Actividad UT2_02: JDBC y MySQL. Base de datos de alumnos

1) Crear un proyecto en NetBeans, de nombre **UT2_ACT02_MYSQL_ALUMNOS**. Modificar la aplicación de la actividad UT2_01 para que funcione con MySQL.

   La tabla alumnos tiene los siguientes campos:

   a) *id*: identificador del alumno, de tipo entero, clave principal.
   b) *nombre*: nombre del alumno, de tipo TEXT, no puede ser NULL.
   c) *ciclo*: ciclo formativo en el que está matriculado, de tipo TEXT, no puede ser NULL.

   Los parámetros de la conexión a la base de datos (servidor, puerto, usuario, contraseña, nombre de la base de datos, el uso de Unicode y codificación UTF-8) se leen de un fichero. Hacer dos versiones:

   a) los datos de la conexión están en un archivo de texto con extensión ".properties", del que se leen y se recuperan mediante un objeto Properties.

   b) los datos se leen de un fichero binario (previamente debes crearte el fichero con los datos de la conexión).

2) Añadir el método **verInfoBD**, que recibe una conexión y muestra por pantalla la siguiente información de la base de datos: nombre y versión del sistema gestor de la base de datos, nombre y versión del driver, la URL de la conexión y el usuario conectado.

3) Añadir el método **verInfoTabla**, que recibe una conexión y el nombre de una tabla, y muestra su clave primaria y la información de todas sus columnas (nombre de la columna y nombre del tipo de datos).

4) En el método main, escribir el código necesario para, utilizando estos métodos, hacer las siguientes operaciones:

   1. Abrir la conexión a la base de datos
   2. Ver la información de la base de datos
   3. Crear la tabla de alumnos
   4. Ver la información de la tabla alumnos
   5. Realizar operaciones CRUD sobre la tabla de alumnos:
      - Guardar los datos de varios alumnos en la tabla, preguntando al usuario cuántos alumnos quiere guardar, o bien repitiendo la operación hasta que el usuario elija terminar. Los id de alumno tomarán valores consecutivos según se van insertando alumnos, empezando por el 1. A cada alumno nuevo se le asigna automáticamente el valor de id inmediatamente posterior al de la última fila en la tabla.
      - Mostrar los datos de todos los alumnos de la tabla.
      - Eliminar alumnos.
      - Actualizar datos de alumnos.

<u>RECORDATORIO DE LAS ESPECIFICACIONES DE LA ACTIVIDAD UT2_01</u>

- **private static Connection abreConexion (String url):** recibe una cadena de conexión a una base de datos y abre la conexión a la base de datos especificada.
- **private static void crearTabla (Connection con, String tname):** recibe una conexión y el nombre de una tabla y crea en la base de datos una tabla con ese nombre y los siguientes campos:
  - ***id*:** identificador del alumno, de tipo entero, clave principal.
  - ***nombre*:** nombre del alumno, de tipo TEXT, no puede ser NULL.
  - ***ciclo*:** ciclo formativo en el que está matriculado, de tipo TEXT, no puede ser NULL.

- **private static void insertarAlumno(Connection con, int id, String nombre, String ciclo, String tablename)**: recibe una conexión, los datos de un alumno y el nombre de una tabla. Inserta un alumno con los datos especificados en la tabla de nombre *tablename*. Utiliza el método executeUpdate() de la interfaz PreparedStatement, y sus métodos setXXX, para para parametrizar las sentencias SQL correspondientes.

- **private static ResultSet consultarTodosLosAlumnos(Connection con, String tablename):** recibe una conexión y el nombre de una tabla y devuelve un ResultSet con todos los alumnos de la tabla. Utiliza el método executeQuery() de la interfaz Statement.

- **private static void mostrarRegistrosAlumnos(ResultSet resul):** recibe un ResultSet con registros de la tabla alumno y muestra todos sus datos por consola con formato tabulado. Si el ResultSet está vacío se indica con un mensaje.

- **private static int ultimoIdAlumno(Connection con, String tablename):** recibe una conexión y el nombre de una tabla y devuelve el id del último alumno de la tabla (es el último id utilizado). Utiliza el método executeQuery de la interfaz Statement.

- **private static void eliminarAlumnoPorId(Connection conexion, String tablename):** recibe una conexión y el nombre de la tabla y borra un alumno con un id determinado, que se pedirá por teclado. Utiliza el método executeUpdate() de la interfaz PreparedStatement, y sus métodos setXXX para parametrizar las sentencias SQL correspondientes. Si no existe el id de alumno se indicará con un mensaje.

- **private static void modificarAlumnoPorId(Connection conexion, String tablename):** recibe una conexión y el nombre de la tabla y modifica los datos de un alumno con un id determinado, que se pedirá por teclado. Se piden los nuevos datos de nombre y ciclo del alumno. Utiliza PreparedStatement y sus métodos setXXX para parametrizar las sentencias SQL correspondientes. Si no existe el id de alumno se indicará con un mensaje.

  En todos los métodos se debe mostrar por consola un mensaje de éxito o error.

En el método main, escribir el código necesario para, utilizando estos métodos, hacer las siguientes operaciones:

6. Abrir la conexión a la base de datos
7. Crear la tabla de alumnos
8. Realizar operaciones CRUD sobre la tabla de alumnos:
   - Guardar los datos de varios alumnos en la tabla, preguntando al usuario cuántos alumnos quiere guardar, o bien repitiendo la operación hasta que el usuario elija terminar. Los id de alumno tomarán valores consecutivos según se van insertando alumnos, empezando por el 1. A cada alumno nuevo se le asigna automáticamente el valor de id inmediatamente posterior al de la última fila en la tabla.
   - Mostrar los datos de todos los alumnos de la tabla.
   - Eliminar alumnos.
   - Actualizar datos de alumnos.