XtraDB/InnoDB Server System Variables

This page documents system variables related to the XtraDB/InnoDB storage engine. See Server System Variables for a complete list of system variables and instructions on them.

See also the Full list of MariaDB options, system and status variables.

have_innodb

- **Description:** If the server supports InnoDB tables, will be set to `YES`, otherwise will be set to `NO`. Removed in MariaDB 10.0, use the Information Schema PLUGINS t SHOW ENGINES instead.
- **Scope:** Global
- **Dynamic:** No
- **Removed:** MariaDB 10.0

---

ignore_builtin_innodb

- **Description:** In older versions of MariaDB, setting this to `1` resulted in the built-in InnoDB storage engine being ignored. In current versions of MariaDB, XtraDB, the performance-enhanced fork of InnoDB, is the default and is always present, so this variable is ignored and setting it results in a warning. From MariaDB 10.0.1 to Maria 10.0.8, when InnoDB was still the default instead of XtraDB, this variable needed to be set.
- **Commandline:** `--ignore-builtin-innodb`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`

---

innodb_adaptive_checkpoint

- **Description:** Replaced with innodb_adaptive_flushing_method. Controls adaptive checkpointing. InnoDB's fuzzy checkpointing can cause stalls, as many dirty blocks a flushed at once as the checkpoint age nears the maximum. Adaptive checkpointing aims for more consistent flushing, approximately `modified age / maximum chec age`. Can result in larger transaction log files
  - `reflex` Similar to innodb_max_dirty_pages_pct flushing but flushes blocks constantly and contiguously based on the oldest modified age. If the age exceeds 1/ maximum age capacity, flushing will be weak contiguous. If the age exceeds 3/4, flushing will be strong. Strength can be adjusted by the variable innodb_io_capa
  - `estimate` The default, and independent of innodb_io_capacity. If the oldest modified age exceeds 1/2 of the maximum age capacity, blocks will be flushed ever at a rate determined by the number of modified blocks, LSN progress speed and the average age of all modified blocks.
  - `keep_average` Attempts to keep the I/O rate constant by using a shorter loop cycle of one tenth of a second. Designed for SSD cards.
- **Commandline:** `--innodb-adaptive-checkpoint=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** `estimate`
- **Valid Values:** `none` or `0`, `reflex` or `1`, `estimate` or `2`, `keep_average` or `3`
- **Introduced:** XtraDB 5.1.54-12.5
- **Removed:** XtraDB 5.5 - replaced with innodb_adaptive_flushing_method

---

innodb_adaptive_flushing

- **Description:** If set to `1`, the default, the server will dynamically adjust the flush rate of dirty pages in the InnoDB buffer pool. This assists to reduce brief bursts of I/O a
- **Commandline:** `--innodb-adaptive-flushing=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`

---

innodb_adaptive_flushing_lwm

- **Description:** Adaptive flushing is enabled when this this low water mark percentage of the redo log capacity is reached.
- **Commandline:** `--innodb-adaptive-flushing-lwm=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `10`
- **Range:** `0` to `70`
- **Introduced:** MariaDB 10.0

---

innodb_adaptive_flushing_method

- **Description:** Determines the method of flushing dirty blocks from the InnoDB buffer pool. If set to `native` or `0`, the original InnoDB method is used. The maximum ch age is determined by the total length of all transaction log files. When the checkpoint age reaches the maximum checkpoint age, blocks are flushed. This can cause lag are many updates per second and many blocks with an almost identical age need to be flushed. If set to `estimate` or `1`, the default, the oldest modified age will be c with the maximum age capacity. If it's more than 1/4 of this age, blocks are flushed every second. The number of blocks flushed is determined by the number of modifie the LSN progress speed and the average age of all modified blocks. It's therefore independent of the innodb_io_capacity for the 1-second loop, but not entirely so for th second loop. If set to `keep_average` or `2`, designed specifically for SSD cards, a shorter loop cycle is used in an attempt to keep the I/O rate constant. Removed in M 10.0/XtraDB 5.6 and replaced with InnoDB flushing method from MySQL 5.6.
- **Commandline:** `innodb-adaptive-flushing-method=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `estimate`
- **Valid Values:** `native` or `0`, `estimate` or `1`, `keep_average` or `2`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced with InnoDB flushing method from MySQL 5.6

---

innodb_adaptive_hash_index

- **Description:** If set to `1`, the default, the InnoDB hash index is enabled.
- **Commandline:** `--innodb-adaptive-hash-index=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`

---

innodb_adaptive_hash_index_partitions

- **Description:** Specifies the number of partitions for use in adaptive searching. If set to `1`, no extra partitions are created. XtraDB-only. From MariaDB 10.2.6 (which use InnoDB as default instead of XtraDB), this is an alias for innodb_adaptive_hash_index_parts to allow for easier upgrades.
- **Commandline:** `innodb-adaptive-hash-index-partitions=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `1` to `64`
- **Introduced:** MariaDB 5.5.20

---

innodb_adaptive_hash_index_parts

- **Description:** Specifies the number of partitions for use in adaptive searching. If set to `1`, no extra partitions are created.
- **Commandline:** `innodb-adaptive-hash-index-parts=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `8`
- **Range:** `1` to `512`
- **Introduced:** MariaDB 10.2.2

---

innodb_adaptive_max_sleep_delay

- **Description:** Maximum time in microseconds to automatically adjust the innodb_thread_sleep_delay value to, based on the workload. Useful in extremely busy system hundreds of thousands of simultaneous connections.
- **Commandline:** `--innodb-adaptive-max-sleep-delay=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `150000`
- **Range:** `0` to `1000000`
- **Introduced:** MariaDB 10.0

---

innodb_additional_mem_pool_size

- **Description:** Size in bytes of the InnoDB memory pool used for storing information about internal data structures. Defaults to 8MB, if your application has many tables large structure, and this is exceeded, operating system memory will be allocated and warning messages written to the error log, in which case you should increase this Deprecated in MariaDB 10.0 and removed in MariaDB 10.2 along with InnoDB's internal memory allocator.
- **Commandline:** `--innodb-additional-mem-pool-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `8388608`

- **Range:** `2097152` to `4294967295`
- **Deprecated:** MariaDB 10.0
- **Removed:** MariaDB 10.2.2

---

innodb_api_bk_commit_interval

- **Description:** Time in seconds between auto-commits for idle connections using the InnoDB memcached interface (not implemented in MariaDB).
- **Commandline:** `--innodb-api-bk-commit-interval=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `5`
- **Range:** `1` to `1073741824`
- **Introduced:** MariaDB 10.0
- **Removed:** MariaDB 10.2.4

---

innodb_api_disable_rowlock

- **Description:** For use with MySQL's memcached (not implemented in MariaDB)
- **Commandline:** `--innodb-api-disable-rowlock=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0
- **Removed:** MariaDB 10.2.4

---

innodb_api_enable_binlog

- **Description:** For use with MySQL's memcached (not implemented in MariaDB)
- **Commandline:** `--innodb-api-enable-binlog=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0
- **Removed:** MariaDB 10.2.4

---

innodb_api_enable_mdl

- **Description:** For use with MySQL's memcached (not implemented in MariaDB)
- **Commandline:** `--innodb-api-enable-mdl=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0
- **Removed:** MariaDB 10.2.4

---

innodb_api_trx_level

- **Description:** For use with MySQL's memcached (not implemented in MariaDB)
- **Commandline:** `--innodb-api-trx-level=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Introduced:** MariaDB 10.0
- **Removed:** MariaDB 10.2.4

---

innodb_auto_lru_dump

- **Description:** Renamed innodb_buffer_pool_restore_at_startup since XtraDB 5.5.10-20.1, which was in turn replaced by innodb_buffer_pool_load_at_startup in MariaD
- **Commandline:** `--innodb-auto-lru-dump=#`
- **Removed:** XtraDB 5.5.10-20.1

---

innodb_autoextend_increment

- **Description:** Size in MB to increment an auto-extending shared tablespace file when it becomes full. If innodb_file_per_table was set to `1`, this setting does not apply resulting per-table tablespace files, which are automatically extended in their own way.
- **Commandline:** `--innodb-autoextend-increment=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `64` (from MariaDB 10.0) `8` (before MariaDB 10.0),
- **Range:** `1` to `1000`

---

innodb_autoinc_lock_mode

- **Description:** Locking mode used for generating auto-increment values. `0` is the traditional lock mode, `1` the consecutive, and `2` the interleaved. See AUTO_INCREI handling in XtraDB/InnoDB for more on the lock modes. In order to use Galera, the mode needs to be set to `2`.
- **Commandline:** `--innodb-autoinc-lock-mode=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:**
  - `2` (>= MariaDB 10.2.4)
  - `1` (<= MariaDB 10.2.3)
- **Range:** `0` to `2`

---

innodb_background_scrub_data_check_interval

- **Description:** Check if spaces needs scrubbing every innodb_background_scrub_data_check_interval seconds. See Data Scrubbing.
- **Commandline:** `--innodb-background-scrub-data-check-interval=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `3600`
- **Range:** `1` to `4294967295`
- **Introduced:** MariaDB 10.1.3

---

innodb_background_scrub_data_compressed

- **Description:** Enable scrubbing of compressed data by background threads (same as encryption_threads). See Data Scrubbing.
- **Commandline:** `--innodb-background-scrub-data-compressed={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `0`
- **Introduced:** MariaDB 10.1.3

---

innodb_background_scrub_data_interval

- **Description:** Scrub spaces that were last scrubbed longer than this number of seconds ago. See Data Scrubbing.
- **Commandline:** `--innodb-background-scrub-data-interval=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `604800`
- **Range:** `1` to `4294967295`
- **Introduced:** MariaDB 10.1.3

---

innodb_background_scrub_data_uncompressed

- **Description:** Enable scrubbing of uncompressed data by background threads (same as encryption_threads). See Data Scrubbing.
- **Commandline:** `--innodb-background-scrub-data-uncompressed={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `0`
- **Introduced:** MariaDB 10.1.3

---

innodb_blocking_buffer_pool_restore

- **Description:** If set to `1` (`0` is default), XtraDB will wait until the LRU dump is completely restored upon restart before reporting back to the server that it has successfu started up. Available with XtraDB only, not InnoDB.
- **Commandline:** `innodb-blocking-buffer-pool-restore={1|2}`

- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0.0

---

innodb_buf_dump_status_frequency

- **Description:** Determines how often (as a percent) the buffer pool dump status should be printed in the logs. For example, `10` means that the buffer pool dump status when every 10% of the number of buffer pool pages are dumped. The default is `0` (only start and end status is printed).
- **Commandline:** `--innodb-buf-dump-status-frequency=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `100`
- **Introduced:** MariaDB 10.1.6

---

innodb_buffer_pool_chunk_size

- **Description:** Chunk size used for dynamically resizing the buffer pool. Note that changing this setting can change the size of the buffer pool. When large-pages is usec value is effectively rounded up to the next multiple of large-page-size. See Setting Innodb Buffer Pool Size Dynamically
- **Commandline:** `--innodb-buffer-pool-chunk-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `134217728`
- **Range:** `1048576` to innodb_buffer_pool_size/innodb_buffer_pool_instances
- **Introduced:** MariaDB 10.2.2

---

innodb_buffer_pool_dump_at_shutdown

- **Description:** Whether to record pages cached in the buffer pool on server shutdown, which reduces the length of the warmup the next time the server starts. The relate innodb_buffer_pool_load_at_startup specifies whether the buffer pool is automatically warmed up at startup.
- **Commandline:** `--innodb-buffer-pool-dump-at-shutdown=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:**
  - `ON` (>= MariaDB 10.2.2)
  - `OFF` (<= MariaDB 10.2.1)
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_dump_now

- **Description:** Immediately records pages stored in the buffer pool. The related innodb_buffer_pool_load_now does the reverse, and will immediately warm up the buffe
- **Commandline:** `--innodb-buffer-pool-dump-now=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_dump_pct

- **Description:** Dump only the hottest N% of each buffer pool, defaults to 100 until MariaDB 10.2.1. Since MariaDB 10.2.2, defaults to 25% along with the changes to innodb_buffer_pool_dump_at_shutdown and innodb_buffer_pool_load_at_startup.
- **Commandline:** `--innodb-buffer-pool-dump-pct=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:**
  - `25` (>= MariaDB 10.2.2)
  - `100` (<= MariaDB 10.2.1)
- **Range:** `1` to `100`
- **Introduced:** MariaDB 10.1.10

---

innodb_buffer_pool_evict

- **Description:** Evict pages from the buffer pool. If set to "uncompressed" then all uncompressed pages are evicted from the buffer pool. Variable to be used only for testi exists in DEBUG builds.
- **Commandline:** `--innodb-buffer-pool-evict=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** `""`
- **Valid Values:** "" or "uncompressed"
- **Introduced:** MariaDB 5.5

---

innodb_buffer_pool_filename

- **Description:** The file that holds the buffer pool list of page numbers set by innodb_buffer_pool_dump_at_shutdown and innodb_buffer_pool_dump_now.
- **Commandline:** `--innodb-buffer-pool-filename=file`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** `ib_buffer_pool`
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_instances

