**install mysql workbench**

**https://www.youtube.com/watch?v=IqYfmckxeyE&t=417s**

**(instead of custom use developers 1st option because in custom visual studio needed)**

**2.MySQL JDBC driver**

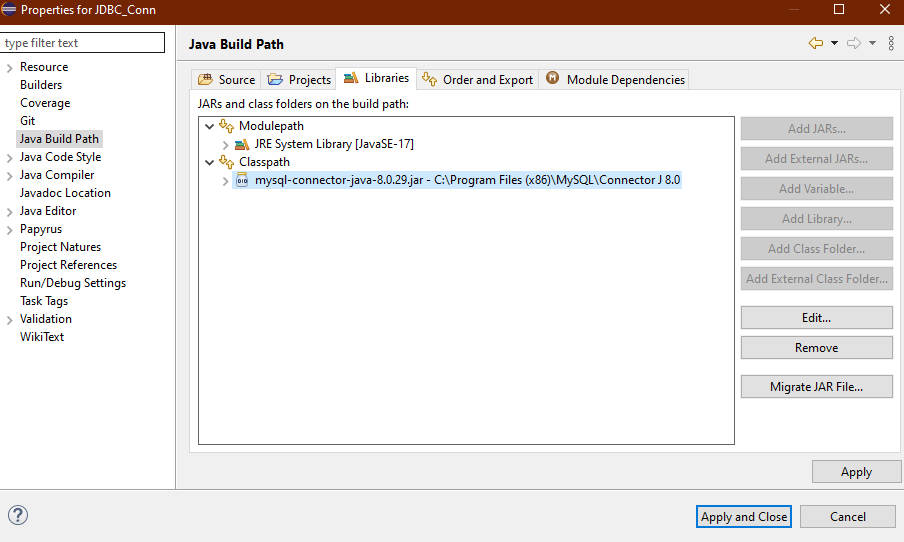
**<https://jar-download.com/artifact-search/mysql-connector-java>**

Graphical user interface, text, application

Description automatically generated

**Using JDBC with Eclipse:**

1. Open Eclipse => chose your workspace => File => New => Other => Search for “Java Project” => Next => Enter project name (Example: MySQLDBConnection) => Finish
2. Now, open the properties of the project. For this you can use shortcut key:- select project then Alt + Enter (Or) Right click on the Project => Properties
3. In the Java Build path, in the Libraries Section => Classpath => Click on “Add External Jars” => Select *mysql-connector-java-<version>.jar* => Apply => Apply and close



1. In the Project Explorer, in that Java Project, in the “Referenced Libraries” section, You can see that *mysql-connector-java-<version>.jar* is available. Eclipse IDE will use it while compilation and execution process.
2. In the Java project => Right-click on “src” => New => Class => ClassName => Finish. Use a reasonable class name, like: *ConnectionTest, Conn\_Class, etc.*
3. Now we need some details of the JDBC driver to develop our program.
   1. JDBC Driver Class Name:: ***com.mysql.cj.jdbc.Driver***
   2. URL for local MySQL:: **jdbc:mysql:///<logical-database-name>**

**NOTE**: You can view the name of the database by using **“show databases”** command

* 1. Username:***<username-of-MySQL-database>***
  2. Password: ***<password-of-MySQL-database>***

**first program “test connection”**

import java.sql.\*;

public class testconn {

public static void main(String[] args) throws Exception {

//import jdbc connector in external jar files

// register MySQL thin driver with DriverManager service

// It is optional for JDBC4.x version

/\* DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver() );

this line is same as Class.forName that automaticlly loads the static method of class com.mysql.cj.jdbc.Driver here we need to create object but in forname we don’t need to create object it will load static block automaticlly

\*/

Class.forName("com.mysql.cj.jdbc.Driver");//driver loading is optional from jdbc 4.0

// From JDBC 4.0 version onwards JDBC driver supports autoloading of driver class.

// One of the great additions in version 4.0 of JDBC,

// we don’t have to explicitly load the driver by calling Class.forName() method

// anymore. When our application attempts to connect the database for the first time,

// DriverManager automatically loads the driver found in the application CLASSPATH.

// variables

final String url = "jdbc:mysql:///jdbc1";

final String user = "root";

final String password = "scd2b@";

// establish the connection

Connection con = DriverManager.getConnection(url, user, password);

// display status message

if (con == null) {

System.out.println("JDBC connection is not established");

return;

} else

System.out.println("Congratulations," +

" JDBC connection is established successfully.\n");

// close JDBC connection

con.close();

}

}

**Second Program “fetch data from DB using query”**

**import** java.sql.\*;

**public** **class** fetchdata {

**public** **static** **void** main(String[] args) **throws** Exception {

// variables

**final** String url = "jdbc:mysql:///jdbc1";

**final** String user = "root";

**final** String password = "scd2b@";

// establish the connection

Connection con = DriverManager.*getConnection*(url, user, password);

// create JDBC statement object

Statement st = con.createStatement();

