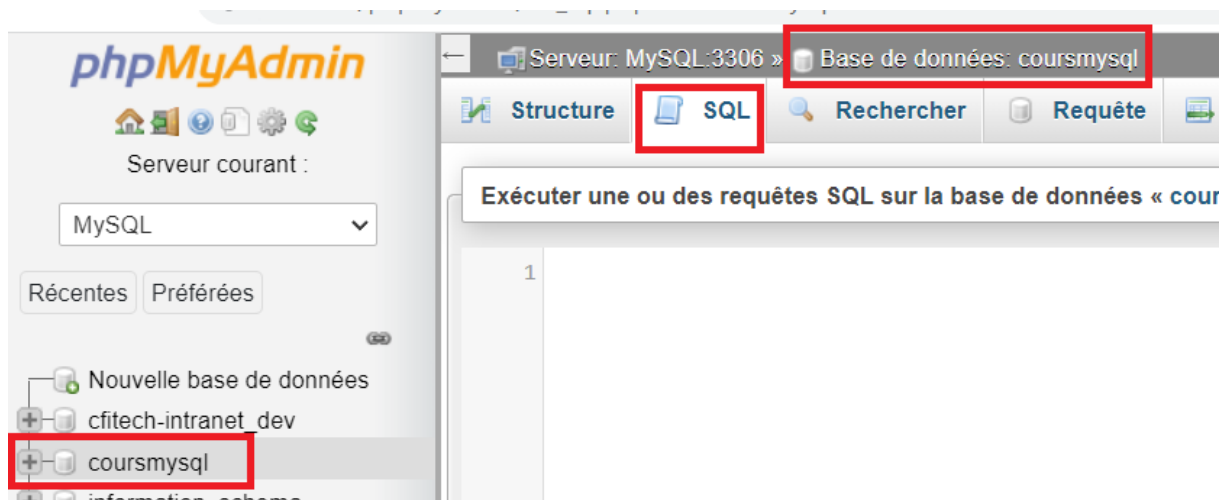




## 8) Création d'une Table en Ligne de commande

Nous avons vu précédemment comment créer une table via l'interface de PhpMyAdmin. Je vais ici vous montrer l'équivalent de ce qu'on a fait, en ligne de commande (**requêtes SQL**).

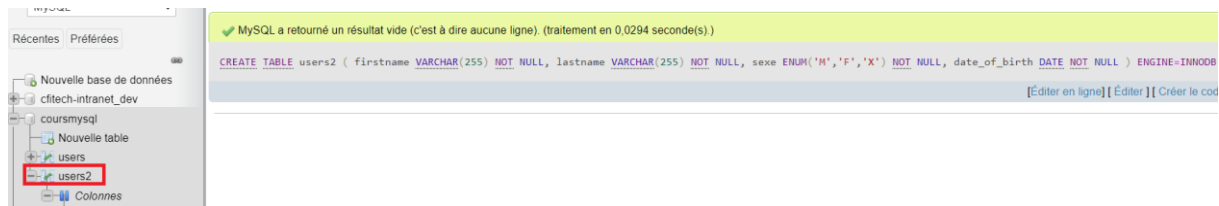
Il suffit de retourner dans la Base de données qu'on a créé pour ce cours : « **coursmysql** ».



Nous allons **créer** une nouvelle **table** d'utilisateur que l'on va appeler cette fois ci « **users2** », en reprenant les mêmes paramètres qu'on avait utilisé pour notre première table users.

```
1 CREATE TABLE users2 (  
2     firstname VARCHAR(255) COLLATE utf8mb4_general_ci NOT NULL,  
3     lastname VARCHAR(255) COLLATE utf8mb4_general_ci NOT NULL,  
4     gender ENUM('M','F','X') COLLATE utf8mb4_general_ci NOT NULL,  
5     date_of_birth DATE NOT NULL  
6 ) ENGINE=InnoDB
```

Une fois que vous aurez réécrit cette requête SQL, **appuyer** sur **exécuter**, vous verrez que dans votre base de données coursmysql il aura créé donc la table users2.



