LETS LEARN HADOOP

(The easiest way)

Training by- Sudhanshu Saxena(er_sudhanshusaxena@yahoo.com)

Type of Data Processing

- HDFS Architecture
- Introduction to Hadoop Distributed File System
- Hadoop Setup

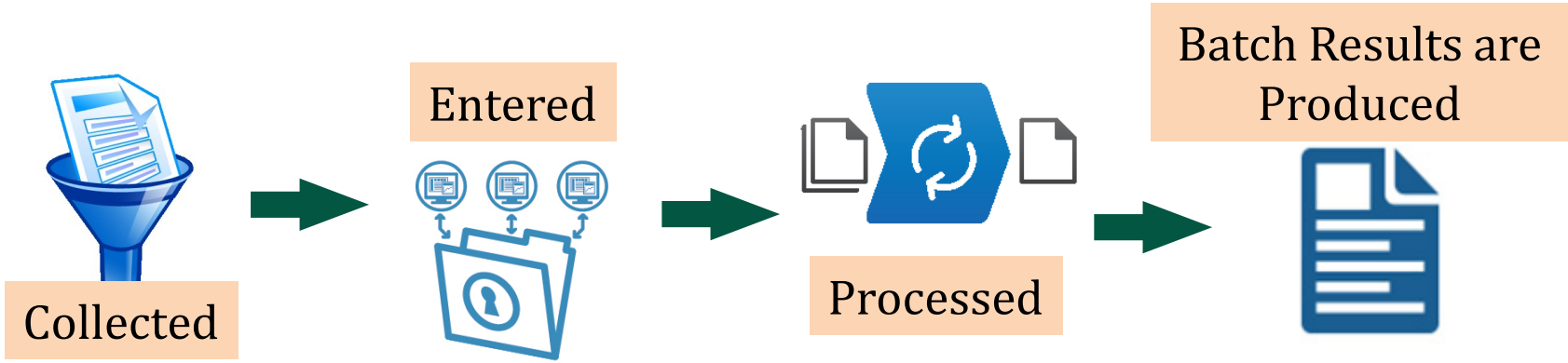
Lets understand Data Processing types

- Batch Processing
- Real time Processing

Batch Processing

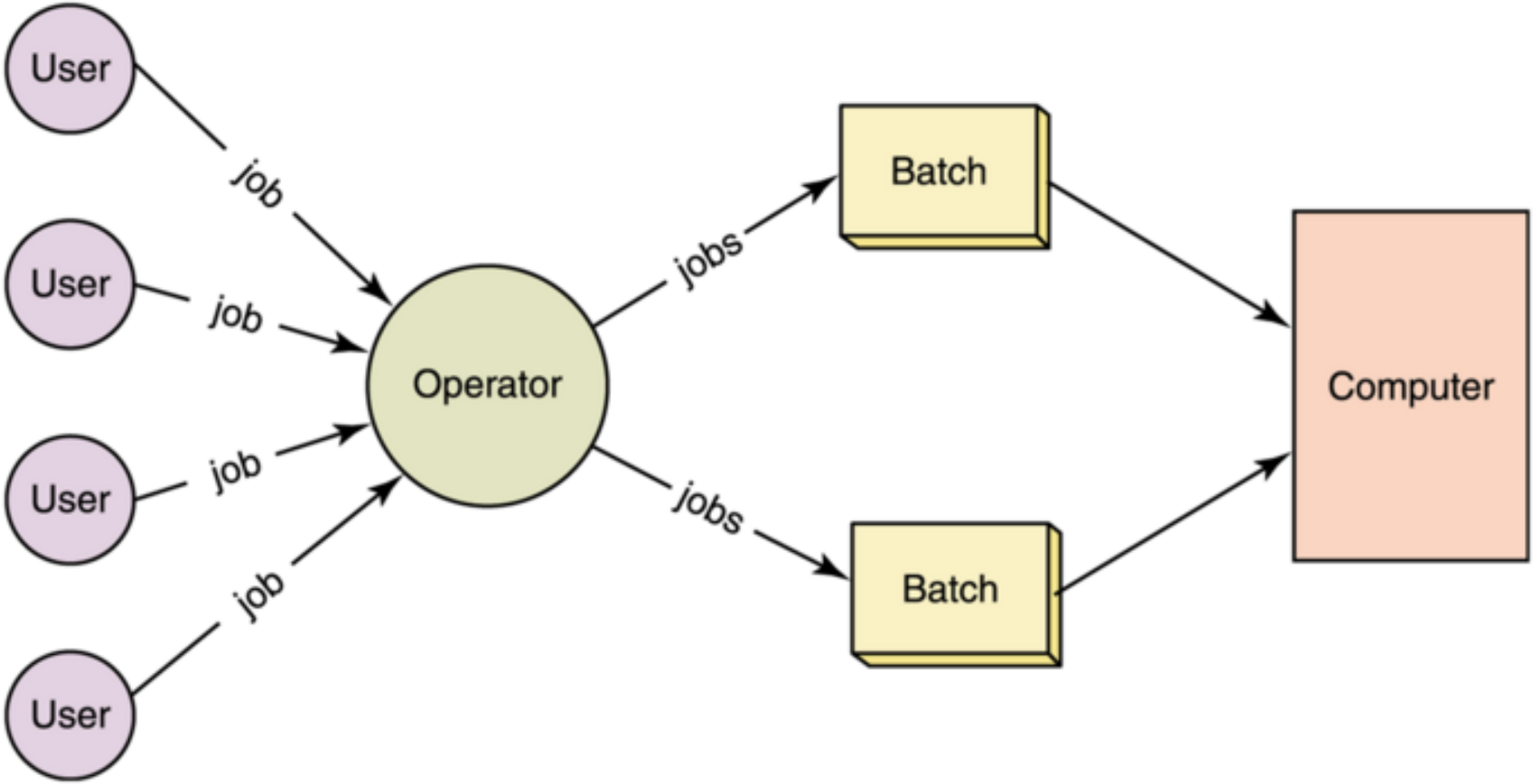
What is Batch Processing?

- Batch data processing is an efficient way of processing high volumes of data is where a group of transactions is collected over a period of time.
- Data is:



- Batch processing requires separate programs for input, process and output.
- An example is payroll and billing systems.

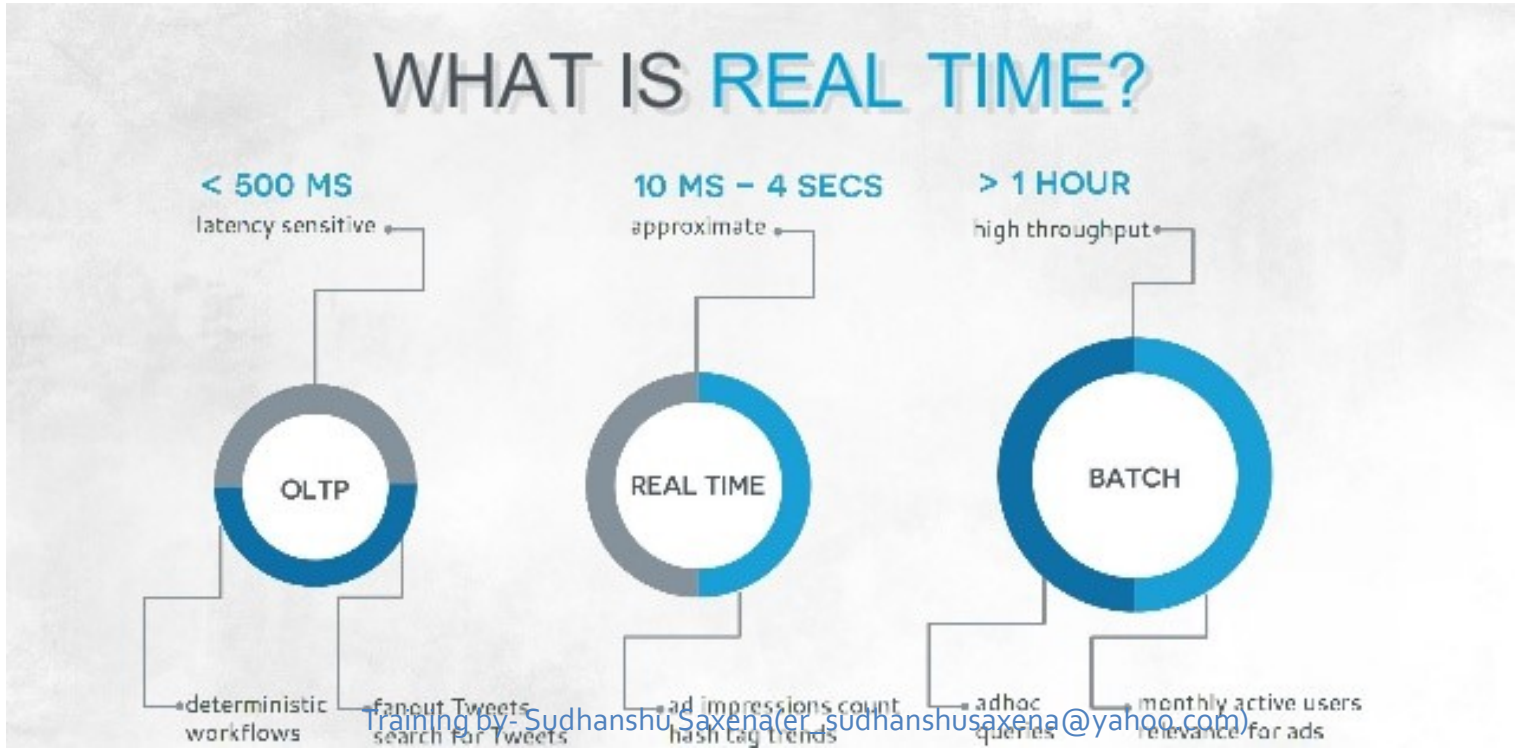
Batch Processing



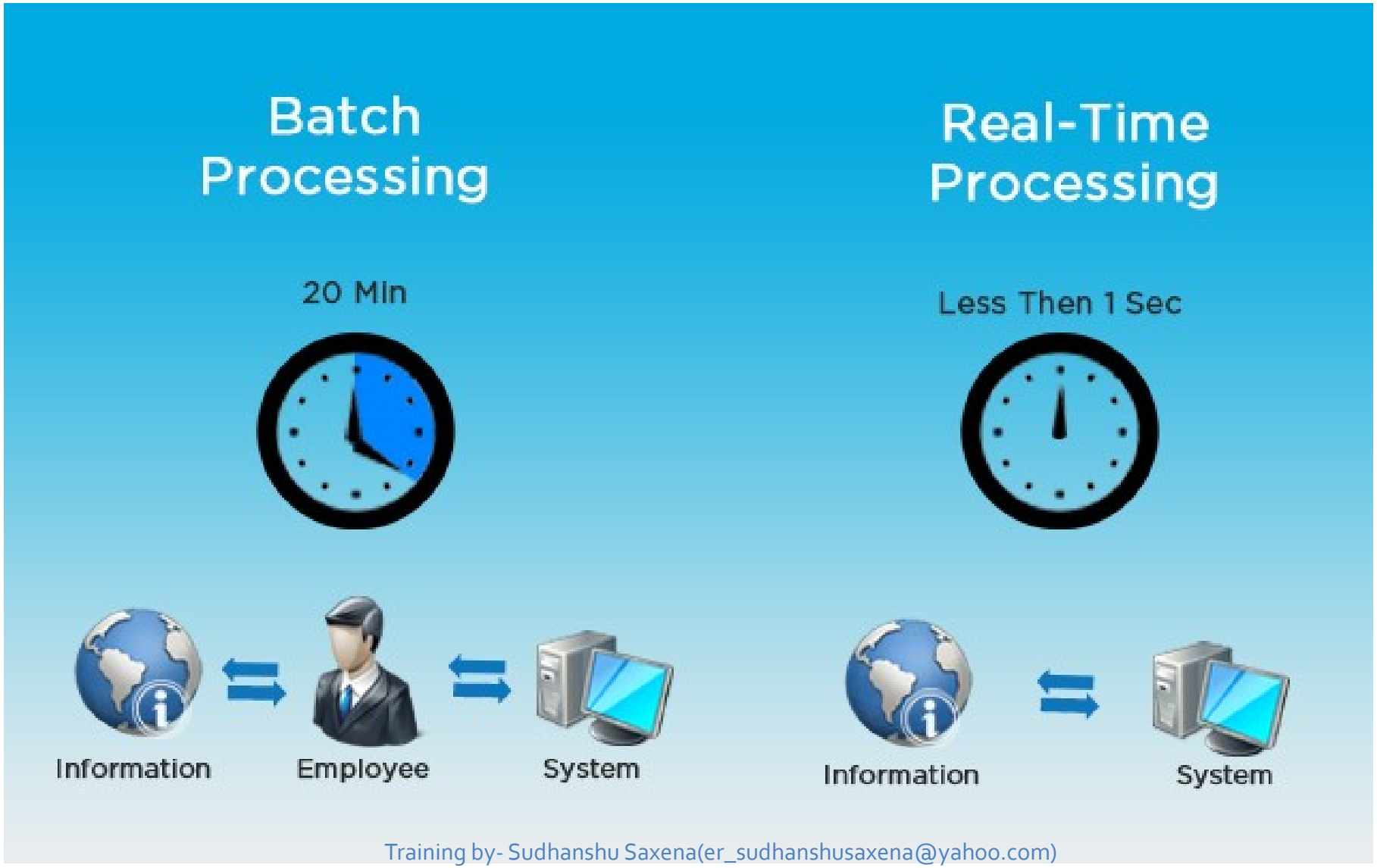
Real Time Processing

What is Real Time Processing?

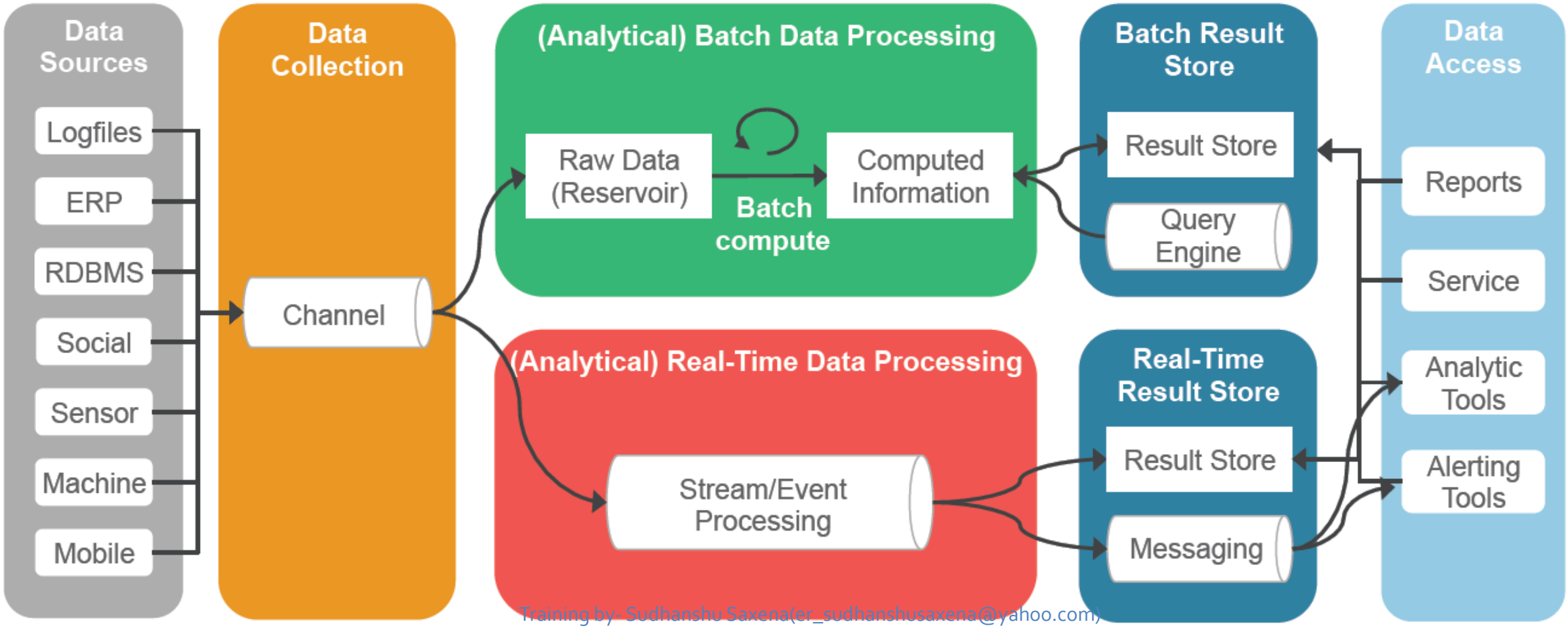
- In contrast, real time data processing involves a continual input, process and output of data.
- Data must be processed in a small time period (or near real time).



Batch Processing Vs Real time Processing



Architecture of Batch & Real Time Processing in Hadoop Ecosystem

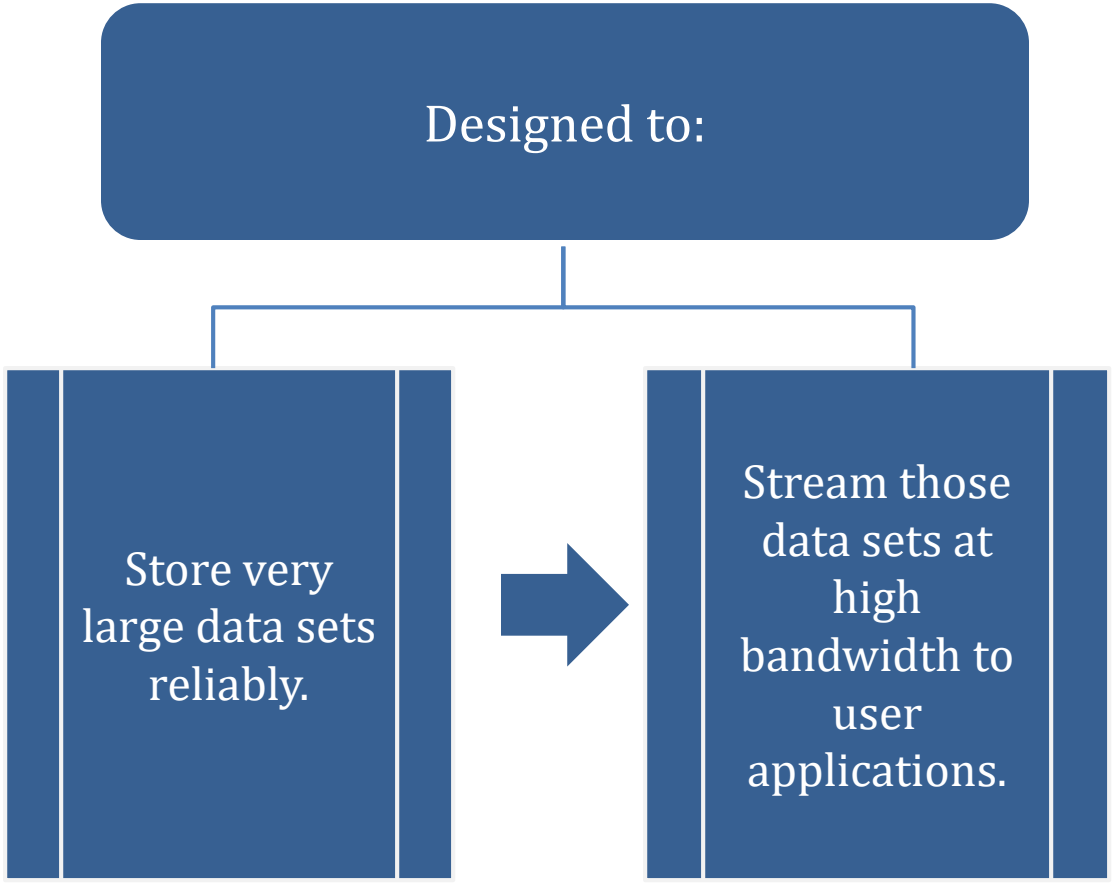


Agenda

In this session, you will learn about:

- Hadoop 1.0 & 2.0 Eco-system
- Hadoop Use Cases
- Where Hadoop Fits?
- Traditional vs. Hadoop Architecture
- RDBMS vs. Hadoop
- When to Use or Not Use Hadoop?
- Hadoop Opportunities

What is HDFS?



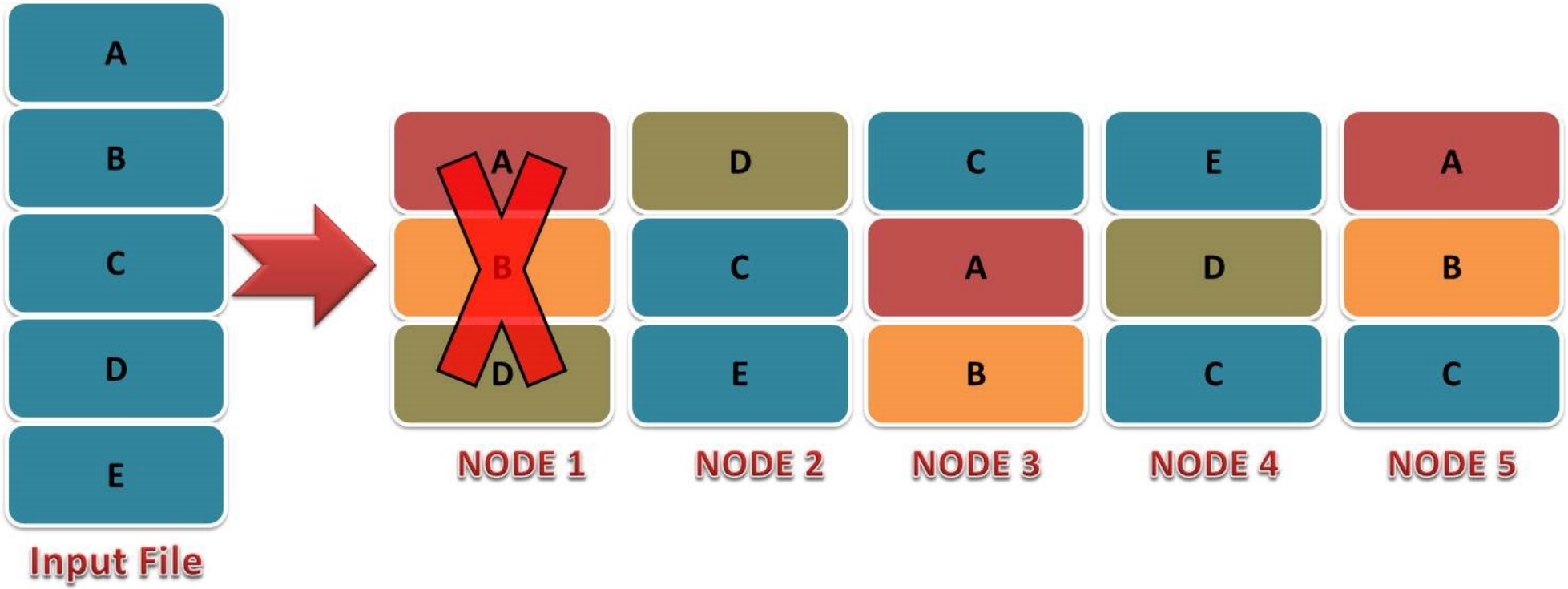
Hadoop's Distributed File System

Derived from GFS

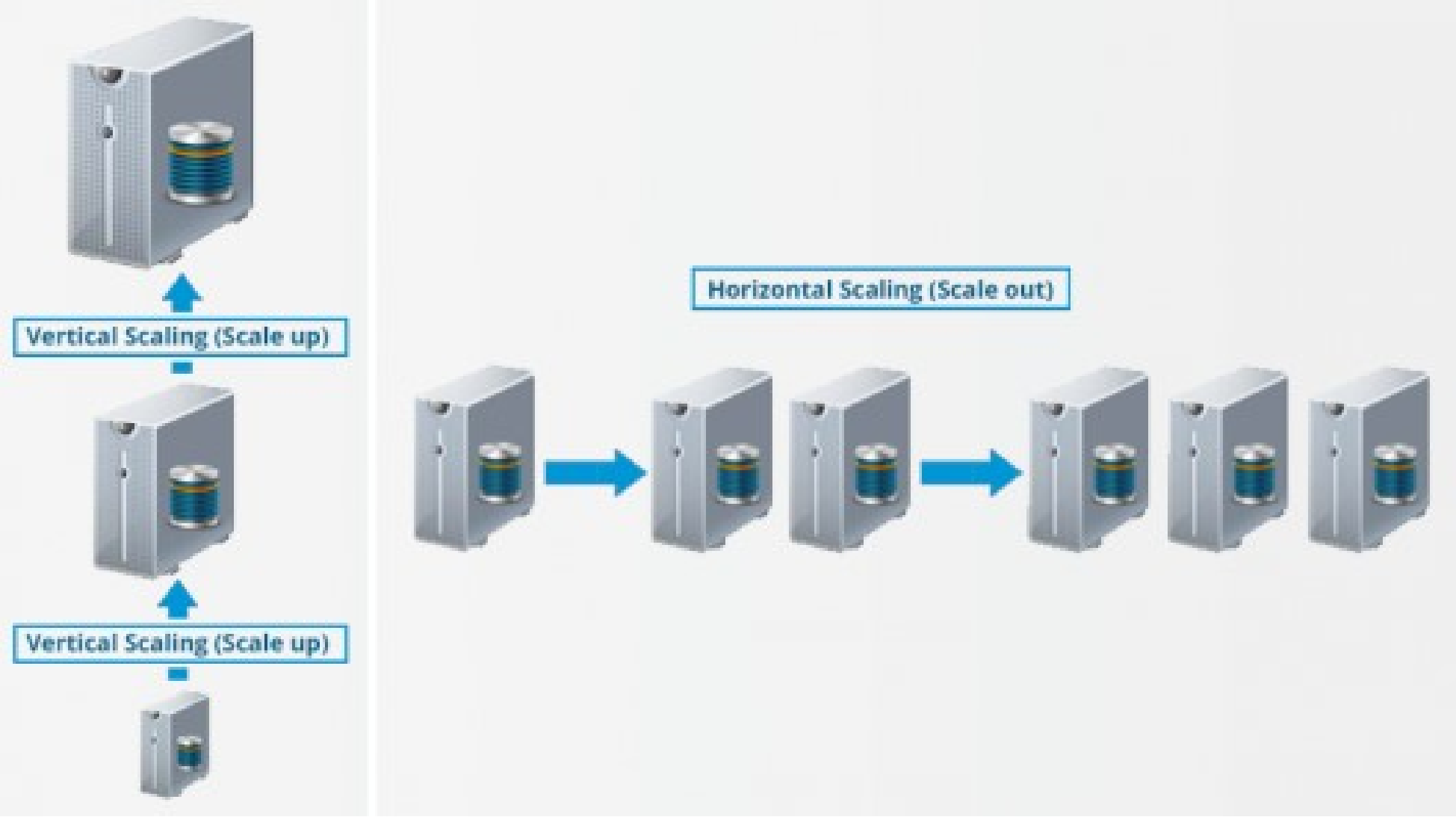
HDFS Features

- Fault Tolerance
- Horizontal Scaling
