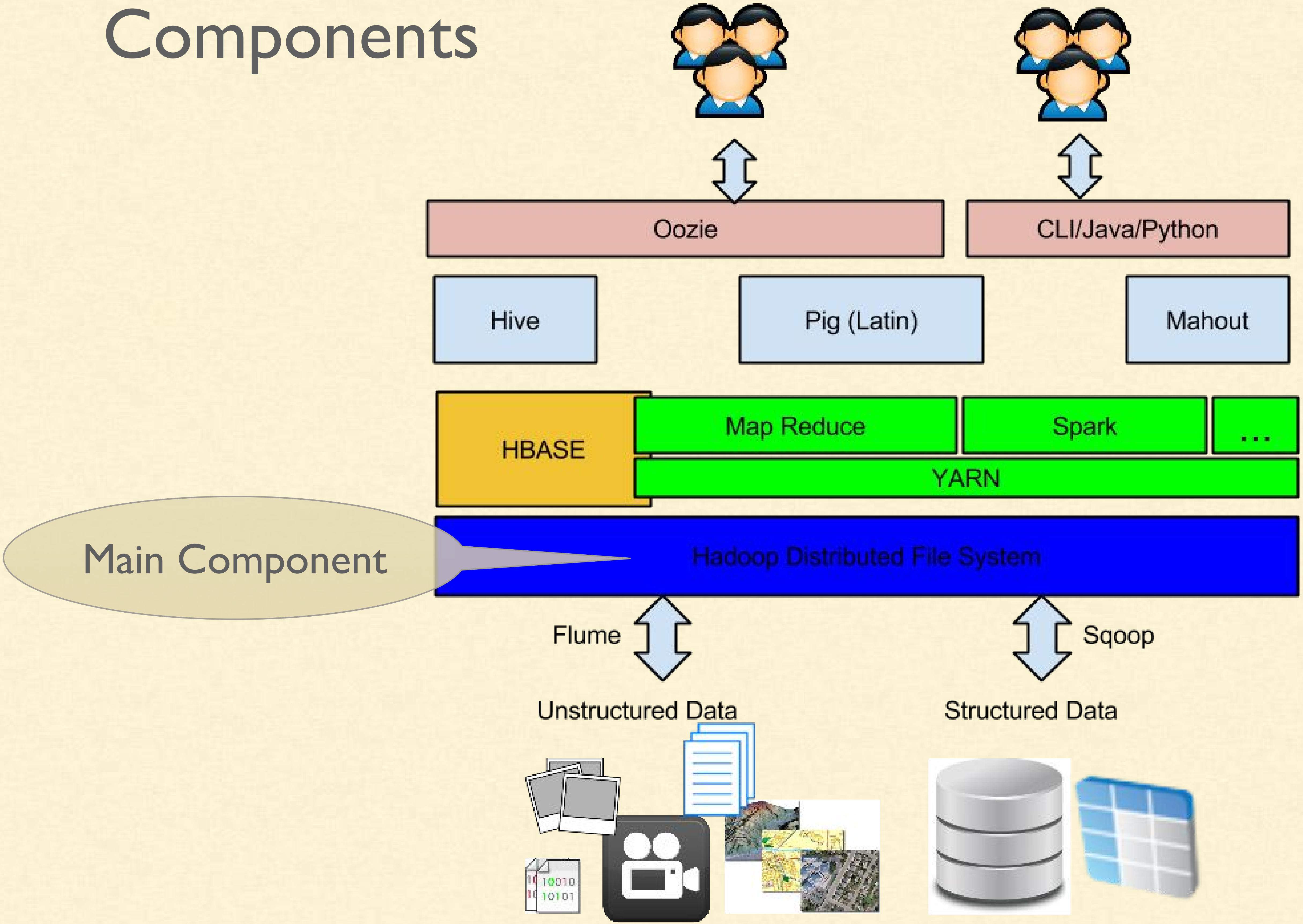
Welcome to Hadoop Distributed File System (HDFS)

Components


Oozie
CLI/Java/Python
Hive
Pig (Latin)
Mahout
HBASE
Map Reduce
Spark
...
YARN
Main Component
Hadoop Distributed File System
Flume
Sqoop
Unstructured Data
Structured Data

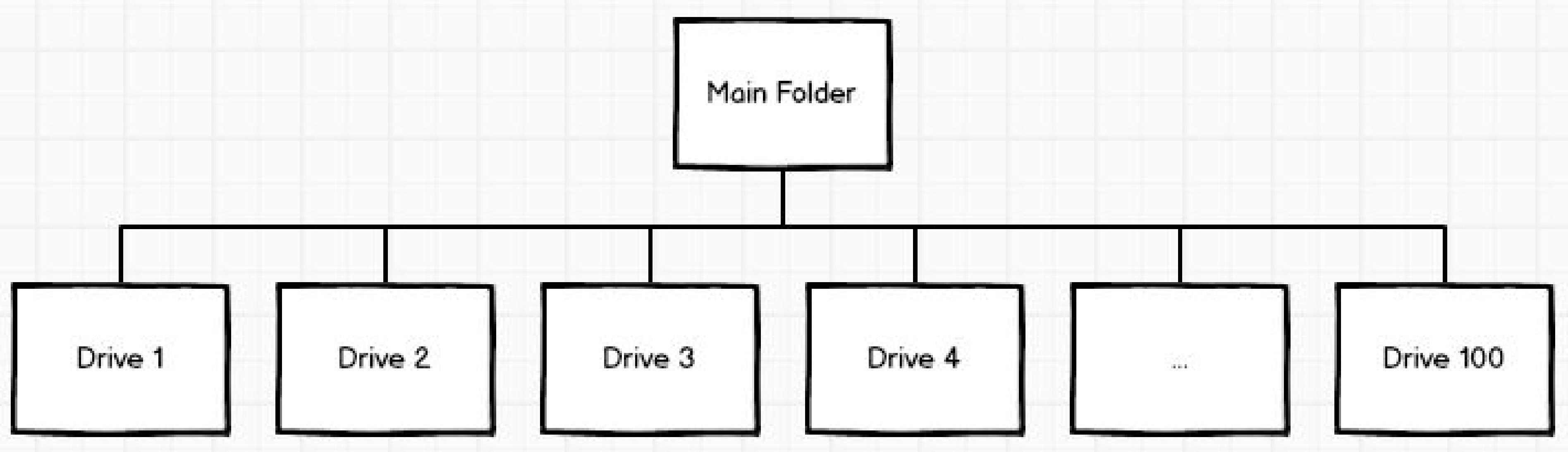
Big Data Problem - Storage

Question: If you have 100TB data, How would you store it?

Big Data Problem - Storage - Approach

Build NAS or SAN

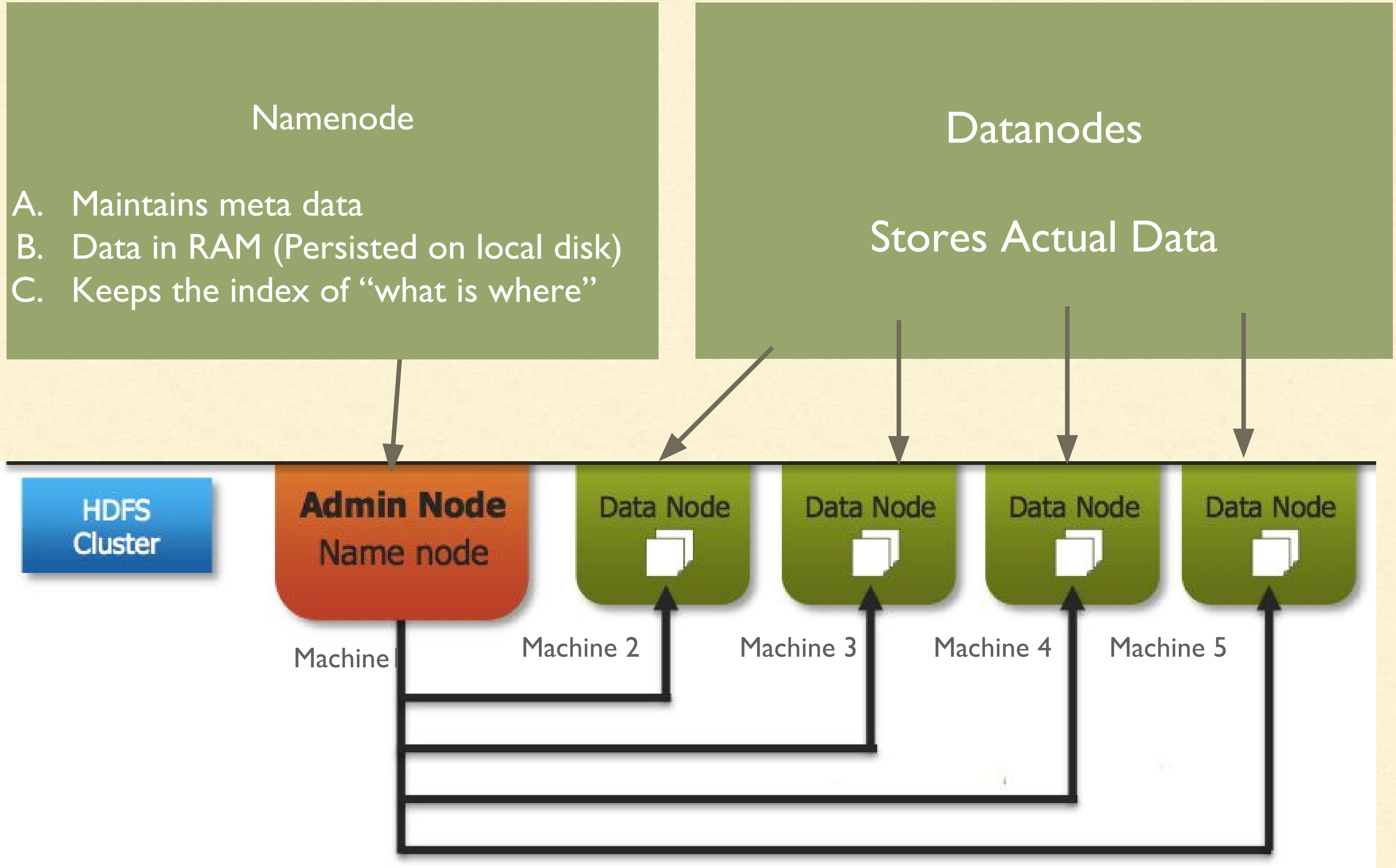
Have 100 1TB drives and make 100 subfolders mount these.



