Project 1.C

I. Project Description

Figure 1 shows an ER diagram for University database. This is the same in Project 1.A and 1.B.

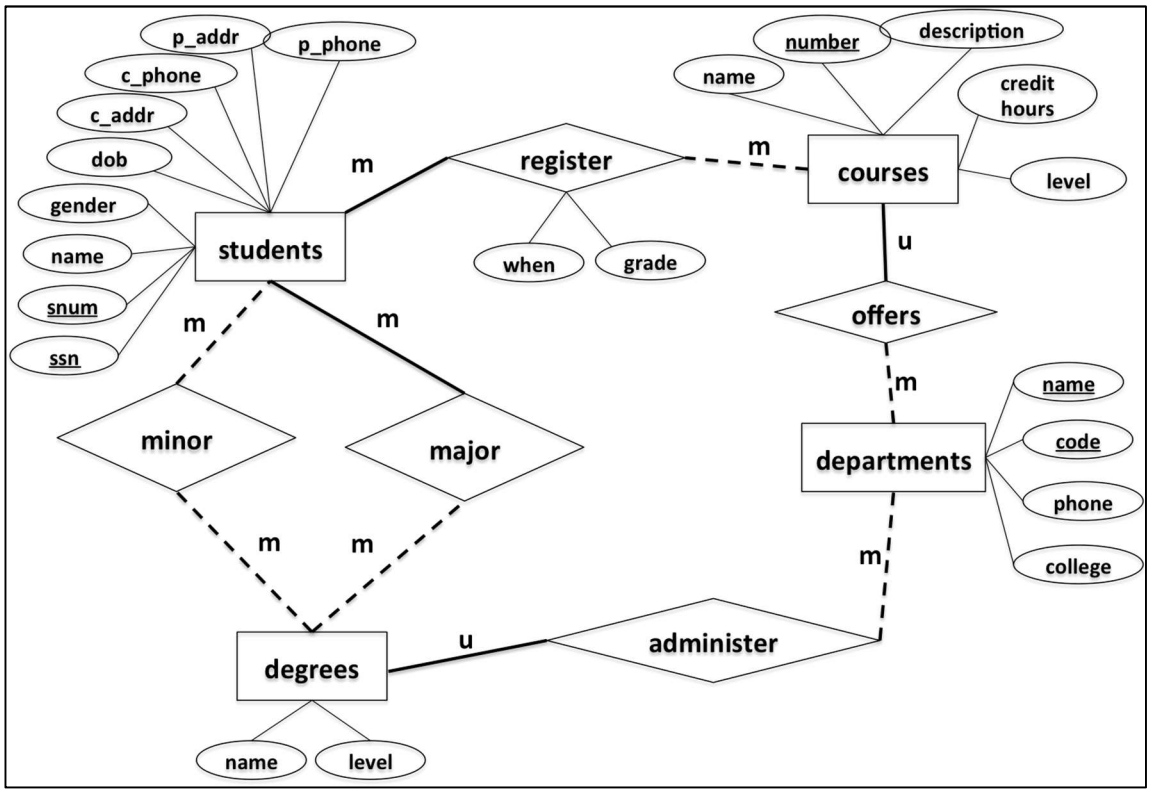


Figure 1. ER-diagram

This project is to implement the above design using a relational data model. Specifically, you are asked to write the following JavaServer Pages (JSP).

1. CreateTables.jsp and CreateTablesResult.jsp [Points: 15]

CreateTables.jsp

This web page has one submit button (See Figure below). When clicking the button, it creates the following tables and redirect to CreateTablesResult.jsp. Those seven tables are:

- students
  a) Attribute, type and length: *snum: integer, ssn: integer, name: varchar(10), gender: varchar(1), dob: datetime, c_addr: varchar(20), c_phone: varchar(10), p_addr: varchar(20), p_phone: varchar(10)*
  b) Primary key: *snum*
  c) Candidate key: *ssn*

d) Foreign key: *N/A*

- departments
  a) Attribute, type and length: *code: integer, name: varchar(50), phone: varchar(10), college: varchar(20)*
  b) Primary key: *code*
  c) Candidate key: *name*
  d) Foreign key: *N/A*
- degrees
  a) Attribute, type and length: *name: varchar(50), level: varchar(5),* department_code: integer
  b) Primary key: *name, level*
  c) Candidate key: *N/A*
  d) Foreign key: *department_code refers to code in table departments*
- courses
  a) Attribute, type and length: *number: integer, name: varchar(50), description: varchar(50), credithours: integer, level: varchar(20), department_code: integer*
  b) Primary key: *number*
  c) Candidate key: *name*
  d) Foreign key: *department_code refers to code in table departments*
- register
  a) Attribute, type and length: *snum: integer, course_number: integer, when: varchar(20), grade: integer*
  b) Primary key: *snum, course_number*
  c) Candidate key: *N/A*
  d) Foreign key: *snum refers to snum in table students, course_number refers to number in table courses*
- major
  a) Attribute, type and length: *snum: integer, name: varchar(50), level: varchar(5)*
  b) Primary key: *snum, name, level*
  c) Candidate key: *N/A*
  d) Foreign key: *snum refers to snum in table students, name & level refer to name & level in table degrees*
- minor
  a) Attribute, type and length: *snum: integer, name: varchar(50), level: varchar(5)*
  b) Primary key: *snum, name, level*
  c) Candidate key: *N/A*
  d) Foreign key: *snum refers to snum in table students, name & level refer to name & level in table degrees*

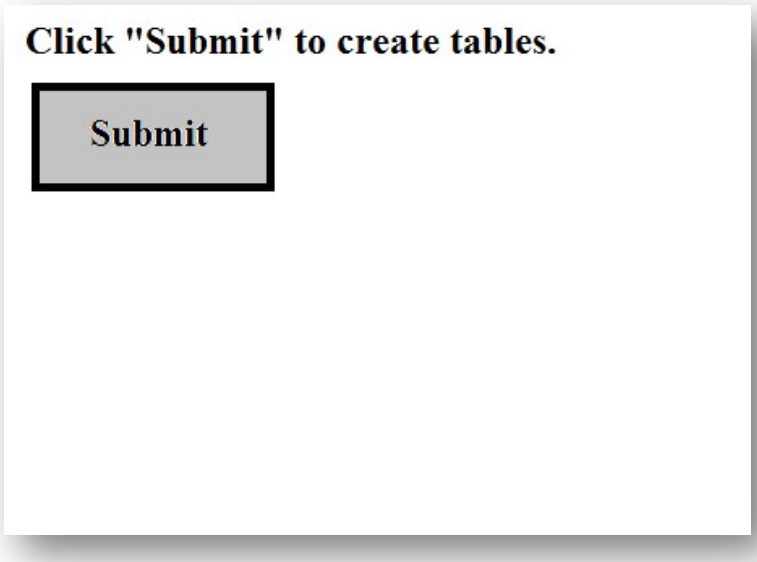


Figure 2: Example of CreateTables.jsp

CreateTablesResult.jsp

This page will display a result of tables creation (See Figure below).

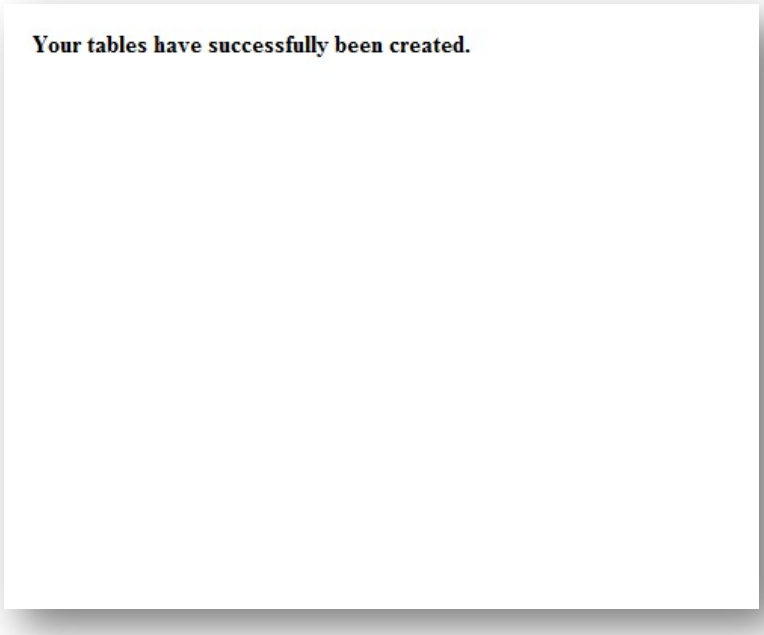


Figure 3: Example of CreateTablesResult.jsp

2. InsertRecords.jsp and InsertRecordsResult.jsp [Points: 15]

InsertRecords.jsp

A web page has one submit button (See Figure below). When clicking the button, your web page must insert data records based on entered SQL syntax. Finally, it will redirect the user to InsertRecordsResult.jsp. Those data records are:

- **students**

snum	ssn	name	gender	dob	c_addr	c_phone	p_addr	p_phone
1001	654651234	Randy	M	2000/12/01	301 E Hall	5152700988	121 Main	7083066321
1002	123097834	Victor	M	2000/05/06	270 W Hall	5151234578	702 Walnut	7080366333
1003	978012431	John	M	1999/07/08	201 W Hall	5154132805	888 University	5152012011
1004	746897816	Seth	M	1998/12/30	199 N Hall	5158891504	21 Green	5154132907
1005	186032894	Nicole	F	2001/04/01	178 S Hall	5158891155	13 Gray	5157162071
1006	534218752	Becky	F	2001/05/16	12 N Hall	5157083698	189 Clark	2034367632
1007	432609519	Kevin	M	2000/08/12	75 E Hall	5157082497	89 National	7182340772

- **departments**

code	name	phone	college
401	Computer Science	5152982801	LAS
402	Mathematics	5152982802	LAS
403	Chemical Engineering	5152982803	Engineering
404	Landscape Architect	5152982804	Design

- **degrees**

name	level	department_code
Computer Science	BS	401
Software Engineering	BS	401
Computer Science	MS	401
Computer Science	PhD	401
Applied Mathematics	MS	402
Chemical Engineering	BS	403
Landscape Architect	BS	404

- **major**

snum	name	level
1001	Computer Science	BS
1002	Software Engineering	BS
1003	Chemical Engineering	BS
1004	Landscape Architect	BS
1005	Computer Science	MS
1006	Applied Mathematics	MS
1007	Computer Science	PhD

- **minor**

snum	name	level
1007	Applied Mathematics	MS
1005	Applied Mathematics	MS
1001	Software Engineering	BS

- **courses**

number	name	description	credithours	level	department_code
113	Spreedsheet	Microsoft Excel and Access	3	Undergraduate	401
311	Algorithm	Design and Analysis	3	Undergraduate	401
531	Theory of Computation	Theorem and Probability	3	Graduate	401
363	Database	Design Principle	3	Undergraduate	401
412	Water Management	Water Management	3	Undergraduate	404
228	Special Topics	Interesting Topics about CE	3	Undergraduate	403
101	Calculus	Limit and Derivative	4	Undergraduate	402

- **register**

snum	course_number	when	grade
1001	363	Fall2015	3
1002	311	Fall2015	4
1003	228	Fall2015	4
1004	363	Spring2015	3
1005	531	Sprng2015	4
1006	363	Fall2015	3
1007	531	Spring2015	4

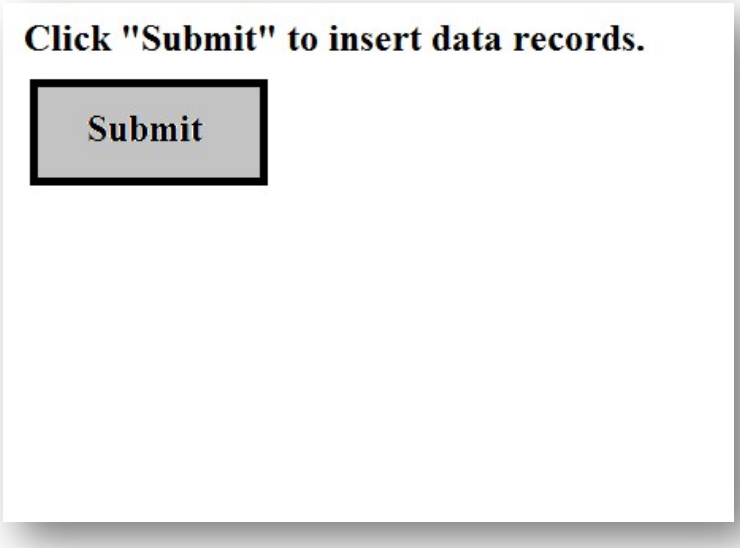


Figure 4: Example of InsertRecords.jsp

InsertRecordsResult.jsp

This page will display a result of records insertion (See Figure below).