- Data Replication
- Rack Awareness

Fault Tolerance

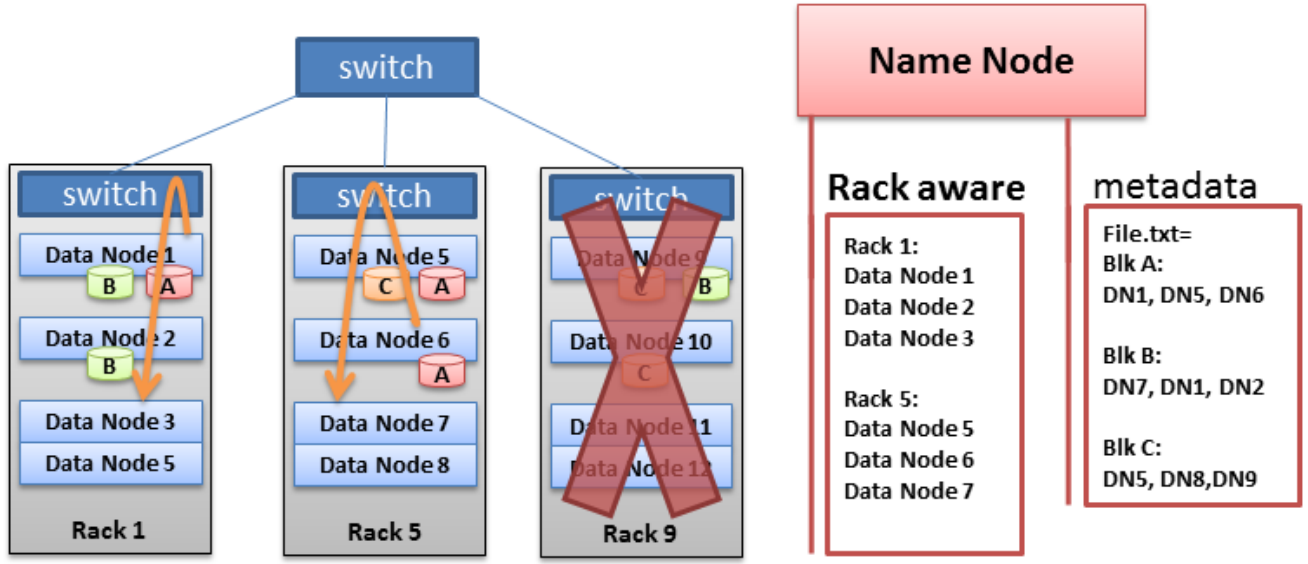


Horizontal Scaling



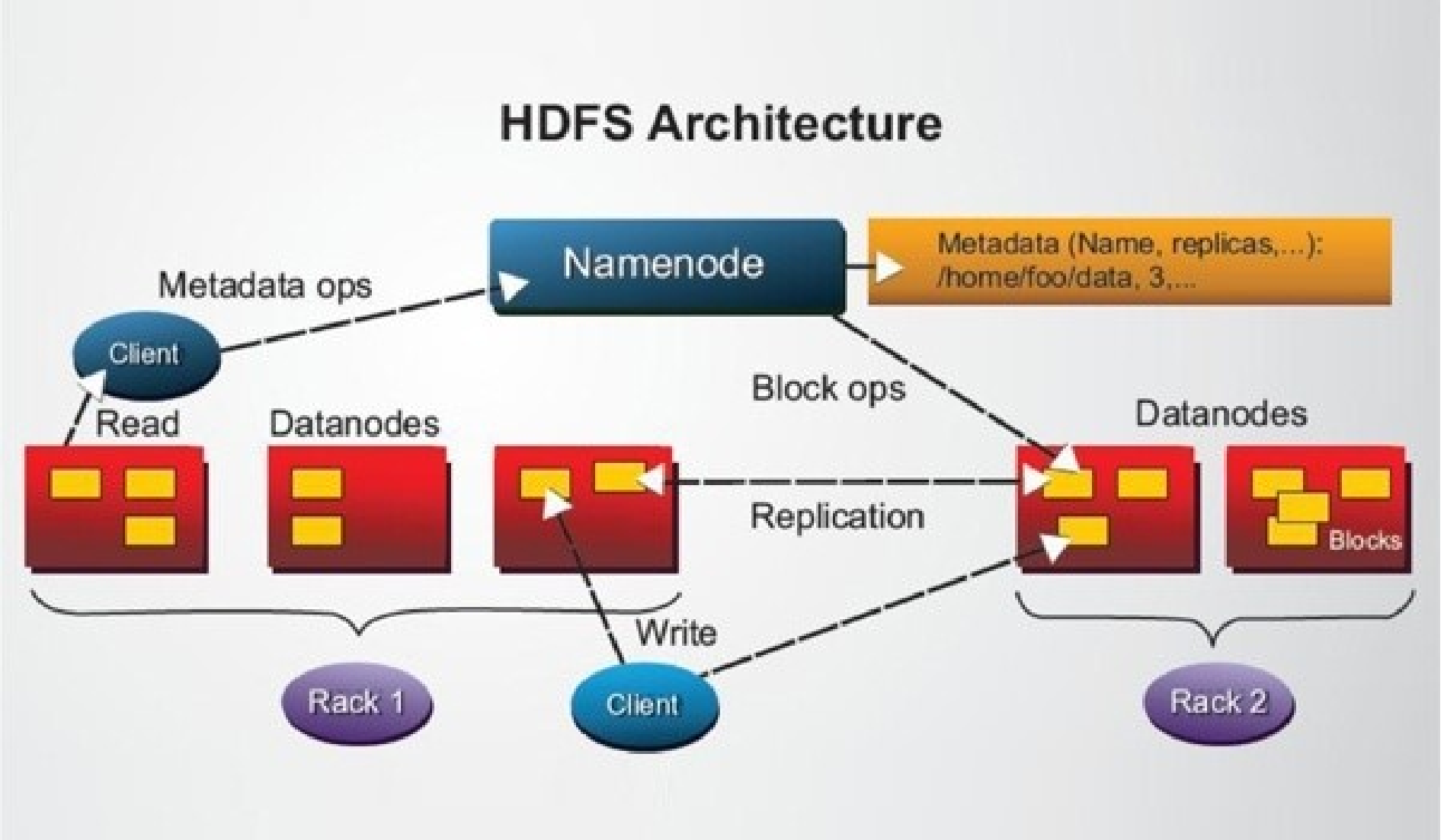
Rack Awareness

Hadoop Rack Awareness – Why?

