4.5.4 **mysqldump** — A Database Backup Program

The **mysqldump** client utility performs logical backups, producing a set of SQL statements that can be executed to reproduce the original database object definitions and table data. It dumps one or more MySQL databases for backup or transfer to another SQL server. The **mysqldump** command can also generate output in CSV, other delimited text, or XML format.

mysqldump requires at least the `SELECT` privilege for dumped tables, `SHOW VIEW` for dumped views, `TRIGGER` for dumped triggers, and `LOCK TABLES` if the `--single-transaction` option is not used. Certain options might require other privileges as noted in the option descriptions.

To reload a dump file, you must have the privileges required to execute the statements that it contains, such as the appropriate `CREATE` privileges for objects created by those statements.

mysqldump output can include `ALTER DATABASE` statements that change the database collation. These may be used when dumping stored programs to preserve their character encodings. To reload a dump file containing such statements, the `ALTER` privilege for the affected database is required.

> **Note**
>
> A dump made using PowerShell on Windows with output redirection creates a file that has UTF-16 encoding:
>
> ```
> shell> mysqldump [options] > dump.sql
> ```
>
> However, UTF-16 is not permitted as a connection character set (see Impermissible Client Character Sets), so the dump file will not load correctly. To work around this issue, use the `--result-file` option, which creates the output in ASCII format:
>
> ```
> shell> mysqldump [options] --result-file=dump.sql
> ```

Performance and Scalability Considerations

`mysqldump` advantages include the convenience and flexibility of viewing or even editing the output before restoring. You can clone databases for development and DBA work, or produce slight variations of an existing database for testing. It is not intended as a fast or scalable solution for backing up substantial amounts of data. With large data sizes, even if the backup step takes a reasonable time, restoring the data can be very slow because replaying the SQL statements involves disk I/O for insertion, index creation, and so on.

For large-scale backup and restore, a physical backup is more appropriate, to copy the data files in their original format that can be restored quickly:

- If your tables are primarily `InnoDB` tables, or if you have a mix of `InnoDB` and `MyISAM` tables, consider using the **mysqlbackup** command of the MySQL Enterprise Backup product. (Available as part of the Enterprise subscription.) It provides the best performance for `InnoDB` backups with minimal disruption; it can also back up tables from `MyISAM` and other storage engines; and it provides a number of convenient options to accommodate different backup scenarios. See Section 30.2, “MySQL Enterprise Backup Overview”.

mysqldump can retrieve and dump table contents row by row, or it can retrieve the entire content from a table and buffer it in memory before dumping it. Buffering in memory can be a problem if you are dumping large tables. To dump tables row by row, use the `--quick` option (or `--opt`, which enables `--quick`). The `--opt` option (and hence `--quick`) is enabled by default, so to enable memory buffering, use `--skip-quick`.

If you are using a recent version of **mysqldump** to generate a dump to be reloaded into a very old MySQL server, use the `--skip-opt` option instead of the `--opt` or `--extended-insert` option.

For additional information about **mysqldump**, see Section 7.4, “Using mysqldump for Backups”.

Invocation Syntax

There are in general three ways to use **mysqldump**—in order to dump a set of one or more tables, a set of one or more complete databases, or an entire MySQL server—as shown here:

```
shell> mysqldump [options] db_name [tbl_name ...]
shell> mysqldump [options] --databases db_name ...
shell> mysqldump [options] --all-databases
```

To dump entire databases, do not name any tables following ***db_name***, or use the `--databases` or `--all-databases` option.

To see a list of the options your version of **mysqldump** supports, issue the command **mysqldump --help**.

Option Syntax - Alphabetical Summary

mysqldump supports the following options, which can be specified on the command line or in the `[mysqldump]` and `[client]` groups of an option file. For information about option files used by MySQL programs, see Section 4.2.2.2, “Using Option Files”.

Table 4.14 mysqldump Options

Option Name	Description	Introduced	Deprecated	Removed
--add-drop-database	Add DROP DATABASE statement before each CREATE DATABASE statement			
--add-drop-table	Add DROP TABLE statement before each CREATE TABLE statement			
--add-drop-trigger	Add DROP TRIGGER statement before each CREATE TRIGGER statement			
--add-locks	Surround each table dump with LOCK TABLES and UNLOCK TABLES statements			
--all-databases	Dump all tables in all databases			

Option Name	Description	Introduced	Deprecated	Removed
--allow-keywords	Allow creation of column names that are keywords			
--apply-slave-statements	Include STOP SLAVE prior to CHANGE MASTER statement and START SLAVE at end of output			
--bind-address	Use specified network interface to connect to MySQL Server			
--character-sets-dir	Directory where character sets are installed			
--column-statistics	Write ANALYZE TABLE statements to generate statistics histograms	8.0.2		
--comments	Add comments to dump file			
--compact	Produce more compact output			
--compatible	Produce output that is more compatible with other database systems or with older MySQL servers			
--complete-insert	Use complete INSERT statements that include column names			
--compress	Compress all information sent between client and server		8.0.18	
--compression-algorithms	Permitted compression algorithms for connections to server	8.0.18		
--create-options	Include all MySQL-specific table options in CREATE TABLE statements			
--databases	Interpret all name arguments as database names			
--debug	Write debugging log			
--debug-check	Print debugging information when program exits			
--debug-info	Print debugging information, memory, and CPU statistics when program exits			
--default-auth	Authentication plugin to use			
--default-character-set	Specify default character set			
--defaults-extra-file	Read named option file in addition to usual option files			
--defaults-file	Read only named option file			
--defaults-group-suffix	Option group suffix value			
--delete-master-logs	On a master replication server, delete the binary logs after performing the dump operation			
--disable-keys	For each table, surround INSERT statements with statements to disable and enable keys			

Option Name	Description	Introduced	Deprecated	Removed
--dump-date	Include dump date as "Dump completed on" comment if --comments is given			
--dump-slave	Include CHANGE MASTER statement that lists binary log coordinates of slave's master			
--enable-cleartext-plugin	Enable cleartext authentication plugin			
--events	Dump events from dumped databases			
--extended-insert	Use multiple-row INSERT syntax			
--fields-enclosed-by	This option is used with the --tab option and has the same meaning as the corresponding clause for LOAD DATA			
--fields-escaped-by	This option is used with the --tab option and has the same meaning as the corresponding clause for LOAD DATA			
--fields-optionally-enclosed-by	This option is used with the --tab option and has the same meaning as the corresponding clause for LOAD DATA			
--fields-terminated-by	This option is used with the --tab option and has the same meaning as the corresponding clause for LOAD DATA			
--flush-logs	Flush MySQL server log files before starting dump			
--flush-privileges	Emit a FLUSH PRIVILEGES statement after dumping mysql database			
--force	Continue even if an SQL error occurs during a table dump			
--get-server-public-key	Request RSA public key from server	8.0.3		
--help	Display help message and exit			
--hex-blob	Dump binary columns using hexadecimal notation			
--host	Host on which MySQL server is located			
--ignore-error	Ignore specified errors			
--ignore-table	Do not dump given table			
--include-master-host-port	Include MASTER_HOST/MASTER_PORT options in CHANGE MASTER statement produced with --dump-slave			
--insert-ignore	Write INSERT IGNORE rather than INSERT statements			
--lines-terminated-by	This option is used with the --tab option and has the same meaning as the corresponding clause for LOAD DATA			
--lock-all-tables	Lock all tables across all databases			
--lock-tables	Lock all tables before dumping them			

Option Name	Description	Introduced	Deprecated	Removed
--log-error	Append warnings and errors to named file			
--login-path	Read login path options from .mylogin.cnf			
--master-data	Write the binary log file name and position to the output			
--max-allowed-packet	Maximum packet length to send to or receive from server			
--net-buffer-length	Buffer size for TCP/IP and socket communication			
--network-timeout	Increase network timeouts to permit larger table dumps	8.0.1		
--no-autocommit	Enclose the INSERT statements for each dumped table within SET autocommit = 0 and COMMIT statements			
--no-create-db	Do not write CREATE DATABASE statements			
--no-create-info	Do not write CREATE TABLE statements that re-create each dumped table			
--no-data	Do not dump table contents			
--no-defaults	Read no option files			
--no-set-names	Same as --skip-set-charset			
--no-tablespaces	Do not write any CREATE LOGFILE GROUP or CREATE TABLESPACE statements in output			
--opt	Shorthand for --add-drop-table --add-locks --create-options --disable-keys --extended-insert --lock-tables --quick --set-charset.			
--order-by-primary	Dump each table's rows sorted by its primary key, or by its first unique index			
--password	Password to use when connecting to server			
--pipe	Connect to server using named pipe (Windows only)			
--plugin-dir	Directory where plugins are installed			
--port	TCP/IP port number for connection			
--print-defaults	Print default options			
--protocol	Connection protocol to use			
--quick	Retrieve rows for a table from the server a row at a time			
--quote-names	Quote identifiers within backtick characters			
--replace	Write REPLACE statements rather than INSERT statements			
--result-file	Direct output to a given file			
--routines	Dump stored routines (procedures and functions) from dumped databases			

Option Name	Description	Introduced	Deprecated	Removed
--secure-auth	Do not send passwords to server in old (pre-4.1) format			8.0.3
--server-public-key-path	Path name to file containing RSA public key	8.0.4		
--set-charset	Add SET NAMES default_character_set to output			
--set-gtid-purged	Whether to add SET @@GLOBAL.GTID_PURGED to output			
--shared-memory-base-name	Name of shared memory to use for shared-memory connections			
--show-create-skip-secondary-engine	Exclude SECONDARY ENGINE clause from CREATE TABLE statements	8.0.18		
--single-transaction	Issue a BEGIN SQL statement before dumping data from server			
--skip-add-drop-table	Do not add a DROP TABLE statement before each CREATE TABLE statement			
--skip-add-locks	Do not add locks			
--skip-comments	Do not add comments to dump file			
--skip-compact	Do not produce more compact output			
--skip-disable-keys	Do not disable keys			
--skip-extended-insert	Turn off extended-insert			
--skip-opt	Turn off options set by --opt			
--skip-quick	Do not retrieve rows for a table from the server a row at a time			
--skip-quote-names	Do not quote identifiers			
--skip-set-charset	Do not write SET NAMES statement			
--skip-triggers	Do not dump triggers			
--skip-tz-utc	Turn off tz-utc			
--socket	Unix socket file or Windows named pipe to use			
--ssl-ca	File that contains list of trusted SSL Certificate Authorities			
--ssl-capath	Directory that contains trusted SSL Certificate Authority certificate files			
--ssl-cert	File that contains X.509 certificate			
--ssl-cipher	Permissible ciphers for connection encryption			
--ssl-crl	File that contains certificate revocation lists			
--ssl-crlpath	Directory that contains certificate revocation-list files			
--ssl-fips-mode	Whether to enable FIPS mode on client side	8.0.11		
--ssl-key	File that contains X.509 key			

Option Name	Description	Introduced	Deprecated	Removed
--ssl-mode	Desired security state of connection to server			
--tab	Produce tab-separated data files			
--tables	Override --databases or -B option			
--tls-ciphersuites	Permissible TLSv1.3 ciphersuites for encrypted connections	8.0.16		
--tls-version	Permissible TLS protocols for encrypted connections			
--triggers	Dump triggers for each dumped table			
--tz-utc	Add SET TIME_ZONE='+00:00' to dump file			
--user	MySQL user name to use when connecting to server			
--verbose	Verbose mode			
--version	Display version information and exit			
--where	Dump only rows selected by given WHERE condition			
--xml	Produce XML output			
--zstd-compression-level	Compression level for connections to server that use zstd compression	8.0.18		

Connection Options

The **mysqldump** command logs into a MySQL server to extract information. The following options specify how to connect to the MySQL server, either on the same machine or a remote system.

- `--bind-address=`***`ip_address`***

  On a computer having multiple network interfaces, use this option to select which interface to use for connecting to the MySQL server.

- `--compress`, `-C`

  Compress all information sent between the client and the server if possible. See Section 4.2.6, “Connection Compression Control”.

  As of MySQL 8.0.18, this option is deprecated. It will be removed in a future MySQL version. See Legacy Connection Compression Configuration.

- `--compression-algorithms=`***`value`***

  The permitted compression algorithms for connections to the server. The available algorithms are the same as for the `protocol_compression_algorithms` system variable. The default value is `uncompressed`.

