Introduction à PHP

PHP et SGBD : un exemple avec Mysql (MariaDB)

monnerat@u-pec.fr

3 avril 2026

IUT de Fontainebleau

Sommaire

Introduction

Mysql (MariaDB)

API mysqli

Requêtes préparées

Introduction

Problématique

Sites dynamiques $\rightsquigarrow$ Informations dynamiques

Comment la gérer ?

C(reate), R(ead), U(pdate), D(elete)

Plusieurs solutions :

- Fichiers : difficultés à accéder/organiser l'information logiquement. Suffisant par exemple pour un compteur de visite !
- Utilisation d'un SGBD : il en existe beaucoup !

Interfaçage de l'application WEB (scripts php) avec le(s) sgbd(s)

2 solutions

$\rightsquigarrow$ Utilisation d'une api spécifique au sgbd
api MySQL, api Oracle, api Access, api PostGreSQL, etc.

$\rightsquigarrow$ Utilisation d'une api indépendante du sgbd
Une seule interface : c'est le cas avec ODBC (Open DataBase Connectivity), ou encore PDO (PHP Data Objects).

Mysql (MariaDB)

MariaDB

`MariaDB` est une base de données (disponible sur toutes les plates-formes) implémentant le langage de requête SQL un langage relationnel très connu. Cette partie suppose connue les principes des bases de données relationnelles.

Quelques liens utiles :

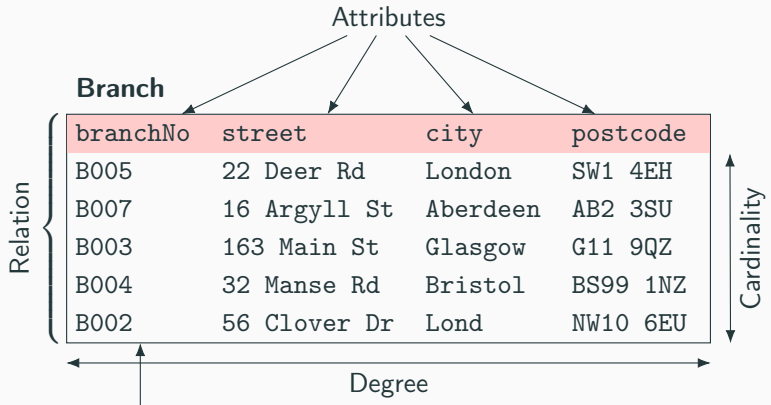
- phpmyadmin : outil libre et gratuit développé par la communauté des programmeurs libres permettant l'administration aisée des bases de données MariaDB/MySQL avec php. Il est disponible sur : `http://www.phpmyadmin.net`.
- Le site `https://mariadb.com/`
- La documentation de `https://mariadb.com/` est disponibles à `https://mariadb.com/kb/fr/documentation-de-mariadb/`.

Mysql (MariaDB)

Modèle relationnel

Modèle relationnel

- Modèle le plus répandu et le plus classique.
- Les données sont organisées en tables, chacune des colonnes représentant un attribut des données.



Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QZ
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	Lond	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

- Chaque attribut (colonne) est typé.
- SQL (Structured Query Language) : langage standard de requête et de mise à jour des données (petites variantes suivant les SGBD)

SGBD relationnels les plus connus

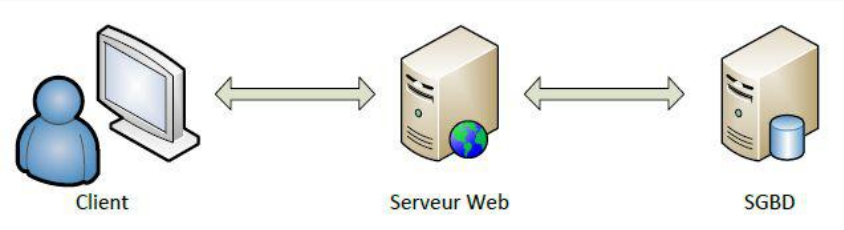
sgbds	
Oracle	commercial, le plus utilisé en entreprise
IBM DB2	commercial
Microsoft SQL Server	commercial
Microsoft Access	commercial, pauvre en fonctionnalités
MySQL	libre (?), le plus utilisé sur le Web
MariaDB	libre (fork de MySQL)
PostGreSQL	libre, plus abouti (?) que MySQL
Sqlite	libre, pas besoin de serveur


Je vous renvoie à votre cours !

Mysql (MariaDB)

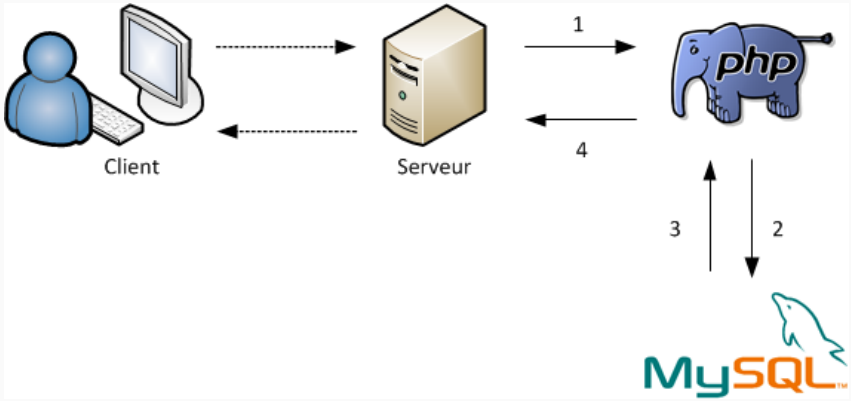
Architecture

Architecture



L'application utilise des données gérées par un sgbd.

Le php intègre une api mariadb (mysql) qui lui permet de faire des requêtes sur un serveur mysql et d'exploiter leurs résultats.



Architecture Client/serveur, communication via TCP/IP.

- Un serveur (daemon) `mysqld` (port 3306).

```
[denis@portable_denis ~]$ ps  -u mysql u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mysql      751  0.0  1.1 533740 48468 ?        Ssl  11:16   0:00 /usr/bin/mysqld --pid-file=/run/mysqld/mysqld.pid
[denis@portable_denis ~]$
```

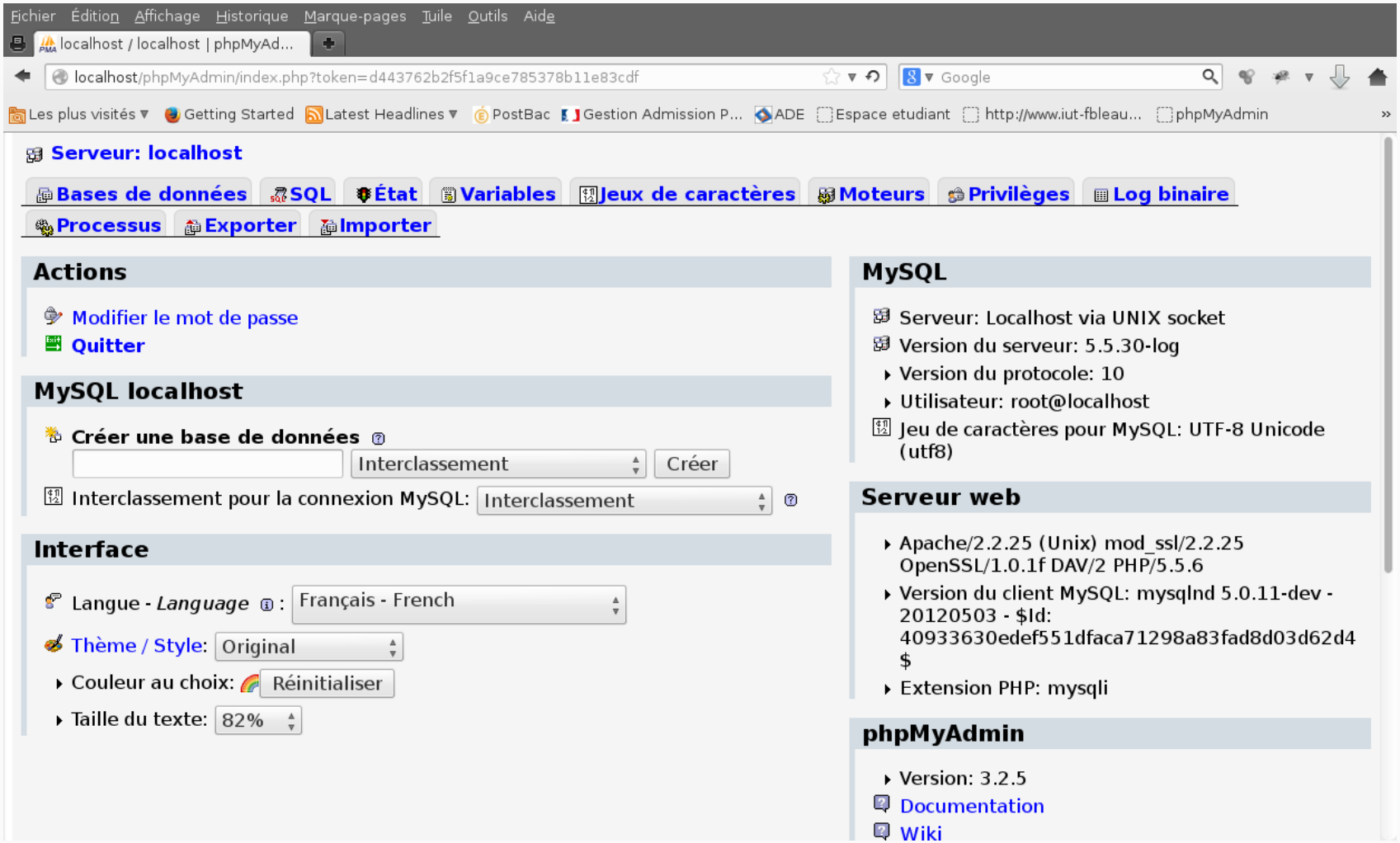
- Une multitude de clients : client en mode console : `mysql`

```
|       220 | Van Cleef       | Lee           |       NULL |
|       221 | Volonte         | Gian Maria    |       NULL |
|       224 | Swank           | Hillary       |       1974 |
+-----------+-----------------+---------------+------------+
193 rows in set (0.07 sec)

mysql> select * from Artiste LIMIT 0,10;
+-----------+-----------+-----------+------------+
| idArtiste | nom       | prenom    | anneeNaiss |
+-----------+-----------+-----------+------------+
|         1 | Lynch     | David     |       1946 |
|         3 | Hitchcock | Alfred    |       1899 |
|         4 | Scott     | Ridley    |       1937 |
|         5 | Weaver    | Sigourney |       1949 |
|         6 | Cameron   | James     |       1954 |
|         9 | Tarkovski | Andrei    |       1932 |
|        10 | Woo       | John      |       1946 |
|        11 | Travolta  | John      |       1954 |
|        12 | Cage      | Nicolas   |       1964 |
|        13 | Burton    | Tim       |       1958 |
+-----------+-----------+-----------+------------+
10 rows in set (0.00 sec)

mysql>
```

