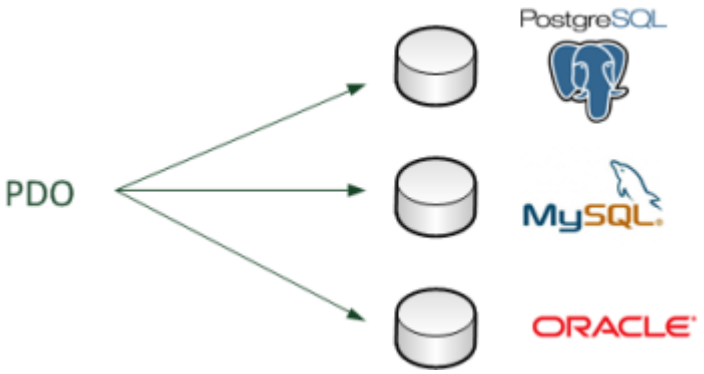
BDD - MySQL et PHP

[Mise à jour le 28/8/2024]

Sources

- Documentation de référence sur php.net
- Tutoriel sur le site **Openclassrooms** : Concevez votre site web avec PHP et MySQL

Mots-clés : SGBD, base, table, enregistrement, champ, requêtes, SQL.



1. Introduction

En PHP, on se connecte à une base de données avec les méthodes suivantes:

1. Avec l'extension mysqli_ (modèle procédural lié à MySQL)
2. Avec l'extension **PDO** (modèle objet) : à privilégier, car plus moderne et permet l'accès à divers types de BDD (**SQLite**, **MySQL**, etc.) avec la même syntaxe.

PDO (**PHP D**ata **O**Bjects) comprend trois classes :

- *PDO* pour créer des objets de connexion à la base disposant de méthodes pour l'envoi de requêtes, etc.
- *PDOStatement* permettant de gérer des requêtes préparées et des résultats de requête.
- *PDOException* qui permet de gérer et d'afficher des informations sur les erreurs.

Extrait de la table *jeux_video* de la base *test* utilisée dans les exemples

ID	nom	possesseur	console	prix	nbre_joueurs_max	commentaires
1	Super Mario Bros	Florent	NES	4	1	Un jeu d'anthologie !
2	Sonic	Patrick	Megadrive	2	1	Pour moi, le meilleur jeu du monde !
3	Zelda : ocarina of time	Florent	Nintendo 64	15	1	Un jeu grand, beau et complet comme on en voit rar...
4	Mario Kart 64	Florent	Nintendo 64	25	4	Un excellent jeu de kart !
5	Super Smash Bros Melee	Michel	GameCube	55	4	Un jeu de baston délirant !

jeux_video

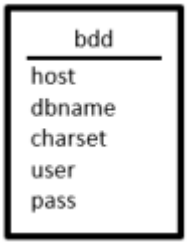
On utilise le SGBD MySQL pour les exemples de la base *test*. La table jeux_video peut être téléchargé ici.

2. Accès à une base de données MySQL avec PHP

L'exploitation d'une base de données se fait à travers les actions suivantes :

1. **Connexion** à la base
2. **Envoi** de requêtes au serveur
3. **Traitement** du résultat des requêtes
4. **Fermeture** de la connexion

2.1 Connexion à la base



Connexion

La connexion à la base se fait **une seule fois** en créant un **objet** de la classe PDO.

- *Syntaxe*
  nomConnect = **new PDO**('mysql:host=**$host** dbname=**$base**',user,pass);

Objet PDO

L'objet *$<nomConnect>* représente la connexion au serveur. Il est utilisé pour toutes les opérations à effectuer sur la base.

- *Exemple 1* : Connexion à la base *test* avec le login *root*, sans mot de passe et sans la gestion des erreurs

*.php

```
$bdd = new PDO('mysql:host=localhost;dbname=test;charset=utf8', 'root',
```

```
'');
```

- *Exemple 2* : Même connexion avec la gestion des d'erreurs

*.php

```
try
{
    $bdd = new PDO('mysql:host=localhost;dbname=test;charset=utf8',
'root', '');
}
catch (Exception $e)
{
        die('Erreur : ' . $e->getMessage()); // En cas d'erreur, on
affiche un message et on arrête tout
}
```

2.2 Envoi d'une requête

> **Variable**
> L'information liée au bon fonctionnement de la requête ou son résultat sont placés dans une variable pour être traités.

