

Job Post ✓

Brand Building ✓

Hadoop

2 things

1. lots of data

2. Cluster

Hadoop is an open source framework designed to handle massive amounts of data in a scalable and distributed way

(-) with earlier soln

→ storage limits

→ Processing issue

→ expensive to add more memory

2000

Doug
Caserella

GFS → distributed storage → HDPS

MapReduce → distributed data processing → MR

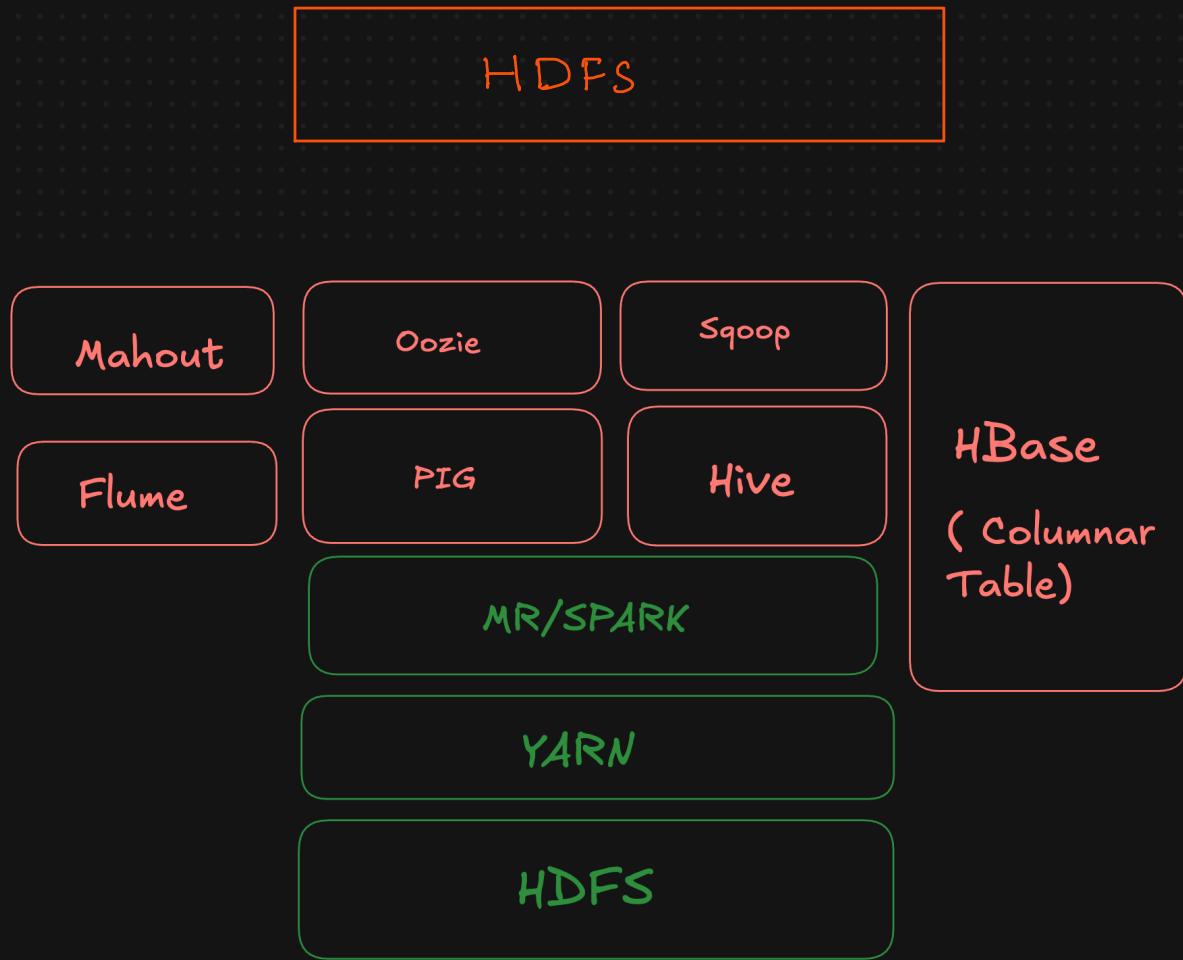
Hadoop can store and process data across many cheap, commodity hardware m/c working together

