Gestión de bases de datos relacionales

```
public class Conexion {
    private static Connection conn;
    private static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    private static final String USER = "conexion";
    private static final String PASS = "";
    private static final String BD="fruteria";
    private static final String DB_URL = "jdbc:mysql://localhost:3306/"+BD;

    private static class ConexionPoseedor {
        public static final Conexion INSTANCE = new Conexion();
    }
    private Conexion()
    {
        try{
            Class.forName(JDBC_DRIVER);
            Properties properties = new Properties();
            properties.setProperty("user", USER);
            properties.setProperty("password",PASS);
            properties.setProperty("useSSL", "false");
            properties.setProperty("autoReconnect", "true");
            conn = (Connection)DriverManager.getConnection(DB_URL, properties);
            if(conn!=null){
                System.out.println("Conexión a la base de datos "+DB_URL+".....CORRECTA");
            }
        }
        catch(SQLException ex){
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("ErrorCode: " + ex.getErrorCode());
        }
        catch(ClassNotFoundException ex){
            System.out.println(ex.toString());
        }
    }
```

Para enviar sentencias SQL al controlador de la BD se utiliza el objeto **Statement**, suministrando el método SQL con la sentencia a ejecutar.

- Statement sentencia=conn.createStatement();

Para sentencias insert, delete, update, create o create table, el método a utilizar es **executeUpdate()**

- Statement sentencia= con.createStatement();
- sentencia.executeUpdate ("CREATE TABLE CAFES " +

"(CAFE_NAME VARCHAR(32), " + "SUP_ID INT, PRECIO FLOAT, " + "DESCUENTO INT, "+

"TOTAL INT)");

```
public void crearTablas(){
    try {
        Statement sentencia=conn.createStatement();
        // en String tabla codigo sql con el create table
        String tabla="create table ejemplo(\n" +
                     "  codigo smallint NOT NULL AUTO_INCREMENT,\n" +
                     "  producto varchar(25) NOT NULL,\n" +
                     "  cantidad int NOT NULL,\n" +
                     "  PRIMARY KEY (`codigo`)\n" +
                     ") ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8mb4 ; "
        sentencia.executeUpdate(tabla);

        System.out.println("Tablas creadas con éxito!!");
        sentencia.close();

    } catch (SQLException ex) {
        System.out.println("Error al ejecutar la sentencia");
    }

}
```

Cuando ejecutas insert, update y delete, que también se hace con executeUpdate(), devuelve el **número de filas afectadas**.

```
public void insertarDatos(){
    try {
        Statement sentencia=conn.createStatement();
        // dentro de executeUpdate codigo del insert, update o delete
        int resul=sentencia.executeUpdate("INSERT INTO ejemplo"
                + "(producto,cantidad)"
                + "VALUES ('melocotones',8),"
                + "('platanos',12),('peras',3);");
        System.out.println("Filas afectadas: "+resul);
        sentencia.close();
    } catch (SQLException ex) {
        System.out.println("Error en la inserción de datos");
    }
}
```

```
public void mostrarDatos() {
    Statement sentencia;
    try {
     sentencia = conn.createStatement();
     // dentro de executeQuery Codigo de la select
     ResultSet rs = sentencia.executeQuery("select codigo,producto,cantidad"
                + "from ejemplo");
     while (rs.next()) {
            //cada columna se indica, el tipo en el get, y que posicion o
            //que nombre tiene en el argumento
            System.out.print(rs.getInt(1) + " ");
            System.out.print(rs.getString("producto") + " ");
            System.out.println(rs.getInt(3));
        }
     rs.close();
     sentencia.close();
    } catch (SQLException ex) {
        System.out.println("Error en la consulta");
    }
}
```

JDBC devuelve los datos en un objeto **ResultSet**, donde se almacenarán los datos obtenidos de la consulta.

Para obtener cada uno de los datos recuperados de la consulta se usará el método **next**, que permitirá ir posicionándonos en cada una de las filas devueltas.

Con los métodos **getXXX** obtendremos cada uno de los campos de la fila.

El objeto ResultSet tiene entre otros los siguientes métodos:

- **next( )**;// accede al siguiente registro y devuelve false cuando no hay más registros
- **first( )**; // accede al primero
- **previous( )**;// al anterior
- **last( )**;// al último
- **getInt("campo") ó getInt(indice columna)**; devuelve el contenido numérico entero de la columna
- **getDouble("campo") ó getDouble(indice columna)**;// devuelve el contenido numérico double de la columna
- **getString(("campo") ó getString (indice columna)**;// devuelve el contenido de la cadena de la columna etc……
- **getObject (índice columna)**;//devuelve un objeto en general de cualquier tipo.

NOTA : los índices de columna siempre comienzan por 1.

Cuando se vayan a ejecutar varias consultas similares es más apropiado utilizar objetos **PreparedStatement** que hereda de la clase **Statement**, ya que se reducirá el tiempo de ejecución. Cuando se vaya a usar, basta con reemplazar los parámetros

```
try {

    System.out.println("Introduce la cantidad:");
    int cantidad = new Scanner(System.in).nextInt();

    //Consulta preparada
    String sql = "SELECT codigo,producto,cantidad from ejemplo "
            + "where cantidad > ? ";
    PreparedStatement ps = conn.prepareStatement(sql);
    // indico que para el primer parámetro el valor de la nota
    ps.setInt(1, cantidad);
    ResultSet rs = ps.executeQuery();

    System.out.println("Productos con cantidad > que : "
            + cantidad+ "\n");
    while (rs.next()) {
        System.out.println(rs.getString("producto")
                + " con cantidad: "
                + rs.getInt(3));
    }
    rs.close();
    conn.close();
} catch (SQLException ex) {
    LOG.log(Level.SEVERE, null, ex);
}
```