mySQL

Table des matières

L'intérêt majeur de PHP est son interfaçage avec un grand nombre de bases de données d'une manière relativement simple et efficace.

1. Introduction

MySQL est un SGBDR, qui utilise le langage SQL. C'est un des SGBDR les plus utilisés car popularité est due en grande partie au fait qu'il s'agit d'un logiciel Open Source.

Le développement de MySQL commence en 1994 par David Axmark et Michael Widenius. En 2008, MySQL AB est rachetée par la société Sun Microsystems, qui est elle-même rachetée par Oracle Corporation en 2010.

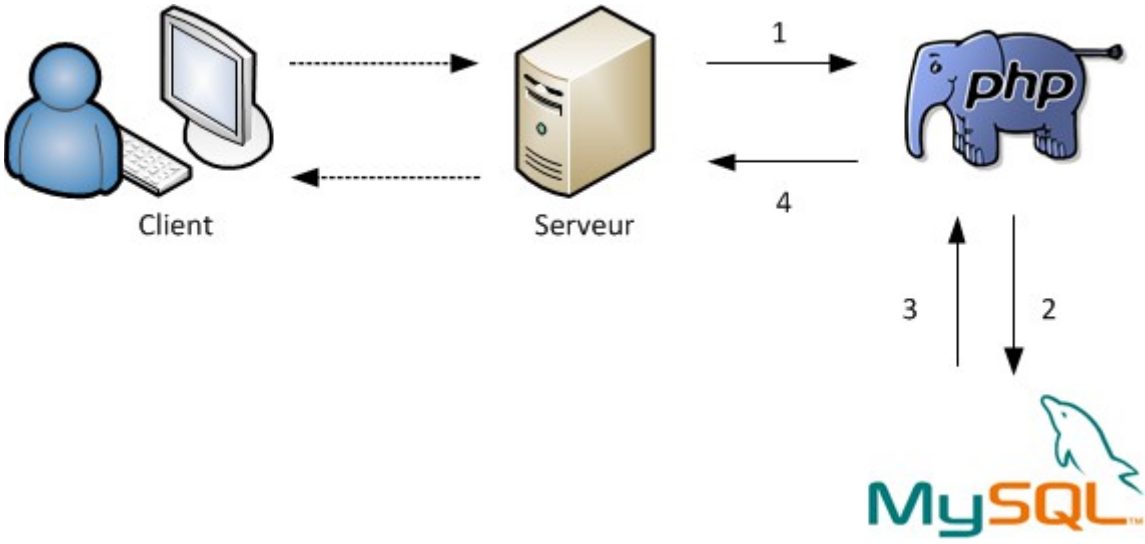
MySQL ne suit pas toujours la norme officielle. Certaines syntaxes peuvent donc être propres à MySQL et ne pas fonctionner sous d'autres SGBDR. Par ailleurs, il n'implémente pas certaines fonctionnalités avancées, qui peuvent être utiles pour un projet un tant soit peu ambitieux.

Il existe des dizaines de SGBDR, chacun ayant ses avantages et ses inconvénients. Voici succinctement quatre d'entre eux, parmi les plus connus :

- Oracle database, édité par Oracle Corporation.
- PostgreSQL, logiciel Open Source qui a longtemps été disponible uniquement sous Unix.
- MS Access, édité par Microsoft qui ne fonctionne que sous Windows et qui n'est pas adapté pour gérer un grand volume de données.
- SQLite, ce logiciel stocke toutes les données dans de simples fichiers.

2. PHP et les bases de Données

Si votre application PHP a besoin de communiquer avec un serveur de base de données, vous devez écrire du code pour pouvoir vous connecter, envoyer des requêtes au serveur, etc. Un logiciel est nécessaire pour assurer l'interface que PHP va utiliser et gérer les communications entre votre application et le serveur de base de données : éventuellement, des bibliothèques intermédiaires sont nécessaires. Ces logiciels sont appelés de manière générique des connecteurs, car ils permettent de se connecter à un serveur de base de données.



Voici ce qui peut se passer lorsque le serveur a reçu une demande d'un client :

1. le serveur utilise toujours PHP, il lui fait donc passer le message ;

2. PHP effectue les actions demandées et se rend compte qu'il a besoin de MySQL. En effet, le code PHP contient à un endroit « Va demander à MySQL d'enregistrer ce message ». Il fait donc passer le travail à MySQL ;
3. MySQL fait le travail que PHP lui avait soumis et lui répond « O.K., c'est bon ! » ;
4. PHP renvoie au serveur que MySQL a bien fait ce qui lui était demandé.

