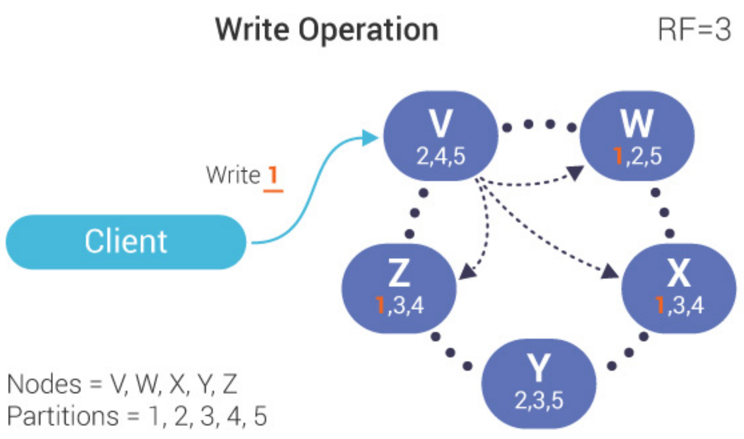
Scylla Architecture - Fault Tolerance

Scylla replicates data according to a replication (../../glossary/#term-replication) strategy that you choose. This strategy will determine the placement of the replicated data. Scylla runs nodes in a hash ring. All nodes are equal: there are no master, slave, or replica sets.

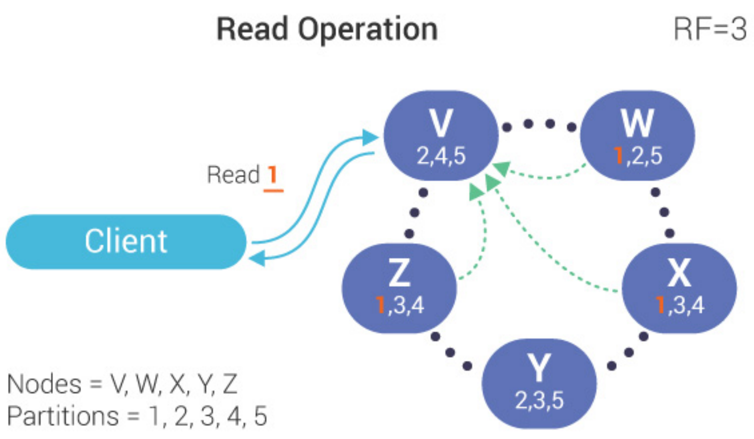
The Replication Factor (RF) (../../glossary/#term-replication-factor-rf) is equivalent to the number of nodes where data (rows and partitions) are replicated. Data is replicated to multiple (RF=N) nodes.

An RF of `1` means there is only one copy of a row in a cluster and there is no way to recover the data if the node is compromised or goes down. RF=2 means that there are two copies of a row in a cluster. An RF of at least `3` is used in most systems or similar.

Data is always replicated automatically. **Read** or **write** operations can occur to data stored on any of the replicated nodes.



In the example above, our client sends a request to write partition *1* to node *V*; 1's data is replicated to nodes *W*, *X*, and *Z*. We have a Replication Factor (RF) of `3`. In this drawing, *V* is a coordinator node but not a replicator node. However, replicator nodes can also be coordinator nodes, and often are.



During a read operation (../../glossary/#term-read-operation) , the client sends a request to the coordinator. Effectively because the RF=3, 3 nodes respond to the read request.

The Consistency Level (CL) (../../glossary/#term-consistency-level-cl) determines how many replicas in a cluster must acknowledge read or write operations (../../glossary/#term-write-operation) before it is considered successful.

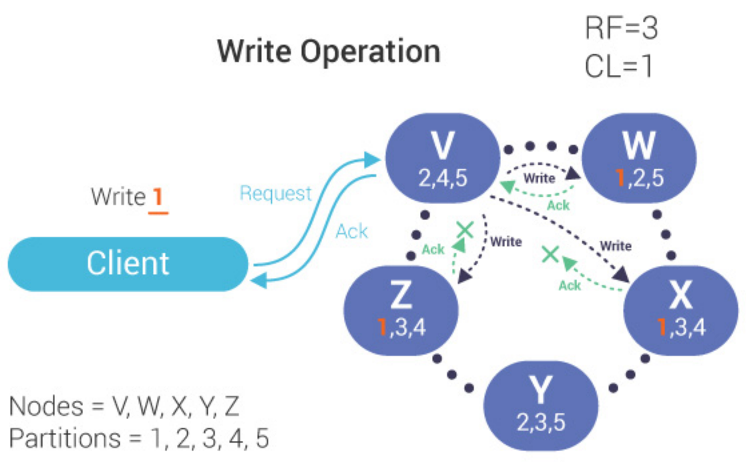
For the CQL Shell (CQLsh (/getting-started/cqlsh)), the consistency level defaults to ONE for read and write operations.

