

Laboratorio. Introducción al manejo de JDBC y MySQL

Objetivo:

Desarrollar una aplicación que permita crear una conexión con una base de datos utilizando JDBC para realizar operaciones básicas de manejo de información en una base de datos a través de sentencias en MySQL.

Pasos a seguir:

1. Crear la base de datos
2. Configurar el entorno de desarrollo
3. Programación de las clases

1. Crear la base de datos

Es necesario haber creado la base de datos en MySQL y las tablas en donde se almacenará la información.

- Crear una base de datos llamada **laboratorio_01** con la instrucción

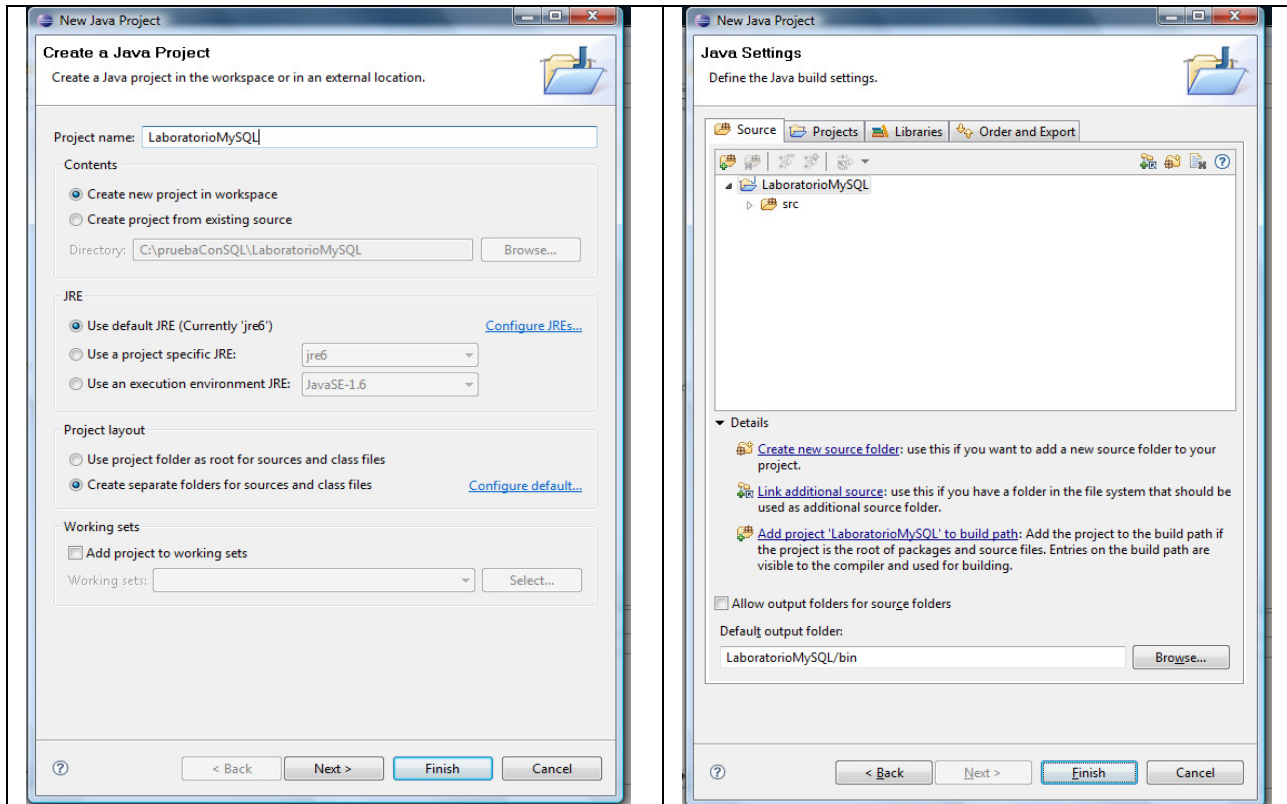
```
CREATE DATABASE laboratorio_01
```

- Crear una tabla llamada **persona** que contenga los campos para almacenar
 - Nombre
 - Apellido paterno
 - Apellido materno
 - Teléfono
 - Email
 - RFC (como llave primaria)

```
CREATE TABLE persona(  
nombre varchar(15),  
RFC varchar(15),  
apellidoPaterno varchar(20),  
apellidoMaterno varchar(20),  
telefono varchar(15),  
email varchar(40),  
PRIMARY KEY(RFC)  
)
```