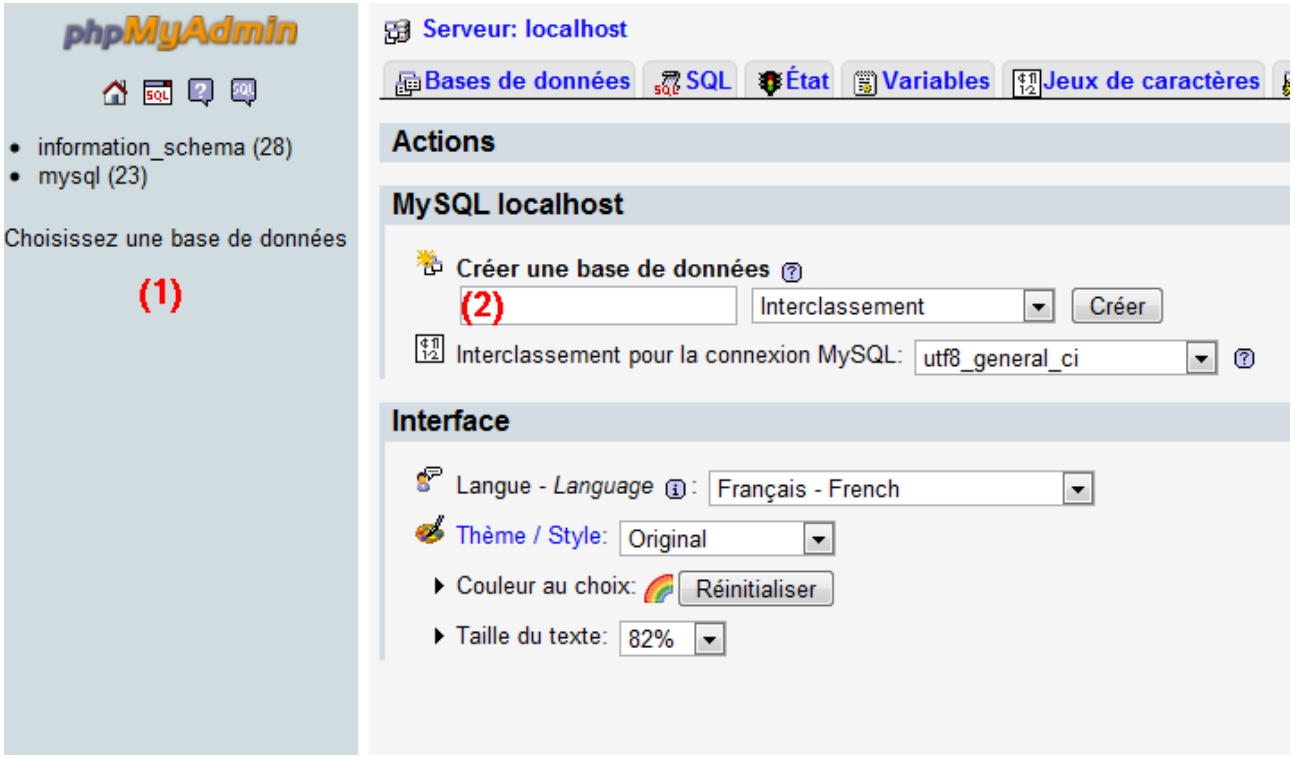
3. phpMyAdmin

3.1. Créer une base de données

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes de gestion de base de données MySQL réalisée en PHP. Il s'agit de l'une des plus célèbres interfaces pour gérer une base de données MySQL sur un serveur PHP.

Cette interface permet d'exécuter, très facilement et sans grandes connaissances dans le domaine des bases de données, de nombreuses requêtes comme les créations de table de données, les insertions, les mises à jour, les suppressions, les modifications de structure de la base de données. Ce système est très pratique pour sauvegarder une base de données sous forme de fichier .sql et ainsi transférer facilement ses données. De plus celui-ci accepte la formulation de requêtes SQL directement en langage SQL, cela permet de tester ses requêtes par exemple lors de la création d'un site et ainsi de gagner un temps précieux.

L'accueil de phpMyAdmin ressemble à la figure suivante :



Deux endroits importants sont signalés par des numéros :

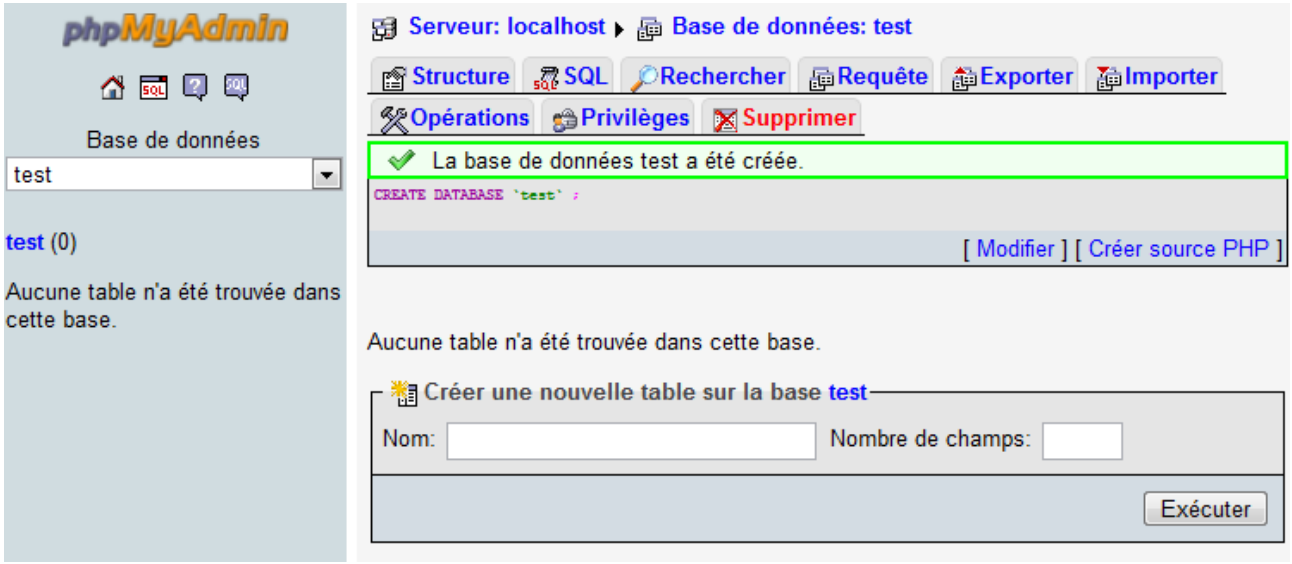
1. **Liste des bases** : c'est la liste de vos bases de données. Le nombre entre parenthèses est le

nombre de tables qu'il y a dans la base.

2. **Créer une base** : pour créer une nouvelle base de données, entrez un nom dans le champ de formulaire à droite, cliquez sur « Créer » et hop ! c'est fait.