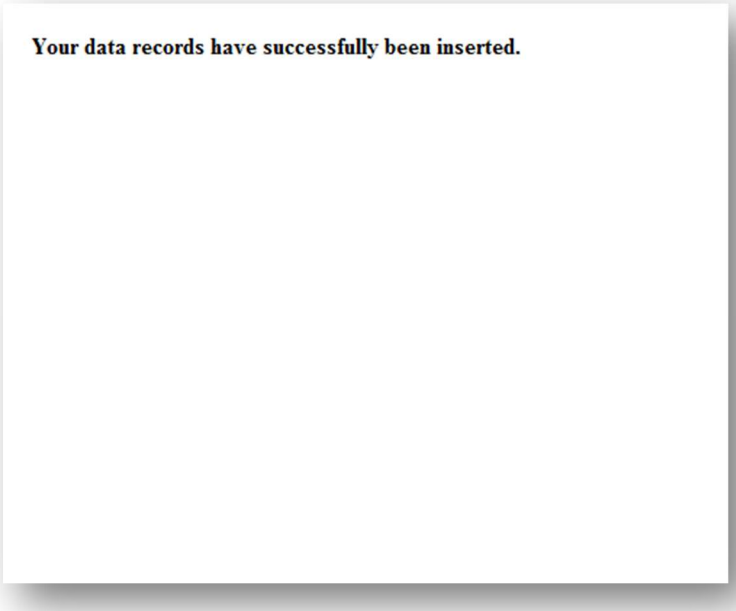


Figure 5: Example of InsertRecordsResult.jsp

3. Query.jsp, QueryResult1.jsp, QueryResult2.jsp, QueryResult3.jsp [Points: 55]

Query.jsp

A web page has **THREE** buttons (See Figure below). When clicking any button, your web page must query data records based on corresponding query. Finally, it will redirect the user to either QueryResult1.jsp or QueryResult2.jsp or QueryResult3.jsp based on corresponding query to show query results. Those three queries are:

1) The student number and ssn of the student whose name is "Becky"
2) All degree names and levels offered by the department of Computer Science
3) The course numbers and names of all courses offered by either Department of Computer Science or Department of Landscape Architect.

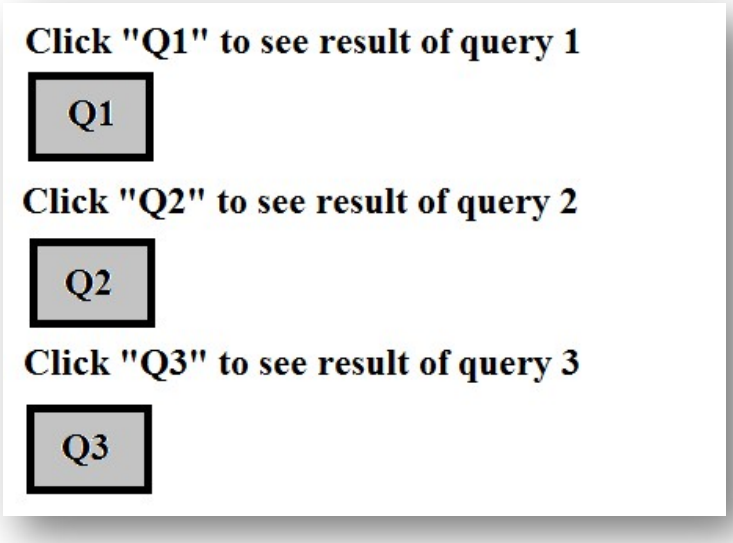


Figure 6: Example of Query.jsp

QueryResult1.jsp, QueryResult2.jsp, QueryResult3.jsp

This page will display a result of corresponding query (See Figure below).