↓
Massive data storage ↓
↓
Cluster parallel processing

Properties

1. Scalability
2. Fault Tolerance
3. Distributed Processing
4. Cost-effectiveness

Hadoop Ecosystem / framework



2 imp things

1. Loosely coupled framework :
2. Integration

SQL

(write heavy)

			a
			b
			c
			d
			a
			b
			c

+

--

Database

OLTP

Columns



Seek time

Data warehouse

OLAP

Task for students

⇒ Master Basic

Linux Command

HDFS

hadoop distributed file system

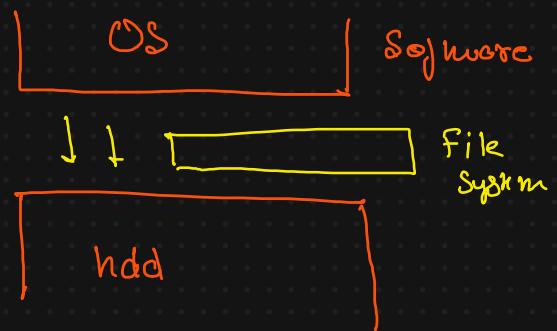
GFS

1. What is a file system?

Windows NTFS / FAT32

Apple APFS

Linux ext



Dual boot
win NTFS linux ext

2. What is a block?

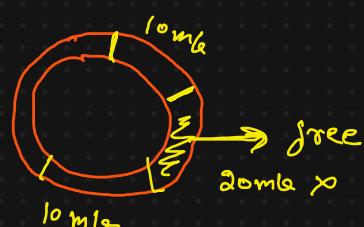
Smallest unit of data storage in a file system

block size \Rightarrow depends on file system

10 mb \rightarrow 3 blocks

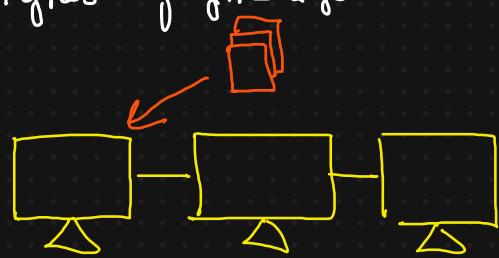
NTFS \rightarrow 4 mb

Hadoop / HDFS \rightarrow 128 mb

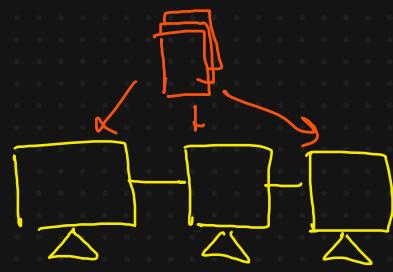


50 mb

3. Types of file system.



Standalone



Distributed

4. Cluster and Node?

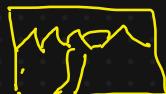
5. Process & Daemon Process?