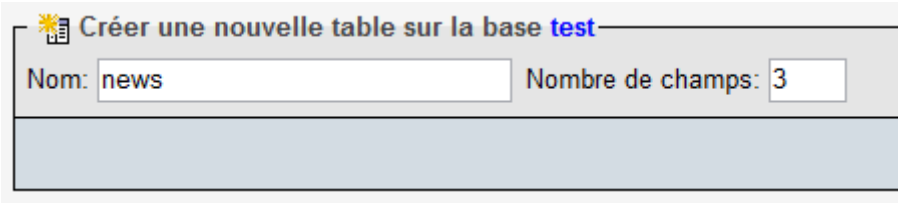
Pour le moment, deux bases existent déjà : information_schema et mysql. N'y touchez pas, elles servent au fonctionnement interne de MySQL.

Nous allons maintenant créer une nouvelle base test dans laquelle nous travaillerons tout le temps par la suite. Utilisez le formulaire à droite pour créer cette base : entrez le nom test et cliquez sur le bouton Créer.



La base de test a été créée, vide.

Dans le champ « Créer une nouvelle table sur la base test », entrez le nom news et le nombre de champs 3, comme vous le montre la figure suivante.



Cliquez sur « Exécuter ».

La table n'est pas immédiatement créée : il faut maintenant indiquer le nom des champs et les données qu'ils peuvent contenir. Je vous propose de faire simple car pour l'instant on cherche juste à tester phpMyAdmin. Pour cette table, on va créer les trois champs suivants :

1. id : numéro d'identification qui servira de clef primaire.
2. titre : ce champ contiendra le titre de la news.
3. contenu : ce champ contiendra la news elle-même.

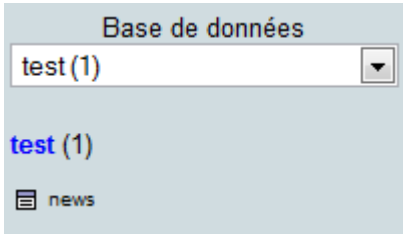
Champ	id	titre	contenu
Type	INT	VARCHAR	TEXT
Taille/Valeurs*1		255	
Défaut2	Aucun	Aucun	Aucun
Interclassement			
Attributs			
Null	☐	☐	☐
Index	PRIMARY	---	---
AUTO_INCREMENT	☑	☐	☐
Commentaires			

Chaque colonne représente un champ. Nous avons demandé trois champs, il y a donc trois colonnesdont les sections les plus intéressantes sont :

1. Champ : permet de définir le nom du champ ;
2. Type : le type de données que va stocker le champ (nombre entier, texte, date…) ;
3. Taille/Valeurs : permet d'indiquer la taille maximale du champ, utile pour le type VARCHAR notamment, afin de limiter le nombre de caractères autorisés ;
4. Index : active l'indexation du champ.
5. AUTO_INCREMENT : permet au champ de s'incrémenter tout seul à chaque nouvelle entrée (cocher AUTO_INCREMENT et définir un index PRIMARY sur le champ id).

Cliquez sur le bouton Sauvegarder en bas de la page et la table est créée !

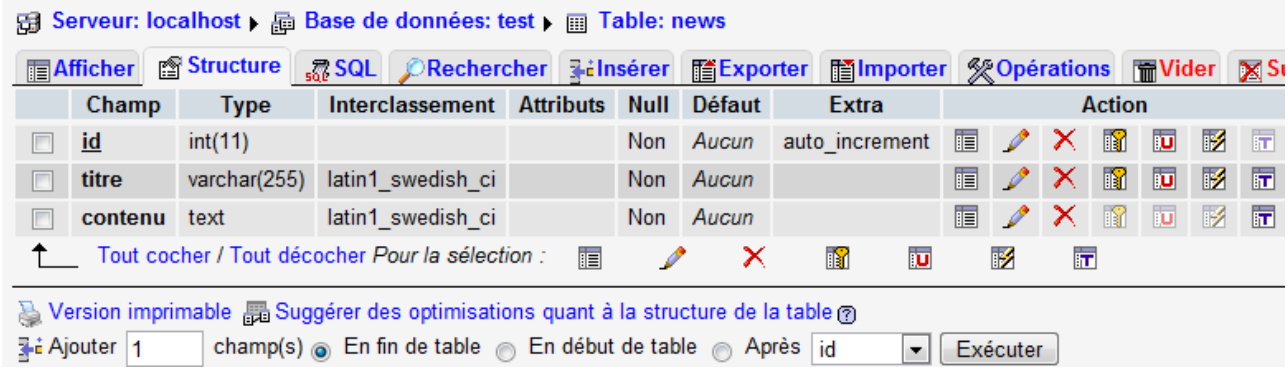
3.2. Modifier une table



À gauche de l'écran, la table « news » que vous venez de créer devient visible.

