PLATE RED RETROSPOT
22412	METAL SIGN NEIGHBOURHOOD WITCH
21171	BATHROOM METAL SIGN
84086C	PINK/PURPLE RETRO RADIO

Item-based collaborative filtering

In [15]:
```
item_item_sim_matrix = pd.DataFrame(cosine_similarity(customer_item_matrix.T))
item_item_sim_matrix.columns = customer_item_matrix.T.index

item_item_sim_matrix['StockCode'] = customer_item_matrix.T.index
item_item_sim_matrix = item_item_sim_matrix.set_index('StockCode')
```

In [16]:
```
print(item_item_sim_matrix)
```

```
StockCode     10002     10080     10120     10125     10133     10135  \
StockCode
10002      1.000000  0.000000  0.094868  0.090351  0.062932  0.098907
10080      0.000000  1.000000  0.000000  0.032774  0.045655  0.047836
10120      0.094868  0.000000  1.000000  0.057143  0.059702  0.041703
10125      0.090351  0.032774  0.057143  1.000000  0.042644  0.044682
10133      0.062932  0.045655  0.059702  0.042644  1.000000  0.280097
...             ...       ...       ...       ...       ...       ...
C2         0.029361  0.000000  0.000000  0.000000  0.036955  0.019360
DOT        0.000000  0.000000  0.000000  0.000000  0.000000  0.104257
M          0.066915  0.016182  0.070535  0.070535  0.070185  0.066184
PADS       0.000000  0.000000  0.000000  0.000000  0.049752  0.000000
POST       0.078217  0.000000  0.010993  0.070669  0.021877  0.034383

StockCode     11001     15030     15034     15036  ...    90214V  90214W  \
StockCode                                          ...
10002      0.095346  0.047673  0.075593  0.090815  ...  0.000000     0.0
10080      0.000000  0.000000  0.082261  0.049413  ...  0.000000     0.0
10120      0.060302  0.060302  0.095618  0.028718  ...  0.000000     0.0
10125      0.043073  0.000000  0.051224  0.030770  ...  0.000000     0.0
10133      0.045002  0.060003  0.071358  0.057152  ...  0.000000     0.0
...             ...       ...       ...       ...  ...       ...     ...
C2         0.055989  0.000000  0.000000  0.039996  ...  0.000000     0.0
DOT        0.150756  0.000000  0.000000  0.000000  ...  0.000000     0.0
M          0.106335  0.063801  0.059013  0.086089  ...  0.049875     0.0
PADS       0.000000  0.000000  0.000000  0.000000  ...  0.000000     0.0
POST       0.058004  0.016573  0.026278  0.051301  ...  0.038866     0.0

StockCode    90214Y  90214Z  BANK CHARGES        C2  DOT         M      PADS  \
StockCode
10002      0.000000     0.0      0.000000  0.029361  0.0  0.066915  0.000000
10080      0.000000     0.0      0.000000  0.000000  0.0  0.016182  0.000000
10120      0.000000     0.0      0.000000  0.000000  0.0  0.070535  0.000000
10125      0.000000     0.0      0.000000  0.000000  0.0  0.070535  0.000000
10133      0.000000     0.0      0.000000  0.036955  0.0  0.070185  0.049752
...             ...     ...           ...       ...  ...       ...       ...
C2         0.000000     0.0      0.000000  1.000000  0.0  0.026196  0.000000
DOT        0.000000     0.0      0.000000  0.000000  1.0  0.000000  0.000000
M          0.040723     0.0      0.089220  0.026196  0.0  1.000000  0.000000
PADS       0.000000     0.0      0.000000  0.000000  0.0  0.000000  1.000000
POST       0.031734     0.0      0.017381  0.020413  0.0  0.077539  0.000000

StockCode      POST
StockCode
10002      0.078217
10080      0.000000
10120      0.010993
10125      0.070669
10133      0.021877
...             ...
C2         0.020413
DOT        0.000000
M          0.077539
PADS       0.000000
POST       1.000000

[3665 rows x 3665 columns]
```

Making Recommendations

In [17]:
```
top_10_similar_items = list(item_item_sim_matrix.loc[23166].sort_values(ascending=

print(top_10_similar_items)
```

```
print()
print(df.loc[
    df['StockCode'].isin(top_10_similar_items),
    ['StockCode', 'Description']
].drop_duplicates().set_index('StockCode').loc[top_10_similar_items])
```

```
[23166, 23165, 23167, 22993, 23307, 22722, 22720, 22666, 23243, 22961]

                                Description
StockCode
23166           MEDIUM CERAMIC TOP STORAGE JAR
23165            LARGE CERAMIC TOP STORAGE JAR
23167           SMALL CERAMIC TOP STORAGE JAR
22993             SET OF 4 PANTRY JELLY MOULDS
23307      SET OF 60 PANTRY DESIGN CAKE CASES
22722        SET OF 6 SPICE TINS PANTRY DESIGN
22720        SET OF 3 CAKE TINS PANTRY DESIGN
22666          RECIPE BOX PANTRY YELLOW DESIGN
23243      SET OF TEA COFFEE SUGAR TINS PANTRY
22961                  JAM MAKING SET PRINTED
```

In [ ]: