

# MySQL

## MySQL Administration

Contributed by W.J. Gilmore

1999-03-30

advertisement

### Article Index:

MySQL Administration correctly administering the MySQL server, including the manipulation the privilege system, and making intelligent decisions about the capabilities (permissions) given to users. So you've finally correctly installed the MySQL server. The next step is to configure the server to handle clients, in addition to making the server a secure one. Yet how does one go about the addition of clients, in addition to providing necessary privileges for the client's intended use of the server? This article will focus upon some of the most daunting (yet easy and important!) tasks of correctly administering the MySQL server, including the manipulation the privilege system, and making intelligent decisions about the capabilities (permissions) given to users.

I will make the assumption that you have at least a rough knowledge of SQL basics. For example, what is an INSERT, SELECT, or UPDATE command. Also, you should have a basic understanding of the Unix-flavored directory structure before continuing this article.

### Have you correctly installed MySQL?

For those who are not sure if MySQL has been correctly installed, follow along with the below commands. (Don't worry if you have no clue as to what is taking place. We will discuss these concepts throughout the article):

```
$ cd /usr/local/bin $ ./mysqlshow +-----+ | Databases | +-----+
| mysql | +-----+
```

Assuming that checks out, try:

```
$ ./mysqlshow mysql Database: mysql +-----+ | Tables | +-----+ | db | |
host | | user | +-----+
```

And finally:

```
$ ./mysql -e "select host,db,user from db" mysql +-----+ +-----+ +-----+ | host |
db | user | +-----+ +-----+ +-----+ | % | test | | | % | test_% | |
+-----+ +-----+ +-----+
```

Assuming the above tests have produced the expected output, we can now begin discussing the concepts that one must understand to fully benefit from MySQL's efficient administration system. If the results were *not* the same as those seen above:

1. Have you executed `MYSQL_INSTALL_DB`? If you have never heard of the command, check out the [MySQL documentation](#).
2. If you have in fact executed `MYSQL_INSTALL_DB`, again, [consult the docs](#), as there are a number of possible problems arising from the command that are out of the scope of this article.

Assuming everything is in order, and MySQL has in fact been installed correctly, let's take a look at the dynamics behind MySQL administration.

Administering the MySQL server involves the maintenance of the database server (hosts, users, and databases). These duties can be better termed as administering MySQL's privilege system.

The privilege system is in fact, a simple concept. Through designating certain '*privileges*', a certain user on a certain host can perform certain commands within a certain database. These privileges give a user a certain set of rules with which to use the MySQL server. For example, a user could have privileges to insert information, but not to delete it. Another user may have privileges to create tables, but cannot destroy (drop) them. These hosts, users, databases, and certain privileges are denoted within the '*host*', '*user*' and '*db*' tables respectively, otherwise known as the '*privilege*' tables.

### The privilege tables

As stated above, all privileges are stored in three tables: '*user*', '*host*' and '*db*'. A good way to look at these three tables is as an order of hierarchy:

First level: host

Second level: user

Third level: db

These tables operate much like normal MySQL tables. They are easily modified using normal `INSERT`, `UPDATE`, and `DELETE` commands. In fact, they are so easy to manipulate that all previous fears about these tables will most likely be erased after you have read the following paragraphs.

### Entering MySQL as the administrator for the first time:

Assuming you have just installed MySQL, run the following command:

---

```
$ mysql -u root mysql
```

---

This will allow you to enter the `mysql` table of the MySQL database. From here, the administrator can execute all necessary administration commands.

### The host table:

The '*host*' table determines which hosts are able to enter the MySQL server. For those running only one server, one would only have to enter two hosts, the '*localhost*' and the

actual host name of the server.

Columns contained within the 'host' table:

Host – Which host?

Db – Name of database?

Again, the following are determined with a 'Y' or 'N' (Default 'N'). These allow the host to execute certain instructions while using the specified database.