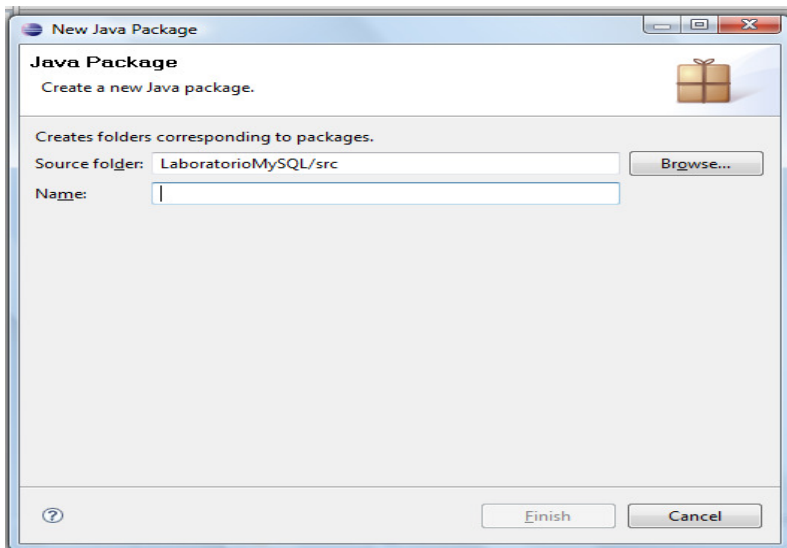
2. Configurar el entorno de desarrollo

- Arrancar la aplicación **eclipse**
- Crear un nuevo proyecto de Java (por el momento no es necesaria una aplicación web)



En la carpeta **src**, crear los paquetes necesarios para contener las clases

- Clic derecho sobre la carpeta **src**
- Seleccionar **New**
- Seleccionar **Package**

