KOENIG
step forward

Causal Clustering in Neo4J

What is Clustering?

Enables to utilize a Neo4j database in production

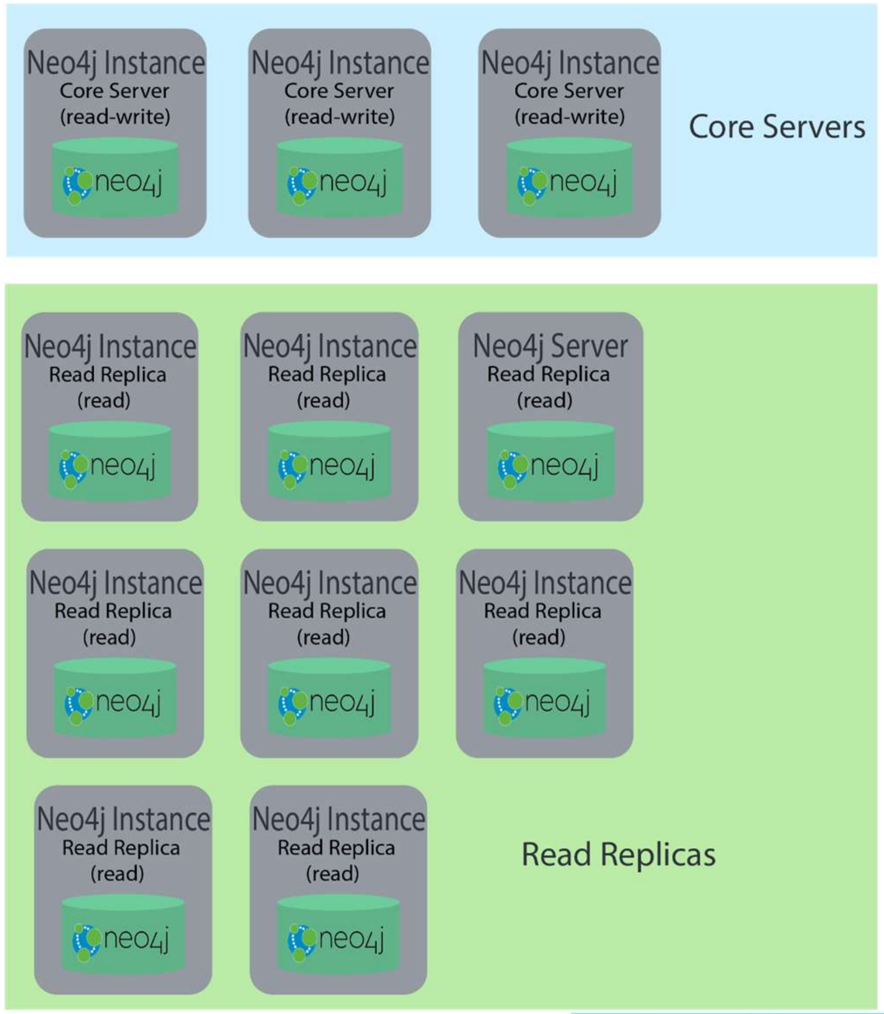
Provides

- High-availability
- High-scalability

What is Causal Clustering?

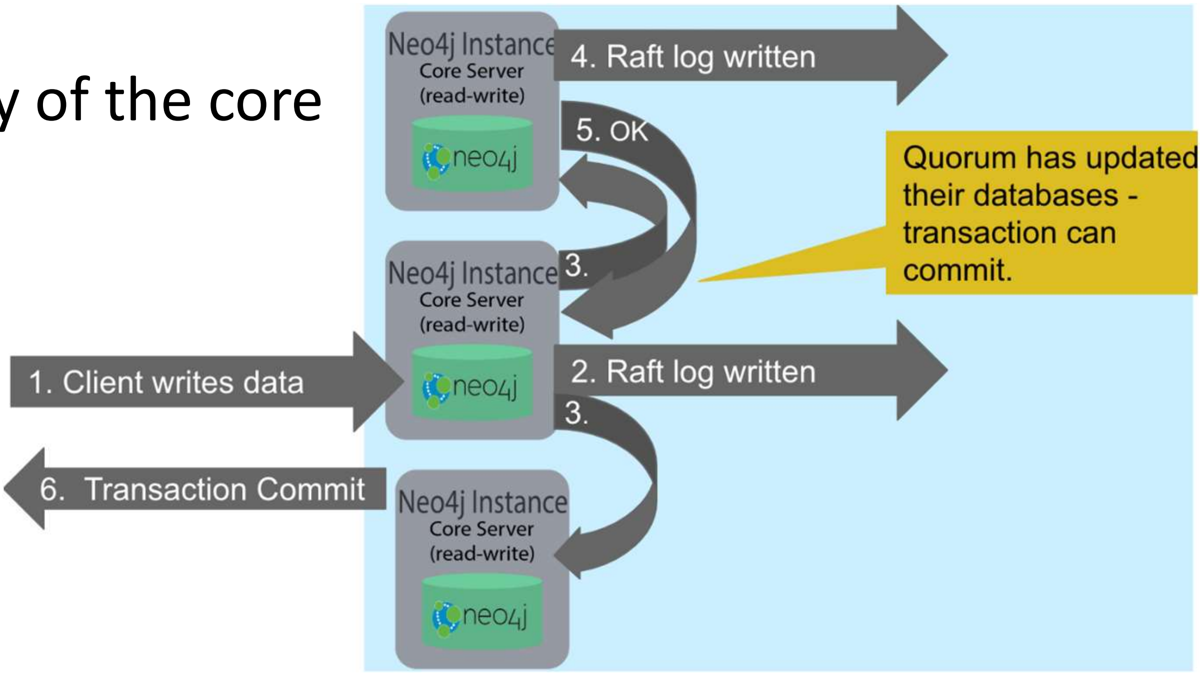
- Consistency model used in distributed computing
- Ensures that causally related operations are seen by every instance in the system in the same order
- Allow clients to treat them as a single (logical) server.
- Makes it possible
    - To write to Core Servers and
    - Read those writes from Read Replica