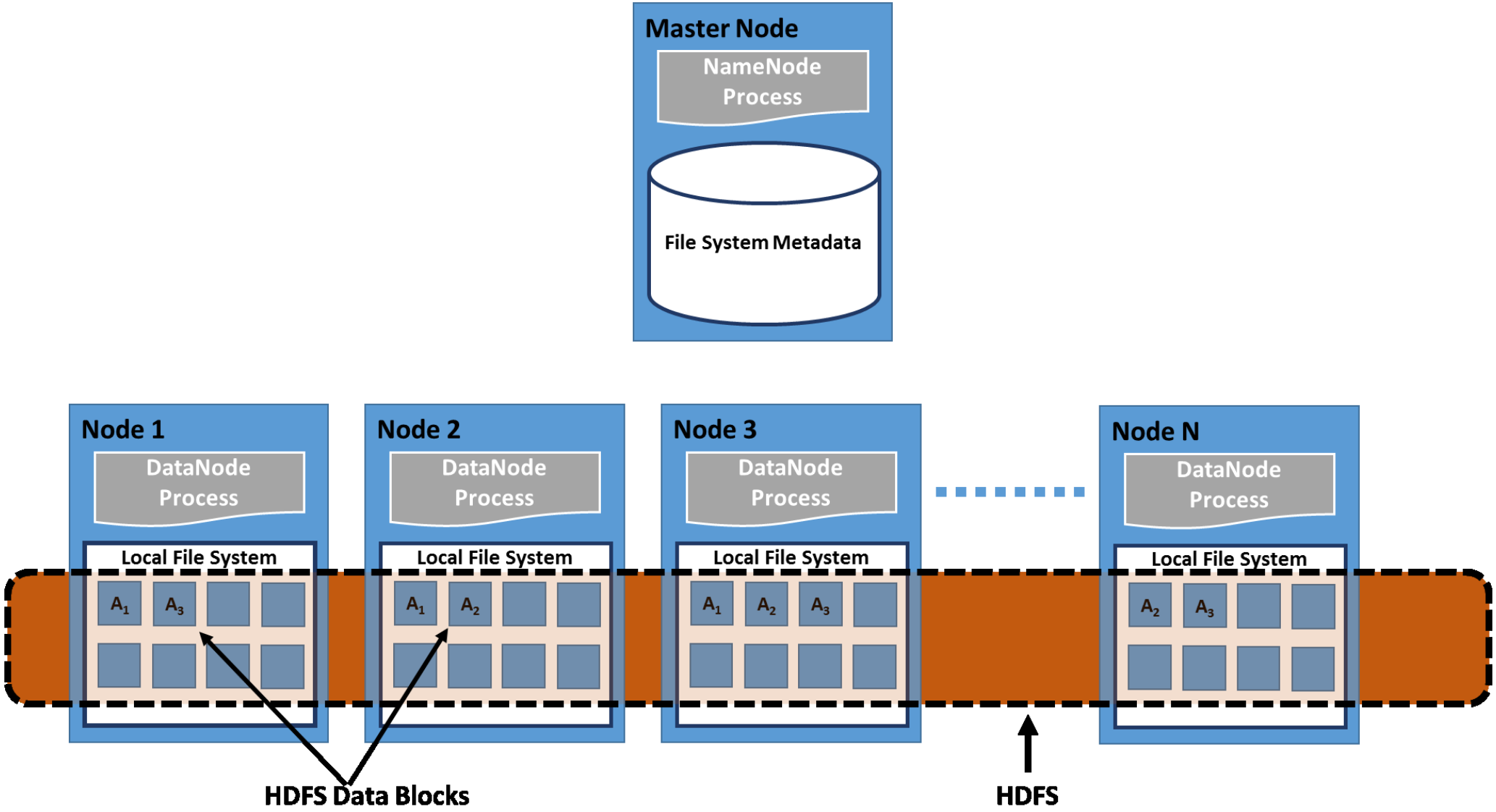


- Never loose all data if entire rack fails
- Keep bulky flows in-rack when possible
- Assumption that in-rack is higher bandwidth, lower latency

