SUMMER 2005 CSC 343 INTRODUCTION TO DATABASES

ASSIGNMENT 3

STUDENT 1

STUDENT NUMBER____

LAST NAME ________________________ FIRST NAME_________________________

STUDENT 1

STUDENT NUMBER____

LAST NAME ________________________ FIRST NAME_________________________

SECTION (circle only one): AFTERNOON EVENING

SUBMISSION DIRECTORY____

PART 1_______/45

PART 2_______/35

PART 3_______/70

TOTAL_______/150

For this assignment you can work in groups of up to two people. Each group must electronically submit one copy of the following files(with theses exact names):

- **group.txt**
- **populate .sqc**
- **query1.sqc**
- **query2.sqc**
- **query3.sqc**
- **part2.ddl**
- **part3 (a file in ASCII, ps or pdf format)**
- **Makefile**

To submit use the command

submit –c csc343h –a A3 group.txt, populate.sqc query1.sqc query2.sqc query3.sqc part2.ddl part3 Makefile

For more information on **submit** use **man submit.**

The printout submission (must be stapled) are to be placed (without envelopes) in the course dropbox in BA2220 and must contain:

1. the cover page provided
2. printouts of all the files submitted electronically

PART 1 Embedded SQL (45 points)

- This part of the assignment must be done on db2.

- The database schema is provided at : www3.cs.toronto.edu/~drosu/a3.ddl

- You are to write a number of programs using Embedded SQL in C. Your programs will modify and query the underlying database. Your programs will take input parameters and output the answer to STDOUT.

1.1 populate.sqc (10 points)

Write an Embedded SQL C program named populate.sqc which can be used to populate all the table in the database following the SQL conventions used for representing values of type integer, float, string, date, etc.

syntax:
populate <table_name> <value1> <value2> …

output:
"Tuple (<value1>, <value2>, …) was successfully inserted into <table_name>"

Example: inserting a tuple into table WorksOn specifying that the employee with employee number 211 works on the project with project number 300:

populate worksOn 211 300

1.2 query1.sqc (10 points)

Write and Embedded SQL C program named **query1.sqc** that will return for a given company the names of the employees who have changed their work place (department) inside the company more than once per year for the past 3 calendaristic years (the current year is included, even if incomplete). It is assumed that an employee is not working in the same time for two departments in the same company.

syntax: query1 <cID>

output:

eID1
eID2
…..

1.3 query2.sqc (10 points)

Write and Embedded SQL C program named **query2.sqc** that will return for a given company the department (departments if there are more) with the highest number of projects still under development that have a monthly budget greater or equal to the average monthly budgets of all the projects already completed in that department.

syntax: query2 <cID>

output:

dID1
dID2

1.4 query3.sqc (15 points)

Write and Embedded SQL C program named **query3.sqc** that will return for a given project all its direct and indirect subprojects. If there are no subprojects, "There are no subprojects" message should be returned.

syntax: query3 <prID>

output:

<prID> <prID1>
<prID> <prID2>
……

Note: a pair (project id, subproject id) should appear only once in the listing.

Important: Part of the marking will be automated. Use the correct syntax for parameters and output. If an invalid call is executed, the program should catch it and display an error message.

Part 2 Triggers (35)

- This part of the assignment must be done on db2.

- The database schema is provided at : www3.cs.toronto.edu/~drosu/a3.ddl

- You are to write a number of triggers and include them in part2.ddl

2.1 trigger1 (10 points)

Write a trigger named **trigger1** that will check **after** an update of the deptID on table worksFor weather an employee is listed as working in the same time for two different departments in the same company. If this is true, the update must be reversed back to the original value.

2.2 trigger2 (10 points)

Write a trigger named **trigger2** that will check **before** a delete is made on the table worksFor if that deletion will result in a project not having its manager listed as working for the project anymore. If that is the case, reject the delete operation and return the message "Delete operation rejected: the employee is the manager of the project".

2.3 trigger3 (15 points)
Assume that the budget of a project is the sum of the costs of its direct subprojects.

Write a trigger named **trigger3** that, **after** the update on table **project** of the **cost** of a base project (a project that does not have any subprojects), will propagate the change up to the immediate ancestors of the project whose cost was updated.

Part 3 Database Design (70 points)

Consider the following database schema: R(A, B, C, D, E, F, G) with the set of functional dependencies {A → E, AD → BE, AC → E, E → B, BG → F, BE → D, BDH → E, F → A, D → H, CD → A}

If X, Y are sets of attributes from R, such that R = X U Y, a join dependency is a constraint of the form $R = X \bowtie Y$. A database instance **r** over R satisfies this constraint if $r = \pi_X(r) \bowtie \pi_Y(r)$.

3.1 (10 points) Find the candidate keys of this schema.

3.2 (15 points) Find a 3NF decomposition of this schema (list both the relations and the corresponding set of functional dependencies). Show the details of your work.

3.3 (15 points) Find a BCNF decomposition of this schema (list both the relations and the corresponding set of functional dependencies). Show the details of your work. Is it dependency preserving? Explain why?

3.4 Let X, Y, S be subsets of attributes in R such that R = X U Y.

3.4.1 (15 points) Let X, Y, S be sets of attributes in R, and r be a instance over R that satisfies the nontrivial dependency $R = X \bowtie Y$ (neither X or Y is a subset of the other). **Prove that if X** n **Y is a subset of S, the projection of r on the attributes of S satisfies the join dependency S = (S** n **X)** $\bowtie$ **(S** n **Y).**

3.4.2 (15 points) Let X, Y, S be sets of attributes in R such that X n Y is not a subset of S. **Give an example of a database instance r over R that satisfies the nontrivial dependency R = X** $\bowtie$ **Y (neither S or Y is a subset of the other), but does not satisfy S = (S** n **X)** $\bowtie$ **(S n Y).**