National University of Computer & Emerging Sciences, Karachi
Computer Science Department
Fall 2024, Lab Manual - 06

Course Code: CL-2005	Course : Database Systems Lab
Instructor(s) :	Mehak Mazhar

Contents:
Connectivity:

1. PHP with MYSQL.
2. JAVA with MYSQL.
3. C# with SQL Server.

PHP Connectivity with MYSQL Using XAMPP:

Xampp:

XAMPP is one of the widely used cross-platform web servers, which helps developers to create and test their programs on a local webserver. It was developed by the Apache Friends. XAMPP is an abbreviation where X stands for Cross-Platform, A stands for Apache, M stands for MYSQL, and the Ps stand for PHP and Perl, respectively. It is an open-source package of web solutions that includes Apache distribution for many servers and command-line executable along with modules such as Apache server, MariaDB, PHP, and Perl.

Download Xampp: https://www.apachefriends.org/download.html

MYSQL:

MySQL is a relational database management system based on SQL – Structured Query Language. The application is used for a wide range of purposes, including data warehousing, e-commerce, and logging applications

PHP with MySQL:

PHP 5 and later can work with a MySQL database using MYSQLi and PDO, both are Object Oriented:

MySQLi extension (the "i" stands for improved) will work only for MySQL databases.

PDO (PHP Data Objects) PDO will work on 12 different database systems.

MYSQLi is further bifurcated to facilitate users:

i) **MYSQL Object Oriented**   ii) **MysQli (Procedural)**

In this lab activity, we will use the MYSQLi in a procedural manner.

Step-01) For this, First open Xampp control Panel and start the Apache and MYSQL servers.

Step-02) Create a database on the phpmyadmin:

Open the browser and enter **: localhost//phpmyadmin,** and then create a database by clicking the NEW button on the left pane. Enter the name of db to be "**Connection_db".**

Step-03) Create a table inside the Connection_db named Students, with std_id, std_name, std_courses as fields.

Step-01: Creating a Connection to the Database:

Create a Connection.php file in VS Code, containing Following code:

<?php

$servername = "localhost";

$username = "root";

$password = "";

// Create connection

$conn = mysqli_connect($servername, $username, $password);

// Check connection

if   (!$conn)   {   die("Connection   failed: "   .

mysqli_connect_error());

}

echo "Connected Successfully";

?>

Save this file in Xampp/htdocs, run the script file using : localhost/filename.php Creating a Database using PHP:

//Creating a Database through php: $sql=

"Create   Database   Connection_db";

mysqli_query($conn,   Squery)   if

(mysqli_query($conn, $sql))   {   echo

"Database created successfully";

}   else   {   echo   "Error   creating   database: "   .

mysqli_error($conn);

}

Creating a table:

//Creating a Database through php: $sql=

"Create   Database   Connection_db";

mysqli_query($conn,   Squery)   if

(mysqli_query($conn, $sql))   {   echo

"Database created successfully";

}   else   {   echo   "Error   creating   database: "   .

mysqli_error($conn);

}

Creating Tables: Two ways

1. Directly on phpMyAdmin

2. Front end (Text Editor) 1. Direct on phpMyAdmin:

Follow these steps:

1. Select the database in which you want to create a table.

2. Fill out the table name and quantity of fields then click Go.

3. Give every field a proper name ,data type, size and Constraint (if any).

4. Click Go.

 Step-02) Front-End: Add the string type "$db_name" variable to your $Conn string holding value of your current database. In our case: "Connection_db"

$db_name ="Connection_db";

$conn = mysqli_connect($servername, $username, $password, $db_name); then add query:

```
// sql to create table

$sql = "CREATE TABLE Students (

std_id   INT(11)   PRIMARY   KEY,

std_name  VARCHAR(30)  NOT  NULL,

std_courses TEXT(30) NOT NULL

)";

if (mysqli_query($conn, $sql)) {    echo "Table

Students created successfully";

}  else  {        echo  "Error  creating  table:  "  .

mysqli_error($conn);

}
```

After the database table is created , we may now add data into it:

Inserting Data into Table:

```
$sql = "INSERT INTO Students (std_id, std_name, std_courses)

VALUES ('Your numeric Roll number', 'Your Name', 'Any Course')"; if

(mysqli_query($conn, $sql)) {  echo "New record created successfully";


}  else  {      echo  "Error: "  .  $sql  .  "<br>"  .

mysqli_error($conn);

}
```

Inserting Multiple Rows:

```
$sql = "INSERT INTO Students (std_id, std_name, std_courses)

VALUES ('Your numeric Roll number', 'Your Name', 'Any Course')";

$sql. = "INSERT INTO Students (std_id, std_name, std_courses)

VALUES ('Your friends numeric Roll number', 'Your  Friends Name', 'Any Course')";
```

Prepared Statements and Bound Parameters:

• A prepared statement is a feature used to execute the same (or similar) SQL statements repeatedly with high efficiency.

