Causal Clustering in Neo4J

What is Clustering?

Enables to utilize a Neo4j database in production

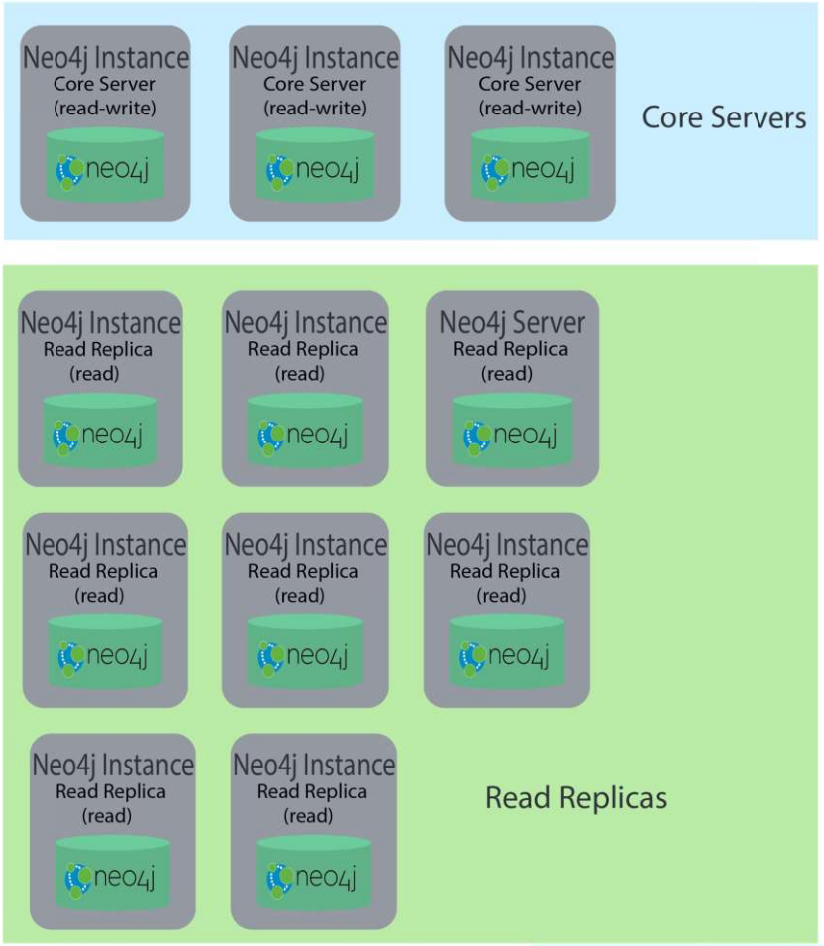
Provides

- High-availability
- High-scalability

What is Causal Clustering?

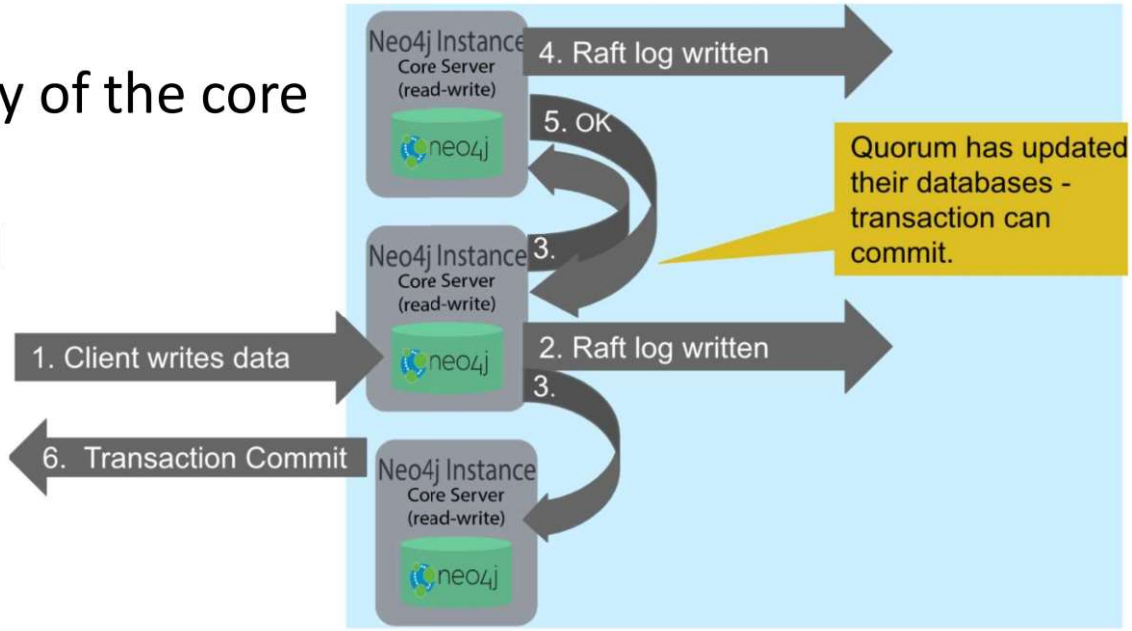
- Consistency model used in distributed computing
- Ensures that causally related operations are seen by every instance in the system in the same order
- Allow clients to treat them as a single (logical) server.
- Makes it possible
  - To write to Core Servers and
  - Read those writes from Read Replica