Select\_priv

Insert\_priv

Update\_priv

Delete\_priv

Create\_priv

Drop\_priv

For example, if your hostname is 'devshed', and the only intended database is 'mydb', you would do as the following to make the server accessible from the actual server, as well as the devshed hostname:

---

```
mysql> insert into    -> host(host,db,Select_priv,Insert_priv,Update_priv,    ->
Delete_priv,Create_priv,Drop_priv)    -> values('localhost','mydb','Y', 'Y', 'Y', 'Y', 'Y', 'Y');
```

---

---

```
mysql> insert into    -> host(host,db,Select_priv,Insert_priv,Update_priv,    ->
Delete_priv,Create_priv,Drop_priv) values('devshed','mydb','Y', 'Y', 'Y', 'Y', 'Y', 'Y');
```

---

**[see notes below for clarification by Monty]**

*The host table is used as an extension of the db table when you want a given db table entry to apply to several hosts. For example, if you want a user to be able to use a database from several hosts in your network, leave the Host value empty in the user's db table entry, then populate the host table with an entry for each of those hosts.*

In other words; The host table is only useful if you have a network with many hosts and you want to have completely different permissions for different hosts and you don't want to add a lot of rows into the db table for every host+user combination.

**FAQ:** In which form(s) can I enter a hostname?

A hostname may be entered as localhost, an actual hostname, an IP number, or a string containing wildcards. Thus, the following are all valid possibilities for a hostname:

195.103.124.193

incluso.com

localhost

incluso%

*Note:* It is recommended that wildcards be avoided, as they can become potential security threats. For example, someone from an unrelated site such as inclusotonno.com or

incluso.yahou.com could enter the MySQL server without problem, even though the administrator did not intend for them to gain access. Considering there are just three pieces obstacles a hacker must overcome in order to enter a site (hostname, username, password), giving the hostname away simply by inserting a wildcard might not be such a brilliant idea.

**FAQ: What is a 'localhost'?**

The localhost can be considered a synonym for hostname if the client and the server are using the same host. i.e. The same computer as that in which MySQL is installed.

**The User table:**

The '*user*' table contains all host+user combinations which are allowed to enter the MySQL server. Basically, a host+user combination could be considered a unique identity, much like a fingerprint or an id number, which tells the server exactly who is, and who is not, allowed server access. Those listed within the user table are capable of entering, and everyone else is not. Simple enough.

The user table contains data relevant to the following information:

Columns contained within the '*user*' table:

Host – From which host is the connection being made?

User – From which user?

Password – The password to make the connection?

Each of the following privileges are determined with a 'Y' or 'N' (Default 'N'):

Select\_priv

Insert\_priv

Update\_priv

Delete\_priv

Create\_priv

Drop\_priv

Reload\_priv

Shutdown\_priv

Process\_priv – Allows user to watch scrolling list of commands being executed on the server.

File\_priv – allows user to write files to the server

The last two commands should bring panic, fear, and alarm into the eyes of any security-minded administrator. A user granted the *Process\_priv* would be able to watch commands as they are executed, simply by typing `mysqladmin proc`. For example, one with this privilege could watch as a user's (or even root's) password is being modified within the user table.

*File\_priv* would grant the user permission to write files to the server itself. This is obviously not a good idea. **But**, this also allows a user to run sometimes necessary commands such as `LOAD DATA INFILE`. This is a command which quickly fills databases with a tab delimited textfile, such as one converted from an Excel database. Typing in the 20,000 records by hand would be the only alternative to using `LOAD DATA INFILE`. With that in mind, just be careful as to who is given permission to use these commands.

**The db table:**

The 'db' table contains which information pertaining to which table a host+user listed in the 'user' table is allowed to enter.

Columns contained within the 'db' table:

Host – Which host?

Db – Name of database?

User – Which user?

Again, the following are determined with a 'Y' or 'N' (Default 'N')

Select\_priv

Insert\_priv

Update\_priv

Delete\_priv

Create\_priv

Drop\_priv

We have up to this point covered the terminology and structure lying behind the privilege system. Let's put that knowledge into practice by running through a Real-Life Example. Suppose we want to allow user 'dario' to access the server via hosts 'localhost' and 'www.devshed.com', which lies on the 'dv1' host. He wants to access the database 'pasta' strictly from 'localhost', but he wants to access the database 'chicken' from both hosts. Finally, he wants to use the password 'mamamia'.

### Step 1: Set up the host table (assuming it has not yet been set up)

The host table is of considerable importance when administering larger networks, yet it needs to be configured for every server. Assuming you have just one server, you will have to insert just two hostnames, the localhost, and your server name. Otherwise, you will have to list each server that you would like to give access to the MySQL server.