Big Data Problem - Storage - Challenges

- What about failovers & backups?
- How do we distribute the data uniformly across disks?
- Is this best use of resources?
- How do we handle frequent access to files?
- How do we scale out?

HDFS - Namenodes and Datanodes



Namenode

A. Maintains meta data
B. Data in RAM (Persisted on local disk)
C. Keeps the index of “what is where”

Datanodes

Stores Actual Data

HDFS - Design

1. Very large files
2. Streaming data access
3. Commodity hardware

HDFS - Limitations

1. Low-latency data access

2. Lots of small files

3. Multiple writers, arbitrary file modifications

HDFS - Blocks

1. The files are split into a chunk of 128M blocks.

HDFS - Blocks

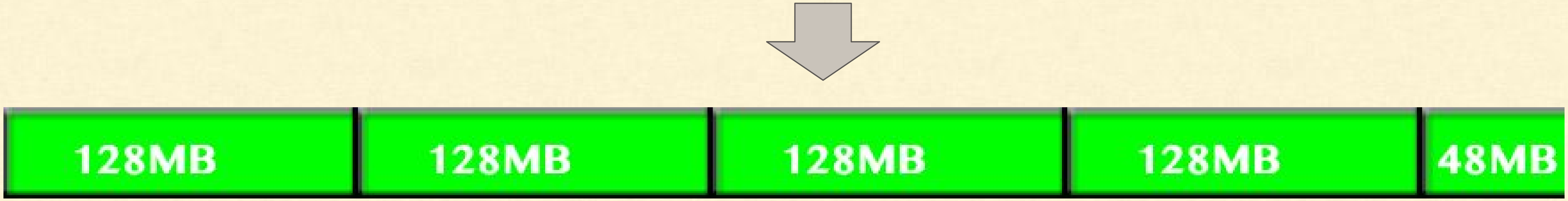
1. The files are split into a chunk of 128M blocks.

560 MB File => How Many Blocks and of what sizes?

HDFS - Blocks

1. The files are split into a chunk of 128M blocks.

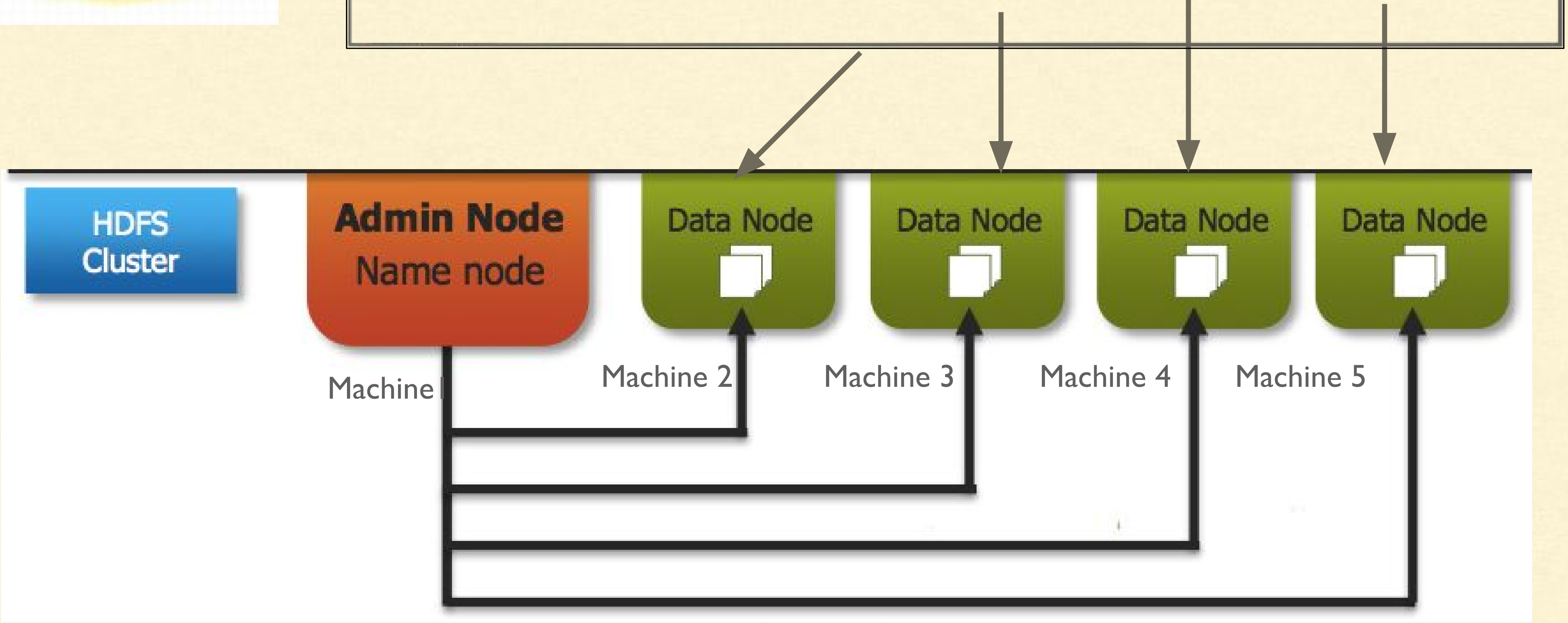
560 MB File => How Many Blocks and of what sizes?



128MB 128MB 128MB 128MB 48MB

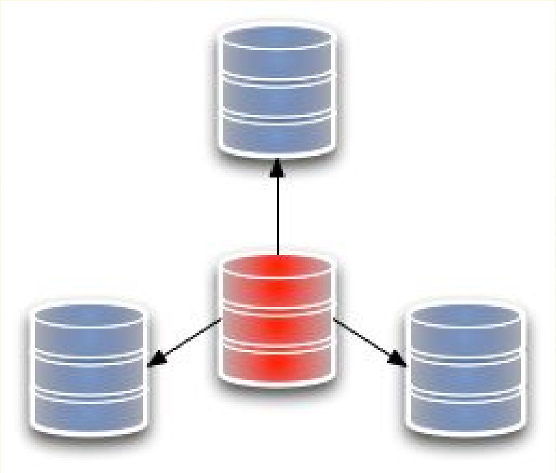
HDFS - Blocks - Advantages

1. Helps fitting big files into small discs
2. Leaves less unused space on the disc.
3. Optimises the transfer
4. Distributes the load to multiple machines