Some of the most common Consistency Levels used are:

- ANY (../../glossary/#term-consistency-level-any)
- QUORUM (../../glossary/#term-consistency-level-quorum)
- ONE (../../glossary/#term-consistency-level-one)
- LOCAL_ONE (../../glossary/#term-consistency-level-local-one)
- LOCAL_QUORUM (../../glossary/#term-consistency-level-local-quorum)
- EACH_QUORUM (../../glossary/#term-consistency-level-each-quorum)
- ALL (../../glossary/#term-consistency-level-all)

> NOTE
>
> Regardless of the **Consistency Level**, a write is always sent to *all* replicas, as set by the **Replication Factor**. Consistency Level control *when* a client acknowledged, not how many replicas are updated.

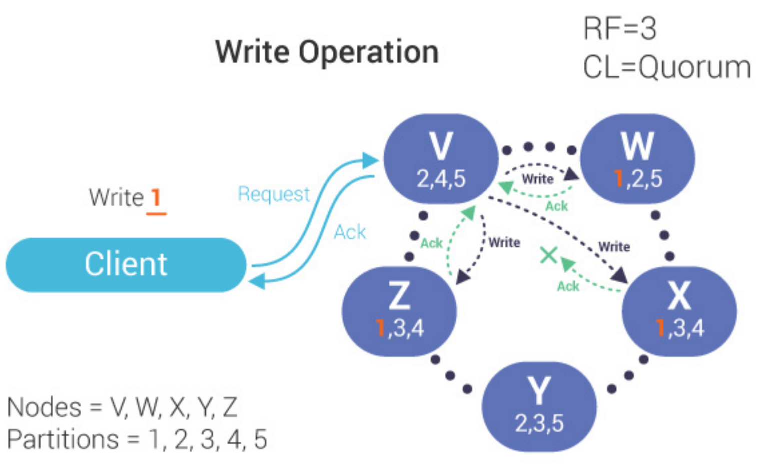
During a write operation, the coordinator communicates with the replicas (the number of which depends on the Replication Factor). The write is successful when the specified number of replicas confirm the write.



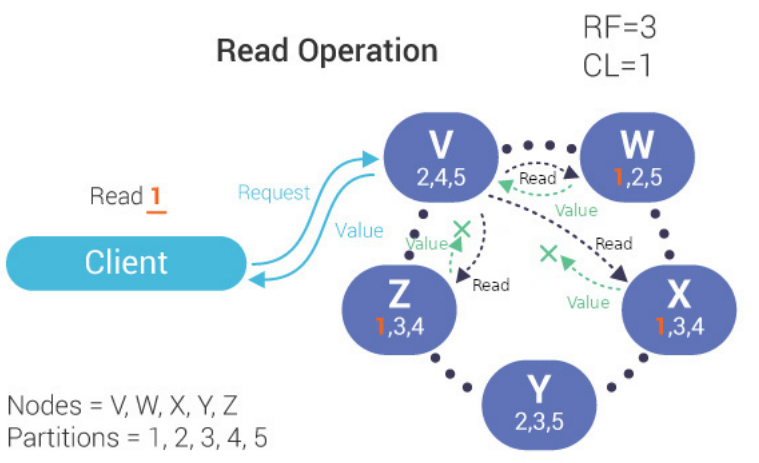
In the above diagram, the double arrows indicate the write operation request going into the coordinator from the client and the acknowledgment being returned. Since the Consistency Level is one, the coordinator, *V*, must only wait for the write to be sent to and responded by a single node in the cluster which is *W*.

Since RF=3, our partition 1 is also written to nodes *X* and *Z*, but the coordinator does not need to wait for a response from them to confirm a successful write operation. In practice, acknowledgments from nodes *X* and *Z* can arrive to the coordinator at a later time, after the coordinator acknowledges the client.

When our Consistency Level is set to `QUORUM`, the coordinator must wait for a majority of nodes to acknowledge the write before it is considered successful. Since our Replication Factor is 3, we must wait for 2 acknowledgments (the third acknowledgment does not need to be returned):



During a read operation, the coordinator communicates with just enough replicas to guarantee that the required Consistency Level is met. Data is then returned to the client.



The Consistency Level is tunable per operation in CQL. This is known as tunable consistency (../../glossary/#term-tunable-consistency) . Sometimes response latency is more important, making it necessary to adjust settings on a per-query or operation level to override keyspace or even data center-wide consistency settings. In other words, the Consistency Level setting allows you to choose a point in the consistency vs. latency tradeoff.

> NOTE
>
> Quorum is a global consistency level across the *entire* cluster. This means that if you have two data centers, all nodes in both datacenters count towards the quorum majority. For example, there is a cluster with two DCs with three nodes in one DC and two nodes in the other. If the smaller DC fails, requests will still pass under Quorum as 3 > 5/2.

The Consistency Level and Replication Factor both impact performance. The **lower** the Consistency Level and/or Replication Factor, the **faster** the read or write operation. However, there will be less fault tolerance if a node goes down.

