

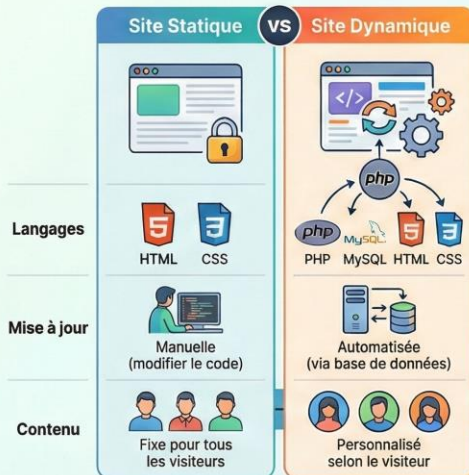
Débuter en PHP : Votre Guide pour Créer des Sites Web Dynamiques

Introduire les concepts de base du développement de sites web dynamiques avec PHP, de la configuration de l'environnement local aux premières lignes de code.

Étape 1 : Comprendre les Concepts Clés

Qu'est-ce que le PHP ?

Un langage exécuté par le serveur pour générer des pages HTML dynamiques et interactives.



Étape 2 : Préparer Votre Environnement Local

Installez un serveur local avec Laragon



Laragon

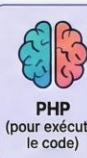
Un outil tout-en-un qui simplifie l'installation d'Apache, PHP et MySQL sur votre ordinateur.

Les 3 composants essentiels inclus



Apache

(serveur web)



PHP

(pour exécuter le code)



MySQL

(pour les bases de données)

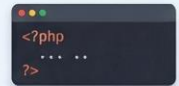
Accès et Fichiers



Vos projets se trouvent dans C:\Laragon\www et sont accessibles via "localhost" dans votre navigateur.

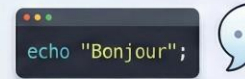
Étape 3 : Écrire Vos Premières Lignes de PHP

Syntaxe de base



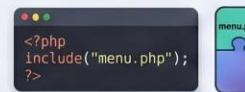
Le code PHP est placé entre les balises `<?php` et `?>`. Chaque instruction se termine par un point-virgule (;).

Afficher du texte avec echo



Utilisez `echo "Bonjour";` pour afficher du contenu directement sur la page web.

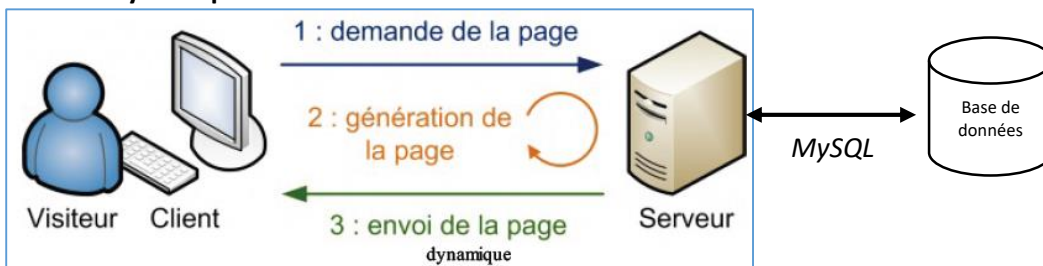
Éviter la répétition avec include



Intégrez des éléments récurrents (menu, pied de page) avec `<?php include("menu.php"); ?>`.

NotebookLM

Les sites dynamiques :



Le **PHP** est un langage exécuté par le serveur.

Il permet de personnaliser la page en fonction du visiteur et/ou d'un contenu dans une base de données, d'effectuer des calculs, etc. Il génère une page HTML.

Les Variables en PHP

En PHP, une **variable** est une information temporaire stockée en mémoire, identifiée par un nom commençant par le symbole '\$'. Elle permet de conserver des données (texte, nombres, etc.) le temps qu'une page web se génère, puis est supprimée.

