

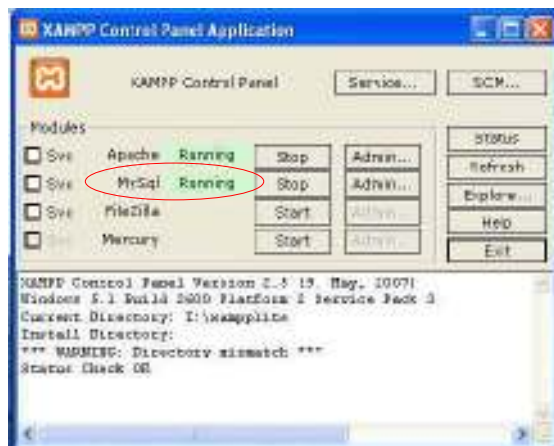
Pour utiliser MYSQL en interactif,

Comment dialoguer avec le serveur Mysql ?

La commande **mysql** exécute un utilitaire qui permet de dialoguer avec le serveur MySQL. Ce mode de gestion s'effectue en ligne de commande et est appelé le mode console.

Lancer cet utilitaire mysql. A ce stade que l'on soit root ou utilisateur quelconque, aucun mot de passe n'est demandé. Et l'on se trouve devant le prompt **mysql>** ce qui signifie que l'interpréteur de commandes SQL attend nos requêtes.

```
$ mysql
Welcome to the MySQL monitor. Commands ends with ; or \g
Your Mysql connection id is ...
tape help; ou \h for help
mysql>
```



Utilisation de la console mysql avec xampp

Ouvrir une console, se placer dans le répertoire `mysql\bin\` et taper

```
mysql -u root
```

Cette méthode est assez compliquée et je suggère une fichier **mysqlRoot.bat** placé à la racine de xampp et contenant:

```
mysql\bin\mysql -u root
```

Vous êtes connecté

La commande **mysql -u root** exécute un utilitaire qui permet de dialoguer avec le serveur MySQL. Ce mode de gestion s'effectue en ligne de commande et est appelé le mode console.

Lancer cet utilitaire mysql. A ce stade que l'on soit root ou utilisateur quelconque, aucun mot de passe n'est demandé. Et l'on se trouve devant le prompt **mysql>** ce qui signifie que l'interpréteur de commandes SQL attend nos requêtes.

```
$ mysql
Welcome to the MySQL monitor. Commands ends with ; or \g
Your Mysql connection id is ...
tape help; ou \h for help
mysql>
```

Pour sortir proprement quit:

```
mysql> quit
Bye
#
```

Quels sont les premiers utilisateurs ?

Un super-utilisateur du serveur MySQL, appelé `root@localhost`

Celui-ci possède **tous les droits** sur les bases de données et en particulier sur la base d'administration nommée elle aussi `mysql`

Or, à l'installation root peut se connecter sans mot de passe ! Il faudra de toute urgence lui en imposer un !

Remarque : Les commandes SQL et en particulier celles que l'on passe dans l'utilitaire `mysql`, ne sont pas sensibles à la casse. Mais par convention, il est d'usage d'écrire les mots-clés SQL en majuscules.

Ouvrir en tant qu'administrateur:

En tant qu'**administrateur (root)** de `mysql`, cela donne

```
mysql -u root -p
```

-p car **l'administrateur devrait avoir un mot de passe au moins**

Remarque : Les commandes SQL et en particulier celles que l'on passe dans l'utilitaire `mysql`, ne sont pas sensibles à la casse. Mais par convention, il est d'usage d'écrire les mots-clés SQL en majuscules.

