Accéder aux bases de données avec Mysqli de PHP

1. Objectifs

- Être capable d'utiliser les API php pour accéder à une base de données MySQL

2. Prérequis

- Pour pouvoir comprendre au mieux ce tutoriel, quelques connaissances sont requises.
  - **MySQL**: Vous devez savoir ce qu'est une BDD (base de données), ainsi que tout le vocabulaire comme : table, champ, ligne, requête SQL, ...
  - Il vous est aussi conseillé de connaître les bases du langage SQL pour pouvoir comprendre les requêtes de base.

3. Présentation

- Il existe trois API principales pour se connecter à MySQL :

  - **L'extension `mysql` (pour les versions PHP < 4.1.3)**
    - L'extension `mysql_`: ce sont des fonctions qui permettent d'accéder à une base de données MySQL, et donc de communiquer avec MySQL.
    - Leur nom commence toujours par **mysql_**.
    - Toutefois, ces fonctions sont vieilles, et on recommande de ne plus les utiliser aujourd'hui.
  - **L'extension `mysqli` (mysql improved pour les versions PHP > 5)**
    - L'extension **`mysqli_`**: ce sont des fonctions améliorées d'accès à MySQL.
  - **PHP Data Objects (PDO)**
    - PHP Data Objects (PDO) qui a des fonctionnalités moins avancées que mysqli mais permet de travailler avec différentes bases de données en conservant le même code.

> *Il est recommandé d'utiliser soit l'extension mysqli, soit l'extension PDO_MySQL. Il n'est pas recommandé d'utiliser l'ancienne extension mysql pour de nouveaux*

apcpedagogie.com

> *développements sachant qu'elle est obsolète depuis PHP 5.5.0, et a été supprimée en PHP 7.*

- Il existe une différence notable entre ces deux API : l'extension **MySQLi** ne va fonctionner qu'avec les bases de données MySQL tandis que **PDO** va fonctionner avec 12 systèmes de bases de données différents.
- Nous utiliserons dans ce cours l'API **PDO**.
- L' extension **Mysqli** peut être utilisée de **manière procédurale** ou **orientée objet**. De manière procédurale, les fonctions sont appelées, tandis que de manière orientée objet, un objet classe est utilisé pour effectuer des opérations sur la base de données.

4. **Exemple #1 Comparaison des 3 APIs MySQL**

5. Notions des APIs procédurales et APIs orientées objet

- Les APIs PHP pour la connexion à MySQL et la sélection de la base de données peuvent être conçues de deux façons principales : **procédurale** ou **orientée objet**.

- **API procédurale**

  - Une **API procédurale** est une interface de programmation d'application qui permet aux développeurs d'écrire des fonctions ou des procédures pour interagir avec une base de données. Dans le cas de PHP et de MySQL, l'API procédurale permet d'établir une connexion avec une base de données MySQL et d'exécuter des requêtes SQL.
  - Dans **une API procédurale**, vous pouvez utiliser des fonctions telles que `mysqli_connect()` pour établir une connexion à une base de données **MySQL**, et `mysqli_select_db()` pour sélectionner une base de données spécifique.
  - Voici un exemple de code utilisant l'API procédurale pour se connecter à une base de données MySQL dans PHP :
    Script Php

```
<?php
// Identifiants de connexion à la base de données
$servername = "localhost";
$username = "nom_utilisateur";
$password = "mot_de_passe";
$dbname = "nom_base_de_données";
```

apcpedagogie.com

```
// Connexion à la base de données
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Vérification de la connexion
if (!$conn) {
    die("Connexion échouée : " . mysqli_connect_error());
}

// Exécution d'une requête SQL
$sql = "SELECT * FROM ma_table";
$result = mysqli_query($conn, $sql);

// Traitement des résultats de la requête
if (mysqli_num_rows($result) > 0) {
    while($row = mysqli_fetch_assoc($result)) {
        echo "Colonne1: " . $row["colonne1"]. " - Colonne2: " .
$row["colonne2"]. "<br>";
    }
} else {
    echo "0 résultats";
}

// Fermeture de la connexion
mysqli_close($conn);
?>
```

- Ce code établit une connexion à une base de données MySQL en utilisant les identifiants de connexion fournis, puis exécute une requête SQL pour récupérer les données de la table "**ma_table**". Les résultats sont ensuite affichés à l'écran avant que la connexion ne soit fermée.

- **API orientée objet**

- Une API orientée objet pour se connecter à une base de données MySQL en PHP est une interface de programmation d'application (API) qui permet à un développeur d'interagir avec une base de données MySQL à travers des objets et des méthodes dans le langage de programmation PHP.
- Il existe plusieurs bibliothèques PHP qui implémentent des API orientées objet pour se connecter à MySQL, mais la plus couramment utilisée est l'extension PHP MySQLi (**MySQL Improved**). Cette extension offre une interface orientée objet ainsi qu'une interface procédurale pour interagir avec les bases de données

apcpedagogie.com

MySQL.

- Dans une **API orientée objet**, vous pouvez créer un objet qui représente la connexion à la base de données en utilisant la classe mysqli, puis appeler des méthodes de cet objet pour effectuer des opérations telles que la sélection de la base de données. Par exemple, vous pouvez créer un objet mysqli en utilisant le constructeur new mysqli(), puis appeler la méthode select_db() sur cet objet pour sélectionner une base de données spécifique.
- Voici un exemple de code utilisant l'API orientée objet MySQLi pour se connecter à une base de données MySQL :
  Script Php

```
<?php
// Connexion à la base de données
$mysqli = new mysqli("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Vérification de la connexion
if ($mysqli->connect_errno) {
    echo "Échec de la connexion à MySQL: " . $mysqli->connect_error;
    exit();
}

// Exécution d'une requête SQL
$resultat = $mysqli->query("SELECT * FROM table");

// Traitement du résultat
while ($row = $resultat->fetch_assoc()) {
    echo $row["champ1"] . " " . $row["champ2"];
}

// Fermeture de la connexion
$mysqli->close();
?>
```

- Dans cet exemple, la classe **mysqli** est utilisée pour instancier un objet de connexion à la base de données MySQL. Les méthodes de cette classe, telles que **connect_errno**, **query** et **fetch_assoc**, sont ensuite utilisées pour interagir avec la base de données et récupérer des données.

- Les APIs PHP orientées objet peuvent être plus complexes à utiliser que les API procédurales, car vous devez comprendre les concepts de la programmation orientée objet pour les utiliser correctement. Cependant, les API orientées objet offrent également

apcpedagogie.com

des avantages, notamment une encapsulation plus forte des données et une réutilisabilité accrue du code.

6. Les étapes pour établir une connexion à une base de données MySQL

- Voici les étapes pour établir une connexion à une base de données MySQL :
  - ***1-*** *Installer le pilote MySQL :*
    - Tout d'abord, il est nécessaire d'avoir un serveur MySQL installé et en cours d'exécution sur votre ordinateur ou sur un serveur distant.
    - Avant de pouvoir établir une connexion à une base de données MySQL, il est nécessaire d'installer le pilote MySQL approprié pour votre langage de programmation ou votre plate-forme.
  - ***2-*** *Vérifier les informations de connexion :*
    - Avant de pouvoir établir une connexion à une base de données MySQL, vous devez disposer des informations suivantes :

      l'**adresse IP** ou le nom d'**hôte** du serveur MySQL, le numéro de **port** MySQL, le nom de la **base de données**, le nom d'**utilisateur** et le **mot de passe**.
  - ***3-****Créer une connexion :*
    - Une fois que vous avez les informations d'identification, vous pouvez établir une connexion à la base de données en utilisant une API de base de données comme MySQLi ou PDO. Pour cela, vous devez instancier un objet de connexion en fournissant les informations d'identification appropriées.
    - Vous pouvez créer une connexion à une base de données MySQL en utilisant les API de connexion fournies par le pilote MySQL. En général, cela implique de fournir les informations de connexion et d'initialiser une connexion à la base de données.
  - ***4-****Exécuter des requêtes :*
    - Après avoir instancié l'objet de connexion, vous pouvez utiliser les méthodes de cette classe pour interagir avec la base de données, telles que l'exécution de requêtes SQL, la récupération de données, etc.
    - Une fois que vous avez établi une connexion à la base de données MySQL, vous pouvez exécuter des requêtes SQL pour récupérer, insérer, mettre à jour ou supprimer des données dans la base de données.
  - ***5-*** *Fermer la connexion :*
    - Enfin, une fois que vous avez terminé d'interagir avec la base de données, vous devez fermer la connexion en appelant la méthode appropriée. Cela permet de libérer les ressources système et de fermer la connexion à la base

apcpedagogie.com

de données.