Les Types de Données Fondamentaux



string (Chaîne de caractères)
Pour stocker du texte, en utilisant des guillemets simples ou doubles.

```
$nom = "Toto";
```



int & float (Nombres)
int pour les nombres entiers, float pour les décimaux (avec un point).

```
$age = 17;  
$poids = 57.3;
```



bool (Booléen)
Ne peut avoir que deux valeurs : true (vrai) ou false (faux).

```
$ext_majour = true;
```

Les Tableaux (Arrays) : Organiser les Données



Tableau Scalaire (Index Numérique)
Chaque valeur est identifiée par un indice numérique, qui commence à 0.

```
$prenoms = ['Mathilde', 'Pierre'];  
echo $prenoms[0]; // Affiche Mathilde
```



Tableau Associatif (Clé Textuelle)
Chaque valeur est associée à une clé personnalisée, comme une étiquette.

```
$ages = ['Pierre' => 29];  
echo $ages['Pierre']; // Affiche 29
```



Tableau Multidimensionnel
Un tableau qui contient d'autres tableaux pour structurer des données complexes.

```
$user = ['nom' => 'Tom',  
        'mail' => 'tom@email.com'];
```

Utilisation et Opérations



Afficher une variable avec echo
La commande echo permet d'afficher la valeur d'une variable dans la page.

```
echo "Le visiteur a $age ans";
```



Opérations Mathématiques
Utilisez les opérateurs +, -, *, / pour les calculs de base.

```
$resultat = (10 + 5) * 2;  
// $resultat vaut 30
```



Le Modulo %
Calcule le reste d'une division entière. Très utile pour les calculs pairs/impairs.

```
$reste = 10 % 3;  
// $reste vaut 1
```

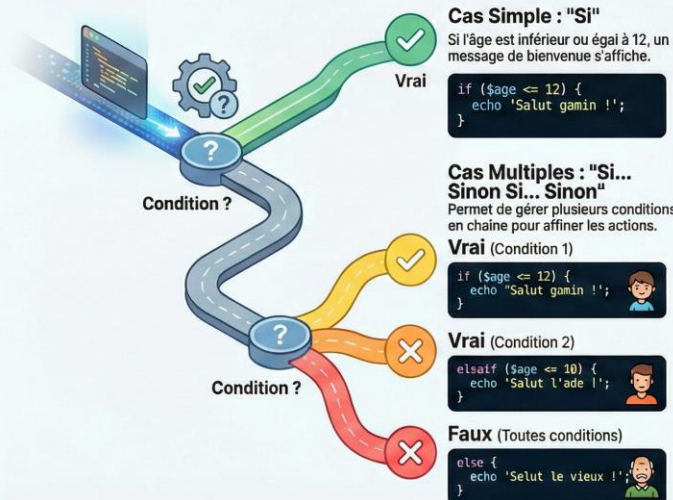
NotebookLM

Programmation : Maîtriser les Conditions Logiques

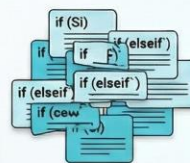
La Structure "Si... Alors... Sinon"

Le Test Logique Fondamental

Vérifie si une condition est vraie ou fausse pour exécuter une action spécifique.

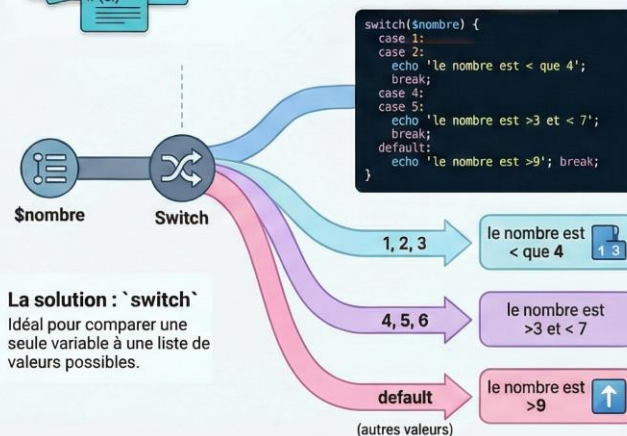


La Structure "Selon... Cas Où" (Switch)



Un problème de lisibilité

Une longue série de `elseif` peut rendre le code lourd et difficile à lire.



