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 (6 points)

Write an Embedded SQL C program named populate.sqc which can be used to populate tables **baseProject** and **subProject**.

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

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

Example: inserting a tuple into table subProject specifying that the project with id 30 is a direct subproject of the project with id 40

populate subProject 40 30

1.2 query2.sqc (8 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 each 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

1.3 query1.sqc (9 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 at least once every year for the past 3 calendar 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
…..

1.4 query3.sqc (22 points)

Write and Embedded SQL C program named **query3.sqc** that will return for a given project all its direct and indirect subprojects. A project prID1 is an indirect subproject of a project prID2 if there is a non-empty set of subprojects $prID_{i_1}$, $prID_{i_2}$, …, $prID_{i_{n-1}}$, $prID_{i_n}$ such that prID1 is a direct subproject of $prID_{i_1}$, $prID_{i_n}$ is a direct subproject of prID2, and for each $0<j<n$, $prID_{i_j}$ is a direct subproject of $prID_{i_{j+1}}$ If there are no subprojects, "Project <prID> has no subprojects" message should be returned, where prID is the id of the project given by the user.

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 endDate on table worksFor if the employee whose work endDate was updated becomes 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 cause the manager of a project to be deleted from the list of people working on that project. If that is the case, the delete operation must be rejected and the message “Delete operation rejected: the employee is the manager of the project” must be returned.

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, will propagate the change up to the immediate ancestors of the project whose cost was updated. A project prID1 is an immediate ancestor of project prID2 if prID2 is a direct descendent of prID1.

Part 3 Database Design (70 points)

3.1

Consider relation R(A, B, C, D, E, F, G) and 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}

3.1.1 (10 points) Find the candidate keys of this schema.

3.1.2 (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.

3.1.3 (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?

3.2

Let **S**(A1, A2,…, An) be a relation and X1, X2,.., Xn be subsets of {A1, A2, .., An} such that X1 U X2 U … U Xn = {A1, A2, .., An}. An instance **s** of **S** satisfies a join dependency constraint of the form S = X1 ⊳⊲ X2 ⊳⊲ … ⊳⊲ Xn if and only if $s = \pi_{X1}(s) \bowtie \pi_{X2}(s) \bowtie .. \bowtie \pi_{Xn}(s)$.

Multivalued dependencies are a special case of join depency: if {A1, A2, .., An} = X1 U X2 U X3, a instance **s** of **S** satisfies a multivalued dependency X1→→X2 if and only if $s = \pi_{X1X2}(s) \bowtie \pi_{X1X3}(s)$.

3.4.1 (15 points) Let X, Y, Z be sets of attributes in {A1, A2, .., An} such that X U Y = {A1, A2, .., An} and X n Y is a subset of Z. Let **s** be a instance over **S** that satisfies the nontrivial dependency $S = X \bowtie Y$ (neither X or Y is a subset of the other). **Prove that the projection of s on the attributes of Z satisfies the join dependency S = (Z** n **X) ⊳⊲ (Z** n **Y).**

3.4.2 (15 points) Let X, Y, Z be sets of attributes in {A1, A2, .., An} such that X U Y = {A1, A2, .., An} and X n Y is not a subset of Z. **For n = 6, give an example of a database instance s over S that satisfies the nontrivial dependency S = X ⊳⊲ Y (neither X or Y is a subset of the other), but does not satisfy S = (Z** n **X) ⊳⊲ (Z n Y).**