Acceso a datos

ActUT03. Conectores a sistemas gestores de bases de datos

Francisco José García Cutillas | 2FPGS_DAM

Índice

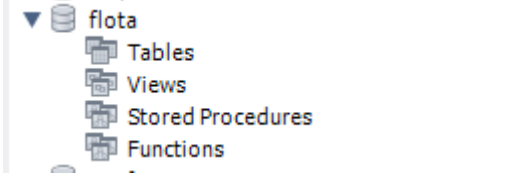
Parte 1

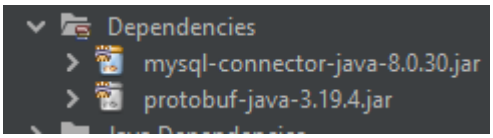
Ejercicio 1

Genera el esquema de base de datos FLOTA y realiza la conexión al mismo.

- **La generación del esquema en MySQL puedes hacerlo mejor con Workbench.**

```
1 •   create database FLOTA;
```






```java
package com.mycompany.conexion;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author Fran
 */
public class MySQLConnection {

    private static String driver = "jdbc:mysql";
    private static String hostPuerto = "localhost:3306";

    private MySQLConnection(){};

    public static Connection newInstance(String user, String pass, String bbdd) throws SQLException{

        String cadenaConexion = driver + "://" + hostPuerto + "/" + bbdd;
        Connection con = DriverManager.getConnection( url: cadenaConexion, user, password: pass);
        return con;

    }

}
```

```java
package com.mycompany.garcia_cutillas_franciscojose_actut03;

import com.mycompany.conexion.MySQLConnection;
import java.sql.Connection;
import java.sql.SQLException;

/**
 *
 * @author Fran
 */
public class Garcia_Cutillas_FranciscoJose_ActUt03 {

    public static void main(String[] args) {

        Connection con = null;

        try {

            con = MySQLConnection.newInstance( user: "root",  pass: "3979199",  bbdd: "flota");
            System.out.println("Conectado: " + con.isValid( timeout: 3));

        } catch (SQLException ex) {

            System.out.println("Error al conectarse a la base de datos: " + ex.getMessage());

        }

    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.258 s
Finished at: 2023-12-06T19:28:04+01:00
------------------------------------------------------------------------
```

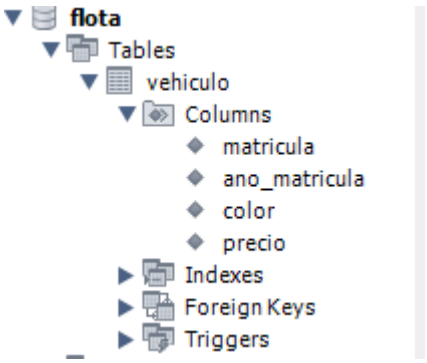
Ejercicio 2

Crea una tabla VEHICULO en la que se almacenarán los distintos vehículos que componen la flota de la empresa. Debemos almacenar su matrícula, año de matrícula, color y precio de compra. Elige los tipos de datos, así como la clave primaria y otras claves que tú consideres.

```java
package com.mycompany.operaciones;

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

/**
 *
 * @author Fran
 */
public class JDBCOperaciones {

    private JDBCOperaciones() {
    }

    ;

    public static int crearTabla(Connection con, String orden) {

        try {

            Statement st = con.createStatement();

            try {

                st.execute( sql: orden);
                return 0;

            } catch (SQLException ex) {

                if (ex.getMessage().contains( s: "already exists")) {

                    System.out.println( x: "La tabla ya existe");
                    return -1;

                } else {

                    System.out.println("Error al crear la tabla: " + ex.getMessage());
                    return -2;

                }

            }

        } catch (SQLException ex) {

            System.out.println("Error al crear el Statement: " + ex.getMessage());
            return -3;

        }

    }

}
```

```java
public class Garcia_Cutillas_FranciscoJose_ActUt03 {

    public static void main(String[] args) {

        Connection con = null;

        try {

            con = MySQLConnection.newInstance( user: "root",  pass: "3979199",  bbdd: "flota");
            System.out.println("Conectado: " + con.isValid( timeout: 3));

        } catch (SQLException ex) {

            System.out.println("Error al conectarse a la base de datos: " + ex.getMessage());

        }

        String tablaVehiculo = "create table vehiculo(\n" +
                               "        matricula varchar(10) primary key,\n" +
                               "    ano_matricula numeric(4) not null,\n" +
                               "    color varchar(50),\n" +
                               "    precio numeric(10,2)\n" +
                               ");";

        int operacion = JDBCOperaciones.crearTabla(con,  orden: tablaVehiculo);

        if (operacion == 0) {

            System.out.println( x: "Tabla creada correctamente");

        }

    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Tabla creada correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.573 s
Finished at: 2023-12-06T22:54:08+01:00
------------------------------------------------------------------------
```