- ***1- Ouverture d'une connexion à MySQL et sélection de la base de données avec MySqli***

  - Pour se connecter à la base de données il faut fournir un **nom de domaine**, une **adresse IP** ou un alias,un **login** et un **mot de passe**.
    - **host** (hôte) : Facultatif. Spécifie un nom d'hôte ou une adresse IP
    - **username**(nom d'utilisateur) : Facultatif. Spécifie le nom d'utilisateur MySQL
    - **password** (mot de passe) : Facultatif. Spécifie le mot de passe MySQL
    - **dbname** (nom base de données): Facultatif. Spécifie la base de données par défaut à utiliser
    - **port**: Facultatif. Spécifie le numéro de port pour tenter de se connecter au serveur MySQL
    - **socket**: Facultatif.
      - Un socket est un canal de communication bidirectionnel qui permet à deux processus de communiquer entre eux sur une même machine ou sur un réseau.
      - Dans le contexte de MySQL, le socket est un fichier qui permet aux applications d'établir une connexion avec le serveur de bases de données MySQL via un protocole de communication appelé protocole de socket.
      - Les fichiers socket sont généralement des fichiers Unix situés dans le répertoire **/var/run/mysqld/ ou /tmp/**.

```
$mysqli = new mysqli('localhost', 'username', 'password', 'dbname',
null, '/var/run/mysqld/mysqld.sock');
```

- **Syntaxe**
  - **Style orienté objet:**

```
$mysqli = new mysqli(host, username, password, dbname, port, socket)
```

  - **Style procédural:**

```
$mysqli_connect(host, username, password, dbname, port, socket)
```

7. ***2- Vérifier la connexion***

   - **Style procédurale:**

Script Php

```
// Établir la connexion à la base de données
$connexion = mysqli_connect("localhost", "utilisateur", "motdepasse",
 "basededonnees");

// Vérifier si la connexion a réussi
if (!$connexion) {
    echo "Échec de la connexion à la base de données : ".
mysqli_connect_error();
} else {
    echo "Connexion réussie à la base de données.";
}
```

Interprétation

- Dans cet exemple, nous utilisons la fonction **mysqli_connect** fournie par PHP pour établir une connexion à la base de données en passant les informations de connexion nécessaires comme arguments.
- Nous vérifions ensuite si la connexion a réussi en vérifiant la valeur de retour de la fonction **mysqli_connect**. Si la connexion a échoué, nous affichons le message d'erreur approprié en utilisant la fonction **mysqli_connect_error**. Sinon, nous affichons un message indiquant que la connexion a réussi.

- **Style orienté objet:**

Script Php

```
// Créer une instance de la classe mysqli
$mysqli = new mysqli("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Vérifier si la connexion a réussi
if ($mysqli->connect_errno) {
    echo "Échec de la connexion à la base de données : " . $mysqli-
>connect_error;
} else {
    echo "Connexion réussie à la base de données.";
}
```

Interprétation

- Dans cet exemple, nous utilisons la classe mysqli fournie par PHP pour créer une instance de l'objet mysqli et nous passons les informations de connexion nécessaires au constructeur.

- Nous vérifions ensuite si la connexion a réussi en vérifiant la valeur de la propriété connect_errno de l'objet mysqli. Si la connexion a échoué, nous affichons le message d'erreur approprié en utilisant la propriété connect_error. Sinon, nous affichons un message indiquant que la connexion a réussi.

8. ***3-* Sélectionner des enregistrements**

   - Les enregistrements de la base de données peuvent être exploités de manière procédurale ou orientée objet.
   - La fonction `mysqli_query` est utilisée pour effectuer une requête sur la base de données.
   - Une fois connecté à la base de données, à l' objet de connexion une requête est transmise à `mysqli_query` et renvoie un ensemble de résultats.
   - Il existe plusieurs fonctions permettant de récupérer les résultats à partir de l'objet ou du descripteur de résultat. Cet objet ou ce descripteur est essentiel pour accéder aux lignes renvoyées par la requête

   - **mysqli_fetch_assoc**

     - La fonction `mysqli_fetch_assoc()` est une fonction procédurale utilisée avec la bibliothèque **MySQLi** pour récupérer une ligne de résultat sous forme de tableau associatif.
     - Si vous préférez une syntaxe orientée objet, vous pouvez utiliser la méthode **fetch_assoc()** de l'objet **mysqli_result**.
     - La fonction **`mysqli_fetch_assoc ()`** est utilisée pour renvoyer un tableau associatif représentant la ligne suivante dans le jeu de résultats pour le résultat représenté par le paramètre `resultat`, où chaque clé du tableau représente le nom d'une des colonnes du jeu de résultats.
     - L'ensemble de résultats est transmis à la fonction **`mysqli_fetch_assoc`** et renvoie une ligne de données sous forme de **tableau associatif**.
     - **Style procédural:**

       Script Php

```
// Connexion à la base de données
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Vérification de la connexion
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
```

```
// Exécution de la requête
$sql = "SELECT id, nom, prenom FROM utilisateurs";
$result = mysqli_query($conn, $sql);

// Récupération des résultats sous forme de tableau associatif
while ($row = mysqli_fetch_assoc($result)) {
    echo "ID : " . $row["id"] . " - Nom : " . $row["nom"] . " - Prénom :
" . $row["prenom"] . "
";
}

// Fermeture de la connexion
mysqli_close($conn);
```

