

[version 8.4](#)

[8.3](#)

[8.0](#)

[5.7](#)

[8.0 Japanese](#)

[MySQL 8.4 Reference Manual](#) / ... / [MySQL Server Administration](#) / [The MySQL Server](#) / Server System Variables

7.1.8 Server System Variables

The MySQL server maintains many system variables that affect its operation. Most system variables can be set at server startup using options on the command line or in an option file. Most of them can be changed dynamically at runtime using the [SET](#) statement, which enables you to modify operation of the server without having to stop and restart it. Some variables are read-only, and their values are determined by the system environment, by how MySQL is installed on the system, or possibly by the options used to compile MySQL. Most system variables have a default value, but there are exceptions, including read-only variables. You can also use system variable values in expressions.

Setting a global system variable runtime value normally requires the [SYSTEM_VARIABLES_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege). Setting a session system runtime variable value normally requires no special privileges and can be done by any user, although there are exceptions. For more information, see [Section 7.1.9.1, “System Variable Privileges”](#)

There are several ways to see the names and values of system variables:

To see the values that a server uses based on its compiled-in defaults and any option files that it reads, use this command:

```
mysqld --verbose --help
```

To see the values that a server uses based only on its compiled-in defaults, ignoring the settings in any option files, use this command:

```
mysqld --no-defaults --verbose --help
```

To see the current values used by a running server, use the [SHOW VARIABLES](#) statement or the Performance Schema system variable tables. See [Section 29.12.14, “Performance Schema System Variable Tables”](#).

This section provides a description of each system variable. For a system variable summary table, see [Section 7.1.5, “Server System Variable Reference”](#). For more information about manipulation of system variables, see [Section 7.1.9, “Using System Variables”](#).

For additional system variable information, see these sections:

[Section 7.1.9, “Using System Variables”](#), discusses the syntax for setting and displaying system variable values.

[Section 7.1.9.2, “Dynamic System Variables”](#), lists the variables that can be set at runtime.

Information on tuning system variables can be found in [Section 7.1.1, “Configuring the Server”](#).

[Section 17.14, “InnoDB Startup Options and System Variables”](#), lists InnoDB system variables.

[Section 25.4.3.9.2, “NDB Cluster System Variables”](#), lists system variables which are specific to NDB Cluster.

For information on server system variables specific to replication, see [Section 19.1.6, “Replication and Binary Logging Options and Variables”](#).

Note

Some of the following variable descriptions refer to “enabling” or “disabling” a variable. These variables can be enabled with the [SET](#) statement by setting them to ON or 1, or disabled by setting them to OFF or 0. Boolean variables can be set at startup to the values ON, TRUE, OFF, and FALSE (not case-sensitive), as well as 1 and 0. See [Section 6.2.2.4, “Program Option Modifiers”](#).

Some system variables control the size of buffers or caches. For a given buffer, the server might need to allocate internal data structures. These structures typically are allocated from the total memory allocated to the buffer, and the amount of space required might be platform dependent. This means that when you assign a value to a system variable that controls a buffer size, the amount of space actually available might differ from the value assigned. In some cases, the amount might be less than the value assigned. It is also possible that the server adjusts a value upward. For example, if you assign a value of 0 to a variable for which the minimal value is 1024, the server sets the value to 1024.

Values for buffer sizes, lengths, and stack sizes are given in bytes unless otherwise specified.

Note

Some system variable descriptions include a block size, in which case a value that is not an integer multiple of the stated block size is rounded down to the next lower multiple of the block size before being stored by the server, that is to [FLOOR\(value\) * block_size](#).

Example: Suppose that the block size for a given variable is given as 4096, and you set the value of the variable to 100000 (we assume that the variable's maximum value is greater than this number). Since $100000 / 4096 = 24.4140625$, the server automatically lowers the value to 98304 ($24 * 4096$) before storing it.

In some cases, the stated maximum for a variable is the maximum allowed by the MySQL parser, but is not an exact multiple of the block size. In such cases, the effective maximum is the next lower multiple of the block size.

Example: A system variable's maximum value is shown as 4294967295 ($2^{32}-1$), and its block size is 1024. $4294967295 / 1024 = 4194303.9990234375$, so if you set this variable to its stated maximum, the value actually stored is $4194303 * 1024 = 4294966272$.

Some system variables take file name values. Unless otherwise specified, the default file location is the data directory if the value is a relative path name. To specify the location explicitly, use an absolute path name. Suppose that the data directory is `/var/mysql/data`. If a file-valued variable is given as a relative path name, it is located under `/var/mysql/data`. If the value is an absolute path name, its location is as given by the path name.

[activate_all_roles_on_login](#)

Command-Line Format	<code>--activate-all-roles-on-login[={OFF ON}]</code>
System Variable	activate_all_roles_on_login
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether to enable automatic activation of all granted roles when users log in to the server:

If [activate_all_roles_on_login](#) is enabled, the server activates all roles granted to each account at login time. This takes precedence over default roles specified with [SET DEFAULT ROLE](#).

If [activate_all_roles_on_login](#) is disabled, the server activates the default roles specified with [SET DEFAULT ROLE](#), if any, at login time.

Granted roles include those granted explicitly to the user and those named in the [mandatory_roles](#) system variable value.

[activate_all_roles_on_login](#) applies only at login time, and at the beginning of execution for stored programs and views that execute in definer context. To change the active roles within a session, use [SET ROLE](#). To change the active roles for a stored program, the program body should execute [SET ROLE](#).

[admin_address](#)

Command-Line Format	<code>--admin-address=addr</code>
System Variable	admin_address
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The IP address on which to listen for TCP/IP connections on the administrative network interface (see [Section 7.1.12.1, "Connection Interfaces"](#)). There is no default [admin_address](#) value. If this variable is not specified at startup, the server maintains no administrative interface. The server also has a [bind_address](#) system variable for configuring regular (nonadministrative) client TCP/IP connections. See [Section 7.1.12.1, "Connection Interfaces"](#).

If [admin_address](#) is specified, its value must satisfy these requirements:

The value must be a single IPv4 address, IPv6 address, or host name.

The value cannot specify a wildcard address format (`*`, `0.0.0.0`, or `:::`).

The value may include a network namespace specifier.

An IP address can be specified as an IPv4 or IPv6 address. If the value is a host name, the server resolves the name to an IP address and binds to that address. If a host name resolves to multiple IP addresses, the server uses the first IPv4 address if there are any, or the first IPv6 address otherwise.

The server treats different types of addresses as follows:

If the address is an IPv4-mapped address, the server accepts TCP/IP connections for that address, in either IPv4 or IPv6 format. For example, if the server is bound to `::ffff:127.0.0.1`, clients can connect using `--host=127.0.0.1` or `--host>::ffff:127.0.0.1`.

If the address is a "regular" IPv4 or IPv6 address (such as `127.0.0.1` or `:::1`), the server accepts TCP/IP connections only for that IPv4 or IPv6 address.

These rules apply to specifying a network namespace for an address:

A network namespace can be specified for an IP address or a host name.

A network namespace cannot be specified for a wildcard IP address.

For a given address, the network namespace is optional. If given, it must be specified as a `/ns` suffix immediately following the address.

An address with no `/ns` suffix uses the host system global namespace. The global namespace is therefore the default.

An address with a `/ns` suffix uses the namespace named `ns`.

The host system must support network namespaces and each named namespace must previously have been set up. Naming a nonexistent namespace produces an error.

For additional information about network namespaces, see [Section 7.1.14, “Network Namespace Support”](#).

If binding to the address fails, the server produces an error and does not start.

The [admin_address](#) system variable is similar to the [bind_address](#) system variable that binds the server to an address for ordinary client connections, but with these differences:

[bind_address](#) permits multiple addresses. [admin_address](#) permits a single address.

[bind_address](#) permits wildcard addresses. [admin_address](#) does not.

[admin_port](#)

Command-Line Format	<code>--admin-port=port_num</code>
System Variable	admin_port
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	33062
Minimum Value	0
Maximum Value	65535

The TCP/IP port number to use for connections on the administrative network interface (see [Section 7.1.12.1, “Connection Interfaces”](#)). Setting this variable to 0 causes the default value to be used.

Setting [admin_port](#) has no effect if [admin_address](#) is not specified because in that case the server maintains no administrative network interface.

[admin_ssl_ca](#)

Command-Line Format	<code>--admin-ssl-ca=file_name</code>
System Variable	admin_ssl_ca
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The [admin_ssl_ca](#) system variable is like [ssl_ca](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_ssl_capath](#)

Command-Line Format	<code>--admin-ssl-capath=dir_name</code>
System Variable	admin_ssl_capath
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Directory name
Default Value	NULL

The [admin_ssl_capath](#) system variable is like [ssl_capath](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_ssl_cert](#)

Command-Line Format	<code>--admin-ssl-cert=file_name</code>
System Variable	admin_ssl_cert
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name

Default Value

NULL

The [admin_ssl_cert](#) system variable is like [ssl_cert](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_ssl_cipher](#)

Command-Line Format

--admin-ssl-cipher=name

System Variable

[admin_ssl_cipher](#)

Scope

Global

Dynamic

Yes

[SET VAR](#) Hint Applies

No

Type

String

Default Value

NULL

The [admin_ssl_cipher](#) system variable is like [ssl_cipher](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

The list specified by this variable may include any of the following values:

ECDHE-ECDSA-AES128-GCM-SHA256

ECDHE-ECDSA-AES256-GCM-SHA384

ECDHE-RSA-AES128-GCM-SHA256

ECDHE-RSA-AES256-GCM-SHA384

ECDHE-ECDSA-CHACHA20-POLY1305

ECDHE-RSA-CHACHA20-POLY1305

ECDHE-ECDSA-AES256-CCM

ECDHE-ECDSA-AES128-CCM

DHE-RSA-AES128-GCM-SHA256

DHE-RSA-AES256-GCM-SHA384

DHE-RSA-AES256-CCM

DHE-RSA-AES128-CCM

DHE-RSA-CHACHA20-POLY1305

Trying to include any values in the cipher list that are not shown here when setting this variable raises an error ([ER_BLOCKED_CIPHER](#)).

[admin_ssl_crl](#)

Command-Line Format

--admin-ssl-crl=file_name

System Variable

[admin_ssl_crl](#)

Scope

Global

Dynamic

Yes

[SET VAR](#) Hint Applies

No

Type

File name

Default Value

NULL

The [admin_ssl_crl](#) system variable is like [ssl_crl](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_ssl_crlpath](#)

Command-Line Format

--admin-ssl-crlpath=dir_name

System Variable

[admin_ssl_crlpath](#)

Scope

Global

Dynamic

Yes

[SET VAR](#) Hint Applies

No

Type

Directory name

Default Value

NULL

The [admin_ssl_crlpath](#) system variable is like [ssl_crlpath](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_ssl_key](#)

Command-Line Format	--admin-ssl-key=file_name
System Variable	admin_ssl_key
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The [admin_ssl_key](#) system variable is like [ssl_key](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

[admin_tls_ciphersuites](#)

Command-Line Format	--admin-tls-ciphersuites=ciphersuite_list
System Variable	admin_tls_ciphersuites
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	NULL

The [admin_tls_ciphersuites](#) system variable is like [tls_ciphersuites](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

The value is a list of zero or more colon-separated ciphersuite names from among those listed here:

TLS_AES_128_GCM_SHA256

TLS_AES_256_GCM_SHA384

TLS_CHACHA20_POLY1305_SHA256

TLS_AES_128_CCM_SHA256

Trying to include any values in the cipher list that are not shown here when setting this variable raises an error ([ER_BLOCKED_CIPHER](#)).

[admin_tls_version](#)

Command-Line Format	--admin-tls-version=protocol_list
System Variable	admin_tls_version
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	TLSv1.2,TLSv1.3

The [admin_tls_version](#) system variable is like [tls_version](#), except that it applies to the administrative connection interface rather than the main connection interface. For information about configuring encryption support for the administrative interface, see [Administrative Interface Support for Encrypted Connections](#).

Important

MySQL 8.4 does not support the TLSv1 and TLSv1.1 connection protocols. See [Removal of Support for the TLSv1 and TLSv1.1 Protocols](#) for more information.

MySQL 8.4 supports the TLSv1.3 protocol, provided that the MySQL server was compiled using OpenSSL 1.1.1 or newer. The server checks the version of OpenSSL at startup, and if it is older than 1.1.1, TLSv1.3 is removed from the default value for the system variable. In that case, the default is TLSv1.2.

[authentication_policy](#)

Command-Line Format	--authentication-policy=value
System Variable	authentication_policy
Scope	Global

Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>*</code> , ,

This variable is used to administer multifactor authentication (MFA) capabilities. It applies to the authentication factor-related clauses of [CREATE USER](#) and [ALTER USER](#) statements used to manage MySQL account definitions, where “factor” corresponds to an authentication method or plugin associated with an account:

[authentication_policy](#) controls the number of authentication factors that accounts may have. That is, it controls which factors are required or permitted.

[authentication_policy](#) also controls, for each factor, which plugins (or methods) are permitted.

[authentication_policy](#), in conjunction with [default_authentication_plugin](#), determines the default authentication plugin for authentication specifications that do not name a plugin explicitly.

Because [authentication_policy](#) applies only when accounts are created or altered, changes to its value have no effect on existing user accounts.

Note

Although the [authentication_policy](#) system variable places certain constraints on the authentication-related clauses of [CREATE USER](#) and [ALTER USER](#) statements, a user who has the [AUTHENTICATION_POLICY_ADMIN](#) privilege is not subject to the constraints. (A warning does occur for statements that otherwise would not be permitted.)

The value of [authentication_policy](#) is a list of 1, 2, or 3 comma-separated elements. Each element present can be an authentication plugin name, an asterisk (*), empty, or missing. (Exception: Element 1 cannot be empty or missing.) In all cases, an element may be surrounded by whitespace characters and the entire list is enclosed in single quotes.

The type of value specified for element *N* in the list has implications for whether factor *N* must be present in account definitions, and which authentication plugins can be used:

If element *N* is an authentication plugin name, an authentication method for factor *N* is required and must use the named plugin.

In addition, the plugin becomes the default plugin for factor *N* authentication methods that do not name a plugin explicitly. For details, see [The Default Authentication Plugin](#).

Authentication plugins that use internal credentials storage can be specified for the first element only, and cannot repeat. For example, the following settings are not permitted:

```
authentication_policy = 'caching_sha2_password, sha256_password'
authentication_policy = 'caching_sha2_password, authentication_webauthn, sha256_password'
```

If element *N* is an asterisk (*), an authentication method for factor *N* is required. It may use any authentication plugin that is valid for element *N* (as described later).

If element *N* is empty, an authentication method for factor *N* is optional. If given, it may use any authentication plugin that is valid for element *N* (as described later).

If element *N* is missing from the list (that is, there are fewer than *N*–1 commas in the value), an authentication method for factor *N* is forbidden. For example, a value of `'*'` permits only a single factor and thus enforces single-factor authentication (1FA) for new accounts created with [CREATE USER](#) or changes to existing accounts made with [ALTER USER](#). In this case, such statements cannot specify authentication for factors 2 or 3.

When an [authentication_policy](#) element names an authentication plugin, the permitted plugin names for the element are subject to these conditions:

Element 1 must name a plugin that does not require a registration step. For example, `authentication_webauthn` cannot be named.

Elements 2 and 3 must name a plugin that does not use internal credentials storage.

For information about which authentication plugins use internal credentials storage, see [Section 8.2.15, “Password Management”](#).

When [authentication_policy](#) element *N* is `*`, the permitted plugin names for factor *N* in account definitions are subject to these conditions:

For factor 1, account definitions can use any plugin. Default authentication plugin rules apply for authentication specifications that do not name a plugin. See [The Default Authentication Plugin](#).

For factors 2 and 3, account definitions cannot name a plugin that uses internal credentials storage. For example, with `'*,*,*'`, [authentication_policy](#) settings, plugins that use internal credentials storage are only permitted for the first factor and cannot repeat.

When [authentication_policy](#) element *N* is empty, the permitted plugin names for factor *N* in account definitions are subject to these conditions:

- For factor 1, this does not apply because element 1 cannot be empty.
- For factors 2 and 3, account definitions cannot name a plugin that uses internal credentials storage.

Empty elements must occur at the end of the list, following a nonempty element. In other words, the first element cannot be empty, and either no element is empty or the last element is empty or the last two elements are empty. For example, a value of ' , , ' is not permitted because it would signify that all factors are optional. That cannot be; accounts must have at least one authentication factor.

The default value of [authentication_policy](#) is ' * , , '. This means that factor 1 is required in account definitions and can use any authentication plugin, and that factors 2 and 3 are optional and each can use any authentication plugin that does not use internal credentials storage.

The following table shows some [authentication_policy](#) values and the policy that each establishes for creating or altering accounts.

Table 7.4 Example authentication_policy Values

authentication_policy Value	Effective Policy
' * '	Permit only creating or altering accounts with one factor.
' * , * '	Permit only creating or altering accounts with two factors.
' * , * , * '	Permit only creating or altering accounts with three factors.
' * , '	Permit creating or altering accounts with one or two factors.
' * , , '	Permit creating or altering accounts with one, two, or three factors.
' * , * , '	Permit creating or altering accounts with two or three factors.
' * , auth_plugin '	Permit creating or altering accounts with two factors, where the first factor can be any authentication method, and the second factor must be the named plugin.
' auth_plugin , * , '	Permit creating or altering accounts with two or three factors, where the first factor must be the named plugin.
' auth_plugin , '	Permit creating or altering accounts with one or two factors, where the first factor must be the named plugin.
' auth_plugin , auth_plugin , auth_plugin '	Permits creating or altering accounts with three factors, where the factors must use the named plugins.

[authentication_windows_log_level](#)

Command-Line Format	--authentication-windows-log-level=#
System Variable	authentication_windows_log_level
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	2
Minimum Value	0
Maximum Value	4

This variable is available only if the `authentication_windows` Windows authentication plugin is enabled and debugging code is enabled. See [Section 8.4.1.6, "Windows Pluggable Authentication"](#).

This variable sets the logging level for the Windows authentication plugin. The following table shows the permitted values.

Value	Description
0	No logging
1	Log only error messages
2	Log level 1 messages and warning messages
3	Log level 2 messages and information notes
4	Log level 3 messages and debug messages

[authentication_windows_use_principal_name](#)

Command-Line Format	--authentication-windows-use-principal-name[={OFF ON}]
System Variable	authentication_windows_use_principal_name
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This variable is available only if the `authentication_windows` Windows authentication plugin is enabled. See [Section 8.4.1.6, “Windows Pluggable Authentication”](#).

A client that authenticates using the `InitSecurityContext()` function should provide a string identifying the service to which it connects (*targetName*). MySQL uses the principal name (UPN) of the account under which the server is running. The UPN has the form *user_id@computer_name* and need not be registered anywhere to be used. This UPN is sent by the server at the beginning of authentication handshake.

This variable controls whether the server sends the UPN in the initial challenge. By default, the variable is enabled. For security reasons, it can be disabled to avoid sending the server's account name to a client as cleartext. If the variable is disabled, the server always sends a 0x00 byte in the first challenge, the client does not specify *targetName*, and as a result, NTLM authentication is used.

If the server fails to obtain its UPN (which happens primarily in environments that do not support Kerberos authentication), the UPN is not sent by the server and NTLM authentication is used.

[autocommit](#)

Command-Line Format	<code>--autocommit[={OFF ON}]</code>
System Variable	autocommit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

The autocommit mode. If set to 1, all changes to a table take effect immediately. If set to 0, you must use [COMMIT](#) to accept a transaction or [ROLLBACK](#) to cancel it. If [autocommit](#) is 0 and you change it to 1, MySQL performs an automatic [COMMIT](#) of any open transaction. Another way to begin a transaction is to use a [START TRANSACTION](#) or [BEGIN](#) statement. See [Section 15.3.1, “START TRANSACTION, COMMIT, and ROLLBACK Statements”](#).

By default, client connections begin with [autocommit](#) set to 1. To cause clients to begin with a default of 0, set the global [autocommit](#) value by starting the server with the `--autocommit=0` option. To set the variable using an option file, include these lines:

```
[mysqld]
autocommit=0
```

[automatic_sp_privileges](#)

Command-Line Format	<code>--automatic-sp-privileges[={OFF ON}]</code>
System Variable	automatic_sp_privileges
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

When this variable has a value of 1 (the default), the server automatically grants the [EXECUTE](#) and [ALTER ROUTINE](#) privileges to the creator of a stored routine, if the user cannot already execute and alter or drop the routine. (The [ALTER ROUTINE](#) privilege is required to drop the routine.) The server also automatically drops those privileges from the creator when the routine is dropped. If [automatic_sp_privileges](#) is 0, the server does not automatically add or drop these privileges.

The creator of a routine is the account used to execute the `CREATE` statement for it. This might not be the same as the account named as the `DEFINER` in the routine definition.

If you start `mysqld` with `--skip-new`, [automatic_sp_privileges](#) is set to OFF.

See also [Section 27.2.2, “Stored Routines and MySQL Privileges”](#).

[auto_generate_certs](#)

Command-Line Format	<code>--auto-generate-certs[={OFF ON}]</code>
System Variable	auto_generate_certs
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This variable controls whether the server autogenerates SSL key and certificate files in the data directory, if they do not already exist.

At startup, the server automatically generates server-side and client-side SSL certificate and key files in the data directory if the [auto_generate_certs](#) system variable is enabled and the server-side SSL files are missing from the data directory. These certificates are always generated in such cases, regardless of the values of any other TLS options. The certificate and key files enable secure client connections using SSL; see [Section 8.3.1, “Configuring MySQL to Use Encrypted Connections”](#).

For more information about SSL file autogeneration, including file names and characteristics, see [Section 8.3.3.1, “Creating SSL and RSA Certificates and Keys using MySQL”](#).

The [sha256_password_auto_generate_rsa_keys](#) and [caching_sha2_password_auto_generate_rsa_keys](#) system variables are related but control autogeneration of RSA key-pair files needed for secure password exchange using RSA over unencrypted connections.

[back_log](#)

Command-Line Format	--back-log=#
System Variable	back_log
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	-1 (signifies autosizing; do not assign this literal value)
Minimum Value	1
Maximum Value	65535

The number of outstanding connection requests MySQL can have. This comes into play when the main MySQL thread gets very many connection requests in a very short time. It then takes some time (although very little) for the main thread to check the connection and start a new thread. The [back_log](#) value indicates how many requests can be stacked during this short time before MySQL momentarily stops answering new requests. You need to increase this only if you expect a large number of connections in a short period of time.

In other words, this value is the size of the listen queue for incoming TCP/IP connections. Your operating system has its own limit on the size of this queue. The manual page for the Unix `listen()` system call should have more details. Check your OS documentation for the maximum value for this variable. [back_log](#) cannot be set higher than your operating system limit.

The default value is the value of [max_connections](#), which enables the permitted backlog to adjust to the maximum permitted number of connections.

[basedir](#)

Command-Line Format	--basedir=dir_name
System Variable	basedir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name
Default Value	parent of mysqld installation directory

The path to the MySQL installation base directory.

[big_tables](#)

Command-Line Format	--big-tables[={OFF ON}]
System Variable	big_tables
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If enabled, the server stores all temporary tables on disk rather than in memory. This prevents most The table *tbl_name* is full errors for [SELECT](#) operations that require a large temporary table, but also slows down queries for which in-memory tables would suffice.

The default value for new connections is `OFF` (use in-memory temporary tables). Normally, it should never be necessary to enable this variable. When in-memory *internal* temporary tables are managed by the `TempTable` storage engine (the default), and the maximum amount of memory that can be occupied by the `TempTable` storage engine is exceeded, the `TempTable` storage engine starts storing data to temporary files on disk. When in-memory temporary tables are managed by the `MEMORY` storage engine, in-memory tables are automatically converted to disk-based tables as required. For more information, see [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

[bind_address](#)

Command-Line Format	--bind-address=addr
System Variable	bind_address
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	*

The MySQL server listens on one or more network sockets for TCP/IP connections. Each socket is bound to one address, but it is possible for an address to map onto multiple network interfaces. To specify how the server should listen for TCP/IP connections, set the [bind_address](#) system variable at server

startup. The server also has an [admin_address](#) system variable that enables administrative connections on a dedicated interface. See [Section 7.1.12.1, "Connection Interfaces"](#).

If [bind_address](#) is specified, it accepts a list of one or more address values, each of which may specify a single non-wildcard IP address or host name. Each address may include a network namespace specifier. If only one address is specified, it may make use of one of the wildcard address formats that permit listening on multiple network interfaces (*, 0.0.0.0, or :). Multiple addresses are separated by commas. When multiple values are listed, each value must specify a single non-wildcard IP address (either IPv4 or IPv6) or a host name, and wildcard address formats (*, 0.0.0.0, or :) are not allowed.

IP addresses can be specified as IPv4 or IPv6 addresses. For any value that is a host name, the server resolves the name to an IP address and binds to that address. If a host name resolves to multiple IP addresses, the server uses the first IPv4 address if there are any, or the first IPv6 address otherwise.

The server treats different types of addresses as follows:

If the address is *, the server accepts TCP/IP connections on all server host IPv4 interfaces, and, if the server host supports IPv6, on all IPv6 interfaces. Use this address to permit both IPv4 and IPv6 connections on all server interfaces. This value is the default. If the variable specifies a list of multiple values, this value is not permitted.

If the address is 0.0.0.0, the server accepts TCP/IP connections on all server host IPv4 interfaces. If the variable specifies a list of multiple values, this value is not permitted.

If the address is :, the server accepts TCP/IP connections on all server host IPv4 and IPv6 interfaces. If the variable specifies a list of multiple values, this value is not permitted.

If the address is an IPv4-mapped address, the server accepts TCP/IP connections for that address, in either IPv4 or IPv6 format. For example, if the server is bound to ::ffff:127.0.0.1, clients can connect using --host=127.0.0.1 or --host>::ffff:127.0.0.1.

If the address is a "regular" IPv4 or IPv6 address (such as 127.0.0.1 or ::1), the server accepts TCP/IP connections only for that IPv4 or IPv6 address.

These rules apply to specifying a network namespace for an address:

A network namespace can be specified for an IP address or a host name.

A network namespace cannot be specified for a wildcard IP address.

For a given address, the network namespace is optional. If given, it must be specified as a /ns suffix immediately following the address.

An address with no /ns suffix uses the host system global namespace. The global namespace is therefore the default.

An address with a /ns suffix uses the namespace named ns.

The host system must support network namespaces and each named namespace must previously have been set up. Naming a nonexistent namespace produces an error.

If the variable value specifies multiple addresses, it can include addresses in the global namespace, in named namespaces, or a mix.

For additional information about network namespaces, see [Section 7.1.14, "Network Namespace Support"](#).

If binding to any address fails, the server produces an error and does not start.

Examples:

```
bind_address=*
```

The server listens on all IPv4 or IPv6 addresses, as specified by the * wildcard.

```
bind_address=198.51.100.20
```

The server listens only on the 198.51.100.20 IPv4 address.

```
bind_address=198.51.100.20,2001:db8:0:f101::1
```

The server listens on the 198.51.100.20 IPv4 address and the 2001:db8:0:f101::1 IPv6 address.

```
bind_address=198.51.100.20,*
```

This produces an error because wildcard addresses are not permitted when [bind_address](#) names a list of multiple values.

```
bind_address=198.51.100.20/red,2001:db8:0:f101::1/blue,192.0.2.50
```

The server listens on the 198.51.100.20 IPv4 address in the red namespace, the 2001:db8:0:f101::1 IPv6 address in the blue namespace, and the 192.0.2.50 IPv4 address in the global namespace.

When [bind_address](#) names a single value (wildcard or non-wildcard), the server listens on a single socket, which for a wildcard address may be bound to multiple network interfaces. When [bind_address](#) names a list of multiple values, the server listens on one socket per value, with each socket bound to a single network interface. The number of sockets is linear with the number of values specified. Depending on operating system connection-acceptance efficiency, long value lists might incur a performance penalty for accepting TCP/IP connections.

Because file descriptors are allocated for listening sockets and network namespace files, it may be necessary to increase the [open_files_limit](#) system variable.

If you intend to bind the server to a specific address, be sure that the `mysql.user` system table contains an account with administrative privileges that you can use to connect to that address. Otherwise, you cannot shut down the server. For example, if you bind the server to `*`, you can connect to it using all existing accounts. But if you bind the server to `:::1`, it accepts connections only on that address. In that case, first make sure that the `'root'@':::1'` account is present in the `mysql.user` table so you can still connect to the server to shut it down.

[block_encryption_mode](#)

Command-Line Format	<code>--block-encryption-mode=#</code>
System Variable	block_encryption_mode
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>aes-128-ecb</code>

This variable controls the block encryption mode for block-based algorithms such as AES. It affects encryption for [AES_ENCRYPT\(\)](#) and [AES_DECRYPT\(\)](#).

[block_encryption_mode](#) takes a value in `aes-keylen-mode` format, where *keylen* is the key length in bits and *mode* is the encryption mode. The value is not case-sensitive. Permitted *keylen* values are 128, 192, and 256. Permitted *mode* values are ECB, CBC, CFB1, CFB8, CFB128, and OFB.

For example, this statement causes the AES encryption functions to use a key length of 256 bits and the CBC mode:

```
SET block_encryption_mode = 'aes-256-cbc';
```

An error occurs for attempts to set [block_encryption_mode](#) to a value containing an unsupported key length or a mode that the SSL library does not support.

[build_id](#)

System Variable	build_id
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Linux

This is a 160-bit SHA1 signature which is generated by the linker when compiling the server on Linux systems with [-DWITH_BUILD_ID=ON](#) (enabled by default), and converted to a hexadecimal string. This read-only value serves as a unique build ID, and is written into the server log at startup.

`build_id` is not supported on platforms other than Linux.

[bulk_insert_buffer_size](#)

Command-Line Format	<code>--bulk-insert-buffer-size=#</code>
System Variable	bulk_insert_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	<code>8388608</code>
Minimum Value	<code>0</code>
Maximum Value (64-bit platforms)	<code>18446744073709551615</code>
Maximum Value (32-bit platforms)	<code>4294967295</code>
Unit	bytes/thread

MyISAM uses a special tree-like cache to make bulk inserts faster for [INSERT ... SELECT](#), `INSERT ... VALUES (...), (...), ...,` and [LOAD DATA](#) when adding data to nonempty tables. This variable limits the size of the cache tree in bytes per thread. Setting it to 0 disables this optimization. The default value is 8MB.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[caching_sha2_password_digest_rounds](#)

Command-Line Format	<code>--caching-sha2-password-digest-rounds=#</code>
System Variable	caching_sha2_password_digest_rounds

Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	5000
Minimum Value	5000
Maximum Value	4095000

The number of hash rounds used by the `caching_sha2_password` authentication plugin for password storage.

Increasing the number of hashing rounds above the default value incurs a performance penalty that correlates with the amount of increase:

Creating an account that uses the `caching_sha2_password` plugin has no impact on the client session within which the account is created, but the server must perform the hashing rounds to complete the operation.

For client connections that use the account, the server must perform the hashing rounds and save the result in the cache. The result is longer login time for the first client connection, but not for subsequent connections. This behavior occurs after each server restart.

[caching_sha2_password_auto_generate_rsa_keys](#)

Command-Line Format	<code>--caching-sha2-password-auto-generate-rsa-keys[={OFF ON}]</code>
System Variable	caching_sha2_password_auto_generate_rsa_keys
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

The server uses this variable to determine whether to autogenerate RSA private/public key-pair files in the data directory if they do not already exist.

