

Tutoriel ultra-minimal : passer des requêtes MySQL en PHP

par [Guillaume Piolle](#)

Date de publication : 25/07/06

Dernière mise à jour : 24/09/06 (version 1.12)

Ce tutoriel est principalement destiné à des débutants en PHP, bien qu'il puisse aussi servir d'aide-mémoire. Il donne les principes de base pour pouvoir utiliser une base de données MySQL depuis un script PHP. Il vise à éviter certaines erreurs courantes, et à bien comprendre les mécanismes de l'interface entre PHP et MySQL. Ce n'est absolument pas un tutoriel avancé, il ne présente que quelques fonctions vitales, et ne se substitue pas à la documentation officielle (elle propose juste une présentation plus synthétique). Ce tutoriel traite pour l'instant exclusivement de l'extension MySQL (et pas mysqli). Ce tutoriel ne traite absolument pas du langage SQL.

- I - Introduction
- II - Connexion au serveur et sélection d'une base
- III - Passage d'une requête SELECT, récupération d'une ressource
- IV - Exploitation de la ressource
 - IV-A - Opérations de base sur les ressources
 - IV-B - Récupération d'un tableau
 - IV-C - Récupération d'un objet
- V - Autres types de requêtes
- VI - Fermeture de la connexion MySQL
- VII - Script complet
- VIII - Les informations utiles en cas de problème
- IX - Remerciements
- X - Références

I - Introduction

Nous allons voir comment passer des requêtes à un serveur MySQL, et comment exploiter le résultat fourni. Nous utiliserons pour cela l'extension MySQL de PHP. Nous supposons ici que vous disposez d'un accès à un serveur MySQL correctement configuré, et que vous avez les privilèges nécessaires pour y passer une requête. Pour nos exemples, nous allons travailler sur une base "developpez", dont la seule table est "tuto_table". Nous avons deux champs dans cette table, `id` et `comment`, et deux enregistrements :

- `id` = 1, `comment` = 'valeur1'
- `id` = 2, `comment` = 'valeur2'

Cette base et cette table sont générées par le code SQL suivant :

Génération de l'environnement de travail

```
CREATE DATABASE `developpez` ;
CREATE TABLE `tuto_table` (
  `id` INT( 10 ) NOT NULL ,
  `comment` VARCHAR( 255 ) NOT NULL ,
  PRIMARY KEY ( `id` )
);
INSERT INTO `tuto_table` ( `id` , `comment` )
VALUES (
  '1', 'valeur1'
), (
  '2', 'valeur2'
);
```

Voilà les étapes nécessaires pour passer une requête SELECT, étapes que nous allons détailler par la suite :

- Connexion au serveur MySQL (authentification) : nous récupérons une *ressource de connexion* sur laquelle nous pourrions travailler.
- Sélection d'une base de données
- Passage de la requête : nous récupérons ici une *ressource*, qui contient (mais pas de manière immédiatement accessible) la réponse à notre requête.
- Vérification de la validité de la ressource récupérée : histoire de s'assurer qu'on a une réponse valide du serveur.
- Extraction des données à partir de la ressource : pour récupérer les informations dans un format exploitable par PHP.
- Fermeture de la connexion.

Durant toutes ces opérations, nous prendrons bien soin d'afficher toutes les erreurs éventuelles, de manière à être mis au courant des problèmes qui peuvent survenir (un principe à toujours respecter en configuration de développement, ne jamais masquer les erreurs). Nous supposons donc que vous avez configuré PHP pour afficher toutes les erreurs PHP, en modifiant de manière adéquate le fichier php.ini ou en plaçant en début de script :

```
error_reporting(E_ALL);
```

Pour les erreurs MySQL, nous utiliserons la fonction `mysql_error()`, qui retourne le texte de la dernière erreur MySQL générée. En effet lorsqu'une erreur MySQL surgit, elle n'est pas affichée automatiquement, à la différence d'une erreur PHP.

II - Connexion au serveur et sélection d'une base

La fonction de connexion est `mysql_connect`. Elle a besoin de trois paramètres fondamentaux, l'adresse du serveur (bien souvent "localhost"), le login et le mot de passe. Ici notre login est "developpez" et notre password est "pass". La connexion se fait ainsi :

Connexion au serveur

```
$link = mysql_connect("localhost", "developpez", "pass");
```

la valeur de retour, `$link`, est une "ressource", un objet PHP qui identifie votre connexion au serveur. Si vous n'utilisez qu'une seule connexion à la fois, ce qui est quand même le cas le plus courant, alors vous pouvez vous dispenser de récupérer la valeur de retour. En cas d'échec de la fonction, c'est une valeur `FALSE` qui est retournée, et PHP affiche un Warning. Voici les erreurs que vous obtiendrez si vous passez des mauvais paramètres :