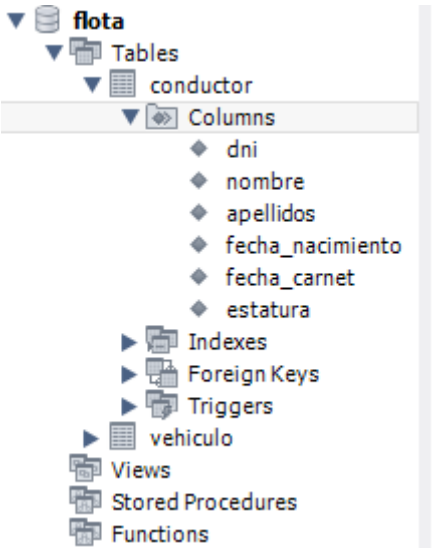


Ejercicio 3

Crea una tabla CONDUCTOR en la que se almacenarán los datos de los posibles conductores de vehículos. Esta tabla debe guardar la información del dni, nombre, apellidos y fecha de nacimiento de este, así como un par de campos más que tú consideres de interés. Elige los tipos de datos, así como la clave primaria y otras claves que tú consideres.

```
43
44          String tablaConductor = "create table conductor(\n" +
45                                  "    dni varchar(10) primary key,\n" +
46                                  "    nombre varchar(50) not null,\n" +
47                                  "    apellidos varchar(100) not null,\n" +
48                                  "    fecha_nacimiento date not null,\n" +
49                                  "    fecha_carnet date not null,\n" +
50                                  "    estatura numeric(3)\n" +
51                                  ");";
52
53          int operacionConductor = JDBCOperaciones.crearTabla(con,  orden: tablaConductor);
54
55          if (operacionConductor == 0) {
56
57              System.out.println( x: "Tabla conductor creada correctamente");
58
59          }
60
61      }
62  }
63
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Tabla conductor creada correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.577 s
Finished at: 2023-12-06T23:07:13+01:00
------------------------------------------------------------------------
```

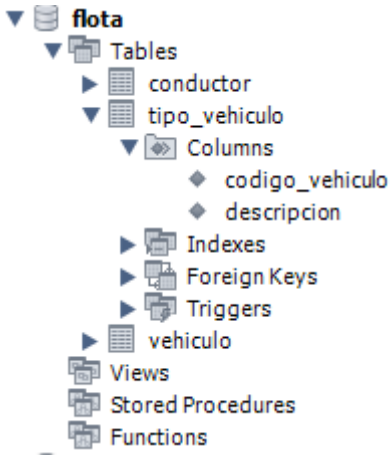


Ejercicio 4

Crea una tabla TIPO_VEHICULO que tendrá únicamente dos campos, un código de vehículo y la descripción del tipo de vehículo correspondiente.

```
60
        String tablaTipoVehiculo = "create table tipo_vehiculo(\n" +
62                                 "    codigo_vehiculo varchar(10) primary key,\n" +
63                                 "    descripcion varchar(100)\n" +
64                                 ");";
65
66      int operacionTipoVehiculo = JDBCOperaciones.crearTabla(con,  orden: tablaTipoVehiculo);
67
68      if (operacionTipoVehiculo == 0) {
69
70          System.out.println( x: "Tabla tipo vehiculo creada correctamente");
71
72      }
73
74  }
75 }
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Tabla tipo vehiculo creada correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.576 s
Finished at: 2023-12-06T23:14:39+01:00
------------------------------------------------------------------------
```



Ejercicio 5

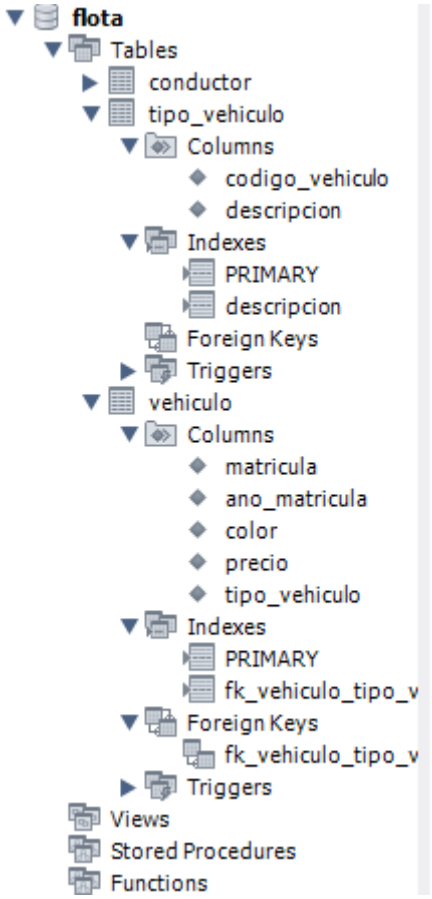
Modifica la tabla VEHICULO para añadir una columna tipo_vehiculo que será clave ajena a la tabla TIPO_VEHICULO.

