CS145 Midterm Examination

Spring 2003, Prof. Widom

- Please read all instructions (including these) carefully.
- There are 8 problems on the exam, with a varying number of points for each problem and subproblem for a total of 75 points to be completed in 75 minutes. *You should look through the entire exam before getting started, in order to plan your strategy.*
- The exam is closed book and closed notes, but you may refer to your three pages of prepared notes.
- Please write your solutions in the spaces provided on the exam. Make sure your solutions are neat and clearly marked. You may use the blank areas and backs of the exam pages for scratch work. Please do not use any additional scratch paper.
- *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.

NAME: _____

In accordance with both the letter and spirit of the Honor Code, I have neither given nor received assistance on this examination.

SIGNATURE: _____

Problem	1	2	3	4	5	6	7	8	TOTAL
Max. points	10	5	10	6	18	4	4	18	75
Points									

1. **Relational Algebra** (10 points)

   Consider a relation *Took*(*name*, *class*, *quarter*) whose tuples record that a student took a given class in a given quarter. You may assume that no two students have the same name, and there is no key for this relation except all three attributes together. Write a relational algebra expression to find all pairs of students who have never taken a class together—i.e., have never taken the same class in the same quarter. Make sure to return each pair of students only once. (For example, if your expression returns `<Mary,Fred>` then it should not also return `<Fred, Mary>` or any additional copies of `<Mary,Fred>`.) Your expression will be graded on simplicity as well as correctness.

2. **SQL** (5 points)

   Consider a one-attribute table $T(A)$. Write the shortest SQL query you can come up with that returns all values for attribute A that appear in table T at least 3 times. You may assume there are no NULLs, and do not include duplicate A values in your result.

3. **SQL Query Equivalence** (10 points)

Consider a SQL table $T(K, V)$ where K is a key and NULL values are not permitted in either column. Consider the following three queries:

```
Q1: select V from T
    where V >= any (select V from T)

Q2: select V from T as T1
    where V > all (select V from T as T2 where T2.K <> T1.K)

Q3: select max(V) from T
```

Circle one of the following statements.

(a) $Q1$, $Q2$, and $Q3$ are all equivalent.

(b) $Q1$ and $Q2$ are equivalent; $Q3$ may produce a different answer on some databases.

(c) $Q1$ and $Q3$ are equivalent; $Q2$ may produce a different answer on some databases.

(d) $Q2$ and $Q3$ are equivalent; $Q1$ may produce a different answer on some databases.

(e) $Q1$, $Q2$, and $Q3$ may all produce different answers on some databases.

If you chose any answer other than (a), show the *simplest* single instance of T you can come up with that demonstrates the nonequivalence(s). In this case, be sure to show the result of all three queries on your instance of T.

4. **SQL and Relational Algebra** (6 points, 3 per part)

Consider a table *Chess*(*player, points*) recording points earned by chess players. Each attribute independently is a key—that is, each player appears only once and there are never ties in points—and there are no NULL values. Consider the following SQL query over this table:

```
select player
from Chess C1
where 2 > (select count(*) from Chess C2
           where C2.points > C1.points)
```

(a) State in English what is returned by this query. Please state the interpretation succinctly—the correct answer requires only a few well-chosen words.

(b) Can an equivalent query be written in relational algebra? Circle one: YES NO

(Even if you circled YES you do *not* need to provide an equivalent query.)

5. **XML, XPath, and XQuery** (18 points)

Consider the following XML DTD:

```
<!ELEMENT Univ (Course+, Prof+)>
<!ELEMENT Course (Title, Eval*)>
<!ATTLIST Course Number ID #REQUIRED Instructor IDREF #IMPLIED>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Eval (#PCDATA)>
<!ATTLIST Eval Score CDATA #REQUIRED>
<!ELEMENT Prof EMPTY>
<!ATTLIST Prof Name ID #REQUIRED Teaches IDREF #IMPLIED>
```

(problem continues on next 2 pages)

(a) (5 points) Show the smallest example XML document you can come up with (i.e., a document containing the fewest possible elements and attributes) such that the document conforms to the above DTD with `!DOCTYPE` (outermost tag) `Univ`.