Cluster architecture



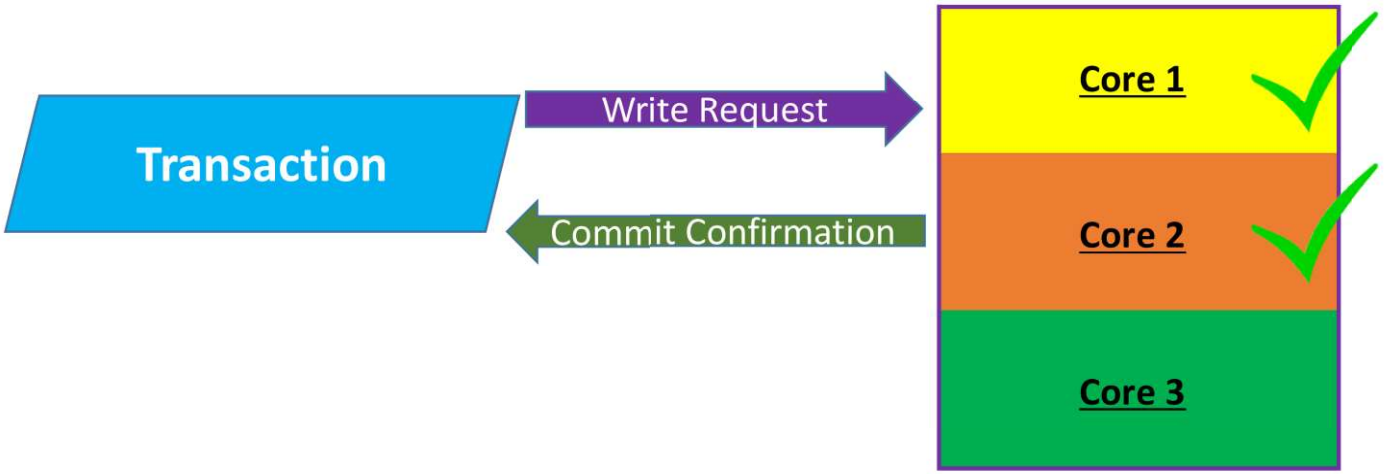
Core servers

- 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

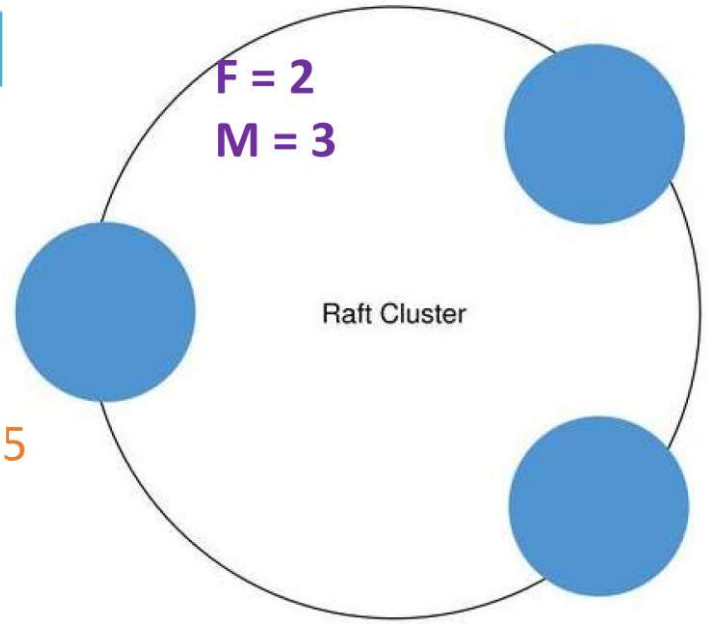


N = 3

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

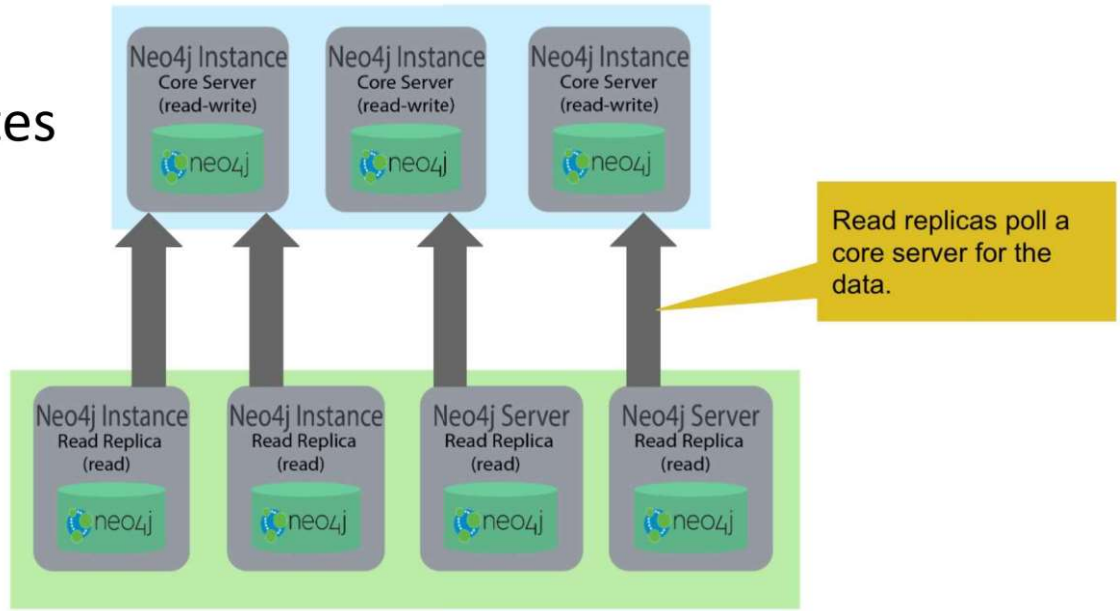
Core servers - Fault Tolerance

- 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

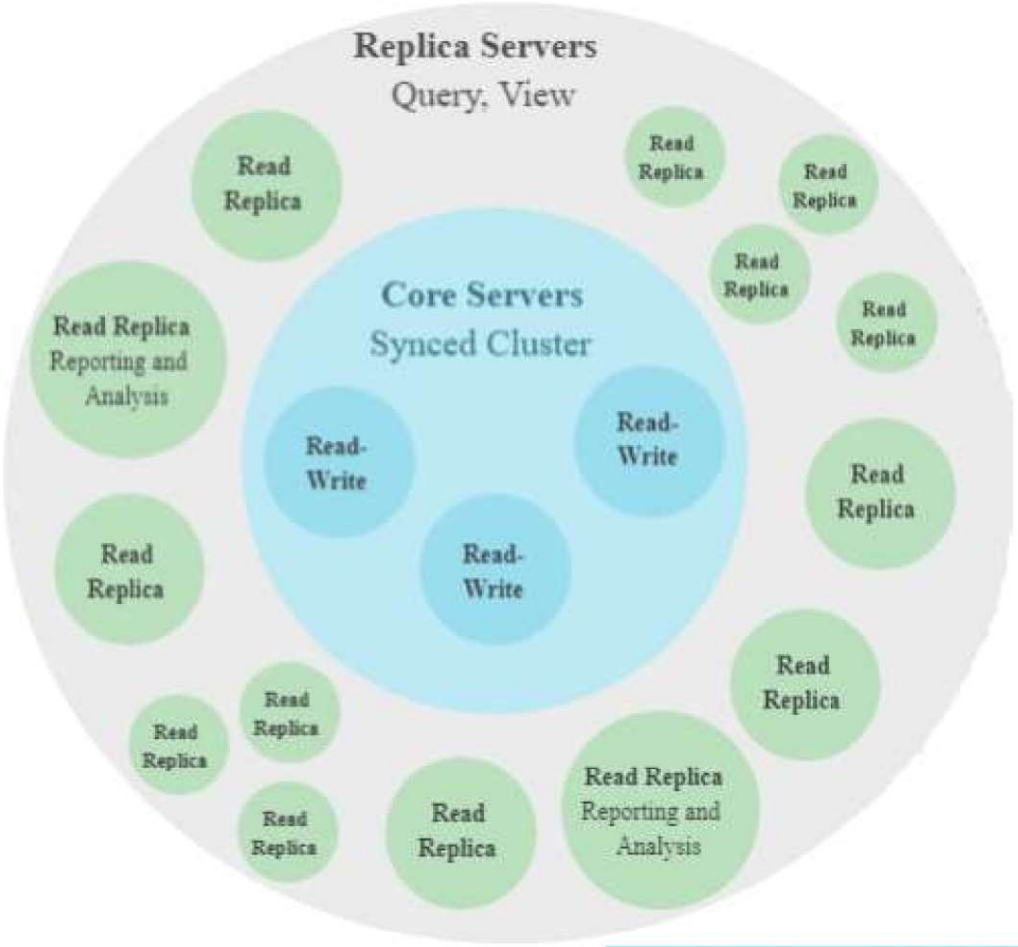


Read replica servers

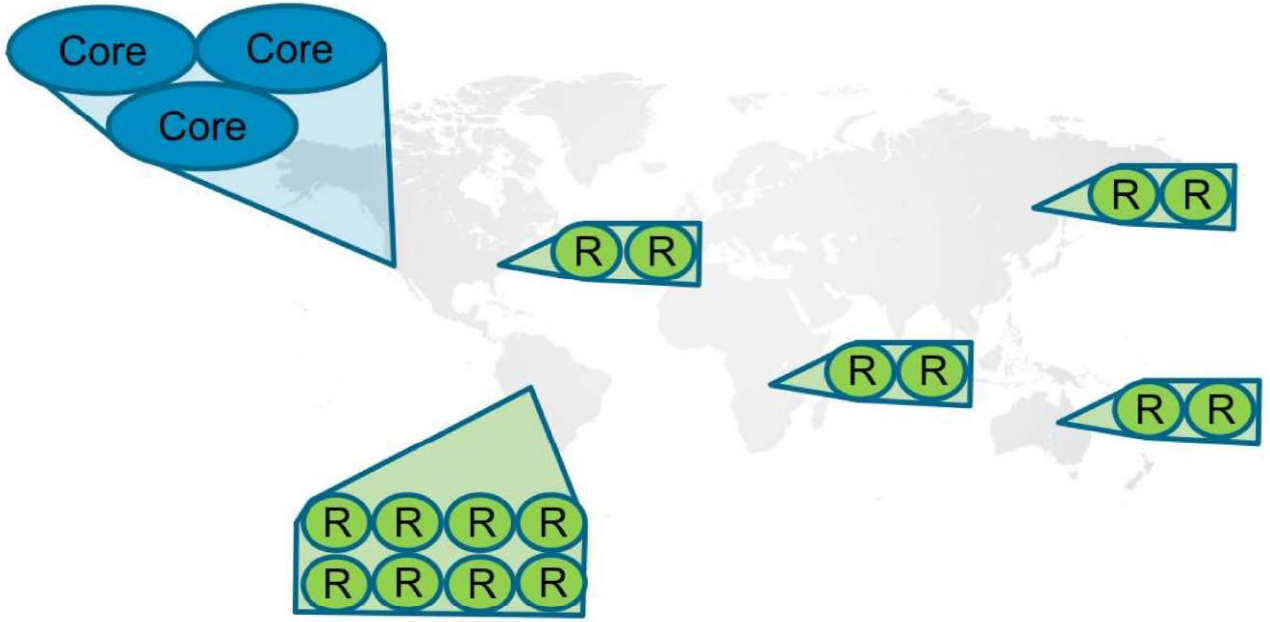
- To scale data across a distributed network
- Only read access to the data
- Regularly poll Core servers for updates
- Distributed cache of the database
- If a read replica fails
  - A new read replica can be started
  - No impact on the data



