Harvey Mudd College
Computer Science 133
Database and Knowledgebase Systems
Fall Semester 2001

Assignment #6 – Indexing and Query Optimization

Due 2:45pm, Friday December 7, 2001

Part I - Indexing

1. Consider the B^+-tree given in Figure 12.15 of the text. (This sort of a tree is often called a **2-3 tree** since each node always contains either 2 or 3 pointers.) You may assume that the key values in the tree form a candidate key for the indexed relation, and that, therefore, the pointers from the leaves of the tree point directly to records in the relation.

   (a) Show what the tree would look like after a record with the key value **Claremont** was inserted.

   (b) Show what the resulting tree would look like after a record with key value **Boston** was inserted.

   (c) Show what the original tree (not the result of the last problem) would look like after the record with the key value **Clearview** was deleted.

   (d) Show what the resulting tree would look like after the record with the key value **Brighton** was deleted.

2. Consider the example of Dynamic Hashing given in Section 12.6 of the text. The values of the hash function for various key values is given in Figure 12.26. Now consider the hash structure as it is given in Figure 12.31 after a series of insertions.

   (a) Show what the hash structure would look like after the insertion of records with the key values **Claremont** (whose hash value begins **1001 ...**) and **Upland** (whose hash value begins **0001 ...**).

   (b) Show what the resulting hash structure would look like after the insertion of a record with the key value **Laverne**, whose hash value begins **0011 ...**).

Part II - Query Optimization

Consider the following SQL query over relations branch(branch_name,assets,branch_city) and account(acc_num,owner_name,branch_name,acc_balance):

```
select owner_name
from branch B, account A
where B.assets >= 5,000,000 and
      B.branch_city = 'Claremont' and
      A.branch_name = B.branch_name and
      A.acc_balance >= 10,000.
```

(a) Write the raw relational algebra relation to which this is directly equivalent.

(b) Use the query optimization techniques discussed in class to give a more efficient version of this query. You may want to show intermediate versions to optimize partial credit.

(c) Assuming there are 100,000 accounts in the account table and 50 branches in the branch table, that 10% of the accounts have balances of at least $10,000, that the accounts are evenly distributed among the 50 branches, that the branches are evenly distributed among 12 cities, and that 75% of the branches have assets of at least $5,000,000.

i. How many tuples do you expect in the result of this query?

ii. If we did not restrict to branches with specific assets, how many tuples would you expect?

iii. If we did not restrict to accounts with specific balances, how many tuples would you expect?