- **Description:** If innodb_buffer_pool_size is set to more than 1GB, innodb_buffer_pool_instances divides the InnoDB buffer pool into this many instances. The default wa MariaDB 5.5, but for large systems with buffer pools of many gigabytes, many instances can help reduce contention concurrency. The default is 8 in MariaDB 10 (excep Windows 32-bit, where it varies according to innodb_buffer_pool_size, or from MariaDB 10.2.2, where it is set to 1 if innodb_buffer_pool_size < 1GB). Each instance ma its own data structures and takes an equal portion of the total buffer pool size, so for example if innodb_buffer_pool_size is 4GB and innodb_buffer_pool_instances is s each instance will be 1GB. Each instance should ideally be at least 1GB in size.
- **Commandline:** `--innodb-buffer-pool-instances=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** <= MariaDB 10.0.3: `1`
- **Default Value:** >= MariaDB 10.0.4: `8` , `1` (>= MariaDB 10.2.2 if innodb_buffer_pool_size < 1GB), or dependent on innodb_buffer_pool_size (Windows 32-bit)
- **Introduced:** MariaDB 5.5.20

---

innodb_buffer_pool_load_abort

- **Description:** Aborts the process of restoring buffer pool contents started by innodb_buffer_pool_load_at_startup or innodb_buffer_pool_load_now.
- **Commandline:** `--innodb-buffer-pool-load-abort=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_load_at_startup

- **Description:** Specifies whether the buffer pool is automatically warmed up when the server starts by loading the pages held earlier. The related innodb_buffer_pool_dump_at_shutdown specifies whether pages are saved at shutdown.
- **Commandline:** `--innodb-buffer-pool-load-at-startup=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:**
  - `ON` (>= MariaDB 10.2.2)
  - `OFF` (<= MariaDB 10.2.1)
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_load_now

- **Description:** Immediately warms up the buffer pool by loading the stored data pages. The related innodb_buffer_pool_dump_now does the reverse, and immediately re pages stored in the buffer pool.
- **Commandline:** `--innodb-buffer-pool-load-now=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0

---

innodb_buffer_pool_populate

- **Description:** When set to `1` (`0` is default), XtraDB will preallocate pages in the buffer pool on starting up so that NUMA allocation decisions are made while the buffer still clean. XtraDB only. This option was made ineffective in MariaDB 10.0.23. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as def instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-buffer-pool-populate={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.0.23

---

innodb_buffer_pool_restore_at_startup

- **Description:** Time in seconds between automatic buffer pool dumps. If set to a non-zero value, XtraDB will also perform an automatic restore of the buffer pool at start to `0` , automatic dumps are not performed, nor automatic restores on startup. Replaced by innodb_buffer_pool_load_at_startup in MariaDB 10.0.
- **Commandline:** `innodb-buffer-pool-restore-at-startup`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range - 32 bit:** `0` to `4294967295`
- **Range - 64 bit:** `0` to `18446744073709547520`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced by innodb_buffer_pool_load_at_startup

---

innodb_buffer_pool_shm_checksum

- **Description:** Used with Percona's SHM buffer pool patch in XtraDB 5.5. Was shortly deprecated in XtraDB XtraDB 5.5.13-20.4, and removed in XtraDB 5.6. XtraDB on
- **Commandline:** `innodb-buffer-pool-shm-checksum={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 5.5
- **Removed:** MariaDB 10.0

---

innodb_buffer_pool_shm_key

- **Description:** Used with Percona's SHM buffer pool patch in XtraDB 5.5. Later deprecated in XtraDB 5.5, and removed in XtraDB 5.6.
- **Commandline:** `innodb-buffer-pool-shm-key={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `0`
- **Introduced:** XtraDB 5.5
- **Deprecated:** XtraDB 5.5.13-20.4
- **Removed:** MariaDB 10.0/XtraDB 5.6

---

innodb_buffer_pool_size

- **Description:** InnoDB buffer pool size in bytes. The primary value to adjust on a database server with entirely/primarily XtraDB/InnoDB tables, can be set up to 80% of t memory in these environments. If set to 2 GB or more, you will probably want to adjust innodb_buffer_pool_instances as well. See the XtraDB/InnoDB Buffer Pool for m setting this variable, and also Setting Innodb Buffer Pool Size Dynamically if doing so dynamically.
- **Commandline:** `--innodb-buffer-pool-size=#`
- **Scope:** Global
- **Dynamic:** Yes (>= MariaDB 10.2.2), No (<= MariaDB 10.2.1)
- **Data Type:** `numeric`
- **Default Value:** `134217728` (128MB)
- **Range:** `5242880` (5MB) to `9223372036854775807` (8192PB)

---

innodb_change_buffer_max_size

- **Description:** Maximum size of the XtraDB/InnoDB Change Buffer as a percentage of the total buffer pool. The default is 25%, and this can be increased up to 50% for with high write activity, and lowered down to 0 for servers used exclusively for reporting.
- **Commandline:** `--innodb-change-buffer-max-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `25`
- **Range:** `0` to `50`

- **Introduced:** MariaDB 10.0

---

innodb_change_buffering

- **Description:** Sets how InnoDB change buffering is performed. See XtraDB/InnoDB Change Buffering for details on the settings.
- **Commandline:** `--innodb-change-buffering=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value - < MariaDB 5.5:** `inserts`
- **Default Value - >= MariaDB 5.5:** `all`
- **Valid Values - <= MariaDB 5.5:** `inserts` , `none`
- **Valid Values - >= MariaDB 5.5:** `inserts` , `none` , `deletes` , `purges` , `changes` , `all`

---

innodb_change_buffering_debug

- **Description:** If set to `1` , an XtraDB/InnoDB Change Buffering debug flag is set. `1` forces all changes to the change buffer, while `2` causes a crash at merge. `0` , the d indicates no flag is set. Only available in debug builds.
- **Commandline:** `--innodb-change-buffering-debug=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `2`

---

innodb_checkpoint_age_target

- **Description:** The maximum value of the checkpoint age. If set to `0` , has no effect. Removed in MariaDB 10.0/XtraDB 5.6 and replaced with InnoDB flushing method fr MySQL 5.6.
- **Commandline:** `innodb-checkpoint-age-target=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` upwards
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced with InnoDB flushing method from MySQL 5.6.

---

innodb_checksum_algorithm

- **Description:** A replacement for InnoDB's innodb_checksums and XtraDB's innodb_fast_checksum, specifies how the InnoDB tablespace checksum is generated and v
  - `innodb` : Backwards compatible with earlier versions.
  - `crc32` : A newer, faster algorithm, but incompatible with earlier versions. Tablespace blocks will be converted to the new format over time, meaning that a mix of checksums may be present.
  - `none` : Writes a constant rather than calculate a checksum.
  - `strict_*` : The `strict_*` options are the same as the regular options, but InnoDB will halt if it comes across a mix of checksum values. These are faster, as t and old checksum values are not required, but can only be used when setting up tablespaces for the first time.
- **Commandline:** `--innodb-checksum-algorithm=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:**
  - `crc32` (>= MariaDB 10.2.2)
  - `innodb` (<= MariaDB 10.2.1)
- **Valid Values:** `innodb` , `crc32` , `none` , `strict_innodb` , `strict_crc32` , `strict_none`
- **Introduced:** MariaDB 10.0

---

innodb_checksums

- **Description:** By default, InnoDB performs checksum validation on all pages read from disk, which provides extra fault tolerance. You would usually want this set to `1` i production environments, although setting it to `0` can provide marginal performance improvements. Deprecated and functionality replaced by innodb_checksum_algor MariaDB 10.0, and should be removed to avoid conflicts. `ON` is equivalent to `--innodb_checksum_algorithm=innodb` and `OFF` to `--innodb_checksum_algorithm=none` .
- **Commandline:** `--innodb-checksums` , `--skip-innodb-checksums`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Deprecated:** MariaDB 10.0

---

innodb_cleaner_lsn_age_factor

- **Description:** XtraDB has enhanced page cleaner heuristics, and with these in place, the default InnoDB adaptive flushing may be too aggressive. As a result, a new LS factor formula has been introduced, controlled by this variable. The default setting, `high_checkpoint`, uses the new formula, while the alternative, `legacy`, uses the algorithm. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-cleaner-lsn-age-factor=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:**
  - `deprecated` (>= MariaDB 10.2.6)
  - `high_checkpoint` (<= MariaDB 10.1)
- **Valid Values:**
  - `high_checkpoint`, `legacy` (<= MariaDB 10.1)
  - `deprecated`, `high_checkpoint`, `legacy` (>= MariaDB 10.2.6)
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_cmp_per_index_enabled

- **Description:** If set to `ON` (`OFF` is default), per-index compression statistics are stored in the INFORMATION_SCHEMA.INNODB_CMP_PER_INDEX table. These are expensive to record, so this setting should only be changed with care, such as for performance tuning on development or slave servers.
- **Commandline:** `--innodb-cmp-per-index-enabled=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0

---

innodb_commit_concurrency

- **Description:** Limit to the number of transaction threads that can can commit simultaneously. 0, the default, imposes no limit. While you can change from one positive li another at runtime, you cannot set this variable to 0, or change it from 0, while the server is running.
- **Commandline:** `--innodb-commit-concurrency=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `1000`

---

innodb_compression_algorithm

- **Description:** Compression algorithm used on page compression. Can only be set to supported values. Not all distributions support all these compression methods. Set alone is not sufficient to compress tables - see Choosing compression algorithm.
  - `none` : Default. Data is not compressed.
  - `zlib` : Pages are compressed with bundled zlib compression method.
  - `lz4` : Pages are compressed using https://code.google.com/p/lz4/ compression method.
  - `lzo` : Pages are compressed using http://www.oberhumer.com/opensource/lzo/ compression method.
  - `lzma` : Pages are compressed using http://tukaani.org/xz/ compression method.
  - `bzip2` : Pages are compressed using http://www.bzip.org/ compression method.
  - `snappy` : Pages are compressed using https://code.google.com/p/snappy/ (from MariaDB 10.1.3).
- **Commandline:** `--innodb-compression-algorithm=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:** `zlib` (>= MariaDB 10.2.4, MariaDB 10.1.22), `none` (<= MariaDB 10.2.3, MariaDB 10.1.21)
- **Valid Values:** `none`, `zlib`, `lz4`, `lzo`, `lzma`, `bzip2` or `snappy` (MariaDB 10.1.3)
- **Introduced:** MariaDB 10.1.0

---

innodb_compression_default

- **Description:** Whether or not compression is the default for new tables. By default `OFF`, which means new tables are not compressed.
- **Commandline:** `--innodb-compression-default={0|1}`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.2.3

---

innodb_compression_failure_threshold_pct

- **Description:** Specifies the percentage cutoff for expensive [compressed table, after which free space is added to each new compressed page, dynamically adjusted up level set by innodb_compression_pad_pct_max. Zero disables checking of compression efficiency and adjusting padding.
- **Commandline:** `--innodb-compression-failure-threshold-pct=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `5`
- **Range:** `0` to `100`
- **Introduced:** MariaDB 10.0

---

innodb_compression_level

- **Description:** The level of zlib compression, from `1` to `9`, used with InnoDB compressed tables and indexes. `1` gives the best speed, `9` the best compression. Defau Note that not all compression methods allow choosing the compression level and in those cases the compression level value is ignored. See InnoDB/XtraDB Page Compression.
- **Commandline:** `--innodb-compression-level=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `6`
- **Range:** `0` to `9`
- **Introduced:** MariaDB 10.0

---

innodb_compression_pad_pct_max

- **Description:** The maximum percentage of reserved free space within each compressed page. Reserved free space is used when the page's data is reorganized and m recompressed. Only used when innodb_compression_failure_threshold_pct is not zero, and the rate of compression failures exceeds its setting.
- **Commandline:** `--innodb-compression-pad-pct-max=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `50`
- **Range:** `0` to `75`
- **Introduced:** MariaDB 10.0

---

innodb_concurrency_tickets

- **Description:** Number of times a newly-entered thread can enter and leave InnoDB until it is again subject to the limitations of innodb_thread_concurrency and may pos queued.
- **Commandline:** `--innodb-concurrency-tickets=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `5000` (from MariaDB 10.0), `500` (before mariaDB 10,0)

---

innodb_corrupt_table_action

- **Description:** When set to `assert`, the default, XtraDB will intentionally crash the server when it detects corrupted data in a single-table tablespace, with an assertion When set to `warn`, it will pass corruption as corrupt table instead of crashing, and disable all further I/O (except for deletion) on the table file. If set to `salvage`, read a permitted, but corrupted pages are ignored. innodb_file_per_table must be enabled for this option. Previously named `innodb_pass_corrupt_table`. XtraDB only. Ad deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-corrupt-table-action=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:**
  - `assert` (<= MariaDB 10.1)
  - `deprecated` (<= MariaDB 10.2.6)
- **Valid Values:**
  - `deprecated`, `assert`, `warn`, `salvage` (>= MariaDB 10.2.6)
  - `assert`, `warn`, `salvage` (<= MariaDB 10.1)
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2.6