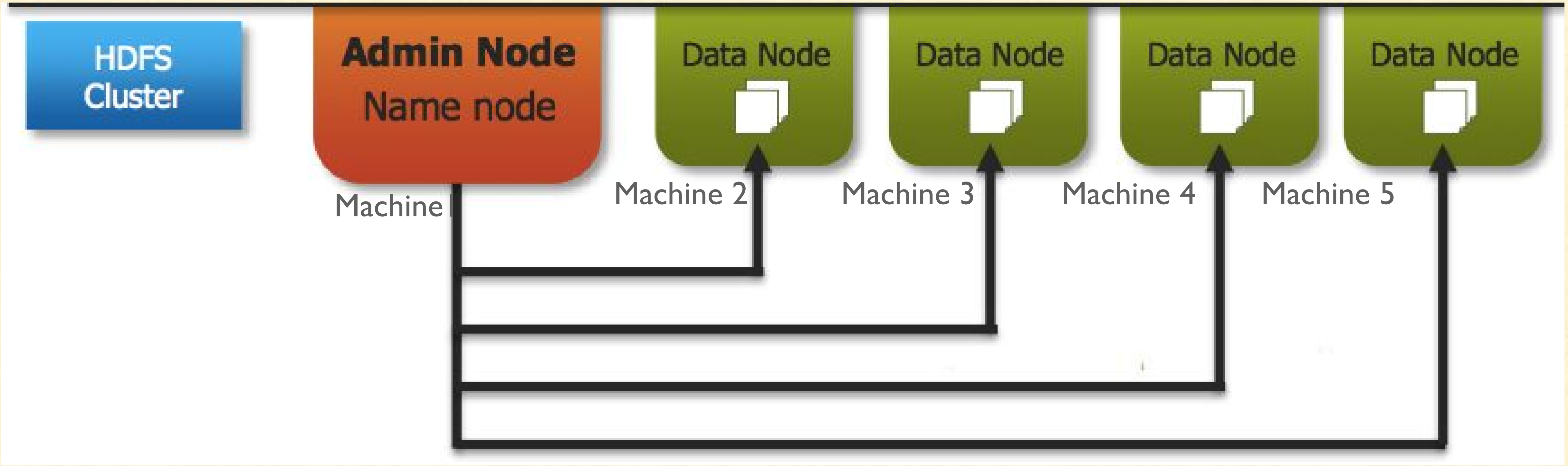


CLOUD x LAB

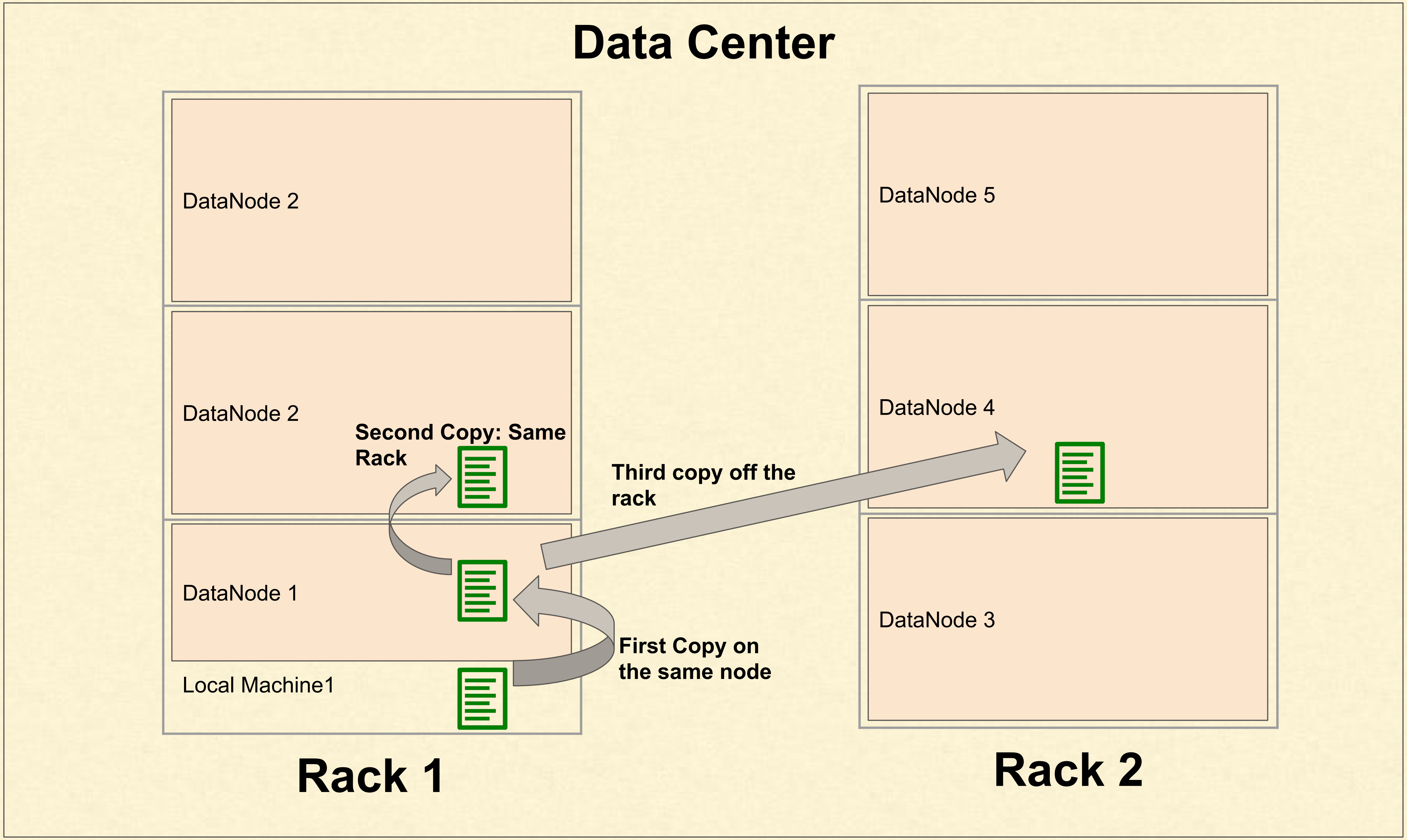
HDFS - Replication



Each block has multiple copies
Called Replication Factor, default 3.
No two copies are on same data node.
Data node fails => The lost blocks are copied to other nodes.

