Stanford University Computer Science Department
2006 Comprehensive Exam in Databases

- The exam is *open book and notes*, but not open computer.
- There are six problems on the exam, with a varying number of points for each problem and subproblem for a total of 60 points (i.e., one point per minute). It is suggested that you look through the entire exam before getting started, in order to plan your strategy.
- Please write your solutions in the spaces provided on the exam. Make sure your solutions are neat and clearly marked.
- *Simplicity and clarity of solutions will count.* You may get as few as 0 points for a problem if your solution is far more complicated than necessary, or if we cannot understand your solution.

MAGIC NUMBER: ____________

Problem	1	2	3	4	5	6	TOTAL
Max. points	4	14	12	12	12	6	60
Points							

1. *Data Models* **(4 points)**

   (a) Why does the E/R model require a notion of "weak entity set"? Be as specific as you can. (For example, "Because some entity sets don't have a key" is not an acceptable answer.)

   (b) Why do object-oriented models like ODL *not* have a notion equivalent to weak entity sets? Again, be as specific as you can.

2. *Relational Algebra* **(14 points)**

   Suppose there are relations $R(A, B)$ and $S(B, C)$. Write the following queries in *relational algebra* using standard operators only. You may write either single expressions or sequences of assignments to temporary variables, the last of which is the desired result. If you want your answers to be considered for partial credit you may provide some explanation of your reasoning, but it is sufficient just to provide the correct relational algebra queries.

   (a) (4 points) The tuples of R such that their B-component is neither a B-component of a tuple of S nor a C-component of a tuple of S.

(b) (4 points) The A-values that appear in at least two distinct tuples of R.

(c) (6 points) The A-values a such that:

i. a appears in at least one tuple of R, and

ii. Every B-value b, for which (a, b) is a tuple of R, appears as a B-value in at least one tuple of S.

3. *Relational Design* **(12 points)**

Suppose we are given relation $R(A, B, C, D)$ with functional dependencies $AB \rightarrow C$, $CD \rightarrow A$, and $B \rightarrow D$. Answer each of the following four questions. In each part, you may choose to justify your answer for partial credit, but it is sufficient just to provide an answer.

(a) (4 points) Find all the (minimal) keys of R.

(b) (2 points) How many superkeys does R have?

(c) (3 points) Of the given FD's, which violate Boyce-Codd normal form?

(d) (3 points) Of the given FD's, which violate third normal form?

4. *XML* **(12 points)**

   Consider the following four DTDs for XML documents containing data about books and authors. You may assume that `isbn` and `ssn` are "keys" for books and authors, in the sense that no two actual books have the same `isbn`, and no two actual authors have the same `ssn`.

```
DTD1: <!DOCTYPE BA [
         <!ELEMENT BA (Book*)>
         <!ELEMENT Book (Title, Author+)>
         <!ELEMENT Title (#PCDATA)>
         <!ELEMENT Author>
         <!ATTLIST Author ssn ID name CDATA> ]>

DTD2: <!DOCTYPE BA [
         <!ELEMENT BA (Author*)>
         <!ELEMENT Author (Book)>
         <!ATTLIST Author ssn ID name CDATA>
         <!ELEMENT Book (Title)>
         <!ELEMENT Title (#PCDATA)> ]>

DTD3: <!DOCTYPE BA [
         <!ELEMENT BA (Book*, Author*)>
         <!ELEMENT Book (Title)>
         <!ATTLIST Book isbn ID author IDREF>
         <!ELEMENT Title (#PCDATA)>
         <!ELEMENT Author>
         <!ATTLIST Author ssn ID name CDATA> ]>

DTD4: <!DOCTYPE BA [
         <!ELEMENT BA (Author*, Book*)>
         <!ELEMENT Author>
         <!ATTLIST Author ssn ID book IDREFS>
         <!ELEMENT Book (Title)>
         <!ATTLIST Book isbn ID>
         <!ELEMENT Title (#PCDATA)> ]>
```

   (a) Which of the DTDs can be used for XML documents that encode *many-many* relationships between books and authors? (circle those that can) `DTD1 DTD2 DTD3 DTD4`

   (b) Which of the DTDs can be used for XML documents that encode *many-one* relationships from books to authors? (circle those that can) `DTD1 DTD2 DTD3 DTD4`

   (c) Which of the DTDs can be used for XML documents that encode *one-many* relationships from books to authors? (circle those that can) `DTD1 DTD2 DTD3 DTD4`

   (d) Which of the DTDs can be used for XML documents that encode *one-one* relationships between books and authors? (circle those that can) `DTD1 DTD2 DTD3 DTD4`

5. *SQL* **(12 points)**

   Consider the following SQL schema for a database about schools in a university and departments in those schools. Data types are omitted for brevity.

   ```
   create table School(schoolName primary key, dean)
   create table Dept(deptName primary key,
                     schoolName references School on delete cascade)
   ```

   Write a *single SQL statement* that deletes all schools with more than 10 departments and also deletes all departments in those schools.

6. *Transactions* **(6 points)**

   Consider a relation `Temperatures(time,temp)` and the following transaction `T` with SQL isolation level `REPEATABLE READ`:

   ```
     T: (Q1) Select Avg(temp) From Temperatures;
        (Q2) Select Avg(temp) From Temperatures;
   ```

   Is it possible for a concurrently-running transaction to cause queries `Q1` and `Q2` to return different values? If so, show the simplest such transaction. If not, explain why not.