Fall 2010 91.573 Database Homework 8 Solution

Due: November 26, 2010

5.13

Suppose you were asked to define a class MetaDisplay in Java, containing a method static void printTable(String r); the method takes a relation name r as input, executes the query “select * from r”, and prints the result out in nice tabular format, with the attribute names displayed in the header of the table.

a. What do you need to know about relation r to be able to print the result in the specified tabular format.

b. What JDBC methods(s) can get you the required information?

c. Write the method printTable(String r) using the JDBC API.

Answer:

a. We need to know the number of attributes and names of attributes of r to decide the number and names of columns in the table.

b. We can use the JDBC methods getColumnCount() and getColumnLabel(int) to get the required information.

c. The method is shown below.
static void printTable(String r) {
    try {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@db.yale.edu:2000:univdb", user, passwd);
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(r);
        ResultSetMetaData rsmd = rs.getMetaData();
        int count = rsmd.getColumnCount();
        System.out.println("<tr>");
        for (int i = 1; i <= count; i++) {
            System.out.println("<td>" + rsmd.getColumnName(i) + "</td>");
        }
        System.out.println("</tr>");
        while (rs.next()) {
            System.out.println("<tr>");
            for (int i = 1; i <= count; i++) {
                System.out.println("<td>" + rs.getString(i) + "</td>");
            }
            System.out.println("</tr>");
        }
        stmt.close();
        conn.close();
    } catch (SQLException sqle) {
        System.out.println("SQLException : " + sqle);
    }
}

5.14

Repeat Exercise 5.13 using ODBC, defining void printTable(char *r) as a function instead of a method.

Answer:

a. Same as for JDBC.

b. The function SQLNumResultCols(hstmt, &numColumn) can be used to find the number of columns in a statement, while the function SQLColAttribute() can be used to find the name, type and other information about any column of a result set set, and the names

c. The ODBC code is similar to the JDBC code, but significantly longer. ODBC code that carries out this task may be found online at the URL http://msdn.microsoft.com/en-us/library/ms713558.aspx (look at the bottom of the page).

9.1
What is the main reason why servlets give better performance than programs that use the common gateway interface (CGI), even though Java programs generally run slower than C or C++ programs?

**Answer:** The CGI interface starts a new process to service each request, which has a significant operating system overhead. On the other hand, servlets are run as threads of an existing process, avoiding this overhead. Further, the process running threads could be the Web server process itself, avoiding interprocess communication which can be expensive. Thus, for small to moderate sized tasks, the overhead of Java is less than the overheads saved by avoiding process creating and communication. For tasks involving a lot of CPU activity, this may not be the case, and using CGI with a C or C++ program may give better performance.

9.15

SQL injection attack occurs when a malicious user (attacker) manages to get an application to execute an SQL query created by the attacker. If an application constructs an SQL query string by concatenating the user supplied parameters, the application is prone to SQL injection attacks. For example, suppose an application constructs and executes a query to retrieve a user's password in the following way:

```java
String userid = request.getParameter("userid");
executeQuery("SELECT password FROM userinfo WHERE userid='" + userid + "'";)
```

Now, if a user types the value for the parameter as:

```
John' OR userid='admin
```

the query constructed will be:

```sql
SELECT password FROM userinfo WHERE userid='John' OR userid='admin';
```
This can reveal unauthorized information to the attacker.

Prevention:
Use prepared statements, with any value that is taken as user input (not just text fields, but even options in drop-down menus) passed as a parameter; user input should never be concatenated directly into a query string. The JDBC, ODBC, ADO.NET, or other libraries that provide prepared statements ensure that special characters like quotes are escaped as appropriate for the target database, so that SQL injection attempts will fail.