Focus sur le langage PHP et ses jointures

Les jointures de BDD

Pourquoi utiliser des jointures ?

Dans votre BDD, vous avez de multiples tables : produits, articles, utilisateurs, commandes, clients…
Il est utile de lier vos informations les unes au autres pour effectuer des recherches ciblées par exemple.

Le but des jointures étant de puiser les données dans chaque table pour avoir des données les plus pertinentes sans altérer les performances.

Le but principal c'est de ne pas avoir 1 table trop surchargée en informations,
Mais plusieurs tables où l'on pourra piocher les informations à l'aide de liens pour optimiser les performances.

A retenir : 1 table de clients/commandes/produits sera moins performante que 3 tables et des jointures : 1 pour les clients, 1 pour les commandes et 1 pour les produits

Les jointures de BDD

Qu’est-ce qu’une jointure ?

Une jointure ou JOIN sert à combiner les données entre plusieurs tables.
C’est le **lien** entre ses tables que l’on appelle jointure.

Il en existe de plusieurs types :

- INNER JOIN
- LEFT or RIGHT or FULL JOIN
- CROSS JOIN

En fonction des besoins, vous utiliserez des jointures différentes.

On utilise les jointures exactement comme un SELECT classique.

INNER JOIN

Jointure interne pour retourner les enregistrements quand la condition est vrai dans les 2 tables.
C'est l'une des jointures les plus communes.

On peut l'écrire de deux manières :

```
SELECT *
FROM table1
INNER JOIN table2 ON table1.id = table2.fk_id
```

```
SELECT *
FROM table1
INNER JOIN table2
WHERE table1.id = table2.fk_id
```

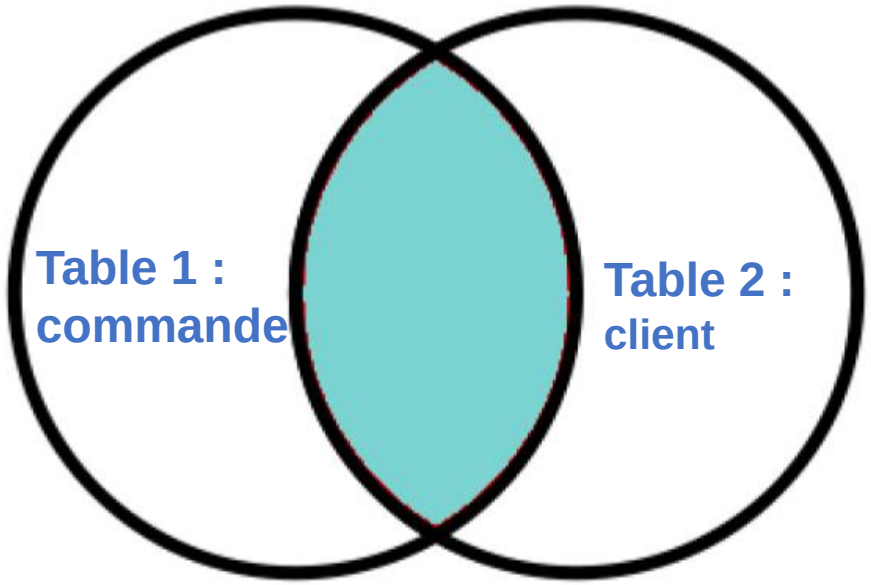


Table 1 :
renseigne les informations et détails des commandes avec un ID_COMMANDE

Table 2 :
renseigne les informations des clients et référence des commandes effectuées avec leur ID_CLIENT et leur ID_COMMANDE

SELECT * FROM client INNER JOIN commande ON client.id_commande = commande.id_commande

LEFT JOIN

LEFT JOIN (aussi appelée LEFT OUTER JOIN) est un type de jointure entre 2 tables. Cela permet de lister tous les résultats de la table de gauche (left = gauche) même s'il n'y a pas de correspondance dans la deuxième table.

