Vietnam National University of HCMC
International University
School of Computer Science and Engineering

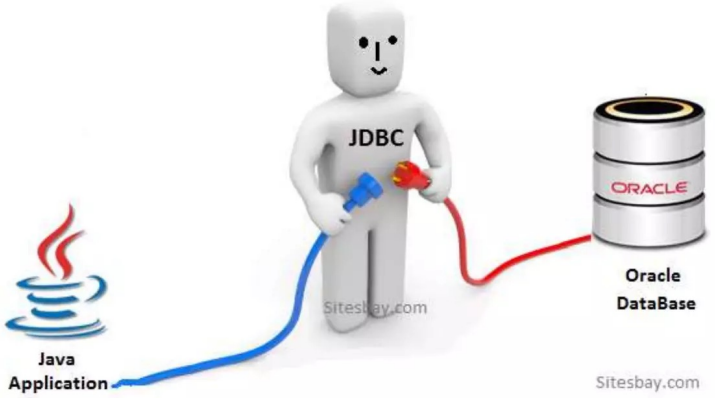
Web Application Development (IT093IU)

Assoc. Prof. Nguyen Van Sinh
Email: nvsinh@hcmiu.edu.vn

(Semester 2, 2023-2024)

Lecture: Database Connection
JAVA <-> JDBC <-> DBMS

- **Introduction to MySQL**
- **Introduction to JDBC**
- **Connect MySQL - JDBC**
- **An example for practice**



RDBMS & MySQL

- Introduction to RDBMS
- Introduction to MySQL
- Terminology
- Managing database

Introduction to RDBMS

- A relational database manages data in table
- Database are managed by a RDBMS
- An RDBMS supports a database language to create and delete database and to manage and search data
- The database language used in almost all DBMS is SQL

Introduction to MySQL

- MySQL is a DBMS
- Inexpensive, Fast, Easy to use, Portable
- MySQL provides:
  - Support for SQL: standard language for working with data that's stored in relational database
  - Support for multiple clients: Java, PhP, C#, etc.,
  - Connectivity: can provide access to data via internet/intranet
  - Security: can protect access to your data
  - Referential integrity: support referential integrity with other DBMS (Oracle or MSSQL)
  - Transaction processing

Features of MySQL

- It is non-procedural, English like language.
- SQL is a high-level language
- It is very flexible language.
- SQL allows users to define the data in database and manipulate that data.
- SQL can be run on any system.
- SQL allows users to access data in relational database management systems.
- MySQL is a freely available Database System.
- MySQL stores data in table, tables further store Data in Rows and Columns.
- MySQL takes less memory space. (800MB for Manager Minimum Disk Space and 500MB for Agent Minimum Disk Space).
- MySQL is scalable and capable of handling more than 50 million rows.
- MySQL has a unique storage engine architecture which makes it faster, cheaper and more reliable.

