MySQLdump Reference Manual

Synopsis

`mysqldump [options] [db_name [tbl_name …]]`

Description

The **mysqldump** client can be used to dump a database or a collection of databases for backup or for transferring the data to another SQL server (not necessarily a MySQL server). The dump contains SQL statements to create the table and/or populate the table.

If you are doing a backup on the server, and your tables all are `MyISAM` tables, you could consider using the **mysqlhotcopy** instead since faster backups and faster restores can be accomplished with the latter. See mysqlhotcopy(1).

There are three general ways to invoke **mysqldump**:

```
shell> mysqldump [options] db_name [tables]
shell> mysqldump [options] --databases DB1 [DB2 DB3...]
shell> mysqldump [options] --all-databases
```

If you do not name any tables or use the `--databases` or `--all-databases` option, entire databases are dumped.

To get a list of the options your version of **mysqldump** supports, execute **mysqldump --help**.

If you run **mysqldump** without the `--quick` or `--opt` option, **mysqldump** loads the whole result set into memory before dumping the result. This probably is a problem if you are dumping a big database. In MySQL 5.1, `--opt` is enabled by default, but can be disabled with `--skip-opt`.

If you are using a recent copy of the **mysqldump** program to generate a dump to be reloaded into a very old MySQL server, you should not use the `--opt` or `-e` options.

mysqldump supports the following options:

- `--help`, `-?`

  Display a help message and exit.

- `--add-drop-database`

  Add a `DROP DATABASE` statement before each `CREATE DATABASE` statement.

- `--add-drop-table`

  Add a `DROP TABLE` statement before each `CREATE TABLE` statement.

⌘ `--add-locks`

Surround each table dump with `LOCK TABLES` and `UNLOCK TABLES` statements. This results in faster inserts when the dump file is reloaded. See the section called “Speed of `INSERT` Statements”.

⌘ `--all-databases`, `-A`

Dump all tables in all databases. This is the same as using the `--databases` option and naming all the databases on the command line.

⌘ `--allow-keywords`

Allow creation of column names that are keywords. This works by prefixing each column name with the table name.

⌘ `--comments[={0|1}]`

If set to `0`, suppresses additional information in the dump file such as program version, server version, and host. `--skip-comments` has the same effect as `--comments=0`. The default value is `1`, which includes the extra information.

⌘ `--compact`

Produce less verbose output. This option suppresses comments and enables the `--skip-add-drop-table`, `--no-set-names`, `--skip-disable-keys`, and `--skip-add-locks` options.

⌘ `--compatible=`*`name`*

Produce output that is more compatible with other database systems or with older MySQL servers. The value of `name` can be `ansi`, `mysql323`, `mysql40`, `postgresql`, `oracle`, `mssql`, `db2`, `maxdb`, `no_key_options`, `no_table_options`, or `no_field_options`. To use several values, separate them by commas. These values have the same meaning as the corresponding options for setting the server SQL mode. See the section called “The Server SQL Mode”.

This option does not guarantee compatibility with other servers. It only enables those SQL mode values that are currently available for making dump output more compatible. For example, `--compatible=oracle` does not map data types to Oracle types or use Oracle comment syntax.

⌘ `--complete-insert`, `-c`

Use complete `INSERT` statements that include column names.

⌘ `--compress`, `-C`

Compress all information sent between the client and the server if both support compression.

⌘ `--create-options`

Include all MySQL-specific table options in the `CREATE TABLE` statements.

⌘ `--databases`, `-B`

Dump several databases. Normally, **mysqldump** treats the first name argument on the

command line as a database name and following names as table names. With this option, it treats all name arguments as database names. `CREATE DATABASE IF NOT EXISTS` *`db_name`* and `USE` *`db_name`* statements are included in the output before each new database.

⌘ `--debug[=`*`debug_options`*`]`, `-# [`*`debug_options`*`]`

Write a debugging log. The *`debug_options`* string is often `'d:t:o,`*`file_name`*`'`.

