

# RESETTING THE QUORUM


Occasionally, you may find your nodes no longer consider themselves part of the [Primary Component](#). For instance, in the event of a network failure, the failure of more than half of the cluster, or a split-brain situation. In these cases, the node come to suspect that there is another Primary Component, to which they are no longer connected.

When this occurs, all nodes return an Unknown command error to all queries. You can check if this is happening using the [wsrep\\_cluster\\_status](#) status variable. Run the following query on each node:

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_status';
```

Variable_name	Value
wsrep_cluster_status	Primary

The return value Primary indicates that it the node is part of the Primary Component. When the query returns any other value it indicates that the node is part of a nonoperational component. If none of the nodes return the value Primary, it means that you need to reset the quorum.

**Note:** Bear in mind that situations where none of the nodes show as part of the Primary Component are very rare. In the event that you do find one or more nodes that return the value Primary, this indicates an issue with network connectivity rather than a need to reset the quorum. Troubleshoot the connection issue. Once the nodes regain network connectivity they automatically resynchronize with the Primary Component.

## FINDING THE MOST ADVANCED NODE

Before you can reset the quorum, you need to identify the most advanced node in the cluster. That is, you must find the node whose local database committed the last transaction. Regardless of the method you use in resetting the quorum, this node serves as the starting point for the new [Primary Component](#).

Identifying the most advanced node in the cluster requires that you find the node with the most advanced sequence number, or seqno. You can determine this using the [wsrep\\_last\\_committed](#) status variable.

From the database client on each node, run the following query:

```
SHOW STATUS LIKE 'wsrep_last_committed';
```


Variable_name	Value
wsrep_last_committed	409745

The return value is the seqno for the last transaction the node committed. The node that provides the highest seqno is the most advanced node in your cluster. Use it as the starting point in the next section when bootstrapping the new Primary Component.

## RESETTING THE QUORUM

When you reset the quorum what you are doing is bootstrapping the [Primary Component](#) on the most advanced node you have available. This node then functions as the new Primary Component, bringing the rest of the cluster into line with its state.

There are two methods available to you in this process: automatic and manual.

**Note:** The preferred method for a quorum reset is the automatic method. Unlike the manual method, automatic bootstraps preserve the write-set cache, or GCache, on each node. What this means is that when the new Primary Component starts, some or all of the joining nodes can provision themselves using the [Incremental State Transfer](#) (IST) method, rather than the much slower [State Snapshot Transfer](#) (SST) method.

### Automatic Bootstrap

Resetting the quorum bootstraps the [Primary Component](#) onto the most advanced node. In the automatic method this is done by enabling [pc.bootstrap](#) under [wsrep\\_provider\\_options](#) dynamically through the database client. This makes the node a new Primary Component.

To perform an automatic bootstrap, on the database client of the most advanced node, run the following command:

```
SET GLOBAL wsrep_provider_options='pc.bootstrap=YES';
```

The node now operates as the starting node in a new Primary Component. Nodes in nonoperational components that have network connectivity attempt to initiate incremental state transfers if possible, state snapshot transfers if not, with this node,

bringing their own databases up-to-date.

## Manual Bootstrap

Resetting the quorum bootstraps the [Primary Component](#) onto the most advanced node. In the manual method this is done by shutting down the cluster, then starting it up again beginning with the most advanced node.

To manually bootstrap your cluster, complete the following steps:

1. Shut down all cluster nodes. For servers that use `init`, run the following command from the console:

```
# service mysql stop
```

For servers that use `systemd`, instead run this command:

```
# systemctl stop mysql
```

2. Start the most advanced node with the `--wsrep-new-cluster` option. For servers that use `init`, run the following command:

```
# service mysql start --wsrep-new-cluster
```

For servers that use `systemd` and Galera Cluster 5.5 or 5.6, instead run this command:

```
# systemctl start mysql --wsrep-new-cluster
```

For servers that use `systemd` and Galera Cluster 5.7, use the following command:

3. Start every other node in the cluster. For servers that use `init`, run the following command:

```
# service mysql start
```

For servers that use `systemd`, instead run this command:

```
# systemctl start mysql
```

When the first node starts with the `--wsrep-new-cluster` option, it initializes a new cluster using the data from the most advanced state available from the previous cluster. As the other nodes start they connect to this node and request state snapshot transfers, to bring their own databases up-to-date.