

TAREA - PDO







Mantenimiento de una Base de Datos MySQL utilizando Php Data Objects (PDO)

Consiste en crear una página web con php que muestre los datos de los clientes existentes en una base de datos, la cual debéis crear para este fin (**siguiendo todos por favor el esquema que os indico más adelante**). Se ha de dar la opción al usuario de la web de **ver, crear, modificar y eliminar** clientes de nuestra Base de Datos.

Cuando entremos a la página principal, veremos algo similar a esto:

MANTENIMIENTO DE CLIENTES

con PDO y usando POO

DNI	Nombre	Dirección	Localidad	Provincia	Teléfono	e-mail	Editar	Borrar
11222777R	Luisa Gómez	Avda. del Rail, 23	Almansa	Albacete	768443322	luisa@hotmail.com		
12345678X	Juana Maria	Illueca 2	Elche	Alacant	977654354	Maria@mimail.com		
22333444B	Juan Jose de la Era	Calle Río Túrria 23	Aspe	Alacant	444555344	juan@hotmail.com		

Nuevo Cliente

Una posible orientación puede consistir en crear los siguientes archivos (o algo similar):

- **funciones.php** ⇒ donde crearemos al menos dos funciones, una para conectarnos a la base de datos (y que devuelva la referencia a la base de datos), y otra para desconectarnos.

No existe un método explícito para **cerrar una conexión** PDO porque, al terminar el script o cuando el objeto PDO deja de estar en uso, la conexión se cierra automáticamente. Sin embargo, si deseas cerrar la conexión de manera explícita antes de que termine el script, puedes hacerlo estableciendo el objeto PDO a null.

Sobre cerrar al conexión manualmente:

- **Establecer \$pdo = null:** Esto libera los recursos asociados con el objeto PDO y cierra la conexión.
- **Finalización del script:** Si no haces esto manualmente, PHP cerrará automáticamente la conexión al finalizar la ejecución del script.
- **Uso del bloque finally:** Es una buena práctica colocar la liberación de recursos en un bloque finally para asegurarte de que siempre se ejecute, incluso si ocurre una excepción.

Esto garantiza que la conexión sea cerrada explícita y oportunamente si necesitas liberar recursos antes de que finalice el script.

```

<?php
try {
    // Crear una nueva conexión PDO
    $pdo = new PDO(.....);

    // Usar la conexión
    $query = $pdo->query('SELECT * FROM tabla');
    foreach ($query as $fila) {
        print_r($fila);
    }


} catch (PDOException $e) {
    echo 'Error: ' . $e->getMessage();
} finally {
    // Cerrar la conexión
    $pdo = null; // Esto cierra explícitamente la conexión
}
?>

```

- **index.php** ⇒ script que mostrará todos los clientes de la base de datos de manera similar a la imagen de arriba, utilizando las funciones creadas en funciones.php para conectarnos a la base de datos. Como veis, hay iconos para editar/borrar cada uno de los clientes, así como un botón para añadir nuevos clientes a nuestra base de datos.
- **clientenuevo.php** ⇒ Llegamos aquí cuando pinchamos en el botón *Nuevo Cliente* del script index.php. Se nos mostrará un formulario para la introducción de los datos del cliente. Este script se debe llamar a si mismo, y aquí se validarán los datos en php. Si todo es correcto, insertamos el nuevo cliente en la base de datos y redirijo a index.php (con lo que ya se verá el nuevo cliente insertado). Si algo falla en la inserción, sigo mostrando el formulario (ojo, dejando los datos correctos y marcando, por ejemplo, los campos incorrectos en rojo). Comprobad al menos que se ha introducido algo en el campo nombre y correo, y que el DNI tiene formato correcto (ocho cifras y una letra al final). Y ojo que no admitimos dni repetidos. Mejores y más exactas comprobaciones las dejo como mejora optativa.
- **editarcliente.php** ⇒ Llegamos aquí si en index.php se pincha sobre el icono *editar* de alguno de los clientes. Mostrará un formulario similar al de *cliente nuevo*, pero es esta ocasión ya con los datos del cliente que se ha seleccionado. Para conseguirlo, a este script le llegará un dato (y sólomente uno) a través de url (query-string): el identificador del cliente sobre el que hemos pulsado la opción *editar*. En este caso será el dni del cliente, que además coincide con la clave primaria de nuestra tabla. A partir de ese dato, el script extraerá el resto de información que necesitamos de la base de datos. Se nos permitirá cambiar los valores del formulario, actualizando la base de datos, después de validarlos. Serán las mismas validaciones que en alta, excepto *dni*, que al ser el identificador y clave primaria de la tabla, no permitiremos que se modifique. Podemos mostrarlo desactivado, por ejemplo...
- **borrarcliente.php** ⇒ Si se elige la opción *borrar* en index.php, llamaremos a este script, en el cual, como siempre que queramos eliminar algo de la BBDD, se ha de pedir confirmación al usuario para que confirme el borrado. Si se confirma, se elimina el cliente. Se confirme o se cancele el borrado, se redirige a index.php (donde además, si se ha procedido al borrado, nos deberá aparecer un mensaje indicando que el cliente de nombre XXX se ha eliminado correctamente (si es que realmente ha sido así).

Seguro que se os ocurren formas de mejorar este “sistema”, o plantearlo de forma diferente. El ejercicio está abierto a mejoras, siempre que se cumpla con la funcionalidad del ejercicio. Comentádmelas en un archivo de texto adjunto.

Cread **tod@s** una base de Datos llamada **clientes_DB**, en ella una sola tabla llamada **clientes**. Un **usuario** llamado **jefe** (contraseña **jefe**) con privilegios para manipular los datos de esa base de datos). Este será el usuario que utilizarán vuestros scripts php. Olvidaos de root, ahora y siempre, para este tipo de accesos desde web. Y la tabla creadla siguiendo este esquema de la imagen.

Nombre	Tipo
dni 	varchar(9)
nombre	varchar(30)
direccion	varchar(50)
localidad	varchar(30)
provincia	varchar(30)
telefono	varchar(9)
email	varchar(30)

Orientaciones/Requisitos:

- A la hora de leer los registros de la base de datos, **debemos hacer un mapeado con nuestra clase cliente** (que debéis crear, claro). **Haced** (además de la función que nos devuelva la conexión, etc) **una función que os devuelva** un array con los clientes de la base de datos, donde cada posición del array será un cliente (un objeto tipo cliente). Así sólo hay que llamar a esa función desde la página principal, y ya tendréis un array con todos los clientes, para poder mostrarlos por pantalla, tal como se pide.
- Os recuerdo: siempre que creamos una clase, preferentemente crearemos todos los **atributos** como **privados** e implementamos, aunque no se pida expresamente, los **set** y los **get públicos**, para acceder a dichos atributos desde fuera de la clase.
- Como siempre, hemos de comprobar si las **acciones se han ejecutado correctamente, y actuar en consecuencia**. Por ejemplo: Si borramos un registro, ¿se ha borrado realmente alguna fila?
- Además hay que **controlar los posibles errores** de pdo con **try-catch**. Por ejemplo: ¿la conexión ha sido exitosa?
- Excepto en la función que nos devuelva todos los clientes de la BBDD, cuya consulta sql no necesita parámetros, en el resto de consultas se ha de utilizar siempre **consultas preparadas**.

La entrega ha de incluir en lo siguiente:

- los scripts de vuestro sitio web que habéis creado.
- scripts para la creación de la base de datos, e inserción de algún registro.
- comentadme cualquier mejora o modificación que hayáis estimado oportuno realizar.
- la corrección se realizará en clase de manera personalizada.