

UD6-UD7. Pt2 – Orientación a objetos y acceso a BD con PHP (2)

En esta práctica se piden algunas mejoras respecto a la aplicación desarrollada en la práctica UD7-Pt1.

Haz una copia del proyecto ud7_pt1. La llamamos por ejemplo ud7_pt2

1) Base de datos.

→ Id del cliente AUTO_INCREMENT.

- Tenemos que hacer algunas mejoras en la BDD. En primer lugar, el campo id de la tabla «cliente» debería ser autoincremental (se debería haber hecho en la Pt1).
- Hay que marcar el checkbox «A_I» en las propiedades del campo.

→ Foreign Key (FK).

- Crea una Foreign Key en la tabla «cuenta» de la Base de datos, que haga referencia a la tabla «cliente».
- Se puede hacer por entorno gráfico con phpMyAdmin.
- Hay que ir a la tabla cuenta → Estructura → Vista de relaciones
- La columna «cliente» tiene que ser una FK que haga referencia a la clave primaria (PK) de la tabla cliente.
- Haz que la FK tenga comportamiento ON DELETE RESTRICT ON UPDATE RESTRICT. De esta forma, no podremos borrar un cliente si existe alguna cuenta que le haga referencia (tampoco podremos modificar el id del cliente).
- Para comprobar el funcionamiento de la FK desde la aplicación, intenta insertar una cuenta con un valor del campo numérico «cliente» que no exista en la tabla cliente de la BDD. Tiene que salir un error de FK.
- También saldrá un error si intentamos modificar una cuenta existente para darle al campo «cliente» un valor inexistente en el id de la tabla cliente.
- Si no nos deja crear la FK o el Auto_increment, puede ser porque ya tenemos valores creados en la BDD que incumplen la restricción.
- Debes tener cuidado con el orden en que aplicas los cambios (Autoincrements y FK). No nos deja hacerlo en cualquier orden.

→ Atributos Unique (UQ).

- En este momento, las dos tablas tienen como clave primaria (PK) un id autoincremental.
- Podemos tener también otras claves alternativas, es decir campos que tienen un valor único para cada registro de la tabla, aunque no sean la PK. Esto, en servidores MySQL, se llama restricción Unique (UQ).
- Haz que el campo código de la tabla «cuenta» sea Unique.
- Para comprobarlo, intenta crear una nueva cuenta con el mismo código que una ya existente. Se debería producir un error de BDD.
- Haz también que la tabla «cliente» tenga una restricción Unique para los campos nombre y

apellidos (para la combinación de los dos campos, es decir 2 clientes no pueden tener el mismo nombre y apellidos).

→ Cuando cambiamos la estructura de la BDD, es conveniente generar un script que servirá como copia de seguridad.

- Genera el script de creación de la BDD por entorno gráfico a partir de phpMyAdmin (pestaña exportar).

2) Mostrar el nombre y apellidos de los clientes (listado de cuentas).