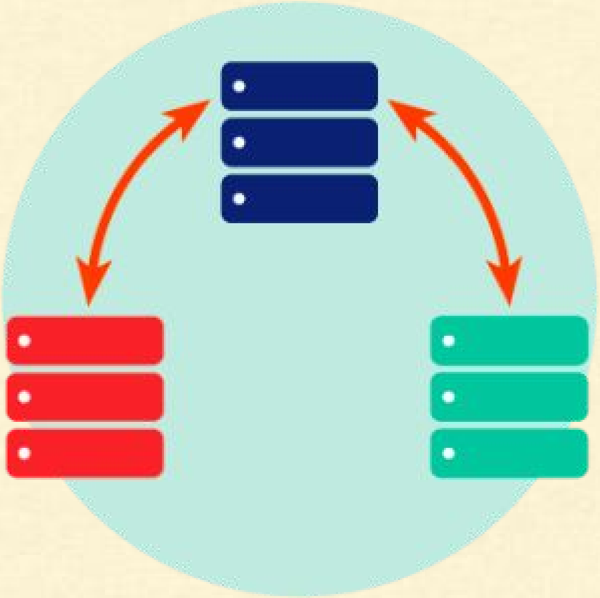


HDFS - Replica Placement Policy



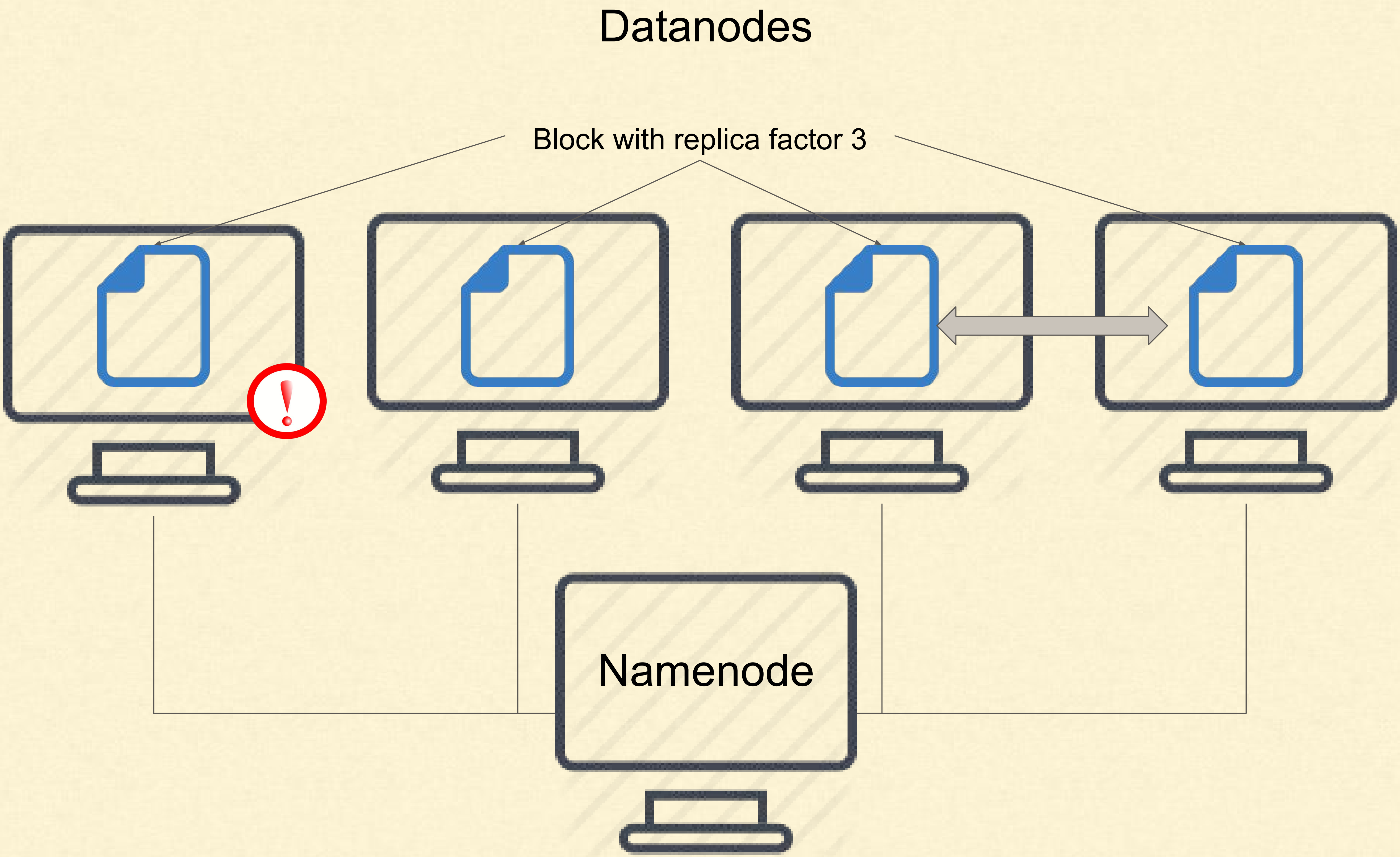
CLOUD x LAB

HDFS - Replication Factor

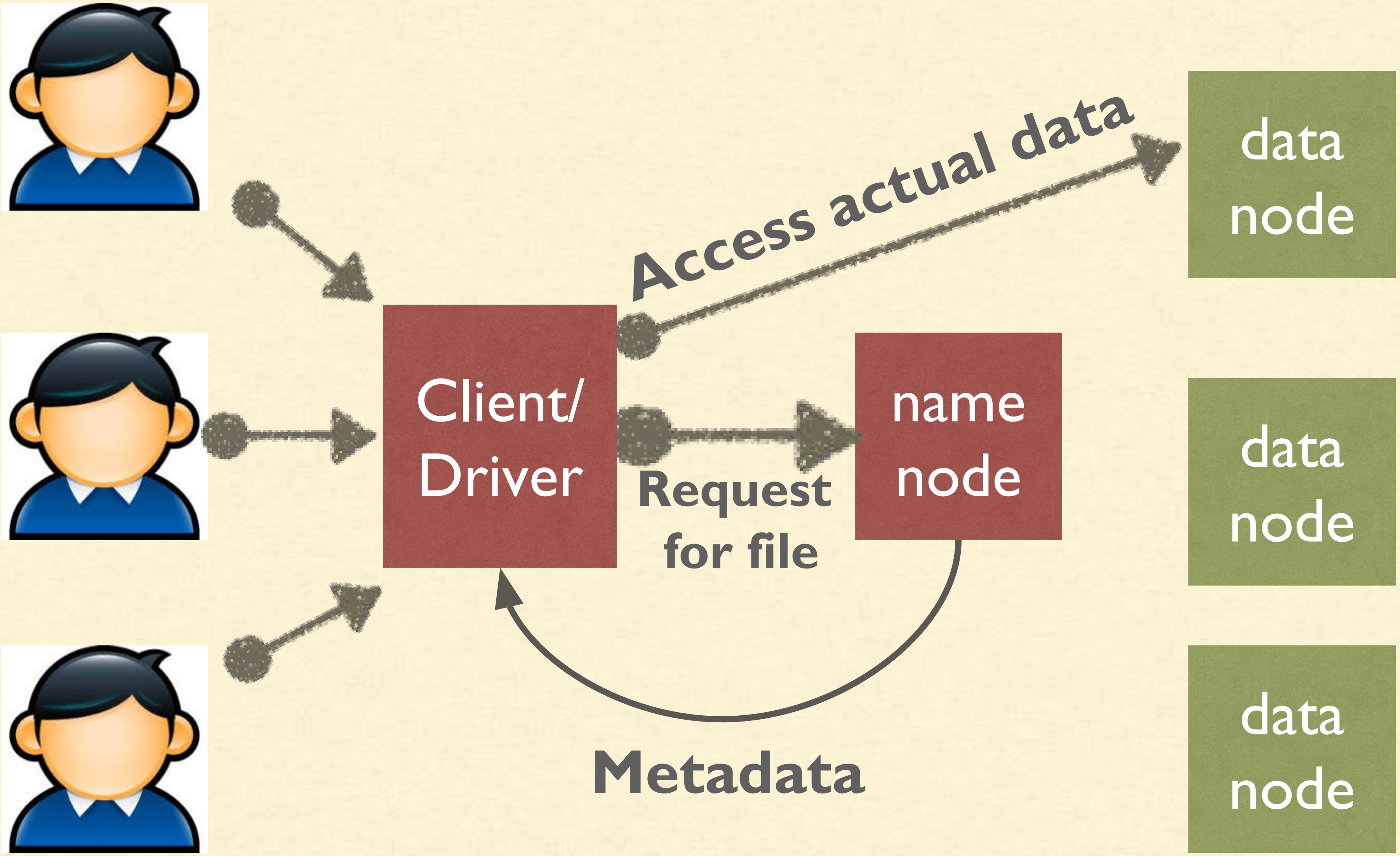


- Default and recommended Replication factor : 3
- Can be set for the entire HDFS
- Can be set for individual files
- For not so important file - you can decrease the replication factor
- Increase replication factor for important files

