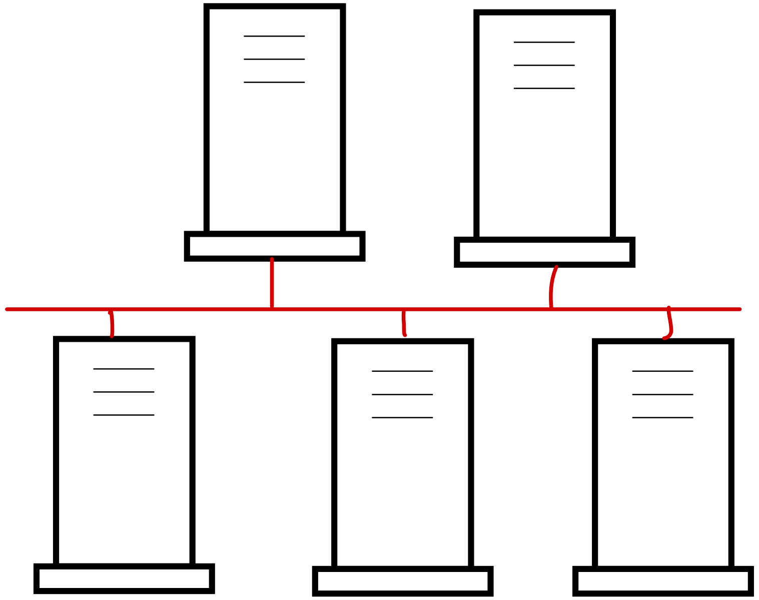
# Big Data Frameworks

Trainer: Mr. Nilesh Ghule

# Distributed Systems

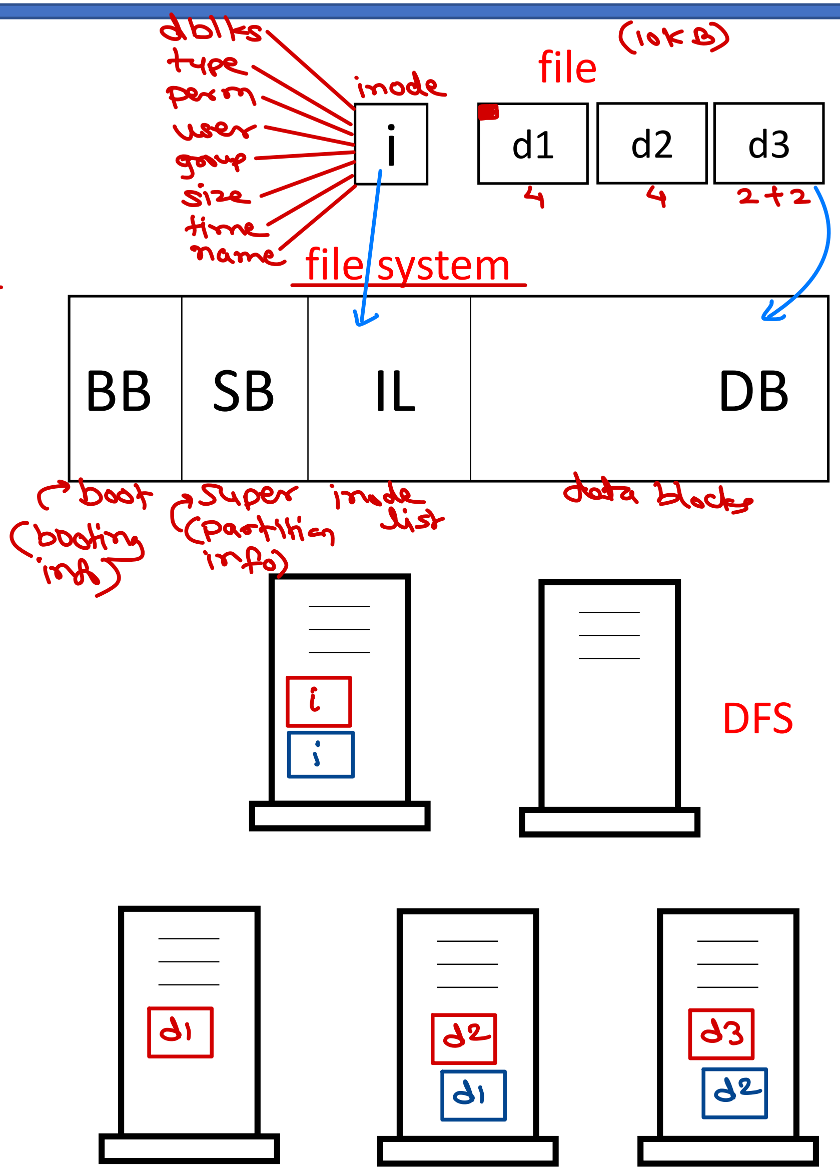
- Most of big data framework are distributed systems.
- Distributed system contains set of computers connected in a network (e.g. LAN). It is also referred as cluster. Each computer in cluster is referred as a node.
- Distributed systems provides
  - High availability, Fault tolerance, Rich computing, High memory.
  - High scalability (Horizontal scaling), Load balancing.
- There are two prime components of distributed system
  - Distributed storage
  - Distributed computing
- Major challenges for distributed systems
  - Node failure
  - Network failure
  - Distributed synchronization



