INSERT : Insérer des données dans une table MySQL

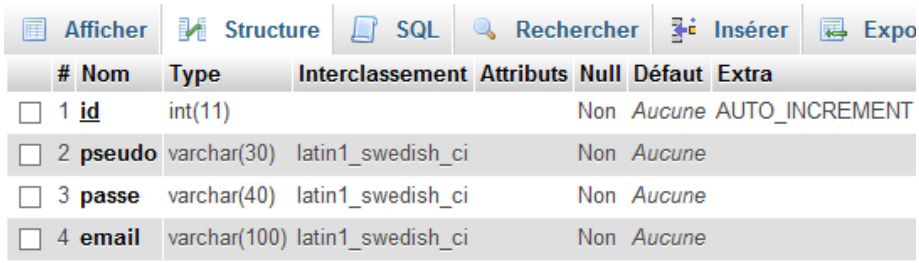
Date de dernière mise à jour : 05/02/2014 à 11:36

INSERT INTO : Insérer des données avec MySQL

Il existe différentes manières d'insérer des données dans une table MySQL. Pour tous ces exemples, nous allons partir d'une table de membres telle que vous pourriez la trouver sur la plupart des sites web, et ne contenant que quelques champs :

-L'identifiant unique du membre : **id**
-Son pseudo : **pseudo**
-Son adresse email : **email**

La structure de la table est la suivante :



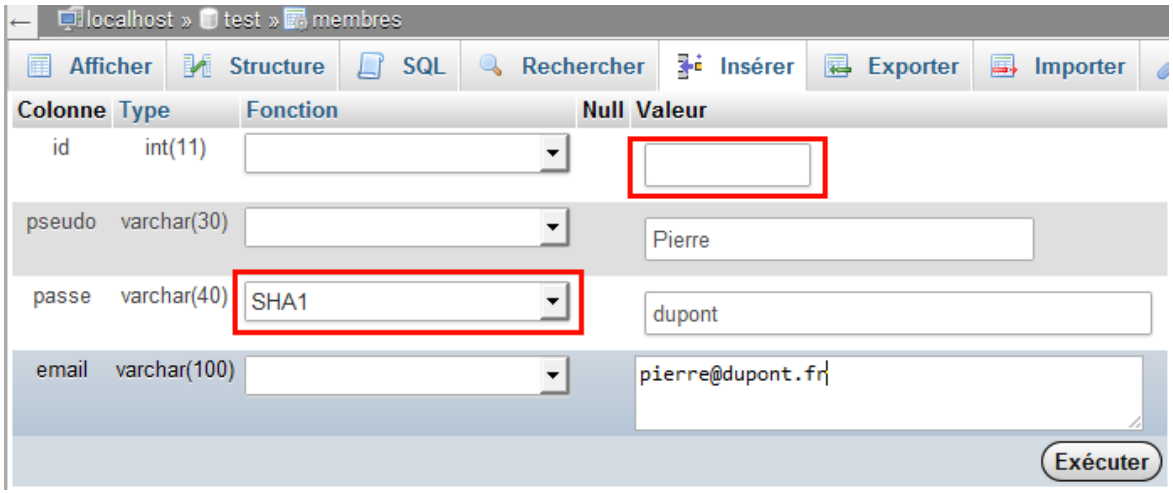
	#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra
☐	1	id	int(11)			Non	*Aucune*	AUTO_INCREMENT
☐	2	pseudo	varchar(30)	latin1_swedish_ci		Non	*Aucune*	
☐	3	passe	varchar(40)	latin1_swedish_ci		Non	*Aucune*	
☐	4	email	varchar(100)	latin1_swedish_ci		Non	*Aucune*	

Si vous souhaitez reproduire cette table, vous pouvez exécuter le code SQL suivant (un outil tel que PHPMyAdmin est recommandé car plus pratique) :

```
CREATE
TABLE
IF
NOT
EXISTS
 `membres` (
  `id
` int
(
11
)
NOT
NULL
 AUTO_INCREMENT,
  `pseudo` varchar
(
30
)
NOT
NULL
,
  `passe` varchar
(
```

```
40
)
NOT
NULL
,
  `email` varchar
(
100
)
NOT
NULL
,
PRIMARY
KEY
(
`id
`)
)
 ENGINE=
InnoDB  DEFAULT
 CHARSET=
latin1;
```

Nous allons insérer un membre dont le pseudo est **Pierre**, le mot de passe sera **dupont** et sera encodé à l'aide de la fonction **SHA1()** de MySQL (afin de ne pas être en clair dans la base), et l'email sera **pierre@dupont.fr**. L'équivalent en cliquant dans PHPMyAdmin serait de faire ceci :



Remarquez que la valeur du champ **id** est vide, car il s'agit d'un champ auto-incrémenté, c'est à dire que MySQL va lui attribuer une valeur automatiquement supérieure à la dernière valeur insérée dans la table. Cela est très pratique et vous évite de devoir effectuer une requête de séletion pour récupérer le maximum du champ id puis de l'incrémenter de 1.

INSERT INTO table VALUES(...)
C'est une syntaxe qui se révèle plutôt courte mais qui, en contrepartie, vous oblige à spécifier les valeurs de tous les champs, dans l'ordre dans lequel ils sont implémentés dans la table MySQL.