---

innodb_data_file_path

- **Description:** Individual InnoDB data files, paths and sizes. The value of innodb_data_home_dir is joined to each path specified by innodb_data_file_path to get the full path. If innodb_data_home_dir is an empty string, absolute paths can be specified here. A file size is specified with K for kilobytes, M for megabytes and G for gigabyte whether or not to autoextend the data file is also specified.
- **Commandline:** `--innodb-data-file-path=name`
- **Scope:** Global

- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `ibdata1:12M:autoextend` (from MariaDB 10.0), `ibdata1:10M:autoextend` (before MariaDB 10.0)

---

innodb_data_home_dir

- **Description:** Directory path for all InnoDB data files in the shared tablespace (assuming innodb_file_per_table is not enabled). File-specific information can be added i innodb_data_file_path, as well as absolute paths if innodb_data_home_dir is set to an empty string.
- **Commandline:** `--innodb-data-home-dir=path`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `directory name`
- **Default Value:** `The MariaDB data directory`

---

innodb_deadlock_detect

- **Description:** By default, the InnoDB deadlock detector is enabled. If set to off, deadlock detection is disabled and MariaDB will rely on innodb_lock_wait_timeout instea may be more efficient in systems with high concurrency as deadlock detection can cause a bottleneck when a number of threads have to wait for the same lock.
- **Commandline:** `--innodb-deadlock-detect`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `1`
- **Introduced:** MariaDB 10.2.6

---

innodb_default_page_encryption_key

- **Description:** Encryption key used for page encryption. See Table and Tablespace Encryption.
- **Commandline:** `--innodb-default-page-encryption-key=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `1` to `255`
- **Introduced:** MariaDB 10.1.3
- **Removed:** MariaDB 10.1.4

---

innodb_default_encryption_key_id

- **Description:** Default encryption key id used for table encryption, See Table and Tablespace Encryption.
- **Commandline:** `--innodb-default-encryption-key-id=#`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `1` to `4294967295`
- **Introduced:** MariaDB 10.1.4

---

innodb_default_row_format

- **Description:** Specifies the default row format to be used for InnoDB tables. The compressed row format cannot be set as the default.
- **Commandline:** `--innodb-default-row-format=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:** `dynamic`
- **Valid Values:** `redundant`, `compact` or `dynamic`
- **Introduced:** MariaDB 10.2.2, MariadB 10.1.32

---

innodb_defragment

- **Description:** When set to `1` (the default is `0`), InnoDB defragmentation is enabled. When set to FALSE, all existing defragmentation will be paused and new defragm commands will fail. Paused defragmentation commands will resume when this variable is set to true again. See Defragmenting InnoDB Tablespaces.
- **Commandline:** `--innodb-defragment=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.1

innodb_defragment_fill_factor

- **Description:**. Indicates how full defragmentation should fill a page. Together with innodb_defragment_fill_factor_n_recs ensures defragmentation won't pack the page t and cause page split on the next insert on every page. The variable indicating more defragmentation gain is the one effective. See Defragmenting InnoDB Tablespaces
- **Commandline:** `--innodb-defragment-fill-factor=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `double`
- **Default Value:** `0.9`
- **Range:** `0.7` to `1`
- **Introduced:** MariaDB 10.1.1

innodb_defragment_fill_factor_n_recs

- **Description:** Number of records of space that defragmentation should leave on the page. This variable, together with innodb_defragment_fill_factor, is introduced so defragmentation won't pack the page too full and cause page split on the next insert on every page. The variable indicating more defragmentation gain is the one effect Defragmenting InnoDB Tablespaces.
- **Commandline:** `--innodb-defragment-fill-factor-n-recs=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `20`
- **Range:** `1` to `100`
- **Introduced:** MariaDB 10.1.1

innodb_defragment_frequency

- **Description:** Maximum times per second for defragmenting a single index. This controls the number of times the defragmentation thread can request X_LOCK on an ir defragmentation thread will check whether 1/defragment_frequency (s) has passed since it last worked on this index, and put the index back in the queue if not enough passed. The actual frequency can only be lower than this given number. See Defragmenting InnoDB Tablespaces.
- **Commandline:** `--innodb-defragment-frequency=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `integer`
- **Default Value:** `40`
- **Range:** `1` to `1000`
- **Introduced:** MariaDB 10.1.1

innodb_defragment_n_pages

- **Description:** Number of pages considered at once when merging multiple pages to defragment. See Defragmenting InnoDB Tablespaces.
- **Commandline:** `--innodb-defragment-n-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `7`
- **Range:** `2` to `32`
- **Introduced:** MariaDB 10.1.1

innodb_defragment_stats_accuracy

- **Description:** Number of defragment stats changes there are before the stats are written to persistent storage. Defaults to zero, meaning disable defragment stats track Defragmenting InnoDB Tablespaces.
- **Commandline:** `--innodb-defragment-stats-accuracy=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `4294967295`
- **Introduced:** MariaDB 10.1.1

innodb_dict_size_limit

- **Description:** Size in bytes of a soft limit the memory used by tables in the data dictionary. Once this limit is reached, XtraDB will attempt to remove unused entries. If s the default and standard InnoDB behavior, there is no limit to memory usage. Removed in MariaDB 10.0/XtraDB 5.6 and replaced by MySQL 5.6's new table_definition implementation.
- **Commandline:** `innodb-dict-size-limit=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`

- **Default Value:** `0`
- **Default Value - 32 bit:** `2147483648`
- **Default Value - 64 bit:** `9223372036854775807`
- **Introduced:** XtraDB 5.0.77-b13
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced by MySQL 5.6's new table_definition_cache implementation.

---

innodb_disable_sort_file_cache

- **Description:** If set to `1` (`0` is default), the operating system file system cache for merge-sort temporary files is disabled.
- **Commandline:** `--innodb-disable-sort-file-cache=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.0

---

innodb_disallow_writes

- **Description:** Tell InnoDB to stop any writes to disk.
- **Commandline:** None
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.3

---

innodb_doublewrite

- **Description:** If set to `1`, the default, to improve fault tolerance InnoDB first stores data to a doublewrite buffer before writing it to data file. Disabling will provide a marg peformance improvement.
- **Commandline:** `--innodb-doublewrite`, `--skip-innodb-doublewrite`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON`

---

innodb_doublewrite_file

- **Description:** The absolute or relative path and filename to a dedicated tablespace for the doublewrite buffer. In heavy workloads, the doublewrite buffer can impact hea the server, and moving it to a different drive will reduce contention on random reads. Since the doublewrite buffer is mostly sequential writes, a traditional HDD is a bett than SSD. This Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Commandline:** `innodb-doublewrite-file=filename`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `filename`
- **Default Value:** `NULL`
- **Introduced:** XtraDB 5.5.8-20.0
- **Removed:** MariaDB 10.0

---

innodb_empty_free_list_algorithm

- **Description:** XtraDB 5.6.13-61 introduced an algorithm to assist with reducing mutex contention when the buffer pool free list is empty, controlled by this variable. If set `backoff`, the default, the new algorithm will be used. If set to `legacy`, the original InnoDB algorithm will be used. XtraDB only. Added as a deprecated and ignored o MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades. See #1651657 for the reasons this was changed back to `legacy` in Xt 5.6.36-82.0.
- **Commandline:** `innodb-empty-free-list-algorithm=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:**
  - `deprecated` (>= MariaDB 10.2.6)
  - `legacy` (>= MariaDB 10.1.24)
  - `backoff` (<= MariaDB 10.1.23)
- **Valid Values:**
  - `deprecated`, `backoff`, `legacy` (>= MariaDB 10.2.6)
  - `backoff`, `legacy` (<= MariaDB 10.1)
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_enable_unsafe_group_commit

- **Description:** Unneeded after XtraDB 1.0.5. If set to `0`, the default, InnoDB will keep transactions between the transaction log and binary logs in the same order. Safer, slower. If set to `1`, transactions can be group-committed, but there is no guarantee of the order being kept, and a small risk of the two logs getting out of sync. In write-environments, can lead to a significant improvement in performance.
- **Commandline:** `--innodb-enable-unsafe-group-commit`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `1`
- **Removed:** Not needed after XtraDB 1.0.5

---

innodb_encrypt_log

- **Description:** Enable redo log encryption/decryption. See Data at Rest Encryption.
- **Commandline:** `--innodb-encrypt-log`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.3

---

innodb_encrypt_tables

- **Description:** Encrypt all tables in the storage engine. See Table and Tablespace Encryption. The `FORCE` option (from 10.1.4) means that XtraDB/InnoDB will refuse to unencrypted tables (CREATE TABLE ... ENCRYPTED=NO will fail).
- **Commandline:** `--innodb-encrypt-tables=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Valid Values:** `ON`, `OFF`, `FORCE` (from MariaDB 10.1.4)
- **Introduced:** MariaDB 10.1.3

---

innodb_encryption_rotate_key_age

- **Description:** Re-encrypt in background any page having a key older than this. When setting up Encryption, this variable must be set to a non-zero value. Otherwise, w enable encryption through `innodb_encrypt_tables` MariaDB won't be able to automatically encrypt any unencrypted tables.
- **Commandline:** `--innodb-encryption-rotate-key-age=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `0` to `4294967295`
- **Introduced:** MariaDB 10.1.3

---

innodb_encryption_rotation_iops

- **Description:** Use this many iops for background key rotation. See Table and Tablespace Encryption.
- **Commandline:** `--innodb-encryption-rotation_iops=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `100`
- **Range:** `0` to `4294967295`
- **Introduced:** MariaDB 10.1.3

---

innodb_encryption_threads

- **Description:** Number of threads performing background key rotation and scrubbing. When setting up Encryption, this variable must be set to a non-zero value. Otherw you enable encryption through `innodb_encrypt_tables` MariaDB won't be able to automatically encrypt any unencrypted tables.
- **Commandline:** `--innodb-encryption-threads=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `4294967295`
- **Introduced:** MariaDB 10.1.3

---

innodb_extra_rsegments

- **Description:** Removed in XtraDB 5.5 and replaced by innodb_rollback_segments. Usually there is one rollback segment protected by single mutex, a source of conten high write environments. This option specifies a number of extra user rollback segments. Changing the default will make the data readable by XtraDB only, and is incon with InnoDB. After modifying, the server must be slow-shutdown. If there is existing data, it must be dumped before changing, and re-imported after the change has tak
- **Commandline:** `--innodb-extra-rsegments=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `126`
- **Introduced:** XtraDB 5.1
- **Removed:** XtraDB 5.5 - replaced by innodb_rollback_segments

---

innodb_extra_undoslots

- **Description:** Usually, InnoDB has 1024 undo slots in its rollback segment, so 1024 transactions can run in parallel. New transactions will fail if all slots are used. Settin variable to `1` expands the available undo slots to 4072. Not recommended unless you get the `Warning: cannot find a free slot for an undo log` error in th log, as it makes data files unusable for ibbackup, or MariaDB servers not run with this option. See also undo log.
- **Commandline:** `--innodb-extra-undoslots=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** XtraDB 5.1

---

innodb_fake_changes

- **Description:** Enables the fake changes feature. In replication, setting up or restarting a slave can cause a replication reads to perform more slowly, as MariaDB is sing threaded and needs to read the data before it can execute the queries. This can be speeded up by prefetching threads to warm the server, replaying the statements an rolling back at commit. This however has a overhead from locking rows only then to undo changes at rollback. Fake changes attempts to reduce this overhead by readi rows for INSERT, UPDATE and DELETE statements but not updating them. The rollback is then very fast with little or nothing to do. XtraDB only. Added as a deprecate ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-fake-changes={0|1}`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2.6

---

innodb_fast_checksum

- **Description:** Implements a more CPU efficient XtraDB checksum algorithm, useful for write-heavy loads with high I/O. If set to `1` on a server with tables that have bee with it set to `0`, reads will be slower, so tables should be recreated (dumped and reloaded). XtraDB will fail to start if set to `0` and there are tables created while set to Replaced with innodb_checksum_algorithm in MariaDB 10.0/XtraDB 5.6.
- **Commandline:** `--innodb-fast-checksum={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Deprecated:** XtraDB 5.5.28-29.2
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced with innodb_checksum_algorithm

---

innodb_fast_shutdown

- **Description:** The shutdown mode. If set to `1`, the default, InnoDB performs a fast shutdown, not performing a full purge or an insert buffer merge. If set to `0`, InnoDB a slow shutdown, including full purge and insert buffer merge. If set to `2`, the logs are flushed and a cold shutdown takes place, similar to a crash. The resulting startup performs crash recovery. Mode 0 can be very slow, even taking hours in extreme cases, but is necessary if upgrading to a new major release before MariaDB 10.2.5 (s MDEV-12289). Mode 2 is extremely fast, in cases of emergency, but risks corruption.
- **Commandline:** `--innodb-fast-shutdown[=#]`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `0` to `2`

---

innodb_fatal_semaphore_wait_threshold

- **Description:** Maximum number of seconds that semaphore times out in InnoDB.
- **Commandline:** `--innodb-fatal-semaphore-wait-threshold=#`

- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `600`
- **Range:** `1` to `4294967295`
- **Introduced:** MariaDB 10.1.2
- **Removed:** MariaDB 10.3.1

---

innodb_file_format

- **Description:** File format for new InnoDB tables. Can either be `Antelope`, the default and the original format, or `Barracuda`, which supports compression. Note that t is also used when a table is re-created with an ALTER TABLE which requires a table copy. See XtraDB/InnoDB File Format for more on the file formats.
- **Commandline:** `--innodb-file-format=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:**
  - `Barracuda` (>= MariaDB 10.2.2)
  - `Antelope` (<= MariaDB 10.2.1)
- **Valid Values:** `Antelope`, `Barracuda`
- **Deprecated:** MariaDB 10.2
- **Removed:** MariaDB 10.3.1

---

innodb_file_format_check

- **Description:** If set to `1`, the default, InnoDB checks the shared tablespace file format tag. If this is higher than the current version supported by XtraDB/InnoDB (for ex Barracuda when only Antelope is supported), XtraDB/InnoDB will will not start. If it the value is not higher, XtraDB/InnoDB starts correctly and the innodb_file_format_m is set to this value. If innodb_file_format_check is set to `0`, no checking is performed. See XtraDB/InnoDB File Format for more on the file formats.
- **Commandline:** `--innodb-file-format-check=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean` (>= MariaDB 5.5)
- **Default Value:** `ON` (>= MariaDB 5.5)
- **Deprecated:** MariaDB 10.2
- **Removed:** MariaDB 10.3.1

---

innodb_file_format_max

- **Description:** The highest XtraDB/InnoDB file format. This is set to the value of the file format tag in the shared tablespace on startup (see innodb_file_format_check. If server later creates a higher table format, innodb_file_format_max is set to that value. See XtraDB/InnoDB File Format for more on the file formats.
- **Commandline:** `--innodb-file-format-max=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** `Antelope`
- **Valid Values:** `Antelope`, `Barracuda`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2
- **Removed:** MariaDB 10.3.1

---

innodb_file_per_table

- **Description:** If set to `1`, new XtraDB/InnoDB tables are created with data and indexes stored in their own .ibd file. If set to `0`, the default, new tables are created in th tablespace. Compression is only available with per table storage. Note that this value is also used when a table is re-created with an ALTER TABLE which requires a ta
- **Commandline:** `--innodb-file-per-table`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON` (>= MariaDB 5.5), `OFF` (<= MariaDB 5.3)

---

innodb_fill_factor

- **Description:** Percentage of B-tree page filled during bulk insert (sorted index build). Used as a hint rather than an absolute value. Setting to `70`, for example, reserves the space on each B-tree page for the index to grow in future.
- **Commandline:** `--innodb-fill-factor=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value**: `100`
- **Range:** `10` to `100`
- **Introduced:** MariaDB 10.2.2

innodb_flush_log_at_timeout

- **Description:** Interval in seconds to write and flush the logs. Before MariaDB 10, this was fixed at one second, which is still the default, but this can now be changed. It's increased to reduce flushing and avoid impacting performance of binary log group commit.
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `0` to `2700`
- **Introduced:** MariaDB 10.0

innodb_flush_log_at_trx_commit

- **Description:** Set to `1`, along with sync_binlog=1 for the greatest level of fault tolerance. The value of innodb_use_global_flush_log_at_trx_commit determines whethe variable can be reset with a SET statement or not.
  - `1` The default, the log buffer is written to the log file and a flush to disk performed after each transaction. This is required for full ACID compliance.
  - `0` Nothing is done on commit; rather the log buffer write and flush are performed once a second. This gives better performance, but a server crash can erase the second of transactions.
  - `2` The log buffer is written after each commit, but flushing takes place once a second. Performance is slightly better, but a OS or power outage can cause the las second's transactions to be lost.
  - `3` (from MariaDB 10.0) Emulates MariaDB 5.5 group commit (3 syncs per group commit). See Binlog group commit and innodb_flush_log_at_trx_commit.
- **Commandline:** `--innodb-flush-log-at-trx-commit[=#]`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `1`
- **Valid Values:** `0`, `1`, `2` or `3` (from MariaDB 10.0)

innodb_flush_method

- **Description:** XtraDB/InnoDB flushing method. Windows always uses async_unbuffered and this variable then has no effect. On Unix, by default fsync() is used to flush logs. Adjusting this variable can give performance improvements, but behavior differs widely on different filesystems, and changing from the default has caused problem some situations, so test and benchmark carefully before adjusting.
  - `O_DSYNC` - O_DSYNC is used to open and flush logs, and fsync() to flush the data files.
  - `O_DIRECT` - O_DIRECT or directio(), is used to open data files, and fsync() to flush data and logs.
  - `fdatasync` - an old default value that follows the default behavior of using fsync(), but replaced with the unset default to avoid confusion between fdatasync() a fsync().
  - `O_DIRECT_NO_FSYNC` - introduced in MariaDB 10.0. Uses O_DIRECT during flushing I/O, but skips fsync() afterwards. Not suitable for XFS filesystems.
  - `ALL_O_DIRECT` - introduced in MariaDB 5.5 / Percona 5.5, and available with XtraDB only. Uses O_DIRECT for opening both data and logs and fsync() to flush not logs. Use with large InnoDB files only, otherwise may cause a performance degradation. Set innodb_log_block_size to 4096 on ext4 filesystems. This is the block size on ext4 and will avoid unaligned AIO/DIO warnings.
- **Commandline:** `--innodb-flush-method=name`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `enumeration`
- **Default Value:** Not set.
- **Valid Values:** `fdatasync`, `O_DSYNC`, `O_DIRECT`, `O_DIRECT_NO_FSYNC` (MariaDB 10.0), `ALL_O_DIRECT` (MariaDB 5.5, XtraDB only)

innodb_flush_neighbor_pages

- **Description:** Determines whether, when dirty pages are flushed to the data file, neighboring pages in the data file are flushed at the same time. If set to `none`, the feat disabled. If set to `area`, the default, the standard InnoDB behavior is used. For each page to be flushed, dirty neighboring pages are flushed too. If there's little head s delay, such as SSD or large enough write buffer, one of the other two options may be more efficient. If set to `cont`, for each page to be flushed, neighboring contiguou are flushed at the same time. Being contiguous, a sequential I/O is used, unlike the random I/O used in `area`. Replaced by innodb_flush_neighbors in MariaDB 10.0/X 5.6.
- **Commandline:** `innodb-flush-neighbor-pages=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `area`
- **Valid Values:** `none` or `0`, `area` or `1`, `cont` or `2`
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced by innodb_flush_neighbors

innodb_flush_neighbors

- **Description:** Determines whether flushing a page from the buffer pool will flush other dirty pages in the same group of pages (extent). In high write environments, if flus not aggressive enough, it can fall behind resulting in higher memory usage, or if flushing is too aggressive, cause excess I/O activity. SSD devices, with low seek times, be less likely to require dirty neighbor flushing to be set.
  - `1` : The default, flushes contiguous dirty pages in the same extent from the buffer pool.
  - `0` : No other dirty pages are flushed.

    - `2` : Flushes dirty pages in the same extent from the buffer pool.
- **Commandline:** `--innodb-flush-neighbors=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `1`
- **Valid Values:** `0` , `1` , `2`
- **Introduced:** MariaDB 10.0.4

---

innodb_flush_sync

- **Description:** If set to `ON` , the default, the innodb_io_capacity setting is ignored for I/O bursts occuring at checkpoints.
- **Commandline:** `--innodb-flush-sync={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value**: `ON`
- **Introduced:** MariaDB 10.2.2

---

innodb_flushing_avg_loops

- **Description:** Determines how quickly adaptive flushing will respond to changing workloads. The value is the number of iterations that a previously calculated flushing s snapshot is kept. Increasing the value smooths and slows the rate that the flushing operations change, while decreasing it causes flushing activity to spike quickly in res workload changes.
- **Commandline:** `--innodb-flushing-avg-loops=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `30`
- **Range:** `1` to `1000`
- **Introduced:** MariaDB 10.0.4

---

innodb_force_load_corrupted

- **Description:** Set to `0` by default, if set to `1` , XtraDB/InnoDB will be permitted to load tables marked as corrupt. Only use this to recover data you can't recover any oth or in troubleshooting. Always restore to `0` when the returning to regular use.
- **Commandline:** `--innodb-force-load-corrupted`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5

---

innodb_force_primary_key

- **Description:** If set to `1` (`0` is default) CREATE TABLEs without a primary or unique key where all keyparts are NOT NULL will not be accepted, and will return an erro
- **Commandline:** `--innodb-force-primary-key`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.0

---

innodb_force_recovery

- **Description:** XtraDB/InnoDB crash recovery mode. `0` is the default. The other modes are for recovery purposes only, and no data can be changed while another mod active. Some queries relying on indexes are also blocked. See XtraDB/InnoDB Recovery Modes for more on mode specifics.
- **Commandline:** `--innodb-force-recovery=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `enumeration`
- **Default Value:** `0`
- **Range:** `0` to `6`

---

innodb_foreground_preflush

- **Description:** Before XtraDB 5.6.13-61.0, if the checkpoint age is in the sync preflush zone while a thread is writing to log, it will try to advance the checkpoint by issuing list flush batch if this is not already being done. XtraDB has enhanced page cleaner tuning, and may already be performing furious flushing, resulting in the flush simply

unneeded mutex pressure. Instead, the thread now waits for the flushes to finish, and then has two options, controlled by this variable. XtraDB only. Added as a deprec ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.

- `exponential_backoff` - thread sleeps while it waits for the flush list flush to occur. The sleep time randomly progressively increases, periodically reset to avoic sleeps.
- `sync_preflush` - thread issues a flush list batch, and waits for it to complete. This is the same as is used when the page cleaner thread is not running.

- **Commandline:** `innodb-foreground-preflush=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:**
  - `deprecated` (>= MariaDB 10.2.6)
  - `exponential_backoff` (<= MariaDB 10.1)
- **Valid Values:**
  - `deprecated`, `exponential_backoff`, `sync_preflush` (>= MariaDB 10.2.6)
  - `exponential_backoff`, `sync_preflush` (<= MariaDB 10.1)
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_ft_aux_table

- **Description:** Diagnostic variable intended only to be set at runtime. It specifies the qualified name (for example `test/ft_innodb`) of an InnoDB table that has a FULI index, and after being set the INFORMATION_SCHEMA tables INNODB_FT_INDEX_TABLE, INNODB_FT_INDEX_CACHE, INNODB_FT_CONFIG, INNODB_FT_DE and INNODB_FT_BEING_DELETED will contain search index information for the specified table.
- **Commandline:** `--innodb-ft-aux-table=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Introduced:** MariaDB 10.0.0

---

innodb_ft_cache_size

- **Description:** Cache size available for a parsed document while creating an InnoDB FULLTEXT index.
- **Commandline:** `--innodb-ft-cache-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `8000000`
- **Introduced:** MariaDB 10.0.0

---

innodb_ft_enable_diag_print

- **Description:** If set to `1`, additional full-text search diagnostic output is enabled.
- **Commandline:** `--innodb-ft-enable-diag-print=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.0

---

innodb_ft_enable_stopword

- **Description:** If set to `1`, the default, a set of stopwords is associated with an InnoDB FULLTEXT index when it is created. The stopword list comes from the table set b session variable innodb_ft_user_stopword_table, if set, otherwise the global variable innodb_ft_server_stopword_table, if that is set, or the built-in list if neither variable
- **Commandline:** `--innodb-ft-enable-stopword=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 10.0.0

