Alternatives for Teaching Web Database Programming: JDBC, SQLJ or CGI

Ming Wang
E-mail: wangm@db.erau.edu
Department of Computing and Mathematics
Embry-Riddle Aeronautical University
600 S. Clyde Morris Blvd.
Daytona Beach, FL 32114

Abstract:
Web database programming was added to the author's undergraduate database course several years ago. From many approaches to accessing data in a database, JDBC, SQLJ and CGI were chosen for teaching in different semesters. The three approaches were compared and contrasted. Oracle programming examples of each of the three approaches are given. This paper also describes the template methodology that the author has been using in her teaching. A template is a sample database application program written with a host language and embedded SQL statements. By modifying the templates provided in class, students were able to do their homework and complete their database programming projects.

Key word: JDBC connectivity, Oracle application programming

1. Introduction

One of the prominent uses of the world-wide web has been web-based querying of databases. Search sites such as Alta Vista and Yahoo provide web users information by retrieving data from their databases. The web users fill out a form on the web page and the search engines return the results. Traditional database course in some institutions may now include web-based querying of databases as a topic.

There are a variety of approaches for a web page to access data in the database. Three of them were taught in the author's database course:

- JDBC (Java Database Connectivity) is made up of a set of Java interfaces that specify the API (Application Programming Interface), and several drivers supplied by database vendors that let Java programs connect to a database. The JDBC API, whose specification was defined by JavaSoft, is a standard data access interface developed by Sun. The JDBC API is a Java application programming interface which is not vendor specific, since JDBC can be obtained from different vendors

- SQLJ (Structured Query Language Java) is another JDBC-based approach that uses Java with embedded SQL. SQLJ is an emerging technology sponsored by Oracle, IBM, Sybase, Tandem and Javasoft for embedding precompiled static SQL statements in Java programs. A SQLJ translator transforms the SQLJ code into standard Java code that accesses the database through a call-level interface.

- CGI (Common Gateway Interface) is a standard interface used to write applications that remote users can fill out an HTML Form on Web browser to communicate with programs on an HTTP server.

Facing up with a variety of choices, which approach is appropriate to teach in a database class? The focus of this paper is to compare these approaches and summarize my experience teaching them.

2. JDBC

The most prominent and mature approach for accessing relational DBMSs from Java appears to be JDBC [1]. With JDBC, Java can be used as the host language for writing database applications. JDBC is a standard Java classes library that queries and modifies data in a database. The JDBC driver acts as a translator. The driver receives the client applications request in Java methods, translates it into a format that the database can understand, then presents the request to the database using the database native protocol. The response is received by the JDBC driver, translated back into Java data format, and presented to the client application. JDBC defines a set of interfaces and classes to be used for communicating with a database. This set of interfaces and classes are all contained in the java.sql package. The entire java.sql package is included in the Java core just as the java.awt or java.lang packages are.

All databases speak SQL. The goal of JDBC API is to give the Java programmer the ability to write applets and applications to access any SQL database servers. The JDBC API consists of a set of classes and interfaces written in the Java programming language that provides a standard API for database developers. JDBC API defines the common ground between the database and application. For example, it defines what commands can be executed, how to execute them, and how data will be formatted. The following JDBC file shows how to list all the employee names from the EMP table.

```
import java.sql.*;
class Employee
{
 public static void main (String args [ ] throws SQLException
 {
   // Load the Oracle JDBC driver
  DriverManager.registerDriver(new
       oracle.jdbc.driver.OracleDriver());

   // Connect to the database
   Connection conn = DriverManager.getConnection
        ("jdbc:oracle:thin:@minna:port:databasename",
         "username", "password");

   // Create a Statement
   Statement stmt = conn.createStatement ();

   // Select the ENAME column from the EMP table
   ResultSet rset = stmt.executeQuery
        ("select ENAME from EMP");

   // Iterate through the result and print the employee names
   while (rset.next ())
       System.out.println (rset.getString (1));

   //Close the statement and connection
   stmt.close();
   conn.close();
 }
}
```

Seven basic steps to query databases with JDBC are:
1. Import the java.sql package
2. Load and register the Driver
3. Establish connection
4. Create a statement
5. Execute a statement
6. Retrieve the results
7. Close the statement and connection

3. SQLJ

SQLJ is a way to embed static SQL in a Java program. The goal of SQLJ is to cut down the development and maintenance cost of Java programs. It is required to connect to a database, provides compile-time checking of SQL statements and reduces the amount of code needed to execute SQL from within Java.