Cluster

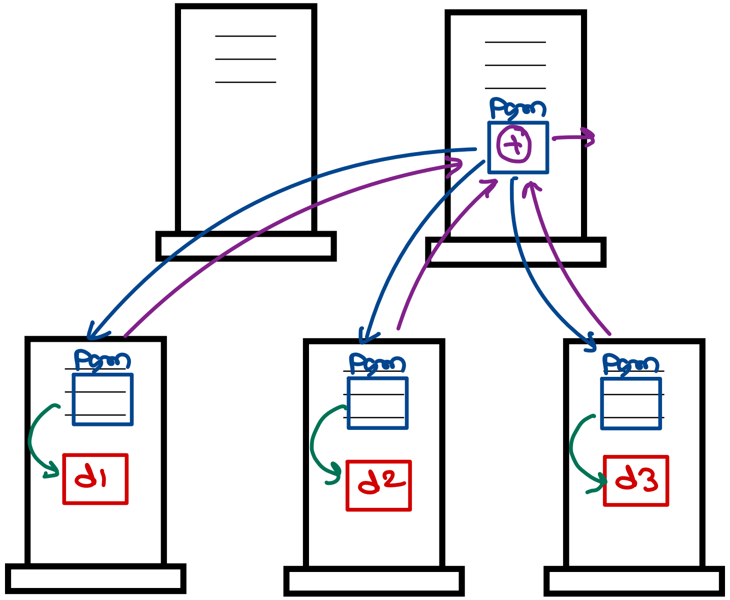
# Distributed Storage

- Each file have data (contents) and metadata (info).
- Information about data blocks is stored into metadata. To read the file first metadata is accessed and then data blocks. To write, data blocks are updated and metadata as well.
- Files are organized into file systems. File systems arrange file's data blocks and inodes in systematic manner for efficient storage and access.
- In distributed file system, data blocks and metadata can be scattered on multiple nodes in the cluster.
- This improves the processing speed of the data.
- However what if any node is failed (containing data) or metadata node is failed? DFS gracefully handle these concerns using replication and/or backup node features.



# Distributed Computing

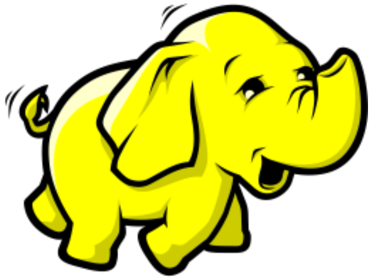
- Traditionally program loads data to be processed from the source and perform operations on it.
- This approach in not suitable for Big Data, considering data size and read/write speed of storage.
- Since data is stored on multiple nodes (distributed storage), program is also executed on multiple nodes processing partial data. These partial results are collected on a node and processed to yield final result.
- Distributed computing follows map-reduce design pattern.
  - Map stage process each record individually.
  - Reduce stage performs aggregation operation.
- Where does individual nodes process the data in memory or on disk? What if any node fails?

