Database Portion of Systems QUAL, May 27, 1999

Problem 1 (15 pts.) U.S. News and World Report annually publishes rankings of Ph.D. programs in various fields and subfields. For each field and subfield, each ranked university receives a score of up to 5.0 points. Suppose this information is stored in a relational database with a separate table for each field and subfield, e.g.:

```
ComputerScience(school, score) // school is a key
ElectricalEngg(school, score)  // school is a key
Databases(school, score)       // school is a key
etc.
```

a) [5 points] Let us begin by assuming that scores are unique within each table, i.e., there are no "ties." Can you write a relational algebra expression that returns the top-scoring school in Databases? (The correct answer is "Stanford," of course.) If you can write such an expression, show it. If not, explain why not.

b) [5 points] Continue to assume that scores are unique within each table. Can you write a single SQL query over the Databases table that takes a school as an argument (denoted `:s` in the query) and returns the school's ranking by score? For example, if the `<school,score>` tuples in the Databases table are:

```
<Stanford, 5.0>
<Berkeley, 4.8>
<Wisconsin, 4.7>
<Maryland, 4.3>
<OGI, 4.1>
<Illinois, 3.8>
```

then for `:s='Stanford'` the query result should be 1 (i.e., a single tuple with a single component with value 1); for `:s='Wisconsin'` the query result should be 3, for `:s='Illinois'` the query result should be 6, and so on. If you can write such a query in SQL, show it. If not, explain why not.

c) [5 points] Now suppose that scores may not be unique within each table. For example, the real data for the top eight schools in Computer Science is:

```
<MIT, 5.0>
<Stanford, 5.0>
<CMU, 4.9>
<Berkeley, 4.9>
<Cornell, 4.5>
<Illinois, 4.5>
<Washington, 4.3>
<Princeton, 4.2>
```

In this case, the rank of each school is the highest possible rank for all schools in its group. That is, for `:s='MIT'` and `:s='Stanford'` the ranking is 1, for `:s='CMU'` and `:s='Berkeley'` the ranking is 3, for `:s='Cornell'` and `:s='Illinois'` the ranking is 5, for `:s='Washington'` the ranking is 7, and for `:s='Princeton'` the ranking is 8.

Now can you write a single SQL query that takes a school as an argument (denoted `:s` in the query) and returns the school's ranking? If you can write such a SQL query, show it. If not, explain why not.

Problem 2 (10 pts.) Consider the following schedule of transactions T_1 and T_2, operating on database objects x, y, and z:

$$S = b\ w_1(x)\ r_2(x)\ w_2(y)\ c_2\ w_1(z)\ c_1\ e$$

Events b and e are the beginning and the end of the schedule, respectively. The following questions ask where certain events may occur in this schedule. Assume that two-phase locking is used. Express your answers by giving the two S events that must precede and follow the event in question. For example, transaction T_1 must lock x between b and $w_1(x)$.

a) Where can T_1 unlock x?

b) Where can T_1 lock z?

c) If undo logging is used, where can T_2 write to the log (on disk) the undo information for its write to y?

d) If redo logging is used, where can T_2 write to the log (on disk) the redo information for its write to y?

e) Schedule S cannot occur if *strict* two phase locking is used. Assuming we can move the T_2 actions in S, where can $w_2(y)$ occur in the new schedule?

Problem 3 (12 pts.) Briefly answer the following questions.

a) The rotational speed of a disk is increased by 50% (e.g., from 2000 RPM to 3000 RPM). What happens to the seek, rotational, and transfer times?

b) A B+ tree is used to index 1,000,000 records in a file. We want the tree to be at most 2 levels high. What is the smallest block (node) size we can use, in bytes? Assume keys are 4 bytes, and pointers are 4 bytes.

c) A file contains 5000 records, each of size 1000 bytes (fixed). The records are stored in blocks of size 2500 bytes, unspanned. What will be the best-case space utilization?

d) A precedence (or dependency) graph for a given execution schedule contains the following arcs (and no other arcs or nodes):

How many different serial schedules are equivalent to this schedule?

Problem 4 (15 pts.) We want to compute the bag-difference of two relations $R - S$, each of which may contain duplicate tuples. The machine on which the operation is executed has 101 main-memory buffers for query execution; put another way, the memory available for execution of this operation is equal to 101 disk blocks. However, it is also important to assume that there is *a little extra memory*, sufficient, for example, to add a mark or count for every tuple that is in main memory. The relations R and S each are packed tightly in some number of disk blocks. For each of the following relation sizes, give an algorithm for executing $R - S$ that is efficient, where cost is measured in the number of disk-block input or output (read or write, respectively) operations ("disk I/O's"). However, you do *not* count the number of disk-write operations needed to write the result of $R - S$ to disk. Estimate the cost of your algorithm in disk I/O's.

a) R occupies 200 blocks and S occupies 100 blocks.

b) R occupies 100 blocks and S occupies 200 blocks.

c) R occupies 12,000 blocks and S occupies 6000 blocks.

Problem 5 (8 pts.) Suppose we have a relation $R(a, b)$. 20% of the queries about R have the form:

```
SELECT *
FROM R
WHERE a = x;
```

where x can be any constant. The other 80% of the queries have the form:

```
SELECT *
FROM R
WHERE b = y;
```

where y can be any constant. We wish to build a partitioned-hash-function data structure on R, that is, a hash table where some number k of the bits of the bucket address are computed solely from the a-value of a tuple, and the remaining bits are computed solely from the b value of the tuple. Assume there are 10 bits total in bucket addresses; i.e., there are 1024 buckets. Each bucket is stored on as few disk blocks as it needs. Find the value of k, the number of bits of the bucket address devoted to a, that minimzes the average number of disk blocks retrieved in response to a random query about R. Explain your reasoning and show your calculation.