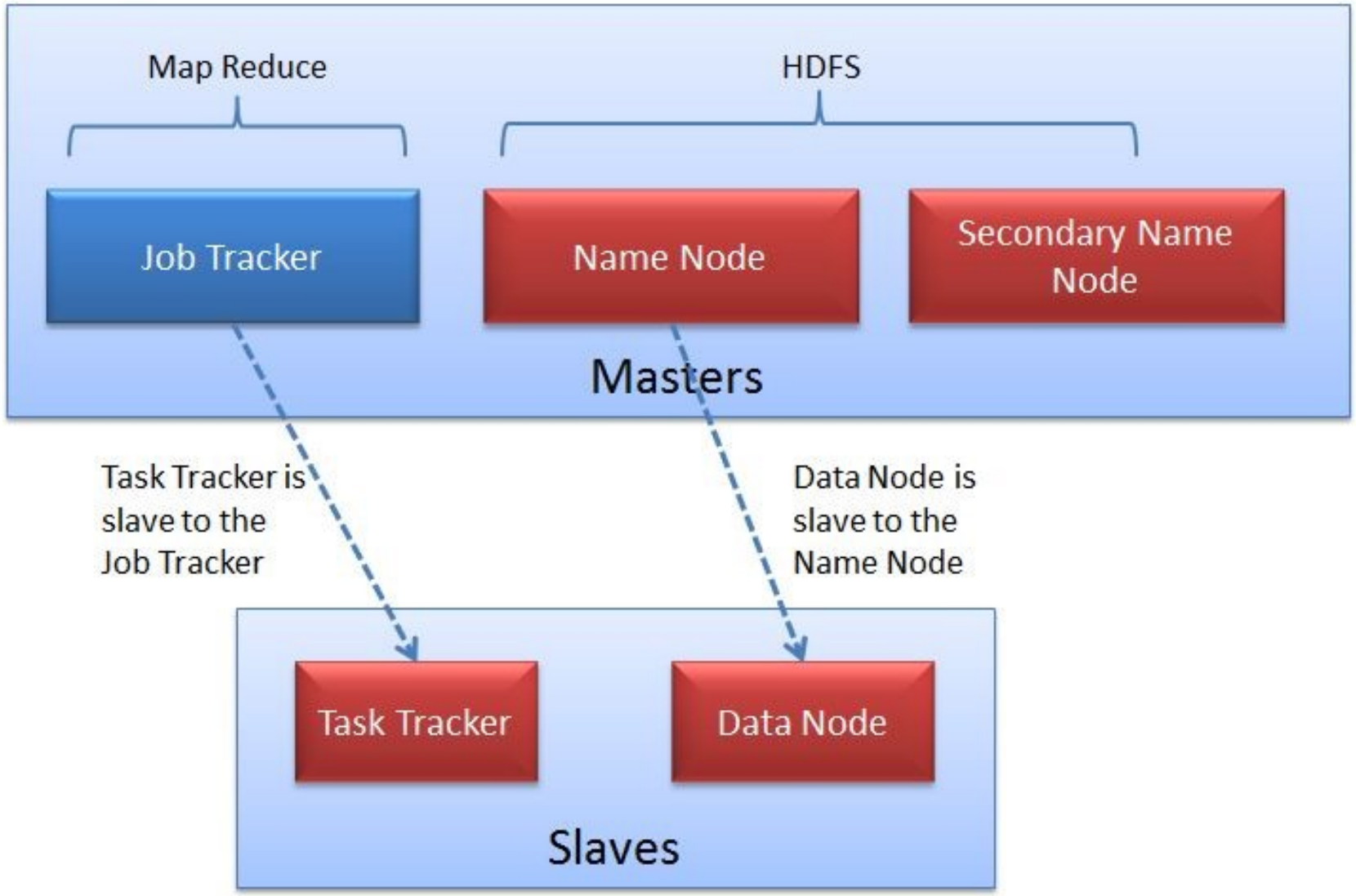
HDFS Architecture



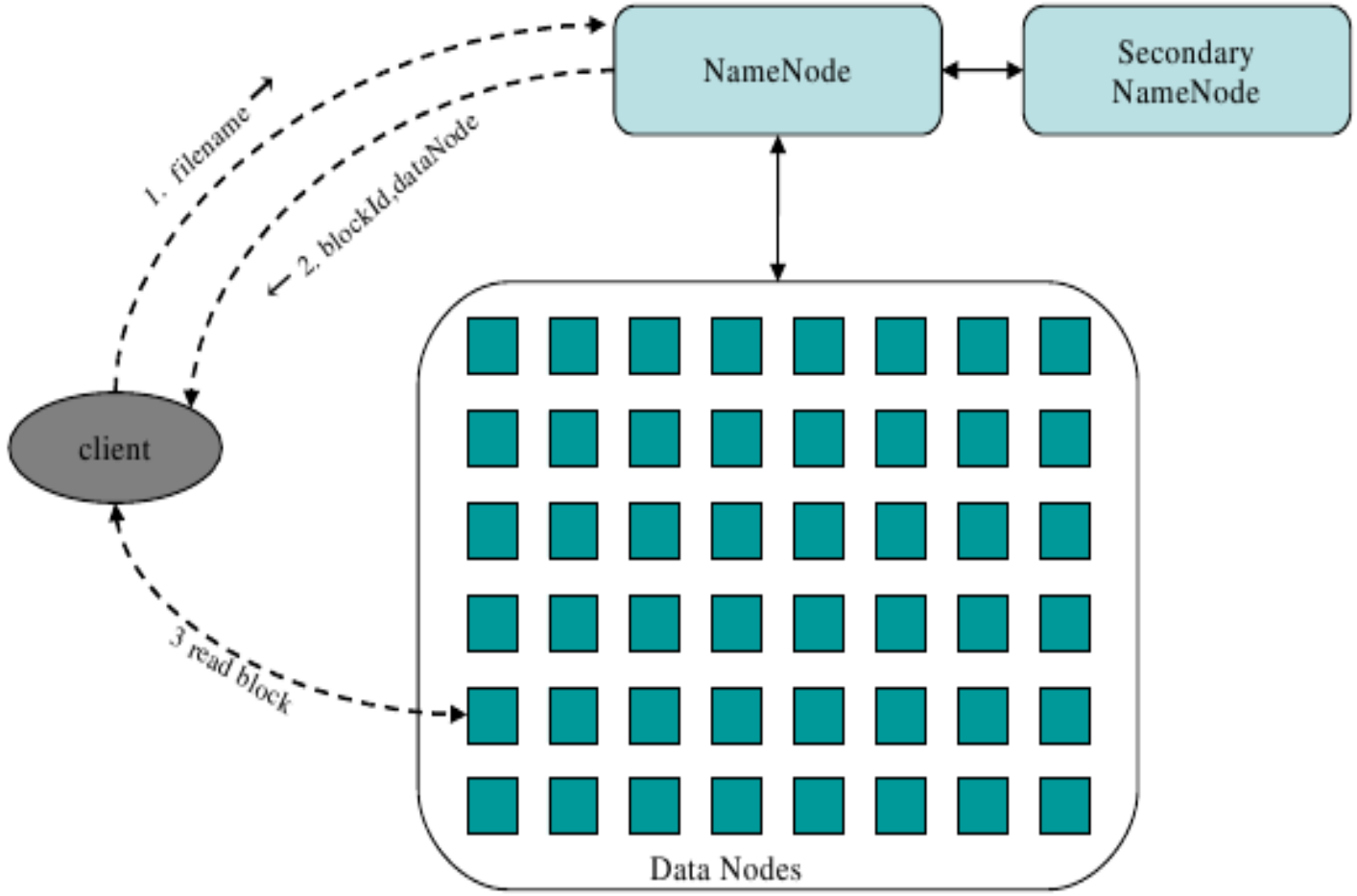
Introduction to Hadoop Distributed File System



Lets Zoom in


Map Reduce
HDFS
Job Tracker
Name Node
Secondary Name Node
Masters
Task Tracker is slave to the Job Tracker
Data Node is slave to the Name Node
Task Tracker
Data Node
Slaves