For more information, see Section 4.2.6, “Connection Compression Control”.

This option was added in MySQL 8.0.18.

- `--default-auth=`***`plugin`***

  A hint about which client-side authentication plugin to use. See Section 6.2.17, “Pluggable Authentication”.

- `--enable-cleartext-plugin`

  Enable the `mysql_clear_password` cleartext authentication plugin. (See Section 6.4.1.4, “Client-Side Cleartext Pluggable Authentication”.)

- `--get-server-public-key`

  Request from the server the public key required for RSA key pair-based password exchange. This option applies to clients that authenticate with the `caching_sha2_password` authentication plugin. For that plugin, the server does not send the public key unless requested. This option is ignored for accounts that do not authenticate with that plugin. It is also ignored if RSA-based password exchange is not used, as is the case when the client connects to the server using a secure connection.

  If `--server-public-key-path=`***`file_name`*** is given and specifies a valid public key file, it takes precedence over `--get-server-public-key`.

  For information about the `caching_sha2_password` plugin, see Section 6.4.1.3, “Caching SHA-2 Pluggable Authentication”.

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

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

- `--login-path=`***`name`***

  Read options from the named login path in the `.mylogin.cnf` login path file. A “login path” is an option group containing options that specify which MySQL server to connect to and which account to authenticate as. To create or modify a login path file, use the **mysql_config_editor** utility. See Section 4.6.7, “**mysql_config_editor** — MySQL Configuration Utility”.

  For additional information about this and other option-file options, see Section 4.2.2.3, “Command-Line Options that Affect Option-File Handling”.

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

  The password of the MySQL account used for connecting to the server. The password value is optional. If not given, **mysqldump** prompts for one. If given, there must be *no space* between `--`

`password=` or `-p` and the password following it. If no password option is specified, the default is to send no password.

Specifying a password on the command line should be considered insecure. To avoid giving the password on the command line, use an option file. See Section 6.1.2.1, “End-User Guidelines for Password Security”.

To explicitly specify that there is no password and that **mysqldump** should not prompt for one, use the `--skip-password` option.

- `--pipe`, `-W`

  On Windows, connect to the server using a named pipe. This option applies only if the server was started with the `named_pipe` system variable enabled to support named-pipe connections. In addition, the user making the connection must be a member of the Windows group specified by the `named_pipe_full_access_group` system variable.

- `--plugin-dir=`***`dir_name`***

  The directory in which to look for plugins. Specify this option if the `--default-auth` option is used to specify an authentication plugin but **mysqldump** does not find it. See Section 6.2.17, “Pluggable Authentication”.

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

  For TCP/IP connections, the port number to use.

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

  The connection protocol to use for connecting to the server. It is useful when the other connection parameters normally result in use of a protocol other than the one you want. For details on the permissible values, see Section 4.2.4, “Connecting to the MySQL Server Using Command Options”.

- `--secure-auth`

  This option was removed in MySQL 8.0.3.

- `--server-public-key-path=`***`file_name`***

  The path name to a file containing a client-side copy of the public key required by the server for RSA key pair-based password exchange. The file must be in PEM format. This option applies to clients that authenticate with the `sha256_password` or `caching_sha2_password` authentication plugin. This option is ignored for accounts that do not authenticate with one of those plugins. It is also ignored if RSA-based password exchange is not used, as is the case when the client connects to the server using a secure connection.

If `--server-public-key-path=`***file_name*** is given and specifies a valid public key file, it takes precedence over `--get-server-public-key`.

For `sha256_password`, this option applies only if MySQL was built using OpenSSL.

For information about the `sha256_password` and `caching_sha2_password` plugins, see Section 6.4.1.2, “SHA-256 Pluggable Authentication”, and Section 6.4.1.3, “Caching SHA-2 Pluggable Authentication”.

- `--socket=`***path***, `-S` ***path***

  For connections to `localhost`, the Unix socket file to use, or, on Windows, the name of the named pipe to use.

  On Windows, this option applies only if the server was started with the `named_pipe` system variable enabled to support named-pipe connections. In addition, the user making the connection must be a member of the Windows group specified by the `named_pipe_full_access_group` system variable.

- `--ssl*`

  Options that begin with `--ssl` specify whether to connect to the server using SSL and indicate where to find SSL keys and certificates. See Command Options for Encrypted Connections.

- `--ssl-fips-mode={OFF|ON|STRICT}`

  Controls whether to enable FIPS mode on the client side. The `--ssl-fips-mode` option differs from other `--ssl-`***xxx*** options in that it is not used to establish encrypted connections, but rather to affect which cryptographic operations are permitted. See Section 6.5, “FIPS Support”.

  These `--ssl-fips-mode` values are permitted:

  - `OFF`: Disable FIPS mode.
  - `ON`: Enable FIPS mode.
  - `STRICT`: Enable “strict” FIPS mode.

  > **Note**
  >
  > If the OpenSSL FIPS Object Module is not available, the only permitted value for `--ssl-fips-mode` is `OFF`. In this case, setting `--ssl-fips-mode` to `ON` or `STRICT` causes the client to produce a warning at startup and to operate in non-FIPS mode.

- `--tls-ciphersuites=`***`ciphersuite_list`***

  The permissible ciphersuites for encrypted connections that use TLSv1.3. The value is a list of one or more colon-separated ciphersuite names. The ciphersuites that can be named for this option depend on the SSL library used to compile MySQL. For details, see Section 6.3.2, “Encrypted Connection TLS Protocols and Ciphers”.

  This option was added in MySQL 8.0.16.

- `--tls-version=`***`protocol_list`***

  The permissible TLS protocols for encrypted connections. The value is a list of one or more comma-separated protocol names. The protocols that can be named for this option depend on the SSL library used to compile MySQL. For details, see Section 6.3.2, “Encrypted Connection TLS Protocols and Ciphers”.

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

  The user name of the MySQL account to use for connecting to the server.

- `--zstd-compression-level=`***`level`***

  The compression level to use for connections to the server that use the `zstd` compression algorithm. The permitted levels are from 1 to 22, with larger values indicating increasing levels of compression. The default `zstd` compression level is 3. The compression level setting has no effect on connections that do not use `zstd` compression.

