**Connecting to csdb.brockport.edu from a Java Program**

**By**

**T.M. Rao and Sandeep Mitra**

**Intro:** Until now you have interacted with a database directly using the phpmyadmin tool or via the command line.

You can access csdb from command-line by going to:

**All apps > CSC-CIS-CPS > CSDB MySQL Access**

**It opens a cmd box and asks for a password.**

**----------------------------------------------------------------**

**Enter password: \*\*\*\*\*\*\*\*\***

**Welcome to the MySQL monitor. Commands end with ; or \g.**

**Your MySQL connection id is 4538**

**Server version: 5.1.73-log Source distribution**

**Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.**

**Oracle is a registered trademark of Oracle Corporation and/or its**

**affiliates. Other names may be trademarks of their respective**

**owners.**

**Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.**

**mysql> show databases;**

**+--------------------+**

**| Database |**

**+--------------------+**

**| information\_schema |**

**| EagleRunSchedule |**

**| ProfClothesCloset |**

**| TMRAO\_2017 |**

**| TMRTest |**

**| test\_DB |**

**| test\_date |**

**| trao\_BRAINS |**

**+--------------------+**

**8 rows in set (0.01 sec)**

In this document we will explain how to use a java program as a front-end to a database. Basically, the java program has to establish a database connection. It sends the SQL queries to the database and receives the results sets via this connection. The SQL statements are sent as String objects and the result sets are received as “Vector of Properties” data structures.

**java.util.Properties class:** In java Properties is a class defined in the java.util package. It is a subclass of the Hashtable class. While the Hashtable class can contain (key, value) pairs where both keys and values can be any objects, a Properties object is used to maintain lists of values in which the key is a String and the value is also a String. Thus, a Properties object may have a list like this: (("Name", "Tom"), ("Age","20"), ("Major","CIS")). Here is a simple Java program that illustrates the use of the Properties class. Study the use of **put() and getProperty()** methods. (<http://www.tutorialspoint.com/java/java_properties_class.htm>)

**java.util.Set interface:** Set is a Java interface that extends the Collection interface. It requires methods such as add, addAll, remove, removeAll, etc. It has an iterator() method that returns an Iterator object. Iterator is an interface that requires an implementer to provide methods: boolean hasNext() and E next() method. We can loop through the set elements using a loop like this:

while (itr.hasNext()) { xyz = ite.next(); /\* process the xyz \*/}

**//----------------------------------------------------------------------**

**import java.util.\*;**

**public class PropDemo {**

**public static void main(String args[]) {**

**Properties capitals = new Properties();**

**String str;**

**capitals.put("Illinois", "Springfield");**

**capitals.put("Missouri", "Jefferson City");**

**capitals.put("Washington", "Olympia");**

**capitals.put("California", "Sacramento");**

**capitals.put("Indiana", "Indianapolis");**

**// Show all states and capitals in hashtable.**

**Set states = capitals.keySet(); // get set-view of keys**

**Iterator itr = states.iterator();**

**while(itr.hasNext()) {**

**str = (String) itr.next();**

**System.out.println("The capital of " +**

**str + " is " + capitals.getProperty(str) + ".");**

**}**

**System.out.println();**

**// look for state not in list -- specify default**

**str = capitals.getProperty("Florida", "Not Found");**

**System.out.println("The capital of Florida is "**

**+ str + ".");**

**}**

**}**

**This would produce the following result:**

**The capital of Missouri is Jefferson City.**

**The capital of Illinois is Springfield.**

**The capital of Indiana is Indianapolis.**

**The capital of California is Sacramento.**

**The capital of Washington is Olympia.**

**The capital of Florida is Not Found.**

**java.util.Vector class:** A vector, also a part of the java.util package, is a dynamic-sized array (Its size can grow or shrink). The following program illustrates the use of the Vector class. (<http://www.tutorialspoint.com/java/java_vector_class.htm>) Study the use of addElement, contains, elements, enumeration methods.

