13) Modification d'un enregistrement

Pour modifier des données, la requête est relativement simple c'est avec les mots clé **UPDATE… SET** :

- **UPDATE [nom table] SET champ=valeur, champ=valeur WHERE [conditions]**

On va modifier le nom et prénom d'un enregistrement.

On peut modifier le prénom du 3ème utilisateur :

```
- UPDATE users SET firstname = "Martha" WHERE id_user = 3;
```

On peut s'amuser, par exemple si on veut modifier la ville de tous les hommes on pourrait faire :

- **UPDATE users SET city= "Anvers" WHERE gender = "M"**

Exos

1) **Créez CET enregistrement (Beyonce…).**

8	Cool	Blues	X	2000-02-24	Anvers
9	Beyonce	Knowles	F	1980-11-20	Kinshasa

2) **Ensuite modifier sa date de naissance (4 Septembre 1981) et sa ville (Houston).**

L'UPDATE, comme toutes les autres commandes, est irréversible donc c'est bien de mettre des **LIMIT** si vous voulez être sûr de votre coup.

14) Sélectionner des données

Maintenant que l'on sait comment manipuler des données dans les tables, on va passer à la sélection de données. C'est une opération que vous allez rencontrer le plus souvent lorsque vous allez ensuite développer votre site web. En effet lorsque vous allez vouloir créer un site, il va vous falloir créer les requêtes qui vont vous permettre de récupérer les articles par exemple ou encore créer une requête pour récupérer une catégorie etc. Vous allez en gros devoir utiliser beaucoup de requêtes de récupération de données.

Pour récupérer les données la requête est relativement simple, on fait **SELECT** suivi des champs à récupérer, dans notre cas, dans la table utilisateurs, on voudra récupérer que les prénoms et les noms puis on précise qu'ils viennent de la table users :

- **SELECT firstname, lastname FROM users**

Dans ma table cela donne ce résultat :

La syntaxe c'est donc :

- **SELECT champ1, champ2,… FROM [nom table] WHERE [conditions]**

A noter que pour sélectionner tous les éléments d'une table il suffit de mettre à la suite du SELECT une ***** :

- **SELECT * FROM users**

Exos

1) **Récupérer le nom et prénom de l'utilisateur qui a comme ID 1.**
2) **Récupérer la ville de toutes les femmes.**
3) **Récupérer la date de naissance de Beyonce.**

Pagination

Pour ce qui est de l'ajout d'une limite, on peut choisir l'endroit où la limite commence.

```
SELECT * FROM `users` LIMIT 1, 2
```

Options

	id_user	firstname	lastname	sexe	date_of_birth	city
Éditer Copier Supprimer	2	Jack	Bauer	M	2004-12-24	New-York
Éditer Copier Supprimer	3	Lara	Croft	F	1997-08-01	Washington

Vous voyez j'ai fait un **SELECT ALL (*)** mais au lieu de m'afficher tous les enregistrements, il m'a juste affiché ce que je demandais dans **LIMIT** donc de 1 à 2 (donc le 2ème et 3ème enregistrement). Ça se lit, je veux 2 enregistrements à partir du 1. Si on avait mis LIMIT 0, 2 ça aurait pris le premier et deuxième enregistrement. Un peu comme les tableaux et listes dans quasi tous les langages de programmation, ça commence à 0.

C'est grâce à ça qu'on peut faire des paginations, par exemple sur une table de 100 enregistrements et qu'on veut les montrer 10 par 10, on pourra faire à chaque fois LIMIT 0, 10, puis pour la page 2 : LIMIT 10, 10 puis LIMIT 20, 10 puis LIMIT 30, 10 etc.

Exos

1) **Affichez 2 enregistrements à partir du 3ème. Montrez-moi une fois finit.**

Ordre d'organisation des données

Pour pouvoir organiser vos enregistrements vous pouvez utiliser le mot clé **ORDER BY**. Si vous voulez afficher tous les utilisateurs en ordonnant par le nom de manière donc **alphabétique (croissant).** Il suffit de faire :

```
SELECT * FROM `users` ORDER BY lastname
```

Tout afficher | Nombre de lignes : 25 Filtrer les lignes: Chercher dans cette table Trier par clé :

+ Options

	id_user	firstname	lastname	sexe	date_of_birth	city
Éditer Copier Supprimer	2	Jack	Bauer	M	2004-12-24	New-York
Éditer Copier Supprimer	8	Cool	Blues	X	2000-02-24	Anvers
Éditer Copier Supprimer	1	James	Bond	M	2007-07-07	London
Éditer Copier Supprimer	3	Lara	Croft	F	1997-08-01	Washington
Éditer Copier Supprimer	9	Beyonce	Knowles	F	1981-09-04	Houston

Ici on voit bien qu'il a bien récupéré les enregistrements par ordre croissant (alphabétique).

