Program 4: FP Tree

1. Consider the given transaction and create the list accordingly. Identify frequent item set using FP tree with minimum support count 2 and confidence 75%. Generate association rules.

```
pip install pyfpgrowth
```

```
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/cola
    Collecting pyfpgrowth
      Downloading pyfpgrowth-1.0.tar.gz (1.6 MB)
         ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 41.1 MB/s eta 0:00
      Preparing metadata (setup.py) ... done
    Building wheels for collected packages: pyfpgrowth
      Building wheel for pyfpgrowth (setup.py) ... done
      Created wheel for pyfpgrowth: filename=pyfpgrowth-1.0-py2.py3-none-any.wh
      Stored in directory: /root/.cache/pip/wheels/29/e2/a7/ccb34c35e2c6737d3f0
    Successfully built pyfpgrowth
    Installing collected packages: pyfpgrowth
    Successfully installed pyfpgrowth-1.0
```

```
#Import Libraries
import pandas as pd
import numpy as np
import pyfpgrowth as fp
```

```
data=[{'Bread','cheese','newspaper'},
      {'Bread','cheese','juice'},
      {'Bread','milk'},
      {'Chesse','juice','milk','coffee'},
      {'Sugar','tea','coffee','biscuits','newspaper'},
      {'Sugar','tea','coffee','biscuits','milk','juice','newspaper'},
      {'Bread','cheese'},
      {'Bread','cheese','juice','coffee'},
      {'Bread','milk'},
      {'Sugar','tea','coffee','bread','milk','juice','newspaper'}]
```

```
data
```

```
[{'Bread', 'cheese', 'newspaper'},
 {'Bread', 'cheese', 'juice'},
 {'Bread', 'milk'},
 {'Chesse', 'coffee', 'juice', 'milk'},
 {'Sugar', 'biscuits', 'coffee', 'newspaper', 'tea'},
 {'Sugar', 'biscuits', 'coffee', 'juice', 'milk', 'newspaper', 'tea'},
 {'Bread', 'cheese'},
 {'Bread', 'cheese', 'coffee', 'juice'},
 {'Bread', 'milk'},
 {'Sugar', 'bread', 'coffee', 'juice', 'milk', 'newspaper', 'tea'}]
```

```
pattern=fp.find_frequent_patterns(data,2)
```

```
pattern
```

```
{('biscuits',): 2,
 ('biscuits', 'tea'): 2,
 ('Sugar', 'biscuits'): 2,
 ('biscuits', 'newspaper'): 2,
 ('biscuits', 'coffee'): 2,
 ('Sugar', 'biscuits', 'tea'): 2,
 ('biscuits', 'newspaper', 'tea'): 2,
 ('biscuits', 'coffee', 'tea'): 2,
 ('Sugar', 'biscuits', 'newspaper'): 2,
 ('Sugar', 'biscuits', 'coffee'): 2,
 ('biscuits', 'coffee', 'newspaper'): 2,
 ('Sugar', 'biscuits', 'newspaper', 'tea'): 2,
 ('Sugar', 'biscuits', 'coffee', 'tea'): 2,
 ('biscuits', 'coffee', 'newspaper', 'tea'): 2,
 ('Sugar', 'biscuits', 'coffee', 'newspaper'): 2,
 ('Sugar', 'biscuits', 'coffee', 'newspaper', 'tea'): 2,
 ('Sugar',): 3,
 ('Sugar', 'newspaper'): 3,
 ('Sugar', 'coffee'): 3,
 ('Sugar', 'juice'): 2,
 ('Sugar', 'milk'): 2,
 ('Sugar', 'coffee', 'newspaper'): 3,
 ('Sugar', 'juice', 'newspaper'): 2,
 ('Sugar', 'milk', 'newspaper'): 2,
 ('Sugar', 'coffee', 'juice'): 2,
 ('Sugar', 'coffee', 'milk'): 2,
 ('Sugar', 'juice', 'milk'): 2,
 ('Sugar', 'coffee', 'juice', 'newspaper'): 2,
