

MySQL Configuration Settings

Server options and system variables

MySQL maintains *well over a thousand server configuration options and system and status variables*—a blizzard of conveniences, interdependencies and opportunities for error:

- *A cross-version master list* of options and system and status variables, with nearly 1,500 entries at last count, lives *here*.
- *Server Command-line Options*: Version 8.0 has 115, listed *here* and by running `mysqld --help`.
- *Option file settings*: Most server command options are also settable in a MySQL option file (*Chapter 3*, Configuring MySQL)
- *System Variables* are readable via SELECT commands and/or by SHOW VARAIBLES. When **dynamic**, they can also be SET via SQL. Running SHOW VARIABLES in 8.0 returns 513 entries. The *Dynamic System Variables page for 5.7* has 284 entries..

Looking a little deeper, we find ...

- Some options are *both* Server Command Options *and* Dynamic Variables, but *beware, they are written differently: with dashes on the command line and in option files, but with underscores when SELECTed or SET*.
- Some can be set only in the server connection string; some can be set only by recompiling the server;
- Variables change significantly from MySQL release to release, as you might expect in a dynamic product; some vanish outright.
- Running `mysqladmin -uUSR -pPWD variables` lists most settings and options. More than half are dynamic. From 5.1.22 to 5.7.6, an `information_schema` query retrieves the same variables; since 5.7.6, query `performance_schema` instead:

```
SELECT variable_name, variable_value
FROM performance_schema.{global|session}_variables
[WHERE variable_name LIKE '...'] [ORDER BY variable_name];
```

Since 8.0, find where a current setting came from by querying `performance_schema.variable_info`.

Table B-1 is a quick-lookup vault for how to read and set MySQL configuration variables.

GLOBAL variable settings (Table B-1 *Type G*) go away when the server shuts down, LOCAL (SESSION) settings (Table B-1 *Type L*) go away with the connection—both unless, since 8.0, set with PERSIST (see below). The usual syntax for *retrieving* SELECTable variable values is ...

SELECT @@[GLOBAL|LOCAL|SESSION].variable_name;

but in some cases, for example `autocommit`, the leading @@ is optional.

Some system variables are dynamic (can be set with SQL), some not. There are two syntaxes for *SETTING* dynamic system variables:

SET [GLOBAL|SESSION|LOCAL] mysql_option=value[,...]

SET @@[GLOBAL.|SESSION.|LOCAL.]mysql_option=value[,...]

SET ONE_SHOT was removed. Since version 8.0, SET...PERSIST persists the setting by writing it to *<datadir>/mysqld-auto.cnf*.

SET ... in the *Syntax for Setting* column of Table B-1 means you can use the above SET syntaxes; [SET ...] means the variable can also be set in an option file or on the command line. Beware exceptions, e.g. in SET NAMES and SET PASSWORD, omit '='. Table B-1 conventions:

- The *Syntax for Setting* column shows '#' for numeric values, '.' for string and date values, and follows these rules:

<i>If the variable ...</i>	<i>then Syntax for Setting shows...</i>	<i>and these conventions apply...</i>
can be SET with SQL ...	SET ... varName=value	If the variable is only GLOBAL or LOCAL, modifier is usually optional. But beware of SET syntax variation from variable to variable!
can be set in an option file or on the server command-line ...	varName [=value]	On the command line, prefix with --. In an option file don't, and spell with underscores not hyphens. <i>Abbr</i> column shows abbreviation if there is one.
can be set by both the above ...	[SET ...] varName [=value]	<i>Type</i> in Table B-1 indicates what [SET ...] accepts.
is set on off by --[skip-]variablename ...	variablename skip-variablename	On the command line, preface with two hyphens. Not in an option file. Most skip-varname settings are reported only by varname.
can be spelled with dashes or underscores ...		Table B-1 shows underscores
cannot be set...	Blank	

- The *Abbr* column shows the abbreviation, if any, for setting the variable on the server command line.
- The *Type* column shows G if the variable can be GLOBAL, L if it can be LOCAL | SESSION.
- The *Shown by Select* column indicates whether the variable is shown by SELECT [@@GLOBAL.|LOCAL.] ... syntax. SELECT ... without GLOBAL | LOCAL | SESSION, MySQL returns the LOCAL value if it exists, otherwise it returns the GLOBAL value.
- The *Shown by Show Variables* column indicates whether SHOW [GLOBAL | LOCAL] VARIABLES lists the variable.
- The *Shown by mysqld -help* column shows whether the command *mysqld --verbose --help* lists the variable.
- The *Shown by mysqladmin variables* column shows whether the command *mysqladmin variables* lists the variable.

