DBMS Course: Web Based Database Administration Tool and Class Projects

Sub Ramakrishnan

Bowling Green State University
Department of Computer Science
Bowling Green, Ohio 43403
Phone: (419) 372 2337 Fax: 419 372 8061
Email: rama@cs.bgsu.edu

Emeka Nwosu

Bowling Green State University
Department of Computer Science
Bowling Green, Ohio 43403
Phone: (419) 372 2337 Fax: 419 372 8061
Email: iyke98@hotmail.com

Abstract

In this paper, we discuss a software tool we have developed for use in undergraduate DBMS courses, that provides: (i) a web-enabled database set up and administration facility for faculty use, and (ii) web-based database projects that may be assigned to students.

We discuss the motivation for our work and the objectives underlying the design of the tool. The significance of our work is two-fold. First, it should help the instructor in the set up, management, and monitoring of student database accounts. Second, it provides the students with a feel for the organization and use of contemporary web-enabled database applications.

Categories and Subject Descriptors

K.3 [Computers & Education]: Computer & Information Science Education - Computer Science Education.

General Terms

Design, Management, Experimentation.

Key Words

Database, Web-based techniques, Client-server communication, User interface.

1 Background

The web draws its audience by providing a standard interface to remote documents and databases. The academic community has recognized the power of web and is integrating web technology in a multitude of classroom setting [4, 11]. It acts as a powerful stimulant for the student and yet it provides an opportunity for the instructors to interact with students. For example, web is an easy to use interface for faculty to manage and organize classroom team projects, and for project members to share files [3].

The ACM curriculum 2001 committee [1] recommends that a minimum of 28 hours be devoted to the covering of two subject areas: (i) Information Management and (ii) Operating Systems. Worldwide revenues of DBMS software are in excess of $5 billion annually. Traditionally, many schools have also recognized the need and have offered courses on databases [5, 8] and operating systems [2]. We believe that the web enables us to assign interesting classroom projects that can relate to multiple CS areas such as operating systems and databases.

This paper describes a web based database application tool that serves a dual purpose. (i) It allows the instructor to administer and control database accounts for students use, and (ii) It provides a couple of sample web enabled database classroom projects that can be assigned to students in a DBMS course. The latter purpose is explored in a slightly different context in [5] where the authors present a tool that allows students to explore the design and functionality of the JDBC API. However, our approach is complementary; we provide specific ideas and code for projects to enable students get hands-on experience with web, JDBC and related technologies.

The objectives of our work in creating a management tool for the instructor and in providing classroom projects are the following: (1) provide an easy to use instructor interface and support audit control options for the instructor to gather statistics on student logins and database use. One might contend that database administration is the responsibility of technology services or ITS, and not of faculty; however, we have learned over the years that though it starts out as an ITS task, faculty wind up doing most of the work because of the various database options we explore in the course and class projects! (2) provide sample classroom projects that are flexible and span a range of topics covered in a DBMS course; the instructor may assign the sample projects exactly as laid out in this tool or assign any combination/subset of these projects. The complete solution for these projects can be obtained from the authors. (3) permit projects to be assigned as team projects; though they may involve ideas from other CS subjects the combined talents of the team members should be adequate to finish the projects. (4) integrate project experience with class presentations; allow students to see the underlying complexities of real world applications and help them gain insight into what they may encounter if their future studies or work is in this area.

2 The Database Course

We discuss the course, organization and other knowledge units to get an idea as to where the proposed course projects might fit in a typical DBMS course offering. DBMS is offered as a one-semester course in which both seniors and some graduate students enroll. The students complete the prerequisites for DBMS in this order: CS 101 - Introduction to Visual Basic, CS 205 and CS 215 - Programming Concepts I and II (coverage of C++, OO design principles, elementary and advanced data structure, ...), all accompanied by a number of hands-on programming projects on Sun Unix machines. A majority of our DBMS students have also had a 1 credit hour Unix course. The web is also the medium for class scheduling for our students, to get Unix accounts, assignments and class resources (CS 215 and beyond).

