Conexión Java con Derby

Apache Derby es un sistema gestor de base de datos relacional escrito en Java que puede ser empotrado en aplicaciones Java y utilizado para procesos de transacciones en línea. Tiene un tamaño de 2 MB de espacio en disco. Actualmente se distribuye como Sun Java DB.

Características

- APIs para JDBC y SQL.
- Su código pesa alrededor de 2MB comprimido.
- Soporta cifrado completo, roles y permisos. Además posee SQL SCHEMAS para separar la información en una única base de datos y control completo de usuarios.
- Soporta internamente procedures, cifrado y compresión.
- Trae soporte multilenguaje y localizaciones específicas.
- A partir de la versión 10.4 trae un sistema simple de replicación maestro-esclavo.
- Transacciones y recuperación ante errores ACID.
- Posee tres productos asociados a la marca:
    - Derby Embedded Database Engine: El motor propiamente dicho.
    - Derby Network Server: Permite convertir Derby en una base de datos que sigue el modelo cliente-servidor tradicional.
    - Database Utilities: Un paquete de utilidades.

Driver Apache Derby

Lo primero que haremos para establecer la conexión Java con Derby es el descargar la base de datos, que viene con el Driver incluido.

https://db.apache.org/derby/derby_downloads.html.

Conectarnos a Apache Derby

Lo siguiente será conectarnos a la base de datos. En este caso vamos a utilizar Apache Derby en memoria, por lo cual la cadena de conexión será:

```
"jdbc:derby:memory:myDB;create=true"
```

Vemos que en la cadena de conexión indicamos que la base de datos está e memoria. Ahora utilizamos el método `.getConnection()` para realizar la conexión mediante el `DriverManager`

```
String sURL = "jdbc:derby:memory:myDB;create=true";
con = DriverManager.getConnection(sURL);
```

Cargar la base de datos

Al ser una base de datos en memoria lo primero que tenemos que hacer es cargarla de datos, ya que no tendrá ninguno. Bueno, en realidad lo primero será crear la tabla para poder insertar los datos:

```
PreparedStatement tabla = con.prepareStatement("CREATE TABLE country (country
varchar(100) not null)");
tabla.execute();
```

Hemos utilizado la sentencia SQL `CREATE TABLE` para llevar a cabo nuestra tarea.

Ahora pasaremos a cargar los datos. En este caso nos hemos apoyado en un batch de inserciones.

```
Statement carga = con.createStatement();
carga.addBatch("INSERT INTO country VALUES ('Spain')");
carga.addBatch("INSERT INTO country VALUES ('France')");
carga.addBatch("INSERT INTO country VALUES ('United States')");
carga.addBatch("INSERT INTO country VALUES ('Brazil')");
carga.addBatch("INSERT INTO country VALUES ('Japan')");
carga.executeBatch();
```

La idea es definir un conjunto de sentencias de inserción mediante `INSERT` que ejecutaremos de forma batch mediante el método `.executeBatch()`

Consultar datos en Apache Derby con Java

Ahora que ya tenemos datos metidos en la base de datos podemos ejecutar una consulta sobre a tabla que hemos creado.

Así podríamos tener el siguiente código fuente:

```
try (PreparedStatement stmt = con.prepareStatement("SELECT country FROM
country")) {
  ResultSet rs = stmt.executeQuery();
  while (rs.next())
    System.out.println (rs.getString("country"));

} catch (SQLException sqle) {
  System.out.println("Error en la ejecución:"
    + sqle.getErrorCode() + " " + sqle.getMessage());
}
```