Cas 1 - Si la requête **ne retourne pas de résultat** (INSERT, UPDATE, DELETE, etc.), on utilise la méthode *exec()*.

- *Syntaxe*
  integer nomConnect → exec(*string* requete)
  nomvar = nomConnect → exec(*string* requete)

> **Requête**
> **exec()** retourne un entier contenant le nombre de lignes concernées par la requête.

- *Exemple* : modification du possesseur de la console NES identifiée par ID=1 (Initialement Florent)

*.php

```
$nb=$bdd->exec('UPDATE jeux_video SET possesseur="Laurent" WHERE
ID="1"');
```

- *Résultat*

ID	nom	possesseur	console	prix	nbre_joueurs_max	commentaires
1	Super Mario Bros	Laurent	NES	4	1	Un jeu d'anthologie !

Cas 2 - Si la requête **retourne un résultat** (SELECT, etc.), on utilise la méthode *query()*.

- *Syntaxe*
  object nomConnect → query(*string* requete)
  nomResultat = nomConnect → query(*string* requete)

> **Erreur**
> Retourne FALSE en cas d'erreur ou un objet représentant l'ensemble des lignes de résultat.

- *Exemple* : *reponse* reçoit le résultat de la requête

*.php

```
$reponse = $bdd->query('SELECT * FROM jeux_video WHERE console="NES"');
```

- *Résultats* issus du traitement décrit au paragraphe suivant
  - Super Mario Bros - Laurent - 4
  - The Rocketeer - Michel - 2
  - Ice Hockey - Michel - 7

2.3 Traitement du résultat de la requête

- **Insertion, suppression, mise à jour**

> **Vérification**
> Pour les opérations d'insertion, de suppression ou de mise à jour des données dans une base, il est utile de vérifier si la requête a bien été exécutée.

*.php

```
echo "<p> $nb ligne(s) modifiée(s) </p>"; // Résultat : 1 ligne(s) modifiée(s)
```

$nb contient le nombre de lignes modifiées dans l'exemple de cas 1 précédent. Cette valeur pourra être testée pour valider la requête.

- **Résultat d'une commande SELECT**

Pour les opérations d'insertion, de suppression ou de mise à jour des données dans une base, il est utile de vérifier si la requête a bien été exécutée.

> **SELECT**
> Lorsqu'il s'agit de lire le résultat d'une requête contenant la commande *SELECT*, la méthode *query()* retourne un objet de type *PDOStatement* ($reponse dans les exemples). La classe *PDOStatement* dispose de méthodes permettant de récupérer des données. La méthode des objets *PDOStatement* couramment utilisée pour lire des données est ***fetch()***.

- *Syntaxe*
  array nomResultat → fetch(*integer* type);

  Cette méthode retourne un tableau :
  - pouvant être indicé si type = PDO::FETCH_NUM
  - dont les clés sont les noms des colonnes de la table interrogée si type = PDO::FETCH_ASSOC
  - dont les clés sont mixtes si type = PDO::FETCH_BOTH

Pour lire toutes les lignes du résultat, il faut créer une boucle while qui lit chaque ligne.

- *Exemple*

*.php

```
while ($donnees = $reponse->fetch()) {
    echo '<p>' . $donnees['nom'].' - '.$donnees['possesseur'].' -
'.$donnees['prix'].'</p>';
}
```

- *Résultats*
  - Super Mario Bros - Laurent - 4
  - The Rocketeer - Michel - 2
  - Ice Hockey - Michel - 7

2.4 Fermeture de la connexion

> **NULL**
> Pour clore la connexion, il suffit de détruire l'objet en assignant **NULL** à la variable qui le gère. Si ce n'est pas fait explicitement, PHP fermera automatiquement la connexion lorsque le script arrivera à la fin.

- *Exemple*

*.php

```
$bdd=null;
```

Résumé

- Pour dialoguer avec MySQL depuis PHP, on fait appel à l'extension PDO de PHP.
- Avant de dialoguer avec MySQL, il faut s'y connecter. On a besoin de l'adresse IP de la machine où se trouve MySQL, du nom de la base de données ainsi que d'un login et d'un mot de passe.
- Il faut faire une boucle en PHP pour récupérer ligne par ligne les données renvoyées par MySQL.