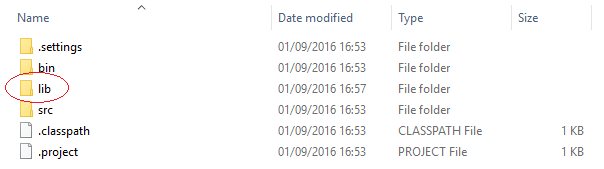
# Data Centric RAD

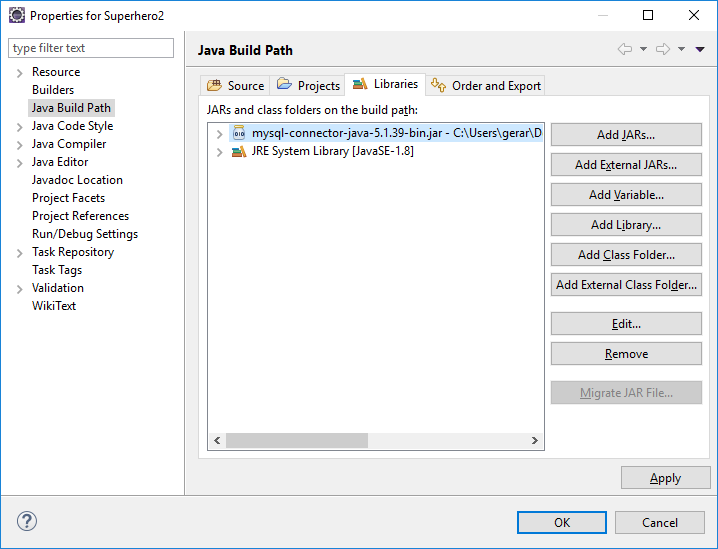
## Lab 4 JDBC

### Part 1 Setting up a JDBC Project

* Open Eclipse Java EE IDE for Web Developers.
* Select *File/New/Other…*
* When the dialog box opens, select *Java Project*.
* Give the Project a *Name* and press *Finish*.
* In the Windows File Explorer, navigate to the project you just created and create a folder called *lib*, so that your project now looks as follows:



* Download the MySQL driver from <https://dev.mysql.com/downloads/connector/j/>.
* After unzipping the file, copy mysql-connector-java-5.1.39-bin.jar to the *lib* folder previously created.
* In Eclipse, select the project you just created and then select *Project/Properties*.
* Click on *Java Build Path*, then the *Libraries* tab, then press the *Add JARs…*
* Select the mysql-connector-java-5.1.39-bin.jar file in the lib directory of the project and press *Open.*
* When the mysql-connector-java-5.1.39-bin.jar appears on the build path (see below) press OK.



* In Eclipse, click on the *src* folder and select *File/New/Class*.
* Enter a Package name, and a Class name. Press *Finish*.
* At the top of the Class just created type the following:

**import** java.sql.Connection;

**import** java.sql.Statement;

**import** java.sql.ResultSet;

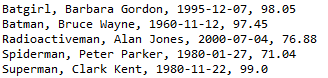
**import** java.sql.SQLException;

If there is no error saying the import cannot be resolved, this means the project is setup correctly.

* Start WAMP Server.
* To run each exercise below, select the .java file and select *Run/Run As/Java Application* in Eclipse.

### Part 2

* Get superheroes\_wk4.sql from Moodle.
* Import it into MySQL as described in Lab 1.
* Using the superheroes database, write a JDBC application to print out all details of all superheroes to the console as follows:



**package** lab4;

**import** java.sql.Connection;

**import** java.sql.Statement;

**import** com.mysql.jdbc.jdbc2.optional.MysqlDataSource;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**public** **class** mongo {

**static** **final** String ***JDBC\_DRIVER*** = "com.mtsql.jdbc.Driver";

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

**try** {

MysqlDataSource mysqlDS = **new** MysqlDataSource();

mysqlDS.setURL("jdbc:mysql://localhost:3306/superheroes");

mysqlDS.setUser("root");

mysqlDS.setPassword("");

Connection conn = mysqlDS.getConnection();

Statement myStmt = conn.createStatement();

String query = "select \* from superhero\_table;";

ResultSet rs = myStmt.executeQuery(query);

**while**( rs.next() ) {

// id = rs.getInt("id");

String name = rs.getString("name");

String realfirstname = rs.getString("real\_first\_name");

String realsurname = rs.getString("real\_surname");

String dob = rs.getString("dob");

**double** powers = rs.getInt("powers");

System.***out***.print(name + ", ");

System.***out***.print(realfirstname+", ");

System.***out***.print(realsurname+", ");

System.***out***.print(dob+", ");

System.***out***.println(powers+", ");

}

} **catch**( SQLException se ) {

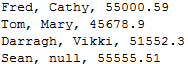
System.***out***.println(se.getMessage());

}

}

Part 3

* Get employee\_kin.sql from Moodle.
* Import it into MySQL as described in Lab 1.
* Using the employee\_kin database, write a JDBC application to print out the Employee Name, Next Of Kin Name, and the Employee Salary of ALL employees on the console as follows:



**package** lab4;

**import** java.sql.Connection;

**import** java.sql.Statement;

**import** com.mysql.jdbc.jdbc2.optional.MysqlDataSource;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**public** **class** mongo {