  For more information, see Section 4.2.6, “Connection Compression Control”.

  This option was added in MySQL 8.0.18.

Option-File Options

These options are used to control which option files to read.

- `--defaults-extra-file=`***`file_name`***

  Read this option file after the global option file but (on Unix) before the user option file. If the file does not exist or is otherwise inaccessible, an error occurs. ***`file_name`*** is interpreted relative to the current directory if given as a relative path name rather than a full path name.

  For additional information about this and other option-file options, see Section 4.2.2.3, “Command-Line Options that Affect Option-File Handling”.

- `--defaults-file=`***`file_name`***

Use only the given option file. If the file does not exist or is otherwise inaccessible, an error occurs. ***file_name*** is interpreted relative to the current directory if given as a relative path name rather than a full path name.

Exception: Even with `--defaults-file`, client programs read `.mylogin.cnf`.

For additional information about this and other option-file options, see Section 4.2.2.3, "Command-Line Options that Affect Option-File Handling".

- `--defaults-group-suffix=`***str***

  Read not only the usual option groups, but also groups with the usual names and a suffix of ***str***. For example, **mysqldump** normally reads the `[client]` and `[mysqldump]` groups. If the `--defaults-group-suffix=_other` option is given, **mysqldump** also reads the `[client_other]` and `[mysqldump_other]` groups.

  For additional information about this and other option-file options, see Section 4.2.2.3, "Command-Line Options that Affect Option-File Handling".

- `--no-defaults`

  Do not read any option files. If program startup fails due to reading unknown options from an option file, `--no-defaults` can be used to prevent them from being read.

  The exception is that the `.mylogin.cnf` file, if it exists, is read in all cases. This permits passwords to be specified in a safer way than on the command line even when `--no-defaults` is used. (`.mylogin.cnf` is created by the **mysql_config_editor** utility. See Section 4.6.7, "**mysql_config_editor** — MySQL Configuration Utility".)

  For additional information about this and other option-file options, see Section 4.2.2.3, "Command-Line Options that Affect Option-File Handling".

- `--print-defaults`

  Print the program name and all options that it gets from option files.

  For additional information about this and other option-file options, see Section 4.2.2.3, "Command-Line Options that Affect Option-File Handling".

DDL Options

Usage scenarios for **mysqldump** include setting up an entire new MySQL instance (including database tables), and replacing data inside an existing instance with existing databases and tables. The following options let you specify which things to tear down and set up when restoring a dump, by encoding various DDL statements within the dump file.

- `--add-drop-database`

  Write a `DROP DATABASE` statement before each `CREATE DATABASE` statement. This option is typically used in conjunction with the `--all-databases` or `--databases` option because no `CREATE DATABASE` statements are written unless one of those options is specified.

- `--add-drop-table`

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

- `--add-drop-trigger`

  Write a `DROP TRIGGER` statement before each `CREATE TRIGGER` statement.

- `--all-tablespaces`, `-Y`

  Adds to a table dump all SQL statements needed to create any tablespaces used by an `NDB` table. This information is not otherwise included in the output from **mysqldump**. This option is currently relevant only to NDB Cluster tables.

- `--no-create-db`, `-n`

  Suppress the `CREATE DATABASE` 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 create each dumped table.

  > **Note**
  >
  > This option does *not* exclude statements creating log file groups or tablespaces from **mysqldump** output; however, you can use the `--no-tablespaces` option for this purpose.

- `--no-tablespaces`, `-y`

  This option suppresses all `CREATE LOGFILE GROUP` and `CREATE TABLESPACE` statements in the output of **mysqldump**.

- `--replace`

  Write `REPLACE` statements rather than `INSERT` statements.

Debug Options

The following options print debugging information, encode debugging information in the dump file, or let the dump operation proceed regardless of potential problems.

- `--allow-keywords`

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

- `--comments`, `-i`

  Write additional information in the dump file such as program version, server version, and host. This option is enabled by default. To suppress this additional information, use `--skip-comments`.

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

  Write a debugging log. A typical *`debug_options`* string is `d:t:o,file_name`. The default value is `d:t:o,/tmp/mysqldump.trace`.

- `--debug-check`

  Print some debugging information when the program exits.

- `--debug-info`

  Print debugging information and memory and CPU usage statistics when the program exits.

- `--dump-date`

  If the `--comments` option is given, **mysqldump** produces a comment at the end of the dump of the following form:

  ```
  -- Dump completed on DATE
  ```

  However, the date causes dump files taken at different times to appear to be different, even if the data are otherwise identical. `--dump-date` and `--skip-dump-date` control whether the date is added to the comment. The default is `--dump-date` (include the date in the comment). `--skip-dump-date` suppresses date printing.

- `--force`, `-f`

  Ignore all errors; continue even if an SQL error occurs during a table dump.

  One use for this option is to cause **mysqldump** to continue executing even when it encounters a view that has become invalid because the definition refers to a table that has been dropped. Without `--force`, **mysqldump** exits with an error message. With `--force`, **mysqldump** prints the

error message, but it also writes an SQL comment containing the view definition to the dump output and continues executing.

If the `--ignore-error` option is also given to ignore specific errors, `--force` takes precedence.

- `--log-error=`***file_name***

  Log warnings and errors by appending them to the named file. The default is to do no logging.

- `--skip-comments`

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

- `--verbose`, `-v`

  Verbose mode. Print more information about what the program does.

Help Options

The following options display information about the **mysqldump** command itself.

- `--help`, `-?`

  Display a help message and exit.

- `--version`, `-V`

  Display version information and exit.

Internationalization Options

The following options change how the **mysqldump** command represents character data with national language settings.

- `--character-sets-dir=`***dir_name***

  The directory where character sets are installed. See Section 10.15, “Character Set Configuration”.

- `--default-character-set=`***charset_name***

  Use ***charset_name*** as the default character set. See Section 10.15, “Character Set Configuration”. If no character set is specified, **mysqldump** uses `utf8`.

- `--no-set-names`, `-N`

  Turns off the `--set-charset` setting, the same as specifying `--skip-set-charset`.

- `--set-charset`

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

Replication Options

The **mysqldump** command is frequently used to create an empty instance, or an instance including data, on a slave server in a replication configuration. The following options apply to dumping and restoring data on replication master and slave servers.