// statement is an interface and createstatement returns instance of statement

// prepare SQL query

//String query = "SELECT ID, NAME FROM student where id = 12";

//String query = "SELECT ID, NAME FROM student";

String query = "SELECT \* FROM student ";

// send and execute SQL query in Database software

ResultSet rs = st.executeQuery(query); //st.executeQuery(query) will return data in tabular structure with column name and data , ResultSet have power to store chunk of data in tabular structure

// process the ResultSet object

**while** (rs.next()) //by default the pointer is pointing the previous record w/o next it will give you error

{

System.***out***.println(rs.getInt(1) + " " + rs.getString(2));

}

//you can also mention column name rs.getInt("id")

// close JDBC objects

rs.close();

st.close();

con.close();

}

}

**Third Program “insert data from java program to DB using query”**

**import** java.sql.\*;

**public** **class** insert {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

Statement st = con.createStatement();

String q = "INSERT INTO teacher VALUES ('7', 'anaya')";

//don’t enter same id again b/c it is //primarykey duplicate insertion will throw //exception

**int** count = st.executeUpdate(q);

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**fourth Program “insert data using variables ”**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.Statement;

**public** **class** insert\_using\_var {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

**int** id = 6;

String name = "ali";

Statement st = con.createStatement();

//separate the string query and variable names using anti quotes

String q = "INSERT INTO teacher VALUES (" + id + ", '" + name +"')";

//in sql string must be in single quotes so the name is placed in single quote

**int** count = st.executeUpdate(q);

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**fifth Program “insert data using scanner ”**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.Statement;

**import** java.util.Scanner;

**public** **class** insert\_scanner {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("enter id");

**int** id = sc.nextInt();

System.***out***.println("enter name");

String name = sc.next();

Statement st = con.createStatement();

String q = "INSERT INTO teacher VALUES (" + id + ", '" + name +"')";

**int** count = st.executeUpdate(q);

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**sixth Program “insert data using prepared statement ”**

// anti quotes are not proper solution if there is 10 columns we will be having alot

//of double quotes and it will be confusing so use prepared statement

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.Statement;

**import** java.util.Scanner;

**public** **class** insert\_preparedstatement {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("enter id");

**int** id = sc.nextInt();

System.***out***.println("enter name");

String name = sc.next();

String q = "INSERT INTO teacher VALUES ( ? , ?)";

PreparedStatement st = con.prepareStatement(q);

st.setInt(1, id);

st.setString(2, name);

**int** count = st.executeUpdate();//don’t pass q here

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**seven Program “update data ”**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.Statement;

**public** **class** update {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

String q = "UPDATE teacher SET name = 'hadi' WHERE (id = 7)";

Statement st = con.createStatement();

**int** count = st.executeUpdate(q);

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**8th Program “update data using preparestatement ”**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.Statement;

**public** **class** update {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

**int** id = 1;

String name = "huma";

**String q = "UPDATE student SET name = ? WHERE id = ?";**

//

PreparedStatement st = con.prepareStatement(q);

st.setString(1, name);

st.setInt(2, id);

**int** count = st.executeUpdate();

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

con.close();

}

}

**9th Program “delete data ”**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.Statement;

**public** **class** delete {

**public** **static** **void** main(String[] args) **throws** Exception {

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

Connection con = DriverManager.*getConnection*("jdbc:mysql:///jdbc1", "root", "scd2b@");

**if** (con == **null**) {

System.***out***.println("JDBC connection is not established");

**return**;

} **else**

System.***out***.println("Congratulations," +

" JDBC connection is established successfully.\n");

String q = "DELETE FROM teacher WHERE (id = 1)";

Statement st = con.createStatement();

**int** count = st.executeUpdate(q);

System.***out***.println(count + "row/s affected");

// close JDBC connection

st.close();

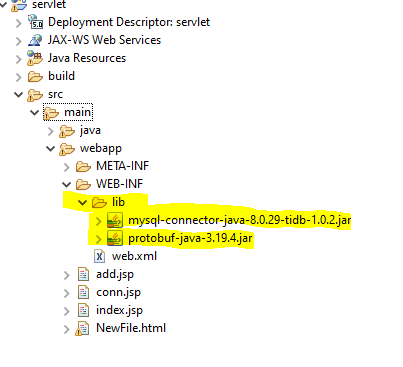
con.close();

}

}

**JDBC and JSP**

**Copy the jar file and paste it in lib folder**



<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<%@ page import= *"java.sql.\*"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<%

**final** String url = "jdbc:mysql:///new\_schema";

**final** String user = "root";

**final** String password = "scd5ab@";

Class.forName("com.mysql.cj.jdbc.Driver");//must include this line

// establish the connection

Connection con = DriverManager.getConnection(url, user, password);

// display status message

**if** (con == **null**) {

out.println("JDBC connection is not established");

**return**;

} **else**

out.println("Congratulations," +

" JDBC connection is established successfully.\n");

%>

</body>

</html>