- **No estoy pidiendo que al crear la tabla en el apartado A02 la crees ya con este campo, sino que ahora ejecutes el comando o comandos necesarios para añadir la columna a una tabla ya creada y crear esta clave ajena.**

```
public static boolean alterTable(Connection con, String orden){

    try{

        Statement st = con.createStatement();

        try{

            st.executeUpdate( sql: orden);
            return true;

        } catch (SQLException ex){

            System.out.println("Error al modificar la tabla: " + ex.getMessage());
            return false;

        }

    } catch (SQLException ex){

        System.out.println("Error al crear el Statement: " + ex.getMessage());
        return false;

    }

}
```

```
    //Para realizar esta operación, necesitamos hacerla en dos pasos:
    //Primero tenemos que añadir un índice a la tabla tipo_vehículo para poder referenciar la
    //foreign key
    String addIndex = "alter table tipo_vehiculo add index descripcion (descripcion);";

    boolean operacionAddTipoVehiculo = JDBCOperaciones.alterTable(con, orden: addIndex);

    if (operacionAddTipoVehiculo) {

        System.out.println( x: "Añadido indice correctamente");

    }

    //Segundo vamos a añadir la columna tipo_vehiculo y su foreign key hacia la tabla tipo_vehiculo
    String addTipoVehiculo = "alter table vehiculo add column tipo_vehiculo varchar(100) not null,\n" +
                             "      add constraint fk_vehiculo_tipo_vehiculo foreign key (tipo_vehiculo)\n" +
                             "      references tipo_vehiculo(descripcion) on delete restrict on update cascade;";

    operacionAddTipoVehiculo = JDBCOperaciones.alterTable(con, orden: addTipoVehiculo);

    if (operacionAddTipoVehiculo) {

        System.out.println( x: "Añadida columna tipo_vehiculo y su foreign key");

    }
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Añadido índice correctamente
Añadida columna tipo_vehiculo y su foreign key
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.566 s
Finished at: 2023-12-07T12:34:06+01:00
```



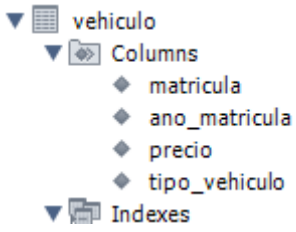
Ejercicio 6

Modifica la tabla VEHICULO para eliminar la columna color ya que posteriormente se ha decidido que no es necesario.

- **No estoy pidiendo que al crear la tabla en el apartado A02 la crees ya sin este campo, sino que ahora ejecutes el comando o comandos necesarios para eliminar la columna.**

```
String delColor = "alter table vehiculo drop column color;";

boolean operaciondelColor = JDBCOperaciones.alterTable(con, orden: delColor);

if (operaciondelColor) {

    System.out.println( x: "Columna color eliminada correctamente.");

}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Columna color eliminada correctamente.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.569 s
Finished at: 2023-12-07T12:40:05+01:00
------------------------------------------------------------------------
```



Ejercicio 7

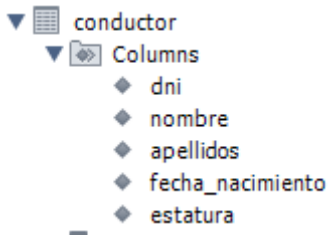
Modifica la tabla CONDUCTOR para añadir una columna fecha_carnet que contendrá la fecha en la que el conductor ha obtenido el carnet y que puede ser nula.

- **No estoy pidiendo que al crear la tabla en el apartado A03 la crees ya con este campo, sino**
- **que ahora ejecutes el comando o comandos necesarios para añadir la columna a una tabla ya creada con las condiciones indicadas.**

En mi caso voy a eliminar la columna y volver a crearla, puesto que en el A03 se exige crear dos campos a mi interés y da la casualidad de que ese campo lo había creado ya en la creación de la tabla.

