

# Chapitre 6 : Accès aux bases de données (sans framework)

PDO : PHP Data Objects

## - Caractéristiques

- Depuis PHP 5
- Orienté objet
- Interface commune à de multiples bases de données (+ de 10), via un driver spécifié dans le DSN

## -- Rappels SQL

- [SQL](#)
- Référence aux types de champs
- Protection des noms des objets : **SELECT \* FROM `nomTable`**

## -- Connexion

### -- Exemple de connexion

```
<?php
$dbo = new PDO('mysql:host=localhost;dbname=test', $user, $pass);
?>
```

### -- DSN

Data Source Name : chaîne définissant les paramètres de connexion à une base

BDD	DSN
Mysql	mysql:host=localhost;port=3307;dbname=testdb
PostgreSQL	pgsql:host=localhost;port=5432;dbname=testdb;user=bruce;password=mypass
Oracle	oci:dbname=localhost:1521/mydb

### -- Gestion des erreurs

```
<?php
try {
    $dbo = new PDO('mysql:host=localhost;dbname=test', $user, $pass);
    $dbo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    print "Error!: " . $e->getMessage() . "<br/>" ;
```

```
    die();
}
?>
```

## -- Paramètres divers

### -- Port par défaut du serveur

```
<?php
    $dbo = new PDO('mysql:host=localhost;dbname=test;port=3306',$user, $pass);
?>
```

### -- Définition de l'encodage

```
<?php
    $dbo = new PDO('mysql:host=localhost;dbname=test;port=3306',$user, $pass);
    $dbo->exec("SET CHARACTER SET utf8");

?>
```

ou

```
<?php
    $dbo=new PDO("mysql:host=localhost;dbname=test;charset=UTF8");
?>
```

## -- Interrogation de données

**PDO::query** retourne un **PDOStatement** (jeu d'enregistrements) qu'il est ensuite possible de parcourir avec un foreach (retourne false en cas d'échec):

```
<?php
    $sql = 'SELECT name, color, calories FROM fruit ORDER BY name';
    $statement=$dbo->query($sql);
    foreach ($statement as $row) {
        print $row['name'] . "\t";
        print $row['color'] . "\t";
        print $row['calories'] . "\n";
    }
?>
```

Produit le résultat :

```
apple    red     150
banana  yellow  250
kiwi    brown   75
lemon   yellow  25
orange  orange  300
pear    green   150
watermelon      pink   90
```

```
<?php
    $sql = 'SELECT name, calories FROM fruit WHERE calories >100 ORDER BY name';
    $statement=$dbo->query($sql);
    foreach ($statement as $row) {
        print $row['name'] . "\t";
        print $row['calories'] . "\n";
    }
?>
```

Produit le résultat :

```
apple    red     150
banana  yellow  250
orange  orange  300
pear    green   150
```

## -- Mise à jour de données

A la différence des instructions de sélection, la mise à jour ne retourne pas de jeu d'enregistrements.

**PDO::exec()** exécute une requête SQL et retourne le nombre de lignes affectées par la requête.

```
<?php
$dbo = new PDO('mysql:host=localhost;dbname=test', $user, $pass);

/* Effacement de toutes les lignes de la table FRUIT dont la couleur est rouge*/
$count = $dbo->exec("DELETE FROM fruit WHERE couleur = 'rouge'");

/* Retourne le nombre de lignes effacées */
print("Retourne le nombre de lignes effacées :\n");
print("Effacement de $count lignes.\n");
?>
```

L'exemple ci-dessus va afficher :

```
Retourne le nombre de lignes effacées :
Effacement de 2 lignes.
```

## -- Injections SQL

Consiste à modifier une instruction SQL existante, de façon malveillante (pour suppression de données,

acquisition de droits, visualisation de données...)

### **Exemple :**

URL **afficheClient.php?id=1**

```
<?php
    $id = $_GET["id"]; // Attention, aucun contrôle !
    $query = "SELECT * FROM clients where id=".$id.";";
    $result = $dbo->query($sql);
?>
```

exemple d'injection possible :

```
1;DELETE FROM clients WHERE 1=1
```

via l'URL **afficheClient.php?id=1;DELETE FROM clients WHERE 1=1**

**Conséquence :** Suppression de toutes les données de la table client

### **-- Solutions pour éviter les injections SQL**

1. Se connecter à la BDD avec un utilisateur ayant des droits limités.
2. Vérifiez que les données ont bien le type attendu. Utiliser **is\_numeric()**, **ctype\_digit()**... ou modifiez les avec **settype()**, ou **sprintf()**
3. Utiliser des requêtes paramétrées

Version corrigée avec **sprintf** de la requête précédente :

```
<?php
    $id = $_GET["id"];
    $query = sprintf("SELECT * FROM clients where id=%d;", $id);
    $result = $dbo->query($sql);
?>
```

### **-- Requêtes paramétrées**

#### **Avantages**

- Anti-injection
- Réutilisation

#### **-- Préparation**

**PDOStatement PDO::prepare ( string \$statement [, array \$driver\_options = array() ] )**

Prépare une requête SQL à être exécutée, et retourne un PDOStatement

```
<?php
    $sql = 'SELECT nom, couleur, calories FROM fruit WHERE calories < :calories AND
couleur = :couleur';
    $sta = $dbo->prepare($sql);
?>
```

## -- Affectation des valeurs

### Avec paramètres nommés

```
<?php
    $sta->bindValue(':calories', $calories, PDO::PARAM_INT);
    $sta->bindValue(':couleur', $couleur, PDO::PARAM_STR);
?>
```

## -- Exécution

```
<?php
    $sta->execute();
?>
```

### Version avec paramètres fictifs

```
<?php
    $calories = 150;
    $couleur = 'rouge';
    $sta = $dbh->prepare('SELECT nom, couleur, calories FROM fruit WHERE calories <
? AND couleur = ?');
    $sta->bindValue(1, $calories, PDO::PARAM_INT);
    $sta->bindValue(2, $couleur, PDO::PARAM_STR);
    $sta->execute();
?>
```

## -- Récupération des enregistrements

**PDOStatement::fetchAll** — Retourne un tableau contenant toutes les lignes du jeu d'enregistrements

```
<?php
$calories=100;
$sta = $dbo->prepare("SELECT nom, couleur FROM fruit WHERE calories> ?");
$sta->bindValue(1, $calories, PDO::PARAM_INT);
$sta->execute();
```

```

/* Récupération de toutes les lignes d'un jeu de résultats */
print("Récupération de toutes les lignes d'un jeu de résultats :\n");
$result = $sta->fetchAll();
print_r($result);
?>

```

## Classes/Enregistrements

### Utilisation de PDO::FETCH\_CLASS

```

<?php
class Fruit {
    public $name;
    public $colour;
}

$sta = $dbo->prepare("SELECT name, colour FROM fruit");
$sta->execute();

$result = $sta->fetchAll(PDO::FETCH_CLASS, "Fruit");
var_dump($result);
?>

```

Résultat :

```

array(3) {
[0]=>
object(Fruit)#1 (2) {
    ["name"]=>
    string(5) "apple"
    ["colour"]=>
    string(5) "green"
}
[1]=>
object(Fruit)#2 (2) {
    ["name"]=>
    string(4) "pear"
    ["colour"]=>
    string(6) "yellow"
}
[2]=>
object(Fruit)#3 (2) {
    ["name"]=>
    string(10) "watermelon"
    ["colour"]=>
    string(4) "pink"
}
}

```

From:  
<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:  
<http://slamwiki2.kobject.net/web/php/chap6>

Last update: **2024/04/08 07:21**

