

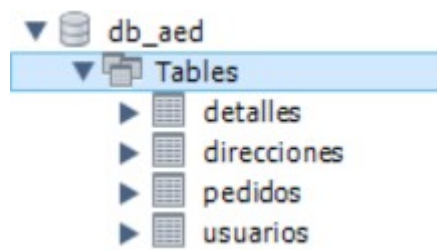
CRUD ORM

Indice

1. Introducción
2. Arquitectura del Sistema
3. Modelo de Datos
4. CRUD
5. Conclusiones

Introducción

En este proyecto he creado un sistema de 4 crud's que vinculan 4 tablas por medio de MySQL. Estas tablas son usuarios, pedidos, direcciones y detalles. Este proyecto hace uso de Java Spring, en mi caso usando Eclipse como IDE, e Ionic angular, así que estas tecnologías y todos los requisitos de estas son necesarias para hacer uso del sistema de Crud's



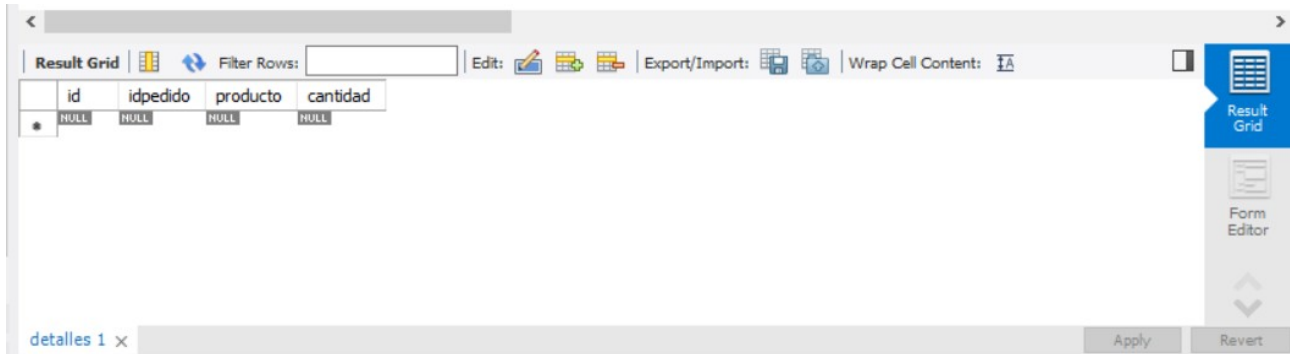
Arquitectura del sistema

El sistema representa una empresa de envíos en la cual el usuario tiene un nombre y correo, tiene vinculada su dirección uno a uno, en la cual se almacena su propia dirección y la ciudad en la que vive. Aparte, el usuario también está vinculado con su lista de pedidos como uno a muchos, en la cual se registra la fecha en la que tomó lugar el pedido (almacenado como DATE) y el estado en el que se encuentra dicha solicitud. A cada pedido se le asigna uno a uno una tabla de detalles, en la cual se registran qué producto es y la cantidad de este.

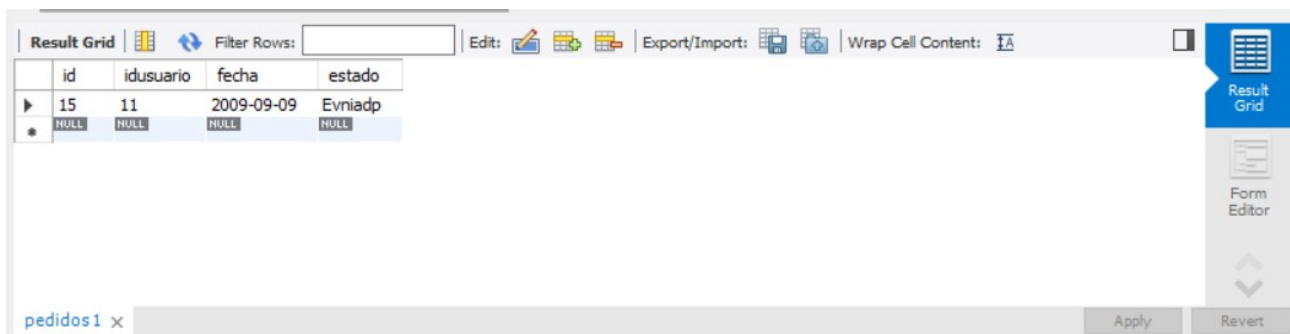
Algo a tener en cuenta es la necesidad de cambiar las credenciales de usuario de MySQL en el backend, dado que cada usuario puede tener las suyas propias. Para esto, solo hay que ir a `aedCrud\src\main\resources\application.properties` y cambiar los campos de `username` y `password`

```
aedCrud > db_aed > src > main > resources > application.properties
1  spring.datasource.url=jdbc:mysql://localhost/db_aed?useSSL=false&useJDBCCompliantTimezon
2  spring.datasource.username=root
3  spring.datasource.password=password
4  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
5  spring.jpa.hibernate.ddl-auto=none
6  logging.level.org.hibernate.SQL=debug
```

