CS145 Final Examination

Autumn 2005, Prof. Widom

- Please read all instructions (including these) carefully.
- There are 11 problems on the exam, with a varying number of points for each problem and subproblem for a total of 120 points to be completed in 120 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	9	10	11	TOTAL
Max. points	16	8	18	10	12	6	10	12	10	10	8	120
Points												

1. **Triggers and Referential Integrity** (16 points; 8 per part)

   Consider tables $R(P, A)$ and $S(F, B)$. Suppose we want to use triggers to enforce a referential integrity constraint: $S.F$ is a foreign key referencing primary key $R.P$.

   (a) Fill in some or all of the blanks in the following skeleton to implement the "*on update cascade*" policy. Write the simplest trigger you can come up with and don't worry about performance.

```
create trigger UpdCascade
after update of P on R
referencing [          ]
for each row
when [          ]
[          ]  (trigger action)
```

   (b) Fill in some or all of the blanks in the following skeleton to implement the "*on delete cascade*" policy. (Note this skeleton does not include "`for each row`", which was included in the previous skeleton.) Write the simplest trigger you can come up with and don't worry about performance.

```
create trigger DelCascade
after delete on R
referencing [          ]
when [          ]
[          ]  (trigger action)
```

2. **Constraints** (8 points)

Consider a table `Pay(salary,bonus)` and the following SQL general assertion:

```
create assertion BonusControl check (
  not exists (select * from Pay
              where bonus > (select max(salary) from Pay)) )
```

Using SQL-99 constraints (not Oracle's), can we enforce this general assertion with one or more tuple-based constraints on table `Pay`? Circle one: YES NO

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

3. **Transactions** (18 points; 6 per part)

Consider a table `Item(name,price)` where `name` is a key. Suppose initially there are two tuples in `Item`: `(A,20)` and `(B,30)`. Consider the following two concurrent transactions, each of which runs once and commits. You may assume there are no other transactions in the system and that individual statements execute atomically.

```
T1: begin transaction
    S1: insert into Item values ('C',40)
    S2: update Item set price = price+30 where name='A'
    commit

T2: begin transaction
    S3: select avg(price) as p1 from Item
    S4: select avg(price) as p2 from Item
    commit
```

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

(problem continues on next page)

(a) If transaction `T2` also executes with isolation level *serializable*, what are the possible pairs of values `p1` and `p2` 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 `p1` and `p2` 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 `p1` and `p2` returned by `T2`? Please carefully indicate all possible pairs.

4. **Indexes** (10 points)

Consider the following tables in a SQL database:

```
Course(courseNum, dept)      // courseNum is a key
Enroll(studentID, courseNum) // <studentID,courseNum> is a key
```

Suppose there are three types of queries commonly asked on this schema:

- Given a course number, find the department offering that course.
- Match each student ID with all of the departments for which the student is enrolled in a course.
- Given a student ID, find all course numbers the student is enrolled in.

Here's the actual problem:

(a) (4 points) What is the minimum number of indexes needed to speed up all three types of queries? (Do not assume indexes are built automatically on keys.)

(b) (6 points) On which attributes should these indexes be created?

5. **Authorization** (12 points)

Consider the following tables in a SQL database:

```
Student(studentID, name, dorm) // studentID is a key
Major(studentID, major)        // <studentID,major> is a key
```

Suppose the owner (creator) of these tables is a user named "Hennessy," and Hennessy wants to grant to a user named "Etchemendy" the ability to read the studentIDs, names and dorms, as well as modify the names and dorms, for students with at least one major containing the string "Science" (and only those students). Is it possible to specify a command or sequence of commands that achieves this goal? If so, show it. If not, explain why not.

6. **Object-Relational SQL** (6 points)

Suppose you've created the following type and table in SQL-99. (Atomic type declarations such as `char` and `integer` have been omitted from the statements.)

```
create type BookInfo (title, author, year)
create table BookSale (book BookInfo, price, seller)
```

You decide to implement an *ordering relationship* for objects of type `BookInfo`, using *method-defined comparison*. (The ordering itself may be based on some combination of title, author, and year, but how it is defined is irrelevant to this problem.)

Once the ordering relationship is in place, you can write SQL queries that include comparisons ($=$, $<$, $>$, `in`, $<$ `all`, etc.) across values from column `BookSale.book`. State three additional SQL features on column `BookSale.book` that are enabled by the ordering relationship.

1: 2:

3:

7. **SQL Recursion** (10 points)

Consider a single-attribute table `Nums(n)` containing a set of numbers, and suppose the numbers in the table are currently 1 through 5, i.e.:

```
Nums(n) = (1),(2),(3),(4),(5)
```

Consider the following `With` statement in SQL-99. (Don't worry about SQL-99 restrictions on allowable types of recursion.)

```
with recursive Mystery(x) as
       (select x from Mystery)
        union
       (select sum(n) as x from Nums
        where n <= (select count(*)+1 from Mystery))
select sum(x) from Mystery
```

Suppose this query is executed over the current table `Nums` as given above. In the box, either write "Nonterminating" if the query may not terminate, or write the query result if it is guaranteed to terminate.

