CS 480: Database Systems

Homework 4

Hard copy due on March 21st in class

Submit Electronic version to blackboard by March 21st 2:00 PM
No late submissions accepted

In this homework you have to write a program that interfaces with a database using embedded SQL. The program should be implemented by Java or C or C++. You can find the help on how to use Java and C/C++ to connect to database in the class web page on blackboard. Your program should operate on the following database with two relations:

employee(eid:integer, name:string, salary:integer)
worksfor(eid:integer, mid:integer)

The *employee* relation gives information about an employee. The *worksfor* relation gives immediate supervisor of an employee. That is, if (100,200) is a tuple in *worksfor* then this shows that the employee with *eid* 200 is the immediate boss of the employee with *eid* 100.

First, you program should create the tables *employee* and *worksfor*.
After setting up the tables, you program has to read an input file called *transfile*. Each line in *transfile* is a transaction. There are four types of transactions. The first character of each line gives the type of the transaction. We call this as transaction code. If the transaction code is 1, then you have to delete an existing employee. The other field in this line is the id of the employee (i.e., the *eid*) to be deleted. In this case, you need to delete the corresponding tuple from the *worksfor* relation also.

If the transaction code is 2, then you have to include a new employee. In this case, the other fields, in the line are *eid, name, salary* followed *by* zero or more *mids*. Each of them is separated by one or more spaces. Here *eid, salary and mids* are integers while *name* is a string without spaces. For this transaction, you have to insert a tuple in *employee* relation and insert zero or more tuples in *worksfor* relation. Note that there can be zero or more mids. This means the employee has zero managers or one or more managers. The number of tuples inserted in *worksfor* relation is as many as the number of distinct mids specified in the transaction.

If the transaction code is 3, then you have to output the average salary of all the employees.
If the transaction code is 4, then you have to output names of all employees that work

under a manager directly or indirectly. In this case, the manager id is given in the line. For example, if the manager id is 100, then you have to output names of employees that work directly for 100, as well as names of employees that work for these employees, and so on.

If the transaction code is 5, then you have to output the average salary of employees that work under a manger. If the average salary is not an integer, round it to an integer. Here also the manager id is given in the input line.

If the transaction code is 6, then you have to check if there is any employee who has more than one manager. In this case, you have to output the names of all such employees. If there is no such employee then output the string "no employees with more than one manager".

When you read a line from *transfile*, if the transaction code is 2, then you have to check that there are no duplicates in employee relation. If the tuples for the new employees are created successfully, then output "done". If there is any problem then output "error". If the transaction code is 1 then you output "error" if no tuple for the employee exists in the *employee* relation. If the employee is successfully deleted, then output "done". Note that , in this case, you have to delete all tuples in *worksfor* relation in which this emloyee id appears as *eid* or as *mid* (i.e., as manager).

When you read a line whose transaction code is 3,4,5 or 6 then you output the results of the transactions. If any problem occurs, output "error".
After you have read all the transactions from *transfile*, your program should drop all the three tables that were created. **Note: Don't forget to drop all tables you created in the end of your program!!!**

We assume the input file is always valid, and you don't need to check if it's correct. We also assume there are at most 100 employees in the database, and each employee's name is a string of maximum 20 characters without any spaces. Every employee's salary is an integer no more than 60000.

A sample input file is as follows:
*****transfile*****
2 100 Mary 50000 50
2 101 Ford 53000 100
2 50 Dell 55000 0
1 100
4 50
5 50
Test your program using your own *transfile*. Save your source code, a readme file and *transfile* into a directory named by your netid. Compress the directory into a rar or zip or tar file and submit it to the blackboard.