Though the primary language of choice for the database course is Java (see below), students adopt to it rather quickly, thanks to OOP in CS 205 and CS 215. In the DBMS course students study both the theory and application aspects of databases [3, 7-8]. We spend a major portion of the class time on design of relational data model and use of structured query languages, including functional dependency and normal forms, EER, interactive and embedded SQL, transaction processing and concurrency control. By the end of the seventh week the students are also given a quick overview of Java.

We deliberately avoid coverage of network and hierarchical models, and limit discussion of security, query optimization and physical implementation. This provides us with valuable class time to discuss web enabled database access and technologies: JDBC and ODBC, Java servlets and JSP. We also provide an overview of three-tier architectures and web-based database applications. Though some schools do not cover this topic in their (perhaps) one and only undergraduate DBMS course [11], we strongly feel this topic is timely and vital to students as they get ready to graduate and explore employment prospects.

In one project students modify the (instructor-supplied) code for a JDBC program to query an MS Access database (using JDBC-ODBC bridge). Then, students work on two other long projects that involve web enabled Oracle database access using Java, JSP and Java servlets as explained later in this paper. They have about 6 weeks to complete them.

3 Tool: Motivation for Choice of hardware and Software

All that the client (or tool user) needs is a web browser. The web server sits on a SUN UNIX machine. The web server interacts with the client (web browser) at one end, and with the Oracle database engine (either on the same as the web server or on a different host) at the backend. Programming environment is Java using Solaris JDK [9].

Though UNIX is a complex tool, its popularity cannot be over emphasized [2]. It is in widespread use in most universities and is extensively documented in the literature. Students in our database course are already familiar with UNIX as users, which constitute a strong motivational factor when they begin work on the course projects. Finally, our (and, it is our belief that most schools') web server runs on a UNIX platform.

The Oracle Academic Initiative [7] provides Oracle database software and other resources to the higher education community, for a very, very nominal cost. Oracle is Java friendly (!) and provides a free JDBC driver.

We use the Apache TomCat web server [10] in our implementation. Java servlets tie the web server to the backend database. Java servlets are attractive for a number of reasons: (i) the language is Java all the way around ensuring portability. (ii) multithreading enables fast interaction, (iii) servlets provide support for http type *get* and *post* interaction through pre-built classes (see program Skeleton I for an example), (iv) Java servlets are precompiled and are preferred to interpreted and less modular scripting languages such as Perl. Applications based on CGI/Perl are somewhat inefficient and scalability of CGI/Perl applications becomes a factor.

Our installation of Oracle and TomCat web server runs on two distinct Unix machines. However, either one or both may be installed on PCs and our tool works just fine with minimal modifications.

4 Tool Description

In this section, we describe the tool in more detail. As noted earlier, it consists of two parts, an easy to use database administration facility and projects for classroom use. The user interface is depicted in Figure 1.

Database Administration by Instructor

The URL provides the name and port # for the web server. The instructor types in authentication information including database host, Oracle listener port # and the database to connect to (Figure 1). Upon login, instructor functionalities are displayed (Figure 2**)**. It provides options for a number of useful tasks that can be administered by the instructor, including database creation, account set up and (automatic) deletion, controlling access rights, auditing student accounts, and other routine tasks.

As an example, the student accounts may be created by checking *Create Users*. The instructor can then specify the number of accounts, account expiration dates, access rights for these accounts and a number of other attributes, as noted in Figure 5. Upon submission of this form, our tool generates the corresponding student accounts and passwords, which may be handed out to students (response is not shown for brevity).

A particularly useful feature (Figure 2) is the audit feature of our interface (activated by *View User Log* button). It can give a summary of the times specific users have logged in to their accounts. There is also an option to look at all the table names, specific table definitions (metadata) and contents of tables and other objects that students have created (see *Display User Tables*). Together, these two options provide a quick check for the instructor to ascertain that students are doing work in their own accounts and adhere to ethical guidelines. Information such as this, and techniques for detecting plagiarism [6], can provide documentary evidence in academic honesty investigations. There are always one or two such cases in nearly every offering of our DBMS course.

