

Programmation WEB avec PHP

Matière : Programmation WEB avec PHP

Date : 13.06.2022

Formateur : Malerba Sylvain

Auteur : Malerba Sylvain

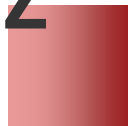
SOMMAIRE

01



La base de
données

02



Connexion à la
base de données

03



Ajout,
Modification et
Suppression

04



Récupération des
données

01

La base de données

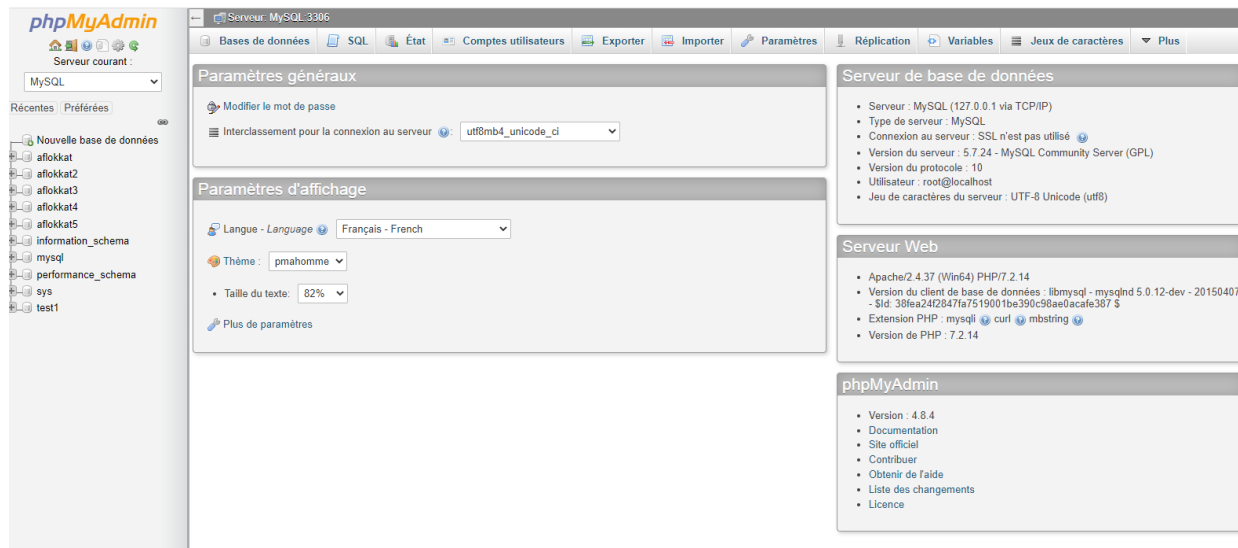
La base de données est:

- **Installée par défaut avec MAMP/WAMP/LAMP**
- **Open Source**
- **Facile à mettre en place**
- **Adaptée aux sites Web**
- **Utilitaire de gestion pré-installé : phpMyAdmin**

01

La base de données

- Accessible à l'adresse <http://localhost/phpmyadmin>



The screenshot displays the phpMyAdmin web interface. The top navigation bar includes tabs for 'Bases de données', 'SQL', 'État', 'Comptes utilisateurs', 'Exporter', 'Importer', 'Paramètres', 'Réplication', 'Variables', 'Jeux de caractères', and 'Plus'. The left sidebar shows a tree view of databases, including 'Nouvelle base de données', 'aflokkat', 'aflokkat2', 'aflokkat3', 'aflokkat4', 'aflokkat5', 'information_schema', 'mysql', 'performance_schema', 'sys', and 'test1'. The main content area is titled 'Serveur: MySQL-3306' and contains three panels:

- Paramètres généraux**: Includes a 'Modifier le mot de passe' link and a dropdown for 'Interclassement pour la connexion au serveur' set to 'utf8mb4_unicode_ci'.
- Paramètres d'affichage**: Includes a language dropdown set to 'Français - French' and a theme dropdown set to 'pmahomme'. There is also a 'Taille du texte' dropdown set to '82%' and a 'Plus de paramètres' link.
- Serveur de base de données**: Lists server details:
 - Serveur : MySQL (127.0.0.1 via TCP/IP)
 - Type de serveur : MySQL
 - Connexion au serveur : SSL n'est pas utilisé
 - Version du serveur : 5.7.24 - MySQL Community Server (GPL)
 - Version du protocole : 10
 - Utilisateur : root@localhost
 - Jeu de caractères du serveur : UTF-8 Unicode (utf8)
- Serveur Web**: Lists web server details:
 - Apache/2.4.37 (Win64) PHP/7.2.14
 - Version du client de base de données : libmysql - mysqlnd 5.0.12-dev - 20150407 - \$Id: 38fe2428471a7519001be390c98ae0acafe387 \$
 - Extension PHP : mysqli, curl, mbstring
 - Version de PHP : 7.2.14
- phpMyAdmin**: Lists application details:
 - Version : 4.8.4
 - Documentation
 - Site officiel
 - Contribuer
 - Obtenir de l'aide
 - Liste des changements
 - Licence

- **Quelques définitions:**

Clé primaire : une **clé primaire** est la donnée qui permet d'identifier de manière unique un enregistrement dans une table.

Clé étrangère : La **clé étrangère** fait référence à la **clé primaire** d'une autre table.

Jointure: Les **jointures** en **SQL** permettent d'associer plusieurs tables dans une même requête

01

Exercice

- Créez une table « users » avec un ensemble de données propres à un utilisateur (nom, prénom, login, password etc...)

01

Exercice

- **Créez une table « commandes » permettant de stocker une liste de commandes.**

Cette table doit être liée avec la table « users ».

02

Connexion à la base de données

02

Connexion à la base de données (procédural)

Fonction

**mysqli_connect() et
mysqli_close()**

```
<?php

class Database
{
    private $db;

    public function __construct(){
        try {
            $this->db = mysqli_connect("localhost", "monLogin", "monMDP",
                                     "aflokkat");
        } catch (RuntimeException $e) {
            error_log($e->getMessage());
            exit(0);
        }
    }

    public function connexion(){
        return $this->db;
    }
}
```

Connexion à la base de données (PDO)

- **PDO (PHP Data Objects)**
 - **wrapper permettant de se connecter à plusieurs SGBD**
 - **pas besoin de réécrire le code si on change de SGBD**
 - **sécurise face aux injections SQL**

02

Connexion à la base de données (PDO)

```
<?php
$servername = 'localhost';
$username = 'monLogin';
$password = 'monMDP';

//On essaie de se connecter
try{
    $conn = new PDO("mysql:host=$servername;dbname=aflokkat"
        , $username, $password);
    //On définit le mode d'erreur de PDO sur Exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::
        ERRMODE_EXCEPTION);
    echo 'Connexion réussie';
}

/*On capture les exceptions si une exception est lancée et
on affiche
*les informations relatives à celle-ci*/
catch(PDOException $e){
    echo "Erreur : " . $e->getMessage();
}
```

02

Execution d'une requête (PDO)

- Utilisation de la méthode exec()

```
<?php
$servername = 'localhost';
$username = 'root';
$password = 'root';

try{
    $dbco = new PDO("mysql:host=$servername", $username, $
        password);
    $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::
        ERRMODE_EXCEPTION);

    $sql = "CREATE DATABASE user_aflokkat";
    $dbco->exec($sql);

    echo 'Base de données créée bien créée !';
}

catch(PDOException $e){
    echo "Erreur : " . $e->getMessage();
}
```

02

Execution d'une requête (procédural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// sql to create table
$sql = "CREATE TABLE user_aflokkat (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL
)";

if (mysqli_query($conn, $sql)) {
    echo "Table user_aflokkat a été créée";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}
```

03

Ajout, Modification et Suppression

INSERT INTO nom_de_table (nom_colonne1, nom_colonne2, nom_colonne3, ...) VALUES (valeur1, valeur2, valeur3, ...)

- Les valeurs de type chaîne de caractère (String) doivent être placées entre apostrophes
- La valeur NULL ne doit pas être placée entre apostrophes
- Les valeurs de type numérique ne doivent pas être placées entre apostrophes
- Les colonnes et valeurs des champs **AUTO-INCREMENT** ou **TIMESTAMP** ne sont pas nécessaires
- Récupération de l'id d'AUTO-INCREMENT avec la méthode **lastInsertId()** ou **mysql_insert_id**

03

Ajout de données (PDO)