**import java.util.\*;**

**public class VectorDemo {**

**public static void main(String args[]) {**

**// initial size is 3, and it increases by 2 each time it hits capacity**

**Vector v = new Vector(3, 2);**

**System.out.println("Initial size: " + v.size());**

**System.out.println("Initial capacity: " +**

**v.capacity());**

**v.addElement(new Integer(1));**

**v.addElement(new Integer(2));**

**v.addElement(new Integer(3));**

**v.addElement(new Integer(4));**

**System.out.println("Capacity after four additions: " +**

**v.capacity());**

**v.addElement(new Double(5.45));**

**System.out.println("Current capacity: " +**

**v.capacity());**

**v.addElement(new Double(6.08));**

**v.addElement(new Integer(7));**

**System.out.println("Current capacity: " +**

**v.capacity());**

**v.addElement(new Float(9.4));**

**v.addElement(new Integer(10));**

**System.out.println("Current capacity: " +**

**v.capacity());**

**v.addElement(new Integer(11));**

**v.addElement(new Integer(12));**

**System.out.println("First element: " +**

**(Integer)v.firstElement());**

**System.out.println("Last element: " +**

**(Integer)v.lastElement());**

**if(v.contains(new Integer(3)))**

**System.out.println("Vector contains 3.");**

**// enumerate the elements in the vector.**

**Enumeration vEnum = v.elements();**

**System.out.println("\nElements in vector:");**

**while(vEnum.hasMoreElements())**

**System.out.print(vEnum.nextElement() + " ");**

**System.out.println();**

**}**

**}**

**This would produce the following result:**

**Initial size: 0**

**Initial capacity: 3**

**Capacity after four additions: 5**

**Current capacity: 5**

**Current capacity: 7**

**Current capacity: 9**

**First element: 1**

**Last element: 12**

**Vector contains 3.**

**Elements in vector:**

**1 2 3 4 5.45 6.08 7 9.4 10 11 12**

**Vector of Properties:** We can use a vector of properties as a data structure to represent the contents of a database table. For example, the table:

| [**Id**](http://csdb.brockport.edu/phpmyadmin/sql.php?db=test&table=student&token=f46fe8c6aef5656405aeba4f6e79aae6&sql_query=SELECT+%2A+FROM+student+ORDER+BY+%60student%60.%60Id%60+ASC) | [**Name**](http://csdb.brockport.edu/phpmyadmin/sql.php?db=test&table=student&token=f46fe8c6aef5656405aeba4f6e79aae6&sql_query=SELECT+%2A+FROM+student+ORDER+BY+%60student%60.%60Name%60+ASC) | [**Address**](http://csdb.brockport.edu/phpmyadmin/sql.php?db=test&table=student&token=f46fe8c6aef5656405aeba4f6e79aae6&sql_query=SELECT+%2A+FROM+student+ORDER+BY+%60student%60.%60Address%60+ASC) | [**Status**](http://csdb.brockport.edu/phpmyadmin/sql.php?db=test&table=student&token=f46fe8c6aef5656405aeba4f6e79aae6&sql_query=SELECT+%2A+FROM+student+ORDER+BY+%60student%60.%60Status%60+ASC) |
| --- | --- | --- | --- |
| **111111111** | **John Doe** | **123 Main St.** | **Freshman** |
| **666666666** | **Joseph Public** | **666 Hollow Rd.** | **Sophomore** |
| **987654321** | **Bart Simpson** | **Fox 5 TV** | **Senior** |
| **123454321** | **Joe Blow** | **6 Yard Ct.** | **Junior** |

Can be represented as

**(("Id", "111111111"), ("Name", "John Doe"), ("Address", "123 Main St."), ("Status", "Freshman")**

**("Id", "666666666"), ("Name", "Joseph Public"), ("Address", "666 Hollow Rd."), ("Status", "Sophomore")**

**("Id", "987654321"), ("Name", "Bart Simpson"), ("Address", "Fox 5 TV"), ("Status", "Senior")**

**("Id", "123454321"), ("Name", "Joe Blow"), ("Address", "6 Yard Ct."), ("Status", "Junior"))**

When the Java program sends a query, the database will return the result set as a Vector of Properties objects. It is up to the Java program to display this to the user.