On peut l'écrire de deux manières :

```
SELECT *
FROM table1
LEFT JOIN table2 ON table1.id = table2.fk_id
```

```
SELECT *
FROM table1
LEFT OUTER JOIN table2 ON table1.id = table2.fk_id
```

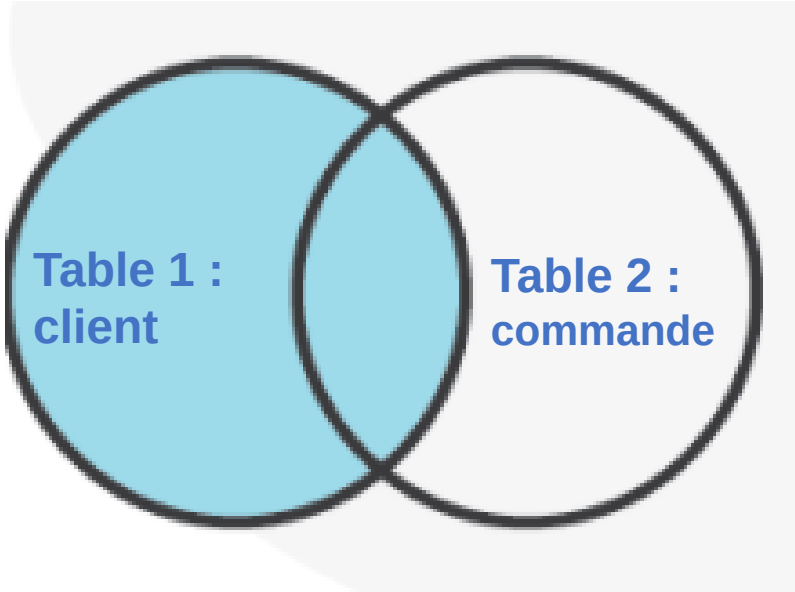


Table 1 :
renseigne les informations des clients et référence des commandes effectuées avec leur ID_CLIENT

Table 2 :
renseigne les informations et détails des commandes avec un ID_CLIENT de renseigné pour chaque commande

SELECT * FROM client LEFT JOIN commande ON client.id_client = commande.id_client

Pour lister tous les clients avec leurs commandes et afficher également les clients qui n'ont pas effectuées de commandes (la donnée sera NULL dans ce cas).

RIGHT JOIN

RIGHT JOIN (aussi appelée RIGHT OUTER JOIN) est un type de jointure entre 2 tables. Cela permet de lister tous les résultats de la table de droite (right = droite) même s'il n'y a pas de correspondance dans la deuxième table.

On peut l'écrire de deux manières :

```
SELECT *
FROM table1
RIGHT JOIN table2 ON table1.id = table2.fk_id
```

```
SELECT *
FROM table1
RIGHT OUTER JOIN table2 ON table1.id = table2.fk_id
```

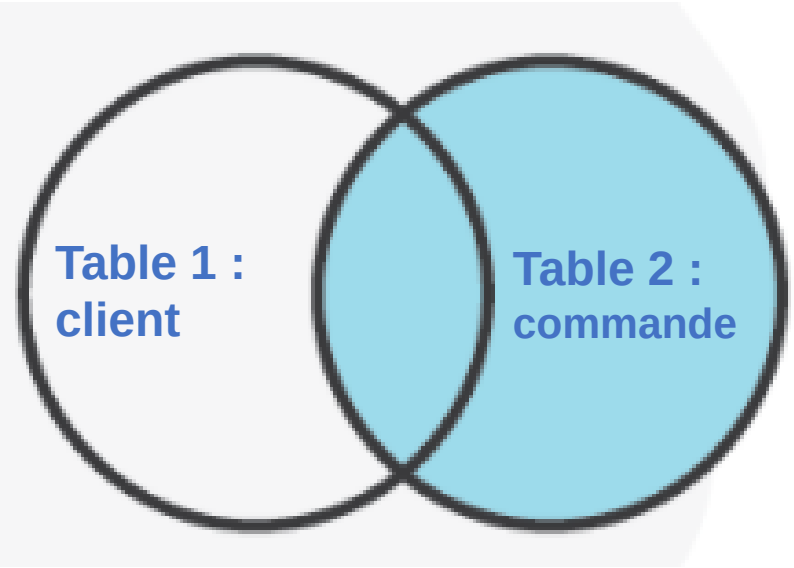


Table 1 :
renseigne les informations des clients et référence des commandes effectuées avec leur ID_CLIENT

Table 2 :
renseigne les informations et détails des commandes avec un ID_CLIENT de renseigné pour chaque commande

SELECT * FROM client RIGHT JOIN commande ON client.id_client = commande.id_client

Pour lister toutes les commandes avec les informations des clients, avec cette requête on peut voir du coup si une commande n'est pas liée à un client (supprimé de la BDD par exemple, ou archivés…).

FULL JOIN

FULL OUTER JOIN est un type de jointures externes qui va récupérer toutes les données pour les colonnes sélectionnées de chacune des deux tables.

