CSCE-608 Database Systems

Fall 2019

Instructor: Dr. Jianer Chen
Office: HRBB 338C
Phone: 845-4259
Email: chen@cse.tamu.edu
Office Hours: TR 10:50am-12:20pm

Teaching Assistant: Yi Liu
Office: Engr. Activities BLDG B 100A
Phone: (203) 393-8835
Email: yl576@tamu.edu
Office Hours: MW 11:30am-1:00pm

COURSE PROJECT I

(Due October 29, 2019)

Overview.

This project provides you with an opportunity to have experience in building a database application for a real-world domain of your own interests. You will create a database using SQL, populate the database, write programs that manipulate the database, and develop an interface for user's queries and modifications on the database.

The project is an individual project. Each student should work on her/his own project.

The project will consist of five parts, described in details as follows.

Part A. E/R Diagram

Your first step is to identify a domain you would like to manage with your database, and to construct an Entity-Relationship diagram for the database. It is suggested that you pick an application that you will enjoy working with. It is especially nice if you pick an application where you can populate your database using real, as opposed to fabricated, data.

Try to pick an application with a schema that is relatively substantial, but not too enormous. For example, your E/R design should have in the range of five or so entity sets, and a similar number of relationship sets. You should certainly include different kinds of relationships (e.g., many-one, many-many) and different kinds of data (strings, integers, etc.), but your application need not necessarily require advanced features such as weak entity sets and "is-a" relationships.

1. Write a one-page description of the database application you propose to work with. Your description should be brief, relatively complete and clear. If there are any unique or particularly difficult aspects of your proposed application, please point them out.
2. Specify an E/R diagram for your proposed database. Underline key attributes for entity sets and include arrowheads indicating the multiplicity of relationship sets. If there are weak entity sets or "is-a" relationships, make sure to notate them appropriately.

The appendix provides some suggested topics of the project for your consideration. You may revise yours to fit the applications of your own interests. You may also consult and discuss with the instructor or TA on your proposed application.

Part B. Relational Schema

In this second part of the project, you will produce a relational schema from the entity-relationship diagram you came up with in Part A.

1. Using the method for translating an E/R diagram to relations, produce a set of relations for your database design.

2. For each relation in your schema, specify the nontrivial functional dependencies for the relation. Any functional dependencies that actually hold in the real-world scenario that you are modeling should be specified. It is fine if some of your relations have no nontrivial functional dependencies.
3. Check if each relation in your schema is in Boyce-Codd Normal Form (BCNF) with respect to the functional dependencies you specified. If not, decompose the relation into smaller relations so that each relation is in BCNF. No matter whether your relations from the E/R diagram are in BCNF or not, you should work through the relation normalization process, report and discuss the process and its results . Report how and why your relations are normalized if the normalization process produces new relations. If no new relations are produced, give descriptions on how your relations pass the BCNF test and provide explanations why your relations have no redundancy.
4. (Optional) Discuss whether the Third Normal Form or the Fourth Normal Form would further improve the structures of your relations. You may further improve your relation structures if you feel needed.

Part C. Creating Your Database using SQL

In this part of the project, you will create a database schema for your database application using SQL, and you will populate the relations in your database with data sets.

The CSE department provides accounts of PostgreSQL and MySQL database servers for each student, which can be activated through CSNET (`http://csnet.cse.tamu.edu`). It is recommended that you read this page `https://wiki.cse.tamu.edu/index.php/How_to_Use_MySQL`, which contains detailed information about how to activate your database account and how to access the database. You may also find other useful information on CSE helpdesk website (`http://helpdesk.cse.tamu.edu`). When connecting to the database, use your NetID as your user name. Also, you must use VPN when accessing CSNET and the database when your are off campus. VPN instructions are also available on CSE helpdesk website.

You may also download the Microsoft SQL Server Developer edition, which is free, from the Internet.

To manage a database, you may use client software such as Unix software: mysql or psql. They are available on the department servers `compute.cs.tamu.edu` and `linux2.cs.tamu.edu`. Graphical user-interface software might be helpful, such as MySQL Workbench, PHPMyAdmin, pgAdmin, or SQL Server Management Studio Express.

To create your database and populate it, you proceed as follows:

1. familiarize yourself with an SQL relational DBMS (referenced above),
2. create relations for your database based on your relational schema from Part B,
3. generate database data: write a program in any programming language you like that creates large files of records for each of your database relations.