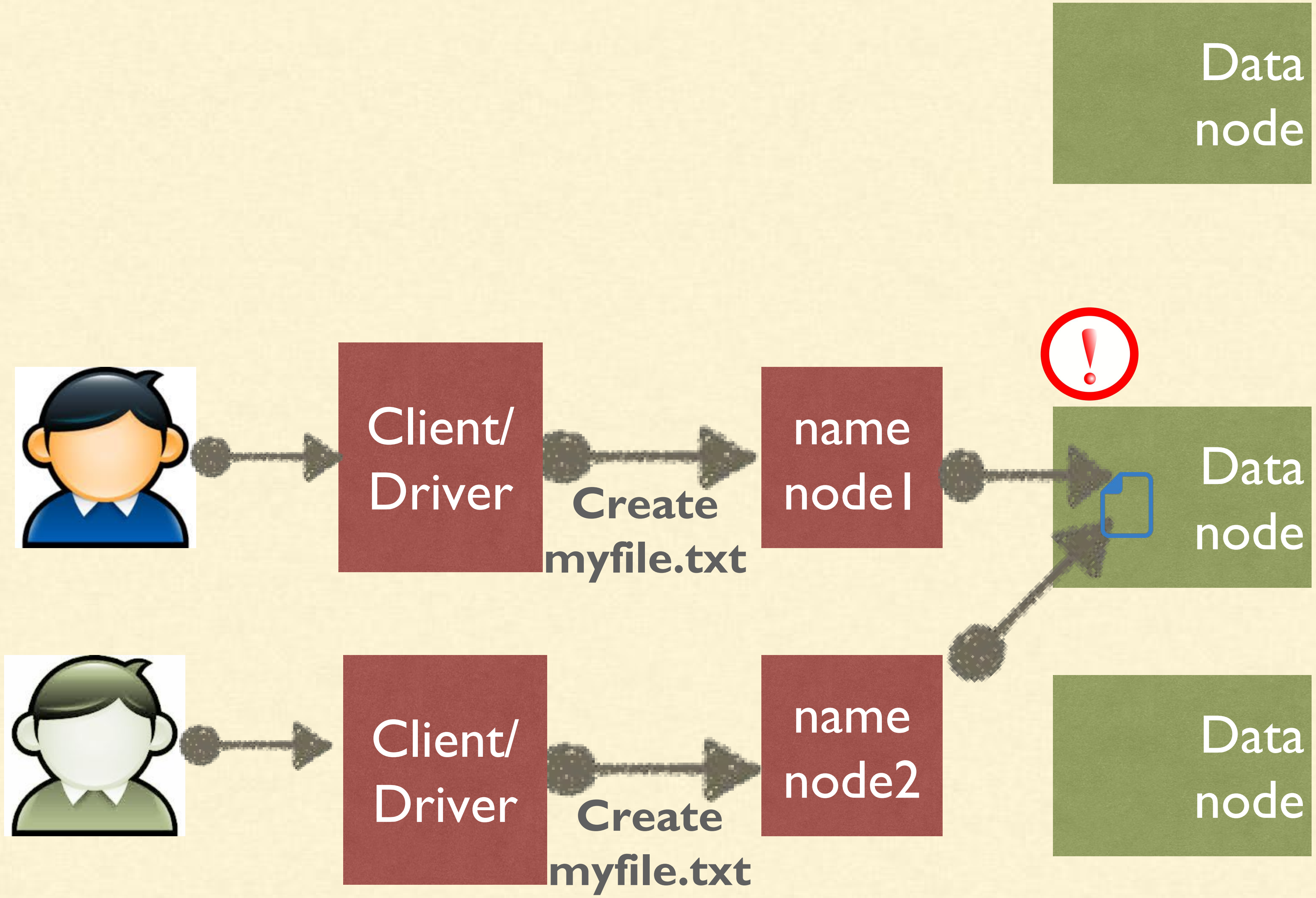
HDFS - Replication - Failover



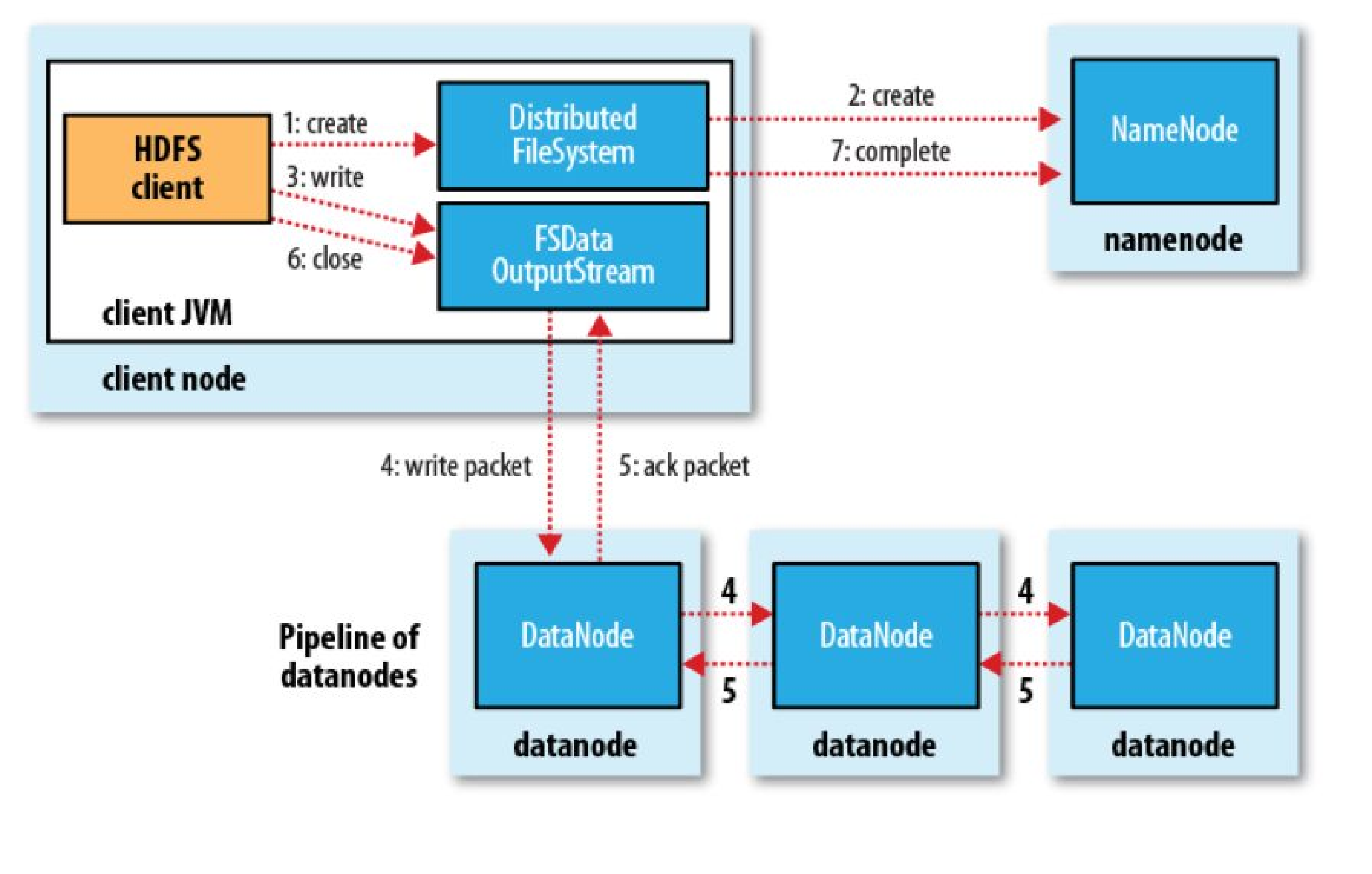
HDFS - Driver


Access actual data
data node
Client/ Driver
name node
Request for file
data node
data node
Metadata

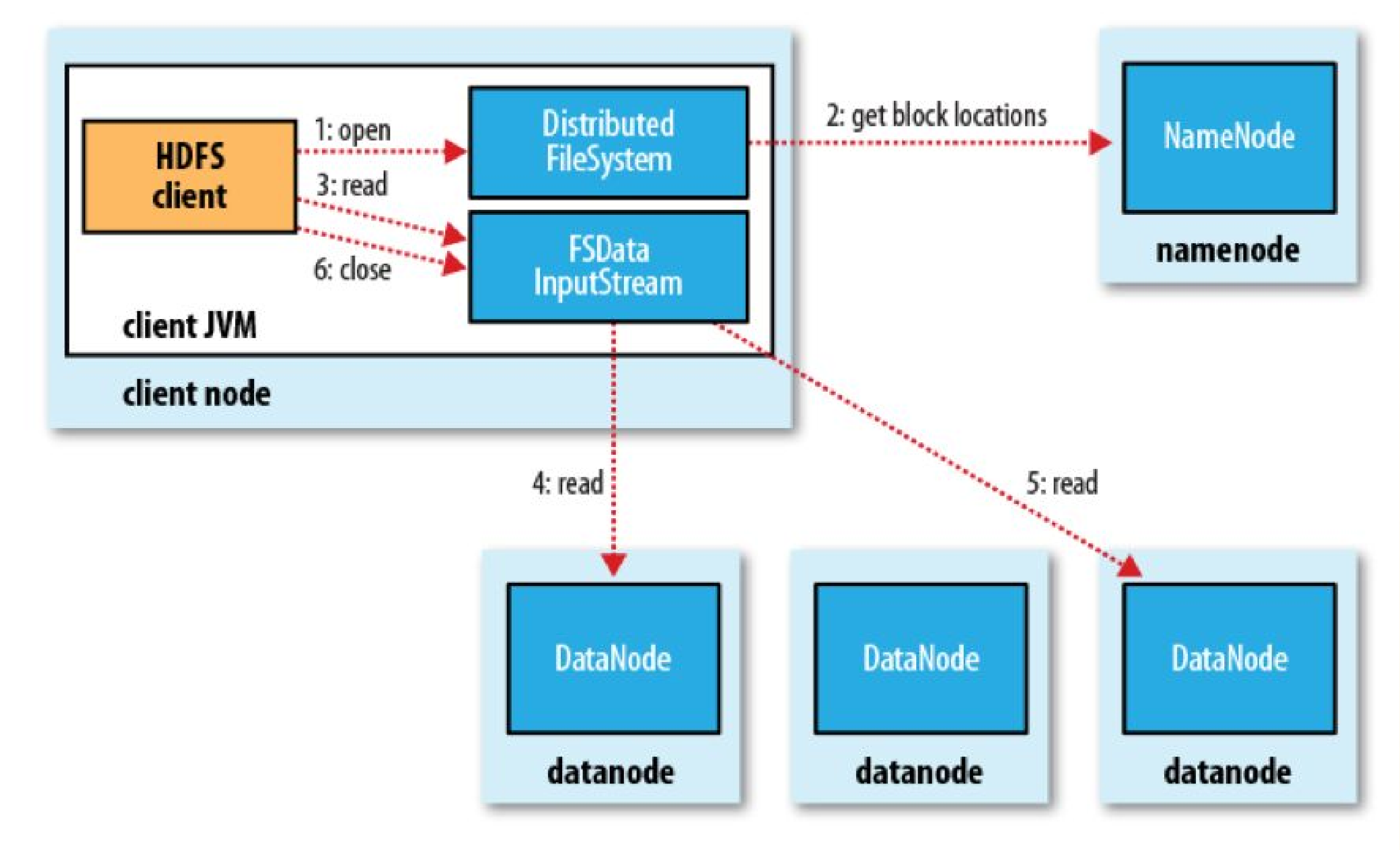
HDFS - If multiple namenodes?



