TEMA 4.5

Consulta con objetos

Consultas a la BD utilizando FETCH_CLASS

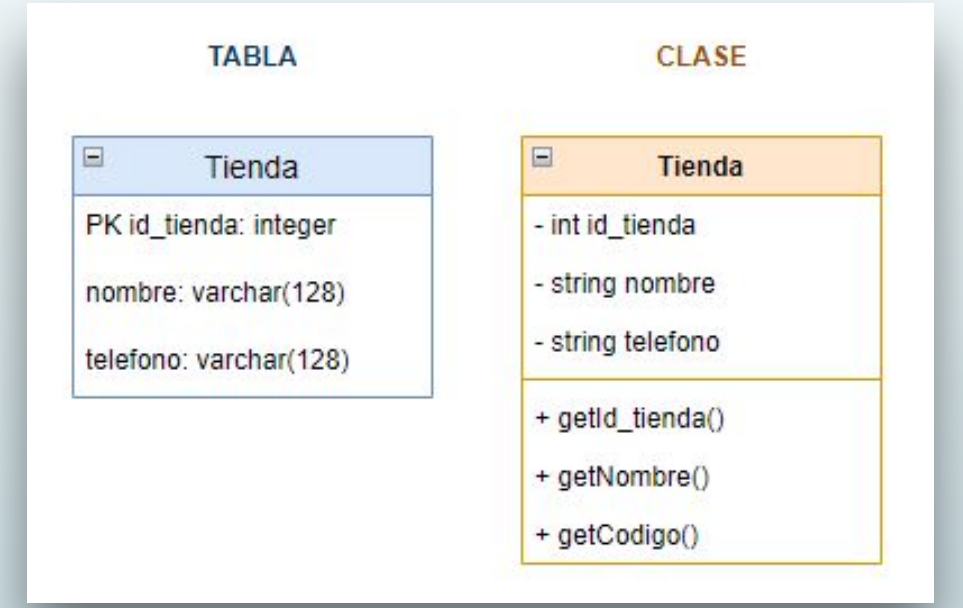
- Como ya hemos visto a lo largo del tema, el método fetch nos permite recuperar el resultado de una query en un array asociativo, con la opción FETCH_ASSOC
- Otra opción es utilizar la opción **FETCH_CLASS**, que nos permite recuperar el resultado de una consulta en un objeto de una clase.
- En este caso, el fetch retorna
  - false si no hay más filas (igual que en el FETCH_ASSOC)
  - La fila siguiente en una instancia de la clase de forma que los datos de las columnas se almacenan en los atributos de la clase.
- Para que este sistema funcione, el nombre de los atributos de la clase debe ser idéntico al nombre de las columnas de la tabla o vista que se está recuperando.
- Si cambia la base de datos, tenemos que adaptar la clase para que nuestro programa siga funcionando.

Respecto al constructor

- Por defecto, fetch rellenará los atributos, y llamará al constructor después
- Si queremos que el constructor se ejecute antes, tenemos que decírselo mediante la propiedad FETCH_PROPS_LATE
- Para configurar la forma en que actuará el método fetch, podemos utilizar el método **setFetchMode**
- Imaginemos que tenemos una clase llamada Instituto, reflejo de la tabla Instituto. Para configurar el fetch de forma que el constructor se ejecute antes de actualizar los valores de los atributos pondríamos:

```
$sentencia->setFetchMode(PDO::FETCH_CLASS || PDO::FETCH_PROPS_LATE, Instituto::class);
```

Un ejemplo de acceso con clases 1: Tabla



- Lo primero crearemos una tabla "tienda" en nuestra base de datos

```sql
CREATE TABLE tienda (
    id_tienda INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(128),
    telefono VARCHAR(128)
) ENGINE INNODB CHARSET=utf8;
```

Un ejemplo de acceso con clases 2: Clase

- Luego creamos la clase.
- Los atributos deben tener igual nombre que los campos de la tabla.

```php
<?php
  // Mis clases
  1 reference | 0 implementations
  class Tienda {
    1 reference
    private int $id_tienda;
    1 reference
    private string $nombre;
    1 reference
    private string $telefono;
    1 reference | 0 overrides
    public function getId_tienda() {
      return $this->id_tienda;
    }
    1 reference | 0 overrides
    public function getNombre() {
      return $this->nombre;
    }
    1 reference | 0 overrides
    public function getTelefono() {
      return $this->telefono;
    }
  }
?>
```