**Vector of Properties Example**

**//-----------------------------------------------------------------**

**import java.util.\*;**

**public class PropertiesDemo**

**{**

**public static void main(String[] args)**

**{**

**/\***

**\* Data Strructure to handle a relational table in Java**

**\* Vector of Properties**

**\* PERSON TABLE**

**\* ID Name Age Gender**

**\* ----------------------------**

**\* 001 Bob 21 M**

**\* 002 Tom 22 M**

**\* 003 Jen 20 F**

**\* ---------------------------**

**\*/**

**Properties row1 = new Properties();**

**row1.put("Id", "001");**

**row1.put("Name", "Bob");**

**row1.put("Age", "21");**

**row1.put("Gender", "M");**

**Properties row2 = new Properties();**

**row2.put("Id", "002");**

**row2.put("Name", "Tom");**

**row2.put("Age", "22");**

**row2.put("Gender", "M");**

**Properties row3 = new Properties();**

**row3.put("Id", "003");**

**row3.put("Name", "Jen");**

**row3.put("Age", "20");**

**row3.put("Gender", "F");**

**printPairs(row1);**

**printPairs(row2);**

**printPairs(row3);**

**Vector<Properties> PERSON = new Vector<Properties>();**

**PERSON.add(row1);**

**PERSON.add(row2);**

**PERSON.add(row3);**

**printValues(PERSON);**

**}**

**//---------------------------------------------------------**

**public static void printPairs(Properties p)**

**{**

**System.out.println("(Id, "+p.getProperty("Id")+")");**

**System.out.println("(Name, "+p.getProperty("Name")+")");**

**System.out.println("(Age, "+p.getProperty("Age")+")");**

**System.out.println("(Gender, "+p.getProperty("Gender")+")");**

**}**

**//---------------------------------------------------------**

**public static void printValues(Vector<Properties> v)**

**{**

**for (Properties p : v)**

**{**

**Enumeration props = p.propertyNames();**

**while (props.hasMoreElements())**

**System.out.print(**

**p.getProperty((String)(props.nextElement()))+"\t");**

**System.out.println();**

**}**

**}**

**//---------------------------------------------------------**

**}**

**The JDBC API:** The JDBC (Java Database Connectivity) is a java API that enables us to connect and communicate with a database. A jar file containing the JDBC will be given to you. It provides important classes and interfaces such as Driver, DriverManager, Connection, Statement, ResultSet, and SQLException. In this class, we do not deal with the JDBC directly. Instead we use a set of packages that are built on top of the JDBC. We will provide you with a directory which has the following structure

**Directory JDBCCode**:

**Sub-directory classes**: This is a directory that contains .class files. You don’t have to do anything with it.

**Sub-directory common:** This contains two classes PropertyFile.java, StringList.java. You don’t need to change anything in the directory. Just let it be there.

**Sub-directory database:** This contains the JDBCBroker.java class. Just leave it as is.

**Sub-directory event:** This contains Event.java and EventLog.java classes. Just leave it as is.

**Classes DatabaseAccessor.java, DatabaseManipulator.java, DatabaseMutator.java:** These are classes that contain the support code to connect to the database.

**Text file dbConfig.ini:** This contains the user-id, password, etc. details needed to establish a connection. You need to edit this file to enter your user-id and password, and also the name of the database and database server you are connecting to. For example,

**username=trao**

**password=\*\*\*\*\***

**dbName=TMRAO\_2017**

**dbServer=csdb.brockport.edu**

**Jar file mysql-connector-java-5.1.7-bin.jar:** This is a jar file (java archive). This contains all the jdbc code. Just leave it as is.

**Java file CollegeDBTesterGUI.java:** This is a tester program that illustrates how to use this infrastructure. We use a database called College. It has tables:

DEPARTMENT\_15(DepartmentId, DepartmentName)

COURSE\_15(DepartmentId, CourseCode, CourseName, CourseDescription)

PROFESSOR\_15(ProfessorId, ProfessorName, DepartmentId)

STUDENT\_15(BannerId, StudentName, Address, Status)

Note: Status can be Freshman, Sophomore, Junior, Senior

TRANSCRIPT\_15(StudentId, CourseCode, TR\_Semester, TR\_Section, TR\_grade)