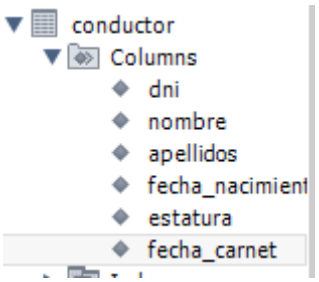
```
String delFechaCarnet = "alter table conductor drop column fecha_carnet;";

boolean operacionDelFechaCarnet = JDBCOperaciones.alterTable(con,  orden: delFechaCarnet);

if (operacionDelFechaCarnet) {

    System.out.println( x: "Columna fecha_carnet eliminada correctamente");

}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Columna fecha_carnet eliminada correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.561 s
Finished at: 2023-12-07T12:55:32+01:00
------------------------------------------------------------------------
```




```
String addFechaCarnet = "alter table conductor add column fecha_carnet date null;";

boolean operacionAddFechaCarnet = JDBCOperaciones.alterTable(con,  orden: addFechaCarnet);

if (operacionAddFechaCarnet) {

    System.out.println( x: "Columna fecha_carnet agregada correctamente");

}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03 ---
Conectado: true
Columna fecha_carnet agregada correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.569 s
Finished at: 2023-12-07T12:56:09+01:00
------------------------------------------------------------------------
```



Parte 2

Adjunto a la tarea tienes un script modulos.sql que te va a generar la base de datos modulos que va a contener información sobre módulos de ciclos formativos (misma tabla y datos que el script de instituto.sql). En netbeans debes realizar lo siguiente.

Ejercicio 1

Genera el patrón DAO e implementa el mismo para realizar las operaciones básicas que hemos visto en teoría (altaModulo, bajaModulo, consultaModulo, consultaTodos y eliminaModulo).

El caso de bajaModulo no se ha realizado porque necesitaríamos en la base de datos una tabla Alumno y otra Matrícula, para poder utilizar este método en el que quitaríamos de la tabla Matrícula, la matrícula de dicho alumno en un módulo.

```java
package com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.conexiones;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author Fran
 */
public class MySQLConnection {

    private static String driver = "jdbc:mysql";
    private static String hostPuerto = "localhost:3306";
    private static String user = "root";
    private static String pass = "3979199";
    private static String bbdd = "modulos";

    private MySQLConnection(){};

    public static Connection newInstance() {

        Connection con = null;

        try{

            String cadenaConexion = driver + "://" + hostPuerto + "/" + bbdd;
            con = DriverManager.getConnection( url: cadenaConexion, user,  password: pass);

        } catch (SQLException ex){

            System.out.println("Error al crear la conexión: " + ex.getMessage());

        }

        return con;

    }

    public static Connection newInstance(String user, String pass, String bbdd) {

        Connection con = null;

        try{

            String cadenaConexion = driver + "://" + hostPuerto + "/" + bbdd;
            con = DriverManager.getConnection( url: cadenaConexion, user,  password: pass);

        } catch (SQLException ex){

            System.out.println("Error al crear la conexión: " + ex.getMessage());

        }

        return con;

    }

```

```
    public static boolean cerrarConexion(Connection con){

        try{

            con.close();
            return true;

        } catch (SQLException ex){

            System.out.println("Error al cerrar la conexión: " + ex.getMessage());
            return false;

        }

    }

}

```

```
package com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.modelo;

/**
 *
 * @author Fran
 */
public class Modulo {

    private int codigo;
    private String nombre;
    private int horasSemanales;
    private int cursoImparte;

    public Modulo(int codigo, String nombre, int horasSemanales, int cursoImparte) {

        this.codigo = codigo;
        this.nombre = nombre;
        this.horasSemanales = horasSemanales;
        this.cursoImparte = cursoImparte;

    }

    public int getCodigo() {
        return codigo;
    }

    public void setCodigo(int codigo) {
        this.codigo = codigo;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public int getHorasSemanales() {
        return horasSemanales;
    }

    public void setHorasSemanales(int horasSemanales) {
        this.horasSemanales = horasSemanales;
    }

    public int getCursoImparte() {
        return cursoImparte;
    }

    public void setCursoImparte(int cursoImparte) {
        this.cursoImparte = cursoImparte;
    }

    @Override
    public String toString() {
        return "        Código = " + codigo + ", Nombre = " + nombre + ", Horas semanales = " + horasSemanales
                + ", Curso que imparte = " + cursoImparte;
    }

```

```java
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null) {
            return false;
        }
        if (getClass() != obj.getClass()) {
            return false;
        }
        final Modulo other = (Modulo) obj;
        return this.codigo == other.codigo;
    }

}
```

```java
package com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.dao;

import com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.modelo.Modulo;
import java.util.ArrayList;

/**
 *
 * @author Fran
 */
public interface ModuloDAO {

    int altaModulo(Modulo modulo);
    Modulo consultaModulo(int codigo);
    ArrayList<Modulo> consultaTodos();
    int eliminaModulo(int codigo);

}
```

