

**Curso: SEPE10-22: Desarrollo web BackEnd**

**Docente:** David Alcolea

## PLA 8\_ DETALLE Y DESCRIPCIÓN

**Nombre del PLA:** Operativa CRUD en una base de datos

### Objetivos del PLA:

- Realizar una operativa CRUD completa sobre una base de datos que nos permita realizar operaciones de alta, consulta, baja y modificación de usuarios utilizando un formulario
- Aprender los procedimientos de acceso a una base de datos MariaDB utilizando la librería mysqli en su paradigma de programación estructurada
- Trabajar con las instrucciones SQL nativas para realizar la operativa

### Competencias asociadas al PLA:

Competencias técnicas	Soft Skills
<ul style="list-style-type: none"> <li>• Realizar correctamente accesos CRUD una base de datos</li> <li>• Uso de la librería mysqli de php</li> <li>• Sentencias SQL nativas para realizar un CRUD</li> </ul>	<ul style="list-style-type: none"> <li>• Resolución de problemas</li> <li>• Interpretar requerimientos</li> <li>• Gestionar un proyecto</li> <li>• Búsqueda, gestión y uso de la información</li> </ul>

### Calendario y horas de dedicación:

Del 09/11/22 al 14/11/22

Síncronas	Webinars	4
	Tutorías presenciales	0
Asíncronas		16
Presenciales		
Totales		20

**Modalidad PLA:** Ejercicio independiente

### Instrucciones que recibirá el alumno para resolver el reto:

A partir de las especificaciones de un usuario y la base de datos que proporcionará el profesor, se confeccionará una plataforma de alta, consulta y mantenimiento de usuarios de una base de datos.

## Instrucciones metodológicas para la entrega de los retos:

Este PLA consta de una actividad consistente en la construcción de una plataforma para el alta, consulta y mantenimiento de usuarios de la base de datos '**banco**' suministrada por el profesor



NIF	Nombre	Apellidos
55545678P	Beatrix	Kiddo's
44608800L	David	Alcolea
45334433T	Gogo	Yubari
44608811L	Joana	O'Donnell
40022199L	Josephine	Delacrua
22345678J	O-Ren	Ishii
99911158P	Peter	Clemenza
40000299L	Pierre	Delacroix
32555678P	Vernita	Green
12345678K	Virgil	Solozzo

El formato de entrega de la actividad es el siguiente:

**Pla8\_tu\_nombre\_tu\_apellido.zip**

La entrega del fichero será a través de la aplicación Moodle del curso.

Las actividades son individuales y intransferibles.

La resolución de dudas o comentarios se podrán realizar a través de las herramientas de comunicación de la aplicación Moodle del curso


La actividad se entregará antes de la fecha límite de entrega.

## Requerimientos de usuario (PLA 08: Operativa CRUD sobre una base de datos)

Se necesita crear una plataforma para realizar una operativa de alta, consulta, modificación y baja de usuarios de la base de datos **banco** y, concretamente de la tabla de **personas**

### Operativa de alta de personas

Al informar los datos del formulario que se adjunta como recurso del PLA y pulsar el botón de '**Alta**', se deberá crear un registro en la tabla **personas** de la base de datos **banco**, teniendo en cuenta las siguientes especificaciones:



Formulario de alta de personas con los siguientes campos:

NIF	40000005Z
Nombre	Apollonia
Apellidos	Corleone
Dirección	Foscarella avenue, 45
Teléfono	
Email	apolonia@mail.com

Botones: Alta, Modificación, Baja, Limpiar

- Todos los datos excepto teléfono serán obligatorios de forma que, si alguno de ellos se encuentra sin informar, se deberá informar la incidencia con un mensaje de error
- Por temas de usabilidad se mostrarán todos los errores de una sola vez tal como se puede ver en el ejemplo siguiente



Mensaje de error de validación:

Alta Modificación Baja Limpiar

Nif obligatorio  
Nombre obligatorio  
Apellidos obligatorios  
Dirección obligatoria  
Email obligatorio

