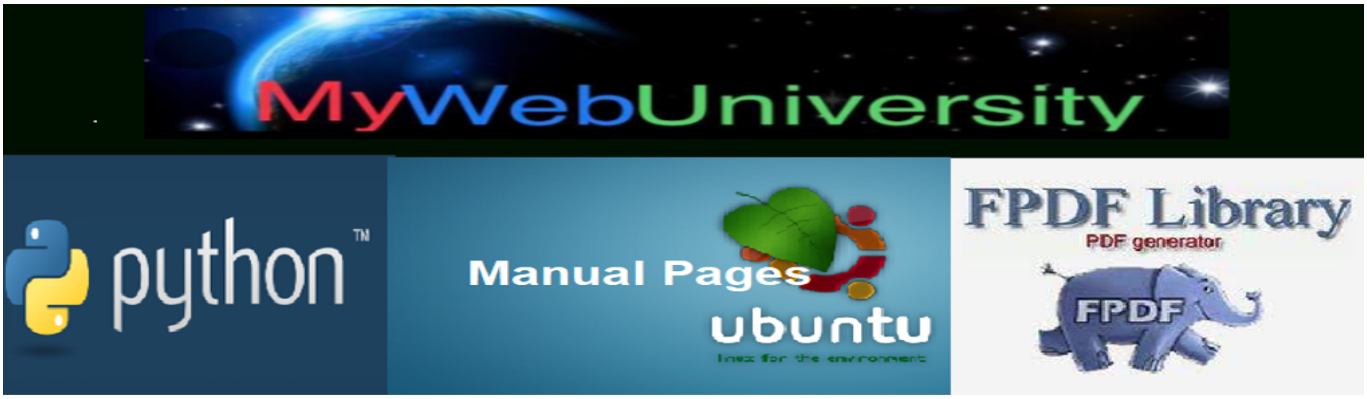

Rocky Enterprise Linux 9.2 Manual Pages on command 'mysqldump.1'

$ man mysqldump.1

MYSQLDUMP(1) MySQL Database System MYSQLDUMP(1)

NAME

mysqldump - a database backup program

SYNOPSIS

mysqldump [options] [db_name [tbl_name ...]]

DESCRIPTION

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.

Tip

Consider using the MySQL Shell dump utilities[1], which provide parallel dumping with multiple threads, file compression, and progress information display, as well as cloud features such as Oracle Cloud Infrastructure Object Storage streaming, and MySQL HeatWave Service compatibility checks and modifications. Dumps can be easily imported into a MySQL Server instance or a MySQL HeatWave Service DB System using the MySQL Shell load dump utilities[2]. Installation instructions for MySQL Shell can be found here[3].

? DDL Options

? Debug Options

? Help Options

? Internationalization Options

? Replication Options

? Format Options

? Filtering Options

? Performance Options

? Transactional Options

? Option Groups

? Examples

? Restrictions

mysqldump requires at least the SELECT privilege for dumped tables, SHOW VIEW for dumped views, TRIGGER for dumped triggers, LOCK TABLES if the --single-transaction option is not used, PROCESS (as of MySQL 8.0.21) if the --no-tablespaces option is not used, and (as of MySQL 8.0.32) the RELOAD or FLUSH_TABLES priviledge with --single-transaction if both gtid_mode=ON and --set-gtid=purged=ON|AUTO.

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:

```
mysqldump [options] > dump.sql
```

However, UTF-16 is not permitted as a connection character set (see the section called ?Impermissible Client Character Sets?), so the dump file cannot be loaded correctly.

To work around this issue, use the --result-file option, which creates the output in ASCII format:

```
mysqldump [options] --result-file=dump.sql
```

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. 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 so that they can be restored quickly.

If your tables are primarily InnoDB tables, or if you have a mix of InnoDB and MyISAM tables, consider using mysqlbackup, which is available as part of MySQL Enterprise. This tool provides high performance for InnoDB backups with minimal disruption; it can also back up tables from MyISAM and other storage engines; it also 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:

```
mysqldump [options] db_name [tbl_name ...]
```

mysqldump [options] --databases db_name ...

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?. 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

??

?Command-Line Format ? --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

??

?Command-Line Format ? --compress[={OFF|ON}] ?

??

?Deprecated ? 8.0.18 ?

??

?Type ? Boolean ?

??

?Default Value ? OFF ?

??