• Prepared statements basically work like this:

> · Prepare: An SQL statement template is created and sent to the database. Certain values are left unspecified, called parameters (labeled "?"). Example: INSERT INTO Student VALUES(?, ?, ?)

> · The database parses, compiles, and performs query optimization on the SQL statement template, and stores the result without executing it

> · Execute: At a later time, the application binds the values to the parameters, andthe database executes the statement. The application may execute the statement as many times as it wants with different values

Advantages:

Compared to executing SQL statements directly, prepared statements have three main advantages:

> · Prepared statements reduces parsing time as the preparation on the query is done only once (although the statement is executed multiple times)

> · Bound parameters minimize bandwidth to the server as you need send only the parameters each time, and not the whole query

> · Prepared statements are very useful against SQL injections, because parameter values, which are transmitted later using a different protocol, neednot be correctly escaped. If the original statement template is not derivedfrom external input, SQL injection cannot occur.

$stmt = $conn->prepare("INSERT INTO Students (std_id, std_name,std_courses) VALUES (?, ?, ?)");

$stmt->bind_param("sss", $id, $name, $course);

// set parameters and execute

$id = = "any value";

$name = "any value";

$course = "any value";

//again for new row

```php
$id = = "any value";

$name =  "any value";

$course = "any value"; $stmt->execute(); echo

"New records created successfully";
```

Updating Data into Table

```php
$sql = "UPDATE Students SET std_name='Sarah' WHERE id=your roll number";

if ($conn->query($sql) === TRUE) {   echo "Record updated successfully";

} else {      echo "Error  updating  record: " .

$conn->error;

}
```

Deleting Data from table:

```php
$sql  =  "DELETE  FROM  Students  WHERE  id=your  roll

number"; if (mysqli_query($conn, $sql)) {     echo "Record

deleted successfully";

} else {         echo  "Error  deleting  record:  "  .

mysqli_error($conn);

}
```

Displaying , Inserting, Updating and Removing Data Using an HTML Form and Table to display records and changes in table.

Create a home.php file in xampp/htdocs folder, containing the following code:

```html
<form action="file to send data to/#" method="POST">

        <h3>Records Table</h3>

        <label for="id">Enter your ID:</label>

        <input type="number" name="id">

        <br>
```

```
    <label for="name">Enter your Name:</label>

     <input type="text" name="name">

   <br>

     <input type="submit" name="submit" class="submit" value="Add Record">

    </form>

      <table cellpadding=1, cellspacing=1>

      <tr> <thead><td>ID</td><td>Name</td></thead></tr>

    <tr>        <?php        while        ($rows=

mysqli_fetch_assoc($result))

{

?>

                <td><?php echo $rows['std_id'] ?></td><td><?php echo $rows['std_name']
?></td>       </tr>

      <?php } ?>

       </table>
```
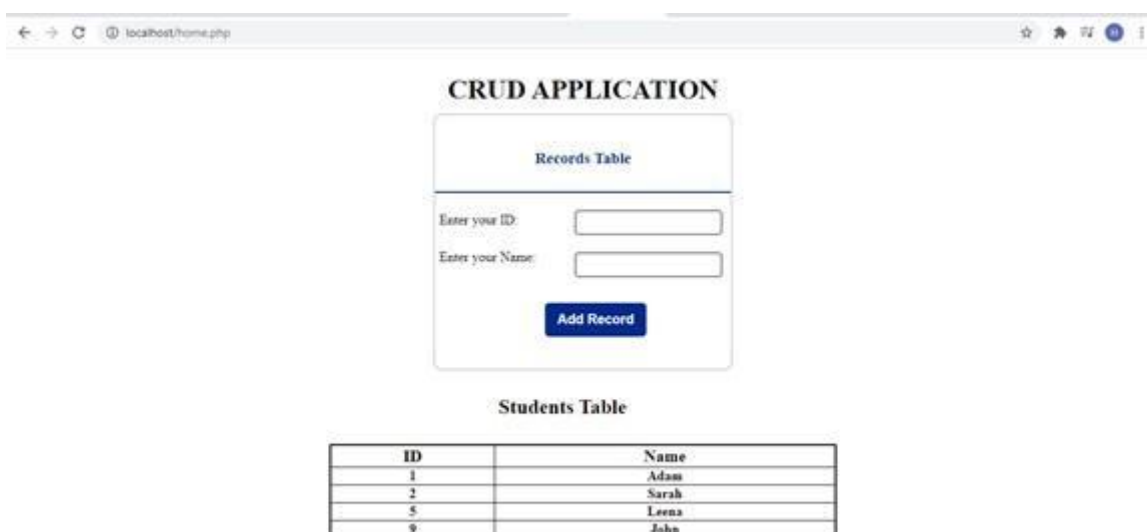
The window should appear like this:

Java Database Connectivity with MySQL

JDBC:

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database.

The current version of JDBC is 4.3. It is the stable release since 21st September, 2017. It is based on the X/Open SQL Call Level Interface. The **java.sql** package contains classes and interfaces for JDBC API.

.

MYSQL FOR JAVA:

MySQL provides connectivity for client applications developed in the Java programming language with MySQL Connector/J, a driver that implements the JDBC API.

MySQL Connector/J is a JDBC Type 4 driver. ie it is a pure java implementation of mysql protocol and does not rely on any client library.

Steps To Connect Java with MYSQL Server:.

There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:

- Register the Driver class

- Create connection

- Create statement

- Execute queries

- Close connection

While working with Visual Studio For Java Connectivity, We will require following files to get going, these are: o Updated JDK as per VS Code requirements, and Java Extension Pack from extensions tab.

o   MYSQL Connecter Jar File

- Now create a new Project in java , then add MySql jar file to its libraries

01) **Register the Driver class**: Here, Java program is loading jdbc driver to establish database

connection: Class.forName("com.mysql.cj.jdbc.Driver");

2) Create the connection object:

Before performing any database operation via JDBC, we have to open a database connection. To open a database connection we can call the getConnection() method of the DriverManager class.

The **getConnection()** method of DriverManager class is used to establish connection with the database.It requires three parameters, they are: Connection url, Database name, username and password.

Connection conn= DriverManager.getConnection(URL, username, password)
Connection con = DriverManager.getConnection ("jdbc:Mysql://localhost/mafaza", "root", "");

3) Create a Statement object:

The JDBC statements are used to execute the SQL or PL/SQL queries against the database. We need a statement for every single query. JDBC API defines the Statement, CallableStatement, and PreparedStatement types of statements.

The createStatement() method of Connection interface is used to create statements. The object of the

statement is responsible to execute queries with the database.   Statement stmt= con.createStatement();

4) Executing the Statement:
·   To show the records using SELECT query , use the executeQuery(String) method.This method returns all records in the form of a result set.
·   The executeUpdate(String SQL) method. This method returns the number of rows matched by the update statement, not the number of rows that were modified.
·   If you do not know ahead of time whether the SQL statement will be a SELECT or an UPDATE/INSERT, then you can use the execute(String SQL) method. This method will return true if the SQL query was a SELECT, or false if it was an UPDATE, INSERT, or DELETE statement.

·   Boolean inserted= stmt.execute("Insert into Student(06,'Thomas', 'Analytical Geometry');");

·   if(inserted){

·   System.out.println("inserted Successful!");

·   }

·   Or

·   **Using a Result Set:**

·   ResultSet rs = stmt.executeQuery("Select * from Connection_db.students");

·   System.out.println("Showing data!");

·   while(rs.next())

·   {

·   System.out.println(rs.getString("std_id")+""+rs.getString("std_name")+" "+rs.getString("std_course"));

·   }

5) Close the Connection:
By closing the connection object statement,the ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.

con.close();

