

**El comando mysqldump del sistema gestor de base de datos MySQL sirve para hacer copias de seguridad. Lo explicamos con detenimiento y vemos sus opciones más importantes...**

**Fuente** <http://www.desarrolloweb.com>

Este comando permite hacer la copia de seguridad de una o múltiples bases de datos. Además permite que estas copias de seguridad se puedan restaurar en distintos tipos de gestores de bases de datos, sin la necesidad de que se trate de un gestor de mysql. Esto lo consigue creando unos ficheros, que contienen todas las sentencias sql necesarias para poder restaurar la tabla, que incluyen desde la sentencia de creación de la tabla, hasta una sentencia insert por cada uno de los registros que forman parte de la misma.

El comando dispone de una amplia variedad de opciones que nos permitirá realizar la copia de la forma más conveniente para el propósito de la misma.

Para poder restaurar la copia de seguridad, bastará con ejecutar todas las sentencias sql que se encuentran dentro del fichero, bien desde la línea de comandos de mysql, o desde la pantalla de creación de sentencias sql de cualquier entorno gráfico como puede ser el Mysql Control Center.

Las limitaciones de la restauración dependerán de las opciones que se han especificado a la hora de hacer la copia de seguridad, por ejemplo, si se incluye la opción --add-drop-table al hacer la copia de seguridad, se podrán restaurar tablas que existen actualmente en el servidor (borrándolas primero). Por lo que es necesario estudiar primero los procedimientos que se utilizarán tanto en la copia como en la restauración, para que todo salga correcto!

**Algunas de las opciones que tiene son:**

### **--add-locks**

Añade LOCK TABLES antes, y UNLOCK TABLE después de la copia de cada tabla.

### **--add-drop-table**

Añade un drop table antes de cada sentencia create

### **-A, --all-databases**

Copia todas las bases de datos. Es lo mismo que utilizar --databases seleccionando todas.

### **-a, --all**

Incluye todas las opciones de creación específicas de Mysql.

### **--allow-keywords**

Permite la creación de nombres de columnas que son palabras clave, esto se realiza poniendo de prefijo a cada nombre de columna, el nombre de la tabla

### **-c, --complete-insert**

Utiliza inserts incluyendo los nombres de columna en cada sentencia (incrementa bastante el tamaño del fichero)

### **-C, --compress**

Comprime la información entre el cliente y el servidor, si ambos soportan compresión.

### **-B, --databases**

Para copiar varias bases de datos. En este caso, no se especifican tablas. El nombre de los argumentos se refiere a los nombres de las bases de datos. Se incluirá USE db\_name en la salida antes de cada base de datos.

### **--delayed**

Inserta las filas con el comando INSERT DELAYED.

### **-e, --extended-insert**

Utiliza la sintaxis de INSERT multilinea. (Proporciona sentencias de insert más compactas y

rápidas.)

### **-, --debug[=option\_string]**

Utilización de la traza del programa (para depuración).

### **--help**

Muestra mensaje de ayuda y termina.

### **--fields-terminated-by=...**

**--fields-enclosed-by=...**

**--fields-optionally-enclosed-by=...**

**--fields-escaped-by=...**

**--lines-terminated-by=...**

Estas opciones se utilizan con la opción -T y tienen el mismo significado que la correspondiente cláusula LOAD DATA INFILE.

### **-F, --flush-logs**

Escribe en disco todos los logs antes de comenzar con la copia

### **-f, --force,**

Continúa aunque se produzca un error de SQL durante la copia.

### **-h, --host=..**

Copia los datos del servidor de Mysql especificado. El servidor por defecto es localhost.

### **-l, --lock-tables.**

Bloquea todas las tablas antes de comenzar con la copia. Las tablas se bloquean con READ LOCAL para permitir inserts concurrentes en caso de las tablas MyISAM. Cuando se realiza la copia de múltiples bases de datos, --lock-tables bloqueará la copia de cada base de datos por separado. De forma que esta opción no garantiza que las tablas serán consistentes lógicamente entre distintas bases de datos. Las tablas en diferentes bases de datos se copiarán en estados completamente distintos.

### **-K, --disable-keys**

Se incluirá en la salida `/*!40000 ALTER TABLE tb_name DISABLE KEYS */;` y `/*!40000 ALTER TABLE tb_name ENABLE KEYS */;` Esto hará que carga de datos en un servidor MySQL 4.0 se realice más rápido debido a que los índices se crearán después de que todos los datos hayan sido restaurados.