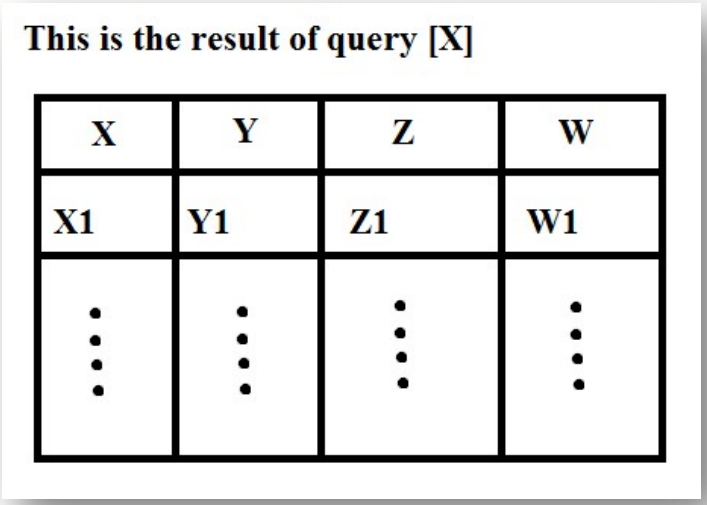


Figure 7: Example of QueryResultX.jsp

4. ModifyRecords.jsp and ModifyRecordsResult.jsp [10]

ModifyRecords.jsp

A web page has one submit button (See Figure below). When clicking, your web page must modify a data record. Finally, it will redirect the user to ModifyRecordsResult.jsp. This is a modification:

1) Change the name of the student with ssn = 746897816 to Scott

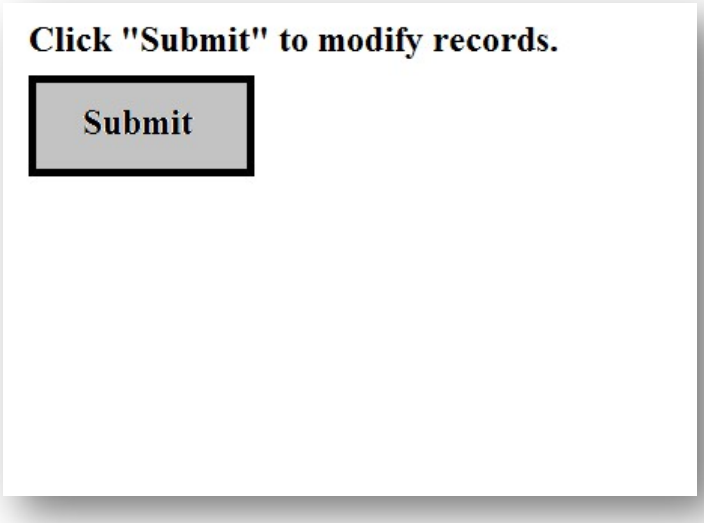


Figure 8: Example of ModifyRecords.jsp

ModifyRecordsResult.jsp

This page will display a result of record modification (See Figure below).

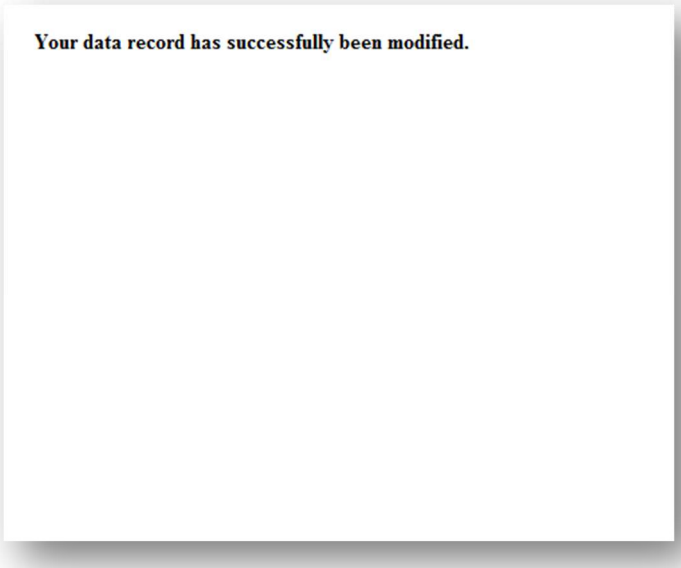


Figure 9: Example of ModifyRecordsResult.jsp

Submission Instruction

***U**pload all JSP files (CreateTables.jsp, CreateTablesResult.jsp, InsertRecords.jsp, InsertRecordsResult.jsp, Query.jsp, QueryResult1.jsp, QueryResult2.jsp, QueryResult3.jsp, ModifyRecords.jsp, and ModifyRecordsResult.jsp) to your account in Canvas.*