 ('Sugar', 'coffee', 'milk', 'newspaper'): 2,
 ('Sugar', 'juice', 'milk', 'newspaper'): 2,
 ('Sugar', 'coffee', 'juice', 'milk'): 2,
 ('Sugar', 'coffee', 'juice', 'milk', 'newspaper'): 2,
 ('tea',): 3,
 ('Sugar', 'tea'): 3,
 ('newspaper', 'tea'): 3,
 ('coffee', 'tea'): 3,
 ('juice', 'tea'): 2,
 ('milk', 'tea'): 2,
 ('Sugar', 'newspaper', 'tea'): 3,
 ('Sugar', 'coffee', 'tea'): 3,
 ('Sugar', 'juice', 'tea'): 2,
 ('Sugar', 'milk', 'tea'): 2,
 ('coffee', 'newspaper', 'tea'): 3,
 ('juice', 'newspaper', 'tea'): 2,
 ('milk', 'newspaper', 'tea'): 2,
 ('coffee', 'juice', 'tea'): 2,
 ('coffee', 'milk', 'tea'): 2,
 ('juice', 'milk', 'tea'): 2,
 ('Sugar', 'coffee', 'newspaper', 'tea'): 3,
 ('Sugar', 'juice', 'newspaper', 'tea'): 2,
 ('Sugar', 'milk', 'newspaper', 'tea'): 2,
 ('Sugar', 'coffee', 'juice', 'tea'): 2,
 ('Sugar', 'coffee', 'milk', 'tea'): 2,
 ('Sugar', 'juice', 'milk', 'tea'): 2,
 ('coffee', 'juice', 'newspaper', 'tea'): 2,
 ('coffee', 'milk', 'newspaper', 'tea'): 2,
 ('juice', 'milk', 'newspaper', 'tea'): 2,
 ('coffee', 'juice', 'milk', 'tea'): 2,
 ('Sugar', 'coffee', 'juice', 'newspaper', 'tea'): 2,
 ('Sugar', 'coffee', 'milk', 'newspaper', 'tea'): 2,
```

```
rules=fp.generate_association_rules(pattern,0.75)
```

```
rules
```

```
    {('biscuits',): (('Sugar', 'coffee', 'newspaper', 'tea'), 1.0),
     ('Sugar', 'biscuits'): (('coffee', 'newspaper', 'tea'), 1.0),
     ('biscuits', 'tea'): (('Sugar', 'coffee', 'newspaper'), 1.0),
     ('biscuits', 'newspaper'): (('Sugar', 'coffee', 'tea'), 1.0),
     ('biscuits', 'coffee'): (('Sugar', 'newspaper', 'tea'), 1.0),
     ('Sugar', 'biscuits', 'newspaper'): (('coffee', 'tea'), 1.0),
     ('Sugar', 'biscuits', 'tea'): (('coffee', 'newspaper'), 1.0),
     ('biscuits', 'newspaper', 'tea'): (('Sugar', 'coffee'), 1.0),
     ('Sugar', 'biscuits', 'coffee'): (('newspaper', 'tea'), 1.0),
     ('biscuits', 'coffee', 'tea'): (('Sugar', 'newspaper'), 1.0),
     ('biscuits', 'coffee', 'newspaper'): (('Sugar', 'tea'), 1.0),
     ('Sugar', 'biscuits', 'coffee', 'newspaper'): (('tea',), 1.0),
     ('Sugar', 'biscuits', 'coffee', 'tea'): (('newspaper',), 1.0),
     ('Sugar', 'biscuits', 'newspaper', 'tea'): (('coffee',), 1.0),
     ('biscuits', 'coffee', 'newspaper', 'tea'): (('Sugar',), 1.0),
     ('Sugar',): (('coffee', 'newspaper', 'tea'), 1.0),
     ('newspaper',): (('coffee',), 0.75),
     ('Sugar', 'coffee'): (('newspaper', 'tea'), 1.0),
     ('Sugar', 'newspaper'): (('coffee', 'tea'), 1.0),
     ('coffee', 'newspaper'): (('Sugar', 'tea'), 1.0),
     ('Sugar', 'juice'): (('coffee', 'milk', 'newspaper', 'tea'), 1.0),
     ('juice', 'newspaper'): (('coffee', 'milk'), 1.0),
     ('Sugar', 'milk'): (('coffee', 'juice', 'newspaper', 'tea'), 1.0),
     ('milk', 'newspaper'): (('coffee', 'juice'), 1.0),
     ('Sugar', 'coffee', 'juice'): (('milk', 'newspaper', 'tea'), 1.0),
     ('Sugar', 'juice', 'newspaper'): (('coffee', 'milk', 'tea'), 1.0),