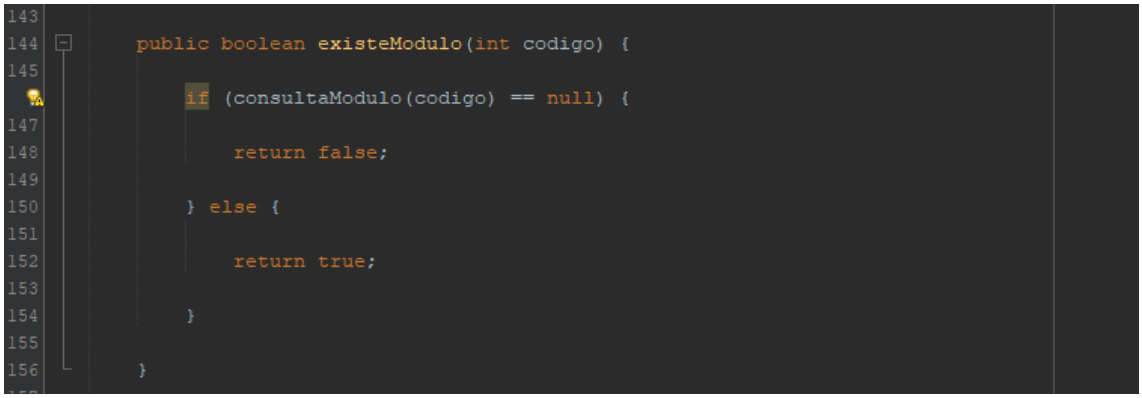
```java
package com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.dao;

import com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.conexiones.MySQLConnection;
import com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.modelo.Modulo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;

/**
 *
 * @author Fran
 */
public class ModuloDAOImpl implements ModuloDAO {

    @Override
    public int altaModulo(Modulo modulo) {

        Connection con = MySQLConnection.newInstance();

        String consulta = "insert into modulo (codigo, nombre, horas_semanales, curso_imparte) values (?,?,?,?);";

        int valorConsulta = 0;

        try {

            PreparedStatement ps = con.prepareStatement(sql: consulta);
            ps.setInt(parameterIndex: 1, x: modulo.getCodigo());
            ps.setString(parameterIndex: 2, x: modulo.getNombre());
            ps.setInt(parameterIndex: 3, x: modulo.getHorasSemanales());
            ps.setInt(parameterIndex: 4, x: modulo.getCursoImparte());
            valorConsulta = ps.executeUpdate();

        } catch (SQLException ex) {

            System.out.println("        Error al insertar el módulo: " + ex.getMessage());

        }

        MySQLConnection.cerrarConexion(con);

        return valorConsulta;

    }
```

```java
    @Override
    public Modulo consultaModulo(int codigo) {

        Connection con = MySQLConnection.newInstance();

        String consulta = "select * from modulo where codigo = ?;";

        Modulo m = null;

        try {

            PreparedStatement ps = con.prepareStatement( sql: consulta);
            ps.setInt( parameterIndex: 1,  x: codigo);
            ResultSet rs = ps.executeQuery();

            while (rs.next()) {

                m = new Modulo( codigo: rs.getInt( columnLabel: "codigo"),  nombre: rs.getString( columnLabel: "nombre"),
                        horasSemanales: rs.getInt( columnLabel: "horas_semanales"),  cursoImparte: rs.getInt( columnLabel: "curso_imparte"));

            }

        } catch (SQLException ex) {

            System.out.println("        Error al consultar el módulo: " + ex.getMessage());

        }

        MySQLConnection.cerrarConexion(con);

        return m;

    }

    @Override
    public ArrayList<Modulo> consultaTodos() {

        Connection con = MySQLConnection.newInstance();

        String consulta = "select * from modulo;";

        ArrayList<Modulo> modulos = new ArrayList<>();
        Modulo m = null;

        try {

            PreparedStatement ps = con.prepareStatement( sql: consulta);

            ResultSet rs = ps.executeQuery();

            while (rs.next()) {

                m = new Modulo( codigo: rs.getInt( columnLabel: "codigo"),  nombre: rs.getString( columnLabel: "nombre"),
                        horasSemanales: rs.getInt( columnLabel: "horas_semanales"),  cursoImparte: rs.getInt( columnLabel: "curso_imparte"));
                modulos.add( e: m);

            }

        } catch (SQLException ex) {

            System.out.println("        Error al consultar el módulo: " + ex.getMessage());

        }

        MySQLConnection.cerrarConexion(con);

        return modulos;

    }

    @Override
    public int eliminaModulo(int codigo) {

        Connection con = MySQLConnection.newInstance();

        String consulta = "delete from modulo where codigo = ?;";

        int valorConsulta = 0;

        try {

            PreparedStatement ps = con.prepareStatement( sql: consulta);
            ps.setInt( parameterIndex: 1,  x: codigo);
            valorConsulta = ps.executeUpdate();

        } catch (SQLException ex) {

            System.out.println("        Error al eliminar el módulo: " + ex.getMessage());

        }

        MySQLConnection.cerrarConexion(con);

        return valorConsulta;

    }
```

