**Gestion de MySQL desde línea de comandos.**

Lo primero que tendremos que hacer es conectar con el sistema gestor de MySQL. Para ello, desde la línea de comandos invocamos a MySQL. Para ello, simplemente tenemos que escribir el comando "mysql" e indicarle unas opciones de conexión.

% mysql

Con el "%" expresamos el principio de la línea de comandos. Ese principio es el prompt que tengamos en nuestra consola de Linux o MsDOS, que puede ser algo como c:\mysql\bin>. El carácter "%", por tanto, no tenemos que escribirlo.

Con esa sentencia se conecta uno con la base de datos con los parámetros por defecto. Es decir, al servidor local, con usuario y password igual a cadenas vacías.

Lo más normal es que tengamos que indicar algún otro dato para conectar con la base de datos, como el usuario, la clave o la dirección del servidor con el que queremos conectar. La sintaxis sería la siguiente:

% mysql -h nombre\_servidor -u nombre\_usuario –p

% mysql -h nombre\_servidor -u nombre\_usuario –p –port puerto\_mysql

**En el comando mysql el puerto siempre tiene que ir al final.**

Si deseamos conectarnos a la base de datos en local y con nombre de usuario root tendríamos que escribir:

% mysql -h localhost -u root –p

% mysql -h localhost -u root –p –port 3310

% mysql -h localhost -u root –p –P 3310

Lo primero que nos preguntará será el password para el usuario root. Una vez introducida la clave, ya estaremos dentro de la línea de comandos de MySQL. Con ello el prompt cambiará a algo como esto:

mysql>

Podríamos haber indicado la contraseña directamente en la línea de comandos para iniciar la sesión, desaconsejado por razones de seguridad. La sintaxis hubiera sido:

% mysql -h localhost -u root –p mi\_clave

Nos fijamos que entre -h y el nombre del host hay un espacio en blanco, igual que entre -u y el nombre de usuario. Sin embargo, entre -p y la clave no debemos poner ningún espacio. Si li que necesitamos es conectarnos remotamente a otro equipo deberemos cambiar el localhost por la ip del servidor de mysql al que queremos conectarnos.

% mysql -h ip\_servidor\_mysql -u root –p mi\_clave

**Dentro de la consola de MySQL**

Una vez dentro, tendremos a nuestra disposición todas las sentencias de MySQL para el trabajo con la base de datos y el lenguaje SQL.

Lo más normal es que primero te tengas que conectar con una base de datos en concreto, de entre todas las que puedes tener creadas en tu servidor MySQL. Eso se hace con el comando use, seguido del nombre de la base de datos que deseas conectar.

mysql> use mibasedatos;

Esto nos conectaría con la base de datos llamada "mibasedatos".

**Nota:** Hay que fijarse que todas las sentencias dentro de la línea de comandos de MySQL acaban en ";". Si no colocamos el punto y coma, lo más seguro es que no se ejecute el comando y nos vuelva a salir el prompt para que sigamos introduciendo el comando. Si lo que queríamos era ejecutar la sentencia que habíamos escrito antes, con simplemente entrar el ";" será suficiente. Es decir, no debemos escribir de nuevo la sentencia entera, sólo el ";" y volver a apretar "enter".

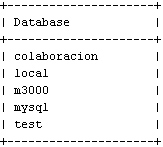
Si queremos ver una lista de las bases de datos alojadas en nuestro servidor podemos escribir el comando **show databases**. Así:

mysql>show databases;

Con esto nos mostraría una lista de las bases de datos de nuestro servidor. Algo como esto:

mysql> show databases

-> ;



5 rows in set (0.02 sec)

Si queremos crear una base datos, podremos hacerlo con el comando "create database" seguido del nombre de la nueva base de datos.

mysql> create database miprueba;

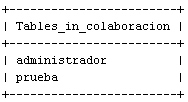
Eso nos creará una base de datos que se llama "miprueba". Como habíamos comentado, si queremos luego usar esa base de datos escribiríamos:

mysql> use miprueba;