Back to Figure 2, another administration task is resetting passwords. We find this interface to be much simpler than what we were used to in years past. Other tasks, while not explained for brevity can be inferred from Figure 2.

Student Web Projects

The tool provides three sample projects and a complete modular solution for each one. They may be assigned as they are (instructor provides some of the modules while students complete the remaining modules) or may be combined easily to provide a variety of other possibilities.

The first project is to design a web interface and associated SQL scripts to build and populate a table (Figures 3 and 4). The web interface specifies the table name, number of attributes, attribute type and name. The backend SQL scripts should be dynamically generated and executed, all in response to user's input. The instructor may choose to give a portion of the code and have the students develop the remaining parts. The project is also suitable as a group project - one student creates the GUI, second student creates interpreting user's choice (see Program Skeleton A: *doPost* method) and sends them to parameterized SQL procedures, while a third student constructs the SQL query and runs it (see Program Skeleton I: *createTable* method).

The second project allows exploration of metadata. The user interface specifies the table to be explored and the response is the metadata for that object. A second option is to display the content of that table. The third project is to run an arbitrary SQL query over the web and display the response. The query is entered over the web and may be a simple select or involving multiple joins.

It may be inferred that the database SQL commands are quite simple. Students need to understand how to construct an html page with a form (use doGet method, not shown for brevity) and how to extract the user interface (html) variables within the backend servlet programs (see Program Skeleton I: *doPost* method). Then, these variables are used in constructing a dynamic SQL statement. The statement is run and results are displayed back to the web user (for example, see the last line of the *try* block of Program Skeleton I). Note that this approach will be used for all of the three projects.

We have tried these projects in our database course offering. The first two were assigned as a group project and we approved the group membership in an attempt to ensure that the combined expertise of the group meets this skill list. There were minor glitches that were later resolved. Students were extremely positive about the experience. Many students felt that these are some of the few skills that will stick with them for years to come.

5 Concluding Remarks

The student projects can also be extended in a number of ways, and we discuss two extensions. (i) Database connection setup is expensive and SQL queries need not explicitly open and close the database connection. Instead, have the students maintain and administer a connection pool (JDBC provides support for this). (ii) Extend the servlets so it sends email upon completion of certain instructor defined project milestones. This also permits nice application of client-server concepts taught in other courses.

After we integrated these ideas in our DBMS offering, the number of students wanting to do web related database work either as an independent study or as a graduate project has increased significantly.

The instructor administration tasks we have provided are fairly comprehensive. However, one may add additional tasks as appropriate and or customize the tool.

References

[1] ACM. Computing Curricula 2001: Computer Science. *The Joint task force on Computing Curricula. IEEE Computer Society and ACM*. December 15, 2001. pp. 236.

[2] Berk, T, A Simple Student Environment for Lightweight Process Concurrent Programming under SunOS. *ACM 27th SIGCSE Proceedings*, February 1996, p. 165.

[3] Curtis, Ronald, A web based Configuration Control System for Team Projects. *ACM 28th SIGCSE Proceedings*. March 1997, pp. 189-193.

[4] Cunningham, S, Internet-Accessible Information Retrieval Tools for Advanced DB/IR Courses. *ACM 27th SIGCSE Proceedings*, February 1996, p. 107.

[5] Dietrich Suzanne, Urban, Susan and Kyriakides, Ion, JDBC Demonstration Courseware Using Servlets and JSP. *ACM 32nd SIGCSE Proceedings*, February 2001, pp. 266-269.

[6] Gitchell, David and Tran, Nicholas, *Sim: A Utility for Detecting Similarity in Computer Programs*. *ACM 30th SIGCSE Proceedings*, March 1999, pp. 266-270.