hadoop is a process

{ JP1
JP2 } HDFS
SP3
JP4
JP5 } MR

6. Metadata

data about data



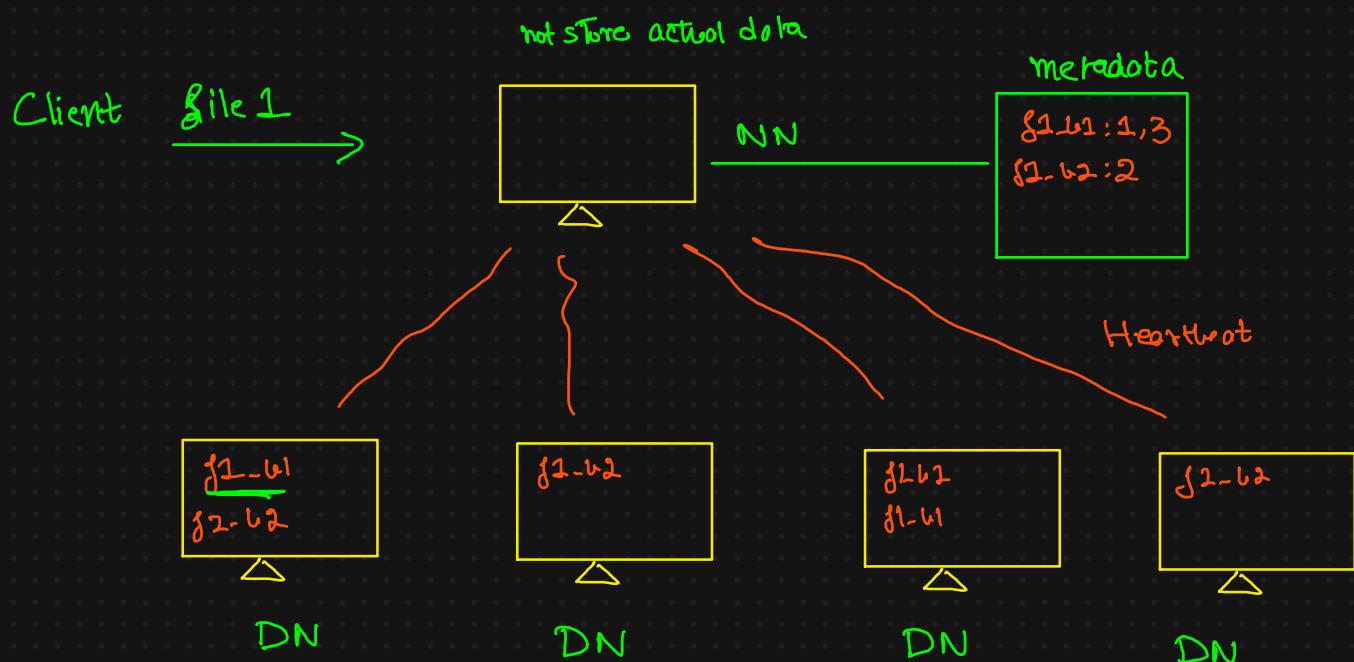
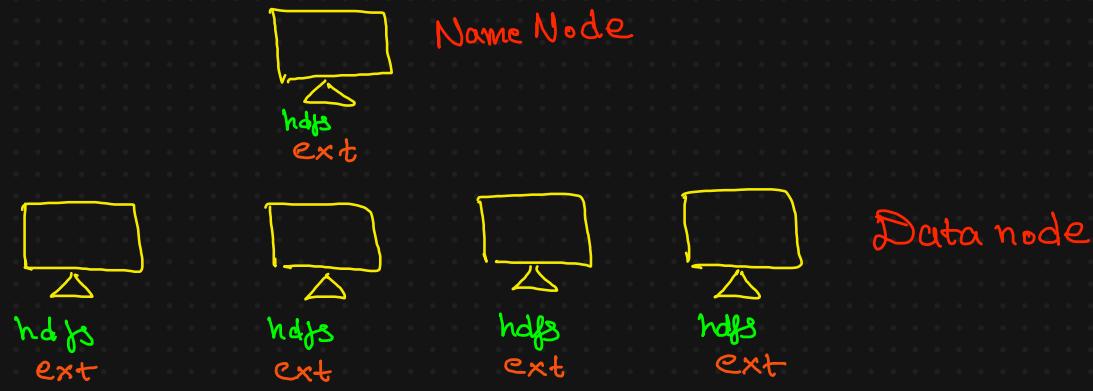
image

size
located at
pixels
dimensions
metadata

7. Replication

making copies of data

HDFS



replication factor = 3
log

Blocks



128 mb by default size

256 mb → 2 blocks

Key Points

1. efficient for large storage file
2. Distributed data

Bazooka / Gun

no hill on count

1Rb

64mb

→ 1 block

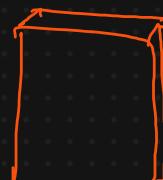
64mb

127mb

1kb = 10 → 10 blocks

Q: Can we inc or dec. block size?