Functions in MySQL

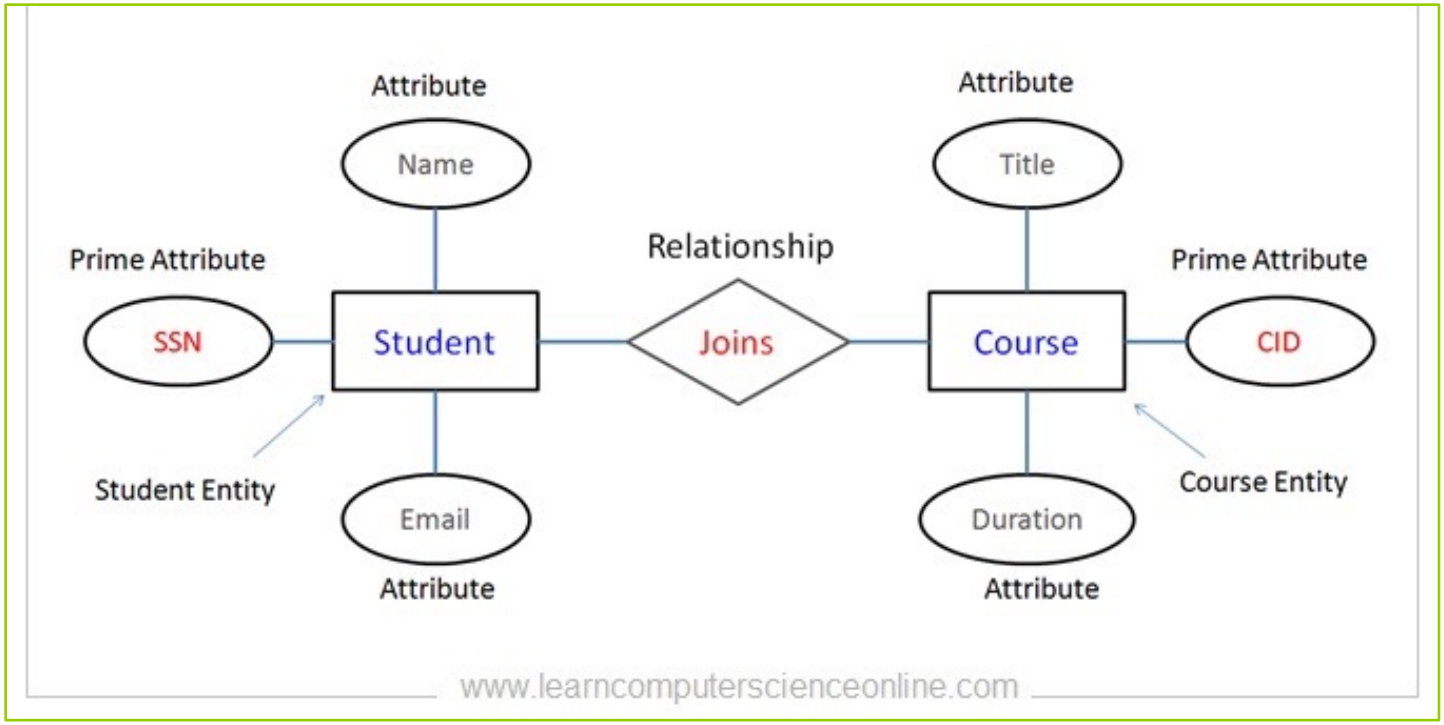
- After creating a database, the most common SQL statement used are:
  - Insert → to add data
  - Update → to change data
  - Delete → to remove data
  - Select → to search data
- A database table may have multiple columns, or attributes, each of which has a name.
- Table usually have a primary key, which is one or more values that uniquely identify each row in a table

Basic command

- CREATE TABLE Command: The CREATE TABLE statement is used to create a new table in a database.
- ALTER TABLE Command: ALTER TABLE statement is to add, delete, or modify columns in an existing table.
- DROP TABLE Command: The DROP TABLE statement is used to drop an existing table in a database.
- TRUNCATE TABLE Command: The TRUNCATE TABLE statement is used to delete the data inside a table, but not the table itself.
- RENAME TABLE Command: The RENAME TABLE statement is used to change the name of the table.
- DESC TABLE Command: DESC TABLE statement is used to describe a table.

An example in MySQL

- A database is modeled using ERD modeling to install in MySQL



Terminology

- Database
  - A repository to store data
- Table
  - The part of a database that store the data. A table has a columns or attributes and the data stored in rows
- Attributes
  - The column in a table. All rows in table entities have the same attributes. For example, a customer table might have the attributes name, address, and city. Each attribute has a data type such as string, integer, or date.

Terminology

- Rows
  - The data entries in a table. Rows contain values for each attribute. For example, a row in a customer table might contain the values "Matthew Richardson, "Punt Road," and "Richmond." Rows are also known as records.
- Relational model
  - A model that uses tables to store data and manage the relationship between tables.
- Relational database management system
  - A software system that manages data in a database and is based on the relational model.

