Introduction to How to Connect Database to PHP

In order to make optimum usage of any database, one should be able to manipulate it as required. The best way to manipulate and manage the databases you created is to connect your database with PHP. First, let us understand what PHP is. PHP is like a control panel that can be used to manage your database. Connecting to PHP gives you the liberty to retrieve data from databases as needed for a particular query. MySQL is the most popular open-source RDBMS, which can be easily connected to PHP. The data in MySQL is arranged in tables and in a row and column structure. It can be easily used for small and large applications and works on the server.

How to Create a Database?

Creating a database in MySQL is as simple as performing a single command in SQL. For beginners, an easy query of "[CREATE DATABASE]" can be used for database creation. Another option to create a database is using another query as "[CREATE SCHEMA]".

For example, suppose you want to create a database named "movies". This can be done by executing the command CREATE DATABASE movies;

```
CREATE DATABASE movies;
```

Adding more parameters and avoiding confusion between existing databases on a single MySQL server, one can use the command "[IF NOT EXISTS]". What this query does is it checks whether any existing database has the same name. If yes, the command will not execute the creation of the database. If there is no such conflict of database names, the below command will be executing and creating a database. Although "[IF NOT EXISTS]" is non-mandatory, it is a good practice.

```
CREATE DATABASEIF NOT EXISTS movies;
```

This newly created database will be empty, waiting for the inclusion of tables with data. SQL command for tables is a simple query "[CREATE TABLE]" with syntax as below.

```
CREATE  TABLE [IF NOT EXISTS] `TableName` (`fieldname` dataType [optional parameters]) ENGIN
E = storage Engine;
```

Connecting Database to PHP

PHP versions below 5 use MySQL extension. But this extension was derogated in 2012.

The 5th version of PHP and newer versions can work with below:

1. MySQLi extension

2. PDO (PHP Data Objects)

Any one of the above can be used according to their own supremacy and one's needs.

The MySQLi extension can be used only with MySQL databases, whereas PDO can be used to connect with 12 different variety of database systems. So, if one is switching databases, PDO might come in handy as it only requires changing a few connections. But in the case of MySQLi, we need to write the entire code and query to switch database. Working with MySQLi requires MySQLi to be enabled on PHP. MySQLi also provides a procedural programming interface along with an object-oriented one. Prepared statements are principal in web security which are allowed in both PDO and MySQLi.

Working with PHP and MySQL

PHP and MySQL are some of the most common stacks for web development. Let's look at a few examples.

1. Object-oriented MySQLi
2. Procedural MySQLi
3. PDO

To start with any of these, it is required to know some important details such as your database system's server address, username, database name, and password. Mainly, we will be using mysqli_connect in all three procedures. This command is used to set up a connection between the database and PHP. First of all, we have to create a separate connection file. This saves time to write code every time you want to insert data and information from the database and insert this data in multiple files.

We can just use the PHP file connection name along with the include function and insert the data instead of rewriting the code each time. This is also useful when you need to transfer your entire project from one system to another. When you change the values in one file, it automatically changes all values in each and every file, as well as saving you the efforts of making changes in every file. Once this is achieved, we have the option of using different procedures to establish a connection with the database.

Let's look into each of them one by one:

1. Using Object-Oriented MySQLi

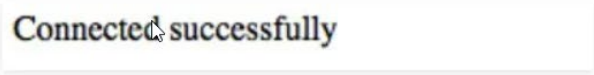
This can be used to build an association with the database using PHP script using the following syntax.

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

// Creating connection
$conn = new mysqli($servername, $username, $password);

// Checking connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Output:



Explanation: Localhost is basically the location of the server. The host can be something else, but in most cases, the server runs on the localhost.

Username is the root and password; it is the same which you are using for php admin.

To establish this link, provide necessary details such as localhost, username, and password. This will create an instance of MySQLi, resulting in a successful connection.

2. Using MySQLi Procedural

The procedure to establish a connection between database and PHP using MySQLi procedural is described as below.

Syntax:

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

// Creating connection
$conn = mysqli_connect($servername, $username, $password);

// Checking connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

Output:

```
Connected successfully
```

Explanation: The main difference for the procedural procedure is that it uses the function mysqli_connect, which fetches the necessary details of the host, username and password, etc. When successfully connected, it will provide with a link identifier.

3. Using PDO

PDO represents PHP Data objects. So in this process of creating a connection, PHP data objects are used as below:

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

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username
    // setting the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

Output:

Connected successfully

Explanation: The exception function in PDO helps to throw in any exceptions that are to be considered and manages any issues that may occur while establishing connections.