Security-related variable names are shown in red, replication-related variable names in plum; obsolete/removed settings are greyed out.

Nobody can remember all this. Keep this compendium close at hand.

Table B-1: How to set and retrieve MySQL system variables

<i>Variable Name</i>	<i>Syntax for Setting</i>	<i>Abbr</i>	<i>Type</i>	<i>Shown by</i>			<i>Meaning</i>	<i>Default value</i>
				SELECT	mysqladmin variables	mysqld --help*		

Variable Name	Syntax for Setting	Abbr	Type	Shown by				Meaning	Default value
				SELECT	mysqldadmin variables	mysqld --help*	Show Variables		
abort-slave-event-count	(mysql-test only) abort-slave-event-count		L	No	No	mysql-test only	Yes	In mysql-test only, count slave aborts	
allow-suspicious-udfs	allow-suspicious-udfs			No	No	Yes	No	Allow UDFs having symbol xxx() and no matching xxx_init() or xxx_deinit(). This permits load any function from any lib, eg exit() from libc.so	OFF
ansi	ansi	-a		No	No	No	No	Run in ANSI mode, where <ul style="list-style-type: none"> • <code> </code> concatenates strings and does not mean OR. • There may be any number of spaces between a function name and its left parenthesis, so all function names are treated as reserved words. • <code>"</code> is an identifier quote character like the single quote, not a quote character for literal strings. • REAL means FLOAT, not DOUBLE. • Default isolation level is SERIALIZABLE. Setting <code>ansi</code> is equivalent to: <code>sql-mode = REAL_AS_FLOAT, PIPES_AS_CONCAT, ANSI_QUOTES, IGNORE_SPACE</code> .	OFF
audit_log_format	audit_log_format=OLD NEW		G	No	Yes	Yes	Yes	Audit log format, requires audit log plugin. 5.7.3.	NEW
audit_log_policy	[SET...] audit_log_policy=ALL NONE LOGINS QUERIES		G	Yes	Yes	Yes	Yes	What the audit log should log. 5.7.3.	ALL
audit_log_rotate_on_size	[SET...] audit_log_rotate_on_size=#		G	Yes	Yes	Yes	Yes	Auto-rotate log past this size, 0=not. 5.7.3.	0
audit_log_strategy	audit_log_strategy=ASYNCHRONOUS PERFORMANCE SEMISYNCHRONOUS SYNCHRONOUS		G	No	Yes	Yes	Yes	ASYCNH: wait for buffer space; SEMISYNCH: synch with caching; PERF: asynch, drop if buffer full. 5.7.3.	SYNCHRONOUS
autocommit	SET LOCAL autocommit=0 1		L	Yes	No	Yes	Yes	Set=0 to enable transaction blocking, set =1 to make all changes instantly permanent.	1
automatic_sp_privileges	SET GLOBAL automatic_sp_privileges=0 1 or [skip-]automatic-skip-privileges		G	Yes	Yes	Yes	Yes	Automatically grant and revoke EXECUTE and ALTER ROUTINE privileges on CREATE and DROP.	TRUE
auto_increment_increment	[SET ...] auto_increment_increment=#		L,G	Yes	Yes	Yes	Yes	auto_increment integer increment value	1
auto_increment_offset	[SET ...] auto_increment_offset=#		L,G	Yes	Yes	Yes	Yes	Starting auto_increment value	1
backupdir	backupdir=.		G	Yes	Yes	Yes	Yes	Path for BACKUP DATABASE, RESTORE. (6.0 only)	datadir
backup_history_log	backup_history_log=ON OFF		G	Yes	Yes	Yes	Yes	Log backup history? (6.0 only).	ON
backup_history_log_file	backup_history_log_file=.		G	Yes	Yes	Yes	Yes	Backup history log file (6.0 only).	
backup_progress_log	backup_progress_log=ON OFF		G	Yes	Yes	Yes	Yes	Log backup progress? (6.0 only).	ON