Cluster architecture



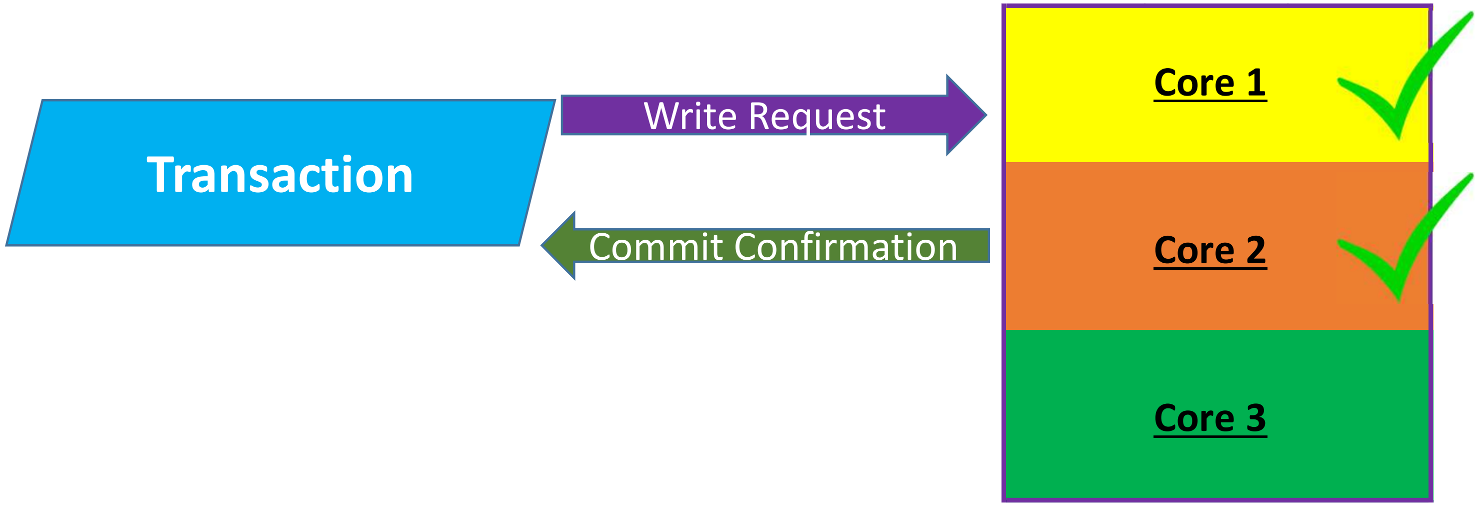
Core servers

- For read & write access
- To synchronize updates
- Transaction is committed if a majority of the core servers have written the data
- Implemented using the Raft protocol
- More core servers
  - Longer a “majority” commit will take



Core servers

- Safeguards data
- Replicate transactions using Raft
- If majority of Core Servers (N/2+1) have accepted the transaction, acknowledge the commit

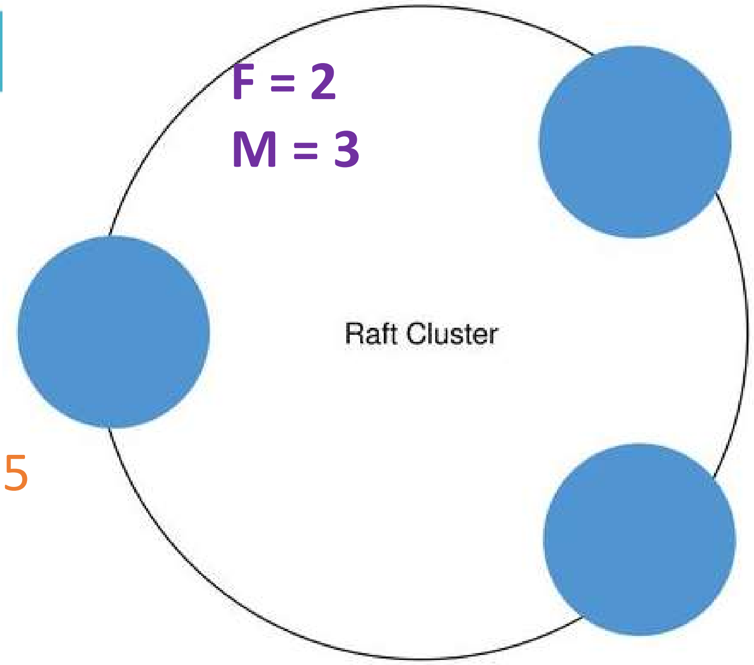


N = 3

N/2+1 = 3/2 + 1 = 1+1 = 2

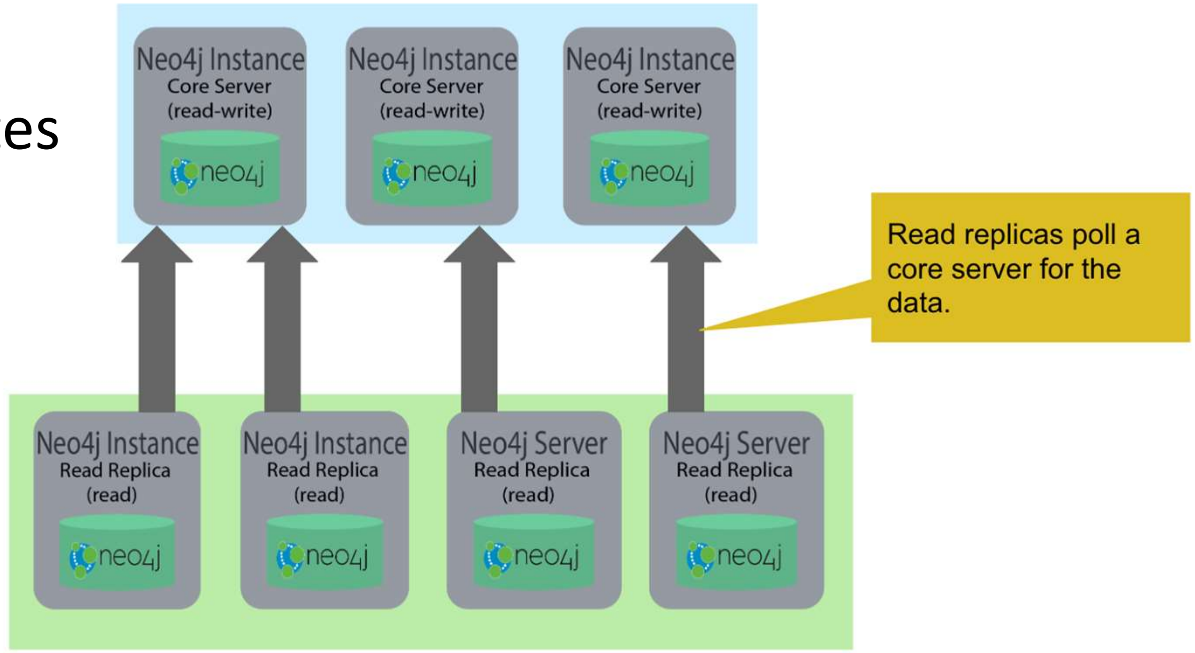
Core servers - Fault Tolerance

- This is calculated with formula: $M = 2F + 1$
  - M: # of Core Servers
  - F: Max # of faults which can be tolerated by cluster
- Example:
  - To tolerate max of 2 failed (F) Core Servers, need to deploy 5 Cores cluster: M:5 = 2 * F:2 +1
  - The smallest fault tolerant cluster must have 3 Cores
    - M:3 = 2 * F:1 +1
- Possible to create Cluster consisting of only two Cores
  - Will not be fault-tolerant
  - If any one server fails, the remaining server will become read-only