Terminology

- Constraints
  - Restrictions or limitations on tables and attributes. For example, a wine can be produced only by one winery, an order for wine can't exist if it isn’t associated with a customer, having a name attribute could be mandatory for a customer.
- Primary key
- Foreign key
- Index

Introduction to JDBC

- **JDBC** is a Java API for communicating with database systems supporting SQL.
- JDBC supports a variety of features for querying and updating data, and for retrieving query results.
- JDBC also supports metadata retrieval, such as querying about relations present in the database and the names and types of relation attributes.
- Model for communicating with the database:
  - Open a connection
  - Create a "statement" object
  - Execute queries using the statement object to send queries and fetch results
  - Exception mechanism to handle errors

Introduction to JDBC

- The JDBC library includes APIs for each of the tasks mentioned below that are commonly associated with database usage.
- Making a connection to a database.
- Creating SQL or MySQL statements.
- Executing SQL or MySQL queries in the database.
- Viewing & Modifying the resulting records.

Connection MySQL <-> JDBC

There are six steps involved in building a JDBC application:

Import the packages:

- This requires that you include the packages containing the JDBC classes needed for database programming.
- Most often, using import java.sql.* will suffice as follows:
- **//STEP 1. Import required packages:** import java.sql.*;

Register the JDBC driver:

- This requires that you initialize a driver so you can open a communications channel with the database.
- **//STEP 2: Register JDBC driver**
- Class.forName("com.mysql.jdbc.Driver");

Open a connection

This requires using the DriverManager.getConnection() method to create a Connection object, which represents a physical connection with the

database as follows:

- //**STEP 3: Open a connection**
- // Database credentials
- static final String USER = "username";
- static final String PASS = "password";
- System.out.println("Connecting to database...");
- conn = DriverManager.getConnection(DB_URL,USER,PASS);

Execute a query

- This requires using an object of type Statement or PreparedStatement for building and submitting an SQL statement to the database as follows:
- **//STEP 4: Execute a query**
- System.out.printin("Creating statement...");
- stmt = conn.createStatement0;
- String sql;
- sql = “SELECT id, first, last, age FROM Employees”;
- ResultSet rs = stmt.executeQuery(sq);
- If there is an SQL UPDATE,INSERT or DELETE statement required, then following code snippet would be required:
- **//STEP 4: Execute a query**
- System.out. printin("Creating statement...");
- stmt = conn.createStatement();
- String sq| = "DELETE FROM Employees";
- ResultSet r = stmt.executeUpdate(sql)

Extract data from result set

- This step is required in case you are fetching data from the database. You can use the appropriate ResultSet.getXXX() method to retrieve the data from the result set as follows:
- **//STEP 5: Extract data from result set**
- while(rs.next0)){
- //Retrieve by column name
- int id = rs.getInt("id");
- int age = rs.getInt("age");
- String first = rs.getString("first");
- String last = rs.getString("last");
- System.out.print("ID: " + id);
- System.out.print(", Age: " + age);
- System.out.print(", First: " + first);
- System.out.println(", Last: " + last); }

