Using Netbeans and the Derby Database for Projects

Contents

1. Prerequisite

To begin with, you should have installed the following software.

1) Java: JDK 7.0
2) NetBeans 7.3.1 with Derby (Java DB)

If not, you should download the JDK (with Java DB included) through JDK official website:

http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

Netbeans can be found on the NetBeans official website (please download the **Java EE** version):

https://netbeans.org/downloads/index.html

2. Creating a Derby Database in Netbeans

a. Accessing Services:

Open NetBeans and you should have the interface shown in Figure 1. The Java DB is located under the tab “Services”. If this tab is not visible, on the main menu: select “Window” and then “Services “as shown in Figure 2.

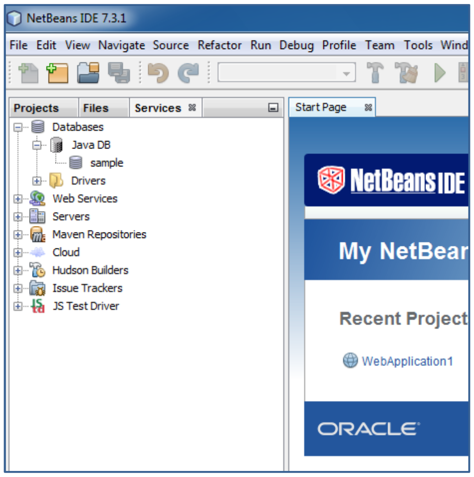


Figure 1: The main Interface of NetBeans

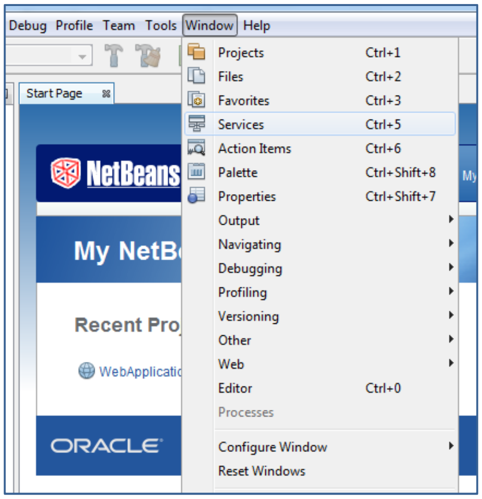
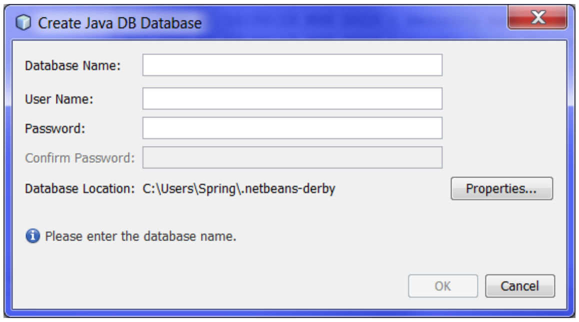


Figure 2: Open Services using Windows→Services

b. To create a new database, in the “services” tab:

1) Expand “Database”
2) Right click “Java DB”, and select “Create Database”.
3) In the Create JavaDB Database window (shown as Figure 3), provide the database name, user name, and password. For class projects, use your name for the database name and use ***IS2560*** as both username and password.
4) When you have created a DBMS, a series of files will be created in your user space as will be described below.



Database Location: C:\Users\Spring\.netbeans-derby

Please enter the database name.

Figure 3: Creating a new database

c. Making a connection through Netbeans

5) After creating a new database, Netbeans will automatically build a connection, shown in the figure 4.

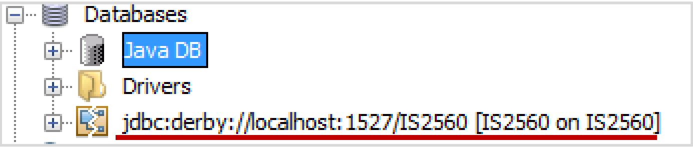


Figure 4: Derby database connections

6) If not, expand "Java DB", you will see a list of existing databases (in our case one called "IS2560"
7) Right click "IS2560" and select "Connect…" (Figure 5)
8) A new connection will be built as shown in Figure 4.

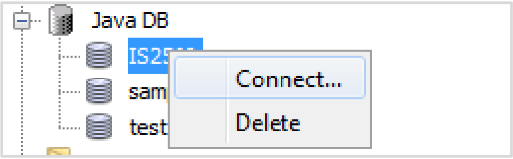


Figure 5: Making a new connection

d. Creating tables

9) Keep in mind, before you can create tables you must make a connection to your DBMS
10) Expand existing connection "jdbc:derby://localhost:1527/IS2560 [IS2560 on IS2560]". If you are not able to expand this connection, make sure the server is started.