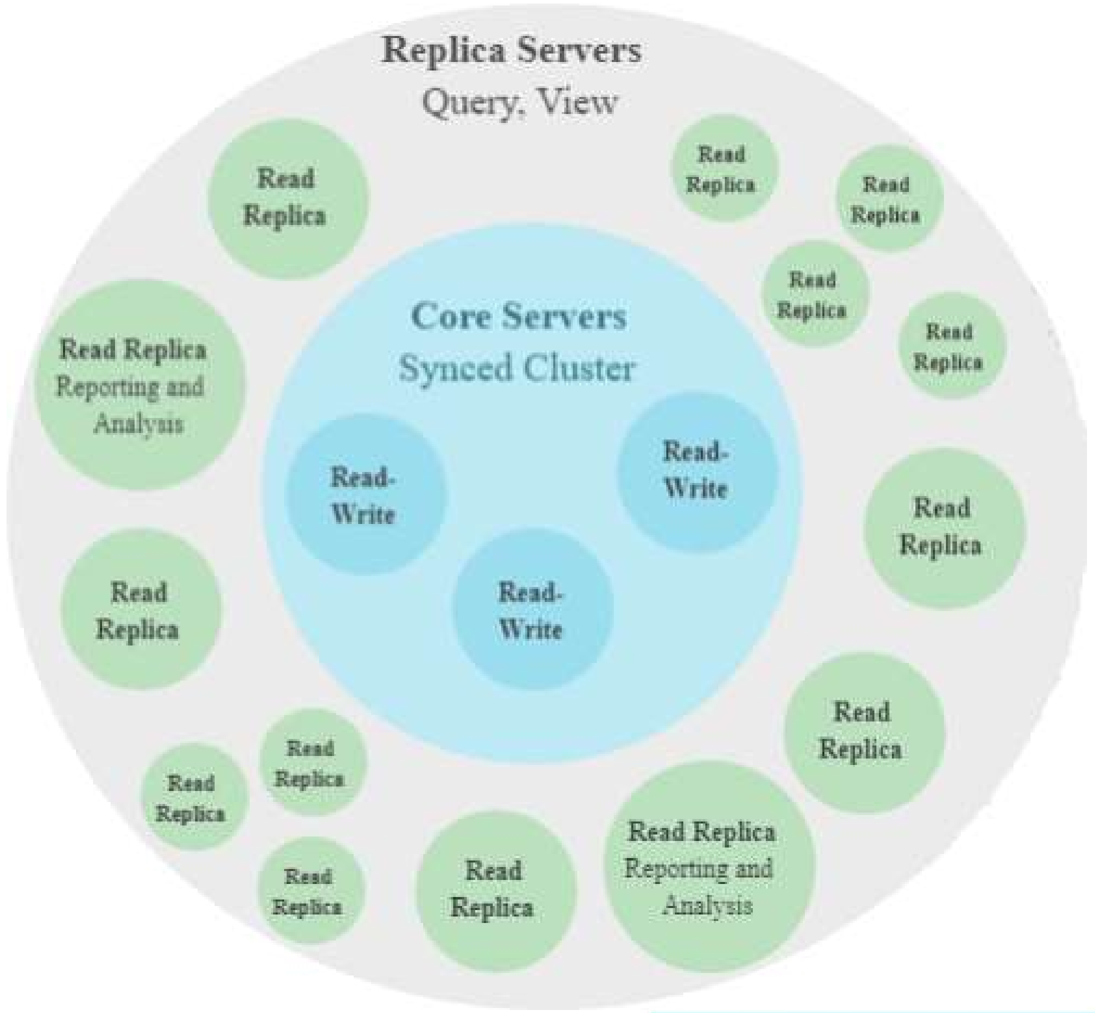


Read replica servers

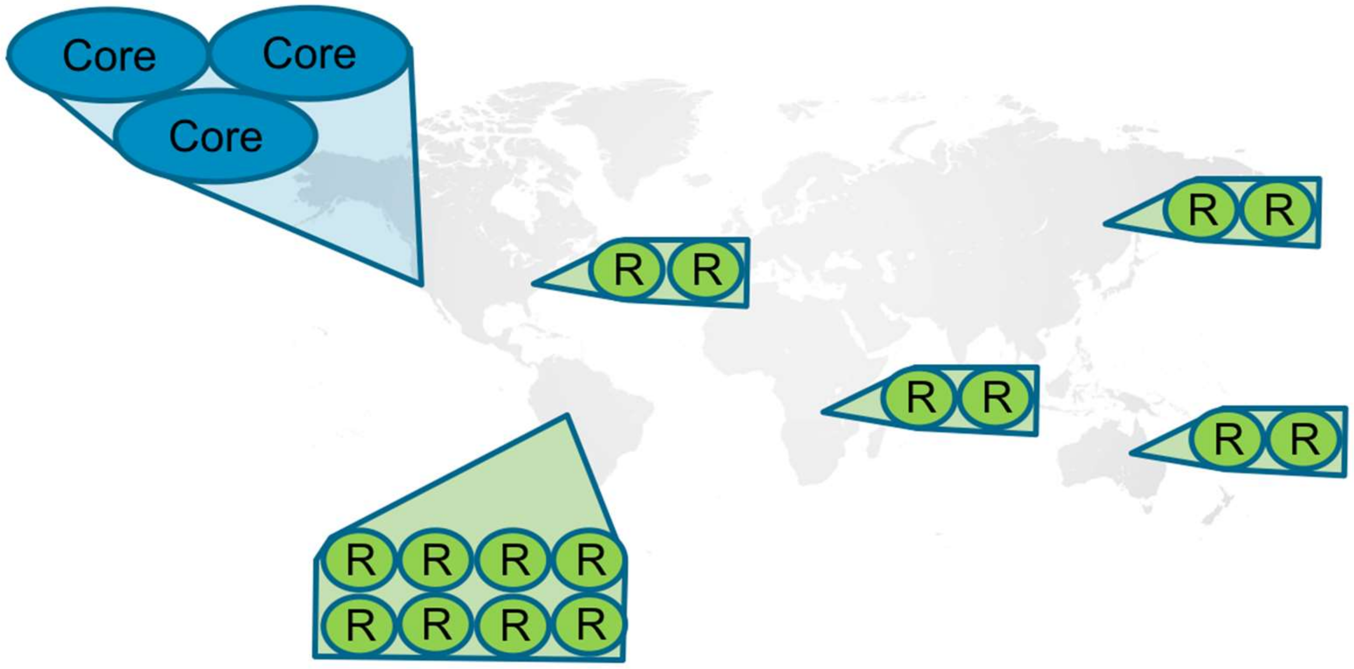
- To scale data across a distributed network
- Only read access to the data
- Regularly poll Core servers for updates
- Distributed cache of the database
- If a read replica fails
  - A new read replica can be started
  - No impact on the data



