Brandon Buchanan

# CSCI 2020 Database Fundamentals
# Lab12

Submit your source code as a text document called *last-name_lab_jdbc* with a .java extender to the drop box. Be sure all **SQL AND JAVA** style guidelines are complied with. You may use your notes, SQL files, Oracle documentation, and books.

**NOTE re-use of code does NOT mean re-use of comments. Deductions will be made for 'borrowed' comments.**

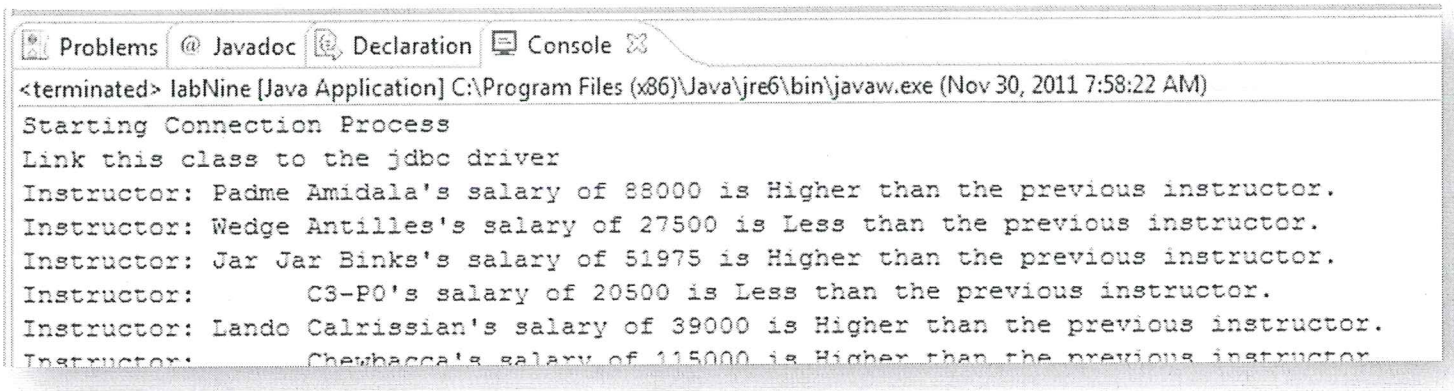
## TASKS

1. Build a project called lab_Twelve that includes a link to the JDBC Oracle driver on the lab machine.

2. Create a class called lab_Twelve that will list the wages, given and family name of each instructor. The list is to be sorted alphabetically by the instructor's last name. The output should also include a fourth column that states if the wages for the current instructor is "higher", "lower" or "the same" as the prior instructor's wages. For the first row, you may assume the prior wages were 0.

3. [BONUS] modify the JAVA code to prevent the string NULL from printing when an instructor has no given_name. ***Do not simply compare the retrieved value to the string "NULL", as this is not a portable solution.***

Example output from Eclipse:



Remember the steps we covered in lecture on Monday:

1) Get all your imports in place
2) Declare ***all*** necessary variables
3) Start a try block (because you may have errors)
4) Link to the JDBC driver
5) Establish your connection
6) Create your statement objects
7) Create your result set object
8) Build & execute your query
9) Process one row at a time
10) Close stuff in the reverse order that you opened it
11) End your try block.

