Les jointures en MySQL - Suite

L'ÉCOLE ATYPIQUE

Les jointures en MySQL

Pourquoi utiliser des jointures ?

Objectif : extraire des données de deux ou de plusieurs tables en fonction des relations logiques existant entre ces tables.

=> Les jointures permettent de combiner des données de deux tables ou plus, créant un ensemble de résultats qui inclut des informations des deux sources.

L'ÉCOLE ATYPIQUE

Clé étrangère

Une **clé étrangère** est une colonne (ou un ensemble de colonnes) dans une table qui est utilisée pour établir et appliquer une relation de "clé étrangère" entre cette table et une autre table.

Etudiant :

id	nom	ville
1	Alice	Poitiers
2	Bob	Paris
3	Martin	Lille

Note :

id	valeur	etudiant_id
1	13	2
2	17	1
3	8	2

L'ÉCOLE ATYPIQUE

Les différentes jointures

LEFT JOIN

Everything on the left
+
anything on the right that matches

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI LEFT JOIN

Everything on the left
that is NOT on the right

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_2.KEY IS NULL
```

RIGHT JOIN

Everything on the right
+
anything on the left that matches

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI RIGHT JOIN

Everything on the right
that is NOT on the left

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
```

OUTER JOIN

Everything on the right
+
Everything on the left

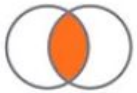
```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI OUTER JOIN

Everything on the left and right
that is unique to each side

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
OR TABLE_2.KEY IS NULL
```

INNER JOIN



Only the things that match on the left AND the right

```
SELECT *
FROM TABLE_1
INNER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

CROSS JOIN

All combination of rows from the right and the left (cartesean product)

```
SELECT *
FROM TABLE_1
CROSS JOIN TABLE_2
```

L'ÉCOLE ATYPIQUE

Tables “Etudiant” et “Note”

Etudiant :

id	nom	ville
1	Alice	Poitiers
2	Bob	Paris
3	Martin	Lille

Note :

id	valeur	etudiant_id
1	13	2
2	17	1

L'ÉCOLE ATYPIQUE

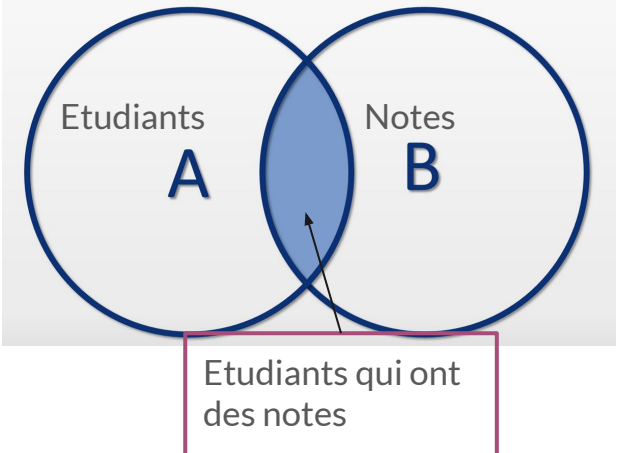
INNER JOIN

Utilisé lorsque vous souhaitez obtenir uniquement les enregistrements ayant une correspondance dans les deux tables.

Si un enregistrement d'une table n'a pas de correspondant dans l'autre table, il ne sera pas inclus dans les résultats.

Exemple : récupérer uniquement les étudiants qui ont des notes.

Si des étudiants n'ont pas de note, ils n'apparaitront pas.



L'ÉCOLE ATYPIQUE

INNER JOIN

