CS145 Final Examination

Autumn 2009, Prof. Widom

- Please read all instructions (including these) carefully.
- There are 12 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. The blank areas and backs of the exam pages may be used for *ungraded* scratch work.
- *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: ____

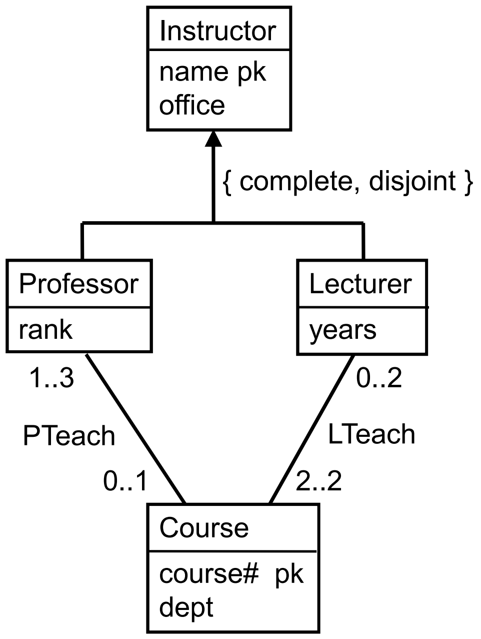
In accordance with both the letter and spirit of the Honor Code, I have neither given nor received assistance on this examination.

SIGNATURE: ____

Problem	1	2	3	4	5	6	7	8	9	10	11	12	TOTAL
Max. points	12	8	12	16	4	8	10	10	6	4	24	6	120
Points													

1. **UML** (12 points)

Consider the following UML diagram.



(a) *(2 points)* According to the diagram, what are the minimum and maximum total number of instructors for a given course?

Minimum: ☐ Maximum: ☐

(b) *(2 points)* According to the diagram, what is the minimum and maximum teaching load (number of courses) for lecturers? For professors?

Lecturer minimum: ☐ Lecturer maximum: ☐

Professor minimum: ☐ Professor maximum: ☐

(problem continues on next page)

(c) *(8 points)* Translate the UML diagram on the previous page to a relational schema.

- There are several possible automatic translations from the diagram to a relational schema. For full credit, use the translations recommended in class for the specific constructs and constraints present in this diagram.
- Underline a minimal key for each relation.
- Suppose that by default attribute values cannot contain NULL. If your translation requires any attributes to permit NULL values, circle them.

Write your relational schema in the box:

2. **Constraints** (8 points)

(a) *(4 points)* Consider the following three schema declarations. 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? Circle exactly one.

- All three schemas are equivalent in the database states and operations they permit.
- 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.

(b) *(4 points)* Consider a table `T(A int)`. Is it possible to write a SQL General Assertion involving only table `T`, such that the same assertion cannot be enforced by declaring one or more tuple-based constraints on `T`?

Either write "YES" and give the simplest example general assertion supporting your answer, or write "NO" and give a brief explanation of why such an assertion is not possible.

3. **Triggers** (12 points)

You are to specify two triggers using the MySQL-supported subset of the SQL standard. The triggers should be syntactically identical, except one of them (`Trigger-A`) uses the keyword `After`, while the other one (`Trigger-B`) uses the keyword `Before`. The trigger action should be a single SQL statement.

Your goal is to specify the simplest trigger-pair you can so that if a triggering database operation occurs and only `Trigger-A` is defined, the final database state may be different than if the same operation occurs on the same initial state and only `Trigger-B` is defined.

Assume that your database contains two tables, `T1(A int)` and `T2(B int)`. Use only these tables, and you are not required to use both of them unless they are both needed.

Fill in all of the blanks. *Remember to use only the MySQL-supported subset of SQL triggers, and give the simplest example you can.*

`Create Trigger Trigger-A` (must use `After`)

`Create Trigger Trigger-B`
assumed same as `Trigger-A` *except with* `Before` *in place of* `After`

Initial contents of table(s) – simplest possible:

(problem continues on next page)

Triggering modification command in SQL:

Final contents of table(s) if only `Trigger-A` is defined:

Final contents of table(s) if only `Trigger-B` is defined:

4. **Transactions** (16 points; 4 per part)

Consider a table `Item(name,price)`. Suppose initially there are two tuples in `Item`: `(iPod,20)` and `(iTouch,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 ('iPhone',40)
    S2: Update Item Set price = price+30 Where name='iPod'
    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`.

(a) If transaction `T2` executes with isolation level `Serializable`, what possible pairs of values `p1` and `p2` are returned by `T2`? Carefully indicate all possible pairs.

(b) If transaction `T2` executes with isolation level `Repeatable Read`, what possible pairs of values `p1` and `p2` are returned by `T2`? Carefully indicate all possible pairs.

(c) If transaction `T2` executes with isolation level `Read Committed`, what possible pairs of values `p1` and `p2` are returned by `T2`? Carefully indicate all possible pairs.

(d) If transaction `T2` executes with isolation level `Read Uncommitted`, what possible pairs of values `p1` and `p2` are returned by `T2`? Carefully indicate all possible pairs.

5. **Indexes** (4 points)

Consider the following table recording movie ratings:

```
Ratings(customer,movie,director,year,rating)
```

The only key is all attributes together. Suppose most queries on this table ask for a list of customers who gave certain ratings to a specific movie. For example, a typical query might look like:

```
Select customer
From Ratings
Where movie = ’New Moon’
And (rating = 5 Or rating = 4)
```

Suppose you are allowed to create exactly one index on table `Ratings` to speed up your queries. (You can assume the table is big enough that indexes never slow down query execution.) Write the `Create Index` command using SQL syntax.