- `--apply-slave-statements`

  For a slave dump produced with the `--dump-slave` option, add a `STOP SLAVE` statement before the `CHANGE MASTER TO` statement and a `START SLAVE` statement at the end of the output.

- `--delete-master-logs`

  On a master replication server, delete the binary logs by sending a `PURGE BINARY LOGS` statement to the server after performing the dump operation. This option automatically enables `--master-data`.

- `--dump-slave[=*value*]`

  This option is similar to `--master-data` except that it is used to dump a replication slave server to produce a dump file that can be used to set up another server as a slave that has the same master as the dumped server. It causes the dump output to include a `CHANGE MASTER TO` statement that indicates the binary log coordinates (file name and position) of the dumped slave's master. The `CHANGE MASTER TO` statement reads the values of `Relay_Master_Log_File` and `Exec_Master_Log_Pos` from the `SHOW SLAVE STATUS` output and uses them for `MASTER_LOG_FILE` and `MASTER_LOG_POS` respectively. These are the master server coordinates from which the slave should start replicating.

  > **Note**
  >
  > Inconsistencies in the sequence of transactions from the relay log which have been executed can cause the wrong position to be used. See Section 17.4.1.33, “Replication and Transaction Inconsistencies” for more information.

  `--dump-slave` causes the coordinates from the master to be used rather than those of the dumped server, as is done by the `--master-data` option. In addition, specfiying this option causes the `--master-data` option to be overridden, if used, and effectively ignored.

  > **Warning**

> This option should not be used if the server where the dump is going to be applied uses `gtid_mode=ON` and `MASTER_AUTOPOSITION=1`.

The option value is handled the same way as for `--master-data` (setting no value or 1 causes a `CHANGE MASTER TO` statement to be written to the dump, setting 2 causes the statement to be written but encased in SQL comments) and has the same effect as `--master-data` in terms of enabling or disabling other options and in how locking is handled.

This option causes **mysqldump** to stop the slave SQL thread before the dump and restart it again after.

In conjunction with `--dump-slave`, the `--apply-slave-statements` and `--include-master-host-port` options can also be used.

- `--include-master-host-port`

  For the `CHANGE MASTER TO` statement in a slave dump produced with the `--dump-slave` option, add `MASTER_HOST` and `MASTER_PORT` options for the host name and TCP/IP port number of the slave's master.

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

  Use this option to dump a master replication server to produce a dump file that can be used to set up another server as a slave of the master. It causes the dump output to include a `CHANGE MASTER TO` statement that indicates the binary log coordinates (file name and position) of the dumped server. These are the master server coordinates from which the slave should start replicating after you load the dump file into the slave.

  If the option value is 2, the `CHANGE MASTER TO` statement is written as an SQL comment, and thus is informative only; it has no effect when the dump file is reloaded. If the option value is 1, the statement is not written as a comment and takes effect when the dump file is reloaded. If no option value is specified, the default value is 1.

  This option requires the `RELOAD` privilege and the binary log must be enabled.

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

  It is also possible to set up a slave by dumping an existing slave of the master, using the `--dump-slave` option, which overrides `--master-data` and causes it to be ignored if both options are used.

- `--set-gtid-purged=`***value***

  This option is for servers that use GTID-based replication (`gtid_mode=ON`). It controls the inclusion of a `SET @@GLOBAL.gtid_purged` statement in the dump output, which updates the value of `gtid_purged` on a server where the dump file is reloaded, to add the GTID set from the source server's `gtid_executed` system variable. `gtid_purged` holds the GTIDs of all transactions that have been applied on the server, but do not exist on any binary log file on the server. **mysqldump** therefore adds the GTIDs for the transactions that were executed on the source server, so that the target server records these transactions as applied, although it does not have them in its binary logs. `--set-gtid-purged` also controls the inclusion of a `SET @@SESSION.sql_log_bin=0` statement, which disables binary logging while the dump file is being reloaded. This statement prevents new GTIDs from being generated and assigned to the transactions in the dump file as they are executed, so that the original GTIDs for the transactions are used.

  If you do not set the `--set-gtid-purged` option, the default is that a `SET @@GLOBAL.gtid_purged` statement is included in the dump output if GTIDs are enabled on the server you are backing up, and the set of GTIDs in the global value of the `gtid_executed` system variable is not empty. A `SET @@SESSION.sql_log_bin=0` statement is also included if GTIDs are enabled on the server.

  In MySQL 5.6 and 5.7, you can replace the value of `gtid_purged` with a specified GTID set, provided that `gtid_executed` and `gtid_purged` are empty. From MySQL 8.0, you can either replace the value of `gtid_purged` with a specified GTID set, or you can add a plus sign (+) to the statement to append a specified GTID set to the GTID set that is already held by `gtid_purged`. **mysqldump**'s `SET @@GLOBAL.gtid_purged` statement includes a plus sign (+) in a version comment that takes effect when the dump file is replayed on releases from MySQL 8.0, meaning that for these releases, the GTID set from the dump file is added to the existing `gtid_purged` value. For MySQL 5.6 and 5.7, the value of `gtid_purged` is replaced with the GTID set from the dump file, which can only happen when `gtid_executed` is the empty set (so when replication has not been started previously, or when replication was not previously using GTIDs). For the exact details of how the `SET @@GLOBAL.gtid_purged` statement operates, see the `gtid_purged` description for the release where the dump file is to be replayed.

  It is important to note that the value that is included by **mysqldump** for the `SET @@GLOBAL.gtid_purged` statement includes the GTIDs of all transactions in the `gtid_executed` set on the server, even those that changed suppressed parts of the database, or other databases on the server that were not included in a partial dump. This can mean that after the `gtid_purged` value has been updated on the server where the dump file is replayed, GTIDs are present that do not relate to any data on the target server. If you do not replay any further dump files on the target server, the extraneous GTIDs do not cause any problems with the future operation of the server, but they make it harder to compare or reconcile GTID sets on different servers in the replication topology. If you do replay a further dump file on the target server that contains the

same GTIDs (for example, another partial dump from the same origin server), any `SET @@GLOBAL.gtid_purged` statement in the second dump file fails. In this case, either remove the statement manually before replaying the dump file, or output the dump file without the statement.

