

SQL – MAJ

Benjamin bailly

Sql – Consultation

- SELECT

SELECT

Pour selectionner tout les attributs d'une table :

`SELECT * FROM table.`

`SELECT nom_attr1, nom_attr2, nom_attr3 FROM table.`

	id_client	prenom	nom	mail
▶	2	Juste	Leblanc	justeleblanc@adrar-formation.com
	3	François	Pignon	françoisignon@adrar-formation.com
	4	Barbara	Stresande	barbarastresand@adrar-formation.com
	5	Jean	Valgean	jeanvalgean@adrar-formation.com
	6	Baptiste	Valgean	valgeanbaptiste@adrar-formation.com
	7	l'asticot	toto	totoasticot@adrar-formation.com
*	NULL	NULL	NULL	NULL

Sql – Consultation

– SELECT

DISTINCT

La commande SELECT peut potentiellement afficher des enregistrements en double.
La commande DISTINCT permet d'éviter des redondances dans les résultats.

Pour que ce soit parlant reprenons la table client, on a pu le voir sur la slide d'avant, j'ai deux personnes nommés Valgean dans ma bdd. Si je fais un select distinct j'aurai cette affichage.

```
USE magasin_v2;  
SELECT distinct nom from client
```

	nom
►	Leblanc
	Pignon
	Stresande
	Valgean
	toto

ex : sans distinct

	nom
►	Leblanc
	Pignon
	Stresande
	Valgean
	Valgean
	toto

AS

Cela permet de faciliter grandement l'écriture et la lecture des requêtes, exemple de nom de table sur un ERP :



The diagram shows a table with two columns. The first column contains the word 'societe' and the second column contains the word 'nom'. A red circle is drawn around the word 'societe' in the first column.

```
SELECT fk_soc as societ  FROM llx_facture as facture;
```

Essayez sur la table de votre choix.

[illegible]

Sql – Consultation

– SELECT

WHERE

La commande WHERE permet d'extraire des enregistrements d'une base de données qui respectent une condition.

Cela permet d'obtenir uniquement les informations désirées.

```
SELECT nom_attribut1, nom_attribut2  
FROM table1  
WHERE condition;
```

Les opérateurs de comparaison :

Opérateur	Description
=	Egale
<>	Pas égale
!=	Pas égale
>	Supérieur à
<	Inférieur à
>=	Supérieur ou égal à
<=	Inférieur ou égal à
IN	Liste de plusieurs valeurs possibles
BETWEEN AND	Valeur comprise dans un intervalle de données
LIKE	Recherche en spécifiant le début, le milieu ou la fin d'un mot
IS NULL	Valeur est nulle
IS NOT NULL	Valeur n'est pas nulle

Sql – Consultation

– SELECT

WHERE AND OR

Le WHERE va être un outil puissant qui va permettre vraiment de cibler nos recherches sur n'importe quel critère souhaiter.

Exemple:

Use magasin_V2;

*SELECT **

FROM client WHERE nom between "A" AND "P";

	id_client	prenom	nom	mail
▶	2	Juste	Leblanc	justeleblanc@adrar-formation.com
✱	NULL	NULL	NULL	NULL

Il est possible de combiner autant de conditions qu'on veut.

Use magasin_V2;

*SELECT * FROM client WHERE nom between "A" AND "Z" AND (id_client < 3 OR id_client = 4);*

	id_client	prenom	nom	mail
▶	2	Juste	Leblanc	justeleblanc@adrar-formation.com
	4	Barbara	Stresande	barbarastresand@adrar-formation.com
✱	NULL	NULL	NULL	NULL

Sql – Consultation

– SELECT

ORDER BY

La commande ORDER BY permet de trier des lignes dans un résultat d'une requête.
Il est possible de trier les données sur un ou plusieurs attributs par ordre ascendant ou descendant.