SQLJ has two major components: The SQLJ translator and SQLJ runtime library. The translator produces a .java file and invokes the Java compiler to create a .class file. The SQL runtime library is invoked automatically at runtime. It implements SQL operations. SQLJ code must be precompiled and translated to Java, using the SQLJ translator. When a SQLJ program is compiled, the SQLJ pre-compiler steps in first, to check the SQL code at compile time and translate the SQLJ statements into pure Java code. This generated Java code needs to be further compiled to byte code. The byte code makes use of JDBC methods, to perform the required database operations, using JDBC as necessary to implement the database operations (See Figure 1 below).

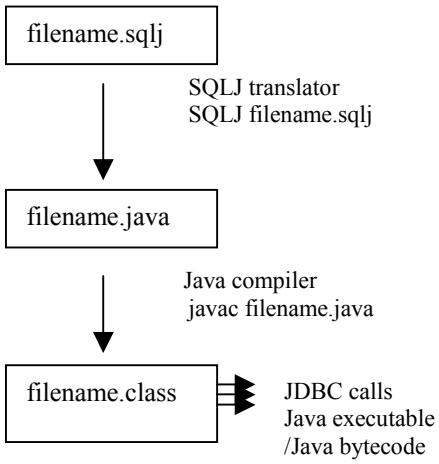

Figure 1: SQLJ Life Cycle

The steps to write a SQLJ file are:
1. Load the JDBC driver,
2. Specify a connection context, 3)
3. Specify host variables in SQLJ
4. Execute the SQL statement,
5. Process result.

The following file is called SimpleExample.sqlj, which retrieves a person's salary:

```
// First, import the SQLJ classes needed:
import sqlj.runtime.*;
import sqlj.runtime.ref.*;
import java.sql.*;

 public class SimpleExample {
  public SimpleExample() {
    try {

  // Register the driver and set the default context
  DriverManager.registerDriver(new
       oracle.jdbc.driver.OracleDriver());
   DefaultContext.setDefaultContext(new DefaultContext(
      "jdbc:oracle:thin:@lt&;myOracleHostgt&;:
       port:databasename", "username","password"));
   } catch (Exception ex) {
    System.err.println("Error running the example: " + ex);
   }
 }

 public static void main (String [] args) throws SQLException {
    SimpleExample o1 = new SimpleExample();
    try {
      o1.runExample();
    }
   catch (SQLException ex) {
      System.err.println("Error running the example: " + ex);
    }
 }
```

```
    void runExample() throws SQLException {
    System.out.println( "Running the example--" );

    // Declare two Java host variables--
    Float salary;
    String ename;

    // Execute a query to retrieve the values.
    // SQLJ uses the standard embedded SQL select ... into syntax-
-
    #sql { SELECT Ename, Sal INTO :ename, :salary FROM Emp
           WHERE Empno = 7499 };
    // Print the results--
    System.out.println("Name is " + ename + ", and Salary is " +
          salary);
  }
}
```

4. Common Gateway Interface (CGI)

CGI is a standard for interfacing applications with a web server. CGI lets HTTP clients interact with programs across the Internet through a Web server [2]. CGI lets HTTP clients receive information from the server and create a child process that handles or stores this information for the CGI program. CGI enables you to add applications to your web pages, retrieve real time data, and communicate with other individual. With CGI, you can provide a method by which visitors to your site can access your databases, store information, and execute external programs. In CGI programming, a request is made for a URL from a location served by the HTTP server.

When using HTML for the interface, the author can define various fields for the user to enter data. This data is sent to a CGI program when the user clicks a submit button. Once the data sent, the CGI program generates a new HTML page describing the result and returns it to the user's browser to be displayed. HTML forms allow you to create a set of data input elements associated with a particular URL. The user can interactively supply additional values by typing into a text field, select a radio button, and so forth. When user clicks the submit button, the active input data are accumulated into a string of the form "name1=val1&name2=Valu2 &…&nameN = valN, which is then transmitted to the specified URL. Action is an HTML element that specifies the URL of the CGI program that will process the FORM data.

You can use any language to write CGI scripts. Perl (Practical Extraction and Report Language) is the most popular language for creating a CGI program. HTML forms provide the only method for a visitor to interact with your CGI scripts. The Action attribute tells the server which CGI script is used to process the form. A method type specifies which method you are using to communicating with the server and script.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

EXEC SQL INCLUDE sqlca;

void sqlerror(void);

int main(void)
{

    EXEC SQL BEGIN DECLARE SECTION;
      varchar username[40];
      varchar password[40];
      char supnum_local[60];
    EXEC SQL END DECLARE SECTION;

    EXEC SQL WHENEVER SQLERROR DO sqlerror();
    EXEC SQL WHENEVER NOT FOUND GOTO end;