> **Note**
>
> For MySQL 5.6 and 5.7, it is not recommended to load a dump file when GTIDs are enabled on the server (`gtid_mode=ON`), if your dump file includes system tables. **mysqldump** issues DML instructions for the system tables which use the non-transactional MyISAM storage engine, and this combination is not permitted when GTIDs are enabled.

If the `SET @@GLOBAL.gtid_purged` statement would not have the desired result on your target server, you can exclude the statement from the output, or (from MySQL 8.0.17) include it but comment it out so that it is not actioned automatically. You can also include the statement but manually edit it in the dump file to achieve the desired result.

The possible values for the `--set-gtid-purged` option are as follows:

`AUTO`

The default value. If GTIDs are enabled on the server you are backing up and `gtid_executed` is not empty, `SET @@GLOBAL.gtid_purged` is added to the output, containing the GTID set from `gtid_executed`. If GTIDs are enabled, `SET @@SESSION.sql_log_bin=0` is added to the output. If GTIDs are not enabled on the server, the statements are not added to the output.

`OFF`

`SET @@GLOBAL.gtid_purged` is not added to the output, and `SET @@SESSION.sql_log_bin=0` is not added to the output. For a server where GTIDs are not in use, use this option or `AUTO`. Only use this option for a server where GTIDs are in use if you are sure that the required GTID set is already present in `gtid_purged` on the target server and should not be changed, or if you plan to identify and add any missing GTIDs manually.

`ON`

If GTIDs are enabled on the server you are backing up, `SET @@GLOBAL.gtid_purged` is added to the output (unless `gtid_executed` is empty), and `SET @@SESSION.sql_log_bin=0` is added to the output. An error occurs if you set this option but GTIDs are not enabled on the server. For a server where GTIDs are in use, use this option or `AUTO`, unless you are sure that the GTIDs in `gtid_executed` are not needed on the target server.

`COMMENTED`

Available from MySQL 8.0.17. If GTIDs are enabled on the server you are backing up, `SET @@GLOBAL.gtid_purged` is added to the output (unless `gtid_executed` is empty), but it is commented out. This means that the value of `gtid_executed` is available in the output, but no action is taken automatically when the dump file is reloaded. `SET @@SESSION.sql_log_bin=0` is added to the output, and it is not commented out. With `COMMENTED`, you can control the use of the `gtid_executed` set manually or through automation. For example, you might prefer to do this if you are migrating data to another server that already has different active databases.

Format Options

The following options specify how to represent the entire dump file or certain kinds of data in the dump file. They also control whether certain optional information is written to the dump file.

- `--compact`

  Produce more compact output. This option enables the `--skip-add-drop-table`, `--skip-add-locks`, `--skip-comments`, `--skip-disable-keys`, and `--skip-set-charset` options.

- `--compatible=`***name***

  Produce output that is more compatible with other database systems or with older MySQL servers. The only permitted value for this option is `ansi`, which has the same meaning as the corresponding option for setting the server SQL mode. See Section 5.1.11, “Server SQL Modes”.

- `--complete-insert`, `-c`

  Use complete `INSERT` statements that include column names.

- `--create-options`

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

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

  These options are used with the `--tab` option and have the same meaning as the corresponding `FIELDS` clauses for `LOAD DATA`. See Section 13.2.7, “LOAD DATA Syntax”.

- `--hex-blob`

  Dump binary columns using hexadecimal notation (for example, `'abc'` becomes `0x616263`). The affected data types are `BINARY`, `VARBINARY`, `BLOB` types, `BIT`, all spatial data types, and other non-binary data types when used with the `binary` character set.

- `--lines-terminated-by=...`

This option is used with the `--tab` option and has the same meaning as the corresponding `LINES` clause for `LOAD DATA`. See Section 13.2.7, "LOAD DATA Syntax".

- `--quote-names`, `-Q`

  Quote identifiers (such as database, table, and column names) within ` characters. If the `ANSI_QUOTES` SQL mode is enabled, identifiers are quoted within " characters. This option is enabled 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_name`***, `-r` ***`file_name`***

  Direct output to the named file. The result file is created and its previous contents overwritten, even if an error occurs while generating the dump.

  This option should be used on Windows to prevent newline `\n` characters from being converted to `\r\n` carriage return/newline sequences.

- `--show-create-skip-secondary-engine=`***`value`***

  Excludes the `SECONDARY ENGINE` clause from `CREATE TABLE` statements. It does so by enabling the `show_create_table_skip_secondary_engine` system variable for the duration of the dump operation. Alternatively, you can enable the `show_create_table_skip_secondary_engine` system variable prior to using **mysqldump**.

  This option was added in MySQL 8.0.18.

- `--tab=`***`dir_name`***, `-T` ***`dir_name`***

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

  > **Note**
  >
  > This option should be used only when **mysqldump** is run on the same machine as the **mysqld** server. Because the server creates `*.txt` files in the directory that you specify, the directory must be writable by the server and the MySQL account that you use must have the `FILE` privilege. Because **mysqldump** creates `*.sql` in the same directory, it must be writable by your system login account.

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-terminated-by` options.

Column values are converted to the character set specified by the `--default-character-set` option.

- `--tz-utc`

  This option enables `TIMESTAMP` columns to be dumped and reloaded between servers in different time zones. **mysqldump** sets its connection time zone to UTC and adds `SET TIME_ZONE='+00:00'` to the dump file. Without this option, `TIMESTAMP` columns are dumped and reloaded in the time zones local to the source and destination servers, which can cause the values to change if the servers are in different time zones. `--tz-utc` also protects against changes due to daylight saving time. `--tz-utc` is enabled by default. To disable it, use `--skip-tz-utc`.

- `--xml`, `-X`

  Write dump output as well-formed XML.

  `NULL`, `'NULL'`, and Empty Values: For a column named ***`column_name`***, the `NULL` value, an empty string, and the string value `'NULL'` are distinguished from one another in the output generated by this option as follows.

Value:	**XML Representation:**
`NULL` (*unknown value*)	`<field name="`***`column_name`***`" xsi:nil="true" />`
`''` (*empty string*)	`<field name="`***`column_name`***`"></field>`
`'NULL'` (*string value*)	`<field name="`***`column_name`***`">NULL</field>`

  The output from the **mysql** client when run using the `--xml` option also follows the preceding rules. (See Section 4.5.1.1, “mysql Client Options”.)

