PHP 4° PARTIE :
BASE DE DONNEES

1. Introduction
2. Présentation de MySQL
3. Principe
4. Connexion
5. Interrogation
6. Extraction des données
7. Fonctions de services
8. Traitement des erreurs
9. Travaux pratiques

1 . Introduction

✓ Parmi les nombreux atouts du langage PHP, un des plus connus est son interfaçage avec la majorité des bases de données du marché.

✓ Parmi les plus connues, on peut citer : **MySQL**, **PostgreSQL**, Oracle, Ingres, Interbase, Informix, Microsoft SQL Server, mSQL, Sybase, FrontBase, dBase, etc …

✓ La base de donnée la plus utilisée avec PHP est sans aucun doute : MySQL, un SGDBR GPL.

2 . Présentation MySQL

MySQL est une base de données implémentant le langage de requête SQL.

SGBDR = Système de Gestion de Base de Données Relationnelle

Remarque : cette partie suppose connue les principes des bases de données relationnelles.

Il existe un outil libre et gratuit développé par la communauté des programmeurs libres : phpMyAdmin, qui permet l'administration aisée des bases de données MySQL avec php. Il est disponible sur : http://sourceforge.net/projects/phpmyadmin/ et http://www.phpmyadmin.net.

Avec MySQL vous pouvez créer plusieurs bases de données sur un serveur. Une base est composée de tables contenant des enregistrements.

Plus d'informations sont disponibles à http://www.mysql.com/.

La documentation de MySQL est disponibles à http://www.mysql.com/documentation/, ainsi qu'en français chez nexen : http://dev.nexen.net/docs/mysql/.

3 . Principe

- **PHP fournit un grand choix de fonctions permettant de manipuler les bases de données. Toutefois, parmi celles-ci quatre fonctions sont essentielles :**
  - **La fonction de connexion au serveur**
  - **La fonction de choix de la base de données**
  - **La fonction de requête**
  - **La fonction de déconnexion**

- **Avec le SGBD *MySQL*, ces fonctions sont les suivantes :**
  - `mysql_connect`
  - `mysql_select_db`
  - `mysql_query`
  - `mysql_close`

- **Evidemment, il faudra traiter le résultat de la requête effectuée et donc transformer le résultat d'un ligne soit sous forme de variables, de tableau, de tableau associatif, d'objets.**

4 . Connexion (I)

Pour se connecter à une base de donnée en php, il faut spécifier un nom de serveur, un nom d'utilisateur, un mot de passe et un nom de base.

Les fonctions de connexion :

`mysql_connect($server,$user,$password)` : permet de se connecter au serveur **`$server`** en tant qu'utilisateur **`$user`** avec le mot de passe **`$password`**, retourne l'identifiant de connexion si succès, FALSE sinon

`mysql_select_db($base[,$id])` : permet de choisir la base **`$base`**, retourne TRUE en cas de succès, sinon FALSE

`mysql_close([$id])` : permet de fermer la connexion

`mysql_pconnect()` : idem que **`mysql_connect()`** sauf que la connexion est persistante, il n'y a donc pas besoin de rouvrir la connexion à chaque script qui travaille sur la même base.

Remarque : les identifiants de connexion ne sont pas nécessaires si on ne se connecte qu'à une seule base à la fois, ils permettent seulement de lever toute ambiguïté en cas de connexions multiples.

4 . Connexion (II)

Exemple simple de connexion :

```
if($db = mysql_connect("localhost", "root", ""))
{
        $id_db = mysql_select_db("test");
                die("Echec de connexion à la base !");
}
else die("Echec de connexion au serveur de base de données");
// code du script
// …
mysql_close($db);
```

4 . Connexion (III)

En pratique, on constate les comportements suivants :

- Utilisation de variables globales de connexion (**`$user, $passwd, $host et $base`**) qui seront placées dans un fichier du style `config.inc.php` et inclus dans chaque script qui en a besoin par un `require()` ou un `include()`.

- Intégration du code de connexion ou plus largement d'interfaçage à la base de données dans un fichier à inclure (par exemple `mysql.inc.php`)

- Utilisation d'un API de haut niveau pour notamment le portage vers d'autres bases de données (pilote ODBC, couche DAO ou des bibliothèques comme PEAR::DB). Remarque : le plus souvent sous forme de classes.

- Utilisation des connexions persistantes : évite d'avoir à rouvrir une connexion dans chaque script. Les connexions sont automatiquement fermées au bout d'un certain temps en cas d'absence de toute activité…

5 . Interrogation

Pour envoyer une requête à une base de donnée, on utilise la fonction : **mysql_query($str)** qui prend pour paramètre une chaîne de caractères qui contient la requête écrite en SQL et retourne un identificateur de résultat ou FALSE en cas d'échec.

Les requêtes les plus couramment utilisées sont : **CREATE** (création d'une table), **SELECT** (sélection), **INSERT** (insertion), **UPDATE** (mise à jour des données), **DELETE** (suppression), **ALTER** (modification d'une table), etc ...

Exemple :

```
$result = mysql_query("SELECT * FROM revues WHERE revueid='$id'");
```

L'identificateur de résultat **`$result`** permettra à d'autres fonctions d'extraire ligne par ligne les données retournées par le serveur.