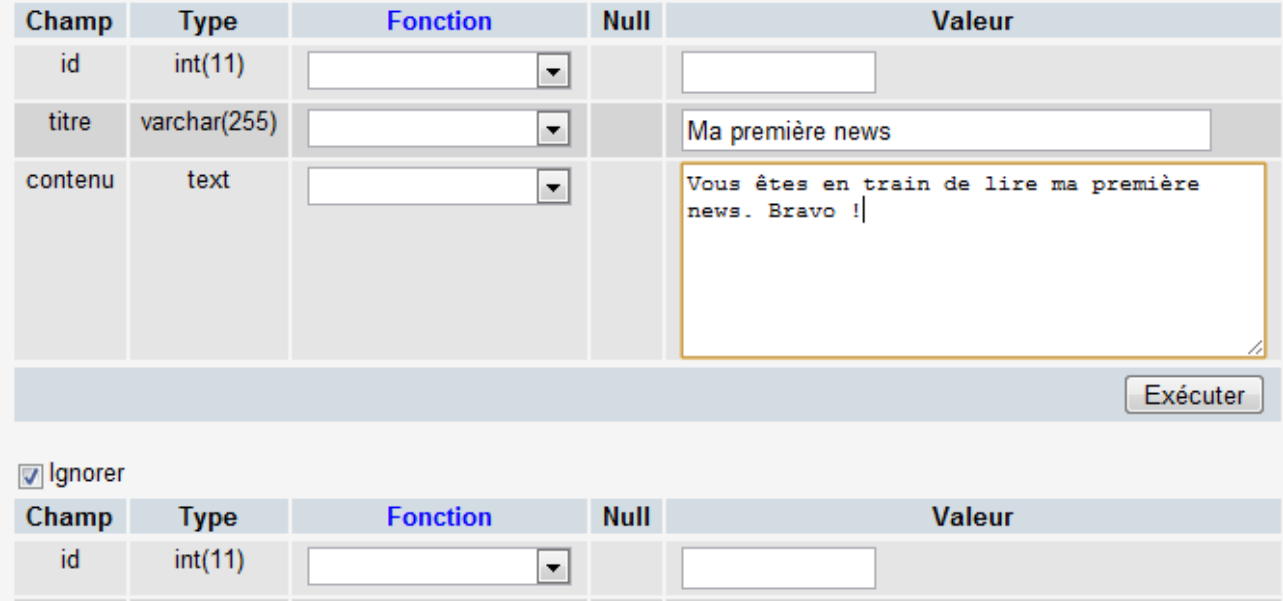
- Si vous cliquez sur le mot « news », le contenu de la table s'affiche à droite de l'écran.
- Si vous cliquez sur la petite image de tableau à gauche, phpMyAdmin vous présentera la structure de la table.

Actuellement, comme la table est vide, c'est la structure de la tablequi s'affichera dans les deux cas.



Ce tableau rappelle de quels champs est constituée la table : c'est sa structure. Le champ id est souligné car c'est la clé primaire de la table.

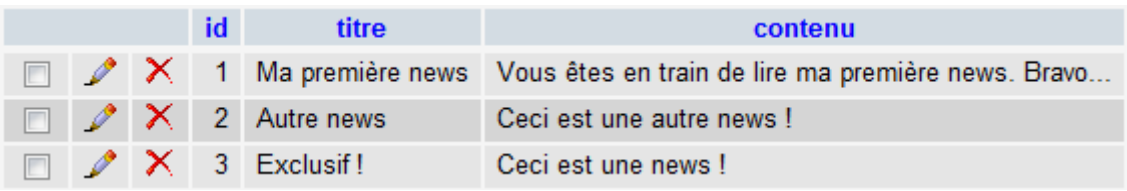
Les onglets du haut : « Insérer » va nous permettre d'insérer la première news :



Seule la colonne « Valeur » nous intéresse. Vous pouvez entrer une valeur pour chacun des trois champs. Il n'y a pas de valeur pour l'id car le numéro d'id est automatiquement calculé grâce à l'option auto_increment.

Une fois le texte saisi, cliquez sur le bouton « Exécuter » de la page.

Pour Afficher le contenu de la table, cliquer soit sur l'onglet « Afficher », en haut, soit sur le nom de la table dans le menu à gauche :



	id	titre	contenu
	1	Ma première news	Vous êtes en train de lire ma première news. Bravo...
	2	Autre news	Ceci est une autre news !
	3	Exclusif !	Ceci est une news !

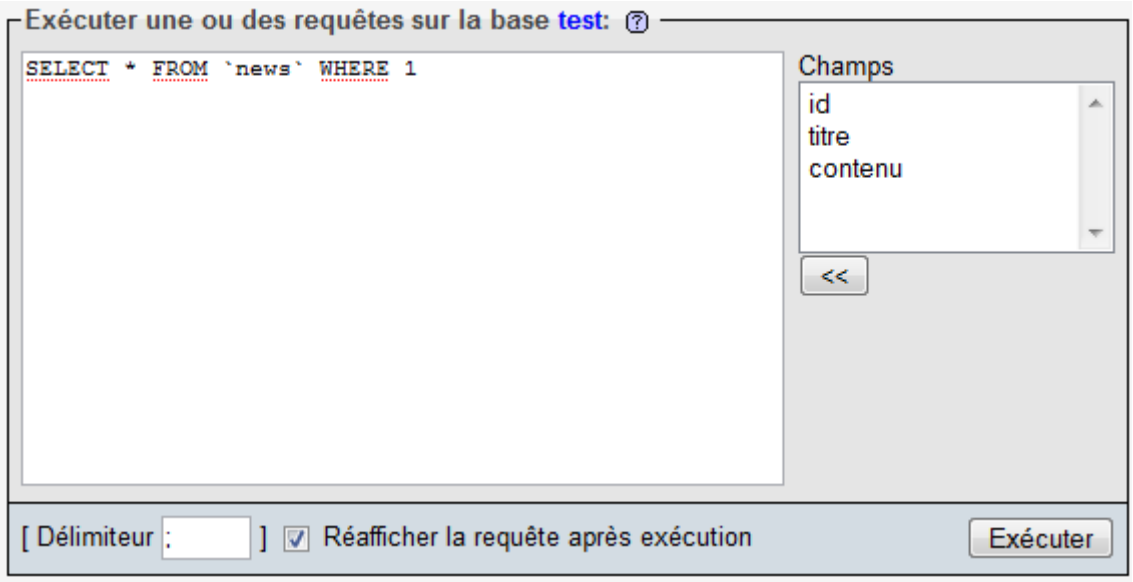
Remarque : les numéros d'id se sont incrémentés automatiquement.

