PHP et les requêtes MySQL

Date de dernière mise à jour : 27/06/2007 à 19:36

Introduction

Après avoir vu ce qu'était une base de données et découvert comment fonctionnait PHPMyAdmin dans le chapitre précédent, nous allons maintenant nous intéresser à la partie PHP, car n'oubliez pas que PHP va vous permettre de communiquer avec votre base de données. Dans ce chapitre, on utilisera toujours PHP pour communiquer avec MySQL, mais sachez qu'il est tout à fait possible d'entrer les requêtes à la main dans ce qu'on appelle une console (pour voir la console Windows, cliquez sur **démarrer**, **Exécuter** et tapez **cmd**).

Avant toute chose, il faut que vous sachiez qu'on ne fait pas directement une requête avec une fonction PHP. Il faut d'abord passer par plusieurs étapes. Un SGBD dispose ainsi de droits d'accès et peut comporter plusieurs bases. Il faut donc tout d'abord s'identifier au serveur MySQL que l'on souhaite utiliser. Une fois cette identification faite, il faut sélectionner la base sur laquelle on souhaite travailler. Et après ça, vous pourrez effectuer vos requêtes ! Voyons comment cela fonctionne.

Connexion au serveur MySQL

La première étape consiste à se connecter au serveur MySQL. Pour cela, on utilise la fonction mysql_connect() de PHP. Elle prend trois arguments :

-L'adresse IP (ou l'alias pointant vers l'ip) du serveur MySQL
-Le nom d'utilisateur
-Le mot de passe

Ces trois paramètres vous sont fournis par votre hébergeur. Lorsque vous travaillez avec EasyPHP ou Wamp, les paramètres à utiliser sont :

-localhost
-root

et il n'y a pas de mot de passe à renseigner.

Voilà comment vous pouvez coder ça pour un usage local :

```
<?php $connexion
=
mysql_connect
(
'localhost'
,
'root'
,
''
) OR die(
'Erreur de connexion'
);
?>
```

La partie **OR die('Erreur de connexion')** n'est pas indispensable, mais elle permet de couper le script si il n'a pas réussi à se connecter au serveur. Il est important pour une application sécurisée de bien gérer les erreurs que vous pouvez avoir à la connexion ou lors de la sélection de base de données (on va voir ça tout de suite)

Sélection de la base de données

Une fois que vous êtes connecté au serveur MySQL, sachant qu'il peut contenir une infinité de bases de données, il faut bien qu'il sâche sur laquelle vous souhaitez travailler. On utilise pour cela la fonction mysql_select_db() en PHP. Elle prend comme paramètre le nom de la base de données que vous souhaitez utiliser.

Pour ceux qui ont suivi le chapitre précédent, j'avais utilisé une base **actualites**. Tous mes exemples vont donc s'appuyer sur cette base que nous avons créée ensemble. Voici comment dire à PHP que nous allons travailler sur cette base :

```
<?php mysql_select_db
(
'actualites'
) OR die(
'Sélection de la base impossible'
);
?>
```

Encore une fois, la partie **OR die('Sélection de la base impossible')** n'est pas indispensable mais conseillée.

Déconnexion du serveur MySQL

Il s'agit d'une étape très importante et trop souvent négligée dans de nombreux scripts. Il faut savoir que MySQL dispose d'un paramètre spécifiant le nombre maximum de connexions simultannées qu'il peut traiter. En local, vous n'aurez pratiquement jamais de problème avec ce paramètre (qui se manifeste par une erreur de **Max user connections** et vous empêche donc d'effectuer vos requêtes). En revanche, chez un hébergeur, ce paramètre est souvent placé à une valeur de 3 ou de 5 (5 étant préférable). Cela veut dire que 5 connexions pourront avoir lieu quasiment simultanément. Vous vous dites "c'est énorme, il n'y aura jamais personne en même temps qui pourra cliquer sur mon site". Le problème, est que la connexion est par défaut, si vous ne la fermez pas, active pendant toute la durée de génération de la page. Si vous ouvrez votre connexion tout en haut de la page et que le serveur met 1 seconde (ce qui est énorme) pour générer la page, votre connexion restera ouverte pendant une seconde. Vous imaginez qu'il devient alors très facile d'avoir 5 connexions à la même seconde pour peu que vous ayez un peu de visiteurs ou des scripts très lents. Il faut donc fermer la connexion le plus tôt possible, après avoir effectué la dernière requête, et **AVANT TOUT TRAITEMENT**.

Vous allez voir tout à l'heure que nous allons utiliser la fonction mysql_query() pour effectuer des requêtes MySQL, et bien voici un schéma qu'il faudrait adopter pour bénéficier d'une optimisation maximum :

```
<?php $connexion
=
mysql_connect
(
'localhost'
,
'root'
,
''
) OR die(
'Erreur de connexion'
);
mysql_select_db
(
'actualites'
) OR die(
'Erreur de sélection de la base'
);
$requete1
=
```

```
mysql_query
(
'....'
);
$requete2
=
mysql_query
(
'....'
);
//Ici vous placez vos autres requêtes
mysql_close
();
//On ferme la connexion à MySQL
    //Ici vous mettez le code PHP qui va aller récupérer les données provenant des requêtes (fonction mysql_fetch_row() par exemple)
?>
```

Comme vous pouvez le voir, on utilise la fonction **mysql_close()** pour fermer la connexion au serveur MySQL. Par défaut, elle ne prend pas de paramètre.