Ejercicio 2

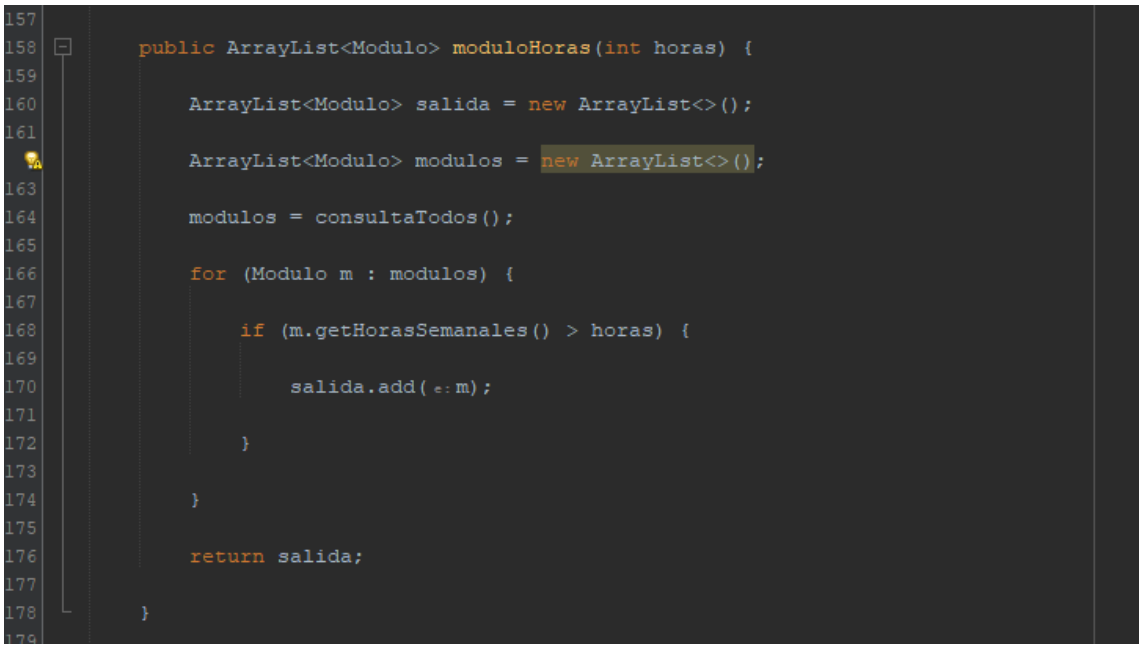
Además de estos, debes implementar las siguientes funcionalidades en el DAO.

- **Método que me indique si existe o no un módulo dado un código.**



```
143
144     public boolean existeModulo(int codigo) {
145
146         if (consultaModulo(codigo) == null) {
147
148             return false;
149
150         } else {
151
152             return true;
153
154         }
155
156     }
```

- **Método que me devuelvan todos los módulos que tengan más de x horas (x será un parámetro).**



```
157
158     public ArrayList<Modulo> moduloHoras(int horas) {
159
160         ArrayList<Modulo> salida = new ArrayList<>();
161
162         ArrayList<Modulo> modulos = new ArrayList<>();
163
164         modulos = consultaTodos();
165
166         for (Modulo m : modulos) {
167
168             if (m.getHorasSemanales() > horas) {
169
170                 salida.add( e: m);
171
172             }
173
174         }
175
176         return salida;
177
178     }
179
```

- **Método que me devuelva todos los módulos que se imparten en un determinado curso x (x será un parámetro).**

```
179
180     public ArrayList<Modulo> moduloCurso(int curso) {
181
182         ArrayList<Modulo> salida = new ArrayList<>();
183
            ArrayList<Modulo> modulos = new ArrayList<>();
185
186         modulos = consultaTodos();
187
188         for (Modulo m : modulos) {
189
190             if (m.getCursoImparte() == curso) {
191
192                 salida.add( e: m);
193
194             }
195
196         }
197
198         return salida;
199
200     }
201
```

