Shri Vile Parle Kelavani Mandal's
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

**A.Y. 2022-2023**

**Subject: Data Mining and Warehousing** **SAP ID: 60004220253 – Devansh Mehta**

## Experiment 06

**Aim:** Implementation of Association rule mining

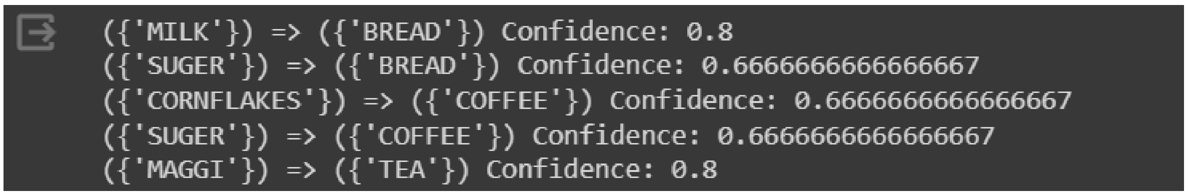
### Apriori Algorithm

**Code:**

```
import pandas as pd
import numpy as np
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder
df = pd.read_csv('../content/sample_data/GroceryStoreDataSet.csv', names = ['products'], sep
= ',')
df.head()
data = list(df["products"].apply(lambda x:x.split(",") ))
data
a = TransactionEncoder()
a_data = a.fit(data).transform(data)
df = pd.DataFrame(a_data,columns=a.columns_)
df = df.replace(False,0)
df
df = apriori(df, min_support = 0.2, use_colnames = True, verbose = 1)
df
df_ar = association_rules(df, metric = "confidence", min_threshold = 0.6)
df_ar

for i in range(len(df_ar)):
    print("",str(df_ar.loc[i, "antecedents"]).replace("frozenset",""),"=>",str(df_ar.loc[i,
"consequents"]).replace("frozenset",""),"Confidence:",str(df_ar.loc[i,
"confidence"]).replace("frozenset",""))
```

**Output:**



```
({'MILK'}) => ({'BREAD'}) Confidence: 0.8
({'SUGER'}) => ({'BREAD'}) Confidence: 0.6666666666666667
({'CORNFLAKES'}) => ({'COFFEE'}) Confidence: 0.6666666666666667
({'SUGER'}) => ({'COFFEE'}) Confidence: 0.6666666666666667
({'MAGGI'}) => ({'TEA'}) Confidence: 0.8
```

Shri Vile Parle Kelavani Mandal's
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

**A.Y. 2022-2023**

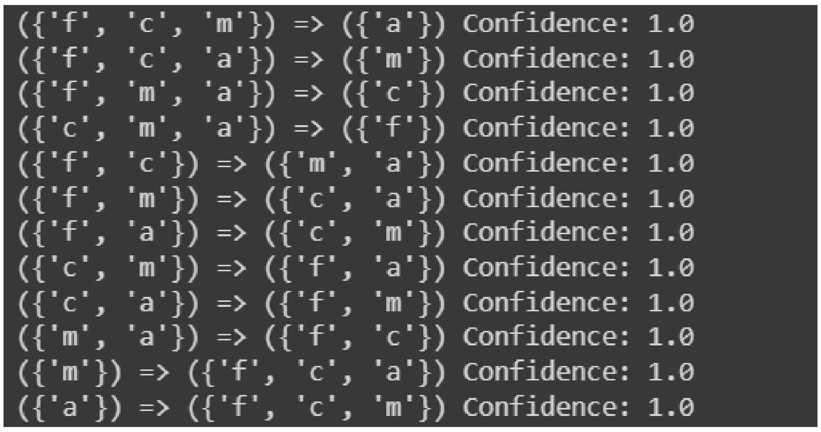
**Subject: Data Mining and Warehousing** **SAP ID: 60004220253 – Devansh Mehta**

# FP Tree

**Code:**

```
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules
dataset = [['f', 'a', 'c', 'd', 'g', 'i', 'm', 'p'],
           ['a', 'b', 'c', 'f', 'l', 'm', 'o'],
           ['b', 'f', 'h', 'j', 'o', 'w'],
           ['b', 'c', 'k', 's', 'p'],
           ['a', 'f', 'c', 'e', 'l', 'p','m', 'n']]
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)
df
result = fpgrowth(df, min_support=0.6,use_colnames=True)
df_ar = association_rules(result, metric="confidence", min_threshold=0.8)
for i in range(len(df_ar)):
   print("",str(df_ar.loc[i, "antecedents"]).replace("frozenset",""),"=>",str(df_ar.loc[i,
"consequents"]).replace("frozenset",""),"Confidence:",str(df_ar.loc[i,
"confidence"]).replace("frozenset",""))
```

**Output:**



```
({'f', 'c', 'm'}) => ({'a'}) Confidence: 1.0
({'f', 'c', 'a'}) => ({'m'}) Confidence: 1.0
({'f', 'm', 'a'}) => ({'c'}) Confidence: 1.0
({'c', 'm', 'a'}) => ({'f'}) Confidence: 1.0
({'f', 'c'}) => ({'m', 'a'}) Confidence: 1.0
({'f', 'm'}) => ({'c', 'a'}) Confidence: 1.0
({'f', 'a'}) => ({'c', 'm'}) Confidence: 1.0
({'c', 'm'}) => ({'f', 'a'}) Confidence: 1.0
({'c', 'a'}) => ({'f', 'm'}) Confidence: 1.0
({'m', 'a'}) => ({'f', 'c'}) Confidence: 1.0
({'m'}) => ({'f', 'c', 'a'}) Confidence: 1.0
({'a'}) => ({'f', 'c', 'm'}) Confidence: 1.0
```