---

innodb_ft_max_token_size

- **Description:** Maximum length of words stored in an InnoDB FULLTEXT index. A larger limit will increase the size of the index, slowing down queries, but permit longer be searched for. In most normal situations, longer words are unlikely search terms.
- **Commandline:** `--innodb-ft-max-token-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `84`
- **Range:** `10` to `252`
- **Introduced:** MariaDB 10.0.0

innodb_ft_min_token_size

- **Description:** Minimum length of words stored in an InnoDB FULLTEXT index. A smaller limit will increase the size of the index, slowing down queries, but permit shorte to be searched for. For data stored in a Chinese, Japanese or Korean character set, a value of 1 should be specified to preserve functionality.
- **Commandline:** `--innodb-ft-min-token-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `3`
- **Range:** `0` to `16`
- **Introduced:** MariaDB 10.0.0

innodb_ft_num_word_optimize

- **Description:** Number of words processed during each OPTIMIZE TABLE on an InnoDB FULLTEXT index. To ensure all changes are incorporated, multiple OPTIMIZE statements could be run in case of a substantial change to the index.
- **Commandline:** `--innodb-ft-num-word-optimize=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `2000`
- **Introduced:** MariaDB 10.0.0

innodb_ft_result_cache_limit

- **Description:** Limit in bytes of the InnoDB FULLTEXT index query result cache per fulltext query. The latter stages of the full-text search are handled in memory, and lim prevents excess memory usage. If the limit is exceeded, the query returns an error.
- **Commandline:** `--innodb-ft-result-cache-limit=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `2000000000`
- **Range:** `10000` to `18446744073709551615`
- **Introduced:** MariaDB 10.0.9

innodb_ft_server_stopword_table

- **Description:** Table name containing a list of stopwords to ignore when creating an InnoDB FULLTEXT index, in the format db_name/table_name. The specified table m before this option is set, and must be an InnoDB table with a single column, a VARCHAR named VALUE. See also innodb_ft_enable_stopword.
- **Commandline:** `--innodb-ft-server-stopword-table=db_name/table_name`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** Empty
- **Introduced:** MariaDB 10.0.0

innodb_ft_sort_pll_degree

- **Description:** Number of parallel threads used when building an InnoDB FULLTEXT index. See also innodb_sort_buffer_size.
- **Commandline:** `--innodb-ft-sort-pll-degree=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `2`
- **Range:** `1` to `32`
- **Introduced:** MariaDB 10.0.0

innodb_ft_total_cache_size

- **Description:**Total memory allocated for the cache for all InnoDB FULLTEXT index tables. A force sync is triggered if this limit is exceeded.
- **Commandline:** `--innodb-ft-total-cache-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `640000000`
- **Range:** `32000000` to `1600000000`
- **Introduced:** MariaDB 10.0.9

innodb_ft_user_stopword_table

- **Description:** Table name containing a list of stopwords to ignore when creating an InnoDB FULLTEXT index, in the format db_name/table_name. The specified table m before this option is set, and must be an InnoDB table with a single column, a VARCHAR named VALUE. See also innodb_ft_enable_stopword.
- **Commandline:** `--innodb-ft-user-stopword-table=db_name/table_name`
- **Scope:** Session
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** Empty
- **Introduced:** MariaDB 10.0.0

---

innodb_ibuf_accel_rate

- **Description:** Allows the insert buffer activity to be adjusted. The following formula is used: [real activity] = [default activity] * (innodb_io_capacity/100) * (innodb_ibuf_accel_rate/100). As `innodb_ibuf_accel_rate` is increased from its default value of `100`, the lowest setting, insert buffer activity is increased. See also innodb_io_capacity. This Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Commandline:** `innodb-ibuf-accel-rate=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `100`
- **Range:** `100` to `999999999`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0

---

innodb_ibuf_active_contract

- **Description:** Specifies whether the insert buffer can be processed before it's full. If set to `0`, the standard InnoDB method is used, and the buffer is not processed unti set to `1`, the default, the insert buffer can be processed before it is full. This Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Commandline:** `innodb-ibuf-active-contract=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `0` to `1`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0

---

innodb_ibuf_max_size

- **Description:** Maximum size in bytes of the insert buffer. Defaults to half the size of the buffer pool so you may want to reduce if you have a very large buffer pool. If set the insert buffer is disabled, which will cause all secondary index updates to be performed synchronously, usually at a cost to performance. This Percona XtraDB variab not been ported to XtraDB 5.6.
- **Commandline:** `innodb-ibuf-max-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** 1/2 the size of the InnoDB buffer pool
- **Range:** `0` to 1/2 the size of the InnoDB buffer pool
- **Removed:** MariaDB 10.0

---

innodb_idle_flush_pct

- **Description:** Up to what percentage of dirty pages should be flushed when innodb finds it has spare resources to do so.
- **Commandline:** `--innodb-idle-flush-pct=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `100`
- **Range:** `0` to `100`
- **Introduced:** MariaDB 10.1.2

---

innodb_immediate_scrub_data_uncompressed

- **Description:** Enable scrubbing of data. See Data Scrubbing.
- **Commandline:** `--innodb-immediate-scrub-data-uncompressed=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.3

innodb_import_table_from_xtrabackup

- **Description:** If set to `1`, permits importing of .ibd files exported with the XtraBackup --export option. Previously named `innodb_expand_import`. Removed in MariaD 10.0/XtraDB 5.6 and replaced with MySQL 5.6's transportable tablespaces.
- **Commandline:** `innodb-import-table-from-xtrabackup=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `1`
- **Introduced:** MariaDB 5.5.20
- **Removed:** MariaDB 10.0

innodb_instrument_semaphores

- **Description:** Enable semaphore request instrumentation. This could have some effect on performance but allows better information on long semaphore wait problems.
- **Commandline:** `--innodb-instrument-semaphores={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.3
- **Deprecated:** MariaDB 10.2.5 (treated as if `OFF`)
- **Removed:** MariaDB 10.3.0

innodb_io_capacity

- **Description:** Limit on I/O activity for XtraDB/InnoDB background tasks, including merging data from the insert buffer and flushing pages. Should be set to around the n I/O operations per second that system can handle, based on the type of drive/s being used. You can also set it higher when the server starts to help with the extra work that time, and then reduce for normal use. Ideally, opt for a lower setting, as at higher value data is removed from the buffers too quickly, reducing the effectiveness of c See also innodb_flush_sync.
- **Commandline:** `--innodb-io-capacity=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `200`
- **Range:** `100` to `18446744073709551615` (2^{64}-1)

innodb_io_capacity_max

- **Description:** Upper limit to which InnodDB can extend innodb_io_capacity in case of emergency. Only applicable if no value was specified for innodb_io_capacity whe server started up.
- **Commandline:** `--innodb-io-capacity-max=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `200`
- **Range :** `100` to `18446744073709551615` (2^{64}-1)
- **Introduced:** MariaDB 10.0

innodb_kill_idle_transaction

- **Description:** Time in seconds before killing an idle XtraDB transaction. If set to `0` (the default), the feature is disabled. Used to prevent accidental user locks. XtraDB c Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `to`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2.6

innodb_large_prefix

- **Description:** If set to `1`, tables that use dynamic and compressed row formats (which require innodb_file_format to be barracuda and innodb_file_per_table to be true permitted to have index key prefixes up to 3072 bytes. If not set, the limit is 767 bytes.
- **Commandline:** `--innodb-large-prefix`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`

- **Default Value:**
  - `ON` (>= MariaDB 10.2.2)
  - `OFF` (<= MariaDB 10.2.1)
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2
- **Removed:** MariaDB 10.3.1

---

innodb_lazy_drop_table

- **Description:** Deprecated and removed in XtraDB 5.6. DROP TABLE processing can take a long time when innodb_file_per_table is set to 1 and there's a large buffer p `innodb_lazy_drop_table` is set to `1` (`0` is default), XtraDB attempts to optimize DROP TABLE processing by deferring the dropping of related pages from the buff until there is time, only initially marking them.
- **Commandline:** `innodb-lazy-drop-table={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `0`
- **Introduced:** XtraDB 5.5.10-20.1
- **Deprecated:** XtraDB 5.5.30-30.2
- **Removed:** MariaDB 10.0.0

---

innodb_lock_schedule_algorithm

- **Description:** Specifies the algorithm that InnoDB/XtraDB uses to decide which of the waiting transactions should be granted the lock once it has been released. The po values are: `FCFS` (First-Come-First-Served) where locks are granted in the order they appear in the lock queue and `VATS` (Variance-Aware-Transaction-Scheduling) v locks are granted based on the Eldest-Transaction-First heuristic. Note that `VATS` should not be used with Galera. From MariaDB 10.1.30, InnoDB will refuse to start if used with Galera. From MariaDB 10.2, `VATS` is default, but from MariaDB 10.2.12, the value will be changed to `FCFS` and a warning produced when using Galera.
- **Commandline:** `--innodb-lock-schedule-algorithm=#`
- **Scope:** Global
- **Dynamic:** No (>= MariaDB 10.2.12, MariaDB 10.1.30), Yes (<= MariaDB 10.2.11, MariaDB 10.1.29)
- **Data Type:** `enum`
- **Valid Values:** `FCFS` , `VATS`
- **Default Value:** `VATS` (10.2), `FCFS` (10.1)
- **Introduced:** MariaDB 10.2.3, MariaDB 10.1.19

---

innodb_lock_wait_timeout

- **Description:** Time in seconds that an InnoDB transaction waits for an InnoDB row lock (not table lock) before giving up with the error `ERROR 1205 (HY000): Lock w timeout exceeded; try restarting transaction` . When this occurs, the statement (not transaction) is rolled back. The whole transaction can be rolled back if th innodb_rollback_on_timeout option is used. Increase this for data warehousing applications or where other long-running operations are common, or decrease for OLTP other highly interactive applications. This setting does not apply to deadlocks, which InnoDB detects immediately, rolling back a deadlocked transaction. `0` (from Maria 10.3.0) means no wait. See WAIT and NOWAIT.
- **Commandline:** `--innodb-lock-wait-timeout=#`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `50`
- **Range:**
  - `0` to `1073741824` (>= MariaDB 10.3)
  - `1` to `1073741824` (<= MariaDB 10.2)

---

innodb_locking_fake_changes

- **Description:** If set to `OFF` , fake transactions (see innodb_fake_changes) don't take row locks. This is an experimental feature to attempt to deal with drawbacks in fak changes blocking real locks. It is not safe for use in all environments. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB a instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-locking-fake-changes`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 5.5.29
- **Deprecated:** MariaDB 10.2.6

---

innodb_locks_unsafe_for_binlog

- **Description:** Set to `0` by default, in which case XtraDB/InnoDB uses gap locking. If set to `1` , gap locking is disabled for searches and index scans. Deprecated in Ma 10.0, use READ COMMITTED transaction isolation level instead.
- **Commandline:** `--innodb-locks-unsafe-for-binlog`
- **Scope:** Global
- **Dynamic:** No

- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Deprecated:** MariaDB 10.0

---

innodb_log_arch_dir

- **Description:** The directory for XtraDB log archiving. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-log-arch-dir=name`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `string`
- **Default Value:** `./`
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_log_arch_expire_sec

- **Description:** Time in seconds since the last change after which the archive log should be deleted. XtraDB only. Added as a deprecated and ignored option in MariaDB (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-log-arch-expire-sec=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_log_archive

- **Description:** Whether or not XtraDB log archiving is enabled. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as defau of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-log-archive=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_log_block_size

- **Description:** Size in bytes of the transaction log records. Generally `512`, the default, or `4096`, are the only two useful values. If the server is restarted and this value i changed, all old log files need to be removed. Should be set to `4096` for SSD cards or if innodb_flush_method is set to `ALL_O_DIRECT` on ext4 filesystems. XtraDB o Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-log-block-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `512`
- **Introduced:** MariaDB 5.1.55
- **Deprecated:** MariaDB 10.2.6

---

innodb_log_buffer_size

- **Description:** Size in bytes of the buffer for writing XtraDB/InnoDB log files to disk. Increasing this means larger transactions can run without needing to perform disk I/O committing.
- **Commandline:** `--innodb-log-buffer-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `16777216` (16MB) >= MariaDB 10.1.9, `8388608` (8MB) <= MariaDB 10.1.8
- **Range:** `262144` to `4294967295` (256KB to 4096MB)

---

innodb_log_checksum_algorithm

- **Description:** Experimental feature (as of MariaDB 10.0.9), this variable specifies how to generate and verify log checksums. XtraDB only. Added as a deprecated and i option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
  - `none` - No checksum. A constant value is instead written to logs, and no checksum validation is performed.