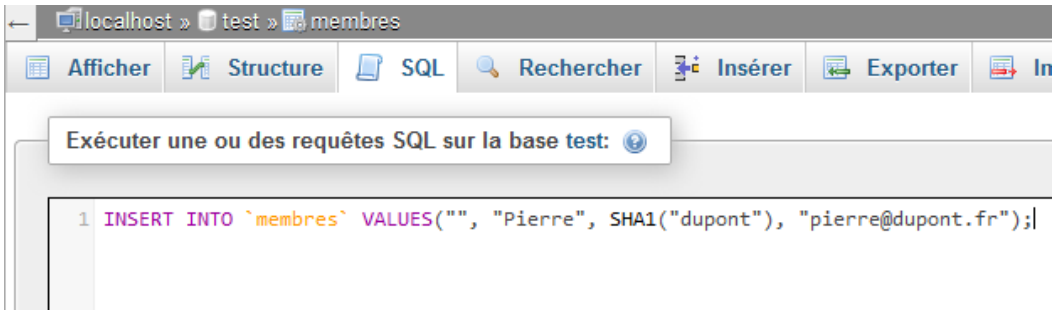
Pour notre exemple ci-dessus, nous sommes donc obligés de spécifier les valeurs des champs **id, pseudo, passe, email** en respectant scrupuleusement cet ordre.

La syntaxe à utiliser est la suivante :

INSERT
INTO
`
membres
`
VALUES
(
""
,
"Pierre"
,
SHA1(
"dupont"
)
,
"pierre@dupont.fr"
)
;

Remarquez que le nom de la table est entouré du caractère d'échappement `. Ceci n'est pas obligatoire dans le cas où le nom de votre table ne contient pas de caractères spéciaux (ce qui est conseillé). Remarquez également l'absence de guillemets doubles autour de la fonction SHA1(). Ceci est normal car il s'agit d'une fonction de MySQL.

Vous pouvez, via l'onglet **SQL** de PHPMyAdmin, saisir entièrement cette syntaxe via l'onglet **SQL**.



INSERT INTO table(champs ...) VALUES(...)
Cette syntaxe est à privilégier car :

-Elle vous permet de spécifier les seuls champs que vous souhaitez remplir (les autres devant avoir une valeur par défaut).
-Elle vous permet de vous affranchir de l'ordre des champs dans la table, vous pouvez donc rajouter ou enlever des colonnes ultérieurement sans toucher à vos requêtes.
-Elle vous permet de savoir ce que vous allez insérer comme données, sans devoir afficher la structure de la table, ce qui est bien plus pratique, surtout avec des tables comportant de nombreux champs.
-Elle vous offre une compatibilité améliorée avec les autres SGBDR- (systèmes de gestion de base de données relationnelles), tels PostgreSQL par exemple. Cela vous évite de devoir réécrire vos requêtes si vous décidez de changer de SGBDR.

La syntaxe à utiliser est la suivante :

INSERT

```
INTO
`
membres
`
(
`
pseudo
`
,
`
passe
`
,
`
email
`
)
VALUES
(
"Pierre"
,
 SHA1(
"dupont"
)
,
"pierre@dupont.fr"
)
;
```

Remarquez la présence du caractère d'échappement entourant les noms de champs. Il n'est pas obligatoire si les noms de vos champs n'ont pas de caractères spéciaux.

Insertion de caractères spéciaux, champs spéciaux et syntaxe

Guillemets doubles

Imaginions que vous souhaitiez insérer un enregistrement contenant des guillemets doubles, par exemple comme pseudo **Pierre "le fou"** , vous devez échapper les guillemets doubles à l'aide du caractère d'échappement antislash ****. Voici un exemple :

```
INSERT
INTO
`
membres
`
(
`
id
`
,
`
pseudo
`
,
`
passe
```

`
,
`
email
`
)
VALUES
(
NULL
,
"Pierre
\"
le fou
\"
"
,
SHA1(
"pierre"
)
,
"pierrelefou@fous.fr"
)
;

Champs numériques

Les champs numériques (INT, FLOAT, etc.) n'ont pas besoin d'être entourés de guillemets lorsque vous effectuez vos requêtes d'insertion. Le caractère séparateur de milliers est généralement le point. Prenons l'exemple d'une table **articles** dont la structure est la suivante :

CREATE
TABLE
IF
NOT
EXISTS
`
articles
`
(
`
id
`
int
(
11
)
NOT
NULL
AUTO_INCREMENT
,
`
libelle
`
varchar
(

```
200
)
NOT
NULL
,
`
prix
`
float
NOT
NULL
,
PRIMARY
KEY
(
`
id
`
)
)
 ENGINE=
InnoDB
DEFAULT
 CHARSET=
latin1;
```

Remarquez le champ **prix** de type **FLOAT**. Vous souhaitez insérer un article dont le libellé est **Mon article** et le prix est de **50.25 €**. La syntaxe à utiliser est la suivante :

```
INSERT
INTO
`
articles
`
(
`
libelle
`
,
`
prix
`
)
VALUES
(
'mon article'
,
50.25
)
;
```