At startup, the server automatically generates RSA private/public key-pair files in the data directory if all of these conditions are true: The [sha256_password_auto_generate_rsa_keys](#) or [caching_sha2_password_auto_generate_rsa_keys](#) system variable is enabled; no RSA options are specified; the RSA files are missing from the data directory. These key-pair files enable secure password exchange using RSA over unencrypted connections for accounts authenticated by the `sha256_password` (deprecated) or `caching_sha2_password` plugin; see [Section 8.4.1.3, “SHA-256 Pluggable Authentication”](#), and [Section 8.4.1.2, “Caching SHA-2 Pluggable Authentication”](#).

For more information about RSA file autogeneration, including file names and characteristics, see [Section 8.3.3.1, “Creating SSL and RSA Certificates and Keys using MySQL”](#)

The [auto_generate_certs](#) system variable is related but controls autogeneration of SSL certificate and key files needed for secure connections using SSL.

[caching_sha2_password_private_key_path](#)

Command-Line Format	<code>--caching-sha2-password-private-key-path=file_name</code>
System Variable	caching_sha2_password_private_key_path
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name
Default Value	<code>private_key.pem</code>

This variable specifies the path name of the RSA private key file for the `caching_sha2_password` authentication plugin. If the file is named as a relative path, it is interpreted relative to the server data directory. The file must be in PEM format.

Important

Because this file stores a private key, its access mode should be restricted so that only the MySQL server can read it.

For information about `caching_sha2_password`, see [Section 8.4.1.2, “Caching SHA-2 Pluggable Authentication”](#).

[caching_sha2_password_public_key_path](#)

Command-Line Format	<code>--caching-sha2-password-public-key-path=file_name</code>
System Variable	caching_sha2_password_public_key_path
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name
Default Value	<code>public_key.pem</code>

This variable specifies the path name of the RSA public key file for the `caching_sha2_password` authentication plugin. If the file is named as a relative path, it is interpreted relative to the server data directory. The file must be in PEM format.

For information about `caching_sha2_password`, including information about how clients request the RSA public key, see [Section 8.4.1.2, “Caching SHA-2 Pluggable Authentication”](#).

[character_set_client](#)

System Variable	character_set_client
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	utf8mb4

The character set for statements that arrive from the client. The session value of this variable is set using the character set requested by the client when the client connects to the server. (Many clients support a `--default-character-set` option to enable this character set to be specified explicitly. See also [Section 12.4, “Connection Character Sets and Collations”](#).) The global value of the variable is used to set the session value in cases when the client-requested value is unknown or not available, or the server is configured to ignore client requests. This can happen when the client requests a character set not known to the server, such as when a Japanese-enabled client requests `sjis` when connecting to a server not configured with `sjis` support.

Some character sets cannot be used as the client character set. Attempting to use them as the [character_set_client](#) value produces an error. See [Impermissible Client Character Sets](#).

[character_set_connection](#)

System Variable	character_set_connection
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	utf8mb4

The character set used for literals specified without a character set introducer and for number-to-string conversion. For information about introducers, see [Section 12.3.8, “Character Set Introducers”](#).

[character_set_database](#)

System Variable	character_set_database
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	utf8mb4
Footnote	This option is dynamic, but should be set only by server. You should not set this variable manually.

The character set used by the default database. The server sets this variable whenever the default database changes. If there is no default database, the variable has the same value as [character_set_server](#).

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

The global [character_set_database](#) and [collation_database](#) system variables are deprecated; expect them to be removed in a future version of MySQL.

Assigning a value to the session [character_set_database](#) and [collation_database](#) system variables is deprecated and assignments produce a warning. Expect the session variables to become read-only (and assignments to them to produce an error) in a future version of MySQL in which it remains possible to access the session variables to determine the database character set and collation for the default database.

[character_set_filesystem](#)

Command-Line Format	<code>--character-set-filesystem=name</code>
System Variable	character_set_filesystem
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	binary

The file system character set. This variable is used to interpret string literals that refer to file names, such as in the [LOAD DATA](#) and [SELECT ... INTO OUTFILE](#) statements and the [LOAD_FILE\(\)](#) function. Such file names are converted from [character_set_client](#) to [character_set_filesystem](#).

before the file opening attempt occurs. The default value is `binary`, which means that no conversion occurs. For systems on which multibyte file names are permitted, a different value may be more appropriate. For example, if the system represents file names using UTF-8, set [character_set_filesystem](#) to `'utf8mb4'`.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, "System Variable Privileges"](#).

[character_set_results](#)

System Variable	character_set_results
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>utf8mb4</code>

The character set used for returning query results to the client. This includes result data such as column values, result metadata such as column names, and error messages.

[character_set_server](#)

Command-Line Format	<code>--character-set-server=name</code>
System Variable	character_set_server
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>utf8mb4</code>

The server's default character set. See [Section 12.15, "Character Set Configuration"](#). If you set this variable, you should also set [collation_server](#) to specify the collation for the character set.

[character_set_system](#)

System Variable	character_set_system
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	<code>utf8mb3</code>

The character set used by the server for storing identifiers. The value is always `utf8mb3`.

[character_sets_dir](#)

Command-Line Format	<code>--character-sets-dir=dir_name</code>
System Variable	character_sets_dir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name

The directory where character sets are installed. See [Section 12.15, "Character Set Configuration"](#).

[check_proxy_users](#)

Command-Line Format	<code>--check-proxy-users[={OFF ON}]</code>
System Variable	check_proxy_users
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	<code>OFF</code>

Some authentication plugins implement proxy user mapping for themselves (for example, the PAM and Windows authentication plugins). Other authentication plugins do not support proxy users by default. Of these, some can request that the MySQL server itself map proxy users according to granted proxy privileges: `mysql_native_password` (deprecated), `sha256_password` (deprecated).

If the [check_proxy_users](#) system variable is enabled, the server performs proxy user mapping for any authentication plugins that make such a request. However, it may also be necessary to enable plugin-specific system variables to take advantage of server proxy user mapping support:

For the deprecated `mysql_native_password` plugin (deprecated), enable [mysql_native_password_proxy_users](#).

For the sha256_password plugin (deprecated), enable [sha256_password_proxy_users](#).

For information about user proxying, see [Section 8.2.19, “Proxy Users”](#).

[collation_connection](#)

System Variable	collation_connection
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

The collation of the connection character set. [collation_connection](#) is important for comparisons of literal strings. For comparisons of strings with column values, [collation_connection](#) does not matter because columns have their own collation, which has a higher collation precedence (see [Section 12.8.4, “Collation Coercibility in Expressions”](#)).

Using the name of a user-defined collation for this variable raises a warning.

[collation_database](#)

System Variable	collation_database
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	utf8mb4_0900_ai_ci
Footnote	This option is dynamic, but should be set only by server. You should not set this variable manually.

The collation used by the default database. The server sets this variable whenever the default database changes. If there is no default database, the variable has the same value as [collation_server](#).

The global [character_set_database](#) and [collation_database](#) system variables are deprecated; expect them to be removed in a future version of MySQL.

Assigning a value to the session [character_set_database](#) and [collation_database](#) system variables is deprecated and assignments produce a warning. Expect the session variables to become read-only (and assignments to produce an error) in a future version of MySQL in which it remains possible to access the session variables to determine the database character set and collation for the default database.

Using the name of a user-defined collation for [collation_database](#) raises a warning.

[collation_server](#)

Command-Line Format	--collation-server=name
System Variable	collation_server
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	utf8mb4_0900_ai_ci

The server's default collation. See [Section 12.15, “Character Set Configuration”](#).

Setting this to the name of a user-defined collation raises a warning.

[completion_type](#)

Command-Line Format	--completion-type=#
System Variable	completion_type
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	NO_CHAIN

NO_CHAIN

CHAIN

RELEASE

Valid Values

0

1

2

The transaction completion type. This variable can take the values shown in the following table. The variable can be assigned using either the name values or corresponding integer values.

Value	Description
NO_CHAIN (or 0)	COMMIT and ROLLBACK are unaffected. This is the default value.
CHAIN (or 1)	COMMIT and ROLLBACK are equivalent to COMMIT AND CHAIN and ROLLBACK AND CHAIN, respectively. (A new transaction starts immediately with the same isolation level as the just-terminated transaction.)
RELEASE (or 2)	COMMIT and ROLLBACK are equivalent to COMMIT RELEASE and ROLLBACK RELEASE, respectively. (The server disconnects after terminating the transaction.)

[completion_type](#) affects transactions that begin with [START TRANSACTION](#) or [BEGIN](#) and end with [COMMIT](#) or [ROLLBACK](#). It does not apply to implicit commits resulting from execution of the statements listed in [Section 15.3.3, "Statements That Cause an Implicit Commit"](#). It also does not apply for [XA COMMIT](#), [XA ROLLBACK](#), or when [autocommit=1](#).

[component_scheduler.enabled](#)

Command-Line Format

--component-scheduler.enabled[=value]

System Variable

[component_scheduler.enabled](#)

Scope

Global

Dynamic

Yes

[SET VAR](#) Hint Applies

No

Type

Boolean

Default Value

ON

When set to OFF at startup, the background thread does not start. Tasks can still be scheduled, but they do not run until [component_scheduler](#) is enabled. When set to ON at startup, the component is fully operational.

It is also possible to set the value dynamically to get the following effects:

ON starts the background thread that begins servicing the queue immediately.

OFF signals a termination of the background thread, which waits for it to end. The background thread checks the termination flag before accessing the queue to check for tasks to execute.

[concurrent_insert](#)

Command-Line Format

--concurrent-insert[=value]

System Variable

[concurrent_insert](#)

Scope

Global

Dynamic

Yes

[SET VAR](#) Hint Applies

No

Type

Enumeration

Default Value

AUTO

NEVER

AUTO

ALWAYS

Valid Values

0

1

2

If AUTO (the default), MySQL permits [INSERT](#) and [SELECT](#) statements to run concurrently for MyISAM tables that have no free blocks in the middle of the data file.

This variable can take the values shown in the following table. The variable can be assigned using either the name values or corresponding integer values.

Value

Description

NEVER (or 0)	Disables concurrent inserts
AUTO (or 1)	(Default) Enables concurrent insert for MyISAM tables that do not have holes
	Enables concurrent inserts for all MyISAM tables, even those that have holes. For a table with a hole, new rows are inserted at the end of the table if it is in use by another thread. Otherwise, MySQL acquires a normal write lock and inserts the row into the hole.
ALWAYS (or 2)	

If you start `mysqld` with `--skip-new, concurrent_insert` is set to NEVER.

See also [Section 10.11.3, “Concurrent Inserts”](#).

[connect_timeout](#)

Command-Line Format	<code>--connect-timeout=#</code>
System Variable	connect_timeout
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	10
Minimum Value	2
Maximum Value	31536000
Unit	seconds

The number of seconds that the `mysqld` server waits for a connect packet before responding with Bad handshake. The default value is 10 seconds.

Increasing the `connect_timeout` value might help if clients frequently encounter errors of the form `Lost connection to MySQL server at 'XXX', system error: errno`.

[connection_memory_chunk_size](#)

Command-Line Format	<code>--connection-memory-chunk-size=#</code>
System Variable	connection_memory_chunk_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8192
Minimum Value	0
Maximum Value	536870912
Unit	bytes

Set the chunking size for updates to the global memory usage counter [Global_connection_memory](#). The status variable is updated only when total memory consumption by all user connections changes by more than this amount. Disable updates by setting `connection_memory_chunk_size = 0`.

The memory calculation is exclusive of any memory used by system users such as the MySQL root user. Memory used by the [InnoDB](#) buffer pool is also not included.

You must have the [SYSTEM_VARIABLES_ADMIN](#) or [SUPER](#) privilege to set this variable.

[connection_memory_limit](#)

Command-Line Format	<code>--connection-memory-limit=#</code>
System Variable	connection_memory_limit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	18446744073709551615
Minimum Value	2097152
Maximum Value	18446744073709551615
Unit	bytes

Set the maximum amount of memory that can be used by a single user connection. If any user connection uses more than this amount, all queries from this connection are rejected with [ER_CONN_LIMIT](#), including any queries currently running.

The limit set by this variable does not apply to system users, or to the MySQL root account. Memory used by the [InnoDB](#) buffer pool is also not included.

You must have the [SYSTEM_VARIABLES_ADMIN](#) or [SUPER](#) privilege to set this variable.

[core_file](#)

System Variable	core_file
------------------------	---------------------------

Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether to write a core file if the server unexpectedly exits. This variable is set by the [--core-file](#) option.

[create_admin_listener_thread](#)

Command-Line Format	--create-admin-listener-thread[={OFF ON}]
System Variable	create_admin_listener_thread
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether to use a dedicated listening thread for client connections on the administrative network interface (see [Section 7.1.12.1, “Connection Interfaces”](#)). The default is `OFF`; that is, the manager thread for ordinary connections on the main interface also handles connections for the administrative interface.

Depending on factors such as platform type and workload, you may find one setting for this variable yields better performance than the other setting.

Setting [create_admin_listener_thread](#) has no effect if [admin_address](#) is not specified because in that case the server maintains no administrative network interface.

[cte_max_recursion_depth](#)

Command-Line Format	--cte-max-recursion-depth=#
System Variable	cte_max_recursion_depth
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1000
Minimum Value	0
Maximum Value	4294967295

The common table expression (CTE) maximum recursion depth. The server terminates execution of any CTE that recurses more levels than the value of this variable. For more information, see [Limiting Common Table Expression Recursion](#).

[datadir](#)

Command-Line Format	--datadir=dir_name
System Variable	datadir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name

The path to the MySQL server data directory. Relative paths are resolved with respect to the current directory. If you expect the server to be started automatically (that is, in contexts for which you cannot know the current directory in advance), it is best to specify the [datadir](#) value as an absolute path.

[debug](#)

Command-Line Format	--debug[=debug_options]
System Variable	debug
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value (Unix)	d:t:i:o,/tmp/mysql.trace
Default Value (Windows)	d:t:i:o,\mysql.trace

This variable indicates the current debugging settings. It is available only for servers built with debugging support. The initial value comes from the value of instances of the [--debug](#) option given at server startup. The global and session values may be set at runtime.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

Assigning a value that begins with + or - cause the value to added to or subtracted from the current value:

```
mysql> SET debug = 'T';
mysql> SELECT @@debug;
+-----+
| @@debug |
+-----+
| T       |
+-----+

mysql> SET debug = '+P';
mysql> SELECT @@debug;
+-----+
| @@debug |
+-----+
| P:T     |
+-----+

mysql> SET debug = '-P';
mysql> SELECT @@debug;
+-----+
| @@debug |
+-----+
| T       |
+-----+
```

For more information, see [Section 7.9.4, “The DBUG Package”](#).

[debug_sync](#)

System Variable	debug_sync
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

This variable is the user interface to the Debug Sync facility. Use of Debug Sync requires that MySQL be configured with the [-DWITH_DEBUG=ON](#) **CMake** option (see [Section 2.8.7, “MySQL Source-Configuration Options”](#)); otherwise, this system variable is not available.

The global variable value is read only and indicates whether the facility is enabled. By default, Debug Sync is disabled and the value of [debug_sync](#) is OFF. If the server is started with [--debug-sync-timeout=N](#), where *N* is a timeout value greater than 0, Debug Sync is enabled and the value of [debug_sync](#) is ON - current signal followed by the signal name. Also, *N* becomes the default timeout for individual synchronization points.

The session value can be read by any user and has the same value as the global variable. The session value can be set to control synchronization points.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

For a description of the Debug Sync facility and how to use synchronization points, see [MySQL Internals: Test Synchronization](#).

[default_authentication_plugin](#)

The default authentication plugin. This must be a plugin that uses internal credentials storage, so these values are permitted:

- caching_sha2_password: Use SHA-256 passwords; see [Section 8.4.1.2, “Caching SHA-2 Pluggable Authentication”](#).
- mysql_native_password (deprecated): Use MySQL native passwords; see [Section 8.4.1.1, “Native Pluggable Authentication”](#).
- sha256_password (deprecated): Use SHA-256 passwords; see [Section 8.4.1.3, “SHA-256 Pluggable Authentication”](#).

For information about which authentication plugins use internal credentials storage, see [Section 8.2.15, “Password Management”](#).

[default_authentication_plugin](#) is used in conjunction with but at a lower precedence than the [authentication_policy](#) system variable. For details, see [The Default Authentication Plugin](#). [default_authentication_plugin](#) is deprecated, and subject to removal in a future MySQL version.

[default_collation_for_utf8mb4](#)

System Variable	default_collation_for_utf8mb4
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration

Valid Values

utf8mb4_0900_ai_ci

utf8mb4_general_ci

Important

The `default_collation_for_utf8mb4` system variable is for internal use by MySQL Replication only.

This variable is set by the server to the default collation for the `utf8mb4` character set. The value of the variable is replicated from a source to a replica so that the replica can correctly process data originating from a source with a different default collation for `utf8mb4`. This variable is primarily intended to support replication from a MySQL 5.7 or older replication source server to a later MySQL replica server, or group replication with a MySQL 5.7 primary node and one or more MySQL 8.0 or later secondaries. The default collation for `utf8mb4` in MySQL 5.7 is `utf8mb4_general_ci`, but `utf8mb4_0900_ai_ci` in later release series. The variable is not present in releases earlier than MySQL 8.0, so if the replica does not receive a value for the variable, it assumes the source is from an earlier release and sets the value to the previous default collation `utf8mb4_general_ci`.

The default `utf8mb4` collation is used in the following statements:

[SHOW COLLATION](#) and [SHOW CHARACTER SET](#).

[CREATE TABLE](#) and [ALTER TABLE](#) having a `CHARACTER SET utf8mb4` clause without a `COLLATION` clause, either for the table character set or for a column character set.

[CREATE DATABASE](#) and [ALTER DATABASE](#) having a `CHARACTER SET utf8mb4` clause without a `COLLATION` clause.

Any statement containing a string literal of the form `_utf8mb4'some text'` without a `COLLATE` clause.

See also [Section 12.9, "Unicode Support"](#).

[default_password_lifetime](#)

Command-Line Format	<code>--default-password-lifetime=#</code>
System Variable	default_password_lifetime
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	65535
Unit	days

This variable defines the global automatic password expiration policy. The default [default_password_lifetime](#) value is 0, which disables automatic password expiration. If the value of [default_password_lifetime](#) is a positive integer *N*, it indicates the permitted password lifetime; passwords must be changed every *N* days.

The global password expiration policy can be overridden as desired for individual accounts using the password expiration option of the [CREATE USER](#) and [ALTER USER](#) statements. See [Section 8.2.15, "Password Management"](#).

[default_storage_engine](#)

Command-Line Format	<code>--default-storage-engine=name</code>
System Variable	default_storage_engine
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	InnoDB

The default storage engine for tables. See [Chapter 18, Alternative Storage Engines](#). This variable sets the storage engine for permanent tables only. To set the storage engine for `TEMPORARY` tables, set the [default_tmp_storage_engine](#) system variable.

To see which storage engines are available and enabled, use the [SHOW ENGINES](#) statement or query the `INFORMATION_SCHEMA` [ENGINES](#) table.

If you disable the default storage engine at server startup, you must set the default engine for both permanent and `TEMPORARY` tables to a different engine, or else the server does not start.

[default_table_encryption](#)

Command-Line Format	<code>--default-table-encryption[={OFF ON}]</code>
System Variable	default_table_encryption
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes

Type	Boolean
Default Value	OFF

Defines the default encryption setting applied to schemas and general tablespaces when they are created without specifying an `ENCRYPTION` clause.

The [default_table_encryption](#) variable is only applicable to user-created schemas and general tablespaces. It does not govern encryption of the `mysql` system tablespace.

Setting the runtime value of [default_table_encryption](#) requires the [SYSTEM_VARIABLES_ADMIN](#) and [TABLE_ENCRYPTION_ADMIN](#) privileges, or the deprecated [SUPER](#) privilege.

[default_table_encryption](#) supports [SET PERSIST](#) and [SET PERSIST ONLY](#) syntax. See [Section 7.1.9.3, “Persisted System Variables”](#).

For more information, see [Defining an Encryption Default for Schemas and General Tablespaces](#).

[default_tmp_storage_engine](#)

Command-Line Format	<code>--default-tmp-storage-engine=name</code>
System Variable	default_tmp_storage_engine
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Enumeration
Default Value	InnoDB

The default storage engine for `TEMPORARY` tables (created with [CREATE TEMPORARY TABLE](#)). To set the storage engine for permanent tables, set the [default_storage_engine](#) system variable. Also see the discussion of that variable regarding possible values.

If you disable the default storage engine at server startup, you must set the default engine for both permanent and `TEMPORARY` tables to a different engine, or else the server does not start.

[default_week_format](#)

Command-Line Format	<code>--default-week-format=#</code>
System Variable	default_week_format
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	7

The default mode value to use for the [WEEK\(\)](#) function. See [Section 14.7, “Date and Time Functions”](#).

[delay_key_write](#)

Command-Line Format	<code>--delay-key-write[={OFF ON ALL}]</code>
System Variable	delay_key_write
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	ON
	OFF
Valid Values	ON
	ALL

This variable specifies how to use delayed key writes. It applies only to `MyISAM` tables. Delayed key writing causes key buffers not to be flushed between writes. See also [Section 18.2.1, “MyISAM Startup Options”](#).

This variable can have one of the following values to affect handling of the `DELAY_KEY_WRITE` table option that can be used in [CREATE TABLE](#) statements.

Option	Description
OFF	<code>DELAY_KEY_WRITE</code> is ignored.
ON	MySQL honors any <code>DELAY_KEY_WRITE</code> option specified in CREATE TABLE statements. This is the default value.
ALL	All new opened tables are treated as if they were created with the <code>DELAY_KEY_WRITE</code> option enabled.

Note

If you set this variable to `ALL`, you should not use `MyISAM` tables from within another program (such as another `MySQL` server or [myisamchk](#)) when the tables are in use. Doing so leads to index corruption.

If `DELAY_KEY_WRITE` is enabled for a table, the key buffer is not flushed for the table on every index update, but only when the table is closed. This speeds up writes on keys a lot, but if you use this feature, you should add automatic checking of all `MyISAM` tables by starting the server with the [myisam_recover_options](#) system variable set (for example, [myisam_recover_options='BACKUP,FORCE'](#)). See [Section 7.1.8, “Server System Variables”](#), and [Section 18.2.1, “MyISAM Startup Options”](#).

If you start `mysqld` with [--skip-new](#), [delay_key_write](#) is set to `OFF`.

Warning

If you enable external locking with [--external-locking](#), there is no protection against index corruption for tables that use delayed key writes.

[delayed_insert_limit](#)

Command-Line Format	<code>--delayed-insert-limit=#</code>
Deprecated	Yes
System Variable	delayed_insert_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	100
Minimum Value	1
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

This system variable is deprecated (because `DELAYED` inserts are not supported), and you should expect it to be removed in a future release.

[delayed_insert_timeout](#)

Command-Line Format	<code>--delayed-insert-timeout=#</code>
Deprecated	Yes
System Variable	delayed_insert_timeout
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	300
Minimum Value	1
Maximum Value	31536000
Unit	seconds

This system variable is deprecated (because `DELAYED` inserts are not supported), and you should expect it to be removed in a future release.

[delayed_queue_size](#)

Command-Line Format	<code>--delayed-queue-size=#</code>
Deprecated	Yes
System Variable	delayed_queue_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1000
Minimum Value	1
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

This system variable is deprecated (because `DELAYED` inserts are not supported), and you should expect it to be removed in a future release.

[disabled_storage_engines](#)

Command-Line Format	<code>--disabled-storage-engines=engine[,engine]...</code>
System Variable	disabled_storage_engines
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	empty string

This variable indicates which storage engines cannot be used to create tables or tablespaces. For example, to prevent new `MyISAM` or `FEDERATED` tables from being created, start the server with these lines in the server option file:

```
[mysqld]
disabled_storage_engines="MyISAM,FEDERATED"
```

By default, [disabled_storage_engines](#) is empty (no engines disabled), but it can be set to a comma-separated list of one or more engines (not case-sensitive). Any engine named in the value cannot be used to create tables or tablespaces with [CREATE TABLE](#) or [CREATE TABLESPACE](#), and cannot be used with [ALTER TABLE ... ENGINE](#) or [ALTER TABLESPACE ... ENGINE](#) to change the storage engine of existing tables or tablespaces. Attempts to do so result in an [ER_DISABLED_STORAGE_ENGINE](#) error.

[disabled_storage_engines](#) does not restrict other DDL statements for existing tables, such as [CREATE INDEX](#), [TRUNCATE TABLE](#), [ANALYZE TABLE](#), [DROP TABLE](#), or [DROP TABLESPACE](#). This permits a smooth transition so that existing tables or tablespaces that use a disabled engine can be migrated to a permitted engine by means such as [ALTER TABLE ... ENGINE permitted_engine](#).

It is permitted to set the [default_storage_engine](#) or [default_tmp_storage_engine](#) system variable to a storage engine that is disabled. This could cause applications to behave erratically or fail, although that might be a useful technique in a development environment for identifying applications that use disabled engines, so that they can be modified.

[disabled_storage_engines](#) is disabled and has no effect if the server is started with any of these options: [--initialize](#), [--initialize-insecure](#), [--skip-grant-tables](#).

[disconnect_on_expired_password](#)

Command-Line Format	<code>--disconnect-on-expired-password[={OFF ON}]</code>
System Variable	disconnect_on_expired_password
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This variable controls how the server handles clients with expired passwords:

If the client indicates that it can handle expired passwords, the value of [disconnect_on_expired_password](#) is irrelevant. The server permits the client to connect but puts it in sandbox mode.

If the client does not indicate that it can handle expired passwords, the server handles the client according to the value of [disconnect_on_expired_password](#):

If [disconnect_on_expired_password](#) is enabled, the server disconnects the client.

If [disconnect_on_expired_password](#) is disabled, the server permits the client to connect but puts it in sandbox mode.

For more information about the interaction of client and server settings relating to expired-password handling, see [Section 8.2.16, "Server Handling of Expired Passwords"](#).

[div_precision_increment](#)

Command-Line Format	<code>--div-precision-increment=#</code>
System Variable	div_precision_increment
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	4
Minimum Value	0
Maximum Value	30

This variable indicates the number of digits by which to increase the scale of the result of division operations performed with the `/` operator. The default value is 4. The minimum and maximum values are 0 and 30, respectively. The following example illustrates the effect of increasing the default value.

```
mysql> SELECT 1/7;
+-----+
| 1/7    |
+-----+
| 0.1429 |
+-----+
mysql> SET div_precision_increment = 12;
mysql> SELECT 1/7;
+-----+
```

```
| 1/7 |
+-----+
| 0.142857142857 |
+-----+
```

[dragnet.log_error_filter_rules](#)

Command-Line Format	--dragnet.log-error-filter-rules=value
System Variable	dragnet.log_error_filter_rules
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	IF prio>=INFORMATION THEN drop. IF EXISTS source_line THEN unset source_line.

The filter rules that control operation of the `log_filter_dragnet` error log filter component. If `log_filter_dragnet` is not installed, [dragnet.log_error_filter_rules](#) is unavailable. If `log_filter_dragnet` is installed but not enabled, changes to [dragnet.log_error_filter_rules](#) have no effect.

The effect of the default value is similar to the filtering performed by the `log_sink_internal` filter with a setting of [log_error_verbosity=2](#).

[dragnet.Status](#) status variable can be consulted to determine the result of the most recent assignment to [dragnet.log_error_filter_rules](#).

[enterprise_encryption.maximum_rsa_key_size](#)

Command-Line Format	--enterprise-encryption.maximum-rsa-key-size=#
System Variable	enterprise_encryption.maximum_rsa_key_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	4096
Minimum Value	2048
Maximum Value	16384

This variable limits the maximum size of RSA keys generated by MySQL Enterprise Encryption. The variable is available only if the MySQL Enterprise Encryption component `component_enterprise_encryption` is installed.

The lowest setting is 2048 bits, which is the minimum RSA key length that is acceptable by current best practice. The default setting is 4096 bits. The highest setting is 16384 bits. Generating longer keys can consume significant CPU resources, so you can use this setting to limit keys to a length that provides adequate security for your requirements while balancing this with resource usage. See [Section 8.6.2, “Configuring MySQL Enterprise Encryption”](#) for more information.

[enterprise_encryption.rsa_support_legacy_padding](#)

Command-Line Format	--enterprise-encryption.rsa_support_legacy_padding[={OFF ON}]
System Variable	enterprise_encryption.rsa_support_legacy_padding
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls whether encrypted data and signatures that MySQL Enterprise Encryption produced using the old `openssl_udf` shared library functions can be decrypted or verified by the MySQL Enterprise Encryption component (`component_enterprise_encryption`). The variable is available only if the MySQL Enterprise Encryption component is installed.

For the component functions to support decryption and verification for content produced by the old `openssl_udf` shared library functions, you must set the system variable padding to `ON`. When `ON` is set, if the component functions cannot decrypt or verify content when assuming it has the RSAES-OAEP or RSASSA-PSS scheme (as used by the component), they make another attempt assuming it has the RSAES-PKCS1-v1_5 or RSASSA-PKCS1-v1_5 scheme (as used by the `openssl_udf` shared library functions). When `OFF` is set, if the component functions cannot decrypt or verify content using their normal schemes, they return null output. See [Section 8.6.2, “Configuring MySQL Enterprise Encryption”](#) for more information.

[end_markers_in_json](#)

Command-Line Format	--end-markers-in-json[={OFF ON}]
System Variable	end_markers_in_json
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean

Default Value	OFF
---------------	-----

Whether optimizer JSON output should add end markers. See [MySQL Internals: The end_markers_in_json System Variable](#).

[eq_range_index_dive_limit](#)

Command-Line Format	--eq-range-index-dive-limit=#
System Variable	eq_range_index_dive_limit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	200
Minimum Value	0
Maximum Value	4294967295

This variable indicates the number of equality ranges in an equality comparison condition when the optimizer should switch from using index dives to index statistics in estimating the number of qualifying rows. It applies to evaluation of expressions that have either of these equivalent forms, where the optimizer uses a nonunique index to look up *col_name* values:

```
col_name IN(val1, ..., valN)
col_name = val1 OR ... OR col_name = valN
```

In both cases, the expression contains *N* equality ranges. The optimizer can make row estimates using index dives or index statistics. If [eq_range_index_dive_limit](#) is greater than 0, the optimizer uses existing index statistics instead of index dives if there are [eq_range_index_dive_limit](#) or more equality ranges. Thus, to permit use of index dives for up to *N* equality ranges, set [eq_range_index_dive_limit](#) to *N* + 1. To disable use of index statistics and always use index dives regardless of *N*, set [eq_range_index_dive_limit](#) to 0.

For more information, see [Equality Range Optimization of Many-Valued Comparisons](#).

To update table index statistics for best estimates, use [ANALYZE TABLE](#).

[error_count](#)

The number of errors that resulted from the last statement that generated messages. This variable is read only. See [Section 15.7.7.18, “SHOW ERRORS Statement”](#).