    How to Start Derby Server : Right click "Java DB", select "Start Server"

11) Expand "IS2560" (IS2560 is displayed in bold)
12) Right click on "Tables" and select "Create Tables…"
13) In the "Create Table" window (Figure 6), provide Table names and fields. In this example, we named the table as "demo" and added two fields: id (numeric, size=10), name (varchar, size=255)

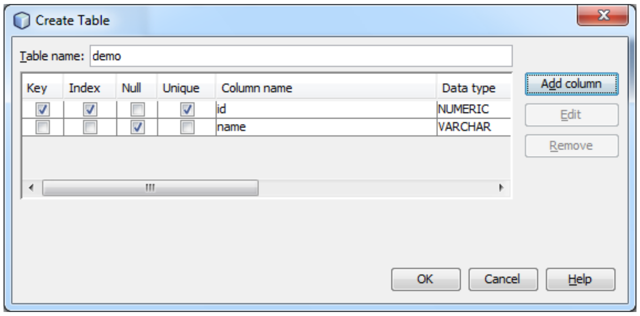


Table name: demo

Key	Index	Null	Unique	Column name	Data type
☑	☑	□	☑	id	NUMERIC
□	□	☑	□	name	VARCHAR

Figure 6: Creating a table schema

14) The created table “demo” will be displayed under your DBMS in the “Services” tab, as shown in Figure 7.

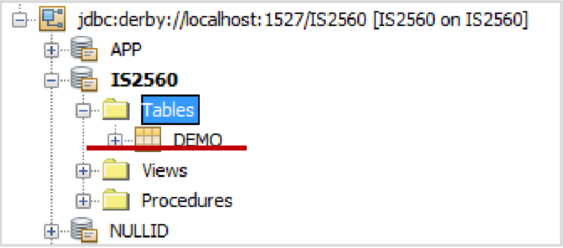


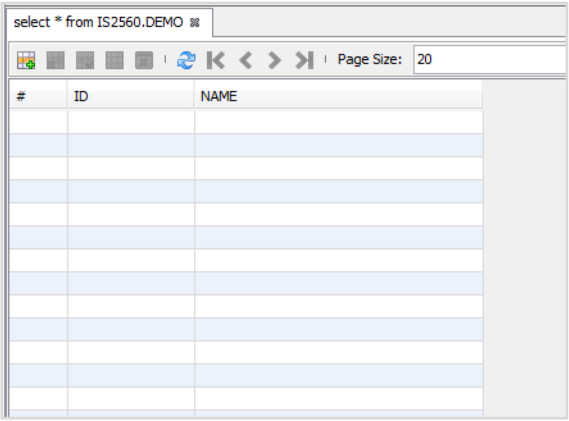
Figure 7: The created table

e. Adding records

15) Expand your DBMS connection “jdbc:derby://localhost:1527/IS2560 [IS2560 on IS2560]”

16) Right click on the table – in our case “DEMO”, select “View Data…”

17) A “select * from IS2560 …” panel (Figure 8) will display on the main workspace



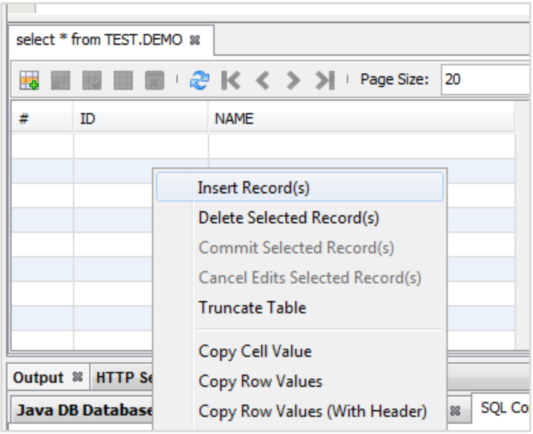
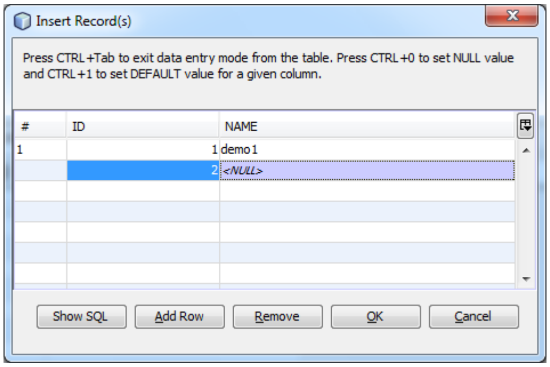


Figure 8: The SQL results panel

Figure 9: Java DB file location

18) Right click on the result panel (Figure 9) and select “Insert records”

19) In the “Insert Records” window (Figure 10), prepare the data you need for IS2560 projects. In this example, we prepared two records: 1, demo1 and 2 demo2.



Press CTRL+Tab to exit data entry mode from the table. Press CTRL+0 to set NULL value and CTRL+1 to set DEFAULT value for a given column.