Variable Name	Syntax for Setting	Abbr	Type	Shown by				Meaning	Default value
				SELECT	mysqldadmin variables	mysqld --help*	Show Variables		
backup_progress_log_file	backup_progress_log=.		G	Yes	Yes	Yes	Yes	Backup progress log file (6.0 only).	
backup_wait_timeout	[SET LOCAL] backup_wait_timeout=#		L	Yes	Yes	No	No	BACKUP RESTORE timeout secs to wait. 6.0 only.	50
back_log	back_log=#			No	Yes	Yes	Yes	No. of stackable listen queue connection requests waiting for new MySQL TCP/IP connection thread.	50
basedir	basedir=.	-b	G	Yes	Yes	Yes	Yes	Root MySQL directory	c:/mysql
big_tables	SET LOCAL big_tables=0 1		L	Yes	Yes	Yes	Yes	Use temp files for result sets that do not fit in memory. May correct 'table full' errors.	0
bind_address***	bind_address=IPaddr		G	Yes	Yes	Yes	Yes	IP address for local connection, 1 only.	NULL
binlog-do-db	binlog-do-db=.			No	No	Yes	No	Log updates on master only for listed database(s)	
binlog_direct_non_transactional_updates	binlog_direct_non_transactional_updates=0 1			No	No	Yes	No	Write transaction-dependent non-transactional statements directly to binary log, from 5.1.44, 5.5.2.	0
binlog-ignore-db	binlog-ignore-db=.			No	No	Yes	No	Do not log updates on master for listed DB(s)	
binlog-row-event-max-size	binlog-row-event-max-size=#			No	No	Yes	No	Max binlog chunk size, multiples of 256. (5.1.5)	1024
binlog_cache_size	[SET ...] binlog_cache_size=#		G	Yes	Yes	Yes	Yes	SQL statement cache for binary log in transaction.	32768
binlog_checksum	[SET ...] binlog_checksum=none crc32		G	Yes	Yes	Yes	Yes	Write event checksums to binary log. Since 5.6.2.	RC32
binlog_format	[SET ...] binlog_format = 'row' 'statement' 'mixed'		G,L	Yes	Yes	Yes	Yes	Replicate by row , statement, or mixed. Since 5.1.5. See details here . SUPER required (5.1.29).	MIXED 5.1.12-28, then STATEMENT
binlog_max_flush_queue_time	[SET GLOBAL] binlog_max_flush_queue_time=#		G	Yes	Yes	Yes	Yes	Microseconds to keep transactions before flushing to binary log, 0...100,000. Since 5.6.6.	0
binlog_order_commits	[SET GLOBAL] binlog_order_commits=0 1		G	Yes	Yes	Yes	Yes	Commit transactions in order logged. Since 5.6.6.	1
binlog_row_image	[SET ...] binlog_row_image=full minimal noblob		G,L	Yes	Yes	Yes	Yes	Columns to log for row replication. Since 5.6.2.	full
block_encryption_mode	[SET ...] block_encryptiion_mode='aes-#-mode		G,L	Yes	Yes	No	Yes	AES encryption mode: # may be 128 192 256. Mode may be ECB CBC CFB CFB8 OFB (openssl) or ECB CBC (yaSSL). Since 5.6.17.	aes-256-ecb
bootstrap	bootstrap			No	No	No	No	mysql_install_db uses to create privilege tables.	
bulk_insert_buffer_size	[SET ...] bulk_insert_buffer_size=#		G,L	Yes	Yes	Yes	Yes	Size of tree cache per thread to optimse bulk insert	8 MB
character set	SET CHARACTER SET CHARSET charsetname DEFAULT	-C		No	No	No	No	SET CHARACTER SET x is equivalent to: SET character_set_client = x; SET character_set_results = x; SET collation_connection=@@collation_database;	cp1251_koi8

To read the rest of this and other chapters, *buy a copy of the book*

[TOC](#) [Previous](#) [Next](#)