    - `innodb` - The default, and the original InnoDB algorithm. This is inefficient, but compatible with all MySQL, MariaDB and Percona versions that don't support oth checksum algorithms.
    - `crc32` - CRC32© is used for log block checksums, which also permits recent CPUs to use hardware acceleration (on SSE4.2 x86 machines and Power8 or late checksums.
    - `strict_*` - Whether or not to accept checksums from other algorithms. If strict mode is used, checksums blocks will be considered corrupt if they don't match t specified algorithm. Normally they are considered corrupt only if no other algorithm matches.
- **Commandline:** `innodb-log-checksum-algorithm=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enum`
- **Default Value:**
    - `deprecated` (>= MariaDB 10.2.6)
    - `innodb` (<= MariaDB 10.1)
- **Valid Values:**
    - `deprecated`, `innodb`, `none`, `crc32`, `strict_none`, `strict_innodb`, `strict_crc32` (>= MariaDB 10.2.6)
    - `innodb`, `none`, `crc32`, `strict_none`, `strict_innodb`, `strict_crc32` (<= MariaDB 10.1)
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_log_checksums

- **Description:** If set to `1`, CRC32C for Innodb or `innodb_log_checksum_algorithm` for XtraDB algorithm is used for redo log pages. If disabled, the checksum field are ignored.
- **Commandline:** `innodb-log-checksums={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 10.2.2

---

innodb_log_compressed_pages

- **Description:** Whether or not images of recompressed pages are stored in the InnoDB redo logs.
- **Commandline:** `--innodb-log-compressed-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:**
    - `ON` (>= MariaDB 10.2.4, >= MariaDB 10.1.26, <= MariaDB 10.1.1)
    - `OFF` (MariaDB 10.2.0 - MariaDB 10.2.3, MariaDB 10.1.2 - MariaDB 10.1.25)
- **Introduced:** MariaDB 10.0.9

---

innodb_log_file_size

- **Description:** Size in bytes of each log file in the log group. The combined size can be no more than 4GB prior to MariaDB 10.0, and no more than 512GB in MariaDB 1 later. Larger values mean less disk I/O due to less flushing checkpoint activity, but also slower recovery from a crash.
- **Commandline:** `--innodb-log-file-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `50331648` (48MB) (from MariaDB 10.0), `5242880` (5MB) (before MariaDB 10.0)
- **Range:** `1048576` to `512GB` (1MB to 512GB) (>= MariaDB 10.0), `1048576` to `4294967295` (1MB to 4096MB) (<= MariaDB 5.5),

---

innodb_log_files_in_group

- **Description:** Number of physical files in the InnoDB redo log.
- **Commandline:** `--innodb-log-files-in-group=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `2`
- **Range:** `1` to `100` (>= MariaDB 10.2.4), `2` to `100` (<= MariaDB 10.2.3)

---

innodb_log_group_home_dir

- **Description:** Path to the XtraDB/InnoDB redo log files. If none is specified, innodb_log_files_in_group files named ib_logfile0 and so on, with a size of innodb_log_file_ created in the data directory.
- **Commandline:** `--innodb-log-group-home-dir=path`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `directory name`

innodb_log_write_ahead_size

- **Description:** Redo log write ahead unit size to avoid read-on-write. Should match the OS cache block IO size.
- **Commandline:** `--innodb-log-write-ahead-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `8192`
- **Range:** `512` to innodb_page_size
- **Introduced:** MariaDB 10.2.2

innodb_lru_scan_depth

- **Description:** Specifies how far down the buffer pool LRU list the cleaning thread should look for dirty pages to flush. This process is performed once a second. In an I/O intensive-workload, can be increased if there is spare I/O capacity, or decreased if in a write-intensive workload with little spare I/O capacity.
- **Commandline:** `--innodb-lru-scan-depth=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1024`
- **Range - 32bit:** `100` to $2^{32}-1$
- **Range - 64bit:** `100` to $2^{64}-1$
- **Introduced:** XtraDB 5.1.66-14.2

innodb_max_bitmap_file_size

- **Description:** Limit in bytes of the changed page bitmap files. For faster incremental backup with Xtrabackup, XtraDB tracks pages with changes written to them accord redo log and writes the information to special changed page bitmap files. These files are rotated when the server restarts or when this limit is reached. XtraDB only. See innodb_track_changed_pages and innodb_max_changed_pages. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-max-bitmap-file-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `4096` (4KB)
- **Range:** `4096` (4KB) to `18446744073709551615` (16EB)
- **Introduced:** MariaDB 5.5.29
- **Deprecated:** MariaDB 10.2.6

innodb_max_changed_pages

- **Description:** Limit to the number of changed page bitmap files (stored in the Information Schema INNODB_CHANGED_PAGES table). Zero is unlimited. See innodb_max_bitmap_file_size and innodb_track_changed_pages. Previously named `innodb_changed_pages_limit`. XtraDB only. Added as a deprecated and ignor in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-max-changed-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1000000`
- **Range:** `0` to `18446744073709551615`
- **Introduced:** MariaDB 5.5.30
- **Deprecated:** MariaDB 10.2.6

innodb_max_dirty_pages_pct

- **Description:** Maximum percentage of unwritten (dirty) pages in the buffer pool. This variable was changed to a double in MariaDB 10.0.15.
- **Commandline:** `--innodb-max-dirty-pages-pct=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `75`
- **Range:** `0` to `99.999` (from MariaDB 10.0.15), `0` to `99` (before MariaDB 10.0.15)

innodb_max_dirty_pages_pct_lwm

- **Description:** Low water mark percentage of dirty pages that will enable preflushing to lower the dirty page ratio. If set to `0`, the default until MariaDB 10.0.15, preflushi disabled. This variable was changed to a double in MariaDB 10.0.15.
- **Commandline:** `--innodb-max-dirty-pages-pct-lwm=#`
- **Scope:** Global
- **Dynamic:** Yes

- **Data Type:** `numeric`
- **Default Value:** `0` (>= MariaDB 10.2.2, <= MariaDB 10.0.14), `0.001000` (>= MariaDB 10.0.15, <= MariaDB 10.2.1)
- **Range:** `0` to `99.999` (>= MariaDB 10.0.15), `0` to `99` (<= MariaDB 10.0.14)
- **Introduced:** MariaDB 10.0.4

---

innodb_max_purge_lag

- **Description:** When purge operations are lagging on a busy server, setting innodb_max_purge_lag can help. By default set to `0`, no lag, the figure is used to calculate lag for each INSERT, UPDATE, and DELETE when the system is lagging. XtraDB/InnoDB keeps a list of transactions with delete-marked index records due to UPDATE DELETE statements. The length of this list is `purge_lag`, and the calculation, performed every ten seconds, is as follows: ((purge_lag/innodb_max_purge_lag)×10)–5 milliseconds.
- **Commandline:** `--innodb-max-purge-lag=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `4294967295`

---

innodb_max_purge_lag_delay

- **Description:** Maximum delay in milliseconds imposed by the innodb_max_purge_lag setting. If set to `0`, the default, there is no maximum.
- **Commandline:** `--innodb-max-purge-lag-delay=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Introduced:** MariaDB 10.0.0

---

innodb_max_undo_log_size

- **Description:** If an undo tablespace is larger than this, it will be marked for truncation if innodb_undo_log_truncate is set.
- **Commandline:** `--innodb-max-undo-log-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:**
  - `10485760` (>= MariaDB 10.2.6)
  - `1073741824` (<= MariaDB 10.2.5)
- **Range:** `10485760` to `18446744073709551615`
- **Introduced:** MariaDB 10.2.2

---

innodb_merge_sort_block_size

- **Description:** Size in bytes of the block used for merge sorting in fast index creation. Replaced in MariaDB 10.0/XtraDB 5.6 by innodb_sort_buffer_size.
- **Commandline:** `innodb-merge-sort-block-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1048576` (1M)
- **Range:** `1048576` (1M) to `1073741824` (1G)
- **Introduced:** MariaDB 5.5.27
- **Removed:** MariaDB 10.0 - replaced by innodb_sort_buffer_size

---

innodb_mirrored_log_groups

- **Description:** Unused. Restored as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Deprecated:** MariaDB 10.0
- **Removed:** MariaDB 10.2.2 - MariaDB 10.2.5

---

innodb_mtflush_threads

- **Description:** Sets the number of threads to use in Multi-Threaded Flush operations. For more information, see Fusion-io Multi-threaded Flush. This feature was deprec version 10.2.9 and removed from version 10.3.2 of MariaDB. Instead, use the `innodb_page_cleaners` system variable.
- **Commandline:** `--innodb-mtflush-threads=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `8`
- **Range:** `1` to `64`

- **Introduced:** MariaDB 10.1.0
- **Deprecated:** MariaDB 10.2.9
- **Removed:** MariaDB 10.3.2

---

innodb_monitor_disable

- **Description:** Disables the specified counters in the INFORMATION_SCHEMA.INNODB_METRICS table.
- **Commandline:** `--innodb-monitor-disable=string`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Introduced:** MariaDB 10.0.0

---

innodb_monitor_enable

- **Description:** Enables the specified counters in the INFORMATION_SCHEMA.INNODB_METRICS table.
- **Commandline:** `--innodb-monitor-enable=string`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Introduced:** MariaDB 10.0.0

---

innodb_monitor_reset

- **Description:** Resets the count value of the specified counters in the INFORMATION_SCHEMA.INNODB_METRICS table to zero.
- **Commandline:** `--innodb-monitor-reset=string`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Introduced:** MariaDB 10.0.0

---

innodb_monitor_reset_all

- **Description:** Resets all values for the specified counters in the INFORMATION_SCHEMA.INNODB_METRICS table.
- **Commandline:** `---innodb-monitor-reset-all=string`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Introduced:** MariaDB 10.0.0

---

innodb_numa_interleave

- **Description:** Whether or not to use the NUMA interleave memory policy to allocate the InnoDB buffer pool. Requires that MariaDB be compiled on a NUMA-enabled Li system.
- **Commandline:** `innodb-numa-interleave={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`

---

innodb_old_blocks_pct

- **Description:** Percentage of the buffer pool to use for the old block sublist.
- **Commandline:** `--innodb-old-blocks-pct=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `37`
- **Range:** `5` to `95`

---

innodb_old_blocks_time

- **Description:** Time in milliseconds an inserted block must stay in the old sublist after its first access before it can be moved to the new sublist. '0' means "no delay". Set non-zero value can help prevent full table scans clogging the buffer pool. See also innodb_old_blocks_pct.
- **Commandline:** `--innodb-old-blocks-time=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `1000` (from MariaDB 10.0), `0` (before MariaDB 10.0)

- **Range:** `0` to $2^{32}-1$

---

innodb_online_alter_log_max_size

- **Description:** The maximum size for temporary log files during online DDL (data and index structure changes). The temporary log file is used for each table being altere index being created, to store data changes to the table while the process is underway. The table is extended by innodb_sort_buffer_size up to the limit set by this variat limit is exceeded, the online DDL operation fails and all uncommitted changes are rolled back. A lower value reduces the time a table could lock at the end of the opera apply all the log's changes, but also increases the chance of the online DDL changes failing.
- **Commandline:** `--innodb-online-alter-log-max-size=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `134217728`
- **Range:** `65536` to $2^{64}-1$
- **Introduced:** MariaDB 10.0.4

---

innodb_open_files

- **Description:** Maximum .ibd files MariaDB can have open at the same time. Only applies to systems with multiple XtraDB/InnoDB tablespaces, and is separate to the ta cache and open_files_limit. In MariaDB 10.0 the default, if innodb_file_per_table is disabled, is 300 or the value of table_open_cache, whichever is higher. It will also au up to the default value if it is set to a value less than `10`.
- **Commandline:** `--innodb-open-files=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `autosized` (from MariaDB 10.0), `300` (before MariaDB 10.0)
- **Range:** `10` to `4294967295`

---

innodb_optimize_fulltext_only

- **Description:** When set to `1` (`0` is default), OPTIMIZE TABLE will only process InnoDB FULLTEXT index data. Only intended for use during fulltext index maintenance
- **Commandline:** `--innodb-optimize-fulltext-only=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.0

---

innodb_page_cleaners

- **Description:** Number of page cleaner threads. The default is `4`, but the value will be set to the number of innodb_buffer_pool_instances if this is lower. If set to `1`, on cleaner thread is used, as was the case until MariaDB 10.2.1. Cleaner threads flush dirty pages from the buffer pool, performing flush list and LRU flushing.
- **Commandline:** `--innodb-page-cleaners=#`
- **Scope:** Global
- **Dynamic:** Yes (>= MariaDB 10.3.3), No (<= MariaDB 10.3.2)
- **Data Type:** `numeric`
- **Default Value:** `4` (or set to innodb_buffer_pool_instances if lower)
- **Range:** `1` to `64`
- **Introduced:** MariaDB 10.2.2

---

innodb_page_size

- **Description:** Size in bytes of the page size for all XtraDB/InnoDB tablespaces. The default, `16k`, is suitable for most uses, but a smaller page size might work more ef in a situation with many small writes (OLTP), or with SSD storage, which usually has smaller block sizes. The page size is set when a MariaDB instance starts, and it re constant afterwards. MariaDB 10.1 allows up to 64K pages for tables with DYNAMIC, COMPACT and REDUNDANT row types, allowing more blob fields to be present Tables with COMPRESSED row type can for now still only be <= 16K page size. Single row size must be still <= 16K and max key length is not affected. Note that Perc previously implemented its own experimental version of `innodb_page_size`, which was deprecated in XtraDB 5.5.30-30.2, and replaced in XtraDB 5.6 with the MySQ version.
- **Commandline:** `--innodb-page-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `enumeration`
- **Default Value:** `16384`
- **Valid Values:** `4k` or `4096`, `8k` or `8192`, `16k` or `16384`. MariaDB 10.1.0 and MySQL 5.7.6 also permits `32k` and `64k`.
- **Introduced:** MariaDB 5.1 (XtraDB), MariaDB 10.0.0 (InnoDB)

---

innodb_pass_corrupt_table

- **Removed:** XtraDB 5.5 - renamed innodb_corrupt_table_action.

