**Experiment 1.4**

**Student Name: Rahul Kumar UID: 20BCS7081**

**Branch: BE-CSE Section/Group:20BCS\_DM-716 B**

**Semester: 6 Date of Performance: 15 March 2023**

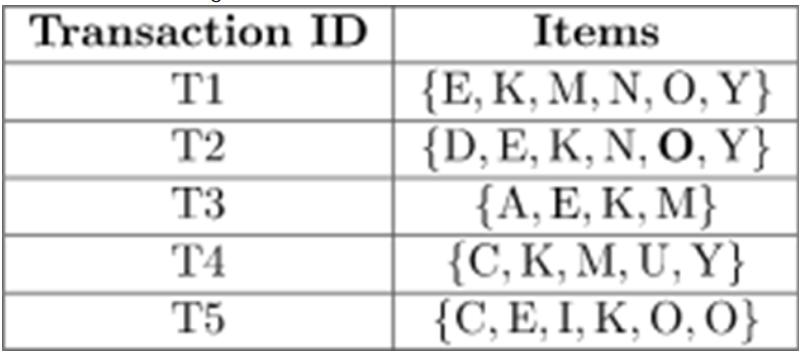
**Subject Name: DM LAB Subject Code: 20CSP\_376**

**AIM :-**

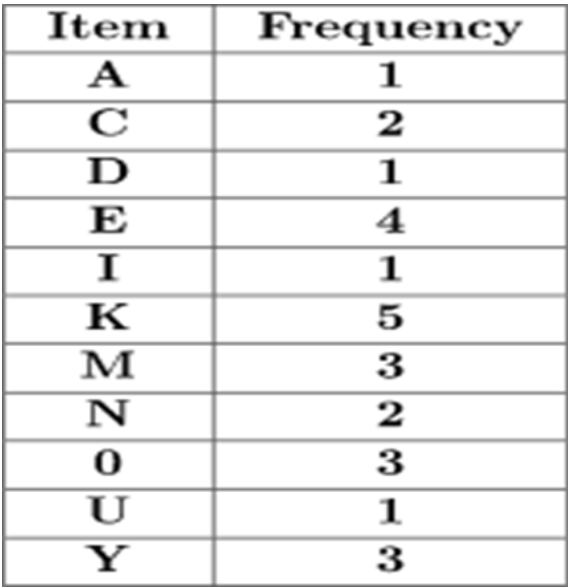
Demonstration of FP Growth algorithm on supermarket data.

**Theory And Output :-**

The two primary drawbacks of the Apriori Algorithm are:   
1.   At each step, candidate sets have to be built.  
  
2.   To build the candidate sets, the algorithm has to repeatedly scan the database.  
These two properties inevitably make the algorithm slower. To overcome these redundant steps, a new association-rule mining algorithm was developed named Frequent Pattern Growth Algorithm. It overcomes the disadvantages of the Apriori algorithm by storing all the transactions in a Trie Data Structure. Consider the following data:-



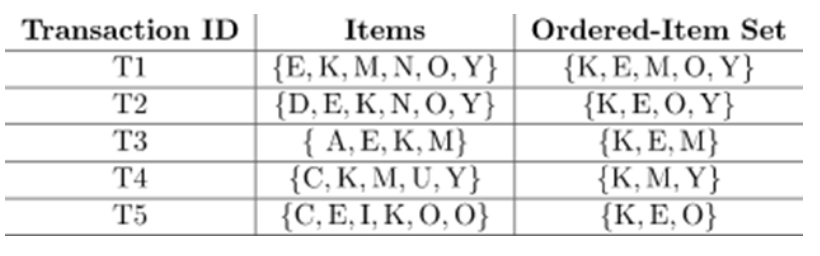
The above-given data is a hypothetical dataset of transactions with each letter representing an item. The frequency of each individual item is computed:-



Let the minimum support be 3. A **Frequent Pattern set** is built which will contain all the elements whose frequency is greater than or equal to the minimum support. These elements are stored in descending order of their respective frequencies. After insertion of the relevant items, the set L looks like this:-

**L = {K : 5, E : 4, M : 3, O : 3, Y : 3}**

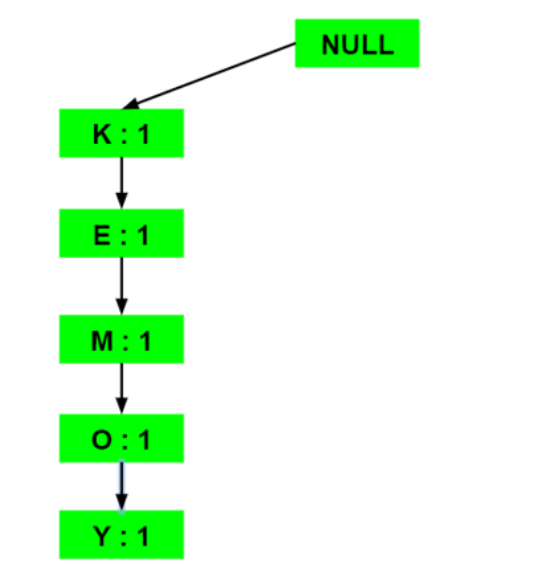
Now, for each transaction, the respective **Ordered-Item set** is built. It is done by iterating the Frequent Pattern set and checking if the current item is contained in the transaction in question. If the current item is contained, the item is inserted in the Ordered-Item set for the current transaction. The following table is built for all the transactions:



Now, all the Ordered-Item sets are inserted into a Trie Data Structure.

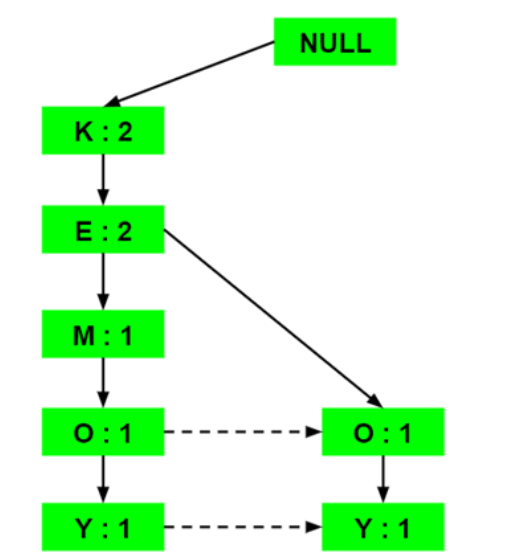
a) **Inserting the set {K, E, M, O, Y}:**

Here, all the items are simply linked one after the other in the order of occurrence in the set and initialize the support count for each item as 1.



b) **Inserting the set {K, E, O, Y}:**

Till the insertion of the elements K and E, simply the support count is increased by 1. On inserting O we can see that there is no direct link between E and O, therefore a new node for the item O is initialized with the support count as 1 and item E is linked to this new node. On inserting Y, we first initialize a new node for the item Y with support count as 1 and link the new node of O with the new node of Y.



Now, for each item, the **Conditional Pattern Base** is computed which is path labels of all the paths which lead to any node of the given item in the frequent-pattern tree. Note that the items in the below table are arranged in the ascending order of their frequencies.