[event_scheduler](#)

Command-Line Format	--event-scheduler[=value]
System Variable	event_scheduler
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	ON
	ON
Valid Values	OFF
	DISABLED

This variable enables or disables, and starts or stops, the Event Scheduler. The possible status values are ON, OFF, and DISABLED. Turning the Event Scheduler OFF is not the same as disabling the Event Scheduler, which requires setting the status to DISABLED. This variable and its effects on the Event Scheduler's operation are discussed in greater detail in [Section 27.4.2, “Event Scheduler Configuration”](#)

[explain_format](#)

Command-Line Format	--explain-format=format
System Variable	explain_format
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	TRADITIONAL
	TRADITIONAL (DEFAULT)
Valid Values	JSON
	TREE

This variable determines the default output format used by [EXPLAIN](#) in the absence of a FORMAT option when displaying a query execution plan. Possible values and their effects are listed here:

TRADITIONAL: Use MySQL's traditional table-based output, as if `FORMAT=TRADITIONAL` had been specified as part of the `EXPLAIN` statement. This is the variable's default value. `DEFAULT` is also supported as a synonym for `TRADITIONAL`, and has exactly the same effect.

Note

`DEFAULT` cannot be used as part of an `EXPLAIN` statement's `FORMAT` option.

JSON: Use the JSON output format, as if `FORMAT=JSON` had been specified.

TREE: Use the tree-based output format, as if `FORMAT=TREE` had been specified.

The setting for this variable also affects `EXPLAIN ANALYZE`. For this purpose, `DEFAULT` and `TRADITIONAL` are interpreted as `TREE`. If the value of `explain_format` is `JSON` and an `EXPLAIN ANALYZE` statement having no `FORMAT` option is issued, the statement raises an error ([ER_NOT_SUPPORTED_YET](#)).

Using a format specifier with `EXPLAIN` or `EXPLAIN ANALYZE` overrides any setting for `explain_format`.

The `explain_format` system variable has no effect on `EXPLAIN` output when this statement is used to display information about table columns.

Setting the session value of `explain_format` requires no special privileges; setting it on the global level requires [SYSTEM_VARIABLES_ADMIN](#) (or the deprecated [SUPER](#) privilege). See [Section 7.1.9.1, "System Variable Privileges"](#).

For more information and examples, see [Obtaining Execution Plan Information](#).

[explain_json_format_version](#)

Command-Line Format	--explain-json-format-version=#
System Variable	explain_json_format_version
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1
Minimum Value	1
Maximum Value	2

Determines the version of the JSON output format used by `EXPLAIN FORMAT=JSON` statements. Setting this variable to 1 causes the server to use Version 1, which is the linear format used for output from such statements in older versions of MySQL; this is the default in MySQL 8.4. Setting `explain_json_format_version` to 2 causes the Version 2 format to be used; this JSON output format is based on access paths, and is intended to provide better compatibility with future versions of the MySQL Optimizer.

For an example of use, see [Obtaining Execution Plan Information](#).

[explicit_defaults_for_timestamp](#)

Command-Line Format	--explicit-defaults-for-timestamp[={OFF ON}]
Deprecated	Yes
System Variable	explicit_defaults_for_timestamp
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This system variable determines whether the server enables certain nonstandard behaviors for default values and `NULL`-value handling in [TIMESTAMP](#) columns. By default, [explicit_defaults_for_timestamp](#) is enabled, which disables the nonstandard behaviors. Disabling [explicit_defaults_for_timestamp](#) results in a warning.

If [explicit_defaults_for_timestamp](#) is disabled, the server enables the nonstandard behaviors and handles [TIMESTAMP](#) columns as follows:

[TIMESTAMP](#) columns not explicitly declared with the `NULL` attribute are automatically declared with the `NOT NULL` attribute. Assigning such a column a value of `NULL` is permitted and sets the column to the current timestamp. *Exception:* Attempting to insert `NULL` into a generated column declared as `TIMESTAMP NOT NULL` is rejected with an error.

The first [TIMESTAMP](#) column in a table, if not explicitly declared with the `NULL` attribute or an explicit `DEFAULT` or `ON UPDATE` attribute, is automatically declared with the `DEFAULT CURRENT_TIMESTAMP` and `ON UPDATE CURRENT_TIMESTAMP` attributes.

[TIMESTAMP](#) columns following the first one, if not explicitly declared with the `NULL` attribute or an explicit `DEFAULT` attribute, are automatically declared as `DEFAULT '0000-00-00 00:00:00'` (the "zero" timestamp). For inserted rows that specify no explicit value for such a column, the column is assigned `'0000-00-00 00:00:00'` and no warning occurs.

Depending on whether strict SQL mode or the [NO_ZERO_DATE](#) SQL mode is enabled, a default value of '0000-00-00 00:00:00' may be invalid. Be aware that the [TRADITIONAL](#) SQL mode includes strict mode and [NO_ZERO_DATE](#). See [Section 7.1.11, “Server SQL Modes”](#).

The nonstandard behaviors just described are deprecated; expect them to be removed in a future MySQL release.

If [explicit_defaults_for_timestamp](#) is enabled, the server disables the nonstandard behaviors and handles [TIMESTAMP](#) columns as follows:

It is not possible to assign a [TIMESTAMP](#) column a value of NULL to set it to the current timestamp. To assign the current timestamp, set the column to [CURRENT_TIMESTAMP](#) or a synonym such as [NOW\(\)](#).

[TIMESTAMP](#) columns not explicitly declared with the NOT NULL attribute are automatically declared with the NULL attribute and permit NULL values. Assigning such a column a value of NULL sets it to NULL, not the current timestamp.

[TIMESTAMP](#) columns declared with the NOT NULL attribute do not permit NULL values. For inserts that specify NULL for such a column, the result is either an error for a single-row insert if strict SQL mode is enabled, or '0000-00-00 00:00:00' is inserted for multiple-row inserts with strict SQL mode disabled. In no case does assigning the column a value of NULL set it to the current timestamp.

[TIMESTAMP](#) columns explicitly declared with the NOT NULL attribute and without an explicit DEFAULT attribute are treated as having no default value. For inserted rows that specify no explicit value for such a column, the result depends on the SQL mode. If strict SQL mode is enabled, an error occurs. If strict SQL mode is not enabled, the column is declared with the implicit default of '0000-00-00 00:00:00' and a warning occurs. This is similar to how MySQL treats other temporal types such as [DATETIME](#).

No [TIMESTAMP](#) column is automatically declared with the DEFAULT CURRENT_TIMESTAMP or ON UPDATE CURRENT_TIMESTAMP attributes. Those attributes must be explicitly specified.

The first [TIMESTAMP](#) column in a table is not handled differently from [TIMESTAMP](#) columns following the first one.

If [explicit_defaults_for_timestamp](#) is disabled at server startup, this warning appears in the error log:

```
[Warning] TIMESTAMP with implicit DEFAULT value is deprecated.
Please use --explicit_defaults_for_timestamp server option (see
documentation for more details).
```

As indicated by the warning, to disable the deprecated nonstandard behaviors, enable the [explicit_defaults_for_timestamp](#) system variable at server startup.

Note

[explicit_defaults_for_timestamp](#) is itself deprecated because its only purpose is to permit control over deprecated [TIMESTAMP](#) behaviors that are to be removed in a future MySQL release. When removal of those behaviors occurs, expect [explicit_defaults_for_timestamp](#) to be removed as well.

For additional information, see [Section 13.2.5, “Automatic Initialization and Updating for TIMESTAMP and DATETIME”](#).

external_user

System Variable	external_user
Scope	Session
Dynamic	No
SET VAR Hint Applies	No
Type	String

The external user name used during the authentication process, as set by the plugin used to authenticate the client. With native (built-in) MySQL authentication, or if the plugin does not set the value, this variable is NULL. See [Section 8.2.19, “Proxy Users”](#).

flush

Command-Line Format	--flush[={OFF ON}]
System Variable	flush
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If ON, the server flushes (synchronizes) all changes to disk after each SQL statement. Normally, MySQL does a write of all changes to disk only after each SQL statement and lets the operating system handle the synchronizing to disk. See [Section B.3.3.3, “What to Do If MySQL Keeps Crashing”](#). This variable is set to ON if you start [mysqld](#) with the [--flush](#) option.

Note

If [flush](#) is enabled, the value of [flush_time](#) does not matter and changes to [flush_time](#) have no effect on flush behavior.

[flush_time](#)

Command-Line Format	--flush-time=#
System Variable	flush_time
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	31536000
Unit	seconds

If this is set to a nonzero value, all tables are closed every [flush_time](#) seconds to free up resources and synchronize unflushed data to disk. This option is best used only on systems with minimal resources.

Note

If [flush](#) is enabled, the value of [flush_time](#) does not matter and changes to [flush_time](#) have no effect on flush behavior.

[foreign_key_checks](#)

System Variable	foreign_key_checks
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	ON

If set to 1 (the default), foreign key constraints are checked. If set to 0, foreign key constraints are ignored, with a couple of exceptions. When re-creating a table that was dropped, an error is returned if the table definition does not conform to the foreign key constraints referencing the table. Likewise, an [ALTER TABLE](#) operation returns an error if a foreign key definition is incorrectly formed. For more information, see [Section 15.1.20.5, "FOREIGN KEY Constraints"](#).

Setting this variable has the same effect on [NDB](#) tables as it does for InnoDB tables. Typically you leave this setting enabled during normal operation, to enforce [referential integrity](#). Disabling foreign key checking can be useful for reloading InnoDB tables in an order different from that required by their parent/child relationships. See [Section 15.1.20.5, "FOREIGN KEY Constraints"](#).

Setting `foreign_key_checks` to 0 also affects data definition statements: [DROP SCHEMA](#) drops a schema even if it contains tables that have foreign keys that are referred to by tables outside the schema, and [DROP TABLE](#) drops tables that have foreign keys that are referred to by other tables.

Note

Setting `foreign_key_checks` to 1 does not trigger a scan of the existing table data. Therefore, rows added to the table while [foreign_key_checks = 0](#) are not verified for consistency.

Dropping an index required by a foreign key constraint is not permitted, even with [foreign_key_checks=0](#). The foreign key constraint must be removed before dropping the index.

[ft_boolean_syntax](#)

Command-Line Format	--ft-boolean-syntax=name
System Variable	ft_boolean_syntax
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	+ -><()~* : " "&

The list of operators supported by boolean full-text searches performed using `IN BOOLEAN MODE`. See [Section 14.9.2, "Boolean Full-Text Searches"](#).

The default variable value is `'+ -><()~* : " "&|'`. The rules for changing the value are as follows:

Operator function is determined by position within the string.

The replacement value must be 14 characters.

Each character must be an ASCII nonalphanumeric character.

Either the first or second character must be a space.

No duplicates are permitted except the phrase quoting operators in positions 11 and 12. These two characters are not required to be the same, but they are the only two that may be.

Positions 10, 13, and 14 (which by default are set to :, &, and |) are reserved for future extensions.

[ft_max_word_len](#)

Command-Line Format	--ft-max-word-len=#
System Variable	ft_max_word_len
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	84
Minimum Value	10
Maximum Value	84

The maximum length of the word to be included in a MyISAM FULLTEXT index.

Note

FULLTEXT indexes on MyISAM tables must be rebuilt after changing this variable. Use `REPAIR TABLE tbl_name QUICK`.

[ft_min_word_len](#)

Command-Line Format	--ft-min-word-len=#
System Variable	ft_min_word_len
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	4
Minimum Value	1
Maximum Value	82

The minimum length of the word to be included in a MyISAM FULLTEXT index.

Note

FULLTEXT indexes on MyISAM tables must be rebuilt after changing this variable. Use `REPAIR TABLE tbl_name QUICK`.

[ft_query_expansion_limit](#)

Command-Line Format	--ft-query-expansion-limit=#
System Variable	ft_query_expansion_limit
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	20
Minimum Value	0
Maximum Value	1000

The number of top matches to use for full-text searches performed using `WITH QUERY EXPANSION`.

[ft_stopword_file](#)

Command-Line Format	--ft-stopword-file=file_name
System Variable	ft_stopword_file
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

The file from which to read the list of stopwords for full-text searches on MyISAM tables. The server looks for the file in the data directory unless an absolute path name is given to specify a different directory. All the words from the file are used; comments are *not* honored. By default, a built-in list of stopwords is used (as defined in the `storage/myisam/ft_static.c` file). Setting this variable to the empty string (`' '`) disables stopwords filtering. See also [Section 14.9.4, “Full-Text Stopwords”](#).

Note

FULLTEXT indexes on MyISAM tables must be rebuilt after changing this variable or the contents of the stopwords file. Use `REPAIR TABLE tbl_name QUICK`.

[general_log](#)

Command-Line Format	--general-log[={OFF ON}]
System Variable	general_log
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether the general query log is enabled. The value can be 0 (or OFF) to disable the log or 1 (or ON) to enable the log. The destination for log output is controlled by the [log_output](#) system variable; if that value is NONE, no log entries are written even if the log is enabled.

[general_log_file](#)

Command-Line Format	--general-log-file=file_name
System Variable	general_log_file
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	host_name.log

The name of the general query log file. The default value is *host_name.log*, but the initial value can be changed with the [--general_log_file](#) option.

[generated_random_password_length](#)

Command-Line Format	--generated-random-password-length=#
System Variable	generated_random_password_length
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	20
Minimum Value	5
Maximum Value	255

The maximum number of characters permitted in random passwords generated for [CREATE USER](#), [ALTER USER](#), and [SET PASSWORD](#) statements. For more information, see [Random Password Generation](#).

[global_connection_memory_limit](#)

Command-Line Format	--global-connection-memory-limit=#
System Variable	global_connection_memory_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	18446744073709551615
Minimum Value	16777216
Maximum Value	18446744073709551615
Unit	bytes

Set the total amount of memory that can be used by all user connections; that is, [Global_connection_memory](#) should not exceed this amount. Any time that it does, all queries (including any currently running) from regular users are rejected with [ER_GLOBAL_CONN_LIMIT](#).

Memory used by the system users such as the MySQL root user is included in this total, but is not counted towards the disconnection limit; such users are never disconnected due to memory usage.

Memory used by the [InnoDB](#) buffer pool is excluded from the total.

You must have the [SYSTEM_VARIABLES_ADMIN](#) or [SUPER](#) privilege to set this variable.

[global_connection_memory_tracking](#)

Command-Line Format	--global-connection-memory-tracking={TRUE FALSE}
System Variable	global_connection_memory_tracking
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	FALSE

Determines whether the server calculates [Global connection memory](#). This variable must be enabled explicitly; otherwise, the memory calculation is not performed, and `Global_connection_memory` is not set.

You must have the [SYSTEM_VARIABLES_ADMIN](#) or [SUPER](#) privilege to set this variable.

[group_concat_max_len](#)

Command-Line Format	--group-concat-max-len=#
System Variable	group_concat_max_len
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	1024
Minimum Value	4
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

The maximum permitted result length in bytes for the [GROUP_CONCAT\(\)](#) function. The default is 1024.

[have_compress](#)

YES if the zlib compression library is available to the server, NO if not. If not, the [COMPRESS\(\)](#) and [UNCOMPRESS\(\)](#) functions cannot be used.

[have_dynamic_loading](#)

YES if [mysqld](#) supports dynamic loading of plugins, NO if not. If the value is NO, you cannot use options such as `--plugin-load` to load plugins at server startup, or the [INSTALL PLUGIN](#) statement to load plugins at runtime.

[have_geometry](#)

YES if the server supports spatial data types, NO if not.

[have_profiling](#)

YES if statement profiling capability is present, NO if not. If present, the `profiling` system variable controls whether this capability is enabled or disabled. See [Section 15.7.7.32, "SHOW PROFILES Statement"](#).

This variable is deprecated; you should expect it to be removed in a future MySQL release.

[have_query_cache](#)

[have_query_cache](#) is deprecated, always has a value of NO, and you should expect it to be removed in a future MySQL release.

[have_rtree_keys](#)

YES if RTREE indexes are available, NO if not. (These are used for spatial indexes in MyISAM tables.)

[have_statement_timeout](#)

System Variable	have_statement_timeout
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean

Whether the statement execution timeout feature is available (see [Statement Execution Time Optimizer Hints](#)). The value can be NO if the background thread used by this feature could not be initialized.

[have_symlink](#)

YES if symbolic link support is enabled, NO if not. This is required on Unix for support of the `DATA DIRECTORY` and `INDEX DIRECTORY` table options. If the server is started with the `--skip-symbolic-links` option, the value is DISABLED.

This variable has no meaning on Windows.

Note

Symbolic link support, along with the `--symbolic-links` option that controls it, is deprecated; expect these to be removed in a future version of MySQL. In addition, the option is disabled by default. The related [have_symlink](#) system variable also is deprecated and you should expect it to be removed in a future version of MySQL.

[histogram_generation_max_mem_size](#)

Command-Line Format	--histogram-generation-max-mem-size=#
System Variable	histogram_generation_max_mem_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	20000000
Minimum Value	1000000
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295
Unit	bytes

The maximum amount of memory available for generating histogram statistics. See [Section 10.9.6, “Optimizer Statistics”](#), and [Section 15.7.3.1, “ANALYZE TABLE Statement”](#).

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[host_cache_size](#)

Command-Line Format	--host-cache-size=#
System Variable	host_cache_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	-1 (signifies autosizing; do not assign this literal value)
Minimum Value	0
Maximum Value	65536

The MySQL server maintains an in-memory host cache that contains client host name and IP address information and is used to avoid Domain Name System (DNS) lookups; see [Section 7.1.12.3, “DNS Lookups and the Host Cache”](#).

The [host_cache_size](#) variable controls the size of the host cache, as well as the size of the Performance Schema [host_cache](#) table that exposes the cache contents. Setting [host_cache_size](#) has these effects:

Setting the size to 0 disables the host cache. With the cache disabled, the server performs a DNS lookup every time a client connects.

Changing the size at runtime causes an implicit host cache flushing operation that clears the host cache, truncates the [host_cache](#) table, and unblocks any blocked hosts.

The default value is autosized to 128, plus 1 for a value of [max_connections](#) up to 500, plus 1 for every increment of 20 over 500 in the [max_connections](#) value, capped to a limit of 2000.

[hostname](#)

System Variable	hostname
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The server sets this variable to the server host name at startup. The maximum length is 255 characters.

[identity](#)

This variable is a synonym for the [last_insert_id](#) variable. It exists for compatibility with other database systems. You can read its value with `SELECT @@identity`, and set it using `SET identity`.

[init_connect](#)

Command-Line Format	--init-connect=name
System Variable	init_connect
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

A string to be executed by the server for each client that connects. The string consists of one or more SQL statements, separated by semicolon characters.

For users that have the [CONNECTION_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege), the content of [init_connect](#) is not executed. This is done so that an erroneous value for [init_connect](#) does not prevent all clients from connecting. For example, the value might contain a statement that has a syntax

error, thus causing client connections to fail. Not executing [init_connect](#) for users that have the [CONNECTION_ADMIN](#) or [SUPER](#) privilege enables them to open a connection and fix the [init_connect](#) value.

[init_connect](#) execution is skipped for any client user with an expired password. This is done because such a user cannot execute arbitrary statements, and thus [init_connect](#) execution fails, leaving the client unable to connect. Skipping [init_connect](#) execution enables the user to connect and change password.

The server discards any result sets produced by statements in the value of [init_connect](#).

[information_schema_stats_expiry](#)

Command-Line Format	--information-schema-stats-expiry=#
System Variable	information_schema_stats_expiry
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	86400
Minimum Value	0
Maximum Value	31536000
Unit	seconds

Some `INFORMATION_SCHEMA` tables contain columns that provide table statistics:

```
STATISTICS.CARDINALITY
TABLES.AUTO_INCREMENT
TABLES.AVG_ROW_LENGTH
TABLES.CHECKSUM
TABLES.CHECK_TIME
TABLES.CREATE_TIME
TABLES.DATA_FREE
TABLES.DATA_LENGTH
TABLES.INDEX_LENGTH
TABLES.MAX_DATA_LENGTH
TABLES.TABLE_ROWS
TABLES.UPDATE_TIME
```

Those columns represent dynamic table metadata; that is, information that changes as table contents change.

By default, MySQL retrieves cached values for those columns from the `mysql.index_stats` and `mysql.table_stats` dictionary tables when the columns are queried, which is more efficient than retrieving statistics directly from the storage engine. If cached statistics are not available or have expired, MySQL retrieves the latest statistics from the storage engine and caches them in the `mysql.index_stats` and `mysql.table_stats` dictionary tables. Subsequent queries retrieve the cached statistics until the cached statistics expire. A server restart or the first opening of the `mysql.index_stats` and `mysql.table_stats` tables do not update cached statistics automatically.

The [information_schema_stats_expiry](#) session variable defines the period of time before cached statistics expire. The default is 86400 seconds (24 hours), but the time period can be extended to as much as one year.

To update cached values at any time for a given table, use [ANALYZE TABLE](#).

To always retrieve the latest statistics directly from the storage engine and bypass cached values, set [information_schema_stats_expiry](#) to 0.

Querying statistics columns does not store or update statistics in the `mysql.index_stats` and `mysql.table_stats` dictionary tables under these circumstances:

When cached statistics have not expired.

When [information_schema_stats_expiry](#) is set to 0.

When the server is in [read_only](#), [super_read_only](#), [transaction_read_only](#), or [innodb_read_only](#) mode.

When the query also fetches Performance Schema data.

The statistics cache may be updated during a multiple-statement transaction before it is known whether the transaction commits. As a result, the cache may contain information that does not correspond to a known committed state. This can occur with [autocommit=0](#) or after [START TRANSACTION](#).

[information_schema_stats_expiry](#) is a session variable, and each client session can define its own expiration value. Statistics that are retrieved from the storage engine and cached by one session are available to other sessions.

For related information, see [Section 10.2.3, “Optimizing INFORMATION_SCHEMA Queries”](#).

[init_file](#)

Command-Line Format	<code>--init-file=file_name</code>
System Variable	init_file
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

If specified, this variable names a file containing SQL statements to be read and executed during the startup process. The acceptable format for statements in this file support the following constructs:

`delimiter ;`, to set the statement delimiter to the `;` character.

`delimiter $$`, to set the statement delimiter to the `$$` character sequence.

Multiple statements on the same line, delimited by the current delimiter.

Multiple-line statements.

Comments from a `#` character to the end of the line.

Comments from a `--` sequence to the end of the line.

C-style comments from a `/*` sequence to the following `*/` sequence, including over multiple lines.

Multiple-line string literals enclosed within either single quote (`'`) or double quote (`"`) characters.

If the server is started with the `--initialize` or `--initialize-insecure` option, it operates in bootstrap mode and some functionality is unavailable that limits the statements permitted in the file. These include statements that relate to account management (such as [CREATE USER](#) or [GRANT](#)), replication, and global transaction identifiers. See [Section 19.1.3, "Replication with Global Transaction Identifiers"](#).

Threads created during server startup are used for tasks such as creating the data dictionary, running upgrade procedures, and creating system tables. To ensure a stable and predictable environment, these threads are executed with the server built-in defaults for some system variables, such as [sql_mode](#), [character_set_server](#), [collation_server](#), [completion_type](#), [explicit_defaults_for_timestamp](#), and [default_table_encryption](#).

These threads are also used to execute the statements in any file specified with [init_file](#) when starting the server, so such statements execute with the server's built-in default values for those system variables.

`innodb_xxx`

[InnoDB](#) system variables are listed in [Section 17.14, "InnoDB Startup Options and System Variables"](#). These variables control many aspects of storage, memory use, and I/O patterns for [InnoDB](#) tables, and are especially important now that [InnoDB](#) is the default storage engine.

[insert_id](#)

The value to be used by the following [INSERT](#) or [ALTER TABLE](#) statement when inserting an `AUTO_INCREMENT` value. This is mainly used with the binary log.

[interactive_timeout](#)

Command-Line Format	<code>--interactive-timeout=#</code>
System Variable	interactive_timeout
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	28800
Minimum Value	1
Maximum Value	31536000
Unit	seconds

The number of seconds the server waits for activity on an interactive connection before closing it. An interactive client is defined as a client that uses the `CLIENT_INTERACTIVE` option to [mysql_real_connect\(\)](#). See also [wait timeout](#).

[internal_tmp_mem_storage_engine](#)

Command-Line Format	<code>--internal-tmp-mem-storage-engine=#</code>
System Variable	internal_tmp_mem_storage_engine
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Enumeration
Default Value	TempTable

Valid Values

MEMORY

TempTable

The storage engine for in-memory internal temporary tables (see [Section 10.4.4, "Internal Temporary Table Use in MySQL"](#)). Permitted values are TempTable (the default) and MEMORY.

The [optimizer](#) uses the storage engine defined by [internal_tmp_mem_storage_engine](#) for in-memory internal temporary tables.

Configuring a session setting for [internal_tmp_mem_storage_engine](#) requires the [SESSION_VARIABLES_ADMIN](#) or [SYSTEM_VARIABLES_ADMIN](#) privilege.

[join_buffer_size](#)

Command-Line Format	--join-buffer-size=#
System Variable	join_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	262144
Minimum Value	128
Maximum Value (Windows)	4294967168
Maximum Value (Other, 64-bit platforms)	18446744073709551488
Maximum Value (Other, 32-bit platforms)	4294967168
Unit	bytes
Block Size	128

The minimum size of the buffer that is used for plain index scans, range index scans, and joins that do not use indexes and thus perform full table scans. This variable also controls the amount of memory used for hash joins. Normally, the best way to get fast joins is to add indexes. Increase the value of [join_buffer_size](#) to get a faster full join when adding indexes is not possible. One join buffer is allocated for each full join between two tables. For a complex join between several tables for which indexes are not used, multiple join buffers might be necessary.

The default is 256KB. The maximum permissible setting for [join_buffer_size](#) is 4GB–1. Larger values are permitted for 64-bit platforms (except 64-bit Windows, for which large values are truncated to 4GB–1 with a warning). The block size is 128, and a value that is not an exact multiple of the block size is rounded down to the next lower multiple of the block size by MySQL Server before storing the value for the system variable. The parser allows values up to the maximum unsigned integer value for the platform (4294967295 or $2^{32}-1$ for a 32-bit system, 18446744073709551615 or $2^{64}-1$ for a 64-bit system) but the actual maximum is a block size lower.

Unless a Block Nested-Loop or Batched Key Access algorithm is used, there is no gain from setting the buffer larger than required to hold each matching row, and all joins allocate at least the minimum size, so use caution in setting this variable to a large value globally. It is better to keep the global setting small and change the session setting to a larger value only in sessions that are doing large joins, or change the setting on a per-query basis by using a [SET_VAR](#) optimizer hint (see [Section 10.9.3, "Optimizer Hints"](#)). Memory allocation time can cause substantial performance drops if the global size is larger than needed by most queries that use it.

When Block Nested-Loop is used, a larger join buffer can be beneficial up to the point where all required columns from all rows in the first table are stored in the join buffer. This depends on the query; the optimal size may be smaller than holding all rows from the first tables.

When Batched Key Access is used, the value of [join_buffer_size](#) defines how large the batch of keys is in each request to the storage engine. The larger the buffer, the more sequential access is made to the right hand table of a join operation, which can significantly improve performance.

For additional information about join buffering, see [Section 10.2.1.7, "Nested-Loop Join Algorithms"](#). For information about Batched Key Access, see [Section 10.2.1.12, "Block Nested-Loop and Batched Key Access Joins"](#). For information about hash joins, see [Section 10.2.1.4, "Hash Join Optimization"](#).

[keep_files_on_create](#)

Command-Line Format	--keep-files-on-create[={OFF ON}]
System Variable	keep_files_on_create
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If a MyISAM table is created with no `DATA DIRECTORY` option, the `.MYD` file is created in the database directory. By default, if MyISAM finds an existing `.MYD` file in this case, it overwrites it. The same applies to `.MYI` files for tables created with no `INDEX DIRECTORY` option. To suppress this behavior, set the [keep_files_on_create](#) variable to ON (1), in which case MyISAM does not overwrite existing files and returns an error instead. The default value is OFF (0).

If a MyISAM table is created with a `DATA DIRECTORY` or `INDEX DIRECTORY` option and an existing `.MYD` or `.MYI` file is found, MyISAM always returns an error. It does not overwrite a file in the specified directory.

[key_buffer_size](#)

Command-Line Format	--key-buffer-size=#
System Variable	key_buffer_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8388608
Minimum Value	0
Maximum Value (64-bit platforms)	OS_PER_PROCESS_LIMIT
Maximum Value (32-bit platforms)	4294967295
Unit	bytes

Index blocks for MyISAM tables are buffered and are shared by all threads. [key_buffer_size](#) is the size of the buffer used for index blocks. The key buffer is also known as the key cache.

The minimum permissible setting is 0, but you cannot set [key_buffer_size](#) to 0 dynamically. A setting of 0 drops the key cache, which is not permitted at runtime. Setting [key_buffer_size](#) to 0 is permitted only at startup, in which case the key cache is not initialized. Changing the [key_buffer_size](#) setting at runtime from a value of 0 to a permitted non-zero value initializes the key cache.

[key_buffer_size](#) can be increased or decreased only in increments or multiples of 4096 bytes. Increasing or decreasing the setting by a nonconforming value produces a warning and truncates the setting to a conforming value.

The maximum permissible setting for [key_buffer_size](#) is 4GB–1 on 32-bit platforms. Larger values are permitted for 64-bit platforms. The effective maximum size might be less, depending on your available physical RAM and per-process RAM limits imposed by your operating system or hardware platform. The value of this variable indicates the amount of memory requested. Internally, the server allocates as much memory as possible up to this amount, but the actual allocation might be less.

You can increase the value to get better index handling for all reads and multiple writes; on a system whose primary function is to run MySQL using the [MyISAM](#) storage engine, 25% of the machine's total memory is an acceptable value for this variable. However, you should be aware that, if you make the value too large (for example, more than 50% of the machine's total memory), your system might start to page and become extremely slow. This is because MySQL relies on the operating system to perform file system caching for data reads, so you must leave some room for the file system cache. You should also consider the memory requirements of any other storage engines that you may be using in addition to [MyISAM](#).

For even more speed when writing many rows at the same time, use [LOCK TABLES](#). See [Section 10.2.5.1, “Optimizing INSERT Statements”](#).

You can check the performance of the key buffer by issuing a [SHOW STATUS](#) statement and examining the [Key_read_requests](#), [Key_reads](#), [Key_write_requests](#), and [Key_writes](#) status variables. (See [Section 15.7.7, “SHOW Statements”](#).) The [Key_reads](#)/[Key_read_requests](#) ratio should normally be less than 0.01. The [Key_writes](#)/[Key_write_requests](#) ratio is usually near 1 if you are using mostly updates and deletes, but might be much smaller if you tend to do updates that affect many rows at the same time or if you are using the [DELAY_KEY_WRITE](#) table option.

The fraction of the key buffer in use can be determined using [key_buffer_size](#) in conjunction with the [Key_blocks_unused](#) status variable and the buffer block size, which is available from the [key_cache_block_size](#) system variable:

```
1 - ((Key_blocks_unused * key_cache_block_size) / key_buffer_size)
```

This value is an approximation because some space in the key buffer is allocated internally for administrative structures. Factors that influence the amount of overhead for these structures include block size and pointer size. As block size increases, the percentage of the key buffer lost to overhead tends to decrease. Larger blocks results in a smaller number of read operations (because more keys are obtained per read), but conversely an increase in reads of keys that are not examined (if not all keys in a block are relevant to a query).

It is possible to create multiple MyISAM key caches. The size limit of 4GB applies to each cache individually, not as a group. See [Section 10.10.2, “The MyISAM Key Cache”](#).