- En caso de producirse un error se mantendrán los datos que ya se hayan informado del formulario
- Se validará que el NIF a asignar a la persona no se encuentre ya en la base de datos, de ser así, se especificará de forma expresa mediante un mensaje específico



Formulario de alta de personas con el siguiente mensaje de error:

NIF	55545678P
Nombre	Beatrix
Apellidos	Kiddo's
Dirección	Margheritti street, 101
Teléfono	222311888
Email	beatrix@mail.com

Botones: Alta, Modificación, Baja, Limpiar

El nif ya existe en la base de datos

- Si el alta es correcta se mostrará un mensaje de 'Alta efectuada'

### Operativa de consulta de todas de personas

Al entrar en la plataforma se deberá mostrar debajo del formulario una tabla con todas las personas que existan en la tabla **personas** de la base de datos y con las siguientes especificaciones:

NIF	Nombre	Apellidos
55545678P	Beatrix	Kiddo's
44608800L	David	Alcolea
45334433T	Gogo	Yubari
44608811L	Joana	O'Donnell
40022199L	Josephine	Delacruz

- Se mostrará una persona en cada fila con los datos nif, nombre y apellidos exclusivamente
- Se mostrarán ordenados por nombre y apellidos
- Si no hubiera datos en la tabla se mostrará un mensaje de 'No hay datos' y se mostrará la tabla vacía

No hay datos

Alta Modificación Baja Limpiar

NIF	Nombre
-----	--------

- Al realizar un alta, una modificación de persona o una baja la tabla se actualizará de forma automática sin que el usuario tenga que refrescar la página con F5

### Operativa de consulta de una persona seleccionada en la tabla

A partir de la tabla de personas, cuando el usuario pulse sobre una de las filas, se consultará en la base de datos el detalle de la persona seleccionada y se mostrarán los datos de la persona en el formulario superior para poder modificarlos o darlos de baja más adelante

NIF	Nombre
55545678P	Beatrix
44608800L	David
45334433T	Gogo
44608811L	Joana
40022199L	Josephine
22345678J	O-Ren
99911158P	Peter
40000299L	Pierre
32555678P	Vernita
12345678K	Virgil

NIF

22345678J

Nombre

O-Ren

Apellidos

Ishli

Dirección

Foscarelli avenue, 45

Teléfono

Email

oren@mail.com

Alta Modificación Baja Limpiar

- Al pulsar sobre una fila de la tabla se consultará en la tabla personas los datos de la persona seleccionada utilizando la clave primaria de la tabla
- Los datos de la persona consultada se mostrarán en el formulario

## Operativa de modificación de la persona seleccionada

Una vez hemos consultado los datos de la persona seleccionada en la tabla y, al pulsar el botón de modificar, se trasladará a la base de datos las modificaciones que hayamos realizado en el formulario:

Formulario de modificación de una persona seleccionada. El formulario contiene los siguientes campos:

- NIF: 22345678J
- Nombre: O-Ren
- Apellidos: Ishii
- Dirección: Foscarelli avenue, 45
- Teléfono: (campo vacío)
- Email: oren@mail.com

Debajo de los campos hay cuatro botones: Alta, Modificación (destacado con un recuadro naranja), Baja y Limpiar.

- Se validará que, si se pulsa el botón de modificar, hayamos seleccionado previamente una persona de la tabla

Mensaje de error: Se debe seleccionar una persona válida. Debajo del mensaje hay cuatro botones: Alta, Modificación, Baja y Limpiar.

- Al igual que en la operativa de alta, todos los datos excepto teléfono serán obligatorios de forma que, si alguno de ellos se encuentra sin informar, se deberá informar la incidencia con un mensaje de error
- Por temas de usabilidad se mostrarán todos los errores de una sola vez

Mensaje de error: Nif obligatorio, Nombre obligatorio, Apellidos obligatorios, Dirección obligatoria, Email obligatorio. Debajo del mensaje hay cuatro botones: Alta, Modificación, Baja y Limpiar.