3.3. Autres opérations

Nous avons jusqu'ici découvert le rôle de trois onglets :

- Afficher : affiche le contenu de la table ;
- Structure : présente la structure de la table (liste des champs) ;
- Insérer : permet d'insérer de nouvelles entrées dans la table.

3.3.1. SQL

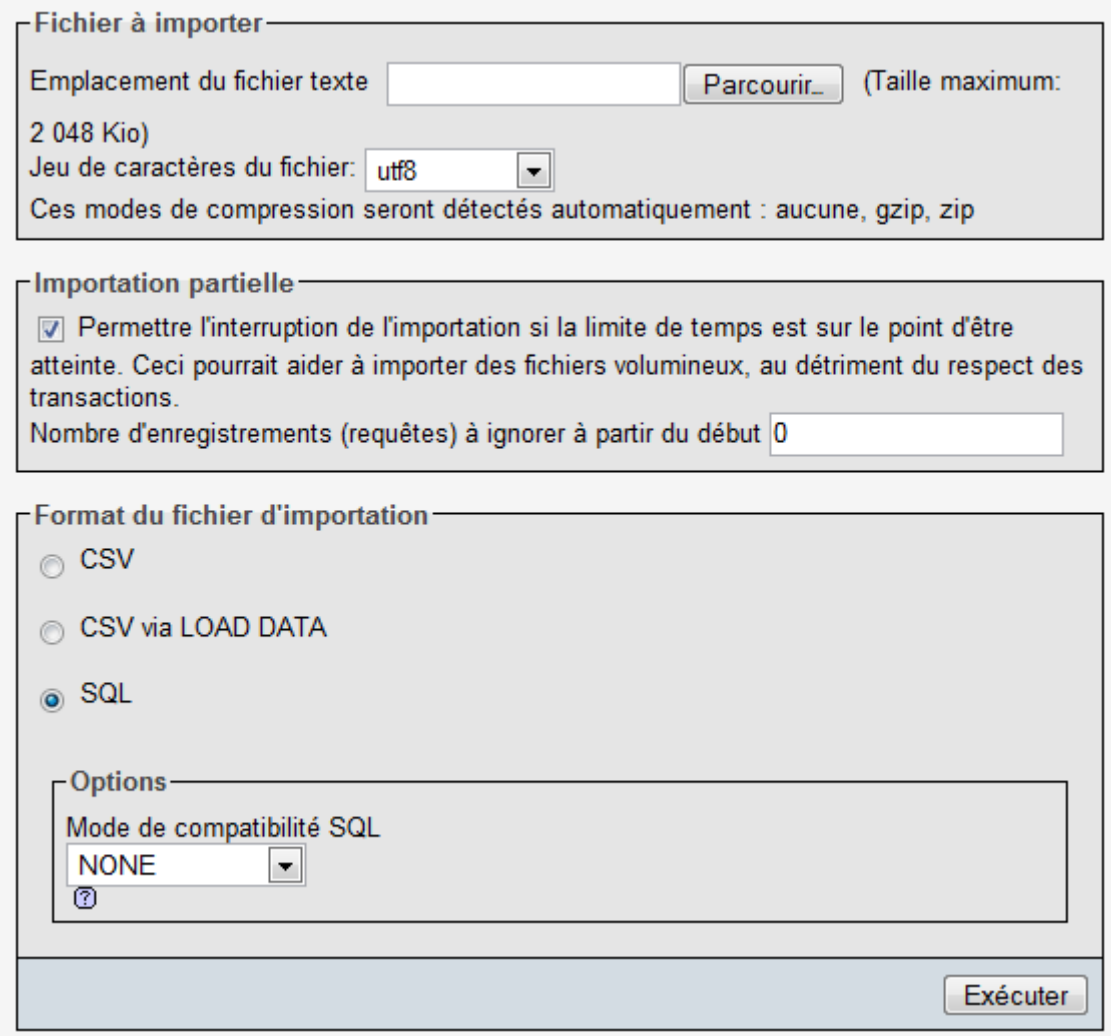


C'est ici que vous pouvez exécuter des requêtes SQL.

3.3.2. Importer

Dans la page qui s'affiche, vous pouvez envoyer un fichier de requêtes SQL (généralement un fichier .sql) à MySQL pour qu'il les exécute.

Il faudra utiliser cet onglet pour restorer une base de données (cf § 3.3.3.).