innodb_prefix_index_cluster_optimization

- **Description:** Enable prefix optimization to sometimes avoid cluster index lookups.
- **Commandline:** `--innodb-prefix-index-cluster-optimization=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.2

innodb_print_all_deadlocks

- **Description:** If set to `1` (`0` is default), all XtraDB/InnoDB transaction deadlock information is written to the error log.
- **Commandline:** `--innodb-print-all-deadlocks=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5.30

innodb_purge_batch_size

- **Description:** Units of redo log records that will trigger a purge operation. Together with innodb_purge_threads has a small effect on tuning.
- **Commandline:** `--innodb-purge-batch-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `20`
- **Range:** `1` to `5000`
- **Introduced:** MariaDB 5.5

innodb_purge_rseg_truncate_frequency

- **Description:** Frequency with which undo records are purged. Set by default to every 128 times, reducing this increases the frequency at which rollback segments are f also innodb_undo_log_truncate.
- **Commandline:** `-- innodb-purge-rseg-truncate-frequency=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `128`
- **Range:** `1` to `128`
- **Introduced:** MariaDB 10.2.2

innodb_purge_threads

- **Description:** Number of background threads dedicated to XtraDB/InnoDB purge operations. Since MariaDB 10.0, the range has been `1` to `32` . At least one backgrou is always used from MariaDB 10.0. The default has been increased from `1` to `4` in MariaDB 10.2.2. Setting to a value greater than 1 creates that many separate purg This can improve efficiency in some cases, such as when performing DML operations on many tables. In MariaDB 5.5, the options are `0` and `1` . If set to `0` , the defau purging is done with the master thread. If set to `1` , purging is done on a separate thread, which could reduce contention. See also innodb_purge_batch_size.
- **Commandline:** `--innodb-purge-threads=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:**
  - `4` (>= MariaDB 10.2.2)
  - `1` (>=MariaDB 10.0 to <= MariaDB 10.2.1)
  - `0` (MariaDB 5.5)
- **Range:** `1` to `32` (>=MariaDB 10.0), `0` to `1` (MariaDB 5.5)
- **Introduced:** MariaDB 5.5

innodb_random_read_ahead

- **Description:** Originally, random read-ahead was always set as an optimization technique, but was removed in MariaDB 5.5. `innodb_random_read_ahead` permits it instated if set to `1` (`0`) is default.
- **Commandline:** `--innodb-random-read-ahead=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`

innodb_read_ahead

- **Description:** If set to `linear`, the default, XtraDB/InnoDB will automatically fetch remaining pages if there are enough within the same extent that can be accessed sequentially. If set to `none`, read-ahead is disabled. `random` has been removed and is now ignored, while `both` sets to both `linear` and `random`. Also see innodb_read_ahead_threshold for more control on read-aheads. Removed in MariaDB 10.0/XtraDB 5.6 and replaced by MySQL 5.6's innodb_random_read_ahead.
- **Commandline:** `innodb-read-ahead=value`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `linear`
- **Valid Values:** `none`, `random`, `linear`, `both`
- **Removed:** MariaDB 10.0/XtraDB 5.6 - replaced by MySQL 5.6's innodb_random_read_ahead

innodb_read_ahead_threshold

- **Description:** Minimum number of pages XtraDB/InnoDB must read from an extent of 64 before initiating an asynchronous read for the following extent.
- **Commandline:** `--innodb-read-ahead-threshold=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `56`
- **Range:** `0` to `64`

innodb_read_io_threads

- **Description:** Number of I/O threads for XtraDB/InnoDB reads. You may on rare occasions need to reduce this default on Linux systems running multiple MariaDB serv avoid exceeding system limits.
- **Commandline:** `--innodb-read-io-threads=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `4`
- **Range:** `1` to `64`

innodb_read_only

- **Description:** If set to `1` (`0` is default), the server will be read-only. For use in distributed applications, data warehouses or read-only media.
- **Commandline:** `--innodb-read-only=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.4

innodb_recovery_stats

- **Description:** If set to `1` (`0` is default) and recovery is necessary on startup, the server will write detailed recovery statistics to the error log at the end of the recovery p This Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Commandline:** No
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Removed:** MariaDB 10.0

innodb_recovery_update_relay_log

- **Description:** If set to `1` (`0` is default), the relay log info file will be overwritten on crash recovery if the information differs from the InnoDB record. Should not be used storage engine types are being replicated. Previously named `innodb_overwrite_relay_log_info`. Removed in MariaDB 10.0/XtraDB 5.6 and replaced by MySQL `relay-log-recovery`
- **Commandline:** `innodb-recovery-update-relay-log={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** XtraDB 5.5.10-20.1
- **Removed:** MariaDB 10.0 - replaced by MySQL 5.6's `relay-log-recovery`

innodb_replication_delay

- **Description:** Time in milliseconds for the slave server to delay the replication thread if innodb_thread_concurrency is reached.
- **Commandline:** `--innodb-replication-delay=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `4294967295`

---

innodb_rollback_on_timeout

- **Description:** InnoDB usually rolls back the last statement of a transaction that's been timed out (see innodb_lock_wait_timeout). If innodb_rollback_on_timeout is set t default), InnoDB will roll back the entire transaction. Before MariaDB 5.5, rolling back the entire transaction was the default behavior.
- **Commandline:** `--innodb-rollback-on-timeout`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `0`

---

innodb_rollback_segments

- **Description:** Specifies the number of rollback segments that XtraDB/InnoDB will use within a transaction (see undo log). Deprecated and replaced by innodb_undo_lo MariaDB 10.0.
- **Commandline:** `--innodb-rollback-segments=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `128`
- **Range:** `1` to `128`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.0

---

innodb_scrub_log

- **Description:** Enable redo log scrubbing. See Data Scrubbing.
- **Commandline:** `--innodb-scrub-log`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.3

---

innodb_scrub_log_interval

- **Description:** Used with Data Scrubbing in 10.1.3 only - replaced in 10.1.4 by innodb_scrub_log_speed. Innodb redo log scrubbing interval in milliseconds.
- **Commandline:** `--innodb-scrub-log-interval=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `56`
- **Range:** `0` to `50000`
- **Introduced:** MariaDB 10.1.3
- **Removed:** MariaDB 10.1.4

---

innodb_scrub_log_speed

- **Description:** Innodb redo log scrubbing speed in bytes/sec. See Data Scrubbing.
- **Commandline:** `--innodb-scrub-log-speed=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `256`
- **Range:** `1` to `50000`
- **Introduced:** MariaDB 10.1.4

---

innodb_sched_priority_cleaner

- **Description:** Set a thread scheduling priority for cleaner and LRU manager threads. The range from `0` to `39` corresponds in reverse order to Linux nice values of `-2` So `0` is the lowest priority (Linux nice value `19`) and `39` is the highest priority (Linux nice value `-20`). XtraDB only. Added as a deprecated and ignored option in Ma

10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-sched-priority-cleaner=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `19`
- **Range:** `0` to `39`
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_show_locks_held

- **Description:** Specifies the number of locks held for each InnoDB transaction to be displayed in SHOW ENGINE INNODB STATUS output. XtraDB only. Added as a de and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-show-locks-held=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `10`
- **Range:** `0` to `1000`
- **Deprecated:** MariaDB 10.2.6

---

innodb_show_verbose_locks

- **Description:** If set to `1`, and innodb_status_output_locks is also ON, the traditional InnoDB behavior is followed and locked records will be shown in SHOW ENGINE I STATUS output. If set to `0`, the default, only high-level information about the lock is shown. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-show-verbose-locks=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `1`
- **Deprecated:** MariaDB 10.2.6

---

innodb_simulate_comp_failures

- **Description:** Simulate compression failures. Used for testing robustness against random compression failures. XtraDB only.
- **Commandline:** None
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `99`
- **Introduced:** MariaDB 10.0.14

---

innodb_sort_buffer_size

- **Description:** Size of the sort buffers used for sorting data when an InnoDB index is created, as well as the amount by which the temporary log file is extended during o operations to record concurrent writes. Before MariaDB 10.0, this was not configurable and the current default setting of 1MB was fixed. The larger the setting, the fewe phases are required between buffers while sorting. When a CREATE TABLE or ALTER TABLE creates a new index, three buffers of this size are allocated, as well as p for the rows in the buffer.
- **Commandline:** `--innodb-sort-buffer-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `1048576` (1M)
- **Range:** `65536` to `67108864`
- **Introduced:** MariaDB 10.0.0

---

innodb_spin_wait_delay

- **Description:** Maximum delay (not strictly corresponding to a time unit) between spin lock polls. Default changed from `6` to `4` in MariaDB 10.3.5, as this was verified t best throughput by OLTP update index and read-write benchmarks on Intel Broadwell (2/20/40) and ARM (1/46/46).
- **Commandline:** `--innodb-spin-wait-delay=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `4` (>= MariaDB 10.3.5), `6` (<= MariaDB 10.3.4)
- **Range:** `0` to `4294967295`

---

innodb_stats_auto_recalc

- **Description:** If set to `1` (the default), persistent statistics are automatically recalculated when the table changes significantly (more than 10% of the rows). Affects table created or altered with STATS_PERSISTENT=1 (see CREATE TABLE), or when innodb_stats_persistent is enabled. innodb_stats_persistent_sample_pages determine much data to sample when recalculating. See InnoDB Persistent Statistics.
- **Commandline:** `--innodb-stats-auto-recalc=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 10.0.4

---

innodb_stats_auto_update

- **Description:** If set to `0` (`1` is default), index statistics will not be automatically calculated except when an ANALYZE TABLE is run, or the table is first opened. Replac innodb_stats_auto_recalc in MariaDB 10.0/XtraDB 5.6.
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `1`
- **Introduced:** XtraDB 5.5.8-20.0
- **Removed:** MariaDB 10.0 - replaced by innodb_stats_auto_recalc.

---

innodb_stats_include_delete_marked

- **Description:** Include delete marked records when calculating persistent statistics.
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.2.6

---

innodb_stats_method

- **Description:** Determines how NULLs are treated for XtraDB/InnoDB index statistics purposes. If set to `nulls_equal`, the default, all NULL index values are treated a group. This is usually fine, but if you have large numbers of NULLs the average group size is slanted higher, and the optimizer may miss using the index for ref accesse would be useful. If set to `nulls_unequal`, the opposite approach is taken, with each NULL forming its own group of one. Conversely, the average group size is slante and the optimizer may use the index for ref accesses when not suitable. Setting to `nulls_ignored` ignores NULLs altogether from index group calculations. See also Statistics, aria_stats_method and myisam_stats_method.
- **Commandline:** `--innodb-stats-method=name`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `enumeration`
- **Default Value:** `nulls_equal`
- **Valid Values:** `nulls_equal`, `nulls_unequal`, `nulls_ignored`
- **Introduced:** MariaDB 5.5

---

innodb_stats_modified_counter

- **Description:** The number of rows modified before we calculate new statistics. If set to `0`, the default, current limits are used.
- **Commandline:** `--innodb-stats-modified-counter=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `18446744073709551615`
- **Introduced:** MariaDB 10.0.15

---

innodb_stats_on_metadata

- **Description:** If set to `1`, the default, XtraDB/InnoDB updates statistics when accessing the INFORMATION_SCHEMA.TABLES or INFORMATION_SCHEMA.STATIST tables, and when running metadata statements such as SHOW INDEX or SHOW TABLE STATUS. If set to `0`, statistics are not updated at those times, which can redu access time for large schemas, as well as make execution plans more stable.
- **Commandline:** `--innodb-stats-on-metadata`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF` (from MariaDB 10.0), `ON` (before MariaDB 10.0)
- **Introduced:** MariaDB 5.5

---

innodb_stats_persistent

- **Description:** ANALYZE TABLE produces index statistics, and this setting determines whether they will be stored on disk, or be required to be recalculated more freque as when the server restarts. This information is stored for each table, and can be set with the STATS_PERSISTENT clause when creating or altering tables (see CREA TABLE). See InnoDB Persistent Statistics.
- **Commandline:** `--innodb-stats-persistent=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 10.0.4

---

innodb_stats_persistent_sample_pages

- **Description:** Number of index pages sampled when estimating cardinality and statistics for indexed columns. Increasing this value will increases index statistics accura use more I/O resources when running ANALYZE TABLE. See InnoDB Persistent Statistics.
- **Commandline:** `--innodb-stats-persistent-sample-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `20`
- **Introduced:** MariaDB 10.0.0

---

innodb_stats_sample_pages

- **Description:** Gives control over the index distribution statistics by determining the number of index pages to sample. Higher values produce more disk I/O, but, especia large tables, produce more accurate statistics and therefore make more effective use of the query optimizer. Lower values than the default are not recommended, as th statistics can be quite inaccurate. Enabling innodb_stats_traditional will help large tables by using more samples. Deprecated in MariaDB 10.0 - use innodb_stats_transient_sample_pages instead.
- **Commandline:** `--innodb-stats-sample-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `8`
- **Range:** `1` to $2^{64}-1$
- **Deprecated:** MariaDB 10.0

---

innodb_stats_traditional

- **Description:** When enabled (the default), traditional statistics calculation based on the number of configured pages is used. When disabled, the innodb_stats_sample_ (MariaDB 5.5) or innodb_stats_transient_sample_pages (MariaDB 10) is multiplied by log2(pages in table) to give a larger sample of pages for larger tables for the purp index statistics calculation.
- **Commandline:** `--innodb-stats-traditional=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 5.5.41/MariaDB 10.0.16