Yes, default is 128mb



1. decrease → 10mb

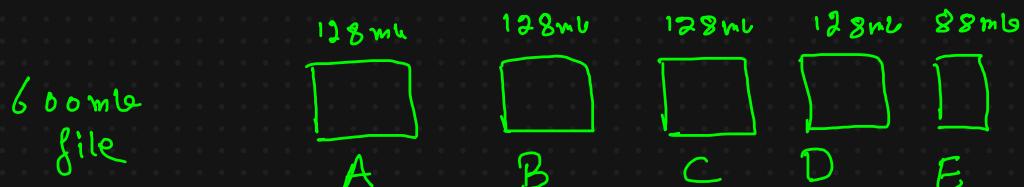
↑↑ parallelism but also ↑↑ metadata



128mb

index	Chapters
metadata	10 pages
metadata ↑↑	1 pages

2. increase : parallelism



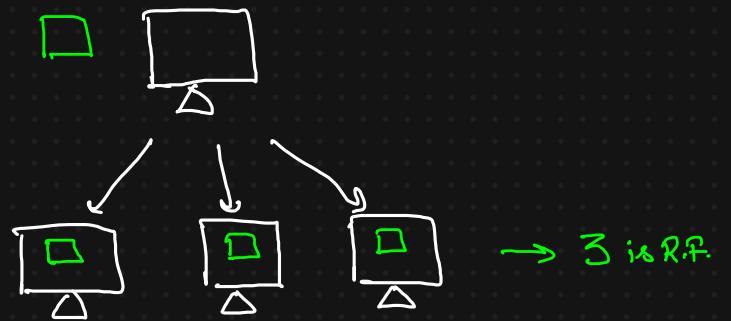
900mb
file

129mb
file

and 64mb
block size

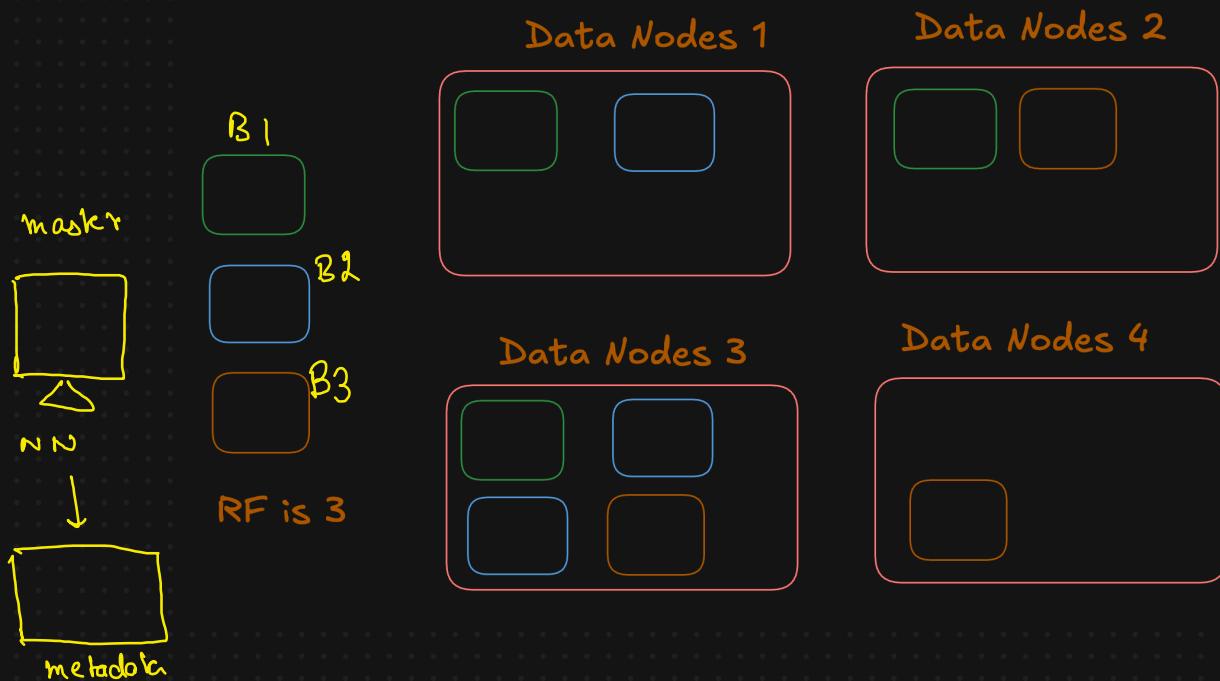
→ 3 blocks

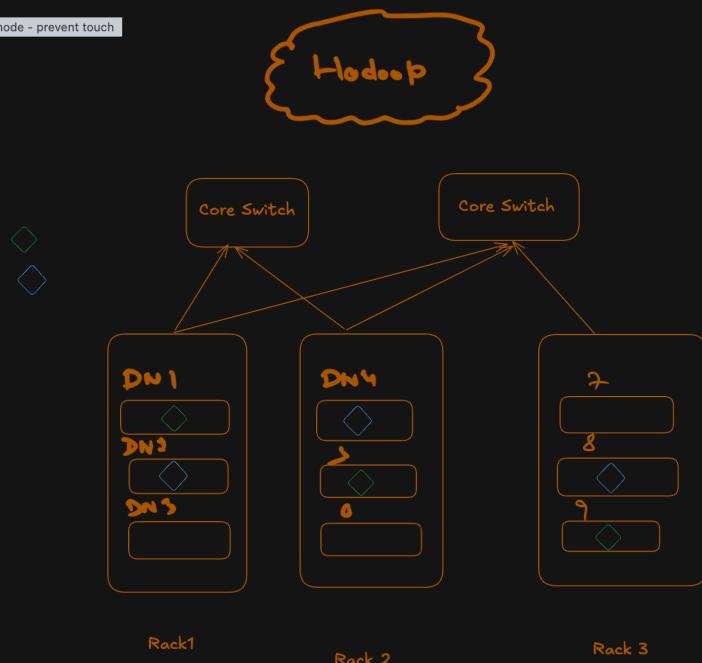
Data Node Failures



1. Replication

R.F. = 3 \Rightarrow 1 original
2 copies





- 1 Fault tolerance
2. Optimized Network Traffic

Q:

1. Temporary failure
 2. Permanent failure
 - Name node failure
- +
- HDFS Procrical
3. YARN + MR

Linux
+
hdfs } Quick
 Guide

Temporary failures in Data Node

- Network outage
- Software issue
- Reboot for maintenance

Process during this failure

1. Detection

temporary failure

heartbeat

2. Replication

Name node identifies the blocks stored on unavailable DN

marks them as under-replicated.