Voir les bases

```
mysql> SHOW databases;
```

```
+-----+
| Database |
+-----+
| mysql    |
+-----+
1 row in set (0.02 sec)
```

Comment créer une nouvelle base ?

root crée une nouvelle base de données nommée `essais` et vérifie sa présence dans la liste des bases

```
mysql> CREATE DATABASE essai;
```

```
Query OK, 1 row affected (0.05 sec)
```

Sélectionner une base de données

La création d'une base de données ne la sélectionne pas pour l'utilisation; vous devez le faire explicitement. Pour rendre `essai` la base courante, utilisez cette commande:

```
mysql> USE essai;
```

```
Database changed
```

Créer la base de données est la partie facile, mais jusque-là elle est vide, comme vous le montre

SHOW TABLES;

```
mysql> SHOW TABLES;
```

```
Empty set (0.00 sec)
```

Comment supprimer une base ?

En mode console, root passe la commande sans recours (attention !)

```
mysql> DROP DATABASE  essai;
Query OK, 0 row affected (0.00 sec)
```

Une curieuse requete : "select"

Voilà une commande simple qui demande au serveur de vous donner son numéro de version et la date courante. Entrez-la comme suit, juste après l'invite `mysql>` puis pressez Enter :

```
mysql> SELECT VERSION(), CURRENT_DATE;
```

```
mysql> SELECT VERSION(), CURRENT_DATE;
+-----+-----+
| VERSION()          | CURRENT_DATE |
+-----+-----+
| 5.1.37-1ubuntu5    | 2010-02-17   |
+-----+-----+
1 row in set (0,00 sec)
```

La requête révèle plusieurs choses à propos de `mysql` :

- Une commande consiste normalement en une commande SQL suivie d'un point-virgule. (Il y a quelques cas où le point-virgule n'est pas requis. `QUIT`, mentionnée plus tôt, en fait partie. Nous verrons les autres plus tard.)
- Lorsque vous entrez une commande, `mysql` l'envoie au serveur pour l'exécution et affiche le résultat, puis affiche un autre `mysql>` pour indiquer qu'il attend une autre commande.
- `mysql` affiche le résultat des requêtes dans une table (lignes et colonnes). La première ligne contient le nom des colonnes. Les lignes suivantes constituent le résultat de la requête. Normalement, les titres des colonnes sont les noms des champs des tables de la base de données que vous avez récupérés. Si vous récupérez la valeur d'une expression au lieu d'une colonne (comme dans l'exemple précédent), `mysql` nomme la colonne en utilisant l'expression elle-même.
- `mysql` vous indique combien de lignes ont été retournées et combien de temps d'exécution la requête a pris, ce qui vous donnera une approximation des performances du serveur. Ces valeurs sont imprécises car elles représentent le temps logiciel (et non le temps processeur ou matériel), et qu'elles sont affectées par des facteurs tels que la charge du serveur ou l'accessibilité du réseau. (Dans un souci de brièveté, la ligne contenant `rows in set` n'est plus montrée dans les exemples suivants de ce chapitre.)

Les mots-clé peuvent être entrés sous n'importe quelle forme de casse. Les requêtes suivantes sont équivalentes :

```
mysql> SELECT VERSION(), CURRENT_DATE;
mysql> select version(), current_date;
mysql> SeLeCt vErSiOn(), current_DATE;
```

Voilà une autre requête. Elle montre que vous pouvez utiliser `mysql` en tant que simple calculatrice :

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
+-----+-----+
| SIN(PI()/4) | (4+1)*5 |
+-----+-----+
| 0.707107    | 25      |
+-----+-----+
```

Les commandes vues jusqu'à présent ont été relativement courtes, et tenaient sur une seule ligne. Vous pouvez même entrer plusieurs requêtes sur une seule ligne. Il suffit de terminer chacune d'elle par un point-virgule :

```
mysql> SELECT VERSION(); SELECT NOW();
```

```
+-----+
| VERSION()      |
+-----+
| 5.1.37-1ubuntu5 |
+-----+
1 row in set (0,00 sec)
```

```
+-----+
| NOW()          |
+-----+
| 2010-02-17 12:20:23 |
+-----+
1 row in set (0,00 sec)
```

Une commande ne doit pas être obligatoirement sur une seule ligne ; les commandes qui exigent plusieurs lignes ne sont pas un problème. **mysql** recherche le point-virgule de terminaison.

Une requête sur plusieurs lignes :