All the above methods help access and manage the database you need.

Checking the Connection

Below syntax can be used to check whether your connection is successfully connected.

```php
<?php

include 'db_connection.php';



echo "Connected Successfully";

mysqli_close($conn);

?>
```

Db_connection is the php file name.

Ending the Connection

Once you establish a connection of the database using PHP scripts, you should also close the connection once your work is finished. With an assumption of reference to the connection is stored in the $ conn variable, below are the closing syntax which can be used in the above-given procedures.

- Using MySQLi object-oriented procedure

```
$conn->close();
```

- Using MySQLi procedural procedure

```
mysqli_close($conn);
```

- Using PDO procedure

```
$conn = null;
```

Conclusion:

Accessing and managing your database is quite easily performed when connected using PHP. It also offers a variety of ways to establish this connection to suit varying levels of requirements. As the connection is established, one can perform queries to extract data from the tables. This information

can be printed out easily. Closing the connection once your work is completed is also important and a part of connecting the database to PHP.

Traduction en français

Introduction à la connexion de la base de données à PHP

Afin d'optimiser l'utilisation de toute base de données, il faut pouvoir la manipuler au besoin. La meilleure façon de manipuler et de gérer les bases de données que vous avez créées est de connecter votre base de données avec PHP. Tout d'abord, laissez-nous comprendre ce qu'est PHP. PHP est comme un panneau de contrôle qui peut être utilisé pour gérer votre base de données. Se connecter à PHP vous donne la liberté de récupérer les données des bases de données nécessaires pour une requête particulière. MySQL est le RDBMS open source le plus populaire, qui peut être facilement connecté à PHP. Les données de MySQL sont disposées en tables et en lignes et en colonnes. Il peut être facilement utilisé pour les petites et grandes applications et fonctionne sur le serveur.

Comment créer une base de données ?

Créer une base de données dans MySQL est aussi simple que d'exécuter une seule commande dans SQL. Pour les débutants, une requête facile de « [CREATE DATABASE] » peut être utilisée pour la création de la base de données. Une autre option pour créer une base de données est d'utiliser une autre requête comme « [CREATE SCHEMA] ».

Par exemple, supposons que vous vouliez créer une base de données appelée « films ». Vous pouvez le faire en exécutant la commande CREATE DATABASE movies;

```
CREATE DATABASE movies;
```

En ajoutant plus de paramètres et en évitant la confusion entre les bases de données existantes sur un seul serveur MySQL, on peut utiliser la commande « [IF NOT EXISTS] ». Cette requête vérifie si une base de données existante a le même nom. Si oui, la commande n'exécutera pas la création de la base de données. S'il n'y a pas un tel conflit de noms de base de données, la commande ci-dessous s'exécute et crée une base de données. Bien que « [IF NOT EXISTS] » ne soit pas obligatoire, c'est une bonne pratique.

```
CREATE DATABASEIF NOT EXISTS movies;
```

Cette base de données nouvellement créée sera vide, en attendant l'inclusion de tableaux avec des données. La commande SQL pour les tables est une simple requête "[CREATE TABLE]" avec la syntaxe ci-dessous.

```
CREATE  TABLE [IF NOT EXISTS] `TableName` (`fieldname` dataType [optional parameters]) ENGIN
E = storage Engine;
```

Connexion de la base de données à PHP

Les versions de PHP ci-dessous 5 utilisent l'extension MySQL. Mais cette extension a été annulée en 2012.

La 5ème version de PHP et les versions plus récentes peuvent fonctionner avec :

1. Extension MySQLi

2. AOP (objets de données PHP)

Chacun de ce qui précède peut-être utilisé selon sa propre suprématie et ses besoins.

L'extension MySQLi ne peut être utilisée qu'avec les bases de données MySQL, tandis que PDO peut être utilisé pour se connecter à 12 systèmes de bases de données différents. Donc, si l'on change de base de données, PDO pourrait être utile car il ne nécessite de changer que quelques connexions. Mais dans le cas de MySQLi, nous devons écrire tout le code et la requête pour changer de base de données. Travailler avec MySQLi nécessite que MySQLi soit activé sur PHP. MySQLi fournit également une interface de programmation procédurale avec une interface orientée objet. Les déclarations préparées sont principales dans la sécurité Web qui sont autorisées dans PDO et MySQLi.

Travailler avec PHP et MySQL

PHP et MySQL sont parmi les piles les plus courantes pour le développement web. Regardons quelques exemples.