### **-n, --no-create-db**

No se incluirá en la salida `CREATE DATABASE /*!32312 IF NOT EXISTS*/ db_name;` Esta línea se incluye si la opción `--databases` o `--all-databases` fue seleccionada.

### **-t, --no-create-info**

No incluirá la información de creación de la tabla (sentencia `CREATE TABLE`).

### **-d, --no-data**

No incluirá ninguna información sobre los registros de la tabla. Esta opción sirve para crear una copia de sólo la estructura de la base de datos.

### **--opt**

Lo mismo que `--quick --add-drop-table --add-locks --extended-insert --lock-tables`. Esta opción le debería permitir realizar la copia de seguridad de la base de datos de la forma más rápida y efectiva.

### **-pyour\_pass, --password[=your\_pass]**

Contraseña utilizada cuando se conecta con el servidor. Si no se especifica, `'=your_pass'`, `mysqldump` preguntará la contraseña.

### **-P, --port=...**

Puerto utilizado para las conexiones TCP/IP

### **--protocol=(TCP | SOCKET | PIPE | MEMORY)**

Especifica el protocolo de conexión que se utilizará.

### **-q, --quick**

No almacena en el buffer la sentencia, la copia directamente a la salida. Utiliza `mysql_use_result()` para realizarlo.

### **-Q, --quote-names**

Entrecomilla las tablas y nombres de columna con los caracteres ``'`'.

### **-r, --result-file=...**

Redirecciona la salida al fichero especificado. Esta opción se debería utilizar en MSDOS, porque previene la conversión de nueva línea ``n'` en nueva línea y retorno de carro ``nr'`.

### **--single-transaction**

Utiliza el comando `BEGIN` antes de realizar la copia desde el servidor. Es muy útil con las tables InnoDB y el nivel de transacción `READ_COMMITTED`, porque en este modo realizará la copia de seguridad en un estado consistente sin necesidad de bloquear las aplicaciones. Consultar el manual para más detalles.

### **-S /path/to/socket, --socket=/path/to/socket**

El fichero de sockets que se especifica al conectar al localhost (que es el host predeterminado).

### **--tables**

sobreescribe la opción `--databases (-B)`.

### **-T, --tab=path-to-some-directory**

Crea un fichero `table_name.sql`, que contiene la sentencia de creación de SQL, y un fichero `table_name.txt`, que contiene los datos de cada tabla. El formato del fichero ``.txt'` se realiza de acuerdo con las opciones `--fields-xxx` y `--lines--xxx options`. Nota: Esta opción sólo funciona si el comando `mysqldump` se ejecuta en la misma máquina que el demonio `mysqld`, el usuario deberá tener permisos para crear y escribir el fichero en la ubicación especificada

### **-u nombre\_usuario, --user=nombre\_usuario**

El nombre de usuario que se utilizará cuando se conecte con el servidor, el valor predeterminado es el del usuario actual.

### **-v, --verbose**

Va mostrando información sobre las acciones que se van realizando (más lento)

### **-w, --where='cláusula where'**

Sirve para realizar la copia de determinados registros

### **-X, --xml**

Realiza la copia de seguridad en un documento xml

### **-x, --first-slave**

Bloquea todas las tablas de todas las bases de datos

## **Ejemplos de comandos mysqldump:**

Para realizar la copia de seguridad de la base de datos mibase al fichero copia\_seguridad.sql

```
mysqldump --opt mibase > copia_seguridad.sql
```

Otro ejemplo más complejo de comando mysqldump para hacer el backup de una base de datos es el siguiente:

```
mysqldump --opt --password=miclave --user=miuser mibasededatos > archivo.sql
```

En este último caso estamos indicando un nombre de usuario y una clave para acceder a la base de datos sobre la que se está haciendo el backup: mibasededatos. Las sentencias SQL para reconstruir esa base de datos se volcarán en el fichero archivo.sql.

## **Restaurar la base de datos**

Si deseamos recuperar la información de un fichero para restaurar una copia de seguridad de la base de datos lo haremos con el comando mysql. Utilizaremos una sintaxis como esta:

```
mysql mibase < archivo.sql
```

En este ejemplo se restauraría la base de de datos mibase con el backup almacenado en el fichero archivo.sql.

Otro ejemplo más complejo de comando para restaurar una base de datos es el siguiente:

```
mysql --password=miclave --user=miuser mibase < archivo.sql
```

Es el mismo ejemplo que el anterior, pero indicando un nombre de usuario y una clave con las que acceder a la base de datos mibase.