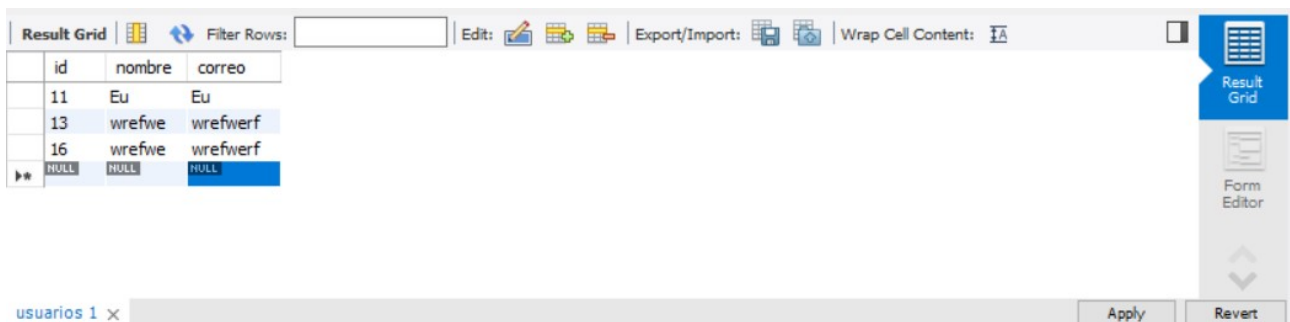
Modelo de Datos



id	idpedido	producto	cantidad
NULL	NULL	NULL	NULL



id	idusuario	fecha	estado
15	11	2009-09-09	Evniadp
NULL	NULL	NULL	NULL



id	nombre	correo
11	Eu	Eu
13	wrefwe	wrefwerf
16	wrefwe	wrefwerf
NULL	NULL	NULL

CRUD

El frontend está hecho con ionic, con una página por cada tabla entre las cuales se navega con el header. Los campos más problemáticos están validados para que se sepa cómo hacer input en caso de error, principalmente la fecha dado que se tiene que registrar como yyyy-mm-dd.

Usuarios

[DIRECCIONES](#)[PEDIDOS](#)[DETALLES](#)

ID	Nombre	Correo	Acciones
11	Eu	Eu	UPDATE DELETE
13	wrefwe	wrefwerf	UPDATE DELETE
16	wrefwe	wrefwerf	UPDATE DELETE

Nombre

Correo

CREAR

Para borrar cada entrada basta con dar click en delete, y todas las conexiones se borran en cascada. Para hacer update hay que escribir los datos en el formulario y clickar update en la entrada a actualizar. Para crear nuevas entradas basta con rellenar el formulario y darle click a crear.

Conclusiones

Este proyecto me ha permitido finalmente pulir mi capacidad de uso de ORMs en CRUDs dado que a pesar de no ser mi primer proyecto haciendo uso de estas, ni siquiera de spring específicamente, siempre me había visto forzado a recurrir a la prueba y error dado que estaba poco familiarizado con la tecnología, pero el haber trabajado este proyecto ha sido el último empujon que necesitaba, y he podido trabajar sin parones gracias a ello.