PHP-PDO : Interfaçage PHP et SGBD au travers d'une interface d'abstraction d'accès aux données.

Apache, PHP et serveur de base de données

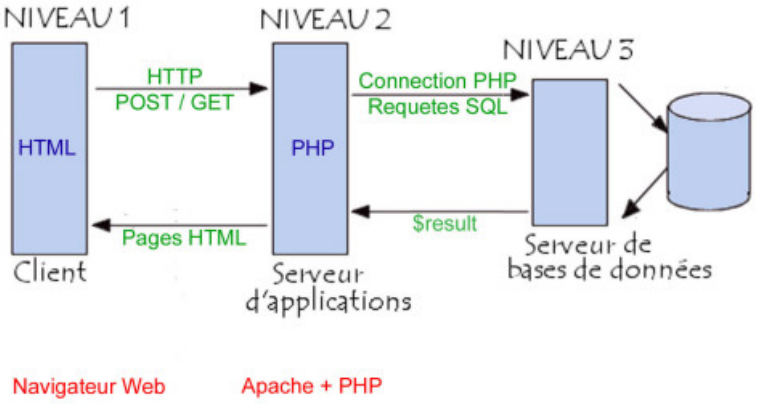

FIGURE 1 – Exemple d'architecture 3-tiers : PHP/BD (commentcamarche.net - © 2003 Pillou - GNU FDL)

L'extension PHP-PDO

L'extension PHP Data Objects (PDO) fournit une interface d'abstraction à l'accès de données, ce qui signifie que vous utilisez les mêmes fonctions pour exécuter des requêtes ou récupérer les données quel que soit la base de données utilisée. PDO ne fournit pas une abstraction de base de données : il ne réécrit pas le SQL, n'émule pas des fonctionnalités manquantes.

PDO définit une excellente interface pour accéder à une base de données depuis PHP. Chaque pilote de base de données implémenté dans l'interface PDO peut utiliser des fonctionnalités spécifiques de chacune des bases de données en utilisant des extensions de fonctions. Notez que vous ne pouvez exécuter aucune fonction de base de données en utilisant l'extension PDO par elle-même ; vous devez utiliser un driver PDO spécifique à la base de données pour accéder au serveur de base de données.

Extrait de l'introduction à PDO dans le manuel PHP

Comment se connecter et envoyer des requêtes

Connexion à la base de données

Pour établir une connexion avec le SGBD, il faut instancier la classe PDO en fournissant au constructeur les différents paramètres :

- nom du serveur : `$server` (`servbdd` pour l'IUT)
- nom du connecteur : `$driver` (`pgsql`)
- nom de la base : `$dbname` (`<pg_login>`)
- nom de l'utilisateur : `$user` (`<login>`, celui de votre session PostgreSQL)
- mot de passe : `$pass` (votre mot de passe de session)

Le code suivant permet cette connexion, en supposant que tous les paramètres que nous venons d'évoquer sont définis dans le script PHP qui se trouve dans `connection_params.php`.

```
include('connection_params.php');
$dbh = new PDO("$driver:host=$server;dbname=$dbname", $user, $pass);
```

Lire des données

Pour récupérer des données, après la connexion, il vous faut en géréral :

1. préparer une requête SQL
2. exécuter cette requette
3. récupérer les résultats

Il arrive que certaines de ces étapes soient confondues.

Voici un exemple de code pour lire le contenu d'une table ligne par ligne :

```
$dbh = new PDO("$driver:host=$server;dbname=$dbname",
            $user, $pass);

foreach($dbh->query("SELECT * from forum1._user") as $row) {
    echo "<pre>"; // pour la version navigateur (présentation brute)
    print_r($row);
    echo "</pre>";
}

$dbh = null; // on "ferme" la connexion
```

Remarquez le contenu du tableau affiché par la commande `print_r`. Il possède à la fois des index numériques et des index associatifs. Si vous souhaitez récupérer uniquement un tableau associatif, il faut l'indiquer avant de lancer la requête par :

```
$dbh->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
```

Voici un autre exemple de code qui récupère les données dans un seul tableau à deux dimensions : des lignes indexées numériquement pour chaque n-uplet et un tableau associatif dans chaque ligne du tableau.

```
$dbh = new PDO("$driver:host=$server;dbname=$dbname", $user, $pass);
$dbh->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,  PDO::FETCH_ASSOC);

$stmt = $dbh->prepare("SELECT * from forum1._user");
$stmt->execute();
$result = $stmt->fetchAll();

echo "<pre>";
print_r($result);
echo "</pre>";
$dbh = null;
```

Modifier des données (update, insert, delete)

Pour exécuter des requêtes de modification de données, il suffit de préparer une requête et de l'exécuter car aucun envoi de données en retour n'est fait. Il est toutefois utile d'inspecter le retour de la méthode exécute pour savoir si tout s'est bien passé ou d'utiliser la capture d'exceptions (voir dans la section suivante sur la gestion des erreurs).