     ('coffee', 'juice', 'newspaper'): (('milk',), 1.0),
     ('Sugar', 'coffee', 'milk'): (('juice', 'newspaper', 'tea'), 1.0),
     ('Sugar', 'milk', 'newspaper'): (('coffee', 'juice', 'tea'), 1.0),
     ('coffee', 'milk', 'newspaper'): (('juice',), 1.0),
     ('Sugar', 'juice', 'milk'): (('coffee', 'newspaper', 'tea'), 1.0),
     ('juice', 'milk', 'newspaper'): (('coffee',), 1.0),
     ('Sugar', 'coffee', 'juice', 'milk'): (('newspaper', 'tea'), 1.0),
     ('Sugar', 'coffee', 'juice', 'newspaper'): (('milk', 'tea'), 1.0),
     ('Sugar', 'coffee', 'milk', 'newspaper'): (('juice', 'tea'), 1.0),
     ('Sugar', 'juice', 'milk', 'newspaper'): (('coffee', 'tea'), 1.0),
     ('coffee', 'juice', 'milk', 'newspaper'): (('Sugar', 'tea'), 1.0),
     ('tea',): (('Sugar', 'coffee', 'newspaper'), 1.0),
     ('Sugar', 'tea'): (('coffee', 'newspaper'), 1.0),
     ('newspaper', 'tea'): (('Sugar', 'coffee'), 1.0),
     ('coffee', 'tea'): (('Sugar', 'newspaper'), 1.0),
     ('juice', 'tea'): (('Sugar', 'coffee', 'milk', 'newspaper'), 1.0),
     ('milk', 'tea'): (('Sugar', 'coffee', 'juice', 'newspaper'), 1.0),
     ('Sugar', 'coffee', 'newspaper'): (('tea',), 1.0),
     ('Sugar', 'coffee', 'tea'): (('newspaper',), 1.0),
     ('Sugar', 'newspaper', 'tea'): (('coffee',), 1.0),
     ('coffee', 'newspaper', 'tea'): (('Sugar',), 1.0),
     ('Sugar', 'juice', 'tea'): (('coffee', 'milk', 'newspaper'), 1.0),
     ('juice', 'newspaper', 'tea'): (('Sugar', 'coffee', 'milk'), 1.0),
     ('Sugar', 'milk', 'tea'): (('coffee', 'juice', 'newspaper'), 1.0),
     ('milk', 'newspaper', 'tea'): (('Sugar', 'coffee', 'juice'), 1.0),
     ('coffee', 'juice', 'tea'): (('Sugar', 'milk', 'newspaper'), 1.0),
     ('coffee', 'milk', 'tea'): (('Sugar', 'juice', 'newspaper'), 1.0),
     ('juice', 'milk', 'tea'): (('Sugar', 'coffee', 'newspaper'), 1.0),
     ('Sugar', 'coffee', 'juice', 'tea'): (('milk', 'newspaper'), 1.0),
     ('Sugar', 'juice', 'newspaper', 'tea'): (('coffee', 'milk'), 1.0),
     ('coffee', 'juice', 'newspaper', 'tea'): (('Sugar', 'milk'), 1.0),
     ('Sugar', 'coffee', 'milk', 'tea'): (('juice', 'newspaper'), 1.0),
     ('Sugar', 'milk', 'newspaper', 'tea'): (('coffee', 'juice'), 1.0),
     ('coffee', 'milk', 'newspaper', 'tea'): (('Sugar', 'juice'), 1.0),
```

Part-2

```
dataset=pd.read_csv("Market_Basket_Original.csv")
```

```
x,y=dataset.shape
```

```
transactions=[]
for i in range (0,x):
  transactions.append([str(dataset.values[i,j])for j in range(0,y)])
```

```
patterns=fp.find_frequent_patterns(transactions,10)
```

```
patterns
```

```
    {('bramble',): 14,
     ('bramble', 'nan'): 175,
     ('frozen vegetables', 'tea'): 10,
     ('frozen vegetables', 'nan', 'tea'): 114,
     ('spaghetti', 'tea'): 11,
     ('nan', 'spaghetti', 'tea'): 121,
     ('mineral water', 'tea'): 11,
     ('mineral water', 'nan', 'tea'): 119,
     ('nan', 'tea'): 365,