8. **OLAP: Cube and Rollup** (12 points; 6 per answer)

Consider a fact table in an OLAP application:

```
Facts(D1, D2, D3, val)
```

where `D1`–`D3` are *dimension attributes* and `val` is a *dependent* attribute. Suppose attributes `D1`, `D2`, and `D3` each take on 2 different values, and all combinations of values are present in table `Facts`.

(a) How many tuples are in the result of the following query? ☐

```
select D1, D2, D3, sum(val)
from Facts
group by D1, D2, D3 WITH CUBE
```

(b) How many tuples are in the result of the following query? ☐

```
select D1, D2, D3, sum(val)
from Facts
group by D1, D2, D3 WITH ROLLUP
```

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

Consider the following *market basket* data, as described in class:

saleID	item
1	beer
1	diapers
2	pretzels
2	diapers
2	beer
3	pretzels
3	soda

Which of the following association rules hold when we require *Support* ≥ 0.4 and *Confidence* ≥ 0.6? Circle the ones that hold.

beer $\rightarrow$ diapers
beer $\rightarrow$ pretzels
beer $\rightarrow$ soda
diapers $\rightarrow$ beer
diapers $\rightarrow$ pretzels
diapers $\rightarrow$ soda
pretzels $\rightarrow$ beer
pretzels $\rightarrow$ diapers
pretzels $\rightarrow$ soda
soda $\rightarrow$ beer
soda $\rightarrow$ diapers
soda $\rightarrow$ pretzels

10. **Data Streams** (10 points; 5 per part)

Consider an online selling-buying service implemented using a Data Stream Management System (DSMS). Specifically, consider the following two streams:

```
stream  ForSale(itemID, seller, price, quantity)
stream  Buy(itemID, buyer)
```

An element on the `ForSale` stream indicates new items for sale including their price and how many there are; `itemID` is a key for this stream. An element on the `Buy` stream indicates the purchase of one item, and there is no key. You may assume every `itemID` appears on the `ForSale` stream before it appears on the `Buy` stream, and that an item is not bought more times than its `quantity` allows.

(a) Consider the following query using the CQL language for continuous queries over streams and relations:

```
select Istream(I.itemID)
from ForSale I [Range 1 Hour], Buy B
where I.itemID = B.itemID
group by I.itemID
having count(*) = max(quantity)
```

State what this query produces. *Do so in one phrase or sentence, but still try to be as precise as possible.*

(b) Now consider the following query, also in the CQL language:

```
select seller, avg(price)
from ForSale [Partition By seller Rows 2]
group by seller
```

The following elements have arrived on the `ForSale` stream, in the following order:

```
(I1,Tom,14,1) (I2,Sue,4,1) (I3,Sue,6,1) (I4,Joe,7,1)
     (I5,Joe,12,1) (I6,Sue,8,1) (I7,Joe,14,1) (I8,Tom,20,1)
```

What is the current result of the query?

11. **Data Integration** (8 points)

Please answer the following questions based on Alon Halevy's CS145 lecture on Data Integration.

(a) (1 point) According to Prof. Halevy, how many databases are being used in an individual company, on average? (circle one) **6 15 49 93**

(b) (1 point) It has been suggested that about 50% of IT budgets in companies are spent on tasks related to data integration. According to Prof. Halevy, what is the other 50% spent on? Write your one-word answer in the box:

(c) (6 points; 2 per part) Consider a simple data integration scenario in which a table from one source:

```
Major(studentID, major) \\ studentID is a key
```

is being integrated with a table from another source:

```
GPA(studentID, GPA) \\ studentID is a key
```

Consider each of the following mediated schemas. For each schema, circle the "**G**" if the schema is amenable to the "global-as-view" approach to data integration, and circle the "**L**" if the schema is amenable to the "local-as-view" approach to data integration. In each case the correct answer may include zero, one, or both letters circled.

- **Schema 1:** A virtual table:
  `Stats(major, avgGPA)`
  containing the average GPA for students, grouped by major.
  Circle zero, one, or both: **G L**

- **Schema 2:** A virtual table:
  `Students(studentID, major, GPA)`
  containing all information about all students (i.e., the natural join of tables `Major` and `GPA`).
  Circle zero, one, or both: **G L**

- **Schema 3:** A virtual table:
  `TopCS(studentID)`
  containing IDs of students with a CS major and a GPA > 3.8.
  Circle zero, one, or both: **G L**