

SQL/MySQL (Create, Alter & Drop)

1. CREATE TABLE : Le langage SQL (Structured Query Language) permet d'interroger une base de données qui supporte ce langage, les plus connues sont MS Access, mSQL, MySQL et PostgreSQL. Comme vous le savez nous utiliserons MySQL (**attention mSQL n'est pas MySQL**), nuance importante.

Comme je vous le disais en introduction nous allons travailler chez [free.fr](http://sql.free.fr/phpMyAdmin/), vous pouvez donc vous rendre à <http://sql.free.fr/phpMyAdmin/> pour accéder à votre base de données MySQL via l'interface [phpMyAdmin](#). Notez que 99% des hébergeurs proposent phpMyAdmin pour administrer votre base de données (cela facilite grandement la vie).

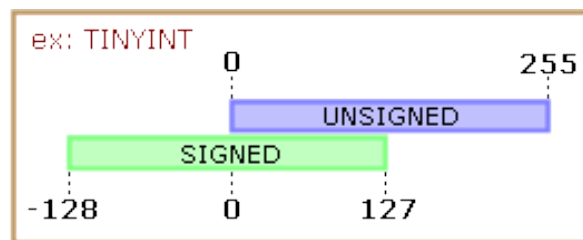
La première chose que nous allons devoir faire c'est de créer une table, c'est la commande **CREATE TABLE**, voyez la syntaxe ci-dessous qui permet de créer **clients_tbl**. Il est important de savoir comment l'on crée une table en SQL avant de passer par l'interface phpMyAdmin pour le faire :

Syntaxe SQL pour créer la table : clients_tbl						
CREATE TABLE clients_tbl (id INT not null AUTO_INCREMENT, prenom VARCHAR (50) not null , nom VARCHAR (50) not null , ne_le DATE not null , ville VARCHAR (90) not null , enfants INT not null , PRIMARY KEY (id))						
Commentons la table en la visualisant sous phpMyAdmin pour MySQL						
Champ	Type	Null	Defaut	Extra	Commentaire	
id	int	Non	0	auto_increment	INT pour Integer (nombre entier) : C'est l'id qui va nous permettre de classer nos enregistrements, l'auto-increment se charge d'affecter un nouveau numéro aux nouveaux enregistrements qui s'ajoutent dans la table.	
prenom	varchar(50)	Non			Varchar pour Chaîne de caractères : Nous l'utilisons quand nous souhaitons que le champ puisse recevoir des chaînes mélangeant texte, nombre, etc ...	
nom	varchar(50)	Non			Idem que PRENOM , la valeur placée entre parenthèses définit le nombre maximum de caractères que le champs accepte, ici 50. Si vous essayez d'insérer une chaîne de 55 caractères, les 5 derniers seront coupés.	
ne_le	date	Non	0000-00-00		Date : Permet de stocker des dates, mais attention TOUJOURS au format US soit : Année/mois/jour, si vous envoyez un format français J/M/A vous obtiendrez une date fausse dans la base.	
ville	varchar(90)	Non			Idem que PRENOM , ici nous avons spécifié 90 caractères car certaines villes comportent beaucoup de caractères (avec 90 nous sommes sûr)	
enfants	int	Non	0		INT (nombre entier) : Dans ce champs n'arriveront que des nombres entiers donc le INT voir SMALL INT et de rigueur.	
Ci-dessous la table : clients_tbl une fois créée						
++++++ + id + prenom + nom + ne_le + ville + enfants + ++++++ Bien sûr pour le moment cette table ne comporte aucun enregistrement ...						