**static** **final** String ***JDBC\_DRIVER*** = "com.mtsql.jdbc.Driver";

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

**try** {

MysqlDataSource mysqlDS = **new** MysqlDataSource();

mysqlDS.setURL("jdbc:mysql://localhost:3306/employee\_kin");

mysqlDS.setUser("root");

mysqlDS.setPassword("");

Connection conn = mysqlDS.getConnection();

Statement myStmt = conn.createStatement();

String query = "select e.ename,n.NOK\_Name,s.salary\r\n"+

"from employee\_table e\r\n" + "left join next\_of\_kin\_table n\r\n" + "on e.NextOfKin = n.NOK\_ID\r\n" + "left join salary s\r\n" + "on s.eid = e.eid;";

ResultSet rs = myStmt.executeQuery(query);

**while**( rs.next() ) {

String name = rs.getString("ename");

String Nokname = rs.getString("NOK\_Name");

**double** salary = rs.getInt("salary");

System.***out***.print(name + ", ");

System.***out***.print(Nokname+", ");

System.***out***.println(salary+", ");

}

} **catch**( SQLException se ) {

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

}

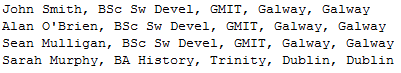
}

}

### Part 4

* Get studentDB3.sql from Moodle.
* Import it into MySQL as described in Lab 1.
* Using the studentdb3 database, write a JDBC application to print out
  + the Student Name
  + the Name of the course he/she is doing
  + the name of the college he/she is attending
  + name of the county and
  + main town of the county in which the college is located

on the console as follows:



The Student Name should be retrieved from the database as *Student*.

The Course Name should be retrieved from the database as *Course*.

The College Name should be retrieved from the database as *Course*.

The County Name should be retrieved from the database as *County*.

The Main Town should be retrieved from the database as *Main Town*.

**package** lab4;

**import** java.sql.Connection;

**import** java.sql.Statement;

**import** com.mysql.jdbc.jdbc2.optional.MysqlDataSource;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**public** **class** mongo {

**static** **final** String ***JDBC\_DRIVER*** = "com.mtsql.jdbc.Driver";

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

**try** {

MysqlDataSource mysqlDS = **new** MysqlDataSource();

mysqlDS.setURL("jdbc:mysql://localhost:3306/studentDB3");

mysqlDS.setUser("root");

mysqlDS.setPassword("");

Connection conn = mysqlDS.getConnection();

Statement myStmt = conn.createStatement();

String query =

"selects.student\_name ,co.course\_name,\r\n"+ "c.college\_name ,cou.county\_name ,cou.main\_town\r\n"+

"from student\_table s\r\n"+

"inner join course\_table co\r\n"+

"on co.course\_id = s.course\_ID\r\n"+

"inner join college\_table c\r\n"+

"on c.college\_id = co.college\_id\r\n"+

"inner join county\_table cou\r\n"+

"on cou.county\_name = c.county ;";

ResultSet rs = myStmt.executeQuery(query);

**while**( rs.next() ) {

String name = rs.getString("student\_name");

String coursename = rs.getString("course\_name");

String collegename = rs.getString("college\_name");

String town = rs.getString("main\_town");

String countyname = rs.getString("county\_name");

System.***out***.print(name + ", ");

System.***out***.print(coursename+", ");

System.***out***.print(collegename+", ");

System.***out***.print(countyname+", ");

System.***out***.println(town+" .");

}

} **catch**( SQLException se ) {

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

}

}

}

### Part 5

* Using the solution to Part 2 as a basis for this answer include the following lines of code after you get the connection to the database but before you execute the query:

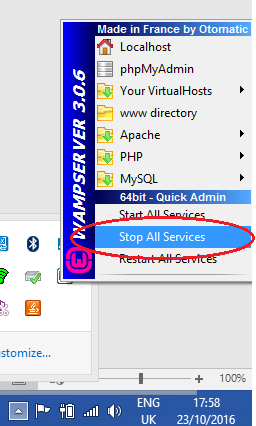
System.out.println("Press Enter on the Console to Continue");

Scanner sc = new Scanner(System.in);

sc.nextLine();

sc.close();

Now run the program and when the line above is displayed on the console, stop the WAMP server as shown:



Now when the database cannot be connected to the only error message to be shown on the console should be:

ERROR: Cannot communicate with superheroes database

### Part 6

* Using the solution to Part 5 as a basis for this answer, update it so that instead of doing a query on the superheroes database, the application inserts the following superhero:

Superhero = Joker

Real Name = Jack Nicholson

DOB = March 21st 1949

Powers = 89.4

If the superhero was successfully inserted into the database, the following should be printed on the console:

Joker successfully inserted into database

And the application should be run again trying to insert the same data into the superhero\_table. This time an error should be displayed on the console saying:

ERROR: Superhero Joker already exists in database

The success/failure messages should apply to any superhero entered i.e. the names shouldn’t be hard-coded.

### Part 7

* Using the solution to Part 6 as a basis for this answer, update it so that the user is prompted for a superhero name to delete from the database.

Depending on the result from the database the following should be printed:







### Part 8

* Using the solution to Part 7 as a basis for this answer, update it so that the user is asked for a Superhero whose powers will then be increased by 1.

If the superhero was updated successfully his/her new powers should be displayed (having first been read from the database); otherwise an error stating that the superhero does not exist should be printed.