[key_cache_age_threshold](#)

Command-Line Format	--key-cache-age-threshold=#
System Variable	key_cache_age_threshold
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	300
Minimum Value	100
Maximum Value (64-bit platforms)	18446744073709551516
Maximum Value (32-bit platforms)	4294967196
Block Size	100

This value controls the demotion of buffers from the hot sublist of a key cache to the warm sublist. Lower values cause demotion to happen more quickly. The minimum value is 100. The default value is 300. See [Section 10.10.2, “The MyISAM Key Cache”](#).

[key_cache_block_size](#)

Command-Line Format	--key-cache-block-size=#
System Variable	key_cache_block_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1024
Minimum Value	512
Maximum Value	16384
Unit	bytes
Block Size	512

The size in bytes of blocks in the key cache. The default value is 1024. See [Section 10.10.2, “The MyISAM Key Cache”](#).

[key_cache_division_limit](#)

Command-Line Format	--key-cache-division-limit=#
System Variable	key_cache_division_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	100
Minimum Value	1
Maximum Value	100

The division point between the hot and warm sublists of the key cache buffer list. The value is the percentage of the buffer list to use for the warm sublist. Permissible values range from 1 to 100. The default value is 100. See [Section 10.10.2, “The MyISAM Key Cache”](#).

[large_files_support](#)

System Variable	large_files_support
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean

Whether [mysqld](#) was compiled with options for large file support.

[large_pages](#)

Command-Line Format	--large-pages[={OFF ON}]
System Variable	large_pages
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Linux
Type	Boolean
Default Value	OFF

Whether large page support is enabled (via the [--large-pages](#) option). See [Section 10.12.3.3, “Enabling Large Page Support”](#).

[large_page_size](#)

System Variable	large_page_size
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	65535
Unit	bytes

If large page support is enabled, this shows the size of memory pages. Large memory pages are supported only on Linux; on other platforms, the value of this variable is always 0. See [Section 10.12.3.3, “Enabling Large Page Support”](#).

[last_insert_id](#)

The value to be returned from [LAST_INSERT_ID\(\)](#). This is stored in the binary log when you use [LAST_INSERT_ID\(\)](#) in a statement that updates a table. Setting this variable does not update the value returned by the [mysql_insert_id\(\)](#) C API function.

[lc_messages](#)

Command-Line Format	<code>--lc-messages=name</code>
System Variable	lc_messages
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	en_US

The locale to use for error messages. The default is en_US. The server converts the argument to a language name and combines it with the value of [lc_messages_dir](#) to produce the location for the error message file. See [Section 12.12, “Setting the Error Message Language”](#).

[lc_messages_dir](#)

Command-Line Format	<code>--lc-messages-dir=dir_name</code>
System Variable	lc_messages_dir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name

The directory where error messages are located. The server uses the value together with the value of [lc_messages](#) to produce the location for the error message file. See [Section 12.12, “Setting the Error Message Language”](#).

[lc_time_names](#)

Command-Line Format	<code>--lc-time-names=value</code>
System Variable	lc_time_names
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

This variable specifies the locale that controls the language used to display day and month names and abbreviations. This variable affects the output from the [DATE_FORMAT\(\)](#), [DAYNAME\(\)](#) and [MONTHNAME\(\)](#) functions. Locale names are POSIX-style values such as 'ja_JP' or 'pt_BR'. The default value is 'en_US' regardless of your system's locale setting. For further information, see [Section 12.16, “MySQL Server Locale Support”](#).

[license](#)

System Variable	license
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	GPL

The type of license the server has.

[local_infile](#)

Command-Line Format	<code>--local-infile[={OFF ON}]</code>
System Variable	local_infile
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls server-side LOCAL capability for [LOAD DATA](#) statements. Depending on the [local_infile](#) setting, the server refuses or permits local data loading by clients that have LOCAL enabled on the client side.

To explicitly cause the server to refuse or permit [LOAD DATA LOCAL](#) statements (regardless of how client programs and libraries are configured at build time or runtime), start [mysqld](#) with [local_infile](#) disabled or enabled, respectively. [local_infile](#) can also be set at runtime. For more information, see [Section 8.1.6, “Security Considerations for LOAD DATA LOCAL”](#).

[lock_wait_timeout](#)

Command-Line Format	<code>--lock-wait-timeout=#</code>
System Variable	lock_wait_timeout

Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	31536000
Minimum Value	1
Maximum Value	31536000
Unit	seconds

This variable specifies the timeout in seconds for attempts to acquire metadata locks. The permissible values range from 1 to 31536000 (1 year). The default is 31536000.

This timeout applies to all statements that use metadata locks. These include DML and DDL operations on tables, views, stored procedures, and stored functions, as well as [LOCK TABLES](#), [FLUSH TABLES WITH READ LOCK](#), and [HANDLER](#) statements.

This timeout does not apply to implicit accesses to system tables in the `mysql` database, such as grant tables modified by [GRANT](#) or [REVOKE](#) statements or table logging statements. The timeout does apply to system tables accessed directly, such as with [SELECT](#) or [UPDATE](#).

The timeout value applies separately for each metadata lock attempt. A given statement can require more than one lock, so it is possible for the statement to block for longer than the [lock_wait_timeout](#) value before reporting a timeout error. When lock timeout occurs, [ER_LOCK_WAIT_TIMEOUT](#) is reported.

[lock_wait_timeout](#) also defines the amount of time that a [LOCK INSTANCE FOR BACKUP](#) statement waits for a lock before giving up.

[locked_in_memory](#)

System Variable	locked_in_memory
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether `mysqld` was locked in memory with [--memlock](#).

[log_error](#)

Command-Line Format	<code>--log-error[=file_name]</code>
System Variable	log_error
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

The default error log destination. If the destination is the console, the value is `stderr`. Otherwise, the destination is a file and the [log_error](#) value is the file name. See [Section 7.4.2, "The Error Log"](#).

[log_error_services](#)

Command-Line Format	<code>--log-error-services=value</code>
System Variable	log_error_services
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>log_filter_internal; log_sink_internal</code>

The components to enable for error logging. The variable may contain a list with 0, 1, or many elements. In the latter case, elements may be delimited by semicolons or commas, optionally followed by space. A given setting cannot use both semicolon and comma separators. Component order is significant because the server executes components in the order listed.

Any loadable (not built in) component named in [log_error_services](#) is implicitly loaded if it is not already loaded. For more information, see [Section 7.4.2.1, "Error Log Configuration"](#).

[log_error_suppression_list](#)

Command-Line Format	<code>--log-error-suppression-list=value</code>
System Variable	log_error_suppression_list
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	<code>empty string</code>

The [log_error_suppression_list](#) system variable applies to events intended for the error log and specifies which events to suppress when they occur with a priority of WARNING or INFORMATION. For example, if a particular type of warning is considered undesirable “noise” in the error log because it occurs frequently but is not of interest, it can be suppressed. This variable affects filtering performed by the `log_filter_internal` error log filter component, which is enabled by default (see [Section 7.5.3, “Error Log Components”](#)). If `log_filter_internal` is disabled, [log_error_suppression_list](#) has no effect.

The [log_error_suppression_list](#) value may be the empty string for no suppression, or a list of one or more comma-separated values indicating the error codes to suppress. Error codes may be specified in symbolic or numeric form. A numeric code may be specified with or without the `MY-` prefix. Leading zeros in the numeric part are not significant. Examples of permitted code formats:

```
ER_SERVER_SHUTDOWN_COMPLETE
MY-000031
000031
MY-31
31
```

Symbolic values are preferable to numeric values for readability and portability. For information about the permitted error symbols and numbers, see [MySQL 8.4 Error Message Reference](#).

The effect of [log_error_suppression_list](#) combines with that of [log_error_verbosity](#). For additional information, see [Section 7.4.2.5, “Priority-Based Error Log Filtering \(log_filter_internal\)”](#).

[log_error_verbosity](#)

Command-Line Format	--log-error-verbosity=#
System Variable	log_error_verbosity
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	2
Minimum Value	1
Maximum Value	3

The [log_error_verbosity](#) system variable specifies the verbosity for handling events intended for the error log. This variable affects filtering performed by the `log_filter_internal` error log filter component, which is enabled by default (see [Section 7.5.3, “Error Log Components”](#)). If `log_filter_internal` is disabled, [log_error_verbosity](#) has no effect.

Events intended for the error log have a priority of ERROR, WARNING, or INFORMATION. [log_error_verbosity](#) controls verbosity based on which priorities to permit for messages written to the log, as shown in the following table.

log_error_verbosity Value	Permitted Message Priorities
1	ERROR
2	ERROR, WARNING
3	ERROR, WARNING, INFORMATION

There is also a priority of `SYSTEM`. System messages about non-error situations are printed to the error log regardless of the [log_error_verbosity](#) value. These messages include startup and shutdown messages, and some significant changes to settings.

The effect of [log_error_verbosity](#) combines with that of [log_error_suppression_list](#). For additional information, see [Section 7.4.2.5, “Priority-Based Error Log Filtering \(log_filter_internal\)”](#).

[log_output](#)

Command-Line Format	--log-output=name
System Variable	log_output
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Set
Default Value	FILE TABLE
Valid Values	FILE NONE

The destination or destinations for general query log and slow query log output. The value is a list one or more comma-separated words chosen from `TABLE`, `FILE`, and `NONE`. `TABLE` selects logging to the [general_log](#) and `slow_log` tables in the `mysql` system schema. `FILE` selects logging to log files. `NONE` disables logging. If `NONE` is present in the value, it takes precedence over any other words that are present. `TABLE` and `FILE` can both be given to select both log output destinations.

This variable selects log output destinations, but does not enable log output. To do that, enable the [general_log](#) and [slow_query_log](#) system variables. For FILE logging, the [general_log_file](#) and [slow_query_log_file](#) system variables determine the log file locations. For more information, see [Section 7.4.1, “Selecting General Query Log and Slow Query Log Output Destinations”](#).

[log_queries_not_using_indexes](#)

Command-Line Format	<code>--log-queries-not-using-indexes[={OFF ON}]</code>
System Variable	log_queries_not_using_indexes
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If you enable this variable with the slow query log enabled, queries that are expected to retrieve all rows are logged. See [Section 7.4.5, “The Slow Query Log”](#). This option does not necessarily mean that no index is used. For example, a query that uses a full index scan uses an index but would be logged because the index would not limit the number of rows.

[log_raw](#)

Command-Line Format	<code>--log-raw[={OFF ON}]</code>
System Variable	log_raw
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

The [log_raw](#) system variable is initially set to the value of the `--log-raw` option. See the description of that option for more information. The system variable may also be set at runtime to change password masking behavior.

[log_slow_admin_statements](#)

Command-Line Format	<code>--log-slow-admin-statements[={OFF ON}]</code>
System Variable	log_slow_admin_statements
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Include slow administrative statements in the statements written to the slow query log. Administrative statements include [ALTER TABLE](#), [ANALYZE TABLE](#), [CHECK TABLE](#), [CREATE INDEX](#), [DROP INDEX](#), [OPTIMIZE TABLE](#), and [REPAIR TABLE](#).

[log_slow_extra](#)

Command-Line Format	<code>--log-slow-extra[={OFF ON}]</code>
System Variable	log_slow_extra
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If the slow query log is enabled and the output destination includes FILE, the server writes additional fields to log file lines that provide information about slow statements. See [Section 7.4.5, “The Slow Query Log”](#). TABLE output is unaffected.

[log_timestamps](#)

Command-Line Format	<code>--log-timestamps=#</code>
System Variable	log_timestamps
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	UTC
	UTC
Valid Values	SYSTEM

This variable controls the time zone of timestamps in messages written to the error log, and in general query log and slow query log messages written to files. It does not affect the time zone of general query log and slow query log messages written to tables (`mysql.general_log`, `mysql.slow_log`). Rows retrieved from those tables can be converted from the local system time zone to any desired time zone with [CONVERT_TZ\(\)](#) or by setting the session

[time_zone](#) system variable.

Permitted [log_timestamps](#) values are UTC (the default) and SYSTEM (the local system time zone).

Timestamps are written using ISO 8601 / RFC 3339 format: `YYYY-MM-DDThh:mm:ss.aaaaaa` plus a tail value of `Z` signifying Zulu time (UTC) or `±hh:mm` (an offset from UTC).

[log_throttle_queries_not_using_indexes](#)

Command-Line Format	<code>--log-throttle-queries-not-using-indexes=#</code>
System Variable	log_throttle_queries_not_using_indexes
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	4294967295

If [log_queries_not_using_indexes](#) is enabled, the [log_throttle_queries_not_using_indexes](#) variable limits the number of such queries per minute that can be written to the slow query log. A value of 0 (the default) means “no limit”. For more information, see [Section 7.4.5, “The Slow Query Log”](#).

[long_query_time](#)

Command-Line Format	<code>--long-query-time=#</code>
System Variable	long_query_time
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Numeric
Default Value	10
Minimum Value	0
Maximum Value	31536000
Unit	seconds

If a query takes longer than this many seconds, the server increments the [Slow_queries](#) status variable. If the slow query log is enabled, the query is logged to the slow query log file. This value is measured in real time, not CPU time, so a query that is under the threshold on a lightly loaded system might be above the threshold on a heavily loaded one. The minimum and default values of [long_query_time](#) are 0 and 10, respectively. The maximum is 31536000, which is 365 days in seconds. The value can be specified to a resolution of microseconds. See [Section 7.4.5, “The Slow Query Log”](#).

Smaller values of this variable result in more statements being considered long-running, with the result that more space is required for the slow query log. For very small values (less than one second), the log may grow quite large in a small time. Increasing the number of statements considered long-running may also result in false positives for the “excessive Number of Long Running Processes” alert in MySQL Enterprise Monitor, especially if Group Replication is enabled. For these reasons, very small values should be used in test environments only, or, in production environments, only for a short period.

[mysqldump](#) performs a full table scan, which means its queries can often exceed a [long_query_time](#) setting that is useful for regular queries. If you want to exclude most or all of the queries generated by [mysqldump](#) from the slow query log, you can use `--mysqld-long-query-time` to change the session value of the system variable to a higher value.

[low_priority_updates](#)

Command-Line Format	<code>--low-priority-updates[={OFF ON}]</code>
System Variable	low_priority_updates
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If set to 1, all [INSERT](#), [UPDATE](#), [DELETE](#), and `LOCK TABLE WRITE` statements wait until there is no pending [SELECT](#) or `LOCK TABLE READ` on the affected table. The same effect can be obtained using `{INSERT | REPLACE | DELETE | UPDATE} LOW_PRIORITY ...` to lower the priority of only one query. This variable affects only storage engines that use only table-level locking (such as MyISAM, MEMORY, and MERGE). See [Section 10.11.2, “Table Locking Issues”](#).

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[lower_case_file_system](#)

System Variable	lower_case_file_system
Scope	Global
Dynamic	No

[SET VAR](#) Hint Applies
Type

No
Boolean

This variable describes the case sensitivity of file names on the file system where the data directory is located. `OFF` means file names are case-sensitive, `ON` means they are not case-sensitive. This variable is read only because it reflects a file system attribute and setting it would have no effect on the file system.

[lower_case_table_names](#)

Command-Line Format	<code>--lower-case-table-names [=#]</code>
System Variable	lower_case_table_names
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value (macOS)	2
Default Value (Unix)	0
Default Value (Windows)	1
Minimum Value	0
Maximum Value	2

If set to 0, table names are stored as specified and comparisons are case-sensitive. If set to 1, table names are stored in lowercase on disk and comparisons are not case-sensitive. If set to 2, table names are stored as given but compared in lowercase. This option also applies to database names and table aliases. For additional details, see [Section 11.2.3, “Identifier Case Sensitivity”](#).

The default value of this variable is platform-dependent (see [lower_case_file_system](#)). On Linux and other Unix-like systems, the default is 0. On Windows the default value is 1. On macOS, the default value is 2. On Linux (and other Unix-like systems), setting the value to 2 is not supported; the server forces the value to 0 instead.

You should *not* set [lower_case_table_names](#) to 0 if you are running MySQL on a system where the data directory resides on a case-insensitive file system (such as on Windows or macOS). It is an unsupported combination that could result in a hang condition when running an `INSERT INTO ... SELECT ... FROM tbl_name` operation with the wrong `tbl_name` lettercase. With `MyISAM`, accessing table names using different lettercases could cause index corruption.

An error message is printed and the server exits if you attempt to start the server with `--lower_case_table_names=0` on a case-insensitive file system.

The setting of this variable affects the behavior of replication filtering options with regard to case sensitivity. For more information, see [Section 19.2.5, “How Servers Evaluate Replication Filtering Rules”](#).

It is prohibited to start the server with a [lower_case_table_names](#) setting that is different from the setting used when the server was initialized. The restriction is necessary because collations used by various data dictionary table fields are determined by the setting defined when the server is initialized, and restarting the server with a different setting would introduce inconsistencies with respect to how identifiers are ordered and compared.

It is therefore necessary to configure [lower_case_table_names](#) to the desired setting before initializing the server. In most cases, this requires configuring [lower_case_table_names](#) in a MySQL option file before starting the MySQL server for the first time. For APT installations on Debian and Ubuntu, however, the server is initialized for you, and there is no opportunity to configure the setting in an option file beforehand. You must therefore use the `debconf-set-selections` utility prior to installing MySQL using APT to enable [lower_case_table_names](#). To do so, run this command before installing MySQL using APT:

```
$> sudo debconf-set-selections <<< "mysql-server mysql-server/lowercase-table-names select Enabled"
```

[mandatory_roles](#)

Command-Line Format	<code>--mandatory-roles=value</code>
System Variable	mandatory_roles
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	empty string

Roles the server should treat as mandatory. In effect, these roles are automatically granted to every user, although setting [mandatory_roles](#) does not actually change any user accounts, and the granted roles are not visible in the `mysql.role_edges` system table.

The variable value is a comma-separated list of role names. Example:

```
SET PERSIST mandatory_roles = '`role1`@`%`,`role2`,role3,role4@localhost`';
```

Setting the runtime value of [mandatory_roles](#) requires the [ROLE_ADMIN](#) privilege, in addition to the [SYSTEM_VARIABLES_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege) normally required to set a global system variable runtime value.

Role names consist of a user part and host part in `user_name@host_name` format. The host part, if omitted, defaults to `%`. For additional information, see [Section 8.2.5, “Specifying Role Names”](#).

The [mandatory_roles](#) value is a string, so user names and host names, if quoted, must be written in a fashion permitted for quoting within quoted strings.

Roles named in the value of [mandatory_roles](#) cannot be revoked with [REVOKE](#) or dropped with [DROP ROLE](#) or [DROP USER](#).

To prevent sessions from being made system sessions by default, a role that has the [SYSTEM_USER](#) privilege cannot be listed in the value of the [mandatory_roles](#) system variable:

If [mandatory_roles](#) is assigned a role at startup that has the [SYSTEM_USER](#) privilege, the server writes a message to the error log and exits.

If [mandatory_roles](#) is assigned a role at runtime that has the [SYSTEM_USER](#) privilege, an error occurs and the [mandatory_roles](#) value remains unchanged.

Mandatory roles, like explicitly granted roles, do not take effect until activated (see [Activating Roles](#)). At login time, role activation occurs for all granted roles if the [activate_all_roles_on_login](#) system variable is enabled; otherwise, or for roles that are set as default roles otherwise. At runtime, [SET ROLE](#) activates roles.

Roles that do not exist when assigned to [mandatory_roles](#) but are created later may require special treatment to be considered mandatory. For details, see [Defining Mandatory Roles](#).

[SHOW GRANTS](#) displays mandatory roles according to the rules described in [Section 15.7.7.22, "SHOW GRANTS Statement"](#).

[max_allowed_packet](#)

Command-Line Format	--max-allowed-packet=#
System Variable	max_allowed_packet
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	67108864
Minimum Value	1024
Maximum Value	1073741824
Unit	bytes
Block Size	1024

The maximum size of one packet or any generated/intermediate string, or any parameter sent by the [mysql_stmt_send_long_data\(\)](#) C API function. The default is 64MB.

The packet message buffer is initialized to [net_buffer_length](#) bytes, but can grow up to [max_allowed_packet](#) bytes when needed. This value by default is small, to catch large (possibly incorrect) packets.

You must increase this value if you are using large [BLOB](#) columns or long strings. It should be as big as the largest [BLOB](#) you want to use. The protocol limit for [max_allowed_packet](#) is 1GB. The value should be a multiple of 1024; nonmultiples are rounded down to the nearest multiple.

When you change the message buffer size by changing the value of the [max_allowed_packet](#) variable, you should also change the buffer size on the client side if your client program permits it. The default [max_allowed_packet](#) value built in to the client library is 1GB, but individual client programs might override this. For example, [mysql](#) and [mysqldump](#) have defaults of 16MB and 24MB, respectively. They also enable you to change the client-side value by setting [max_allowed_packet](#) on the command line or in an option file.

The session value of this variable is read only. The client can receive up to as many bytes as the session value. However, the server does not send to the client more bytes than the current global [max_allowed_packet](#) value. (The global value could be less than the session value if the global value is changed after the client connects.)

[max_connect_errors](#)

Command-Line Format	--max-connect-errors=#
System Variable	max_connect_errors
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	100
Minimum Value	1
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

After [max_connect_errors](#) successive connection requests from a host are interrupted without a successful connection, the server blocks that host from further connections. If a connection from a host is established successfully within fewer than [max_connect_errors](#) attempts after a previous connection was interrupted, the error count for the host is cleared to zero. To unblock blocked hosts, flush the host cache; see [Flushing the Host Cache](#).

[max_connections](#)

Command-Line Format	--max-connections=#
System Variable	max_connections
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	151
Minimum Value	1
Maximum Value	100000

The maximum permitted number of simultaneous client connections. The maximum effective value is the lesser of the effective value of [open_files_limit](#) - 810, and the value actually set for `max_connections`.

For more information, see [Section 7.1.12.1, “Connection Interfaces”](#).

[max_delayed_threads](#)

Command-Line Format	--max-delayed-threads=#
Deprecated	Yes
System Variable	max_delayed_threads
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	20
Minimum Value	0
Maximum Value	16384

This system variable is deprecated (because `DELAYED` inserts are not supported) and subject to removal in a future MySQL release.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[max_digest_length](#)

Command-Line Format	--max-digest-length=#
System Variable	max_digest_length
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	1024
Minimum Value	0
Maximum Value	1048576
Unit	bytes

The maximum number of bytes of memory reserved per session for computation of normalized statement digests. Once that amount of space is used during digest computation, truncation occurs: no further tokens from a parsed statement are collected or figure into its digest value. Statements that differ only after that many bytes of parsed tokens produce the same normalized statement digest and are considered identical if compared or if aggregated for digest statistics.

Warning

Setting [max_digest_length](#) to zero disables digest production, which also disables server functionality that requires digests, such as MySQL Enterprise Firewall.

Decreasing the [max_digest_length](#) value reduces memory use but causes the digest value of more statements to become indistinguishable if they differ only at the end. Increasing the value permits longer statements to be distinguished but increases memory use, particularly for workloads that involve large numbers of simultaneous sessions (the server allocates [max_digest_length](#) bytes per session).

The parser uses this system variable as a limit on the maximum length of normalized statement digests that it computes. The Performance Schema, if it tracks statement digests, makes a copy of the digest value, using the [performance_schema_max_digest_length](#) system variable as a limit on the maximum length of digests that it stores. Consequently, if [performance_schema_max_digest_length](#) is less than [max_digest_length](#), digest values stored in the Performance Schema are truncated relative to the original digest values.

For more information about statement digesting, see [Section 29.10, “Performance Schema Statement Digests and Sampling”](#).

[max_error_count](#)

Command-Line Format	--max-error-count=#
System Variable	max_error_count
Scope	Global, Session

Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	1024
Minimum Value	0
Maximum Value	65535

The maximum number of error, warning, and information messages to be stored for display by the [SHOW ERRORS](#) and [SHOW WARNINGS](#) statements. This is the same as the number of condition areas in the diagnostics area, and thus the number of conditions that can be inspected by [GET DIAGNOSTICS](#).

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[max_execution_time](#)

Command-Line Format	--max-execution-time=#
System Variable	max_execution_time
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	4294967295
Unit	milliseconds

The execution timeout for [SELECT](#) statements, in milliseconds. If the value is 0, timeouts are not enabled.

[max_execution_time](#) applies as follows:

The global [max_execution_time](#) value provides the default for the session value for new connections. The session value applies to [SELECT](#) executions executed within the session that include no [MAX_EXECUTION_TIME\(N\)](#) optimizer hint or for which *N* is 0.

[max_execution_time](#) applies to read-only [SELECT](#) statements. Statements that are not read only are those that invoke a stored function that modifies data as a side effect.

[max_execution_time](#) is ignored for [SELECT](#) statements in stored programs.

[max_heap_table_size](#)

Command-Line Format	--max-heap-table-size=#
System Variable	max_heap_table_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	16777216
Minimum Value	16384
Maximum Value (64-bit platforms)	18446744073709550592
Maximum Value (32-bit platforms)	4294966272
Unit	bytes
Block Size	1024

This variable sets the maximum size to which user-created [MEMORY](#) tables are permitted to grow. The value of the variable is used to calculate [MEMORY](#) table [MAX_ROWS](#) values.

Setting this variable has no effect on any existing [MEMORY](#) table, unless the table is re-created with a statement such as [CREATE TABLE](#) or altered with [ALTER TABLE](#) or [TRUNCATE TABLE](#). A server restart also sets the maximum size of existing [MEMORY](#) tables to the global [max_heap_table_size](#) value.

This variable is also used in conjunction with [tmp_table_size](#) to limit the size of internal in-memory tables. See [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

[max_heap_table_size](#) is not replicated. See [Section 19.5.1.21, “Replication and MEMORY Tables”](#), and [Section 19.5.1.39, “Replication and Variables”](#), for more information.

[max_insert_delayed_threads](#)

Deprecated	Yes
System Variable	max_insert_delayed_threads
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No

Type	Integer
Default Value	0
Minimum Value	20
Maximum Value	16384

This variable is a synonym for [max_delayed_threads](#). Like [max_delayed_threads](#), it is deprecated (because DELAYED inserts are not supported) and subject to removal in a future MySQL release.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[max_join_size](#)

Command-Line Format	--max-join-size=#
System Variable	max_join_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	18446744073709551615
Minimum Value	1
Maximum Value	18446744073709551615

This represents a limit on the maximum number of row accesses in base tables made by a join. If the server's estimate indicates that a greater number of rows than `max_join_size` must be read from the base tables, the statement is rejected with an error.

Setting this variable to a value other than DEFAULT resets the value of [sql_big_selects](#) to 0. If you set the `sql_big_selects` value again, the `max_join_size` variable is ignored.

[max_length_for_sort_data](#)

Command-Line Format	--max-length-for-sort-data=#
Deprecated	Yes
System Variable	max_length_for_sort_data
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	4096
Minimum Value	4
Maximum Value	8388608
Unit	bytes

This variable is deprecated, and has no effect in MySQL 8.4.

[max_points_in_geometry](#)

Command-Line Format	--max-points-in-geometry=#
System Variable	max_points_in_geometry
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	65536
Minimum Value	3
Maximum Value	1048576

The maximum value of the `points_per_circle` argument to the [ST_Buffer_Strategy\(\)](#) function.

[max_prepared_stmt_count](#)

Command-Line Format	--max-prepared-stmt-count=#
System Variable	max_prepared_stmt_count
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	16382
Minimum Value	0
Maximum Value	4194304

This variable limits the total number of prepared statements in the server. It can be used in environments where there is the potential for denial-of-service attacks based on running the server out of memory by preparing huge numbers of statements. If the value is set lower than the current number of prepared statements, existing statements are not affected and can be used, but no new statements can be prepared until the current number drops below the limit. Setting the value to 0 disables prepared statements.

[max_seeks_for_key](#)

Command-Line Format	--max-seeks-for-key=#
System Variable	max_seeks_for_key
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value (Windows)	4294967295
Default Value (Other, 64-bit platforms)	18446744073709551615
Default Value (Other, 32-bit platforms)	4294967295
Minimum Value	1
Maximum Value (Windows)	4294967295
Maximum Value (Other, 64-bit platforms)	18446744073709551615
Maximum Value (Other, 32-bit platforms)	4294967295

Limit the assumed maximum number of seeks when looking up rows based on a key. The MySQL optimizer assumes that no more than this number of key seeks are required when searching for matching rows in a table by scanning an index, regardless of the actual cardinality of the index (see [Section 15.7.7.23, “SHOW INDEX Statement”](#)). By setting this to a low value (say, 100), you can force MySQL to prefer indexes instead of table scans.

[max_sort_length](#)

Command-Line Format	--max-sort-length=#
System Variable	max_sort_length
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	1024
Minimum Value	4
Maximum Value	8388608
Unit	bytes

The number of bytes to use when sorting string values which use PAD SPACE collations. The server uses only the first [max_sort_length](#) bytes of any such value and ignores the rest. Consequently, such values that differ only after the first [max_sort_length](#) bytes compare as equal for GROUP BY, ORDER BY, and DISTINCT operations. (This behavior differs from previous versions of MySQL, where this setting was applied to all values used in comparisons.)

Increasing the value of [max_sort_length](#) may require increasing the value of [sort_buffer_size](#) as well. For details, see [Section 10.2.1.16, “ORDER BY Optimization”](#).

[max_sp_recursion_depth](#)

Command-Line Format	--max-sp-recursion-depth[=#]
System Variable	max_sp_recursion_depth
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	255

The number of times that any given stored procedure may be called recursively. The default value for this option is 0, which completely disables recursion in stored procedures. The maximum value is 255.

Stored procedure recursion increases the demand on thread stack space. If you increase the value of [max_sp_recursion_depth](#), it may be necessary to increase thread stack size by increasing the value of [thread_stack](#) at server startup.

[max_user_connections](#)

Command-Line Format	--max-user-connections=#
System Variable	max_user_connections
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0

Minimum Value	0
Maximum Value	4294967295

The maximum number of simultaneous connections permitted to any given MySQL user account. A value of 0 (the default) means “no limit.”

This variable has a global value that can be set at server startup or runtime. It also has a read-only session value that indicates the effective simultaneous-connection limit that applies to the account associated with the current session. The session value is initialized as follows:

If the user account has a nonzero `MAX_USER_CONNECTIONS` resource limit, the session [max_user_connections](#) value is set to that limit.

Otherwise, the session [max_user_connections](#) value is set to the global value.

Account resource limits are specified using the [CREATE USER](#) or [ALTER USER](#) statement. See [Section 8.2.21, “Setting Account Resource Limits”](#).

[max_write_lock_count](#)

Command-Line Format	<code>--max-write-lock-count=#</code>
System Variable	max_write_lock_count
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value (Windows)	4294967295
Default Value (Other, 64-bit platforms)	18446744073709551615
Default Value (Other, 32-bit platforms)	4294967295
Minimum Value	1
Maximum Value (Windows)	4294967295
Maximum Value (Other, 64-bit platforms)	18446744073709551615
Maximum Value (Other, 32-bit platforms)	4294967295

After this many write locks, permit some pending read lock requests to be processed in between. Write lock requests have higher priority than read lock requests. However, if [max_write_lock_count](#) is set to some low value (say, 10), read lock requests may be preferred over pending write lock requests if the read lock requests have already been passed over in favor of 10 write lock requests. Normally this behavior does not occur because [max_write_lock_count](#) by default has a very large value.

[mecab_rc_file](#)

Command-Line Format	<code>--mecab-rc-file=file_name</code>
System Variable	mecab_rc_file
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

The `mecab_rc_file` option is used when setting up the MeCab full-text parser.

