CS145 Final Examination

Spring 2002, Prof. Widom

- Please read all instructions (including these) carefully.
- There are 10 problems on the exam, with a varying number of points for each problem and subproblem for a total of 120 points. *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	9	10	TOTAL
Max. points	8	6	12	8	10	24	13	17	10	12	120
Points											

1. **E/R and Relational Design** (8 points: 4 each for a and b)

   Consider an entity-relationship (E/R) diagram containing two entity sets *E1* and *E2* with key attributes *A1* and *A2* respectively. Consider a relationship R that is many-one from *E1* to *E2* with referential integrity. The standard translation from this E/R diagram to relations would produce three relations, one each corresponding to *E1*, *E2*, and R.

   (a) Does it make sense to combine the relation for R with either of the relations for *E1* or *E2*? If so, with which one(s)? If not, write "no."

   (b) If your answer to (a) did not include one of *E1* or *E2* (or both), what specific relational design criterion may be violated by combining R with the entity set?

2. **ODL and E/R** (6 points)

   Consider the following ODL class definition. Assume that ODL classes `D` and `E` have already been defined.

```
class C (key (a)) {
  attribute integer a;
  relationship D rel1;
  relationship Set<E> rel2; }
```

   You can think of the two relationships in class `C` as encoding a three-way relationship between classes `C`, `D`, and `E`. Draw an entity-relationship (E/R) diagram that depicts this relationship. The diagram should contain entity sets corresponding to classes `C`, `D`, and `E` (attributes can be omitted), and one three-way relationship. Specify arrows in your diagram to indicate any multiplicities you can determine from the class definition above.

3. **Transactions** (12 points: 4 for each part)

Consider a relation `Employee(ID,salary)` where `ID` is the key. Suppose there are two tuples in `Employee` with initial values `(A,20)` and `(B,30)`. Consider the following two concurrent transactions:

```
T1: begin transaction;
    update Employee set salary = 2*salary where ID='A';
    update Employee set salary = salary+10 where ID='A';
    commit;

T2: begin transaction;
    select avg(salary) as sal1 from Employee;
    select avg(salary) as sal2 from Employee;
    commit;
```

Suppose that transaction T1 executes with isolation level *serializable*.

(a) If transaction T2 also executes with isolation level *serializable*, what are the possible pairs of values `sal1` and `sal2` returned by T2? Please carefully indicate all possible pairs.

(b) If transaction T2 executes with isolation level *read committed*, what are the possible pairs of values `sal1` and `sal2` returned by T2? Please carefully indicate all possible pairs.

(c) If transaction T2 executes with isolation level *read uncommitted*, what are the possible pairs of values `sal1` and `sal2` returned by T2? Please carefully indicate all possible pairs.

4. **Constraints** (8 points)

Consider two relations `R(A integer)` and `S(B integer)` and the following constraint:

```
create assertion Mystery check (
   not exists (select * from R
               where A > (select avg(B) from S)) )
```

Using SQL-99 constraints (not Oracle's), can this general assertion be rewritten as one or more or tuple-level constraints on R and S? Circle one: YES   NO

If you chose YES, write the constraint(s). If you chose NO, briefly explain why not.

5. **Triggers** (10 points: 5 each for a and b)

Consider again the relation `Employee(ID,salary)` where `ID` is the key. Suppose the following SQL-99 trigger is defined on this relation:

```
create trigger AutoRaise
after insert on Employee
referencing new table as NT
update Employee
  set salary = salary + (select avg(salary) from NT)
```

Suppose that before the trigger is created the relation contains one tuple: `(A,50)`. Then, after trigger creation, two new tuples are inserted as the result of one statement: `(B,70)` and `(C,30)`.

(continued on next page)

(a) After the insertions and trigger execution(s), what are the final salaries for the three employees?

A: [ ] B: [ ] C: [ ]

(b) Suppose we add "`for each row`" to the trigger. Now, after the insertions and trigger execution(s), what are the final salaries for the three employees?

A: [ ] B: [ ] C: [ ]

6. **Recursion** (24 points: 4 for each part)

Consider a relation `Parent(X,Y)` where a tuple `(x,y)` in `Parent` specifies that person `x` is the parent of person `y`. The only key for `Parent` consists of both attributes together. We are interested in writing a recursive query to find all descendants of the person named "Eve." Parts (a)–(f) below each contain a `with` statement, without the final `select` query. For each `with` statement, choose one of the following three options and write the number in the box:

(1) Adding the final query "`select Y from Descendant`" produces the desired result (all descendants of "Eve").

(2) Adding the final query "`select Y from Descendant where X='Eve'`" produces the desired result, but the query in (1) does not produce the desired result.

(3) Neither the query in (1) nor the query in (2) produces the desired result.

For the purposes of this problem, assume that nonlinear recursion is permitted in SQL-99.