Interprétation

- Dans cet exemple, nous avons utilisé la fonction **mysqli_connect()** pour établir une connexion à la base de données et stocké le résultat dans la variable **$conn**.
- Nous avons ensuite vérifié que la connexion était réussie à l'aide de la fonction **mysqli_connect_error()**.
- Nous avons exécuté la requête SELECT à l'aide de la fonction **mysqli_query()** et stocké le résultat dans la variable **$result**.
- Nous avons ensuite utilisé la fonction **mysqli_fetch_assoc()** pour récupérer chaque ligne de résultat sous forme de tableau associatif dans la boucle while. Enfin, nous avons fermé la connexion à la base de données à l'aide de la fonction **mysqli_close()**.
- Notez que la syntaxe procédurale peut être plus concise pour certains développeurs, mais elle peut également être plus difficile à lire et à comprendre pour d'autres.

- **Style orienté objet:**

  Script Php

```
// Connexion à la base de données
$conn = new mysqli($servername, $username, $password, $dbname);

// Vérification de la connexion
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

apcpedagogie.com

```
// Exécution de la requête
$sql = "SELECT id, nom, prenom FROM utilisateurs";
$result = $conn->query($sql);

// Récupération des résultats sous forme de tableau associatif
while ($row = $result->fetch_assoc()) {
    echo "ID : " . $row["id"] . " - Nom : " . $row["nom"] . " - Prénom :
" . $row["prenom"] . "
";
}

// Fermeture de la connexion
$conn->close();
```

Interprétation

- Dans cet exemple, nous avons utilisé la méthode query() de l'objet mysqli pour exécuter la requête SELECT et stocké le résultat dans l'objet mysqli_result $result. Nous avons ensuite utilisé la méthode fetch_assoc() de cet objet pour récupérer chaque ligne de résultat sous forme de tableau associatif dans la boucle while. Enfin, nous avons fermé la connexion à la base de données à l'aide de la méthode close() de l'objet mysqli.
- Notez que la syntaxe orientée objet peut être plus facile à lire et à comprendre pour certains développeurs, mais elle peut également être légèrement plus verbeuse que la syntaxe procédurale.

- **mysqli_fetch_array**

  - La fonction **mysqli_fetch_array** est utilisée pour récupérer les enregistrements de la base de données. Il récupère une seule ligne de l'ensemble de résultats
  - La fonction **mysqli_fetch_array** retourne une ligne de résultat sous la forme d'un **tableau associatif**, d'un **tableau indexé**, ou **les deux**
  - **MYSQLI_BOTH**, **MYSQLI_ASSOC** et **MYSQLI_NUM** sont des constantes utilisées comme arguments facultatifs dans la fonction **mysqli_fetch_array**. Elles définissent le type de tableau renvoyé par la fonction.
    - `MYSQLI_BOTH`: renvoie un tableau associatif et numérique combiné (par défaut)
    - `MYSQLI_ASSOC`: renvoie un tableau associatif
    - `MYSQLI_NUM`: renvoie un tableau numérique

apcpedagogie.com

```
$query = "SELECT * FROM users";
$result = mysqli_query($conn, $query);

while ($row = mysqli_fetch_array($result, MYSQLI_BOTH)) {
  echo $row[0] . " " . $row['nom'] . " " . $row[2] . "
";
}

while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
  echo $row['id'] . " " . $row['nom'] . " " . $row['email'] . "
";
}

while ($row = mysqli_fetch_array($result, MYSQLI_NUM)) {
  echo $row[0] . " " . $row[1] . " " . $row[2] . "
";
}
```

- **Style procédural:**
  - La fonction `mysqli_fetch_array()` en style procédural est utilisée pour récupérer une ligne de résultats sous forme de tableau associatif, de tableau numérique, ou des deux, à partir d'un résultat de requête **MySQLi**.

    Script Php

```
// Connexion à la base de données
$conn = mysqli_connect("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if (!$conn) {
    die("La connexion a échoué: " . mysqli_connect_error());
}

// Exécution d'une requête
$resultat = mysqli_query($conn, "SELECT * FROM ma_table");

// Boucle à travers les résultats
while ($row = mysqli_fetch_array($resultat, MYSQLI_BOTH)) {
    // Affichage des résultats
    echo $row['id'] . " - " . $row['nom'] . " - " . $row['email'] . "
";
```

apcpedagogie.com