Anatomy of a File Write

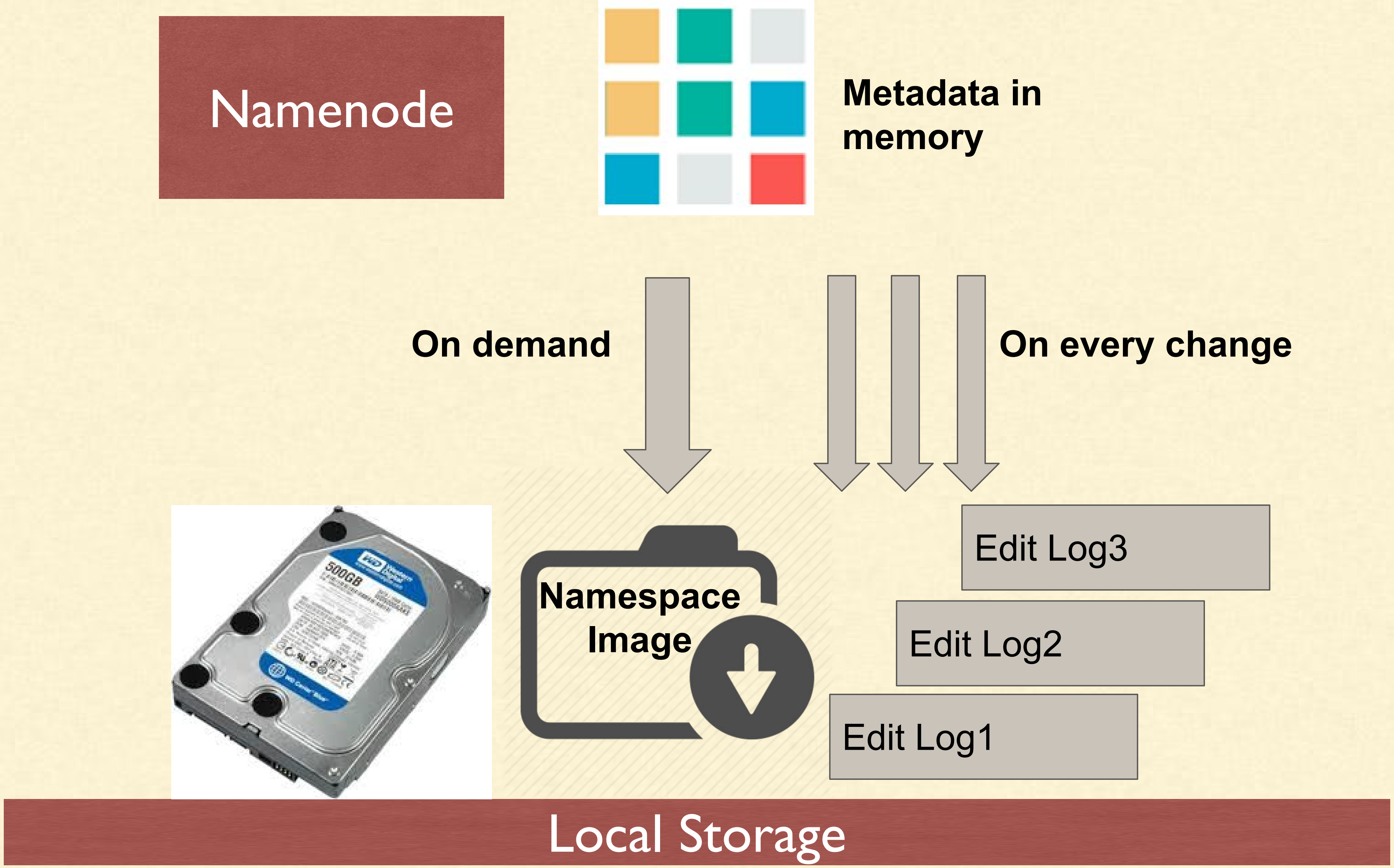


Anatomy of a File Read



Namenode - Metadata

1. Keeping a backup of namespace image & editlogs on two locations



Namenode

Metadata in memory

On demand

On every change

Namespace Image

Edit Log3

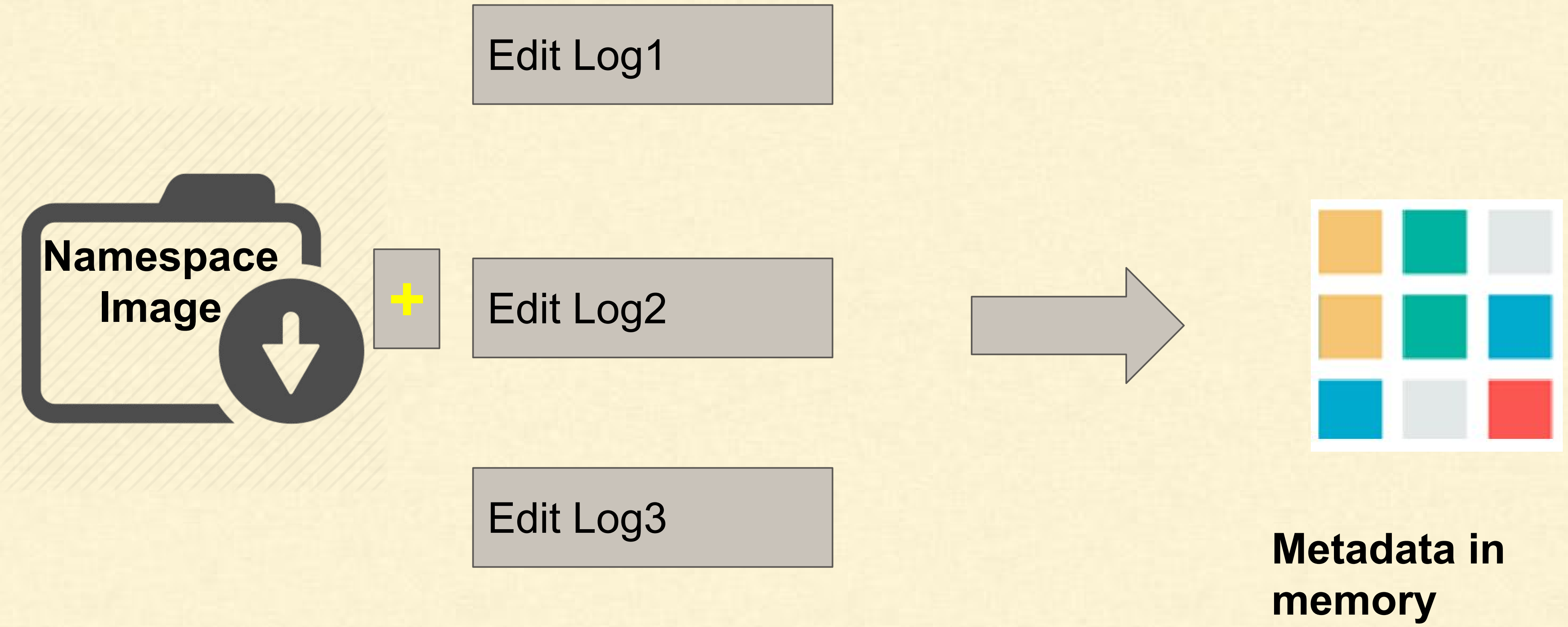
Edit Log2

Edit Log1

Local Storage

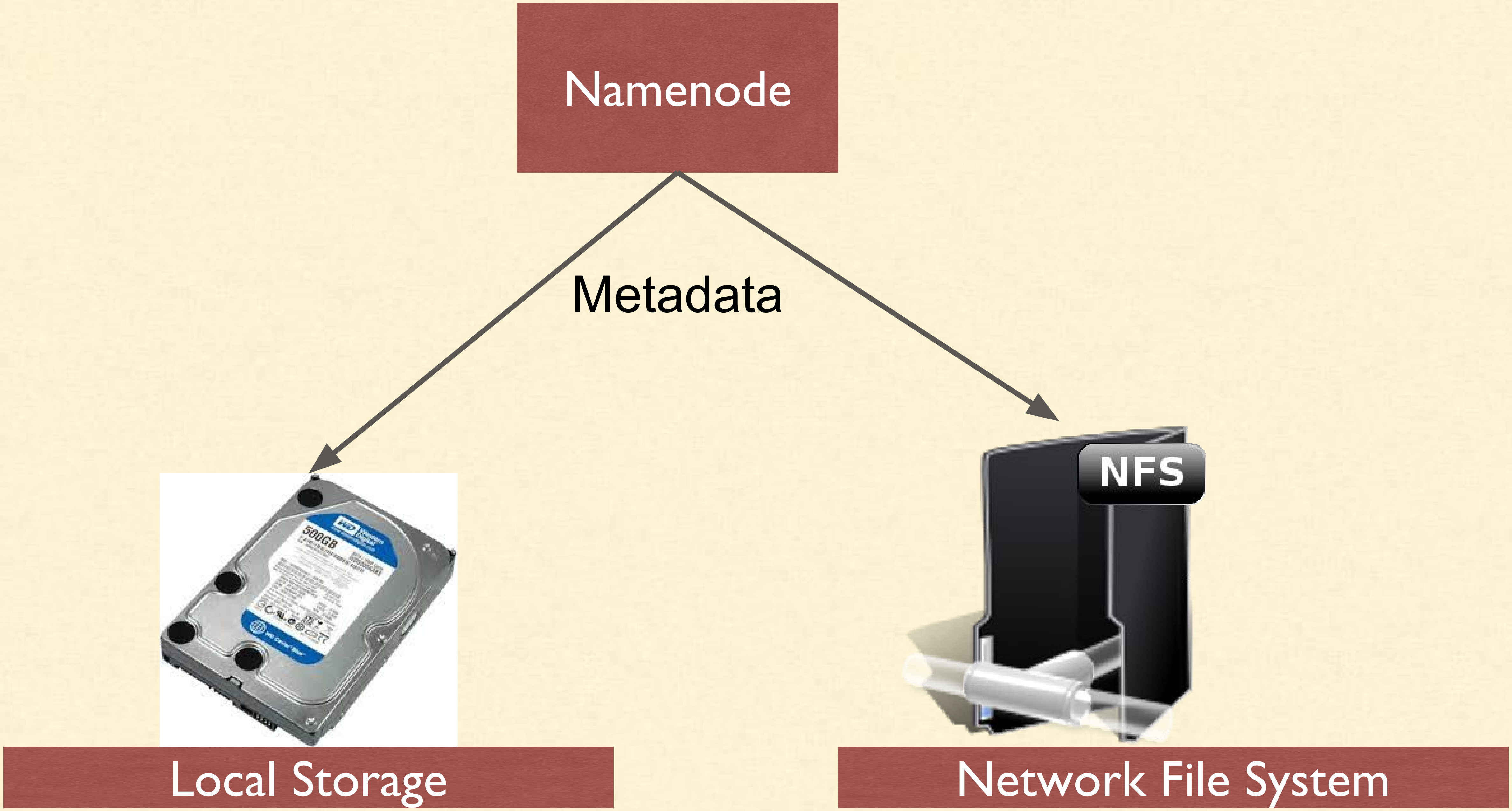
Namenode - Single Point of Failure

1. To recover metadata we needed to merge the image with editlogs.