---

```
$ mysql mysql mysql> insert into    -> host(host,db,Select_priv,Insert_priv,Update_priv,
    -> Delete_priv,Create_priv,Drop_priv)    -> values('localhost','%', 'Y', 'Y', 'Y', 'Y', 'Y',
'Y'); mysql> insert into    -> host(host,db,Select_priv,Insert_priv,Update_priv,    ->
Delete_priv,Create_priv,Drop_priv)    -> values('dv1','%', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y');
```

---

### Step 2: Update the 'user' table, for reason of granting access to a new host+user combination.

---

```
mysql> insert into user (host,user,password)    ->
values('localhost','dario',password('mamamia')); mysql> insert into user (host,user,password)
    -> values('www.devshed.com','dario',password('mamamia'));
```

---

### Step 3: Update the database (db) table.

---

```
mysql> insert into db    -> (host,db,user,Select_priv,Insert_priv,Update_priv,Delete_priv,
    -> Create_priv,Drop_priv)    -> values ('localhost','pasta','dario','Y','Y','Y','Y','Y','Y');
mysql> insert into db    -> (host,db,user,Select_priv,Insert_priv,Update_priv,Delete_priv,
    -> Create_priv,Drop_priv)    -> values ('%', 'chicken','dario','Y','Y','Y','Y','Y','Y');
```

---

---

**Step 4: Create the necessary tables, using mysqladmin.**

Exit from the MySQL server (q). Now, you must use a tool called mysqladmin to create the actual database.

---

```
$ mysqladmin -u root -p create pasta enter password: ***** Database "pasta" created.
```

---

Don't forget to do the same thing for the 'chicken' database!

**Step 5: Use mysqladmin again.**

After all necessary modifications have been made, the command mysqladmin reload must be executed. If you do not execute this command, the changes will not take effect.

---

```
mysql> mysqladmin -u root -p reload; enter password: *****
```

---

That's all there is to it. Dario, not working on the 'localhost' and entering the server via telnet, should be able to access the pasta and chicken databases. However, if Dario enters via host 'www.devshed.com', he will only be able to enter the chicken database. If he attempts to enter the pasta database, an error will occur.

**FAQ:** Why do I have to enter "password('mamamia')" instead of just "mamamia" into the password variable of the 'user' table?

This is because MySQL, like any security-minded server, stores the password encrypted. Thus, to allow the user to continue to use the password 'mamamia', it must be entered as above.

**FAQ:** Previously you stated that there were 10 privileges within the user table, but in the example you only listed 6. Why?

This is because the default of every privilege is 'N'. Thus, if it is not listed within the insert command, it is considered to be 'NO' ('N').

**FAQ:** What happens if I leave host or db empty?

Both 'host' and 'db' can handle wildcards. Thus, if one or the other are left empty ("), it will be entered as '%'. This is obviously dangerous, unless you really know what you are doing. Therefore, be especially careful when entering data into these tables.

**FAQ:** What if I want to delete the inserted user and db info?

Easy. Just follow the above instructions, except using the delete syntax instead of the insert. Finally, use mysqladmin to delete the database. Don't forget to execute 'mysqladmin reload' after you're done.

**FAQ:** Isn't there an easier way to do this?

Believe it or not, yes. There are a number of programs included along with the MySQL distribution (within the contrib directory), including `xmysqladmin`, `mysql_webadmin`, `mysqladmin` and even `xmysql` to modify values within the privilege tables.. For more tips you may also read our article on [MySQL Grant Tables](#)

### Using MySQLAdmin

As noted in the above example, one uses the MySQLAdmin to carry out very important administrative tasks, such as finalizing modifications on the server, and creating databases. At the UNIX command line, try typing:

---

```
$ mysqladmin
```

---

A list of commands will scroll down the screen. These commands are carried out as the 'reload' or 'create databasename'. BE CAREFUL of these commands, as they are capable of erasing or shutting down the database server. However, be sure to study these commands carefully, as they are indispensable to running MySQL administration.

You now should be able to add (and thus modify and delete information from the privilege tables. Yet how do you enter these databases, and what are some security measures you can take to keep the 'bad guys' out? This is the subject of the next section.