     ('nan', 'nan', 'tea'): 2262,
     ('chutney', 'spaghetti'): 11,
     ('chutney', 'nan', 'spaghetti'): 116,
     ('chutney', 'eggs'): 11,
     ('chutney', 'eggs', 'nan'): 127,
     ('chutney', 'mineral water'): 13,
     ('chutney', 'mineral water', 'nan'): 145,
     ('chutney', 'nan'): 418,
     ('chutney', 'nan', 'nan'): 2865,
     ('mashed potato', 'mineral water'): 11,
     ('mashed potato', 'mineral water', 'nan'): 143,
     ('mashed potato', 'spaghetti'): 11,
     ('mashed potato', 'nan', 'spaghetti'): 116,
     ('mashed potato', 'nan'): 424,
     ('mashed potato', 'nan', 'nan'): 2902,
     ('chocolate bread', 'mineral water'): 14,
     ('chocolate bread', 'mineral water', 'nan'): 171,
     ('chocolate bread', 'nan'): 438,
     ('chocolate bread', 'nan', 'nan'): 2913,
     ('dessert wine', 'spaghetti'): 10,
     ('dessert wine', 'nan', 'spaghetti'): 133,
     ('dessert wine', 'mineral water'): 12,
     ('dessert wine', 'mineral water', 'nan'): 156,
     ('dessert wine', 'nan'): 458,
     ('dessert wine', 'nan', 'nan'): 3086,
     ('ketchup', 'mineral water'): 10,
     ('ketchup', 'mineral water', 'nan'): 106,
     ('ketchup', 'milk'): 10,
     ('ketchup', 'milk', 'nan'): 103,
     ('ketchup', 'spaghetti'): 11,
     ('ketchup', 'nan', 'spaghetti'): 102,
     ('ketchup', 'pancakes'): 12,
     ('ketchup', 'nan', 'pancakes'): 130,
     ('ketchup', 'nan'): 396,
     ('ketchup', 'nan', 'nan'): 2398,
     ('chocolate', 'oatmeal'): 10,
     ('chocolate', 'nan', 'oatmeal'): 110,
     ('mineral water', 'oatmeal'): 13,
     ('mineral water', 'nan', 'oatmeal'): 154,
     ('nan', 'oatmeal'): 426,
     ('nan', 'nan', 'oatmeal'): 2688,
     ('sandwich',): 34,
     ('nan', 'sandwich'): 499,
     ('babies food', 'chocolate'): 13,
     ('babies food', 'chocolate', 'nan'): 167,
     ('babies food', 'nan'): 494,
     ('babies food', 'nan', 'nan'): 3569,
     ('asparagus', 'milk'): 10,
     ('asparagus', 'milk', 'nan'): 128,
     ('asparagus', 'mineral water'): 16,
     ('asparagus', 'mineral water', 'nan'): 194
```

```
rules=fp.generate_association_rules(patterns,0.8)
```

```
rules
```

```
     {('bramble',): (('nan',), 12.5),
      ('frozen vegetables', 'tea'): (('nan',), 11.4),
      ('spaghetti', 'tea'): (('nan',), 11.0),
      ('mineral water', 'tea'): (('nan',), 10.818181818181818),
      ('nan', 'tea'): ((), 6.197260273972603),
      ('chutney', 'spaghetti'): (('nan',), 10.545454545454545),
      ('chutney', 'eggs'): (('nan',), 11.545454545454545),
      ('chutney', 'mineral water'): (('nan',), 11.153846153846153),
      ('chutney', 'nan'): ((), 6.854066985645933),
      ('mashed potato', 'mineral water'): (('nan',), 13.0),
      ('mashed potato', 'spaghetti'): (('nan',), 10.545454545454545),
      ('mashed potato', 'nan'): ((), 6.84433962264151),
      ('chocolate bread', 'mineral water'): (('nan',), 12.214285714285714),
