title: ClickHouse Monitoring Integration description: Comprehensive view of your database's health and performance with Sematext ClickHouse monitoring integration. Our ClickHouse integration provides support to monitor ClickHouse current metrics, profile events, asynchronous metrics, database/table level replication and replica metrics.

Integration

- Instructions: https://apps.sematext.com/ui/howto/ClickHouse/overview

More about ClickHouse Monitoring

- Key Metrics for Monitoring ClickHouse
- ClickHouse Monitoring Tools
- Monitoring ClickHouse with Sematext

Metrics

Metric Name Key *(Type)* *(Unit)*	Description
Max relative replica queue delay**clickhouse.repl.queue.delay.relative.max** *(double gauge)* *(ms)*	Relative delay is the maximum difference of absolute delay from any other replica
Max absolute replica queue delay**clickhouse.repl.queue.delay.absolute.max** *(double gauge)* *(ms)*	Maximum replica queue delay relative to current time
Max active part count**clickhouse.part.count.max** *(double gauge)*	Maximum number of active parts in partitions
Mark cache size**clickhouse.cache.mark.size** *(double gauge)* *(bytes)*	Mark cache - Cache of 'marks' for StorageMergeTree. Marks is an index structure that addresses ranges in column file, corresponding to ranges of primary key
Heap size**clickhouse.heap.size** *(double gauge)* *(bytes)*	Number of bytes in the heap (current_allocated_bytes + fragmentation + freed memory regions)
Current allocated memory**clickhouse.current.allocated.bytes** *(double gauge)* *(bytes)*	Number of bytes currently allocated by application
Allocated bytes**clickhouse.dict.allocated.bytes** *(long gauge)* *(bytes)*	The amount of memory used by the dictionary.

Metric Name Key *(Type)* *(Unit)*	Description
Element count**clickhouse.dict.element.count** *(long gauge)*	The number of items stored in the dictionary.
Load factor**clickhouse.dict.load.factor** *(double gauge)*	The filled percentage (0.0 - 1.0) of the dictionary (for a hashed dictionary, it is the filled percentage of the hash table).
Total query count**clickhouse.query.count** *(long counter)*	Total number of queries started to be interpreted and may be executed.
Select query count**clickhouse.query.select.count** *(long counter)*	Number of SELECT queries started to be interpreted and may be executed.
Insert query count**clickhouse.query.insert.count** *(long counter)*	Number of INSERT queries started to be interpreted and may be executed.
Failed file opens**clickhouse.file.open.failed** *(long counter)*	
Read buffer failed file reads**clickhouse.buffer.read.fd.failed** *(long counter)*	Number of times the read (read/pread) from a file descriptor has failed.
Write buffer failed file writes**clickhouse.buffer.write.fd.failed** *(long counter)*	Number of times the write (write/pwrite) to a file descriptor has failed.
Inserted rows**clickhouse.insert.rows** *(long counter)*	Number of rows inserted to all tables.
Inserted bytes**clickhouse.insert.bytes** *(long counter) (bytes)*	Number of uncompressed bytes inserted to all tables.
Merged rows**clickhouse.merge.rows** *(long counter)*	Rows read for background merges. This is the number of rows before merge.
Mark cache hits**clickhouse.cache.mark.hits** *(long counter)*	Mark cache - Cache of ‘marks’ for merge tree storage engine. Marks is an index structure that addresses ranges in column file, corresponding to ranges of primary key
Mark cache misses**clickhouse.cache.mark.misses** *(long counter)*	Mark cache - Cache of ‘marks’ for merge tree storage engine. Marks is an index structure that addresses ranges in column file, corresponding to ranges of primary key

