ALTER CLUSTER Syntax

```
ALTER CLUSTER ADD 'ip' [, 'ip'] .
..

ALTER CLUSTER SOFTFAIL nodeid [,
nodeid] ...
ALTER CLUSTER UNSOFTFAIL nodeid [
, nodeid] ...
ALTER CLUSTER REFORM

ALTER CLUSTER DROP nodeid

ALTER CLUSTER RESIZE DEVICES size


ALTER CLUSTER nodeid [, nodeid] .
.. ZONE zone

ALTER CLUSTER SET MAX_FAILURES =
number of simultaneous failures

ALTER CLUSTER RELOAD SSL
```

Use ALTER CLUSTER to:

- ALTER CLUSTER ADD (Flex Up): Add node(s) to increase cluster size
- ALTER CLUSTER SOFTFAIL (Flex Down): Remove node(s) from your cluster.
- ALTER CLUSTER DROP: Forcibly remove node(s) from your cluster.
- ALTER CLUSTER RESIZE DEVICES: Increase the size of the file allocated for permanent database storage.
- ALTER CLUSTER ZONE: Assign nodes to a zone.
- ALTER CLUSTER SET MAX_FAILURES: Specify the number of simultaneous failures that can be safely tolerated.
- ALTER CLUSTER RELOAD SSL: re-load SSL keys and certificates

One change may be made at a time. Users must have the SUPER privilege to use ALTER CLUSTER.

ALTER CLUSTER ADD (Flex Up)

```
ALTER CLUSTER ADD '
ip' [, 'ip'] ...
```

Use ALTER CLUSTER ADD to add new node(s) to your cluster. For full instructions, see Expanding Your Cluster's Capacity - Flex Up.

ALTER CLUSTER SOFTFAIL (Flex Down)

```
ALTER CLUSTER SOFTFAIL no
deid [, nodeid] ...
ALTER CLUSTER UNSOFTFAIL
nodeid [, nodeid] ...
ALTER CLUSTER REFORM
```

ALTER CLUSTER SOFTFAIL directs the Rebalancer to move all data from the designated node(s) to active nodes in the cluster ensuring that replicas are preserved.

ALTER CLUSTER UNSOFTFAIL cancels a previous SOFTFAIL request. The node(s) are again made available for use and the Rebalancer will work to relocate data to the node(s).

ALTER CLUSTER REFORM removes softfailed nodes from the cluster and performs a group change. For full instructions see Reducing Your Cluster's Capacity - Flex Down.

Softfailing a Zone

To softfail a zone, mark all the nodes in the zone as softfailed.

ALTER CLUSTER DROP

```
ALTER CLUSTER DROP
nodeid
```

Use ALTER CLUSTER DROP to immediately remove a node from the cluster without ensuring all data has sufficient replicas. This should only be used in emergency situations. Additional information can be found in Administering Failure and Recovery.

ALTER CLUSTER DROP should be used with caution as this operation cannot be undone. Dropping more than one node before reprotect is finished can result in permanent data loss. When possible, use the Flex Down procedure instead of ALTER CLUSTER DROP.

ALTER CLUSTER RESIZE DEVICES

```
ALTER CLUSTER RESIZE DEVICES
size
```

Use ALTER CLUSTER RESIZE DEVICES to expand the device1 file on all online nodes. size is the total calculated bytes or a rounded whole integer suffixed by k/m/g for kilobytes, megabytes, or gigabytes.

```
sql> ALTER CLUSTER RESIZE DEVICES 50g;
```

All device1 files should be the same size cluster-wide and are all modified by the resize command in parallel. This command does not affect the device1-temp file that is used for sorting and grouping large query results. See Managing File Space and Database Capacity for more information.

ALTER CLUSTER RESIZE DEVICES does not support reducing the size of the device1 file. See Decreasing device1 Size for how to perform this operation.

See Managing File Space and Database Capacity for additional information.

Clustrix recommends resizing devices during off-peak hours.

ALTER CLUSTER ZONE

ClustrixDB allows nodes to be grouped into zones to improve fault tolerance, where a zone can be availability zones within the same AWS Region, different server racks, or separate servers in different data centers. Once you have determined your target zone configuration, use ALTER CLUSTER ZONE to assign nodes of your cluster to a zone.

```
ALTER CLUSTER
nodeid [,
nodeid] ...
ZONE zone
```

Assigning a node to a zone allows ClustrixDB to have more options for fault tolerance. By ensuring that replicas are placed across zones, no data is lost if a zone becomes unavailable.