`phpmyadmin` : interface web (en php) d'administration

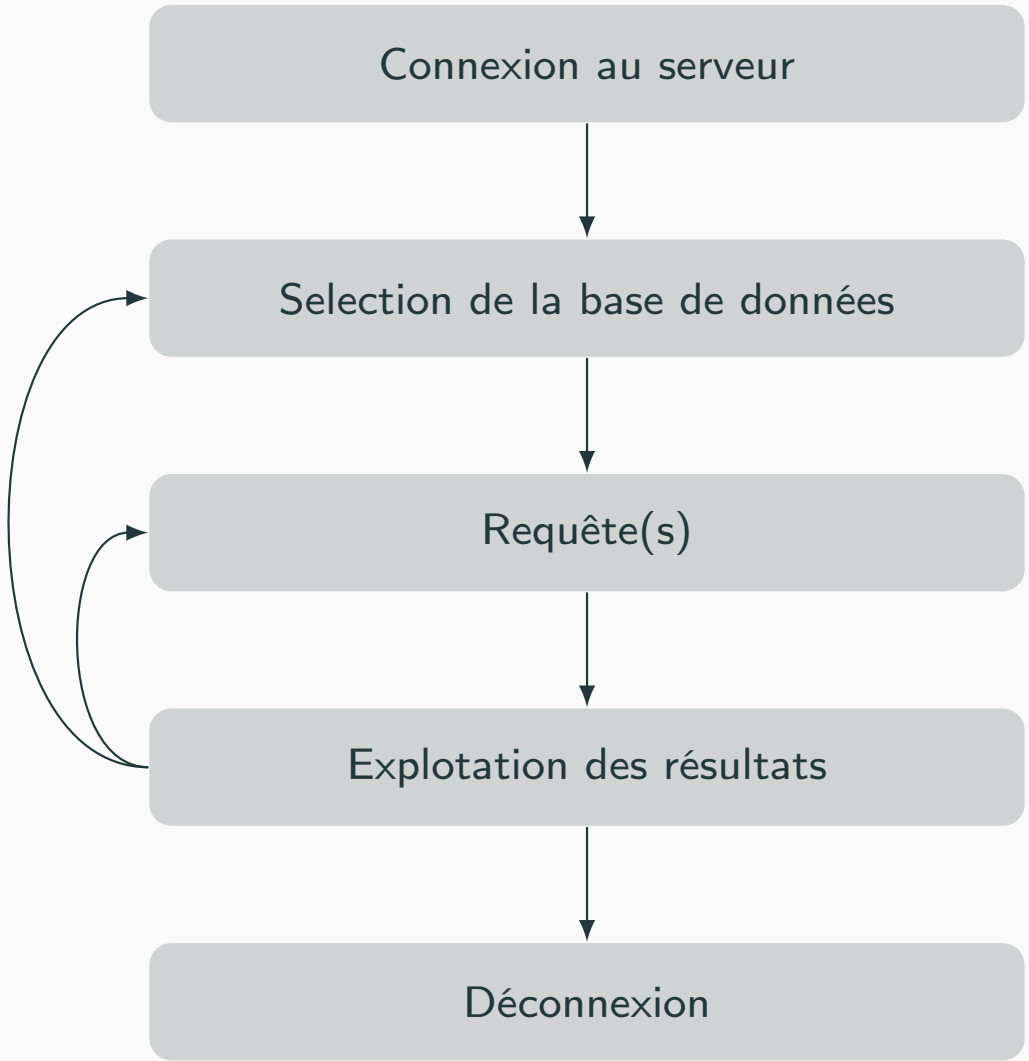


API mysqli

API mysqli

Vue générale

Intéraction Script(s)↔SGBD



php avec mysql

L'extension procédurale `mysql` est obsolète depuis PHP 5.5.0, car elle ne prend pas en charge les nouvelles fonctionalités de mysql5 : procédures stockées, fonctions, transactions. Pas de requêtes préparées, gestion manuelle des échappements. Elle sera supprimée dans un futur proche.

À la place, soit l'extension mysqli ou PDO_MySQL devrait être utilisée.

Dans la suite, on présente l'extension **mysqli**. C'est une extension objet, mais qui offre une version procédurale que l'on utilisera.

Dans la plupart des cas, les noms de fonctions ne diffèrent (de celle de mysql) que par leurs préfixes. Quelques fonctions mysqli prennent un gestionnaire de connexion comme premier argument, alors que la fonction correspondante de l'ancienne interface mysql le prennait comme argument optionnel en dernière position.