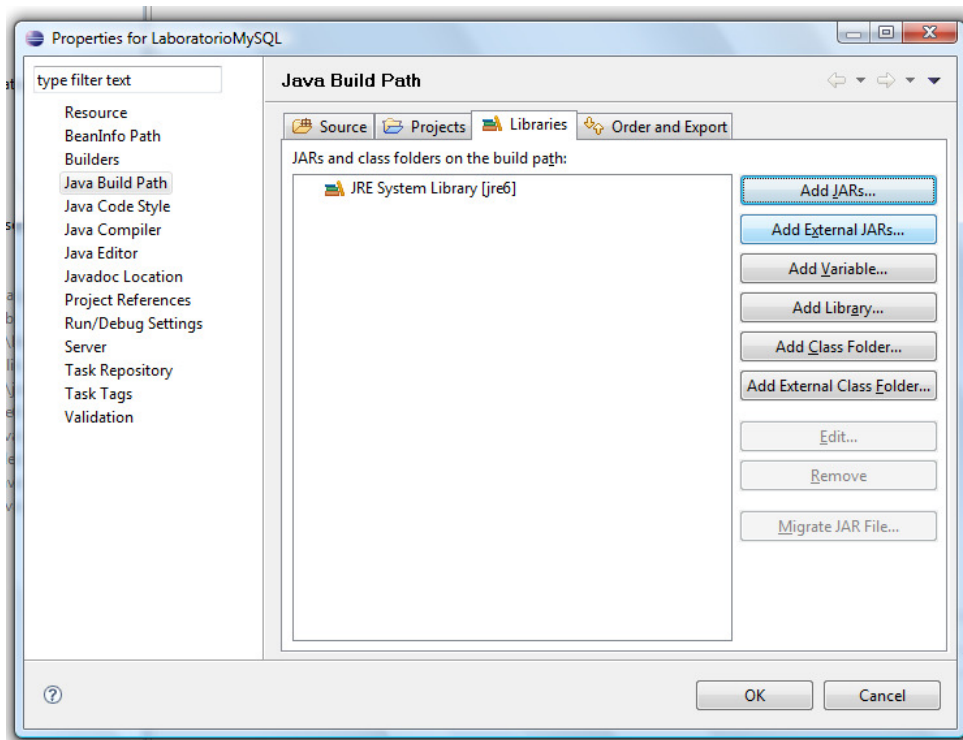


Crear los siguientes paquetes:

- **temas.laboratorio.clasesPlanas**
- **temas.laboratorio.operacionesBaseDeDatos**
- **temas.laboratorio.prueba**

Antes de comenzar con la programación de las clases, es necesario agregar las bibliotecas necesarias para trabajar con JDBC y MySQL

- Clic derecho sobre el nombre del proyecto
- Seleccionar **Properties**
- Seleccionar **Java Build Path**
- Seleccionar la pestaña **Libraries**
- Seleccionar **Add External JARs...**



- Buscar y seleccionar el archivo **mysql-connector-java-5.1.7-bin.jar**
- Seleccionar **OK**
- Se habrá agregado la biblioteca en **Referenced Libraries**

3. Programación de las clases

- Crear una clase llamada **Persona** que tenga los datos necesarios para almacenar el nombre, apellido paterno, apellido materno, teléfono, e-mail y RFC (los mismos que se tienen en la tabla **persona** en la base de datos)
- Agregar los métodos **get()** y **set()** para cada uno de los datos miembro
- Crear una clase principal llamada **Main** que será la encargada del control de la aplicación
- Crear una clase llamada **ManejoDeBaseDeDatos** que será la encargada de las operaciones relacionadas con la conexión
- Crear una clase llamada **OperacionesPersona** que se encargará del manejo de las diferentes operaciones a realizar con los datos de una persona

Código fuente de la clase **Persona**

```
package ts.laboratorio.clasesPlanas;

public class Persona {

    private String nombre;
    private String apellidoPaterno;
    private String apellidoMaterno;
    private String telefono;
    private String email;
    private String RFC;
    public String getNombre() {
        return nombre;
    }
    public String getApellidoPaterno() {
        return apellidoPaterno;
    }
    public String getApellidoMaterno() {
        return apellidoMaterno;
    }
    public String getTelefono() {
        return telefono;
    }
    public String getEmail() {
        return email;
    }
    public String getRFC() {
        return RFC;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public void setApellidoPaterno(String apellidoPaterno) {
        this.apellidoPaterno = apellidoPaterno;
    }
    public void setApellidoMaterno(String apellidoMaterno) {
        this.apellidoMaterno = apellidoMaterno;
    }
    public void setTelefono(String telefono) {
        this.telefono = telefono;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public void setRFC(String rfc) {
        RFC = rfc;
    }

    public void imprimePersona(){

        System.out.println(apellidoPaterno + " " + apellidoMaterno +
                           " " + nombre + " " + telefono +
                           " " + email + " " + RFC + "\n");
    }
}
```

Código fuente de la clase **ManejoDeBaseDeDatos**

```
package ts.laboratorio.operacionesBaseDeDatos;
import ts.laboratorio.clasesPlanas.*;
import java.sql.*;

public class ManejoDeBaseDeDatos {

    private Connection conexion=null;

    public ManejoDeBaseDeDatos() {

    }

    public Connection crearConexion(){

        try {
            DriverManager.registerDriver(new
com.mysql.jdbc.Driver());
            conexion = DriverManager.getConnection(
                "jdbc:mysql://localhost/laboratorio_01",
                "root", "josue");
        }
        catch (Exception e) {
            System.out.println("Error al crear la conexion");
            e.printStackTrace();
            conexion= null;
        }

        return conexion;
    }

    public void cerrarConexion() {
        try {
            conexion.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

Código fuente de la clase OperacionesPersona