Clean up the environment

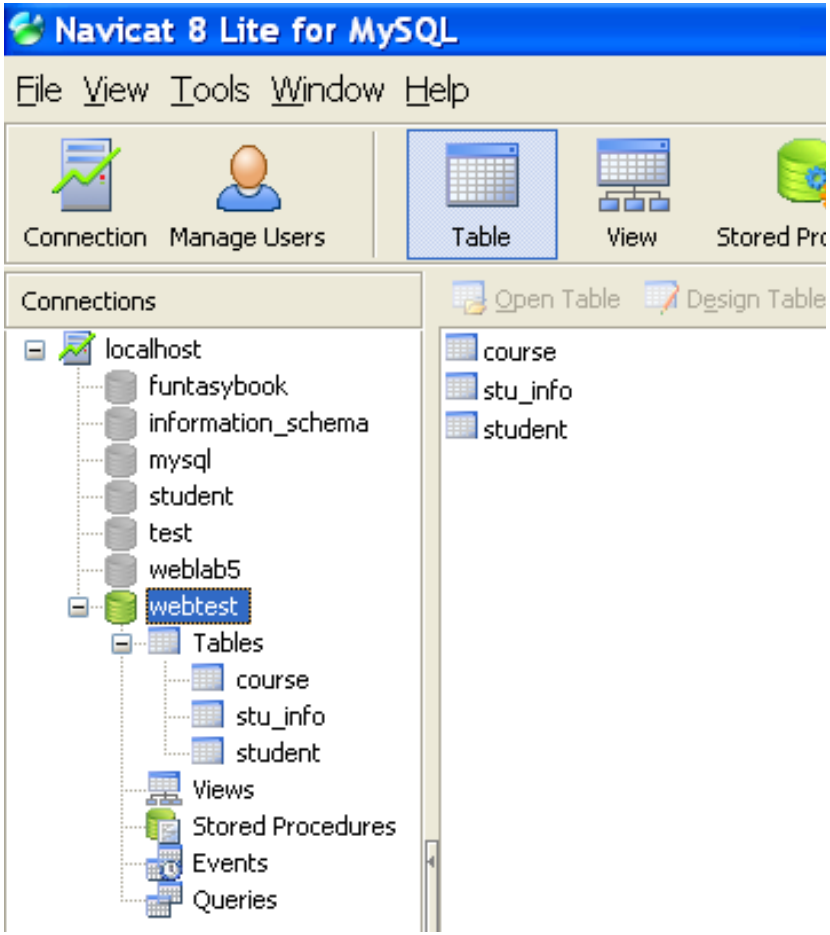
- You should explicitly close all database resources versus relying on the JVM's garbage collection as follows:
- **//STEP 6: Clean-up environment**
- rs.close(;
- stmt.close();
- conn.close();

Example for practice

- This section guide to create an example as in practice
- A small application that can be processed with 5 functions of a dynamic web:
  - Input data,
  - View data,
  - Update data,
  - Delete data
  - Search data

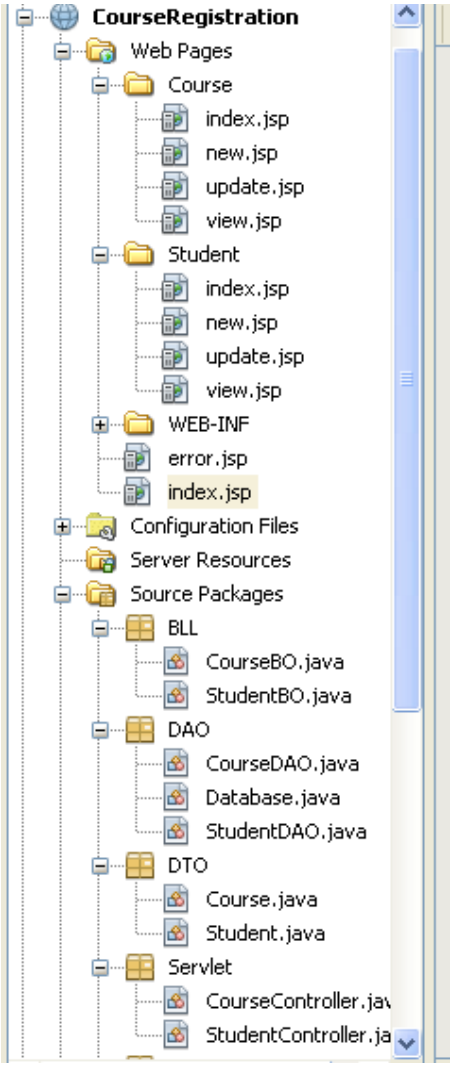
  (In detail the work in the Lab section 5)