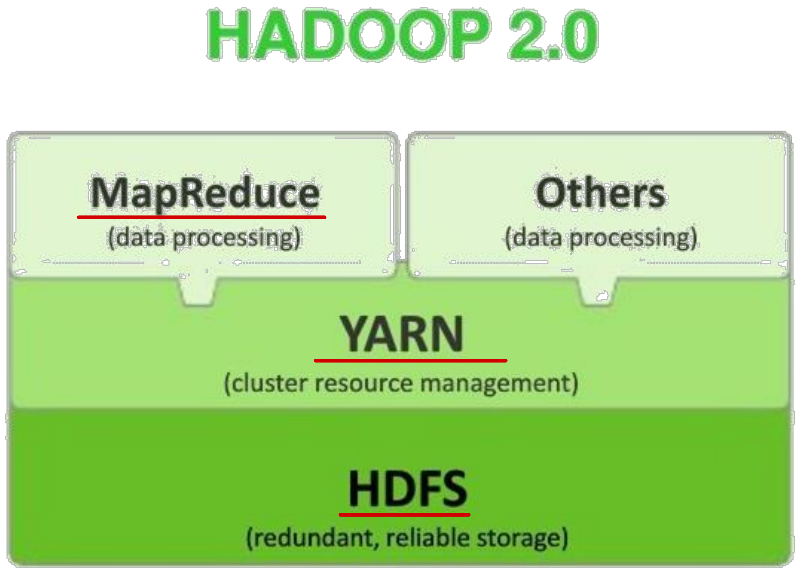


Distributed computing

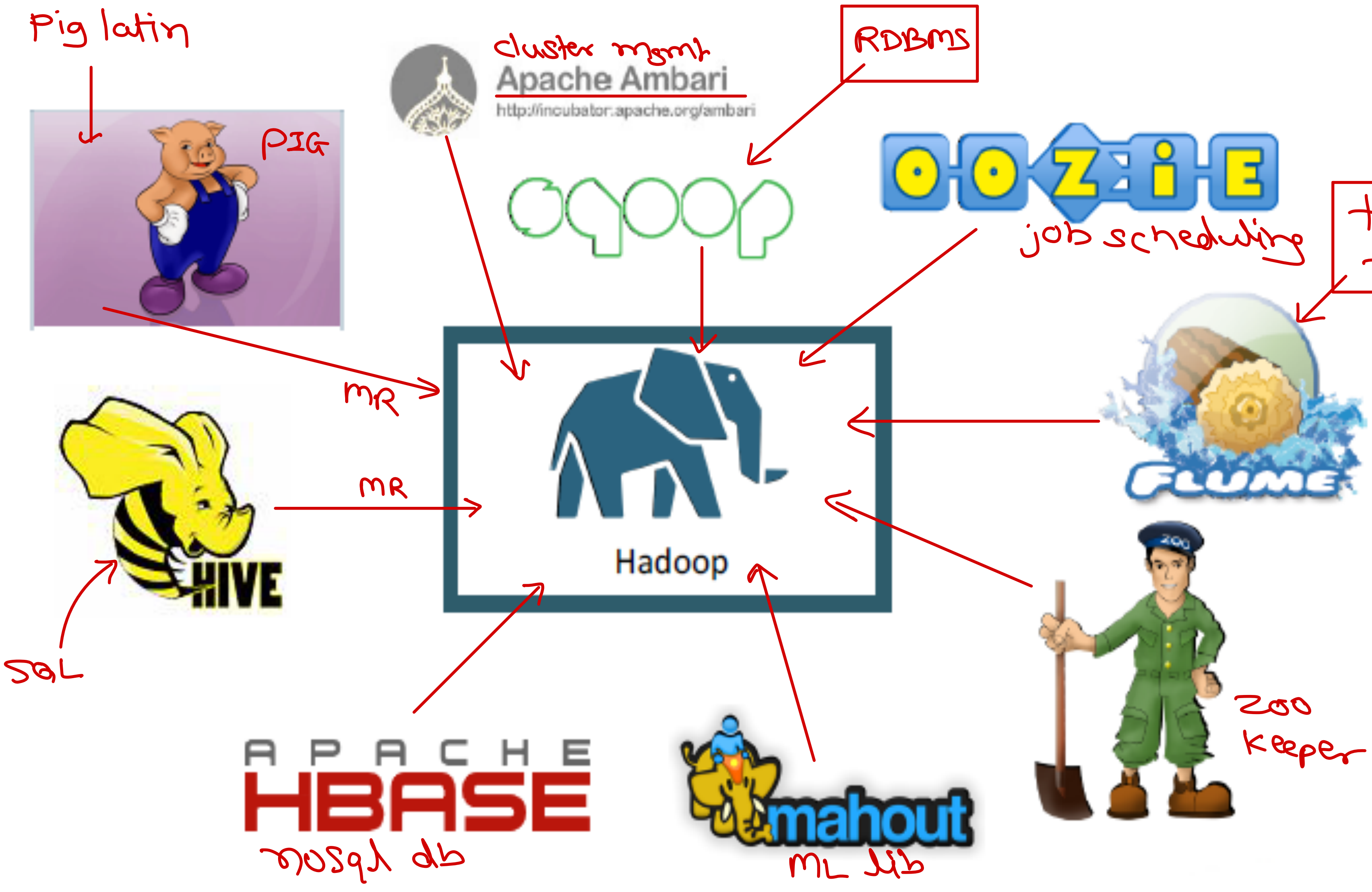
# Apache Hadoop

- Hadoop is developed by Doug cutting.
  - Web crawler – Nutch
  - Distributed computing and storage needed to process huge data produced by the crawler.
  - Joined Yahoo. Developed and open sourced under Apache license.
- Hadoop 1.x 2006
  - Distributed storage: HDFS
  - Distributed computing Map-reduce
- Hadoop 2.x
  - Distributed storage: HDFS
  - Distributed computing Map-reduce