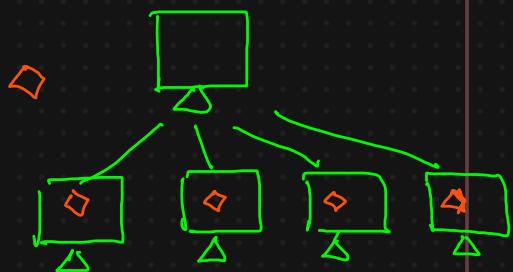
→ updating metadata

3. When the DN recovers

the DN sends a Block report

extra replica is deleted.

The metadata is updated dynamically to ensure consistency.



The client will have no idea about failure

Permanent failure

- Hardware failure
- Disk corruption
- Decommissioning the node

> 10 min heartbeat

Process during Permanent failure

1. Detection

NN will declare DN as dead due to missing heartbeat

2. Replication

- re-replicate the lost blocks to other data nodes.
- NN updates its metadata to point to new copies & remove reference from previous failed DN

3. If failed Machine comes back:

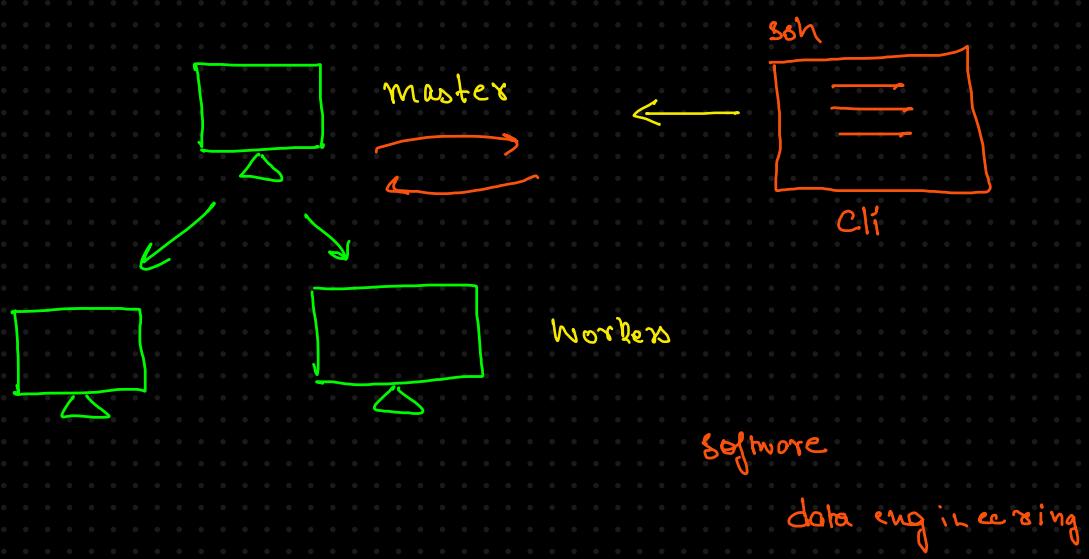
the returning DN is not trusted & its blocks are ignored and cleaned up.

Aspect	Temporary Failure	Permanent Failure
Failure Cause	Network issues, software crashes, or reboots.	Hardware failures, disk corruption, or decommissioning.
Detection	DataNode stops sending heartbeat signals; marked as temporarily unavailable.	DataNode stops sending heartbeat signals for an extended period; marked as dead.
Replication Trigger	Blocks on the failed DataNode are marked as "under-replicated," and new replicas are created.	Blocks on the failed DataNode are treated as lost, and new replicas are created.
Returning DataNode Behavior	Sends a block report to the NameNode. Excess replicas are deleted to restore the replication factor.	Returning DataNode is not trusted. Its blocks are treated as stale and eventually cleaned up.
Metadata Updates	Metadata dynamically updates to include new replicas and remove excess copies.	Metadata permanently removes blocks from the failed DataNode and updates mappings for new replicas.
Library Analogy	A locked bookshelf temporarily restricts access. Extra copies are reconciled when it becomes available.	A destroyed bookshelf is permanently removed from the catalog, and its books are replaced elsewhere.

