CS145 Final Examination

Spring 2004, 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	14	5	6	10	15	5	10	14	10	24	7	120
Points												

1. **UML** (14 points)

Consider the following relational schema describing professors, courses, and classes taught:

```
Prof(name Primary Key, dept, office, phone)
Course(cNum Primary Key, title, dept)
Teaches(name Primary Key, cNum, quarter, #students)
```

(a) (8 points) Draw a UML diagram from which this relational schema could have been produced. Do not draw a diagram with three independent classes—your diagram should be fully connected, and it should be as detailed as possible from the information you have.

(b) (3 points) Now suppose for relation `Teaches` we instead have:

```
Teaches(name, cNum, quarter, #students,
        Primary Key (name,cNum))
```

If your UML diagram changes as a result, state or show the change.

(problem continues on next page)

(c) (3 points) Continuing with the new version of `Teaches`, now suppose we have some additional information with our relational schema: There is a referential integrity constraint from `Prof.Name` to `Teaches.name`, and there is a referential integrity constraint from `Course.cNum` to `Teaches.cNum`. If your UML diagram changes as a result of this additional information, state or show the change.

2. **Constraints** (5 points)

Consider the following three schema declarations. You may assume that all attributes have the same type, which has been omitted for brevity.

```
Schema 1: Create Table R(A Primary Key, B)
          Create Table S(C, D References R(A))

Schema 2: Create Table R(A Primary Key, B)
          Create Table S(C, D Check(D In (Select A From R)))

Schema 3: Create Table R(A Primary Key, B)
          Create Table S(C, D)
          Create Assertion A Check(
            Not Exists (Select * From S
                        Where D Not In (Select A From R)))
```

Which of the following statements is true? (Please circle exactly one.)

- All three schemas are equivalent in terms of their behavior.
- Schemas 1 and 2 are equivalent but Schema 3 is different.
- Schemas 1 and 3 are equivalent but Schema 2 is different.
- Schemas 2 and 3 are equivalent but Schema 1 is different.
- None of the three schemas are equivalent.

3. **More Constraints** (6 points; 3 per part)

Consider a SQL relation $R(A, B)$. For the following two general assertions, write in the box provided the constraint that is enforced by the assertion. Please be specific about the actual attributes involved in the constraint. In both cases the correct answer corresponds to a single concept from the course and can be specified in a few words.

(a) General Assertion:

```
Create Assertion WarmUp AS
   (Not Exists (Select A From R
                Group By A
                Having Count(*) > 1))
```

Constraint:

(b) General Assertion:

```
Create Assertion Tricky AS
   (Not Exists ((Select * From R)
                Except
                (Select * From R Where A = A)))
```

Constraint:

4. **Triggers** (10 points; 5 per part)

Consider a simple one-attribute relation $R(A)$ containing two tuples, $R = \{(10), (11)\}$, and another one-attribute relation $S(B)$ containing three tuples, $S = \{(20), (21), (22)\}$.

(a) Suppose we create the following trigger:

```
Create Trigger T1
After Insert On R
Referencing New Table as NT
Update R Set A = A + (Select Count(*) From NT)
```

and then we execute the following SQL statement:

```
Insert into R (Select * From S)
```

What tuples are in relation R after the SQL insertion statement and any subsequent trigger activities?

(b) Suppose instead we create the following trigger:

```
Create Trigger T2
After Insert On R
Referencing New Table as NT
For Each Row
Update R Set A = A + (Select Count(*) From NT)
```

and then we execute the following SQL statement:

```
Insert into R (Select * From S)
```

What tuples are in relation R after the SQL insertion statement and any subsequent trigger activities?