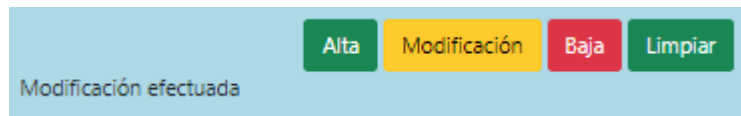
- En caso de producirse un error se mantendrán los datos que ya se hayan informado del formulario
- Si modificamos el nif se validará que éste no se encuentre ya en la base de datos, de ser así, se especificará de forma expresa mediante un mensaje específico

Mensaje de error: El nif ya existe en la base de datos. Debajo del mensaje hay cuatro botones: Alta, Modificación, Baja y Limpiar.

- Si, por el motivo que sea, la persona no existe en la base de datos o bien pulsamos el botón de modificar sin haber modificado ningún dato mostraremos un mensaje de error

Mensaje de error: El cliente no existe en la base de datos o no se han modificado datos. Debajo del mensaje hay cuatro botones: Alta, Modificación, Baja y Limpiar.

- Si la modificación es correcta se mostrará un mensaje de 'Modificación efectuada'



### Operativa de baja de la persona seleccionada

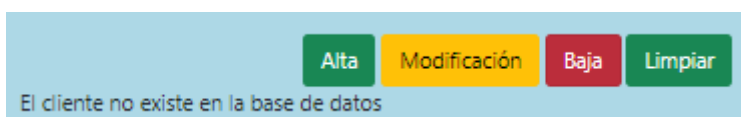
Una vez hemos consultado los datos de la persona seleccionada en la tabla y, al pulsar el botón de baja, borraremos la persona en la base de datos:

A form with a light blue background and a green border. It contains the following fields: NIF (44608800L), Nombre (David), Apellidos (Alcolea), Dirección (Gran Via, 77), Teléfono (338998477), and Email (david@mail.com). Below the fields are four buttons: 'Alta' (green), 'Modificación' (yellow), 'Baja' (red, highlighted with a red border), and 'Limpiar' (green). The text 'Modificación efectuada' is at the bottom left.

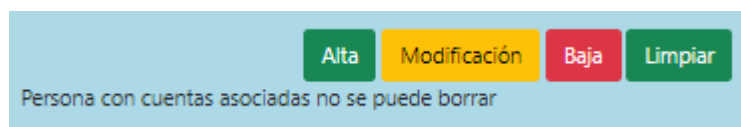
- Se validará que, si se pulsa el botón de baja, hayamos seleccionado previamente una persona de la tabla



- Si, por el motivo que sea, la persona no existe en la base de datos



- Tendremos que validar, si el cliente tiene cuentas asignadas en la tabla de cuentas y, de ser así, mostraremos un mensaje de error avisando que el cliente no puede borrarse



- Si la baja es correcta se mostrará un mensaje de 'Baja efectuada'

## Especificaciones técnicas

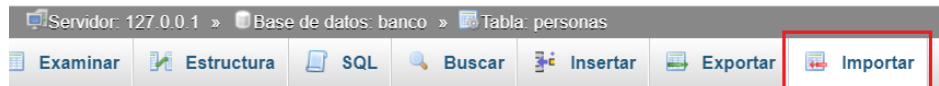
Se enumeran a continuación una serie de requisitos de confección de la plataforma que se deben cumplir para realizar correctamente el ejercicio.

- Se recomienda organizar el código utilizando funciones para el alta, baja, consulta y modificación que tendremos en ficheros externos que incorporaremos en el programa principal ***index.php*** utilizando **require** o **include**
- El tratamiento de errores lo haremos con **try...catch**. En este ejercicio utilizaremos una estructura de tratamiento de errores para cada una de las operativas
- Por temas de usabilidad se recomienda que todos los errores de validación aparezcan de una sola vez
- Se deberá tener en cuenta en todos los datos de tipo alfanumérico que pueden venir informados con apóstrofes y tendremos, por tanto, que escaparlos para evitar que se produzca un error en la sentencia SQL o, peor aun, que se produzca inyección de código SQL
- Sería interesante que, mediante algún mecanismo, los botones de baja y modificación solo se activen en el momento de seleccionar una persona de la tabla y se desactiven al realizar una baja o un alta
- Para realizar la consulta de la persona seleccionada en la tabla veréis un formulario oculto al final del documento **html** y un **script** que ya se proporciona, no obstante, podéis utilizar cualquier otro sistema que os resulte más cómodo o que hayáis descubierto