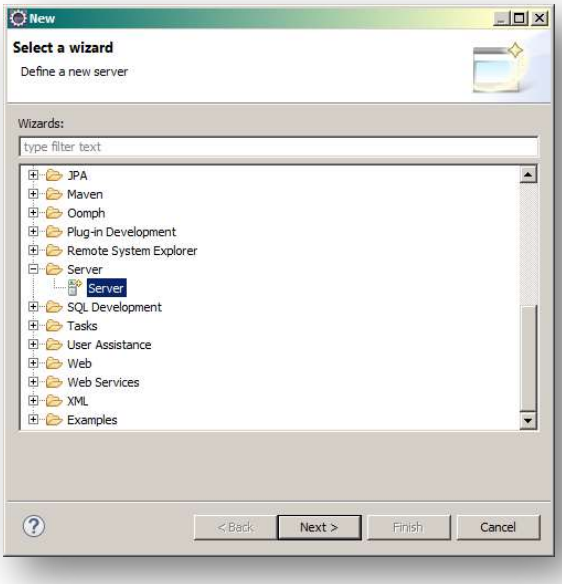
II. Set up a local working environment using Eclipse

SSG is supposed to set up a working environment for us. However, they are not helping anything yet. So let's set up a local working environment. Specifically, you follow these steps.

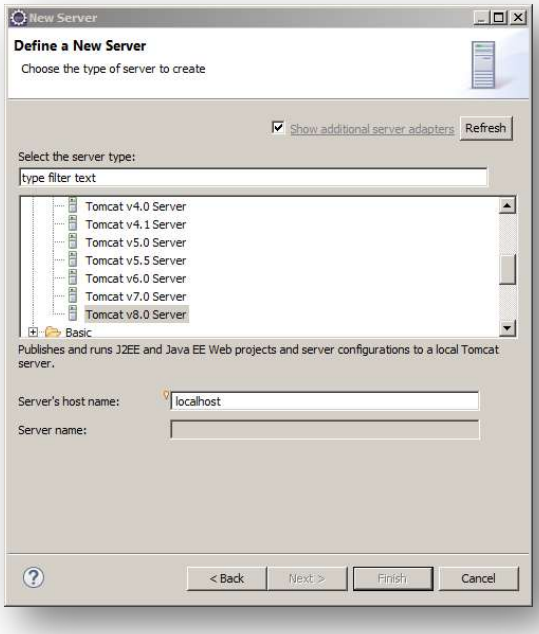
1. Make sure that you have Java JDK installed in your computer, if not, you can get Java JDK at http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Download and install Eclipse IDE at https://eclipse.org/downloads/ (choose "Eclipse IDE for Java EE Developers")
3. Download and install Connector J at https://dev.mysql.com/downloads/connector/j/
4. Download Tomcat Server (Zipped file) at http://tomcat.apache.org/download-80.cgi and unzip it to your preferred directory.

III. Set up Tomcat Server and Create Dynamic Web Project in Eclipse

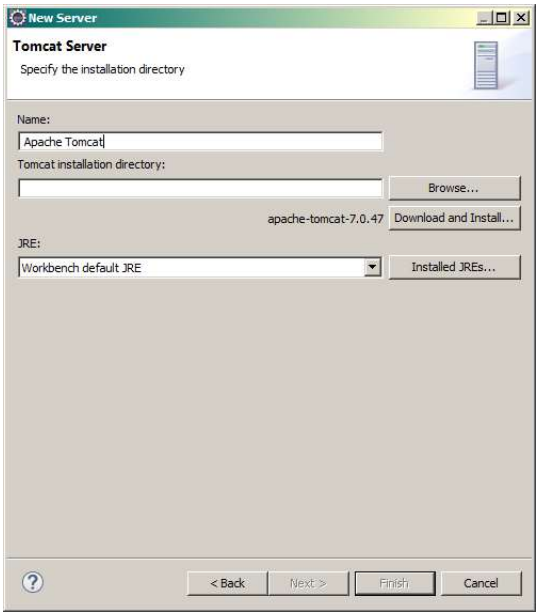
1. Create new Server in Eclipse by go to “New > Server > Server”, then, click “Next”.