Causal Cluster Architecture



Distributed architecture

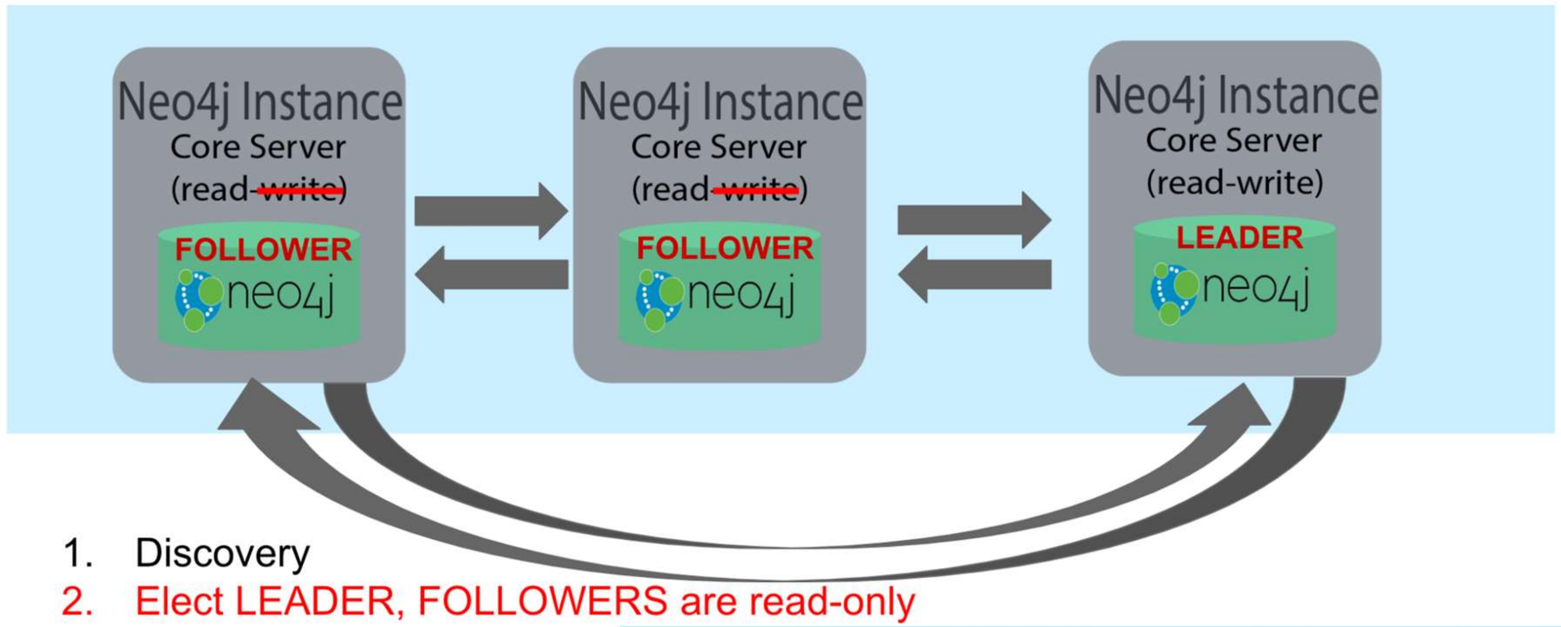


Configure clustering

- By updating the neo4j.conf file on each server
- Types of properties to configure for a cluster:
  - Core or read replica server?
  - Public address for the server?
  - Domain name of the servers in the core server membership?
  - Ports used for communicating between the members?
  - Published ports for bolt, http, https?
  - # of core servers in the cluster?

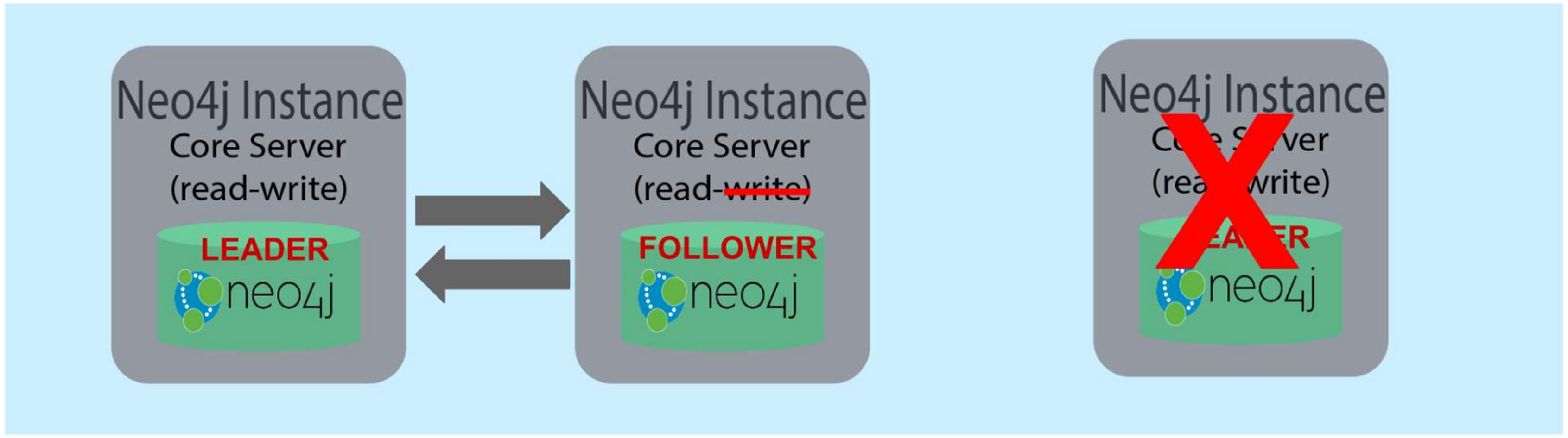
Core server startup

- When a core server starts, it first uses a discovery protocol to join the network
- Exactly one core server is elected to be the LEADER
- LEADER is coordinator of all communication
- All other core servers are FOLLOWERS



Core server shutdown

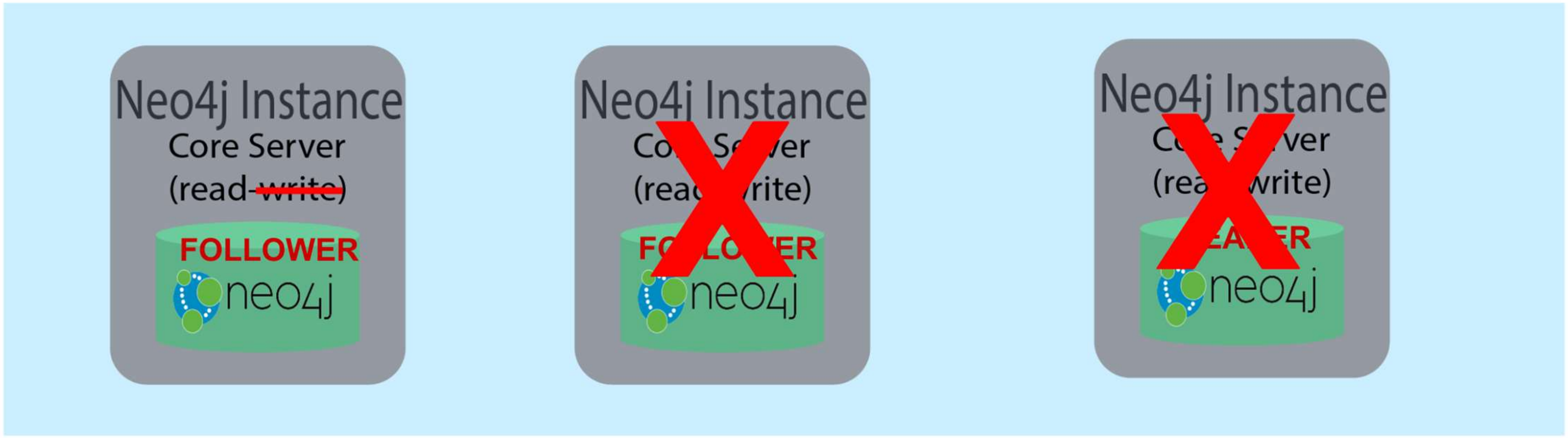
- FOLLOWER
  - LEADER incorporates information into its operations with the other core servers
- LEADER
  - Remaining core servers communicate with each other to elect LEADER



1. Server shutdown
2. Elect new LEADER

Cluster below quorum

- The LEADER becomes inoperable for writing to the database
- This is a serious matter that needs to be addressed by administrator



1. Server shutdown
2. Below quorum - no LEADER, cluster inoperable for writes

Core server updates database

- A core server updates its database based upon the requests from clients
- The client’s transaction is not complete until a quorum of core servers have updated their databases
- Subsequent to the completion of the transaction, the remaining core servers will also be updated
- The Raft protocol to share updates
- Application use bolt+routing protocol
  - With this protocol, applications can write to any core server in the cluster, but the LEADER will always coordinate updates

Administrative tasks for clustering

Modify the neo4j.conf files for each core server.

Start the core servers in the cluster.

Seed the core server (add initial data).

Ensure each core server has the data.

Modify the neo4j.conf files for each read replica server.

