ACCES AUX BASES DE DONNEES EN PHP

1- Se connecter à une base de données de type MySQL/MariaDB :
API : *Application Programming Interface*
L'API est le module bibliothèque ajouté au langage pour s'interfacer avec le monde extérieur.
Dans notre cas, le monde extérieur est un serveur de base de données.

L'API pour PHP et SQL s'appelle PDO (*Php Data Object*). C'est une bibliothèque orientée objet.

On doit créer un OBJET PDO. Cet objet gère les événements ce qui rend possible son utilisation au travers du système try/catch pour afficher les éventuelles erreurs :

```
$serveur = '10.69.88.1';
$base =    'tpPhpSession';
$login =   'tpPhpSession';
$mdp =     'tpPhpSession';

try {
     $bdd = new PDO("mysql:host=$serveur;dbname=$base", $login, $mdp);
}
catch (Exception $e) {
     echo "ERR Exception : ". $e->getMessage();
     die();
}
```

➔ Une fois la connexion établie, la variable $**bdd** est utilisée comme point d'accès à la base de données.

2-Formuler une requête :
On dispose de 2 méthodes pour les requêtes :

a) Requêtes instantanées, avec la méthode PDO **query**
b) Requêtes préparées, avec les méthodes PDO **prepare, bindparam, execute**

a- **Requête instantanée** :

```
$req = "INSERT INTO utilisateurs VALUES('Dupont', 'Bob',
'bdupont','yep')" ;

if ( ! $bdd->query($req) )
{
echo "Echec"; die() ;
}
```

b- **Requête préparée** :

On utilise cette technique quand une requête doit être répétée plusieurs fois, avec des paramètres différents. Les paramètres sont transmis au moment de la requête.
Une requête préparée est également plus sécurisée puisqu'elle est protégée contre les attaques par injection de code.

Là encore, 2 façons de faire :
Exemple avec les **paramètres nommés** :

```
$sql = $bdd->prepare("INSERT INTO utilisateurs VALUES( :nom, :prenom, :login, :mdp)");
$sql->bindParam (':nom',$nom);
$sql->bindParam (':prenom',$pre);
$sql->bindParam (':login', $utilisateur);
$sql->bindParam (':mdp', $mot_de_passe);

$nom = 'Dupont';
$pre = 'Bob';
$utilisateur = 'bdupont';
$mot_de_passe = 'yep';

$sql->execute();
```

Autre possibilité : avec **tableau de données**

```
$sql = $bdd->prepare("INSERT INTO utilisateurs VALUES( ?, ?, ?, ?)";

$nom = 'Dupont';
$pre = 'Bob';
$utilisateur = 'bdupont';
$mot_de_passe = 'yep';

$sql->execute( array($nom, $pre, $utilisateur, $mot_de_passe) );
```

➔ Pour chaque requête, il suffit de réutiliser $sql->execute() avec les nouvelles valeurs des paramètres.

3-Analyser la réponse : le FETCH.

On distingue 2 types de requêtes SQL :

a) Requête avec réponse en tableau (SELECT, DESCRIBE, …)
b) Requête avec réponse booléenne ok/ko (INSERT, DELETE, UPDATE, …)

Pour une requête à réponse booléenne, il est prudent de vérifier si la requête a fonctionné ou non, avec un simple test « if ».

Pour une requête avec réponse en tableau (SELECT, DESCRIBE, …), il y a des choix à faire pour l'analyse du résultat. Là encore, il y a plusieurs façons de faire. En voici 2 principales :

- Si on utilise la requête instantanée **query**, voici une technique :

```
$req = 'SELECT * FROM utilisateurs';

foreach ( $bdd->query($req) as $ligne)
{
      echo 'nom:' . $ligne['nom'] . 'Prénom:' . $ligne['prenom'];
      echo '</br>';
}
```

- Si on utilise la requête **préparée**, voici une technique équivalente :

```
$sql = $bdd->prepare('SELECT * FROM utilisateurs');
$sql->execute();

while ($ligne  = $sql->fetch(PDO::FETCH_ASSOC))
{
      echo 'nom:' . $ligne['nom'] . 'Prénom:' . $ligne['prenom'];
      echo '</br>';
}
```

PDO::FETCH_ASSOC : indique le <u>type de tableau</u> que l'on veut récupérer. Ici, ce sera un tableau associatif (c-à-d le nom des champs sera le nom des colonnes)
PDO::FETCH_NUM : ce sera un tableau à index numérique (colonnes numérotées de 0 à …)
PDO::FETCH_OBJ : ce sera un objet tableau (array_object) qui utilisera les nom des champ comme propriétés de l'objet.

- Voici le même principe mais cette fois on récupère <u>TOUT le tableau en 1 fois</u> et on le lit APRES :

```
$sql = $bdd->prepare('SELECT * FROM utilisateurs');
$sql->execute();

$resul = $sql->fetchAll(PDO::FETCH_ASSOC);

foreach ($resul as $ligne)
{
      echo 'nom:' . $ligne['nom'] . 'Prénom:' . $ligne['prenom'];
      echo '</br>';
}
```

NOTES :

- Le choix WHILE ou FOREACH dépend de la façon de demander le résultat :
  - FOREACH : Pour un résultat complet (FETCHALL, QUERY)
  - WHILE : Pour un résultat ligne à ligne (FETCH)

- Dans un tableau complet (FETCHALL), on peut aussi récupérer directement 1 valeur si on connait sa position dans le tableau. Pas besoin de faire une boucle dans ce cas :
  Exemple : récupérer l'information de la 3e ligne du tableau (index [2]) :

```
$sql = $bdd->prepare('SELECT * FROM utilisateurs');
$sql->execute();

$resul = $sql->fetchAll(PDO::FETCH_ASSOC);

echo 'nom:' . $resul[2]['nom'] . 'Prénom:' . $resul[2]['prenom'];
```