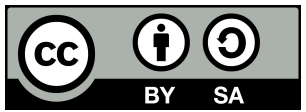
CURSO .PHP

ACCESO A DATOS

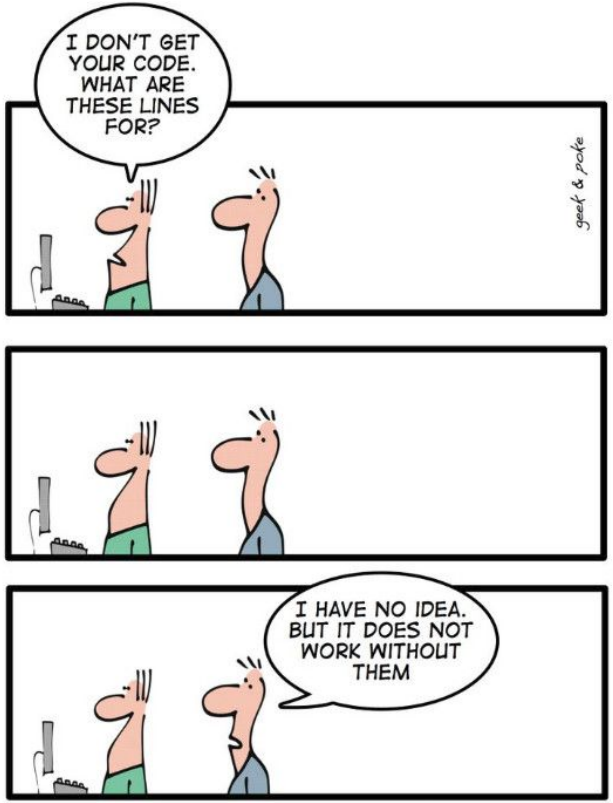


Autor: Jon Vadillo

www.jonvadillo.com

Contenidos

- Ejecutar sentencias SQL
- Establecer conexión
- Preparar la sentencia
- Cerrar conexión
- Tratar los resultados



Author: Geek and Poke

Ejecutar sentencias SQL

1. **Establecer conexión** con la base de datos
2. **Preparar la sentencia**
3. **Ejecutar sentencia** (asociando parámetros si fuese necesario)
4. Opcional: **tratar el resultado** de la sentencia ejecutada.

Establecer conexión

```
function connect(){
    try {
        # MS SQL Server
        $dbh= new PDO("mssql:host=$host;dbname=$dbname, $user,
$pass");

        # MySQL
        $dbh= new PDO("mysql:host=$host;dbname=$dbname", $user,
$pass);

        # SQLite Database
        $dbh= new PDO("sqlite:my/database/path/database.db" );
    }
    catch(PDOException $e) {
        echo $e->getMessage();
    }
}
```

Preparar la sentencia

```php
$stmt = $dbh->prepare("SELECT nombre, apellidos FROM alumnos" );

// Incluir parámetros con la sintaxis :nombre
$stmt = $dbh->prepare("
     SELECT nombre, apellidos
     FROM alumnos
     WHERE edad > :edad" );

// Otro ejemplo de un INSERT
$stmt= $dbh->prepare("
     INSERT INTO alumnos(nombre, apellidos)
     values (:nombre, :apellidos)" );
```

Ejecutar la sentencia

```php
// Ejemplo simple sin parámetros
$stmt = $dbh->prepare("SELECT nombre, apellidos FROM alumnos");
$stmt->execute();

// Ejemplo con parámetros
$data = array( 'nombre' => 'Mikel', 'edad' => 15 );
$stmt = $dbh->prepare("
     SELECT nombre, apellidos
     FROM alumnos
     WHERE nombre = :nombre AND edad = :edad");
$stmt->execute($data);
```

Cerrar conexión

```
function close(){
    /**
     * Una conexión a base de datos con PDO
     * permanecerá abierta mientras exista
     * el objeto PDO creado
     * */
    $dbh = null;
}
```

Tratamiento de resultados

- Una vez ejecutada la sentencia **`execute()`** podremos acceder a los resultados obtenidos de la base de datos.
- PDO ofrece la posibilidad de recibir los resultados en distintos formatos: Para indicar el modo se utiliza el método **`setFetchMode(String mode)`**:
  - PDO::FETCH_ASSOC: devuelve un **array indexado** por nombre de columna
  - PDO::FETCH_CLASS: Asigna los valores de las columnas a las **propiedades de la clase indicada**.
  - PDO::FETCH_OBJ: devuelve **objetos anónimos** que tendrán como propiedades las columnas obtenidas.

fetch()

- Una vez indicado el cómo queremos los datos, utilizaremos el método `fetch()` para acceder a la información.
- El método `fetch()` obtiene **la siguiente fila** de un conjunto de resultados, por lo que se deberá iterar por los resultados.