#	ID	NAME
1	1	demo1
	2	<NULL>

Figure 10: The “Insert Records” window

3. Using the examples provided by the instructor

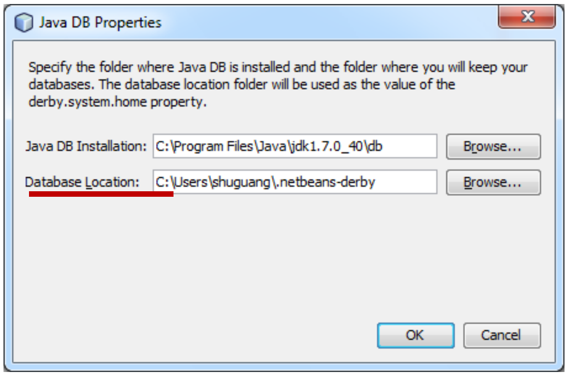
On the class website, several code examples using the Derby database in servlets are provided. This section shows how to use these samples. This example uses "SimpleServlet_DBMS_Derby".

a. Get the files

1) Download the zip file from the class website
2) Unzip it to a location of your choice – the code will be in a folder with the same name as the zip file – in our example"SimpleServlet_DBMS_Derby"

b. Restore the database

3) Open the main folder ("SimpleServlet_DBMS_Derby") and then the folder named "dbms", inside the "dbms" and you will find a folder that is the Derby database – in this case"ss_message". Copy this folder to the Derby DBMS data file location.
    a. To **locate your Java DB file location**, please click "Services" tab
    b. Expand "Database" and select "Java DB"
    c. Right click the "Java DB" and select "properties"
    d. The "Java DB properties" window (Figure 11) provides the Database Location. In our case (in Windows), it is located in "C:\Users\username\.netbeans-derby". If you are in Mac, the location should be different.
    e. Navigate to this folder and paste project database folder "ss_message".



Specify the folder where Java DB is installed and the folder where you will keep your databases. The database location folder will be used as the value of the derby.system.home property.

Java DB Installation: C:\Program Files\Java\jdk1.7.0_40\db

Database Location: C:\Users\shuguang\.netbeans-derby

Figure 11 Locate Java DB database location

Note: In both Mac and Windows, if you cannot find this folder, mostly, it is because this folder is hidden in your computer. You can use this link for view the hidden files:
http://guides.macrumors.com/Viewing_hidden_files_on_a_Mac (Mac)
http://windows.microsoft.com/en-us/windows/show-hidden-files#show-hidden-files=windows-7 (Windows)

c. Run the Project in Netbeans

4) Start Netbeans. (Be sure you have moved the database folder before starting Netbeans – it only looks for these files at startup time. If you are already running Netbeans, you can open the Java DB properties window and change the database location -- I take the y off derby. The DB will indicate there are no DBs. If you now change it back to the correct location, it will find the new DB you added – you also have to restart Java DB.)

5) Right click on "Projects" tab, select "Open Project…" (Figure 12) and select "SimpleServlet_DBMS_Derby" project (Figure 13)

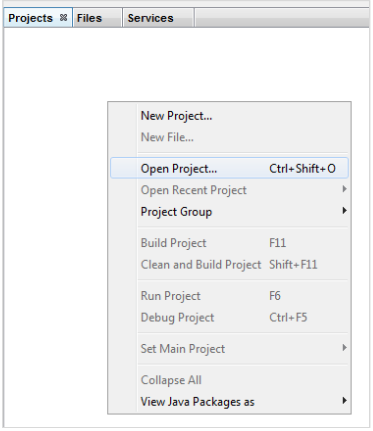
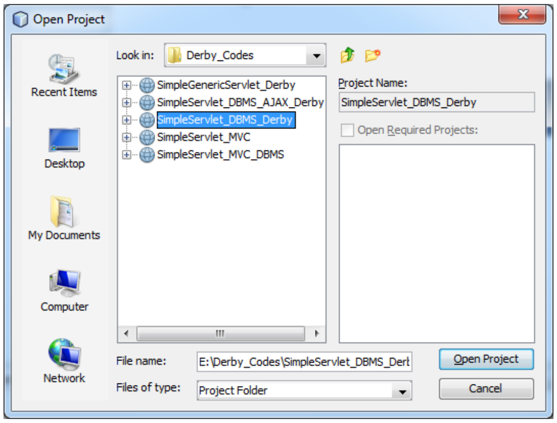


Figure 12 Open existing project



Look in: Derby_Codes

Project Name: SimpleServlet_DBMS_Derby

File name: E:\Derby_Codes\SimpleServlet_DBMS_Dert

Files of type: Project Folder

Figure 13 Open "SimpleServlet_DBMS_Derby" project

6) You may encounter errors because your JDK version is different from what we used in this project. Generally you can right click on project and select "Properties" (Figure 14) to fix such "errors".

