

## **Acceso a bases de datos relacionales desde Java**

Vamos a proceder a explicar cómo acceder a una base de datos relacional desde Java en dos entornos y con dos sistemas gestores de bases de datos

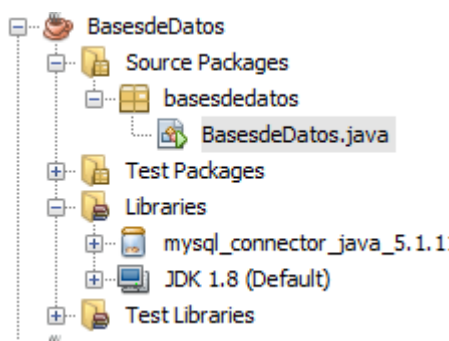
Accedemos desde NetBeans al SGBD Mysql Workbench:

Primero, debemos encender el servidor, en startup/shutdown

Después debemos utilizar Create Schema clientes,

Por último, tenemos que hacer una tabla -persona-con new table, en tables debajo de cliente. Tras definir allí nuestra tabla, procedemos a la inclusión de la fila desde nuestro programa Java

El driver de conexión a mysql debe estar cargado:



A continuación, tenemos el código que explica cómo realizar la conexión:

```
Source History
1 package pruebamysql;
2 /*
3  * PruebaMySQL.java
4  *
5  * Programa de prueba para conexión a una base de datos de MySQL.
6  * Presupone que el servidor de base de datos está arrancado, disponible,
7  * en el puerto por defecto.
8  * El usuario y password de conexión con la base de datos debe cambiarse.
9  * En la base de datos se supone que hay una base de datos llamada prueba y que
10 * tiene una tabla persona con tres campos, de esta manera:
11 * mysql> create database clientes;
12 * mysql> use clientes;
13 * mysql> create table persona (id smallint auto_increment, nombre varchar(60),
14 *   nacimiento date, primary key(id));
15 */
16
17
18 import java.sql.*;
19
20 /**
21 * Clase de prueba de conexión con una base de datos MySQL
22 */
23 public class PruebaMySQL {
24
25     /**
26     * Crea una instancia de la clase MySQL y realiza todo el código
27     * de conexión, consulta y muestra de resultados.
28     */
29 }
```

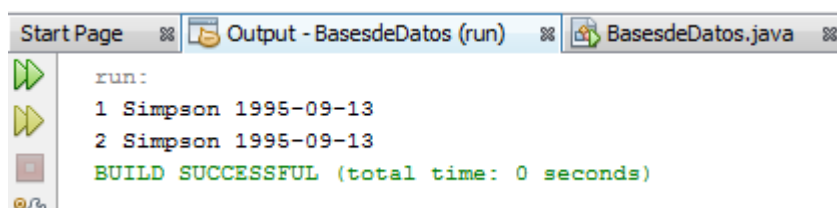
```
29 public PruebaMySQL()
30 {
31     // Se mete todo en un try por los posibles errores de MySQL
32     try
33     {
34         // Se registra el Driver de MySQL
35         DriverManager.registerDriver(new com.mysql.jdbc.Driver());
36
37         // Se obtiene una conexión con la base de datos. Hay que
38         // cambiar el usuario "root" y la clave "la_clave" por las
39         // adecuadas a la base de datos que estemos usando.
40         Connection conexion = DriverManager.getConnection (
41             "jdbc:mysql://localhost/clientes","root", "");
42
43         // Se crea un Statement, para realizar la consulta
44         Statement s = conexion.createStatement();
45
46         //Introduce dato
47         s.executeUpdate("INSERT INTO persona " + "VALUES (null,'Simpson', '1995-09-13')");
48
49         // Se realiza la consulta. Los resultados se guardan en el
50         // ResultSet rs
51         ResultSet rs = s.executeQuery ("select * from persona");
52
53         // Se recorre el ResultSet, mostrando por pantalla los resultados.
54         while (rs.next())
55         {
56             System.out.println (rs.getInt ("Id") + " " + rs.getString (2)+
57                 " " + rs.getDate(3));
58         }
59     }
60 }
```

```

60
61         // Se cierra la conexión con la base de datos.
62         conexion.close();
63     }
64     catch (Exception e)
65     {
66         e.printStackTrace();
67     }
68 }
69
70 /**
71  * Método principal, instancia una clase PruebaMySQL
72  *
73  * @param args the command line arguments
74  */
75 public static void main(String[] args)
76 {
77     new PruebaMySQL();
78 }
79
80 }

```

La ejecución de este programa nos da:

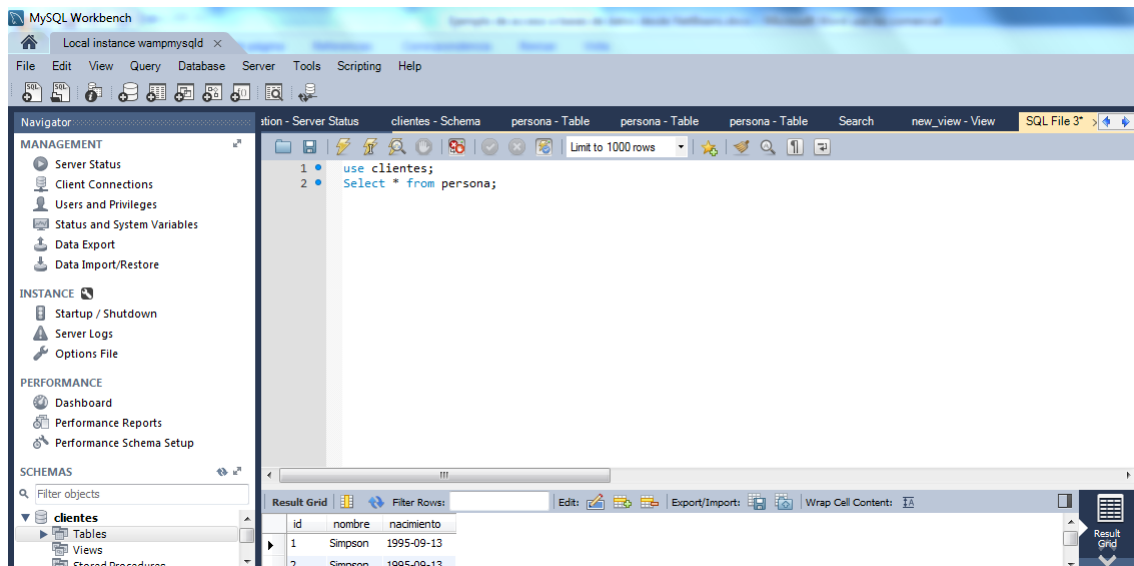


```

run:
1 Simpson 1995-09-13
2 Simpson 1995-09-13
BUILD SUCCESSFUL (total time: 0 seconds)