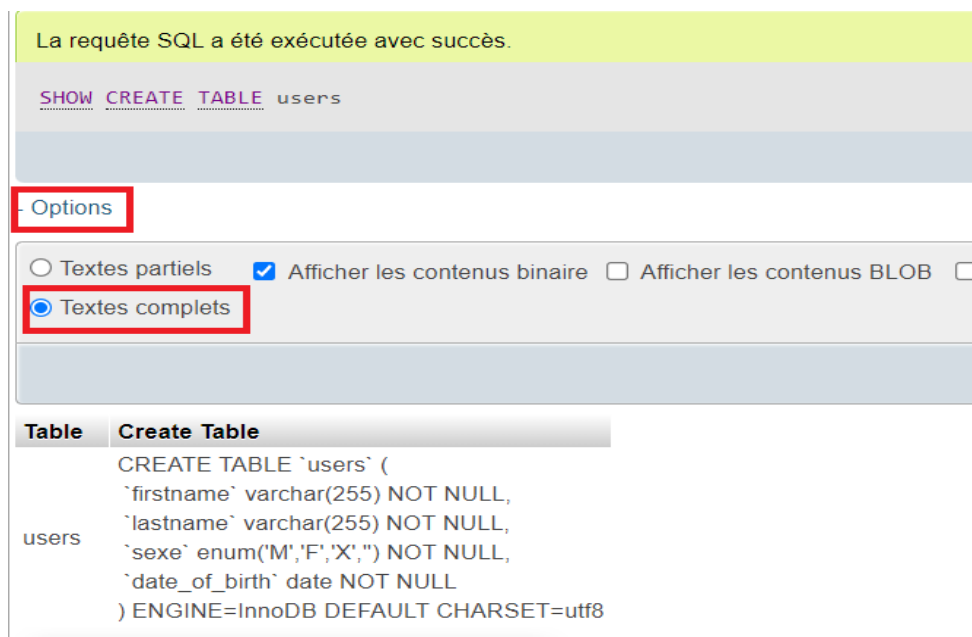
On pourra voir aussi notre commande écrite sur une seule ligne, moi ici j'ai agencé de manière à ce que ce soit plus clair.

Vous voyez que c'est plus simple et plus rapide de créer en utilisant l'interface de PhpMyAdmin.

Comme je vous l'ai déjà montré au début, tout ce qu'on fait en passant par l'interface visuel de phpMyAdmin, derrière il y a une requête SQL. Ici je vais vous montrer qu'on aurait pu voir directement comment on a créé la table « users ». Il suffit de cliquer sur la base de données « **coursmysql** ». D'ouvrir la ligne de commande SQL. Et d'exécuter cette requête :

- **SHOW CREATE TABLE users**

En gros, montre-moi comment la table « **users** » a été créé.



Pour voir le texte complet, cliquez sur option supplémentaire, sélectionnez textes complets ensuite exécutez. Vous voyez donc ici qu'il nous donne le code qui a permis de générer cette table « **users** ».

Pour supprimer une table, c'est pareil que pour supprimer une base de données comme on a vu, il suffit d'exécuter cette requête :

- **DROP TABLE users2**

## 9) Modification d'une table

Si on veut modifier une table, on peut le faire de manière visuelle via l'interface PhpMyAdmin.

Il y a plusieurs manières de le faire, je vais vous en montrer deux.

La première, Il faut aller dans notre table users, une fois cliquer dessus, vous allez dans l'onglet Structure et vous pourrez voir, en bas de notre structure de table, l'option ajouter nombre de colonne avant ou après une colonne selon votre choix.