   If you have available real data for your database, then your program will need to transform the data into files of records conforming to your database schema and to MySQL's load format. Otherwise, you will need to write a program to fabricate data: your program will generate records conforming to your schema. Note that it is both fine and expected for your data values, strings especially, to be meaningless gibberish. The point of generating large amounts of data is so that you can experiment with a database of realistic size. The data you generate and load should be on the order of:

   - At least two relations with thousands of tuples
   - At least one relation with hundreds of tuples

If your application naturally includes relations that are expected to be relatively small (e.g., schools within a university), then it is fine to use some small relations, but please ensure that you have relations of the sizes prescribed above as well.

When writing a program to fabricate data, there are two important points to keep in mind:

- Make sure not to generate duplicate values for key attributes.
- Your database almost certainly includes relations that are expected to join with each other. For example, you may have a STUDENT relation with an attribute COURSENUM that is expected to join with attribute NUMBER in relation COURSE. When generating data, be sure to generate values that actually do join – otherwise all of your interesting queries will have empty results! To order to ensure this, you may generate records for multiple relations (e.g., COURSE and STUDENT) at the same time, or generate the records for one relation first, and then use the joining values for the other relation. For example, you could generate records for relation COURSE first, then use the COURSE.NUMBER values when creating values for STUDENT.COURSENUM.

Part D. Developing a user interface

The application should have a clear purpose, and the interface must be user-friendly. Users do not need to know the details of the database or the SQL language to manipulate the data. Graphical interface is more desired than a text-based interface. Your program should consist of a continuous loop in which:

- A list of at least five alternative options is offered to the user. (An additional alternative should be QUIT.)
- The user selects an alternative.
- The system prompts the user for appropriate input values.
- The system accesses the database to perform the appropriate queries and/or modifications.
- Data or an appropriate acknowledgment is returned to the user.

Both input and output in the interface should be in a format more convenient and pleasing than raw interactive SQL. Please include some "interesting" queries or modifications, i.e., operations that require some of the more complex SQL constructs such as subqueries, aggregates, set operators, etc. As a general example, if your database is a campus applicant database, then your interface might include in its menu a number of useful queries on the database, with some queries performing statistical analysis requiring multiple levels of grouping, and other queries.

To build the user interface,

1. you can choose any combination of a programming language and a database system for your application,
2. set up or install a web server if you choose to write a web application. The department provides a web server that hosts web pages written in Perl, PHP and C/C++,
3. your software or web pages connect to the database using a connector. You will need to install a connector specific to the programming language of your application. CSE web server has the Perl & PHP connector installed for MySQL and PostgreSQL database. Some other connectors include:
   - Java JDBC connector for MySQL server
     `http://dev.mysql.com/downloads/connector/j/`
   - .NET Framework 4 and Microsoft SQL server (You can use languages VB, C#)
     `http://msdn.microsoft.com/en-us/library/kb9s9ks0.aspx`
4. There are also online examples of connecting to a database through a program:

- PHP - MySQL example and function documentation
  `http://dev.mysql.com/doc/apis-php/en/index.html`
- C# Documentation of connecting to Microsoft SQL server
  `http://msdn.microsoft.com/en-us/library/bb655891%28v=VS.90%29.aspx`

If you decide to build a website for this project, you may develop it in your local computer, and demonstrate to the TA. Alternatively, you may upload your wibpages to cloud service HEROKU and provide a link to the website in your project report. Check the documentation in `devecenter.heroku.com` for more details.

Part E. Submitting your project

The project submission should include a README of how to install and run your programs, and a project report that provides all details of the project. If you are submitting a web application, please provide the URL of your website.

The project report should be printed single-spaced, and include the following components:

(a) Project description of at least one page: it describes the application background of your system, and the functions and services your system will provide;
(b) The Entity-Relationship diagram of your database. The E-R diagram should contain at least 5 entity sets and a similar number of relationships. Give discussions on how the diagram is produced and how it reflects the applications you described in item (a);
(c) Table normalization: construct the relations based on the E-R diagram given in (b), and apply the table normalization process on the tables. Your tables should be at least in Boyce-Codd Normal Form. Discuss how the table normalization process changes/improves table structures. If the table normalization process does not change the table structures at all, explain why it does not;
(d) Data collection: describe how the data are collected/generated. Explain how you ensure the uniqueness of key attributes and interesting joins among multiple relations;
(e) User interface: describe how you build the system user interface and how users use your system. Give a list of functions that are offered by your system to the users. Explain how the functions are implemented in SQL;
(f) Project source code: include the source code that implements your system;
(g) Discussion: share your experience in developing a database application: what were the difficulties you encountered in the process and how you got over them; what you had learned from doing the project.

Grading policy (based on your submission in Part E)

(a) Application Description: 10%,
(b) E/R Diagram: 10%,
(c) Database Schema and Normalization: 15%,
(d) Creating Tables/Data: 20%,
(e) User Interface and Functions: 30%,
(f) Project Report: 15%,