#### **Correct ways to enter database:**

To test your new permission setup, try to enter the database using the following commands:

---

```
$ mysql -u user database
```

---

(Assuming a password was not established.) Note: Do not establish an account without designating a password, unless you have a *very* good reason for doing so.

ex.

Assuming the username is 'dario', and database 'pasta' (Note: this would only work from the localhost, as we have designated pasta to only be reachable via localhost):

---

```
$ mysql -u dario pasta
```

---

or, if a password is set:

---

```
$ mysql -u user -p password database
```

---

ex.

Assuming the username is 'dario', password 'mamamia', and database 'chicken' (Note: this would only work from both localhost and dv1, as we have designated chicken to be accessible from both):

---

```
$ mysql -u dario -p chicken enter password: mamamia
```

---

---

If you have correctly configured the tables, you will see the following message (more or less):

---

Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 5602 to server version: 3.21.23-beta-log Type 'help' for help. mysql>

---

Assuming you have learned tons from this article ;) , and everything went smooth, great. If not, check out the following section, troubleshooting the privilege tables.

### **Troubleshooting:**

*Troubleshooting #1:*

If:

---

```
$ mysql -u username test
```

---

works, but:

---

```
$ mysql -h your_hostname -u username test
```

---

returns the famous 'Access denied' message, then the wrong hostname is entered in the 'user' table.

### **[see notes below for clarification by Monty]**

*If 'mysql -u username -h hostname databasename' give the error:*

Access to database denied

Then this means that you don't have a row with 'user=username', 'host=hostname' and 'db=databasename' in the db table.

Remember that just because your organization name may be 'devshed', the host name is 'devshed.com'! So don't type 'devshed' as the hostname and assume it will work. The hostname must be entered as 'devshed.com'!

*Troubleshooting #2:*

If you receive the 'Access denied' message after entering:

---

```
$ mysql -u user database
```

---

There is a problem with the user table. Check the data entered in the user table, and you will most likely find the answer to your problem.

### **[see notes below for clarification by Monty]**

*If you think that you have set up all privileges correct and they still doesn't work, try doing 'mysqladmin reload' and try again.*



*Troubleshooting #3:*

If a client needs to use LOAD DATA INFILE or SELECT INTO OUTFILE, but keeps getting the 'Access to database denied' command, then the File\_priv command isn't set to 'Y' within the 'User' table.

*Troubleshooting #4:*

IF within the 'user' table, a " or '%' has been inserted as the host, the server will look to the 'host' table for the specific privileges. All three privilege tables will sort by non-wild card hosts first, and then by wild-card, choosing the *first* row that fits the bill.

Let's assume we have the following within the 'user' table:

Host	User
%	root
%	dario
localhost	root
localhost	

The table will be sorted as follows:

localhost/root  
localhost/any  
any/dario  
any/root

Thus, which set of privileges will be used if Dario attempts to connect from the localhost? The one with user as dario, right? Wrong. It will use the row with localhost as host and 'any' as user. Thus, pay attention to which privileges are being assigned to users.

**[see notes below for clarification by Monty]**

*This description is slightly wrong. The host table is only accessed if the db table has a host="" entry. host='%' will match all hosts!*

**Basic Security tips: (READ: IMPORTANT)**

1. Above all, take a moment to update the root privileges, and give it a password!  
Someone managing to get on the localhost can enter simply by typing: `mysql -u root mysql`, if a password is not established.
2. Assign passwords for every user.
3. If the user doesn't have a specific need for it, don't assign permission to use File\_priv. This gives the user the ability to write files to the MySQL server. In addition, don't give permission to the process\_priv, unless there is a good reason for doing so.
4. Don't use '%' within the host names. It allows a user to enter that server from any outside host. In addition, don't use wildcards (i.e. devshed%) It would theoretically allow someone to use `http://devshed.badcracker.com` to aid them in entering the server.

Hopefully, this article has at least helped somewhat in providing new MySQL administrator's some insight into the privilege tables. I guess the best advice I can give you, however, is to

become an active member of the MySQL mailing list, and read the documentation. These two resources will greatly aid you in quickly taking control of your MySQL server. If you have any further questions, feel free to email me.