Few Takeaways

1. HDFS handles failures gracefully.
2. Metadata is the backbone of hdfs.
3. Replication factor is strictly maintained.

1. Google cloud → \$300 credits + 3 month trial
⇒ Credit/debit
UPI
Netbanking
2. Dataproc → managed Hadoop clusters
↳ enable the dataproc API



Name node failures

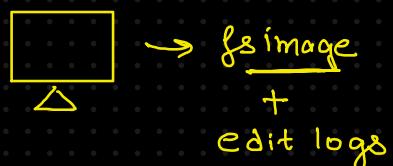
SPOF → Single point of failure

NN is the most critical component of your HDFS

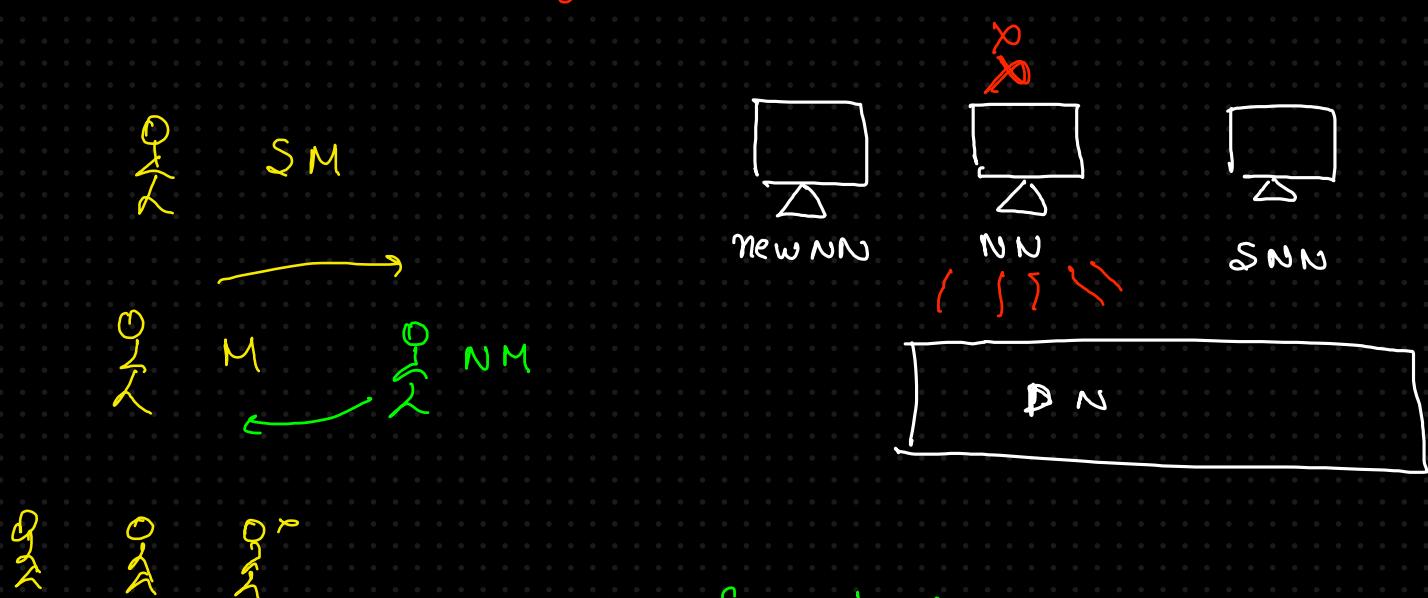
Q: What is the difference between a Secondary NN & a Standby NN?

Secondary Name node

it is a helper node that checkpoints
the primary Name node.



When our main NN dies, SNN will have all information
to bring up a new NN upto the point where it can
behave like previous NN. It is not a failover mechanism,
it cannot take role of NN.



Some downtime will be here
till the new NN has
been updated via SNN.

Librarian \Rightarrow

helper

Standby NN

Standby NN takes role of Active as soon as Active Name Node gets killed

\rightarrow hot backup for NN

\rightarrow constant connection

\rightarrow makes copy of namenode image in memory

1. High Availability

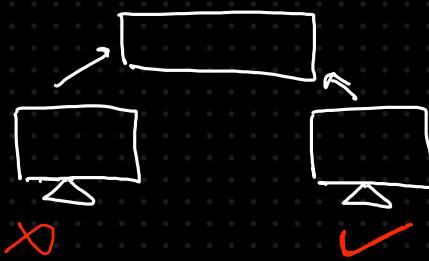
1. Standby NN will maintain

a synchronized copy of metadata in memory.