1. MySQLi orientée objet

2. Procédure MySQLi

3. OAP

Pour commencer, il est nécessaire de connaître certains détails importants tels que l'adresse du serveur, le nom d'utilisateur, le nom de la base de données et le mot de passe de votre système de base de données. Principalement, nous utiliserons mysqli_connect dans les trois procédures. Cette commande est utilisée pour configurer une connexion entre la base de données et PHP. Tout d'abord, nous devons créer un fichier de connexion séparé. Cela permet d'économiser du temps pour écrire du code chaque fois que vous voulez insérer des données et des informations de la base de données et insérer ces données dans plusieurs fichiers.

Nous pouvons simplement utiliser le nom de connexion du fichier PHP avec la fonction include et insérer les données au lieu de réécrire le code à chaque fois. Ceci est également utile lorsque vous avez besoin de transférer l'ensemble de votre projet d'un système à un autre. Lorsque vous modifiez les valeurs d'un fichier, il modifie automatiquement toutes les valeurs de chaque fichier, ainsi que vous sauvez les efforts de faire des changements dans chaque fichier. Une fois cela réalisé, nous avons la possibilité d'utiliser différentes procédures pour établir une connexion avec la base de données.

Examinons chacun d'eux un par un :

1. Utilisation de MySQLi orientée objet

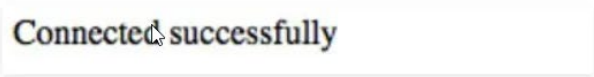
Ceci peut être utilisé pour construire une association avec la base de données en utilisant le script PHP en utilisant la syntaxe suivante.

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

// Creating connection
$conn = new mysqli($servername, $username, $password);

// Checking connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Output:



Explication : Localhost est essentiellement l'emplacement du serveur. L'hôte peut être autre chose, mais dans la plupart des cas, le serveur fonctionne sur l'hôte local.

Le nom d'utilisateur est la racine et le mot de passe; c'est le même que vous utilisez pour phpMyadmin.

Pour établir ce lien, fournir les détails nécessaires tels que localhost, nom d'utilisateur et mot de passe. Cela créera une instance de MySQLi, résultant en une connexion réussie.

2. Utilisation de la procédure MySQLi

La procédure pour établir une connexion entre la base de données et PHP en utilisant la procédure MySQLi est décrite ci-dessous.

Syntax:

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

// Creating connection
$conn = mysqli_connect($servername, $username, $password);

// Checking connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

Output:

```
Connected successfully
```

Explication : La principale différence pour la procédure est qu'elle utilise la fonction mysqli_connect, qui récupère les détails nécessaires de l'hôte, du nom d'utilisateur et du mot de passe, etc. Une fois connecté avec succès, il fournit un identifiant de lien.

3. Utilisation du PDO

PDO représente les objets de données PHP. Ainsi, dans ce processus de création d'une connexion, les objets de données PHP sont utilisés comme ci-dessous :

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

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username
    // setting the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

Output:

Connected successfully

Explication : La fonction d'exception dans PDO aide à ajouter les exceptions qui doivent être prises en compte et gère les problèmes qui peuvent survenir lors de l'établissement de connexions.

Toutes les méthodes ci-dessus permettent d'accéder et de gérer la base de données dont vous avez besoin.

Vérification de la connexion

La syntaxe ci-dessous peut être utilisée pour vérifier si votre connexion est correctement connectée.

```php
<?php

include 'db_connection.php';



echo "Connected Successfully";

mysqli_close($conn);

?>
```

Db_connection est le nom du fichier php.

Mettre fin à la connexion

Une fois que vous établissez une connexion de la base de données en utilisant des scripts PHP, vous devez également fermer la connexion une fois que votre travail est terminé. Avec une hypothèse de référence à la connexion est stocké dans la variable $ conn, ci-dessous se trouve la syntaxe de fermeture qui peut être utilisée dans les procédures ci-dessus.

- Utilisation de la procédure orientée objet MySQLi

```
$conn->close();
```

- Procédure d'utilisation de MySQLi

```
mysqli_close($conn);
```

- Utilisation de la procédure AOP

```
$conn = null;
```

Conclusion :

L'accès et la gestion de votre base de données se fait assez facilement lorsqu'elle est connectée avec PHP. Il offre également une variété de façons d'établir ce lien pour répondre à divers niveaux d'exigences. Comme la connexion est établie, on peut effectuer des requêtes pour extraire des données des tables. Cette information peut être imprimée facilement. Fermer la connexion une fois votre travail terminé est également important et une partie de la connexion de la base de données à PHP.