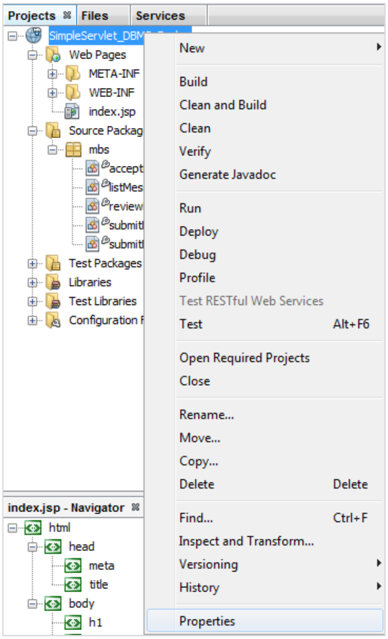


Figure 14: Check project properties

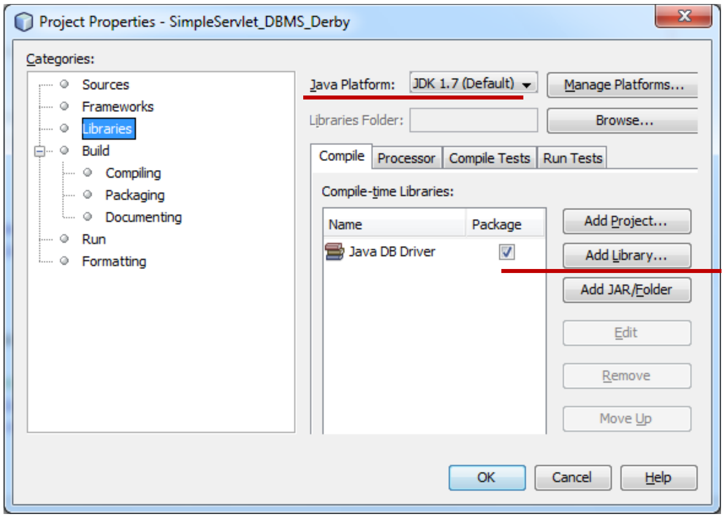
Two common issues relate to the JDK and the Java DB. In the "Project Properties" window , select Libraries, you will see Figure 15In "Java Platform", select the "JDK1.* (Default)"; In "Add Library…", select "Java DB Driver"

7) Click "Services" Tab, expand "Databases" , then right click on "Java DB", choose "Start Server" (The first time you run a sample project, you may also want to connect to the database to make sure the username and password are correct – for our samples we always use TEST and TEST,

8) Click "Projects" Tab, right click on "SimpleServlet_DBMS_Derby" , choose "run".

9) A system will be pop out in your web browser, as shown in Figure 16. You should be able to add messages and list messages in the database.

10) You might want to run the project with the debugger, setting a break point in one of the files so you can walk through the project and see what is going on. The next section will explain how to write your own code.

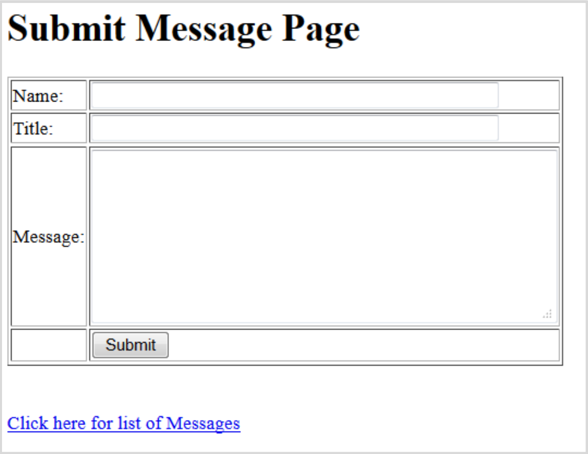


Java Platform: JDK 1.7 (Default)

Compile-time Libraries:

Name	Package
Java DB Driver	☑

Figure 15: Choose the right project libraries



Submit Message Page

Name:	
Title:	
Message:	
	Submit

Click here for list of Messages

Figure 16: System interface

4. Connecting to derby DBMS from a servlet or other Java program

In this section, we will provide examples on how to connect a derby database through a java program. We will use a Java application Web project as an example.

a. Connecting to an existing DBMS schema

1) Click the “Projects” tab
2) Right click in a blank area, or click on the File menu and select “New Project”
3) In the “New Project” window (Figure 17), select “Java Web” and choose “Web Application” (Figure 18)
4) Create a new web application -- in our case named “DerbyWeb”

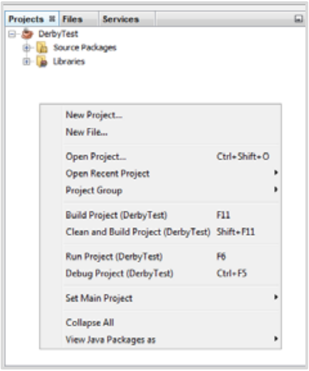
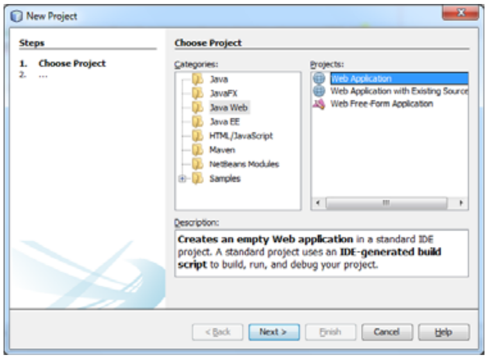

Projects
Files
Services
DerbyTest
Source Packages
Libraries
New Project...
New File...
Open Project... Ctrl+Shift+O
Open Recent Project
Project Group
Build Project (DerbyTest) F11
Clean and Build Project (DerbyTest) Shift+F11
Run Project (DerbyTest) F6
Debug Project (DerbyTest) Ctrl+F5
Set Main Project
Collapse All
View Java Packages as

Figure 17: Create a new project



Creates an empty Web application in a standard IDE project. A standard project uses an **IDE-generated build script** to build, run, and debug your project.

Figure 18: The new project window

5) In order to use Derby in web applications, we need to add Derby drivers to our project library. Expand the project, right click on “Libraries”, and select “Add Library” (Figure 19). (You can also open the project properties window, go to Library and add it there.
6) In the “Add library” window, select “Java DB Driver” (Figure 20).