Compress all information sent between the client and the server if possible. See Section 4.2.8, ?Connection Compression Control?.

As of MySQL 8.0.18, this option is deprecated. Expect it to be removed in a future version of MySQL. See the section called ?Configuring Legacy Connection Compression?.

? --compression-algorithms=value

??

?Command-Line Format ? --compression-algorithms=value ?

??

?Introduced ? 8.0.18 ?

??

?Type ? Set ?

??

?Default Value ? uncompressed ?

??

?Valid Values ? ?

? ? zlib ?

? ? ?

? ? zstd ?

? ? ?

? ? uncompressed ?

??

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 4.2.8, ?Connection Compression Control?.

This option was added in MySQL 8.0.18.

? --default-auth=plugin

??

?Command-Line Format ? --default-auth=plugin ?

??

?Type ? String ?

??

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

? --enable-cleartext-plugin

??

?Command-Line Format ? --enable-cleartext-plugin ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

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

? --get-server-public-key

??

?Command-Line Format ? --get-server-public-key ?

??

?Type ? Boolean ?

??

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 6.4.1.2, ?Caching SHA-2 Pluggable Authentication?.

? --host=host_name, -h host_name

??????????????????????????????

?Command-Line Format ? --host ?

??????????????????????????????

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

? --login-path=name

??

?Command-Line Format ? --login-path=name ?

??

?Type ? String ?

??

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 mysql_config_editor(1).

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]

??

?Command-Line Format ? --password[=password] ?

??

?Type ? String ?

??

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.

? --password1[=pass_val] The password for multifactor authentication factor 1 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 --password1= 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-password1 option.

--password1 and --password are synonymous, as are --skip-password1 and

--skip-password.

? --password2[=pass_val] The password for multifactor authentication factor 2 of the MySQL account used for connecting to the server. The semantics of this option are similar to the semantics for --password1; see the description of that option for details.

? --password3[=pass_val] The password for multifactor authentication factor 3 of the MySQL account used for connecting to the server. The semantics of this option are similar to the semantics for --password1; see the description of that option for details.

? --pipe, -W

??????????????????????????????

?Command-Line Format ? --pipe ?

??????????????????????????????

?Type ? String ?

??????????????????????????????

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-authentication-kerberos-client-mode=value

??

?Command-Line Format ? --plugin-authentication- ?

? ? kerberos-client-mode ?

??

?Introduced ? 8.0.32 ?

??

?Type ? String ?

??

?Default Value ? SSPI ?

??

?Valid Values ? GSSAPI ?

??

On Windows, the authentication_kerberos_client authentication plugin supports this

plugin option. It provides two possible values that the client user can set at runtime: SSPI and GSSAPI.

The default value for the client-side plugin option uses Security Support Provider Interface (SSPI), which is capable of acquiring credentials from the Windows in-memory cache. Alternatively, the client user can select a mode that supports Generic Security Service Application Program Interface (GSSAPI) through the MIT Kerberos library on Windows. GSSAPI is capable of acquiring cached credentials previously generated by using the kinit command.

For more information, see Commands for Windows Clients in GSSAPI Mode.

? --plugin-dir=dir_name

??

?Command-Line Format ? --plugin-dir=dir_name ?

??

?Type ? Directory 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

??

?Command-Line Format ? --port=port_num ?

??

?Type ? Numeric ?

??

?Default Value ? 3306 ?

??

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

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

??

?Command-Line Format ? --protocol=type ?

??

?Type ? String ?

??

?Default Value ? [see text] ?

??????????????????????????????????????

?Valid Values ? ?

? ? TCP ?

? ? ?

? ? SOCKET ?

? ? ?

? ? PIPE ?

? ? ?

? ? MEMORY ?

??????????????????????????????????????

The transport 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 4.2.7, ?Connection Transport Protocols?.

? --server-public-key-path=file_name

??????????????????????????????????????

?Command-Line Format ? --server-public-key- ?

? ? path=file_name ?

??????????????????????????????????????

?Type ? File name ?

??????????????????????????????????????

The path name to a file in PEM format containing a client-side copy of the public key required by the server for RSA key pair-based password exchange. 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 6.4.1.3, ?SHA-256 Pluggable Authentication?, and Section 6.4.1.2, ?Caching

SHA-2 Pluggable Authentication?.

? --socket=path, -S path

??

?Command-Line Format ? --socket={file_name|pipe_name} ?

