Développement Web
Atelier Interfaçage base de données: PHP/Mysql

Objectif de l'atelier:
Etre capable de mettre en place une interface PHP d'une base de données sous Mysql.
- se connecter à un serveur de base de données mysql,
- executer une requête sur un serveur,
- exploitater le résultat d'une requête

Prérequis conseillés:
Ateliers de transmission de données
Fonctionnement des bases de données relationnelles.
Durée : 3 h

1. Introduction

L'intérêt majeur de PHP est son interfaçage avec un grand nombre de serveurs de bases de données (MySQL, Postgres, Oracle, utilisation de ODBC, ...) d'une manière relativement simple et efficace. Pratiquement tous les SGBD sont pris en charge mais PHP s'utilise le plus souvent avec le serveur Mysql. Le logiciel MySQL (TM) est un serveur de base de données SQL très rapide, multi-threadé, multi-utilisateur et robuste.

2. Création d'une base de données de gestion des inscriptions

En utilisant l'outil phpMyadmin créer sur le serveur MySQL une base de données GestionEtu contenant les tables etudiants, classes et inscription pour la gestion des inscriptions des étudiants et une table utilisateurs (id, login, password) pour la gestion de l'espace administrateur. Vous expliquerez le choix de chacune des propriétés de champs (type, valeurs par défaut, auto increment, ...). Pour plus d'information sur MySQL: http://www.mysql.com/documentation

3. Fonctionnement de base

PHP est un langage côté serveur, qui peut être utilisé pour créer des pages Web dynamiques. Il fournit un support d'accès à plusieurs systèmes de bases de données, dont MySQL. L'exploitation de MySQL avec PHP peut s'effectuer en 5 étapes :

3.1. Connexion au serveur MySQL (authentification)

Vous demandez au serveur l'autorisation de vous connecter. Deux réponses possibles :
- refus: le serveur vous renvoie un false
- accordé: le serveur vous renvoie une ressource (différent de false) identifiant votre connexion.

mysql_connect est la fonction de connexion. Elle a besoin de trois paramètres fondamentaux: l'adresse du serveur ("localhost" dans l'exemple 1), le login (toto dans l'exemple 1) et le mot de passe (tati dans l'exemple 1). La fonction mysql_connect retourne l'identifiant de la connexion au serveur MySQL en cas de succès ou FALSE en cas d'échec.

Exemple 1

```
<?php
$link = mysql_connect("localhost", "toto", "tati") or
die("Impossible de se connecter : " . mysql_error());  //== echo puis exit
echo 'Connexion réussie';
?>
```

3.1.1. Application

- Quel est le message d'erreur si le nom de la machine est incorrect, le login ou le mot de passe?
- Quel est le message d'erreur si le login n'est pas spécifié?
- Utiliser le caractère @ devant la fonction *mysql_connect*. Quel est le résultat obtenu?

- Discussion...

3.2. Sélection d'une base de données

Après la connexion au serveur Mysql vous choisissez la base de données à utiliser. Toutes les requêtes lancées ensuite concerneront uniquement cette base de données. La fonction *mysql_select_db* permet de spécifier la base que vous utilisez. Elle dispose d'un paramètre obligatoire: le nom de la base de données et d'un paramètre optionnel: l'identifiant de la connexion (par défaut la dernière connection). La fonction *mysql_select_db* retourne TRUE en cas de succès et FALSE en cas d'échec.

Exemple 2

```
<?php
//code exemple 1 d'abord
// Rendre gestionEtu, la base courante
$db_selected = mysql_select_db('gestionEtu', $link);
if (!$db_selected) {
die ('Impossible de sélectionner cette base de données : ' . mysql_error());
?>
```

3.3. Application

- Quel est le message d'erreur si la base n'existe pas?
- Remplacez le if de l'exemple 2 par un or die comme dans l'exemple 1.

3.4. Lancement d'une requête

La requête concerne la base de données en cours. La fonction *mysql_query* est utilisée pour envoyer une requête à une base de donnée Mysql. Cette fonction dispose d'un paramètre obligatoire: la requête SQL et d'un paramètre optionnel: l'identifiant de la connection (par défaut la dernière connection). Si la requête à exécuter est de type SELECT, SHOW, DESCRIBE ou EXPLAIN la fonction *mysql_query* renvoie une ressource en cas de succès, ou FALSE en cas d'erreur. Pour les autres types de requêtes (UPDATE, DELETE, DROP, …) la fonction *mysql_query* renvoie TRUE en cas de succès ou FALSE en cas d'erreur.

Exemple 3

```
<?php
// code exemple 2 ensuite
$requet = 'SELECT nom,prenom FROM etudiants';
if($res = mysql_query($requete))
//traitement resultat voir exemple 4
else
echo "Erreur de requête de base de données.";
?>
```

3.4.1. Application

- Quelles sont les données récupérées après l'exécution de cette requête ?
- Expliquez le comportement du script de l'exemple 3.

