HIBERNATE: CRUD

Por Mikel Jorge y Eneko Martinez

Vamos crear de nuevo un CRUD sobre una base de datos, pero esta vez haremos uso de Hibernate. Crearemos las tablas que representen las entidades con las características requeridas a partir de los esquemas y análisis proporcionados previamente en la propia actividad.

Antes de empezar…

Debemos asegurarnos de haber creado previamente la base de datos a la cual accederá el programa. La base de datos podemos llamarla como nos guste, aunque va a gestionar ventas entre clientes y vendedores. La base de datos deberá por lo demás estar vacía.

Creación del proyecto

Partiremos de un proyecto Maven vacío. En el fichero *pom.xml* pondremos las dependencias que necesita Hibernate para funcionar con nuestra base de datos, o sea, Hibernate Core, Log4j Api y Core, y el conector de MySQL. El proyecto tendrá la estructura de directorios anteriormente vista para un CRUD, esto es, conexión, modelo, DAO y servicios, además del archivo de configuración en la carpeta de recursos.

¿Qué queremos que haga nuestro programa?

Nuestra aplicación va a crear en una base de datos vacía las tablas que representen a las entidades (y sólo a las entidades) de la base de datos requerida en los esquemas proporcionados previamente con la estructura y tipos de la base de datos del cliente. En esta actividad **no tendremos en cuenta las relaciones existentes entre tablas**. Por lo tanto, **no incluiremos** aquellos atributos que se creen para cumplir las relaciones.

Posteriormente, en el programa principal daremos opción, mediante un menú de texto, a probar las operaciones CRUD para cada una de las tablas que hayamos creado.

1. Información previa relativa a la base de datos del cliente

Nos proporcionan los esquemas de la base de datos que requiere el cliente.

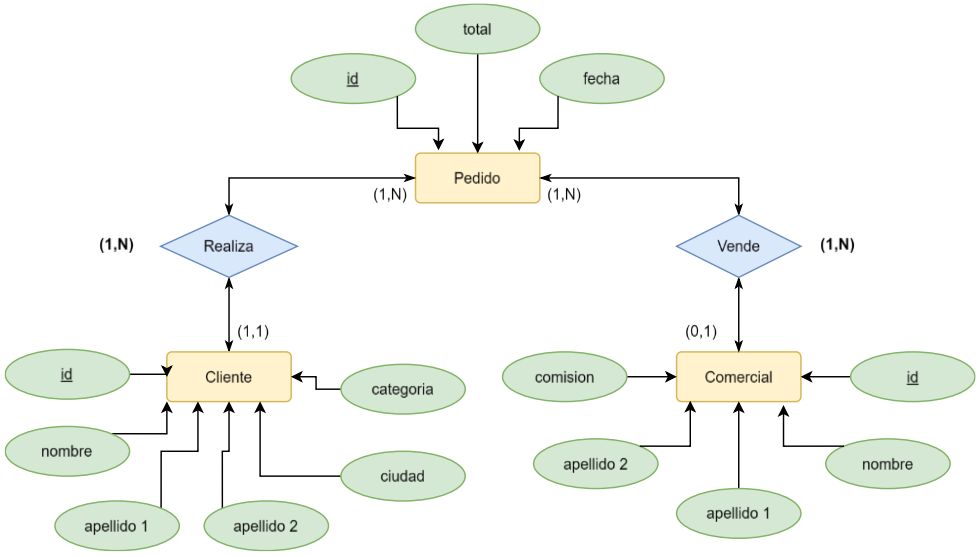

DIAGRAMA 1: CONCEPTUAL (E/R)

