Open in Colab

In [1]:

```
!curl https://raw.githubusercontent.com/HeptaDecane/LP2_SEM7/main/A03/MarketBasket.csv -
-output MarketBasket.csv
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  784k  100  784k    0     0  2731k      0 --:--:-- --:--:-- --:--:-- 2731k
```

In [2]:

```
!pip3 install apyori
```

```
Requirement already satisfied: apyori in /usr/local/lib/python3.7/dist-packages (1.1.2)
```

In [3]:

```
import numpy as np
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

from apyori import apriori
```

In [4]:

```
df = pd.read_csv('MarketBasket.csv')
df
```

Out[4]:

	Item(s)	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10	I
0	4	citrus fruit	semi-finished bread	margarine	ready soups	NaN	NaN	NaN	NaN	NaN	NaN	I
1	3	tropical fruit	yogurt	coffee	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
2	1	whole milk	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
3	4	pip fruit	yogurt	cream cheese	meat spreads	NaN	NaN	NaN	NaN	NaN	NaN	I
4	4	other vegetables	whole milk	condensed milk	long life bakery product	NaN	NaN	NaN	NaN	NaN	NaN	I
...	...	...	...	...	...	...	...	...	...	...	...	
9830	17	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	whipped/sour cream	f
9831	1	cooking chocolate	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
9832	10	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	cling film/bags	I
9833	4	semi-finished bread	bottled water	soda	bottled beer	NaN	NaN	NaN	NaN	NaN	NaN	I
9834	5	chicken	tropical fruit	other vegetables	vinegar	shopping bags	NaN	NaN	NaN	NaN	NaN	I

9835 rows × 33 columns

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9835 entries, 0 to 9834
Data columns (total 33 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   Item(s)  9835 non-null   int64 
 1   Item 1   9835 non-null   object
 2   Item 2   7676 non-null   object
 3   Item 3   6033 non-null   object
 4   Item 4   4734 non-null   object
 5   Item 5   3729 non-null   object
 6   Item 6   2874 non-null   object
 7   Item 7   2229 non-null   object
 8   Item 8   1684 non-null   object
 9   Item 9   1246 non-null   object
 10  Item 10  896 non-null    object
 11  Item 11  650 non-null    object
 12  Item 12  468 non-null    object
 13  Item 13  351 non-null    object
 14  Item 14  273 non-null    object
 15  Item 15  196 non-null    object
 16  Item 16  141 non-null    object
 17  Item 17  95 non-null     object
 18  Item 18  66 non-null     object
 19  Item 19  52 non-null     object
 20  Item 20  38 non-null     object
 21  Item 21  29 non-null     object
 22  Item 22  18 non-null     object
 23  Item 23  14 non-null     object
 24  Item 24  8 non-null      object
 25  Item 25  7 non-null      object
 26  Item 26  7 non-null      object
 27  Item 27  6 non-null      object
 28  Item 28  5 non-null      object
 29  Item 29  4 non-null      object
 30  Item 30  1 non-null      object
 31  Item 31  1 non-null      object
 32  Item 32  1 non-null      object
dtypes: int64(1), object(32)
memory usage: 2.5+ MB
```

In [6]:

```
records = []
np_arr = df.drop(columns='Item(s)').values

for row in np_arr:
    row = row[~pd.isna(row)]
    records.append([entry for entry in row])
```

In [7]:

```
association_rules = apriori(records,min_support = 0.005, min_confidence = 0.2, min_lift
= 3)
association_results = list(association_rules)
```

In [8]:

```
for x in association_results[0]:
    print(x)
```

```
frozenset({'beef', 'root vegetables'})
0.017386883579054397
[OrderedStatistic(items_base=frozenset({'beef'}), items_add=frozenset({'root vegetables'}
), confidence=0.3313953488372093, lift=3.0403668431100312)]
```

In [9]:

```
association_df = pd.DataFrame(columns = ['items_base','items_add','support','confidence'
,'lift'])

for entry in association_results:
    ordered_statistics = entry.ordered_statistics[0]
    association_df = association_df.append({
        'items_base': ', '.join(ordered_statistics.items_base),
        'items_add': ', '.join(ordered_statistics.items_add),
        'support': entry[1],
        'confidence': ordered_statistics.confidence,
        'lift': ordered_statistics.lift
    }, ignore_index=True)

association_df
```

Out[9]:

	items_base	items_add	support	confidence	lift
0	beef	root vegetables	0.017387	0.331395	3.040367
1	berries	whipped/sour cream	0.009049	0.272171	3.796886
2	herbs	root vegetables	0.007016	0.431250	3.956477
3	sliced cheese	sausage	0.007016	0.286307	3.047435
4	other vegetables, beef	root vegetables	0.007931	0.402062	3.688692
5	beef, whole milk	root vegetables	0.008033	0.377990	3.467851
6	whole milk, butter	domestic eggs	0.005999	0.217712	3.431409
7	other vegetables, butter	root vegetables	0.006609	0.329949	3.027100
8	other vegetables, butter	whipped/sour cream	0.005796	0.289340	4.036397
9	whole milk, butter	whipped/sour cream	0.006711	0.243542	3.397503
10	chicken, whole milk	root vegetables	0.005999	0.341040	3.128855
11	citrus fruit, other vegetables	root vegetables	0.010371	0.359155	3.295045
12	other vegetables, tropical fruit	citrus fruit	0.009049	0.252125	3.046248
13	citrus fruit, pip fruit	tropical fruit	0.005592	0.404412	3.854060
14	citrus fruit, root vegetables	tropical fruit	0.005694	0.321839	3.067139
15	tropical fruit, curd	yogurt	0.005287	0.514851	3.690645
16	whole milk, curd	whipped/sour cream	0.005897	0.225681	3.148329
17	margarine, whole milk	domestic eggs	0.005186	0.214286	3.377404
18	other vegetables, domestic eggs	root vegetables	0.007321	0.328767	3.016254
19	other vegetables, domestic eggs	whipped/sour cream	0.005084	0.228311	3.185012
20	other vegetables, frozen vegetables	root vegetables	0.006101	0.342857	3.145522
21	other vegetables, onions	root vegetables	0.005694	0.400000	3.669776
22	other vegetables, pip fruit	tropical fruit	0.009456	0.361868	3.448613
23	pip fruit, whipped/sour cream	other vegetables	0.005592	0.604396	3.123610
24	other vegetables, tropical fruit	root vegetables	0.012303	0.342776	3.144780
25	other vegetables, tropical fruit	whipped/sour cream	0.007829	0.218130	3.042995
26	other vegetables, yogurt	whipped/sour cream	0.010168	0.234192	3.267062
27	pip fruit, root vegetables	tropical fruit	0.005287	0.339869	3.238967
28	pip fruit, yogurt	tropical fruit	0.006406	0.355932	3.392048
29	rolls/buns, shopping bags	sausage	0.005999	0.307292	3.270794
30	yogurt, root vegetables	tropical fruit	0.008134	0.314961	3.001587
31	yogurt, root vegetables	whipped/sour cream	0.006406	0.248031	3.460127

	items_base	items_add	support	confidence	lift
32	whipped/sour cream, tropical fruit	yogurt	0.006202	0.448529	3.215224
33	citrus fruit, other vegetables	whole milk, root vegetables	0.005796	0.200704	4.103796
34	other vegetables, fruit/vegetable juice	whole milk, yogurt	0.005084	0.241546	4.311441
35	other vegetables, pip fruit	whole milk, root vegetables	0.005491	0.210117	4.296254
36	pip fruit, yogurt	other vegetables, whole milk	0.005084	0.282486	3.774794
37	rolls/buns, root vegetables	other vegetables, whole milk	0.006202	0.255230	3.410582
38	tropical fruit, root vegetables	other vegetables, whole milk	0.007016	0.333333	4.454257
39	whipped/sour cream, root vegetables	other vegetables, whole milk	0.005186	0.303571	4.056556
40	yogurt, root vegetables	other vegetables, whole milk	0.007829	0.303150	4.050919
41	other vegetables, tropical fruit	whole milk, yogurt	0.007626	0.212465	3.792358
42	whipped/sour cream, yogurt	other vegetables, whole milk	0.005592	0.269608	3.602708
43	tropical fruit, root vegetables	yogurt, whole milk	0.005694	0.270531	4.828814

In [10]:

```
association_df.to_csv('Association.csv')
```