6. **Authorization** (8 points)

   Consider a database with three users, `U1`, `U2`, `U3`, and no tables created yet. Write the shortest and simplest sequence of SQL commands you can come up with such that the last command is a `Revoke` command with the `Restrict` option, and it generates an error. Preface each command with the user issuing it, for example "`U1: Create Table ...`"

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

Consider a table `Edge(N1 int, N2 int)`, 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. Make no assumptions about the data in the table. 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) *(6 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) *(4 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 "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

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

Consider the following list of courses taken by students.

studentID	**course**
123	CS103
123	CS106A
123	CS106X
234	CS103
234	CS106A
234	CS106B
234	CS110
345	CS106A
345	CS106B
456	CS106B
456	CS110
567	CS103
567	CS106A

Treat the set of courses taken by each student as a "market basket." Specify all of the association rules that can be deduced from this data with *Support* > 0.5 and *Confidence* > 0.5. To limit your search, only consider association rules that have exactly one course on the left-hand side and one course on the right-hand side. Write your rules in the box next to the data.

9. **MapReduce** (6 points)

Consider a table `Scores(student,score)` where each student may have numerous scores. We want to run a *MapReduce* job to compute the result of the following SQL query:

```
Select student, Avg(score)
From Scores
Group By student
```

Fill in the blanks in the following pseudocode for the MapReduce job. We are not concerned about efficiency or about precise notation.

```
Map(key,value):
  (student,score) = parseScoresRow(value)
  emitIntermediate(                                    )

Reduce(key,values):
  [                                              ] (initialize variables)
  for each x in values:
      [                                          ]
  end for
  emit(                                          )
```

10. **Non-Traditional Data Stores** (4 points)

The non-traditional approaches to OLAP data management discussed by the guest speaker are understood to have slower query performance than traditional approaches. The speaker explained some advantages of these approaches that offset the disadvantages. Name two of the advantages:

1:

2:

11. **OLAP and Views** (24 points)

Consider the following fact table, reporting total enrollments for certain classes in certain years.

`Enrollments(year,dept,course,students) // key is [year,dept,course]`

Let `Enrollments` have the following contents (data is fictitious):

year	*dept*	*course*	*students*
2005	CS	110	60
2005	ME	100	70
2006	CS	103	50
2006	CS	110	50
2006	EE	110	90
2007	EE	112	50
2007	ME	100	75
2007	ME	112	95

Consider the following three materialized views:

```
Create Materialized View V1 As
  Select year, dept, course, Sum(students)
  From Enrollments
  Group By year, dept, course

Create Materialized View V2 As
  Select year, dept, course, Sum(students)
  From Enrollments
  Group By year, dept, course With Cube

Create Materialized View V3 As
  Select year, dept, course, Sum(students)
  From Enrollments
  Group By year, Rollup(dept, course)
```

(problem continues on next page)

(a) *(3 points each)* Given the data in table `Enrollments` on the previous page, how many tuples are in each of the three views?

V1: $\boxed{}$ V2: $\boxed{}$ V3: $\boxed{}$

(b) *(3 points each)* Now suppose we want to answer the following query:

```
Select dept, Sum(students)
From Enrollments
Group By dept
```

Suppose we wish to answer this query using one of the materialized views, instead of directly on the `Enrollments` 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: $\boxed{}$ V2: $\boxed{}$ V3: $\boxed{}$

(c) *(2 points each)* Now suppose views `V1`, `V2`, and `V3` are created as *virtual views* instead of materialized views. For each of the following 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.)

(i) `Select year, Sum(students) From V1 Group By year`

How many tuples? $\boxed{}$

(ii) `Select year, Sum(students) From V2 Group By year`

How many tuples? $\boxed{}$

(iii) `Select year, Sum(students) From V3 Group By year`

How many tuples? $\boxed{}$

12. **More Views** (6 points; 2 per answer)

Continue with the table from the previous problem:

```
Enrollments(year,dept,course,students) // key is [year,dept,course]
```

and now suppose we have a second table:

```
Courses(course,year,instructor) // key is [course,year]
```

Make no assumptions about the tables except for the specified keys.

Consider the three view definitions below. Using the restrictions imposed by MySQL (*not* the SQL standard restrictions), state whether a modification of the specified type for each view:

- ALWAYS generates a view-modification error;
- SOMETIMES (but not always) generates a view-modification error; or
- NEVER generates a view-modification error

(a) View:
```
Create View V1 As
    Select E.year, dept, E.course
    From Enrollments E, Courses C
    Where E.year = C.year And E.course = C.course
    And C.instructor = 'Widom'
```

Modification: `Update V1.year`

Generates view-modification error? Circle one: ALWAYS SOMETIMES NEVER

(b) View:
```
Create View V2 As
    Select year, dept, course
    From Enrollments
    Where students > 50
    With Check Option
```

Modification: `Insert Into V2`

Generates view-modification error? Circle one: ALWAYS SOMETIMES NEVER

(c) View:
```
Create View V3 As
    Select course, year
    From Courses C1
    Where instructor Not In
       (Select instructor from Courses C2
        Where C2.year < C1.year)
```

Modification: `Delete From V3`

Generates view-modification error? Circle one: ALWAYS SOMETIMES NEVER