Exemple :

Use magasin_V2;
SELECT * from CLIENT ORDER BY nom ;

	id_client	prenom	nom	mail
▶	2	Juste	Leblanc	justeleblanc@adrar-formation.com
	3	François	Pignon	françoisignon@adrar-formation.com
	4	Barbara	Stresande	barbarastresand@adrar-formation.com
	7	l'asticot	toto	totoasticot@adrar-formation.com
	5	Jean	Valgean	jeanvalgean@adrar-formation.com
	6	Baptiste	Valgean	valgeanbaptiste@adrar-formation.com
*	NULL	NULL	NULL	NULL

Par défaut, les résultats sont classés par ordre ascendant.

Pour trier le résultat par ordre décroissant nous utilisons le suffixe DESC

SELECT * from CLIENT ORDER BY nom DESC

	id_client	prenom	nom	mail
▶	5	Jean	Valgean	jeanvalgean@adrar-formation.com
	6	Baptiste	Valgean	valgeanbaptiste@adrar-formation.com
	7	l'asticot	toto	totoasticot@adrar-formation.com
	4	Barbara	Stresande	barbarastresand@adrar-formation.com
	3	François	Pignon	françoisignon@adrar-formation.com
	2	Juste	Leblanc	justeleblanc@adrar-formation.com
*	NULL	NULL	NULL	NULL

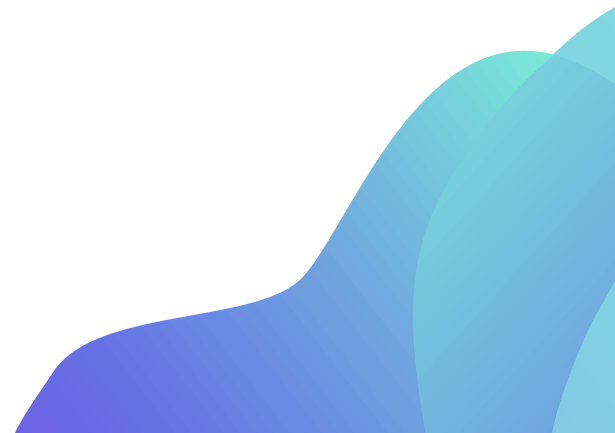
Sql – Consultation

– Jointure

JOINTURES :

Les jointures permettent d'associer plusieurs tables dans une requête par la clé étrangère.

Types de jointure :

- INNER JOIN
 - LEFT JOIN
 - RIGHT JOIN
 - FULL JOIN
- 

Sql – Consultation

– INNER JOIN

INNER JOIN

Cette commande retourne les enregistrements quand la condition est vraie dans les deux tables.

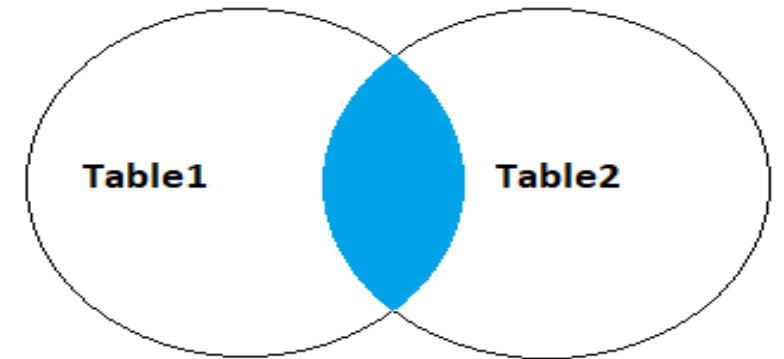
Elle retournera tous les enregistrements qui se trouvent dans la table 1 et qui ont forcément un lien (clé étrangère not null) avec la table 2.

Ex :

```
USE magasin_v2;
```

```
SELECT * from jeuxvideo
```

```
INNER JOIN jouer ON jeuxvideo.id_jeuxvideo = jouer.id_jeuxvideo
```



Sql – Consultation

– LEFT JOIN

LEFT JOIN

C'est une jointure externe gauche.

Cette commande retourne tous les enregistrements de la table 1 même si la condition n'est pas vérifiée dans l'autre table.