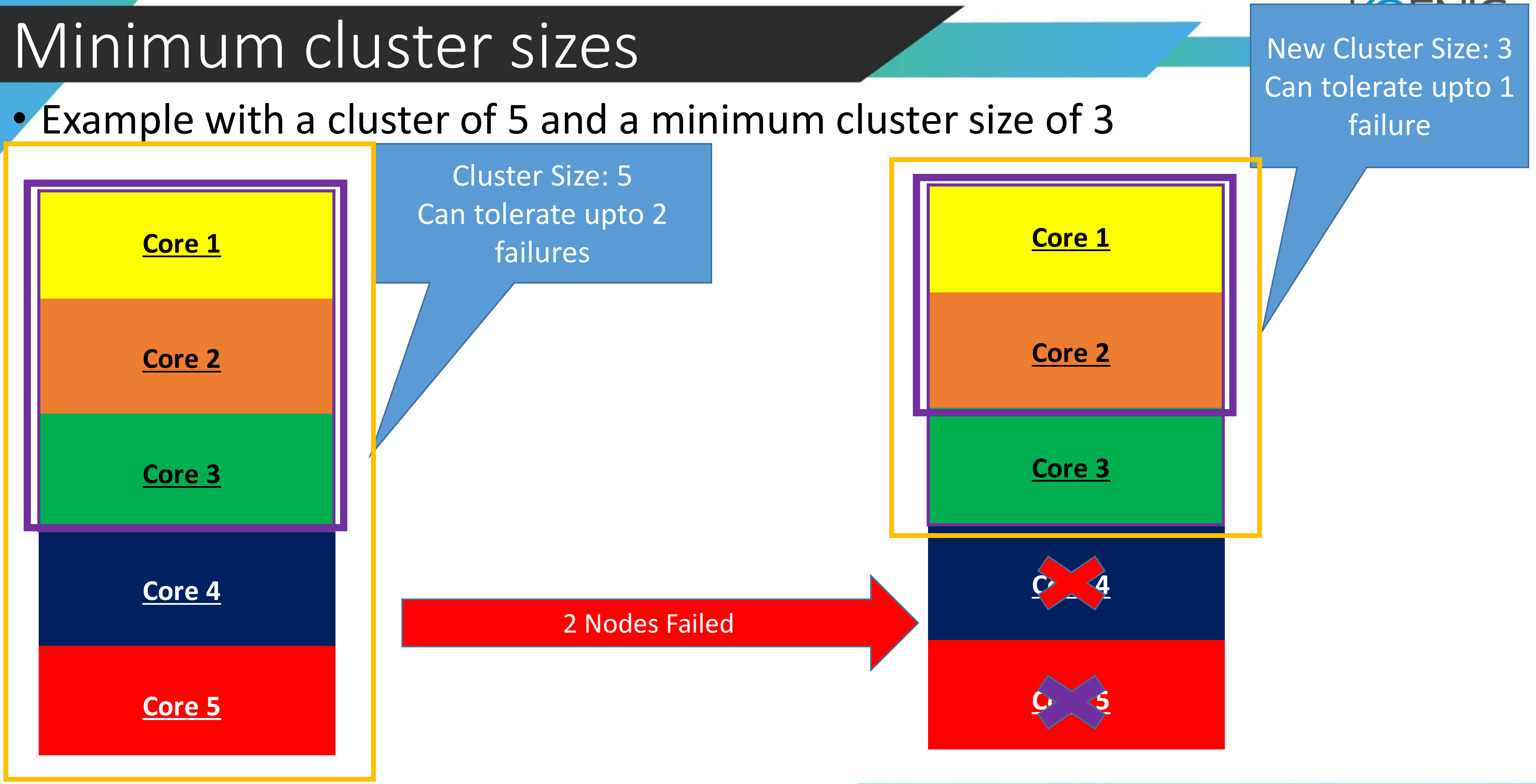
Start the read replica servers.

Ensure each read replica server has the data.

Test updates to the database.

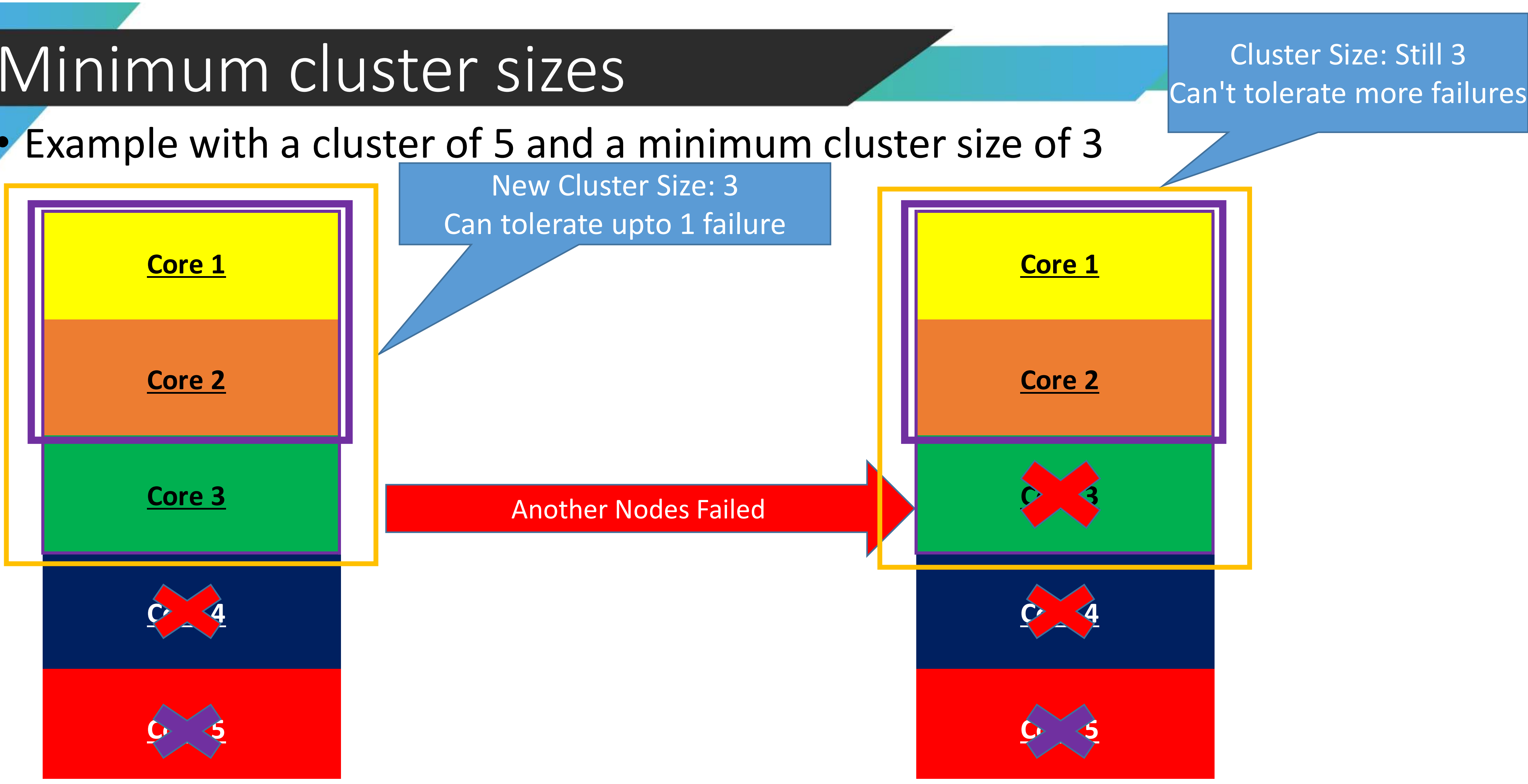
Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3