  XML output from **mysqldump** includes the XML namespace, as shown here:

```
shell> mysqldump --xml -u root world City
<?xml version="1.0"?>
<mysqldump xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<database name="world">
<table_structure name="City">
<field Field="ID" Type="int(11)" Null="NO" Key="PRI" Extra="auto_increme
<field Field="Name" Type="char(35)" Null="NO" Key="" Default="" Extra=""
<field Field="CountryCode" Type="char(3)" Null="NO" Key="" Default="" Ex
```

```
<field Field="District" Type="char(20)" Null="NO" Key="" Default="" Extr
<field Field="Population" Type="int(11)" Null="NO" Key="" Default="0" Ex
<key Table="City" Non_unique="0" Key_name="PRIMARY" Seq_in_index="1" Col
Collation="A" Cardinality="4079" Null="" Index_type="BTREE" Comment="" /
<options Name="City" Engine="MyISAM" Version="10" Row_format="Fixed" Row
Avg_row_length="67" Data_length="273293" Max_data_length="18858823439613
Index_length="43008" Data_free="0" Auto_increment="4080"
Create_time="2007-03-31 01:47:01" Update_time="2007-03-31 01:47:02"
Collation="latin1_swedish_ci" Create_options="" Comment="" />
</table_structure>
<table_data name="City">
<row>
<field name="ID">1</field>
<field name="Name">Kabul</field>
<field name="CountryCode">AFG</field>
<field name="District">Kabol</field>
<field name="Population">1780000</field>
</row>

...

<row>
<field name="ID">4079</field>
<field name="Name">Rafah</field>
<field name="CountryCode">PSE</field>
<field name="District">Rafah</field>
<field name="Population">92020</field>
</row>
</table_data>
</database>
</mysqldump>
```

Filtering Options

The following options control which kinds of schema objects are written to the dump file: by category, such as triggers or events; by name, for example, choosing which databases and tables to dump; or even filtering rows from the table data using a `WHERE` clause.

- `--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.

  Prior to MySQL 8.0, the `--routines` and `--events` options for **mysqldump** and **mysqlpump** were not required to include stored routines and events when using the `--all-databases` option: The dump included the `mysql` system database, and therefore also the `mysql.proc` and `mysql.event` tables containing stored routine and event definitions. As of MySQL 8.0, the `mysql.event` and `mysql.proc` tables are not used. Definitions for the corresponding objects are

stored in data dictionary tables, but those tables are not dumped. To include stored routines and events in a dump made using `--all-databases`, use the `--routines` and `--events` options explicitly.

- `--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` and `USE` statements are included in the output before each new database.

  This option may be used to dump the `performance_schema` database, which normally is not dumped even with the `--all-databases` option. (Also use the `--skip-lock-tables` option.)

- `--events`, `-E`

  Include Event Scheduler events for the dumped databases in the output. This option requires the `EVENT` privileges for those databases.

  The output generated by using `--events` contains `CREATE EVENT` statements to create the events.

- `--ignore-error=`***`error[,error]...`***

  Ignore the specified errors. The option value is a list of comma-separated error numbers specifying the errors to ignore during **mysqldump** execution. If the `--force` option is also given to ignore all errors, `--force` takes precedence.

- `--ignore-table=`***`db_name.tbl_name`***

  Do not dump the given table, which must be specified using both the database and table names. To ignore multiple tables, use this option multiple times. This option also can be used to ignore views.

- `--no-data`, `-d`

  Do not write any table row information (that is, do not dump table contents). This is useful if you want to dump only the `CREATE TABLE` statement for the table (for example, to create an empty copy of the table by loading the dump file).

- `--routines`, `-R`

  Include stored routines (procedures and functions) for the dumped databases in the output. This option requires the global `SELECT` privilege.

  The output generated by using `--routines` contains `CREATE PROCEDURE` and `CREATE FUNCTION` statements to create the routines.

- `--tables`

  Override the `--databases` or `-B` option. **mysqldump** regards all name arguments following the option as table names.

- `--triggers`

  Include triggers for each dumped table in the output. This option is enabled by default; disable it with `--skip-triggers`.

  To be able to dump a table's triggers, you must have the `TRIGGER` privilege for the table.

  Multiple triggers are permitted. **mysqldump** dumps triggers in activation order so that when the dump file is reloaded, triggers are created in the same activation order. However, if a **mysqldump** dump file contains multiple triggers for a table that have the same trigger event and action time, an error occurs for attempts to load the dump file into an older server that does not support multiple triggers. (For a workaround, see Downgrade Notes; you can convert triggers to be compatible with older servers.)

- `--where='`***`where_condition`***`',-w '`***`where_condition`***`'`

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

  Examples:

```
--where="user='jimf'"
-w"userid>1"
-w"userid<1"
```

Performance Options

The following options are the most relevant for the performance particularly of the restore operations. For large data sets, restore operation (processing the `INSERT` statements in the dump file) is the most time-consuming part. When it is urgent to restore data quickly, plan and test the performance of this stage in advance. For restore times measured in hours, you might prefer an alternative backup and restore solution, such as MySQL Enterprise Backup for `InnoDB`-only and mixed-use databases.

Performance is also affected by the transactional options, primarily for the dump operation.

- `--column-statistics`

  Add `ANALYZE TABLE` statements to the output to generate histogram statistics for dumped tables when the dump file is reloaded. This option is disabled by default because histogram generation for large tables can take a long time.

- `--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 only for nonunique indexes of `MyISAM` tables.

- `--extended-insert`, `-e`

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

- `--insert-ignore`

  Write `INSERT IGNORE` statements rather than `INSERT` statements.

- `--max-allowed-packet=`***value***

  The maximum size of the buffer for client/server communication. The default is 24MB, the maximum is 1GB.

- `--net-buffer-length=`***value***

  The initial size of the buffer for client/server communication. When creating multiple-row `INSERT` statements (as with the `--extended-insert` or `--opt` option), **mysqldump** creates rows up to `--net-buffer-length` bytes long. If you increase this variable, ensure that the MySQL server `net_buffer_length` system variable has a value at least this large.