```
}

// Libération de la mémoire
mysqli_free_result($resultat);

// Fermeture de la connexion
mysqli_close($conn);
```

Interprétation

- Dans cet exemple, nous nous connectons à la base de données MySQL en utilisant **mysqli_connect()**, exécutons une requête SELECT pour récupérer toutes les lignes d'une table, et nous bouclons à travers les résultats en utilisant **mysqli_fetch_array()** pour afficher chaque ligne de résultats. Enfin, nous libérons la mémoire et fermons la connexion.

- **Style orienté objet:**
  - La fonction `mysqli_fetch_array()` en style orienté objet est utilisée pour récupérer une ligne de résultats sous forme de tableau associatif, de tableau numérique, ou des deux, à partir d'un résultat de requête **MySQLi**.

Script Php

```
// Connexion à la base de données
$conn = new mysqli("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if ($conn->connect_error) {
    die("La connexion a échoué: " . $conn->connect_error);
}

// Exécution d'une requête
$resultat = $conn->query("SELECT * FROM ma_table");

// Boucle à travers les résultats
while ($row = $resultat->fetch_array(MYSQLI_BOTH)) {
    // Affichage des résultats
    echo $row['id'] . " - " . $row['nom'] . " - " . $row['email'] . "
";
}

// Libération de la mémoire
$resultat->free();
```

apcpedagogie.com

```
// Fermeture de la connexion
$conn->close();
```

Interprétation

- Dans cet exemple, nous nous connectons à la base de données MySQL en utilisant la classe **mysqli**, exécutons une requête SELECT pour récupérer toutes les lignes d'une table, et nous bouclons à travers les résultats en utilisant la méthode **fetch_array()** pour afficher chaque ligne de résultats. Enfin, nous libérons la mémoire et fermons la connexion.

- **mysqli_fetch_all**

  - La fonction **mysqli_fetch_all()** est utilisée pour récupérer toutes les lignes et renvoie l'ensemble de résultats sous forme de tableau associatif.
  - **Style procédural:**
    - La fonction `mysqli_fetch_all()` en style procédural est utilisée pour récupérer toutes les lignes de résultats sous forme de tableau associatif, de tableau numérique, ou des deux, à partir d'un résultat de requête **MySQLi**.

      Script Php

```
// Connexion à la base de données
$conn = mysqli_connect("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if (!$conn) {
    die("La connexion a échoué: " . mysqli_connect_error());
}

// Exécution d'une requête
$resultat = mysqli_query($conn, "SELECT * FROM ma_table");

// Récupération de toutes les lignes de résultats
$rows = mysqli_fetch_all($resultat, MYSQLI_ASSOC);

// Affichage des résultats
foreach ($rows as $row) {
    echo $row['id'] . " - " . $row['nom'] . " - " . $row['email'] . "
";
}
```

apcpedagogie.com

```
// Libération de la mémoire
mysqli_free_result($resultat);

// Fermeture de la connexion
mysqli_close($conn);
```

Interprétation

- Dans cet exemple, nous nous connectons à la base de données MySQL en utilisant mysqli_connect(), exécutons une requête SELECT pour récupérer toutes les lignes d'une table, et nous utilisons la fonction mysqli_fetch_all() pour récupérer toutes les lignes de résultats sous forme de tableau associatif. Nous parcourons ensuite chaque ligne de résultats et les affichons.
- Enfin, nous libérons la mémoire et fermons la connexion.

- **Style orienté objet:**
  - La fonction `mysqli_fetch_all()` en style orienté objet est utilisée pour récupérer toutes les lignes de résultats sous forme de tableau associatif, de tableau numérique, ou des deux, à partir d'un résultat de requête **MySQLi**.

Script Php

```
// Connexion à la base de données
$conn = new mysqli("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if ($conn->connect_error) {
    die("La connexion a échoué: " . $conn->connect_error);
}

// Exécution d'une requête
$resultat = $conn->query("SELECT * FROM ma_table");

// Récupération de toutes les lignes de résultats
$rows = $resultat->fetch_all(MYSQLI_ASSOC);

// Affichage des résultats
foreach ($rows as $row) {
    echo $row['id'] . " - " . $row['nom'] . " - " . $row['email'] . "
";
}
```

apcpedagogie.com

```
// Libération de la mémoire
$resultat->free();