Voici un code permettant d'insérer un n-uplet :

```
$dbh = new PDO("$driver:host=$server;dbname=$dbname",
            $user, $pass);
$nickname   = 'Arnaud';
$email      = "arnaud@dataland.org";
$pass       = "duanra";

$stmt = $dbh->prepare(
        "INSERT INTO forum1._user(nickname, email, pass) VALUES('$nickname', '$email','$pass')"
    );
$stmt->execute();
$dbh = null;
```

Attention, cette forme d'envoi de requêtes ne vous met pas à l'abri des injections SQL dans les formulaires. Pour palier ce problème, nous vous conseillons d'utiliser des requêtes préparées comme décrit cette partie de la documentation PHP.

Gestion des erreurs et exceptions venant du SGBD

Nous avons évoqué le fait de récupérer les retours des méthodes de PDO pour décider si tout s'était bien passé lors de la connexion, ou de l'exécution de requêtes.

Capturer les erreurs

Un moyen plus efficace, parce qu'il vous permet de récupérer des codes d'erreurs et des messages correspondants entre autres, est d'utiliser la capture d'exception `PDOException`dans PHP. Notre premier script de lecture des données deviendrait :

```php
try {
    $dbh = new PDO("$driver:host=$server;dbname=$dbname",
            $user, $pass);
    $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $dbh->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
    foreach($dbh->query("SELECT * from forum1._user")
                as $row) {
        echo "<pre>";
        print_r($row);
        echo "</pre>";
    }
    $dbh = null;
} catch (PDOException $e) {
    print "Erreur !: " . $e->getMessage() . "<br/>";
    die();
}
```

Les erreurs de connexion, ainsi que celles concernant la requête (table manquante par exemple) seront capturées ici et gérées dans la section `catch`. Nous faisons simple pour la gestion des erreurs ici : affichage du message et assassinat du script (`die()`).

Remarquez que nous avons élevé le niveau de retour d'erreurs en changeant la valeur de l'attribut PDO `PDO::ATTR_ERRMODE` pour avoir des retours plus complets.

Gérer les retours d'erreur

PostgreSQL associe un certain nombre de codes et de messages d'erreurs en fonction de la situation. Par exemple, une contrainte NOT NULL non-respectée enverra un code et un message différents de la violation d'une contrainte de clé primaire. Vous pourrez trouver la liste des codes et messages dans l'Annexe A. Codes d'erreurs de PostgreSQL de la documentation officielle.

Depuis une procédure PL/pgsql, pour envoyer un message et un code d'erreur personnalisé, nous pouvons utiliser la commande :

```
raise 'message erreur' using errcode = '23000';
```

Nous pouvons ensuite, à l'aide des méthodes `getCode()` et `getMessage()` de l'objet `PDOException` récupérer les codes et messages d'erreur respectivement, et décider quoi faire dans la section `catch` : faire une redirection vers une page plus sympathique ppour l'utilisateur de l'application web, plutôt que de laisser PHP afficher un message d'erreur incompréhensible et arrêter le scirpt.

Exercices de mise en application

Vous allez vous appuyer sur le schéma du TP de S2 soncernant l'entreprise Distribill, ses départements et ses employés. Les script de création de la base, nommé `cr_schema_distribill_s3.sql` est fourni avec cet énoncé sur l'ENT. Ce script crée un schéma `distribill_s3`.

1. Concevez un script PHP permettant de récupérer les données de la table `_dept` du schéma `distribill_s3` et de stocker ces données dans un tableau associatif à deux dimensions.
2. Concevez un script PHP permettant d'insérer un nouveau département dans le schéma `distribill_s3`. Observez le résultat avec SQLWorkbench/J, ou avec le script précédent.
3. Concevez un script PHP qui affichera l'ensemble des employés,
4. Concevez un script PHP permettant de mettre à jour le nom d'un employé en indiquant son numéro de matricule dans le schéma `distribill_s3` et gérez le retour d'erreur si l'employé n'existe pas.
5. Concevez un script PHP permettant de supprimer un employé en bas de la hiérarchie en indiquant son numéro de matricule dans le schéma `distribill_s3`.
6. Prolongement : concevez un script permettant de créer un nouvel emloyé à partir des toutes informations nécessaires fournies dans un tableau associatif à une dimension. Vous pourrez refaire cet exercice après avoir étudié les transactions à l'aide de la documentation PDO). Vous pourrez faire de même pour la suppression d'un employé.