[DepartmentId](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=SELECT+%2A+FROM+%60COURSE_15%60%0AORDER+BY+%60COURSE_15%60.%60DepartmentId%60+ASC&session_max_rows=30&token=56f1ae1344c4bac1f376ec0a042a4af3)[CourseCode](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=SELECT+%2A+FROM+%60COURSE_15%60%0AORDER+BY+%60COURSE_15%60.%60CourseCode%60+ASC&session_max_rows=30&token=56f1ae1344c4bac1f376ec0a042a4af3)[CourseName](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=SELECT+%2A+FROM+%60COURSE_15%60%0AORDER+BY+%60COURSE_15%60.%60CourseName%60+ASC&session_max_rows=30&token=56f1ae1344c4bac1f376ec0a042a4af3)[CourseDescription](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=SELECT+%2A+FROM+%60COURSE_15%60%0AORDER+BY+%60COURSE_15%60.%60CourseDescription%60+ASC&session_max_rows=30&token=56f1ae1344c4bac1f376ec0a042a4af3)[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27CIS202%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27CIS202%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27CIS202%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[CSC](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27CSC%27&token=56f1ae1344c4bac1f376ec0a042a4af3)CIS202Fundamentals of Information Systbla bla[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27CSC203%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27CSC203%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27CSC203%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[CSC](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27CSC%27&token=56f1ae1344c4bac1f376ec0a042a4af3)CSC203Fundamentals of Computer Sciencebla bla[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH201%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH201%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH201%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[MTH](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27MTH%27&token=56f1ae1344c4bac1f376ec0a042a4af3)MTH201Calculus 1bla bla[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH281%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH281%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27MTH281%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[MTH](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27MTH%27&token=56f1ae1344c4bac1f376ec0a042a4af3)MTH281Discrete Mathematics 1bla bla[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH110%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH110%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH110%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[PSH](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27PSH%27&token=56f1ae1344c4bac1f376ec0a042a4af3)PSH110Principles of Psychologybla bla[Edit Edit](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH201%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=update&token=56f1ae1344c4bac1f376ec0a042a4af3)[Copy Copy](https://csdb.brockport.edu/phpmyadmin/tbl_change.php?db=TMRAO_2017&table=COURSE_15&where_clause=%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH201%27&clause_is_unique=1&sql_query=SELECT+%2A+FROM+%60COURSE_15%60&goto=sql.php&default_action=insert&token=56f1ae1344c4bac1f376ec0a042a4af3)[Delete Delete](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=COURSE_15&sql_query=DELETE+FROM+%60TMRAO_2017%60.%60COURSE_15%60+WHERE+%60COURSE_15%60.%60CourseCode%60+%3D+%27PSH201%27&message_to_show=The+row+has+been+deleted&goto=sql.php%3Fdb%3DTMRAO_2017%26table%3DCOURSE_15%26sql_query%3DSELECT%2B%252A%2BFROM%2B%2560COURSE_15%2560%26message_to_show%3DThe%2Brow%2Bhas%2Bbeen%2Bdeleted%26goto%3Dtbl_sql.php%26token%3D56f1ae1344c4bac1f376ec0a042a4af3&token=56f1ae1344c4bac1f376ec0a042a4af3)[PSH](https://csdb.brockport.edu/phpmyadmin/sql.php?db=TMRAO_2017&table=DEPARTMENT_15&pos=0&sql_query=SELECT+%2A+FROM+%60TMRAO_2017%60.%60DEPARTMENT_15%60+WHERE+%60DepartmentId%60+%3D+%27PSH%27&token=56f1ae1344c4bac1f376ec0a042a4af3)PSH201Introduction to Statisticsbla blaBottom of Form

Top of Form

Bottom of Form

Here are some of the methods in this class:

**//--------------------------------------------------------------------------**

**// This is a supporting method used by the retrieve method**

**// Convert a Vector of Properties to a printable String**

**private static String vecToString(Vector<Properties> data)**

**{**

**String result = "";**

**// Now, we have to print out these rows in a user-understandable form**

**if ((data == null) || (data.size() == 0))**

**{**

**return ("No results were returned from database for this query");**

**}**

**else**

**{**

**result = "==============================================\n";**

**Properties p1 = data.firstElement();**

**Enumeration props1 = p1.propertyNames();**

**while (props1.hasMoreElements())**

**result += (props1.nextElement()+"\t");**

**result += "\n";**

**result += "----------------------------------------------\n";**

**// Now go thru the entire 'data' Vector,**

**// get each Properties object out of it**

**// and print out the contents of the Properties object**

**for (Properties p : data)**

**{**

**Enumeration props = p.propertyNames();**

**while (props.hasMoreElements())**

**result += p.getProperty**

**((String)(props.nextElement()))+"\t");**

**result += "\n";**

**}**

**result += "==============================================";**

**}**

**return result;**

**}**

**//--------------------------------------------------------------------------**

**// The following four methods implement the retrieve, insert, update and**

**// delete operations**

**//--------------------------------------------------------------------------**

**public static String retrieveFromTable(String queryString)**

**{**

**// First, set up an instance of the DatabaseAccessor class**

**DatabaseAccessor dbAcc = new DatabaseAccessor();**

**// Now that you have created the query string,**

**// you set that on the DatabaseAccessor object you created**

**// using the 'setSQLStatement()' method as shown below**

**dbAcc.setSQLStatement(queryString);**

**// Then invoke the method 'executeSQLSelectStatement()'**

**// on the DatabaseAccessor object as shown below to run the query.**

**// The result of running this query is a Vector of**

**// Properties objects. Each Properties object in this Vector contains**

**// the data from one of the db table rows matching the query**

**Vector<Properties> returnedValues = dbAcc.executeSQLSelectStatement();**

**return vecToString(returnedValues);**

**}**

**//---------------------------------------------------------------------------**

**public static void insertIntoTable(String insertQueryString )**

**{**

**DatabaseMutator dbMut = new DatabaseMutator();**

**dbMut.setSQLStatement(insertQueryString);**

**Integer returnedValue = dbMut.executeSQLMutateStatement();**

**if (returnedValue != 1)**

**System.out.println("Error in db insertion");**

**else**

**System.out.println("Row inserted successfully");**

**}**

**//---------------------------------------------------------------------------**

**public static void updateTable(String updateQueryString)**

**{**

**DatabaseMutator dbMut = new DatabaseMutator();**

**dbMut.setSQLStatement(updateQueryString);**

**Integer returnedValue = dbMut.executeSQLMutateStatement();**

**if (returnedValue < 0)**

**System.out.println("Error in db update");**

**else**

**System.out.println("Row updated successfully");**

**}**

**//------------------------------------------------------------------------**

**public static void deleteFromTable(String deleteQueryString)**

**{**

**DatabaseMutator dbMut = new DatabaseMutator();**

**dbMut.setSQLStatement(deleteQueryString);**

**Integer returnedValue = dbMut.executeSQLMutateStatement();**

**if (returnedValue < 0)**

**System.out.println("Error in db Delete");**

**else**

**System.out.println("Row deleted successfully");**

**}**

**How to compile and run these programs from command line:**

Firstly, you only edit and make changes into the Tester class. In our example we have called it DepartmentDBTester. You can call it whatever depending on the database tables you are working on. Observe that the JDBCCode directory has two batch files: **compileTester.bat** and **runTester.bat**.

The compileTester.bat has the following DOS commands:

**javac -d classes -classpath classes;. common\\*.java event\\*.java database\\*.java**

**javac -classpath classes;. DatabaseManipulator.java**

**javac -classpath classes;. DatabaseAccessor.java**

**javac -classpath classes;. DatabaseMutator.java**

**javac -classpath classes;. CollegeDBTesterGUI.java**

**Open a DOS window by typing 'cmd' in the START menu.**

**In the DOS window, type compileTester as a command. This will compile all the Java code. (Get all the errors out!! – you know that, right?)**

The runTester.bat has the following DOS commands:

**java -cp mysql-connector-java-5.1.7-bin.jar;classes;. DepartmentDBTester**

Type runTester as a DOS command. This will run the program.

You will have to edit these files appropriately to include the name of your tester class in place of DepartmentDBtester. You double-click on compileTester.bat to compile all the files. Once you have gotten rid of all compile errors, you can double-click on runtester.bat to run your program.

**How to do Database Connection in Eclipse:**

1. Create a new Project called **DatabaseConnector. Make sure that the workspace you create this project is in your p: drive (i.e. filecity).**
2. How to copy the **mysql-connector-java-5.1.7-bin.jar** file into Eclipse:

* Right-click on the project name (i.e. DatabaseConnector)
* A dialog opens, Click on “Build Path”
* Click on “Add External Archives”. It gives you a dialog. Choose Browse and navigate to the location of the .jar file and click open. This will create the jar file under referenced libraries.

1. How to create packages “event”, “common”, “database” (Create them in this order):
   1. For each of the packages, first create a package in Eclipse: (say event)
      1. Click on src, right click on src, then click on new, then click on package, then enter the package name (i.e., event)
   2. Click on the package name (event), right click on it, select import, (may have to click on “General” first) choose “File System”, navigate to the directory on your computer where you have the Java files for this package, and click on it.
   3. Within Eclipse click on checkbox containing the package name to select all files
   4. Click finish.
   5. Do the same for all the other packages
2. Create these classes in the default package in this order and copy-paste the code as described below: InvalidReturnValueException.java, DatabaseManipulator.java, DatabaseAccessor.java, DatabaseMutator.java
   1. Create the class in Eclipse (say : InvalidReturnValueException.java)
      1. Right-click on src, select new, select class, enter the name of class InvalidReturnValueException.
   2. Copy-paste the code from wherever it exists on your computer (you might want to open this file in a program like Notepad, then select all its contents, select copy and paste all the content into the Eclipse window for this class).
   3. Do the same for all other classes mentioned above.
3. How to create the application program (say DepartmentDBTester.java)
   1. Create the class as described above and write the code. See the code provided in DepartmentDBTester.java for a model.
4. Where to store the dbconfig.ini file
   1. Click project name (DatabaseConnector)
   2. Right click, new, file. It opens a dialog. Enter the filename dbconfig.ini and click “Finish”. It opens a notepad. Copy the dbconfig.ini contents and save it.

Now you can compile and run your tester program using Eclipse.