```
package ts.laboratorio.operacionesBaseDeDatos;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import ts.laboratorio.clasesPlanas.Persona;

public class OperacionesPersona {

    private ManejoDeBaseDeDatos baseDeDatos;

    public OperacionesPersona(){
        baseDeDatos=new ManejoDeBaseDeDatos();
    }

    public void insertarElemento(Persona p) {

        Statement s;

        try {
            baseDeDatos=new ManejoDeBaseDeDatos();
            Connection con=baseDeDatos.crearConexion();
            s=con.createStatement();
            String parametro = construirParametros(p);
            s.executeUpdate("INSERT INTO persona
(apellidoPaterno,apellidoMaterno,nombre,email,telefono,RFC) "
+ parametro);
            s.close();
            baseDeDatos.cerrarConexion();

        } catch (Exception e1) {
            e1.printStackTrace();
        }

    }

}
```

```

public void despliegaLista() {

    Statement s;
    Persona persona = new Persona();
    try {
        Connection con = baseDeDatos.crearConexion();
        s=con.createStatement();
        ResultSet rs = s.executeQuery("SELECT * FROM persona
                                      order by RFC");

        while (rs.next()) {
            persona.setNombre(rs.getString("nombre"));

            persona.setApellidoPaterno(rs.getString("apellidoPaterno"));

            persona.setApellidoMaterno(rs.getString("apellidoMaterno"));
            persona.setEmail(rs.getString("email"));
            persona.setTelefono(rs.getString("telefono"));
            persona.setRFC(rs.getString("RFC"));

            persona.imprimePersona();

        }

        s.close();
        baseDeDatos.cerrarConexion();
    } catch (Exception e) {

        e.printStackTrace();
    }
}

```



```

public void buscarPorApellidoPaterno(String apPaterno) {

    Statement s;
    Connection con=baseDeDatos.crearConexion();
    Persona persona = new Persona();

    try {
        s=con.createStatement();
        ResultSet rs = s.executeQuery("SELECT * FROM persona
                                        WHERE apellidoPaterno='" +
                                        apPaterno + "'");

        //Si no hay elementos, lo indica
        if(!rs.first())
            System.out.println("No existen coincidencias");

        //Es necesario regresar al primer elemento antes de
recorrer

        else{
            rs.beforeFirst();
            while (rs.next()) {
                persona.setNombre(rs.getString("nombre"));

                persona.setApellidoPaterno(rs.getString("apellidoPaterno"));

                persona.setApellidoMaterno(rs.getString("apellidoMaterno"));
                persona.setEmail(rs.getString("email"));
                persona.setTelefono(rs.getString("telefono"));
                persona.setRFC(rs.getString("RFC"));

                persona.imprimePersona();

            }
        }

        s.close();
        baseDeDatos.cerrarConexion();
    }

    catch (Exception e) {
        e.printStackTrace();
    }

}

```

```

public Persona buscarPorRFC(String rfc) {

    Statement s;
    Connection con=baseDeDatos.crearConexion();
    Persona persona = new Persona();

    try {
        s=con.createStatement();
        ResultSet rs = s.executeQuery("SELECT * FROM persona
                                         WHERE RFC='" +
                                         rfc + "'");

        //Si no hay elementos, lo indica
        if(!rs.first()){
            System.out.println("No existen coincidencias");
            persona = null;
        }

        //Es necesario regresar al primer elemento antes de recorrer
        else{
            rs.beforeFirst();
            while (rs.next()) {
                persona.setNombre(rs.getString("nombre"));

                persona.setApellidoPaterno(rs.getString("apellidoPaterno"));

                persona.setApellidoMaterno(rs.getString("apellidoMaterno"));
                persona.setEmail(rs.getString("email"));
                persona.setTelefono(rs.getString("telefono"));
                persona.setRFC(rs.getString("RFC"));
            }