2. Synchronization happens via shared edits: Your ANN and Standby NN has access to a shared directory (Journal Node)

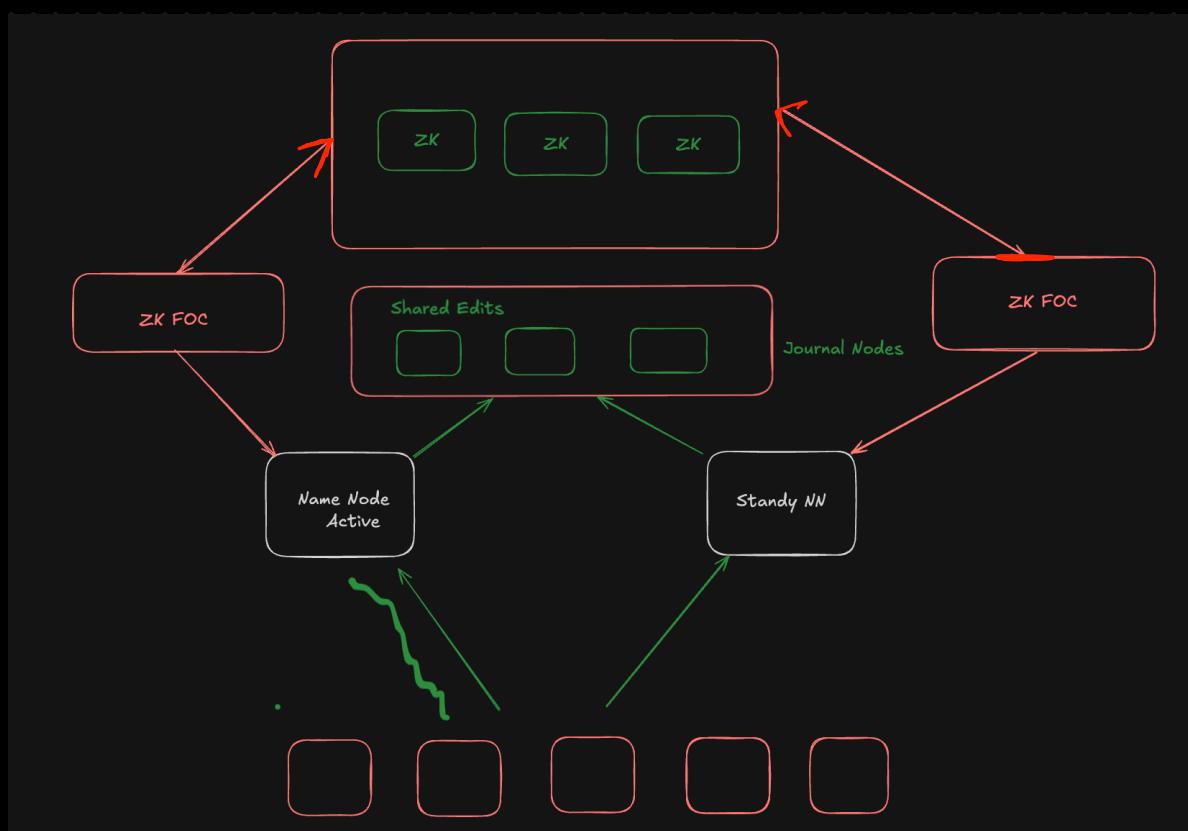
2. Failure mechanism

Standby Name node can take role of Active NN, ensuring minimum downtime & data loss.



Aspect	Secondary NameNode	Standby NameNode
Role	Helper for checkpointing and edits log management.	Hot backup for Active NameNode, enabling seamless failover.
High Availability	Does not provide HA; cannot replace the NameNode.	Ensures HA by taking over the Active NameNode's role on failure.
Synchronization	No continuous synchronization; works periodically.	Continuously synchronized with the Active NameNode.
Failover Capability	Not capable of failover; only provides updated fsimage.	Fully capable of taking over in case of Active NameNode failure.
Introduced In	Hadoop 1.x	Hadoop 2.x

Hadoop High Availability architecture



Component	Role	Explanation
Active NameNode	Main controller managing HDFS namespace and block information.	Handles all client operations like file creation, deletion, and metadata management.
Standby NameNode	Hot backup of the Active NameNode, ready to take over during failure.	Synchronizes its state with the Active NameNode using edit logs stored in JournalNodes.
DataNodes (DN)	Store actual data blocks and handle data read/write operations.	Send regular block reports to both Active and Standby NameNodes.
JournalNodes (JN)	Maintain a shared edit log that keeps both NameNodes synchronized.	Facilitate consistent updates to the Standby NameNode to ensure it has the latest changes in the HDFS namespace.
Zookeeper (ZK)	Coordinates failover and prevents split-brain scenarios.	Monitors NameNode states and triggers leader election during failover.
FailoverController	Monitors the health of NameNodes and initiates failover.	Works with Zookeeper to perform automatic failover when the Active NameNode is unresponsive.
Heartbeat Mechanism	Periodic signals exchanged between components to check health.	Ensures timely detection of failures in NameNodes or DataNodes.
Block Reports	Periodic reports from DataNodes to NameNodes about stored blocks.	Ensures both Active and Standby NameNodes have accurate information about the location of data blocks in the cluster.
Clients	Users or applications interacting with the Hadoop cluster.	Access HDFS files through the Active NameNode. Failover is transparent to clients, ensuring uninterrupted service.