Un exemple

On dispose d'une base de données `cinema` qui permet de gérer une filmothèque. En particulier, la table `Film` décrite par la relation suivante :

FILM (idFilm, titre, annee, *#idMes*, *#genre*, resume, *#codePays*)

permet de stocker le titre et l'année des films de la filmothèque.

But

Écrire une page qui affiche la liste de tous nos titres, avec l'année de sortie.

1. Connexion

```
$link=mysqli_connect("host","user","passwd","cinema");
if(!$link){
  die("<p>connexion impossible</p>");
}
```

2. Requête

```
$resultat=mysqli_query($link,
  "SELECT titre,annee
  FROM Film");
```

Un exemple

3. Exploitation

```php
if($resultat) {
  echo "<ul>";
  while($film=mysqli_fetch_assoc($resultat)){
    echo "<li>{$film['titre']} en {$film['annee']} </li>";
  }
  echo "</ul>";
} else
  die("<p>Erreur dans l'éxécution de la requête.</p>");
```

4. Déconnexion

```php
mysqli_close($link);
```

API mysqli

Connexion

Connexion

Fonction de connexion

```
mysqli mysqli_connect (
  [ string $host = ini_get("mysqli.default_host")
  [, string $username = ini_get("mysqli.default_user")
  [, string $passwd = ini_get("mysqli.default_pw")
  [, string $dbname = ""
  [, int $port = ini_get("mysqli.default_port")
  [, string $socket = ini_get("mysqli.default_socket") ]]]]]] )
```

Exemple :

```php
$link = mysqli_connect(
   'localhost',
   'my_user',
   'my_password',
   'my_db'
);

if (!$link) {
   die('Erreur de connexion (' . mysqli_connect_errno() . ') '
      . mysqli_connect_error());
}

echo 'Succès... ' . mysqli_get_host_info($link) . "\n";

mysqli_close($link);
```

- La fonction

```
mysqli_connect()
```

  renvoie un identifiant de connexion, généralement nécessaire par la suite. Renvoie `FALSE` en cas d'échec.
- La fonction

```
mysqli_select_db()
```

  permet de selectionner une base de données. Renvoie `TRUE` ou `FALSE`.

API mysqli

Requetes

Requêtes

Il existe plusieurs types de requêtes :

- Lecture.
- Modification/Insertion.
- Suppression.

Avec PHP (pour l'instant)

On utilise directement le langage SQL à partir de php.

La fonction

```
mixed mysqli_query (mysqli $link , string $query )
```

permet d'envoyer une requête au serveur, et retourne le resultat de l'éxecution sous forme d'un objet `mysqli_result` ou un booléen.

Caractères spéciaux

Certains caractères dans des paramètres de bases de données mysql doivent être protéger (apostrophe par exemple). Il faut (on peut) utiliser une fonction spécifique à mysqli :

```
string mysqli_real_escape_string (
  mysqli $link ,
  string $escapestr );
```

```
$nom = mysqli_real_escape_string($l,$_POST['nom']);
$prenom = mysqli_real_escape_string($l,$_POST['prenom']);
$adresse = mysqli_real_escape_string($l,$_POST['adresse']);
$conn = mysqli_connect("localhost","user","pass","db");
mysqli_query($conn, "INSERT INTO etudiants VALUES
            ('$nom','$prenom','$adresse')"
);
```

API mysqli

Extraction

Sélection

Pour une requête de selection, la fonction `mysqli_query` renvoie un `mysqli_result` qui représente le jeu de résultats selectionnés (analogie avec les curseurs en PL/SQL). On peut récupérer un à un les résultats avec :

`mysqli_fetch_array()`	retourne un enregistrement sous la forme d'un tableau associatif, numérique, ou les deux.
`mysqli_fetch_object()`	retourne un enregistrement sous la forme d'un objet
`mysqli_fetch_row()`	retourne un enregistrement sous la forme d'un tableau numérique
`mysqli_fetch_assoc()`	retourne un enregistrement sous la forme d'un tableau associatif