      ('chocolate bread', 'nan'): ((), 6.6506849315068495),
      ('dessert wine', 'spaghetti'): (('nan',), 13.3),
      ('dessert wine', 'mineral water'): (('nan',), 13.0),
      ('dessert wine', 'nan'): ((), 6.737991266375546),
      ('ketchup', 'mineral water'): (('nan',), 10.6),
      ('ketchup', 'milk'): (('nan',), 10.3),
      ('ketchup', 'spaghetti'): (('nan',), 9.272727272727273),
      ('ketchup', 'pancakes'): (('nan',), 10.833333333333334),
      ('ketchup', 'nan'): ((), 6.055555555555555),
      ('chocolate', 'oatmeal'): (('nan',), 11.0),
      ('mineral water', 'oatmeal'): (('nan',), 11.846153846153847),
      ('nan', 'oatmeal'): ((), 6.309859154929577),
      ('sandwich',): (('nan',), 14.676470588235293),
      ('babies food', 'chocolate'): (('nan',), 12.846153846153847),
      ('babies food', 'nan'): ((), 7.224696356275303),
      ('asparagus', 'milk'): (('nan',), 12.8),
      ('asparagus', 'mineral water'): (('nan',), 12.125),
      ('asparagus', 'nan'): ((), 6.727650727650728),
      ('chocolate', 'corn'): (('nan',), 11.9),
      ('corn', 'mineral water'): (('nan',), 12.545454545454545),
      ('corn', 'spaghetti'): (('nan',), 11.727272727272727),
      ('corn', 'nan'): ((), 6.647916666666666),
      ('salad',): (('nan',), 14.583333333333334),
      ('cauliflower', 'spaghetti'): (('nan',), 10.545454545454545),
      ('cauliflower', 'mineral water'): (('nan',), 12.071428571428571),
      ('cauliflower', 'nan'): ((), 6.179039301310044),
      ('shampoo', 'spaghetti'): (('nan',), 11.181818181818182),
      ('mineral water', 'shampoo'): (('nan',), 10.454545454545455),
      ('nan', 'shampoo'): ((), 6.229787234042553),
      ('chocolate', 'hand protein bar'): (('nan',), 10.272727272727273),
      ('hand protein bar', 'spaghetti'): (('nan',), 10.0),
      ('hand protein bar', 'mineral water'): (('nan',), 11.733333333333333),
      ('hand protein bar', 'nan'): ((), 6.456692913385827),
      ('mineral water', 'mint green tea'): (('nan',), 11.9),
      ('mint green tea', 'spaghetti'): (('nan',), 12.181818181818182),
      ('milk', 'mint green tea'): (('nan',), 11.75),
      ('french fries', 'mint green tea'): (('nan',), 11.857142857142858),
      ('mint green tea', 'nan'): ((), 7.031932773109244),
      ('burger sauce', 'chocolate'): (('nan',), 11.7),
      ('burger sauce', 'milk'): (('nan',), 10.363636363636363),
      ('burger sauce', 'ground beef'): (('nan',), 9.0),
      ('burger sauce', 'chicken'): (('nan',), 9.083333333333334),
      ('burger sauce', 'eggs'): (('nan',), 11.75),
      ('burger sauce', 'spaghetti'): (('nan',), 10.0),
      ('burger sauce', 'mineral water', 'spaghetti'): (('nan',), 9.0),
      ('burger sauce', 'nan'): ((), 6.086792452830188),
      ('burger sauce', 'mineral water', 'nan'): ((), 5.065573770491803)
```

Inference- The FP tree algorithm is executed for the given dataset and sample dataset

Colab paid products - Cancel contracts here