```

Y el resultado de consultarlo en el SGBD (Usamos MySQL Workbench) es:



Esto también puede realizarse con el XAMPP:

Además, debemos haber creado la base de datos en MySql para poder trabajar con ella

```
XAMPP for Windows - mysql -u root -p

Setting environment for using XAMPP for Windows.
alfredo@ALFREDO-HP C:\XAMPP_Final\xampplite
# cd mysql

alfredo@ALFREDO-HP C:\XAMPP_Final\xampplite\mysql
# cd bin

alfredo@ALFREDO-HP C:\XAMPP_Final\xampplite\mysql\bin
# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.1.41 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

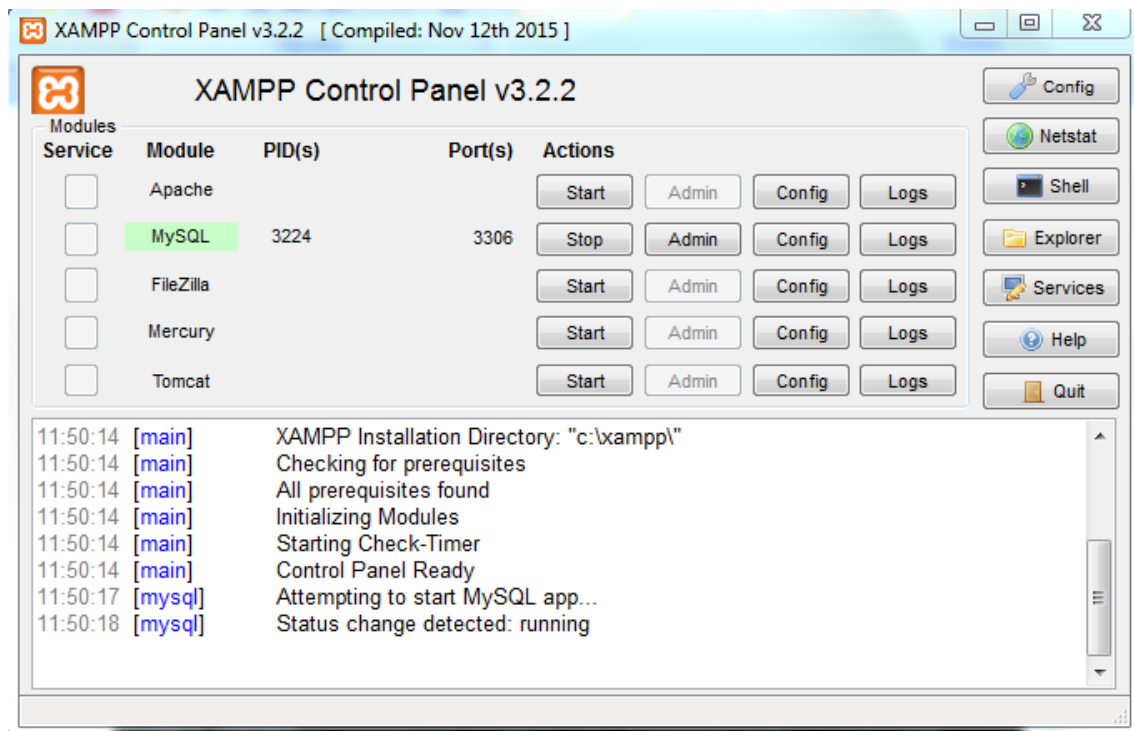
mysql> create database clientes;
Query OK, 1 row affected (0.05 sec)

mysql> create database cliente
-> ;
Query OK, 1 row affected (0.00 sec)

mysql> use cliente;
Database changed
mysql> create table persona(id smallint auto_increment,nombre varchar(60), nacim
iento date,primary key(id));
Query OK, 0 rows affected (0.42 sec)

mysql>
```

También puede ocurrir que accedamos a través de una shell de XAMPP, que use MariaDB



```

C:\xampp> mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4
Server version: 10.1.16-MariaDB mariadb.org binary distribution
Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database clientes;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> use clientes;
Database changed
MariaDB [clientes]> create table persona(id smallint auto_increment,nombre varchar(60),nacimient
o date,primary key(id));
Query OK, 0 rows affected (0.38 sec)

MariaDB [clientes]>

```

En ambos casos el código es el mismo que el anterior.

Una última forma, utilizando el entorno de desarrollo proporcionado por Coding Ground

```

1 import java.sql.*;
2 public class JDBCExample {
3     static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
4     static final String DB_URL = "jdbc:mysql://localhost/CODINGGROUND";
5
6     static final String USER = "root";
7     static final String PASS = "root";
8
9     public static void main(String[] args) {
10         Connection conn = null;
11         Statement stmt = null;
12
13         try{
14             Class.forName("com.mysql.jdbc.Driver");
15             System.out.println("Connecting to database...");
16             conn = DriverManager.getConnection(DB_URL,USER,PASS);

```

```

17             System.out.println("Creating statement...");
18             stmt = conn.createStatement();
19             String sql;
20             sql = "select * from users";
21             String sql2="create database alumnos";
22             ResultSet rs = stmt.executeQuery(sql);
23
24             while(rs.next()){
25                 int id = rs.getInt("id");
26                 int age = rs.getInt("age");
27                 String name = rs.getString("name");
28                 String sex = rs.getString("sex");
29
30                 System.out.print("ID: " + id);
31                 System.out.print(", Name: " + name);

```

```

32                 System.out.print(", Age: " + age);
33                 System.out.println(", Sex: " + sex);
34             }
35
36
37
38             stmt.executeUpdate("use alumnos");
39             //stmt.executeUpdate("CREATE TABLE ALUMNOS (exp INTEGER,nombre VARCHAR(32),sexo CHAR(1),PRIMARY KEY (exp))");
40             //stmt.executeUpdate("INSERT INTO ALUMNOS VALUES(209,\"Pepe\", \"M\")");
41             ResultSet rs2=stmt.executeQuery("select * from ALUMNOS");
42             while(rs2.next()){
43                 int id = rs2.getInt("exp");
44
45                 String name = rs2.getString("nombre");
46                 String sex = rs2.getString("sexo");