After all nodes are assigned to a zone, verify that there is an equal number of nodes in each zone and that no nodes are assigned to zone 0. Clustrix supports configuring a minimum of 3 and a maximum of 5 zones.

```
sql> SELECT * FROM system.nodeinfo ORDER BY zone;
```

Changes to zone configurations do not take effect until a group change or ALTER CLUSTER REFORM. This will cause a group change and an interruption in service. See Zones for more information.

```
sql> ALTER CLUSTER
REFORM;
```

If you no longer wish to use zones, simply assign all nodes to zone 0 using ALTER CLUSTER ... ZONE followed by `ALTER CLUSTER REFORM;`

Clustrix recommends allocating enough disk space so that should a zone fail, there is sufficient space to reprotect. See Allocating Disk Space for Fault Tolerance and Availability.

ALTER CLUSTER SET MAX_FAILURES

The max_failures global variable determines the number of failures that can occur **simultaneously** while ensuring that no data is lost. By default, this is the number of node failures that can be tolerated. If zones are in use, this is the number of node or zone failures tolerated. For example, if MAX_FAILURES = 1 (default), the cluster can lose one node or one zone, regardless of the number of nodes in that zone. The value of max_failures is also used to determine the number of replicas created by default for a table or index. If MAX_FAILURES = 1, new database entities are created with REPLICAS = 2. See MAX_FAILURES for additional information.

max_failures is a read-only global variable that can only be set using ALTER CLUSTER.

Name	Description	Default Value
max_failures	Number of simultaneous failures that the cluster can withstand while maintaining transaction resolution and without suffering data loss.	1

Change the Value of MAX_FAILURES

Increasing the value for max_failures increases the number of replicas created, which can have a significant performance impact to writes and requires additional disk space. Due to the high performance overhead, Clustrix does not recommend exceeding MAX_FAILURES = 2.

To change the value of the global max_failures, perform the steps outlined below. In this sample, the number of allowable failures is modified from the default of 1 to 2. This will cause all new tables and indexes to be created with REPLICAS = 3.

Step 1: Ensure there is sufficient disk space

Ensure that the cluster has sufficient disk space for additional replicas. See Allocating Disk Space for Fault Tolerance and Availability for more details on how to determine disk space required.

Step 2: Set the Cluster-Wide Failure Threshold

```
sql>  ALTER CLUSTER SET MAX_FAILURES = 2;
```

Running this command will result in a group change and an interruption in service.

Step 3: Alter Existing Tables

Tables created after the value for max_failures has been updated will automatically have sufficient replicas. However, tables created before max_failures was updated may not have sufficient replicas and need to be altered. This query generates ALTER statements for all representations that are under-protected.

```
sql> SELECT concat ('ALTER TABLE ',`database`, '.', `Table`, ' REPLICAS = MAX_FAILURES
+ 1;')
     FROM system.table_replicas
     where database not in ('system', 'clustrix_dbi', 'clustrix_statd', '_replication')
     GROUP BY `table`, `database`
          having (count(1) / count(distinct slice)) < MAX_FAILURES + 1;
```

The resulting SQL will look like:

```
sql> ALTER TABLE foo REPLICAS = 3;
```

Run the generated script and monitor the Rebalancer as it creates additional replicas. See Managing the Rebalancer.

Log Messages

When the value for max_failures is modified, you will see an entry in clustrix.log that notes the number of failures that are configured:

```
INFO tm/gtm_resolve.c:168 gtm_r_validate_paxos_f(): group 1cfffe supports 2 simultaneous failures
```

ALTER CLUSTER RELOAD SSL

To configure ClustrixDB to use encrypted connections:

Create keys and certificates (using your choice of method) and copy them as the root user to every node:

```
shell> scp server-cert.pem root@hostname:/data/clustrix
shell> scp server-key.pem root@hostname:/data/clustrix
```

On each node, transfer ownership of those files to the clxd user:

```
shell> sudo chown clxd server-*.pem
```

Certificates and keys must be in the same location on every node.

Configure ClustrixDB to use these certificates, keys, and SSL:

```
sql> SET GLOBAL ssl_cert = '/data/clustrix/server-cert.pem';
sql> SET GLOBAL ssl_key = '/data/clustrix/server-key.pem';
sql> ALTER CLUSTER RELOAD SSL;
sql> SET GLOBAL ssl_enabled = TRUE;
```

ALTER CLUSTER RELOAD SSL validates the location of the certificates and keys. If this command fails, the clustrix.log may include more detail.