Namenode - Single Point of Failure

1. Keeping a backup of namespace image & editlogs on two locations



Namenode

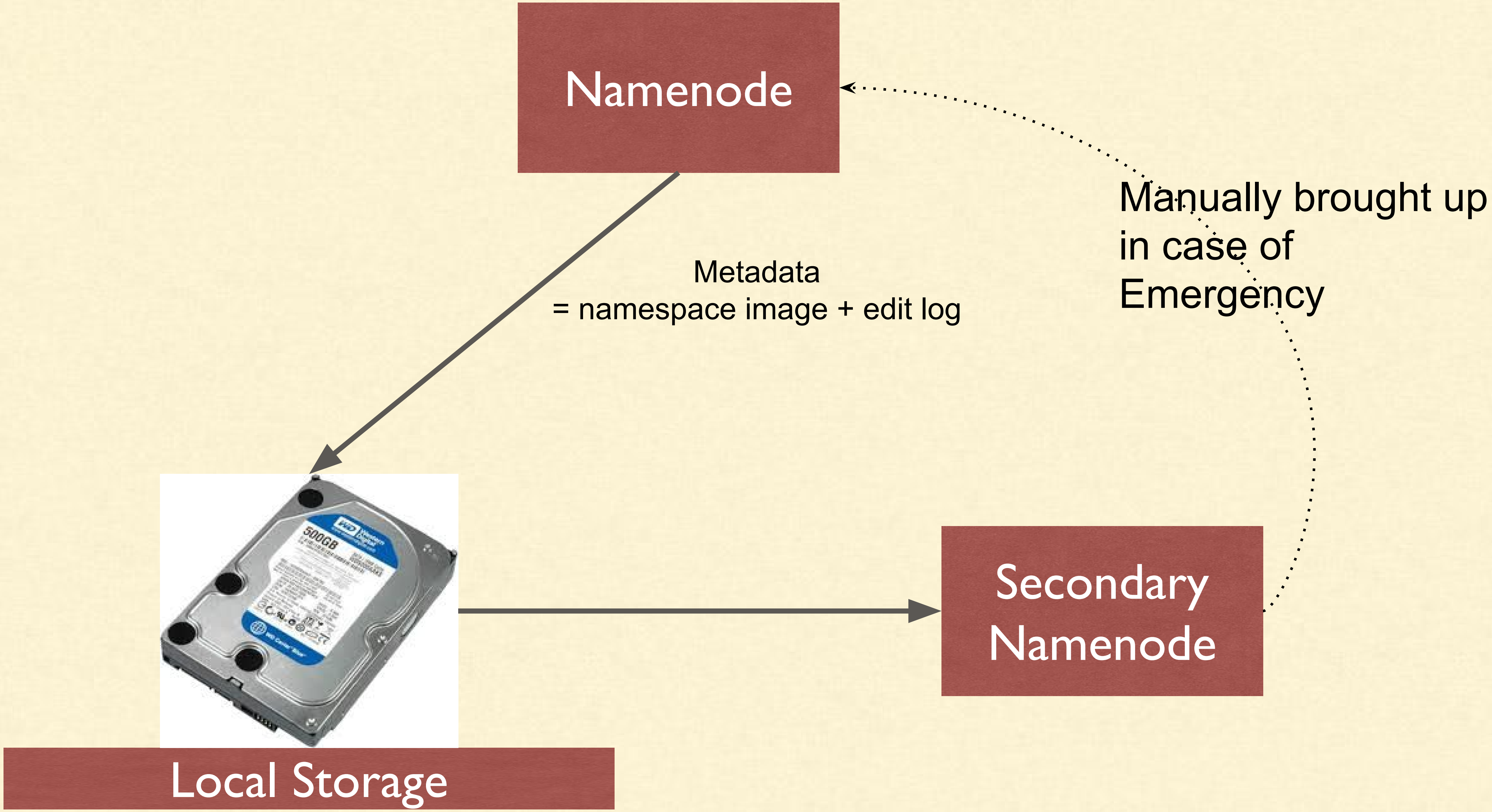
Metadata

Local Storage

Network File System

Namenode - Single Point of Failure

2. Keeping A Secondary Namenode



Namenode

Metadata
= namespace image + edit log

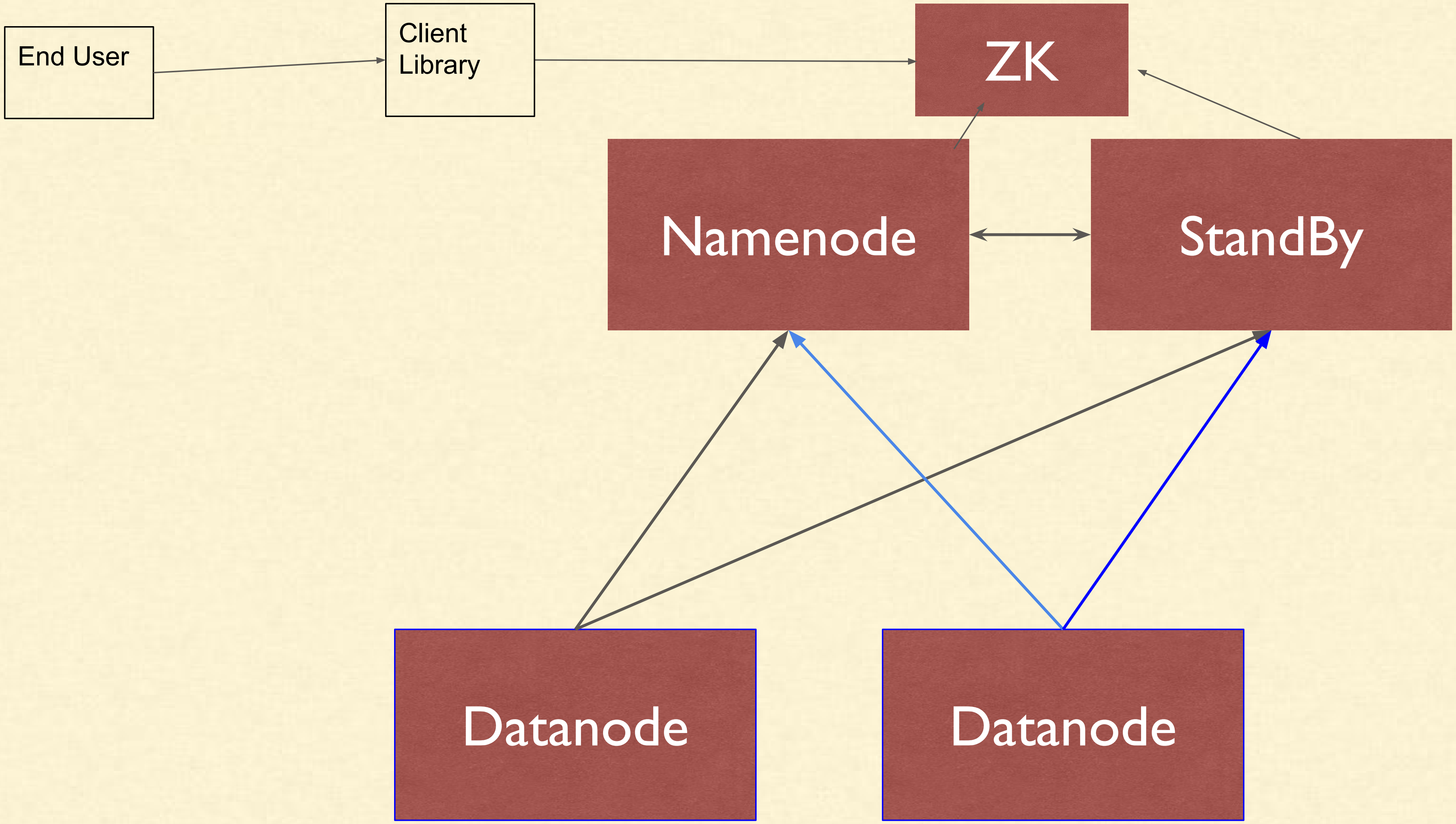
Manually brought up in case of Emergency

Local Storage

Secondary Namenode

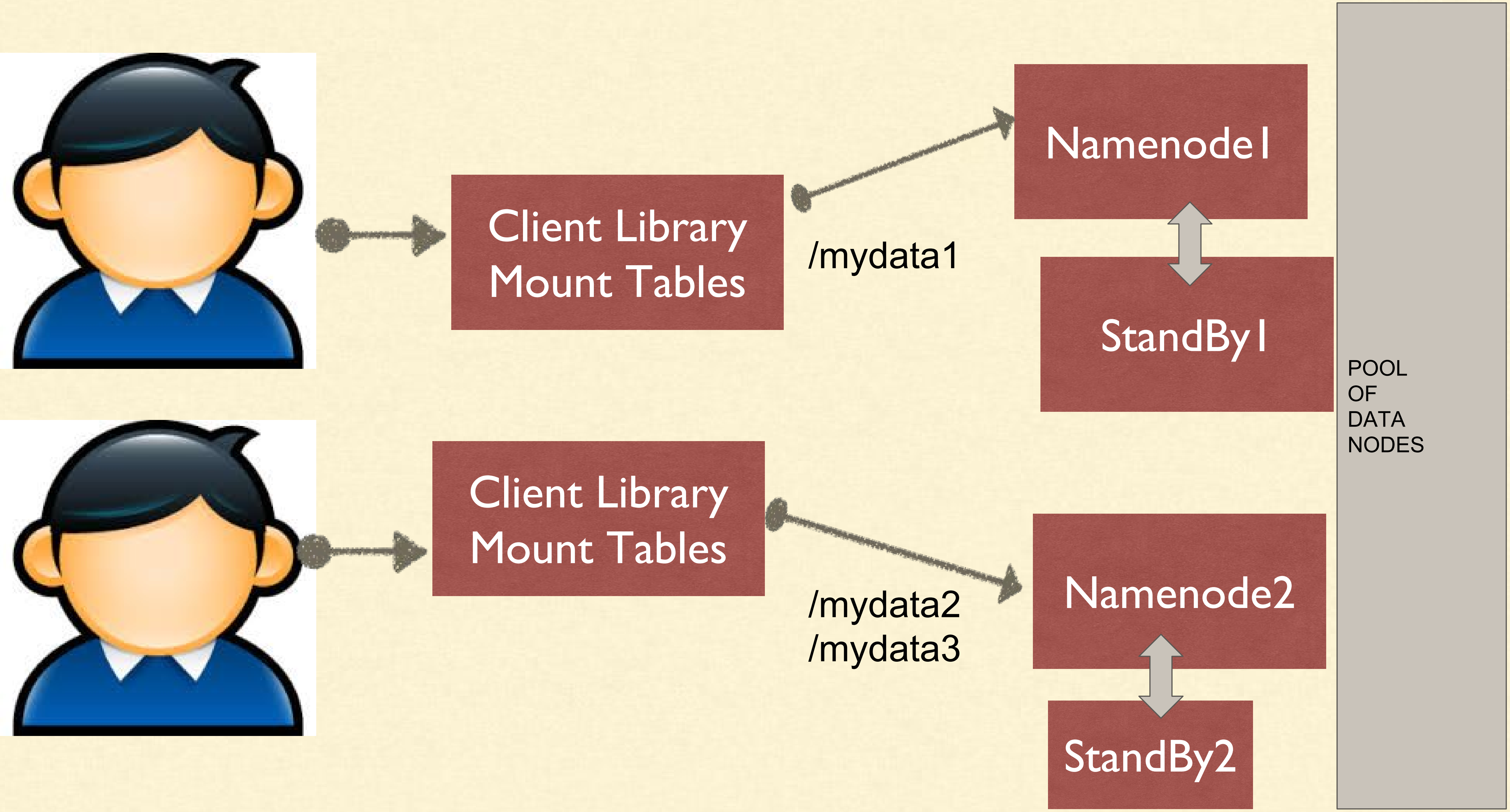
Namenode - Single Point of Failure?