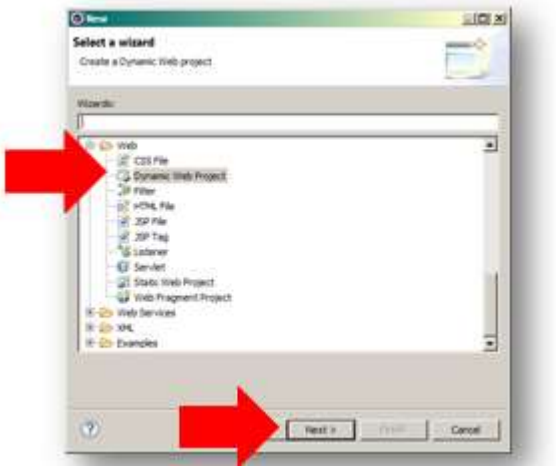
2. Select “Tomcat v8.0 Server”, then, click “Next”.



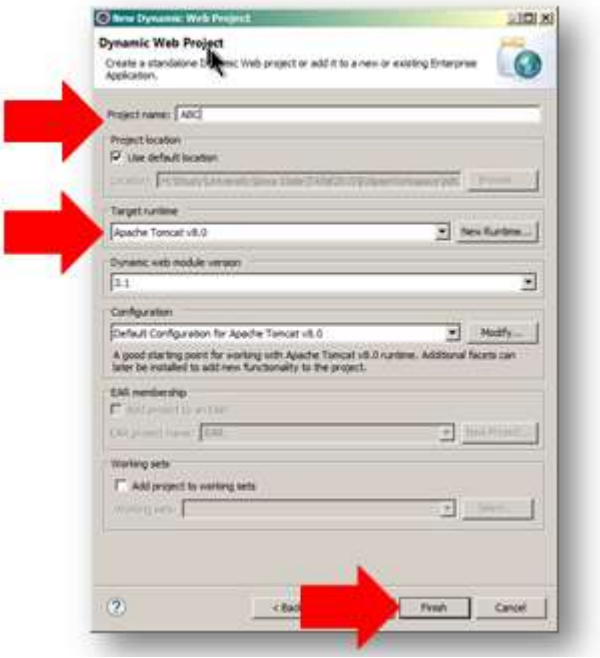
3. Click “Browse…”, then, select the directory that you unzipped Apache Tomcat Server. Click “Finish”.



4. Create new Dynamic Web Project by go to “File > New > Other”, then select Dynamic Web Project under “Web” folder.

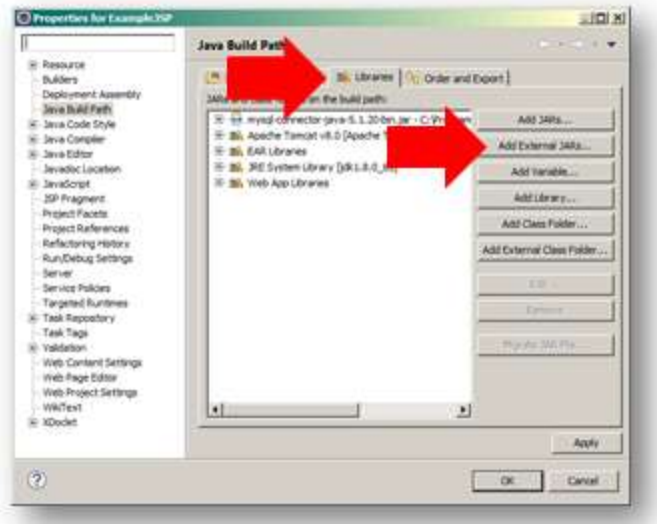


5. Specify Project name and Target runtime (Tomcat server that installed in your Eclipse at step 3). Click “Finish”.

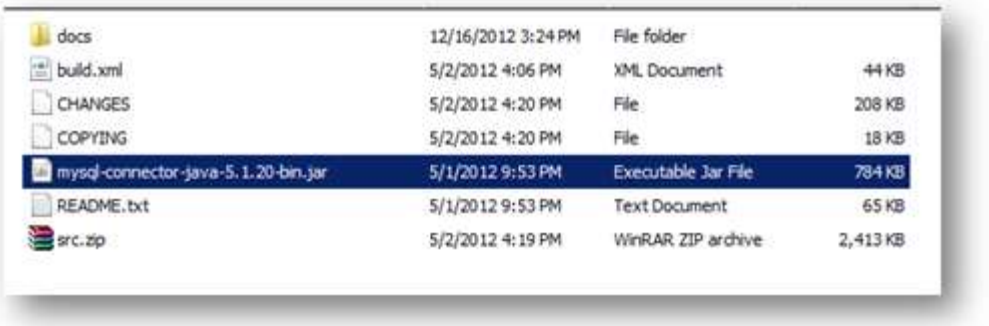


6. You will get the Project Explorer like this:

7. Add Connector J JAR file to your build path by right-click at your project and select “Build Path > Configure Build Path”.
8. Select Libraries tab and click “Add External JARs”.