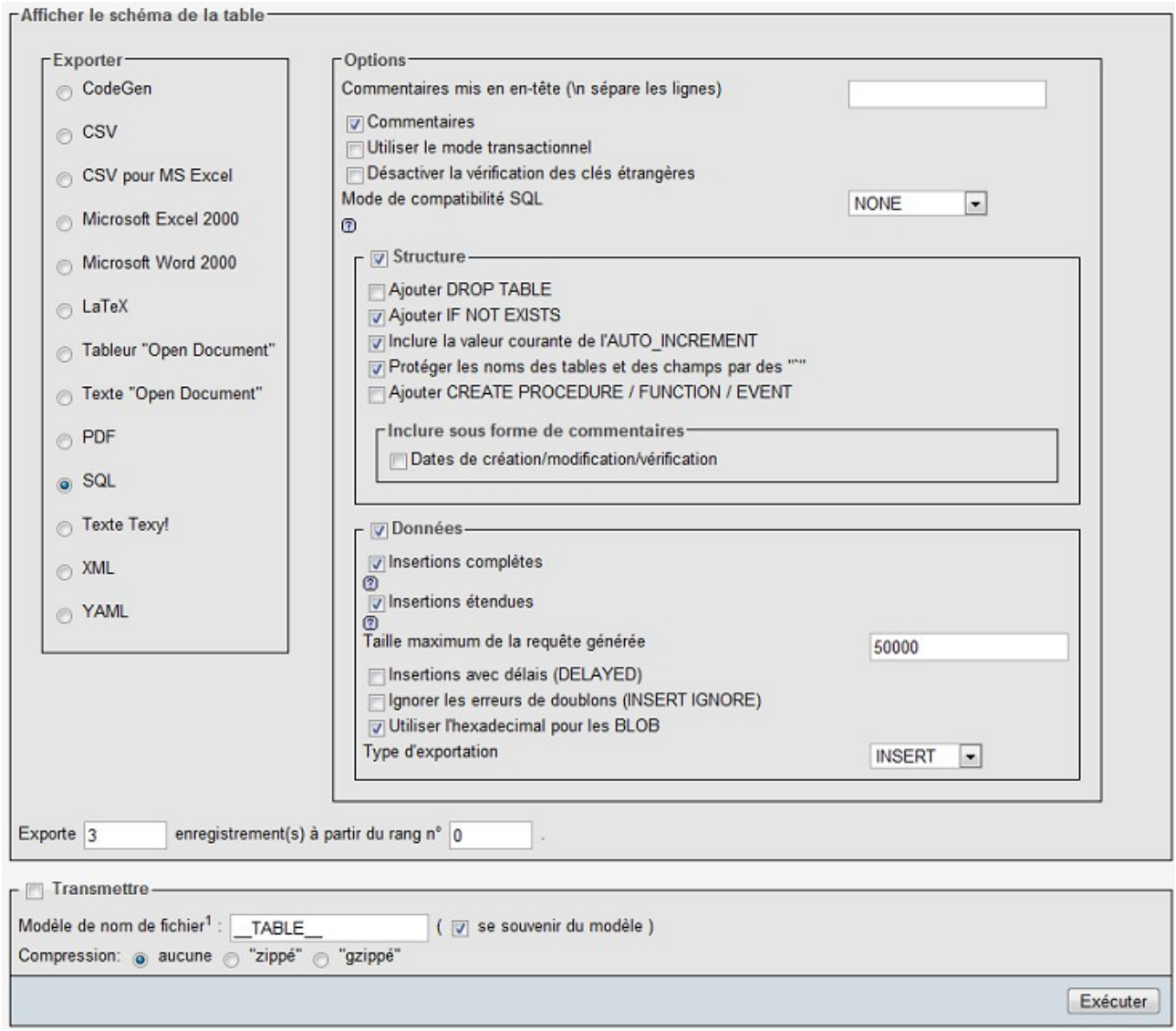
Le premier champ permet d'indiquer un fichier sur le disque dur contenant des requêtes SQL à exécuter.

Cliquez ensuite sur le bouton « Exécuter » tout en bas.

3.3.3. Exporter

On peut s'en servir pour deux choses :

1. transmettre la base de données sur Internet pour l'installer sur un serveur distant.
2. faire une copie de sauvegarde de la base de données sous forme de fichier texte .sql pour la restorer en cas de besoin.



La structure d'une table se résume en quelques lignes : ce sont en fait les noms des champs, leurs types, etc. Les données correspondent aux entrées. Pour faire une sauvegarde complète, il faut donc prendre la structure ET les données.

Puis cocher la case « Transmettre » en bas. À noter que vous pouvez demander une compression, ce qui est utile si votre table est très grosse.

3.3.4. Opérations

Vous pouvez effectuer diverses opérations sur la table. Exemple :

- changer le nom de la table : indiquez le nouveau nom pour cette table ;
- déplacer la table vers : si vous voulez placer cette table dans une autre base de données ;
- copier la table : faire une copie de la table, dans une autre base ou dans la même (attention : dans ce cas, il faudra qu'elle ait un nom différent) ;
- optimiser la table : à force d'utiliser une table, surtout si elle est grosse, on finit par avoir des « pertes » qui font que la table n'est plus bien organisée. Un clic là-dessus et hop ! c'est de

nouveau arrangé.

3.3.5. Vider

Vide tout le contenu de la table. Toutes les entrées vont disparaître, seule la structure de la table restera (c'est-à-dire les champs).

Attention ! Il n'est pas possible d'annuler cette opération !

3.3.6. Supprimer

Pour supprimer la totalité de la table (structure et données). Il n'est pas possible d'annuler cette opération !

4. Se connecter à une base de données

4.1. Connexion en PHP

Pour pouvoir travailler avec une base de données en PHP, il faut d'abord s'y connecter.

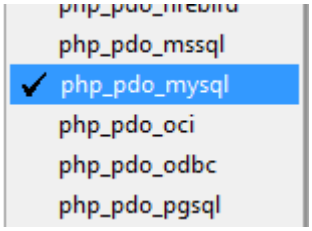
Il va donc falloir que PHP s'authentifie : on dit qu'il établit une connexion avec MySQL. Une fois que la connexion sera établie, vous pourrez faire toutes les opérations que vous voudrez sur la base de données.

PHP propose plusieurs moyens de se connecter à une base de données MySQL.

- L'extension mysql_ : ce sont des fonctions qui permettent d'accéder à une base de données MySQL et donc de communiquer avec MySQL. Leur nom commence toujours par mysql_. Toutefois, ces fonctions sont vieilles et on recommande de ne plus les utiliser aujourd'hui.
- L'extension mysqli_ : ce sont des fonctions améliorées d'accès à MySQL. Elles proposent plus de fonctionnalités et sont plus à jour.
- L'extension PDO[1] : c'est un outil complet qui permet d'accéder à n'importe quel type de base de données. On peut donc l'utiliser pour se connecter aussi bien à MySQL que PostgreSQL ou Oracle.