    /* Connect to the SQL database */
    strcpy((char*)username.arr,"kelly");
    username.len=strlen((char*)username.arr);
    strcpy((char*)password.arr,"kelly");
    password.len=strlen((char*)password.arr);

  EXEC SQL CONNECT :username IDENTIFIED BY
     :password;

  EXEC SQL DECLARE supnum_query CURSOR FOR
    SELECT supnum FROM supplier;
  EXEC SQL OPEN supnum_query;

  for ( ;;)
  {
     EXEC SQL FETCH supnum_query INTO :supnum_local;
     printf("<supnum = >%s\n", supnum_local);
  }

end:

    EXEC SQL CLOSE supnum_query;
    EXEC SQL COMMIT WORK RELEASE;
    return(0);
}

void sqlerror(void)
{
       EXEC SQL WHENEVER SQLERROR CONTINUE;
       printf("\n Oracle Error Detcted \n");
       printf("\n %70s", sqlca.sqlerrm.sqlerrmc);
       exit(1);
}
```

5. Comparison of the three approaches

5.1 SQLJ Compared with JDBC

SQLJ is based on static embedded SQL while JDBC is based on both static and dynamic embedded SQL. Dynamic SQL statements are not directly embedded in the source program. Instead, they are stored in character strings input into, or built by, the program at runtime.

SQLJ provides syntax checking and schema checking, which may help produce more reliable application programs. A server connection is used to check your embedded SQL code to verify that it is syntactically and semantically correct.

SQLJ offers a simple and straightforward way to write Java programs that interact with databases. SQLJ source programs are more concise and smaller than equivalent JDBC programs.

SQLJ provides stronger typing than JDBC code, because the SQL code is checked at compile time rather than run time. SQLJ provides strong typing of query outputs and returns parameters, and allows type checking on calls. JDBC passes values to and from SQL without compiling-time checks. Therefore, SQLJ provides less error-prone static SQL constructs than JDBC.

In using dynamic SQL APIs such as JDBC, you will not find out if your SQL is syntactically or semantically wrong until you run it.

5.2 CGI Compared with JDBC/SQLJ

The main advantage of CGI is its simplicity, language independence, Web server independence, and its wide acceptance. Despite these advantages, there are problems associated with the CGI-based approach.

The first problem is that the communication between a client and server must always go through the web server in the middle, which may possibly cause a bottleneck if there are a large number of users accessing the web server simultaneously. For every request submitted by a web server or every response delivered by the database server, the web server has to convert data from or to an HTML document. This certainly adds a significant overhead to query processing.

The second problem is the lack of efficiency and transaction support in a CGI-based approach. For every query submitted through CGI, the database server has to perform the same logon and logout procedure, even for subsequent queries submitted by the same user.

Another important disadvantage is that the server has to generate a new process or thread for each CGI script. for a popular site that can easily acquire dozens of hits almost simultaneously, this can be significant overhead, with processes competing for memory, disk and processor time

Finally, if appropriate measures are not taken, security can be serious drawback with CGI. Many of these problems relate to the data that is input by the user at the browser end, which the developer of the CGI script did not anticipate [3].

Overall, the author made comparisons of the three approaches as follows:

A comparison of the three approaches from the Web user's point of view is listed as follows:

	CGI	JDBC	SQLJ
Process on The server	X		
Process on The client		X	X
Need the Java Enabled web browser		X	X

Table 1 Web user' point of view

A comparison of the three approaches from the programmer's point of view is listed as follows:

	CGI	JDBC	SQLJ
Language independent	X		
Web server independent	X	X	X
SQL debugging			X
Strong type			X
Error message at run time		X	
Easiness to create interface	X		

Table 2 Programmer's point of view

6. Database Environment

All the three approaches were implemented within the Oracle DBMS environment.

The CGI Program was written with Pro-c and compiled with a Pro-c-compiler in Oracle7 on a Salaries system. The generated c code was compiled with a GCC compiler to generate executable CGI code. The executable cgi code runs on the web server when URL in the Action in the form is called. Two disadvantages are:

The HTML form has to be defined and wrapped in the proc code, which makes code more complicated. The size of the executable is very large.

One executable cgi program with one query can be as large as 1 megabyte because the calls to Oracle library are embedded in the file.

JDBC and SQLJ were run in the Oracle 8 on NT 4.0. Defining the interface is more complicated; Java beans have to be used.

Oracle provides two types of JDBC driver: “JDBC thin driver” and “JDBC” OCI (Oracle call interface) driver. The Oracle JDBC thin driver is for Java applets and applications. The Oracle JDBC thin driver is written in 100% pure Java and therefore platform independent. The thin driver is self-contained, requiring no Oracle specific software or files on the client side.

Oracle provides native support for Java in the DBMS. That is, Oracle has developed its own Java VM that integrates closely with the database structure for high performance. In addition, the database system natively supports JDBC and SQLJ. The oracle 8I also comes with a web server; so the database system can be an http listener.