9. Go to the directory that you stored Connector J JAR file and select it. Then click “OK”.
10. Next, go to that directory again and **COPY** your Connector J JAR file manually.

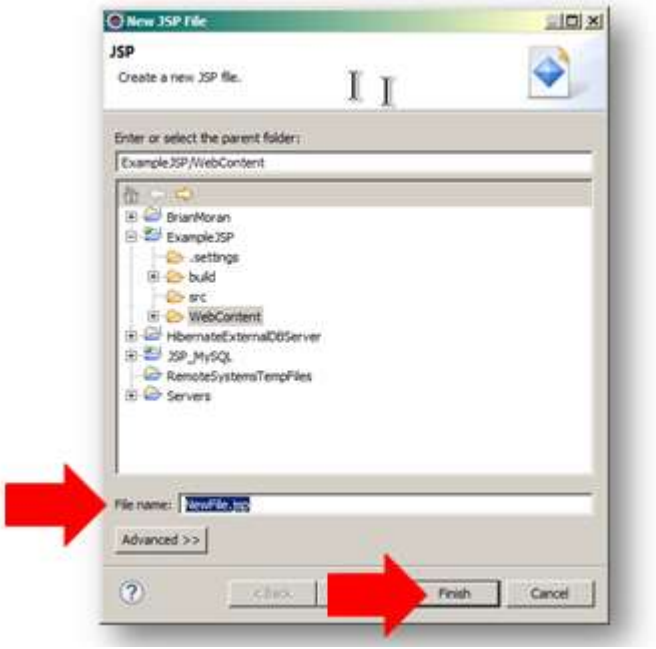


11. Go to “[Your Project Folder]\WebContent\WEB-INF\lib\” and paste Connector J JAR file to that folder.

IV. Example of JSP Codes

In order to create JSP page, follow these steps:

1. Right-click at WebContent folder (in Project Explorer) and select “WebContent > New > JSP File”.
2. Specify the name of your new JSP file and click “Finish”.



3. Next, in order to connect to database, you need to have these statements in your JSP code:
   - To import necessary libraries:

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page import="javax.servlet.http.*,javax.servlet.*" %>
```

   - To establish a JDBC connection:

```
//Establishing a connection
String connectionURL = "jdbc:mysql://[Server Name]/[DB Name]";
Connection connection = null;
Class.forName("com.mysql.jdbc.Driver").newInstance();
connection = DriverManager.getConnection(connectionURL,"[User
Name]","[Password]");
```

- To select data from database:

```
String allEmployeeSQL = "[Your SQL select syntax]";
Statement stm = connection.createStatement();
ResultSet rs = stm.executeQuery(allEmployeeSQL);
while(rs.next())
{
        out.println(rs.getString("[Column Name]");
}
```

- To insert/update/delete data:

```
String insertSQL = "INSERT INTO [Table Name] VALUES('" +
[column1] + "','" + [column2] + … + "')";
PreparedStatement statement =
connection.prepareStatement(insertSQL);
statement.executeUpdate();
```

- To get value from HTML form

```
String value1 = request.getParameter("[Input Name]");
String value2 = request.getParameter("[Input Name]");
```

4. Please note that JSP page consists of both HTML code and Java code. Every time you write any Java code inside JSP page, you have to wrap all your Java codes inside the scripting delimiter:

```
<% --- Your Java Code ---%>
```

V. Tutorial

1. This page shows you how to create and run simple JSP application in Eclipse: http://www.srccodes.com/p/article/2/JSP-Hello-World-Program-using-Eclipse-IDE-and-Tomcat-web-server
2. This page shows you how to use “Form” to retrieve values from any control (textbox, check box, etc.) and send those values to another JSP page: http://www.tutorialspoint.com/jsp/jsp_form_processing.htm