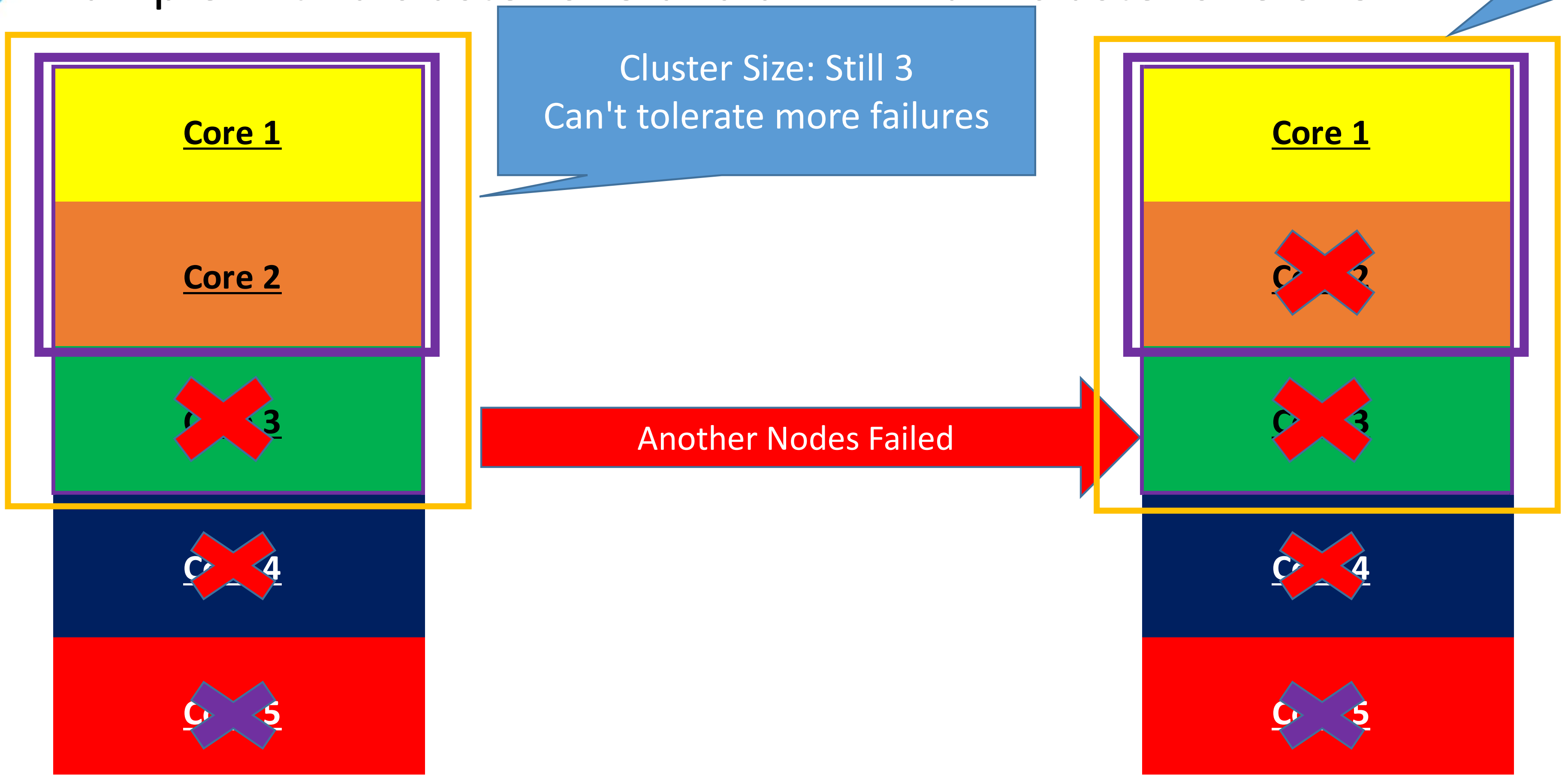
Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3



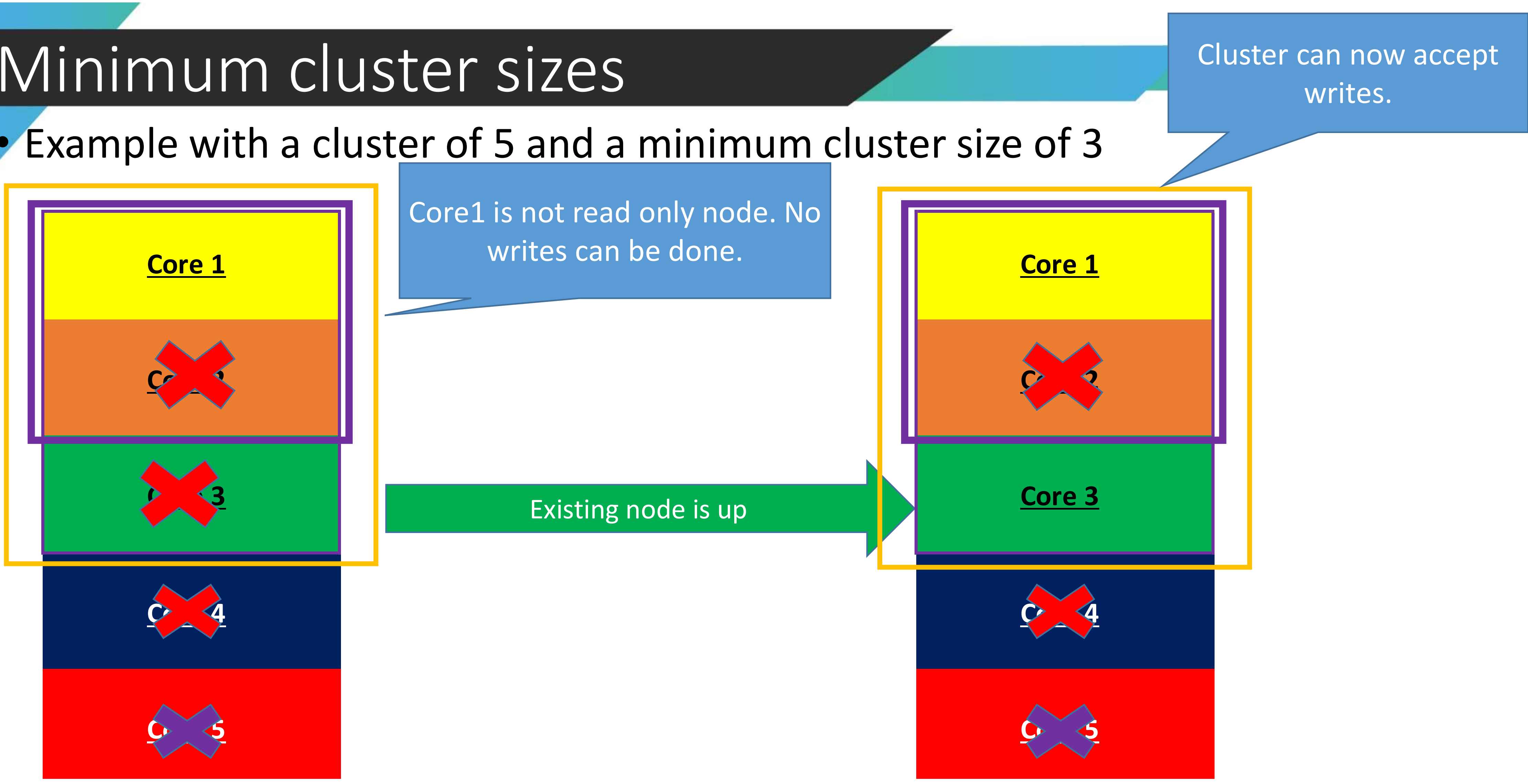
Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3