// Fermeture de la connexion
$conn->close();
```

Interprétation

- Dans cet exemple, nous nous connectons à la base de données MySQL en utilisant la classe **mysqli**, exécutons une requête SELECT pour récupérer toutes les lignes d'une table, et nous utilisons la méthode **fetch_all()** pour récupérer toutes les lignes de résultats sous forme de tableau associatif. Nous parcourons ensuite chaque ligne de résultats et les affichons. Enfin, nous libérons la mémoire et fermons la connexion.

- **mysqli_fetch_object()**

  - La fonction `mysqli_fetch_object()` est une fonction de l'API MySQLi de PHP qui permet de récupérer une ligne de résultat sous la forme d'un objet. Les propriétés de l'objet correspondent aux colonnes de la ligne de résultat. Cette fonction peut être utilisée de manière procédurale ou orientée objet.
  - En utilisant `mysqli_fetch_object()`, on peut accéder aux propriétés des objets retournés par la requête en utilisant la notation fléchée (->). Par exemple, si la requête retourne une colonne nommée "nom", on peut accéder à cette propriété en utilisant $objet->nom.
  - **Style procédural:**
    - La fonction `mysqli_fetch_object()` en style procédural est utilisée pour récupérer une ligne de résultat sous forme d'un objet.

    Script Php

```
// Connexion à la base de données
$mysqli = mysqli_connect("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Exécution de la requête
$resultat = mysqli_query($mysqli, "SELECT nom, prenom, age FROM
utilisateurs");

// Récupération des résultats sous forme d'objets
while ($objet = mysqli_fetch_object($resultat)) {
```

```
    echo $objet->nom . " " . $objet->prenom . ", " . $objet->age . " ans
";
}

// Fermeture de la connexion
mysqli_close($mysqli);
```

Interprétation

- Dans cet exemple, nous exécutons une requête qui sélectionne les colonnes "nom", "prenom" et "age" de la table "utilisateurs". Nous récupérons ensuite les résultats sous forme d'objets en utilisant la fonction **mysqli_fetch_object()**. Nous itérons sur les différents objets retournés par cette fonction à l'aide d'une boucle while et nous affichons les valeurs des propriétés "nom", "prenom" et "age" de chaque objet à l'écran. Enfin, nous fermons la connexion à la base de données à l'aide de la fonction mysqli_close().

- **Style orienté objet:**
  - La fonction `mysqli_fetch_object()` est particulièrement utile si l'on souhaite manipuler les résultats d'une requête SQL en utilisant un style orienté objet. Elle permet de créer des objets à partir des résultats d'une requête, ce qui facilite leur manipulation dans le code.

    Script Php

```
// Connexion à la base de données
$conn = mysqli_connect("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Exécution de la requête
$resultat = mysqli_query($conn, "SELECT * FROM utilisateurs");

// Récupération des résultats sous forme d'objets
while ($utilisateur = mysqli_fetch_object($resultat)) {
    echo $utilisateur->nom . " " . $utilisateur->prenom . "
";
}

// Fermeture de la connexion
mysqli_close($conn);
```

Interprétation

- Dans cet exemple, nous exécutons une requête pour sélectionner toutes les lignes de la table "utilisateurs". Nous récupérons ensuite les résultats sous forme d'objets à l'aide de la fonction **mysqli_fetch_object()**. Nous parcourons les résultats à l'aide d'une boucle while et nous affichons le nom et le prénom de chaque utilisateur à l'écran.
- Notez que pour accéder aux propriétés de chaque objet, nous utilisons la syntaxe de l'objet suivi du nom de la propriété (par exemple $utilisateur->nom).

9. ***4-* Vérification d'existence des données dans la base de données**

   - Pour vérifier l'existence de données dans une base de données MySQL en utilisant PHP, vous pouvez exécuter une requête SELECT et vérifier si des résultats sont retournés.
   - Voici quelques fonctions PHP qui permettent de vérifier l'existence de données dans une base de données MySQL :
     - `mysqli_num_rows()`: Cette fonction renvoie le nombre de lignes dans un résultat de requête. Elle peut être utilisée pour vérifier si une requête a retourné des résultats ou non.
     - `mysqli_fetch_array()` : Cette fonction renvoie la prochaine ligne du résultat d'une requête sous forme de tableau associatif ou numérique. Elle peut être utilisée pour vérifier si une requête a retourné des données ou non.
     - `mysqli_fetch_assoc()`: Cette fonction renvoie la prochaine ligne du résultat d'une requête sous forme de tableau associatif. Elle peut être utilisée pour vérifier si une requête a retourné des données ou non.
     - `mysqli_fetch_row()`: Cette fonction renvoie la prochaine ligne du résultat d'une requête sous forme de tableau numérique. Elle peut être utilisée pour vérifier si une requête a retourné des données ou non.
     - `mysqli_affected_rows()`: Cette fonction renvoie le nombre de lignes affectées par la dernière requête MySQL. Elle peut être utilisée pour vérifier si une requête a modifié des données ou non.
     - `mysqli_query()`: Cette fonction exécute une requête MySQL et renvoie un résultat (si la requête est une requête de sélection) ou un booléen (si la requête est une requête de modification ou de suppression). Elle peut être utilisée pour vérifier si une requête a été exécutée avec succès ou non.
     - Il est important de noter que toutes les fonctions ci-dessus ne vérifient pas directement l'existence de données dans la base de données, mais plutôt si des résultats ont été renvoyés par une requête donnée. Vous devez donc utiliser ces fonctions en conjonction avec une requête appropriée pour vérifier l'existence de

apcpedagogie.com

données dans votre base de données.

- **mysqli_num_rows()**
  - La fonction `mysqli_num_rows()` est une fonction intégrée en PHP qui est utilisée pour renvoyer le nombre de lignes présentes dans le jeu de résultats.
  - **Style procédural:**

    Script Php