Lire des données dans une table

Avant de lire des données dans une table, il faut que votre table contienne des enregistrements. On va se servir toujours de la même table que précédemment, c'est à dire la table **news**. Vous pouvez y insérer quelques enregistrements via PHPMyAdmin, comme je vous l'ai déjà montré au chapitre précédent.

Qu'il s'agisse d'une lecture, d'une écriture ou d'une modification de données ou de paramètres, il vous faudra passer par la fonction mysql_query(). Query signifie requête en anglais. C'est cette fonction qui va vous permettre d'interagir avec MySQL.

```
<?php    $requete
=
mysql_query
(
'Ici   votre requête SQL'
);
?>
```

Le résultat de la requête sera retourné dans la variable $requete. Attention, il s'agit d'une variable de type resource. Vous ne pourrez donc pas faire un echo de cette variable, ça ne vous renverra pas ce que vous attendez. On va en reparler juste après ;)

Voyons comment effectuer une requête de sélection de certains champs :

En SQL, lorsque vous souhaitez sélectionner des données provenant d'une table, on utilise tout d'abord le mot SELECT, qui veut dire que vous vous apprêtez à récupérer des données. Ensuite, vous devez indiquer à MySQL la liste des champs de la table que vous souhaitez voir dans votre résultat, séparés par des virgules. Vous devez ensuite indiquer à quelle table vous souhaitez prendre les données via le mot clé FROM suivi du nom de la table

Si nous souhaitons récupérer les champs titre et texte de la table news, voici comment nous allons procéder :

```
<?php    $requete
=
mysql_query
(
'SELECT titre, texte FROM news'
```

```
);
?>
```

Bon c'est bien beau tout ça, mais comment fait-on pour récupérer le résultat d'une requête sous forme textuelle ? Et bien on utilise pour cela quatre fonctions de PHP (en fait on en utilise une parmis les quatre). Voilà les trois fonctions que vous pouvez utiliser :

-mysql_fetch_row() : cette fonction va retourner un tableau avec des indices numériques (l'indice 0 correspond au premier champ de la requête, l'indice 1 au second champ ...). Je vous conseille d'utiliser cette fonction pour des tables dans lesquelles vous sélectionnez peu de champs (un ou deux), car c'est la fonction la plus rapide.
-mysql_fetch_array() : cette fonction va retourner un tableau avec par défaut les indices numériques et associatifs. Elle est plus lente que mysql_fetch_row() mais plus simple d'utilisation. Pour la rendre plus rapide, on peut utiliser la fonction mysql_fetch_assoc() qui va uniquement retourner un tableau avec les indices associatifs.
-mysql_fetch_object() : cette fonction retourne un objet qui aura pour attributs les champs de la requête MySQL.

Voyons voir ce que ça donne pour sélectionner tous les enregistrements de la table actualites, et les afficher :

```
<?php mysql_connect
(
'localhost'
,
'root'
,
''
) OR die(
'Erreur de connexion à la base'
);
mysql_select_db
(
'actualites'
) OR die(
'Erreur de sélection de la base'
);
$requete
=
mysql_query
(
'SELECT titre, texte FROM news'
) OR die(
'Erreur de la requête MySQL'
);
mysql_close
();
/** * On récupère les données * Tant qu'une ligne sera présente, la boucle continuera */
while(
$resultat
=
mysql_fetch_object
(
$requete
)) { echo
'Titre : '
.
```

```
$resultat
->
titre
.
'. Texte : '
.
$resultat
->
texte
.
'


'
;    }
?>
```

Avec la fonction **mysql_fetch_row()**, voici ce que ça aurait donné :

```
<?php      mysql_connect
(
'localhost'
,
'root'
,
''
) OR die(
'Erreur de connexion à la base'
);
mysql_select_db
(
'actualites'
) OR die(
'Erreur de sélection de la base'
);
$requete
=
mysql_query
(
'SELECT titre, texte FROM news'
) OR die(
'Erreur de la requête MySQL'
);
mysql_close
();
/**      * On récupère les données      * Tant qu'une ligne sera présente, la boucle continuera      */
while(
$resultat
=
mysql_fetch_row
(
$requete
))     {          echo
'Titre : '
```

```
.
$resultat
[

].
'. Texte : '
.
$resultat
[
1
].
'

'
;     }
?>
```

Et avec la fonction **mysql_fetch_assoc()** (équivalente à la fonction **mysql_fetch_array()** à laquelle on passe une constante en second paramètre : **MYSQL_ASSOC**) :

```
<?php       mysql_connect
(
'localhost'
,
'root'
,
''
) OR die(
'Erreur de connexion à la base'
);
mysql_select_db
(
'actualites'
) OR die(
'Erreur de sélection de la base'
);
$requete
=
mysql_query
(
'SELECT titre, texte FROM news'
) OR die(
'Erreur de la requête MySQL'
);
mysql_close
();
/**       * On récupère les données       * Tant qu'une ligne sera présente, la boucle continuera       */
while(
$resultat
=
mysql_fetch_assoc
(
$requete
```

```
))
//équivalent à  while($resultat = mysql_fetch_array($requete, MYSQL_ASSOC))
{           echo
'Titre : '
.
$resultat
[
'titre'
].
'. Texte : '
.
$resultat
[
'texte'
].
'


'
;      }
?>
```

Pour tout ce qui est insertion de données, je vous conseille de lire les requêtes MySQL associées dans ce cours : requêtes MySQL d'insertion de données. Le principe est le même, sauf que là on ne récupère aucun résultat.

Chapitre suivant

-Annexes

Optimiser PHP

Contrer les failles de vos scripts PHP

Upload de données sécurisé

FAQ (foire aux questions) pour les problèmes PHP les plus courants