---

innodb_stats_transient_sample_pages

- **Description:** Gives control over the index distribution statistics by determining the number of index pages to sample. Higher values produce more disk I/O, but, especia large tables, produce more accurate statistics and therefore make more effective use of the query optimizer. Lower values than the default are not recommended, as th statistics can be quite inaccurate. Enabling innodb_stats_traditional will help large tables by using more samples. If persistent statistics are used on a table (see innodb_stats_persistent), the setting from innodb_stats_persistent_sample_pages) applies instead.
- **Commandline:** `--innodb-stats-transient-sample-pages=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `8`
- **Range:** `1` to $2^{64}-1$
- **Introduced:** MariaDB 10.0/MySQL 5.6

---

innodb_stats_update_need_lock

- **Description:** Setting to `0` (`1` is default) may help reduce contention of the `&dict_operation_lock`, but also disables the *Data_free* option in SHOW TABLE STATU Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `1`

- **Introduced:** XtraDB 5.5.8-20.0
- **Removed:** MariaDB 10.0/XtraDB 5.6

---

innodb_status_output

- **Description:** Enable InnoDB monitor output to the error log.
- **Commandline:** `--innodb-status-output={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.11

---

innodb_status_output_locks

- **Description:** Enable InnoDB lock monitor output to the error log and SHOW ENGINE INNODB STATUS. Also requires innodb_status_output=ON to enable output to th log.
- **Commandline:** `--innodb-status-output-locks={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.11

---

innodb_strict_mode

- **Description:** If set to `1` (`0` is the default before MariaDB 10.2.2), XtraDB/InnoDB will return errors instead of warnings in certain cases, similar to strict SQL mode.
- **Commandline:** `--innodb-strict-mode=#`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:**
  - `ON` (>= MariaDB 10.2.2)
  - `OFF` (<= MariaDB 10.2.1)

---

innodb_support_xa

- **Description:** If set to `1` , the default, XA transactions are supported. XA support ensures data is written to the binary log in the same order to the actual database, whic critical for replication and disaster recovery, but comes at a small performance cost. If your database is set up to only permit one thread to change data (for example, or replication slave with only the replication thread writing), it is safe to turn this option off. Removed in MariaDB 10.3, XA transactions are always supported.
- **Commandline:** `--innodb-support-xa`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Deprecated:** MariaDB 10.2
- **Removed:** MariaDB 10.3.0

---

innodb_sync_array_size

- **Description:** By default `1` , can be increased to split internal thread co-ordinating, giving higher concurrency when there are many waiting threads.
- **Commandline:** `--innodb-sync-array-size=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `1` to `1024`
- **Introduced:** MariaDB 10.0/MySQL 5.6

---

innodb_sync_spin_loops

- **Description:** The number of times a thread waits for an XtraDB/InnoDB mutex to be freed before the thread is suspended.
- **Commandline:** `--innodb-sync-spin-loops=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `30`
- **Range:** `0` to `4294967295`

---

innodb_table_locks

- **Description:** If autocommit is set to to `0` (`1` is default), setting innodb_table_locks to `1`, the default, will cause XtraDB/InnoDB to lock a table internally upon a LOCK
- **Commandline:** `--innodb-table-locks`
- **Scope:** Global, Session
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`

---

innodb_thread_concurrency

- **Description:** Once this number of threads is reached (excluding threads waiting for locks), XtraDB/InnoDB will place new threads in a wait state in a first-in, first-out qu execution, in order to limit the number of threads running concurrently. A setting of `0`, the default, permits as many threads as necessary. A suggested setting is twice t number of CPU's plus the number of disks.
- **Commandline:** `--innodb-thread-concurrency=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `1000`

---

innodb_thread_concurrency_timer_based

- **Description:** If set to `1`, thread concurrency will be handled in a lock-free timer-based manner rather than the default mutex-based method. Depends on atomic op bu being available. This Percona XtraDB variable has not been ported to XtraDB 5.6.
- **Commandline:** `innodb-thread-concurrency-timer-based={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Removed:** MariaDB 10.0/XtraDB 5.6

---

innodb_thread_sleep_delay

- **Description:** Time in microseconds that XtraDB/InnoDB threads sleep before joining the queue. Setting to `0` disables sleep.
- **Commandline:** `--innodb-thread-sleep-delay=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `10000`

---

innodb_temp_data_file_path

- **Description:**
- **Commandline:** `--innodb-temp-data-file-path=path`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `string`
- **Default Value:** `ibtmp1:12M:autoextend`
- **Introduced:** MariaDB 10.2.2

---

innodb_tmpdir

- **Description:** Allows an alternate location to be set for temporary non-tablespace files. If not set (the default), files will be created in the usual tmpdir location.
- **Commandline:** `--innodb-tmpdir=path`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `string`
- **Default Value:** Empty
- **Introduced:** MariaDB 10.1.14, MariaDB 10.2.1

---

innodb_track_changed_pages

- **Description:** For faster incremental backup with Xtrabackup, XtraDB tracks pages with changes written to them according to the redo log and writes the information to changed page bitmap files. This read-only variable is used for controlling this feature. See also innodb_max_changed_pages and innodb_max_bitmap_file_size. XtraD Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-track-changed-pages={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`

- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.0.9
- **Deprecated:** MariaDB 10.2.6

---

innodb_track_redo_log_now

- **Description:** Available on debug builds only. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of Xtral allow for easier upgrades.
- **Commandline:** `innodb-track-redo-log-now={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Deprecated:** MariaDB 10.2.6

---

innodb_undo_directory

- **Description:** Path to the directory (relative or absolute) that InnoDB uses to create separate tablespaces for the undo logs. `.` (the default value before 10.2.2) leaves t logs in the same directory as the other log files. From MariaDB 10.2.2, the default value is NULL, and if no path is specified, undo tablespaces will be created in the dire defined by datadir. Use together with innodb_undo_logs and innodb_undo_tablespaces. Undo logs are most usefully placed on a separate storage device.
- **Commandline:** `--innodb-undo-directory=name`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `string`
- **Default Value:** NULL (>= 10.2.2), `.` (<= 10.2.1)
- **Introduced:** MariaDB 10.0.0

---

innodb_undo_log_truncate

- **Description:** When enabled, undo tablespaces that are larger than innodb_max_undo_log_size are marked for truncation. See also innodb_purge_rseg_truncate_freq
- **Commandline:** `--innodb-undo-log-truncate[={0|1}]`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.2.2

---

innodb_undo_logs

- **Description:** Specifies the number of rollback segments that XtraDB/InnoDB will use within a transaction (or the number of active undo logs). By default set to the max `128`, it can be reduced to avoid allocating unneeded rollback segments. See the Innodb_available_undo_logs status variable for the number of undo logs available. Se innodb_undo_directory and innodb_undo_tablespaces. Replaces innodb_rollback_segments in MariaDB 10.0/MySQL 5.6. The Information Schema XTRADB_RSEG T contains information about the XtraDB rollback segments.
- **Commandline:** `--innodb-undo-logs=#`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `numeric`
- **Default Value:** `128`
- **Range:** `0` to `128`
- **Introduced:** MariaDB 10.0.0

---

innodb_undo_tablespaces

- **Description:** Number of tablespaces files used for dividing up the undo logs. By default, undo logs are all part of the system tablespace, which contains one undo table more than the `innodb_undo_tablespaces` setting. When the undo logs can grow large, splitting them over multiple tablespaces will reduce the size of any single tabl Must be set before InnoDB is initialized, or else MariaDB will fail to start, with an error saying that `InnoDB did not find the expected number of undo tablesp` The files are created in the directory specified by innodb_undo_directory, and are named `undoN`, N being an integer. The default size of an undo tablespace is 10MB. innodb_undo_logs must have a non-zero setting for `innodb_undo_tablespaces` to take effect.
- **Commandline:** `--innodb-undo-tablespaces=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `0`
- **Range:** `0` to `95` (>= MariaDB 10.2.2), `0` to `126` (<= MariaDB 10.2.1)
- **Introduced:** MariaDB 10.0.0

---

innodb_use_atomic_writes

- **Description:** Implement atomic writes on FusionIO devices. See atomic write support for other variables affected when this is set.

- **Commandline:** `innodb-use-atomic-writes={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON` (>= MariaDB 10.2.4), `OFF` (<= MariaDB 10.2.3)
- **Introduced:** MariaDB 5.5.31

---

innodb_use_fallocate

- **Description:** Preallocate files fast, using operating system functionality. On POSIX systems, posix_fallocate system call is used. Automatically set to `1` when innodb_use_atomic_writes is set - see FusionIO DirectFS atomic write support.
- **Commandline:** `innodb-use-fallocate={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5.31
- **Deprecated:** MariaDB 10.2.5 (treated as if `ON`)
- **Removed:** MariaDB 10.3.0

---

innodb_use_global_flush_log_at_trx_commit

- **Description:** Determines whether a user can set the variable innodb_flush_log_at_trx_commit. If set to `1`, a user cannot reset the value with a SET command, while if `1`, a user can reset the value of `innodb_flush_log_at_trx_commit`. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoD default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `innodb-use-global-flush-log-at-trx_commit={0|1}`
- **Scope:** Global
- **Dynamic:** Yes
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 5.5
- **Deprecated:** MariaDB 10.2.6

---

innodb_use_mtflush

- **Description:** Whether to enable Multi-Threaded Flush operations. For more information, see Fusion. This feature was deprecated in version 10.2.9 and removed from 10.3.2 of MariaDB. Instead, use `innodb_page_cleaners` system variable.
- **Commandline:** `--innodb-use-mtflush=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 10.1.0
- **Deprecated:** MariaDB 10.2.9
- **Removed:** MariaDB 10.3.2

---

innodb_use_native_aio

- **Description:** For Linux systems only, specified whether to use Linux's asynchronous I/O subsystem. Set to `1` by default, it may be changed to `0` at startup if XtraDB/I detects a problem
- **Commandline:** `--innodb-use-native-aio=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Introduced:** MariaDB 5.5

---

innodb_use_purge_thread

- **Description:** Usually with InnoDB, data changed by a transaction is written to an undo space to permit read consistency, and freed when the transaction is complete. M large, transactions, can cause the main tablespace to grow dramatically, reducing performance. This option, introduced in XtraDB 5.1 and removed for 5.5, allows multi threads to perform the purging, resulting in slower, but much more stable performance.
- **Commandline:** `--innodb-use-purge-thread=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `1`
- **Range:** `0` to `32`
- **Introduced:** XtraDB 5.1
- **Removed:** XtraDB 5.5

---

innodb_use_stacktrace

- **Description:** If set to `ON` (`OFF` is default), a signal handler for SIGUSR2 is installed when the InnoDB server starts. When a long semaphore wait is detected at sync/sync0array.c, a SIGUSR2 signal is sent to the waiting thread and thread that has acquired the RW-latch. For both threads a full stacktrace is produced as well as i possible. XtraDB only. Added as a deprecated and ignored option in MariaDB 10.2.6 (which uses InnoDB as default instead of XtraDB) to allow for easier upgrades.
- **Commandline:** `--innodb-use-stacktrace=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `OFF`
- **Introduced:** MariaDB 5.5.34
- **Deprecated:** MariaDB 10.2.6

---

innodb_use_sys_malloc

- **Description:** If set the `1` , the default, XtraDB/InnoDB will use the operating system's memory allocator. If set to `0` it will use its own. Deprecated in MariaDB 10.0 and in MariaDB 10.2 along with InnoDB's internal memory allocator.
- **Commandline:** `--innodb-use-sys-malloc=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON`
- **Deprecated:** MariaDB 10.0
- **Removed**: MariaDB 10.2.2

---

innodb_use_sys_stats_table

- **Description:** If set to `1` (`0` is default), XtraDB will use the SYS_STATS system table for extra table index statistics. When a table is opened for the first time, statistics be loaded from SYS_STATS instead of sampling the index pages. Statistics are designed to be maintained only by running an ANALYZE TABLE. Replaced by MySQL Persistent Optimizer Statistics.
- **Commandline:** `innodb-use-sys-stats-table={0|1}`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `0`
- **Introduced:** XtraDB 5.5.8-20.0
- **Removed:** MariaDB 10.0/XtraDB 5.6

---

innodb_use_trim

- **Description:** Use trim to free up space of compressed blocks. See InnoDB/XtraDB Page Compression - Persistent Trim). Note that this setting works best with NVMFS InnoDB holepunch compression vs the filesystem in MariaDB 10.1).
- **Commandline:** `--innodb-use-trim=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `boolean`
- **Default Value:** `ON` (>= MariaDB 10.2.4), `OFF` (<= MariaDB 10.2.3)
- **Introduced:** MariaDB 10.1.0, MariaDB 10.0.15 Fusion-io
- **Deprecated:** MariaDB 10.2.4
- **Removed:** MariaDB 10.3.0

---

innodb_version

- **Description:** InnoDB version number.
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `string`

---

innodb_write_io_threads

- **Description:** Number of I/O threads for XtraDB/InnoDB writes. You may on rare occasions need to reduce this default on Linux systems running multiple MariaDB serv avoid exceeding system limits.
- **Commandline:** `--innodb-write-io-threads=#`
- **Scope:** Global
- **Dynamic:** No
- **Data Type:** `numeric`
- **Default Value:** `4`
- **Range:** `1` to `64`

Comments

No comments