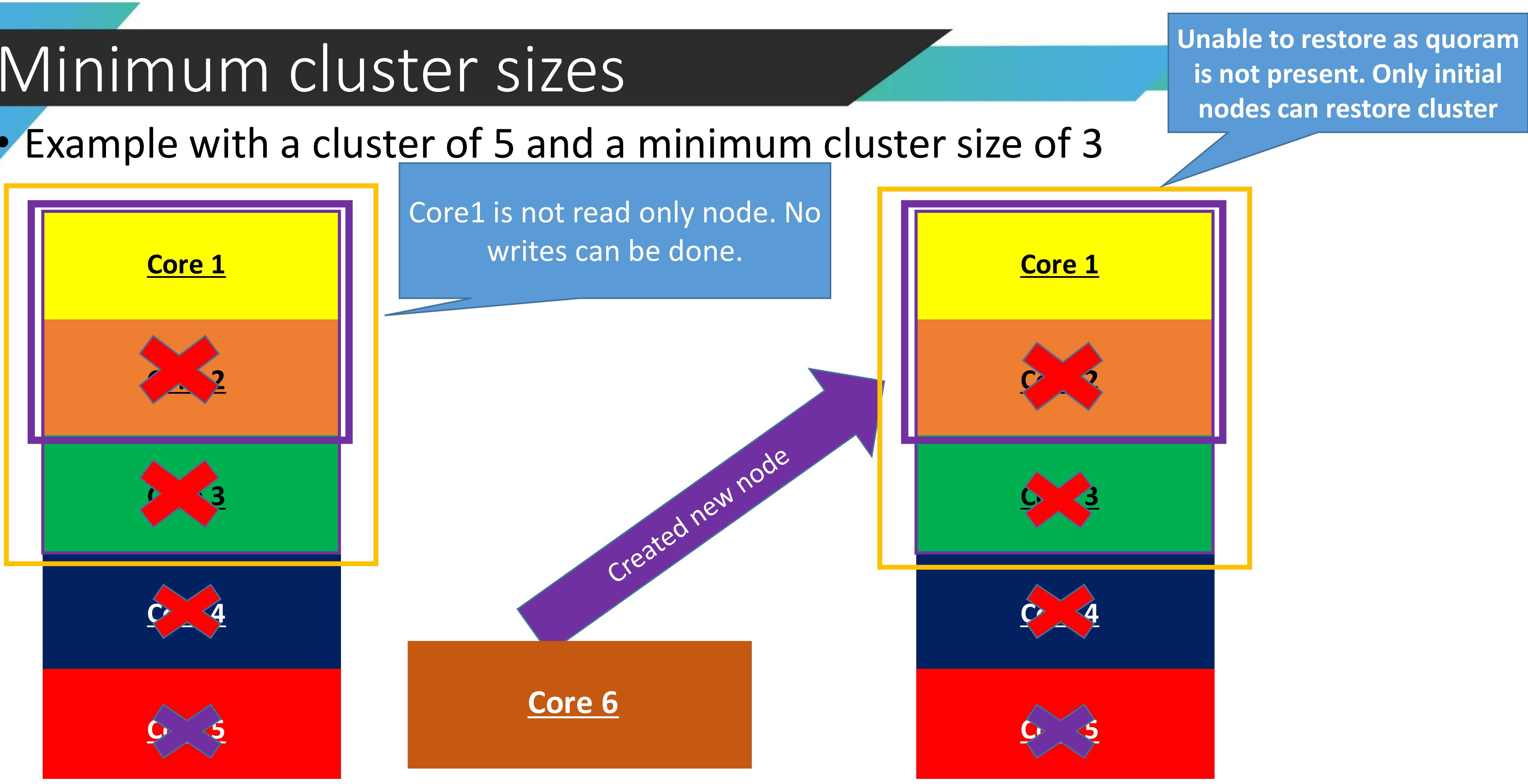
Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3


Core1 is not read only node. No writes can be done.
Cluster can now accept writes.
Core 1
Core 3
Existing node is up
Core 1
Core 3

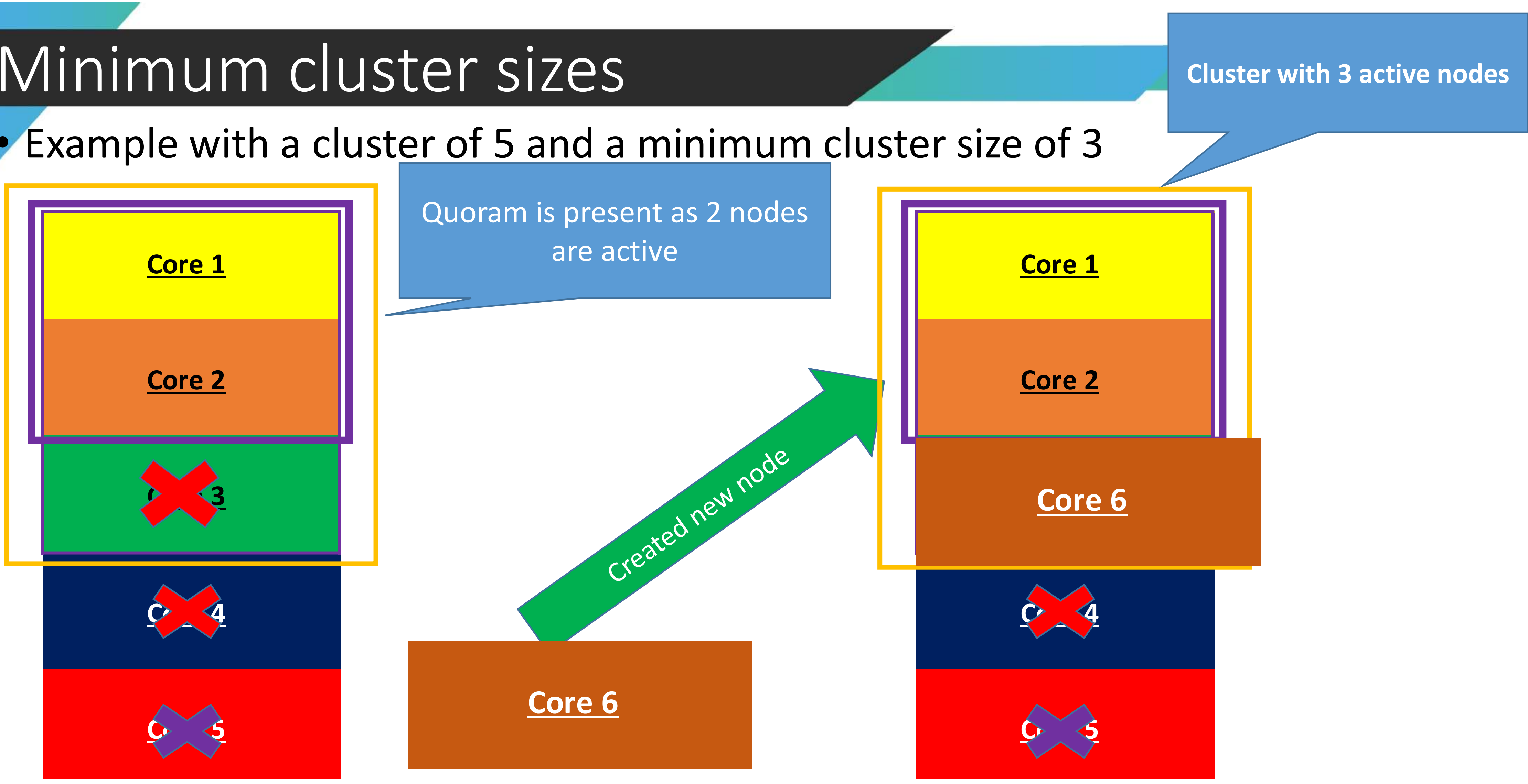
Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3