## Explicación técnica del Reto (PLA 08: Operativa CRUD sobre una base de datos)

### EJERCICIO 0: CARGA DE LA BASE DE DATOS

El primer paso será cargar la base de datos **banco.sql** que encontraréis en los recursos del PLA. Para ello pulsamos sobre la opción de importar de phpMyAdmin, seleccionamos el archivo de base de datos y pulsamos en continuar.



### Importando en la tabla "personas"

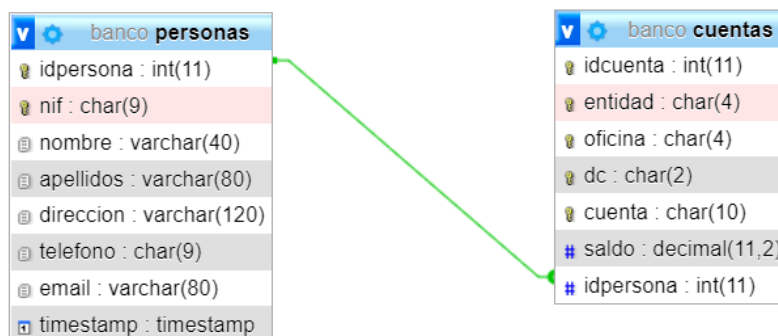
#### Archivo a importar:

El archivo puede ser comprimido (gzip, bzip2, zip) o descomprimido.

Un archivo comprimido tiene que terminar en **[formato].[compresión]**. Por ejemplo: **.sql.zip**

Buscar en su ordenador:  Ningún archi... seleccionado (Máximo: 40MB)

Una vez cargada la base de datos podemos ver su estructura en la vista de diseñador



### EJERCICIO 1: ALTA DE PERSONA

Al pulsar el botón de alta recuperaremos los datos del formulario, los validaremos y daremos de alta el registro en la tabla personas siguiendo los siguientes pasos:

- Inicializar todas las variables php que vamos a utilizar después en el documento html para mantener los datos en el formulario en caso de error y mostrar los diferentes mensajes.

Para ello recordad que tendremos que incluir en los atributos **value** de los controles del formulario el **echo** de la variable php

- Verificar con **isset** si hemos pulsado el botón de alta y, de ser así, incorporaremos toda la operativa de alta de persona (recuperar datos, validarlos, y confeccionar la sentencia SQL)
- Recuperar los datos utilizando **addslashes()** si utilizáis **\$\_POST** o **FILTER\_SANITIZE\_ADD\_SLASHES** si utilizáis **filter\_input()** para evitar la inyección de código

```
$nombre = addslashes($_POST['nombre']);
```

```
$nombre = filter_input(INPUT_POST, 'nombre', FILTER_SANITIZE_ADD_SLASHES);
```



- Validar todos los datos obligatorios y, en caso de que exista algún error de validación, incorporar los textos de error en una variable para lanzar la excepción al final de las validaciones y así mostrar todas a la vez. Ejemplo

```
if (empty($direccion)) {  
    $errores .= "Dirección obligatoria<br>";  
}  
if (empty($email)) {  
    $errores .= "Email obligatorio<br>";  
}  
if (!empty($errores)) {  
    throw new Exception($errores); //aquí lanzamos la excepción si $errores no está vacía  
}
```

- Para que el código quede más ordenado mi sugerencia es que utilizéis funciones para todas las operativas, Por ejemplo, para el alta podríais estructurar el código de la siguiente forma:

```
if (isset($_POST['alta'])) {  
    //recuperar los datos del formulario  
    try {  
        //dentro de la función validaremos los datos y damos de alta el registro en la bbdd  
        altaPersona($nif, $nombre, $apellidos, $direccion, $telefono, $email);  
        $mensajes = 'Alta efectuada';  
    } catch (Exception $e) {  
        $mensajes = $e->getMessage();  
    }  
}
```