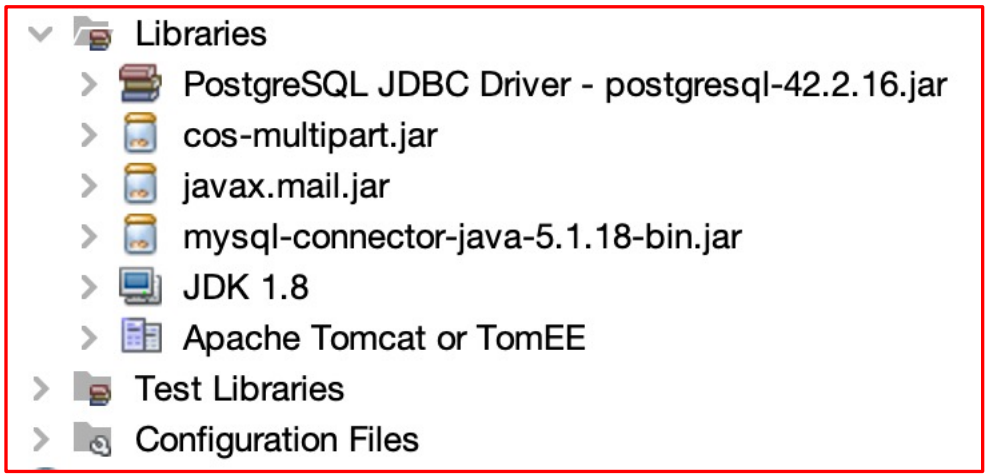
The context of application



- Create an application for student registration system (SRS)
- DBMS: MySQL; JSP and Servlet for handle
- Create a DB with 3 tables (as in Figure)
- One student can register more than one course; a course can be registered by more than one student