Causal Cluster Architecture



Distributed architecture

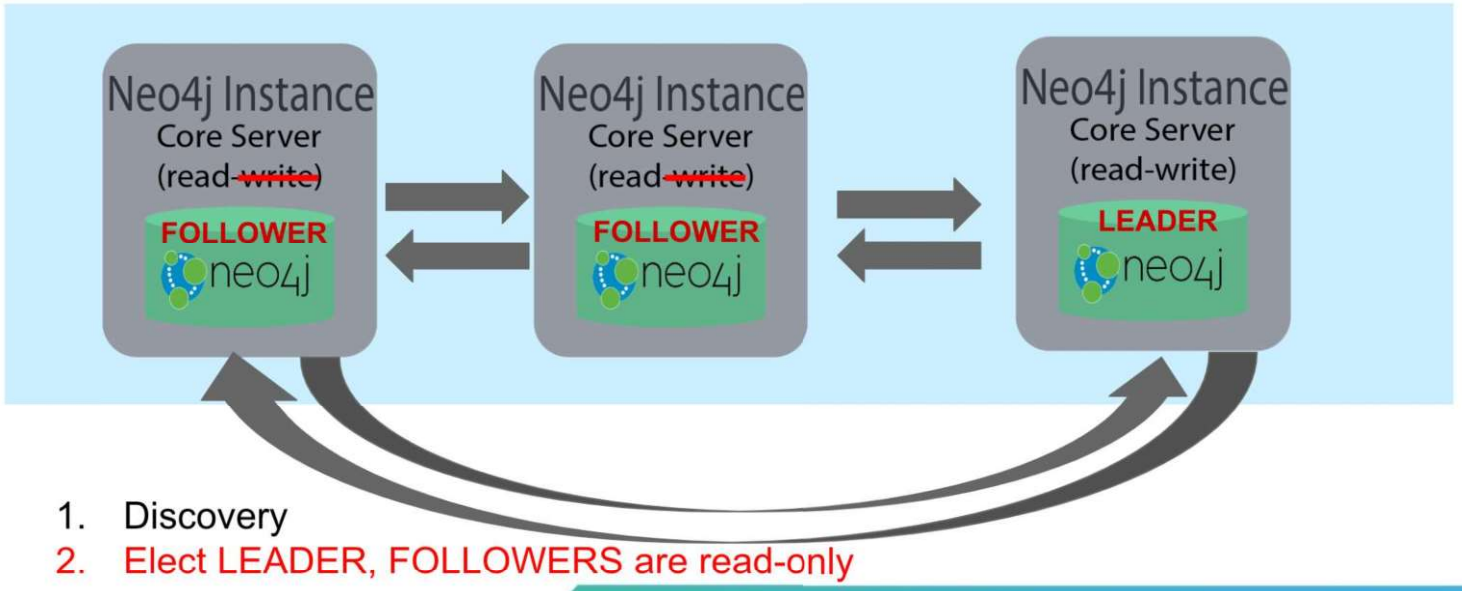


Configure clustering

- 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



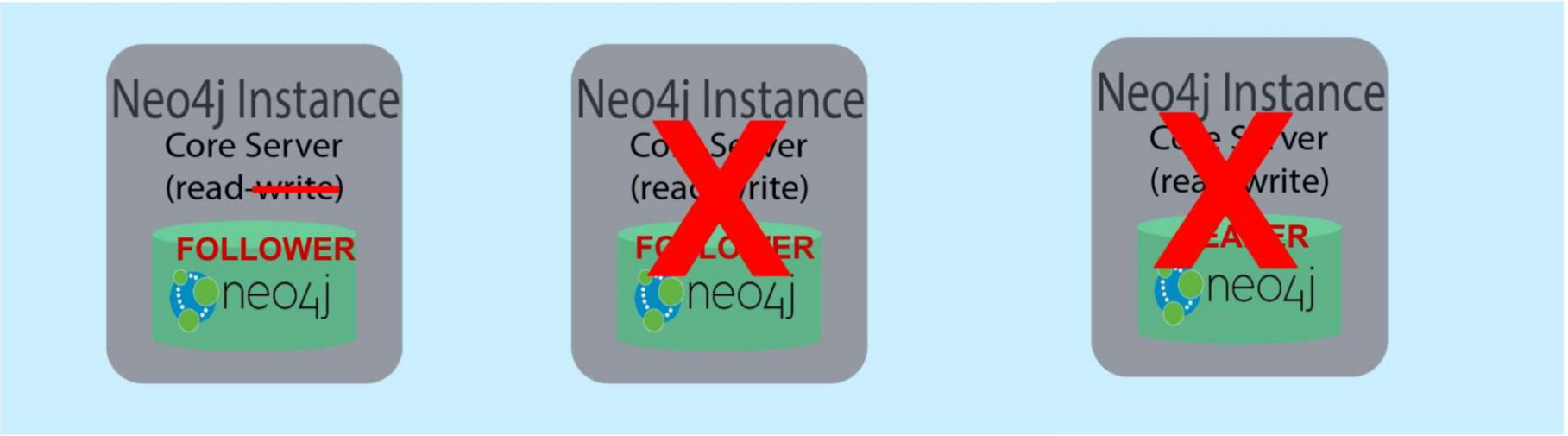
Core server shutdown

- 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.

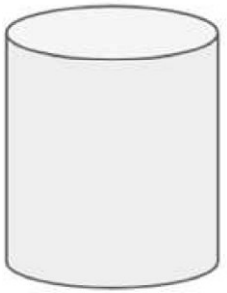
Configuring core servers

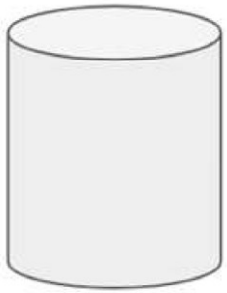
- Identify core servers
- Specify cluster membership
- Configuration properties
- Minimum cluster sizes
- Starting the core servers
- Viewing the status of the cluster

Identify core servers

- Server 1: vm1
- dbms.connector.http.advertised_address=vm1:7474
- dbms.connector.bolt.advertised_address=vm1:7687
- causal_clustering.transaction_advertised_address=vm1:6000
- causal_clustering.raft_advertised_address=vm1:7000
- causal_clustering.discovery_advertised_address=vm1:5000

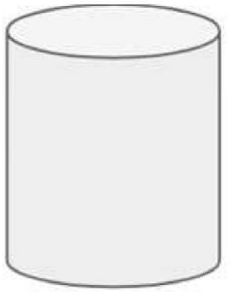
- causal_clustering.initial_discovery_members=
- vm1:5000, vm2:5000, vm3:5000



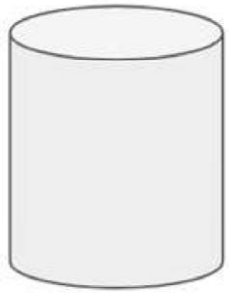


Identify core servers

- Server 2: vm2
- dbms.connector.http.advertised_address=vm2:7474
- dbms.connector.bolt.advertised_address=vm2:7687
- causal_clustering.transaction_advertised_address= vm2:6000
- causal_clustering.raft_advertised_address= vm2:7000
- causal_clustering.discovery_advertised_address= vm2:5000

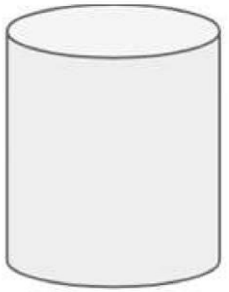


- causal_clustering.initial_discovery_members=
- vm1:5000, vm2:5000, vm3:5000

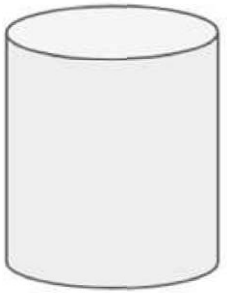


Identify core servers

- Server 3: vm3
- dbms.connector.http.advertised_address=vm3:7474
- dbms.connector.bolt.advertised_address=vm3:7687
- causal_clustering.transaction_advertised_address=vm3:6000
- causal_clustering.raft_advertised_address=vm3:7000
- causal_clustering.discovery_advertised_address=vm3:5000

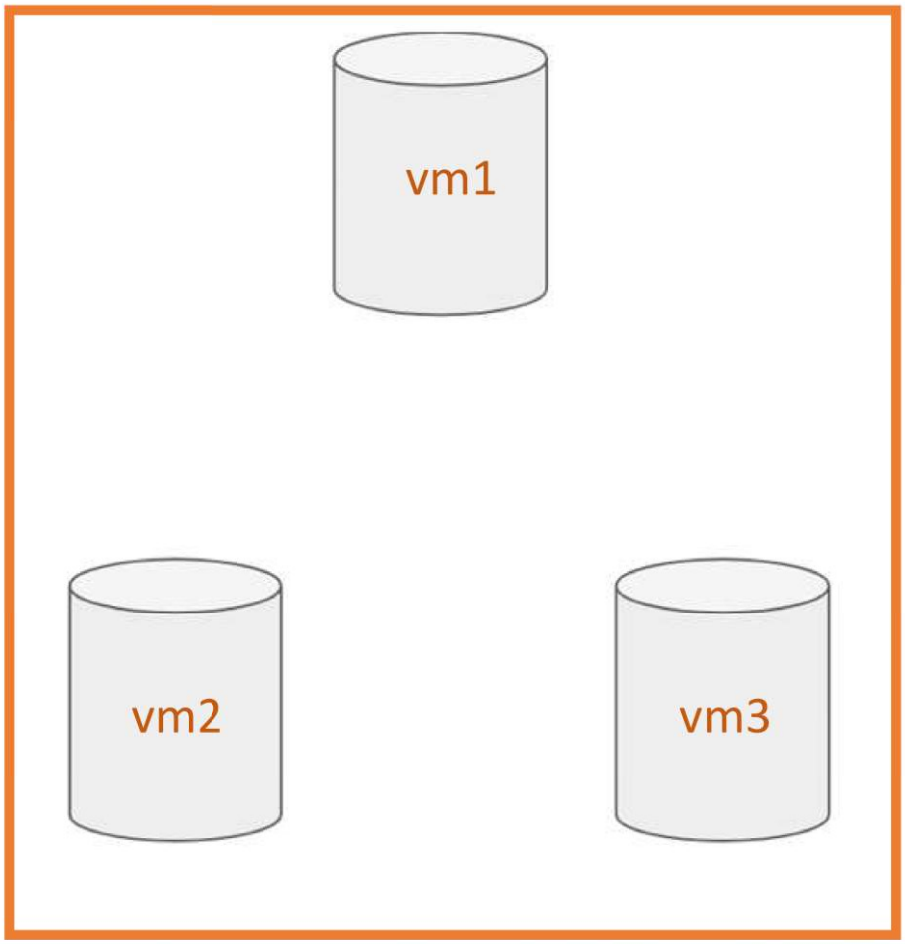


- causal_clustering.initial_discovery_members=
- vm1:5000, vm2:5000, vm3:5000



Specify cluster membership

- On all Core Servers:
  - causal_clustering.initial_discovery_members=
  - vm1:5000,
  - vm2:5000,
  - vm3:5000



