`University of Wisconsin-Madison
Computer Sciences Department

Database Qualifying Exam
Spring 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. Concurrency Control (Spring 2002)

(a) In optimistic concurrency control as proposed by Kung and Robinson, one of the conditions that guarantees serializability can be stated as: if TN(Ti) < TN(Tj), then Write-Phase(Ti) ends before Write-Phase(Tj) begins and Write-Set(Ti) and Read-Set(Tj) do not overlap. Suppose that during the execution of a set of transactions T1, T2, …, Tn, each pair of transactions Ti and Tj in this set satisfy the condition. Explain why this schedule is serializable (it is not sufficient to just cite Kung and Robinson.)

(b) What is the relationship between the set of schedules allowed by Kung and Robinson and the set of schedules allowed at degree 3 consistency as defined by Gray?

(c) Is non-strict 2PL identical to any of the levels of consistency defined by Gray et al.? Explain your answer.

2. Expressive Power of SQL (spring 02)

Consider a subset of the SQL:1999 query language that includes recursive queries, but does not include grouping, aggregate operations, or cursors (i.e., no order-by). Nested queries and set-operations (e.g., union and difference) are allowed, and you can use views to store intermediate results. You can use the DISTINCT clause if you wish. You cannot write programs in a host language and embed SQL calls; you must stay strictly within the SQL query language with the above restrictions. Consider the relation Employees(eid, ename, mid, dept, sal) and answer the following questions; eid is the key and mid identifies the manager of the employee.

(a) Can you count the number of employee tuples?

(b) Can you find out if any departments employ more than one person?

(c) Can you identify departments that employ more than one person?

(d) Can you identify which department pays the highest salary?

(e) Can you identify employees who make less than someone they (directly or indirectly) manage?

2. View Updates (Fall 03)

The following questions deal with updating relational views.

(a) Define what it means to translate an update on a view.

(b) It is desirable that a translation of an update on a view should not change the database unnecessarily. State such a condition precisely.

(c) Give an example of a view update for which there is no reasonable translation.

(d) Give an example of a view update for which there is more than one reasonable translation.

4. XML and XQUERY (Fall 06)

Consider a database consisting of a collection of XML book elements as shown below

```
<bib>
    <book>
        <title> book1 </title>
        <publisher>  Morgan Kauffmann
        </publisher>
        <year> 1998 </year>
        <author> author1 </author>
        <author>  author2 </author>
        <author>  author3 </author>
    </book>
    <book>
        <title> book2 </title>
        <publisher>  Morgan Kauffmann
        </publisher>
        <year> 1996 </year>
        <author> author1 </author>
        <author>  author4 </author>
    </book>
```

-
-
-

```
</bib>
```

There are two competing approaches for storing and querying collections of XML documents. One is to construct a "native" database system designed specifically for XML documents. The second is to use a relational database system.

a. Design a relational schema for the XML document above. Make sure that your design can handle an arbitrary number of author elements for each book (your design should be extensible to handle an arbitrary number of each element type in general).

b. Given your mapping, translate the following XML query into SQL.

```
FOR $x IN document("bib.xml")/bib/book
WHERE $x/year > 1995
RETURN $x/author
```

c. Comment on the relative efficiency of storing XML documents as trees on disk (perhaps with each element as a separate record on a slotted page) versus the relational approach with respect to the efficiency of query execution.

Consider the following query in XQuery:

```
FOR $p IN distinct(document("bib.xml")//publisher)
LET $b := document("bib.xml")/book[publisher = $p]
   WHERE count($b) > 100
RETURN $p
```

d. Explain what it computes.

e. Modify the above query to only consider books in which some paragraph (accessed as .../book//paragraph) contains the word "sailing".

5. Web Search Engines (spring 05)

(a) How is a document represented in the vector-space model?

(b) What is TF*IDF? How is it related to the vector space model?

(c) Given a document d and a collection of documents D, how would you rank the documents in D by similarity to d using the vector space model?

(d) Google uses a measure called page rank to order search results. Explain what page rank is and describe the underlying intuition. Also, explain the relative roles of page rank and traditional vector-space ranking in Google.

(e) Search engines must scale to billions of documents and millions of queries per day. Describe how a typical search engine can be parallelized to achieve these goals, using a cluster of commodity computers (e.g., PCs running Linux).