Metric Name Key *(Type)* *(Unit)*	Description
Replicated part fetches**clickhouse.repl.part.fetches** *(long counter)*	Number of times a data part was downloaded from the replica of a ReplicatedMergeTree table.
Failed replicated part fetches**clickhouse.repl.part.fetches.failed** *(long counter)*	
Obsolete replicated parts**clickhouse.repl.part.obsolete** *(long counter)*	Replicated parts that are replaced/rendered obsolete by fetching new parts.
Replicated part merges**clickhouse.repl.part.merge.count** *(long counter)*	
Fetches of merged replicated parts**clickhouse.repl.part.fetches.merged** *(long counter)*	Number of times the system prefers to download an already merged part from the replica of ReplicatedMergeTree table.
Replicated part checks**clickhouse.repl.part.checks** *(long counter)*	
Failed replicated part checks**clickhouse.repl.part.checks.failed** *(long counter)*	
Lost replicated parts**clickhouse.repl.part.lost** *(long counter)*	Replicated parts lost forever (possible if on all the replicas where the part was, is deteriorated), detected during part checks.
Distributed Connection Retries**clickhouse.connection.dist.retries** *(long counter)*	Count of connection retries in replicated DB connection pool
Distributed Connection Fails**clickhouse.connection.dist.fails** *(long counter)*	Count of connection failures after all retries in replicated DB connection pool
Uncompressed bytes merged**clickhouse.merge.bytes.uncompressed** *(long counter) (bytes)*	Uncompressed bytes that was read for background merges. This is the number before merge.
Merge time**clickhouse.merge.time** *(long counter) (ms)*	Total time spent for background merges.
RW Lock acquired read locks**clickhouse.lock.rw.acquired.reads** *(long counter)*	Count of acquired read locks on table storage. RW Locks are used to control concurrent access to table structure and data
RW Lock reader wait time**clickhouse.lock.rw.reader.wait.time** *(long counter) (ms)*	Total time waited to get read locks on table storage. RW locks are used to control concurrent access to table structure and data

Metric Name Key *(Type)* *(Unit)*	Description
RW Lock acquired write locks**clickhouse.lock.rw.acquired.writes** *(long counter)*	Count of acquired write locks on table. RW locks are used to control concurrent access to table structure
RW Lock write wait time**clickhouse.lock.rw.writer.wait.time** *(long counter) (ms)*	Total time waited to get write locks on table storage. RW locks are used to control concurrent access to table structure
Delayed inserts**clickhouse.insert.delayed** *(long counter)*	Part inserts that are delayed because the current `Max active part count` is more than `parts_to_delay_insert` setting
Rejected inserts**clickhouse.insert.rejected** *(long counter)*	Part inserts that are rejected because the current `Max active part count` is more than `parts_to_throw_insert` setting
ZooKeeper wait time**clickhouse.zk.wait.time** *(long counter) (microseconds)*	Time spent in waiting for ZooKeeper operations
ZooKeeper exceptions**clickhouse.zk.exceptions** *(long counter)*	Count of exceptions during ZooKeeper operations
ZooKeeper ephemeral node removal failures**clickhouse.zk.nodes.ephemeral.remove.fails** *(long counter)*	Count of ZooKeeper ephemeral node removal failures
Network errors**clickhouse.network.errors** *(long counter)*	Count of network errors (timeouts and connection failures) during query execution, background pool tasks and DNS cache update
Distributed Sync insertion timeouts**clickhouse.distributed.sync.insert.timeout** *(long counter)*	Count of sync distributed insert wait timeout exceeded errors in distributed storage engine
Cache dictionary expired keys**clickhouse.dict.cache.keys.expired** *(long counter)*	
Cache dictionary keys not found**clickhouse.dict.cache.keys.notfound** *(long counter)*	
Cache dictionary keys hits**clickhouse.dict.cache.keys.hits** *(long counter)*	
TCP Connections**clickhouse.connection.tcp.count** *(long gauge)*	Number of connections to TCP server (clients with native interface)