Structure de table

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut
1	firstname	varchar(255)	utf8_general_ci		Non	Aucun(e)
2	lastname	varchar(255)	utf8_general_ci		Non	Aucun(e)
3	sexe	enum('M', 'F', 'X', '')	utf8_general_ci		Non	Aucun(e)
4	date_of_birth	date			Non	Aucun(e)

Ajouter 1 colonnes(s) après date\_of\_birth Exécuter

Vous pourrez ensuite appuyer sur « **exécuter** », et vous arriverez sur cette page

Nom: country, Type: VARCHAR, Taille/Valeurs: 255, Valeur par défaut: Aucun(e)

Structure

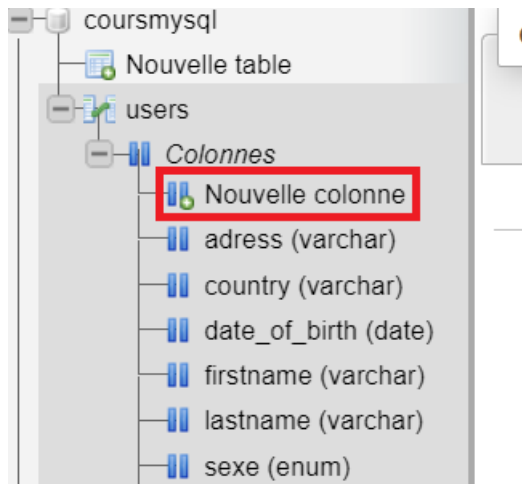
Vous pourrez rajouter donc comme nom de colonne « **country** », comme type « **VARCHAR** », de taille 255. Ensuite exécuter.

Vous verrez que la colonne country s'est rajoutée et comme d'habitude vous verrez la requête SQL équivalente qu'on aurait pu taper pour rajouter country en ligne de commande :

```
✓ La table users a été modifiée avec succès.
```

```
ALTER TABLE `users` ADD `country` VARCHAR(255) NOT NULL AFTER `date_of_birth`;
```

La deuxième, il suffit de cliquer sur votre table users, ensuite cliquer sur colonnes, et là on voit directement « **nouvelle colonne** ». En cliquant dessus on retombe sur l'interface d'ajout de nouveau champ (donc colonne).



A noter qu'ici on ne rajoute qu'une seule table et on n'a pas non plus la possibilité de choisir si on rajoute la colonne après par exemple lastname, etc. Donc ici il rajoute juste la colonne à la fin.

Sur cette image ci-dessus vous pouvez voir que c'est organisé par ordre alphabétique mais quand on clique dans l'onglet Structure on verra par ordre dans lequel on a introduit les données

<div> <div>Parcourir</div> <div>Structure</div> <div>SQL</div> <div>Rechercher</div> </div>					
Structure de table		Vue relationnelle			
	#	Nom	Type	Interclassement	Attribut
<input type="checkbox"/>	1	firstname	varchar(255)	utf8_general_ci	
<input type="checkbox"/>	2	lastname	varchar(255)	utf8_general_ci	
<input type="checkbox"/>	3	sexe	enum('M', 'F', 'X', '')	utf8_general_ci	
<input type="checkbox"/>	4	date_of_birth	date		
<input type="checkbox"/>	5	country	varchar(255)	utf8_general_ci	

Pareil vous pourrez aussi ajouter une colonne via la ligne de commande en passant la requête :

- **ALTER TABLE users ADD address VARCHAR(255) NOT NULL**

Pour supprimer suffit de faire un **DROP** :

- **ALTER TABLE users DROP address**

Pour renommer un champ, il suffit de faire en ligne de commande :

- **ALTER TABLE users CHANGE country city VARCHAR (255) NOT NULL**

Il nous aura changer country en city dans notre table.

Via l'interface PhpMyAdmin, il suffit de cliquer sur la colonne que l'on veut modifier ou dans Structure sur modifier, et il ouvrira la page de ce champ :



Là on a vu donc tout ce qu'on peut faire avec les champs (créer, modifier, supprimer).