3. HDFS high availability (HA) - Quorum journal manager (QJM) (in 2.x)



HDFS Federation (2.x)

Multiple Namenodes - different namenode for different sub folders



Client Library
Mount Tables

/mydata1

Namenode1

StandBy1

Client Library
Mount Tables

/mydata2
/mydata3

Namenode2

StandBy2

POOL
OF
DATA
NODES

Namenode Metadata

Meta-data

1. The entire metadata is in main memory.

Types of Metadata

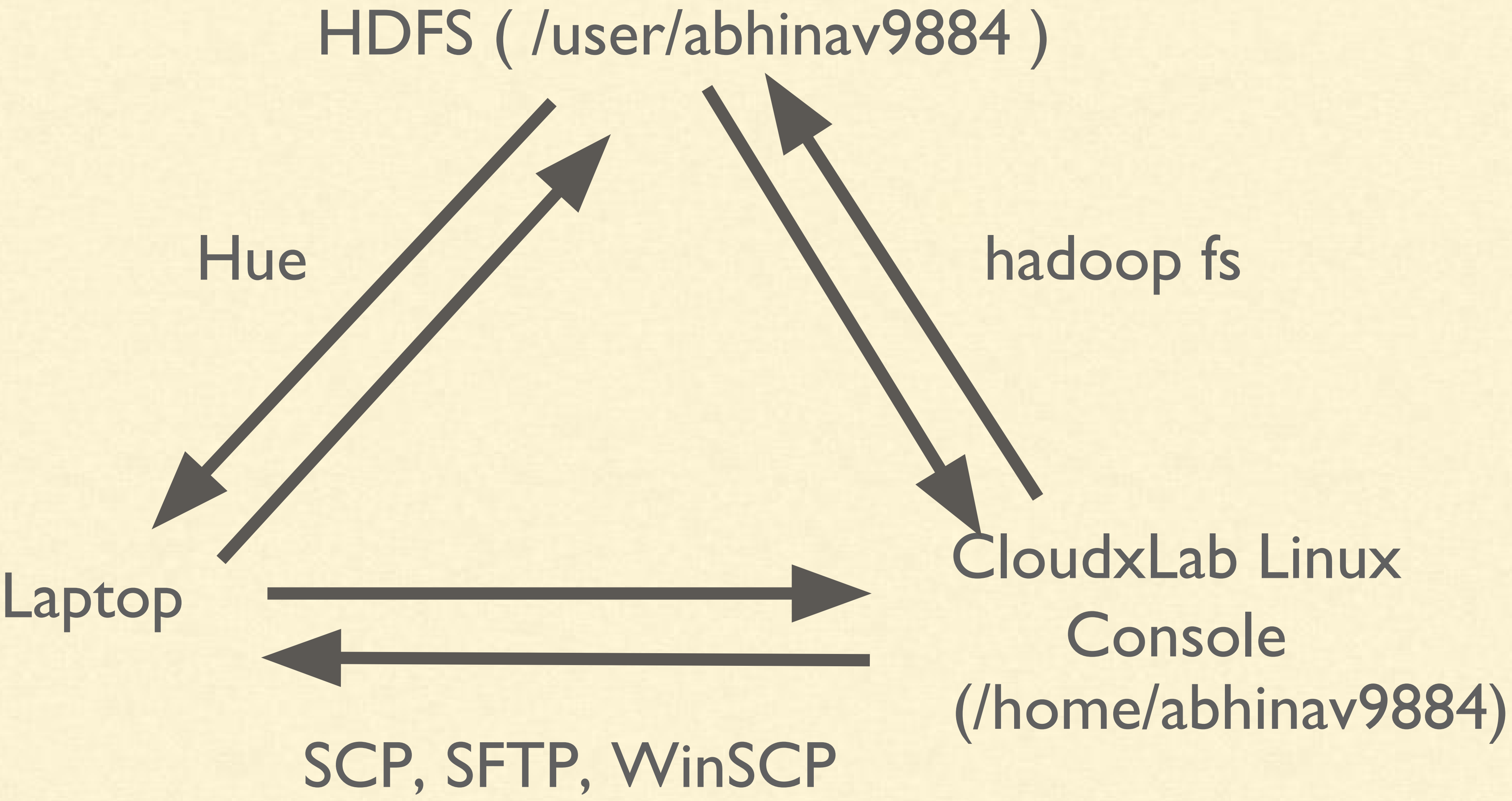
1. List of files
2. List of Blocks for each file
3. List of DataNode for each block
4. File attributes, e.g. access time, replication factor, file size, file name, directory name

A Transaction Log or Edit Log

1. Records file creations, file deletions. etc

HDFS - Hands-on

How to Transfer Files?



HDFS (/user/abhinav9884)

Hue

hadoop fs

Laptop

CloudxLab Linux Console (/home/abhinav9884)

SCP, SFTP, WinSCP

HDFS - Access files

1. *hadoop fs -ls sample.txt*
2. *hadoop fs -ls /user/abhinav9884/sample.txt*
3. *hadoop fs -ls hdfs:///user/abhinav9884/sample.txt*
4. *hadoop fs -ls hdfs://ip-172-31-53-48.ec2.internal:8020/user/abhinav9884/sample.txt*

Where *ip-172-31-53-48.ec2.internal* is the namenode host

Where are the blocks?

```
hdfs fsck -blocks -locations -racks  -files /user/abhinav9884/sample.txt
```

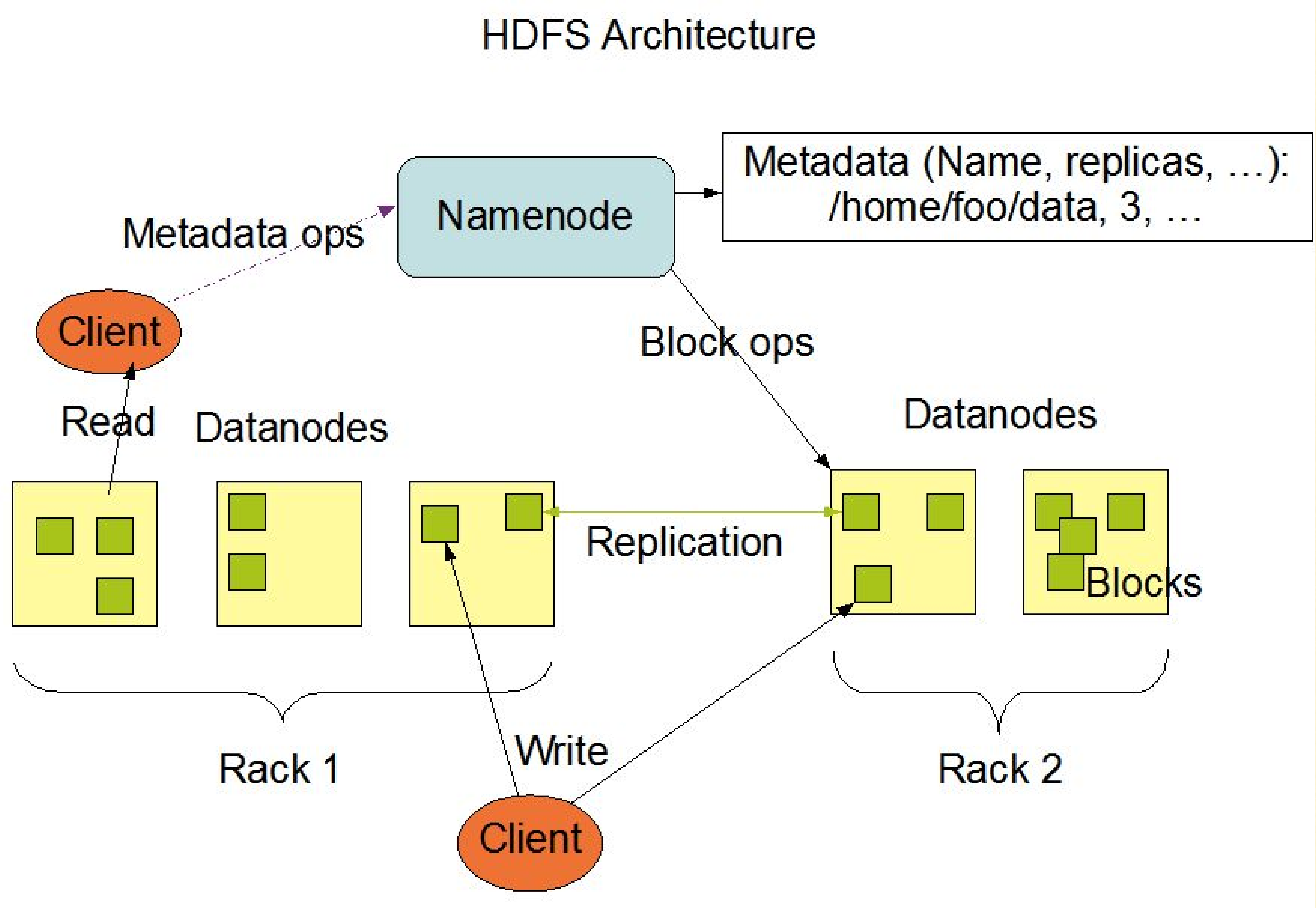
Set Replication

```
# To set replication factor as 1

hadoop fs -setrep -w 1 /user/abhinav9884/sample.txt

# Check blocks

hdfs fsck -blocks -locations -racks  -files /user/abhinav9884/sample.txt
```

Summary

- HDFS
- HDFS - Design
- HDFS - Namenode, Datanode, Secondary Namenode, Standby Namenode and High Availability
- Hands-on demos

Thank You!


HDFS Architecture
Metadata ops
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

TODAY'S CLASS

- Recap
    - HDFS Architecture
    - Hadoop 1.0 Architecture
    - Hadoop 2.0 / Yarn
- Cluster Overview
    - Hue
- Using HDFS, Hive, Pig, Oozie
    - From Web
    - From Command