Nous allons utiliser PDO car c'est cette méthode d'accès aux bases de données qui va devenir la plus utilisée dans les prochaines versions de PHP. D'autre part, le gros avantage de PDO est que vous pouvez l'utiliser de la même manière pour vous connecter à n'importe quel autre type de base de données (PostgreSQL, Oracle…).

Normalement, PDO est activé par défaut. Pour le vérifier, faites un clic gauche sur l'icône de WAMP[2] dans la barre des tâches, puis allez dans le menu PHP / Extensions PHP et vérifiez que php_pdo_mysql est bien coché.



Si vous n'utilisez pas WAMP, vous pouvez ouvrir le fichier de configuration de PHP (php.ini) et

1 Php Data Object. Il s'agit une couche d'abstraction des fonctions d'accès aux bases de données.
2 Windows, Apache, MySQL, PHP : néologisme basé sur LAMP.

rechercher la ligne qui contient php_pdo_mysql. Enlevez le point-virgule devant s'il y en a un pour activer l'extension :

```
;extension=php_pdo_firebird.dll
;extension=php_pdo_mssql.dll
extension=php_pdo_mysql.dll
;extension=php_pdo_oci.dll
;extension=php_pdo_odbc.dll
```

Sous GNU/Linux, recherchez la ligne qui commence par pdo_mysql.default_socket et complétez-la comme ceci :

```
pdo_mysql.default_socket = /opt/lampp/var/mysql/mysql.sock
```

Enregistrez le fichier puis redémarrez PHP.

4.2. Se connecter à MySQL avec PDO

Pour se connecter à MySQL, il faut quatre renseignements :

1. le nom de l'hôte : c'est l'adresse de l'ordinateur où MySQL est installé (ex : sql.hebergeur.com) ;
2. la base : c'est le nom de la base de données à laquelle vous voulez vous connecter (dans notre cas, la base s'appelle test) ;
3. le login : il permet de vous identifier. Le plus souvent, c'est le même login utilisé pour le FTP ;
4. le mot de passe.

Pour l'instant, nous faisons des tests sur notre ordinateur « en local ». Par conséquent, le nom de l'hôte sera localhost.

Quant au login et au mot de passe, par défaut le login est root et il n'y a pas de mot de passe.

Voici donc comment on doit faire pour se connecter à MySQL via PDO sur la base test :

```
<?php
$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '');
?>
```

La ligne de code crée un objet$bdd qui représente la connexion à la base de données. On crée la connexion en indiquant dans l'ordre dans les paramètres :

1. le nom d'hôte (localhost) ;
2. la base de données (test) ;
3. le login (root) ;
4. le mot de passe (ici il n'y a pas de mot de passe, donc chaîne vide).

Le premier paramètre (qui commence par mysql) s'appelle le DSN : Data Source Name. C'est généralement le seul qui change en fonction du type de base de données auquel on se connecte.

S'il y a une erreur (vous vous êtes trompés de mot de passe ou de nom de base de données, par

exemple), PHP risque d'afficher toute la ligne qui pose l'erreur, ce qui inclut le mot de passe !

Il est donc préférable de traiter l'erreur :

```
<?php
try
{
	$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '');
}

catch (Exception $e)
{
	die('Erreur : ' . $e->getMessage());
}
?>
```

PHP essaie d'exécuter les instructions à l'intérieur du bloc try. S'il y a une erreur, il rentre dans le bloc catch et fait ce qu'on lui demande (ici, on arrête l'exécution de la page en affichant un message décrivant l'erreur).

5. Lire des données

5.1. Récupérer les données

Une fois créé l'objet représentant la connexion à la base, on demandeà effectuer une requête sur la base de données comme ceci :

```
$reponse = $bdd->query('Tapez votre requête SQL ici');
```
[3]

On récupère ce que la base de données nous a renvoyé dans un autre objet que l'on a appelé ici $reponse.

Exemple :

```
<?php
$reponse = $bdd->query('SELECT * FROM test');
?>
```

$reponse contient maintenant la réponse de MySQL qui correspondant à une entrée de la table.

Pour récupérer une entrée, on utilise la méthode fetch(), qui renvoie la première ligne ou faux (false) lorsqu'elle est arrivée à la fin des données, c'est-à-dire que toutes les entrées ont été passées en revue.

```
<?php
$donnees = $reponse->fetch();
?>
```
[4]

$donnees est un tableau qui contient champ par champ les valeurs de la première entrée. Par exemple, si on s'intéresse au champ titre, in utilisera l'array $donnees['titre'].

3 query en anglais signifie « requête ».
4 fetch en anglais signifie « va chercher ».

Il faut faire une boucle pour parcourir les entrées une à une. Chaque fois que $reponse->fetch() est appelée, on passe à l'entrée suivante. La boucle est donc répétée autant de fois qu'il y a d'entrées dans la table.

Exemple :

```
<?php
try
{
	// On se connecte à MySQL
	$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '');
}

catch(Exception $e)
{
	// En cas d'erreur, on affiche un message et on arrête tout
	die('Erreur : '.$e->getMessage());
}

// Si tout va bien, on peut continuer
// On récupère tout le contenu de la table jeux_video
$reponse = $bdd->query('SELECT * FROM test');

// On affiche chaque entrée une à une
while ( $donnees = $reponse->fetch() )
{
	print("titre : ". $donnees['titre'] ."<br />") ;
}

// Termine le traitement de la requête
$reponse->closeCursor();
?>
```

Remarque : la méthode closeCursor() provoque la « fermeture du curseur d'analyse des résultats » afin d'éviter d'avoir des problèmes à la requête suivante.