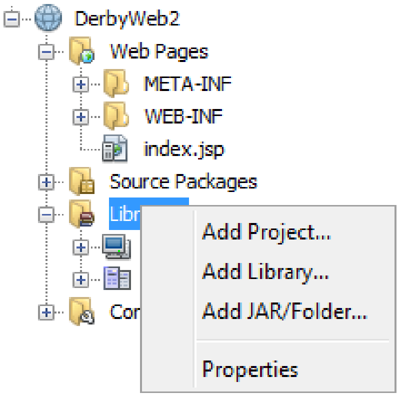


Figure 19: The new project window

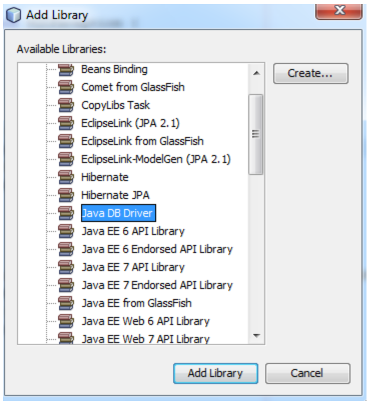


Figure 20: Add Java DB Drivers

7) The following Java code (Figure 21) is used to connect the database. (This is basically the same code used to connect to any database – only the driver changes.)

```
String connectionURL = "jdbc:derby://localhost:1527/IS2560";
//ConnectionURL, username and password should be specified in getConnection()
try {
   Connection conn = DriverManager.getConnection(connectionURL, "IS2560", "IS2560");
   System.out.println("Connect successfully ! ");
   conn.close();
} catch (SQLException ex) {
   System.out.println("Connect failed ! ");
}
```

Figure 21: Derby Connection

8) Normally, running a project will cause the Derby Database to start. If you get the message “connection failed” it may because the Derby server was not started. To Start Derby Server manually, go to the services tab, right click “Java DB”, select “Start Server”

b. Selecting, adding and modifying

9) Through the standard SQL scripts, we could add, modify or select data from database.

10) Figure 22, 23 and 24 provided examples of using “SELECT”, “INSERT” and “UPDATE” to select, add and modify data. An important note: in selecting data from database, use java.sql.Statement.executeQuery method, while for updating and adding new data, use java.sql.Statement.executeUpdate method instead.

```
String connectionURL = "jdbc:derby://localhost:1527/IS2560";
 //ConnectionURL, username and password should be specified in getConnection()
 try {
    Connection conn = DriverManager.getConnection(connectionURL, "IS2560", "IS2560");
    String sql = "SELECT * FROM demo";
    Statement st = conn.createStatement();
            ResultSet rs=null;
            rs=st.executeQuery(sql);
            while(rs.next()){
                System.out.println(rs.getInt("id")+"\t"+rs.getString("name"));
             }
             rs.close();
             st.close();
             conn.close();
 } catch (SQLException ex) {
    System.out.println("Connect failed ! ");
 }
```

Figure 22: Test SQL SELECT to select data

```
 String connectionURL = "jdbc:derby://localhost:1527/IS2560";
//ConnectionURL, username and password should be specified in getConnection()
 try {
     Connection conn = DriverManager.getConnection(connectionURL, "IS2560", "IS2560");
     String sql = "INSERT INTO IS2560.DEMO (ID, NAME) VALUES (3, 'demo3')";
     Statement st = conn.createStatement();
               st.executeUpdate(sql);
               st.close();
               conn.close();
 } catch (SQLException ex) {
      System.out.println("Connect failed ! ");
 }
```

Figure 23: Test SQL INSERT to add new data

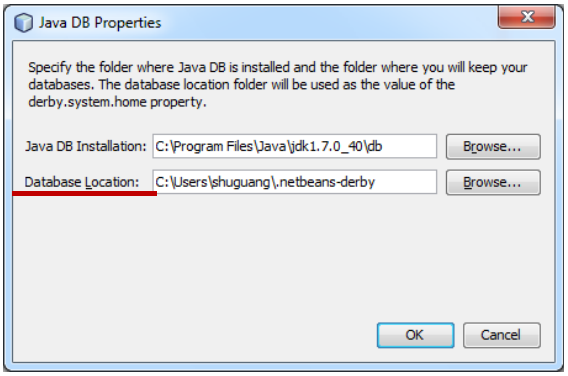
```
String connectionURL = "jdbc:derby://localhost:1527/IS2560";
//ConnectionURL, username and password should be specified in getConnection()
try {
    Connection conn = DriverManager.getConnection(connectionURL, "IS2560", "IS2560");
    String sql = "UPDATE IS2560.DEMO SET NAME='demo2_new' WHERE ID=2";
    Statement st = conn.createStatement();
              st.executeUpdate(sql);
               st.close();
               conn.close();
} catch (SQLException ex) {
    System.out.println("Connect failed ! ");
}
```

Figure 24: Test SQL UPDATE to modify existing records

5. Packaging an servlet assignment for IS2560

a. Copy the DBMS folder

1) To **locate your Java DB file location**, please click “Services” tab
2) Expand “Database” and select “Java DB”
3) Right click the “Java DB” and select “properties”
4) The “Java DB properties” window (Figure 25) provides the Database Location. In our case (in **Windows**), it is located in “C:\Users\username\.netbeans-derby”. If you are in **Mac**, the location should be different.
5) Navigate to this folder and copy the folder named “IS2560”, or the database name you chose for your project.
6) **Note**: In both Mac and Windows, if you cannot find this folder, mostly, it is because this folder is hidden in your computer. You can use this link for view the hidden files:
   http://guides.macrumors.com/Viewing_hidden_files_on_a_Mac (Mac)
   http://windows.microsoft.com/en-us/windows/show-hidden-files#show-hidden-files=windows-7 (Windows)



Java DB Properties

Specify the folder where Java DB is installed and the folder where you will keep your databases. The database location folder will be used as the value of the derby.system.home property.