5. **Transactions** (15 points; 3 per part)

Consider a simple relation `Student(ID,GPA)`, and the following transaction `T`:

```
T: (Q1) Select Avg(GPA) From Student
   <read-only activity>
   (Q2) Select Avg(GPA) From Student
   commit
```

(a) Suppose all other transactions in the system are declared as `Serializable` and `Read-Only`. What is the weakest isolation level needed for transaction `T` to ensure that queries `Q1` and `Q2` will always get the same result? Circle one:

```
Read Uncommitted
Read Committed
Repeatable Read
Serializable
```

(b) Suppose all other transactions in the system are declared as `Serializable`, and they only involve queries, updates, and deletions. What is the weakest isolation level needed for transaction `T` to ensure that queries `Q1` and `Q2` will always get the same result? Circle one:

```
Read Uncommitted
Read Committed
Repeatable Read
Serializable
```

(c) Suppose all other transactions in the system are declared as `Serializable`, and we know nothing else about them. What is the weakest isolation level needed for transaction `T` to ensure that queries `Q1` and `Q2` will always get the same result? Circle one:

```
Read Uncommitted
Read Committed
Repeatable Read
Serializable
```

(problem continues on next page)

(d) Now consider the following variation, where the two queries are in two different transactions:

```
T1: (Q1) Select Avg(GPA) From Student
    <read-only activity>
    commit

T2: (Q2) Select Avg(GPA) From Student
    <read-only activity>
    commit
```

Suppose both transactions `T1` and `T2` are declared as `Read Committed`. Consider in turn scenarios (a), (b), and (c) above for all other transactions in the system. For which of these scenarios, if any, are we guaranteed to always get the same result for `Q1` and `Q2`?

Circle all the scenarios that guarantee identical results: (a) (b) (c)

(e) Continuing with the two-transaction variant, now suppose both transactions `T1` and `T2` are declared as `Serializable`. For which scenarios, if any, are we guaranteed to always get the same result for `Q1` and `Q2`?

Circle all the scenarios that guarantee identical results: (a) (b) (c)

6. **Views and Authorization** (5 points)

Consider two tables `Student(ID,name)` and `Lives(ID,dorm)`, where `ID` is a key for each table. Consider the following pair of SQL statements. Assume "JW" is a valid user, and the statements are issued by a single user who is the owner of both tables `Student` and `Lives`.

```
Create View NoHome As
  Select * From Student
  Where ID Not In (Select ID From Lives)

Grant Delete On NoHome to JW With Grant Option
```

Is this pair of statements legal in SQL? Write and circle "Yes", or write "No" along with a brief explanation of why the statements are not legal.

7. **Object-Relational SQL** (10 points)

Consider the following SQL-99 object-relational schema describing students and assignments. Assume that you can compare dates using the normal operators ($=$, $<$, etc.).

```
Create Type StudInfo (name, major)
Create Type StudType (ID, info StudInfo)
Create Table Student Of Type StudType (Primary Key (ID))

Create Type AssignType (kind, number, due-date)
Create Table Assignment Of Type AssignType
             (Primary Key (kind, number))

Create Table Score(student Ref(StudType) Scope Student,
                   assignment Ref(AssignType) Scope Assignment,
                   date-in, score)
```

Write a SQL-99 query to find the IDs (no duplicates please) of all non-CS majors who earned a score > 5 on at least one assignment that was turned in on time and whose kind is "challenge". Remember that your solution will be graded on simplicity as well as correctness.

8. **Recursion** (14 points)

Consider a relation `Edge(N1 integer, N2 integer)`, where a tuple (n_1, n_2) in `Edge` indicates that there is an edge from node number n_1 to node number n_2 in a directed graph. Consider the following `With` statement in SQL-99.

```
With Recursive Mystery as
  ((Select * from Edge)
   Union
   (Select Mystery.N1, Edge.N2
    From Mystery, Edge
    Where Mystery.N2 = Edge.N1))
Select N1 From Mystery Where N1 = N2
```

(a) (8 points) In one sentence or less, state in English what is computed by the above with statement. The correct answer is simple and brief.

(b) (6 points) Consider the following similar `With` statement:

```
With Recursive Mystery as
  ((Select * from Edge)
   Union
   (Select M1.N1, M2.N2
    From Mystery M1, Mystery M2
    Where M1.N2 = M2.N1))
Select N1 From Mystery Where N1 = N2
```

(1) Is this `With` statement legal in SQL-99? Write and circle "Yes", or write "No" along with a brief explanation of why it is not legal.