```

```
47
48         System.out.print("ID: " + id);
49         System.out.print(", Name: " + name);
50
51         System.out.println(", Sex: " + sex);
52     }
53     rs.close();
54     rs2.close();
55     stmt.close();
56     conn.close();
57 }catch(SQLException se){
58     se.printStackTrace();
59 }catch(Exception e){
60     e.printStackTrace();
61 }finally{
62
```

```
62     try{
63         if(stmt!=null)
64             stmt.close();
65     }catch(SQLException se2){
66         // nothing we can do
67     }try{
68         if(conn!=null)
69             conn.close();
70     }catch(SQLException se){
71         se.printStackTrace();
72     }//end finally try
73 }//end try
74 }
75 }
76
```

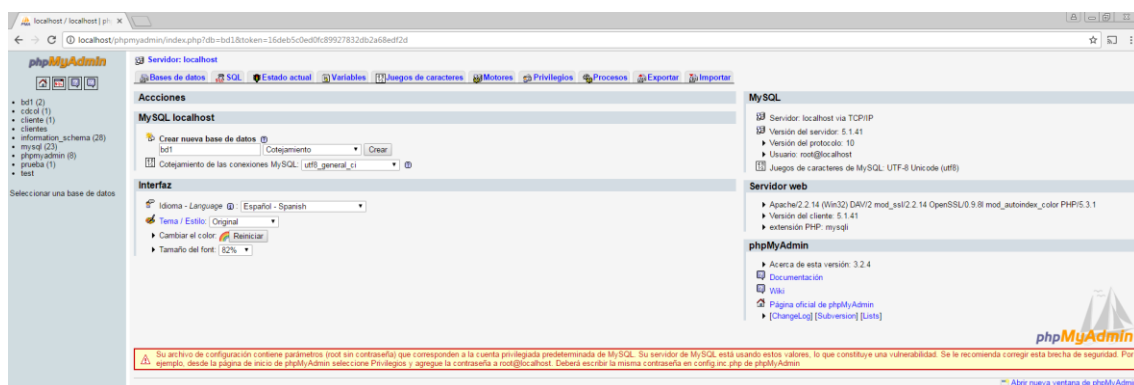
```

ID: 1, Name: Gopal, Age: 28, Sex: M
sh-4.3$ java -Xmx128M -Xms16M JDBCExample
Connecting to database...
Creating statement...
ID: 1, Name: Gopal, Age: 28, Sex: M
ID: 2, Name: Manisha, Age: 26, Sex: F
ID: 3, Name: Javed, Age: 22, Sex: M
ID: 4, Name: Raju, Age: 22, Sex: M
ID: 5, Name: Satish, Age: 29, Sex: M
ID: 6, Name: Zara, Age: 13, Sex: F
ID: 7, Name: Nuha, Age: 4, Sex: F
ID: 1, Name: Pepe, Sex: M
ID: 8, Name: Pepe, Sex: M
ID: 208, Name: Pepe, Sex: M
ID: 209, Name: Pepe, Sex: M
sh-4.3$

```

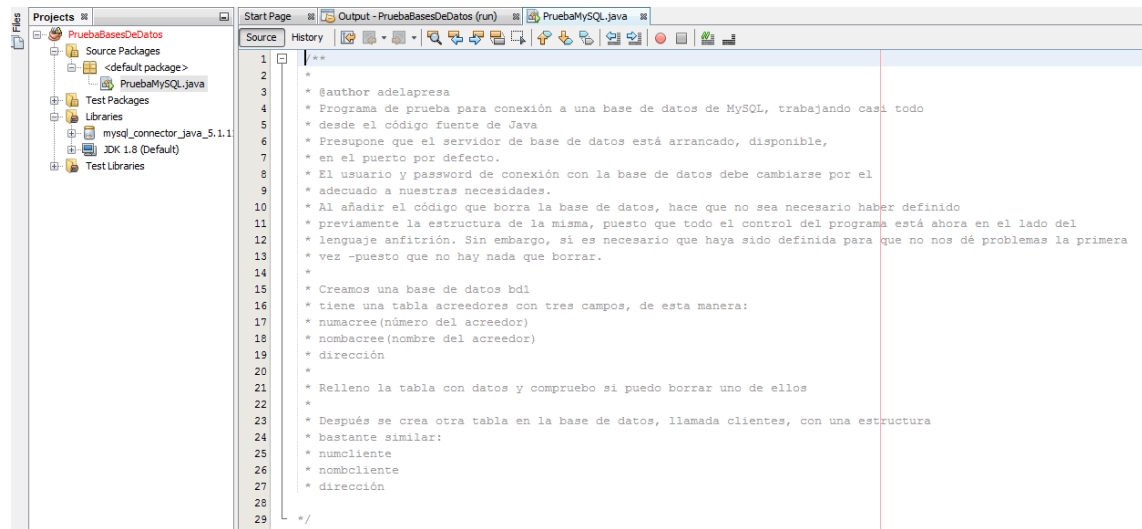
Ahora modificamos nuestro anterior programa, haciendo que casi todo el peso caiga sobre la parte de programación de Java, dejando sólo la creación de la base de datos en el servidor -sin definir más-para acceder a la misma. El código necesario para ello incluye las órdenes SQL DROP DATABASE, CREATE DATABASE, y USE DATABASE

En esta ocasión, utilizaremos el XAMPP y el PHPMyAdmin para generar la base de datos y desde código la borraremos y reescribiremos cuantas veces nos sea preciso