  - Cluster manager: YARN
- Hadoop is like a Kernel/Platform on which many different applications are built (eco-systems).





# Hadoop Eco-System & Hadoop distributions



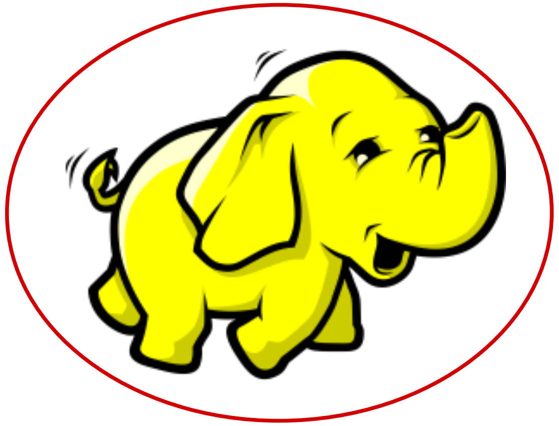
- Eco-Systems
  - HBase
  - Sqoop
  - Flume
  - Pig
  - Hive
  - Impala
  - Hue
  - Oozie
  - ZooKeeper
  - Spark
- Hadoop Distributions
  - Cloudera ✓
  - Hotonworks ✓
  - AWS EMR ✓
  - MapR ✓

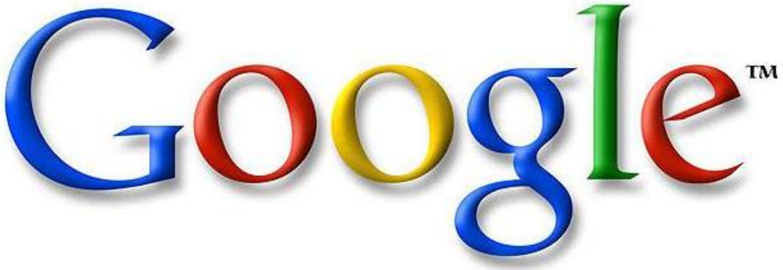
# Big Data – Hadoop

Trainer: Mr. Nilesh Ghule.