⌘ `--default-character-set=`*`charset`*

Use *`charset`* as the default character set. See the section called "The Character Set Used for Data and Sorting". If not specified, **mysqldump** in MySQL 5.1 uses `utf8`.

⌘ `--delayed-insert`

Insert rows using `INSERT DELAYED` statements.

⌘ `--delete-master-logs`

On a master replication server, delete the binary logs after performing the dump operation. In MySQL 5.1, this option automatically enables `--master-data`.

⌘ `--disable-keys`, `-K`

For each table, surround the `INSERT` statements with `/*!40000 ALTER TABLE` *`tbl_name`* `DISABLE KEYS */;` and `/*!40000 ALTER TABLE` *`tbl_name`* `ENABLE KEYS */;` statements. This makes loading the dump file faster because the indexes are created after all rows are inserted. This option is effective for `MyISAM` tables only.

⌘ `--extended-insert`, `-e`

Use multiple-row `INSERT` syntax that include several `VALUES` lists. This results in a smaller dump file and speeds up inserts when the file is reloaded.

⌘ `--fields-terminated-by=...`, `--fields-enclosed-by=...`, `--fields-optionally-enclosed-by=...`, `--fields-escaped-by=...`, `--lines-terminated-by=...`

These options are used with the `-T` option and have the same meaning as the corresponding clauses for `LOAD DATA INFILE`. See the section called "`LOAD DATA INFILE` Syntax".

⌘ `--first-slave`, `-x`

Deprecated, now renamed to `--lock-all-tables`.

- `--flush-logs`, `-F`

  Flush the MySQL server log files before starting the dump. This option requires the `RELOAD` privilege. Note that if you use this option in combination with the `--all-databases` (or `-A`) option, the logs are flushed *for each database dumped*. The exception is when using `--lock-all-tables` or `--master-data`: In this case, the logs are flushed only once, corresponding to the moment that all tables are locked. If you want your dump and the log flush to happen at exactly the same moment, you should use `--flush-logs` together with either `--lock-all`

  or `--master-data`.

- `--force`, `-f`

  Continue even if an SQL error occurs during a table dump.

- `--host=`*`host_name`*, `-h` *`host_name`*

  Dump data from the MySQL server on the given host. The default host is `localhost`.

- `--hex-blob`

  Dump binary string columns using hexadecimal notation (for example, `'abc'` becomes `0x616263`). The affected columns in MySQL 5.1 are `BINARY`, `VARBINARY`, `BLOB`.

- `--lock-all-tables`, `-x`

  Lock all tables across all databases. This is achieved by acquiring a global read lock for the duration of the whole dump. This option automatically turns off `--single-transaction` and `--lock-tables`.

- `--lock-tables`, `-l`

  Lock all tables before starting the dump. The tables are locked with `READ LOCAL` to allow concurrent inserts in the case of `MyISAM` tables. For transactional tables such as `InnoDB` and `BDB`, `--single-transaction` is a much better option, because it does not need to lock the tables at all.

  Please note that when dumping multiple databases, `--lock-tables` locks tables for each database separately. So, this option does not guarantee that the tables in the dump file are logically consistent between databases. Tables in different databases may be dumped in completely different states.

- `--master-data[=`*`value`*`]`

  This option causes the binary log position and filename to be written to the output. This option requires the `RELOAD` privilege and the binary log must be enabled. If the option value is equal to 1, the position and filename are written to the dump output in the form of a `CHANGE MASTER` statement that makes a slave server start from the correct position in the master's binary logs if you use this SQL dump of the master to set up a slave. If the option value is equal to 2, the `CHANGE MASTER` statement is written as an SQL comment. This is the default action if *`value`* is omitted.

The `--master-data` option turns on `--lock-all-tables`, unless `--single-transaction` also is specified (in which case, a global read lock is only acquired a short time at the beginning of the dump. See also the description for `--single-transaction`. In all cases, any action on logs happens at the exact moment of the dump. This option automatically turns off `--lock-tables`.