La fonction `mysqli_num_rows()` permet de connaître le nombre d'enregistrements retournés par une requête de selection.

```php
$result=mysqli_query($conn,"SELECT nom,login FROM etudiant");

// version mysql_fetch_array
while ($ligne=mysql_fetch_assoc($result)){
  echo "<li>Nom : {$ligne['nom']}
    login : {$ligne['login']}</li>";
}

// version mysql_fetch_row
while ($ligne=mysql_fetch_row($result)){
  echo "<li>Nom : {$ligne[0]}
    login : {$ligne[1]}</li>";
}

// version mysql_fetch_object
while ($ligne=mysql_fetch_object($result)){
  echo "<li>Nom : {$ligne->nom} login :
    {$ligne->login}</li>";
}
```

Remarque : depuis php 5.4, il est possible d'utiliser `foreach` pour itérer un `mysqli_result` :

```php
<?php

$resultat=mysqli_query($link,"SELECT titre,annee
                              FROM Film");

if($resultat) {
  echo "<ul>";
  foreach($resultat as $enr){
    echo "<li>{$enr['titre']} en {$enr['annee']} </li>";
  }
  echo "</ul>";
}
?>
```

Insertion/Modification

Les requêtes de type INSERT, UPDATE, REPLACE ou DELETE ne retournent normalement pas de données. Néanmoins, il est possible de savoir combien d'enregistrement effectivement insérés, modifiés ou supprimés lors de la **dernière requête** avec la fonction

```
mysqli_affected_rows()
```

```
$query="UPDATE etudiant
    SET age=age+1
    WHERE anniversisaire='1995-11-11';";
mysqli_query($conn,$query);
echo mysqli_affected_rows($conn)."etudiants concernes";
```

Dans le cas d'un SELECT `myqsli_affected_rows()` se comporte comme `mysqli_num_rows`.

Remarque : lors d'une requête d'insertion dans une table contenant un champs autoincrémenté, on peut récupérer sa valeur pour la dernière insertion avec

```
mysqli_insert_id()
```

```
<?php
$query="INSERT INTO etudiant ....";
mysqli_query($conn,$query);
echo "Identifiant etudiant : ".mysql_insert_id($conn);
?>
```

API mysqli

Déconnexion

Déconnexion

La connexion au serveur est fermé automatiquement à la fin du script PHP. On peut néanmoins le faire manuellement à l'aide la fonction

```
mysqli_close()
```

Interférence HTML/PHP avec SQL

- `mysqli_real_escape_tring`

  pour protéger les caractères apostrophes, antislashes et guillemets pour mysql.

- `trim`

  qui supprime les espaces en début et fin de mot, avant insertion dans la base.

- `htmlentities`

  qui permet de convertir les caractères spéciaux d'HTML : par exemple `&,<,>,",'`.

- `nl2br`

  de manière à convertir les sauts lignes en balise `<br />`.

Interférence HTML/PHP avec SQL

Valider et nettoyer les données "externes" avec les fonctions `filter_var` et/ou `filter_input`.

API mysqli

listes des fonctions mysqli

Pour une liste exhaustive, `http://php.net/manual/fr/mysqli.summary.php`

`mysqli_affected_row()`	nombre d'enregistrements concernés par la dernière requête.
`mysqli_close()`	Deconnexion.
`mysqli_connect()`	Connexion à un serveur.
`mysqli_data_seek()`	déplace le pointeur interne de résultat.
`mysqli_query()`	Exécution d'une requête.
`mysqli_error()`	Retourne le texte associé à l'erreur provoqué par la dernière requête.
`mysqli_escape_string()`	Protège une chaîne pour la passer à `mysql_query`.
`mysqli_fetch_array()`	Lit une ligne de résultat dans tableau.
`mysqli_fetch_assoc()`	Idem, dans un tableau associatif.
`mysqli_fetch_object()`	Idem, dans un objet.
`mysqli_fetch_row()`	Idem, dans un tableau numérique.

Suite...

