ECLAT Algorithm for Frequent Itemset Mining

ECLAT Algorithm

- **Equivalence Class Clustering and bottom up Lattice Traversal- ECLAT**
- Method for Frequent Itemset Generation
- Searches in a DFS manner.
- Represent the data in **vertical format.**

ECLAT

To Improve the Efficiency of Apriori : (Scalable Algorithms)

- ✓ FPGrowth
- ✓ ECLAT
- ✓ Mining Close Frequent Patterns and Maxpatterns

- Both Apriori and FP-growth use **horizontal data format**
- Alternatively data can also be represented in **vertical format**

TID	Items
1	Bread,Butter,Jam
2	Butter,Coke
3	Butter,Milk
4	Bread,Butter,Coke
5	Bread,Milk
6	Butter,Milk
7	Bread,Milk
8	Bread,Butter,Milk,Jam
9	Bread,Butter,Milk

Item Set	TID set
Bread	1,4,5,7,8,9
Butter	1,2,3,4,6,8,9
Milk	3,5,6,7,8,9
Coke	2,4
Jam	1,8

Eclat: algorithm

1. Get tidlist for each item (DB scan)
2. Tidlist of {a} is exactly the list of transactions containing {a}
3. Intersect tidlist of {a} with the tidlists of all other items, resulting in tidlists of {a,b}, {a,c}, {a,d}, …
   = {a}-conditional database (if {a} removed)
4. Repeat from 1 on {a}-conditional database
5. Repeat for all other items

Frequent 1-itemsets

min_sup=2

Item Set	TID Set
Bread	1,4,5,7,8,9
Butter	1,2,3,4,6,8,9
Milk	3,5,6,7,8,9
Coke	2,4
Jam	1,8

Frequent 2-itemsets

Item Set	TID set
{Bread,Butter}	1,4,8,9
{Bread,Milk}	5,7,8,9
{Bread,Coke}	4
{Bread,Jam}	1,8
{Butter,Milk}	3,6,8,9
{Butter,Coke}	2,4
{Butter,Jam}	1,8
{Milk,Jam}	8

Frequent 3-itemsets

Item Set	TID Set
{Bread,Butter,Milk}	8,9
{Bread,Butter,Jam}	1,8

- This process repeats, with k incremented by 1 each time, until no frequent items or no candidate itemsets can be found.

- Depth-first search reduces memory requirements
- Usually (considerably) faster than Apriori
- No need to scan the database to find the support of (k+1) itemsets, for k>=1

Disadvantage

- The TID-sets can be quite long, hence expensive to manipulate