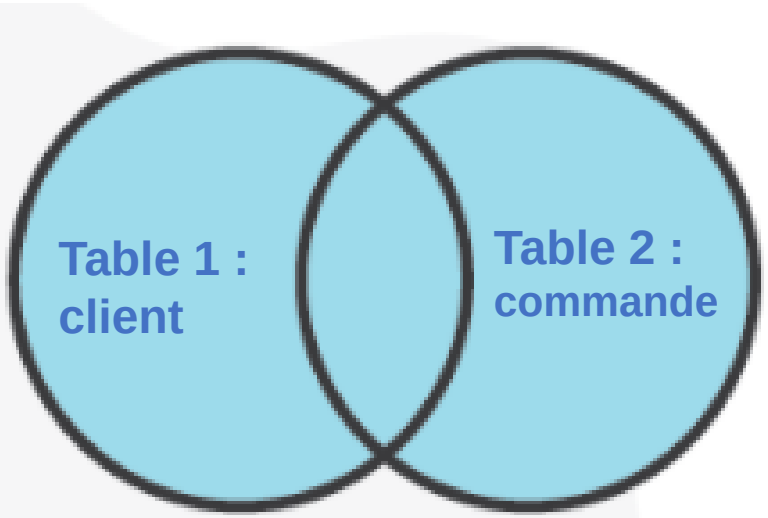
Ainsi, elle permet de combiner les résultats des 2 tables, les associer entre elles grâce à une condition et remplir avec des valeurs NULL si la condition n'est pas respectée.

Attention cependant : ce type de jointures n'est pas supporté en MySQL.

<u>On peut l'écrire de deux manières</u> :

```
SELECT *
FROM table1
FULL JOIN table2 ON table1.id = table2.fk_id
```

```
SELECT *
FROM table1
FULL OUTER JOIN table2 ON table1.id = table2.fk_id
```



SELECT * FROM client FULL JOIN commande ON client.id_client = commande.id_client

Pour lister toutes les données des 2 tables, on aura l'ensemble des clients et des commandes. Si une commande est liée à un client alors les lignes seront combinées, si une commande n'est pas liée à un client nous aurons des données à NULL pour le client, et vis versa si un client n'a effectué aucune commande alors nous aurons des données NULL pour les commandes associées au client.

CROSS JOIN

Jointure retournant une liste de résultat qui va être le produit des entrées des deux tables jointes lorsqu'aucune clause WHERE n'est utilisée.

Attention, le nombre de résultat peut facilement être très élevé sans clause WHERE. S'il est effectué sur des tables avec beaucoup d'enregistrements, cela peut ralentir sensiblement le serveur. Cette jointure est rarement utilisée.

On peut l'écrire de deux manières :

```
SELECT *
FROM table1
CROSS JOIN table2
```

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

Table 1 : 5 enregistrements
Commandes_informatiques

Table 2 : 10 enregistrements
Commandes_téléphones

SELECT * FROM Commandes_informatiques
CROSS JOIN Commandes_téléphones;

Vous aurez les enregistrements des 2 tables comme suit :
5 x 10 !!!

INNER JOIN

TABLE

COLONNE

```php
<?php

$req = $pdo->prepare('SELECT proprio.nom, proprio.prenom, voiture.marque, voiture.modele
                      FROM proprio
                      INNER JOIN voiture ON proprio.id = voiture.ID_proprio');

$req->execute();

$res = $req->fetchALL();

echo '<pre>';
print_r($res);
echo '</pre>';

?>
```

JOINTURE

INSTRUCTION D'EXECUTION DE LA REQUETE

A la différence du fetch qui retourne 1 enregistrement, le fetchall retourne un tableau contenant toutes les lignes du jeu d'enregistrements du SELECT avec ses jointures.

RIGHT JOIN

```php
<?php

$req = $pdo->prepare('SELECT proprio.nom, proprio.prenom, voiture.marque, voiture.modele
                      FROM proprio RIGHT JOIN voiture ON proprio.id = voiture.ID_proprio');

$req->execute();

$res = $req->fetchALL();

echo '<pre>';
print_r($res);
echo '</pre>';

?>
```

CROSS JOIN

```php
<?php

$req = $pdo->prepare('SELECT proprio.nom, proprio.prenom, voiture.marque, voiture.modele
                      FROM proprio
                      CROSS JOIN voiture');

$req->execute();

$res = $req->fetchALL();

echo '<pre>';
print_r($res);
echo '</pre>';

?>
```