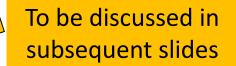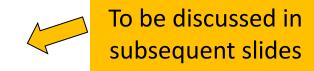# Extensions or Improvements of Apriori

Apriori: Improvements and Alternatives

❑ Reduce passes of transaction database scans

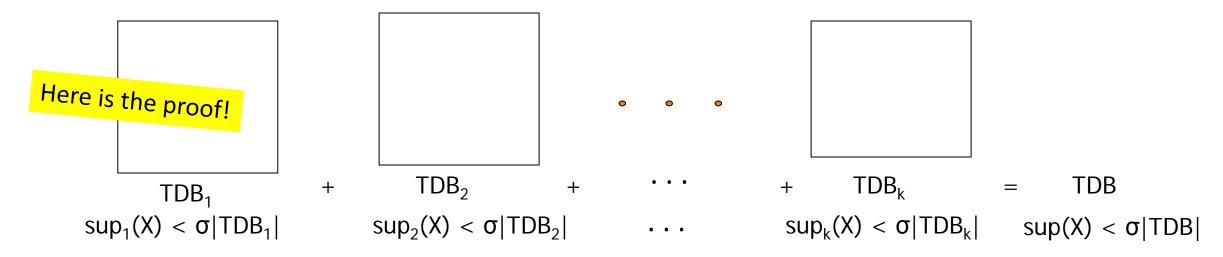   ❑ Partitioning (e.g., Savasere, et al., 1995)

To be discussed in subsequent slides

   ❑ Dynamic itemset counting (Brin, et al., 1997)

❑ Shrink the number of candidates

   ❑ Hashing (e.g., DHP: Park, et al., 1995)

To be discussed in subsequent slides

   ❑ Pruning by support lower bounding (e.g., Bayardo 1998)

   ❑ Sampling (e.g., Toivonen, 1996)

❑ Exploring special data structures

   ❑ Tree projection (Agarwal, et al., 2001)

   ❑ H-miner (Pei, et al., 2001)

   ❑ Hypecube decomposition (e.g., LCM: Uno, et al., 2004)

Partitioning: Scan Database Only Twice

❑ Theorem: *Any itemset that is potentially frequent in TDB must be frequent in at least one of the partitions of TDB*

Here is the proof!

TDB_1 + TDB_2 +   . . .   + TDB_k =   TDB

$sup_1(X) < \sigma|TDB_1|$ $sup_2(X) < \sigma|TDB_2|$ . . . $sup_k(X) < \sigma|TDB_k|$ $sup(X) < \sigma|TDB|$

❑ Method: (A. Savasere, E. Omiecinski and S. Navathe, *VLDB'95)*

   ❑ Scan 1: Partition database (how?) and find local frequent patterns

   ❑ Scan 2: Consolidate global frequent patterns (how to?)

❑ Why does this method guarantee to scan TDB only twice?

Direct Hashing and Pruning (DHP)

❑ DHP (Direct Hashing and Pruning): Reduce the number of candidates (J. Park, M. Chen, and P. Yu, SIGMOD'95)

❑ Observation: A *k*-itemset whose corresponding hashing bucket count is below the threshold cannot be frequent

  ❑ Candidates: a, b, c, d, e

  ❑ Hash entries

    ❑ {ab, ad, ae}

    ❑ {bd, be, de}

    ❑ …

Itemsets	Count
{ab, ad, ae}	35
{bd, be, de}	298
……	…
{yz, qs, wt}	58

Hash Table

  ❑ Frequent 1-itemset: a, b, d, e

  ❑ ab is not a candidate 2-itemset if the sum of count of {ab, ad, ae} is below support threshold