The `mecab_rc_file` option defines the path to the `mecabrc` configuration file, which is the configuration file for MeCab. The option is read-only and can only be set at startup. The `mecabrc` configuration file is required to initialize MeCab.

For information about the MeCab full-text parser, see [Section 14.9.9, “MeCab Full-Text Parser Plugin”](#).

For information about options that can be specified in the MeCab `mecabrc` configuration file, refer to the [MeCab Documentation](#) on the [Google Developers](#) site.

[min_examined_row_limit](#)

Command-Line Format	<code>--min-examined-row-limit=#</code>
System Variable	min_examined_row_limit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

Queries that examine fewer than this number of rows are not logged to the slow query log.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[myisam_data_pointer_size](#)

Command-Line Format	--myisam-data-pointer-size=#
System Variable	myisam_data_pointer_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	6
Minimum Value	2
Maximum Value	7
Unit	bytes

The default pointer size in bytes, to be used by [CREATE TABLE](#) for MyISAM tables when no MAX_ROWS option is specified. This variable cannot be less than 2 or larger than 7. The default value is 6. See [Section B.3.2.10, “The table is full”](#).

[myisam_max_sort_file_size](#)

Command-Line Format	--myisam-max-sort-file-size=#
System Variable	myisam_max_sort_file_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value (Windows)	2146435072
Default Value (Other, 64-bit platforms)	9223372036853727232
Default Value (Other, 32-bit platforms)	2147483648
Minimum Value	0
Maximum Value (Windows)	2146435072
Maximum Value (Other, 64-bit platforms)	9223372036853727232
Maximum Value (Other, 32-bit platforms)	2147483648
Unit	bytes

The maximum size of the temporary file that MySQL is permitted to use while re-creating a MyISAM index (during [REPAIR TABLE](#), [ALTER TABLE](#), or [LOAD DATA](#)). If the file size would be larger than this value, the index is created using the key cache instead, which is slower. The value is given in bytes.

If MyISAM index files exceed this size and disk space is available, increasing the value may help performance. The space must be available in the file system containing the directory where the original index file is located.

[myisam_mmap_size](#)

Command-Line Format	--myisam-mmap-size=#
System Variable	myisam_mmap_size
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value (64-bit platforms)	18446744073709551615
Default Value (32-bit platforms)	4294967295
Minimum Value	7
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295
Unit	bytes

The maximum amount of memory to use for memory mapping compressed [MyISAM](#) files. If many compressed MyISAM tables are used, the value can be decreased to reduce the likelihood of memory-swapping problems.

[myisam_recover_options](#)

Command-Line Format	--myisam-recover-options[=list]
System Variable	myisam_recover_options
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Enumeration
Default Value	OFF

Valid Values

OFF
 DEFAULT
 BACKUP
 FORCE
 QUICK

Set the MyISAM storage engine recovery mode. The variable value is any combination of the values of OFF, DEFAULT, BACKUP, FORCE, or QUICK. If you specify multiple values, separate them by commas. Specifying the variable with no value at server startup is the same as specifying DEFAULT, and specifying with an explicit value of "" disables recovery (same as a value of OFF). If recovery is enabled, each time [mysqld](#) opens a MyISAM table, it checks whether the table is marked as crashed or was not closed properly. (The last option works only if you are running with external locking disabled.) If this is the case, [mysqld](#) runs a check on the table. If the table was corrupted, [mysqld](#) attempts to repair it.

The following options affect how the repair works.

Option	Description
OFF	No recovery.
DEFAULT	Recovery without backup, forcing, or quick checking.
BACKUP	If the data file was changed during recovery, save a backup of the <i>tbl_name.MYD</i> file as <i>tbl_name-datetime.BAK</i> .
FORCE	Run recovery even if we would lose more than one row from the .MYD file.
QUICK	Do not check the rows in the table if there are not any delete blocks.

Before the server automatically repairs a table, it writes a note about the repair to the error log. If you want to be able to recover from most problems without user intervention, you should use the options BACKUP, FORCE. This forces a repair of a table even if some rows would be deleted, but it keeps the old data file as a backup so that you can later examine what happened.

See [Section 18.2.1, “MyISAM Startup Options”](#).

[mysam_sort_buffer_size](#)

Command-Line Format	--mysam-sort-buffer-size=#
System Variable	mysam_sort_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8388608
Minimum Value	4096
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295
Unit	bytes

The size of the buffer that is allocated when sorting MyISAM indexes during a [REPAIR TABLE](#) or when creating indexes with [CREATE INDEX](#) or [ALTER TABLE](#).

[mysam_stats_method](#)

Command-Line Format	--mysam-stats-method=name
System Variable	mysam_stats_method
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	nulls_unequal
	nulls_unequal
Valid Values	nulls_equal
	nulls_ignored

How the server treats NULL values when collecting statistics about the distribution of index values for MyISAM tables. This variable has three possible values, nulls_equal, nulls_unequal, and nulls_ignored. For nulls_equal, all NULL index values are considered equal and form a single value group that has a size equal to the number of NULL values. For nulls_unequal, NULL values are considered unequal, and each NULL forms a distinct value group of size 1. For nulls_ignored, NULL values are ignored.

The method that is used for generating table statistics influences how the optimizer chooses indexes for query execution, as described in [Section 10.3.8, “InnoDB and MyISAM Index Statistics Collection”](#).

[mysam_use_mmap](#)

Command-Line Format	--myisam-use-mmap[={OFF ON}]
System Variable	myisam_use_mmap
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Use memory mapping for reading and writing MyISAM tables.

[mysql_native_password_proxy_users](#)

Command-Line Format	--mysql-native-password-proxy-users[={OFF ON}]
System Variable	mysql_native_password_proxy_users
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls whether the [mysql_native_password](#) built-in authentication plugin (deprecated) supports proxy users. It has no effect unless the [check_proxy_users](#) system variable and the `mysql_native_password` plugin are enabled. For information about user proxying, see [Section 8.2.19, “Proxy Users”](#).

[named_pipe](#)

Command-Line Format	--named-pipe[={OFF ON}]
System Variable	named_pipe
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Windows
Type	Boolean
Default Value	OFF

(Windows only.) Indicates whether the server supports connections over named pipes.

[named_pipe_full_access_group](#)

Command-Line Format	--named-pipe-full-access-group=value
System Variable	named_pipe_full_access_group
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Windows
Type	String
Default Value	empty string empty string
Valid Values	valid Windows local group name *everyone*

(Windows only.) The access control granted to clients on the named pipe created by the MySQL server is set to the minimum necessary for successful communication when the [named_pipe](#) system variable is enabled to support named-pipe connections. Some MySQL client software can open named pipe connections without any additional configuration; however, other client software may still require full access to open a named pipe connection.

This variable sets the name of a Windows local group whose members are granted sufficient access by the MySQL server to use named-pipe clients. The default value is an empty string, which means that no Windows user is granted full access to the named pipe.

A new Windows local group name (for example, `mysql_access_client_users`) can be created in Windows and then used to replace the default value when access is absolutely necessary. In this case, limit the membership of the group to as few users as possible, removing users from the group when their client software is upgraded. A non-member of the group who attempts to open a connection to MySQL with the affected named-pipe client is denied access until a Windows administrator adds the user to the group. Newly added users must log out and log in again to join the group (required by Windows).

Setting the value to `'*everyone*'` provides a language-independent way of referring to the Everyone group on Windows. The Everyone group is not secure by default.

[net_buffer_length](#)

Command-Line Format	--net-buffer-length=#
System Variable	net_buffer_length
Scope	Global, Session

Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	16384
Minimum Value	1024
Maximum Value	1048576
Unit	bytes
Block Size	1024

Each client thread is associated with a connection buffer and result buffer. Both begin with a size given by [net_buffer_length](#) but are dynamically enlarged up to [max_allowed_packet](#) bytes as needed. The result buffer shrinks to [net_buffer_length](#) after each SQL statement.

This variable should not normally be changed, but if you have very little memory, you can set it to the expected length of statements sent by clients. If statements exceed this length, the connection buffer is automatically enlarged. The maximum value to which [net_buffer_length](#) can be set is 1MB.

The session value of this variable is read only.

[net_read_timeout](#)

Command-Line Format	--net-read-timeout=#
System Variable	net_read_timeout
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	30
Minimum Value	1
Maximum Value	31536000
Unit	seconds

The number of seconds to wait for more data from a connection before aborting the read. When the server is reading from the client, [net_read_timeout](#) is the timeout value controlling when to abort. When the server is writing to the client, [net_write_timeout](#) is the timeout value controlling when to abort. See also [replica_net_timeout](#).

[net_retry_count](#)

Command-Line Format	--net-retry-count=#
System Variable	net_retry_count
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	10
Minimum Value	1
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

If a read or write on a communication port is interrupted, retry this many times before giving up. This value should be set quite high on FreeBSD because internal interrupts are sent to all threads.

[net_write_timeout](#)

Command-Line Format	--net-write-timeout=#
System Variable	net_write_timeout
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	60
Minimum Value	1
Maximum Value	31536000
Unit	seconds

The number of seconds to wait for a block to be written to a connection before aborting the write. See also [net_read_timeout](#).

[ngram_token_size](#)

Command-Line Format	--ngram-token-size=#
System Variable	ngram_token_size
Scope	Global
Dynamic	No
SET VAR Hint Applies	No

Type	Integer
Default Value	2
Minimum Value	1
Maximum Value	10

Defines the n-gram token size for the n-gram full-text parser. The `ngram_token_size` option is read-only and can only be modified at startup. The default value is 2 (bigram). The maximum value is 10.

For more information about how to configure this variable, see [Section 14.9.8, “ngram Full-Text Parser”](#).

[offline_mode](#)

Command-Line Format	<code>--offline-mode[={OFF ON}]</code>
System Variable	offline_mode
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

In offline mode, the MySQL instance disconnects client users unless they have relevant privileges, and does not allow them to initiate new connections. Clients that are refused access receive an [ER_SERVER_OFFLINE_MODE](#) error.

To put a server in offline mode, change the value of the [offline_mode](#) system variable from OFF to ON. To resume normal operations, change [offline_mode](#) from ON to OFF. To control offline mode, an administrator account must have the [SYSTEM_VARIABLES_ADMIN](#) privilege and the [CONNECTION_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege, which covers both these privileges). [CONNECTION_ADMIN](#) is required, to prevent accidental lockout.

Offline mode has these characteristics:

Connected client users who do not have the [CONNECTION_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege) are disconnected on the next request, with an appropriate error. Disconnection includes terminating running statements and releasing locks. Such clients also cannot initiate new connections, and receive an appropriate error.

Connected client users who have the [CONNECTION_ADMIN](#) or [SUPER](#) privilege are not disconnected, and can initiate new connections to manage the server.

If the user that puts a server in offline mode does not have the [SYSTEM_USER](#) privilege, connected client users who have the [SYSTEM_USER](#) privilege are also not disconnected. However, these users cannot initiate new connections to the server while it is in offline mode, unless they have the [CONNECTION_ADMIN](#) or [SUPER](#) privilege as well. It is only their existing connection that cannot be terminated, because the [SYSTEM_USER](#) privilege is required to kill a session or statement that is executing with the [SYSTEM_USER](#) privilege.

Replication threads are permitted to keep applying data to the server.

[old_alter_table](#)

Command-Line Format	<code>--old-alter-table[={OFF ON}]</code>
System Variable	old_alter_table
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

When this variable is enabled, the server does not use the optimized method of processing an [ALTER TABLE](#) operation. It reverts to using a temporary table, copying over the data, and then renaming the temporary table to the original, as used by MySQL 5.0 and earlier. For more information on the operation of [ALTER TABLE](#), see [Section 15.1.9, “ALTER TABLE Statement”](#).

`ALTER TABLE ... DROP PARTITION` with [old_alter_table=ON](#) rebuilds the partitioned table and attempts to move data from the dropped partition to another partition with a compatible `PARTITION ... VALUES` definition. Data that cannot be moved to another partition is deleted. In earlier releases, `ALTER TABLE ... DROP PARTITION` with [old_alter_table=ON](#) deletes data stored in the partition and drops the partition.

[open_files_limit](#)

Command-Line Format	<code>--open-files-limit=#</code>
System Variable	open_files_limit
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	5000, with possible adjustment
Minimum Value	0

Maximum Value

platform dependent

The number of file descriptors available to [mysqld](#) from the operating system:

At startup, [mysqld](#) reserves descriptors with `setrlimit()`, using the value requested at by setting this variable directly or by using the `--open-files-limit` option to [mysqld safe](#). If [mysqld](#) produces the error `Too many open files`, try increasing the [open_files_limit](#) value. Internally, the maximum value for this variable is the maximum unsigned integer value, but the actual maximum is platform dependent.

At runtime, the value of [open_files_limit](#) indicates the number of file descriptors actually permitted to [mysqld](#) by the operating system, which might differ from the value requested at startup. If the number of file descriptors requested during startup cannot be allocated, [mysqld](#) writes a warning to the error log.

The effective [open_files_limit](#) value is based on the value specified at system startup (if any) and the values of [max_connections](#) and [table_open_cache](#), using these formulas:

$10 + \text{max_connections} + (\text{table_open_cache} * 2)$. Using the defaults for these variables yields 8161.

On Windows only, 2048 (the value of the C Run-Time Library file descriptor maximum) is added to this number. This totals 10209, again using the default values for the indicated system variables.

$\text{max_connections} * 5$

The operating system limit.

The server attempts to obtain the number of file descriptors using the maximum of those values, capped to the maximum unsigned integer value. If that many descriptors cannot be obtained, the server attempts to obtain as many as the system permits.

The effective value is 0 on systems where MySQL cannot change the number of open files.

On Unix, the value cannot be set greater than the value displayed by the `ulimit -n` command. On Linux systems using `systemd`, the value cannot be set greater than `LimitNOFILE` (this is `DefaultLimitNOFILE`, if `LimitNOFILE` is not set); otherwise, on Linux, the value of `open_files_limit` cannot exceed `ulimit -n`.

[optimizer_prune_level](#)

Command-Line Format	<code>--optimizer-prune-level=#</code>
System Variable	optimizer_prune_level
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	1
Minimum Value	0
Maximum Value	1

Controls the heuristics applied during query optimization to prune less-promising partial plans from the optimizer search space. A value of 0 disables heuristics so that the optimizer performs an exhaustive search. A value of 1 causes the optimizer to prune plans based on the number of rows retrieved by intermediate plans.

[optimizer_search_depth](#)

Command-Line Format	<code>--optimizer-search-depth=#</code>
System Variable	optimizer_search_depth
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	62
Minimum Value	0
Maximum Value	62

The maximum depth of search performed by the query optimizer. Values larger than the number of relations in a query result in better query plans, but take longer to generate an execution plan for a query. Values smaller than the number of relations in a query return an execution plan quicker, but the resulting plan may be far from being optimal. If set to 0, the system automatically picks a reasonable value.

[optimizer_switch](#)

Command-Line Format	<code>--optimizer-switch=value</code>
System Variable	optimizer_switch
Scope	Global, Session
Dynamic	Yes

[SET VAR](#) Hint Applies

Type

Yes

Set

batched_key_access={on|off}

block_nested_loop={on|off}

condition_fanout_filter={on|off}

derived_condition_pushdown={on|off}

derived_merge={on|off}

duplicateweedout={on|off}

engine_condition_pushdown={on|off}

firstmatch={on|off}

hash_join={on|off}

index_condition_pushdown={on|off}

index_merge={on|off}

index_merge_intersection={on|off}

index_merge_sort_union={on|off}

index_merge_union={on|off}

loosescan={on|off}

materialization={on|off}

mrr={on|off}

mrr_cost_based={on|off}

prefer_ordering_index={on|off}

semi_join={on|off}

skip_scan={on|off}

subquery_materialization_cost_based={on|off}

subquery_to_derived={on|off}

use_index_extensions={on|off}

use_invisible_indexes={on|off}

Valid Values

Valid Values

batched_key_access={on|off}
block_nested_loop={on|off}
condition_fanout_filter={on|off}
derived_merge={on|off}
duplicateweedout={on|off}
engine_condition_pushdown={on|off}
firstmatch={on|off}
hash_join={on|off}
index_condition_pushdown={on|off}
index_merge={on|off}
index_merge_intersection={on|off}
index_merge_sort_union={on|off}
index_merge_union={on|off}
loosescan={on|off}
materialization={on|off}
mrr={on|off}
mrr_cost_based={on|off}
prefer_ordering_index={on|off}
semijoin={on|off}
skip_scan={on|off}
subquery_materialization_cost_based={on|off}
subquery_to_derived={on|off}
use_index_extensions={on|off}
use_invisible_indexes={on|off}

Valid Values

batched_key_access={on|off}
block_nested_loop={on|off}
condition_fanout_filter={on|off}
derived_merge={on|off}
duplicateweedout={on|off}
engine_condition_pushdown={on|off}
firstmatch={on|off}
hash_join={on|off}
index_condition_pushdown={on|off}
index_merge={on|off}
index_merge_intersection={on|off}
index_merge_sort_union={on|off}
index_merge_union={on|off}
loosescan={on|off}
materialization={on|off}
mrr={on|off}
mrr_cost_based={on|off}
semi_join={on|off}
skip_scan={on|off}
subquery_materialization_cost_based={on|off}
use_index_extensions={on|off}
use_invisible_indexes={on|off}

batched_key_access={on|off}
 block_nested_loop={on|off}
 condition_fanout_filter={on|off}
 derived_merge={on|off}
 duplicateweedout={on|off}
 engine_condition_pushdown={on|off}
 firstmatch={on|off}
 index_condition_pushdown={on|off}
 index_merge={on|off}
 index_merge_intersection={on|off}
 index_merge_sort_union={on|off}
 index_merge_union={on|off}
 loosescan={on|off}
 materialization={on|off}
 mrr={on|off}
 mrr_cost_based={on|off}
 semijoin={on|off}
 skip_scan={on|off}
 subquery_materialization_cost_based={on|off}
 use_index_extensions={on|off}
 use_invisible_indexes={on|off}

Valid Values

The [optimizer_switch](#) system variable enables control over optimizer behavior. The value of this variable is a set of flags, each of which has a value of `on` or `off` to indicate whether the corresponding optimizer behavior is enabled or disabled. This variable has global and session values and can be changed at runtime. The global default can be set at server startup.

To see the current set of optimizer flags, select the variable value:

```
mysql> SELECT @@optimizer_switch\G
***** 1. row *****
@@optimizer_switch: index_merge=on,index_merge_union=on,
                    index_merge_sort_union=on,index_merge_intersection=on,
                    engine_condition_pushdown=on,index_condition_pushdown=on,
                    mrr=on,mrr_cost_based=on,block_nested_loop=on,
                    batched_key_access=off,materialization=on,semijoin=on,
                    loosescan=on,firstmatch=on,duplicateweedout=on,
                    subquery_materialization_cost_based=on,
                    use_index_extensions=on,condition_fanout_filter=on,
                    derived_merge=on,use_invisible_indexes=off,skip_scan=on,
                    hash_join=on,subquery_to_derived=off,
                    prefer_ordering_index=on,hypergraph_optimizer=off,
                    derived_condition_pushdown=on,hash_set_operations=on

1 row in set (0.00 sec)
```

For more information about the syntax of this variable and the optimizer behaviors that it controls, see [Section 10.9.2, “Switchable Optimizations”](#).

[optimizer_trace](#)

Command-Line Format	--optimizer-trace=value
System Variable	optimizer_trace
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

This variable controls optimizer tracing. For details, see [MySQL Internals: Tracing the Optimizer](#).

[optimizer_trace_features](#)

Command-Line Format	--optimizer-trace-features=value
System Variable	optimizer_trace_features
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String

This variable enables or disables selected optimizer tracing features. For details, see [MySQL Internals: Tracing the Optimizer](#).

[optimizer_trace_limit](#)

Command-Line Format	--optimizer-trace-limit=#
System Variable	optimizer_trace_limit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1
Minimum Value	0
Maximum Value	2147483647

The maximum number of optimizer traces to display. For details, see [MySQL Internals: Tracing the Optimizer](#).

[optimizer_trace_max_mem_size](#)

Command-Line Format	--optimizer-trace-max-mem-size=#
System Variable	optimizer_trace_max_mem_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	1048576
Minimum Value	0
Maximum Value	4294967295
Unit	bytes

The maximum cumulative size of stored optimizer traces. For details, see [MySQL Internals: Tracing the Optimizer](#).

[optimizer_trace_offset](#)

Command-Line Format	--optimizer-trace-offset=#
System Variable	optimizer_trace_offset
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	-1
Minimum Value	-2147483647
Maximum Value	2147483647

The offset of optimizer traces to display. For details, see [MySQL Internals: Tracing the Optimizer](#).

[performance_schema_xxx](#)

Performance Schema system variables are listed in [Section 29.15, “Performance Schema System Variables”](#). These variables may be used to configure Performance Schema operation.

[parser_max_mem_size](#)

Command-Line Format	--parser-max-mem-size=#
System Variable	parser_max_mem_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value (64-bit platforms)	18446744073709551615
Default Value (32-bit platforms)	4294967295
Minimum Value	10000000
Maximum Value (64-bit platforms)	18446744073709551615
Maximum Value (32-bit platforms)	4294967295

Unit	bytes
------	-------

The maximum amount of memory available to the parser. The default value places no limit on memory available. The value can be reduced to protect against out-of-memory situations caused by parsing long or complex SQL statements.

[partial_revokes](#)

Command-Line Format	--partial-revokes[={OFF ON}]
System Variable	partial_revokes
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF (if partial revokes do not exist)
	ON (if partial revokes exist)

Enabling this variable makes it possible to revoke privileges partially. Specifically, for users who have privileges at the global level, [partial_revokes](#) enables privileges for specific schemas to be revoked while leaving the privileges in place for other schemas. For example, a user who has the global [UPDATE](#) privilege can be restricted from exercising this privilege on the `mysql` system schema. (Or, stated another way, the user is enabled to exercise the [UPDATE](#) privilege on all schemas except the `mysql` schema.) In this sense, the user's global [UPDATE](#) privilege is partially revoked.

Once enabled, [partial_revokes](#) cannot be disabled if any account has privilege restrictions. If any such account exists, disabling [partial_revokes](#) fails:

For attempts to disable [partial_revokes](#) at startup, the server logs an error message and enables [partial_revokes](#).

For attempts to disable [partial_revokes](#) at runtime, an error occurs and the [partial_revokes](#) value remains unchanged.

To disable [partial_revokes](#) in this case, first modify each account that has partially revoked privileges, either by re-granting the privileges or by removing the account.

Note

In privilege assignments, enabling [partial_revokes](#) causes MySQL to interpret occurrences of unescaped `_` and `%` SQL wildcard characters in schema names as literal characters, just as if they had been escaped as `_` and `\%`. Because this changes how MySQL interprets privileges, it may be advisable to avoid unescaped wildcard characters in privilege assignments for installations where [partial_revokes](#) may be enabled.

In addition, use of `_` and `%` as wildcard characters in grants is deprecated, and you should expect support for them to be removed in a future version of MySQL.

For more information, including instructions for removing partial revokes, see [Section 8.2.12, "Privilege Restriction Using Partial Revokes"](#).

[password_history](#)

Command-Line Format	--password-history=#
System Variable	password_history
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	4294967295

This variable defines the global policy for controlling reuse of previous passwords based on required minimum number of password changes. For an account password used previously, this variable indicates the number of subsequent account password changes that must occur before the password can be reused. If the value is 0 (the default), there is no reuse restriction based on number of password changes.

Changes to this variable apply immediately to all accounts defined with the `PASSWORD HISTORY DEFAULT` option.

The global number-of-changes password reuse policy can be overridden as desired for individual accounts using the `PASSWORD HISTORY` option of the [CREATE USER](#) and [ALTER USER](#) statements. See [Section 8.2.15, "Password Management"](#).

[password_require_current](#)

Command-Line Format	--password-require-current[={OFF ON}]
System Variable	password_require_current
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No

Type	Boolean
Default Value	OFF

This variable defines the global policy for controlling whether attempts to change an account password must specify the current password to be replaced.

Changes to this variable apply immediately to all accounts defined with the `PASSWORD REQUIRE CURRENT DEFAULT` option.

The global verification-required policy can be overridden as desired for individual accounts using the `PASSWORD REQUIRE` option of the [CREATE USER](#) and [ALTER USER](#) statements. See [Section 8.2.15, “Password Management”](#).

[password_reuse_interval](#)

Command-Line Format	<code>--password-reuse-interval=#</code>
System Variable	password_reuse_interval
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	4294967295
Unit	days

This variable defines the global policy for controlling reuse of previous passwords based on time elapsed. For an account password used previously, this variable indicates the number of days that must pass before the password can be reused. If the value is 0 (the default), there is no reuse restriction based on time elapsed.

Changes to this variable apply immediately to all accounts defined with the `PASSWORD REUSE INTERVAL DEFAULT` option.

The global time-elapsed password reuse policy can be overridden as desired for individual accounts using the `PASSWORD REUSE INTERVAL` option of the [CREATE USER](#) and [ALTER USER](#) statements. See [Section 8.2.15, “Password Management”](#).

[persisted_globals_load](#)

Command-Line Format	<code>--persisted-globals-load[={OFF ON}]</code>
System Variable	persisted_globals_load
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

Whether to load persisted configuration settings from the `mysqld-auto.cnf` file in the data directory. The server normally processes this file at startup after all other option files (see [Section 6.2.2.2, “Using Option Files”](#)). Disabling [persisted_globals_load](#) causes the server startup sequence to skip `mysqld-auto.cnf`.

To modify the contents of `mysqld-auto.cnf`, use the [SET PERSIST](#), [SET PERSIST ONLY](#), and [RESET PERSIST](#) statements. See [Section 7.1.9.3, “Persisted System Variables”](#).

[persist_only_admin_x509_subject](#)

Command-Line Format	<code>--persist-only-admin-x509-subject=string</code>
System Variable	persist_only_admin_x509_subject
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	empty string

[SET PERSIST](#) and [SET PERSIST ONLY](#) enable system variables to be persisted to the `mysqld-auto.cnf` option file in the data directory (see [Section 15.7.6.1, “SET Syntax for Variable Assignment”](#)). Persisting system variables enables runtime configuration changes that affect subsequent server restarts, which is convenient for remote administration not requiring direct access to MySQL server host option files. However, some system variables are nonpersistible or can be persisted only under certain restrictive conditions.

The [persist_only_admin_x509_subject](#) system variable specifies the SSL certificate X.509 Subject value that users must have to be able to persist system variables that are persist-restricted. The default value is the empty string, which disables the Subject check so that persist-restricted system variables cannot be persisted by any user.

If [persist_only_admin_x509_subject](#) is nonempty, users who connect to the server using an encrypted connection and supply an SSL certificate with the designated Subject value then can use [SET PERSIST ONLY](#) to persist persist-restricted system variables. For information about persist-restricted system variables and instructions for configuring MySQL to enable [persist_only_admin_x509_subject](#), see [Section 7.1.9.4, “Nonpersistible and Persist-Restricted System Variables”](#).

[persist_sensitive_variables_in_plaintext](#)

Command-Line Format	<code>--persist_sensitive_variables_in_plaintext[={OFF ON}]</code>
System Variable	persist_sensitive_variables_in_plaintext
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

`persist_sensitive_variables_in_plaintext` controls whether the server is permitted to store the values of sensitive system variables in an unencrypted format, if keyring component support is not available at the time when [SET PERSIST](#) is used to set the value of the system variable. It also controls whether or not the server can start if the encrypted values cannot be decrypted. Note that keyring plugins do not support secure storage of sensitive system variables; a keyring component (see [Section 8.4.4, “The MySQL Keyring”](#)) must be enabled on the MySQL Server instance to support secure storage.

The default setting, `ON`, encrypts the values if keyring component support is available, and persists them unencrypted (with a warning) if it is not. The next time any persisted system variable is set, if keyring support is available at that time, the server encrypts the values of any unencrypted sensitive system variables. The `ON` setting also allows the server to start if encrypted system variable values cannot be decrypted, in which case a warning is issued and the default values for the system variables are used. In that situation, their values cannot be changed until they can be decrypted.

The most secure setting, `OFF`, means sensitive system variable values cannot be persisted if keyring component support is unavailable. The `OFF` setting also means the server does not start if encrypted system variable values cannot be decrypted.

For more information, see [Persisting Sensitive System Variables](#).

[pid_file](#)

Command-Line Format	<code>--pid-file=file_name</code>
System Variable	pid_file
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

The path name of the file in which the server writes its process ID. The server creates the file in the data directory unless an absolute path name is given to specify a different directory. If you specify this variable, you must specify a value. If you do not specify this variable, MySQL uses a default value of `host_name.pid`, where `host_name` is the name of the host machine.

The process ID file is used by other programs such as [mysqld_safe](#) to determine the server's process ID. On Windows, this variable also affects the default error log file name. See [Section 7.4.2, “The Error Log”](#).

[plugin_dir](#)

Command-Line Format	<code>--plugin-dir=dir_name</code>
System Variable	plugin_dir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name
Default Value	<code>BASEDIR/lib/plugin</code>

The path name of the plugin directory.

If the plugin directory is writable by the server, it may be possible for a user to write executable code to a file in the directory using [SELECT ... INTO DUMPFILE](#). This can be prevented by making [plugin_dir](#) read only to the server or by setting [secure_file_priv](#) to a directory where [SELECT](#) writes can be made safely.

[port](#)

Command-Line Format	<code>--port=port_num</code>
System Variable	port
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	3306
Minimum Value	0
Maximum Value	65535

The number of the port on which the server listens for TCP/IP connections. This variable can be set with the `--port` option.

[preload_buffer_size](#)

Command-Line Format	--preload-buffer-size=#
System Variable	preload_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	32768
Minimum Value	1024
Maximum Value	1073741824
Unit	bytes

The size of the buffer that is allocated when preloading indexes.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[print_identified_with_as_hex](#)

Command-Line Format	--print-identified-with-as-hex[={OFF ON}]
System Variable	print_identified_with_as_hex
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Password hash values displayed in the `IDENTIFIED WITH` clause of output from [SHOW CREATE USER](#) may contain unprintable characters that have adverse effects on terminal displays and in other environments. Enabling [print_identified_with_as_hex](#) causes [SHOW CREATE USER](#) to display such hash values as hexadecimal strings rather than as regular string literals. Hash values that do not contain unprintable characters still display as regular string literals, even with this variable enabled.

[profiling](#)

If set to 0 or OFF (the default), statement profiling is disabled. If set to 1 or ON, statement profiling is enabled and the [SHOW PROFILE](#) and [SHOW PROFILES](#) statements provide access to profiling information. See [Section 15.7.7.32, “SHOW PROFILES Statement”](#).

This variable is deprecated; expect it to be removed in a future MySQL release.

[profiling_history_size](#)

The number of statements for which to maintain profiling information if [profiling](#) is enabled. The default value is 15. The maximum value is 100. Setting the value to 0 effectively disables profiling. See [Section 15.7.7.32, “SHOW PROFILES Statement”](#).

This variable is deprecated; expect it to be removed in a future MySQL release.

[protocol_compression_algorithms](#)

Command-Line Format	--protocol-compression-algorithms=value
System Variable	protocol_compression_algorithms
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Set
Default Value	zlib,zstd,uncompressed
	zlib
Valid Values	zstd
	uncompressed

The compression algorithms that the server permits for incoming connections. These include connections by client programs and by servers participating in source/replica replication or Group Replication. Compression does not apply to connections for FEDERATED tables.