Minimum cluster sizes

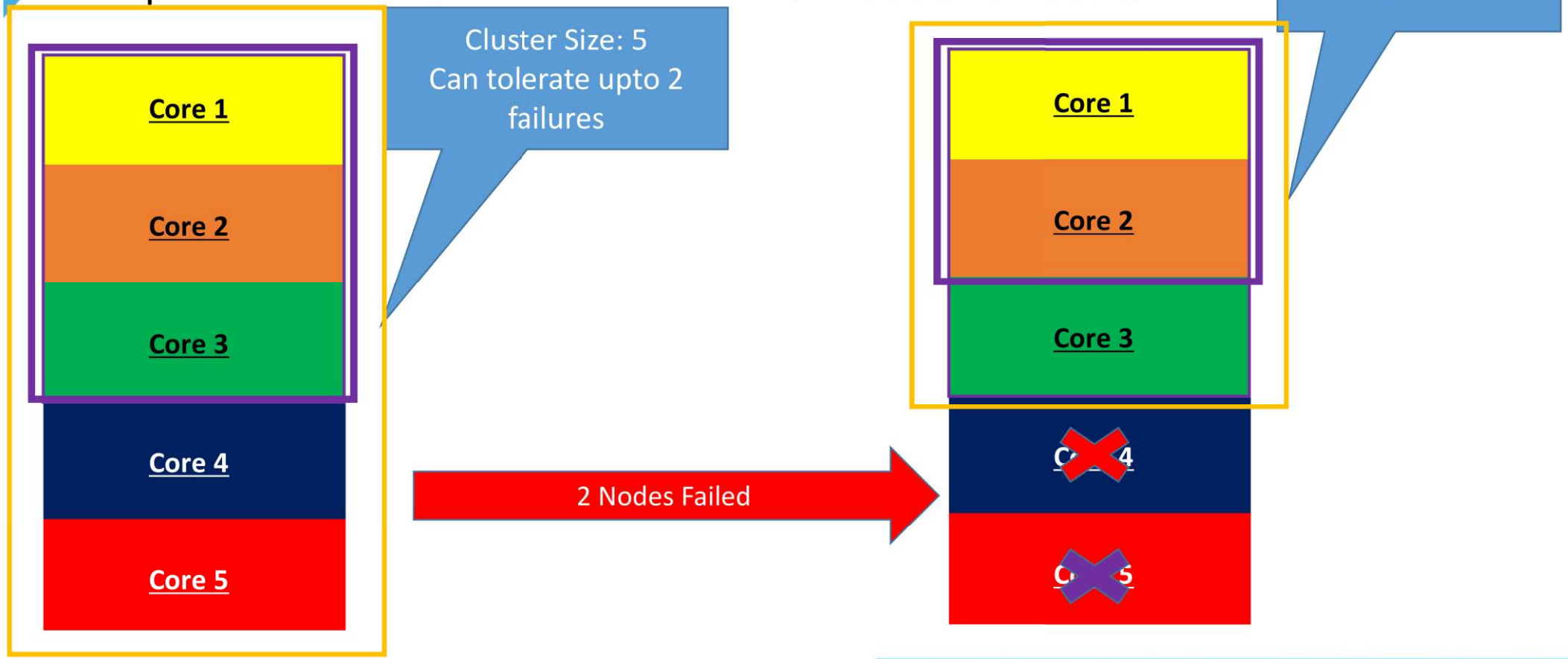
- minimum_core_cluster_size_at_formation
  - Cluster will form when at least three Core members have discovered each other
- minimum_core_cluster_size_at_runtime
  - Specify that the cluster should not try to dynamically adjust below three Core members
  - The cluster size will stay at 3 and will not shrink to 2. The offline instance is still considered a member of the cluster even if it's not available
  - With only 2 out of 3 instances online, if another instance fails we lose quorum and write capability
  - If a different core instance gets added it can still be voted in, since we still have quorum of 2 instances

Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 3
  - If we started out with a 5 instance cluster, quorum would be 3 of the 5 instances, and we can tolerate 2 instance failures while keeping quorum.
  - In the event of losing up to 2 instances of those 5, a member vote-out would take place and succeed, since a quorum is present
  - The cluster size would then scale down accordingly to a 3-instance cluster
    - With a new quorum size of 2 out of 3 and
    - Ability to tolerate just one more instance failure