# Apache Hadoop History

- Hadoop is developed by Doug cutting & Mike Cafarella.
  - Core code of distributed storage and distributed computing in Hadoop is borrowed from Nutch project.
  - Nutch project is web crawler developed by Doug & Mike.
  - Distributed computing and storage needed to process huge data produced by the crawler.
  - Doug Cutting joined Yahoo.
  - Hadoop 0.1.0 is released in April 2006.
  - Hadoop open sourced under Apache license.
- Development of Hadoop is inspired from Google white-papers on GFS (2003) & MapReduce (2004).
- Hadoop is implemented in Java.
- Hadoop is named after Doug Cutting's son's toy elephant.
- Hadoop has major components HDFS & MapReduce.



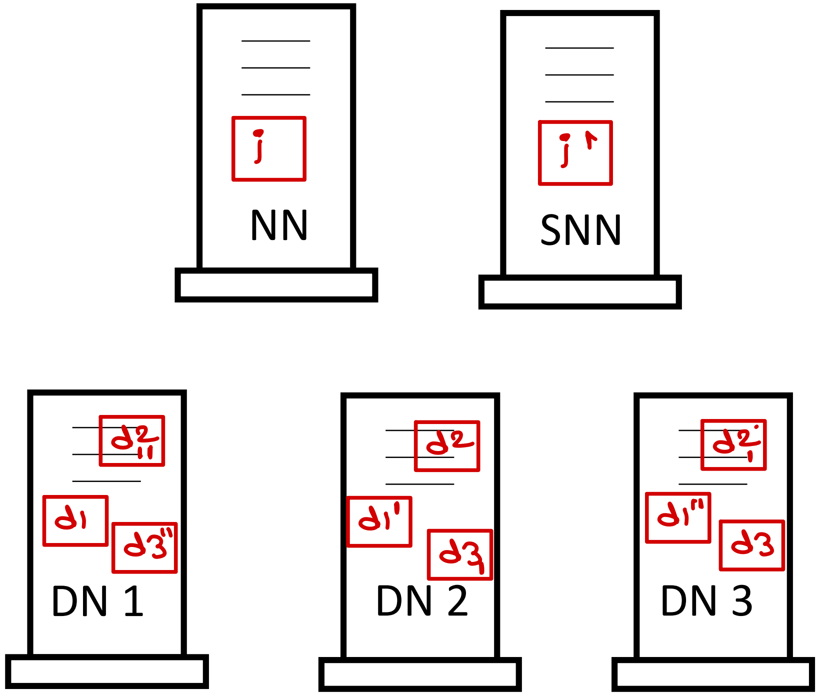


# Hadoop Distributed File System

- HDFS is fault tolerant, redundant distributed file system.
- It is implemented following white-paper on Google File System.
- HDFS has three components
  - Name Node – Manage file metadata.
  - Data Node – Manage files data.
  - Secondary Name Node – Metadata backup.
- HDFS stores file's data into data blocks. Size of data block is 64 MB or 128 MB.

1·x 2·x/3·x

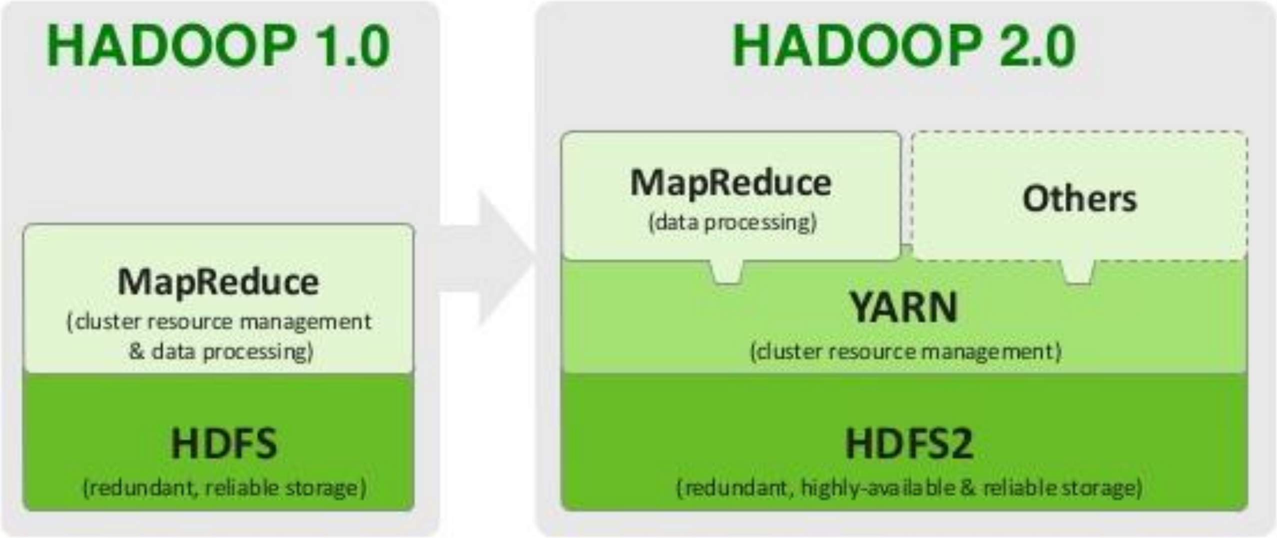
- Each HDFS block is replicated on 3 nodes (while write operation). It ensures that if any node fails, data can be taken from some replica node.