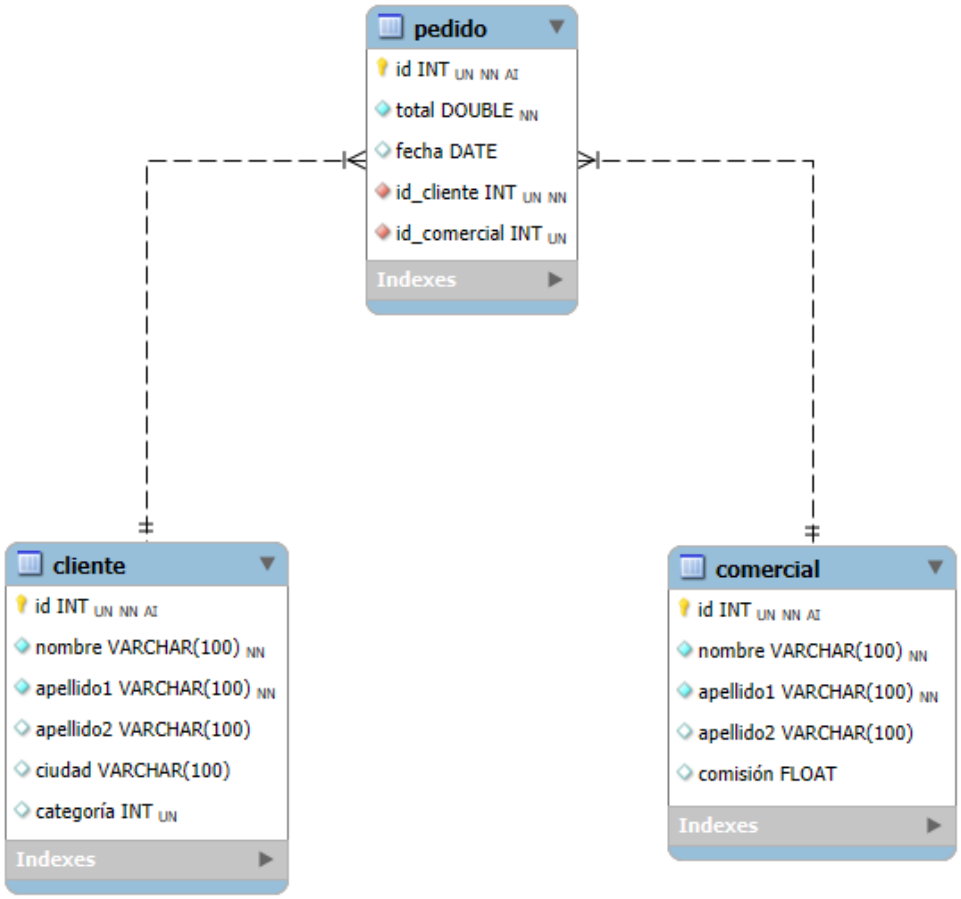

DIAGRAMA 2: RELACIONAL

***Nota**: 'UN' especifica un entero sin signo (UNSIGNED). Investiga de qué manera podríamos conseguir que nos lo añadiera a través de JPA.*

2. Clase de conexión y archivo de configuración de Hibernate

Tal como hemos visto en los apuntes, usaremos una clase para gestionar la conexión con la base de datos. Eso sí, tendremos que cambiar los parámetros adecuados dentro del archivo de configuración de Hibernate que debería estar en la carpeta *resources*.

3. Clases del modelo

Tendremos que crear las clases que modelen las tablas. Por lo tanto, en este caso tendremos 3 clases. Como no vamos a hacer las relaciones entre clases todavía, **no pongáis aquellos atributos que hagan referencia a otras tablas**.

Haremos uso de JPA para el mapeado de las clases (anotaciones). Revisa bien qué tipos necesitamos en la base de datos y su correspondencia en java. También debemos fijarnos en las restricciones de cada atributo.

Por último, recuerda añadir la entrada correspondiente para cada tabla en el archivo de configuración de Hibernate, en este caso indicando el mapeo de las clases.

4. Clases DAO

Crearemos una interfaz y una clase abstracta de implementación genérica que realice las operaciones CRUD. Posteriormente crearemos un archivo DAO específico para cada clase del modelo que extienda el DAO genérico, cada uno haciendo uso de la correspondiente clase modelo.

5. Clases de servicios

Tal como se ha visto en los apuntes, tendremos que crear una interfaz y una clase de implementación genérica de servicio que haga uso de los métodos DAO para las operaciones con la base de datos. Seguidamente crearemos una clase de servicio para cada una de las clases del modelo-DAO.

6. Programa principal

Crea un menú que nos permita elegir qué tabla queremos comprobar y los métodos necesarios que hagan uso de todas las clases de servicio para hacer una comprobación de funcionamiento (en manera similar a lo que aparece en los apuntes-presentación del tema).