(2) Ignoring legality, does this `With` statement compute exactly the same result as the original `With` statement on all databases?

Circle one: YES NO

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

Consider the following market-basket data represented in a two-attribute relation.

trans-ID	*item*
1	diapers
1	milk
2	beer
2	chips
2	diapers
2	eggs
3	beer
3	chips
4	beer
4	diapers
4	milk
5	beer
5	diapers

Specify all of the association rules that can be deduced from this data with *Support* > 0.3 and *Confidence* > 0.5. To limit your search, you need only consider association rules that have exactly one item on the left-hand side and one item on the right-hand side. Write your rules in the box next to the data.

(ungraded scratch space below)

10. **Views, OLAP, Cube, and Rollup** (24 points; 2 per answer)

Consider the following fact table in a star schema, reporting total sales by store:

```
Sales(state, city, store#, total)
```

The only key for this table is the three dimension attributes together: `(state,city,store#)`. Let `Sales` have the following contents:

state	*city*	*store#*	*total*
Ohio	Columbus	1	600
Ohio	Cincinnati	1	700
Indiana	Columbus	1	500
Indiana	Columbus	2	500
Indiana	Bloomington	2	900
Illinois	Bloomington	1	500
Illinois	Chicago	1	750
Illinois	Chicago	3	950

Consider the following five materialized views:

```
Create Materialized View V1 As
   Select state, city, store#, sum(total)
   From Sales
   Group By state, city, store#

Create Materialized View V2 As
   Select state, city, store#, sum(total)
   From Sales
   Group By state, city, store# With Cube

Create Materialized View V3 As
   Select state, city, store#, sum(total)
   From Sales
   Group By state, Cube(city, store#)

Create Materialized View V4 As
   Select state, city, store#, sum(total)
   From Sales
   Group By state, city, store# With Rollup

Create Materialized View V5 As
   Select state, city, store#, sum(total)
   From Sales
   Group By state, Rollup(city, store#)
```

(a) How many tuples are in each of the five views?

V1: [ ] V2: [ ] V3: [ ] V4: [ ] V5: [ ]

(b) Now suppose we want to answer the following query:

```
Select city, sum(total)
From Sales
Group By city
```

Suppose we wish to answer this query using one of the materialized views, instead of directly on the `Sales` table. For each view, state the minimum number of tuples in the view that must be looked at in order to compute the result. (Assume you will not "look at" any tuples that do not contribute directly to the result.) If the query result cannot be computed from the view, write the number 0.

V1: [ ] V2: [ ] V3: [ ] V4: [ ] V5: [ ]

(c) Now suppose views `V1` and `V3` are created as *virtual views* instead of materialized views. For each of the following two queries, state how many different physically-stored tuples must be looked at in order to compute the result. (Again assume you will not "look at" any tuples that do not contribute directly to the result.)

(1) `Select state, sum(total) From V1 Group By state`

How many tuples? [ ]

(2) `Select state, sum(total) From V3 Group By state`

How many tuples? [ ]

11. **Peer-to-Peer Systems** (7 points; 1 per part)

In class we learned about three different types of Peer-to-Peer architectures:

- Hybrid (H) – like Napster
- Pure (P) – like Gnutella
- Distributed Hash-Table Based (DHT) – like Overnet

For each of the following three scenarios, state which architecture is *best suited* for the scenario. Write "H", "P", or "DHT" in the box.

(a) Peers stay connected for long periods but are weak (low network bandwidth and/or low processing power).

(b) You are looking for a file that almost everybody has, you only need to find one peer with the file, and no centralization is allowed.

(c) You want to be guaranteed to find a file no matter where it exists in the network, and no centralization is allowed.

For each of the following four scenarios, state which architecture is *worst suited* for the scenario. Write "H", "P", or "DHT" in the box.

(d) Peers stay connected for short periods.

(e) You are searching for a file that only one peer in 10,000 has.

(f) You are worried about "single points of failure."

(g) The system has 1000 nodes, and only 10 keywords total are used to identify files.