??

?Type ? String ?

??

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 encryption and indicate where to find SSL keys and certificates. See the section called ?Command Options for Encrypted Connections?.

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

??

?Command-Line Format ? --ssl-fips-mode={OFF|ON|STRICT} ?

??

?Deprecated ? 8.0.34 ?

??

?Type ? Enumeration ?

??

?Default Value ? OFF ?

??

?Valid Values ? ?

? ? 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 to permit. See Section 6.8, ?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.

As of MySQL 8.0.34, this option is deprecated. Expect it to be removed in a future version of MySQL.

? --tls-ciphersuites=ciphersuite_list

??

?Command-Line Format ? --tls- ?

? ? ciphersuites=ciphersuite_list ?

??

?Introduced ? 8.0.16 ?

??

?Type ? String ?

??

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

??

?Command-Line Format ? --tls-version=protocol_list ?

??

?Type ? String ?

??

?Default Value (? 8.0.16) ? ?

? ? TLSv1,TLSv1.1,TLSv1.2,TLSv1.3 ?

? ? (OpenSSL 1.1.1 or ?

? ? higher) ?

? ? ?

? ? TLSv1,TLSv1.1,TLSv1.2 ?

? ? (otherwise) ?

??

?Default Value (? 8.0.15) ? TLSv1,TLSv1.1,TLSv1.2 ?

??

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

??????????????????????????????????????

?Command-Line Format ? --user=user_name ?

??????????????????????????????????????

?Type ? String ?

??????????????????????????????????????

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

If you are using the Rewriter plugin with MySQL 8.0.31 or later, you should grant this user the SKIP_QUERY_REWRITE privilege.

? --zstd-compression-level=level

??

?Command-Line Format ? --zstd-compression-level=# ?

??

?Introduced ? 8.0.18 ?

??

?Type ? Integer ?

??

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 4.2.8, ?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

??

?Command-Line Format ? --defaults-extra-file=file_name ?

??

?Type ? 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. If file_name is not an absolute path name, it is interpreted relative to the current directory.

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

??

?Command-Line Format ? --defaults-file=file_name ?

??

?Type ? File name ?

??

Use only the given option file. If the file does not exist or is otherwise inaccessible, an error occurs. If file_name is not an absolute path name, it is interpreted relative to the current directory.

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

??

?Command-Line Format ? --defaults-group-suffix=str ?

??

?Type ? String ?

??

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 this option is given as --defaults-group-suffix=_other, 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

??????????????????????????????????????

?Command-Line Format ? --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 is read in all cases, if it exists. This permits passwords to be specified in a safer way than on the command line even when --no-defaults is used. To create .mylogin.cnf, use the mysql_config_editor utility. See mysql_config_editor(1).

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

??????????????????????????????????????

?Command-Line Format ? --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

??

?Command-Line Format ? --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.

Note

In MySQL 8.0, the mysql schema is considered a system schema that cannot be dropped by end users. If --add-drop-database is used with --all-databases or with --databases where the list of schemas to be dumped includes mysql, the dump file contains a DROP DATABASE `mysql` statement that causes an error when the dump file is reloaded.

Instead, to use --add-drop-database, use --databases with a list of schemas to be dumped, where the list does not include mysql.

? --add-drop-table

??

?Command-Line Format ? --add-drop-table ?

??

Write a DROP TABLE statement before each CREATE TABLE statement.

? --add-drop-trigger

??

?Command-Line Format ? --add-drop-trigger ?

??

Write a DROP TRIGGER statement before each CREATE TRIGGER statement.

? --all-tablespaces, -Y

??

?Command-Line Format ? --all-tablespaces ?

??

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

??????????????????????????????????????

?Command-Line Format ? --no-create-db ?

??????????????????????????????????????

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

??????????????????????????????????????

?Command-Line Format ? --no-create-info ?

??????????????????????????????????????

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

??????????????????????????????????????

?Command-Line Format ? --no-tablespaces ?

??????????????????????????????????????

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

? --replace

??????????????????????????????????

?Command-Line Format ? --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

??????????????????????????????????????

?Command-Line Format ? --allow-keywords ?

??????????????????????????????????????

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

? --comments, -i

??????????????????????????????????

?Command-Line Format ? --comments ?

??????????????????????????????????

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]

??

?Command-Line Format ? --debug[=debug_options] ?

??

?Type ? String ?

??