```
mysql> SELECT
-> user(),
-> CURRENT_DATE
-> ;
+-----+-----+
| user()      | CURRENT_DATE |
+-----+-----+
| root@localhost | 2010-02-17   |
+-----+-----+
1 row in set (0,00 sec)
```

Si vous décidez d'annuler une commande que vous êtes en train de taper, faites-le en entrant **\c** :

```
mysql> SELECT
-> USER()
-> \c
mysql>
```

Ici aussi, portez votre attention sur l'invite. Elle se transforme à nouveau en **mysql>** après que vous ayez entré **\c**, vous informant que **mysql** est prêt pour une nouvelle requête.

Le tableau suivant montre les différentes invites que vous pourrez voir et résume leur signification quand à l'état dans lequel se trouve **mysql** :

Invite	Signification
mysql>	Prêt pour une nouvelle commande.
->	En attente de la ou des lignes terminant la commande.
'>	En attente de la prochaine ligne collectant une chaîne commencée par une apostrophe ('').
">	En attente de la prochaine ligne collectant une chaîne commencée par un guillemet ("").

```
` > En attente de la prochaine ligne collectant une chaîne commencée par un guillemet oblique ('').
```

Les commandes sur plusieurs lignes sont la plupart du temps des accidents, lorsque vous voulez faire une commande sur une seule ligne et que vous oubliez le point-virgule de fin. Dans ce cas, `mysql` attend la suite de votre saisie :

```
mysql> SELECT USER()  
->
```

Si cela arrive, il est fort probable que `mysql` attende le point-virgule.

Liste des bases

Utilisez la commande `SHOW` pour trouver quelles bases existent déjà sur le serveur :

```
mysql> SHOW DATABASES;  
+-----+  
| Database |  
+-----+  
| mysql    |  
| test     |  
| tmp      |  
+-----+
```

La liste des bases de données est probablement différente sur votre machine, mais les bases `mysql` et `test` y figurent sûrement. La base `mysql` est requise car elle gère les accès et les privilèges. La base `test` est souvent fournie pour que les utilisateurs y effectuent leurs tests.

Notez que **vous ne pourrez voir toutes les bases de données si vous n'avez pas le privilège** `SHOW DATABASES`.

Utiliser une base

```
mysql> use mysql;  
Database changed
```

Voir les tables

```
mysql> show tables;
```

```
+-----+  
| Tables_in_mysql |  
+-----+  
| columns_priv    |  
| db              |  
| func            |  
| host            |  
| tables_priv     |  
| user            |  
+-----+
```

6 rows in set (0.02 sec)

Afficher les champs user et password de la table user:

```
mysql> select user,password from user;
```

```
+-----+-----+  
| user | password |  
+-----+-----+  
| root |          |  
|      |          |  
+-----+-----+
```

2 rows in set (0.02 sec)

idem avec le nom du host en plus

```
mysql> select user,password,host from user;
```

```
+-----+-----+-----+  
| user | password | host      |  
+-----+-----+-----+  
| root |          | localhost |  
|      |          | localhost |  
+-----+-----+-----+
```

2 rows in set (0.00 sec)

On voit ici un utilisateur **anonyme** de mysql.

Changer l'ordre et trier

```
mysql> select host, user,password from user order by user;
```

```
+-----+-----+-----+  
| host      | user | password |  
+-----+-----+-----+  
| localhost |      |          |  
| localhost | root |          |  
+-----+-----+-----+
```

Tri **descendant**

```
mysql> select host, user,password from user order by user desc;
```

```
+-----+-----+-----+  
| host      | user | password |  
+-----+-----+-----+  
| localhost | root |          |  
| localhost |      |          |  
+-----+-----+-----+
```

2 rows in set (0.02 sec)

Ceci est un exemple, cliquez sur le lien de téléchargement pour obtenir le cours complet.