`mysqli_free_result()`	Efface le résultat de la mémoire.
`mysqli_insert_id()`	Retourne l''identifiant de la dernière requête INSERT.
`mysqli_num_fields()`	Retourne le nombre de champs d'un résultat.
`mysqli_fetch_fields`	Retourne le prochain champs dans le jeu de résultats.
`mysqli_num_rows()`	Retourne le nombre de lignes d'un resultat de SELECT.
`mysqli_select_db()`	Selectionne une base de données.

Requêtes préparées

La base de données MySQL supporte les requêtes préparées. Une requête préparée ou requête paramétrable est utilisée pour exécuter la même requête plusieurs fois, avec une grande efficacité.

Deux étapes : la préparation et l'exécution.

Lors de la préparation, un template de requête est envoyé au serveur de base de données. Le serveur effectue une vérification de la syntaxe, et initialise les ressources internes du serveur pour une utilisation ultérieure.

Le serveur MySQL supporte le mode anonyme, avec des marqueurs de position utilisant le caractère ?.

Préparation

```
mysqli_stmt mysqli_prepare (mysqli $link , string $query)
```

```php
$link = mysqli_connect("localhost","","","cinema");
if (!$link) die ("pb");
$stmt = mysqli_prepare(
 $link,
 "INSERT
  INTO Artiste (nom,prenom,anneeNaiss)
  VALUES (?,?,?)"
);
if (!$stmt) die ("pb");
```

Notez bien la présence des marqueurs " ?".

Liaison et exécution

```
bool mysqli_stmt_bind_param (
  mysqli_stmt $stmt ,
  string $types ,
  mixed &$var1 [, mixed &$var2, ... ] )
```

types est une chaîne de caractères qui spécifie les types des variables à lier :

Caractère	Description
i	correspond à une variable de type entier
d	correspond à une variable de type nombre décimal
s	correspond à une variable de type chaîne de caractères
b	correspond à une variable de type BLOB, qui sera envoyé par paquets

Exécution

```
bool mysqli_execute ( mysqli_stmt $stmt )
```

```php
$nom = "Ford";
$prenom = "John";
$annee = 1894;
mysqli_stmt_bind_param($stmt,"ssi",$nom,$prenom,$annee);
mysqli_execute($stmt);
```

Si la requête est UPDATE, DELETE ou INSERT, le nombre total de lignes affectées est disponible via la fonction

```
int mysqli_stmt_affected_rows ( mysqli_stmt $stmt )
```

Si la fonction génère un résultat, il sera disponible via la fonction

```
bool mysqli_stmt_fetch ( mysqli_stmt $stmt )
```

après avoir lié le résultat avec des variables par

```
bool mysqli_stmt_bind_param (
  mysqli_stmt $stmt ,
  string $types ,
  mixed &$var1 [, mixed &$var2, ... ] )
```

Récupérer les résultats un à un

```php
$stmt = mysqli_prepare(
  $link,
  "SELECT nom,prenom,anneeNaiss
   FROM Artiste
   WHERE anneeNaiss > 1960");
if (!$stmt) die("pb");
mysqli_execute($stmt);
mysqli_stmt_bind_result($stmt,$nom,$prenom,$annee);
while(mysqli_stmt_fetch($stmt)){
  echo "$nom $prenom : $annee";
}
```

Compter le nombre de résultats

```php
mysqli_stmt_store_result($stmt);
$nb = mysqli_stmt_num_rows($stmt);
```

On peut aussi récupérer le résultat avec

```
mysqli_result mysqli_stmt_get_result ( mysqli_stmt $stmt )
```

que l'on peut traiter de façon usuelle :

```
$stmt = mysqli_prepare(
  $link,
  "SELECT nom,prenom,anneeNaiss
   FROM Artiste
   WHERE anneeNaiss > 1960");
mysqli_execute($stmt);

$res = mysqli_stmt_get_result($stmt);
foreach ($res as $enr){
  echo "{$enr['nom']} {$enr['prenom']}";
}
```