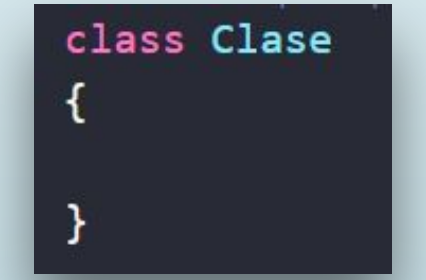
Un ejemplo de acceso con clases 3: Acceso

```php
<?php
  require_once("pdo.php");
  $sql = "SELECT * FROM tienda";
  $sentencia=$pdo->prepare($sql);
  // Configuramos la operatividad del método fetch, indicando que
  // se retornará una instancia de la clase Tienda
  $sentencia->setFetchMode(PDO::FETCH_CLASS, Tienda::class);
  // Ejecutamos la sentencia
  $sentencia->execute();
  // Mostramos cada fila, recuperándola en una instancia de la clase con fetch
  while ($tienda = $sentencia->fetch()) {
      // Primero hago un echo de los atributos del objeto, para ver que los carga bien
    echo "Id_tienda: " . $tienda->getId_tienda() . "<br>";
    echo "Nombre: " . $tienda->getNombre() . "<br>";
    echo "Telefono: " . $tienda->getTelefono() . "<br>";
      // Hago un var_dump del objeto, para ver su información estructurada
    echo "<pre>";
    var_dump($tienda);
    echo "</pre>";
  }
?>
```

- Preparamos la sentencia como siempre, con un prepare.
- Establecemos el modo en que funcionará el fetch con setFetchMode
- Ejecutamos la sentencia
- Mediante un bucle, recuperamos cada registro con fetch en un objeto de la clase Tienda.

Otra opción: Clase vacía

- Todo lo visto anteriormente también funciona si definimos una clase vacía
- Al hacer fetch, PHP creará los atributos de la clase, con el mismo nombre que las columnas de la query. Luego actualiza la información de los campos creados con el valor recuperado en la query
- Al comportamiento anterior se le llama "**set**"
- Este sistema tiene la siguiente ventaja:
  - Podemos utilizar la misma clase para diversas consultas
  - Si la clase que indicamos no existe, PHP guardará la información en un array asociativo, en vez de en un objeto.



Método mágico: __set()

- Si hemos definido una clase con determinados atributos para almacenar el resultado, y por algún motivo no existe uno de los atributos:
  - PHP rellena los atributos existentes, luego *crea* el que falta y lo rellena con el valor recuperado en la query.
  - La creación del atributo que falta lo hace PHP mediante un comportamiento "set". Existe la posibilidad de sobreescribir el comportamiento de "set" con el método mágico
  - `__set($nombreCampo, $valorCampo)`
  - Si creamos un método mágico **__set()** vacío, no se crea el atributo y no se recupera esa información. Tampoco falla.

Recuperar los datos en un array de objetos: fetchAll

- Si en vez de un fetch, utilizamos el método **fetchAll**, PHP creará un array de objetos, cada uno de los cuales contendrá uno de los registros retornados por la query.
- Podemos recorrer el array de objetos con un foreach

```php
$sql = "SELECT * FROM tienda";
$sentencia=$pdo->prepare($sql);
$sentencia->setFetchMode(PDO::FETCH_CLASS, Tienda::class);
$sentencia->execute();
$tiendas = $sentencia->fetchAll();
foreach ($tiendas as $tienda) {
  echo "<pre>";
  var_dump($tienda);
  echo "</pre>";
}
```

```
object(Tienda)#2 (3) {
  ["id_tienda":"Tienda":private]=>
  int(1)
  ["nombre":"Tienda":private]=>
  string(5) "chino"
  ["telefono":"Tienda":private]=>
  string(11) "333 333 444"
}

object(Tienda)#4 (3) {
  ["id_tienda":"Tienda":private]=>
  int(2)
  ["nombre":"Tienda":private]=>
  string(10) "telepostre"
  ["telefono":"Tienda":private]=>
  string(11) "111 333 444"
}

object(Tienda)#5 (3) {
  ["id_tienda":"Tienda":private]=>
  int(3)
  ["nombre":"Tienda":private]=>
  string(7) "cositas"
  ["telefono":"Tienda":private]=>
  string(11) "666 333 444"
}
```

Ejercicio 1:

Acceso con FETCH_CLASS

Vas a programar varias formas de acceso a la tabla "institutos" que has utilizado en ejercicios anteriores:

- Recuperarás todos los registros de la tabla
- Mostrarás todos los registros de la tabla en forma de tabla

1. La clase creada tendrá un atributo por campo de la tabla. Utilizarás fetch para recuperar la información
2. Crearás una clase con un atributo menos, sin sobreescribir set. Utilizarás fetch.
3. Finalmente, crearás una clase vacía, y recuperarás la información con fetchAll

https://www.php.net/docs.php

https://www.php.net/manual/es/book.pdo.php

Manual Oficial: https://www.phptutorial.net/php-pdo/pdo-fetch_class

Tutorial Interesante: https://phpdelusions.net/pdo/objects