CS 386/586 Winter 2013 Assignment 6

Assigned: Wednesday, February 27, 2013

Due: Wednesday, March 13, 2013 at 11:59PM (midnight)

General Information:
Submission: you **must post a note** to the "**instructors**" with the **attachment** (listed below) in Piazza. You **must** put your note with the attachment in the **turn-in-assign-here6** folder.

You are encouraged to do this assignment in teams of two students. If you work with a team, just turn in one assignment – with the name of both students included.

Assignment:

Database Design Problem:

You are to design a database schema for the following application. Your design will be represented using an ER diagram with cardinality constraints, attributes on relationships (if needed), isa relationships (if needed), and keys.

The database is to keep track of courses, classes (sections of courses offered in particular quarters), quarters, rooms, time/day slots, students, instructors, enrollments, transcripts, and teaching assignments.

A **course** is identified uniquely by the subject prefix and the course number, e.g., CS 386 or CS 162. (We will ignore classes like CS510 that have many different titles.) Each course has a title, a description, an indication of the number of credits, and an indication of whether or not the course is a graduate or undergraduate course. (We will ignore other kinds of courses; we will ignore courses that are both graduate and undergraduate.) A course may have zero or more other courses (that are also in the database) as prerequisites. (You must represent the prerequisite relationship in your database design.

A class (or section) of a course must be associated with the course that it is a section of and the quarter that it is being taught in. A class will also be associated with the instructor or team of instructors that will teach the class. (The instructors are not always known when the class is first created.) A class will be associated with each student that enrolls in the class; the enrollment should also record the grade that the student earned in the class – with the course subject prefix and course number as well as the quarter that the class was taken.

A student is represented by a student id and a name. There are two kinds of students: graduate students and undergraduate students. To keep things simple, we assume that a student is

either an undergraduate student or a graduate student but not both. There are no other kinds of students.

An instructor is represented by an employee id number, a name, and a rank.

A room is represented by the building, the room number, and the capacity of the room in terms of number of students.

A timeslot is represented by the time/day combination. For example, our class meets in the timeslot from 17:30 – 21:05.

The following constraints also exist for this application. You may or may not be able to support the enforcement of each of these constraints in your ER diagram.

1. A room cannot be booked for more than one class (section) at the same time slot.
2. The number of students enrolled in a class cannot exceed the capacity of the room assigned to the class.
3. Students may not take more than 21 credits in any one quarter. (PSU rules may be more flexible.)
4. A student can't enroll in two sections at the same time slot in a quarter.
5. A student can't enroll in two sections of the same class in a quarter.
6. A student can't enroll in a class unless they have earned a grade of C or higher in the prerequisite courses.
7. A student can't take the same course more than three times.

Once you have designed your database by drawing an ER diagram, translate your ER diagram into a set of tables. Describe your tables using create table statements – with all primary key, foreign key, NOT NULL, and other constraints shown.

For any of the above constraints listed, explain how the constraint will be enforced. For example, is it fully supported by a foreign key constraint? etc? Or will there need to be a trigger or database procedure written to check/enforce the constraint? Or would you prefer to enforce the constraint in the embedded SQL program that allows this database to be modified. For each constraint, you need to describe – in English how you would have it enforced – by one or more of these methods.

Once you have your tables defined, write SQL queries for the following:
Q1: Write a query that will show the class roster for each class.
Q2: Write a query that will show the transcript for each student.

Please turn in: your ER diagram, the CREATE TABLE statement for all of your tables (that you use to represent all of the information from your ER diagram, a description of how all of the constraints are supported, and the two SQL queries.

Normalization problem

Consider the following table which will be introduced into the Spy database to replace the team and mission table.

Current tables (which will be removed from the database):

Team(team_id, name, meeting_frequency)
Mission(mission_id, name, access_id, team_id, mission_status)

New table which will replace those two:

Mission2(mission_id, mission_name, access_id, team_id, mission_status, team_name, team_meeting_frequency)

For this table:

1. Identify all of the non-trivial FDs that hold (based on the application).
2. Identify the key(s) for this table.
3. Identify any "troublesome" FDs that prevent this table from being in BCNF.
4. Describe one insert anomaly, one update anomaly, and one delete anomaly that can arise with this table.
5. Given that the system designers have decided to use this table and knowing that there are redundancies, describe (in English) the triggers that would need to be implemented in order to correctly manage the redundancy. You want to make sure that a given piece of information (if it is represented redundantly) is always consistent. That is, you want to make sure that all of the copies of any redundant information have the same, most up-to-date value. You want to make sure that if information is deleted, all of the copies are deleted, etc.

Note: if you were to normalize this table, you would end up with the Mission and Team tables of the original Spy database.

Turn in your answers to 1-5 above.