- `--network-timeout`, `-M`

  Enable large tables to be dumped by setting `--max-allowed-packet` to its maximum value and network read and write timeouts to a large value. This option is enabled by default. To disable it, use `--skip-network-timeout`.

- `--opt`

  This option, enabled by default, is shorthand for the combination of `--add-drop-table` `--add-locks` `--create-options` `--disable-keys` `--extended-insert` `--lock-tables` `--quick` `--set-charset`. It gives a fast dump operation and produces a dump file that can be reloaded into a MySQL server quickly.

  Because the `--opt` option is enabled by default, you only specify its converse, the `--skip-opt` to turn off several default settings. See the discussion of `mysqldump` option groups for information about selectively enabling or disabling a subset of the options affected by `--opt`.

- `--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.

- `--skip-opt`

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

Transactional Options

The following options trade off the performance of the dump operation, against the reliability and consistency of the exported data.

- `--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 Section 8.2.5.1, “Optimizing INSERT Statements”.

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

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

- `--flush-privileges`

  Add a `FLUSH PRIVILEGES` statement to the dump output after dumping the `mysql` database. This option should be used any time the dump contains the `mysql` database and any other database that depends on the data in the `mysql` database for proper restoration.

  > **Note**
  >
  > For upgrades to MySQL 5.7.2 or higher from older versions, do not use `--flush-privileges`. For upgrade instructions in this case, see Section 2.11.4, “Changes in MySQL 8.0”.

- `--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

  For each dumped database, lock all tables to be dumped before dumping them. The tables are locked with READ LOCAL to permit concurrent inserts in the case of MyISAM tables. For transactional tables such as InnoDB, --single-transaction is a much better option than --lock-tables because it does not need to lock the tables at all.

  Because --lock-tables locks tables for each database separately, 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.

  Some options, such as --opt, automatically enable --lock-tables. If you want to override this, use --skip-lock-tables at the end of the option list.

- --no-autocommit

  Enclose the INSERT statements for each dumped table within SET autocommit = 0 and COMMIT statements.

- --order-by-primary

  Dump each table's rows sorted by its primary key, or by its first unique index, if such an index exists. This is useful when dumping a MyISAM table to be loaded into an InnoDB table, but makes the dump operation take considerably longer.

- --shared-memory-base-name=***name***

  On Windows, the shared-memory name to use for connections made using shared memory to a local server. The default value is MYSQL. The shared-memory name is case-sensitive.

  This option applies only if the server was started with the shared_memory system variable enabled to support shared-memory connections.

- --single-transaction

  This option sets the transaction isolation mode to REPEATABLE READ and sends a START TRANSACTION SQL statement to the server before dumping data. It is useful only with transactional tables such as InnoDB, because then it dumps the consistent state of the database at the time when START TRANSACTION 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 MEMORY tables dumped while using this option may still change state.

While a `--single-transaction` dump is in process, to ensure a valid dump file (correct table contents and binary log coordinates), no other connection should use the following statements: `ALTER TABLE`, `CREATE TABLE`, `DROP TABLE`, `RENAME TABLE`, `TRUNCATE TABLE`. A consistent read is not isolated from those statements, so use of them on a table to be dumped can cause the `SELECT` that is performed by **mysqldump** to retrieve the table contents to obtain incorrect contents or fail.

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 large tables, combine the `--single-transaction` option with the `--quick` option.

Option Groups

- The `--opt` option turns on several settings that work together to perform a fast dump operation. All of these settings are on by default, because `--opt` is on by default. Thus you rarely if ever specify `--opt`. Instead, you can turn these settings off as a group by specifying `--skip-opt`, the optionally re-enable certain settings by specifying the associated options later on the command line.
- The `--compact` option turns off several settings that control whether optional statements and comments appear in the output. Again, you can follow this option with other options that re-enable certain settings, or turn all the settings on by using the `--skip-compact` form.

When you selectively enable or disable the effect of a group option, order is important because options are processed first to last. For example, `--disable-keys --lock-tables --skip-opt` would not have the intended effect; it is the same as `--skip-opt` by itself.

Examples

To make a backup of an entire database:

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

To load the dump file back into the server:

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

Another way to reload the dump file:

```
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
```

You can dump several databases with one command:

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

To dump all databases, use the `--all-databases` option:

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

For `InnoDB` tables, **mysqldump** provides a way of making an online backup:

```
shell> mysqldump --all-databases --master-data --single-transaction > all_da
```

This backup acquires 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 the lock is released. If long updating statements are running when the `FLUSH` statement is issued, the MySQL server may get stalled until those statements finish. After that, the dump becomes lock free and does not disturb reads and writes on the tables. If the update statements that the MySQL server receives are short (in terms of execution time), the initial lock period should not be noticeable, even with many updates.

For point-in-time recovery (also known as “roll-forward,” when you need to restore an old backup and replay the changes that happened since that backup), it is often useful to rotate the binary log (see Section 5.4.4, “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 `--master-data` and `--single-transaction` options can be used simultaneously, which provides a convenient way to make an online backup suitable for use prior to point-in-time recovery if tables are stored using the `InnoDB` storage engine.

For more information on making backups, see Section 7.2, “Database Backup Methods”, and Section 7.3, “Example Backup and Recovery Strategy”.

- To select the effect of `--opt` except for some features, use the `--skip` option for each feature. To disable extended inserts and memory buffering, use `--opt --skip-extended-insert --skip-quick`. (Actually, `--skip-extended-insert --skip-quick` is sufficient because `--opt` is on by default.)
- To reverse `--opt` for all features except index disabling and table locking, use `--skip-opt --disable-keys --lock-tables`.

Restrictions

mysqldump does not dump the `performance_schema` or `sys` schema by default. To dump any of these, name them explicitly on the command line. You can also name them with the `--databases` option. For `performance_schema`, also use the `--skip-lock-tables` option.

mysqldump does not dump the `INFORMATION_SCHEMA` schema.

mysqldump does not dump `InnoDB` `CREATE TABLESPACE` statements.

mysqldump does not dump the NDB Cluster `ndbinfo` information database.

mysqldump includes statements to recreate the `general_log` and `slow_query_log` tables for dumps of the `mysql` database. Log table contents are not dumped.

If you encounter problems backing up views due to insufficient privileges, see Section 24.9, “Restrictions on Views” for a workaround.