UNIVERSITY OF WISCONSIN-MADISON
COMPUTER SCIENCES DEPARTMENT

DATABASE QUALIFYING EXAMS
FALL 1992

BREADTH EXAM

Answer any *four* (4) of the following *five* (5) questions. You may *not* select questions from the Depth Exam. Before beginning to answer a question, be sure to read it carefully. If you are confused about what the question means, state any assumptions you have made in formulating your answer. Good Luck!

(1) Concurrency Control

Consider a quorum consensus algorithm that employs locking for concurrency control in distributed database systems.

(a) Describe the operation of this algorithm.

(b) What problem does it address, and how?

(c) Does it prevent deadlocks in a distributed DBMS? Explain.

(2) Buffering Strategies

Consider the following simple buffer replacement policy based upon LOVE/HATE hints: whenever a page is brought into the buffer pool, it is either marked as a "LOVED" page or a "HATED" page. If it becomes necessary to remove a page from the buffer pool, the buffer manager always prefers a "HATED" page to a "LOVED" page. (That is, a "LOVED" page will be removed from the buffer pool only if there is currently no unpinned "HATED" page in the buffer pool.) Within the HATED pages, LRU is used to pick a page to remove; LRU is also used within the LOVED pages.

(a) Consider the following four operations in a relational database system: an indexed exact-match lookup, a sort-merge join, a nested-loops with index join, and a file scan. For each, tell which of the pages accessed should be marked as "LOVED" and which should be marked as "HATED".

(b) For each of the operations discussed in (a), describe how the performance of the DBMIN policy would compare with the performance of the LOVE/HATE policy.

(c) Discuss the relative performance of the DBMIN and the LOVE/HATE buffer management policies in a multi-user (i.e. multiple concurrent queries) environment.

(d) Propose an extension to the LOVE/HATE algorithm intended to fix any problems you may have cited in its performance in part (c).

(3) **Database System History**

System R, developed at IBM San Jose Research Center, was one of the first implementations of a full-function relational DBMS. What were the important results and lessons from the System R project? That is, list both the lasting contributions and the major mistakes/problems that were encountered. For each item on your list, support the item with a brief but clear explanation.

(4) **Relational Queries**

Consider the following relational schema, where keys are italicized:

SUPPLIER(*sno*, sname, city, state)
PART(*pno*, pname, color, weight)
CATALOG(*supno*, *partno*, price)

Write the following query in each of the query languages indicated below.

QUERY: Print the names and cities of each California supplier from whom one could order all — not just any, but **all** — of the purple-colored parts.

(a) Write the above query in the relational algebra.
(b) Now write the above query in the tuple relational calculus.
(c) Next, write the above query in SQL.
(d) Finally, write the above query in QUEL.

(5) **Physical Database Design**

Assume that a database system supports both primary and secondary B^+-trees, extendible hashing, and linear hashing as its storage structures. Let the following be the schema of a database managed by that system, storing information about some company, where the meaning of the attributes is straightforward and the primary keys are indicated in italics:

EMP (*ssn*, name, sal, sex, dpt)
DEPT (*dpt*, name, floor, mgr_ssn)

The following queries make up the bulk of the workload on this database (we ignore **from** clauses whenever they are unnecessary for clarity):

(Q1) **select** EMP.name **where** EMP.sal > 50K
(Q2) **select** EMP.name **where** EMP.sex="f"
(Q3) **select** EMP.name **where** EMP.ssn=DEPT.mgr_ssn **and** DEPT.floor = 8
(Q4) **select** EMP.name **where** EMP.dpt=DEPT.dpt **and** DEPT.floor = 8
(Q5) **select** EMP.name **where** EMP.sal **in**
(**select** avg(e.sal) **from** EMP e **where** e.sal > 30K **and** e.ssn=DEPT.mgr_ssn)

Also assume that the EMP relation contains 100,000 tuples, the DEPT relation contains 1000

tuples, employee salaries are uniformly distributed between 20K and 100K, and that the company is an equal opportunity employer (e.g., there is no discrimination based on sex). For each of the following questions, provide an answer together with an explanation, **independent** of your answers to the other questions.

(a) For (Q1), would you build a B^+-tree or an extendible hashing index on EMP.sal?

(b) For (Q2), would you build an extendible or a linear hashing index on EMP.sex?

(c) For (Q3) and (Q4), you decide that one of EMP.ssn and EMP.dpt will have a primary linear hashing index and the other will have a secondary linear hashing index. Which one has what?

(d) For (Q5), would you build a B^+-tree or a linear hashing index on EMP.sal? Assume that the only other index is a secondary linear hashing index on EMP.ssn.

DEPTH EXAM

Begin by taking the breadth exam, following the instructions given there. Then answer *three* (3) of the following *four* (4) questions.

(1) **Join Algorithm**

(a) Consider a hybrid hash join of relations R and S. Suppose that we have available M memory pages, and that the number of pages in R is RPAGES while the number of pages in S is SPAGES, where RPAGES < SPAGES. Let a sequential I/O take IOSEQ msec, while a random I/O takes IORAND msec. Give a formula for the I/O time for the hybrid hash join of R and S (your formula should involve the parameter "numPartitions", which is the number of partitions used by the join algorithm). Explain the derivation of your formula.

(b) Discuss when increasing the number of partitions will improve the performance of the hybrid hash join algorithm (if it ever will!), and when it will hurt the performance of the algorithm. Based on this, how many partitions should be used?

(c) [Part (c) is not intended to make use of any of your results from parts (a) and (b).] Suppose we are joining two relations, both of which have more pages than the square of the number of memory pages. Describe a hash-based join algorithm for such a join, and discuss the motivation for the decisions that you made in designing the algorithm.

(2) **Models of Persistence**

In AT&T's object-oriented database system ODE, persistence of objects is termed to be "persistence by allocation" while Altair's O2 database system implements "persistence by reachability".

(a) Explain what both these concepts mean.

(b) Illustrate the differences with one or more examples.

(c) What are the advantages/disadvantages of each?

(d) Does ODE provide persistence orthogonal to type? Explain, giving examples to clarify your explanation.

(3) **Parallel Database Systems**

The GAMMA parallel DBMS was designed for use on a "shared nothing" machine. Consider instead the problem of designing/implementing a parallel relational DBMS for a "shared everything" machine, where all of the processors are able to access all of the system's memory and disk drives. Sketch a design for such a system by explaining what you would do the same way, and what you would do differently, relative to Gamma. What DBMS components/problems would be easier (or harder) to handle in a "shared everything" context? How would such a system compare to Gamma in terms of potential advantages and disadvantages?

(4) **Recursion**

Consider a directed acyclic graph that is seven levels deep and its representation as a binary relation. Computing the transitive closure of this graph can be represented by a linear recursive logic program. The seminaive algorithm applied on this linear program computes the transitive closure by performing seven joins and six unions.

(a) Is there an algorithm that computes the transitive closure in a fewer number of joins? If there is one, describe it, explain why it involves fewer operations, and elaborate on whether that algorithm is likely to win in practice or not. If there is none, explain why.

(b) Give the equivalent expression of the seminaive algorithm in relational algebra.

(c) What is the relationship between the seminaive algorithm and magic sets?