Lógicamente, esta base de datos recién creada estará vacía, pero si estuviéramos usando una base de datos ya creada y queremos ver las tablas que tiene escribiríamos el comando "show tables".

mysql> show tables;

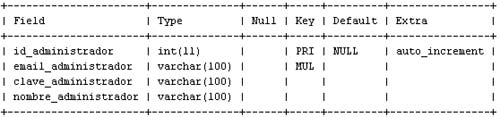
Si no hay tablas, nos dirá algo como "Empty set", pero si tenemos varias tablas dadas de alta en la base de datos que estamos usando, nos saldrá una lista de ellas:



2 rows in set (0.00 sec)

Ahora, si deseamos obtener información sobre una tabla, para saber qué campos tiene y de qué tipo, podremos utilizar el comando describe seguido del nombre de la tabla.

mysql> describe administrador;



4 rows in set (0.11 sec)

**Ejecución de sentencias DDL y DML**

Desde la consola de MySQL podemos indicar por línea de comandos todo tipo de sentencias en lenguaje SQL, como selecciones, inserciones, actualizaciones, creación de tablas, etc. El mecanismo es el que se puede deducir. Simplemente colocamos la sentencia a ejecutar seguida del punto y coma. Veamos una serie de sentencias seguidas y el resultado de ejecutarlas:

mysql> create table prueba (id\_prueba int);

Query OK, 0 rows affected (0.08 sec)

mysql> insert into prueba (id\_prueba) values (1);

Query OK, 1 row affected (0.00 sec)

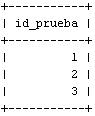
mysql> insert into prueba (id\_prueba) values (2);

Query OK, 1 row affected (0.00 sec)

mysql> insert into prueba (id\_prueba) values (3);

Query OK, 1 row affected (0.00 sec)

mysql> select \* from prueba;



3 rows in set (0.00 sec)

En definitiva, podemos ejecutar todas las sentencias que se han aprendido en el [manual del lenguaje SQL](http://www.desarrolloweb.com/manuales/9/)

**Para salir de la línea de comandos de MySQL**

Una vez hemos terminado de trabajar con MySQL, si queremos cerrar la conexión con el servidor, simplemente escribimos "quit" desde el prompt de MySQL:

mysql> quit

### Instrucciones SQL para la gestión de las estructuras.

Quien tenga el trabajo de crear scripts y programas que interactúan con MySQL seguramente está familiarizado con estas peticiones, pero ya que utilizo mi propio sitio web como referencia pondré aquí algunas peticiones comunes.

**Listar tablas existentes en la base de datos**

show tables;

**Mostrar información de las tablas en la base de datos**

show tables solo nos mostrará los nombres de las tablas en la base de datos, para ver toda la información sobre las tablas, usa:

show table status;

La información presentada sobre la tabla es:

* Name - El nombre de la tabla
* Engine - Mecanismo de la tabla (MyISAM, InnoDB, Memory, CVS, etc.)
* Version - Número de versión del archivo .frm de la tabla
* Row\_format - El formato de almacenamiento de las filas (Dinámico, Redundante, etc.)
* Rows - Número de filas en la tabla
* Avg\_row\_length - Longitud promedio de las filas
* Data\_length - Longitud del archivo de datos
* Max\_data\_length - La máxima longitud del archivo de datos
* Index\_length - La longitud del archivo índice
* Data\_free - Número de bytes asignados pero no usados
* Auto\_increment - El próximo valor de auto-incremento
* Create\_time - Cuando fue creada la tabla
* Update\_time - Cuando fue actualizado el archivo de datos por última vez
* Check\_time - Cuando fue revisada la tabla por última vez
* Collation - El set de caracteres y la colación de la tabla
* Checksum - El checksum vivo
* Create\_options - Opciones extras utilizadas cuando fue creada la tabla
* Comment - El comentario de la tabla