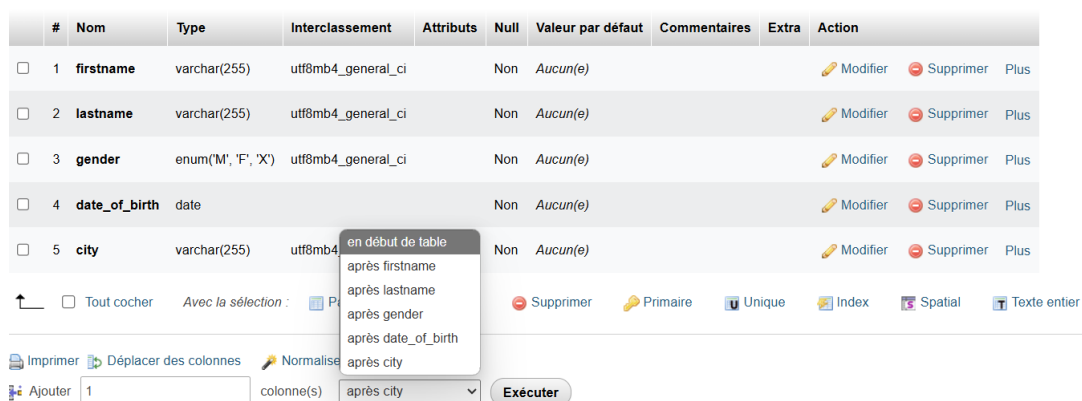
## 10) Le concept de clé

Les clés, c'est tout simplement quelque chose qui permet de vous identifier.

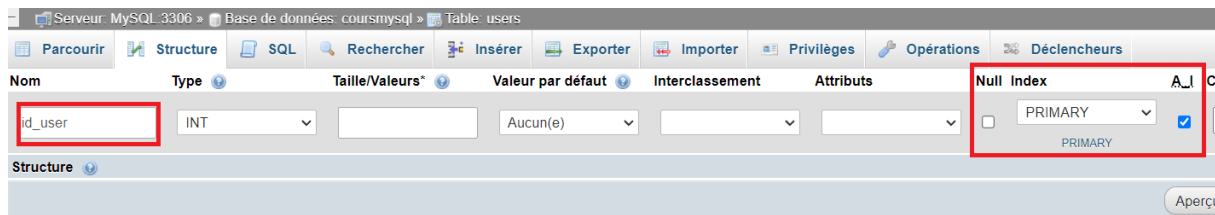
Le problème lorsqu'il y a plusieurs utilisateurs, il peut y avoir des concurrences c'est-à-dire qu'on aura peut-être des utilisateurs avec le même nom et prénom. Du coup comment faire pour différencier ces personnes. Dans la vie de tous les jours par exemple pour différencier deux personnes ayant le même nom et prénom, on utilise des « ID » qui sont des chiffres qui permettent d'identifier une personne, comme le numéro de registre national.

Du coup ce que l'on veut pouvoir faire sur MySQL, c'est créer un champ qui soit unique pour chacun des utilisateurs de notre table.

On va rajouter un champ en début de table



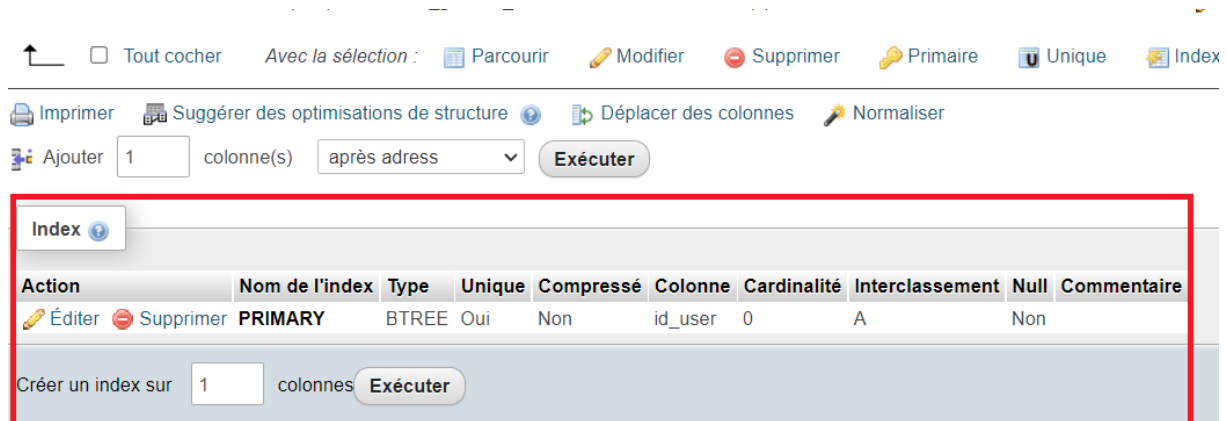
Ce champ on va lui donner un « id ». C'est quelque chose que vous allez revoir et entendre très souvent. C'est ce qui permettra d'identifier les personnes ou les articles ou les voitures si on avait une table voiture etc.