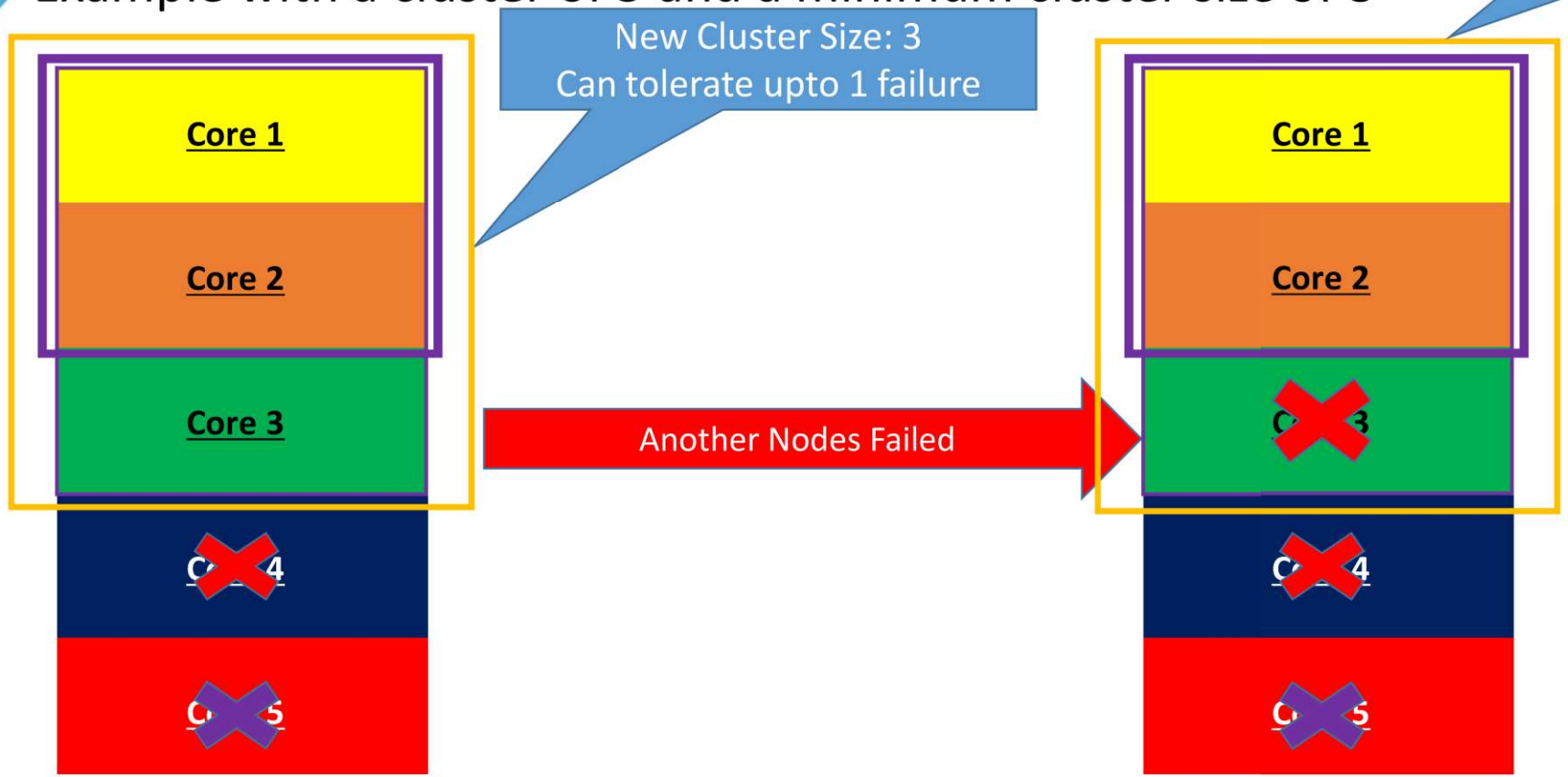
Minimum cluster sizes

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



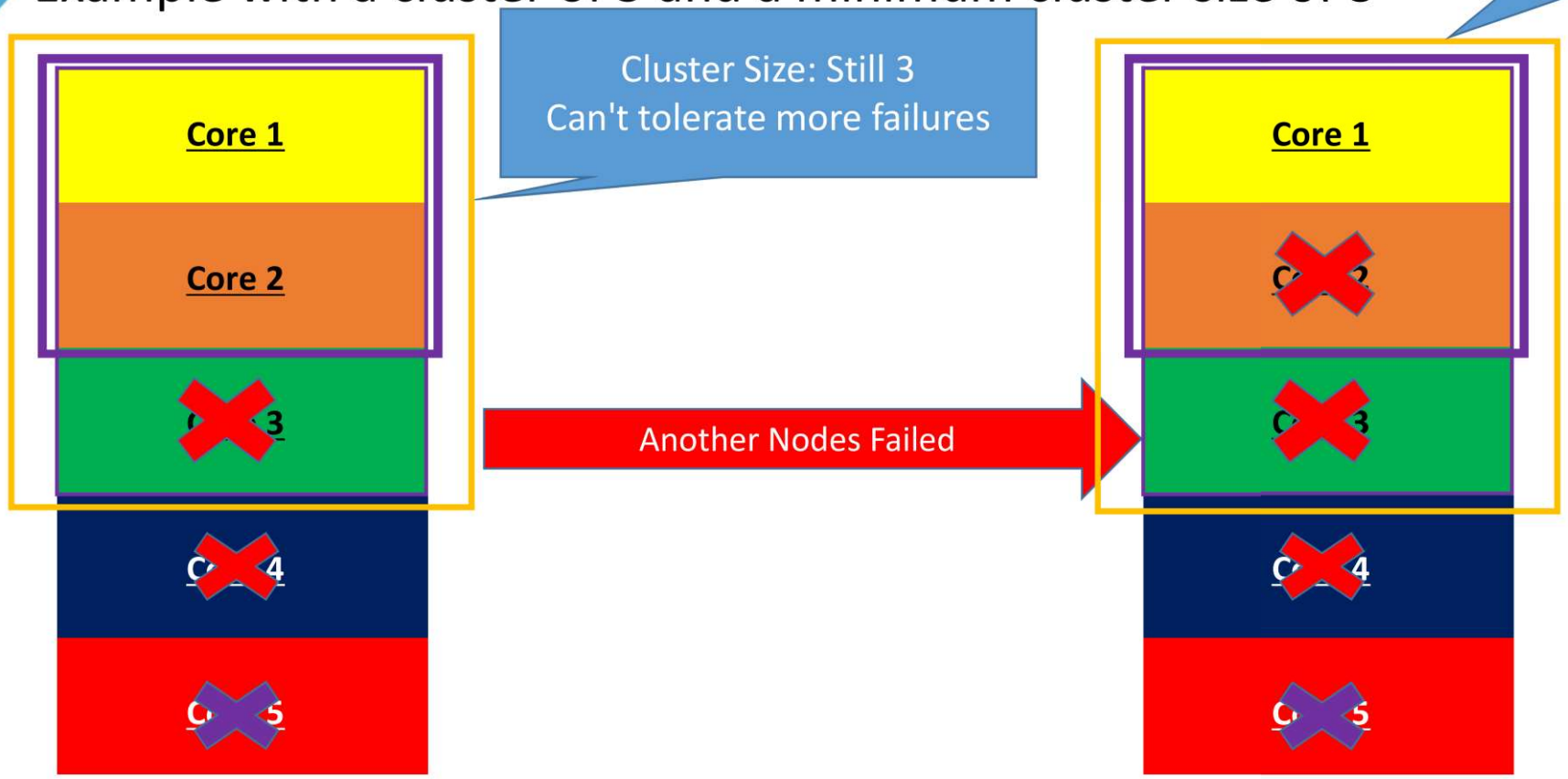
Minimum cluster sizes

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



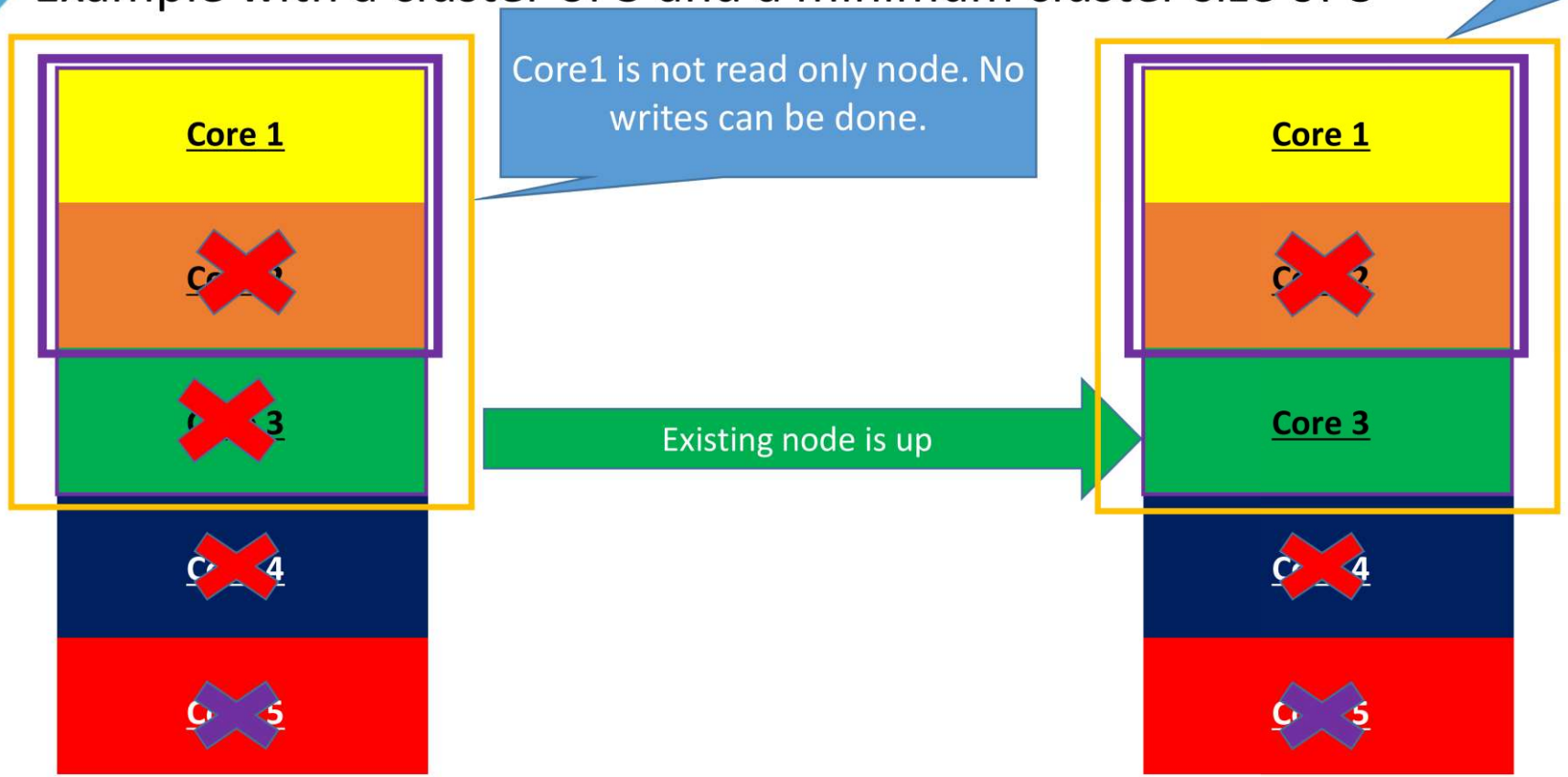
Minimum cluster sizes

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



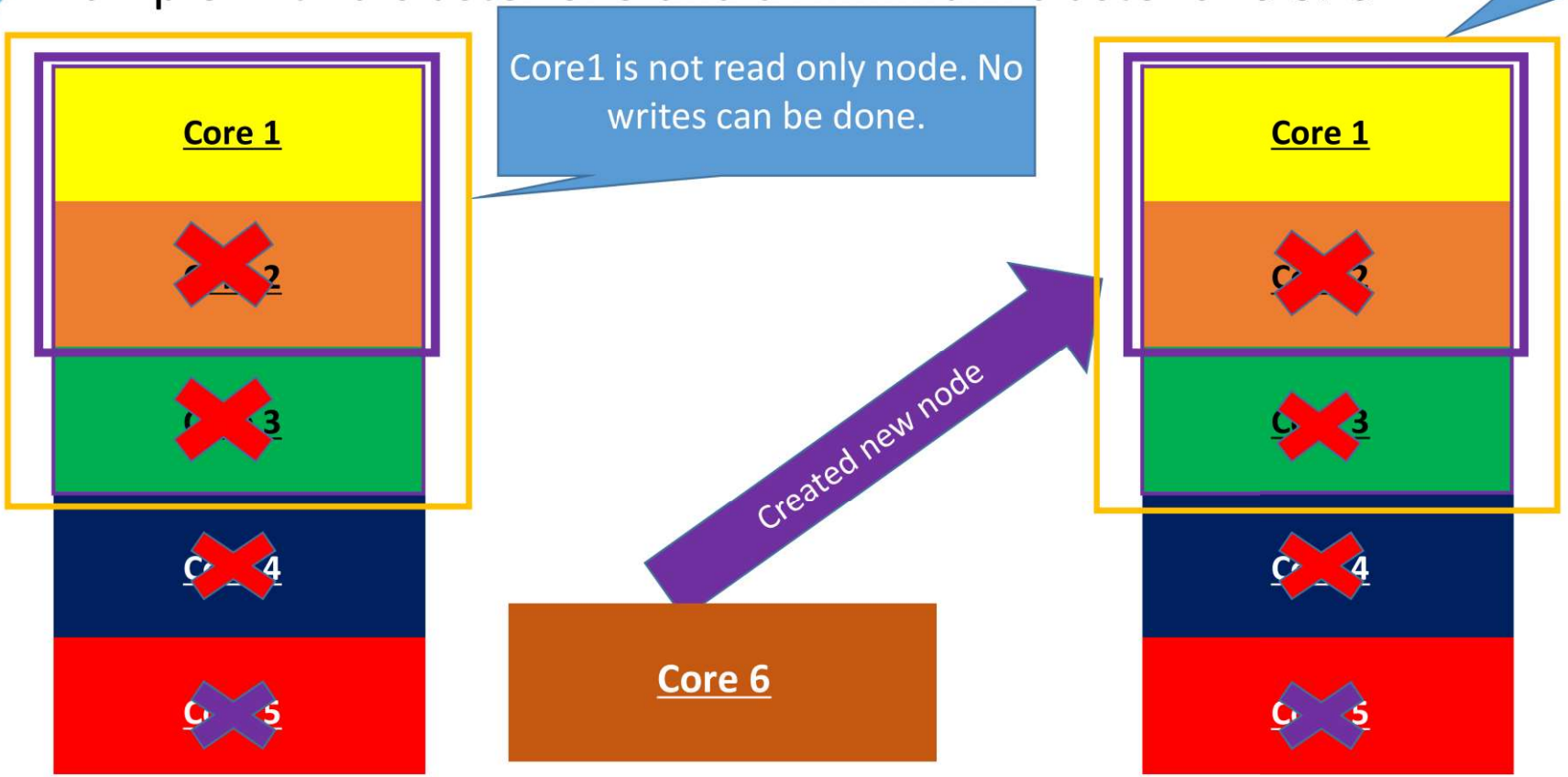
Minimum cluster sizes

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



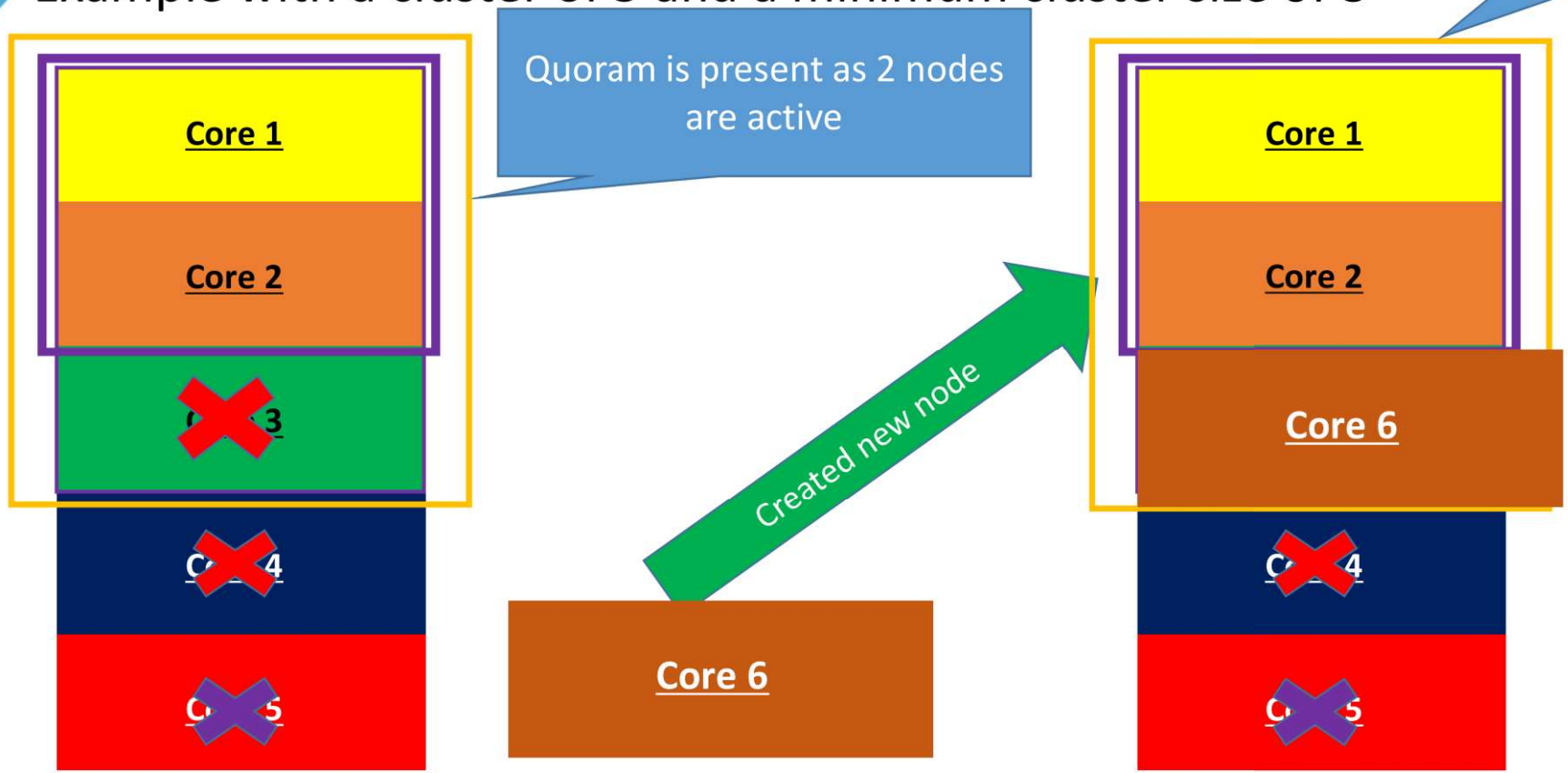
Minimum cluster sizes