The Consistency Level itself impacts availability. A **higher** Consistency Level (more nodes required to be online) means less availability with less tolerance to tolerate node failures. A **lower** Consistency Level means more availability and more fault tolerance.

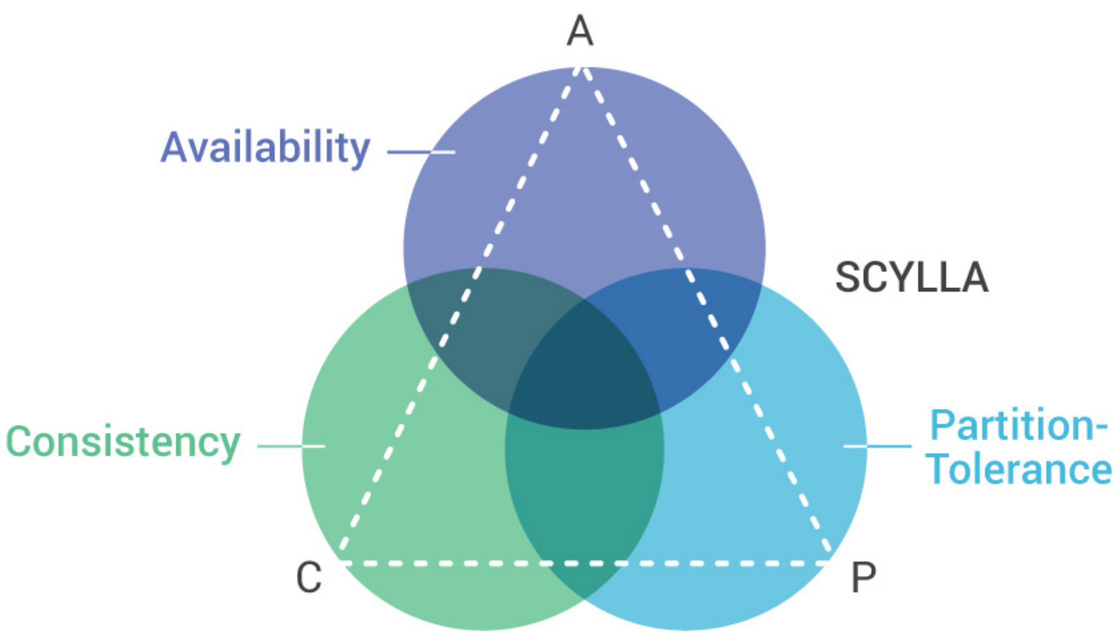
The following table shows what Consistency Levels are available for a read or write operation:

Consistency Level	Read	Write
Any	No	Yes
1	Yes	Yes

`2`	Yes	Yes
`3`	Yes	Yes
`QUORUM`	Yes	Yes
`LOCAL_ONE`	Yes	Yes
`LOCAL_QUORUM`	Yes	Yes
`EACH_QUORUM`	No	Yes
`ALL`	Yes	Yes

Scylla, as do many distributed database systems, adheres to the CAP Theorem (../../glossary/#term-cap-theorem) . The **CAP Theorem** is the notion that **Consistency**, **Availability** and **Partition Tolerance** of data are mutually dependent in a distributed system. Increasing any 2 of these factors will reduce the third.

Scylla adheres to the CAP theorem in the following way:



Scylla chooses availability and partition tolerance over consistency, such that:

- It’s impossible to be both consistent and highly available during a network partition;
- If we sacrifice consistency, we can be highly available.

You’ll need to design your application around Scylla’s data modeling, but the net result is an application that will never go down.

Additional Resources

Consistency Level Console Demo (../console_CL_full_demo/) .

You can also learn more in the Consistency Level lesson on Scylla University.

- Getting Started (../../getting-started/)
- Scylla for Administrators (../../operating-scylla/)
- Scylla for Developers (../../using-scylla/)
- Scylla Cloud (../../scylla-cloud/)
- Scylla Architecture (../)
  - Scylla Ring Architecture (../ringarchitecture/)
  - Scylla Fault Tolerance
  - Consistency Level Console Demo (../console_CL_full_demo/)

DOCS (HTTPS://DOCS.SCYLLADB.COM) CONTACT US (HTTPS://WWW.SCYLLADB.COM/COMPANY/CONTACT-US/) ABOUT US (HTTPS://

(https://www.scylladb.com)

REPORT AN ISSUE ON THIS PAGE (HTTPS://GITHUB.COM/SCYLLADB/SCYLLA-DOC-ISSUES/ISSUES/NEW?TITLE=ISSUE IN PAGE S
TOLERANCE&&BODY=I%20WOULD%20LIKE%20TO%20REPORT%20AN%20ISSUE%20IN%20PAGE%20HTTPS://DOCS.SCYLLADB.COM/
FAULT-TOLERANCE%0A%0A%23%23%23%20PROBLEM%0A%0A%23%23%23%20%20SUGGEST%20A%2