3.5. Exploitation (affichage, test, ...) des résultats de la requête

Il s'agit généralement d'afficher le résultat de la requête où de renseigner sur le nombre d'insertion, de suppression, … réalisées. Pour extraire les données retournées plusieurs fonctions peuvent être utilisées selon le format voulu:

- mysql_fetch_row: .La ligne est retournée sous la forme d'un tableau. Chaque colonne est enregistrée sous la forme d'un tableau commençant à la position 0;
- mysql_fetch_assoc:Retourne une ligne de résultat MySQL sous la forme d'un tableau associatif;
- mysql_fetch_array: Retourne une ligne de résultat MySQL sous la forme d'un tableau associatif, d'un tableau indexé, ou les deux. Le deuxième paramètre permet de spécifier le type de tableau attendu (MYSQL_BOTH (défaut), MYSQL_ASSOC, MYSQL_NUM);
- mysql_fetch_object: Retourne une ligne de résultat MySQL sous la forme d'un objet.

Ces fonctions prennent comme paramètre la ressource retournée par mysql_query. La récupération se fait enregistrement par enregistrement et ces fonctions renvoient false si la fin des

enregistrements est atteinte. Généralement ces fonctions sont utilisées à l'intérieur d'une boucle pour parcourir l'ensemble du résultat.

Exemple 4

```
<?php //cas 1
// code exemple 3
$row=mysql_fetch_row($res);
echo $row[0]; // le nom
echo $row[1]; // le prenom
?>
```

```
<?php //cas 2
// code exemple 3
while ($row =
mysql_fetch_assoc($res))
{
echo $row["nom"];
echo $row["prenom"];
}
mysql_free_result($res);
?>
```

```
<?php //cas 3
// code exemple 3
while ($row =
mysql_fetch_array($res,
MYSQL_BOTH))
{
printf ("nom : %s adresse : %s",
$row[0], $row["prenom"]);
}
mysql_free_result($result);
?>
```

```
<?php //cas 4
// code exemple 3
while ($row =
mysql_fetch_object($result))
{
echo $row->nom;
echo $row->adresse;
}
mysql_free_result($result);
?>
```

```
<?php //cas 5
// code exemple 2
$login= htmlspecialchars ($_POST["login"]);
$password= htmlspecialchars ($_POST["password"]);
$requete= "select * from utilisateurs where login='$login' and
password='$password'";
if($res = mysql_query($requete))
{
 if(mysql_num_rows($res)==1) //si un enregistrement dans la base
 {
  header("Location:saisiEtudiant.php");
 }
 else
  header("Location:login.php?erreur=1");
}
mysql_free_result($result);
?>
```

3.6. Application

- Dans le repertoire atelier7 à la racine du documentRoot créer pour chaque exemple de cette section un fichier php (exemple4cas1.php, exemple4cas2.php, …). Intégrer le cas 5 dans le script login.php pour gerer les utilisateur à partir de la table *utilisateurs* de la base gestionEtu.

3.7. Fermeture de la connexion.

La fermeture de la connection qui permet de liberer la ressource n'est habituellement pas nécessaire. En effet les connexions non persistantes ouvertes sont automatiquement fermées à la fin de l'exécution du script. La fonction *mysql_close* est utilisée pour fermer une connection. Elle retourne TRUE en cas de succès et FALSE sinon.

3.8. Remarque

- A chaque étape (authentification, sélection de la base, …) il convient de vérifier et de traiter les erreurs éventuelles (erreur de connexion au serveur, erreur de sélection de la base,...).
- Le processus décrit pour Mysql est identique pour les autres SGBD... Seules les fonctions PHP à utiliser différent.

4. Autres fonctions php/mysql

Il existe de nombreuses autres fonctions pour faciliter la manipulation des bases de données mysql. Le tableau suivant en liste quelques unes avec des exemples d'utilisation.

Fonctions	Exemple	Description
mysql_num_rows	mysql_num_rows($res)	renvoie le nombre d'enregistrement du résultat
mysql_affected_rows	mysql_affected_rows()	renvoie le nombre d'enregistrement affecté par la dernière requête (insertion, suppression, mise à jour)
mysql_data_seek	mysql_data_seek($res,$i)	positionne le curseur sur le numéro d'enregistrement $i donné
mysql_field_name	mysql_field_name($res,$i)	renvoie le nom d'une colonne dont le rang $i est donné
mysql_pconnect	mysql_pconnect($serv,$login,$pass)	ouvre une connection persistante

Application

- Créer dans le repertoire atelier des scripts de test des différentes fonctions citées dans le tableau précédent.

5. Questions et Applications:

5.1. Questions

5.1.1. Quelles sont les différentes étapes pour executer une requête sur un serveur Mysql ?

5.1.2. Quelles différences existent ils entre les fonctions mysql_connect et mysql_pconnect ?

5.2. Applications

5.2.1. Scripts insertion : Completer les scripts insererClasse.php, insererEtudiant.php, inscription pour respectivement ajouter une classe, un étudiant ou inscrire un étudiant dans une classe donnée.

5.2.2. Scripts listes: Modifier les scripts listeClasses.php et listeEtudiants.php pour afficher respectivement la liste des classes et la liste des étudiants d'une classe (zone de selection pour choisir la classe).

5.2.3. Sécuriser les pages d'insertion pour qu'elles ne soient utilisées que par des administrateurs (utilisateur ayant un compte dans la table utilisateurs). Les pages liste sont disponibles pour les visiteurs du site.