- Si utilizáis funciones recordad que las variables que necesitéis dentro de éstas tendréis que pasarlas como parámetros de entrada (ver ejemplo anterior para `altaPersona`) o bien incorporarlas utilizando `global`

Mi consejo es que incorporéis el fichero de conexión en la base de datos solo una vez dentro del flujo principal del programa y, dentro de cada función utilizéis `global $conexionBanco` para poder utilizar el la variable con el objeto de conexión

- Confeccionar la sentencia `INSERT` y ejecutarla con la función `mysqli_query`. Podéis controlar si el registro ya existe (el nif, que es clave única, se encuentra ya en la base de datos) utilizando

```
if (!mysqli_query($conexionBanco, $sql)) {  
    if ($conexionBanco->errno == 1062) {  
        throw new Exception("El nif ya existe en la base de datos");  
    }  
    //texto del error, código de error  
    throw new Exception($conexionBanco->error, $conexionBanco->errno);  
}
```

## EJERCICIO 2: CONSULTA DE TODAS LAS PERSONAS

Al entrar en la plataforma se consultarán todas las personas de la base de datos siguiendo los siguientes pasos:

- Confeccionar la sentencia **SELECT** para recuperar todos los atributos sin olvidar incorporar la cláusula **ORDER\_BY** para que las filas aparezcan ordenadas por nombre y apellidos
- Recordad que las consultas es la única operativa para la que tenemos que recoger de forma expresa el resultado que nos entrega la base de datos (que serán las filas de la tabla personas)

**\$objetoDatos = mysqli\_query(\$conexionBanco, \$sql)**

Si observamos este objeto veremos que no es utilizable directamente ya que se trata de un objeto especial mysqli del cual tendremos que extraer los datos que realmente necesitamos

```
mysqli_result Object ( [current_field] => 0 [field_count] => 8 [lengths] => [num_rows] => 6 [type] => 0 )
```

- Fijaos que, del objeto anterior, podemos extraer el número de filas que nos devuelve la consulta utilizando el atributo **num\_rows** de forma que, si necesitamos conocer si la consulta nos devuelve alguna fila podemos utilizar:

**if (\$objetoDatos->num\_rows == 0) { ... }**

- Para extraer los datos del objeto anterior tendremos que ejecutar una función mysqli adicional:

**\$personas = mysqli\_fetch\_all(\$objetoDatos, MYSQLI\_ASSOC);**

Que nos devolverá un array asociativo que ya podemos utilizar

```
Array
(
    [0] => Array
        (
            [idpersona] => 74
            [nif] => 32345678P
            [nombre] => Beatrix
            [apellidos] => Kiddo
            [direccion] => Margheritti street, 101
            [telefono] => 222311888
            [email] => beatrix@mail.com
            [timestamp] => 2021-10-06 18:01:01
        )
    [1] => Array
        (
            [idpersona] => 36
            [nif] => 44608800L
            [nombre] => David
            [apellidos] => Alcolea
            [direccion] => Gran Via, 77
            [telefono] => 338888477
            [email] => david@mail.com
            [timestamp] => 2021-10-06 17:08:47
        )
)
```

- Por último, con el array anterior, confeccionaremos las etiquetas **<tr>** de la tabla del documento html y con la siguiente estructura:

**<tr data-id='idpersona'><td>Nif</td><td>Nombre</td><td>Apellidos</td></tr>**

donde idpersona, nif, nombre y apellidos son los datos correspondientes de cada una de las personas del array

Observad que en la etiqueta `tr` hemos utilizado un atributo `data-id` que informaremos con el valor de la columna `idpersona` y que utilizaremos después para activar la consulta de la persona seleccionada en la tabla

### EJERCICIO 3: CONSULTA DE LA PERSONA SELECCIONADA

Al seleccionar una persona de la tabla tendremos que enviar una consulta al servidor con el id de la persona seleccionada para traernos el detalle de la persona que trasladaremos al formulario para su edición:

Este ejercicio consta de dos partes:

- Activación de la selección de una fila de la tabla de personas en el documento html
- Realización de la consulta de la persona seleccionada en el servidor para informar el formulario

#### Selección de la persona

Esta operativa ya se encuentra completa en los recursos del PLA ya que es exclusivamente JavaScript. No obstante, vamos a revisarla para entender su funcionamiento

- Confeccionamos un pequeño formulario con un solo control que, además no será visible ya que se trata de un formulario técnico de uso interno. En este formulario informaremos el id de la persona a consultar para poder enviarlo al servidor más adelante

```
<form id='formconsulta' method='post' action='#'>  
  <input type='hidden' id='consulta' name='consulta'>  
</form>
```

- Necesitamos detectar cuando el usuario pulsa sobre una etiqueta `<td>` para poder buscar el id de la persona de la fila a la que pertenece la celda y trasladarlo al formulario oculto anterior

- Activamos un listener sobre la propia tabla

```
document.querySelector('table#listapersonas').onclick = function(ev) { ... }
```

- Cada vez que pulsemos sobre un elemento de la tabla (cualquier etiqueta que se encuentre dentro de ésta) se ejecutará la función anónima asociada al evento `onclick`. No obstante nos interesa solo cuando el usuario pulsa sobre una etiqueta `<td>` y para detectar esta etiqueta utilizamos al siguiente expresión:

```
document.querySelector('table#listapersonas').onclick = function(ev) {  
  //comprobar cuando hemos pulsado sobre una etiqueta <td>  
  if (ev.target.nodeName == 'TD') {  
    consultaPersona(ev.target)  
  }  
}
```

- Sólo cuando el usuario pulsa sobre una etiqueta TD se ejecutará la función `consultaPersona` pasando como parámetro el objeto que se crea automáticamente cada vez que javascript detecta un evento (en este caso es el evento `onclick`)

- La función `consultaPersona` realiza las siguientes operativas

Localizar la etiqueta `<tr>` de la fila que contiene la celda pulsada

```
let tr = td.closest('tr')
```

Recuperar el atributo `data-id` de la `<tr>`

```
let idpersona = tr.getAttribute('data-id')
```

Trasladar el id al formulario oculto

```
document.querySelector('#consulta').value = idpersona
```

Submit del formulario oculto para enviar el id al servidor

```
document.querySelector('#formconsulta').submit()
```

### Consulta de la persona en el servidor

Una vez hemos seleccionado una persona de la tabla y hemos realizado el submit del formulario oculto con JavaScript tendremos que preguntar en el servidor si nos llega el id de la persona para consultar sus datos

- Preguntaremos si nos llega el índice 'consulta' con `if (isset($_POST['consulta'])) {...}`
- Recuperamos el id de la persona a consultar (nos llega en `$_POST['consulta']`) y validamos que se encuentre informado, sea numérico y tenga un valor mayor que cero (no existen claves primarias que sean menores o iguales a cero)
- Realizamos la consulta en la base de datos con `SELECT` pero, esta vez, para extraer los datos no necesitamos un array de dos dimensiones ya que solo obtendremos una fila (o ninguna si el id no existe). Para ello utilizaremos:

```
$personas = mysqli_fetch_assoc($objetoDatos);
```

```
Array
(
    [idpersona] => 74
    [nif] => 32345678P
    [nombre] => Beatrix
    [apellidos] => Kiddo
    [direccion] => Margheritti street, 101
    [telefono] => 222311888
    [email] => beatrix@mail.com
    [timestamp] => 2021-10-06 18:01:01
)
```

NIF	32345678P
Nombre	Beatrix
Apellidos	Kiddo
Dirección	Margheritti street, 101
Teléfono	222311888
email	beatrix@mail.com

- Como vemos en un array asociativo de una sola dimensión que utilizaremos para informar el formulario con los datos de la persona seleccionada