?Default Value ? d:t:o,/tmp/mysqldump.trace ?

??

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.

This option is available only if MySQL was built using WITH_DEBUG. MySQL release binaries provided by Oracle are not built using this option.

? --debug-check

??????????????????????????????????????

?Command-Line Format ? --debug-check ?

??????????????????????????????????????

?Type ? Boolean ?

??????????????????????????????????????

?Default Value ? FALSE ?

??????????????????????????????????????

Print some debugging information when the program exits.

This option is available only if MySQL was built using WITH_DEBUG. MySQL release binaries provided by Oracle are not built using this option.

? --debug-info

??????????????????????????????????????

?Command-Line Format ? --debug-info ?

??????????????????????????????????????

?Type ? Boolean ?

??????????????????????????????????????

?Default Value ? FALSE ?

??????????????????????????????????????

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

This option is available only if MySQL was built using WITH_DEBUG. MySQL release binaries provided by Oracle are not built using this option.

? --dump-date

??????????????????????????????????????

?Command-Line Format ? --dump-date ?

??????????????????????????????????????

?Type ? Boolean ?

??????????????????????????????????????

?Default Value ? TRUE ?

??????????????????????????????????????

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

??????????????????????????????????

?Command-Line Format ? --force ?

??????????????????????????????????

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

??

?Command-Line Format ? --log-error=file_name ?

??

?Type ? File name ?

??

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

? --skip-comments

??????????????????????????????????????

?Command-Line Format ? --skip-comments ?

??????????????????????????????????????

See the description for the --comments option.

? --verbose, -v

??????????????????????????????????

?Command-Line Format ? --verbose ?

??????????????????????????????????

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

Help Options

The following options display information about the mysqldump command itself.

? --help, -?

??????????????????????????????

?Command-Line Format ? --help ?

??????????????????????????????

Display a help message and exit.

? --version, -V

??????????????????????????????

?Command-Line Format ? --version ?

??????????????????????????????????

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

??

?Command-Line Format ? --character-sets-dir=dir_name ?

??

?Type ? Directory name ?

??

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

? --default-character-set=charset_name

??

?Command-Line Format ? --default-character- ?

? ? set=charset_name ?

??

?Type ? String ?

??

?Default Value ? utf8 ?

??

Use charset_name as the default character set. See Section 10.15, ?Character Set Configuration?. If no character set is specified, mysqldump uses utf8mb4.

? --no-set-names, -N

??????????????????????????????????????

?Command-Line Format ? --no-set-names ?

??????????????????????????????????????

?Deprecated ? Yes ?

??????????????????????????????????????

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

? --set-charset

??????????????????????????????????????

?Command-Line Format ? --set-charset ?

??????????????????????????????????????

?Disabled by ? skip-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 replica server in a replication configuration. The following options apply to dumping and restoring data on replication source servers and replicas.

? --apply-replica-statements

??

?Command-Line Format ? --apply-replica-statements ?

??

?Introduced ? 8.0.26 ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

From MySQL 8.0.26, use --apply-replica-statements, and before MySQL 8.0.26, use --apply-slave-statements. Both options have the same effect. For a replica dump produced with the --dump-replica or --dump-slave option, the options add a STOP REPLICA (or before MySQL 8.0.22, STOP SLAVE) statement before the statement with the binary log coordinates, and a START REPLICA statement at the end of the output.

? --apply-slave-statements

??

?Command-Line Format ? --apply-slave-statements ?

??

?Deprecated ? 8.0.26 ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

Use this option before MySQL 8.0.26 rather than --apply-replica-statements. Both options have the same effect.

? --delete-source-logs

??

?Command-Line Format ? --delete-source-logs ?

??

?Introduced ? 8.0.26 ?

??

From MySQL 8.0.26, use --delete-source-logs, and before MySQL 8.0.26, use --delete-master-logs. Both options have the same effect. On a replication source server, the options delete the binary logs by sending a PURGE BINARY LOGS statement to the server after performing the dump operation. The options require the RELOAD privilege as well as privileges sufficient to execute that statement. The options automatically enable --source-data or --master-data.

? --delete-master-logs

??

?Command-Line Format ? --delete-master-logs ?

??

?Deprecated ? 8.0.26 ?

??

Use this option before MySQL 8.0.26 rather than --delete-source-logs. Both options have the same effect.

? --dump-replica[=value]

??