Ejercicio 3

Para utilizar estos mecanismos de comunicación con la base de datos vamos a generar un programa principal a modo menú hasta que el usuario decida terminar marcando determinada opción. Por ejemplo, si el usuario pulsa la opción 1 podrá ver todos los módulos que tiene en base de datos, si pulsa la opción 2 le pedirá un código de módulo y le devolverá su información, etc. Se deben utilizar todos ellos pero no hace falta que sean directamente, por ejemplo, el de consultar un módulo puede utilizarse primero el de comprobar si existe o no el módulo y si existe, entonces mostrar la información correspondiente al mismo.

```java
package com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.aplicacion;

import com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.dao.ModuloDAOImpl;
import com.mycompany.garcia_cutillas_franciscojose_actut03_parte2.modelo.Modulo;
import java.util.ArrayList;
import java.util.Scanner;

/**
 *
 * @author Fran
 */
public class Garcia_Cutillas_FranciscoJose_ActUt03_parte2 {

    private static Scanner sc = new Scanner( source: System.in,  charsetName: "ISO-8859-1");
    private static int codigo;
    private static String nombre;
    private static int horas_semanales;
    private static int curso_imparte;
    private static ModuloDAOImpl mdi = new ModuloDAOImpl();
    private static ArrayList<Modulo> modulos = new ArrayList<>();

    public static void main(String[] args) {

        int seleccion = -1;

        //Instrucciones de la aplicación:
        /*
            Para finalizar pulsar 0
            Para dar de alta un módulo pulsar 1
            Para eliminar un módulo pulsar 2
            Para consultar un módulo pulsar 3
            Para consultar todos los módulos pulsar 4
            Para consultar módulos que tengan más de x horas pulsar 5
            Para consultar módulos que se imparten en un determinado curso pulsar 6
        */

        System.out.println( x: "***** Consultas a la base de datos modulos *****\n");
        System.out.println("    Para finalizar pulse 0\n" +
                           "    Para dar de alta un módulo pulse 1\n" +
                           "    Para eliminar un módulo pulse 2\n" +
                           "    Para consultar un módulo pulse 3\n" +
                           "    Para consultar todos los módulos pulse 4\n" +
                           "    Para consultar módulos que tengan más de x horas pulse 5\n" +
                           "    Para consultar módulos que se imparten en un determinado curso pulse 6\n");

        do {

            System.out.print( s: "    \nSeleccione un valor (0 - 6):");
            seleccion = sc.nextInt();
            sc.nextLine();

            switch (seleccion) {
                case 1:

                    altaModulo();
                    break;

                case 2:

                    eliminaModulo();
                    break;
```

```
                case 3:

                    consultaModulo();
                    break;

                case 4:

                    consultaModulos();
                    break;

                case 5:

                    consultaHoras();
                    break;

                case 6:

                    consultaCurso();
                    break;

                default:

                    if (seleccion != 0) {

                        System.out.println( x: "    Debe seleccionar un valor del rango preestablecido");

                    }

                    break;
            }

        } while (seleccion != 0);

    }

    public static void altaModulo(){

        System.out.println( x: "    *** Alta de módulo ***");

        System.out.print( s: "        Introduce el código del módulo: ");
        codigo = sc.nextInt();
        sc.nextLine();

        System.out.print( s: "        Introduce el nombre del módulo: ");
        nombre = sc.nextLine();

        System.out.print( s: "        Introduce las horas semanales del módulo: ");
        horas_semanales = sc.nextInt();
        sc.nextLine();

        System.out.print( s: "        Introduce el curso en el que se imparte el módulo: ");
        curso_imparte = sc.nextInt();
        sc.nextLine();

        Modulo m = new Modulo(codigo, nombre,  horasSemanales: horas_semanales,  cursoImparte: curso_imparte);

        if (mdi.altaModulo( modulo: m) != 0) {

            System.out.println( x: "        Módulo añadido");

        } else {

            System.out.println( x: "        No se ha añadido ningún módulo");

        }
```

```
    }

    public static void eliminaModulo(){

        System.out.println( x: "    *** Eliminar un módulo ***");

        System.out.print( s: "        Introduce el código del módulo: ");
        codigo = sc.nextInt();
        sc.nextLine();

        if (mdi.eliminaModulo(codigo) != 0) {

            System.out.println( x: "        Módulo eliminado");

        } else {

            System.out.println( x: "        No se ha eliminado ningún módulo");

        }

    }

    public static void consultaModulo(){

        System.out.println( x: "    *** Consultar un módulo ***");

        System.out.print( s: "        Introduce el código del módulo: ");
        codigo = sc.nextInt();
        sc.nextLine();

        if (mdi.existeModulo(codigo)) {

            Modulo m = mdi.consultaModulo(codigo);
            System.out.println( x: m.toString());

        } else {

            System.out.println("        El módulo con código " + codigo + " no se encuentra en la base de datos");