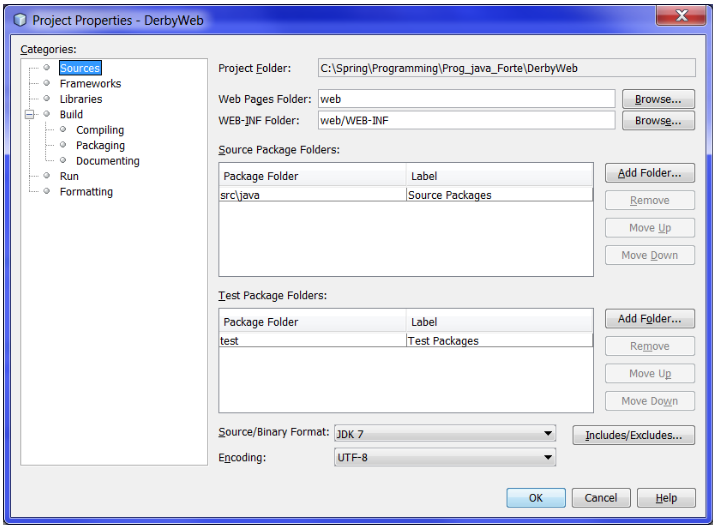
Java DB Installation: C:\Program Files\Java\jdk1.7.0_40\db

Database Location: C:\Users\shuguang\.netbeans-derby

Figure 25: Locate Java DB database location

b. Zip the files

7) Click “Projects” tab, and select your homework project. In our example, we select “DerbyWeb”.
8) Right click on “DerbyWeb”, select “Properties” to see where your project is located.
9) Go to the location of your project, create a folder called “DBMS”.
10) Go to that folder and paste your DMBS
11) Navigate back to the main project location
12) Zip your project files with database files. This is what we expect you to submit.
13)



Project Folder: C:\Spring\Programming\Prog_java_Forte\DerbyWeb

Web Pages Folder: web

WEB-INF Folder: web/WEB-INF

Source Package Folders:

Package Folder	Label
src\java	Source Packages

Test Package Folders:

Package Folder	Label
test	Test Packages

Source/Binary Format: JDK 7

Encoding: UTF-8

Figure 26: Locate Java project location

6. A little more on DMBS connections

Mysql and Access are other two commonly used databases. In this section, we introduce how to connect with these two databases using Java code.

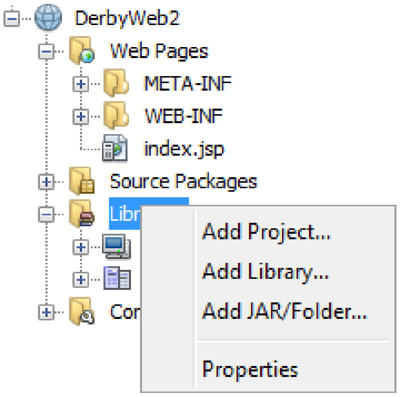
a. Installing Mysql

To use Mysql, you must install a Mysql Server and register it with netbeans. This step is not included here, but you can find online help at:

- Installing Mysql: https://netbeans.org/kb/docs/ide/install-and-configure-mysql-server.html
- Registering Mysql: https://netbeans.org/kb/docs/ide/mysql.html

b. Connecting Mysql

1) In order to use Mysql in web applications, we need to add Mysql JDBC drivers to our project library. Click “Projects” tab and select your homework projects
2) Right click on “Libraries”, and select “Add Library” (Figure 27)
3) In the “Add library” window, select “Mysql JDBC Driver” (Figure 28).



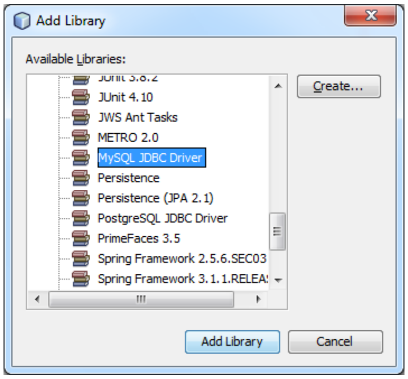


Figure 27: The new project window

Figure 28: Add Drivers

4) The following Java codes (Figure 29) allow you to connect the Mysql database. If the console printed connection failed, it may because the Mysql server is stopped.
5) The selecting, adding and modifying data using Mysql database are the same as using Derby after the connection was built. Please refer to Figure 22-24.

```
String url = "jdbc:mysql://localhost:3306/";
 String dbName = "IS2560";
 String driver = "com.mysql.jdbc.Driver";
 String userName = "IS2560";
 String password = "IS2560";
 try {
    Class.forName(driver).newInstance();
    Connection conn = DriverManager.getConnection(url+dbName,userName,password);
    System.out.println("Connected to the database");
    //You can add some functions here
    conn.close();
  } catch (Exception se)
 {
        System.out.println(“Connect failed !”);
 }
```

Figure 29: Test Derby Connections

c. Connecting to Access

6) Different from Derby and Mysql, JDK provided the database connection drivers for Access. There is no need to add external drivers.