?Command-Line Format ? --dump-replica[=value] ?

??

?Introduced ? 8.0.26 ?

??

?Type ? Numeric ?

??

?Default Value ? 1 ?

??

?Valid Values ? ?

? ? 1 ?

? ? ?

? ? 2 ?

??

From MySQL 8.0.26, use --dump-replica, and before MySQL 8.0.26, use --dump-slave. Both options have the same effect. The options are similar to --source-data, except that they are used to dump a replica server to produce a dump file that can be used to set up another server as a replica that has the same source as the dumped server. The options cause the dump output to include a CHANGE REPLICATION SOURCE TO statement (from MySQL 8.0.23) or CHANGE MASTER TO statement (before MySQL 8.0.23) that indicates the binary log coordinates (file name and position) of the dumped replica's source. The CHANGE REPLICATION SOURCE TO statement reads the values of Relay_Master_Log_File and Exec_Master_Log_Pos from the SHOW REPLICA STATUS output and uses them for SOURCE_LOG_FILE and SOURCE_LOG_POS respectively. These are the replication source server coordinates from which the replica starts 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 17.5.1.34, ?Replication and Transaction Inconsistencies? for more information.

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

Warning

--dump-replica and --dump-slave should not be used if the server where the dump is going to be applied uses gtid_mode=ON and SOURCE_AUTO_POSITION=1 or MASTER_AUTO_POSITION=1.

The option value is handled the same way as for --source-data. Setting no value or 1 causes a CHANGE REPLICATION SOURCE TO statement (from MySQL 8.0.23) or CHANGE MASTER TO statement (before MySQL 8.0.23) to be written to the dump. Setting 2 causes the

statement to be written but encased in SQL comments. It has the same effect as --source-data in terms of enabling or disabling other options and in how locking is handled.

--dump-replica and --dump-slave cause mysqldump to stop the replication SQL thread before the dump and restart it again after.

--dump-replica and --dump-slave send a SHOW REPLICA STATUS statement to the server to obtain information, so they require privileges sufficient to execute that statement.

--apply-replica-statements and --include-source-host-port options can be used in conjunction with --dump-replica and --dump-slave.

? --dump-slave[=value]

??

?Command-Line Format ? --dump-slave[=value] ?

??

?Deprecated ? 8.0.26 ?

??

?Type ? Numeric ?

??

?Default Value ? 1 ?

??

?Valid Values ? ?

? ? 1 ?

? ? ?

? ? 2 ?

??

Use this option before MySQL 8.0.26 rather than --dump-replica. Both options have the same effect.

? --include-source-host-port

??

?Command-Line Format ? --include-source-host-port ?

??

?Introduced ? 8.0.26 ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

From MySQL 8.0.26, use --include-source-host-port, and before MySQL 8.0.26, use --include-master-host-port. Both options have the same effect. The options add the SOURCE_HOST | MASTER_HOST and SOURCE_PORT | MASTER_PORT options for the host name and TCP/IP port number of the replica's source, to the CHANGE REPLICATION SOURCE TO statement (from MySQL 8.0.23) or CHANGE MASTER TO statement (before MySQL 8.0.23) in a replica dump produced with the --dump-replica or --dump-slave option.

? --include-master-host-port

??

?Command-Line Format ? --include-master-host-port ?

??

?Deprecated ? 8.0.26 ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

Use this option before MySQL 8.0.26 rather than --include-source-host-port. Both options have the same effect.

? --source-data[=value]

??

?Command-Line Format ? --source-data[=value] ?

??

?Introduced ? 8.0.26 ?

??

?Type ? Numeric ?

??

?Default Value ? 1 ?

??

?Valid Values ? ?

? ? 1 ?

? ? ?

? ? 2 ?

??

From MySQL 8.0.26, use --source-data, and before MySQL 8.0.26, use --master-data. Both options have the same effect. The options are used to dump a replication source server to produce a dump file that can be used to set up another server as a replica of the source. The options cause the dump output to include a CHANGE REPLICATION SOURCE TO statement (from MySQL 8.0.23) or CHANGE MASTER TO statement (before MySQL 8.0.23) that indicates the binary log coordinates (file name and position) of the dumped server. These are the replication source server coordinates from which the replica should start replicating after you load the dump file into the replica.

If the option value is 2, the CHANGE REPLICATION SOURCE TO | 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.

