Programmation Web Dynamique 2

12. Interactions avec MySQL dans PHP

Interaction avec MySQL

- Pour interagir avec MySQL, il existe plusieurs façons de faire. La façon de base est d'utiliser la librairie libmysqli.dll (si on travaille avec MySQL 4.1 ou plus récent).

Connection à MySQL

- Pour interagir avec MySQL, il faut commencer par ouvrir une connexion vers la base que l'on veut interroger. Ceci est fait à l'aide de la fonction mysqli_connect(), qui prend en paramètres le nom de l'hôte, le nom d'usager et le mot de passe de cet usager. La fonction mysqli_connect() retourne un objet représentant la connexion qui permettra de spécifier aux autres opérations une connexion en particulier si plusieurs d'entre elles sont ouvertes en parallèle.
  - Ex : $linkID = mysqli_connect(“localhost”, “gharvey”, “monpassword”);

- Comme avec les fichiers, à la fin de l'interaction avec la base, il faut fermer la connection à la base de données avec la fonction mysqli_close(), qui prend en paramètre l‘objet retourné par mysqli_connect(). Si aucun objet n'est envoyé en paramètres, mysqli_close() ferme la dernière connexion ouverte. La fonction mysqli_close() retourne TRUE si elle a fonctionné, FALSE sur erreur.
  - Ex : mysqli_close($linkID);

Sélection d'une base MySQL

- Après l'ouverture d'une connexion au serveur de base de données, il faut sélectionner à quelle base on désire se connecter. Pour se faire, on utilise la fonction mysqli_select_db(), qui prend en paramètres l‘objet de connexion retourné par mysqli_connect() et le nom de la base de données que l’on veut accéder. Elle retourne TRUE si tout a bien fonctionné.
  - Ex : $selected = mysqli_select_db($linkID, “mabase”);

Requêtes MySQL dans PHP

- Lorsque la base de données est sélectionnée, il est possible d'utiliser mysqli_query(). Celle-ci prend en paramètres la requête SQL que l'on veut exécuter (sans le point-virgule à la fin) et l'objet de connexion retourné par mysqli_connect(). Cette fonction retourne un objet mysqli_result pour la requête SQL envoyée en paramètres.
  - Ex : $resultat = mysqli_query($linkID, "SELECT * FROM usagers WHERE nom = '$nomUsager'");

- L'objet retourné pourra ensuite être utilisé, dans le cas d'un SELECT, par exemple, pour accéder aux résultats retournés par la requête. On y accède via la fonction mysqli_fetch_row(), qui prend en paramètre l'objet de résultat retourné par mysqli_query() et qui retourne la prochaine rangée du résultat de la requête.
  - Ex : while($rangeeData = mysqli_fetch_row($resultat))

```
{
        echo "Premiere colonne : $rangeeData[0]";
        echo "Deuxieme colonne : $rangeeData[1]";
}
```

Requêtes MySQL (suite)

- On peut aussi utiliser mysqli_fetch_assoc(), qui retourne un tableau dont les éléments sont accessibles par index via le nom de la colonne dans la table interrogée.
  - Ex : while($rangeeData = mysqli_fetch_assoc ($resultat))

```
{
        echo “Nom de l'Usager :” . $rangeeData[“nomUsager”];
        echo “Courriel :”. $rangeeData[“courriel”];
}
```

Requêtes SQL (fin)

- Finalement, il est possible d'obtenir le nombre de résultats retournés par une requête en utilisant mysqli_num_rows() qui prend en paramètres l'identifiant de résultat retourné par mysqli_query().
  - Ex : $numRows = mysqli_num_rows($resultat);

- Si notre requête ne retourne pas de résultat, mais affecte des rangées dans la base de données (comme un DELETE, un UPDATE ou un INSERT), il faut utiliser mysqli_affected_rows() pour savoir combien de rangées ont été affectées.
  - $rangeesModifiees = mysqli_affected_rows($connexion);
  - Exception : celle-ci retourne 0 dans le cas d'un DELETE sans WHERE qui supprime tous les éléments d'une table.