        }



    }

    public static void consultaModulos(){

        System.out.println( x: "    *** Consultar todos los módulos ***");

        modulos = mdi.consultaTodos();

        mostrarModulos(modulos);

    }

    public static void consultaHoras(){

        System.out.println( x: "    *** Consultar módulos de más de x horas ***");

        System.out.print( s: "        Introduce las horas: ");
        int horas = sc.nextInt();
        sc.nextLine();

        modulos = mdi.moduloHoras(horas);
```

```
        mostrarModulos(modulos);

    }

    public static void consultaCurso(){

        System.out.println( x: "    *** Consultar módulos por curso ***");

        System.out.print( s: "        Introduce el curso: ");
        int curso = sc.nextInt();
        sc.nextLine();

        modulos = mdi.moduloCurso(curso);

        mostrarModulos(modulos);

    }

    public static void mostrarModulos(ArrayList<Modulo> modulos){

        for(Modulo m : modulos){

            System.out.println( x: m.toString());

        }

    }

}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03_parte2 ---
***** Consultas a la base de datos modulos *****

    Para finalizar pulse 0
    Para dar de alta un módulo pulse 1
    Para eliminar un módulo pulse 2
    Para consultar un módulo pulse 3
    Para consultar todos los módulos pulse 4
    Para consultar módulos que tengan más de x horas pulse 5
    Para consultar módulos que se imparten en un determinado curso pulse 6


Seleccione un valor (0 - 6):0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  6.569 s
Finished at: 2023-12-07T20:32:44+01:00
------------------------------------------------------------------------
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Garcia_Cutillas_FranciscoJose_ActUt03_parte2 ---
***** Consultas a la base de datos modulos *****

    Para finalizar pulse 0
    Para dar de alta un módulo pulse 1
    Para eliminar un módulo pulse 2
    Para consultar un módulo pulse 3
    Para consultar todos los módulos pulse 4
    Para consultar módulos que tengan más de x horas pulse 5
    Para consultar módulos que se imparten en un determinado curso pulse 6


Seleccione un valor (0 - 6):1
    *** Alta de módulo ***
        Introduce el código del módulo: 6
        Introduce el nombre del módulo: Programación multimedia y dispositivos móviles
        Introduce las horas semanales del módulo: 5
        Introduce el curso en el que se imparte el módulo: 2
        Módulo añadido

Seleccione un valor (0 - 6):4
    *** Consultar todos los módulos ***
        Código = 1, Nombre = Programación, Horas semanales = 7, Curso que imparte = 1
        Código = 2, Nombre = Base de Datos, Horas semanales = 5, Curso que imparte = 1
        Código = 3, Nombre = Entornos de Desarrollo, Horas semanales = 3, Curso que imparte = 1
        Código = 4, Nombre = Acceso a Datos, Horas semanales = 5, Curso que imparte = 2
        Código = 5, Nombre = Lenguaje de Marcas, Horas semanales = 6, Curso que imparte = 2
        Código = 6, Nombre = Programación multimedia y dispositivos móviles, Horas semanales = 5, Curso que imparte = 2

Seleccione un valor (0 - 6):3
    *** Consultar un módulo ***
        Introduce el código del módulo: 3
        Código = 3, Nombre = Entornos de Desarrollo, Horas semanales = 3, Curso que imparte = 1

Seleccione un valor (0 - 6):2
    *** Eliminar un módulo ***
        Introduce el código del módulo: 2
        Módulo eliminado

Seleccione un valor (0 - 6):4
    *** Consultar todos los módulos ***
        Código = 1, Nombre = Programación, Horas semanales = 7, Curso que imparte = 1
        Código = 3, Nombre = Entornos de Desarrollo, Horas semanales = 3, Curso que imparte = 1
        Código = 4, Nombre = Acceso a Datos, Horas semanales = 5, Curso que imparte = 2
        Código = 5, Nombre = Lenguaje de Marcas, Horas semanales = 6, Curso que imparte = 2
        Código = 6, Nombre = Programación multimedia y dispositivos móviles, Horas semanales = 5, Curso que imparte = 2

Seleccione un valor (0 - 6):5
    *** Consultar módulos de más de x horas ***
        Introduce las horas: 5
        Código = 1, Nombre = Programación, Horas semanales = 7, Curso que imparte = 1
        Código = 5, Nombre = Lenguaje de Marcas, Horas semanales = 6, Curso que imparte = 2

Seleccione un valor (0 - 6):6
    *** Consultar módulos por curso ***
        Introduce el curso: 1
        Código = 1, Nombre = Programación, Horas semanales = 7, Curso que imparte = 1
        Código = 3, Nombre = Entornos de Desarrollo, Horas semanales = 3, Curso que imparte = 1
```

```
Seleccione un valor (0 - 6):2
    *** Eliminar un módulo ***
        Introduce el código del módulo: 7
        No se ha eliminado ningún módulo

Seleccione un valor (0 - 6):3
    *** Consultar un módulo ***
        Introduce el código del módulo: 2
        El módulo con código 2 no se encuentra en la base de datos

Seleccione un valor (0 - 6):1
    *** Alta de módulo ***
        Introduce el código del módulo: 3
        Introduce el nombre del módulo: Programación de servicios y procesos
        Introduce las horas semanales del módulo: 6
        Introduce el curso en el que se imparte el módulo: 2
        Error al insertar el módulo: Duplicate entry '3' for key 'modulo.PRIMARY'
        No se ha añadido ningún módulo

Seleccione un valor (0 - 6):0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  06:40 min
Finished at: 2023-12-07T20:39:52+01:00
------------------------------------------------------------------------
```