University of Wisconsin-Madison
Computer Sciences Department

Database Qualifying Exam
Spring 2014

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** four (4) 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. **Conjunctive Queries:**

(a) Consider the following three conjunctive queries expressed in Datalog. 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.) Note: you may find it useful to use techniques similar to the tableau mapping techniques presented in the Aho et al. "Equivalence Among Relational Expressions" paper.

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

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

III. q3(X,Y,Z) :- a(X,Y), 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 why it does not.

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

2. ARIES Recovery:

Part 1: During the REDO pass, when we are deciding whether to redo the operation in a log record *l* for a page *p*, the claim is that we don't even have to check the pageLSN for *p* if *p* is not in the Dirty Page Table, or if the page's recoveryLSN in the DPT is greater than the LSN of *l*.

a) Why is this true?
b) How is it even possible that we are considering an update log record for a page *p* but *p* is not in the DPT?

Part 2: Consider the three points in the log:

i) beginning of the analysis phase,
ii) beginning of the REDO phase
iii) earliest log record considered by the UNDO phase.

Is it possible that these three points could appear, from earliest to latest, in the order (i), (ii), (iii)? Explain your answer.

3. Concurrency Control

Jane Zany has just taken on a job in a new database startup, DataDrop, and is charged with improving the concurrency control component of their storage manager. The current DataDrop storage manager implements the classic six locking modes that are discussed in the Gray et al.'76 paper. Jane is asked to consider adding Increment (I) and Decrement (D) lock modes to the storage manager. An I (D) lock is used to protect an increment (decrement) operation.

Part 1: How would such lock modes work with the existing lock modes? You need to provide a clear answer about the compatibility of these locks with the existing locks and with each other, and any changes to the hierarchical locking protocol.

Part 2: Now assume that this DataDrop system also keeps a materialized count aggregate views on a view definition that involves a single primary key-foreign key join and a GROUP BY clause. Would the addition of I/D locks help with keeping this view updated when there are changes to the underlying base tables? If yes, explain when it would be advantageous to have these I/D locks. If no, explain why.

4. Entity Matching

This problem focuses on the topic of entity matching (a.k.a. record linkage, entity resolution, data matching, etc.). Consider two tables *A* and *B* with identical schemas that describe persons:

A(firstName, lastName, phone, affiliation, position)
B(firstName, lastName, phone, affiliation, position)

a) Briefly describe a rule-based approach to match Tables *A* and *B*, that is, to find pairs of tuples *(x,y)*, where *x* is from Table *A* and *y* is from Table *B*, that refer to the same real-world person.

b) Briefly describe a learning-based approach to match Tables *A* and *B*. Compare and contrast the rule-based approach with the learning-based one (that is, the pros and cons of each).

c) In practice, a developer often wants to pre-process the tables before matching them, to improve matching accuracy. A common pre-processing step is to identify synonyms and replace them with the same canonical string. For example, a developer may determine that strings "UW-Madison", "UWisc", and "Univ of Wisconsin" in column "*affiliation*" are synonyms, and replace all of them with the canonical string "University of Wisconsin Madison."

Manually examining all the values in the column “affiliation”, say, to find synonyms is often not feasible, because each table can have tens of thousands or hundreds of thousands of tuples. Describe a semi-automatic or automatic algorithm that helps the developer find synonyms in the column “affiliation” (and more generally, in any column of two tables to be matched). Describe how that synonym-discovery algorithm can be used in the process of matching the two tables *A* and *B*.