```
<?php
try {
    $dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

    $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = 'INSERT INTO user (`login`, `password`, `nom`, `prenom`) VALUES (\`test@test.
        com\`, \'$2y$10$SgdPZ7S8G2re0vgI.U.oWe/y5w6rql40ZIlf7GCuqa0CLCucAPM5u%\`, \'
        GRANGER\`, \'Hermione\`)\';

    $dbco->exec($sql);
    echo 'User ' . $dbco->lastInsertId() . ' bien créé !';
} catch (PDOException $e) {
    echo "Erreur : " . $e->getMessage();
}
```

03

Ajout de données (procédural)

```
$requete = "INSERT INTO `user` ( `nom`,`prenom` ) VALUES ('GRANGER','Hermione')";  
mysqli_query($GLOBALS['database'], $requete) or die;
```

- **Lors d'une succession de requêtes, les transactions permettent de s'assurer que TOUTES les requêtes seront traitées**
- **Sinon, l'opération complète sera annulée**
 - **Début de transaction : méthode beginTransaction()**
 - **Fin de transaction : méthode commit()**
 - **Annulation de la transaction : méthode rollBack()**

03

Ajout de données

```
try {
    $dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

    $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $dbco->beginTransaction();

    $sql = 'INSERT INTO user (`login`, `password`, `nom`, `prenom`) VALUES (\''
        . hermione@granger.com\' , \'$2y$10$SgdPZ7S8G2re0vgI.U.oWe/'
        . y5w6rql40ZIlf7GCuqa0CLCucAPM5u%\', \'GRANGER\', \'Hermione\')';

    $sql = 'INSERT INTO user (`login`, `password`, `nom`, `prenom`) VALUES (\''
        . potter.com\' , \'$2y$10$SgdPZ7S8G2re0vgI.U.oWe/y5w6rql40ZIlf7GCuqa0CLCucAPM5u%\''
        . ', \'POTTER\', \'Harry\')';

    $dbco->exec($sql);

    $dbco->commit();

    echo 'Users bien créés !';
} catch (PDOException $e) {
    $dbco->rollBack();
    echo "Erreur : " . $e->getMessage();
}
```

- Chaîne de caractères court-circuitant la requête pour lui donner un autre comportement
- Exemple : lors du formulaire d'inscription, l'utilisateur entre dans son prénom cette chaîne :

```
'\'); DELETE FROM `user`;
```

Injection SQL: solution (PDO)

- **Utilisation de marqueurs nommés** (:prenom) **ou interrogatifs** (?)
- **Avantages** : gain de performance + risque d'injection SQL diminuée
- **Méthode prepare()** : définit un template de la requête avec les marqueurs sans les valeurs
- **Méthode execute()** : tableau qui fait le mapping avec les valeurs, compile et exécute la requête

03

Injection SQL: solution (PDO)

```
try {
    $dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password)
        ;
    $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $dbco->beginTransaction();
    $preparedSql = $dbco->prepare('INSERT INTO `user` (`login`, `password`, `nom`
        , `prenom`) VALUES (:login, :password, :nom, :prenom)');

    $preparedSql->execute(array(
        'login' => 'hermione@granger.com',
        'password' => '$2y$10$SgdPZ7S8G2re0vgI.U.oWe/
            y5w6rql40ZIlf7GCuqa0CLCucAPM5u%',
        'nom' => 'GRANGER',
        'prenom' => 'Hermione'
    ));

    $dbco->commit();

    echo 'User bien créé !';
} catch (PDOException $e) {
    $dbco->rollBack();
    echo "Erreur : " . $e->getMessage();
}
```

03

Injection SQL: solution (PDO) bindValue()

```
$dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

$dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$dbco->beginTransaction();
$login = 'homer@test.com';
$password = '$2y$10$SgdPZ7S8G2re0vgI.U.oWe/y5w6rq140ZIlf7GCuqa0CLCucAPM5u%';

