NYU Tandon School of Engineering **October 2, 2018**
Computer Science and Engineering
CS6083, Fall 2018

Problem Set #2 (due October 15)

Problem 1:

In this problem, you have to write SQL and RA queries for a database modeling student organizations (student clubs) at a university, given by the following relational schema:

Student (sid, sname, semail, sphone);
Club (cid, cname, cdescription);
Event (eid, ename, edescription, edate, memprice, nonmemprice, maxpeople)
Membership (sid, cid, semester, year, memberfee);
HoldsEvent (eid, cid);
Register(eid, sid, price, rating);

In the schema, we have students identified by an *sid*, and clubs identified by a *cid*. Students can join clubs on a per-semester basis by paying a membership fee. Clubs organize events, such as "Annual Dinner" or "Camping Trip", where each event is identified by an *eid*. An event can be jointly organized by several clubs, and the price of taking part in the event usually depends on whether the student is a member of one of the organizing clubs or not. Finally, the schema keeps track of which student registers for which event, and afterwards the student can rate an event from one to five stars.

(a) Draw an ER diagram that is consistent with the above schema. Identify any weak entities.

(b) Identify suitable foreign-key relationships for the above tables.

(c) How would you change the above schema, if every event is organized by only one club?

(d) How would you change the schema if for jointly organized events, the different clubs can charge different fees for their members? E.g., if an event is jointly organized by the chess club and the tennis club, the chess club might charge their members $5 to attend, while the tennis club charges their members $10.

(e) Create the above schema in a database system, choose appropriate attributes types, and define primary keys, foreign keys, and other constraints. Data for this schema will be made available on NYU Classes; please use that data. Load the data into the database using either insert statements or the bulk-loading facilities. You may use any mainstream relational database system.

(f) Write the following SQL queries and execute them in your database system. **Show the queries and the results**:

(i) For each student, output the number of events the student registered for in Fall 2018, and the total price of all the registrations.

(ii) Output the eid and name of the event that was co-organized by the largest number of clubs.
(iii) Output the sid of any student who was not member of the chess club in Fall 2017, but who would have saved money by becoming a member (because the membership fee would have been cheaper than the extra nonmember registration fees she paid).
(iv) Output the sid and cid for any case where a student has been a member of a club for every semester since Fall 2015. (Assume there are two semesters each year, Spring and Fall.)
(v) Output the name of any club that has never held an event with average rating below 4.
(vi) Output the names of all pairs of clubs that have organized at least 5 events together.
(vii) Output the name of any student who has attended an event jointly organized by the Chinese and Japanese Student Associations while not being a member of either one.

(g) Write expressions in Relational Algebra for queries (iii) to (vii).

(h) Write SQL statements to perform the following updates to the database:
(i) Write a query to give a complementary membership for Fall 2018 to any student who has attended more than 5 events as a nonmember during Fall 2018.
(ii) Delete all students who have never attended an event and who have never been a member of any club.
(iii) Suppose the chess club and the tennis club decide to jointly organize a new event called "Halloween Party" for up to 50 people. Write insertion statements to register this event in the database. (You may choose values for the date and price, and use the cids of the clubs directly in the query, instead of looking them up based on the club names.)

Problem 2:

In this problem, you have to create views and then write queries on the views, and create triggers, for the schema in Problem 1. Execute everything on your database system and report the results.

(a) Define a view that shows all students and their club memberships for Fall 2018. The view should consists of the attributes sid, sname, cid, and cname. Using this view, answer the following queries:
(i) Output the number of Fall 2018 club memberships for the student with name "John Myers".
(ii) Define another view on top of this view and the tables Register and HoldsEvent that contains the sid, sname, and number of events a student registered for during Fall 2018 while being a member of a club organizing the event.
(iii) Write any interesting query of your choice that uses this new view.
(iv) Drop both views.

(b) Can you write a trigger that automatically deletes students that have not attended any events for two years? (You should check your database system manual to see what kind of functionality it offers to implemented this or something similar.)

(c) Write a trigger that automatically increases the nonmember price of an event by 50% as soon as the number of registrations reaches 80% of the maximum (maxpeople). Note that this should only change the price in Event, not change the price of already existing nonmember registrations.

Problem 3:

In this problem, you need to design a relational database for a chain of printing and copying shops, that is, shops that offer services such as printing, scanning, copying, binding, shipping, etc. Each shop has a unique shop number, an address, and a phone number. Different shops may offer different services, and you need to store which services are offered at which shop. Each service has an ID, a name (e.g., black-and-white printing, color printing, copying, binding, scanning, shipping, etc.), and a short description. There is also a price per unit (say, 5c per black-and-white copy, 10c per page for color printing) that may be different in different shops. Each shop has a number of employees, with each employee having an employee ID, a name, and a shop to which they belong; you may assume that each employee works at only one shop.

Customers have to register and provide their name, phone number, and a credit card. Customers can then request work orders. Each work order consists of one or more jobs, where each job is about only one type of service. For example, a work order could consist of a printing job followed by binding of the prints, followed by shipping them to customers. For each job, you need to store the type of service and the amount (e.g., 5000 copies), and a short description provided by the customer. For each work order, you also need to store the order date, the requested and actual completion date, the shop that is in charge of the work order, and the total price. Finally, for each work order that is completed, we need to store which employees worked on it, where a work order could be done by one or more employees.

(a) Design an ER diagram that can model the above scenario. Identify suitable keys and the cardinalities of the relationships. Also identify any weak entities. Discuss any assumptions that you are making in your design.

(b) Convert your ER diagram into a relational schema. Show primary keys and foreign key constraints.

(c) Write statements in SQL for the following queries. Note that if your schema does not allow you to answer a query, you may have to go back and change your design.

(i) For customer “John Myers”, output the number of work orders and their total cost in 2017.
(ii) Output the shop number of any shop who offers every service offered by the chain.
(iii) For each employee, output the employee ID and the percentage of work orders on which this employee worked that were more than one day late (i.e., actual completion date was more than one day after requested completion date).
(iv) Output the cid and name of any customers who made a work order involving both “color copying” and “shipping”.

(d) Create the tables in your database system, and insert some sample data (maybe 5-10 tuples per table). Choose an interesting and meaningful data set. Then execute the queries from part (c). Submit your sample data, and screenshots or logs of the queries and outputs.