[protocol_compression_algorithms](#) does not control connection compression for X Protocol. See [Section 22.5.5, “Connection Compression with X Plugin”](#) for information on how this operates.

The variable value is a list of one or more comma-separated compression algorithm names, in any order, chosen from the following items (not case-sensitive):

`zlib`: Permit connections that use the `zlib` compression algorithm.

`zstd`: Permit connections that use the `zstd` compression algorithm.

uncompressed: Permit uncompressed connections. If this algorithm name is not included in the [protocol_compression_algorithms](#) value, the server does not permit uncompressed connections. It permits only compressed connections that use whichever other algorithms are specified in the value, and there is no fallback to uncompressed connections.

The default value of `zlib`, `zstd`, `uncompressed` indicates that the server permits all compression algorithms.

For more information, see [Section 6.2.8, “Connection Compression Control”](#).

[protocol_version](#)

System Variable	protocol_version
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	10
Minimum Value	0
Maximum Value	4294967295

The version of the client/server protocol used by the MySQL server.

[proxy_user](#)

System Variable	proxy_user
Scope	Session
Dynamic	No
SET VAR Hint Applies	No
Type	String

If the current client is a proxy for another user, this variable is the proxy user account name. Otherwise, this variable is `NULL`. See [Section 8.2.19, “Proxy Users”](#).

[pseudo_replica_mode](#)

System Variable	pseudo_replica_mode
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean

[pseudo_replica_mode](#) is for internal server use. It assists with the correct handling of transactions that originated on older or newer servers than the server currently processing them. [mysqlbinlog](#) sets the value of [pseudo_replica_mode](#) to true before executing any SQL statements.

Setting the session value of [pseudo_replica_mode](#) is a restricted operation. The session user must have either the [REPLICATION_APPLIER](#) privilege (see [Section 19.3.3, “Replication Privilege Checks”](#)), or privileges sufficient to set restricted session variables (see [Section 7.1.9.1, “System Variable Privileges”](#)). However, note that the variable is not intended for users to set; it is set automatically by the replication infrastructure.

[pseudo_replica_mode](#) has the following effects on the handling of prepared XA transactions, which can be attached to or detached from the handling session (by default, the session that issues [XA START](#)):

If true, and the handling session has executed an internal-use [BINLOG](#) statement, XA transactions are automatically detached from the session as soon as the first part of the transaction up to [XA PREPARE](#) finishes, so they can be committed or rolled back by any session that has the [XA RECOVER ADMIN](#) privilege.

If false, XA transactions remain attached to the handling session as long as that session is alive, during which time no other session can commit the transaction. The prepared transaction is only detached if the session disconnects or the server restarts.

[pseudo_replica_mode](#) has the following effects on the `original_commit_timestamp` replication delay timestamp and the [original_server_version](#) system variable:

If true, transactions that do not explicitly set `original_commit_timestamp` or [original_server_version](#) are assumed to originate on another, unknown server, so the value 0, meaning unknown, is assigned to both the timestamp and the system variable.

If false, transactions that do not explicitly set `original_commit_timestamp` or [original_server_version](#) are assumed to originate on the current server, so the current timestamp and the current server's version are assigned to the timestamp and the system variable.

[pseudo_replica_mode](#) has the following effects on the handling of a statement that sets one or more unsupported (removed or unknown) SQL modes:

If true, the server ignores the unsupported mode and raises a warning.

If false, the server rejects the statement with [ER_UNSUPPORTED_SQL_MODE](#).

[pseudo_slave_mode](#)

Deprecated	Yes
System Variable	pseudo_slave_mode
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean

Deprecated alias for [pseudo_replica_mode](#).

[pseudo_thread_id](#)

System Variable	pseudo_thread_id
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	2147483647
Minimum Value	0
Maximum Value	2147483647

This variable is for internal server use.

Warning

Changing the session value of the [pseudo_thread_id](#) system variable changes the value returned by the [CONNECTION_ID\(\)](#) function.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[query_alloc_block_size](#)

Command-Line Format	--query-alloc-block-size=#
System Variable	query_alloc_block_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8192
Minimum Value	1024
Maximum Value	4294966272
Unit	bytes
Block Size	1024

The allocation size in bytes of memory blocks that are allocated for objects created during statement parsing and execution. If you have problems with memory fragmentation, it might help to increase this parameter.

The block size for the byte number is 1024. A value that is not an exact multiple of the block size is rounded down to the next lower multiple of the block size by MySQL Server before storing the value for the system variable. The parser allows values up to the maximum unsigned integer value for the platform (4294967295 or $2^{32}-1$ for a 32-bit system, 18446744073709551615 or $2^{64}-1$ for a 64-bit system) but the actual maximum is a block size lower.

[query_prealloc_size](#)

Command-Line Format	--query-prealloc-size=#
Deprecated	Yes
System Variable	query_prealloc_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8192
Minimum Value	8192
Maximum Value (64-bit platforms)	18446744073709550592
Maximum Value (32-bit platforms)	4294966272
Unit	bytes
Block Size	1024

[query_prealloc_size](#) is deprecated, and setting it has no effect; you should expect its removal in a future release of MySQL.

[rand_seed1](#)

System Variable	rand_seed1
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	N/A
Minimum Value	0
Maximum Value	4294967295

The [rand_seed1](#) and [rand_seed2](#) variables exist as session variables only, and can be set but not read. The variables—but not their values—are shown in the output of [SHOW VARIABLES](#).

The purpose of these variables is to support replication of the [RAND\(\)](#) function. For statements that invoke [RAND\(\)](#), the source passes two values to the replica, where they are used to seed the random number generator. The replica uses these values to set the session variables [rand_seed1](#) and [rand_seed2](#) so that [RAND\(\)](#) on the replica generates the same value as on the source.

[rand_seed2](#)

See the description for [rand_seed1](#).

[range_alloc_block_size](#)

Command-Line Format	--range-alloc-block-size=#
System Variable	range_alloc_block_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	4096
Minimum Value	4096
Maximum Value (64-bit platforms)	18446744073709550592
Maximum Value	4294966272
Unit	bytes
Block Size	1024

The size in bytes of blocks that are allocated when doing range optimization.

The block size for the byte number is 1024. A value that is not an exact multiple of the block size is rounded down to the next lower multiple of the block size by MySQL Server before storing the value for the system variable. The parser allows values up to the maximum unsigned integer value for the platform (4294967295 or $2^{32}-1$ for a 32-bit system, 18446744073709551615 or $2^{64}-1$ for a 64-bit system) but the actual maximum is a block size lower.

[range_optimizer_max_mem_size](#)

Command-Line Format	--range-optimizer-max-mem-size=#
System Variable	range_optimizer_max_mem_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8388608
Minimum Value	0
Maximum Value	18446744073709551615
Unit	bytes

The limit on memory consumption for the range optimizer. A value of 0 means “no limit.” If an execution plan considered by the optimizer uses the range access method but the optimizer estimates that the amount of memory needed for this method would exceed the limit, it abandons the plan and considers other plans. For more information, see [Limiting Memory Use for Range Optimization](#).

[rbr_exec_mode](#)

System Variable	rbr_exec_mode
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	STRICT
Valid Values	STRICT IDEMPOTENT

For internal use by [mysqlbinlog](#). This variable switches the server between IDEMPOTENT mode and STRICT mode. IDEMPOTENT mode causes suppression of duplicate-key and no-key-found errors in [BINLOG](#) statements generated by [mysqlbinlog](#). This mode is useful when replaying a row-based

binary log on a server that causes conflicts with existing data. [mysqlbinlog](#) sets this mode when you specify the [--idempotent](#) option by writing the following to the output:

```
SET SESSION RBR_EXEC_MODE=IDEMPOTENT;
```

[read_buffer_size](#)

Command-Line Format	--read-buffer-size=#
System Variable	read_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	131072
Minimum Value	8192
Maximum Value	2147479552
Unit	bytes
Block Size	4096

Each thread that does a sequential scan for a MyISAM table allocates a buffer of this size (in bytes) for each table it scans. If you do many sequential scans, you might want to increase this value, which defaults to 131072. The value of this variable should be a multiple of 4KB. If it is set to a value that is not a multiple of 4KB, its value is rounded down to the nearest multiple of 4KB.

This option is also used in the following context for all other storage engines with the exception of [InnoDB](#):

For caching the indexes in a temporary file (not a temporary table), when sorting rows for ORDER BY.

For bulk insert into partitions.

For caching results of nested queries.

[read_buffer_size](#) is also used in one other storage engine-specific way: to determine the memory block size for [MEMORY](#) tables.

[select_into_buffer_size](#) is used for the I/O cache buffer for [SELECT INTO DUMPFILE](#) and SELECT INTO OUTFILE statements. ([read_buffer_size](#) is used for the I/O cache buffer size in all other cases.)

For more information about memory use during different operations, see [Section 10.12.3.1, “How MySQL Uses Memory”](#).

[read_only](#)

Command-Line Format	--read-only[={OFF ON}]
System Variable	read_only
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If the [read_only](#) system variable is enabled, the server permits no client updates except from users who have the [CONNECTION_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege). This variable is disabled by default.

The server also supports a [super_read_only](#) system variable (disabled by default), which has these effects:

If [super_read_only](#) is enabled, the server prohibits client updates, even from users who have the [CONNECTION_ADMIN](#) or [SUPER](#) privilege.

Setting [super_read_only](#) to ON implicitly forces [read_only](#) to ON.

Setting [read_only](#) to OFF implicitly forces [super_read_only](#) to OFF.

When [read_only](#) is enabled and when [super_read_only](#) is enabled, the server still permits these operations:

Updates performed by replication threads, if the server is a replica. In replication setups, it can be useful to enable [read_only](#) on replica servers to ensure that replicas accept updates only from the source server and not from clients.

Writes to the system table `mysql.gtid_executed`, which stores GTIDs for executed transactions that are not present in the current binary log file.

Use of [ANALYZE TABLE](#) or [OPTIMIZE TABLE](#) statements. The purpose of read-only mode is to prevent changes to table structure or contents. Analysis and optimization do not qualify as such changes. This means, for example, that consistency checks on read-only replicas can be performed with [mysqlcheck --all-databases --analyze](#).

Use of [FLUSH STATUS](#) statements, which are always written to the binary log.

Operations on `TEMPORARY` tables.

Inserts into the log tables (`mysql.general_log` and `mysql.slow_log`); see [Section 7.4.1, “Selecting General Query Log and Slow Query Log Output Destinations”](#).

Updates to Performance Schema tables, such as [UPDATE](#) or [TRUNCATE TABLE](#) operations.

Changes to [read_only](#) on a replication source server are not replicated to replica servers. The value can be set on a replica independent of the setting on the source.

The following conditions apply to attempts to enable [read_only](#) (including implicit attempts resulting from enabling [super_read_only](#)):

The attempt fails and an error occurs if you have any explicit locks (acquired with [LOCK TABLES](#)) or have a pending transaction.

The attempt blocks while other clients have any ongoing statement, active `LOCK TABLES WRITE`, or ongoing commit, until the locks are released and the statements and transactions end. While the attempt to enable [read_only](#) is pending, requests by other clients for table locks or to begin transactions also block until [read_only](#) has been set.

The attempt blocks if there are active transactions that hold metadata locks, until those transactions end.

[read_only](#) can be enabled while you hold a global read lock (acquired with [FLUSH TABLES WITH READ LOCK](#)) because that does not involve table locks.

[read_rnd_buffer_size](#)

Command-Line Format	<code>--read-rnd-buffer-size=#</code>
System Variable	read_rnd_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	262144
Minimum Value	1
Maximum Value	2147483647
Unit	bytes

This variable is used for reads from `MyISAM` tables, and, for any storage engine, for Multi-Range Read optimization.

When reading rows from a `MyISAM` table in sorted order following a key-sorting operation, the rows are read through this buffer to avoid disk seeks. See [Section 10.2.1.16, “ORDER BY Optimization”](#). Setting the variable to a large value can improve `ORDER BY` performance by a lot. However, this is a buffer allocated for each client, so you should not set the global variable to a large value. Instead, change the session variable only from within those clients that need to run large queries.

For more information about memory use during different operations, see [Section 10.12.3.1, “How MySQL Uses Memory”](#). For information about Multi-Range Read optimization, see [Section 10.2.1.11, “Multi-Range Read Optimization”](#).

[regexp_stack_limit](#)

Command-Line Format	<code>--regexp-stack-limit=#</code>
System Variable	regexp_stack_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8000000
Minimum Value	0
Maximum Value	2147483647
Unit	bytes

The maximum available memory in bytes for the internal stack used for regular expression matching operations performed by [REGEXP_LIKE\(\)](#) and similar functions (see [Section 14.8.2, “Regular Expressions”](#)).

[regexp_time_limit](#)

Command-Line Format	<code>--regexp-time-limit=#</code>
System Variable	regexp_time_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	32

Minimum Value	0
Maximum Value	2147483647

The time limit for regular expression matching operations performed by [REGEXP_LIKE\(\)](#) and similar functions (see [Section 14.8.2, “Regular Expressions”](#)). This limit is expressed as the maximum permitted number of steps performed by the match engine, and thus affects execution time only indirectly. Typically, it is on the order of milliseconds.

[require_row_format](#)

System Variable	require_row_format
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable is for internal server use by replication and [mysqlbinlog](#). It restricts DML events executed in the session to events encoded in row-based binary logging format only, and temporary tables cannot be created. Queries that do not respect the restrictions fail.

Setting the session value of this system variable to ON requires no privileges. Setting the session value of this system variable to OFF is a restricted operation, and the session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[require_secure_transport](#)

Command-Line Format	--require-secure-transport[={OFF ON}]
System Variable	require_secure_transport
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether client connections to the server are required to use some form of secure transport. When this variable is enabled, the server permits only TCP/IP connections encrypted using TLS/SSL, or connections that use a socket file (on Unix) or shared memory (on Windows). The server rejects nonsecure connection attempts, which fail with an [ER_SECURE_TRANSPORT_REQUIRED](#) error.

This capability supplements per-account SSL requirements, which take precedence. For example, if an account is defined with `REQUIRE SSL`, enabling [require_secure_transport](#) does not make it possible to use the account to connect using a Unix socket file.

It is possible for a server to have no secure transports available. For example, a server on Windows supports no secure transports if started without specifying any SSL certificate or key files and with the [shared_memory](#) system variable disabled. Under these conditions, attempts to enable [require_secure_transport](#) at startup cause the server to write a message to the error log and exit. Attempts to enable the variable at runtime fail with an [ER_NO_SECURE_TRANSPORTS_CONFIGURED](#) error.

See also [Configuring Encrypted Connections as Mandatory](#).

[restrict_fk_on_non_standard_key](#)

Command-Line Format	--restrict-fk-on-non-standard-key
Deprecated	Yes
System Variable	restrict_fk_on_non_standard_key
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This variable, when ON (the default), prevents the use of non-unique keys or partial keys as foreign keys. To allow such keys to be used as foreign keys in the current session, use `SET @@session.restrict_fk_on_non_standard_key=OFF`; to allow them to be used globally, set the global variable or start the server with `--skip-restrict-fk-on-non-standard-key`.

Using non-unique or partial keys as foreign keys in a [CREATE TABLE](#) or [ALTER TABLE](#) statement is deprecated, and you should expect support for it to be removed in a future version of MySQL. When [restrict_fk_on_non_standard_key](#) is ON, attempts to do so are rejected with [ER_FK_NO_INDEX_PARENT](#); when it is OFF, this usage is permitted but still raises [ER_WARN_DEPRECATED_NON_STANDARD_KEY](#) as a warning.

[restrict_fk_on_non_standard_key](#) is deprecated, and subject to removal in a future version of MySQL. Setting it raises a deprecation warning.

Implication for MySQL Replication. When a foreign key is created on a nonstandard key on the primary because [restrict_fk_on_non_standard_key](#) is OFF, the statement succeeds on the replica regardless of any setting on the replica for this variable.

[resultset_metadata](#)

System Variable	resultset_metadata
------------------------	------------------------------------

Scope	Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	FULL
	FULL
Valid Values	NONE

For connections for which metadata transfer is optional, the client sets the [resultset_metadata](#) system variable to control whether the server returns result set metadata. Permitted values are FULL (return all metadata; this is the default) and NONE (return no metadata).

For connections that are not metadata-optional, setting [resultset_metadata](#) to NONE produces an error.

For details about managing result set metadata transfer, see [Optional Result Set Metadata](#).

[secondary_engine_cost_threshold](#)

System Variable	secondary_engine_cost_threshold
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Numeric
Default Value	100000.000000
Minimum Value	0
Maximum Value	DBL_MAX (maximum double value)

The optimizer cost threshold for query offload to a secondary engine.

For use with HeatWave. See [MySQL HeatWave User Guide](#).

[schema_definition_cache](#)

Command-Line Format	--schema-definition-cache=#
System Variable	schema_definition_cache
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	256
Minimum Value	256
Maximum Value	524288

Defines a limit for the number of schema definition objects, both used and unused, that can be kept in the dictionary object cache.

Unused schema definition objects are only kept in the dictionary object cache when the number in use is less than the capacity defined by `schema_definition_cache`.

A setting of 0 means that schema definition objects are only kept in the dictionary object cache while they are in use.

For more information, see [Section 16.4, “Dictionary Object Cache”](#).

[secure_file_priv](#)

Command-Line Format	--secure-file-priv=dir_name
System Variable	secure_file_priv
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value	platform specific empty string
Valid Values	dirname NULL

This variable is used to limit the effect of data import and export operations, such as those performed by the [LOAD DATA](#) and [SELECT ... INTO OUTFILE](#) statements and the [LOAD_FILE\(\)](#) function. These operations are permitted only to users who have the [FILE](#) privilege.

[secure_file_priv](#) may be set as follows:

If empty, the variable has no effect. This is not a secure setting.

If set to the name of a directory, the server limits import and export operations to work only with files in that directory. The directory must exist; the server does not create it.

If set to NULL, the server disables import and export operations.

The default value is platform specific and depends on the value of the [INSTALL_LAYOUT](#) CMake option, as shown in the following table. To specify the default [secure_file_priv](#) value explicitly if you are building from source, use the [INSTALL_SECURE_FILE_PRIVDIR](#) CMake option.

INSTALL_LAYOUT Value	Default secure_file_priv Value
STANDALONE	empty
DEB, RPM, SVR4	/var/lib/mysql-files
Otherwise	mysql-files under the CMAKE_INSTALL_PREFIX value

The server checks the value of [secure_file_priv](#) at startup and writes a warning to the error log if the value is insecure. A non-NULL value is considered insecure if it is empty, or the value is the data directory or a subdirectory of it, or a directory that is accessible by all users. If [secure_file_priv](#) is set to a nonexistent path, the server writes an error message to the error log and exits.

[select_into_buffer_size](#)

Command-Line Format	--select-into-buffer-size=#
System Variable	select_into_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	131072
Minimum Value	8192
Maximum Value	2147479552
Unit	bytes
Block Size	4096

When using [SELECT INTO OUTFILE](#) or `SELECT INTO DUMPFILE` to dump data into one or more files for backup creation, data migration, or other purposes, writes can often be buffered and then trigger a large burst of write I/O activity to the disk or other storage device and stall other queries that are more sensitive to latency. You can use this variable to control the size of the buffer used to write data to the storage device to determine when buffer synchronization should occur, and thus to prevent write stalls of the kind just described from occurring.

`select_into_buffer_size` overrides any value set for [read_buffer_size](#). (`select_into_buffer_size` and `read_buffer_size` have the same default, maximum, and minimum values.) You can also use [select_into_disk_sync_delay](#) to set a timeout to be observed afterwards, each time synchronization takes place.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[select_into_disk_sync](#)

Command-Line Format	--select-into-disk-sync={ON OFF}
System Variable	select_into_disk_sync
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	OFF
Valid Values	OFF
	ON

When set on ON, enables buffer synchronization of writes to an output file by a long-running [SELECT INTO OUTFILE](#) or `SELECT INTO DUMPFILE` statement using [select_into_buffer_size](#).

[select_into_disk_sync_delay](#)

Command-Line Format	--select-into-disk-sync-delay=#
System Variable	select_into_disk_sync_delay
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	31536000
Unit	milliseconds

When buffer synchronization of writes to an output file by a long-running [SELECT INTO OUTFILE](#) or `SELECT INTO DUMPFILE` statement is enabled by [select into disk sync](#), this variable sets an optional delay (in milliseconds) following synchronization. 0 (the default) means no delay.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[session_track_gtids](#)

Command-Line Format	--session-track-gtids=value
System Variable	session_track_gtids
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	OFF
	OFF
Valid Values	OWN_GTID
	ALL_GTIDS

Controls whether the server returns GTIDs to the client, enabling the client to use them to track the server state. Depending on the variable value, at the end of executing each transaction, the server's GTIDs are captured and returned to the client as part of the acknowledgement. The possible values for [session_track_gtids](#) are as follows:

OFF: The server does not return GTIDs to the client. This is the default.

OWN_GTID: The server returns the GTIDs for all transactions that were successfully committed by this client in its current session since the last acknowledgement. Typically, this is the single GTID for the last transaction committed, but if a single client request resulted in multiple transactions, the server returns a GTID set containing all the relevant GTIDs.

ALL_GTIDS: The server returns the global value of its [gtid_executed](#) system variable, which it reads at a point after the transaction is successfully committed. As well as the GTID for the transaction just committed, this GTID set includes all transactions committed on the server by any client, and can include transactions committed after the point when the transaction currently being acknowledged was committed.

[session_track_gtids](#) cannot be set within transactional context.

For more information about session state tracking, see [Section 7.1.18, “Server Tracking of Client Session State”](#).

[session_track_schema](#)

Command-Line Format	--session-track-schema[={OFF ON}]
System Variable	session_track_schema
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

Controls whether the server tracks when the default schema (database) is set within the current session and notifies the client to make the schema name available.

If the schema name tracker is enabled, name notification occurs each time the default schema is set, even if the new schema name is the same as the old.

For more information about session state tracking, see [Section 7.1.18, “Server Tracking of Client Session State”](#).

[session_track_state_change](#)

Command-Line Format	--session-track-state-change[={OFF ON}]
System Variable	session_track_state_change
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Controls whether the server tracks changes to the state of the current session and notifies the client when state changes occur. Changes can be reported for these attributes of client session state:

The default schema (database).

Session-specific values for system variables.

- User-defined variables.
- Temporary tables.
- Prepared statements.

If the session state tracker is enabled, notification occurs for each change that involves tracked session attributes, even if the new attribute values are the same as the old. For example, setting a user-defined variable to its current value results in a notification.

The [session_track_state_change](#) variable controls only notification of when changes occur, not what the changes are. For example, state-change notifications occur when the default schema is set or tracked session system variables are assigned, but the notification does not include the schema name or variable values. To receive notification of the schema name or session system variable values, use the [session_track_schema](#) or [session_track_system_variables](#) system variable, respectively.

Note

Assigning a value to [session_track_state_change](#) itself is not considered a state change and is not reported as such. However, if its name listed in the value of [session_track_system_variables](#), any assignments to it do result in notification of the new value.

For more information about session state tracking, see [Section 7.1.18, “Server Tracking of Client Session State”](#).

[session_track_system_variables](#)

Command-Line Format	--session-track-system-variables=#
System Variable	session_track_system_variables
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	time_zone, autocommit, character_set_client, character_set_results, character_set_connection

Controls whether the server tracks assignments to session system variables and notifies the client of the name and value of each assigned variable. The variable value is a comma-separated list of variables for which to track assignments. By default, notification is enabled for [time_zone](#), [autocommit](#), [character_set_client](#), [character_set_results](#), and [character_set_connection](#). (The latter three variables are those affected by [SET NAMES](#).)

To enable display of the Statement ID for each statement processed, use the `statement_id` variable. For example:

```
mysql> SET @@SESSION.session_track_system_variables='statement_id'
mysql> SELECT 1;
+----+
| 1 |
+----+
| 1 |
+----+
1 row in set (0.0006 sec)
Statement ID: 603835
```

The special value * (asterisk) causes the server to track assignments to all session variables. If given, this value must be specified by itself without specific system variable names. This value also enables display of the Statement ID for each successful statement processed.

To disable notification of session variable assignments, set [session_track_system_variables](#) to the empty string.

If session system variable tracking is enabled, notification occurs for all assignments to tracked session variables, even if the new values are the same as the old.

For more information about session state tracking, see [Section 7.1.18, “Server Tracking of Client Session State”](#).

[session_track_transaction_info](#)

Command-Line Format	--session-track-transaction-info=value
System Variable	session_track_transaction_info
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	OFF
	OFF
Valid Values	STATE
	CHARACTERISTICS

Controls whether the server tracks the state and characteristics of transactions within the current session and notifies the client to make this information available. These [session_track_transaction_info](#) values are permitted:

OFF: Disable transaction state tracking. This is the default.

STATE: Enable transaction state tracking without characteristics tracking. State tracking enables the client to determine whether a transaction is in progress and whether it could be moved to a different session without being rolled back.

CHARACTERISTICS: Enable transaction state tracking, including characteristics tracking. Characteristics tracking enables the client to determine how to restart a transaction in another session so that it has the same characteristics as in the original session. The following characteristics are relevant for this purpose:

ISOLATION LEVEL
 READ ONLY
 READ WRITE
 WITH CONSISTENT SNAPSHOT

For a client to safely relocate a transaction to another session, it must track not only transaction state but also transaction characteristics. In addition, the client must track the [transaction_isolation](#) and [transaction_read_only](#) system variables to correctly determine the session defaults. (To track these variables, list them in the value of the [session_track_system_variables](#) system variable.)

For more information about session state tracking, see [Section 7.1.18, “Server Tracking of Client Session State”](#).

[set_operations_buffer_size](#)

Command-Line Format	--set-operations-buffer-size=#
System Variable	set_operations_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	256K
Minimum Value	16K
Maximum Value	1 GB

Sets the buffer size for [INTERSECT](#) and [EXCEPT](#) operations that use hash tables when the [hash_set_operations](#) optimizer switch is ON. In general, increasing the size of this buffer improves performance of these operations when the hashing optimization is enabled.

[sha256_password_auto_generate_rsa_keys](#)

Command-Line Format	--sha256-password-auto-generate-rsa-keys [= {OFF ON}]
Deprecated	Yes
System Variable	sha256_password_auto_generate_rsa_keys
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

The server uses this variable to determine whether to autogenerate RSA private/public key-pair files in the data directory if they do not already exist.

At startup, the server automatically generates RSA private/public key-pair files in the data directory if all of these conditions are true: The [sha256_password_auto_generate_rsa_keys](#) or [caching_sha2_password_auto_generate_rsa_keys](#) system variable is enabled; no RSA options are specified; the RSA files are missing from the data directory. These key-pair files enable secure password exchange using RSA over unencrypted connections for accounts authenticated by the [sha256_password](#) (deprecated) or [caching_sha2_password](#) plugin; see [Section 8.4.1.3, “SHA-256 Pluggable Authentication”](#), and [Section 8.4.1.2, “Caching SHA-2 Pluggable Authentication”](#).

For more information about RSA file autogeneration, including file names and characteristics, see [Section 8.3.3.1, “Creating SSL and RSA Certificates and Keys using MySQL”](#).

The [auto_generate_certs](#) system variable is related but controls autogeneration of SSL certificate and key files needed for secure connections using SSL.

[sha256_password_private_key_path](#)

Command-Line Format	--sha256-password-private-key-path=file_name
Deprecated	Yes
System Variable	sha256_password_private_key_path
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name

Default Value	private_key.pem
----------------------	-----------------

The value of this variable is the path name of the RSA private key file for the sha256_password (deprecated) authentication plugin. If the file is named as a relative path, it is interpreted relative to the server data directory. The file must be in PEM format.

Important

Because this file stores a private key, its access mode should be restricted so that only the MySQL server can read it.

For information about sha256_password, see [Section 8.4.1.3, “SHA-256 Pluggable Authentication”](#).

[sha256_password_proxy_users](#)

Command-Line Format	--sha256-password-proxy-users[={OFF ON}]
Deprecated	Yes
System Variable	sha256_password_proxy_users
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls whether the sha256_password (deprecated) built-in authentication plugin supports proxy users. It has no effect unless the [check_proxy_users](#) system variable is enabled. For information about user proxying, see [Section 8.2.19, “Proxy Users”](#).

[sha256_password_public_key_path](#)

Command-Line Format	--sha256-password-public-key-path=file_name
Deprecated	Yes
System Variable	sha256_password_public_key_path
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	File name
Default Value	public_key.pem

The value of this variable is the path name of the RSA public key file for the sha256_password (deprecated) authentication plugin. If the file is named as a relative path, it is interpreted relative to the server data directory. The file must be in PEM format. Because this file stores a public key, copies can be freely distributed to client users. (Clients that explicitly specify a public key when connecting to the server using RSA password encryption must use the same public key as that used by the server.)

For information about sha256_password (deprecated), including information about how clients specify the RSA public key, see [Section 8.4.1.3, “SHA-256 Pluggable Authentication”](#).

[shared_memory](#)

Command-Line Format	--shared-memory[={OFF ON}]
System Variable	shared_memory
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Windows
Type	Boolean
Default Value	OFF

(Windows only.) Whether the server permits shared-memory connections.

[shared_memory_base_name](#)

Command-Line Format	--shared-memory-base-name=name
System Variable	shared_memory_base_name
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Platform Specific	Windows
Type	String
Default Value	MYSQL

(Windows only.) The name of shared memory to use for shared-memory connections. This is useful when running multiple MySQL instances on a single physical machine. The default name is MYSQL. The name is case-sensitive.

This variable applies only if the server is started with the [shared_memory](#) system variable enabled to support shared-memory connections.

[show create table skip secondary engine](#)

Command-Line Format	<code>--show-create-table-skip-secondary-engine[={OFF ON}]</code>
System Variable	show create table skip secondary engine
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	OFF

Enabling [show create table skip secondary engine](#) causes the `SECONDARY ENGINE` clause to be excluded from [SHOW CREATE TABLE](#) output, and from [CREATE TABLE](#) statements dumped by the [mysqldump](#) utility.

[mysqldump](#) provides the `--show-create-skip-secondary-engine` option. When specified, it enables the [show create table skip secondary engine](#) system variable for the duration of the dump operation.

For use with HeatWave. See [MySQL HeatWave User Guide](#).

[show create table verbosity](#)

Command-Line Format	<code>--show-create-table-verbosity[={OFF ON}]</code>
System Variable	show create table verbosity
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

[SHOW CREATE TABLE](#) normally does not show the `ROW_FORMAT` table option if the row format is the default format. Enabling this variable causes [SHOW CREATE TABLE](#) to display `ROW_FORMAT` regardless of whether it is the default format.

[show gipk in create table and information schema](#)

Command-Line Format	<code>--show-gipk-in-create-table-and-information-schema[={OFF ON}]</code>
System Variable	show gipk in create table and information schema
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

Whether generated invisible primary keys are visible in the output of [SHOW](#) statements and in Information Schema tables. When this variable is set to `OFF`, such keys are not shown.

This variable is not replicated.

For more information, see [Section 15.1.20.11, “Generated Invisible Primary Keys”](#).

[skip external locking](#)

Command-Line Format	<code>--skip-external-locking[={OFF ON}]</code>
System Variable	skip external locking
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

This is `OFF` if [mysqld](#) uses external locking (system locking), `ON` if external locking is disabled. This affects only [MyISAM](#) table access.