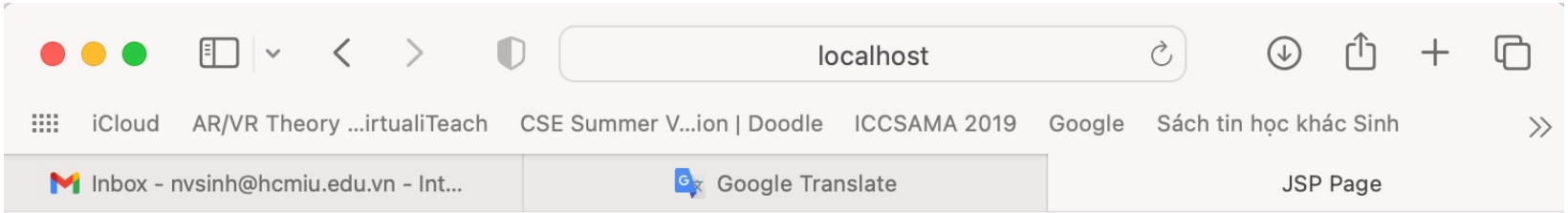
The context of application





Right-click at Libraries folder.
→ add JDBC

The UI of application



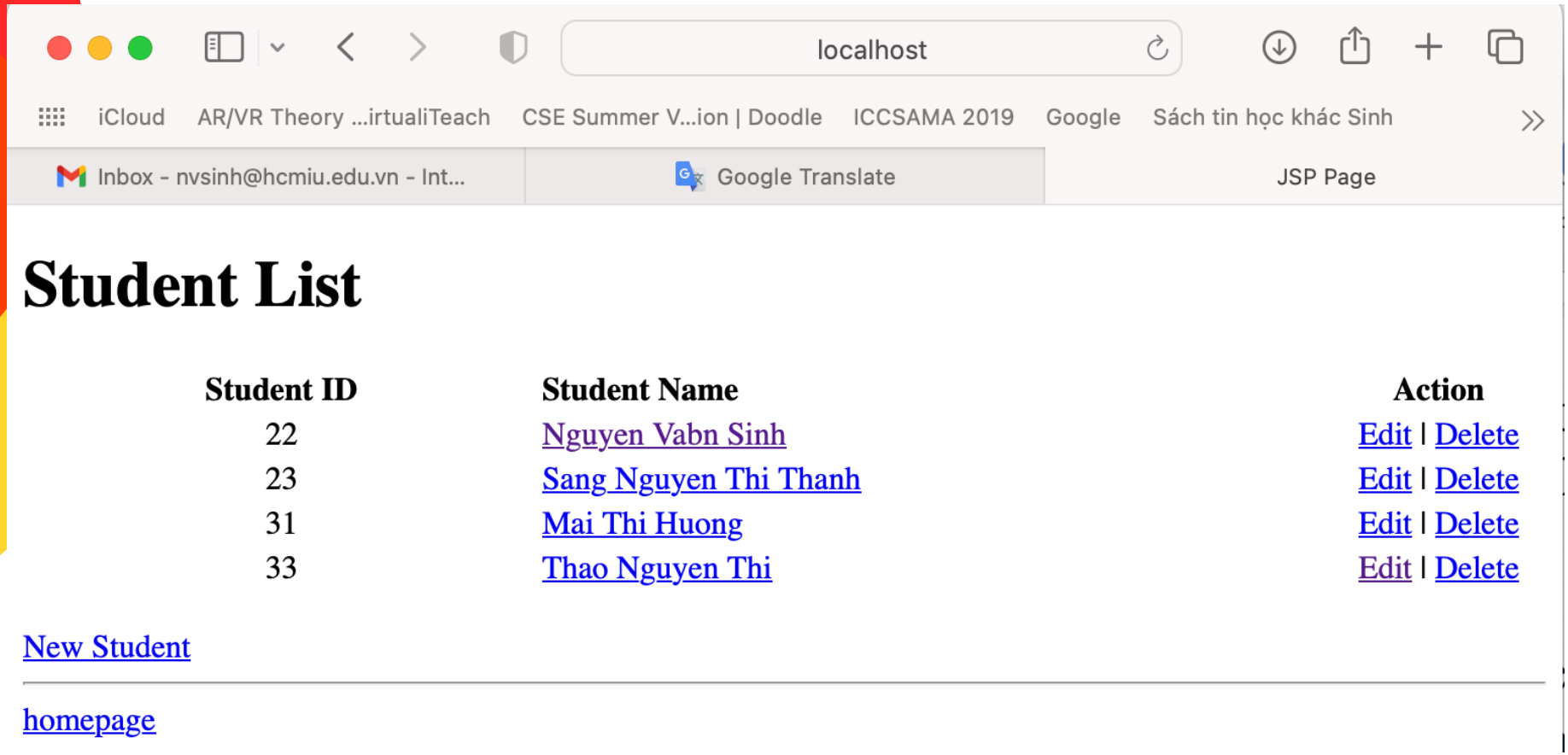
Course List

Course ID	Course Name	Action
1	Advanced Computer Network	Edit | Delete
5	database	Edit | Delete
6	C++ Programming	Edit | Delete
10	Object Oriented Programming	Edit | Delete
14	Advanced Computer Network	Edit | Delete
16	Java Programming	Edit | Delete
23	Discrete Maths	Edit | Delete
28	OOP	Edit | Delete
29	Discrete math	Edit | Delete
32	PPL	Edit | Delete

New Course

homepage

The UI of application



Step to build DB page with Java

- Build database and tables
- Build JSP or Servlet pages
- Write source code to connect from JSP to database
- Write source code to manipulate data (insert, delete, update, retrieve) in that pages

