`University of Wisconsin-Madison
Computer Sciences Department

Database Qualifying Exam
Fall 2008

GENERAL INSTRUCTIONS

Answer each question in a separate book.

Indicate on the cover of *each* book the area of the exam, your code number, and the question answered in that book. On *one* of your books list the numbers of *all* the questions answered. Return all answer books in the folder provided. Additional answer books are available if needed.

Do not write your name on any answer book.

SPECIFIC INSTRUCTIONS

Answer **all** five (5) questions. Before beginning to answer a question make sure that you read it carefully. If you are confused about what the question means, state any assumptions that you have made in formulating your answer. Good luck!

The grade you will receive for each question will depend on both the correctness of your answer and the quality of the writing of your answer.

Policy on misprints and ambiguities:

The Exam Committee tries to proofread the exam as carefully as possible. Nevertheless, the exam sometimes contains misprints and ambiguities. If you are convinced a problem has been stated incorrectly, mention this to the proctor. If necessary, the proctor can contact a representative of the area to resolve problems during the *first hour* of the exam. In any case, you should indicate your interpretation of the problem in your written answer. Your interpretation should be such that the problem is nontrivial.

1. **Datalog and conjunctive queries:**

(a) Consider the following three conjunctive queries. For each pair of queries q and q', state if q contained in q', q' is contained in q, or they are incommensurate (neither is contained in the other.)

I. q1(X,Y,Z) :- a(X,W), b(W,Y), c(X,Z).

II. q2(X,Y,Y) :- a(X,W), b(W,Y), c(X,Z).

III. q3(X,Y,Z) :- a(X,W), b(U,Y), c(X,Z).

b. Does the following recursive Datalog program have an equivalent conjunctive query? If your answer is yes, give an equivalent query; if your answer is no, argue (informally is fine) why it does not.

t(X,Y) :- e(X,W), t(Z,Y).
t(X,Y) :- e(X,Y).

2. Transactional Memory and Database Transactions:

Recently the computer architecture community has been very interested in something they call “transactional memory.” A very simplified explanation of transactional memory is that it allows a programmer to declare a sequence of instructions to be a “transaction.” Two transactions “conflict” if either's write set overlaps with the other's read set or write set. (Note: the items read or written are memory words or cache lines, not logical objects like records, etc.) Transactional memory guarantees that only one of a pair of conflicting transactions will complete (the other will abort and all changes will be undone.) The primary purpose of transactional memory is to provide mutual exclusion without operating system locks or semaphores.

A. What are some key differences between transactional memory transactions and database transactions?

B. Would transactional memory be helpful in implementing database transactions? Why or why not?

C. Can you think of places in a DBMS other than concurrency control that transactional memory might be useful to someone implementing a DBMS?

3. Indexing Spatial Objects

Consider indexing two-dimensional spatial polylines (like roads or rivers) using a spatial index. You are told that 10% of the spatial objects are long, and run diagonally across the space. The remaining objects are short and have a random orientation. The queries on this data set are window queries, i.e., the query specifies a rectangular box, and we need to find all objects that overlap with the query box.

Propose an efficient indexing technique for this data set. Analyze the impact on your solution as the percentage of the long objects increases.

4. Multicore Parallel Joins

Consider a database server with a very large main memory, large enough that the database completely fits in main memory. Now consider running the DBMS on the following hypothetical processor that crudely resembles many of the current processors, which have multiple processing cores. The processor has 16 processing cores. Each core has a CPU and a private L1 cache. This cache is 64KB in size. The entire processor also has a shared L2 cache, which is 64MB in size. Data in the L1 and L2 caches is organized in units of 64bytes, so when the CPU requests even a single byte of memory, 64 bytes of data is fetched into the cache (conceptually for the purpose of caching the memory is seen as an array of 64-byte blocks). Assume that both the L1 and L2 caches use an LRU replacement policy. When the processor requests a piece of data that is not in the L1 cache, the L2 cache is searched for that data. If the data is not present in the L2 cache then the data is fetched from main memory to the L2 and then to L1 (so there are two copies of the data). Because of sharing, it is possible that the copy of the L2 data is replaced from that cache while some L1cache still has that copy.

How would you adapt hybrid hash join to run in this environment? Describe each component of the algorithm, including your choice of the algorithm for joining each partition, the partitioning strategy, and the number of partitions.

5. Temporal Databases

a) Briefly define the following terms, as discussed in the paper “A Taxonomy of Time in Databases”: transaction time, valid time, user-defined time, and temporal databases.

b) Let D be a static database whose schema consists of two tables: EMPS(eid, ename, salary, dept-id) and DEPTS(did, dname, location). Discuss how you can revise D's schema to support transaction time.

c) Supposed D can be modified (i.e., tuples inserted, deleted, or updated) by users u1, u2... or un (where each ui is a user id). Can you use the above revised schema of D to also capture user updates? That is, can you capture the facts that a particular user carried out a modification to the database at a particular time (e.g., user u3 deleted all tuples whose salary are under 20K from table EMPS at time 11:03pm on Sep 3, 2008)?

If not, show how you can revise D's schema to capture both transaction time and user activities. Briefly discuss any possible limitations of your solution.