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



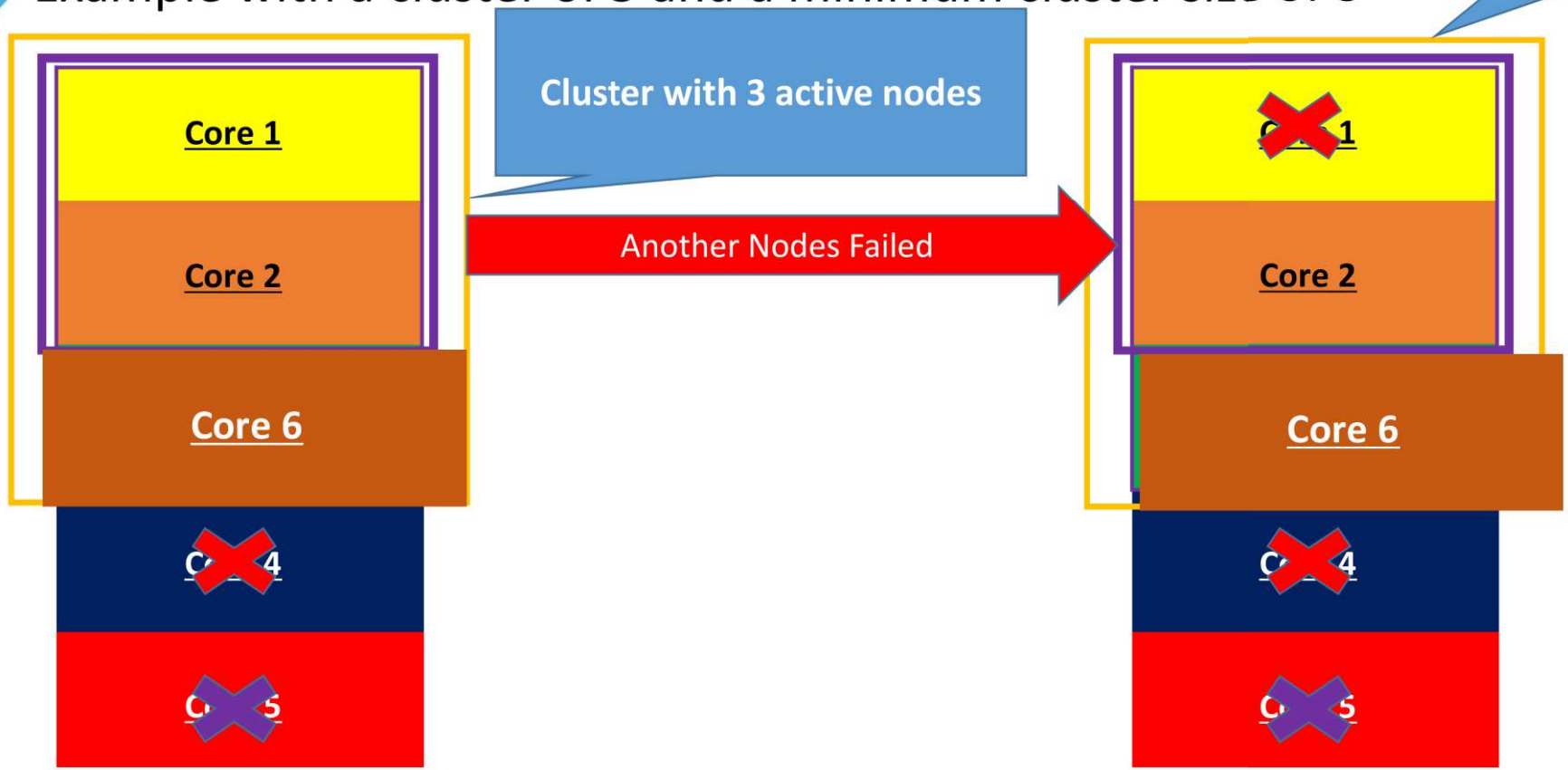
Minimum cluster sizes

- 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

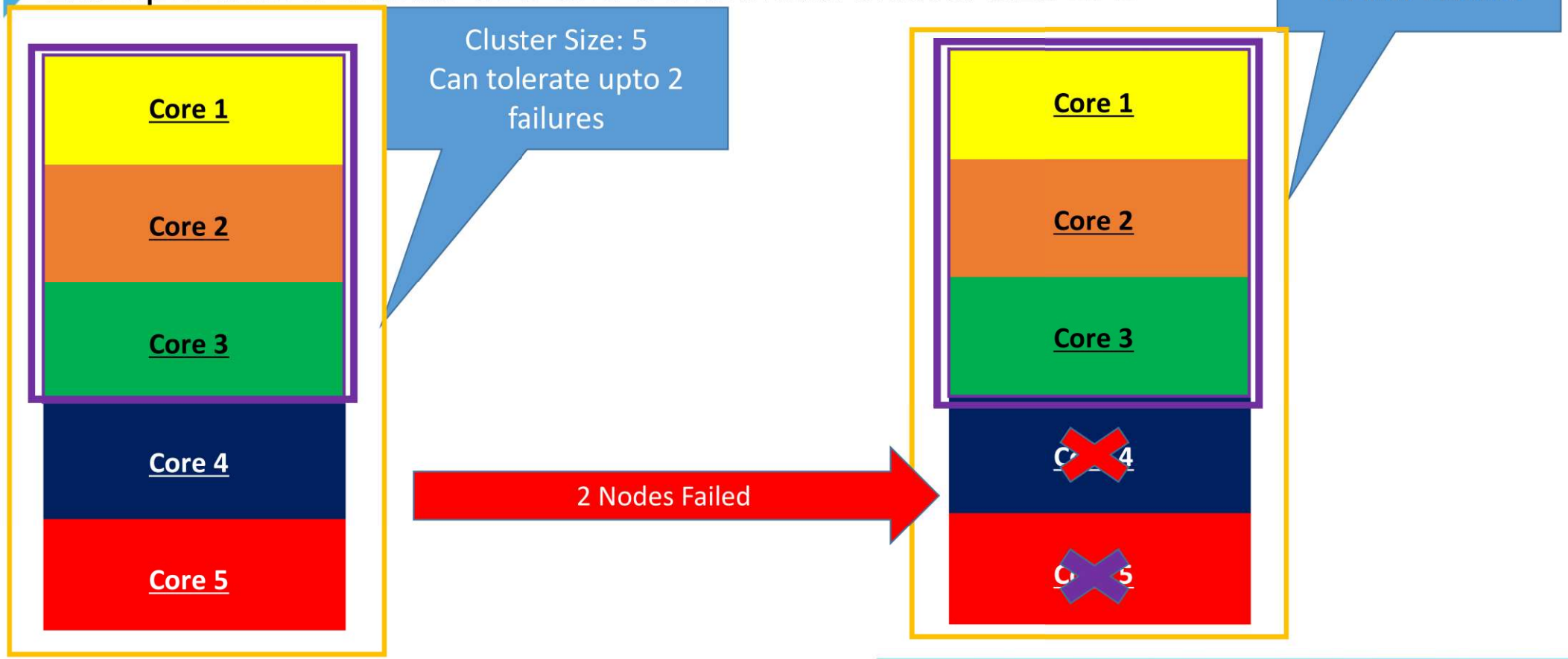


Minimum cluster sizes

- Example with a cluster of 5 and a minimum cluster size of 5
  - If we started with a 5 instance cluster and minimum cluster size of 5, we would not be able to scale down to a smaller cluster with instance failures.
  - While we could tolerate up to 2 simultaneous instance failures while keeping quorum, no cluster scaling would occur