```
// Connexion à la base de données
$conn = mysqli_connect("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Vérification de la connexion
if (!$conn) {
    die("Connexion échouée: " . mysqli_connect_error());
}

// Exécution de la requête
$resultat = mysqli_query($conn, "SELECT * FROM table");

// Vérification du nombre de lignes
if (mysqli_num_rows($resultat) > 0) {
    // Des données existent dans la table
} else {
    // Aucune donnée n'existe dans la table
}

// Fermeture de la connexion
mysqli_close($conn);
```

    Interprétation Dans cet exemple, nous nous connectons à la base de données en utilisant la fonction mysqli_connect(). Nous exécutons ensuite une requête SELECT pour récupérer toutes les données de la table. Nous vérifions ensuite le nombre de lignes renvoyées par la requête en utilisant la fonction mysqli_num_rows(). Si la valeur renvoyée est supérieure à zéro, cela signifie qu'il y a des données dans la table. Sinon, cela signifie qu'il n'y a aucune donnée dans la table. Enfin, nous fermons la connexion à la base de données en utilisant la fonction mysqli_close().
  - **Style orienté objet:**

    Script Php

apcpedagogie.com

```
// Connexion à la base de données
$conn = new mysqli("localhost", "utilisateur", "motdepasse",
"basededonnees");

// Vérification de la connexion
if ($conn->connect_error) {
    die("Connexion échouée: " . $conn->connect_error);
}

// Exécution de la requête
$resultat = $conn->query("SELECT * FROM table");

// Vérification du nombre de lignes
if ($resultat->num_rows > 0) {
    // Des données existent dans la table
} else {
    // Aucune donnée n'existe dans la table
}

// Fermeture de la connexion
$conn->close();
```

Interprétation Dans cet exemple, nous créons une instance de la classe **mysqli** pour se connecter à la base de données en utilisant le constructeur de la classe. Nous exécutons ensuite une requête SELECT pour récupérer toutes les données de la table en utilisant la méthode query(). Nous vérifions ensuite le nombre de lignes renvoyées par la requête en utilisant la propriété num_rows. Si la valeur renvoyée est supérieure à zéro, cela signifie qu'il y a des données dans la table. Sinon, cela signifie qu'il n'y a aucune donnée dans la table. Enfin, nous fermons la connexion à la base de données en utilisant la méthode close().

10. ***5-* Fermeture de la connexion à MySQL**

- Normalement, la fermeture de la connexion se fait automatiquement, mais afin de s'en assurer, on peut utiliser la méthode close()
- **Style orienté objet:**
  - Pour fermer une connexion MySQL en PHP orienté objet, vous pouvez utiliser la méthode close() de l'objet mysqli que vous avez créé pour établir la connexion.

    Script Php

```
// Création de l'objet mysqli
$mysqli = new mysqli("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if ($mysqli->connect_errno) {
  echo "Echec lors de la connexion à MySQL : " . $mysqli->connect_error;
  exit();
}

// Utilisation de la connexion

// Fermeture de la connexion
$mysqli->close();
```

Interprétation

- La méthode **close()** ferme la connexion en cours et libère toutes les ressources associées à la connexion. Il est recommandé de toujours fermer la connexion après avoir fini d'utiliser la base de données pour éviter de surcharger le serveur MySQL avec des connexions inutiles.

- **Style procédural:**
  - Pour fermer une connexion MySQL en PHP style procédural, vous pouvez utiliser la fonction mysqli_close() en passant en paramètre le lien de la connexion.

Script Php

```
// Connexion à la base de données
$link = mysqli_connect("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if (!$link) {
    die("La connexion à MySQL a échoué : " . mysqli_connect_error());
}