6 . Extraction des données (I)

Une fois la requête effectuée et l'identificateur de résultat obtenu, il ne reste plus qu'à extraire les données retournées par le serveur.

Sous MySQL (comme pour beaucoup de SGBD), le traitement des résultats d'une requête se fait ligne par ligne. Une boucle permettra de recueillir chacune des lignes à partir de l'identifiant de résultat.

revueid	*nom*
1	Programmez!
2	Linux Magazine
3	LOGIN:

Une ligne contient une ou plusieurs valeurs correspondants aux différents attributs retournés par la requête. Ainsi, une ligne de résultat pourra être sous la forme de variables, d'un tableau, d'un tableau associatif, ou d'un objet.

6 . Extraction des données (II)

`mysql_fetch_row($result)` : retourne une ligne de résultat sous la forme d'un tableau. Les éléments du tableau étant les valeurs des attributs de la ligne. Retourne FALSE s'il n'y a plus aucune ligne.

Exemple :

```
$requete = "SELECT * FROM revues";
if($result = mysql_query($requete))
{          while($ligne = mysql_fetch_row($result))
           {          $revueid = $ligne[0];
                      $nom = $ligne[1];
                      echo "$revueid -&gt; $nom<br />";
           }
}
```

Ici, on accède aux valeurs de la ligne par leur indice dans le tableau.

Une pratique courante est d'utiliser la fonction **`list()`** pour avoir automatiquement les champs sous forme de variables :

```
           while(list($revueid, $nom) = mysql_fetch_row($result))
           {
                      echo "$revueid -&gt; $nom<br />";
           }
```

6 . Extraction des données (III)

`mysql_fetch_array($result)` : retourne un tableau associatif. Les clés étant les noms des attributs et leurs valeurs associées leurs valeurs respectives. Retourne FALSE s'il n'y a plus aucune ligne.

Exemple :

```
$requete = "SELECT * FROM revues";
if($result = mysql_query($requete))
{
        while($ligne = mysql_fetch_array($result))
        {
                $revueid = $ligne["revueid"];
                $nom = $ligne["nom"];
                echo "$revueid -&gt; $nom<br />";
        }
}
```

Ici, on accède aux valeurs de la ligne par l'attribut dans le tableau associatif.

6 . Extraction des données (IV)

`mysql_fetch_object($result)` : retourne un objet. Les attributs de l'objet correspondent à ceux de la ligne de résultat. Et les valeurs des attributs de l'objet correspondent à ceux de la ligne de résultat.

Retourne FALSE s'il n'y a plus aucune ligne.

Exemple :

```
$requete = "SELECT * FROM revues";
if($result = mysql_query($requete))
{
        while($ligne = mysql_fetch_object($result))
        {
                $revueid = $ligne->revueid;
                $nom = $ligne->nom;
                echo "$revueid -&gt; $nom<br />";
        }
}
```

Ici, on accède aux valeurs par leur attribut dans l'objet.

7 . Fonctions de service

Quelques fonctions supplémentaires très utiles :

`mysql_free_result($result)` : efface de la mémoire du serveur les lignes de résultat de la requête identifiées par **`$result`**.

`mysql_insert_id([$id])` : retourne l'identifiant d'un attribut clé primaire AUTO_INCREMENT de la dernière insertion.

`mysql_num_fields($result)` : retourne le nombre d'attributs du résultats.

`mysql_num_rows($result)` : retourne le nombre de lignes du résultats.

8 . Traitement des erreurs

Généralement, on préfixe les appels aux fonctions `mysql_` avec @ pour éviter les affichages dans le navigateur des messages de WARNING ou d'erreur (sauf en débuggage).

Par contre, il faut systématiquement tester et traiter les erreurs en provenance de la base de données. Pour cela, on pourra utiliser :

`mysql_errno()` : retourne le numéro de message d'erreur de la dernière opération MySQL
`mysql_error()` : retourne le texte associé à l'erreur générée lors de la dernière requête.

Exemple :

```
mysql_connect("TF1"); // ajouter le if
echo mysql_errno().": ".mysql_error()."<br>";
mysql_select_db("StarAcademy"); // ajouter le if
echo mysql_errno().": ".mysql_error()."<br>";
$result = mysql_query("SELECT * FROM artistes"); // ajouter le if
echo mysql_errno().": ".mysql_error()."<br>";
```

9 . Travaux pratiques

1. Créer la table revues comportant les 2 champs décris dans le cours. Insérer quelques enregistrements.

2. Réaliser un script revues.php qui permet de se connecter à la base et afficher la liste complète des revues disponibles ainsi que le nombre total de revues.

3. Ecrire l'application qui réalisera les fonctions de base suivantes : ajouter des revues, modifier des revues existantes, supprimer des revues existantes et afficher les revues disponibles. On structurera son application en utilisant des fichiers séparés.

4. Compléter l'application précédente en ajoutant un script d'installation permettant de créer la table sur le serveur en vérifiant préalablement son existence. Dans ce cas, le script permettra, suivant le choix de l'utilisateur, de la détruire puis de la créer ou de la conserver.

5. Aller plus loin : normalement, ce type d'application doit permettre de stocker des articles parues dans les revues existantes. Un minimum est de conserver un sommaire et de pouvoir y faire une recherche suivant des mots clés …