5.2. Construire des requêtes en fonction de variables

Les variables ne transitent pas toujours via un formulaire mais bien souvent par l'URL via la méthode GET. Dans ce cas, les variables et les valeurs qu'elles prennent sont déclarées directement dans l'URL c'est à dire via la balise de lien HTML . Les variables sont ensuite exploitables sur la page cible en PHP.

Cependant, si la variable $_GET['possesseur'] a été modifiée par un visiteur, il y a un gros risque de faille de sécurité qu'on appelle injection SQL[5]. Pour y remédier, nous allons utiliser les requêtes préparées.

Le système de requêtes préparées a l'avantage d'être beaucoup plus sûr mais aussi plus rapide pour la base de données si la requête est exécutée plusieurs fois

5 http://fr.wikipedia.org/wiki/Injection_SQL

4.4.1. marqueurs « ? »

Dans un premier temps, on va « préparer » la requête sans sa partie variable, que l'on représentera avec un marqueur sous forme de point d'interrogation. Au lieu d'exécuter la requête avec query(), on appelle ici prepare().

La requête est alors prête, sans sa partie variable. Nous allons ensuite exécuter la requête en appelant la méthode execute() et en lui transmettant la liste des paramètres :

```
<?php
$req = $bdd->prepare("SELECT contenu FROM test WHERE titre = '?' ");
$req->execute(array($_GET['titre']));
?>
```

La requête est alors exécutée à l'aide des paramètres que l'on a indiqués sous forme de tableau.

S'il y a plusieurs marqueurs, il faut indiquer les paramètres dans le bon ordre :

```
<?php
$req = $bdd->prepare("SELECT contenu FROM test WHERE titre = '?' LIMIT ?");
$req->execute(array($_GET['titre'], $_GET['limit']));
?>
```

Le premier point d'interrogation de la requête sera remplacé par le contenu de la variable $_GET['titre'], et le second par le contenu de $_GET['limit']. Le contenu de ces variables aura été automatiquement sécurisé pour prévenir les risques d'injection SQL.

4.4.2. marqueurs nominatifs

Si la requête contient beaucoup de parties variables, il peut être plus pratique de nommer les marqueurs plutôt que d'utiliser des points d'interrogation.

```
<?php
$req = $bdd->prepare('SELECT contenu FROM test WHERE titre = :titre AND limit <= :limit');
$req->execute(array('titre' => $_GET['titre'], 'limit' => $_GET['limit']));
?>
```

Les points d'interrogation ont été remplacés par les marqueurs nominatifs :titre et :limit.

Ces marqueurs sont remplacés par les variables à l'aide d'un array associatif.

5.3. Traquer les erreurs

Lorsqu'une requête SQL « plante », PHP dira qu'il y a eu une erreur à la ligne du fetch :

Fatal error: Call to a member function fetch() on a non-object in C:\wamp\www\tests\index.php on line 13

Ce n'est pas la ligne du fetch qui est en cause : c'est plutôt la requête SQL quelques lignes plus haut.

Pour afficher des détails sur l'erreur avec un message beaucoup plus clair, il faut activer les erreurs lors de la connexion à la base de données via PDO à l'aide d'un paramètre :

```
<?php
$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '', array(PDO::ATTR_ERRMODE =>
```

```
PDO::ERRMODE_EXCEPTION));
?>
```

6. Écrire des données

On peut ajouter et modifier des données dans la base à l'aide des requêtes SQL fondamentales : INSERT, UPDATE et DELETE.

Utilisons ces requêtes SQL au sein d'un script PHP. Cette fois, au lieu de faire appel à query(), on va utiliser exec() qui est prévue pour exécuter des modifications sur la base de données.

6.1. INSERT : ajouter des données

La requête INSERT INTO permet d'ajouter une entrée.

Vous remarquerez que pour le premier champ (ID), j'ai laissé des apostrophes vides. C'est voulu : le champ a la propriété auto_increment, MySQL mettra donc le numéro d'ID lui-même. On pourrait même se passer du champ ID dans la requête :

```
<?php
try
{
	$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '');
}
catch(Exception $e)
{
	die('Erreur : '.$e->getMessage());
}

// On ajoute une entrée dans la table test
$bdd->exec("INSERT INTO test('', 'nouvelle news', 'J\'ai appris à programmer en PHP !')");
?>
```

Remarques :

- le premier champ (ID) a des apostrophes vides car le champ a la propriété auto_increment. MySQL mettra donc le numéro d'ID lui-même.
- il faut rajouter des antislashs \ si la valeur du champ contient des apostrophes.

NB : si on choisit d'utiliser une requête préparée, le fonctionnement est en fait exactement le même que dans le chapitre précédent.

```
<?php
$req = $bdd->prepare('INSERT INTO test VALUES(:titre, :contenu)');

$req->execute(array(
	'titre' => $titre,
	'contenu' => $contenu
	));
?>
```

6.2. UPDATE : modifier des données

La requête UPDATE permet de modifier une entrée.

De la même manière, en PHP on fait appel à exec() pour effectuer des modifications :

```
<?php
$nb_modifs = $bdd->exec("
        UPDATE test SET contenu = 'J\'ai appris à programmer en PHP !'
        WHERE titre = 'nouvelle news'
        LIMIT 1
        ");

echo $nb_modifs . ' entrée a été modifiée<br/>';
?>
```

Remarque : cet appel renvoie le nombre de lignes modifiées.

6.3. DELETE : supprimer des données

La requête DELETE FROM permet de supprimer une entrée. Il n'y a aucun moyen de récupérer les données.

De la même manière, en PHP on fait appel à exec() pour effectuer des modifications :

```
<?php
$bdd->exec("DELETE FROM test WHERE titre = 'nouvelle news' LIMIT 1");
?>
```

NB : si la clause WHERE est oubliée, toutes les entrées seront supprimées. Cela équivaut à vider la table.