// Utilisation de la connexion

// Fermeture de la connexion
mysqli_close($link);
```

Interprétation

apcpedagogie.com

- La fonction **mysqli_close()** ferme la connexion en cours et libère toutes les ressources associées à la connexion. Il est recommandé de toujours fermer la connexion après avoir fini d'utiliser la base de données pour éviter de surcharger le serveur MySQL avec des connexions inutiles.

11. Applications

App 01

- **Énoncé**
  - Vous travaillez pour une entreprise de développement de logiciels qui utilise une base de données MySQLi pour stocker des informations sur les clients. Dans cet examen, vous allez devoir écrire du code PHP pour effectuer des opérations sur la base de données.
    - ***1-*** Écrire une requête SQL pour sélectionner tous les clients dont l'adresse e-mail contient le domaine "gmail.com".
    - ***2-***Écrire le code PHP qui exécute la requête SQL de la question précédente et qui affiche le nombre de clients trouvés.
    - ***3-***Écrire le code PHP qui utilise la fonction **mysqli_fetch_array()** pour afficher les résultats de la requête de la question 1 sous forme de tableau HTML.

  Solution
- Requête SQL pour sélectionner tous les clients dont l'adresse e-mail contient le domaine "gmail.com" :

```
SELECT * FROM clients WHERE email LIKE '%@gmail.com';
```

- Code PHP qui exécute la requête SQL précédente et qui affiche le nombre de clients trouvés :

```
// Connexion à la base de données
$link = mysqli_connect("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if (mysqli_connect_errno()) {
    printf("Échec de la connexion : %s\n", mysqli_connect_error());
```

apcpedagogie.com

```
    exit();
}

// Exécution de la requête
$query = mysqli_query($link, "SELECT * FROM clients WHERE email LIKE
'%@gmail.com'");

// Récupération du nombre de résultats
$num_rows = mysqli_num_rows($query);

// Affichage du nombre de résultats
echo "Nombre de clients trouvés : " . $num_rows;
```

- Code PHP qui utilise la fonction mysqli_fetch_array() pour afficher les résultats de la requête précédente sous forme de tableau HTML :

```
// Connexion à la base de données
$link = mysqli_connect("localhost", "utilisateur", "mot_de_passe",
"ma_base_de_donnees");

// Vérification de la connexion
if (mysqli_connect_errno()) {
    printf("Échec de la connexion : %s\n", mysqli_connect_error());
    exit();
}

// Exécution de la requête
$query = mysqli_query($link, "SELECT * FROM clients WHERE email LIKE
'%@gmail.com'");

// Affichage des résultats sous forme de tableau HTML
echo "<table>";
while ($row = mysqli_fetch_array($query)) {
    echo "<tr>";
    echo "<td>" . $row['id'] . "</td>";
    echo "<td>" . $row['nom'] . "</td>";
    echo "<td>" . $row['email'] . "</td>";
    echo "<td>" . $row['telephone'] . "</td>";
    echo "</tr>";
}
echo "</table>";
```

apcpedagogie.com

- **App 02**

  - **Énoncé**

    - Soit une base de données **gestion_apprenants** mySQL
    - Cette base est représentée par le schéma suivant:

    - Donner le code PHP pour se connecter à cette base de données en utilisant l'API **mysqli** de PHP avec ses deux styles (Chaque style dans une page comme suit)
      - Une page **connexion_p.php** contenant un script php pour se connecter au base de données avec l'API mysql en utilisant le style procédural
      - Une page **connexion_oo.php** contenant un script php pour se connecter au base de données avec l'API mysql en utilisant le style orienté Objet
    - Créer une page **index.php** contenant deux boutons appelants les deux pages crées.
    - Dans la page **index.php**, créer les boutons suivants sous celles crées précédemment

Num ligne	Type	Nom	couleur	Valeur
2	submit	name="affichage_assoc_p"	warning	Affichage avec mysqli_fetch_assoc_pro
2	submit	name="affichage_assoc_oo"	warning	Affichage avec mysqli_fetch_assoc_OO
2	submit	name="affichage_all_p"	info	Affichage avec mysqli_fetch_all_pro
2	submit	name="affichage_all_oo"	info	Affichage avec mysqli_fetch_all_OO
3	submit	name="affichage_array_p"	success	Affichage avec mysqli_fetch_array_pro

apcpedagogie.com

Num ligne	Type	Nom	couleur	Valeur
3	submit	name="affichage_array_oo"	success	Affichage avec mysqli_fetch_array_OO ▪ Le résultat souhaité sera

- **Solution**
  - Essayez de faire l'exercice de votre côté avant de regarder la Solution !

Cours PHP

Source : https://apcpedagogie.com/acceder-aux-bases-de-donnees-avec-php · téléchargé le 07/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com