UNIVERSITY OF WISCONSIN - MADISON
COMPUTER SCIENCES DEPARTMENT

DATABASE QUALIFYING EXAM
SPRING 1996

Answer any *four* (4) of the following *five* (5) questions. 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 Queries

Consider the following relations:

Student(*snum:* `integer`, *sname:* `string`, *major :* `string`, *level:* `string`, *age:* `integer`)
Class(*name:* `string`, *meets_at:* `time`, *room:* `string`, *fid:* `integer`)
Enrolled(*snum:* `integer`, *cname:* `string`)
Faculty(*fid*: `integer`, *fname:* `string`, *deptid :* `integer`)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. The keys are underlined in each relation.

1. Write the following queries in SQL. No duplicates should be printed in any of the answers.
   (a) Find the names of faculty members who teach in every room in which some class is taught.
   (b) Print the Level and the average age of students for that Level, for each Level.
   (c) Find the names of students who are enrolled in the maximum number of classes.
2. For each of the above queries, write it in relational algebra if this is possible. If not, briefly explain why.

2 Materialized Views in Query Optimization

A *materialized view* is the result of executing a view definition that is explicitly stored in the database; database updates are propagated to the materialized view to keep it consistent.

1. Given a view definition, explain when one would want to materialize it and when not.
2. Are there any restrictions in indexing materialized views or can it be treated as a regular relation in that respect?
3. Consider a materialized view V that is simply defined as the join between two relations R and S (no selection, no projection). Give an example of a query where one could use V to process it but using R and S is cheaper, and one where the opposite holds.
4. Consider materialized views that are simply defined as joins between relations (no selection, no projection). How would the Selinger-style query optimization algorithm need to be changed to accomodate the use of such materialized views to process queries?
5. Consider materialized views that are defined using arbitrary select-project-join queries. What new difficulties arise in using such views in query answering?

3 Concurrency Control in Spatial Access Methods

With the increasing interest in geographic information systems, R-Trees and their variants have become very popular spatial indexing methods.

1. Describe the R-tree indexing structure and how lookup operations work.
2. Consider the operation of an R-tree in a multiuser environment and assume that you wish to support page-level locking using a non-2PL for more concurrency.
   (a) Design a non-2PL algorithm for R-trees. Explain how your algorithm increases concurrency for both readers, writers, and a mix of readers and writers.
   (b) How is your algorithm different from the corresponding B-tree algorithm? What feature of the R-Tree structure motivated this difference?

4 Relational Database Design

1. Consider a relation R with five attributes ABCDE. You are given the following dependencies: A $\rightarrow$ B, BC $\rightarrow$ E, ED $\rightarrow$ A.
   (a) List all keys for R.
   (b) Is R in 3NF?
   (c) Is R in BCNF?
2. Prove that if R is in 3NF and every key is simple (i.e., consists of a single attribute) then R is in BCNF.

3. Give an algorithm for testing whether a relation scheme is in BCNF. The algorithm should be polynomial in the size of the set of given FDs. (The 'size' is the sum over all FDs of the number of attributes that appear in the FD.)

4. Is there a polynomial algorithm for testing whether a relation scheme is in 3NF? Explain briefly.

5 Logic and Databases

Consider the following recursive program for the transitive closure:

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

Assume that e is a base relation, while t is defined only by these two rules.

1. Give a four-tuple instance of e such that seminaive evaluation of t and naive evaluation of t make the same number of tuple inferences.

2. Give a four-tuple instance of e such that the seminaive evaluation of t makes far fewer tuple inferences than does naive.

Now consider the query t(1,Y)? over this same transitive closure program.

1. Show the magic sets rewriting of the program for this query.

2. Give a four-tuple instance of e such that the seminaive evaluation of the rewritten program materializes the same set of tuples for t as does seminaive applied to the original program.

3. Give a four-tuple instance of e such that the seminaive evaluation of the rewritten program materializes fewer tuples of t than does seminaive applied to the original program.