DB – Homework 2

Prateek Agarwal (prat0318@gmail.com)

Sol 1: Basically the tuples from A have now to be mapped to two keys ‘x’ and ‘y’. This can be done by creating two separate hash tables one for all ‘x’ keys and another for ‘y’ keys, and the keys point to the tuples shared with both of them.

At B, all the rows are iterated as in Classical Hash Join, for each row, ‘x’ value is checked in the hash table created above for ‘x’, if an entry is found then join the row with the key’s value. If no entry in hash is found, then ‘y’ value is checked in the hash table for ‘y’, if it matches then join the rows, else skip the row and continue.

Sol2: I would use ‘binary search tree’ instead of ‘hash-table’ data structure in the Classical Hash Join algorithm.

Instead of building hash-table of table A, create a binary search tree of table A, and use it for the lookups of B. On using binary search tree, we reduce the lookup time to k*O(lg n) for each row of B. As a field lookup in a BST takes an average of O(lg n). [where k is the number of matches of A for a row in B, and n is the number of rows in A]

Sol 3a: CREATE VIEW KIMTEMP AS (

SELECT A.w from A group by w

MINUS

SELECT C.w from C group by w

);

SELECT A.x FROM A JOIN KIMTEMP where A.q = 40;

3b) SELECT A.x FROM A LEFT OUTER JOIN C ON A.w = C.w WHERE A.q = 40 AND C.w is null;

Sol 4) The query gives the pname of all parts, which were either bought by all customers of Austin or by no one in Austin.