Mauvais nom de serveur

```
Warning: mysql_connect() [function.mysql-connect]:
Unknown MySQL server host 'locdlocalhost' (11001) in [...]script.php on line 3
```

Mauvais login

```
Warning: mysql_connect() [function.mysql-connect]:
Access denied for user 'developpezzzz'@'localhost' (using password: YES) in [...]script.php on line
3
```

Mauvais password

```
Warning: mysql_connect() [function.mysql-connect]:
Access denied for user 'developpez'@'localhost' (using password: YES) in [...]script.php on line 3
```

Pas de password spécifié

```
Warning: mysql_connect() [function.mysql-connect]:
Access denied for user 'developpez'@'localhost' (using password: NO) in [...]script.php on line 3
```

Ni login ni password spécifié

```
Warning: mysql_connect() [function.mysql-connect]:
Access denied for user 'ODBC'@'localhost' (using password: NO) in [...]script.php on line 3
```

Pouvez-vous vous connecter à un serveur distant ? Bien souvent non, pour des raisons de sécurité, et c'est pourquoi le script PHP qui fait appel à MySQL devra se trouver sur la même machine, avec "localhost" comme nom de serveur de connexion.

Une fois connecté, il faut spécifier la base que vous utilisez. Cette étape est facultative si l'utilisateur que vous utilisez pour vous connecter n'a accès qu'à une seule base. Dans le cas contraire, sélectionnez votre base avec `mysql_select_db` (ici notre base est nommée "developpez", comme notre utilisateur, ce qui est courant) :

Sélection d'une base

```
mysql_select_db("developpez", $link) or die(mysql_error());
```

Le second paramètre est la ressource de connexion, qui est facultative. Vous aurez remarqué le "or die", qui affiche l'erreur MySQL en cas d'échec (en effet dans ce cas `mysql_select_db` renvoie `FALSE`, et le "or die" est exécuté). C'est nécessaire car une fois que la connexion est ouverte, PHP n'affichera pas les erreurs qui viennent de MySQL. Si vous sélectionnez une base qui n'existe pas, vous aurez l'erreur suivante :

Mauvaise base

```
Access denied for user 'tuto'@'localhost' to database 'developpez'
```

III - Passage d'une requête SELECT, récupération d'une ressource

C'est uniquement une fois connecté que vous pouvez passer une requête au serveur. C'est la fonction `mysql_query` qui est utilisée :

passage de la requête

```
$query = "SELECT * FROM `tuto_table`";  
$result = mysql_query($query, $link) or die($query . " - " . mysql_error());
```

C'est une bonne habitude que de stocker la requête dans une chaîne de caractères avant de la passer au serveur. Les noms des champs doivent être entre apostrophes inversées (backquotes), et les valeurs de chaînes entre apostrophes (guillemets simples). Encore une fois, `$link` est facultatif. Ici en cas de succès, on récupère `$result`, qui est encore une fois une *ressource*, qui n'est pas immédiatement exploitable mais dont nous allons pouvoir extraire les données retournées. Encore une fois, le "or die" affiche l'erreur MySQL en cas d'échec de la requête, ainsi que la requête effectivement passée au serveur, afin de vérifier que la chaîne de caractères, qui a pu être générée de manière complexe, contient bien la requête que l'on attend. Voilà un message d'erreur que l'on pourrait recevoir :

Erreur de syntaxe SQL

```
SELECTTTT * FROM `tuto_table` - You have an error in your SQL syntax;  
check the manual that corresponds to your MySQL server version for the right syntax to use near  
'SELECTTTT * FROM `tuto_table`' at line 1
```

Cette étape d'affichage de l'erreur est cruciale, c'est souvent à ce moment-là que les choses se gâtent avec MySQL, et si l'on n'affiche pas les erreurs on ne s'en rend pas compte, et on se demande plus tard pourquoi des erreurs bizarres arrivent !

IV - Exploitation de la ressource

IV-A - Opérations de base sur les ressources

La seule opération que nous allons voir sur la ressource résultat (à part l'extraction des résultats proprement dits, évidemment), est le comptage du nombre d'enregistrements retournés. Dans notre exemple nous avons fait un SELECT, et nous voulons savoir combien d'entrées ont été trouvées. La fonction `mysql_num_rows` fera notre bonheur :

Nombre d'enregistrements retournés

```
$nbResults = mysql_num_rows($result);  
echo $nbResults;
```

La fonction retourne un entier, bien évidemment, et il n'y a pas de difficulté particulière à son utilisation. Elle est bien pratique parfois pour débuser des erreurs dans les requêtes.

IV-B - Récupération d'un tableau

L'extension MySQL permet d'extraire les résultats sous différentes formes. La plus commune est le tableau, qui peut être associatif ou non. La fonction `mysql_fetch_array`, appliquée à l'objet résultat, extrait **un enregistrement et un seul**, et le retourne dans un tableau associatif (mais que l'on peut également utiliser comme un tableau non associatif). Les clés du tableau PHP sont les noms des champs dans la table MySQL. Si nous voulons retourner tous les enregistrements, alors il nous faut mettre la fonction dans une boucle while.