The output of the following Code :
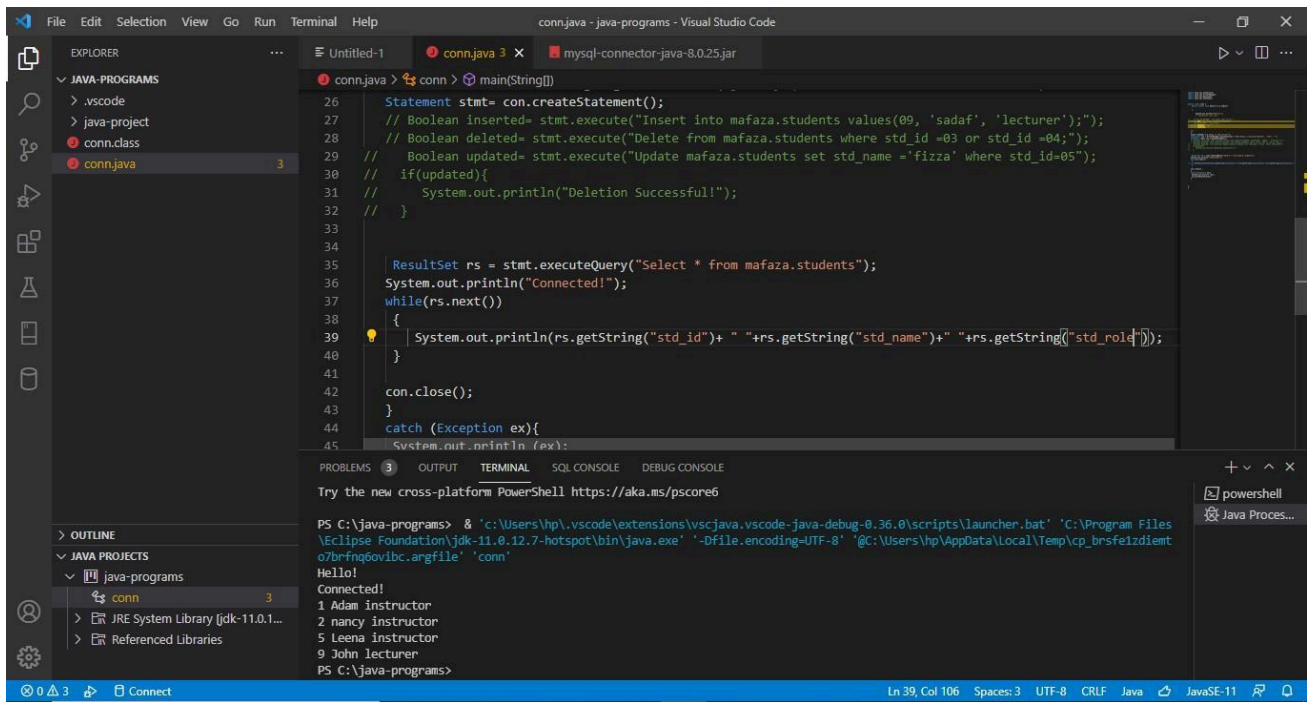
C# Connectivity with SQL Server:

C# and .Net can work with a majority of databases, the most common being Oracle and Microsoft SQL Server. But with every database, the logic behind working with all of them is mostly the same.

In this lab ,we will look at working with Microsoft SQL Server as our database. For learning purposes, you can download and use the Microsoft SQL Server Express Edition, which is a free database software provided by Microsoft from the following link:

https://www.microsoft.com/en-pk/download/details.aspx?id=42299 For Visual Studio

Download: https://visualstudio.microsoft.com/vs/community/

In working with databases, the following are the concepts which are common to all databases.

Connection – To work with the data in a database, the first obvious step is the connection. The connection to a database normally consists of the belowmentioned parameters.

SQL Command in C#

SqlCommand in C# allow the user to query and send the commands to the database. SQL command is specified by the SQL connection object. Two methods are used, ExecuteReader method for results of query and ExecuteNonQuery for insert, Update, and delete commands. It is the method that is best for the different commands.

How to connect C# to Database

Let"s now look at the code, which needs to be kept in place to create a connection to a database. In our example, we will connect to a database which has the name of Connection_db.

We will see a simple Windows forms application to work with databases. We will have a simple button called "Test" which will be used to connect/(Insert) to the database.

So let"s follow the below steps to achieve this

Step 1) The first step involves the creation of a new project in Visual Studio. After launching Visual Studio, you need to choose the menu option New->Project.

C# SQL SERVER Database

Step 2) The next step is to choose the project type as a Windows Forms application.
Here, we also need to mention the name and location of our project.

In the project dialog box, we can see various options for creating different types of projects in Visual Studio. Select Windows Forms Application.

We then give a name for the application which in our case is "WindowsFormApp2". Finally, we click

the „OK" button to let Visual Studio to create our project.