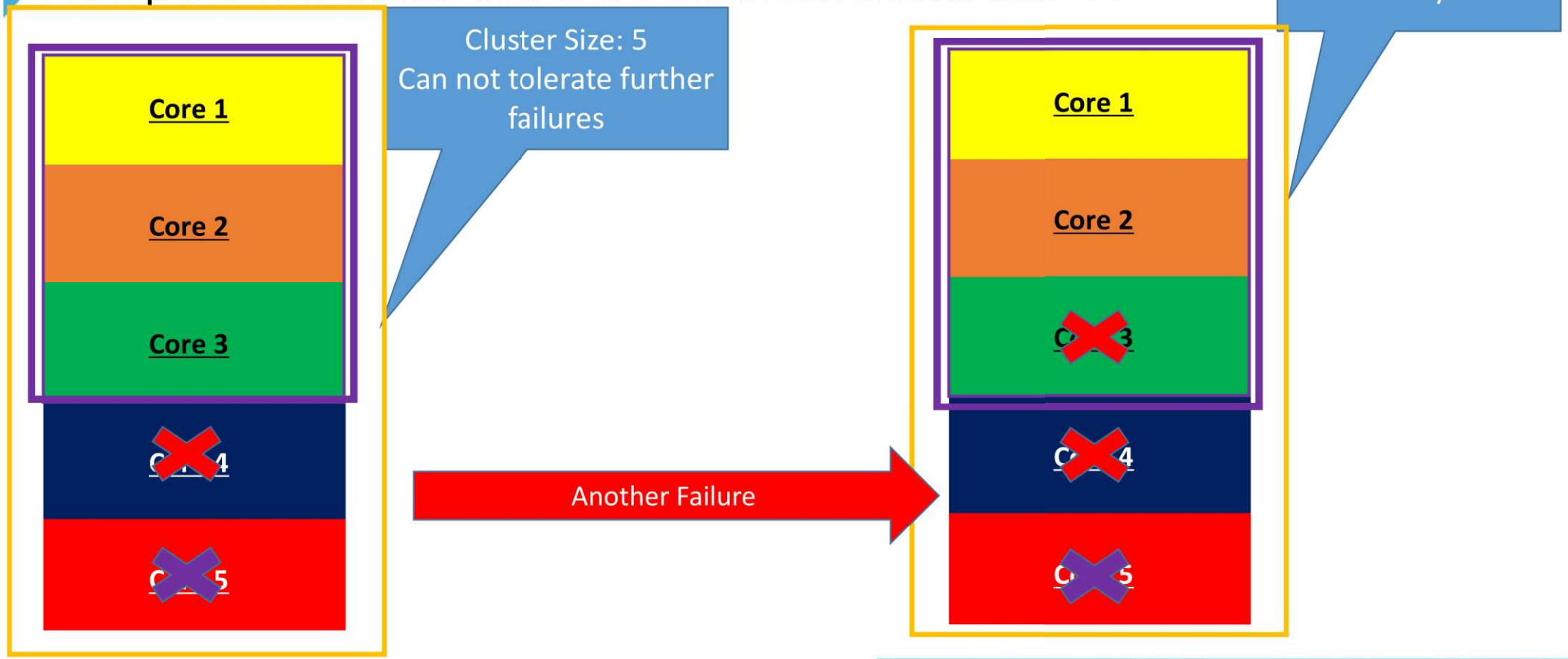
Minimum cluster sizes

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



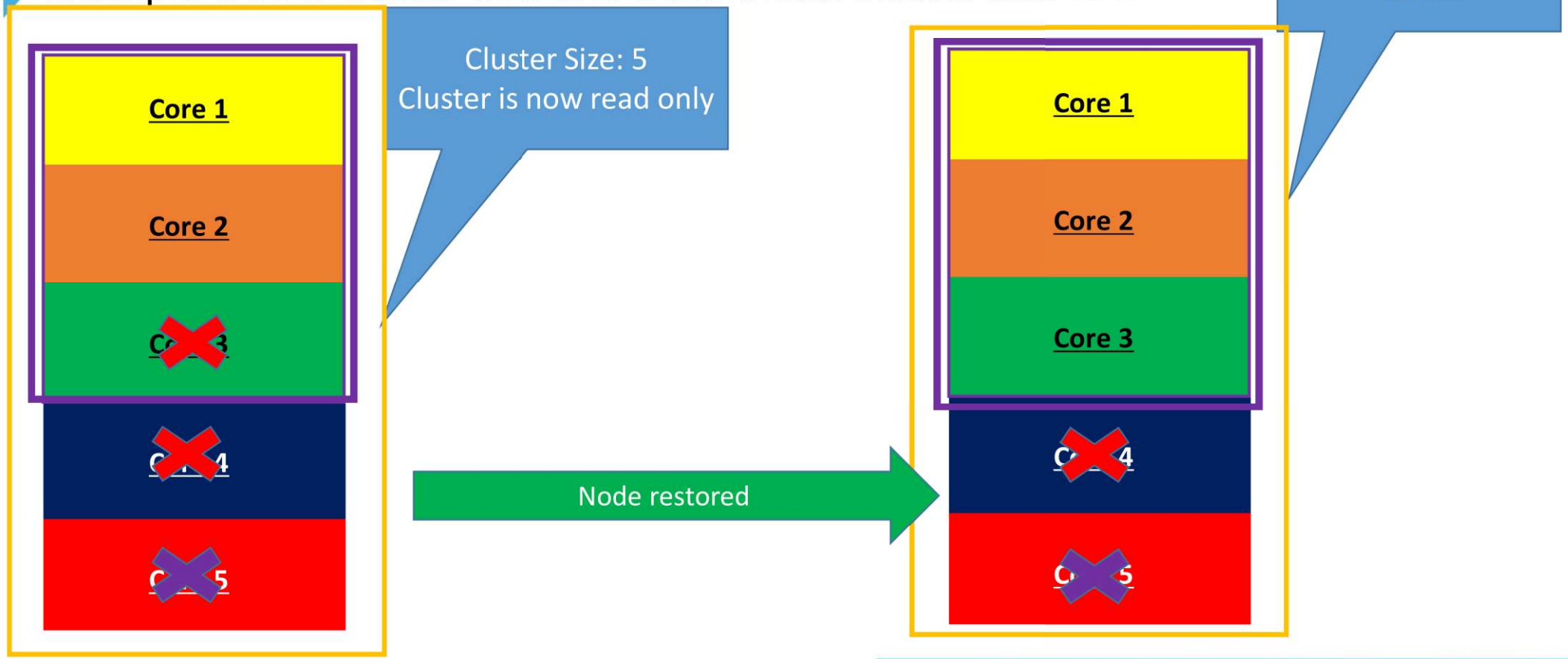
Minimum cluster sizes

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



Minimum cluster sizes

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

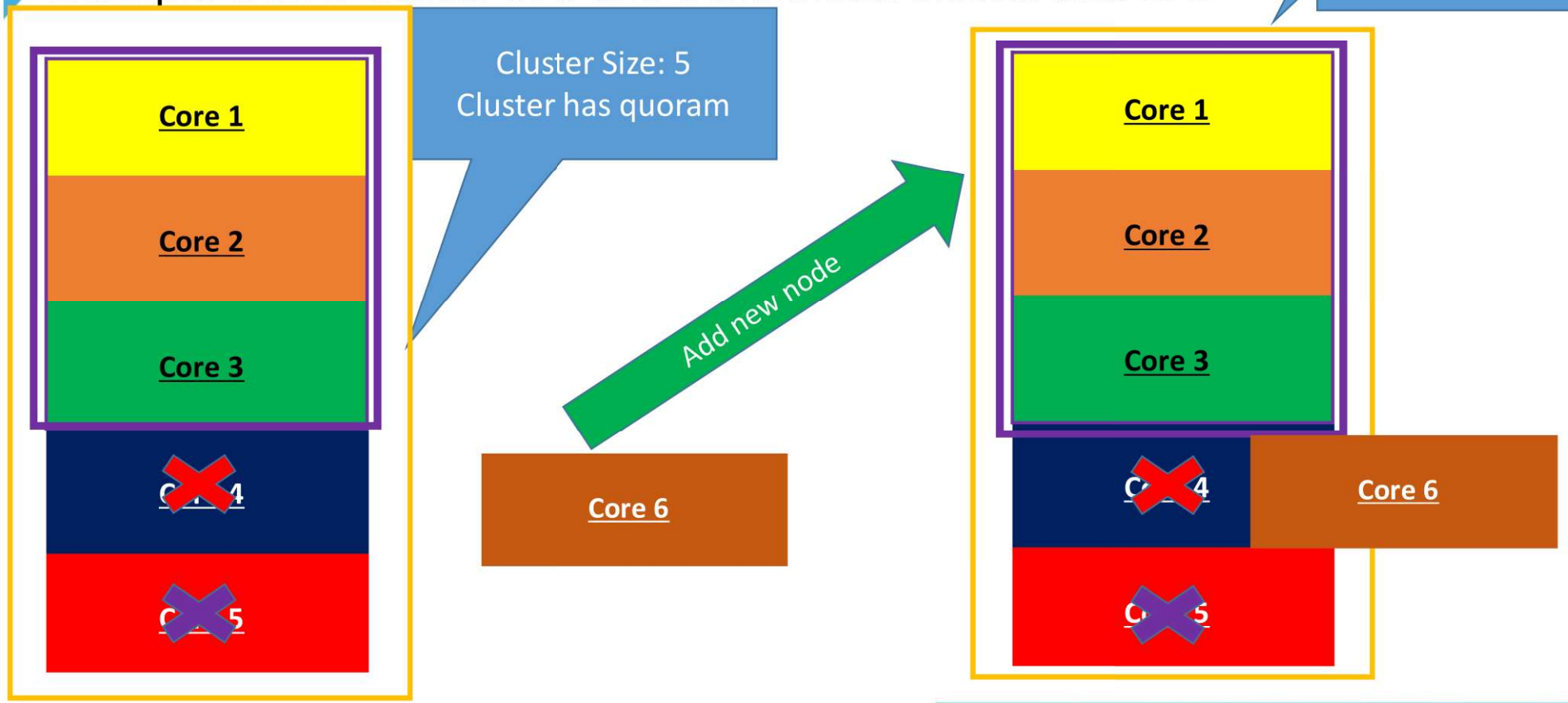


Minimum cluster sizes

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

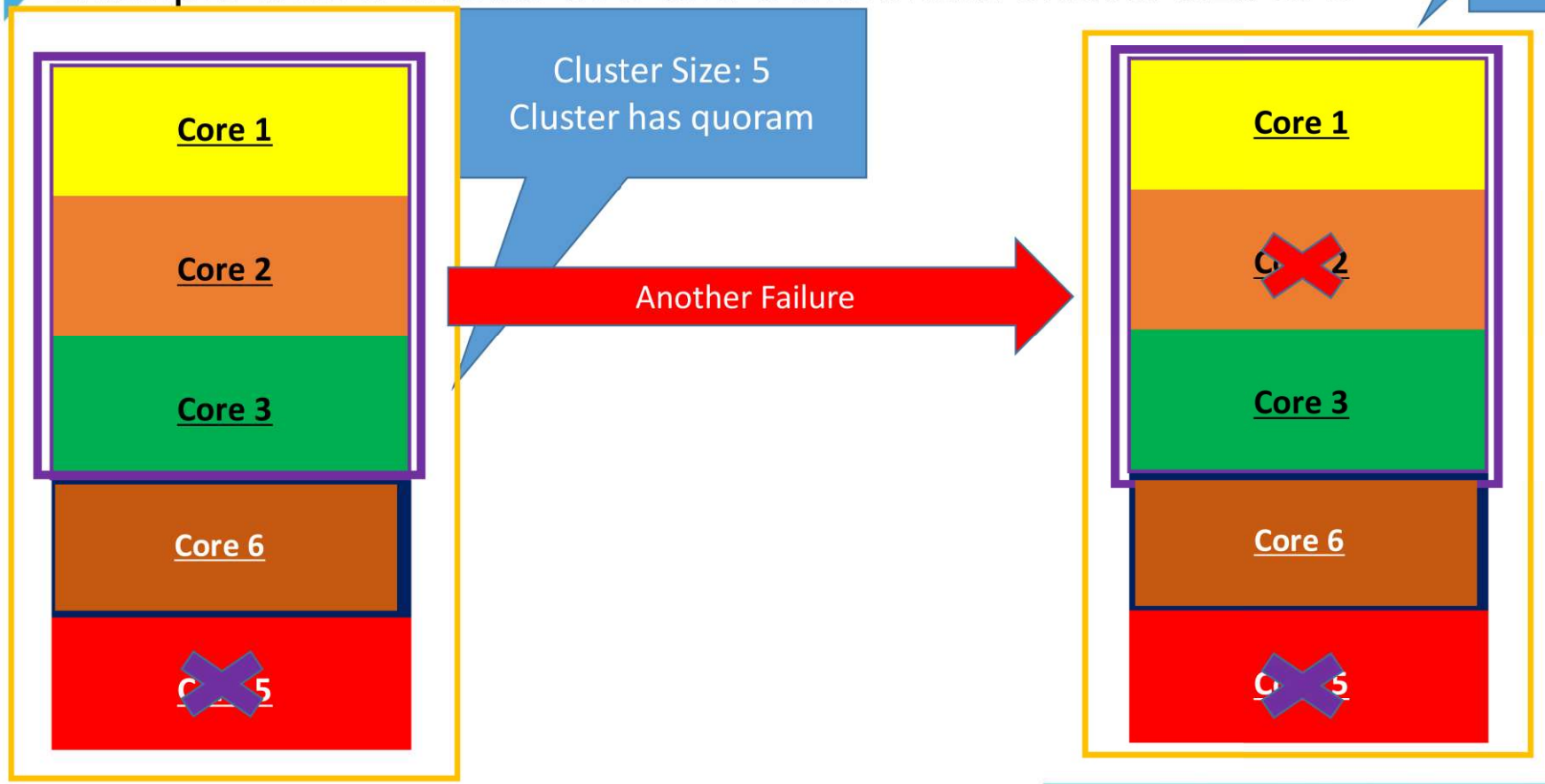
Minimum cluster sizes

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



Minimum cluster sizes

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



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

Configure a Core-only cluster

- Refer:
  - 12-Causal Clustering in Neo4j-B-Via-Individual-VMs

Add a Core Server to existing cluster

- Refer:
  - 12-Causal Clustering in Neo4j-B-Via-Individual-VMs - Add Core Server.txt

Add Read Replica to existing cluster

- Refer:
  - 12-Causal Clustering in Neo4j-B-Via-Individual-VMs - Read Replica.txt

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

Loading the data

1. neo4j stop
2. neo4j-admin unbind
3. Load the data using one of restore, load, or import commands of neo4j-admin

1. neo4j stop
2. neo4j-admin unbind
3. Load the data using one of restore, load, or import commands of neo4j-admin

1. neo4j stop
2. neo4j-admin unbind
3. Load the data using one of restore, load, or import commands of neo4j-admin

Loading the data

- Open any node on browser:
  - http://ag-20201029-vm1:7474/browser/
- Insert some data:
  - :USE neo4j
  - CREATE (:Movie {title: 'Batman Begins'})
  - CREATE (:Movie:Action {title: 'Batman Begins'})
  - CREATE (m:Movie:Action {title: ' Batman Begins'}) RETURN m.title
  - CREATE (:Person {name: 'Michael Caine', born: 1933}), (:Person {name: 'Liam Neeson', born: 1952}), (:Person {name: 'Katie Holmes', born: 1978}), (:Person {name: 'Benjamin Melniker', born: 1913})

Loading the data

- Confirm that the data has been loaded into the database.
  - match(n) return n


bolt+routing

Bolt

Neo4j's proprietary protocol

For communication between client applications and database servers

For direct connection to specific server

Bolt Routing

- A variant of the bolt protocol
- Automatic routing to an available server
- Load balancing of requests between the available servers
- Automatic retries
- Can specify a FOLLOWER as recipient and driver will route updates to LEADER

Bolt & Bolt+Routing at runtime

- For a 3 node cluster and core1 as the LEADER, application can write to only core1 using the bolt protocol

- You can successfully read from any server

Server Types and Operations

Server Type	Protocol	Read Allowed?	Write Allowed?
Core Leader	Bolt	Yes	Yes
Core Follower	Bolt	Yes	No
Reader	Bolt	Yes	No
Core Leader	Bolt + Route	Yes	Yes
Core Follower	Bolt + Route	Yes	Yes
Reader	Bolt + Route	Yes	Yes

Configuring read replica servers

- Change
  - dbms.mode=CORE
- To