Récupération des résultats sous forme de tableau

```
while ($stab = mysql_fetch_array($result)) {  
    echo $stab['id'] . " : " . $stab['comment'];  
    echo "<br />";  
}
```

Trace du script

```
1 : valeur1  
2 : valeur2
```

Si l'on a pas pris soin d'afficher toutes les erreurs lors du passage de la requête, on risque en cas d'erreur de voir apparaître l'erreur suivante :

L'erreur classique qui apparaît quand on a été très vilain

```
Warning: mysql_fetch_array():  
supplied argument is not a valid MySQL result resource in [...]/script.php on line 14
```

Qu'est-ce que ça veut dire ? Exactement ce qui est écrit : la ressource sur laquelle on travaille n'est pas valide, c'est à dire qu'elle n'a pas été correctement retournée par `mysql_query`, probablement parce que la requête est défectueuse. Cette erreur ne devrait pas apparaître si l'on procède comme décrit dans ce tutoriel pour ce qui relève de l'affichage des erreurs. Toutes les fonctions qui s'appliquent à des ressources résultat, comme `mysql_num_rows`, peuvent générer ce type d'erreur.

IV-C - Récupération d'un objet

En remplaçant `mysql_fetch_array` par `mysql_fetch_object`, on peut récupérer un objet à la place d'un tableau. Il faut donc accéder aux données d'une manière différente par la suite, mais les principes sont exactement les mêmes,

ainsi que le risque de générer une erreur si l'on n'a pas fait attention en passant sa requête. Ici ce sont les attributs de l'objet qui portent le nom des champs de la table MySQL. La sortie est exactement la même que pour le script précédent.

Récupération des résultats sous forme d'objet PHP

```
while ($stab = mysql_fetch_object($result)) {  
    echo $stab->id . " : " . $stab->comment;  
    echo "<br />";  
}
```

V - Autres types de requêtes

Pour les requêtes de type SHOW, DESCRIBE ou EXPLAIN, une ressource est retournée par `mysql_query`, de la même manière que pour SELECT. Pour les autres types de requêtes, en revanche, seul TRUE est renvoyé en cas de succès. Le nombre de lignes modifiées ou supprimées par un UPDATE, un DELETE, un INSERT ou un UPDATE s'obtient en appelant la fonction `mysql_affected_rows`, sans paramètre (ou bien en passant en paramètre, si besoin est, la ressource de connexion, `$link` dans notre exemple). Cette fonction retourne uniquement l'entier correspondant.

VI - Fermeture de la connexion MySQL

Soyez gentils, éteignez la lumière en partant, et fermez la connexion !

Fermeture de connexion

```
mysql_close($link);
```

Encore une fois, le passage de la ressource de connexion en paramètre est optionnel si l'on en utilise une seule.

VII - Script complet

```
<?php

$link = mysql_connect("localhost", "developpez", "pass");
mysql_select_db("developpez", $link) or die(mysql_error());

$query = "SELECT * FROM `tuto_table`";
$result = mysql_query($query, $link) or die($query . " - " . mysql_error());

$numResults = mysql_num_rows($result);
echo $numResults;
echo "<br /><br />";

while ($tab = mysql_fetch_array($result)) {
    echo $tab['id'] . " : " . $tab['comment'];
    echo "<br />";
}

echo "<br />";

$result = mysql_query($query, $link) or die($query . " - " . mysql_error());

while ($tab = mysql_fetch_object($result)) {
    echo $tab->id . " : " . $tab->comment;
    echo "<br />";
}

mysql_close($link);

?>
```

VIII - Les informations utiles en cas de problème

Si vous avez des problèmes lorsque vous travaillez avec PHP/MySQL, les informations qui vous seront le plus utiles sont les affichages des erreurs que nous avons mis en place, ainsi que le réaffichage de la requête envoyée par `mysql_query`. Si vous devez demander de l'aide sur le forum de [developpez.com](#) ou n'importe où ailleurs sur internet, veuillez à respecter les principes simples donnés ici et à bien donner tous les affichages lorsque vous posez votre question.

Bon développement !

IX - Remerciements

Je remercie pour leur effort de relecture et leurs remarques constructives les personnes suivantes : [JWhite](#), [Maxoo](#), [Yogui](#).

X - Références

[Protégez vos caractères spéciaux en passant vos requêtes !](#)

[La documentation officielle de l'interface MySQL de PHP](#)

[La FAQ PHP/MySQL de developpez.com](#)

[Récupérer les valeurs de champ auto_increment](#)

[La section SQL de developpez.com](#)

[La documentation officielle de MySQL](#)