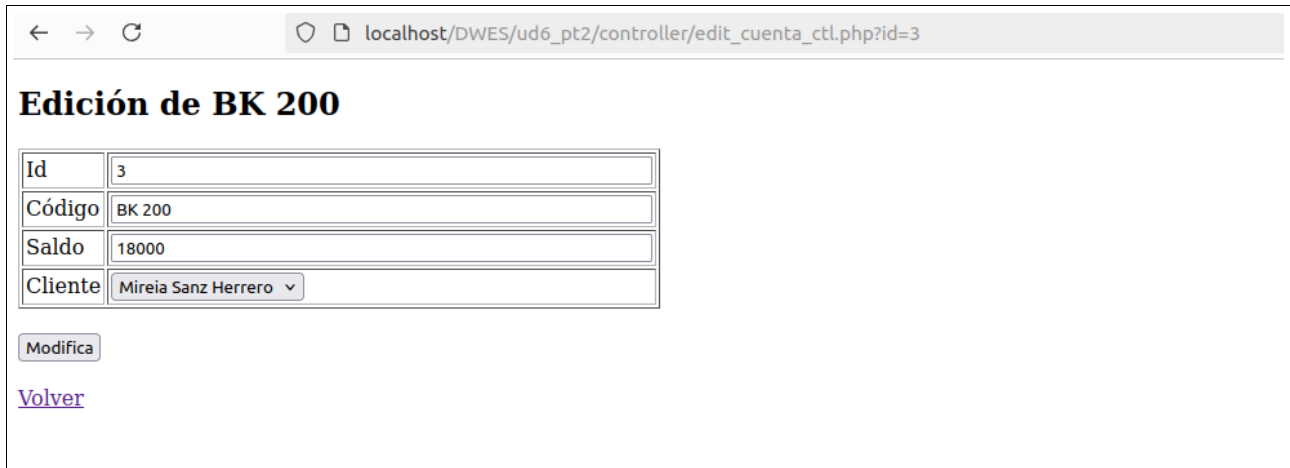
→ Es evidente que no resulta adecuado mostrar en las vistas de la cuenta el id del cliente referenciado.

→ En lugar de esto, se pide mostrar el nombre y apellidos del cliente en el listado de cuentas.

- Sugerencia: crear un método *getNombreApellidosCliente()* en la clase cuenta que retorne el concatenado del nombre y apellidos del cliente asociado a la cuenta, realizando la correspondiente consulta a la BDD con un objeto *clienteDAO*.

3) Mostrar el nombre y apellidos de los clientes (desplegable en los formularios).

→ También se pide mostrar un desplegable con el nombre y apellidos de los clientes en los formularios de inserción y edición de la cuenta.



← → ↻ localhost/DWES/ud6_pt2/controller/edit_cuenta_ctl.php?id=3

Edición de BK 200

Id	3
Código	BK 200
Saldo	18000
Cliente	Mireia Sanz Herrero ▾

[Volver](#)

→ Este desplegable (elemento **<select>** de HTML), se tendrá que cargar con un array de clientes, obtenido de una consulta a la BDD.

→ Habrá que hacer un foreach sobre este array, para mostrar las diferentes opciones del desplegable. Por cada **<option>** del desplegable, mostraremos el nombre y apellidos del cliente, y enviaremos al controlador su id (en caso de ser la opción seleccionada).

→ El valor por defecto en el desplegable lo definimos con el atributo «selected» de HTML:

- El formulario de edición tendrá como valor por defecto el cliente actual de la cuenta.

4) Fecha de nacimiento del cliente.

Se pide mejorar la presentación de la fecha de nacimiento del cliente (consulta la documentación de **php.net** para ver cómo hacerlo):

→ formato **día-mes-año** en la lista de clientes (en lugar del formato actual: año-mes-día).

Ayuda: puedes utilizar la clase *DateTime* de PHP.

- Haz también que en el formulario para insertar un nuevo cliente aparezca como valor por defecto la fecha de hoy.

5) Herencia de clases.

- Se propone simplificar la estructura de la capa persistence, haciendo que todas las clases DAO hereden de la clase db:

```
class cuentaDAO extends db
```

- Así, desde la clase DAO podemos acceder a los métodos *consulta()* y *close()*, sin necesidad de instanciar un objeto *db*
- Lo podemos hacer con la ayuda de la pseudo-variable *\$this* que hace referencia al objeto actual.

6) Inputs number.

- Por último, se pide una mejora en el HTML de los formularios de insertar / modificar una cuenta.
- El campo **saldo** de la cuenta es un integer en la BDD, pero en los formularios es un input de tipo text.
- Si escribimos en el formulario un valor del saldo con caracteres no numéricos, nos sale un mensaje de error de MySQL.
- Para evitarlo, cambiamos el tipo del input de «text» a «number». Así, en el formulario se valida que el usuario escriba un valor numérico.

Entrega de la práctica:

- script generado a partir de la BDD **bank**.
- carpeta comprimida con los ficheros de código fuente.