Conditionnelles

```
if ( )
{

} else {

}
```

```
if ( )
{

} elseif ( )
{

} else {

}
```

```
switch ( )
{
    case 1 :
    case 2 :
    case 3 : ...
        break;
    default : ...
}
```

Les répétitives

Les Boucles en PHP : while, for & foreach

Les boucles (ou structures répétitives) sont essentielles pour exécuter un bloc d'instructions plusieurs fois en PHP.
Voici les trois boucles fondamentales adaptées à des besoins spécifiques.

La Boucle while

Répète tant qu'une condition est vraie.

Idéale lorsque le nombre d'itérations n'est pas connu à l'avance.



Syntaxe de base

Répète tant qu'une condition.

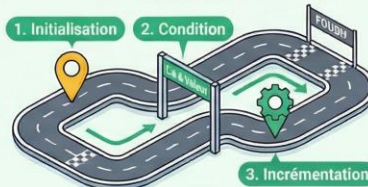
```
while ($chiffre <= 10) {  
    ...  
    $chiffre++;  
}
```

Exemple : compter de 1 à 10.

La Boucle for

Répète pour un nombre de tours défini.

Parfaite pour itérer avec un compteur que l'on maîtrise.



Une structure en trois temps

1. Initialisation, 2. Condition, 3. Incréméntation.

Syntaxe de base

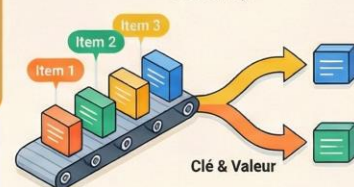
```
for ($i = 1; $i <= 100; $i++) {  
    ...  
}
```

Exemple : afficher 100 lignes.

La Boucle foreach

La spécialiste pour parcourir les tableaux.

Simplifie la lecture de chaque élément d'un array.



Deux façons de l'utiliser

Récupérer juste la valeur **OU** récupérer la clé et la valeur.

Syntaxe de base

```
foreach ($tableau as $cle => $element) {  
    ...  
}
```

NotebookLM

Les fonctions

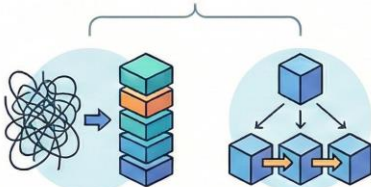
Les Fonctions en PHP : Le Guide Essentiel

Qu'est-ce qu'une fonction PHP ?



Un bloc d'instructions réutilisable

Elle effectue des actions spécifiques et peut retourner une valeur (nombre, texte, etc.).



Les deux avantages principaux

1. Clarifier le code : Rend le programme plus lisible.
2. Réemployer le code : Évite de réécrire les mêmes instructions.

Créer et Utiliser une Fonction

```
function nom(arguments)  
{  
    instructions;  
    return;  
}
```

La Structure de Base (Syntaxe)

Utilisez le mot-clé `function`, donnez un nom, puis définissez les arguments entre parenthèses.

Exemple de Déclaration

```
function VolumeCone($rayon, $hauteur) {  
    $volume = ...;  
    return $volume;  
}
```

Comment l'appeler ?

```
VolumeCone(3, 1);
```

Utilisez simplement son nom avec les valeurs des arguments : `VolumeCone(3, 1);`

Exemples et Bonnes Pratiques



Fonctions intégrées utiles

PHP en possède de nombreuses pour la date (`date('Y')`) ou les textes (`strlen`, `strtoupper`).



Bonne Pratique n°1 : Commentez

Expliquez ce que fait votre fonction, ses arguments et ce qu'elle retourne.

Bonne Pratique n°2 : Organisez

Regroupez vos fonctions un fichier séparé et incluez-le avec l'instruction `'include'`.

NotebookLM

Les formulaires

Formulaires HTML & PHP : De la saisie au traitement des données

Les formulaires HTML collectent les données utilisateur, qui sont transmises à une page PHP pour être traitées.
Le choix de la méthode de transmission, GET ou POST, est crucial pour la sécurité et la capacité.