(a)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent)
     union
    (select Descendant.X, Parent.Y
     from Descendant, Parent
     where Descendant.Y=Parent.X) )
```

[ ]

(b)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent where X='Eve')
     union
    (select Descendant.X, Parent.Y
     from Descendant, Parent
     where Descendant.Y=Parent.X) )
```

(c)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent)
     union
    (select Parent.X, Descendant.Y
     from Parent, Descendant
     where Parent.Y = Descendant.X) )
```

(d)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent)
     union
    (select Parent.X, Descendant.Y
     from Parent, Descendant
     where Parent.Y = Descendant.X and Parent.X='Eve') )
```

(e)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent where X='Eve')
     union
    (select D1.X, D2.Y
     from Descendant D1, Descendant D2
     where D1.Y = D2.X) )
```

(f)
```
with recursive Descendant(X,Y) as
  ( (select X,Y from Parent)
     union
    (select D1.X, D2.Y
     from Descendant D1, Descendant D2
     where D1.Y = D2.X and D1.X='Eve') )
```

7. **Temporal Databases** (13 points: 2 for a, 3 for b, 8 for c)

Consider a temporal relation `login(user, machine, T)`, where `T` is the specially treated *timestamp* attribute. This relation records the times at which particular `user`s are logged into particular `machine`s. Consider the following query Q over temporal relation `login`. Q is written using temporal relational algebra as covered in class.

$$\Pi_{\texttt{user,m1,m2}}(\sigma_{\texttt{m1}<\texttt{m2}}(\rho_{\texttt{L1(user,m1,T)}}(\texttt{login}) \bowtie \rho_{\texttt{L2(user,m2,T)}}(\texttt{login})))$$

Suppose temporal relation `login` contains the following tuples, where for simplicity let us assume that times are denoted by integers.

user	machine	T
Susan	saga	[1,4], [8,12]
Susan	elaine	[2,5]
Susan	myth	[6,9], [12,15]
Fred	myth	[17,25]
Fred	elaine	[1,4], [9,20]

(a) What is the key for relation `login`?

(b) What is the schema of the result of query Q?

(c) List the tuples in the result of query Q.

8. **OLAP** (17 points: 2 each for a–d, 5 for e, 4 for f)

Consider the following *star schema*:

```
Scores(studentID REFERENCES Students(ID), quarter,
       courseNum REFERENCES Courses(num), exam#, score)
  // only key is [studentID,quarter,courseNum,exam#]
Students(ID, name)  // ID is key
Courses(num, dept)  // num is key
```

(a) Which table(s) in the schema are the *fact* table(s)?

(b) Which table(s) in the schema are the *dimension* table(s)?

(c) Which attribute(s) of the fact table(s) are the *dimension* attribute(s)?

(d) Which attribute(s) of the fact table(s) are the *dependent* attribute(s)?

(e) Consider the following three queries, intended to find the total exam score for all CS courses taken any spring quarter.

```
Q1: select sum(score)
    from Scores, Courses
    where courseNum=num and dept='CS' and quarter='spring'

Q2: select sum(score)
    from CUBE(Scores), Courses
    where courseNum=num and dept='CS'
    and quarter='spring' and studentID='*' and exam#='*'

Q3: select sum(score)
    from CUBE(Scores), Courses
    where courseNum=num and dept='CS'
    and quarter='spring' and studentID<>'*' and exam#<>'*'
```

Do all three queries return the same result? Circle one: YES NO

If your answer is NO, which one(s) return the same result and which one(s) return different results?

(f) Regardless of equivalent results, which of the three queries in part (e) would you expect to be the most efficient to execute? Think of efficiency as measured by the number of tuples satisfying the query's `where` clause. Circle one: Q1 Q2 Q3

9. **Data Mining** (10 points)

Consider the following *market basket* data:

`saleID`	`item`
1	beer
1	diapers
2	chips
2	diapers
2	soda
3	chips
3	soda

Which of the following association rules hold when we require *Support* > 0.4 and *Confidence* > 0.5? Circle the ones that hold.

beer $\rightarrow$ diapers
beer $\rightarrow$ chips
beer $\rightarrow$ soda
diapers $\rightarrow$ beer
diapers $\rightarrow$ chips
diapers $\rightarrow$ soda
chips $\rightarrow$ beer
chips $\rightarrow$ diapers
chips $\rightarrow$ soda
soda $\rightarrow$ beer
soda $\rightarrow$ diapers
soda $\rightarrow$ chips

10. **XSLT** (12 points: 4 for each part)

Consider the following XML data:

```
<ExamRatings>
  <Question number='1'>
     <Part number='a'>Hard</Part>
     <Part number='b'>Easy</Part>
  </Question>
  <Question number='2'>
     <Part number='a'>Trivial</Part>
     <Part number='b'>Terrifying</Part>
  </Question>
</ExamRatings>
```

Recall that XSLT contains the following two default rules (templates), with lowest priority:

```
<xsl:template match="*|/">
  <xsl:apply-templates/>
</xsl:template>

<xsl:template match="text()|@*">
  <xsl:value-of select="." />
</xsl:template>
```

For the purposes of this problem, assume that every XSLT result data set is bracketed by `<xsl:result>` and `</xsl:result>`.

(a) If only the default rules are applied, what is the result of XSLT processing over the above XML data?

(continued on next page)

(b) Suppose we add the following rule:

```
<xsl:template match="/">
</xsl:template>
```

Now what is the result of XSLT processing over the above XML data?

(c) Suppose we do not add the rule in part (b), but we do add the following two rules:

```
<xsl:template match="Part">
  <Rating> <xsl:value-of select="." /> </Rating>
</xsl:template>

<xsl:template match="@number">
</xsl:template>
```

Now what is the result of XSLT processing over the above XML data?