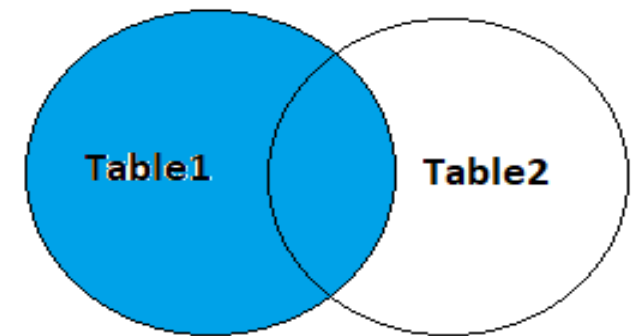
Elle retournera tous les enregistrements de la table 1 qu'ils aient un lien ou non avec la table 2.

Ex :

```
USE magasin_v2;
```

```
SELECT * from jeuxvideo
```

```
LEFT JOIN jouer ON jeuxvideo.id_jeuxvideo = jouer.id_jeuxvideo
```



Sql – Consultation

– RIGHT JOIN

RIGHT JOIN

C'est une jointure externe droite.

Cette commande retourne tous les enregistrements de la table 2 même si la condition n'est pas vérifiée dans l'autre table.

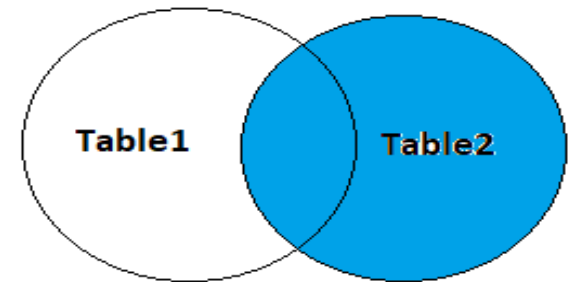
Elle retournera tous les enregistrements de la table 2 qu'ils aient un lien ou non avec la table 1.

Ex :

```
USE magasin_v2;
```

```
SELECT * from stock
```

```
RIGHT JOIN jeuxvideo ON jeuxvideo.id_jeuxvideo = stock.id_jeuxvideo
```



Sql – Consultation – Requête imbriqué !

Admettons que vous voulez savoir combien il y a de stock de jeu diablo 2, nous n'allons pas à chaque requête aller en base de données vérifier l'id des jeux afin d'aller chercher dans la table stock.

Nous allons désormais utiliser les requêtes imbriquées.

Ex :

```
USE magasin_v2;
```

```
SELECT * from stock
```

```
WHERE id_jeuxvideo = (SELECT id_jeuxvideo from jeuxvideo WHERE nom = "diablo2")
```

Sql – Consultation – Fonctions d'agrégation

Les fonctions d'agrégations permettent d'effectuer des stats/calculs sur le résultat de vos requêtes.

COUNT – Pour compter le nombre d'enregistrement

SUM – Pour calculer la somme sur un ensemble d'enregistrement

AVG – Pour faire la moyenne sur un ensemble d'enregistrement

MIN – Pour récupérer la valeur minimum sur un ensemble d'enregistrement

MAX – Pour récupérer la valeur max sur un ensemble d'enregistrement



Sql – Consultation

– GROUP BY

GROUP BY est utilisé pour regrouper des résultats suite à un calcul.

Voici un exemple pour afficher le solde de compte bancaire à l'aide d'une table regroupant toute les transactions.

Ex :

```
USE dolibarr_mcneil;  
SELECT round(sum(amount)) as total, fk_account as account  
FROM llx_bank as bank  
GROUP BY account
```

	total	account
▶	0	1
	17169	2
	0	3
	363	4
	129159	5

Sql – Consultation

– having

Having va fonctionner un peu comme le **where** mais va permettre de mettre une restriction après le calcul fait par la fonction d'agregation.

Ex :

```
USE dolibarr_mcneil;  
SELECT round(sum(amount)) as total, fk_account as account  
FROM llx_bank as bank  
GROUP BY account  
HAVING total > 10000
```

	total	account
▶	17169	2
	129159	5

Merci