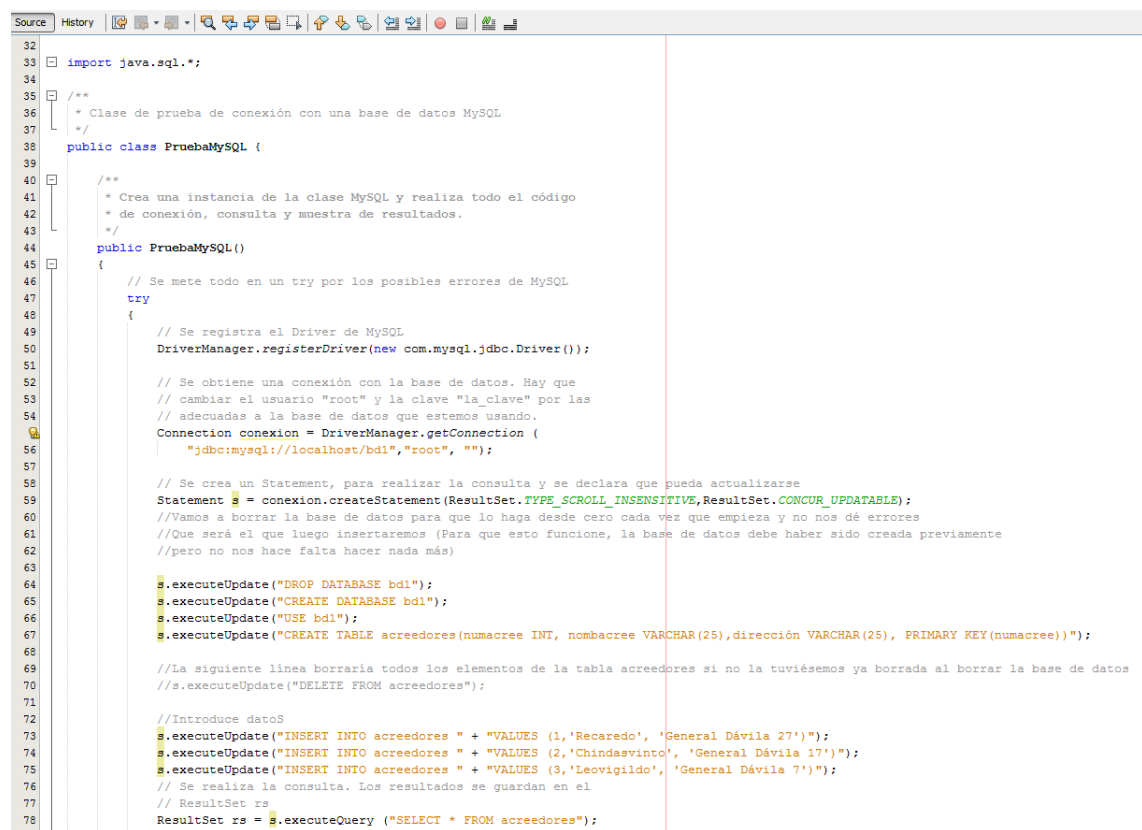


Haciendo clic en el botón “Crear” nos crea la base bd1 que luego emplearemos en el código siguiente:



The screenshot shows an IDE with a project named 'PruebaBasesDeDatos'. The 'Source Packages' view on the left shows the project structure. The main editor displays the 'PruebaMySQL.java' file. The code is a Java class with a comment block at the top explaining the purpose of the program: to test a MySQL connection. The code includes comments about the database structure, including a table named 'acreedores' with fields 'numacree', 'nombacree', and 'direccion'. The code also includes comments about the database structure, including a table named 'clientes' with fields 'numcliente', 'nombcliente', and 'direccion'.

```
1  /**
2  *
3  * @author adelapresa
4  * Programa de prueba para conexión a una base de datos de MySQL, trabajando casi todo
5  * desde el código fuente de Java
6  * Presupone que el servidor de base de datos está arrancado, disponible,
7  * en el puerto por defecto.
8  * El usuario y password de conexión con la base de datos debe cambiarse por el
9  * adecuado a nuestras necesidades.
10 * Al añadir el código que borra la base de datos, hace que no sea necesario haber definido
11 * previamente la estructura de la misma, puesto que todo el control del programa está ahora en el lado del
12 * lenguaje anfitrión. Sin embargo, si es necesario que haya sido definida para que no nos dé problemas la primera
13 * vez -puesto que no hay nada que borrar.
14 *
15 * Creamos una base de datos bd1
16 * tiene una tabla acreedores con tres campos, de esta manera:
17 * numacree(número del acreedor)
18 * nombacree(nombre del acreedor)
19 * dirección
20 *
21 * Relleno la tabla con datos y compruebo si puedo borrar uno de ellos
22 *
23 * Después se crea otra tabla en la base de datos, llamada clientes, con una estructura
24 * bastante similar:
25 * numcliente
26 * nombcliente
27 * dirección
28 *
29 */
```



The screenshot shows the continuation of the 'PruebaMySQL.java' file. The code starts with an import statement for 'java.sql.\*'. It then defines a class 'PruebaMySQL' with a constructor. The constructor contains a try-catch block that attempts to establish a connection to a MySQL database. The code includes comments explaining the steps: registering the MySQL driver, obtaining a connection, creating a statement, and executing SQL commands to drop and create the database, and then inserting data into the 'acreedores' table. The code also includes a comment about the database structure, including a table named 'clientes' with fields 'numcliente', 'nombcliente', and 'direccion'.

```
32
33 import java.sql.*;
34
35 /**
36 * Clase de prueba de conexión con una base de datos MySQL
37 */
38 public class PruebaMySQL {
39
40     /**
41     * Crea una instancia de la clase MySQL y realiza todo el código
42     * de conexión, consulta y muestra de resultados.
43     */
44     public PruebaMySQL()
45     {
46         // Se mete todo en un try por los posibles errores de MySQL
47         try
48         {
49             // Se registra el Driver de MySQL
50             DriverManager.registerDriver(new com.mysql.jdbc.Driver());
51
52             // Se obtiene una conexión con la base de datos. Hay que
53             // cambiar el usuario "root" y la clave "la_clave" por las
54             // adecuadas a la base de datos que estemos usando.
55             Connection conexion = DriverManager.getConnection (
56                 "jdbc:mysql://localhost/bd1","root", "");
57
58             // Se crea un Statement, para realizar la consulta y se declara que pueda actualizarse
59             Statement s = conexion.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
60             //Vamos a borrar la base de datos para que lo haga desde cero cada vez que empieza y no nos dé errores
61             //Que será el que luego insertaremos (Para que esto funcione, la base de datos debe haber sido creada previamente
62             //pero no nos hace falta hacer nada más)
63
64             s.executeUpdate("DROP DATABASE bd1");
65             s.executeUpdate("CREATE DATABASE bd1");
66             s.executeUpdate("USE bd1");
67             s.executeUpdate("CREATE TABLE acreedores(numacree INT, nombacree VARCHAR(25),direccion VARCHAR(25), PRIMARY KEY(numacree))");
68
69             //La siguiente línea borraría todos los elementos de la tabla acreedores si no la tuviésemos ya borrada al borrar la base de datos
70             //s.executeUpdate("DELETE FROM acreedores");
71
72             //Introduce datos
73             s.executeUpdate("INSERT INTO acreedores " + "VALUES (1,'Recaredo', 'General Dávila 27')");
74             s.executeUpdate("INSERT INTO acreedores " + "VALUES (2,'Chindasvinto', 'General Dávila 17')");
75             s.executeUpdate("INSERT INTO acreedores " + "VALUES (3,'Leovigildo', 'General Dávila 7')");
76             // Se realiza la consulta. Los resultados se guardan en el
77             // ResultSet rs
78             ResultSet rs = s.executeQuery ("SELECT * FROM acreedores");
```

```
Source History
81 // Se recorre el ResultSet, mostrando por pantalla los resultados.
82 while (rs.next())
83 {
84     System.out.println (rs.getInt ("numacree") + " " + rs.getString (2)+
85         " " + rs.getString(3));
86     //Vamos ahora a borrar el segundo de los registros insertados
87     //mediante el método ResultSet
88     if (rs.getString(2).equals("Chindasvinto")){
89         rs.deleteRow();//Esta orden me borra el registro cuyo segundo valor sea Chindasvinto
90     }
91 }
92
93 System.out.println("Ahora vamos a visualizar el listado de acreedores para comprobar que efectivamente se ha borrado");
94 ResultSet rs2=s.executeQuery("select numacree, nombacree, dirección from acreedores");
95
96 while (rs2.next())
97 {
98     System.out.println (rs2.getInt ("numacree") + " " + rs2.getString (2)+
99         " " + rs2.getString(3));
100
101     //Creamos una tabla nueva
102     s.executeUpdate("CREATE TABLE clientes(numcliente INT, nombcliente VARCHAR(25),dirección VARCHAR(25))");
103
104     // Se cierra la conexión con la base de datos.
105     conexion.close();
106
107 }
108
109 catch (Exception e)
110 {
111     e.printStackTrace();
112 }
113
114 //Vamos a crear una tabla nueva clientes
115
116 }
117
118 /**
119  * Método principal, instancia una clase PruebaMySQL
120  *
121  * @param args the command line arguments
122  */
123 public static void main(String[] args)
124 {
125     new PruebaMySQL();
126 }
127
128
129 }
```