On va lui donner comme nom « **id\_user** », certains développeurs donnent juste le nom « **id** », mais ici je préfère préciser le nom de la table, mais les deux manières sont bonnes.

L'id est de type entier. On va préciser que c'est la clé primaire donc celle qui identifiera l'utilisateur. On n'oubliera pas non plus de cocher A.I. pour Auto-Incrémenté ce qui veut dire qu'à chaque nouvel utilisateur qu'on rajoutera, il donnera un numéro automatiquement en incrémentant de 1 à chaque fois. Donc premier utilisateur aura l'id\_user à 1, le deuxième utilisateur à 2, le troisième à 3 etc.

Quand vous aurez créé la clé primaire, si vous retourner dans structure vous verrez qu'on a maintenant une partie index et dedans on verra notre clé primaire :



Il existe d'autre type de clé et propriété. Si vous allez dans plus pour un champ.



On verra qu'il y a clé primaire, Unique, Index, Spatial etc.

Unique veut dire tout simplement que cette valeur doit être unique. C'est un peu comme une clé primaire mais ça précise juste une contrainte d'unicité. Par exemple une adresse mail.

La notion d'index c'est tout simplement de dire que MySQL va aussi organiser les données suivant ce champ-là. Lorsque vous allez ajouter un index ça va avoir un effet positif qui va tout simplement faire en sorte que ça va aller plus vite. Ça sert vraiment à améliorer les performances de votre table. Il faut le mettre sur un champ qui ne prend pas beaucoup de valeurs, par exemple sur le genre ici.

Spatial on ne le verra pas.

Ensuite on a Texte Entier ou fulltext, on peut faire des recherches avec une notion de pertinence dessus. C'est plus utilisé avec le moteur de base de données MyISAM. Nous ici on utilise INNODB.

## Exos

- 1) Modifier aussi la table « articles » en lui rajoutant une clé primaire id\_article.
- 2) Le nom de l'article devra être unique.
- 3) Modifier le nom du champ creation\_date, remplacer le par created\_at. Donnez-lui le bon format date et heure (allez relire les explications des types DATE dans le PDF 1). Par défaut ça prendra la date et heure actuelle.
- 4) Rajoutez un champ updated\_at qui représentera la date et heure de modification de l'article. Dès qu'on le modifiera il mettra à jour la date et l'heure. Et lui aussi par défaut il nous donnera la date et heure actuelle.
- 5) Si vous n'avez pas de table articles, aller voir la fin du PDF part 1.

## 11) Insertion de données dans nos tables

Maintenant que nous avons créé notre base de données et nos différentes tables. On va pouvoir rentrer dans le vif du sujet c'est à dire sauvegarder, modifier et récupérer des données. On va faire via l'interface graphique et via les commandes SQL.

Il y a pas mal de nouveau mot clé à apprendre, mais ne vous inquiétez pas, au début vous allez peut-être passer pas mal de temps à rechercher à chaque fois comment faire telle ou telle opération. Au fur et à mesure que vous allez utiliser ces requêtes là, vous allez les retenir.

On va commencer avec la table des utilisateurs. On va insérer un nouvel utilisateur.