90

```
1 /* Buchanan_Lab_JDBC
2 -------------------------------------------------------------------------
3 Programmer's Name: Brandon Buchanan
4 Course: CSCI2020                          Section Number: 002
5 Creation Date: 11/30/2011        Date of Last Modification: 11/30/2011
6 E-mail address: zbwb8@goldmail.etsu.edu
7 -------------------------------------------------------------------------
8 Purpose -
9  Lab 12
10
11  SQL integration with Java programs
12 -------------------------------------------------------------------------
13 Notes on specifications, special algorithms, and assumptions:
14 Designed to work on the database created by the schooldb.sql
15 script
16 */
17
18 import java.sql.DriverManager;  // DriverManager for establishing linkage to DB
19 import java.sql.SQLException;   // SQLException handles exception for DB classes
20 import java.sql.Connection;     // Connection establishes a logon to the DB
21 import java.sql.Statement;      // Holds and executes the sql statements
22 import java.sql.ResultSet;      // Holds the results of a query
23
24 public class lab_Twelve {
25
26     // Global fields
27     private static String myConnStr;    // holds the connection string
28     private static String myUser;       // holds the user name
29     private static String myPasswd;     // holds the user password
30     private static Connection myCon;    // holds sql connection object
31
32     /**
33      * Main method
34      * @param args
35      */
36     public static void main(String[] args)
37     {
38         // Connect to the SQL server
39         openConnection();
40
41         // Get the results of the query
42         ResultSet results = query();
43
44         // Display retreived information to the console screen
45         displayResults(results);
46
47         // Close the SQL connection
48         closeConnection();
49     }
50
51     /**
52      * Displays the results of the INSTRUCTOR query.
53      * @param results
54      */
```

```
55      private static void displayResults(ResultSet results)
56      {
57          // Store a instructor's salary to be compared with the next instructor
58          int previousSalary = 0;
59
60          try
61          {
62              // Loop through the result set
63              while (results.next())
64              {
65                  // Store the first name, last name, and salary of each instructor
66                  String firstName = results.getString("GIVEN_NAME");
67                  String lastName = results.getString("FAMILY_NAME");
68                  int salary = results.getInt("SALARY");
69
70                  // Create the output string
71                  String output = "Instructor: " + firstName + " " + lastName + "'s
    salary of " + salary;
72                  output += " is ";
73                  if(salary > previousSalary)
74                      output += "higher ";
75                  else if(salary < previousSalary)
76                      output += "lower ";
77                  else
78                      output += "the same ";
79                  output += "than the previous instructor.";
80
81                  // Display the output to the console
82                  System.out.println(output);
83
84                  // Set this instructor's salary to be compared with the next
85                  previousSalary = salary;
86              }
87          }
88          catch(SQLException sqle)
89          {
90              System.out.println("Unable to retrieve a result.");
91          }
92
93      }
94
95      /**
96       * Queries the SQL server for the INSTRUCTOR information.
97       * @return
98       */
99      private static ResultSet query()
100     {
101         String strQuery = "";
102
103         strQuery = "select INSTRUCTOR.GIVEN_NAME, INSTRUCTOR.FAMILY_NAME,
    INSTRUCTOR.SALARY";
104         strQuery += " from INSTRUCTOR";
105         strQuery += " order by INSTRUCTOR.FAMILY_NAME asc";
106
```

Never declare variables inside a loop

-10

```
107         return getResult(strQuery);
108     }
109
110     /**
111      * Retrieves the result of a given SQL query.
112      *
113      * @param query the query to run
114      * @return
115      */
116     private static ResultSet getResult(String query)
117     {
118         Statement myQueryStmt = null;
119
120         try
121         {
122             myQueryStmt = myCon.createStatement();
123         }
124         catch(SQLException sqle)
125         {
126             System.out.println("Unable to create SQL statement.");
127         }
128         catch(NullPointerException npe)
129         {
130             System.out.println("Unable to create SQL statement because the statement
    was null.");
131         }
132
133         ResultSet myResults = null;
134         try
135         {
136             myResults = myQueryStmt.executeQuery(query);
137         }
138         catch(SQLException sqle)
139         {
140             System.out.println("Unable to retrieve the results of the query.");
141         }
142
143         return myResults;
144     }
145
146     /**
147      * Connects to ETSU's Pythia SQL server.
148      */
149     private static void openConnection()
171
172     /**
173      * Logs into ETSU's Pythia SQL server.
174      *
175      * @param server server path to connect to
176      * @param userName user name to be used
177      * @param password password for the user account
178      * @return
179      */
180     private static Connection login(String server, String userName, String password)
```

```
181     {
182         Connection sqlConn = null;
183 
184         try
185         {
186             sqlConn = DriverManager.getConnection(server,userName,password);
187         }
188         catch(SQLException sqle)
189         {
190             System.out.println("Unable to establish the SQL connection.");
191         }
192 
193         return sqlConn;
194     }
195 
196     /**
197      * Closes the SQL connection.
198      */
199     private static void closeConnection()
200     {
201         System.out.println("Closing the SQL connection...");
202 
203         try
204         {
205             myCon.close();
206         }
207         catch(SQLException sqle)
208         {
209             System.out.println("Unable to close the SQL connection.");
210         }
211 
212         System.out.println("SQL connection closed.");
213     }
214 }
```