Create a database with tables

- Start mySQL by run some tools such as MyAdmin… or mysqld.exe in console window.
- Connect to database (use some accounts) by Navicat Lite. Default server running at localhost:3306
- Create a database schema
- Create tables to contain data
- Insert data into table

Build JSP or Servlet pages

- Create a JSP or servlet class as normal

Connect to database server

- Using JDBC (Java DataBase Connectivity)
  - Make reference to package jar file
    - MySQL JDBC Driver at "C:\Program Files\NetBeans 7.0\ide11\modules\ext\mysql-connector-java-5.1.6-bin.jar"
    - Or add library "MySQL JDBC Driver" to your project. This library is built-in of Netbean, but it is older version
- Follow by steps:
  - Import packages
    - Put following code to your program
      - `import java.sql.*;`
  - Put mysql package reference to your project
  - Load the driver (JDBC)
    - Code: `Class.forName("com.mysql.jdbc.Driver");`

Connect to database server (cont.)

- Define the connection URI
  - `String URI = "jdbc:mysql://localhost:3306/weblab5“;`
- Establish the connection
  - `Connection conn = DriverManager.getConnection(URI,"root", "");`

Steps to manipulate data

- Create a statement object from connection object
  - `Statement stmt = conn.createStatement();`
- Execute a query
  - `String sql = "select * from … ";`
  - `ResultSet rs = stmt.executeQuery(sql);`
  - Or `stmt.execute("insert into…");`

Code example for Login page

```
3   <%@page import="java.sql.*"%>
4   <%@page contentType="text/html" pageEncoding="UTF-8"%>
5   <!DOCTYPE html>
6   <html>
7       <head>
8           <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9           <title>Login Page</title>
10      </head>
11      <body>
12          <h1>Check Connection to Database</h1><hr>
13              <%
14              String connectionURL="jdbc:mysql://localhost:8889/weblab5?user=root;password=root"; //step 2
15              Connection connection = null;       //step 3
16              Statement statement = null; //step 4
17              ResultSet rs = null;
    %>
19  <%
20              //Class.forName("org.gjt.mm.mysql.Driver").newInstance();
21              Class.forName("com.mysql.jdbc.Driver").newInstance();       //step 1
22              connection = DriverManager.getConnection(connectionURL, "root", "root");
23              statement = connection.createStatement();
24              rs = statement.executeQuery("SELECT * FROM user");  //step 5
25
26              out.println("Your userName and passWord are:<br>");
27              while (rs.next()) {
28                  out.println(rs.getString("userName"));
29                  out.println(rs.getString("passWord")+"<br>");
30              }   //step 6
31              rs.close(); //step 7
    %>
33      </body>
34  </html>
```