FETCH_ASSOC

```php
$data = array( 'nombre' => 'Mikel', 'edad' => 15 );
$stmt = $dbh->prepare("
   SELECT nombre, apellidos FROM alumnos
   WHERE nombre = :nombre AND edad = :edad" );
// Establecemos el modo en el que queremos recibir los datos
// Este tipo de fetch crea un array asociativo, indexado por el
nombre de la columna.
$stmt->setFetchMode(PDO::FETCH_ASSOC);
// Ejecutamos la sentencia
$stmt->execute($data);
// Mostramos los resultados obtenidos
while($row = $stmt->fetch()) {
   echo $row['nombre'] . "\n";
   echo $row['apellidos'] . "\n";
   echo $row['edad '] . "\n";
}
```

FETCH_OBJ

```php
$data = array( 'nombre' => 'Mikel', 'edad' => 15 );
$stmt = $dbh->prepare("
    SELECT nombre, apellidos FROM alumnos
    WHERE nombre = :nombre AND edad = :edad" );
// Establecemos el modo en el que queremos recibir los datos
$stmt->setFetchMode(PDO::FETCH_OBJ);
// Ejecutamos la sentencia
$stmt->execute($data);

// Mostramos los resultados obtenidos
while($row = $stmt->fetch()) {
    echo $row->nombre . "\n";
    echo $row->apellidos . "\n";
    echo $row->edad . "\n";
}
```

FETCH_CLASS

```php
class Alumno {

    public $nombre;
    public $apellidos;
    public $edad;
    public $otraInformacion;

    function __construct($otraInformacion= '') {
        // El constructor se ejecutará después de asociar los valores obtenidos de la base de datos al objeto. Por lo tanto, podemos tratar esos valores dentro del constructor.
        $this->nombre = strtoupper($this->nombre);
        $this->otraInformacion = $otraInformacion;
    }
}
```

FETCH_CLASS

```php
$data = array( 'nombre' => 'Mikel', 'edad' => 15 );
$stmt = $dbh->prepare("SELECT nombre, apellidos FROM alumnos WHERE
nombre = :nombre AND edad = :edad" );
// Establecemos el modo en el que queremos recibir los datos
$stmt->setFetchMode(PDO::FETCH_CLASS, 'Alumno');
// Ejecutamos la sentencia
$stmt->execute($data);

// Mostramos los resultados
while($obj = $stmt->fetch()) {
    echo $obj->addr;
}
```

Método abreviado query()

- En consultas que no reciban parámetros, podemos utilizar el método abreviado query() el cual ejecutará la sentencia y nos devolverá el conjunto de resultados directamente.
- Es decir, **no es necesario hacer la operación en 2 pasos (prepare() y execute())** como hacíamos hasta ahora.

```php
$stmt = $dbh->query('
    SELECT nombre, apellidos, edad
    FROM empleado');

// Establecemos el modo en el que queremos
recibir los datos
$stmt->setFetchMode(PDO::FETCH_ASSOC);

while($row = $stmt->fetch()) {
    echo $row['nombre'] . "\n";
    echo $row['apellidos'] . "\n";
    echo $row['edad '] . "\n";
}
```

fetchObject()

- Alternativa al método fetch() la cual devolverá los resultados como objetos anónimos (PDO::FETCH_OBJ) u objetos de la clase indicada (PDO::FETCH_CLASS).

```php
$stmt = $dbh->query('
    SELECT nombre, apellidos, edad
    FROM empleado');

while($persona = $stmt
->fetchObject())
{
   echo $persona->nombre;
   echo $persona->apellido;
}
```

```php
$stmt = $dbh->query('
    SELECT nombre, apellidos, edad
    FROM empleado');

while($persona = $stmt
->fetchObject('Alumno')) {
   echo $persona->nombre;
   echo $persona->apellido;
}
```

fetchAll()

- A diferencia del método fetch(), fetchAll() trae todos los datos de golpe, sin abrir ningún puntero, almacenándolos en un array.
- Se recomienda cuando no se esperan demasiados resultados que podrían provocar problemas de memoria.

```php
// $resultado contendrá un array asociativo con todos los datos
$resultado = $stmt->fetchAll(PDO::FETCH_ASSOC);

// Para leer las filas podemos recorrer el array y acceder a la información.
foreach ($resultado as $row){
    echo $row["nombre"]." ".$row["apellido"].PHP_EOL;
}
```

Hands on!

- Lista de la compra: crea una aplicación que muestre una lista de la compra almacenada en base de datos. La tabla de base de datos únicamente tendrá dos columnas, una con el ID y otra un VARCHAR con el texto (será el nombre del producto).
- Añade a la aplicación anterior un formulario para introducir nuevos productos en la lista.
- Añade a la aplicación anterior un enlace a cada producto de la lista para que se pueda eliminar de la lista.

Sources

- Github jvadillo: https://github.com/jvadillo/guia-php-pdo
- WikiBooks PHP: https://en.wikibooks.org/wiki/PHP_Programming