Pour avoir par ordre **décroissant** (du plus grand ou plus petit), il faut juste rajouter **DESC**.

```
SELECT * FROM `users` ORDER BY lastname DESC
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PH

Tout afficher | Nombre de lignes : 25 Filtrer les lignes: Che

+ Options

←T→	id_user	firstname	lastname 1	sex
Éditer Copier Supprimer	9	Beyonce	Knowles	F
Éditer Copier Supprimer	3	Lara	Croft	F
Éditer Copier Supprimer	1	James	Bond	M
Éditer Copier Supprimer	8	Cool	Blues	X
Éditer Copier Supprimer	2	Jack	Bauer	M

A savoir que par **défaut** quand on met juste **ORDER BY nom_champ**, c'est équivalent à mettre **ORDER BY nom_champ ASC**. Testez-le.

On sait aussi mettre plusieurs ordres, si on obtient par exemple un champ reprit dans plusieurs enregistrements différents, on pourra lui demander de trier sur un second ordre.

(Exos)

1) **Affichez les noms et prénoms de tous les utilisateurs par ordre alphabétique de leurs prénoms. Montrez-moi une fois finis.**
2) **Affichez les noms et dates de naissance des 3 premiers utilisateurs ordonné par leurs dates de naissance, du plus jeune au plus vieux. Montrez-moi une fois finis.**
3) **Affichez les noms, prénoms et villes du 2ème au 5ème utilisateurs ordonné par leur ville, par ordre décroissant.**
4) **Affichez les noms et genre des utilisateurs par ordre croissant par rapport au genre et par ordre décroissant par rapport au nom. Vous constaterez un petit problème.**

Le comptage COUNT()

On peut aussi savoir combien il y a d'utilisateur, combien d'utilisateur venant par exemple d'une même ville, où il y a combien d'homme ou de femme etc. C'est avec le mot clé **COUNT() :**

La requête SQL a été exécutée avec succès.

```
SELECT COUNT(*) FROM `users`
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQ

+ Options

COUNT(*)
5

On voit bien qu'il y a 5 utilisateurs dans ma table. Pour savoir par exemple le nombre de personne s'appelant Jack, il faut mettre une condition :

```
SELECT COUNT(`firstname`) FROM `users` Where firstname="Jack"
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code so

+ Options

COUNT(`firstname`)
1

Les Alias (as)

On voit ici que le nom est archaïque, **count('firstname')**, si vous voulez vous pouvez renommer le nom de la table pour y voir plus clair en utilisant un alias. C'est représenté par un « **as** ».

```
SELECT COUNT(`firstname`) as "Le nombre de prenom Jack" FROM `users` Where
firstname="Jack"
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actuali

+ Options

Le nombre de prenom Jack
1

(Exos)

1) **Affichez le nombre de femmes en donnant comme titre de colonne le_nbr_femme.**
2) **Affichez le nombre d'hommes en donnant comme titre de colonne le_nbr_homme.**

Le GROUP BY

Le **GROUP BY** va permettre de grouper les éléments. Par exemple si on veut en même temps comptez le nombre d'homme et de femme.

```
SELECT COUNT(*) as le_nombre, sexe FROM `users` GROUP BY sexe
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source

Tout afficher | Nombre de lignes : 25 Filtrer les lignes:

+ Options

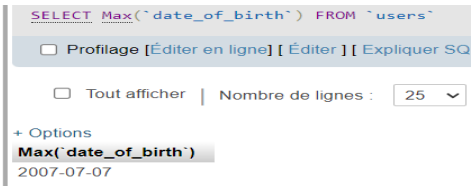
le_nombre	sexe
2	M
2	F
1	X

On peut faire un **GROUP BY** qui va regrouper tous les M qu'il va retrouver, donc dans la colonne le_nombre on aura 2 qui représentera les 2 hommes de ma table. Il fera pareil pour toutes les femmes et tous les autres.

Attention dans ma table tous mes utilisateurs ont un prénom, par contre si on avait mis prénom pouvait avoir un champ NULL et qu'on en avait un. En faisant **COUNT(firstname)** j'aurai eu 4 comme résultat au lieu 5. Il est donc préférable de mettre **COUNT(*).** En gros le count ne compte pas les lignes qui ont pour valeur NULL quand on le met le nom du champ.

Maximum, Minimum et Average

On a la possibilité aussi de chercher les valeurs maximum ou minimum d'un champ. Par exemple pour la date de naissance **MAX(champ)** :



On peut essayer de rajouter « **group by gender** » par exemple, et on verra donc la date maximum par genre.

Bien entendu c'est bien mieux avec des champs comme l'âge ou le prix etc. On peut faire pareil pour le minimum, **MIN(champ).**

On peut aussi faire une moyenne d'un champ, par exemple si on avait un champ âge ou poids ou prix pour un article etc. Il suffirait d'utiliser **AVG(champ)** :

- **SELECT AVG(nom_champ) FROM nomtable**

On peut aussi demander la somme en faisant **SUM(champ).**

Petit rappel, nous somme en Case Insensitive donc vous pouvez mettre en minuscule ou majuscule c'est pareil même pour les champs.

Exos

1) **Ajoutez en SQL une colonne « weight_kg » de type entier de taille 5 et NOT NULL.**
2) **Ajoutez des poids différends à tous les utilisateurs sauf le 1er et le dernier qui auront le même poids.**
3) **Affichez le poids Minimum des hommes en donnant comme nom de colonne « Le poids minimum des hommes ».**
4) **Affichez le poids maximum des femmes.**
5) **Affichez la somme des poids des utilisateurs.**
6) **Affichez la moyenne des poids des utilisateurs en donnant comme nom de colonne « La moyenne des poids des utilisateurs ».**
7) **Affichez la moyenne des poids des femmes, des hommes et des autres en mettant comme nom de colonne la_moyenne.**