Overview: In this assignment, you will build a database-driven web information management system from ground up. We will give you an application domain to work on, your goal is to design the underlying database and define the application functionalities you will provide with the database, and lastly implement this application as a web-based system.

Assignment: In this assignment you are to implement a three-tier client-server architecture.

1. **Database Back-End**, which runs the Oracle DBMS on aloes.cs.arizona.edu. Your job is to design the database relational schema, create tables and populate your tables with some initial data. We are requiring that you create an ER diagram, analyze the FDs of each table and apply table normalization techniques to your schema to justify that your schema satisfies 3NF, and if possible, BCNF.

2. **The business logic and data processing layer**, which is the middle tier that runs on an application server, which, in this assignment, will be lectura.cs.arizona.edu running the Tomcat web server. This layer sits in the middle, receives requests from client application and generates response back to client application. The response generation may involve accessing the back-end database you have created. Though there are many server-side techniques available for use, in this assignment we are requiring that you use Java and JavaServer Pages (JSP).

3. **Web Front-End**, which is the client user-interface. You need to design webpages appropriately to handle all the required functionalities. Your client application can run in any machine within the CS department with a web browser installed.

Application Domain: The problem description for the project is as follows:

A dentist’s office needs to keep information about patients, the number of visits they make to the office, work that must be performed, procedures performed during the visit, charges and payments for the treatment(s), and laboratory supplies and services.

Assume there is only one dentist, so there is no need to store information about the dentist in the database. There are several hundred patients. Patients make many visits, and the database should store information about the services performed during each visit, and the charges for each of the services. There is a standard list of charges for different services and each service or procedure needs several supplies. Make your own charge table and keep it in the database. The office uses three dental laboratories that provides supplies and services, such as fabricating dentures.

This description does not describe every detail. These are the essentials; we expect that your team will create logical and conceptual designs that incorporate these features, at minimum. You are free to add additional details that you feel are appropriate.

We realize that you are not dental professionals, and as such are not knowledgeable about all aspects of a dental practice. Neither are we! However, you’ve all visited a dentist several times (we hope!); use those experiences, and some Googling, to help you. As a start, we’ve provided a link to a charge sheet at the end of this handout. We’re far more interested in your DB design and implementation than in how a real dentist’s office operates.

(Continued...)

http://www.cs.arizona.edu/classes/cs460/fall16

Program #4: Database-driven Web Application

Due Date: December 6th, 2016, at the beginning of class

Designed by Zhenge Zhao
Required functionalities:

1. Record insertion
   Your application should support inserting a new data record via web interface.

2. Record deletion
   Your application should support deleting an existing data record via web interface.

3. Record update
   Your application should support updating an existing data record via web interface.

4. Record query
   Your application should support querying your database via the web interface for the problem description given above. You are required to implement at least five different queries, each of your own design, but with the following restrictions: At least one must be constructed using at least one piece of information gathered from the user. At least one must involve at least two relations. And, there should no trivial queries (for example, simply selecting everything from a table); your queries should be constructed to answer questions that real users would be expected to ask.

More specifically, you should support record insertion and deletion for all tables you designed, and support record update for no more than one table on your own choice. For each table you create, you need to populate a reasonable number of tuples in order to test your queries. (What is ‘reasonable’ is difficult to define; a few dozen tuples per relation certainly would be; just a handful per relation may not provide sufficient variety.)

Work in Groups: In industry, such a project is usually the work of multiple developers, since it involves several different components. Good communication is a vital key to the success of the project. This homework provides such an opportunity for teamwork. Therefore, it is required that groups of 2-4 members should be formed. Working solo (that is, a team of one) is not acceptable.

You need to come up with a reasonable work load distribution scheme. More importantly, you need to come up with a well-formed design at the beginning. This will minimize conflicts and debugging effort in the actual implementation.

Note: Each team should email the TA (Zhenge Zhao) at zhengezhao@email.arizona.edu the members of the team no later than Monday, November 21.

Hand In: You are required to submit a .tar file of your well-documented application program file(s) via turnin to the folder cs460p4. The tar file should contain the following exactly:

1. A directory called “ROOT”, which contains all the source code for the application. The structure of it should follow exactly as what you will see in the simple demo application (see below).

2. A directory called “doc”, containing a PDF document including these sections in this order:
   
   (a) Conceptual database design: ER diagram along with your design rationale and any necessary high-level text description of the data model (e.g., constraints or anything not able to show in the ER diagram but is necessary to help people understand your database design).
   
   (b) Logical database design: converting an ER schema into a relational database schema. Show the schemas of the tables resulted in this step.
   
   (c) Normalization analysis: show FDs of all your tables and justify why your design adheres to 3NF.
   
   (d) Query description: describe your five queries; what questions they are answering?

3. A ReadMe.txt describing how the TA can operate your website to see the required functionalities, and work load distribution among team members (that is, who is responsible for what?).

Each team should schedule a time slot (10 minutes) to meet with the TA (Zhenge Zhao) and demonstrate your system. We will let you know how to sign up later.

(Continued...)
A Simple Demo Application: To speed up the development, I’ve put a simple demo application under /home/cs460/fall16/2016p4/, please read the HowTo.txt within it to see how to run the demo. The demo contains a simple web page with a button you can click. By clicking the button, it will retrieve all the records from table mccann.employee and display the content on another web page.

You should run this demo because 1) it will let you install the Tomcat web server under your account which is needed for your application to run; 2) it will help you get familiar with the techniques you are going to use for this assignment.

Grading: Total: 100

1. Coding style: 10
2. Database design: 20
   - ER diagram: 10
   - Normalization analysis: 10
3. Implementation: 70
   - Record insertion: 15
   - Record deletion: 15
   - Record update: 15
   - Record query: 15
   - web front-end: 10

Note: We won’t put too much weight on the look of the pages. The main point of the assignment is the DB design, the web side is a nice bonus. You should put your focus on web page functionality. Don’t worry if your web pages don’t look “nice”.

Late days: Late days can be used on this assignment; how many a team has to use is determined as follows: Team members total their remaining late days, and divide by the number of members in the team (integer division), producing the number of late days the team has available, to a max of two days. (Justification: The TA needs to get grading done soon after the due date, and you need time to study for your final exam. As our final is at 1:00 p.m. Friday and the due date is 12:30 p.m. Tuesday, a late-day deadline of 12:30 Thursday (Reading Day) leaves ~24 hours for studying.)

For example, a team whose three members have 1, 1, and 3 late days remaining have \( \left\lfloor \frac{1+1+3}{3} \right\rfloor = 1 \) late day to use to get their project materials submitted.

Want to Learn More?
- Here’s a small dental charge sheet (w/o prices), to give you an idea what one contains: www.vhcf.org/wp-content/uploads/2011/05/D-4_Dental-Charge-Form-share.xls