        }

        s.close();
        baseDeDatos.cerrarConexion();
        return persona;
    }

    catch (Exception e) {
        e.printStackTrace();
        return null;
    }

}

```

```

public void actualizar(Persona p){

    String parametro = construirParametros(p);
    Statement s;
    Connection con=baseDeDatos.crearConexion();

    try {
        s=con.createStatement();
        s.executeUpdate("UPDATE persona SET " +
            "nombre='" + p.getNombre() + "',"+
            "apellidoPaterno='" + p.getApellidoPaterno() +
            "', "+ "apellidoMaterno='" +
            p.getApellidoMaterno() + "', "+
            "email='" + p.getEmail() + "', "+
            "telefono='" + p.getTelefono() + "'" +
            "WHERE RFC='" + p.getRFC() + "'");
        s.close();
        baseDeDatos.cerrarConexion();

    } catch (Exception e) {
        e.printStackTrace();
    }

}

public void eliminarPorRFC(String rfc){

    Statement s;
    Connection con=baseDeDatos.crearConexion();

    try {
        s=con.createStatement();
        int res = s.executeUpdate("DELETE FROM persona WHERE
                                   RFC='" + rfc + "'");
        if(res==0)
            System.out.println("El elemento no existe");
        else
            System.out.println("Elemento borrado
                               correctamente");

        s.close();
        baseDeDatos.cerrarConexion();
    } catch (Exception e) {
        e.printStackTrace();
    }

}

```

```

    public String construirParametros(Persona p){

        String parametro="";
        parametro = parametro.concat(" VALUES (" +
            "'" + p.getApellidoPaterno() + "', " +
            "'" + p.getApellidoMaterno() + "', " +
            "'" + p.getNombre() + "', " +
            "'" + p.getEmail() + "', " +
            "'" + p.getTelefono() + "', " +
            "'" + p.getRFC() + "'" );

        return parametro;

    }
}

```

Código fuente de la clase **Main**

```

package ts.laboratorio.prueba;

import ts.laboratorio.clasesPlanas.Persona;
import ts.laboratorio.operacionesBaseDeDatos.ManejoDeBaseDeDatos;
import ts.laboratorio.operacionesBaseDeDatos.OperacionesPersona;

public class Main {

    static OperacionesPersona manejoPersona=new OperacionesPersona();

    public static void main(String[] args) {

        //inserta();
        //despliegaLista();
        //buscarApellidoPaterno();
        //eliminaRFC();
        //actualiza();
        despliegaLista();

    }
}

```

```

    public static void inserta() {

        Persona persona = new Persona();
        persona.setApellidoPaterno("Figueroa");
        persona.setApellidoMaterno("Gonzalez");
        persona.setNombre("Ivonne");
        persona.setEmail("ivonne.figueroa@gmail.com");
        persona.setTelefono("57899810");
        persona.setRFC("FIGI780718HD6");

        System.out.println(manejoPersona.construirParametros(persona));
        manejoPersona.insertarElemento(persona);
    }

    public static void despliegaLista() {
        manejoPersona.despliegaLista();
    }

    public static void buscarApellidoPaterno(){
        manejoPersona.buscarPorApellidoPaterno("Figueroa");
    }

    public static void eliminaRFC(){
        //manejoPersona.eliminarPorRFC("FIGI780718HD6");
        //manejoPersona.eliminarPorRFC("FIGI780718HD6");
    }

    public static void actualiza(){
        Persona p = new Persona();
        p = manejoPersona.buscarPorRFC("FIGI780718HD6");
        if(p!=null){
            p.imprimePersona();
            p.setNombre("Ivonne");
            p.setEmail("ivonne.figueroa@hotmail.com");
            manejoPersona.actualizar(p);
        }
        else{
            System.out.println("No existe la persona");
        }
    }
}

```

Consideraciones.

El despliegue de la lista de personas se realiza dentro del mismo método que hace la consulta, en caso de que se deseen regresar los resultado, estos pueden introducirse en alguna de las colecciones de Java como Listas Ligada, Arreglos o Mapas Hash.