Online Schema Changes

ClustrixDB performs online schema changes without blocking reads or writes to a table.

ClustrixDB uses MVCC (multi-version concurrency control) to avoid locking to tables undergoing a schema change. New temporary container(s) are created, and while data is being copied, a temporary transaction log records any writes run against the old containers. Once all the original records have been copied and logged transactions have been processed, the ALTER transaction completes, the new table is available, and the old table is discarded. All of these mechanics are performed automatically when you execute your ALTER SQL.

ClustrixDB maintains read consistency during the ALTER – read and write queries that run against the table before the ALTER commits see the original table schema. Reads and writes after the ALTER commits see the new schema. From the perspective of any single query or user, the ALTER is instantaneous.

Best Practices

While ClustrixDB is designed to easily support online schema changes for tables of any size, the following best practices help to minimize negative and unforeseen impacts to your application. This includes:

1. Since an ALTER on a large table can take a while to complete, we suggest using screen or some other detachable terminal.
2. Test the new schema with your application's queries in a non-production environment. Compare the output of the EXPLAIN plan before and after the change.
3. Perform schema change(s) during off-peak hours or during a maintenance window.
4. Ensure there is adequate disk space (see below).
5. Understand impacts to replication and revise plan accordingly.

Estimating Disk Space Requirements

Online schema changes pins BigC, which means that undo log is not cleaned up until the ALTER completes. This means that for the ALTER to complete, there must be sufficient space. To calculate how much free space is required for an ALTER you should review the following:

- The current amount of space used.
- The minimum amount of free space desired.
  - Plan to reserve a minimum of at least 10% free space throughout your schema changes.
  - ClustrixDB will kill long-running transactions (including the ALTER) if free space falls under 5%.
- Ensure there is sufficient space to store a full copy of the table, including replicas. The size of a table can be estimated from system.table_sizes.
- The estimated amount of time it will take to complete the ALTER.
- New data growth rate.
- Undo log size. The undo log for all INSERT, UPDATE, and DELETE operations will accumulate during the ALTER.
- Binlog size and growth rate. During the ALTER, binlog trims will still run, but the disk space will not be freed until BigC is unpinned.

When you add up all the planned growth and size of the table copy plus the current space consumed, you should still have at least 10% of disk remaining. If not, trimming binlogs, pruning tables, or expanding the cluster with additional servers is recommended before initiating the ALTER.

Example: Disk Space required for an ALTER

In the following example, a column will be added to a 1TB table that is part of a 10TB cluster.

Description of Space Requirement	TB Required
Our cluster is currently 10 TB of usable space and we know the system reserves 5% for internal operations. If we exceed this, writes can be blocked and the ALTER will roll-back.	0.5 TB
Our users have requested we leave at least 10% of space free at all times to handle surges in load.	1.0 TB
The new table will consume a bit over 1 TB due to the additional column. We could calculate rows times bytes, but let's assume 1.1 TB. Because this is a copy, we have to reserve this space until the ALTER is complete.	1.1 TB
The entire cluster receives approximately ~1.5 TB per day of write transactions (including writes to the binlogs) so (1.5 / 24) * 25 hours = 1.5625 TB	1.6 TB
Total free space required to safely begin the ALTER:	**4.2 TB**

A conservative estimate would be that there should be at least 4.5 TB available space on the cluster to allow the ALTER to complete.

Monitoring the Progress of an ALTER

To view the status of an ALTER in-process, use this SQL.

```
sql> select * from system.alter_progress;
```

Replication Concerns

Performing ALTER operations in replicated environments requires additional planning. Whenever possible, it is recommended to allow the ALTER to run over the replication stream.

Running the ALTER over replication means that the ALTER must execute sequentially, first on the master, then on the slave. Furthermore, the slave must process the ALTER sequentially within the replication stream, and all other writes are paused while the slave processes through the ALTER. For large tables, ALTER operations can cause the slave to fall behind the master. Plan to adjust binlog retention on the master and monitor replication regularly to ensure the slave does not exceed your master's binlog retention during this process.

Column Matching

When replicating via row-based binlogs, column synchronization is critical. RBR (row-based replication) sends both the old and new rows to the slave and the old must match before the new is applied. ALTER operations that modify columns must be executed within replication and in sequence to avoid slave errors trying to applying writes for one schema to the other.

When replicating via statement-based (SBR) binlogs, more flexibility is allowed, provided that the statements execute equally well on the old and new schema. However, for environments with high concurrency, it is extremely difficult to assess whether all statements are equivalent on the old and new schemas.

Exceptions to these column matching concerns include ALTER operations that add an index, change the storage type, or modify the number of slices. Since these do not affect the insertion or update of rows, they may be executed in parallel outside the replication stream.

Scale-Out Concerns for Online Schema Changes

The performance of ALTER on user data will vary depending on your particular environment and workload. Take the following as guidance and adapt based on your own experience and testing.

Modifying a Simple, Small Table

For tables around N*10^6 rows or with light concurrency, ALTER can be done live and should complete within minutes depending on the cluster size, row size, and overall load. Cache and query plan refreshes occur regularly and the automatic refresh should handle any performance issues.

Medium Scale-Out Issues with Online Schema Changes

Tables that are being accessed with high concurrency or have more than N*10^7 rows may experience degraded cluster performance immediately after the ALTER completes. ClustrixDB stores statistics and query plans for each table, and these values are cached. ClustrixDB's periodic refresh and flush of that information may not occur quickly enough and may consequently impact performance. To avoid this, immediately flush the cache once the ALTER is complete by using the following commands:

```
sql> ALTER TABLE ...

shell> clx cmd 'mysql system -e "call
pdcache_update()"';

shell> clx cmd 'mysql system -e "call qpc_flush()"
';
```

Note that qpc_flush and pdcache_update are done on a per-node basis. Therefore, these should be executed using the clx cmd utility to ensure they are run on all nodes.

Chaining these commands together in the shell as command && command && command is recommended to avoid delays between the completion of the A LTER and flushing of the caches.