This variable is set by the `--external-locking` or `--skip-external-locking` option. External locking is disabled by default.

External locking affects only [MyISAM](#) table access. For more information, including conditions under which it can and cannot be used, see [Section 10.11.5, “External Locking”](#).

[skip name resolve](#)

Command-Line Format	<code>--skip-name-resolve[={OFF ON}]</code>
System Variable	skip name resolve
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean

Default Value

OFF

Whether to resolve host names when checking client connections. If this variable is OFF, [mysqld](#) resolves host names when checking client connections. If it is ON, [mysqld](#) uses only IP numbers; in this case, all `Host` column values in the grant tables must be IP addresses. See [Section 7.1.12.3, “DNS Lookups and the Host Cache”](#).

Depending on the network configuration of your system and the `Host` values for your accounts, clients may need to connect using an explicit `--host` option, such as `--host=127.0.0.1` or `--host:::1`.

An attempt to connect to the host `127.0.0.1` normally resolves to the `localhost` account. However, this fails if the server is run with [skip_name_resolve](#) enabled. If you plan to do that, make sure an account exists that can accept a connection. For example, to be able to connect as `root` using `--host=127.0.0.1` or `--host:::1`, create these accounts:

```
CREATE USER 'root'@'127.0.0.1' IDENTIFIED BY 'root-password';
CREATE USER 'root'@':::1' IDENTIFIED BY 'root-password';
```

[skip_networking](#)

Command-Line Format	<code>--skip-networking[={OFF ON}]</code>
System Variable	skip_networking
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls whether the server permits TCP/IP connections. By default, it is disabled (permit TCP connections). If enabled, the server permits only local (non-TCP/IP) connections and all interaction with [mysqld](#) must be made using named pipes or shared memory (on Windows) or Unix socket files (on Unix). This option is highly recommended for systems where only local clients are permitted. See [Section 7.1.12.3, “DNS Lookups and the Host Cache”](#).

Because starting the server with [--skip-grant-tables](#) disables authentication checks, the server also disables remote connections in that case by enabling [skip_networking](#).

[skip_show_database](#)

Command-Line Format	<code>--skip-show-database</code>
System Variable	skip_show_database
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This prevents people from using the [SHOW DATABASES](#) statement if they do not have the [SHOW DATABASES](#) privilege. This can improve security if you have concerns about users being able to see databases belonging to other users. Its effect depends on the [SHOW DATABASES](#) privilege: If the variable value is ON, the [SHOW DATABASES](#) statement is permitted only to users who have the [SHOW DATABASES](#) privilege, and the statement displays all database names. If the value is OFF, [SHOW DATABASES](#) is permitted to all users, but displays the names of only those databases for which the user has the [SHOW DATABASES](#) or other privilege.

Caution

Because any static global privilege is considered a privilege for all databases, any static global privilege enables a user to see all database names with [SHOW DATABASES](#) or by examining the [SCHEMATA](#) table of `INFORMATION_SCHEMA`, except databases that have been restricted at the database level by partial revokes.

[slow_launch_time](#)

Command-Line Format	<code>--slow-launch-time=#</code>
System Variable	slow_launch_time
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	2
Minimum Value	0
Maximum Value	31536000
Unit	seconds

If creating a thread takes longer than this many seconds, the server increments the [Slow_launch_threads](#) status variable.

[slow_query_log](#)

Command-Line Format	<code>--slow-query-log[={OFF ON}]</code>
----------------------------	--

System Variable	slow_query_log
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether the slow query log is enabled. The value can be 0 (or OFF) to disable the log or 1 (or ON) to enable the log. The destination for log output is controlled by the [log_output](#) system variable; if that value is NONE, no log entries are written even if the log is enabled.

“Slow” is determined by the value of the [long_query_time](#) variable. See [Section 7.4.5, “The Slow Query Log”](#).

[slow_query_log_file](#)

Command-Line Format	<code>--slow-query-log-file=file_name</code>
System Variable	slow_query_log_file
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	host_name-slow.log

The name of the slow query log file. The default value is `host_name-slow.log`, but the initial value can be changed with the `--slow_query_log_file` option.

[socket](#)

Command-Line Format	<code>--socket={file_name pipe_name}</code>
System Variable	socket
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String
Default Value (Windows)	MySQL
Default Value (Other)	/tmp/mysql.sock

On Unix platforms, this variable is the name of the socket file that is used for local client connections. The default is `/tmp/mysql.sock`. (For some distribution formats, the directory might be different, such as `/var/lib/mysql` for RPMs.)

On Windows, this variable is the name of the named pipe that is used for local client connections. The default value is MySQL (not case-sensitive).

[sort_buffer_size](#)

Command-Line Format	<code>--sort-buffer-size=#</code>
System Variable	sort_buffer_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	262144
Minimum Value	32768
Maximum Value (Windows)	4294967295
Maximum Value (Other, 64-bit platforms)	18446744073709551615
Maximum Value (Other, 32-bit platforms)	4294967295
Unit	bytes

Each session that must perform a sort allocates a buffer of this size. [sort_buffer_size](#) is not specific to any storage engine and applies in a general manner for optimization. At minimum the [sort_buffer_size](#) value must be large enough to accommodate fifteen tuples in the sort buffer. Also, increasing the value of [max_sort_length](#) may require increasing the value of [sort_buffer_size](#). For more information, see [Section 10.2.1.16, “ORDER BY Optimization”](#).

If you see many [Sort merge passes](#) per second in `SHOW GLOBAL STATUS` output, you can consider increasing the [sort_buffer_size](#) value to speed up ORDER BY or GROUP BY operations that cannot be improved with query optimization or improved indexing.

The optimizer tries to work out how much space is needed but can allocate more, up to the limit. Setting it larger than required globally slows down most queries that perform sorts. It is best to increase it as a session setting, and only for the sessions that need a larger size. On Linux, there are thresholds of 256KB and 2MB where larger values may significantly slow down memory allocation, so you should consider staying below one of those values. Experiment to find the best value for your workload. See [Section B.3.3.5, “Where MySQL Stores Temporary Files”](#).

The maximum permissible setting for [sort_buffer_size](#) is 4GB–1. Larger values are permitted for 64-bit platforms (except 64-bit Windows, for which large values are truncated to 4GB–1 with a warning).

[sql_auto_is_null](#)

System Variable	sql_auto_is_null
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	OFF

If this variable is enabled, then after a statement that successfully inserts an automatically generated `AUTO_INCREMENT` value, you can find that value by issuing a statement of the following form:

```
SELECT * FROM tbl_name WHERE auto_col IS NULL
```

If the statement returns a row, the value returned is the same as if you invoked the [LAST_INSERT_ID\(\)](#) function. For details, including the return value after a multiple-row insert, see [Section 14.15, “Information Functions”](#). If no `AUTO_INCREMENT` value was successfully inserted, the [SELECT](#) statement returns no row.

The behavior of retrieving an `AUTO_INCREMENT` value by using an [IS NULL](#) comparison is used by some ODBC programs, such as Access. See [Obtaining Auto-Increment Values](#). This behavior can be disabled by setting [sql_auto_is_null](#) to `OFF`.

The default value of [sql_auto_is_null](#) is `OFF`.

[sql_big_selects](#)

System Variable	sql_big_selects
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	ON

If set to `OFF`, MySQL aborts [SELECT](#) statements that are likely to take a very long time to execute (that is, statements for which the optimizer estimates that the number of examined rows exceeds the value of [max_join_size](#)). This is useful when an inadvisable `WHERE` statement has been issued. The default value for a new connection is `ON`, which permits all [SELECT](#) statements.

If you set the [max_join_size](#) system variable to a value other than `DEFAULT`, [sql_big_selects](#) is set to `OFF`.

[sql_buffer_result](#)

System Variable	sql_buffer_result
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	OFF

If enabled, [sql_buffer_result](#) forces results from [SELECT](#) statements to be put into temporary tables. This helps MySQL free the table locks early and can be beneficial in cases where it takes a long time to send results to the client. The default value is `OFF`.

[sql_generate_invisible_primary_key](#)

Command-Line Format	--sql-generate-invisible-primary-key[={OFF ON}]
System Variable	sql_generate_invisible_primary_key
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Whether this server adds a generated invisible primary key to any [InnoDB](#) table that is created without one.

This variable is not replicated. In addition, even if set on the replica, it is ignored by replication applier threads; this means that, by default, a replica does not generate a primary key for any replicated table which, on the source, was created without one. You can cause the replica to generate invisible primary keys for such tables by setting `REQUIRE_TABLE_PRIMARY_KEY_CHECK = GENERATE` as part of a [CHANGE REPLICATION SOURCE TO](#) statement, optionally specifying a replication channel.

For more information and examples, see [Section 15.1.20.11, “Generated Invisible Primary Keys”](#).

[sql_log_off](#)

System Variable	sql_log_off
Scope	Global, Session

Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF
	OFF (enable logging)
Valid Values	ON (disable logging)

This variable controls whether logging to the general query log is disabled for the current session (assuming that the general query log itself is enabled). The default value is `OFF` (that is, enable logging). To disable or enable general query logging for the current session, set the session [sql_log_off](#) variable to `ON` or `OFF`.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

[sql_mode](#)

Command-Line Format	--sql-mode=name
System Variable	sql_mode
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Set
	ONLY_FULL_GROUP_BY STRICT_TRANS_TABLES NO_ZERO_IN_DATE
	NO_ZERO_DATE ERROR_FOR_DIVISION_BY_ZERO
	NO_ENGINE_SUBSTITUTION
	ALLOW_INVALID_DATES
	ANSI_QUOTES
	ERROR_FOR_DIVISION_BY_ZERO
	HIGH_NOT_PRECEDENCE
	IGNORE_SPACE
	NO_AUTO_VALUE_ON_ZERO
	NO_BACKSLASH_ESCAPES
	NO_DIR_IN_CREATE
	NO_ENGINE_SUBSTITUTION
	NO_UNSIGNED_SUBTRACTION
	NO_ZERO_DATE
	NO_ZERO_IN_DATE
	ONLY_FULL_GROUP_BY
	PAD_CHAR_TO_FULL_LENGTH
	PIPES_AS_CONCAT
	REAL_AS_FLOAT
	STRICT_ALL_TABLES
	STRICT_TRANS_TABLES
	TIME_TRUNCATE_FRACTIONAL
Valid Values	

The current server SQL mode, which can be set dynamically. For details, see [Section 7.1.11, “Server SQL Modes”](#).

Note

MySQL installation programs may configure the SQL mode during the installation process.

If the SQL mode differs from the default or from what you expect, check for a setting in an option file that the server reads at startup.

[sql_notes](#)

System Variable	sql_notes
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No

Type	Boolean
Default Value	ON

If enabled (the default), diagnostics of `Note` level increment `warning_count` and the server records them. If disabled, `Note` diagnostics do not increment `warning_count` and the server does not record them. [mysqldump](#) includes output to disable this variable so that reloading the dump file does not produce warnings for events that do not affect the integrity of the reload operation.

[sql_quote_show_create](#)

System Variable	sql_quote_show_create
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

If enabled (the default), the server quotes identifiers for [SHOW CREATE TABLE](#) and [SHOW CREATE DATABASE](#) statements. If disabled, quoting is disabled. This option is enabled by default so that replication works for identifiers that require quoting. See [Section 15.7.7.11, “SHOW CREATE TABLE Statement”](#), and [Section 15.7.7.7, “SHOW CREATE DATABASE Statement”](#).

[sql_require_primary_key](#)

Command-Line Format	<code>--sql-require-primary-key[={OFF ON}]</code>
System Variable	sql_require_primary_key
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	OFF

Whether statements that create new tables or alter the structure of existing tables enforce the requirement that tables have a primary key.

Setting the session value of this system variable is a restricted operation. The session user must have privileges sufficient to set restricted session variables. See [Section 7.1.9.1, “System Variable Privileges”](#).

Enabling this variable helps avoid performance problems in row-based replication that can occur when tables have no primary key. Suppose that a table has no primary key and an update or delete modifies multiple rows. On the replication source server, this operation can be performed using a single table scan but, when replicated using row-based replication, results in a table scan for each row to be modified on the replica. With a primary key, these table scans do not occur.

[sql_require_primary_key](#) applies to both base tables and `TEMPORARY` tables, and changes to its value are replicated to replica servers. The table must use MySQL storage engines that can participate in replication.

When enabled, [sql_require_primary_key](#) has these effects:

Attempts to create a new table with no primary key fail with an error. This includes `CREATE TABLE ... LIKE`. It also includes `CREATE TABLE ... SELECT`, unless the `CREATE TABLE` part includes a primary key definition.

Attempts to drop the primary key from an existing table fail with an error, with the exception that dropping the primary key and adding a primary key in the same `ALTER TABLE` statement is permitted.

Dropping the primary key fails even if the table also contains a `UNIQUE NOT NULL` index.

Attempts to import a table with no primary key fail with an error.

The `REQUIRE_TABLE_PRIMARY_KEY_CHECK` option of the [CHANGE REPLICATION SOURCE TO](#) statement enables a replica to select its own policy for primary key checks. When the option is set to `ON` for a replication channel, the replica always uses the value `ON` for the [sql_require_primary_key](#) system variable in replication operations, requiring a primary key. When the option is set to `OFF`, the replica always uses the value `OFF` for the [sql_require_primary_key](#) system variable in replication operations, so that a primary key is never required, even if the source required one. When the `REQUIRE_TABLE_PRIMARY_KEY_CHECK` option is set to `STREAM`, which is the default, the replica uses whatever value is replicated from the source for each transaction. With the `STREAM` setting for the `REQUIRE_TABLE_PRIMARY_KEY_CHECK` option, if privilege checks are in use for the replication channel, the `PRIVILEGE_CHECKS_USER` account needs privileges sufficient to set restricted session variables, so that it can set the session value for the [sql_require_primary_key](#) system variable. With the `ON` or `OFF` settings, the account does not need these privileges. For more information, see [Section 19.3.3, “Replication Privilege Checks”](#).

[sql_safe_updates](#)

System Variable	sql_safe_updates
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean

Default Value

OFF

If this variable is enabled, [UPDATE](#) and [DELETE](#) statements that do not use a key in the WHERE clause or a LIMIT clause produce an error. This makes it possible to catch [UPDATE](#) and [DELETE](#) statements where keys are not used properly and that would probably change or delete a large number of rows. The default value is OFF.

For the [mysql](#) client, [sql_safe_updates](#) can be enabled by using the [--safe-updates](#) option. For more information, see [Using Safe-Updates Mode \(--safe-updates\)](#).

[sql_select_limit](#)

System Variable	sql_select_limit
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	18446744073709551615
Minimum Value	0
Maximum Value	18446744073709551615

The maximum number of rows to return from [SELECT](#) statements. For more information, see [Using Safe-Updates Mode \(--safe-updates\)](#).

The default value for a new connection is the maximum number of rows that the server permits per table. Typical default values are $(2^{32})-1$ or $(2^{64})-1$. If you have changed the limit, the default value can be restored by assigning a value of DEFAULT.

If a [SELECT](#) has a LIMIT clause, the LIMIT takes precedence over the value of [sql_select_limit](#).

[sql_warnings](#)

System Variable	sql_warnings
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

This variable controls whether single-row [INSERT](#) statements produce an information string if warnings occur. The default is OFF. Set the value to ON to produce an information string.

[ssl_ca](#)

Command-Line Format	--ssl-ca=file_name
System Variable	ssl_ca
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The path name of the Certificate Authority (CA) certificate file in PEM format. The file contains a list of trusted SSL Certificate Authorities.

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_capath](#)

Command-Line Format	--ssl-capath=dir_name
System Variable	ssl_capath
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Directory name
Default Value	NULL

The path name of the directory that contains trusted SSL Certificate Authority (CA) certificate files in PEM format.

This variable is can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_cert](#)

Command-Line Format	--ssl-cert=file_name
----------------------------	----------------------

System Variable	ssl_cert
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The path name of the server SSL public key certificate file in PEM format.

If the server is started with [ssl_cert](#) set to a certificate that uses any restricted cipher or cipher category, the server starts with support for encrypted connections disabled. For information about cipher restrictions, see [Connection Cipher Configuration](#).

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_cipher](#)

Command-Line Format	--ssl-cipher=name
System Variable	ssl_cipher
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	NULL

The list of permissible encryption ciphers for connections that use TLSv1.2. If no cipher in the list is supported, encrypted connections that use this TLS protocol do not work.

The list may include any of the following values:

```

ECDHE-ECDSA-AES128-GCM-SHA256
ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-RSA-AES128-GCM-SHA256
ECDHE-RSA-AES256-GCM-SHA384
ECDHE-ECDSA-CHACHA20-POLY1305
ECDHE-RSA-CHACHA20-POLY1305
ECDHE-ECDSA-AES256-CCM
ECDHE-ECDSA-AES128-CCM
DHE-RSA-AES128-GCM-SHA256
DHE-RSA-AES256-GCM-SHA384
DHE-RSA-AES256-CCM
DHE-RSA-AES128-CCM
DHE-RSA-CHACHA20-POLY1305

```

Trying to include any values in the cipher list that are not shown here when setting this variable raises an error ([ER_BLOCKED_CIPHER](#)).

For greatest portability, the cipher list should be a list of one or more cipher names, separated by colons. The following example shows two cipher names separated by a colon:

```

[mysqld]
ssl_cipher="DHE-RSA-AES128-GCM-SHA256:AES128-SHA"

```

OpenSSL supports the syntax for specifying ciphers described in the OpenSSL documentation at <https://www.openssl.org/docs/manmaster/man1/ciphers.html>.

For information about which encryption ciphers MySQL supports, see [Section 8.3.2, "Encrypted Connection TLS Protocols and Ciphers"](#).

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_crl](#)

Command-Line Format	--ssl-crl=file_name
System Variable	ssl_crl
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The path name of the file containing certificate revocation lists in PEM format.

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_crlpath](#)

Command-Line Format	--ssl-crlpath=dir_name
System Variable	ssl_crlpath
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Directory name
Default Value	NULL

The path of the directory that contains certificate revocation-list files in PEM format.

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_fips_mode](#)

Command-Line Format	--ssl-fips-mode={OFF ON STRICT}
Deprecated	Yes
System Variable	ssl_fips_mode
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Enumeration
Default Value	OFF
	OFF (or 0)
Valid Values	ON (or 1)
	STRICT (or 2)

Controls whether to enable FIPS mode on the server side. The [ssl_fips_mode](#) system variable differs from other `ssl_xxx` system variables in that it is not used to control whether the server permits encrypted connections, but rather to affect which cryptographic operations are permitted. See [Section 8.8, “FIPS Support”](#).

These [ssl_fips_mode](#) values are permitted:

OFF (or 0): Disable FIPS mode.

ON (or 1): Enable FIPS mode.

STRICT (or 2): 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 at startup causes the server to produce an error message and exit.

This option is deprecated and made read-only. Expect it to be removed in a future version of MySQL.

[ssl_key](#)

Command-Line Format	--ssl-key=file_name
System Variable	ssl_key
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	File name
Default Value	NULL

The path name of the server SSL private key file in PEM format. For better security, use a certificate with an RSA key size of at least 2048 bits.

If the key file is protected by a passphrase, the server prompts the user for the passphrase. The password must be given interactively; it cannot be stored in a file. If the passphrase is incorrect, the program continues as if it could not read the key.

This variable can be modified at runtime to affect the TLS context the server uses for new connections established after the execution of [ALTER INSTANCE RELOAD TLS](#) or after a restart if the variable value was persisted. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

[ssl_session_cache_mode](#)

Command-Line Format	--ssl_session_cache_mode={ON OFF}
System Variable	ssl_session_cache_mode
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON
	ON
Valid Values	OFF

Controls whether to enable the session cache in memory on the server side and session-ticket generation by the server. The default mode is ON (enable session cache mode). A change to the [ssl_session_cache_mode](#) system variable has an effect only after the [ALTER INSTANCE RELOAD TLS](#) statement has been executed, or after a restart if the variable value was persisted.

These [ssl_session_cache_mode](#) values are permitted:

ON: Enable session cache mode.

OFF: Disable session cache mode.

The server does not advertise its support for session resumption if the value of this system variable is OFF. When running on OpenSSL 1.0.x the session tickets are always generated, but the tickets are not usable when [ssl_session_cache_mode](#) is enabled.

The current value in effect for [ssl_session_cache_mode](#) can be observed with the [Ssl_session_cache_mode](#) status variable.

[ssl_session_cache_timeout](#)

Command-Line Format	--ssl_session_cache_timeout
System Variable	ssl_session_cache_timeout
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	300
Minimum Value	0
Maximum Value	84600
Unit	seconds

Sets a period of time during which prior session reuse is permitted when establishing a new encrypted connection to the server, provided the [ssl_session_cache_mode](#) system variable is enabled and prior session data is available. If the session timeout expires, a session can no longer be reused.

The default value is 300 seconds and the maximum value is 84600 (or one day in seconds). A change to the [ssl_session_cache_timeout](#) system variable has an effect only after the [ALTER INSTANCE RELOAD TLS](#) statement has been executed, or after a restart if the variable value was persisted. The current value in effect for [ssl_session_cache_timeout](#) can be observed with the [Ssl_session_cache_timeout](#) status variable.

[stored_program_cache](#)

Command-Line Format	--stored-program-cache=#
System Variable	stored_program_cache
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	256
Minimum Value	16
Maximum Value	524288

Sets a soft upper limit for the number of cached stored routines per connection. The value of this variable is specified in terms of the number of stored routines held in each of the two caches maintained by the MySQL Server for, respectively, stored procedures and stored functions.

Whenever a stored routine is executed this cache size is checked before the first or top-level statement in the routine is parsed; if the number of routines of the same type (stored procedures or stored functions according to which is being executed) exceeds the limit specified by this variable, the corresponding cache is flushed and memory previously allocated for cached objects is freed. This allows the cache to be flushed safely, even when there are dependencies between stored routines.

The stored procedure and stored function caches exist in parallel with the stored program definition cache partition of the [dictionary object cache](#). The stored procedure and stored function caches are per connection, while the stored program definition cache is shared. The existence of objects in the stored procedure and stored function caches have no dependence on the existence of objects in the stored program definition cache, and vice versa. For more information, see [Section 16.4, “Dictionary Object Cache”](#).

[stored_program_definition_cache](#)

Command-Line Format	--stored-program-definition-cache=#
System Variable	stored_program_definition_cache
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	256
Minimum Value	256
Maximum Value	524288

Defines a limit for the number of stored program definition objects, both used and unused, that can be kept in the dictionary object cache.

Unused stored program definition objects are only kept in the dictionary object cache when the number in use is less than the capacity defined by [stored_program_definition_cache](#).

A setting of 0 means that stored program definition objects are only kept in the dictionary object cache while they are in use.

The stored program definition cache partition exists in parallel with the stored procedure and stored function caches that are configured using the [stored_program_cache](#) option.

The [stored_program_cache](#) option sets a soft upper limit for the number of cached stored procedures or functions per connection, and the limit is checked each time a connection executes a stored procedure or function. The stored program definition cache partition, on the other hand, is a shared cache that stores stored program definition objects for other purposes. The existence of objects in the stored program definition cache partition has no dependence on the existence of objects in the stored procedure cache or stored function cache, and vice versa.

For related information, see [Section 16.4, “Dictionary Object Cache”](#).

[super_read_only](#)

Command-Line Format	--super-read-only[={OFF ON}]
System Variable	super_read_only
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

If the [read_only](#) system variable is enabled, the server permits no client updates except from users who have the [CONNECTION_ADMIN](#) privilege (or the deprecated [SUPER](#) privilege). If the [super_read_only](#) system variable is also enabled, the server prohibits client updates even from users who have [CONNECTION_ADMIN](#) or [SUPER](#). See the description of the [read_only](#) system variable for a description of read-only mode and information about how [read_only](#) and [super_read_only](#) interact.

Client updates prevented when [super_read_only](#) is enabled include operations that do not necessarily appear to be updates, such as `CREATE FUNCTION` (to install a loadable function), `INSTALL PLUGIN`, and `INSTALL COMPONENT`. These operations are prohibited because they involve changes to tables in the `mysql` system schema.

Similarly, if the Event Scheduler is enabled, enabling the [super_read_only](#) system variable prevents it from updating event “last executed” timestamps in the `events` data dictionary table. This causes the Event Scheduler to stop the next time it tries to execute a scheduled event, after writing a message to the server error log. (In this situation the [event_scheduler](#) system variable does not change from `ON` to `OFF`. An implication is that this variable rejects the DBA *intent* that the Event Scheduler be enabled or disabled, where its actual status of started or stopped may be distinct.). If [super_read_only](#) is subsequently disabled after being enabled, the server automatically restarts the Event Scheduler as needed.

Changes to [super_read_only](#) on a replication source server are not replicated to replica servers. The value can be set on a replica independent of the setting on the source.

[syseventlog.facility](#)

Command-Line Format	--syseventlog.facility=value
System Variable	syseventlog.facility
Scope	Global

Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	daemon

The facility for error log output written to `syslog` (what type of program is sending the message). This variable is unavailable unless the `log_sink_syseventlog` error log component is installed. See [Section 7.4.2.8, “Error Logging to the System Log”](#).

The permitted values can vary per operating system; consult your system `syslog` documentation.

This variable does not exist on Windows.

[syseventlog.include_pid](#)

Command-Line Format	<code>--syseventlog.include-pid[={OFF ON}]</code>
System Variable	syseventlog.include_pid
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

Whether to include the server process ID in each line of error log output written to `syslog`. This variable is unavailable unless the `log_sink_syseventlog` error log component is installed. See [Section 7.4.2.8, “Error Logging to the System Log”](#).

This variable does not exist on Windows.

[syseventlog.tag](#)

Command-Line Format	<code>--syseventlog.tag=tag</code>
System Variable	syseventlog.tag
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	empty string

The tag to be added to the server identifier in error log output written to `syslog` or the Windows Event Log. This variable is unavailable unless the `log_sink_syseventlog` error log component is installed. See [Section 7.4.2.8, “Error Logging to the System Log”](#).

By default, no tag is set, so the server identifier is simply `MySQL` on Windows, and `mysqld` on other platforms. If a tag value of `tag` is specified, it is appended to the server identifier with a leading hyphen, resulting in a `syslog` identifier of `mysqld-tag` (or `MySQL-tag` on Windows).

On Windows, to use a tag that does not already exist, the server must be run from an account with Administrator privileges, to permit creation of a registry entry for the tag. Elevated privileges are not required if the tag already exists.

[system_time_zone](#)

System Variable	system_time_zone
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The server system time zone. When the server begins executing, it inherits a time zone setting from the machine defaults, possibly modified by the environment of the account used for running the server or the startup script. The value is used to set [system_time_zone](#). To explicitly specify the system time zone, set the `TZ` environment variable or use the `--timezone` option of the [mysqld_safe](#) script.

In addition to startup time initialization, if the server host time zone changes (for example, due to daylight saving time), [system_time_zone](#) reflects that change, which has these implications for applications:

Queries that reference [system_time_zone](#) will get one value before a daylight saving change and a different value after the change.

For queries that begin executing before a daylight saving change and end after the change, the [system_time_zone](#) remains constant within the query because the value is usually cached at the beginning of execution.

The [system_time_zone](#) variable differs from the [time_zone](#) variable. Although they might have the same value, the latter variable is used to initialize the time zone for each client that connects. See [Section 7.1.15, “MySQL Server Time Zone Support”](#).

[table_definition_cache](#)

Command-Line Format	<code>--table-definition-cache=#</code>
System Variable	table_definition_cache

Scope	Global
Dynamic	Yes
<u>SET VAR</u> Hint Applies	No
Type	Integer
Default Value	-1 (signifies autosizing; do not assign this literal value)
Minimum Value	400
Maximum Value	524288

The number of table definitions that can be stored in the table definition cache. If you use a large number of tables, you can create a large table definition cache to speed up opening of tables. The table definition cache takes less space and does not use file descriptors, unlike the normal table cache. The minimum value is 400. The default value is based on the following formula, capped to a limit of 2000:

```
MIN(400 + table_open_cache / 2, 2000)
```

For [InnoDB](#), the [table_definition_cache](#) setting acts as a soft limit for the number of table instances in the dictionary object cache and the number file-per-table tablespaces that can be open at one time.

If the number of table instances in the dictionary object cache exceeds the [table_definition_cache](#) limit, an LRU mechanism begins marking table instances for eviction and eventually removes them from the dictionary object cache. The number of open tables with cached metadata can be higher than the [table_definition_cache](#) limit due to table instances with foreign key relationships, which are not placed on the LRU list.

The number of file-per-table tablespaces that can be open at one time is limited by both the [table_definition_cache](#) and [innodb_open_files](#) settings. If both variables are set, the highest setting is used. If neither variable is set, the [table_definition_cache](#) setting, which has a higher default value, is used. If the number of open tablespaces exceeds the limit defined by [table_definition_cache](#) or [innodb_open_files](#), an LRU mechanism searches the LRU list for tablespace files that are fully flushed and not currently being extended. This process is performed each time a new tablespace is opened. Only inactive tablespaces are closed.

The table definition cache exists in parallel with the table definition cache partition of the [dictionary object cache](#). Both caches store table definitions but serve different parts of the MySQL server. Objects in one cache have no dependence on the existence of objects in the other. For more information, see [Section 16.4, "Dictionary Object Cache"](#).

[table_encryption_privilege_check](#)

Command-Line Format	--table-encryption-privilege-check[={OFF ON}]
System Variable	table_encryption_privilege_check
Scope	Global
Dynamic	Yes
<u>SET VAR</u> Hint Applies	No
Type	Boolean
Default Value	OFF

Controls the [TABLE_ENCRYPTION_ADMIN](#) privilege check that occurs when creating or altering a schema or general tablespace with encryption that differs from the [default_table_encryption](#) setting, or when creating or altering a table with an encryption setting that differs from the default schema encryption. The check is disabled by default.

Setting [table_encryption_privilege_check](#) at runtime requires the [SUPER](#) privilege.

[table_encryption_privilege_check](#) supports [SET PERSIST](#) and [SET PERSIST ONLY](#) syntax. See [Section 7.1.9.3, "Persisted System Variables"](#).

For more information, see [Defining an Encryption Default for Schemas and General Tablespaces](#).

[table_open_cache](#)

Command-Line Format	--table-open-cache=#
System Variable	table_open_cache
Scope	Global
Dynamic	Yes
<u>SET VAR</u> Hint Applies	No
Type	Integer
Default Value	4000
Minimum Value	1
Maximum Value	524288

The number of open tables for all threads. Increasing this value increases the number of file descriptors that [mysqld](#) requires. The effective value of this variable is the greater of the effective value of [open_files_limit](#) - 10 - the effective value of [max_connections](#) / 2, and 400; that is

```
MAX(
    (open_files_limit - 10 - max_connections) / 2,
    400
)
```

You can check whether you need to increase the table cache by checking the [Opened tables](#) status variable. If the value of [Opened tables](#) is large and you do not use [FLUSH TABLES](#) often (which just forces all tables to be closed and reopened), then you should increase the value of the [table_open_cache](#) variable. For more information about the table cache, see [Section 10.4.3.1, “How MySQL Opens and Closes Tables”](#).

[table_open_cache_instances](#)

Command-Line Format	--table-open-cache-instances=#
System Variable	table_open_cache_instances
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	16
Minimum Value	1
Maximum Value	64

The number of open tables cache instances. To improve scalability by reducing contention among sessions, the open tables cache can be partitioned into several smaller cache instances of size [table_open_cache](#) / [table_open_cache_instances](#) . A session needs to lock only one instance to access it for DML statements. This segments cache access among instances, permitting higher performance for operations that use the cache when there are many sessions accessing tables. (DDL statements still require a lock on the entire cache, but such statements are much less frequent than DML statements.)