La méthode POST est préconisée pour les formulaires.
Elle masque les informations et permet d'envoyer des données plus volumineuses.



Méthode GET

Visibilité des données
Dans l'URL (visible)



Taille maximale:
Limitée à 255 caractères



Envoi de fichiers:
Non



Récupération en PHP:
'\$_GET[variable]'



Méthode POST

Visibilité des données
Cachées dans la requête



Taille maximale:
Illimitée (en pratique)



Envoi de fichiers:
Oui



Récupération en PHP:
'\$_POST[variable]'

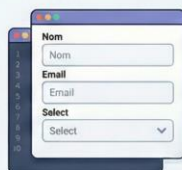


Le processus POST en 3 étapes



1. Construire le formulaire en HTML

Utiliser la balise `<form>` avec `method="post"` et `action="page_de_traitement.php"`.



2. Définir les champs de saisie

Chaque champ (`<input>`, `<select>`) doit avoir un attribut `name` qui deviendra le nom de la variable.



3. Récupérer les données en PHP

Sur la page de traitement, accéder aux valeurs via le tableau `$_POST[nom_du_champ]`.



NotebookLM

Méthode POST

```
<form action="resultat.php" method="post">
Entrez votre prénom : <input type="text" name="prenom" />
<input type="submit" value="valider" />
</form>
```

Récupération de la valeur d'un champ de formulaire :

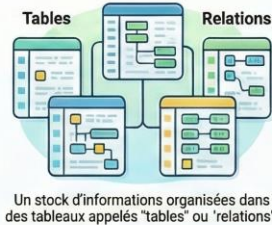
```
$_POST["prenom"]
```

Base de données

Les Bases de Données Relationnelles Simplifiées

Les Concepts Fondamentaux

Qu'est-ce qu'une base de données relationnelle ?



Anatomy d'une table

Elle est composée de colonnes (attributs) et de lignes (enregistrements).

Table : Personnes

ID (Clé Primaire)	Nom	CodePostal (Clé Étrangère)
1	Pierre	1300
2	Jacques	1400
3	Michel	1370

Colonnes (attributs)



Lignes (enregistrements)

Le rôle crucial des clés



Clé primaire
Identifie un enregistrement unique

Clé étrangère
Crée un lien entre les tables

Table : Localités

CP (Clé Primaire)	Nom
1300	Wavre
1400	Nivelle
1370	Limal

L'Interaction et la Gestion



NotebookLM

Les instructions SQL ne sont pas sensibles à la casse

Anatomie d'une Table SQL : Le Guide Essentiel

Cette infographie décompose le processus de création d'une table SQL. Elle présente la syntaxe fondamentale, aide à choisir les types de données appropriés pour chaque colonne, et explique comment établir des relations entre les tables pour garantir l'intégrité des données.

La Commande CREATE TABLE

CREATE TABLE Nom_Table (...)

Options de Colonnes Essentielles



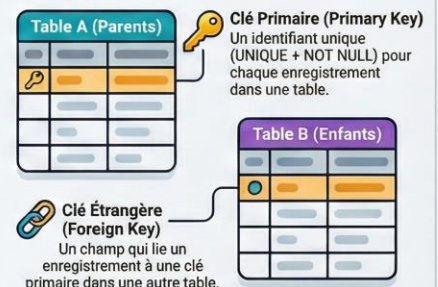
Exemple de Création de Table

```
CREATE TABLE T_Utilisateur  
(ID INT AUTO_INCREMENT PRIMARY KEY,  
Nom VARCHAR(50) NOT NULL,  
Pays VARCHAR(30) DEFAULT 'France')  
;
```