$nom = 'Simpson';
$prenom = 'Homer';
$preparedSql = $dbco->prepare('INSERT INTO `user` (`login`, `password`, `nom`, `
    prenom`) VALUES (:login, :password, :nom, :prenom)');
$preparedSql->bindValue(':login', $login);
$preparedSql->bindValue(':password', $password);
$preparedSql->bindValue(':nom', $nom);
$preparedSql->bindValue(':prenom', $prenom);
$preparedSql->execute();
$preparedSql = $dbco->prepare('INSERT INTO `user` (`login`, `password`, `nom`, `
    prenom`) VALUES (?, ?, ?, ?)');
$preparedSql->bindValue(1, $login);
$preparedSql->bindValue(2, $password);
$preparedSql->bindValue(3, $nom);
$preparedSql->bindValue(4, $prenom);
$preparedSql->execute();
$dbco->commit();
echo 'User bien créé !';
```


Injection SQL: solution (procédural)

- Utilisation de la fonction `mysqli_real_escape_string`

03

Injection SQL: solution (procédural)

```
$requete = "INSERT INTO `user` ( `mail` ) VALUES ('". mysqli_real_escape_string($  
    GLOBALS['database'],$email) ."')";  
  
mysqli_query($GLOBALS['database'], $requete) or die;echo "Erreur : " . $e->getMessage  
    ();
```

03

Modification (PDO)

```
$dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

$dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$dbco->beginTransaction();
$preparedSql = $dbco->prepare('UPDATE `user` SET prenom=:prenom WHERE login=:login');
$preparedSql->bindValue(':login', 'albus@dumbledore.com');
$preparedSql->bindValue(':prenom', 'Albus');
$preparedSql->execute();
$count = $preparedSql->rowCount();
$dbco->commit();
echo $count . ' user(s) modifié(s) !' ;
```

03

Modification (procédural)

```
$requete = "UPDATE `user` SET `prenom`='". mysqli_real_escape_string($GLOBALS['  
    database'],$prenom) ."' WHERE `login`='". $login ."'";  
  
mysqli_query($GLOBALS['database'], $requete) or die;
```

03

Suppression (PDO)

```
$dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

$dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$dbco->beginTransaction();
$preparedSql = $dbco->prepare('DELETE FROM `user` WHERE login=:login');
$preparedSql->bindValue(':login', 'hermione@granger.com');
$preparedSql->execute();
$count = $preparedSql->rowCount();
$dbco->commit();
echo $count . ' user(s) supprimé(s) !';
```

03

Suppression (procédural)

```
$requete = "DELETE `user` WHERE `login`='". $login ."'";  
mysqli_query($GLOBALS['database'], $requete) or die;
```

04

Récupération des données

Récupération des données (PDO)

```
try {  
  
    $dbco = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);  
  
    $dbco->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
    $dbco->beginTransaction();  
    $preparedSql = $dbco->prepare('SELECT `id`, `prenom`, `nom` FROM `user` WHERE  
        login=:login ORDER BY `id` DESC');  
    $preparedSql->bindValue("login", "hermione@granger.com");  
    $preparedSql->execute();  
    $dbco->commit();  
    $rows = $preparedSql->fetchAll(PDO::FETCH_ASSOC); // Renvoie un tableau  
    associatif  
    foreach ($rows as $row) {  
        echo "Id : " . $row["id"] . " " . $row["prenom"] . " " . $row["nom"] . "<br>";  
    }  
} catch (PDOException $e) {  
    $dbco->rollBack();    echo "Erreur : " . $e->getMessage();  
}
```


Récupération des données (procédural)

```
$requete = "SELECT * FROM `admin` WHERE `login` = '" . filter($this->login) . "'";

$result = mysqli_query($GLOBALS['database'], $requete) or die;

if ($data = mysqli_fetch_array($result)) {

    $this->login = $data['login'];
    $this->mail = $data['mail'];
    $this->nom = $data['nom'];

}
```

04

Récupération des données (procédural)

```
$requete = "SELECT * FROM `user` WHERE `id` = '". mysqli_real_escape_string($GLOBALS['  
    database'],$this->id) ."'";  
  
$result = mysqli_query($GLOBALS['database'], $requete) or die;  
  
while ($data = mysqli_fetch_assoc($result)) {  
    $liste_user[] = $data;  
}
```