(b) (5 points) Write an XPath expression to find all boring CS classes. More specifically, return the course number for all courses that have "CS" in their title and at least two evaluations that include the word "boring."

(c) (4 points) State in English what is returned by the following query in XQuery. Please state the English interpretation succinctly and in terms of the real-world objects being represented rather than details of the data itself. Under no circumstances should your description discuss procedural aspects of matching the query against the data.

```
let $as := //@Score
for $c in /Univ/Course[Eval]
let $cs := $c/Eval/@Score
where min($cs) > avg($as)
return $c
```

(d) (4 points) State in English what is returned by the following query in XQuery. Please state the English interpretation succinctly and "declaratively." Under no circumstances should your description discuss procedural aspects of matching the query against the data.

```
for $c in /Univ/Course
for $p in /Univ/Prof
where ($c/@Instructor = $p/@Name and $p/@Teaches <> $c/@Number)
   or ($p/@Teaches = $c/@Number and $c/@Instructor <> $p/@Name)
return <Pair>
         <Course>{$c/@Number}</Course>
         <Prof>{$p/@Name}</Prof>
       </Pair>
```

6. **Multivalued Dependencies** (4 points)

Consider a relation $R(A, B, C)$ where the only values for each attribute are 0 or 1. Consider an instance of R with 8 distinct tuples. Does $A \twoheadrightarrow B$ hold?

Circle one: YES NO

(ungraded scratch space)

7. **Normal Forms** (4 points, 2 per part)

Consider a relation R that always contains exactly one tuple.

(a) Is R in Boyce-Codd Normal Form (BCNF)? Circle one:

- Yes
- No
- Can't tell without seeing the actual schema and FD's
- Can't tell without seeing the actual data
- Can't tell without seeing the actual schema, FD's, and data

(b) Is R in Fourth Normal Form (4NF)? Circle one:

- Yes
- No
- Can't tell without seeing the actual schema, FD's, and MVD's
- Can't tell without seeing the actual data
- Can't tell without seeing the actual schema, FD's, MVD's, and data

8. **Dependencies and Normal Forms** (18 points, 6 per part)

Consider the following two relational schemas:

```
Schema 1: R(A,B,C)
Schema 2: R1(A,B), R2(A,C)
```

(a) Suppose that the only dependencies (functional or multivalued) that hold on the relations in these schemas are $A \rightarrow BC$ and all dependencies that follow from this one. Circle exactly two of the following statements.

- Schema 1 is in neither BCNF nor 4NF.
- Schema 1 is in BCNF but not 4NF.
- Schema 1 is in 4NF but not BCNF.
- Schema 1 is in both BCNF and 4NF.
- Schema 2 is in neither BCNF nor 4NF.
- Schema 2 is in BCNF but not 4NF.
- Schema 2 is in 4NF but not BCNF.
- Schema 2 is in both BCNF and 4NF.

(problem continues on next page)

(b) Now suppose that the only dependencies (functional or multivalued) that hold on the relations in these schemas are $BC \rightarrow A$, $A \rightarrow C$, and all dependencies that follow from these two. Circle exactly two of the following statements.

- Schema 1 is in neither BCNF nor 4NF.
- Schema 1 is in BCNF but not 4NF.
- Schema 1 is in 4NF but not BCNF.
- Schema 1 is in both BCNF and 4NF.
- Schema 2 is in neither BCNF nor 4NF.
- Schema 2 is in BCNF but not 4NF.
- Schema 2 is in 4NF but not BCNF.
- Schema 2 is in both BCNF and 4NF.

(c) Now suppose that the only dependencies (functional or multivalued) that hold on the relations in these schemas are $A \twoheadrightarrow B$, $A \twoheadrightarrow C$, and all dependencies that follow from these two. Circle exactly two of the following statements.

- Schema 1 is in neither BCNF nor 4NF.
- Schema 1 is in BCNF but not 4NF.
- Schema 1 is in 4NF but not BCNF.
- Schema 1 is in both BCNF and 4NF.
- Schema 2 is in neither BCNF nor 4NF.
- Schema 2 is in BCNF but not 4NF.
- Schema 2 is in 4NF but not BCNF.
- Schema 2 is in both BCNF and 4NF.

(ungraded scratch space)