7) The following Java codes (Figure 30) help to connect the Access database.

```
String dbName = "IS2560";
String driver = " sun.jdbc.odbc.JdbcOdbcDriver";
String userName = "IS2560";
String password = "IS2560";
try {
   Class.forName(driver).newInstance();
   Connection conn = DriverManager.getConnection(“jdbc:odbc:”+ dbName,userName,password);
   System.out.println("Connected to the database");
   //You can add some functions here
   conn.close();
 } catch (Exception se)
{
       System.out.println(“Connect failed !”);
}
```

Figure 30: Test Derby Connections

7. More on Backup and Restore in Windows for Derby

a. Copy Data Folders (See section 4)

b. Using Java Code

1) For adding Derby drivers, please refer to section 3.
2) The codes displayed in Figure 31 implemented the Derby database backup function. To **backup** database “IS2560”, set variable backup=true and run the program. A folder named “IS2560” will appear on your folder: “C:/dbbackups/”.

```
    boolean backup = true;
    Connection conn = null;
    if(backup)
    {
      //jdbc:derby://localhost:1527/test is the database connection,
      //test is the database name
      String connectionURL = "jdbc:derby://localhost:1527/ IS2560";

      //username and password should be specified in getConnection()
      //test, test are username and password in this case
      conn = DriverManager.getConnection(connectionURL, " IS2560", " IS2560");
      String sqlstmt = "CALL SYSCS_UTIL.SYSCS_BACKUP_DATABASE(?)";
      CallableStatement cs = conn.prepareCall(sqlstmt);

      //backup the database to C:/dbbackups/
      cs.setString(1,"C:/dbbackups/");
      cs.execute();
      cs.close();
      conn.close();
     }else{
      String dbURL = "jdbc:derby://localhost:1527/ IS2560;restoreFrom=C:/dbbackups/ IS2560";
      conn = DriverManager.getConnection(dbURL, " IS2560", " IS2560");
      conn.close();
     }
  }
}
```

Figure 31: Backup and Restore database.

3) To simulate the restore process, we delete the database “IS2560”: Expand “JavaDB” and select “IS2560” Right click “delete”.
4) To **restore** database “IS2560”, set variable backup=false and run the codes. Close and Reopen NetBeans, you will see “IS2560” database in your Java DB.

c. Using dblook and ij tool

There are several available derby tools for operating derby database, among which the dblook and ij are two important tools. There are three different ways of using dblook and ij, as provided in the following link. In this document, we will demonstrate the second method.

https://db.apache.org/derby/docs/10.8/getstart/tgsrunningdblook.html

The dblook can be used to dump and restore the database scheme (not data files)and ij tool to export and import data in tables (no scheme).

1) Locate the derbyrun.jar file. In NetBeans, click the “Services” tab, right click the Java DB and choose “Properties…” The Java DB Installation path suggests derby installed in “C:\Program Files\Java\jdk1.7.0_40\db”. To run **dblook** and **ij** tools, we need to run the jar files located in the folder “C:\Program Files\Java\jdk1.7.0_40\db\lib\derbyrun.jar”
2) To test the derby **dblook** and derby **ij**, we create a temporary folder in “C:\” named “derbytest”. To test on it, we should open the command line, and navigate to “C:\derbytest”: click the Start, run ”cmd” and use “cd” command to redirect to “C:\derbytest\”

For data backup

3) Run dblook: (it will dump sql scheme files to the folder “C:\derbytest\”)

```
C:\derbytest>java -jar "C:\Program Files\Java\jdk1.7.0_40\db\lib\derbyrun.jar" dblook –d
"jdbc:derby://localhost:1527/test;user=test;password=test" -o ‘C:\derbytest\my.sql’
```

4) Run ij: (we want to dump the table “demo” in the database “test”, and it will be dumped to demo.sql)

```
C:\derbytest>java -jar "C:\Program Files\Java\jdk1.7.0_40\db\lib\derbyrun.jar" ij
ij> connect ‘jdbc:derby://localhost:1527/test;user=test;password=test’;
ij> CALL SYSCS_UTIL.SYSCS_EXPORT_TABLE('TEST', 'DEMO', 'C:\derbytest\demo.sql',
null, null, 'UTF-8');
```

For data restore (delete the database before restore)

5) Restore data scheme using ij tool

```
C:\derbytest>java -jar "C:\Program Files\Java\jdk1.7.0_40\db\lib\derbyrun.jar" ij
ij> connect 'jdbc:derby://localhost:1527/test;user=test;password=test;create=true’;
ij> run ‘C:\derbytest\my.sql’;
```