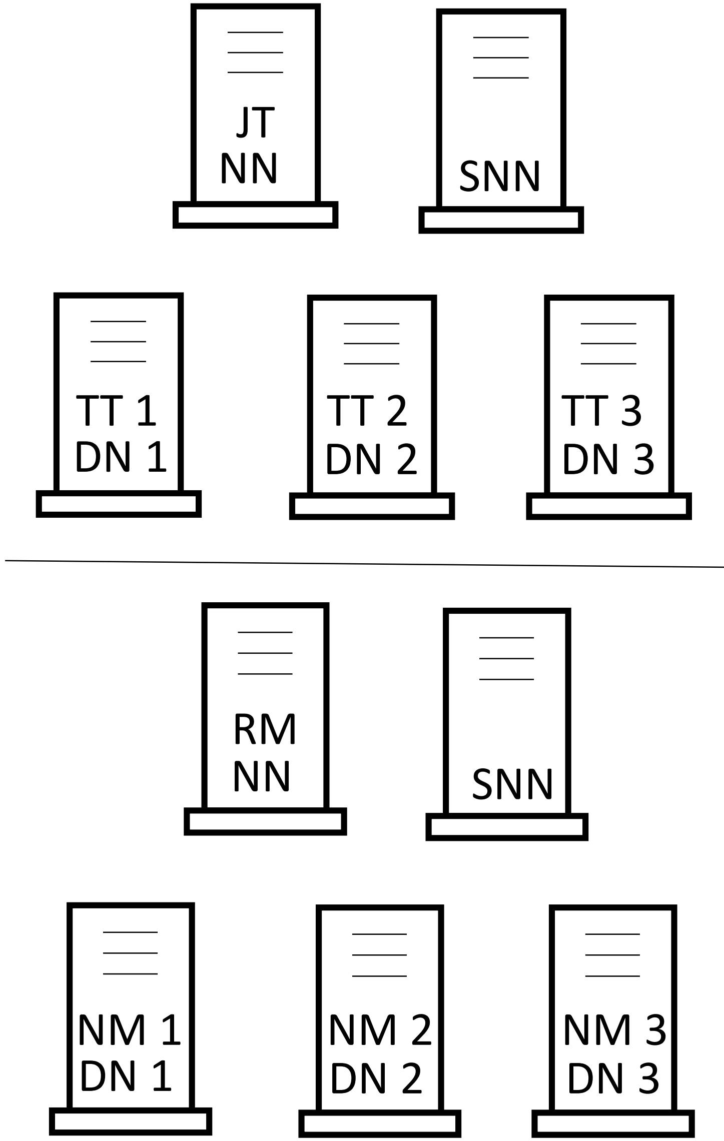


- The metadata backup is maintained on secondary name node. In case of name node failure, metadata can be retrieved from secondary name node.
- This makes HDFS fault-tolerant.