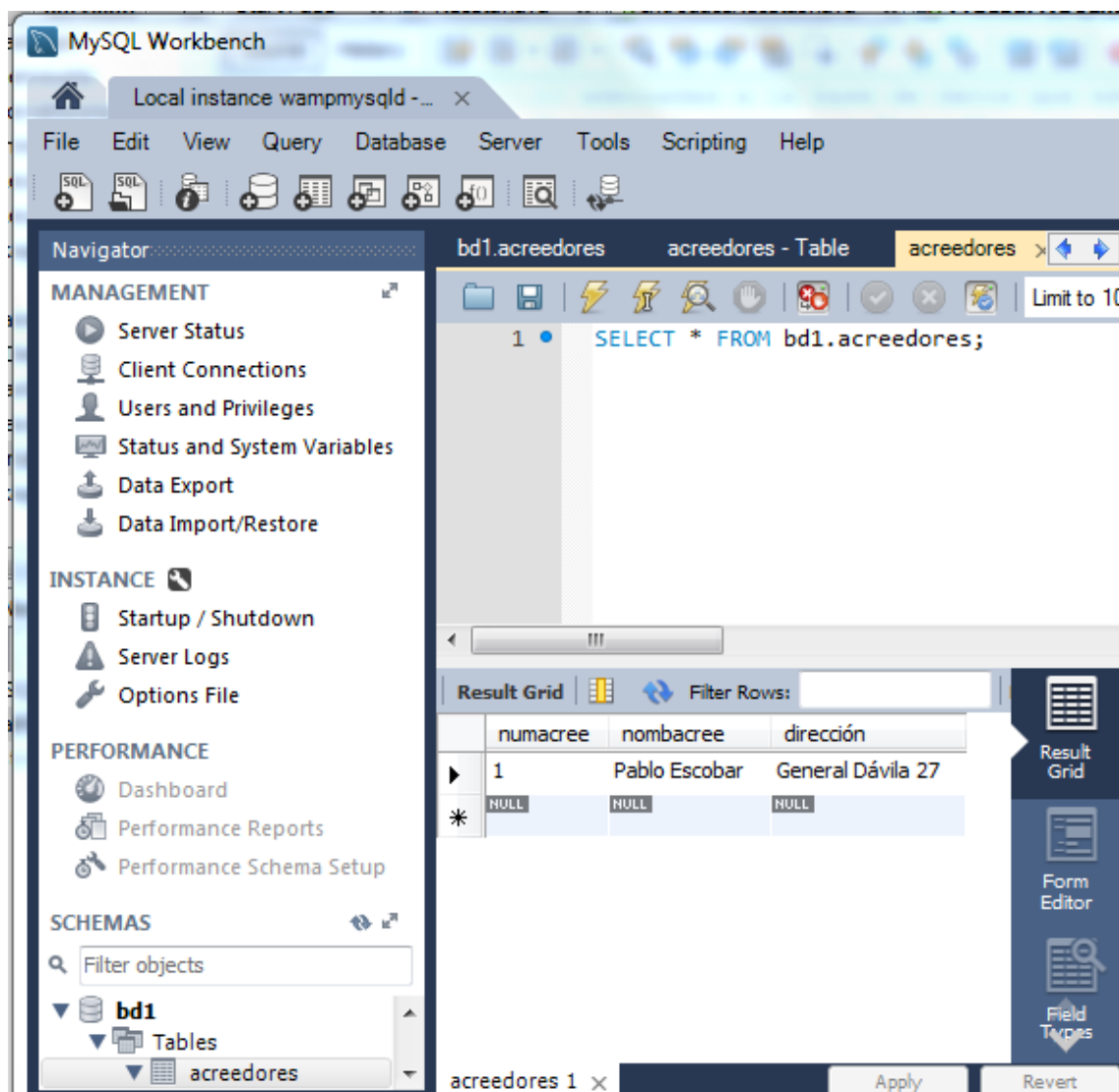
Cuya ejecución nos produce la siguiente salida:

```
Start Page Output - PruebaBasesDeDatos (run) PruebaMySQL.java
run:
1 Recaredo General Dávila 27
2 Chindasvinto General Dávila 17
3 Leovigildo General Dávila 7
Ahora vamos a visualizar el listado de acreedores para comprobar que efectivamente se ha borrado
1 Recaredo General Dávila 27
3 Leovigildo General Dávila 7
BUILD SUCCESSFUL (total time: 0 seconds)
|
```

## Ejemplo 1

Creamos una base de datos db1, formada, al principio, por una única tabla:

- “acreedores”, formada por los siguientes campos: “numacree” (número del acreedor), nombacree(nombre del acreedor) y dirección
- Borramos los clientes de la tabla. (Orden DELETE)
- Insertamos dos clientes de la tabla (Orden INSERT INTO nombre\_tabla VALUES())
- Visualizamos el contenido de la tabla (Orden SELECT \* FROM nombre\_tabla)



```
Source History
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6
7   /**
8   *
9   * @author adelapresa
10  */
11  /*
12   * PruebaMySQL.java
13   *
14   * Programa de prueba para conexión a una base de datos de MySQL.
15   * Presupone que el servidor de base de datos está arrancado, disponible,
16   * en el puerto por defecto.
17   * El usuario y password de conexión con la base de datos debe cambiarse.
18   * En la base de datos se supone que hay una base de datos llamada bd1 y que
19   * tiene una tabla acreedores con tres campos, de esta manera:
20   * numacree(número del acreedor)
21   * nombacree(nombre del acreedor)
22   * dirección
23
24  */
25
26
27
28  import java.sql.*;
29
30  /**
```

```
Source History
31  * Clase de prueba de conexión con una base de datos MySQL
32  */
33  public class PruebaMySQL {
34
35      /**
36       * Crea una instancia de la clase MySQL y realiza todo el código
37       * de conexión, consulta y muestra de resultados.
38       */
39      public PruebaMySQL()
40      {
41          // Se mete todo en un try por los posibles errores de MySQL
42          try
43          {
44              // Se registra el Driver de MySQL
45              DriverManager.registerDriver(new com.mysql.jdbc.Driver());
46
47              // Se obtiene una conexión con la base de datos. Hay que
48              // cambiar el usuario "root" y la clave "la clave" por las
49              // adecuadas a la base de datos que estemos usando.
50              Connection conexion = DriverManager.getConnection (
51                  "jdbc:mysql://localhost/bd1","root", "");
52
53              // Se crea un Statement, para realizar la consulta y se declara que pueda actualizarse
54              Statement s = conexion.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
55              //Vamos a borrar todos los datos de la tabla para que sólo haya uno en la misma
56              //Que será el que luego insertaremos
57              s.executeUpdate("DELETE FROM acreedores");
58          }
59          catch (SQLException e)
60          {
61              e.printStackTrace();
62          }
63      }
64  }
```

```

Source History
59 //Introduce dato
60 s.executeUpdate("INSERT INTO acreedores " + "VALUES (1,'Pablo Escobar', 'General Dávila 27')");
61 s.executeUpdate("INSERT INTO acreedores " + "VALUES (2,'Juan Valdés', 'General Dávila 17')");
62 // Se realiza la consulta. Los resultados se guardan en el
63 // ResultSet rs
64 ResultSet rs = s.executeQuery ("select numacree, nombacree, dirección from acreedores");
65
66
67 // Se recorre el ResultSet, mostrando por pantalla los resultados.
68 while (rs.next())
69 {
70     System.out.println (rs.getInt ("numacree") + " " + rs.getString (2)+
71         " " + rs.getString(3));
72     //Vamos ahora a borrar el segundo de los registros insertados
73     //mediante el método ResultSet
74     if (rs.getString(2).equals("Juan Valdés")){
75         rs.deleteRow();
76     }
77 }
78
79 System.out.println("Ahora vamos a visualizar el listado de acreedores para comprobar que efectivamente seha borrado");
80 ResultSet rs2=s.executeQuery("select numacree, nombacree, dirección from acreedores");
81
82 while (rs2.next())
83 {
84     System.out.println (rs2.getInt ("numacree") + " " + rs2.getString (2)+
85         " " + rs2.getString(3));}
86
87 // Se cierra la conexión con la base de datos.
88 conexion.close();
89

```

```

89     }
90     catch (Exception e)
91     {
92         e.printStackTrace();
93     }
94 }
95
96 /**
97  * Método principal, instancia una clase PruebaMySQL
98  *
99  * @param args the command line arguments
100  */
101 public static void main(String[] args)
102 {
103     new PruebaMySQL();
104 }
105
106 }

```

En ejecución este programa nos produce la siguiente salida:

```

Output - PruebaBasesDeDatos (run)
run:
1 Pablo Escobar General Dávila 27
2 Juan Valdés General Dávila 17
Ahora vamos a visualizar el listado de acreedores para comprobar que efectiva
1 Pablo Escobar General Dávila 27
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Ejemplo 2

softwProyectos es el nombre de una base de datos. Está formada, al principio, por las siguientes tablas:

“empresas”. Formada por los campos “NombEmpresa”, “teléfono”

“Proyectos”. Formada por los siguientes campos:  
“NombreProyecto”, “NombreEmpresa”, “Presupuesto”

Escribe un programa que se conecte a la base de datos “sofwProyectos” con el fin de pedir un número y visualizar los datos del proyecto que está en esa posición, los datos del proyecto que está en la posición anterior y los datos del proyecto que está en la posición siguiente. Se van a estar pidiendo distintas posiciones hasta que el usuario indique que no quiere visualizar más proyectos. Al final del programa se visualizará el listado completo de todos los proyectos.

Creamos la base de datos:

Tenemos la base de datos:

phpMyAdmin

Servidor: localhost Base de datos: softwproyectos Tabla: proyectos

Examinar Estructura SQL Buscar Insertar Exportar Importar Operaciones Vaciar Eliminar

Mostrando registros 0 - 3 (4 total. La consulta tardó 0.0007 seg)

```
SELECT * FROM 'proyectos' LIMIT 0 , 30
```

Mostrar: 30 filas empezando de 0 en modo horizontal y repetir los encabezados cada 100 celdas

+ Opciones

	NombreProyecto	NombreEmpresa	Presupuesto
<input type="checkbox"/>	Manhattan	Microsoft	2000
<input type="checkbox"/>	EcoTecnó	Apple	2000
<input type="checkbox"/>	Watchmen	Dc Comics	2000
<input type="checkbox"/>	La Torre Oscura	Stephen King SL	2000

← Marcar todos/as / Desmarcar todos Para los elementos que están marcados:

Mostrar: 30 filas empezando de 0 en modo horizontal y repetir los encabezados cada 100 celdas

Operaciones sobre los resultados de la consulta

Vista de impresión Previsualización para imprimir (documento completo) Exportar CREATE VIEW

Guardar esta consulta en favoritos

Etiqueta:  ☐ Permitir que todo usuario pueda acceder a este favorito

El código para realizar lo que nos piden es el siguiente:

```

1 package softwproyectos;
2 /**
3  *
4  * @author alfredo
5  */
6 import java.util.*;
7 import java.sql.*;
8 public class MuestraProyectos {
9     Connection conexion;
10
11     public static void main(String[] args) {
12         try {
13             DriverManager.registerDriver(new com.mysql.jdbc.Driver());
14
15             Connection conexion = DriverManager.getConnection (
16                 "jdbc:mysql://localhost/softwproyectos","root", "");
17
18             Statement instruccion=conexion.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
19
20             ResultSet rs= instruccion.executeQuery("SELECT * FROM proyectos");
21             Scanner sc= new Scanner(System.in);
22
23             do {
24                 System.out.println("Introduce el número de la posición del proyecto\nEso sacará tanto el siguiente como el anterior si existen\nIntroduce 0 es para terminar");
25                 opcion=sc.nextInt();
26                 if ((opcion!=1)&&(opcion!=0)){
27                     rs.absolute(opcion-1);
28                     System.out.println("Los datos del elemento previo al que pides son los siguientes");
29                     System.out.println("Nombre Proyecto: "+rs.getString(1));
30                     System.out.println("Nombre empresa: "+rs.getString(2));
31                     System.out.println("Presupuesto proyecto: "+rs.getInt(3));
32                 }
33                 System.out.println("-----");
34                 rs.absolute(opcion);
35                 System.out.println("Los datos del elemento en el que estás son los siguientes");
36                 System.out.println("Nombre Proyecto: "+rs.getString(1));
37                 System.out.println("Nombre empresa: "+rs.getString(2));
38                 System.out.println("Presupuesto proyecto: "+rs.getInt(3));
39                 System.out.println("-----");
40             } while (opcion!=0);
41
42             System.out.println("Los datos del elemento siguiente al que pides son estos");
43             System.out.println("Nombre Proyecto: "+rs.getString(1));
44             System.out.println("Nombre empresa: "+rs.getString(2));
45             System.out.println("Presupuesto proyecto: "+rs.getInt(3));
46             System.out.println("-----");
47             } while (opcion!=0);
48         }
49     }
50
51     catch(SQLException e){}
52
53     try {
54         Connection conexion = DriverManager.getConnection (
55             "jdbc:mysql://localhost/softwProyectos","root", "");
56         Statement instruccion=conexion.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
57         ResultSet rs= instruccion.executeQuery("SELECT * FROM proyectos");
58         System.out.println("Lista Final de Proyectos");
59
60         while (rs.next()){
61             System.out.println("Los datos del proyecto");
62             System.out.println("Nombre Proyecto: "+rs.getString(1));
63             System.out.println("Nombre empresa: "+rs.getString(2));
64             System.out.println("Presupuesto proyecto: "+rs.getInt(3));
65         }
66     } catch(SQLException e){}
67
68 }
69
70
71
72
73
74
75
76
77
78

```

La ejecución de este programa nos produce:

```
run:
Introduce el número de la posición del proyecto
Eso sacará tanto el siguiente como el anterior si existen
Introduce 0 es para terminar
1
-----
Los datos del elemento en el que estás son los siguientes
Nombre Proyecto: Manhattan
Nombre empresa: Microsoft
Presupuesto proyecto: 2000
-----
Los datos del elemento siguiente al que pides son estos
Nombre Proyecto: EcoTecno
Nombre empresa: Apple
Presupuesto proyecto: 2000
-----
Introduce el número de la posición del proyecto
Eso sacará tanto el siguiente como el anterior si existen
Introduce 0 es para terminar
2
Los datos del elemento previo al que pides son los siguientes
Nombre Proyecto: Manhattan
Nombre empresa: Microsoft
Presupuesto proyecto: 2000
-----
Los datos del elemento en el que estás son los siguientes
Nombre Proyecto: EcoTecno
Nombre empresa: Apple
Presupuesto proyecto: 2000
-----
Los datos del elemento siguiente al que pides son estos
Nombre Proyecto: Watchmen
Nombre empresa: Dc Comics
Presupuesto proyecto: 2000
-----
Introduce el número de la posición del proyecto
Eso sacará tanto el siguiente como el anterior si existen
Introduce 0 es para terminar
4
Los datos del elemento previo al que pides son los siguientes
Nombre Proyecto: Watchmen
Nombre empresa: Dc Comics
Presupuesto proyecto: 2000
-----
Los datos del elemento en el que estás son los siguientes
Nombre Proyecto: La Torre Oscura
Nombre empresa: Stephen King SL
Presupuesto proyecto: 2000
-----
Introduce el número de la posición del proyecto
Eso sacará tanto el siguiente como el anterior si existen
Introduce 0 es para terminar
0
-----
```



```
-----  
Lista Final de Proyectos  
Los datos del proyecto  
Nombre Proyecto: Manhattan  
Nombre empresa: Microsoft  
Presupuesto proyecto: 2000  
Los datos del proyecto  
Nombre Proyecto: EcoTecno  
Nombre empresa: Apple  
Presupuesto proyecto: 2000  
Los datos del proyecto  
Nombre Proyecto: Watchmen  
Nombre empresa: Dc Comics  
Presupuesto proyecto: 2000  
Los datos del proyecto  
Nombre Proyecto: La Torre Oscura  
Nombre empresa: Stephen King SL  
Presupuesto proyecto: 2000  
BUILD SUCCESSFUL (total time: 33 seconds)
```

### Ejemplo 3.

#### Enunciado 1

Escribe un programa que se conecte a la base de datos “softwProyectos” con el fin de cambiar el presupuesto de todos los proyectos a 2000 Euros.

#### Enunciado 2

Crea una tabla en la base de datos softwProyectos. Dicha tabla guardará información de los empleados, de los cuales nos interesan: número de empleado, nombre y edad.

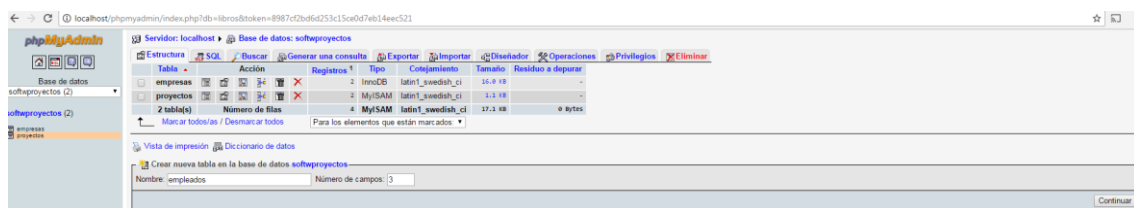
Posteriormente, el programa insertará en la tabla los datos de varios empleados, tantos como el usuario quiera. Dicha inserción se hará usando la clase ResultSet.

Por último, se visualizará el contenido de la tabla

#### Enunciado 3

Escribe un programa que gestione la base de datos softwProyectos, borrando todos los registros de la tabla empleado que tengan menos de 25 años

Creemos la tabla empleado, con tres campos



localhost / localhost / sql x

localhost/phpmyadmin/index.php?db=libros&token=8987cf2bd6d253c15ce0d7eb14eec521

phpMyAdmin

Base de datos  
softwproyectos (3)

empleados  
empresas  
proyectos

Servidor: localhost ▶ Base de datos: softwproyectos ▶ Tabla: empleados

Examinar Estructura SQL Buscar Insertar Exportar Importar Operaciones Vaciar Eliminar

✓ La Tabla 'softwproyectos'. 'empleados' se creó.

