BASE DE DONNÉES MYSQL

Niv ĺI

PDO PHP DATA OBJECT

Mysqli est de moins en moins utilisé au bénéfice de l'accès en PDO. L'avantage est qu'il permet d'accéder à tout type de serveur de base de données

```
<?php
$pdo = new PDO('mysql:host=localhost;dbname=votre_base', 'utilisateur', 'mot_de_passe');
?>
```

Jusque là aucune nouvelle information si ce n'est l'écriture de la connexion avec new PDO.

Si on veut utiliser un serveur postgresql, on remplacera mysql par pgsql.

PDO PHP DATA OBJECT

Gestion des erreurs de connexion

```
<?php
try {
    $dbh = new PDO('mysql:host=localhost;dbname=test', $user, $pass);
    foreach($dbh->query('SELECT * from FOO') as $row) {
        print_r($row);
    }
    $dbh = null;
} catch (PDOException $e) {
    print "Erreur !: " . $e->getMessage() . "<br/>";
    die();
}?>
```

$dbh=null; permet la fermeture de la connexion au serveur.

Indispensable lorsque nous n'aurons plus besoin d'accéder à la BDD.

PDO PHP DATA OBJECT

Les requêtes SQL sont les mêmes qu'en mysqli, par contre pour les exécuter nous aurons:

```
require 'sqlconnect.php';

$sql = 'DELETE FROM accouns WHERE type = "member"';
$nb = $pdo->exec($sql);

echo $nb.' membres ont été supprimés.';
```

Jusque là aucune nouvelle information si ce n'est l'écriture de la connexion avec new PDO.

Si on veut utiliser un serveur postgresql, on remplacera mysql par pgsql.

PDO PHP DATA OBJECT

Récupérer des données

Pour récupérer les données dans une base de données, nous allons utiliser la méthode query() de la classe PDO. Cette méthode retourne un objet du type « PDOStatement ».

La classe PDOStatement propose une méthode permettant de récupérer ligne par ligne les résultats : fetch().

```
require 'sqlconnect.php';
$sql = 'SELECT * FROM membres';
$req = $pdo->query($sql);
while($row = $req->fetch()) {
echo '<a href="membre-'.$row['id'].'.html">'.$row['pseudo'].'</a><br/>';
}
$req->closeCursor();
```

Nous fermons le curseur car nous n'avons plus besoin de cette variable

```
$sql = 'SELECT * FROM membres WHERE pseudo = '.$pdo->quote($pseudoMembre);
```

PDO PHP DATA OBJECT

Multitables - jointures

Les jointures permettent d'exploiter pleinement le modèle relationnel des tables d'une base de données.

Elle sont faites pour mettre en relation deux (ou plus) tables concourant à rechercher la réponse à des interrogations. Une jointure permet donc de combiner les colonnes de plusieurs tables.

Rappel de la syntaxe du **SELECT** :

```
SELECT [DISTINCT ou ALL] * ou liste_de_colonnes FROM nom_des_tables_ou_des_vues
```

Tâchons donc de récupérer les n° des téléphones associés aux clients.

```
SELECT CLI_NOM, TEL_NUMERO
FROM T_CLIENT, T_TELEPHONE
WHERE T_CLIENT.CLI_ID = T_TELEPHONE.CLI_ID
```

PDO PHP DATA OBJECT

Multitables - jointures

Plus nous ajouterons de tables plus nous aurons de clauses where:

```
SELECT C.CLI_ID, C.CLI_NOM, T.TEL_NUMERO, E.EML_ADRESSE, A.ADR_VILLE
FROM T_CLIENT C, T_TELEPHONE T, T_ADRESSE A, T_EMAIL E
WHERE C.CLI_ID = T.CLI_ID
  AND C.CLI_ID = A.CLI_ID
  AND C.CLI_ID = E.CLI_ID
```

PDO PHP DATA OBJECT

Multitables - jointures

Nous pouvons utiliser dans la clause WHERE, le mot clé LIKE.

Ce mot-clé permet d'effectuer une recherche sur un modèle particulier. Il est par exemple possible de rechercher les enregistrements dont la valeur d'une colonne commence par telle ou telle lettre. Les modèles de recherches sont multiple.

```
SELECT *
FROM table
WHERE colonne LIKE modele
```

Le but n'est pas de trouver des égalités mais plus des contients.

Ex: Mot commençant par une lettre ou terminant par une autre. Pour ça, on utilisera les métas caractères % ou _

% remplace tous les caractères possibles

_ remplace un seul et unique caractères

PDO PHP DATA OBJECT

Multitables - jointures

Exemples:

LIKE ‘%a’ : le caractère « % » est un caractère joker qui remplace tous les autres caractères. Ainsi, ce modèle permet de rechercher toutes les chaines de caractère qui se termine par un « a ».
LIKE ‘a%’ : ce modèle permet de rechercher toutes les lignes de « colonne » qui commence par un « a ».
LIKE ‘%a%’ : ce modèle est utilisé pour rechercher tous les enregistrement qui utilisent le caractère « a ».
LIKE ‘pa%on’ : ce modèle permet de rechercher les chaines qui commence par « pa » et qui se terminent par « on », comme « pantalon » ou « pardon ».
LIKE ‘a_c’ : peu utilisé, le caractère « _ » (underscore) peut être remplacé par n’importe quel caractère, mais un seul caractère uniquement (alors que le symbole pourcentage « % » peut être remplacé par un nombre incalculable de caractères . Ainsi, ce modèle permet de retourner les lignes « aac », « abc » ou même « azc ».

```
SELECT titre, annee, codePays, genre, idMes
FROM Film, Artiste
WHERE titre LIKE '%$titre%'
AND nom LIKE '%nomMes%'
AND annee BETWEEN $anneemin AND $anneemax
AND idMes=id
```

PDO PHP DATA OBJECT

Multitables - jointures

AS

```
SELECT titre, nomRole
FROM Film AS f, Role AS r
WHERE f.titre=r.titre
```

LIMIT permet de limiter le nombre de données retournées:

```
SELECT *
FROM Film
LIMIT 3
```

Ici uniquement les 3 premiers résultats de la requête seront renvoyés

```
SELECT *
FROM Film
LIMIT 1,3
```

Le premier chiffre indique le numéro de la ligne à partir de laquelle la limite s'applique, les lignes étant numérotées à partir de 0.

PDO PHP DATA OBJECT

get_magic_quotes_gpc()

```
<?php
// Si les guillemets magiques sont actifs
echo $_POST['lastname'];           // O\'reilly
echo addslashes($_POST['lastname']); // O\\\'reilly
// Utilisation pour toutes les versions de PHP
if (get_magic_quotes_gpc()) {
   $lastname = stripslashes($_POST['lastname']);
}
else {
   $lastname = $_POST['lastname'];
}
// Si vous utilisez MySQL
$lastname = mysql_real_escape_string($lastname);
echo $lastname; // O\'reilly
$sql = "INSERT INTO lastnames (lastname) VALUES ('$lastname')";
?>
```