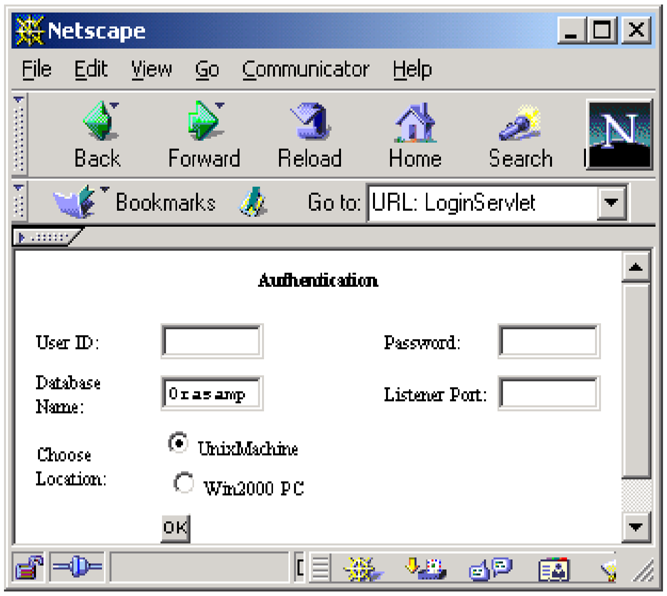
[7] Oracle Academic Initiative. http://oai.oracle.com/.

[8] Ramakrishnan, S, Rao, BM. Classroom Projects on Database Connectivity and the Web. *ACM 28th SIGCSE Proceedings*, February 1997, pp. 116-120.

[9] Sun Microsystems, http://www.sun.com

[10] TomCat Web server. http://jakarta.apache.org

[11] Urban, Susan and Dietrich Suzanne, Advanced Database Topics for Undergraduates: Experience with Teaching a Second Course. *ACM 32nd SIGCSE Proceedings*, February 2001, pp. 357-361.

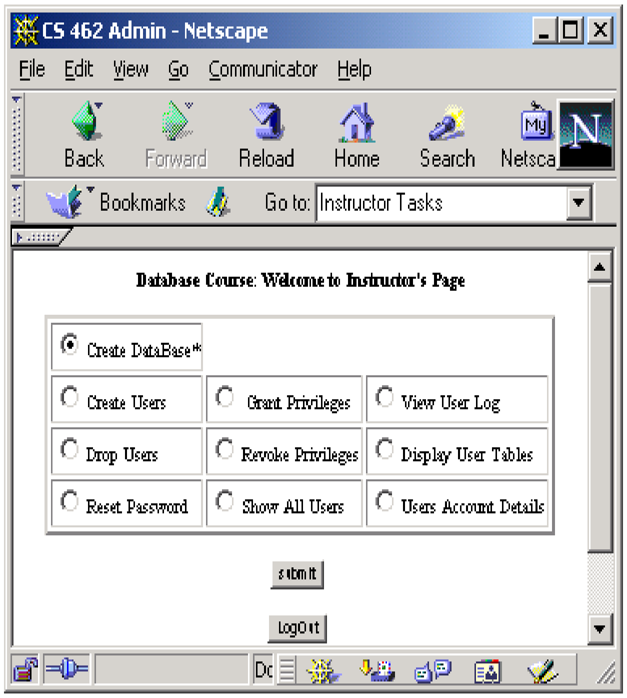

Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search
Bookmarks Go to: URL: LoginServlet

Authentication

User ID:
Password:
Database Name: Orasamp
Listener Port:
Choose Location: [✓ UnixMachine] Win2000 PC

OK

Figure 1: Authentication Interface


CS 462 Admin - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netsca
Bookmarks Go to: Instructor Tasks