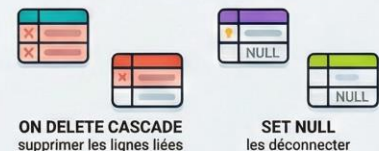
Choisir le Bon Type de Données



Lier les Tables avec des Contraintes



Gérer l'Intégrité Référentielle



NotebookLM

RAPPEL – MEMENTO

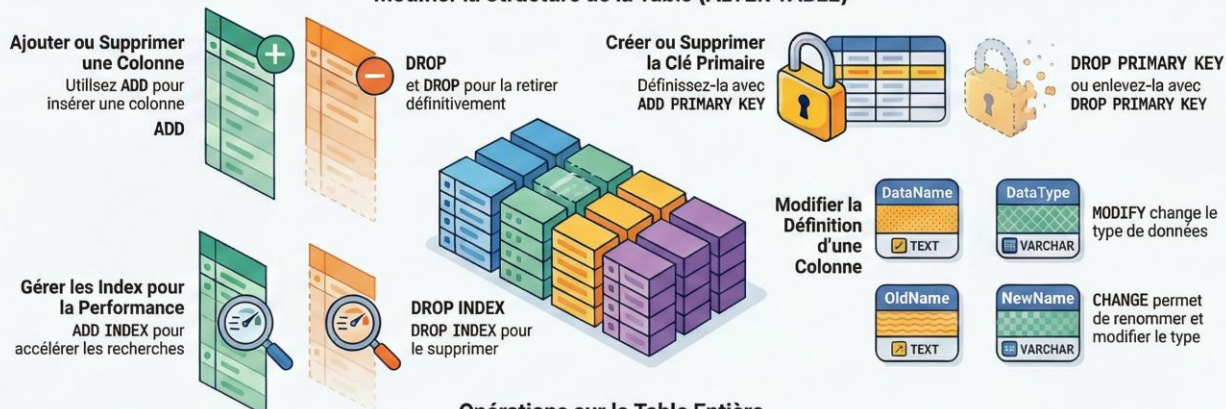
5XBAK - Approche développement backend

Création d'une base de données	CREATE DATABASE
Création d'une table	CREATE TABLE (...) Champs et type de données Entiers INT / Flottants FLOAT / Chaîne VARCHAR TEXT / Date et Heure DATE DATETIME / Ensemble ENUM
Création des relations entre les tables <i>Clé primaire (exemple ISBN, Reg Nat, ADN ...)</i>	PRIMARY KEY
Attribut NOT NULL / UNIQUE Valeur par défaut DEFAULT INDEX → <u>recherches et les tris</u>	
Clé étrangère	CONSTRAINT fk FOREIGN KEY (fkey) REFERENCES TABLE (id)
Mise à jour d'une base de données	ALTER TABLE ... (modification) DROP TABLE ... (suppression)

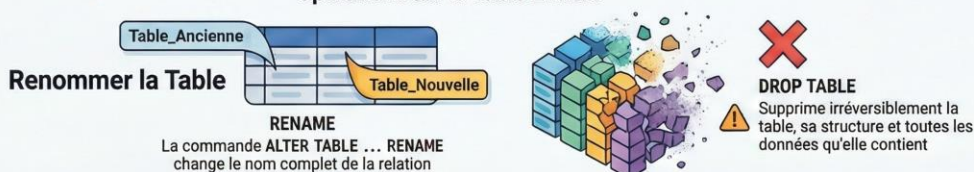
Modifier une Table SQL : Le Guide Rapide

Ce guide visuel présente les principales opérations de modification d'une table SQL avec la commande 'ALTER TABLE', de l'ajout de colonnes à la gestion des clés et des index, jusqu'à la suppression complète.

Modifier la Structure de la Table (ALTER TABLE)



Opérations sur la Table Entière



Gérer les Données en SQL : Les Commandes Essentielles

Ce guide rapide présente les opérations fondamentales pour ajouter, modifier et supprimer des enregistrements dans une base de données : INSERT, UPDATE, et DELETE.

Ajouter des Données (INSERT)



La commande **INSERT INTO** ajoute de nouvelles lignes (enregistrements) dans une table.

```
INSERT INTO Personnes(nom, prenom)
VALUES('Martin', 'Paolo');
```

Spécifiez les colonnes pour éviter les erreurs.



Attention à l'ordre des valeurs !