# Hadoop 1.x vs Hadoop 2.x

- HDFS of 2.x can be configured to be highly available. (HA)
  - Secondary NameNode process WAL to ensure that no data is lost.
  - Standby NameNode actively takes backup and can become active namenode immediately in case of failure of NameNode.
- MapReduce of 2.x introduce YARN scheduler.
  - YARN does uniform cluster/resource management.
  - Individual job progress is tracked by MRAppMaster.


HADOOP 1.0
MapReduce
(cluster resource management & data processing)
HDFS
(redundant, reliable storage)
HADOOP 2.0
MapReduce
(data processing)
Others
YARN
(cluster resource management)
HDFS2
(redundant, highly-available & reliable storage)



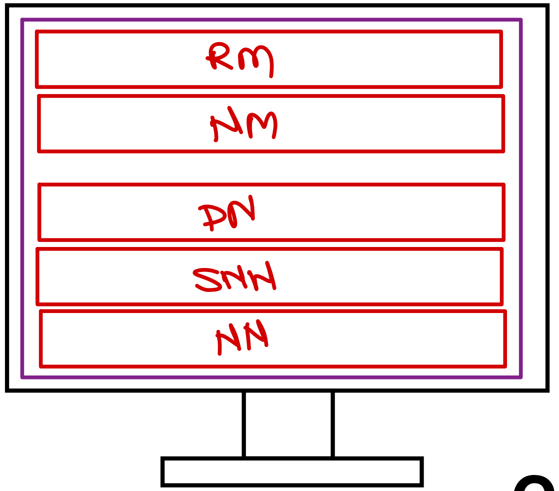
# Hadoop 2.x Daemons & Hadoop installation modes

- Hadoop daemons are background processes implemented in Java.
- HDFS Daemons
  - DataNode ✓
  - NameNode ✓
  - SeconadaryNameNode ✓
- YARN/MapReduce Daemons
  - NodeManager ✓
  - ResourceManager ✓
- All daemons are configurable via XML configuration files.

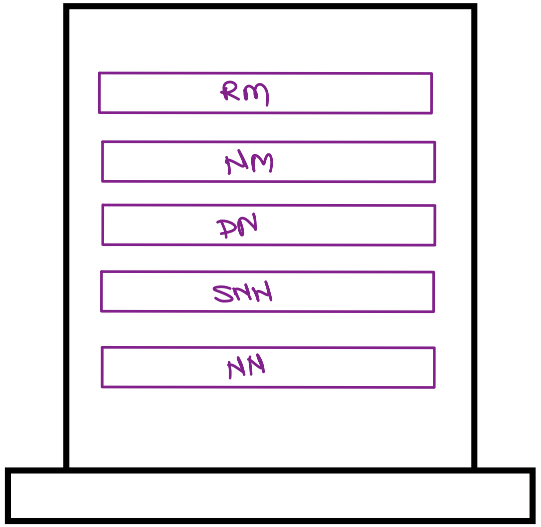
- Hadoop can be installed in 3 possible ways. It mainly differs in its applications and execution of Hadoop daemons.
- Local mode
  - All daemons runs in single Java process.
  - Can access only LocalFileSystem. (/)
  - Used for unit testing of MR jobs & prototyping.
- Pseudo distribution mode (Single node cluster)
  - All daemons runs as independent Java processes on the single machine.
  - Used as developer machine setup.
- Full distribution mode (Multi node cluster)
  - All daemons runs as independent Java processes on the multiple machines in the network.
  - This is production cluster setup to run jobs.