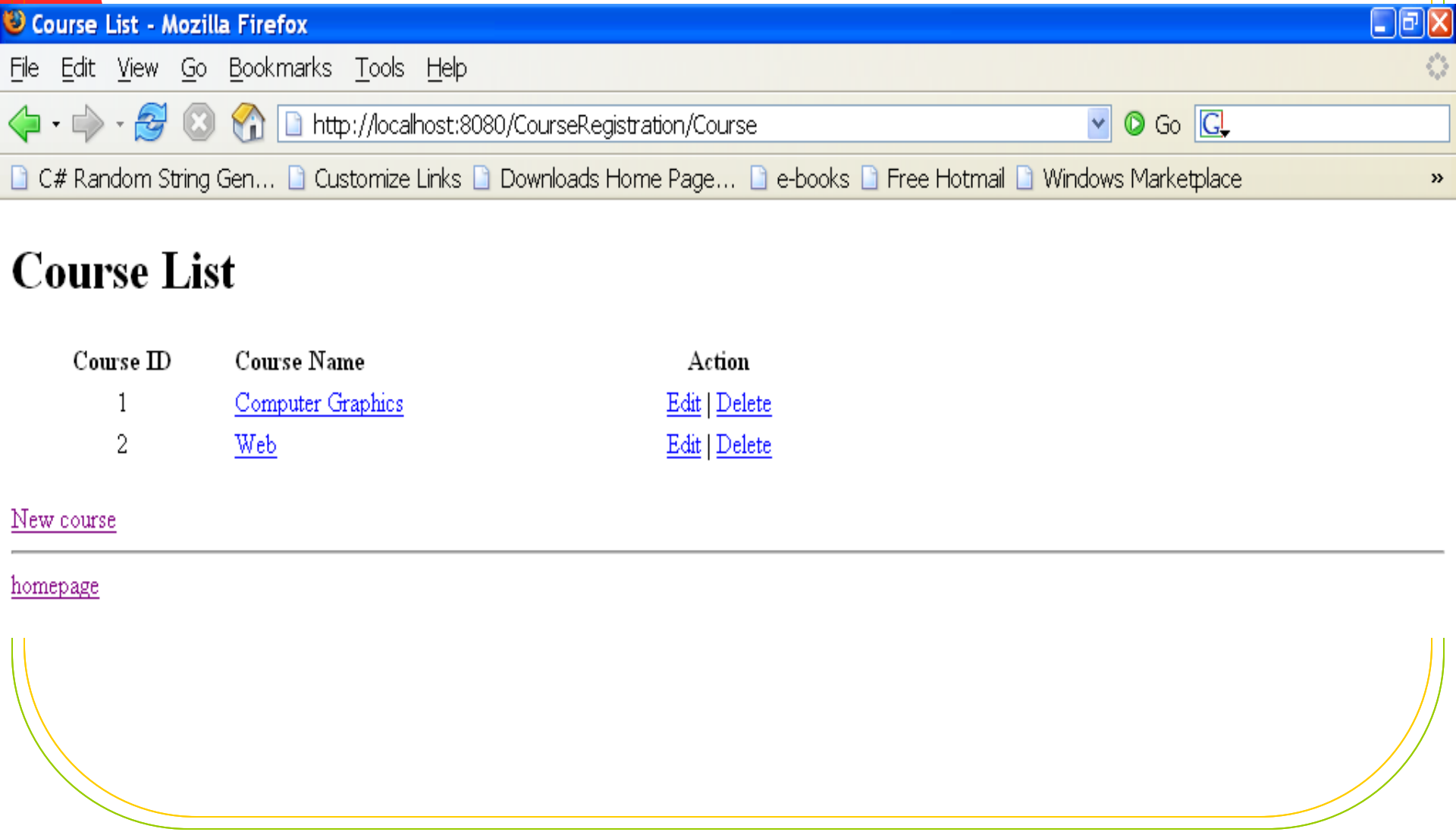
Steps to manipulate data (cont.)

- Process a results (display for edit, new…)
  - ```
    while (rs.next()) {
      String ID = rs.getString (1);
      //or String ID = rs.getString("ID");
      long YOB = rs.getLong(2);
      //or long YOB = rs.getLong("YOB");
      }
    ```
- Close the objects (should be closed when application is exited)
  - `rs.close();`
  - `stmt.close()`
  - `conn.close()`
    ```

Hint: how to edit a row in table

- Create a hyperlink.
- If you have an action file “action.jsp” and you want edit on studentID, you can do as follow:
  - When you display row, you can add an additional link such as
    <a href=
    “action.jsp?studentID=09023283”> Edit

Ex: editing and deleting row



https://mvnrepository.com/artifact/org.apache.maven.plugins/maven-war-plugin

- Netbeans version 16: maven-war-plugin
- copy the latest code of last version
- <!-- https://mvnrepository.com/artifact/org.apache.maven.plugins/maven-war-plugin -->
- <dependency>
- <groupId>org.apache.maven.plugins</groupId>
- <artifactId>maven-war-plugin</artifactId>
- <version>3.3.2</version>
- </dependency>
- Put the source into the pom.xml in: <plugin> ... </plugin>