- ⌘ `--no-create-db`, `-n`

  This option suppresses the `CREATE DATABASE /*!32312 IF NOT EXISTS*/ db_name` statements that are otherwise included in the output if the `--databases` or `--all-databases` option is given.

- ⌘ `--no-create-info`, `-t`

  Do not write `CREATE TABLE` statements that re-create each dumped table.

- ⌘ `--no-data`, `-d`

  Do not write any row information for the table. This is very useful if you want to get a dump of only the structure for a table.

- ⌘ `--opt`

  This option is shorthand; it is the same as specifying `--add-drop-table --add-locks --create-options --disable-keys --extended-insert --lock-tables --quick --set-charset`. It should give you a fast dump operation and produce a dump file that can be reloaded into a MySQL server quickly. *In MySQL 5.1, `--opt` is on by default, but can be disabled with `--skip-opt`*. To disable only certain of the options enabled by `--opt`, use their `--skip` forms; for example, `--skip-add-drop-table` or `--skip-quick`.

- ⌘ `--password[=`*`password`*`]`, `-p[`*`password`*`]`

  The password to use when connecting to the server. If you use the short option form (`-p`), you *cannot* have a space between the option and the password. If you omit the *`password`* value following the `--password` or `-p` option on the command line, you are prompted for one.

- ⌘ `--port=`*`port_num`*, `-P` *`port_num`*

  The TCP/IP port number to use for the connection.

- ⌘ `--protocol={TCP | SOCKET | PIPE | MEMORY}`

  The connection protocol to use.

- ⌘ `--quick`, `-q`

  This option is useful for dumping large tables. It forces **mysqldump** to retrieve rows for a table from the server a row at a time rather than retrieving the entire row set and buffering it in memory before writing it out.

⌘ `--quote-names`, `-Q`

Quote database, table, and column names within '`' characters. If the server SQL mode includes the `ANSI_QUOTES` option, names are quoted within '"' characters. In MySQL 5.1, `--quote-names` is on by default. It can be disabled with `--skip-quote-names`, but this option should be given after any option such as `--compatible` that may enable `--quote-names`.

⌘ `--result-file=`*`file`*, `-r` *`file`*

Direct output to a given file. This option should be used on Windows, because it prevents newline '`\n`' characters from being converted to '`\r\n`' carriage return/newline sequences.

⌘ `--set-charset`

Add `SET NAMES` *`default_character_set`* to the output. This option is enabled by default. To suppress the `SET NAMES` statement, use `--skip-set-charset`.

⌘ `--single-transaction`

This option issues a `BEGIN` SQL statement before dumping data from the server. It is useful only with transactional tables such as `InnoDB` and `BDB`, because then it dumps the consistent state of the database at the time when `BEGIN` was issued without blocking any applications.

When using this option, you should keep in mind that only `InnoDB` tables are dumped in a consistent state. For example, any `MyISAM` or `HEAP` tables dumped while using this option may still change state.

The `--single-transaction` option and the `--lock-tables` option are mutually exclusive, because `LOCK TABLES` causes any pending transactions to be committed implicitly.

To dump big tables, you should combine this option with `--quick`.

⌘ `--socket=`*`path`*, `-S` *`path`*

The socket file to use when connecting to `localhost` (which is the default host).

⌘ `--skip-comments`

See the description for the `--comments` option.

⌘ `--tab=`*`path`*, `-T` *`path`*

Produce tab-separated data files. For each dumped table, **mysqldump** creates a *`tbl_name`*`.sql` file that contains the `CREATE TABLE` statement that creates the table, and a *`tbl_name`*`.txt` file that contains its data. The option value is the directory in which to write the files.

By default, the `.txt` data files are formatted using tab characters between column values and a newline at the end of each line. The format can be specified explicitly using the `--fields-`*`xxx`* and `--lines--`*`xxx`* options.

Note: This option should be used only when **mysqldump** is run on the same machine as the **mysqld** server. You must have the `FILE` privilege, and the server must have permission to write files in the directory that you specify.

⌘ `--tables`

Override the `--databases` or `-B` option. All arguments following the option are regarded as table names.