Metric Name Key *(Type)* *(Unit)*	Description
HTTP Connections**clickhouse.connection.http.count** *(long gauge)*	Number of connections to HTTP server
Interserver Connections**clickhouse.connection.interserver.count** *(long gauge)*	Number of connections from other replicas to fetch parts
Query Threads**clickhouse.query.thread.count** *(long gauge)*	Number of query processing threads
Preempted Queries**clickhouse.query.preempted.count** *(long gauge)*	Number of queries that are stopped and waiting due to 'priority' setting.
BackgroundPool Tasks**clickhouse.background.pool.tasks** *(long gauge)*	Number of active tasks in BackgroundProcessingPool (merges, mutations, fetches or replication queue bookkeeping)
Reads**clickhouse.reads** *(long gauge)*	Number of read (read, pread, io_get events, etc.) syscalls in progress
Writes**clickhouse.writes** *(long gauge)*	Number of write (write, pwrite, io_get events, etc.) syscalls in progress
Memory**clickhouse.memory.tracking** *(long gauge)* *(bytes)*	Total amount of memory (bytes) allocated in currently executing queries. Note that some memory allocations may not be accounted.
Running merges**clickhouse.merge.count** *(long gauge)*	Number of executing background merges (if merged takes very short time, they may not be counted)
Open Files (Read)**clickhouse.files.open.read** *(long gauge)*	Number of files open for reading
Open Files (Write)**clickhouse.files.open.write** *(long gauge)*	Number of files open for writing
Distributed Sends**clickhouse.distributed.send** *(long gauge)*	Number of connections sending data, that was inserted to Distributed tables, to remote servers. Both synchronous and asynchronous mode.
Current leader elections**clickhouse.zk.leader.election** *(long gauge)*	Number of replicas participating in leader election. Equals to total number of replicas in usual cases.
Ephemeral nodes**clickhouse.zk.nodes.ephemeral** *(long gauge)*	Number of ephemeral nodes held in ZooKeeper.

Metric Name Key *(Type)* *(Unit)*	Description
ZooKeeper sessions**clickhouse.zk.sessions** *(long gauge)*	Number of sessions (connections) to ZooKeeper. Should be no more than one.
ZooKeeper watches**clickhouse.zk.watches** *(long gauge)*	Number of watches (event subscriptions) in ZooKeeper.
ZooKeeper requests**clickhouse.zk.requests** *(long gauge)*	Number of requests to ZooKeeper in progress.
Table size on disk**clickhouse.mergetree.table.size** *(long gauge) (bytes)*	
Active part count**clickhouse.mergetree.table.parts** *(long gauge)*	
Row count**clickhouse.mergetree.table.rows** *(long gauge)*	
Replica read-only**clickhouse.replica.readonly** *(long gauge)*	True if the config doesn't have session with ZK, if an unknown error occurred when reinitializing sessions in ZK, and during session reinitialization in ZK.
Replica session expired**clickhouse.replica.session.expired** *(long gauge)*	True if the ZK session expired
Replica future parts**clickhouse.replica.parts.future** *(long gauge)*	The number of data parts that will appear as the result of inserts or merges that haven't been done yet
Replica parts to check**clickhouse.replica.parts.tocheck** *(long gauge)*	The number of data parts in the queue for verification. A part is put in the verification queue if there is suspicion that it might be damaged.
Replica queue size**clickhouse.replica.queue.size** *(long gauge)*	Size of the queue for operations waiting to be performed. Operations include inserting blocks of data, merges, and certain other actions.
Replica queue inserts**clickhouse.replica.queue.inserts** *(long gauge)*	Number of inserts of blocks of data that need to be made. Insertions are usually replicated fairly quickly. If the number is high, something is wrong.
Replica queue merges**clickhouse.replica.queue.merges** *(long gauge)*	The number of merges waiting to be made. Sometimes merges are lengthy, so this value may be greater than zero for a long time

Metric Name Key *(Type)* *(Unit)*	Description
Replica log max index**clickhouse.replica.log.max.index** *(long gauge)*	Maximum entry number in the log of general activity.
Replica log pointer**clickhouse.replica.log.pointer** *(long gauge)*	Maximum entry number in the log of general activity that the replica copied to its execution queue, plus one. If log pointer is much smaller than log max index, something is wrong.
Total replicas**clickhouse.replica.total.replicas** *(long gauge)*	The total number of known replicas of this table.
Active replicas**clickhouse.replica.active.replicas** *(long gauge)*	The number of replicas of this table that have a session in ZooKeeper (i.e., the number of functioning replicas).