A value of 8 or 16 is recommended on systems that routinely use 16 or more cores. However, if you have many large triggers on your tables that cause a high memory load, the default setting for [table_open_cache_instances](#) might lead to excessive memory usage. In that situation, it can be helpful to set [table_open_cache_instances](#) to 1 in order to restrict memory usage.

[tablespace_definition_cache](#)

Command-Line Format	--tablespace-definition-cache=#
System Variable	tablespace_definition_cache
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	256
Minimum Value	256
Maximum Value	524288

Defines a limit for the number of tablespace definition objects, both used and unused, that can be kept in the dictionary object cache.

Unused tablespace definition objects are only kept in the dictionary object cache when the number in use is less than the capacity defined by [tablespace_definition_cache](#).

A setting of 0 means that tablespace definition objects are only kept in the dictionary object cache while they are in use.

For more information, see [Section 16.4, “Dictionary Object Cache”](#).

[temptable_max_mmap](#)

Command-Line Format	--temptable-max-mmap=#
System Variable	temptable_max_mmap
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	2 ⁶⁴ -1
Unit	bytes

Defines the maximum amount of memory (in bytes) the TempTable storage engine is permitted to allocate from memory-mapped temporary files before it starts storing data to InnoDB internal temporary tables on disk. A setting of 0 (default) disables allocation of memory from memory-mapped temporary files. For more information, see [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

Before MySQL 8.4, this option was set to 1 GiB instead of 0.

[temptable_max_ram](#)

Command-Line Format	--temptable-max-ram=#
System Variable	temptable_max_ram
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No

Type	Integer
Default Value	3% of total memory: min 1 GB, max 4 GB
Minimum Value	2097152
Maximum Value	2 ⁶⁴ -1
Unit	bytes

Defines the maximum amount of memory that can be occupied by the TempTable storage engine before it starts storing data on disk. The default value is 3% of total memory available on the server, with a minimum and maximum default range of 1-4 GiB. For more information, see [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

Before MySQL 8.4, the default value was always 1 GiB.

[temptable_use_mmap](#)

Command-Line Format	--temptable-use-mmap[={OFF ON}]
Deprecated	Yes
System Variable	temptable_use_mmap
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Defines whether the TempTable storage engine allocates space for internal in-memory temporary tables as memory-mapped temporary files when the amount of memory occupied by the TempTable storage engine exceeds the limit defined by the [temptable_max_ram](#) variable. When [temptable_use_mmap](#) is disabled (default), the TempTable storage engine uses InnoDB on-disk internal temporary tables instead. For more information, see [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

[thread_cache_size](#)

Command-Line Format	--thread-cache-size=#
System Variable	thread_cache_size
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	-1 (signifies autosizing; do not assign this literal value)
Minimum Value	0
Maximum Value	16384

How many threads the server should cache for reuse. When a client disconnects, the client's threads are put in the cache if there are fewer than [thread_cache_size](#) threads there. Requests for threads are satisfied by reusing threads taken from the cache if possible, and only when the cache is empty is a new thread created. This variable can be increased to improve performance if you have a lot of new connections. Normally, this does not provide a notable performance improvement if you have a good thread implementation. However, if your server sees hundreds of connections per second you should normally set [thread_cache_size](#) high enough so that most new connections use cached threads. By examining the difference between the [Connections](#) and [Threads_created](#) status variables, you can see how efficient the thread cache is. For details, see [Section 7.1.10, “Server Status Variables”](#).

The default value is based on the following formula, capped to a limit of 100:

$$8 + (\text{max_connections} / 100)$$

[thread_handling](#)

Command-Line Format	--thread-handling=name
System Variable	thread_handling
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Enumeration
Default Value	one-thread-per-connection no-threads
Valid Values	one-thread-per-connection loaded-dynamically

The thread-handling model used by the server for connection threads. The permissible values are `no-threads` (the server uses a single thread to handle one connection), `one-thread-per-connection` (the server uses one thread to handle each client connection), and `loaded-dynamically` (set by the thread pool plugin when it initializes). `no-threads` is useful for debugging under Linux; see [Section 7.9, “Debugging MySQL”](#).

[thread_pool_algorithm](#)

Command-Line Format	--thread-pool-algorithm=#
System Variable	thread_pool_algorithm
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	1

This variable controls which algorithm the thread pool plugin uses:

0: Use a conservative low-concurrency algorithm.

1: Use an aggressive high-concurrency algorithm which performs better with optimal thread counts, but performance may be degraded if the number of connections reaches extremely high values.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, "MySQL Enterprise Thread Pool"](#).

[thread_pool_dedicated_listeners](#)

Command-Line Format	--thread-pool-dedicated-listeners
System Variable	thread_pool_dedicated_listeners
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

Dedicates a listener thread in each thread group to listen for incoming statements from connections assigned to the group.

OFF: (Default) Disables dedicated listener threads.

ON: Dedicates a listener thread in each thread group to listen for incoming statements from connections assigned to the group. Dedicated listener threads do not execute queries.

Enabling [thread_pool_dedicated_listeners](#) is only useful when a transaction limit is defined by `thread_pool_max_transactions_limit`. Otherwise, [thread_pool_dedicated_listeners](#) should not be enabled.

MySQL HeatWave Service uses this variable, which is available only with MySQL Enterprise Edition, and not supported in MySQL 8.4.

[thread_pool_high_priority_connection](#)

Command-Line Format	--thread-pool-high-priority-connection=#
System Variable	thread_pool_high_priority_connection
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	1

This variable affects queuing of new statements prior to execution. If the value is 0 (false, the default), statement queuing uses both the low-priority and high-priority queues. If the value is 1 (true), queued statements always go to the high-priority queue.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, "MySQL Enterprise Thread Pool"](#).

[thread_pool_longrun_trx_limit](#)

Command-Line Format	--thread-pool-longrun-trx-limit=#
System Variable	thread_pool_longrun_trx_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	2000
Minimum Value	10
Maximum Value	60*60*24
Unit	ms

When [thread_pool_max_transactions_limit](#) is in use, there is a maximum number of transactions that can be active in each thread group. If entire number available is being used by long-running transactions, any additional transaction assigned to the group blocks until one of the long-running transactions is completed, which users can perceive as an inexplicable hang.

To mitigate this issue, the limit for a given thread group is suspended if all of the threads using up the transaction maximum have been executing longer than the interval (in milliseconds) specified by [thread_pool_longrun_trx_limit](#). When the number of long-running transactions decreases, [thread_pool_max_transactions_limit](#) can be (and is) enabled again. In order for this to happen, the number of ongoing transactions must be less than $\text{thread_pool_max_transactions_limit} / 2$ for the interval defined as shown:

```
MIN( MAX(thread_pool_longrun_trx_limit * 15, 5000), 30000)
```

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_max_active_query_threads](#)

Command-Line Format	--thread-pool-max-active-query-threads
System Variable	thread_pool_max_active_query_threads
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	512

The maximum permissible number of active (running) query threads per group. If the value is 0, the thread pool plugin uses up to as many threads as are available.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_max_transactions_limit](#)

Command-Line Format	--thread-pool-max-transactions-limit
System Variable	thread_pool_max_transactions_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	1000000

The maximum number of transactions permitted by the thread pool plugin. Defining a transaction limit binds a thread to a transaction until it commits, which helps stabilize throughput during high concurrency.

The default value of 0 means that there is no transaction limit. The variable is dynamic but cannot be changed from 0 to a higher value at runtime and vice versa. A non-zero value at startup permits dynamic configuration at runtime. The [CONNECTION_ADMIN](#) privilege is required to configure [thread_pool_max_transactions_limit](#) at runtime.

When you define a transaction limit, enabling [thread_pool_dedicated_listeners](#) creates a dedicated listener thread in each thread group. The additional dedicated listener thread consumes more resources and affects thread pool performance. [thread_pool_dedicated_listeners](#) should therefore be used cautiously.

When the limit defined by [thread_pool_max_transactions_limit](#) has been reached, new connections or transactions on existing connections may appear to hang until one or more existing transactions are completed. It should be possible in many cases to mitigate this issue by setting [thread_pool_longrun_trx_limit](#) so that the transaction maximum can be relaxed when the number of ongoing transactions matches it for a given length of time. If existing connections continue to be blocked or long-running even after attempting this, a privileged connection may be required to access the server to increase the limit, remove the limit, or kill running transactions. See [Privileged Connections](#).

MySQL HeatWave Service uses this variable, which is available only with MySQL Enterprise Edition, and not supported in MySQL 8.4.

[thread_pool_max_unused_threads](#)

Command-Line Format	--thread-pool-max-unused-threads=#
System Variable	thread_pool_max_unused_threads
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	4096

The maximum permitted number of unused threads in the thread pool. This variable makes it possible to limit the amount of memory used by sleeping threads.

A value of 0 (the default) means no limit on the number of sleeping threads. A value of N where N is greater than 0 means 1 consumer thread and $N-1$ reserve threads. In this case, if a thread is ready to sleep but the number of sleeping threads is already at the maximum, the thread exits rather than going to sleep.

A sleeping thread is either sleeping as a consumer thread or a reserve thread. The thread pool permits one thread to be the consumer thread when sleeping. If a thread goes to sleep and there is no existing consumer thread, it sleeps as a consumer thread. When a thread must be woken up, a consumer thread is selected if there is one. A reserve thread is selected only when there is no consumer thread to wake up.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_prio_kickup_timer](#)

Command-Line Format	--thread-pool-prio-kickup-timer=#
System Variable	thread_pool_prio_kickup_timer
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1000
Minimum Value	0
Maximum Value	4294967294
Unit	milliseconds

This variable affects statements waiting for execution in the low-priority queue. The value is the number of milliseconds before a waiting statement is moved to the high-priority queue. The default is 1000 (1 second).

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_query_threads_per_group](#)

Command-Line Format	--thread-pool-query-threads-per-group
System Variable	thread_pool_query_threads_per_group
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	1
Minimum Value	1
Maximum Value	4096

The maximum number of query threads permitted in a thread group. The maximum value is 4096, but if [thread_pool_max_transactions_limit](#) is set, [thread_pool_query_threads_per_group](#) must not exceed that value.

The default value of 1 means there is one active query thread in each thread group, which works well for many loads. When you are using the high concurrency thread pool algorithm (`thread_pool_algorithm = 1`), consider increasing the value if you experience slower response times due to long-running transactions.

The [CONNECTION ADMIN](#) privilege is required to configure [thread_pool_query_threads_per_group](#) at runtime.

If you decrease the value of [thread_pool_query_threads_per_group](#) at runtime, threads that are currently running user queries are allowed to complete, then moved to the reserve pool or terminated. If you increment the value at runtime and the thread group needs more threads, these are taken from the reserve pool if possible, otherwise they are created.

MySQL HeatWave Service uses this variable, which is available only with MySQL Enterprise Edition, and not supported in MySQL 8.4.

[thread_pool_size](#)

Command-Line Format	--thread-pool-size=#
System Variable	thread_pool_size
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	16
Minimum Value	1
Maximum Value	512

The number of thread groups in the thread pool. This is the most important parameter controlling thread pool performance. It affects how many statements can execute simultaneously. If a value outside the range of permissible values is specified, the thread pool plugin does not load and the server writes a message to the error log.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_stall_limit](#)

Command-Line Format	--thread-pool-stall-limit=#
System Variable	thread_pool_stall_limit
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	6
Minimum Value	4
Maximum Value	600
Unit	milliseconds * 10

This variable affects executing statements. The value is the amount of time a statement has to finish after starting to execute before it becomes defined as stalled, at which point the thread pool permits the thread group to begin executing another statement. The value is measured in 10 millisecond units, so the default of 6 means 60ms. Short wait values permit threads to start more quickly. Short values are also better for avoiding deadlock situations. Long wait values are useful for workloads that include long-running statements, to avoid starting too many new statements while the current ones execute.

This variable is available only if the thread pool plugin is enabled. See [Section 7.6.3, “MySQL Enterprise Thread Pool”](#).

[thread_pool_transaction_delay](#)

Command-Line Format	--thread-pool-transaction-delay
System Variable	thread_pool_transaction_delay
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	0
Minimum Value	0
Maximum Value	300000

The delay period before executing a new transaction, in milliseconds. The maximum value is 300000 (5 minutes).

A transaction delay can be used in cases where parallel transactions affect the performance of other operations due to resource contention. For example, if parallel transactions affect index creation or an online buffer pool resizing operation, you can configure a transaction delay to reduce resource contention while those operations are running.

Worker threads sleep for the number of milliseconds specified by `thread_pool_transaction_delay` before executing a new transaction.

The `thread_pool_transaction_delay` setting does not affect queries issued from a privileged connection (a connection assigned to the Admin thread group). These queries are not subject to a configured transaction delay.

[thread_stack](#)

Command-Line Format	--thread-stack=#
System Variable	thread_stack
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Integer
Default Value	1048576
Minimum Value	131072
Maximum Value (64-bit platforms)	18446744073709550592
Maximum Value (32-bit platforms)	4294966272
Unit	bytes
Block Size	1024

The stack size for each thread. The default is large enough for normal operation. If the thread stack size is too small, it limits the complexity of the SQL statements that the server can handle, the recursion depth of stored procedures, and other memory-consuming actions.

[time_zone](#)

System Variable	time_zone
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	String
Default Value	SYSTEM
Minimum Value	-13:59

Maximum Value

+14:00

The current time zone. This variable is used to initialize the time zone for each client that connects. By default, the initial value of this is 'SYSTEM' (which means, "use the value of [system_time_zone](#)"). The value can be specified explicitly at server startup with the [--default-time-zone](#) option. See [Section 7.1.15, "MySQL Server Time Zone Support"](#).

Note

If set to SYSTEM, every MySQL function call that requires a time zone calculation makes a system library call to determine the current system time zone. This call may be protected by a global mutex, resulting in contention.

[timestamp](#)

System Variable	timestamp
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Numeric
Default Value	UNIX_TIMESTAMP()
Minimum Value	1
Maximum Value	2147483647

Set the time for this client. This is used to get the original timestamp if you use the binary log to restore rows. *timestamp_value* should be a Unix epoch timestamp (a value like that returned by [UNIX_TIMESTAMP\(\)](#), not a value in 'YYYY-MM-DD hh:mm:ss' format) or DEFAULT.

Setting [timestamp](#) to a constant value causes it to retain that value until it is changed again. Setting [timestamp](#) to DEFAULT causes its value to be the current date and time as of the time it is accessed.

[timestamp](#) is a DOUBLE rather than BIGINT because its value includes a microseconds part. The maximum value corresponds to '2038-01-19 03:14:07' UTC, the same as for the [TIMESTAMP](#) data type.

SET timestamp affects the value returned by [NOW\(\)](#) but not by [SYSDATE\(\)](#). This means that timestamp settings in the binary log have no effect on invocations of [SYSDATE\(\)](#). The server can be started with the [--sysdate-is-now](#) option to cause [SYSDATE\(\)](#) to be a synonym for [NOW\(\)](#), in which case SET timestamp affects both functions.

[tls_certificates_enforced_validation](#)

Command-Line Format	--tls-certificates-enforced-validation={OFF ON}
System Variable	tls_certificates_enforced_validation
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

During startup, the server ensures that the location of each required SSL certificate file is present in the default data directory if the file locations are not given on the command line. However, the server does not validate the certificate files and, as a result, it is able to start with an invalid certificate. The [tls_certificates_enforced_validation](#) system variable controls whether certificate validation is enforced at startup. Discovery of an invalid certificate halts the startup execution when validation enforcement is enabled. By default, certificate validation enforcement is disabled (OFF).

Validation enforcement can be enabled by specifying the [--tls-certificates-enforced-validation](#) option on the command line with or without the ON value. With validation enforcement enabled, certificates are also validated at the time of reloading them through the [ALTER INSTANCE RELOAD TLS](#) statement. This system variable cannot be persisted across reboots. For more information, see [Configuring Certificate Validation Enforcement](#).

[tls_ciphersuites](#)

Command-Line Format	--tls-ciphersuites=ciphersuite_list
System Variable	tls_ciphersuites
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	NULL

Which ciphersuites the server permits for encrypted connections that use TLSv1.3. The value is a list of zero or more colon-separated ciphersuite names from among those listed here:

```
TLS_AES_128_GCM_SHA256
```

```
TLS_AES_256_GCM_SHA384
```

```
TLS_CHACHA20_POLY1305_SHA256
```


Trying to include any values in the cipher list that are not shown here when setting this variable raises an error ([ER_BLOCKED_CIPHER](#)).

The ciphersuites that can be named for this variable depend on the SSL library used to compile MySQL. If this variable is not set, its default value is `NULL`, which means that the server permits the default set of ciphersuites. If the variable is set to the empty string, no ciphersuites are enabled and encrypted connections cannot be established. For more information, see [Section 8.3.2, “Encrypted Connection TLS Protocols and Ciphers”](#).

[tls_version](#)

Command-Line Format	<code>--tls-version=protocol_list</code>
System Variable	tls_version
Scope	Global
Dynamic	Yes
SET VAR Hint Applies	No
Type	String
Default Value	TLSv1.2,TLSv1.3

Which protocols the server permits for encrypted connections. The value is a list of one or more comma-separated protocol names, which are not case-sensitive. The protocols that can be named for this variable depend on the SSL library used to compile MySQL. Permitted protocols should be chosen such as not to leave “holes” in the list. For details, see [Section 8.3.2, “Encrypted Connection TLS Protocols and Ciphers”](#).

This variable can be modified at runtime to affect the TLS context the server uses for new connections. See [Server-Side Runtime Configuration and Monitoring for Encrypted Connections](#).

Important

MySQL 8.4 does not support the TLSv1 and TLSv1.1 connection protocols. See [Removal of Support for the TLSv1 and TLSv1.1 Protocols](#) for more information.

Support for the TLSv1.3 protocol is available in MySQL 8.4, provided that MySQL Server was compiled using OpenSSL 1.1.1 or higher. The server checks the version of OpenSSL at startup, and if it is lower than 1.1.1, TLSv1.3 is removed from the default value for the system variable. In that case, the default is TLSv1.2.

[tmp_table_size](#)

Command-Line Format	<code>--tmp-table-size=#</code>
System Variable	tmp_table_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Integer
Default Value	16777216
Minimum Value	1024
Maximum Value	18446744073709551615
Unit	bytes

Defines the maximum size of internal in-memory temporary tables created by the `MEMORY` and `TempTable` storage engines. If an internal in-memory temporary table exceeds this size, it is automatically converted to an on-disk internal temporary table.

The [tmp_table_size](#) variable does not apply to user-created `MEMORY` tables. User-created `TempTable` tables are not supported.

When using the `MEMORY` storage engine for internal in-memory temporary tables, the actual size limit is the smaller of [tmp_table_size](#) and [max_heap_table_size](#). The [max_heap_table_size](#) setting does not apply to `TempTable` tables.

Increase the value of [tmp_table_size](#) (and [max_heap_table_size](#) if necessary when using the `MEMORY` storage engine for internal in-memory temporary tables) if you do many advanced `GROUP BY` queries and you have lots of memory.

You can compare the number of internal on-disk temporary tables created to the total number of internal temporary tables created by comparing [Created tmp disk tables](#) and [Created tmp tables](#) values.

See also [Section 10.4.4, “Internal Temporary Table Use in MySQL”](#).

[tmpdir](#)

Command-Line Format	<code>--tmpdir=dir_name</code>
System Variable	tmpdir
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	Directory name

The path of the directory to use for creating temporary files. It might be useful if your default `/tmp` directory resides on a partition that is too small to hold temporary tables. This variable can be set to a list of several paths that are used in round-robin fashion. Paths should be separated by colon characters (:) on Unix and semicolon characters (;) on Windows.

[tmpdir](#) can be a non-permanent location, such as a directory on a memory-based file system or a directory that is cleared when the server host restarts. If the MySQL server is acting as a replica, and you are using a non-permanent location for [tmpdir](#), consider setting a different temporary directory for the replica using the [replica_load_tmpdir](#) variable. For a replica, the temporary files used to replicate [LOAD DATA](#) statements are stored in this directory, so with a permanent location they can survive machine restarts, although replication can now continue after a restart if the temporary files have been removed.

For more information about the storage location of temporary files, see [Section B.3.3.5, “Where MySQL Stores Temporary Files”](#).

[transaction_alloc_block_size](#)

Command-Line Format	<code>--transaction-alloc-block-size=#</code>
System Variable	transaction_alloc_block_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	8192
Minimum Value	1024
Maximum Value	131072
Unit	bytes
Block Size	1024

The amount in bytes by which to increase a per-transaction memory pool which needs memory. See the description of [transaction_prealloc_size](#).

[transaction_isolation](#)

Command-Line Format	<code>--transaction-isolation=name</code>
System Variable	transaction_isolation
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Enumeration
Default Value	REPEATABLE-READ
	READ-UNCOMMITTED
	READ-COMMITTED
Valid Values	REPEATABLE-READ
	SERIALIZABLE

The transaction isolation level. The default is [REPEATABLE-READ](#).

The transaction isolation level has three scopes: global, session, and next transaction. This three-scope implementation leads to some nonstandard isolation-level assignment semantics, as described later.

To set the global transaction isolation level at startup, use the [--transaction-isolation](#) server option.

At runtime, the isolation level can be set directly using the [SET](#) statement to assign a value to the [transaction_isolation](#) system variable, or indirectly using the [SET TRANSACTION](#) statement. If you set [transaction_isolation](#) directly to an isolation level name that contains a space, the name should be enclosed within quotation marks, with the space replaced by a dash. For example, use this [SET](#) statement to set the global value:

```
SET GLOBAL transaction_isolation = 'READ-COMMITTED';
```

Setting the global [transaction_isolation](#) value sets the isolation level for all subsequent sessions. Existing sessions are unaffected.

To set the session or next-level [transaction_isolation](#) value, use the [SET](#) statement. For most session system variables, these statements are equivalent ways to set the value:

```
SET @@SESSION.var_name = value;
SET SESSION var_name = value;
SET var_name = value;
SET @@var_name = value;
```

As mentioned previously, the transaction isolation level has a next-transaction scope, in addition to the global and session scopes. To enable the next-transaction scope to be set, [SET](#) syntax for assigning session system variable values has nonstandard semantics for [transaction_isolation](#):

To set the session isolation level, use any of these syntaxes:

```
SET @@SESSION.transaction_isolation = value;
SET SESSION transaction_isolation = value;
SET transaction_isolation = value;
```

For each of those syntaxes, these semantics apply:

Sets the isolation level for all subsequent transactions performed within the session.

Permitted within transactions, but does not affect the current ongoing transaction.

If executed between transactions, overrides any preceding statement that sets the next-transaction isolation level.

Corresponds to [SET SESSION TRANSACTION ISOLATION LEVEL](#) (with the `SESSION` keyword).

To set the next-transaction isolation level, use this syntax:

```
SET @@transaction_isolation = value;
```

For that syntax, these semantics apply:

Sets the isolation level only for the next single transaction performed within the session.

Subsequent transactions revert to the session isolation level.

Not permitted within transactions.

Corresponds to [SET TRANSACTION ISOLATION LEVEL](#) (without the `SESSION` keyword).

For more information about [SET TRANSACTION](#) and its relationship to the [transaction_isolation](#) system variable, see [Section 15.3.7, "SET TRANSACTION Statement"](#).

[transaction_prealloc_size](#)

Command-Line Format	--transaction-prealloc-size=#
Deprecated	Yes
System Variable	transaction_prealloc_size
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	4096
Minimum Value	1024
Maximum Value	131072
Unit	bytes
Block Size	1024

There is a per-transaction memory pool from which various transaction-related allocations take memory. The initial size of the pool in bytes is `transaction_prealloc_size`. For every allocation that cannot be satisfied from the pool because it has insufficient memory available, the pool is increased by [transaction_alloc_block_size](#) bytes. When the transaction ends, the pool is truncated to `transaction_prealloc_size` bytes. By making `transaction_prealloc_size` sufficiently large to contain all statements within a single transaction, you can avoid many `malloc()` calls.

`transaction_prealloc_size` is deprecated, and setting this variable no longer has any effect. Expect `transaction_prealloc_size` to be removed in a future release of MySQL.

[transaction_read_only](#)

Command-Line Format	--transaction-read-only[={OFF ON}]
System Variable	transaction_read_only
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	OFF

The transaction access mode. The value can be `OFF` (read/write; the default) or `ON` (read only).

The transaction access mode has three scopes: global, session, and next transaction. This three-scope implementation leads to some nonstandard access-mode assignment semantics, as described later.

To set the global transaction access mode at startup, use the [--transaction-read-only](#) server option.

At runtime, the access mode can be set directly using the [SET](#) statement to assign a value to the [transaction_read_only](#) system variable, or indirectly using the [SET TRANSACTION](#) statement. For example, use this [SET](#) statement to set the global value:

```
SET GLOBAL transaction_read_only = ON;
```

Setting the global [transaction_read_only](#) value sets the access mode for all subsequent sessions. Existing sessions are unaffected.

To set the session or next-level [transaction_read_only](#) value, use the [SET](#) statement. For most session system variables, these statements are equivalent ways to set the value:

```
SET @@SESSION.var_name = value;
SET SESSION var_name = value;
SET var_name = value;
SET @@var_name = value;
```

As mentioned previously, the transaction access mode has a next-transaction scope, in addition to the global and session scopes. To enable the next-transaction scope to be set, [SET](#) syntax for assigning session system variable values has nonstandard semantics for [transaction_read_only](#),

To set the session access mode, use any of these syntaxes:

```
SET @@SESSION.transaction_read_only = value;
SET SESSION transaction_read_only = value;
SET transaction_read_only = value;
```

For each of those syntaxes, these semantics apply:

- Sets the access mode for all subsequent transactions performed within the session.
- Permitted within transactions, but does not affect the current ongoing transaction.
- If executed between transactions, overrides any preceding statement that sets the next-transaction access mode.
- Corresponds to [SET SESSION TRANSACTION {READ WRITE | READ ONLY}](#) (with the `SESSION` keyword).

To set the next-transaction access mode, use this syntax:

```
SET @@transaction_read_only = value;
```

For that syntax, these semantics apply:

- Sets the access mode only for the next single transaction performed within the session.
- Subsequent transactions revert to the session access mode.
- Not permitted within transactions.
- Corresponds to [SET TRANSACTION {READ WRITE | READ ONLY}](#) (without the `SESSION` keyword).

For more information about [SET TRANSACTION](#) and its relationship to the [transaction_read_only](#) system variable, see [Section 15.3.7, “SET TRANSACTION Statement”](#).

[unique_checks](#)

System Variable	unique_checks
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	ON

If set to 1 (the default), uniqueness checks for secondary indexes in InnoDB tables are performed. If set to 0, storage engines are permitted to assume that duplicate keys are not present in input data. If you know for certain that your data does not contain uniqueness violations, you can set this to 0 to speed up large table imports to InnoDB.

Setting this variable to 0 does not *require* storage engines to ignore duplicate keys. An engine is still permitted to check for them and issue duplicate-key errors if it detects them.

[updatable_views_with_limit](#)

Command-Line Format	--updatable-views-with-limit[={OFF ON}]
System Variable	updatable_views_with_limit
Scope	Global, Session

Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	1

This variable controls whether updates to a view can be made when the view does not contain all columns of the primary key defined in the underlying table, if the update statement contains a `LIMIT` clause. (Such updates often are generated by GUI tools.) An update is an [UPDATE](#) or [DELETE](#) statement. Primary key here means a `PRIMARY KEY`, or a `UNIQUE` index in which no column can contain `NULL`.

The variable can have two values:

1 or YES: Issue a warning only (not an error message). This is the default value.

0 or NO: Prohibit the update.

[use_secondary_engine](#)

System Variable	use_secondary_engine
Scope	Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Enumeration
Default Value	ON
	OFF
Valid Values	ON
	FORCED

For future use.

Whether to execute queries using a secondary engine.

For use with HeatWave. See [MySQL HeatWave User Guide](#).

`validate_password.xxx`

The `validate_password` component implements a set of system variables having names of the form `validate_password.xxx`. These variables affect password testing by that component; see [Section 8.4.3.2, "Password Validation Options and Variables"](#).

[version](#)

The version number for the server. The value might also include a suffix indicating server build or configuration information. `-debug` indicates that the server was built with debugging support enabled.

[version_comment](#)

System Variable	version_comment
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The **CMake** configuration program has a [COMPILATION_COMMENT_SERVER](#) option that permits a comment to be specified when building MySQL. This variable contains the value of that comment.

[version_compile_machine](#)

System Variable	version_compile_machine
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The type of the server binary.

[version_compile_os](#)

System Variable	version_compile_os
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The type of operating system on which MySQL was built.

[version_compile_zlib](#)

System Variable	version_compile_zlib
Scope	Global
Dynamic	No
SET VAR Hint Applies	No
Type	String

The version of the compiled-in zlib library.

[wait_timeout](#)

Command-Line Format	--wait-timeout=#
System Variable	wait_timeout
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Integer
Default Value	28800
Minimum Value	1
Maximum Value (Windows)	2147483
Maximum Value (Other)	31536000
Unit	seconds

The number of seconds the server waits for activity on a noninteractive connection before closing it.

On thread startup, the session [wait_timeout](#) value is initialized from the global [wait_timeout](#) value or from the global [interactive_timeout](#) value, depending on the type of client (as defined by the `CLIENT_INTERACTIVE` connect option to [mysql_real_connect\(\)](#)). See also [interactive_timeout](#).

[warning_count](#)

The number of errors, warnings, and notes that resulted from the last statement that generated messages. This variable is read only. See [Section 15.7.7.41, “SHOW WARNINGS Statement”](#).

[windowing_use_high_precision](#)

Command-Line Format	--windowing-use-high-precision[={OFF ON}]
System Variable	windowing_use_high_precision
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	Yes
Type	Boolean
Default Value	ON

Whether to compute window operations without loss of precision. See [Section 10.2.1.21, “Window Function Optimization”](#).

[xa_detach_on_prepare](#)

Command-Line Format	--xa-detach-on-prepare[={OFF ON}]
System Variable	xa_detach_on_prepare
Scope	Global, Session
Dynamic	Yes
SET VAR Hint Applies	No
Type	Boolean
Default Value	ON

When set to `ON` (enabled), all XA transactions are detached (disconnected) from the connection (session) as part of [XA PREPARE](#). This means that the XA transaction can be committed or rolled back by another connection, even if the originating connection has not terminated, and this connection can start new transactions.

Temporary tables cannot be used inside detached XA transactions.

When this is `OFF` (disabled), an XA transaction is strictly associated with the same connection until the session disconnects. It is recommended that you allow it to be enabled (the default behavior) for replication.

For more information, see [Section 15.3.8.2, “XA Transaction States”](#).

[PREV](#) [HOME](#) [UP](#) [NEXT](#)

[Related Documentation](#)

[MySQL 8.4 Release Notes](#)

[Download this Manual](#)

[PDF \(US Ltr\)](#) - 39.9Mb

[PDF \(A4\)](#) - 40.0Mb

[Man Pages \(TGZ\)](#) - 258.3Kb

[Man Pages \(Zip\)](#) - 365.2Kb

[Info \(Gzip\)](#) - 4.0Mb

[Info \(Zip\)](#) - 4.0Mb