Unable to restore as quoram is not present. Only initial nodes can restore cluster
Core1 is not read only node. No writes can be done.
Core 1
2
3
4
5
Core 6
Created new node
Core 1
C 2
C 3
C 4
C 5

Minimum cluster sizes

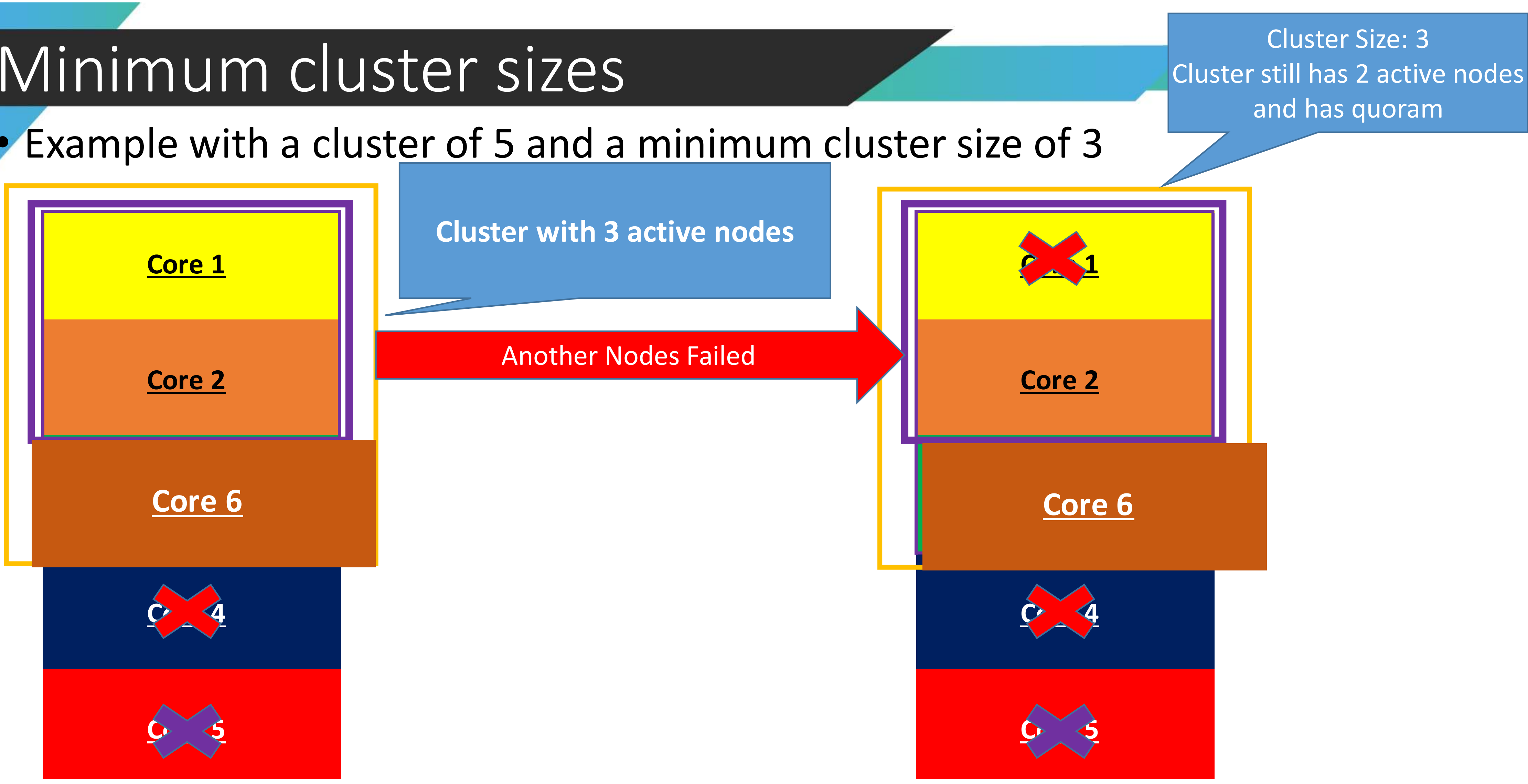
- Example with a cluster of 5 and a minimum cluster size of 3



Cluster with 3 active nodes
Quoram is present as 2 nodes are active
Core 1
Core 2
3
4
5
Created new node
Core 6
Core 1
Core 2
Core 6
4
5

Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3



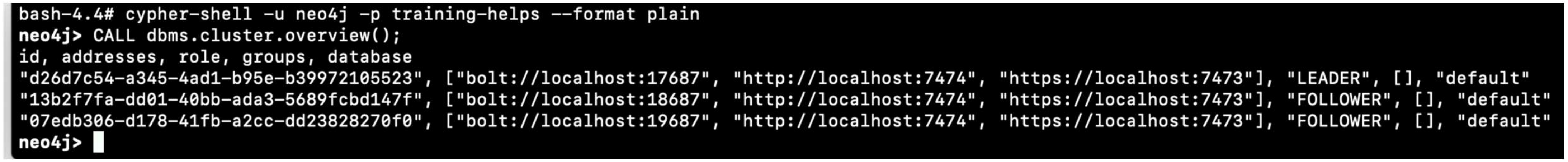
Starting the core servers

- Doesn’t matter what order they are started
- One of the members of the core group will automatically be elected as LEADER.

Viewing status of the cluster

cypher-shell -u neo4j -p secret

CALL dbms.cluster.overview();



```
bash-4.4# cypher-shell -u neo4j -p training-helps --format plain
neo4j> CALL dbms.cluster.overview();
id, addresses, role, groups, database
"d26d7c54-a345-4ad1-b95e-b39972105523", ["bolt://localhost:17687", "http://localhost:7474", "https://localhost:7473"], "LEADER", [], "default"
"13b2f7fa-dd01-40bb-ada3-5689fcbd147f", ["bolt://localhost:18687", "http://localhost:7474", "https://localhost:7473"], "FOLLOWER", [], "default"
"07edb306-d178-41fb-a2cc-dd23828270f0", ["bolt://localhost:19687", "http://localhost:7474", "https://localhost:7473"], "FOLLOWER", [], "default"
neo4j>
```

Configure a cluster

Refer hands-on section

KOENIG
step forward

Seeding data for Cluster

Seeding data for Cluster

- Each Neo4j instance has its own database
- Before seed, must unbind from cluster
  - neo4j stop
  - neo4j-admin unbind --verbose

Loading the data

- 3 Options
  - Restore data using an online backup.
  - Load data using an offline backup.
  - Create data using the import tool

- For relatively small (less than 10M nodes) can also load .csv data into one running core server
  - This will propagate the data to all databases in the cluster to other core servers
- For large data, need to
  - First Import data in Standalone Neo4J server
  - Take backup of the database
  - Load database in all Core Servers in the cluster