# Hadoop installation modes & Configuration files

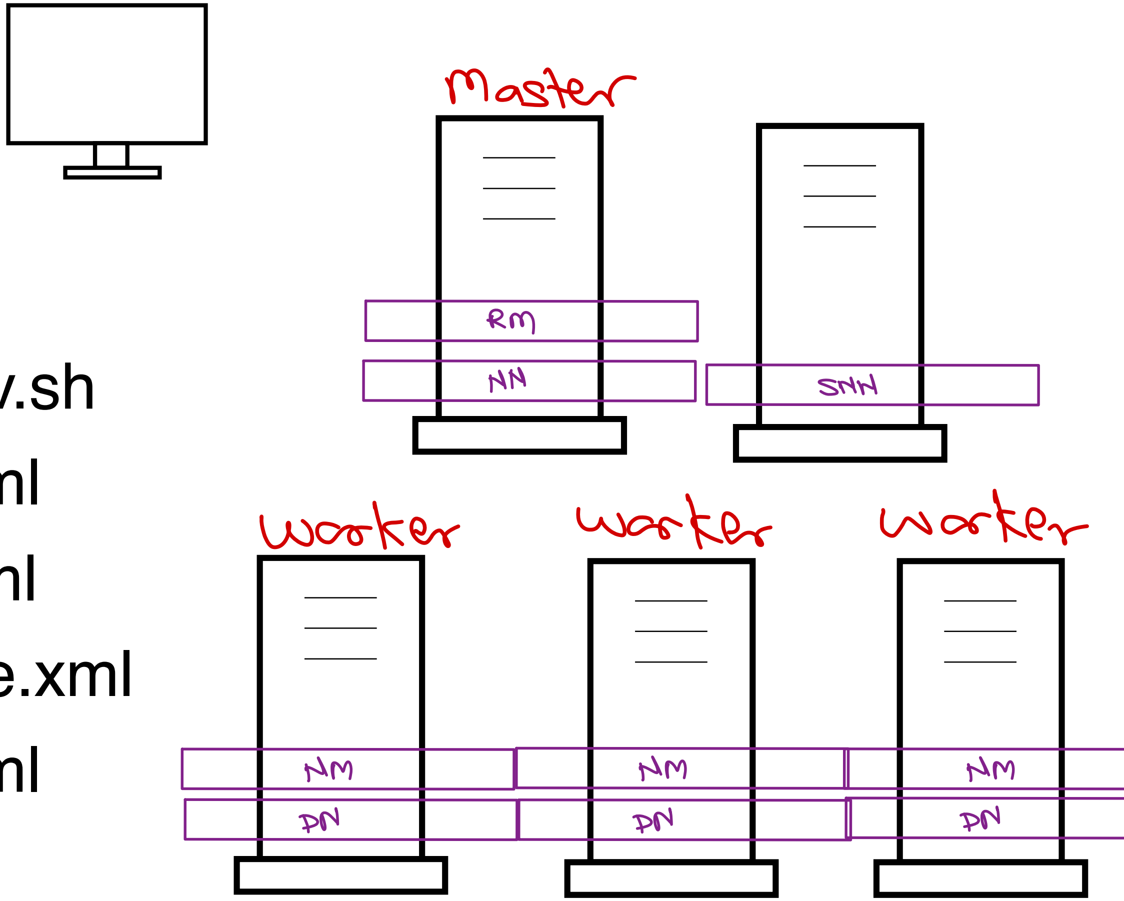
- Local mode



- Pseudo distribution mode



- Full distribution mode
  [https://github.com/nilesh-g/hadoop-cluster-install](https://github.com/nilesh-g/hadoop-cluster-install)



**Config files**

- hadoop-env.sh
- core-site.xml
- hdfs-site.xml
- mapred-site.xml
- yarn-site.xml
- ~/.bashrc

# Using HDFS

- Before using HDFS need to be formatted. It create first (empty) file system image on NameNode.
  - terminal> hdfs namenode –format
- Start all HDFS daemons & verify them
  - terminal> start-dfs.sh
  - terminal> jps
  - browser: http://localhost:50070
- While metadata is loaded into NameNode memory, HDFS is not ready for use. This state is safe mode.

- HDFS user commands
  - terminal> hadoop fs –help
  - syntax: hadoop fs genericoptions command
- Generic options
  - -conf, -fs, …
- HDFS user commands categories
  - ingestion/retrieval: put, get, getmerge
  - directory handling: ls, mkdir, rmdir
  - file data handling: cat, tail, rm, truncate, touchz, stat
  - metadata handling: chmod, chown, setrep
- HDFS admin commands
  - terminal> hdfs –help
  - terminal> hdfs dfsadmin –help

# Thank you!

*Nilesh Ghule <nilesh@sunbeaminfo.com>*