6) Restore data using ij tool

```
ij> connect ‘jdbc:derby://localhost:1527/test;user=test;password=test’;
ij> CALL SYSCS_UTIL.SYSCS_IMPORT_TABLE('TEST','DEMO','C:\derbytest\demo.sql',nul
l,null,'utf-8',0);
```

Note: “SYSCS_UTIL.SYSCS_IMPORT_TABLE” and “SYSCS_UTIL.SYSCS_EXPORT_TABLE” can be found here: http://db.apache.org/derby/docs/10.0/manuals/reference/sqlj02.html#ToC

8. More on Backup and Restore in Mac for Derby

a. Copy Data Folders (See section 4)

b. Using Java Code

1) For adding Derby drivers, please refer to section 3.
2) The codes displayed in Figure 32 implemented the Derby database backup function. To backup database “test”, set variable backup=true and run the program. A folder named “IS2560” will appear on your folder: “/Users/Your Computer Name/ dbbackups”.

```
        boolean backup = true;
        Connection conn = null;
        if(backup)
        {
            //jdbc:derby://localhost:1527/test is the database connection,
            //test is the database name
            String connectionURL = "jdbc:derby://localhost:1527/ IS2560";

            //username and password should be specified in getConnection()
            //test, test are username and password in this case
            conn = DriverManager.getConnection(connectionURL, " IS2560", " IS2560");
            String sqlstmt = "CALL SYSCS_UTIL.SYSCS_BACKUP_DATABASE(?)";
            CallableStatement cs = conn.prepareCall(sqlstmt);

            //backup the database to ," /Users/ComputerName/dbbackups/
            cs.setString(1," ," /Users/ComputerName/dbbackups/");
            cs.execute();
            cs.close();
            conn.close();
         }else{
            String dbURL = "jdbc:derby://localhost:1527/
IS2560;restoreFrom=/Users/ComputerName/dbbackups/IS2560";
            conn = DriverManager.getConnection(dbURL, "test", "test");
            conn.close();
         }
    }
}
```

Figure 32: Backup and Restore database.

3) To simulate the restore process, we delete the database “IS2560”: Expand “JavaDB” and select “IS2560” Right click “delete”.
4) To restore database “IS2560”, set variable backup=false and run the codes. Close and Reopen NetBeans, you will see “IS2560” database in your Java DB.

c. Using dblook and ij tool

There are several available derby tools for operating derby database, among which the dblook and ij are two important tools. There are three different ways of using dblook and ij, as provided in the following link. In this document, we will demonstrate the second method.

https://db.apache.org/derby/docs/10.8/getstart/tgsrunningdblook.html

The dblook can be used to dump and restore the database scheme (not data files)and ij tool to export and import data in tables (no scheme).

5) Locate the derbyrun.jar file. In NetBeans, click the “Services” tab, right click the Java DB and choose “Properties…” The Java DB Installation path suggests derby installed in “/Applications/NetBeans/glassfish-4.0/javadb”. To run **dblook** and **ij** tools, we need to run the jar files located in the folder “/Applications/NetBeans/glassfish-4.0/javadb/lib/derbyrun.jar”
6) To test the derby **dblook** and derby **ij**, we create a temporary folder in “/Users/Your computer name/” named “derbytest”. To test on it, we should open the terminal (Apple Finder → Applications → Utilities → terminal), and navigate to “/Users/Your computer name/derbytest”: run “cd” command to redirect to “derbytest”.

For data backup

7) Run dblook: (it will dump sql scheme files to the folder “Users/Your computer name/derbytest”)

```
derbytest> java -jar "/Applications/NetBeans/glassfish-4.0/javadb/lib/derbyrun.jar" dblook -d "jdbc:derby://localhost:1527/test;user=test;password=test" -o ‘/Users/ computer name /derbytest /my.sql’
```

8) Run ij: (we want to dump the table “demo” in the database “test”, and it will be dumped to demo.sql)

```
derbytest>java -jar "/Applications/NetBeans/glassfish-4.0/javadb/lib/derbyrun.jar " ij
ij> connect ‘jdbc:derby://localhost:1527/test;user=test;password=test’;
ij> CALL SYSCS_UTIL.SYSCS_EXPORT_TABLE('TEST', 'DEMO', ‘Users/Your computer name/derbytest/demo.sql', null, null, 'UTF-8');
```

For data restore (delete the database as we showed above for demonstration)

9) Restore data scheme using ij tool

```
derbytest> java -jar "/Applications/NetBeans/glassfish-4.0/javadb/lib/derbyrun.jar " ij
ij> connect 'jdbc:derby://localhost:1527/test;user=test;password=test;create=true’;
ij> run ‘/Users/ computer name /derbytest /my.sql’;
```

10) Restore data using ij tool

```
ij> connect ‘jdbc:derby://localhost:1527/test;user=test;password=test’;
ij> CALL SYSCS_UTIL.SYSCS_IMPORT_TABLE('TEST','DEMO', ‘Users/Your computer name/derbytest/demo.sql',null,null,'utf-8',0);
```

Note that: “SYSCS_UTIL.SYSCS_IMPORT_TABLE” and “SYSCS_UTIL.SYSCS_EXPORT_TABLE” can be found here: http://db.apache.org/derby/docs/10.0/manuals/reference/sqlj02.html#ToC