--source-data and --master-data send a SHOW MASTER STATUS statement to the server to obtain information, so they require privileges sufficient to execute that statement. This option also requires the RELOAD privilege and the binary log must be enabled.

--source-data and --master-data automatically turn off --lock-tables. They also turn 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 replica by dumping an existing replica of the source, using the --dump-replica or --dump-slave option, which overrides --source-data and --master-data and causes them to be ignored.

? --master-data[=value]

??

?Command-Line Format ? --master-data[=value] ?

??

?Deprecated ? 8.0.26 ?

??

?Type ? Numeric ?

??

?Default Value ? 1 ?

??

?Valid Values ? ?

? ? 1 ?

? ? ?

? ? 2 ?

??

Use this option before MySQL 8.0.26 rather than --source-data. Both options have the same effect.

? --set-gtid-purged=value

??

?Command-Line Format ? --set-gtid-purged=value ?

??

?Type ? Enumeration ?

??

?Default Value ? AUTO ?

??

?Valid Values ? ?

? ? OFF ?

? ? ?

? ? ON ?

? ? ?

? ? AUTO ?

??

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.

You can either replace the value of gtid_purged with a specified GTID set, or add a plus sign (+) to the statement to append a specified GTID set to the GTID set that is already held by gtid_purged. The SET @@GLOBAL.gtid_purged statement recorded by mysqldump includes a plus sign (+) in a version-specific comment, such that MySQL 8.0 (and later) adds the GTID set from the dump file to the existing gtid_purged value.

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.

Before MySQL 8.0.32: Using this option with the --single-transaction option could lead to inconsistencies in the output. If --set-gtid-purged=ON is required, it can be used with --lock-all-tables, but this can prevent parallel queries while mysqldump is being

run.

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

??????????????????????????????????

?Command-Line Format ? --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

??

?Command-Line Format ? --compatible=name[,name,...] ?

??

?Type ? String ?

??

?Default Value ? ?

??

?Valid Values ? ?

? ? ansi ?

? ? ?

? ? mysql323 ?

? ? ?

? ? mysql40 ?

? ? ?

? ? postgresql ?

? ? ?

? ? oracle ?

? ? ?

? ? mssql ?

? ? ?

? ? db2 ?

? ? ?

? ? maxdb ?

? ? ?

? ? no_key_options ?

? ? ?

? ? no_table_options ?

? ? ?

? ? no_key_options ?

??

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

??

?Command-Line Format ? --complete-insert ?

??

Use complete INSERT statements that include column names.

? --create-options

??

?Command-Line Format ? --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=...

??

?Command-Line Format ? --fields-terminated-by=string ?

??

?Type ? String ?

??

??

?Command-Line Format ? --fields-enclosed-by=string ?

??

?Type ? String ?

??

??

?Command-Line Format ? --fields-optionally-enclosed- ?

? ? by=string ?

??

?Type ? String ?

??

??

?Command-Line Format ? --fields-escaped-by ?

??

?Type ? String ?

??

These options are used with the --tab option and have the same meaning as the corresponding FIELDS clauses for LOAD DATA. See Section 13.2.9, ?LOAD DATA Statement?.

? --hex-blob

????????????????????????????????????

?Command-Line Format ? --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.

The --hex-blob option is ignored when the --tab is used.

? --lines-terminated-by=...

??

?Command-Line Format ? --lines-terminated-by=string ?

??

?Type ? String ?

??

This option is used with the --tab option and has the same meaning as the

corresponding LINES clause for LOAD DATA. See Section 13.2.9, ?LOAD DATA Statement?.

? --quote-names, -Q

??????????????????????????????????????

?Command-Line Format ? --quote-names ?

??????????????????????????????????????

?Disabled by ? skip-quote-names ?

??????????????????????????????????????

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

??????????????????????????????????????

?Command-Line Format ? --result-file=file_name ?

??????????????????????????????????????

?Type ? 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

??????????????????????????????????????

?Command-Line Format ? --show-create-skip-secondary- ?

? ? engine ?

??????????????????????????????????????

?Introduced ? 8.0.18 ?

??????????????????????????????????????

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. Attempting a mysqldump operation with the --show-create-skip-secondary-engine option on a release prior to MySQL 8.0.18 that does not support the show_create_table_skip_secondary_engine variable causes an error.

? --tab=dir_name, -T dir_name

??????????????????????????????????????

?Command-Line Format ? --tab=dir_name ?