Podemos especificar de que tabla queremos ver la información utilizando:

show table status like 'nombre\_de\_la\_tabla';

Y podemos buscar en otro campo por cierto valor, por ejemplo, para mostrar todas las tablas que usan el mecanismo de almacenamiento MyISAM, podemos utilizar:

show table status where `Engine` like 'MyISAM';

**Crear una base de datos nueva**

La crecación de schemas es muy sencilla mediante el uso de scripts;

create database nombre\_base\_de\_datos;

create schema nombre\_base\_de\_datos;

Una vez creada, si quiero cambiar el ‘foco’ a dicha base de de datos para que cualquier comando que ejecutemos tenga efecto sobre ella, deberemos ejecutar:

use nombre\_base\_de\_datos;

El uso de este comando es muy importante para poder trabajar de forma indistinta con varias bases de datos dentro de un script.

**Crear una nueva tabla**

Este es un ejemplo del comando para crear una tabla, agregué muchos campos diferentes para la referencia de como declararlos.

create table `nombre\_base\_de\_datos`.`tabla\_de\_prueba` (  
`campo\_id` int( 11 ) unsigned not null auto\_increment comment 'clave primaria',  
`campo\_indice1` int( 11 ) unsigned not null comment 'un índice',  
`campo\_indice2` int( 11 ) unsigned not null comment 'un índice',  
`campo\_indice3` int( 11 ) unsigned not null comment 'un índice',  
`campo\_unico1` int( 11 ) unsigned not null comment 'un campo único',  
`campo\_unico2` int( 11 ) unsigned not null comment 'un campo único',  
`campo\_unico3` int( 11 ) unsigned not null comment 'un campo único',  
`campo\_varchar` varchar( 100 ) not null comment 'un campo varchar',  
`campo\_date` date not null comment 'un campo date',  
`campo\_datetime` datetime not null comment 'un campo datetime',  
`campo\_float` float not null comment 'un campo float',  
`campo\_longtext` longtext not null comment 'un campo longtext',  
`campo\_bool` bool not null comment 'un campo bool',  
`campo\_char` char( 1 ) not null comment 'un campo char',  
`campo\_tinyint` tinyint not null comment 'un campo tinyint',  
primary key ( `campo\_id` ) ,  
index ( `campo\_indice1` , `campo\_indice2`, `campo\_indice3` ) ,  
unique ( `campo\_unico1` , `campo\_unico2`, `campo\_unico3`)  
) engine = myisam character set utf8 collate utf8\_general\_ci comment =  
'comentarios de tabla';

**Listar los campos en una tabla**

show columns from `tabla\_de\_prueba`;

**Cambiar el nombre de un campo en una tabla**

alter table `tabla\_de\_prueba` change `campo\_indice\_1` `nuevo\_nombre\_campo` int(11) unsigned not null;

**Agregar un campo a una tabla y hacerlo un índice**

alter table `tabla\_de\_prueba` add `nuevo\_campo\_indice` int(11) unsigned not null, add index(`nuevo\_campo\_indice`);

**Quitar un índice de una tabla**

alter table `tabla\_de\_prueba` drop index `nuevo\_campo\_indice`;

**Quitar un campo de una tabla**

alter table `tabla\_de\_prueba` drop `nuevo\_campo\_indice`;

**Agregar campos después de un campo especificado**

alter table `tabla\_de\_prueba` add `a\_borrar` varchar(12) not null after `campo\_date`;

**Agregar campos al comienzo de la tabla**

alter table `tabla\_de\_prueba` add `a\_borrar\_2` varchar(12) not null first;

**Agregar múltiples campos a la tabla**

alter table `tabla\_de\_prueba` add `a\_borrar\_3` varchar(12) not null after `a\_borrar`, add `a\_borrar\_4` varchar(12) not null after `a\_borrar\_3`;