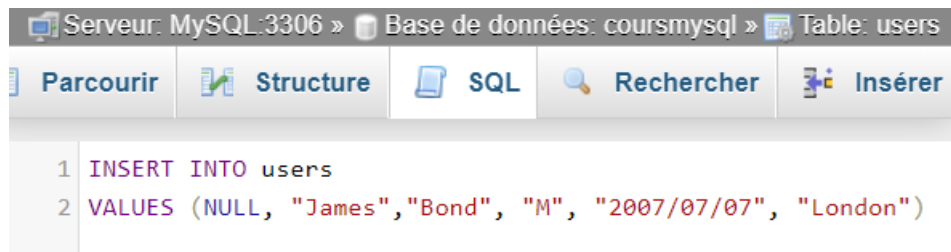
On peut ouvrir la ligne de commande SQL en vérifiant bien qu'on est dans la table users. Pour insérer il faudra donc faire, la requête **INSERT INTO** suivi du nom de la table à utiliser ensuite **VALUES** pour lui donner des valeurs concernant chacun des champs qu'on mettra entre parenthèse séparée d'une virgule.

On doit mettre Null pour la valeur de id\_user par ce qu'on a dit que c'était auto-incrémenté. En mettant null, c'est comme si on lui donne un champ vide.

Pour les chaînes de caractères, comme dans la plupart des langages de programmations, on va les utiliser entourer de guillemets double ou simple. Si on a des guillemets dans la chaîne de caractères même, on peut utiliser un anti slash pour que ça fonctionne :

"Il m'a dit \"Bonjour\" l'autre jour".

Pour ce qui est de la DATE il faut taper en format ISO c'est-à-dire Y/m/d pour année, mois et jour.



The screenshot shows a MySQL client window with the title bar 'Serveur: MySQL:3306 » Base de données: coursmysql » Table: users'. The toolbar includes buttons for 'Parcourir', 'Structure', 'SQL', 'Rechercher', and 'Insérer'. The SQL editor contains the following query:

```
1 INSERT INTO users
2 VALUES (NULL, "James", "Bond", "M", "2007/07/07", "London")
```

Vous verrez une fois cette requête exécutée, qu'on aura notre premier enregistrement.



The screenshot shows the same MySQL client window after executing the query. The 'Parcourir' button is highlighted with a red box. A green status bar indicates 'Affichage des lignes 0 - 0 (total de 1, traitement en 0,0003 seconde(s).)'. The SQL editor shows 'SELECT \* FROM `users`'. Below the editor, there are controls for 'Tout afficher', 'Nombre de lignes: 25', and 'Filtrer les lignes: Chercher dans cette table'. The '+ Options' section is expanded, showing icons for 'Éditer', 'Copier', and 'Supprimer'. A table of results is displayed, with the header row and the first data row highlighted by a red box:

id_user	firstname	lastname	sexe	date_of_birth	city
1	James	Bond	M	2007-07-07	London

Pour pouvoir enregistrer plusieurs valeurs en même temps il suffit de faire comme ceci :



The screenshot shows a MySQL client window with the title bar 'Exécuter une ou des requêtes SQL sur la table « coursmysql.users »:'. The SQL editor contains the following query:

```
1 INSERT INTO users
2 VALUES
3 (NULL, "Jack", "Bauer", "M", "2004/12/24", "New-York"),
4 (NULL, "Lara", "Croft", "F", "2000/08/01", "Washington");
```



On peut donc insérer plusieurs enregistrements à la fois, il suffit de les séparer par une virgule.

+ Options

<div><div>←T→</div><div></div></div>			id_user	firstname	lastname	sexe	date_of_birth	city	
<div><div><div></div></div></div>	<div><div><div></div></div></div> Éditer	<div><div><div></div></div></div> Copier	<div><div><div></div></div></div> Supprimer	1	James	Bond	M	2007-07-07	London
<div><div><div></div></div></div>	<div><div><div></div></div></div> Éditer	<div><div><div></div></div></div> Copier	<div><div><div></div></div></div> Supprimer	2	Jack	Bauer	M	2004-12-24	New-York
<div><div><div></div></div></div>	<div><div><div></div></div></div> Éditer	<div><div><div></div></div></div> Copier	<div><div><div></div></div></div> Supprimer	3	Lara	Croft	F	2000-08-01	Washington