??????????????????????????????????????

?Type ? Directory 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

??????????????????????????????????????

?Command-Line Format ? --tz-utc ?

??????????????????????????????????????

?Disabled by ? skip-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

??????????????????????????????

?Command-Line Format ? --xml ?

??????????????????????????????

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" /> ?

??

? ? ?

? ? <field ?

? ? name="column_name"></field> ?

??

? ? ?

? ? <field ?

? ? name="column_name">NULL</field> ?

??

The output from the mysql client when run using the --xml option also follows the preceding rules. (See the section called ?MYSQL CLIENT OPTIONS?.)

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

```
$> 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_increment" />

<field Field="Name" Type="char(35)" Null="NO" Key="" Default="" Extra="" />

<field Field="CountryCode" Type="char(3)" Null="NO" Key="" Default="" Extra="" />

<field Field="District" Type="char(20)" Null="NO" Key="" Default="" Extra="" />

<field Field="Population" Type="int(11)" Null="NO" Key="" Default="0" Extra="" />

<key Table="City" Non_unique="0" Key_name="PRIMARY" Seq_in_index="1" Column_name="ID"

Collation="A" Cardinality="4079" Null="" Index_type="BTREE" Comment="" />

<options Name="City" Engine="MyISAM" Version="10" Row_format="Fixed" Rows="4079"

Avg_row_length="67" Data_length="273293" Max_data_length="18858823439613951"

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

??????????????????????????????????????

?Command-Line Format ? --all-databases ?

??????????????????????????????????????

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

Note

See the --add-drop-database description for information about an incompatibility of that option with --all-databases.

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

??????????????????????????????????????

?Command-Line Format ? --databases ?

??????????????????????????????????????

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.)

Note

See the --add-drop-database description for information about an incompatibility of that option with --databases.

? --events, -E

??????????????????????????????

?Command-Line Format ? --events ?

??????????????????????????????

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]...

??

?Command-Line Format ? --ignore-error=error[,error]... ?

??

?Type ? String ?

??

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

??

?Command-Line Format ? --ignore-table=db_name.tbl_name ?

??

?Type ? String ?

??

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

??????????????????????????????

?Command-Line Format ? --no-data ?

??????????????????????????????

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

??????????????????????????????????

?Command-Line Format ? --routines ?

??????????????????????????????????

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.

? --skip-generated-invisible-primary-key

??

?Command-Line Format ? --skip-generated-invisible- ?

? ? primary-key ?

??

?Introduced ? 8.0.30 ?

??

?Type ? Boolean ?

??

?Default Value ? FALSE ?

??

This option is available beginning with MySQL 8.0.30, and causes generated invisible primary keys to be excluded from the output. For more information, see Section 13.1.20.11, ?Generated Invisible Primary Keys?.

? --tables

??????????????????????????????????

?Command-Line Format ? --tables ?

??????????????????????????????????

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

? --triggers

??????????????????????????????????????

?Command-Line Format ? --triggers ?

??????????????????????????????????????

?Disabled by ? skip-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[4]; you can convert triggers to be compatible with older servers.)

? --where='where_condition', -w 'where_condition'

??

?Command-Line Format ? --where='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

??

?Command-Line Format ? --column-statistics ?

??

?Type ? Boolean ?

??

?Default Value ? OFF ?

??

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

????????????????????????????????????

?Command-Line Format ? --disable-keys ?

????????????????????????????????????

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

??

?Command-Line Format ? --extended-insert ?

??

?Disabled by ? skip-extended-insert ?

??

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

????????????????????????????????????

?Command-Line Format ? --insert-ignore ?

????????????????????????????????????

Write INSERT IGNORE statements rather than INSERT statements.

? --max-allowed-packet=value

??

?Command-Line Format ? --max-allowed-packet=value ?

??

?Type ? Numeric ?

??

?Default Value ? 25165824 ?

??

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

Note

The value of this option is specific to mysqldump and should not be confused with the MySQL server's max_allowed_packet system variable; the server value cannot be exceeded by a single packet from mysqldump, regardless of any setting for the mysqldump option, even if the latter is larger.

? --mysqld-long-query-time=value

??

?Command-Line Format ? --mysqld-long-query-time=value ?

??

?Introduced ? 8.0.30 ?

??

?Type ? Numeric ?

??

?Default Value ? Server global setting ?

??

Set the session value of the long_query_time system variable. Use this option, which is available from MySQL 8.0.30, if you want to increase the time allowed for mysqldump?s queries before they are logged to the slow query log file. mysqldump performs a full table scan, which means its queries can often exceed a global long_query_time setting that is useful for regular queries. The default global setting is 10 seconds.

You can use --mysqld-long-query-time to specify a session value from 0 (meaning that every query from mysqldump is logged to the slow query log) to 31536000, which is 365

days in seconds. For mysqldump?s option, you can only specify whole seconds. When you do not specify this option, the server?s global setting applies to mysqldump?s queries.

? --net-buffer-length=value

??

?Command-Line Format ? --net-buffer-length=value ?

??

?Type ? Numeric ?

??

?Default Value ? 16384 ?

??

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

??

?Command-Line Format ? --network-timeout[={0|1}] ?

??

?Type ? Boolean ?

??

?Default Value ? TRUE ?

??

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

??????????????????????????????

?Command-Line Format ? --opt ?

??????????????????????????????

?Disabled by ? skip-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

??????????????????????????????????

?Command-Line Format ? --quick ?

??????????????????????????????????

?Disabled by ? skip-quick ?

??????????????????????????????????

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

??????????????????????????????????

?Command-Line Format ? --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

??????????????????????????????????

?Command-Line Format ? --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

??????????????????????????????????

?Command-Line Format ? --flush-logs ?

??????????????????????????????????

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, --source-data or --master-data, or --single-transaction. In these cases, 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, --source-data or --master-data, or --single-transaction.

? --flush-privileges

??????????????????????????????????????

?Command-Line Format ? --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.

Because the dump file contains a FLUSH PRIVILEGES statement, reloading the file requires privileges sufficient to execute that statement.

Note

For upgrades to MySQL 5.7 or higher from older versions, do not use --flush-privileges. For upgrade instructions in this case, see Section 2.10.4, ?Changes in MySQL 8.0?.

? --lock-all-tables, -x

??????????????????????????????????????

?Command-Line Format ? --lock-all-tables ?

??????????????????????????????????????

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

??????????????????????????????????

?Command-Line Format ? --lock-tables ?

??????????????????????????????????????

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

??????????????????????????????????????

?Command-Line Format ? --no-autocommit ?

??????????????????????????????????????

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

? --order-by-primary

??????????????????????????????????????

?Command-Line Format ? --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

??????????????????????????????????????

?Command-Line Format ? --shared-memory-base-name=name ?

??????????????????????????????????????

?Platform Specific ? Windows ?

??????????????????????????????????????

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

??

?Command-Line Format ? --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.

The RELOAD or FLUSH_TABLES privilege is required with --single-transaction if both gtid_mode=ON and --set-gtid=purged=ON|AUTO. This requirement was added in MySQL 8.0.32.

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.

Before 8.0.32: Using --single-transaction together with the --set-gtid-purged option was not recommended; doing so could lead to inconsistencies in the output of mysqldump.

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, then 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:

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

To load the dump file back into the server:

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

Another way to reload the dump file:

```
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:

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

You can dump several databases with one command:

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

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

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

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

```
mysqldump --all-databases --master-data --single-transaction > all_databases.sql
```

or from MySQL 8.0.26:

```
mysqldump --all-databases --source-data --single-transaction > all_databases.sql
```

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:

```
mysqldump --all-databases --master-data=2 > all_databases.sql
```

or from MySQL 8.0.26:

```
mysqldump --all-databases --source-data=2 > all_databases.sql
```

Or:

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

or from MySQL 8.0.26:

```
mysqldump --all-databases --flush-logs --source-data=2 > all_databases.sql
```

The --source-data or --master-data option can be used simultaneously with the --single-transaction option, 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 disabling of indexes 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 25.9, ?Restrictions on Views? for a workaround.

COPYRIGHT

NOTES

1. MySQL Shell dump utilities

   https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-utilities-dump-instance-schema.html

2. MySQL Shell load dump utilities

   https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-utilities-load-dump.html

3. here

   https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-install.html

4. Downgrade Notes

   https://dev.mysql.com/doc/refman/5.7/en/downgrading-to-previous-series.html

SEE ALSO

For more information, please refer to the MySQL Reference Manual, which may already be

installed locally and which is also available online at http://dev.mysql.com/doc/.

AUTHOR

Oracle Corporation (http://dev.mysql.com/).