**Borrar campos en una tabla**

alter table `tabla\_de\_prueba` drop `a\_borrar`, drop `a\_borrar\_2`, drop `a\_borrar\_3`, drop `a\_borrar\_4`;

**Renombrar una tabla**

rename table `nombre\_base\_de\_datos`.`nombre\_original` to `nombre\_base\_de\_datos`.`nuevo\_nombre`;

**Cambiar el comentario de una tabla**

alter table `tabla\_de\_prueba` comment='Los comentarios';

**Cambiar el valor de auto incremento de una tabla**

alter table `tabla\_de\_prueba` auto\_increment=3;

**Borrar todos los registros en una tabla**

truncate table `tabla\_de\_prueba`;

deletre from ` tabla\_de\_prueba `;

**Borrar una tabla**

drop table `tabla\_de\_prueba`;

## ¿Cómo crear un nuevo usuario?

Vamos empezando por crear un usuario nuevo desde la consola de MySQL:

CREATE USER 'nombre\_usuario'@'localhost' IDENTIFIED BY 'contrasena';

CREATE USER 'nombre\_usuario'@'%' IDENTIFIED BY 'contrasena';

Lamentablemente, a este punto el nuevo usuario no tiene permisos para hacer algo con las bases de datos. Por consecuencia si el usuario intenta identificarse (con la contraseña establecida), no será capaz de acceder a la consola de MySQL.

Por ello, lo primero que debemos hacer es proporcionarle el acceso requerido al usuario con la información que requiere.

GRANT ALL PRIVILEGES ON \*.\* TO 'nombre\_usuario'@'localhost';

GRANT ALL PRIVILEGES ON DAI.\* TO 'alumno'@'%';

Los asteriscos en este comando hacen referencia a la base de datos y la tabla (respectivamente) a la cual el nuevo usuario tendrá acceso; específicamente este comando permite al usuario leer, editar, ejecutar y realizar todas las tareas en todas las bases de datos y tablas.

Una vez que has finalizado con los permisos que deseas configurar para tus nuevos usuarios, hay que asegurarse siempre de refrescar todos los privilegios.

FLUSH PRIVILEGES;

Tus cambios ahora surtirán efecto.

## ¿Cómo otorgar permisos de usuario diferentes?

Una pequeña lista del resto de los posibles permisos que los usuarios pueden tener:

* ALL PRIVILEGES: como mencionamos previamente esto permite a un usuario de MySQL acceder a todas las bases de datos asignadas en el sistema.
* CREATE: permite crear nuevas tablas o bases de datos.
* DROP: permite eliminar tablas o bases de datos.
* DELETE: permite eliminar registros de tablas.
* INSERT: permite insertar registros en tablas.
* SELECT: permite leer registros en las tablas.
* UPDATE: permite actualizar registros seleccionados en tablas.

GRANT OPTION: permite asignar privilegios de usuarios.

REVOKE OPTION: permite remover privilegios de usuarios.

Para proporcionar un permiso a usuario específico, puedes utilizar ésta estructura:

GRANT [permiso] ON [nombre de bases de datos].[nombre de tabla] TO ‘[nombre de usuario]’@'localhost’;

Si deseas darle acceso a cualquier base de datos o tabla, asegúrate de insertar un asterisco en lugar del nombre de la base de datos o tabla.

Cada vez que tu actualizas o cambias permisos, asegúrate de refrescar los privilegios mediante FLUSH PRIVILEGES;.

Si necesitas remover un permiso, la estructura es casi idéntica a la que los asigna:

REVOKE [permiso] ON [nombre de base de datos].[nombre de tabla] FROM ‘[nombre de usuario]’@‘localhost’;

Así como puedes borrar bases de datos con DROP, también puedes usar el comando DROP para borrar usuarios:

DROP USER ‘usuario\_prueba’@‘localhost’;

Para probar el nuevo usuario, debes cerrar sesión escribiendo quit y volviendo a iniciar sesión con éste comando en la consola:

mysql -u [nombre de usuario]-p