The Java program is machine independent. It can be compiled with any Java compiler as long as it follows JDK standard. The driver can be changed. The compiled output Java class saved on the disk can be called by an applet which is embedded into a page of HTML. When a web browser reads the page with the embedded applet, it downloads the applet over the network to the local system, and runs the applet in a Java VM which is built into

the browser. Therefore the output of the Java program is displayed on the Internet.

7. Teaching Methodology

Most Computer Science undergraduate curricula offer at least one semester of course work in database management systems. How do you integrate database application programming into the already crowded traditional database course and lead students to programming for web applications? Although there are many ways to teach web database applications, they always involve introducing a new host language, Internet connectivity, database connectivity, embedded SQL, GUI interface or the HTML language. It is practically impossible to teach so many things within two weeks before the end of the semester with even one of the three approaches.

By trial and error, the author found a workable way to teach this emerging technology effectively. The solution was to provide students programming samples, explain the samples in class, let the students compile and run the samples by themselves, and then modify them in their homework assignments. It was successfully used for four semesters in the author's course. Given programming templates and the appropriate database connectivity resources, students were able to learn both non-web and web database programming in two weeks and accomplish their final projects. The prerequisite is that the student should be familiar with database design, with the SQL language, and with at least one programming language proficiency.

Templates are typical database programming examples prepared by the instructor. Every template contains database connectivity and embedded SQL statements. Typical templates covered the following topics:

- Query: Select
- DML: Delete, Insert and Update
- DDM: Create and Insert
- PLSQL: Create and Call subprograms

Teaching with templates includes the following steps:

1. Demonstrate non-web application programming examples in class and explain how they work.
2. Explain the syntax of database connectivity and embedded SQL.
3. Let students run the templates before they start their homework assignments.
4. Encourage students to do the homework by following the examples given in the classroom as templates.
5. Repeat Step 1 to Step 4 to teach web application programming by providing web application program templates.

8. Conclusion

CGI is a method that lets Web pages communicate with programs on an HTTP server. Prior to the advent of ODBC/JDBC, CGI was the only way to create web applications. Java replaces some uses of CGI by allowing truly dynamic web pages doing the process on the client instead of server. SQLJ was recently created to reduce programming code and increase the reliability of the program.

SQLJ is a good place to start out teaching embedded SQL statements since it is the easiest to learn and debug. JDBC is the most powerful approach to do web database programming since it can implement both static and dynamic embedded SQL statements. Although CGI programming has a lot of disadvantages, it is still a popular and general way to create web applications. Once students learn how to create a CGI program using one language, they will be able to pick up any other language to write another CGI program. Some of my students learned using pro-c to write a CGI program in the course. They picked up Perl or C/C++ to write a CGI program by themselves later.

What approaches for web database programming to teach in a database course depends upon how much time the instructor plans to spend on the topic in the course. It might take the whole semester to teach all three approaches. This may be a case for a second database course, a database application course, or a Web application course. Only one approach can be possibly taught if an instructor plans to teach the topic in a first database course due to the time frame provided.

The reasons to teach Java/JDBC for Web database applications are as follows:

1. Java is object-oriented and interpreted in the native machine's instruction set at run-time. Thus it is architecture-neutral and portable.
2. Java has built-in capabilities to prevent memory corruption, program crashes, and viruses. Thus it is robust and secure.
3. Java contains multiple threads that carry out many tasks in parallel.
4. Java was a language created for web application. Therefore, Java has many tools and predefined libraries available for Web applications.

The best experience that the author had from teaching web database programming was to give complete programming examples in class, provide soft code on the web site, let students compile and run by themselves, and then give them homework assignments. Most of the students found it easy to follow because they were guided by the examples. Based on their homework assignments they were able to write their own database applications. Database web application programming was the last topic of our database course. It was also a part of the term project in the author's course. Students liked this exciting part of the project.

References:

[1] Catlell R. & Hamilton G., JDBC database access with Java, Addison Wesley 1997

[2] Internet & World Wide Web: How to program, Deitl, Deitel & Nieto,Prentice Hall, 2000

[3] Connolly, T & Begg, C.& Strachan A. Database systems: A practical Approach to design, Implementation, and management, Addison Wesley, 1999

[4] Morisseau-Leroy, Solomon & Momplaisir, Oracle 8i: SQLJ Programming, Osborne, 2000