Step 3) Now add some buttons and other widgets from the toolbox to the Windows form. Put the text property of the Button as "Test". This is how it will look like:

Before this, Add the following package to your import list, to import all the functionalities of SqlClient. using System.Data.SqlClient;

Step 4) Now save the form and then in the Event handler for test button , paste the following Code:

Add teh following code in the Form Event Handler:

```
public string constring = "Data Source=HP-PC;Initial
Catalog=Connection_db;Integrated Security=True";  SqlConnection conn = new
SqlConnection(constring);conn.Open();
if(conn.State==System.Data.ConnectionState.Open)
    {
            string    q    =    "Insert    into    Test(ID,    NAME)    values
("'+txtID.Text.ToString()+"','"+txtName.Text.ToString()+"')";
```

```
        SqlCommand cmd = new SqlCommand(q, conn);

    cmd.ExecuteNonQuery();

    MessageBox.Show("Inserted Successfully!");

}

    conn.Close();
```

Code Explanation:-

The first step is to create variables, which will be used to create the connection string and the connection to the SQL Server database.

The next step is to create the connection string. The connecting string needs to be specified correctly for C# to understand the connection string. The connection string consists of the following parts

Data Source – This is the name of the server on which the database resides. In our case, it resides on a machine called "HP-PC"

The Initial Catalog is used to specify the name of the database : Initial Catalog=Connection_db

Next, we assign the connecting string to the variable conn. The variable conn, which is of type SqlConnection is used to establish the connection to the database.

Next, we use the Open method of the conn variable to open a connection to the database.

Then we insert the data using the windows form, and click insert.

If the process of insertion is successful, the message of successful insertion will be displayed.

Once the operation is completed successfully, we then close the connection to the database. It is always a good practice to close the connection to the database if nothing else is required to be done on the database.

When the above code is set, and the project is executed using Visual Studio, you will get the below output. Once the form is displayed, click the Test button.

Once we click Test, The output Should be:
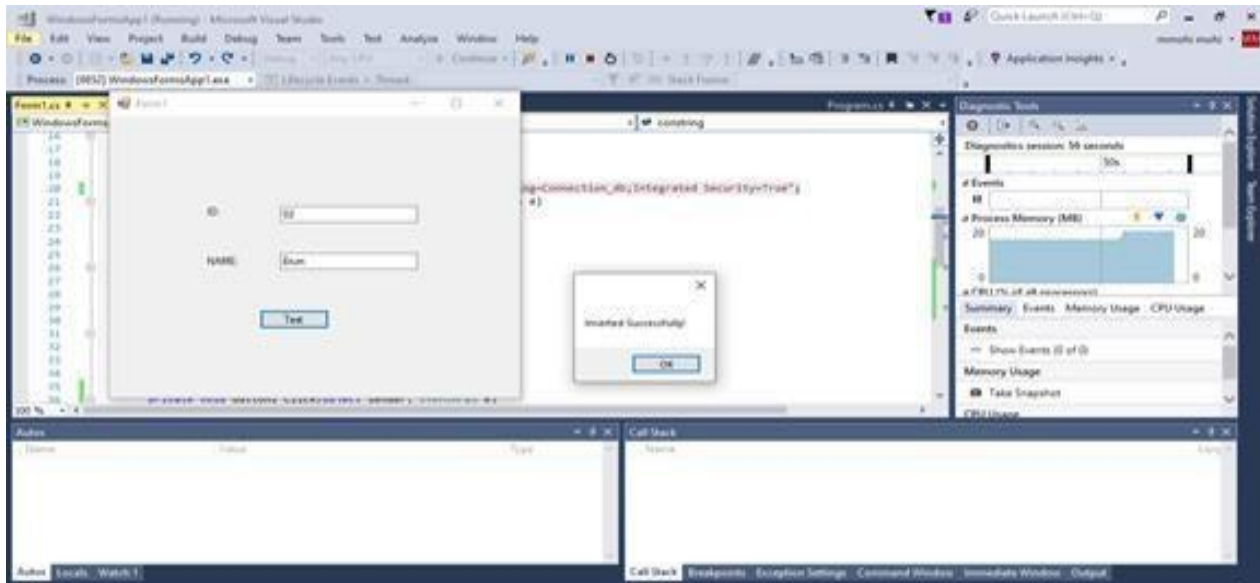
LAB TASKS:

Task-01: Create a table "Books" with fields like ISBN_NO, Title, and Author.

Create an html form to insert data into using PHP., and display the data using an html table.

Task-02: Replicate the same using C# with SQL Server using Windows Form.

Task-03: Perform the above task with Java using MYSQL in Visual Studio Code.