CS351 Introduction to Database Systems **Spring 2013**

Project 2

100pts *Due Date: Friday, Apr 11, 2014*

Designing New University Relations

This project focuses on normalization and application connectivity. We will add to the `university` database used in the previous project. In Project 1, you were given a simple university schema, which contained information on Students, Courses, Departments, and their relationships with each other. There is still quite a bit of information missing from our university database. I will summarize these as follows:

- **Faculty** members are identified by a unique ID. Each should have his/her own name, phone number, email address, salary, and a rank, which can be Assistant Professor, Associate Professor, and Professor.

  A faculty member can belong to one or more departments, although you should make the distinction that only one of these departments is their primary association. They can only have one office space, denoted by a building and room number, regardless how many departments with which they are affiliated. A faculty's research interests might be different for each department they are with.

  Faculty can teach several courses and must hold specific office hours for each. You can use the primary key to the **Course** relation from the previous project to denote the course they are teaching.

- Each faculty can also associate with a set of **Grants**. Each grant should have an award number, award amount, amount left, award title, and begin and end dates. Grants can be affiliated with multiple faculty members.

- The **Academic Coordinators** are also identified by a unique ID. Each should have his/her own name, phone number, email address, salary. They have an office space, denoted by a building and room number. They can belong to only one department and can advise many students.

- Email and phone numbers for faculty and coordinators should be assumed to be multi-valued.

ER and Reduction

Design the full ER diagram for the university database.

1. First, you should include the Student, Department, and Course entities (as well as their relationships) from the previous project into this diagram. Yes, that means you will have to perform a "reverse reduction," which is a good exercise.

2. Next, add the above entities and relationships into the diagram and reduce into a set of schema. In your project report, you should turn in both ER diagram and the schema diagram as figures.

3. Produce the SQLite DDL code to define these tables. Name this file `Proj2-ddl.sql`.

Java Integration

- Create a new Java project in eclipse, and name it `CS351_UnivDB`.
- go to `http://www.xerial.org/maven/repository/artifact/org/xerial/sqlite-jdbc/` and download the latest `sqlite-jdbc-*.*.*.jar`. Last checked, 3.7.2 is the most recent version, and it works as expected. Put it into your Eclipse project directory.
- Add the `jar` file to your build path. To do this, right-click on your project, select "Build Path," and "Add external archives." Then select the jar file.
- Download `UnivDB.java` from the CS351 course page into your Project directory.
- You should familiarize yourself with the sample code I've given you in this file to help you connect and run some simple queries.

Requirements

You will be writing a complete university administration program in Java. Your source files must be well-documented and commented. You should use good rationale for decisions on your object/class design.

- Open up SQLite, and `.read` your DDL and DML files in, then save your database as `UnivDB.db`.
- Once you have created the `UnivDB.db file`, create a new class called **UniversityFunctions** with a `main()` method that will continuously loop on the following options:
  - Enter `EXIT` to exit the program.
  - Enter `STU` for the **Student Menu**
  - Enter `FAC` for the **Faculty Menu**
  - or enter a customized SQL query. (You must only allow SELECT queries here. Raise an error if users try to use a DDL query, or an INSERT or UPDATE).
  - For each SQL query answered, **report the elapsed time** in milliseconds. You may want to look into the java method `System.nanoTime()` for this.
- Inside the **Student Menu**, it will continuously loop the following:
  - Enter `EXIT` to exit to the Main Menu.
  - Enter `ADD` followed by a student ID, name, major, class, and GPA to add a student record.
  - Enter `REMOVE` followed by a student ID to remove a student record.
  - Enter `FIND` followed by a student ID or name. It should pull up the student's personal information (from the Student relation).
  - Enter `ENROLL` followed by a student ID and course ID to enroll a student in a course. You must first check if the student is already taking this course.
  - Enter `WITHDRAW` followed by a student ID and course ID to withdraw a student in a course. You must return an error message if the student is not taking the course you are trying to withdraw from.
  - Enter `SCHEDULE` followed by a student ID or name to list all courses that the student is enrolled in, including each course's information.
- Within the **Faculty Menu**, it will continuously loop the following:

  - You should have the `EXIT`, `ADD`, `REMOVE`, and `FIND` functionalities similar to Students.
  - Enter `SCHEDULE` followed by a faculty ID or name to list all courses the person is currently teaching, including the courses' information.
  - Enter `GRANTS` followed by a faculty ID or name to list all grant information.
- As a bonus (5pts) you can also implement the Academic Coordinators menu.
- Your code must be well documented with useful comments. You will be graded on your commenting.

What You Need to Submit

- Create a `readme.txt` file that contains the full names of your team members. State whether or not you attempted the bonus.
- Zip up your files (there should be four): (1) Readme, (2) the ER and Schema diagrams in one PDF/Word file, (3) your SQLite DDL file, and (4) a zipped archive containing all your Java files. I do not need your `UnivDB.db` file, because I will be re-creating your database from item (3), and populating the database myself using your system (4).
- Submit your file to Angel http://lms.wsu.edu