  - dbms.mode=READ_REPLICA

Read replica server shutdown

- No effect to the operation of the cluster.

Recovering a core server

- Two options:
  - Start a new core server that has not yet been part of the cluster, but is specified in the membership list of the cluster
    - This will only work if the cluster currently has a quorum so the existing core servers can vote to add the core server to the cluster
- Start a new parallel cluster with backup from current read only cluster

- Option (1) is much easier
- Best practice is to
  - Always specify additional hosts that could be used as replacement core servers
  - Enables to add core servers to cluster without needing to stop cluster

Monitoring core servers

- curl -u neo4j:secret <hostname>:7474/db/<dbname>/cluster/writable
- curl -u neo4j:secret <hostname):7474/db/<dbname>/cluster/read-only
- curl -u neo4j:secret <hostname):7474/db/<dbname>/cluster/available
- curl -u neo4j:secret <hostname):7474/db/<dbname>/cluster/status | jq '.'

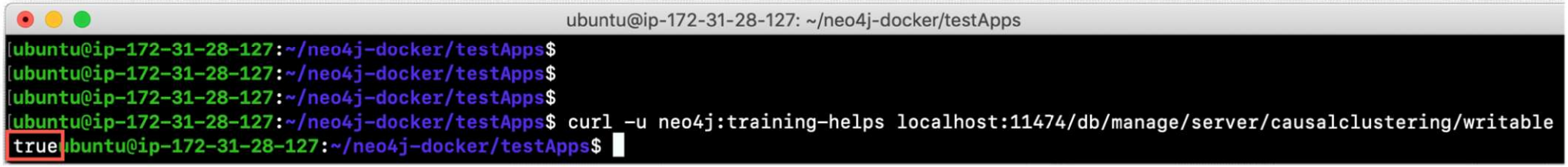
- <hostname> is the host name of the target neo4j server
  - ag-20201029-vm1
  - ag-20201029-vm2
  - ag-20201029-vm3
- <dbname> is the database of which we need to check status
  - neo4j
  - system

Monitoring core servers-Example

- vm1:7474/db/system/cluster/writable
- vm1:7474/db/system/cluster/read-only
- vm1:7474/db/system/cluster/available
- vm1:7474/db/system/cluster/status | jq '.'

Is the core server writable?

- vm1:7474/db/neo4j/cluster/writable
- vm2:7474/db/neo4j/cluster/writable
- vm3:7474/db/neo4j/cluster/writable
- vm4:7474/db/neo4j/cluster/writable
- vm5:7474/db/neo4j/cluster/writable



Understanding quorum

- CALL dbms.cluster.overview();

- Stop the core server that is the LEADER.
- View the cluster overview using another neo4j server:
  - dbms.cluster.overview();

Understanding quorum

- Login to the portal using **Bolt** driver

- Can you write to the database?

Understanding quorum

- Stop few core servers so that the quorum is lost
  - Note: Make sure to maintain 2-3 minutes gap between each server stop
- Make sure that any leader is available
  - CALL dbms.cluster.overview();
- Also check if you can write to any database

Understanding quorum

- Start a core server that you previously stopped.
- View the cluster overview. Is there now a LEADER?
- This cluster is operational because it now has a LEADER

Backing up a cluster

- The database for a cluster is backed up online
- In the configuration for each core server must specify
  - dbms.backup.enabled=true
  - dbms.backup.address=<server-address>:6362

Backup using a read replica

- Best practice is to create and use a read replica server for backups
- The last transaction ID is on a core server vs. a read replica
  - CALL dbms.listTransactions() YIELD transactionId;

- neo4j-admin backup --backup-dir=<backup-path> --name=<backup-name> --from=<core-server:backup-port> --protocol=catchup --check-consistency=true


Cognixia®
THANK YOU