```
CREATE TABLE `softwproyectos`.`empleados` (
  `numEmpleado` INT NOT NULL ,
  `nombre` VARCHAR(255) NOT NULL ,
  `edad` INT NOT NULL ,
  ENGINE = INNODB
)
```

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
<input type="checkbox"/>	numEmpleado	int(11)			No	None		
<input type="checkbox"/>	nombre	varchar(255)	latin1_swedish_ci		No	None		
<input type="checkbox"/>	edad	int(11)			No	None		

Marcar todos/as / Desmarcar todos Para los elementos que están marcados:

Vista de impresión Vista de relaciones Planteamiento de la estructura de tabla

Añadir 1 campo(s) Al final de la tabla Al comienzo de la tabla Después de numEmpleado Continuar

⚠ ¡No se ha definido el índice!

Crear un índice en 1 columna(s) Continuar

Espacio utilizado		Estadísticas de la fila	
Tipo	Uso	Enunciado	Valor
Datos	16,384 Bytes	Formato	Compact
Índice	0 Bytes	Filas	0
Residuo a depurar	4,096.0 KB	Creación	17-05-2017 a las 19:06:54
Efectivo/a	-4,177,920 Bytes		
Total	16,384 Bytes		

El programa que hace uso del ResultSet para insertar los valores

```
Source History
1 package softwproyectos;
2 import java.sql.Connection;
3 import java.sql.DriverManager;
4 import java.sql.ResultSet;
5 import java.sql.SQLException;
6 import java.sql.Statement;
7 import java.util.Scanner;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10
11 /**
12  *
13  * @author alfredo
14  */
15 public class SoftwProyectos {
16
17     /**
18      * @param args the command line arguments
19      */
20     public static void main(String[] args) {
21         int opcion=0;
22         ResultSet rs;
23         Scanner sc= new Scanner(System.in);
24         Scanner sc2=new Scanner (System.in);
25         try {
26             // TODO code application logic here
27             DriverManager.registerDriver(new com.mysql.jdbc.Driver());
28
29             Connection conexion = DriverManager.getConnection (
30                 "jdbc:mysql://localhost/softwProyectos","root", "");
31
32             Statement instruccion=conexion.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
33
34             instruccion.executeUpdate("UPDATE Proyectos SET presupuesto='2000'");
35
36             do{
37                 System.out.println("Carga de usuarios\nIntroduce 0 para salir\nIntroduce cualquier otro número para insertar otro empleado");
38                 opcion=sc.nextInt();
39                 if (opcion!=0){
40                     System.out.println("Introduce número de empleado");
41
42                     int numemp=sc.nextInt();
43                     System.out.println("Introduce nombre del empleado");
44                     String nombemp=sc2.nextLine();
45
46                     System.out.println("Introduce la edad del empleado");
47                     int edad= sc.nextInt();
48                 }
49             } while (opcion!=0);
50         } catch (SQLException ex) {
51             Logger.getLogger(SoftwProyectos.class.getName()).log(Level.SEVERE, null, ex);
52         }
53     }
54 }
```

```

48
49
50 //instruccion.executeUpdate("INSERT INTO empleados VALUES("+numemp+", "+nombemp+", "+edad+"");
51 rs= instruccion.executeQuery("SELECT * FROM empleados");
52 rs.moveToInsertRow();
53 rs.updateInt("numEmpleado", numemp);
54 rs.updateString("nombre", nombemp);
55 rs.updateInt("edad", edad);
56 rs.insertRow();
57
58 }
59
60 //Mostramos la tabla
61 rs= instruccion.executeQuery("SELECT * FROM empleados");
62 while (rs.next()){
63     System.out.println("Empleado número "+rs.getInt(1)+" nombre: "+rs.getString(2)+" edad: "+rs.getInt(3));
64 }
65 }while(opcion!=0);
66
67
68
69 } catch (SQLException ex) {
70     Logger.getLogger(SoftwProyectos.class.getName()).log(Level.SEVERE, null, ex);
71 }
72
73 }
74
75 }
76

```

En ejecución nos da lo siguiente:

```

run:
Carga de usuarios
Introduce 0 para salir
Introduce cualquier otro número para insertar otro empleado
0
Empleado número 1 nombre: Alfredo edad: 23
Empleado número 2 nombre: Monica edad: 22
BUILD SUCCESSFUL (total time: 18 seconds)
|

```

Si optamos por introducir otro empleado:

```

run:
Carga de usuarios
Introduce 0 para salir
Introduce cualquier otro número para insertar otro empleado
1
Introduce número de empleado
3
Introduce nombre del empleado
Recaredo
Introduce la edad del empleado
23
Empleado número 1 nombre: Alfredo edad: 23
Empleado número 2 nombre: Monica edad: 22
Empleado número 3 nombre: Recaredo edad: 23
Carga de usuarios
Introduce 0 para salir
Introduce cualquier otro número para insertar otro empleado
0
Empleado número 1 nombre: Alfredo edad: 23
Empleado número 2 nombre: Monica edad: 22
Empleado número 3 nombre: Recaredo edad: 23
BUILD SUCCESSFUL (total time: 22 seconds)
|

```

Escribe un programa que gestione la base de datos softwProyectos, borrando todos los registros de la tabla empleado que tengan menos de 25 años

Aparte de sacar el listado del bucle para que no nos lo haga más de una vez, ahora, incluimos la orden

```
instruccion.executeUpdate("DELETE FROM empleados WHERE edad<25");
```

Con esto es suficiente para realizarlo, pero mostramos de nuevo la información para verlo desde el programa:

```

rs= instruccion.executeQuery("SELECT * FROM empleados");

while (rs.next()){

    System.out.println("Empleado    número    "+rs.getInt(1)+"    nombre:
"+rs.getString(2)+" edad: "+rs.getInt(3));

}

```

Esto nos queda lo siguiente en ejecución:

---

```
run:
Carga de usuarios
Introduce 0 para salir
Introduce cualquier otro número para insertar otro empleado
1
Introduce número de empleado
4
Introduce nombre del empleado
Chindasvinto
Introduce la edad del empleado
54
Carga de usuarios
Introduce 0 para salir
Introduce cualquier otro número para insertar otro empleado
0
Empleado número 1 nombre: Alfredo edad: 23
Empleado número 2 nombre: Monica edad: 22
Empleado número 3 nombre: Recaredo edad: 23
Empleado número 4 nombre: Chindasvinto edad: 54
Ahora borramos lo menores de 25 años de nuestra tabla de empleados y la mostramos
Empleado número 4 nombre: Chindasvinto edad: 54
BUILD SUCCESSFUL (total time: 18 seconds)
```

Como podemos ver, el único empleado que queda en la tabla al final es Chindasvinto