⌘ `--triggers`

Dump triggers for each dumped table. This option is on by default; disable it with `--skip-triggers`.

⌘ `--tz-utc`

Add `SET TIME_ZONE='+00:00'` to the dump file so that `TIMESTAMP` columns can be dumped and reloaded between servers in different time zones. (Without this option, `TIMESTAMP` columns are dumped and reloaded in the local time zones of the source and destination servers.) `--tz-utc` also also protects against changes due to daylight saving time. `--tz-utc` is enabled by default. To disable it, use `--skip-tz-utc`. This option was added in MySQL 5.1.2.

⌘ `--user=`*`user_name`*, `-u` *`user_name`*

The MySQL username to use when connecting to the server.

⌘ `--verbose`, `-v`

Verbose mode. Print out more information on what the program does.

⌘ `--version`, `-V`

Display version information and exit.

⌘ `--where='`*`where-condition`*`'`, `-w '`*`where-condition`*`'`

Dump only records selected by the given `WHERE` condition. Note that quotes around the condition are mandatory if it contains spaces or characters that are special to your command interpreter.

Examples:

```
"--where=user='jimf'"
"-wuserid>1"
"-wuserid<1"
```

⌘ `--xml`, `-X`

Write dump output as well-formed XML.

You can also set the following variables by using `--`*`var_name`*`=`*`value`* options:

⌘ `max_allowed_packet`

The maximum size of the buffer for client/server communication. In MySQL 5.1, the value of this variable can be up to 1GB.

- ⌘ `net_buffer_length`

  The initial size of the buffer for client/server communication. When creating multiple-row-insert statements (as with option `--extended-insert` or `--opt`), **mysqldump** creates rows up to `net_buffer_length` length. If you increase this variable, you should also ensure that the `net_buffer_length` variable in the MySQL server is at least this large. It is also possible to set variables by using `--set-variable=`*`var_name`*`=`*`value`* or `-O` *`var_name`*`=`*`value`* syntax. However, this syntax now deprecated.

The most common use of **mysqldump** is probably for making a backup of an entire database:

```
shell> mysqldump --opt db_name > backup-file.sql
```

You can read the dump file back into the server like this:

```
shell> mysql db_name < backup-file.sql
```

Or like this:

```
shell> mysql -e "source /path-to-backup/backup-file.sql" db_name
```

mysqldump is also very useful for populating databases by copying data from one MySQL server to another:

```
shell> mysqldump --opt db_name | mysql --host=remote_host -C db_name
```

It is possible to dump several databases with one command:

```
shell> mysqldump --databases db_name1 [db_name2 ...] > my_databases.sql
```

If you want to dump all databases, use the `--all-databases` option:

```
shell> mysqldump --all-databases > all_databases.sql
```

If tables are stored in the `InnoDB` storage engine, `mysqldump` provides a way of making an online backup of these (see command below). This backup just needs to acquire a global read lock on all tables (using `FLUSH TABLES WITH READ LOCK`) at the beginning of the dump. As soon as this lock has been acquired, the binary log coordinates are read and lock is released. So if and only if one long updating statement is running when the `FLUSH...` is issued, the MySQL server may get stalled until that long statement finishes, and then the dump becomes lock-free. So if the MySQL server receives only short (in the sense of "short execution time") updating statements, even if there are plenty of them, the initial lock period should not be noticeable.

```
shell> mysqldump --all-databases --single-transaction > all_databases.sql
```

For point-in-time recovery (also known as “roll-forward”, when you need to restore an old backup and replay the changes which happened since that backup), it is often useful to rotate the binary log (see the section called “The Binary Log”) or at least know the binary log coordinates to which the dump corresponds:

```
shell> mysqldump --all-databases --master-data=2 > all_databases.sql
or
shell> mysqldump --all-databases --flush-logs --master-data=2 > all_databases.sql
```

The simultaneous use of `--master-data` and `--single-transaction` provides a convenient way to make an online backup suitable for point-in-time recovery if tables are stored in the `InnoDB` storage engine.