Sans nom de colonnes, les valeurs doivent correspondre exactement à l'ordre et au type de la table.

Modifier des Données (UPDATE)



La commande **UPDATE** modifie les données des lignes existantes dans une table.

```
UPDATE Personnes SET téléphone = '0156281469'
WHERE id = 102;
```

La clause **WHERE** est cruciale pour ne modifier que les lignes souhaitées.



N'oubliez JAMAIS la clause WHERE !

Sans condition WHERE, TOUS les enregistrements de la table seront modifiés.

Supprimer des Données (DELETE)



La commande **DELETE FROM** efface définitivement des enregistrements d'une table.

```
DELETE FROM Personnes WHERE id = 154;
```

Utilisez un identifiant unique dans la clause WHERE pour une suppression précise.



La suppression est définitive !

Sans clause WHERE, la commande videra entièrement et irréversiblement la table.

NotebookLM

Ajout de données

INSERT INTO TABLE (...) VALUES (...)
(attention à l'ordre des champs)

Modification de données

UPDATE TABLE SET champ= "..." **WHERE** ID =...

Suppression de données

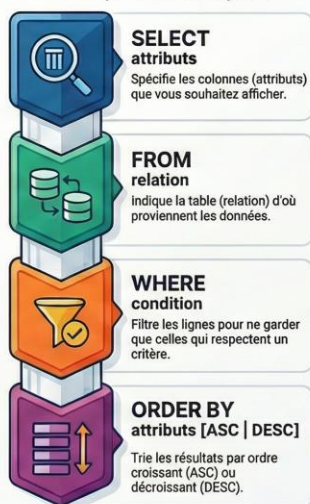
DELETE FROM TABLE WHERE ID =...

Interrogation de base de données

L'Anatomie d'une Requête SQL SELECT

Fournir un guide de référence visuel et rapide sur la structure et les clauses principales utilisées pour interroger une base de données avec la commande SQL SELECT.

Les Briques d'une Requête



Combiner les Données avec les Jointures (JOINS)



INNER JOIN (Jointure Interne)

Retourne uniquement les lignes ayant une correspondance dans les deux tables.



LEFT JOIN (Jointure Externe)

Retourne toutes les lignes de la table de gauche, et les correspondances de la table de droite.

Regrouper et Calculer



GROUP BY attributs

Regroupe les lignes ayant des valeurs identiques pour effectuer un calcul sur chaque groupe.

Fonctions d'Agrégation

S'utilisent avec GROUP BY pour résumer les données.

Fonction	Description
COUNT()	Compte le nombre de lignes.
SUM()	Calcule la somme des valeurs.
AVG()	Calcule la moyenne des valeurs.
MAX() / MIN()	Trouve la valeur maximale / minimale.

NotebookLM

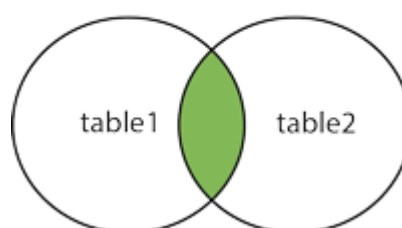
Requêtes de sélection

Simple	SELECT attribut FROM table
Avec filtre	SELECT attribut FROM table WHERE attribut = " ... "
Avec tri	SELECT attribut FROM table ORDER BY attribut ASC
Suppression des doublons	SELECT DISTINCT attribut FROM table
Avec jointure	SELECT attribut FROM table1, table2 WHERE table1.attribut = table2.attribut
Multiple	
Requêtes imbriquées	SELECT * FROM table1 WHERE ID=(SELECT ID FROM table2 WHERE attribut = " ... ")
Requêtes sélection multi-tables	SELECT * FROM table1,table2 WHERE table1.ID = table2.IDREF AND attribut = " ... "
Avec regroupement	SELECT ... FROM table GROUP BY attribut HAVING...
Champs calculés/ Fonctions de MySQL	MIN(x), MAX(x), AVG(x), SUM(x), COUNT(x)...