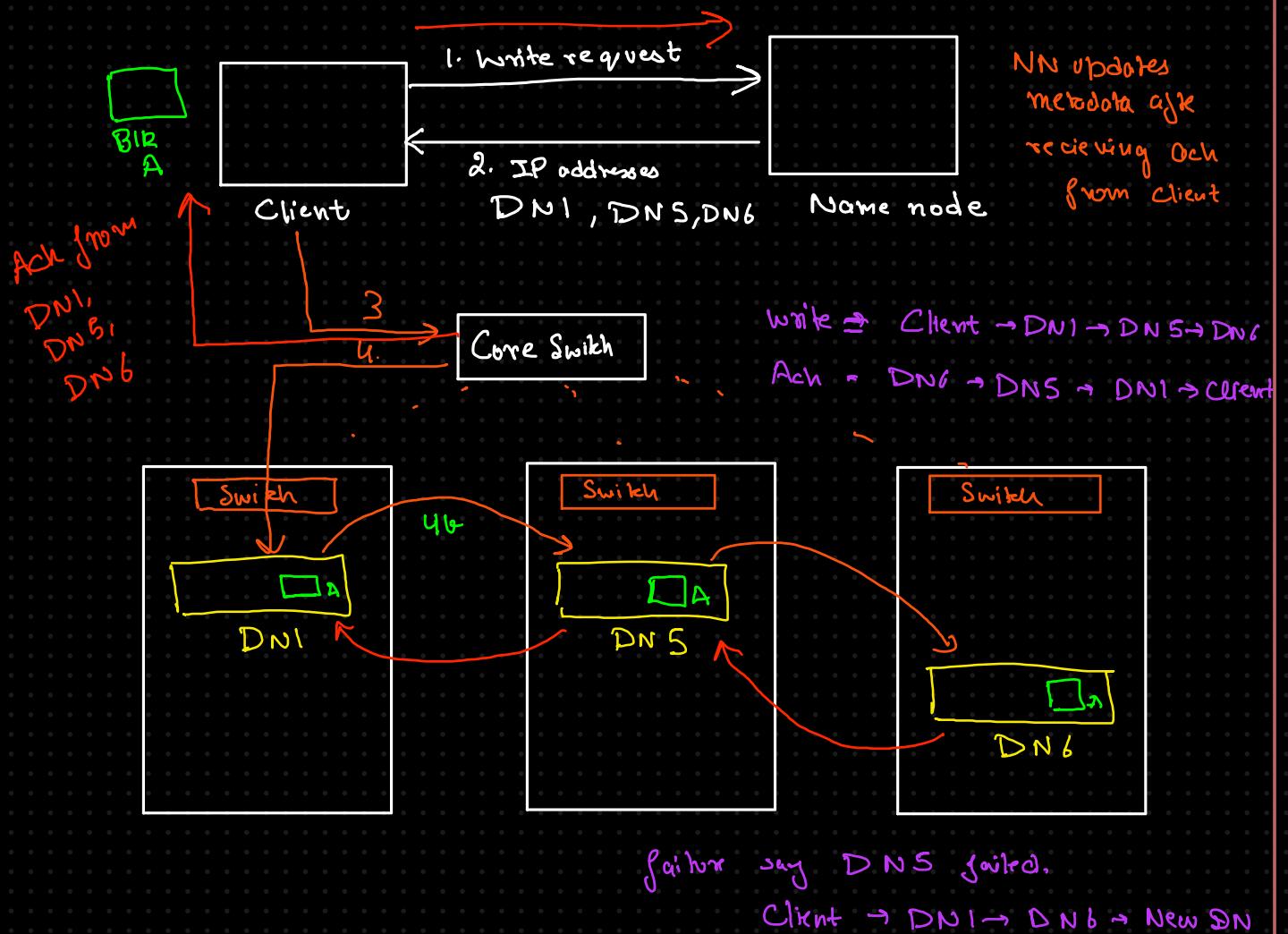
HDFS

- architecture of hdfs
- Replication, blocks, rack
- Data Node failures
- Name Node failures
- hdfs write
- hdfs read

Write operation happens in hdfs

3 steps process

1. Interact with NN
2. moving actual data to DN
3. Acknowledgement



Fault tolerance & Failure handling

1. Data node failed during write