Database Course: Welcome to Instructor's Page

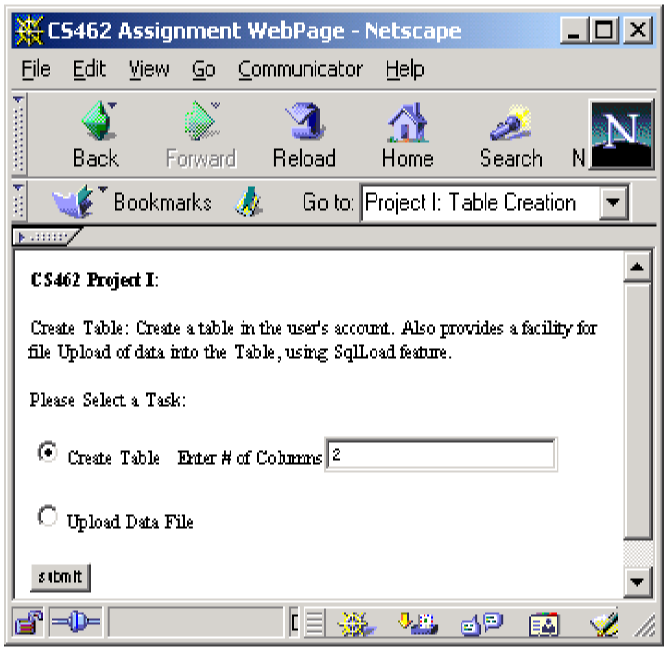
[✓ Create DataBase*]

Create Users	Grant Privileges	View User Log
Drop Users	Revoke Privileges	Display User Tables
Reset Password	Show All Users	Users Account Details

submit

LogOut

Figure 2: Instructor: Possible Administration Tasks


CS462 Assignment WebPage - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search N
Bookmarks Go to: Project I: Table Creation

CS462 Project I:

Create Table: Create a table in the user's account. Also provides a facility for file Upload of data into the Table, using SqlLoad feature.

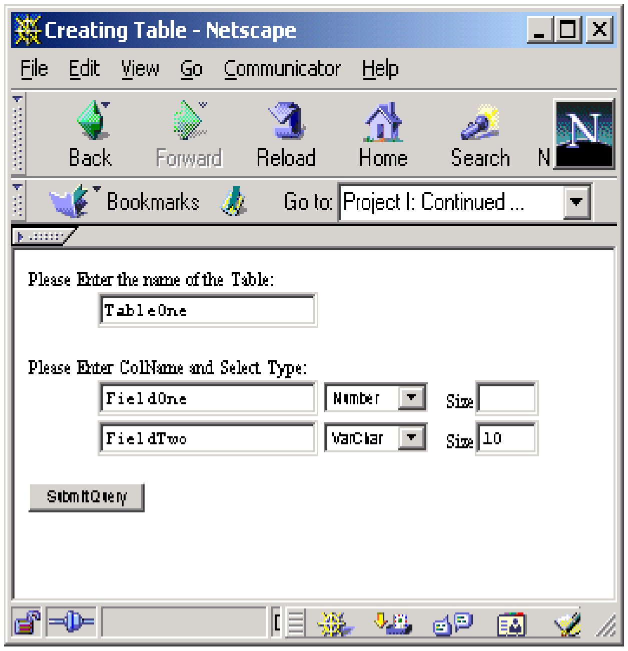
Please Select a Task:

[✓ Create Table] Enter # of Columns 2

Upload Data File

submit

Figure 3: Class Project I: Create Table and Upload Data


Creating Table - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search N
Bookmarks Go to: Project I: Continued ...

Please Enter the name of the Table:

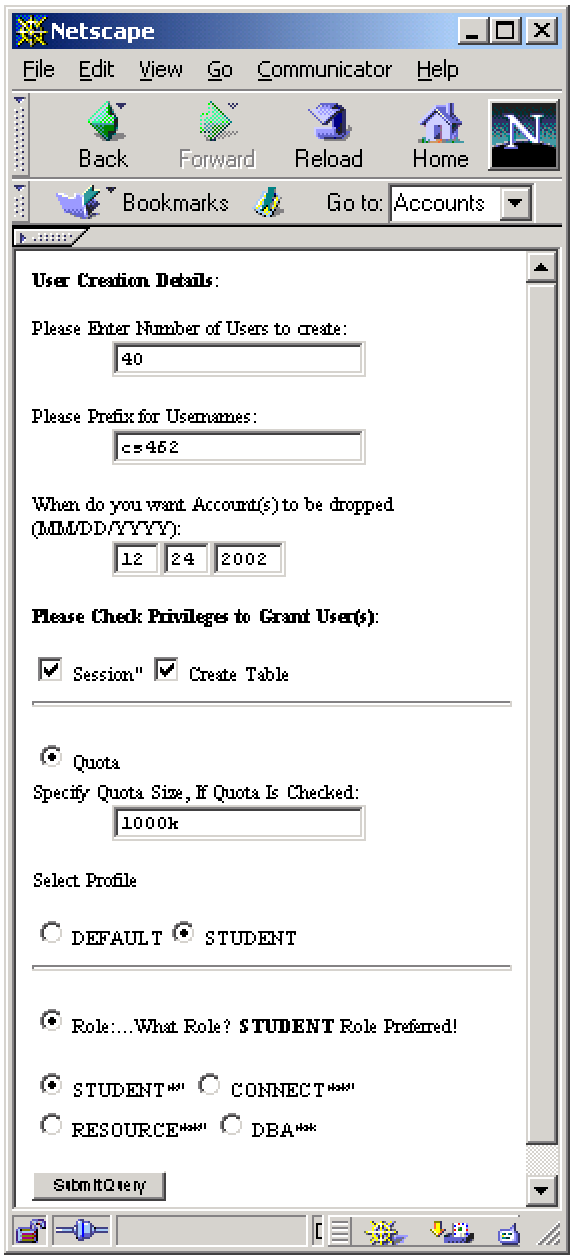
TableOne

Please Enter ColName and Select Type:

FieldOne	Number	Size
FieldTwo	VarChar	Size 10

Submit Query

Figure 4: Class Project I: Specify Table Properties



User Creation Details:

Please Enter Number of Users to create: 40

Please Prefix for Usernames: cs462

When do you want Account(s) to be dropped (MM/DD/YYYY): 12 24 2002

Please Check Privileges to Grant User(s):

[✓ Session"] [✓ Create Table]

[✓ Quota]

Specify Quota Size, If Quota Is Checked: 1000k

Select Profile

DEFAULT [✓ STUDENT]

[✓ Role:...What Role? **STUDENT** Role Preferred!]

[✓ STUDENT**] CONNECT***

RESOURCE*** DBA**

Submit Query

Figure 5: Instructor Creates Student Accounts

```
public void doPost (HttpServletRequest request,
      HttpServletResponse response) {
            String createtableName =
                  request.getParameter("createtableName");
            String colNum =
                  request.getParameter("colNum");
            String[] colNames =
                  request.getParameterValues("colName");
            String[] varType  =
                  request.getParameterValues("varType");
            String[] colType  =
                  request.getParameterValues("colType");
            …
            if(createtableName != null) &&  (colNames != null) )
                  createTable (out, createtableName,
                  colNames, varType, colType, s);
…
}  // end of doPost

public void createTable (PrintWriter out, String
      createTableName, String [] colNames, String [] varType,
      String [] colType, HttpSession s) {
   try{      …
          //Construct SQL query
            query = "create table " + createTableName + "( ";
            for (int x = 0;x < colNames.length ; x++ ) {
                query += colNames[x] + " " + varType[x];
              //If type is non-numeric extract
             //the size and use in Create Table
              if(varType[x].equals("VARCHAR") ||
                         varType[x].equals("CHAR"))
                                              …
               else      …
             } //end of for
             …
            myPrep.execute();    //Run Query - create table
            myConnectionRequest.close();
           myPrep.close(); ou.close();
            out.println("Table " + createTableName  + "
                       Created"); // Response to web user.
     }
    catch {…}
} // end of createTable
```

Program Skeleton I: Retreive Attribute Values from GUI and Construct and Run a SQL Query to Create a Table