## EJERCICIO 4: MODIFICACIÓN DE PERSONA

Una vez hemos realizado la consulta de una persona seleccionada en la tabla, al pulsar el botón de modificar recuperaremos los datos del formulario, los validaremos y trasladaremos la modificación al registro en la tabla personas que corresponda con el id de la persona:

- Verificar con `isset` si hemos pulsado el botón de modificación y, de ser así, incorporaremos toda la operativa de modificación de persona (recuperar datos, validarlos, y confeccionar la sentencia SQL)
- Al igual que en el alta recuperar los datos utilizando `addslashes()` si utilizáis `$_POST` o `FILTER_SANITIZE_ADD_SLASHES` si utilizáis `filter_input()` para evitar la inyección de código
- Validar que el id de la persona llegue informado, sea numérico y su valor sea mayor o igual a uno (para evitar errores posteriores en la ejecución de la sentencia SQL)
- Validar todos los datos obligatorios y, en caso de que exista algún error de validación, incorporar los textos de error en una variable para lanzar la excepción al final de las validaciones y así mostrar todas a la vez.
- Confeccionar la sentencia `UPDATE` y ejecutarla con la función `mysqli_query`. Recordad controlar si el registro ya existe (el nif, que es clave única, se encuentra ya en la base de datos) para indicar esta circunstancia al usuario (ver apartado de alta persona)
- Como validación adicional podemos comprobar que efectivamente el usuario ha modificado algún dato en el formulario utilizando

```
if ($conexionBanco->affected_rows == 0) {...}
```

Si intentamos modificar una fila en la tabla y el sistema gestor detecta que no se cambia ninguno de los valores de los atributos de la tabla, no se llega a realizar la modificación. En este caso no parece lógico informar al usuario con un mensaje de *'Modificación efectuada'* sino que sería más aconsejable mostrar *'No se han modificado datos'*

Pero, ¿y si intentamos modificar una fila que no existe? Pues que el sistema gestor no nos va a devolver ningún error y la única forma de detectarlo es precisamente utilizando el mismo atributo `affected_rows`. Podemos especificar entonces como mensaje de error:

*'Persona no existe o No se han modificado datos'*

## EJERCICIO 5: BAJA DE PERSONA

Una vez hemos realizado la consulta de una persona seleccionada en la tabla, al pulsar el botón de baja recuperaremos el id de la persona mostrada y la daremos de baja en la tabla personas:

- Verificar con `isset` si hemos pulsado el botón de baja y, de ser así, incorporaremos toda la operativa de baja de persona (recuperar id, validarlo, y confeccionar la sentencia SQL)
- Validar que el id de la persona llegue informado, sea numérico y su valor sea mayor o igual a uno (para evitar errores posteriores en la ejecución de la sentencia SQL)
- Confeccionar la sentencia `DELETE` y ejecutarla con la función `mysqli_query`.
- En el caso de la baja tenemos que tener en cuenta la restricción de clave foránea que existe entre la tabla personas y la tabla cuentas y que va a impedir que se borre una persona que tenga cuentas asociadas.

Por ejemplo si intentamos borrar la siguiente persona con una cuenta

idpersona	nif	nombre	apellidos
36	44608800L	David	Alcolea

idcuenta	entidad	oficina	dc	cuenta	saldo	idpersona
3	0010	0100	01	1234567890	200.00	36

Deberíamos informar al usuario de esta circunstancia

Persona con cuentas asociadas no se puede borrar

Tendremos que controlar si el sistema gestor nos devuelve el código 1451

```
if ($conexionBanco->errno == 1451) {  
    throw new Exception("Persona con cuentas asociadas no se puede borrar", 20);  
}
```

- Validaremos, al igual que en la modificación, que el id de la persona exista en la base de datos. En caso que el id no exista el atributo `$conexionBanco->affected_rows` devolverá 0

## Recursos de aprendizaje vinculados:

Toda la documentación (manuales, vídeo tutoriales, esquemas, etc) se encontrarán en el aula virtual así como los recursos necesarios para resolver el reto (imágenes, etc)

- Manual de instrucciones SQL para realizar un CRUD utilizando mysqli
- Base de datos *banco.sql*