```
SELECT etudiant.nom, note.valeur
FROM etudiant
INNER JOIN note ON etudiant.id = note.etudiant_id;
```

Cette requête relie la table `etudiant` à la table `note` via le champ `etudiant_id` dans `note`, qui correspond au champ `id` dans `etudiant`.

Elle retourne les noms des étudiants ainsi que les valeurs de leurs notes, mais uniquement pour les étudiants qui ont des notes enregistrées.

Résultat :

id	nom	note
1	Alice	18
1	Alice	17
2	Bob	15

L'ÉCOLE ATYPIQUE

LEFT JOIN

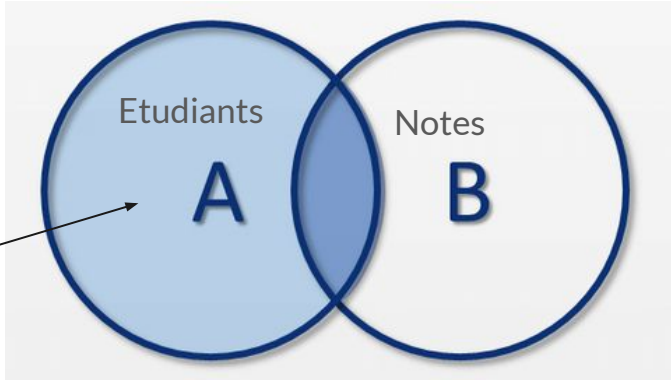
Utilisé pour sélectionner tous les enregistrements d'une table principale (gauche ou droite) et les enregistrements correspondants de l'autre table.

Remarque : Si un enregistrement de la table principale n'a pas de correspondant, le résultat inclura ce enregistrement avec des valeurs NULL pour les colonnes de l'autre table.

Exemple : Récupérer tous les étudiants, qu'ils aient des notes ou non.

Tous les étudiants qu'ils aient des notes ou pas

S'ils ont des notes on verra la note affichée.
Sinon NULL sera affiché



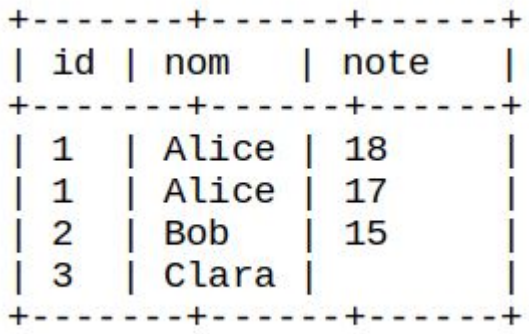
L'ÉCOLE ATYPIQUE

LEFT JOIN

```
SELECT etudiant.nom, note.valeur
FROM etudiant
LEFT JOIN note ON etudiant.id = note.etudiant_id;
```

Dans cet exemple, tous les étudiants seront listés, qu'ils aient ou non des notes attribuées. Les étudiants sans notes auront une valeur `NULL` pour le champ `valeur` de la note.

Résultat :

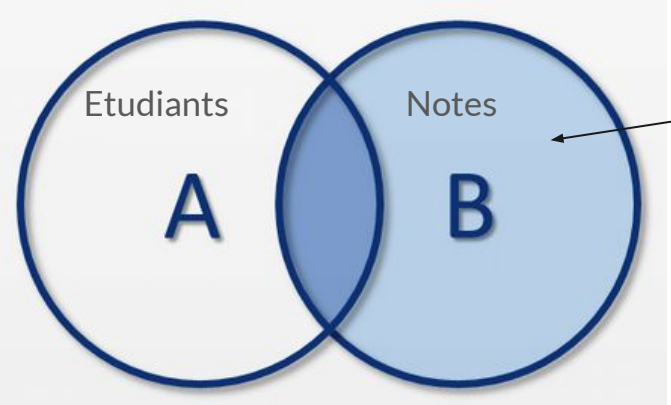


id	nom	note
1	Alice	18
1	Alice	17
2	Bob	15
3	Clara	

RIGHT JOIN

Même principe que le LEFT JOIN mais pas avec les mêmes tables

Exemple : Récupérer toutes les notes, qu'elles soient rattachées ou non à des étudiants.



L'ÉCOLE ATYPIQUE

RIGHT JOIN

```
SELECT etudiant.nom, note.valeur
FROM note
RIGHT JOIN etudiant ON note.etudiant_id = etudiant.id;
```

Toutes les notes, y compris celles qui ne sont pas associées à un étudiant dans la table `etudiant`. Si une note n'est pas associée à un étudiant, le nom de l'étudiant sera affiché comme `NULL`.

Résultat :

```
+----+-------+------+
| id | nom   | note |
+----+-------+------+
| 1  | Alice | 18   |
| 1  | Alice | 17   |
| 2  | Bob   | 15   |
+----+-------+------+
```

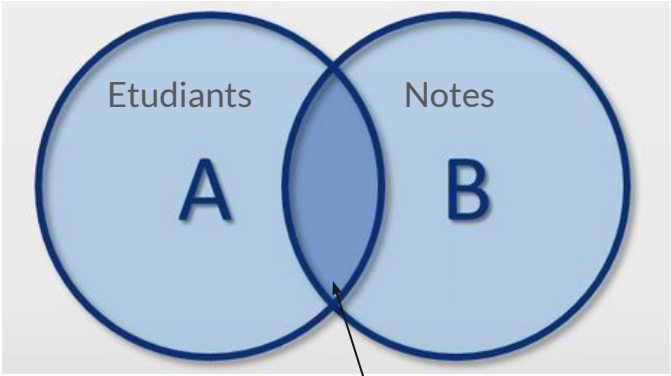
Remarque : dans cet exemple le résultat est le même que celui du INNER JOIN car il ne peut pas y avoir de Note sans étudiant.

L'ÉCOLE ATYPIQUE

FULL JOIN

Combinaison des LEFT et RIGHT JOIN, récupérant tous les enregistrements quand il y a une correspondance dans au moins une des tables.

Exemple : Récupérer toutes les notes et tous les étudiants



QCM

https://aymeric-auberton.fr/academie/mysql/quiz