Les jointures internes

INNER JOIN

INNER JOIN

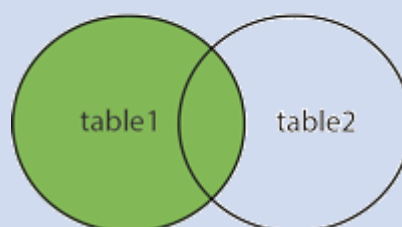


SELECT ...
FROM table1
INNER JOIN table2 **ON** table1.ID = table2.IDREF

Les jointures externes

LEFT OUTER JOIN

LEFT JOIN

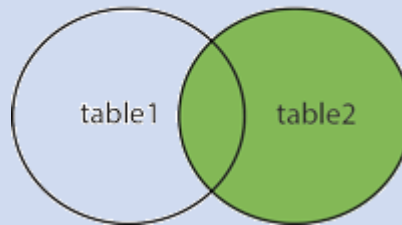


SELECT ...

```
FROM table1
LEFT OUTER JOIN table2 ON table1.ID = table2.IDREF
```

RIGHT OUTER JOIN

RIGHT JOIN



```
SELECT ...
FROM table1
RIGHT OUTER JOIN table2 ON table1.ID = table2.IDREF
```

Requêtes ensemblistes

Union

```
SELECT attribut FROM table1
UNION
SELECT attribut FROM table2
```

Intersection

```
SELECT attribut FROM table1
WHERE attribut IN (SELECT attribut FROM table2)
```

Différence

```
SELECT attribut FROM table1
WHERE attribut NOT IN (SELECT attribut FROM table2)
```

Produit cartésien

```
SELECT * FROM table1,table2
```

PHP & BDD : Maîtriser la Connexion avec PDO et MySQLi



PDO : Le Standard pour l'Accès aux Données
Qu'est-ce que PDO ?

Une classe PHP qui agit comme une interface unifiée pour accéder aux bases de données.



Une Couche d'Abstraction Puissante
Permet de communiquer avec MySQL, Oracle, PostgreSQL, etc., sans changer de code.



Sécurité et Réutilisabilité
Son intérêt majeur est de sécuriser les requêtes grâce aux "requêtes préparées".

En Pratique : Les 4 Étapes Clés avec MySQLi



1. Se Connecter à la Base

Utiliser `mysqli_connect()` avec les informations d'identification du serveur.

2. Exécuter une Requête

Envoyer une commande SQL à la base de données via la méthode `query()`.

3. Traiter les Résultats

Parcourir les lignes de données reçues avec une boucle `while` et `mysqli_fetch_array()`.

4. Manipuler les Données

Utiliser `INSERT`, `UPDATE` ou `DELETE` pour ajouter, modifier ou supprimer des enregistrements.

PDO est une **classe PHP** destinée à permettre à PHP de communiquer avec un serveur de données. (PDO : Php Data Object)

```
<?php

// on se connecte à MySQL et on sélectionne la base
$connection = mysqli_connect('dns', 'utilisateur', 'motdepasse', 'basededonnees'[,
'port']);
?>
```

Pour **sélectionner des enregistrements**, nous utiliserons la méthode `query($requete)`.

```
<?php
// On établit la connexion
Include('connexion.php');

// On envoie la requête
$select = $connection->query("SELECT * FROM t_contact");
?>
```

Affichage (SELECT)

```
<?php
while( $enregistrement=mysqli_fetch_array($select))
{
    // Affichage des champs
    echo '<h1>'. $enregistrement['Nom'] . ' ' . $enregistrement['Prenom'] . '</h1>';
}
?>
```

```
<?php
// Ajout de données
// On crée la requête
$req = "INSERT INTO table(texte) VALUES ('Du texte')";

// on envoie la requête
$res = $conn->query($req);
```

```
// Mise à jour de données
// On crée la requête
$req = "UPDATE FROM table WHERE id=XXX";

// on envoie la requête
$res = $conn->query($req);
```

```
// Suppression de données
// On crée la requête
$req = "DELETE FROM table WHERE id=XXX";

// on envoie la requête
$res = $conn->query($req);
```