Insertion simple :

- **INSERT INTO [nom table] VALUES (valeur1, valeur2, ...) ;**

Insertion multiple :

- **INSERT INTO (nom table) VALUES  
(valeur1, valeur2, ...),  
(valeur1, valeur2, ...),  
(valeur1, valeur2, ...);**

Ici je vais aussi vous montrer une commande propre à MySQL pour rajouter un enregistrement, mais ça ne passe pas sur les autres SGBD :

- **INSERT INTO [nom table] SET champ1= "valeur1", champ2="valeur2"...**

## 12) Suppression d'un enregistrement

Pour supprimer c'est aussi simple :

- **DELETE FROM [nom table] WHERE [conditions]**

Faites cette commande, si on veut supprimer le ou les utilisateurs qui ont comme prénom Lara, il suffit d'écrire :

- **DELETE FROM users WHERE firstname = "Lara"**

Attention écrire juste « **DELETE FROM users** » va supprimer tous les utilisateurs de la table.

Ici donc j'ai indiqué une condition pour pouvoir supprimer. On peut choisir la condition qu'on veut.

Si on a par exemple 10 utilisateurs qui ont comme « city = "Brussel" » et qu'on veut supprimer juste 4 utilisateurs ayant cette ville, on rajoutera après la condition une LIMIT 4 :

- **DELETE FROM users WHERE city = "Brussel" LIMIT 4**

On utilise le LIMIT comme une sécurité au cas où on se trompe dans la requête et que ça supprime définitivement des données.

On peut aussi utiliser plusieurs conditions en rajoutant « OR » ou « AND »

- **DELETE FROM users WHERE city = "Brussel" OR firstname = "John"**

On y reviendra plus tard pour tout ce qui est condition.

Avec le **DELETE** quand on supprime par exemple le dernier utilisateur qu'on a introduit, qui avait comme « id » 10, il ne sera plus en base de données bien entendu. Par contre quand on recréera un utilisateur, l'auto-incrémente ne va pas recréer un nouvel utilisateur avec comme « id » 10 mais bien avec « id 11 ».

Un id ne sera jamais réattribué.

Quand on fait la requête « **DELETE FROM users** », ça supprime tous nos utilisateurs dans la table users. Il va tous les supprimer, la table sera vide par contre quand on recréera un nouvel utilisateur son id ne commencera pas à 1 mais continuera. En gros si j'avais 10 utilisateurs au moment où j'ai supprimé et que le dernier utilisateur avait comme id 10, même si je vide la table avec « **DELETE FROM users** », quand on recréera on aura un utilisateur avec comme id 11.

Pour pouvoir remettre à 0 l'auto-incrémentation, il faut utiliser une autre commande qui supprimera tous les éléments comme un DELETE mais qui aussi réinitialisera les id.

Cette commande c'est :

- **TRUNCATE TABLE users**

## Exos

- 1) Créer 5 utilisateurs différents.
- 2) Supprimer le 3ème et 5ème utilisateur en donnant les bonnes conditions.
- 3) Vider la table users et recrée ces 3 utilisateurs ci-dessous.  
Vous devez avoir comme Id 1, 2 et 3.

+ Options

<div><div><div>←</div><div>T</div><div>→</div></div><div></div></div>				id_user	firstname	lastname	sexe	date_of_birth	city
<div><div><div></div></div><div><div><div></div></div></div><div><div>Éditer</div><div>Copier</div><div>Supprimer</div></div></div>	1	James	Bond	M	2007-07-07	London			
<div><div><div></div></div><div><div><div></div></div></div><div><div>Éditer</div><div>Copier</div><div>Supprimer</div></div></div>	2	Jack	Bauer	M	2004-12-24	New-York			
<div><div><div></div></div><div><div><div></div></div></div><div><div>Éditer</div><div>Copier</div><div>Supprimer</div></div></div>	3	Lara	Croft	F	2000-08-01	Washington			