Dès cet instant vous devez consulter [la documentation officielle de MySQL](#) pour connaître tous les types de champs que vous pouvez utiliser dans une table, voici quelques exemples :

- **TINYINT** : Entier de 0 à 255 (unsigned)
- **SMALLINT** : Entier de 0 à 65535 (unsigned)
- **MEDIUMINT** : Entier de 0 à 16777215 (unsigned)
- **INT** : Entier de 0 à 4294967295 (unsigned)
- **BIGINT** : Entier de 0 à 18446744073709551615 (unsigned)
- **DECIMAL** : Un nombre à virgule flottante
- **DATE** : Une date, va de '1000-01-01' à '9999-12-31'
- **DATETIME** : Date et Heure, va de '1000-01-01 00:00:00' à '9999-12-31 23:59:59'
- **TIMESTAMP** : Date et Heure exprimée en secondes depuis le 1er janvier 1970. Va de '1970-01-01 00:00:00' à quelque part, durant l'année 2037
- **TIME** : Une mesure de l'heure, va de '-838:59:59' à '838:59:59'
- **YEAR** : Une année, va de 1901 à 2155
- **CHAR** : Chaîne de caractère de taille fixe, va de 1 à 255 caractères
- **VARCHAR** : Chaîne de caractère de taille variable, va de 1 à 255 caractères
- **TINYTEXT ou TINYBLOB** : Un objet BLOB ou TEXT, longueur maximale de 255
- **TEXT ou BLOB** : Un objet BLOB ou TEXT, longueur maximale de 65535
- **MEDIUMTEXT ou MEDIUMBLOB** : Un objet BLOB ou TEXT, longueur maximale de 16777215
- **LONGTEXT ou LOB** : Un objet BLOB ou TEXT, longueur maximale de 4294967295

A noter : Concernant la notion de **UNSIGNED** est importante pour les Integer (INT, SMALLINT, etc.) je vous propose un petit schéma simple pour vous expliquer la différence entre un Integer UNSIGNED ou pas:



Le **TINYINT** couvre de 0 à 255 quand il est **UNSIGNED**, mais il couvrira de - 128 à 127 (soit toujours 255) si il ne l'est pas, Ceci est valable pour tous les Integers (nombres entiers).

2. ALTER TABLE : Une fois que votre table est créée vous pourrez bien sûr la modifier en utilisant **ALTER TABLE**, voyez l'exemple ci-dessous pour ajouter un champs :

Syntaxe SQL pour ajouter le champ 'tel' à la table : **clients_tbl**

```
ALTER TABLE clients_tbl ADD tel INT not null
```

Ci-dessous la table **clients_tbl** une fois modifiée

```
+++++
+  id  +  prenom  +  nom  +  ne_le  +  ville  +  enfants  +  tel  +
+++++
```

Voilà ci-dessus notre table a été modifiée grâce à **ALTER TABLE** puis **ADD** (comme ajouter).

Maintenant supprimons le champs tel de la table :

Syntaxe SQL pour supprimer le champ 'tel' de la table : **clients_tbl**

```
ALTER TABLE clients_tbl DROP tel
```

Ci-dessous la table `clients_tbl` une fois modifiée

```
+++++  
+  id  +  prenom  +   nom   +  ne_le  +   ville  +  enfants  +  
+++++
```

Voilà ci-dessus notre table a été modifiée grâce à `ALTER TABLE` puis `DROP`. Cette commande `DROP` va également nous être utile pour supprimer une table complète.

3. DROP TABLE : Il s'agit de la commande qui permet de supprimer une table complète, attention en supprimant une table vous perdez tout ce qu'elle contenait, donc à utiliser avec prudence !

Syntaxe SQL pour supprimer la table : `clients_tbl`

```
DROP TABLE clients_tbl
```

Notre table `clients_tbl` a été complètement effacée avec la commande `DROP TABLE`, dans notre cas elle ne contenait aucune information mais dans le cas contraire, cela signifie la perte de toutes les données de la table.

Pour finir : Nous avons vu ici la majeure partie des choses importantes à connaître en ce qui concerne la création et la modification des tables. Dans les prochains exercices nous verrons comment traiter les données d'une table avec les commandes `SELECT`, `INSERT`, `UPDATE` et `DELETE`.

www.phpdebutant.org © 2006 – L'équipe de phpDebutant