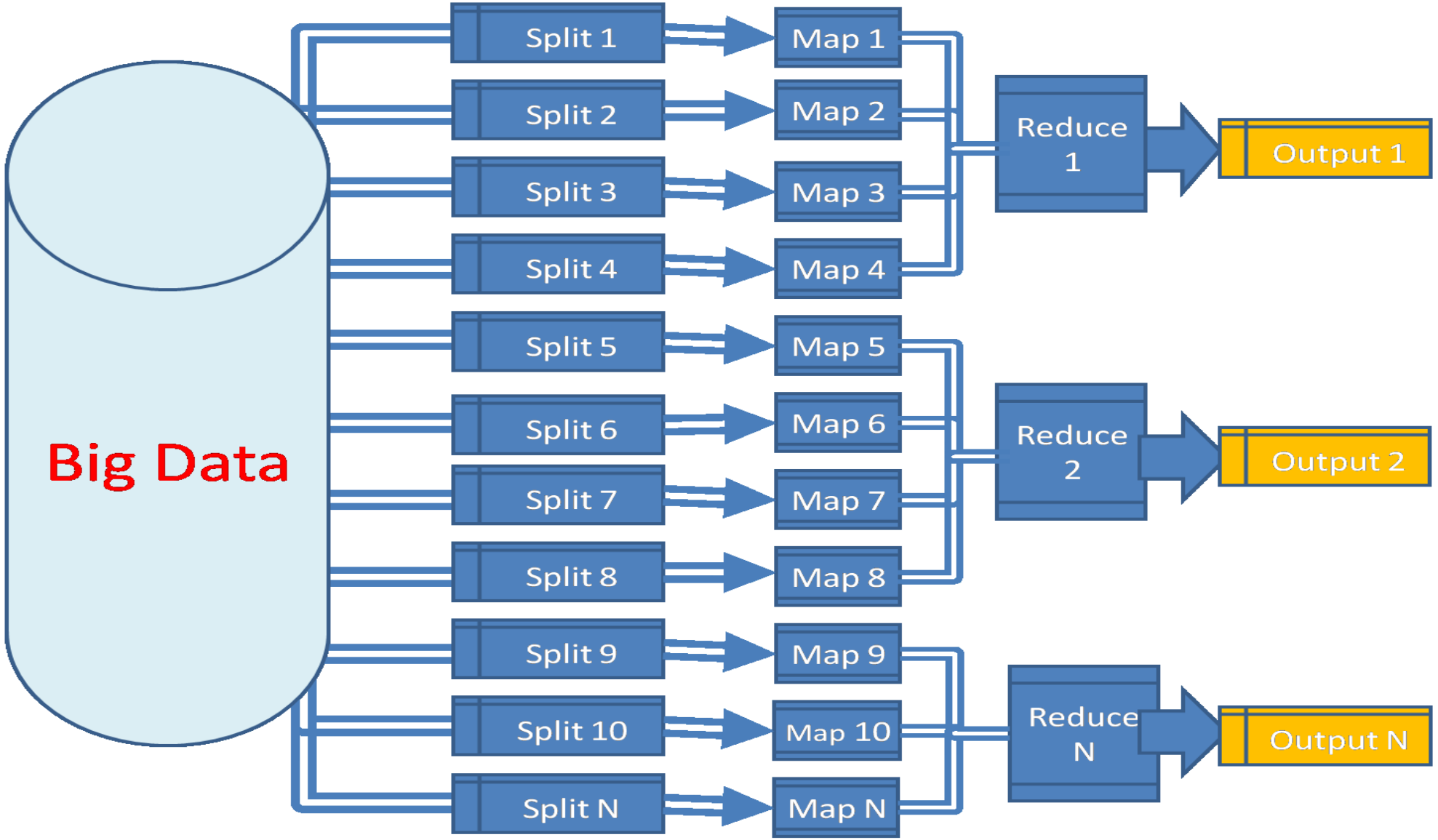
HDFS Architecture



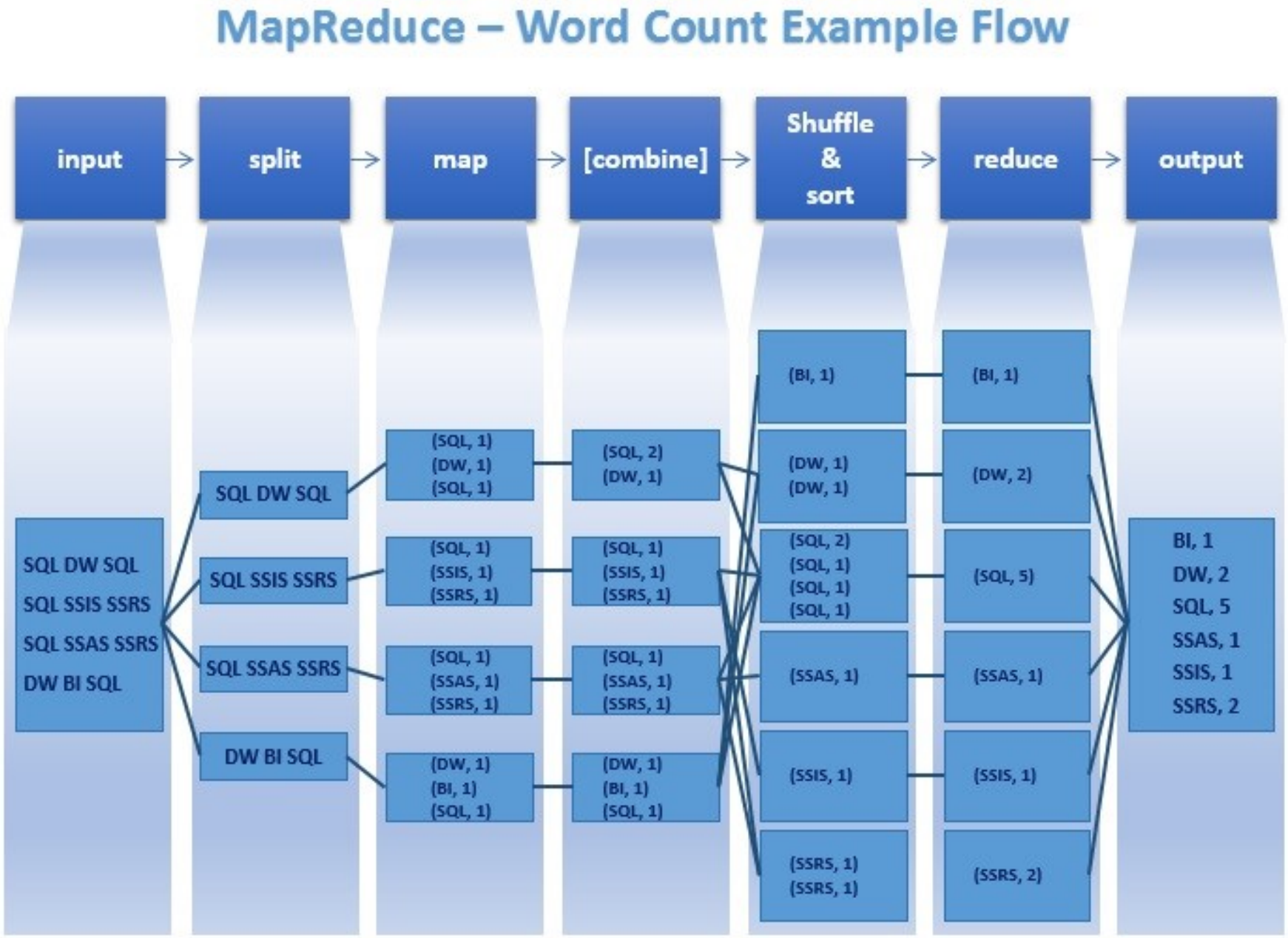
- **NameNode**: Filename, offset> blockid, block > datanode
- **DataNode**: maps block > local disk
- **Secondary NameNode**: Periodically merges edit logs

Block is also called chunk

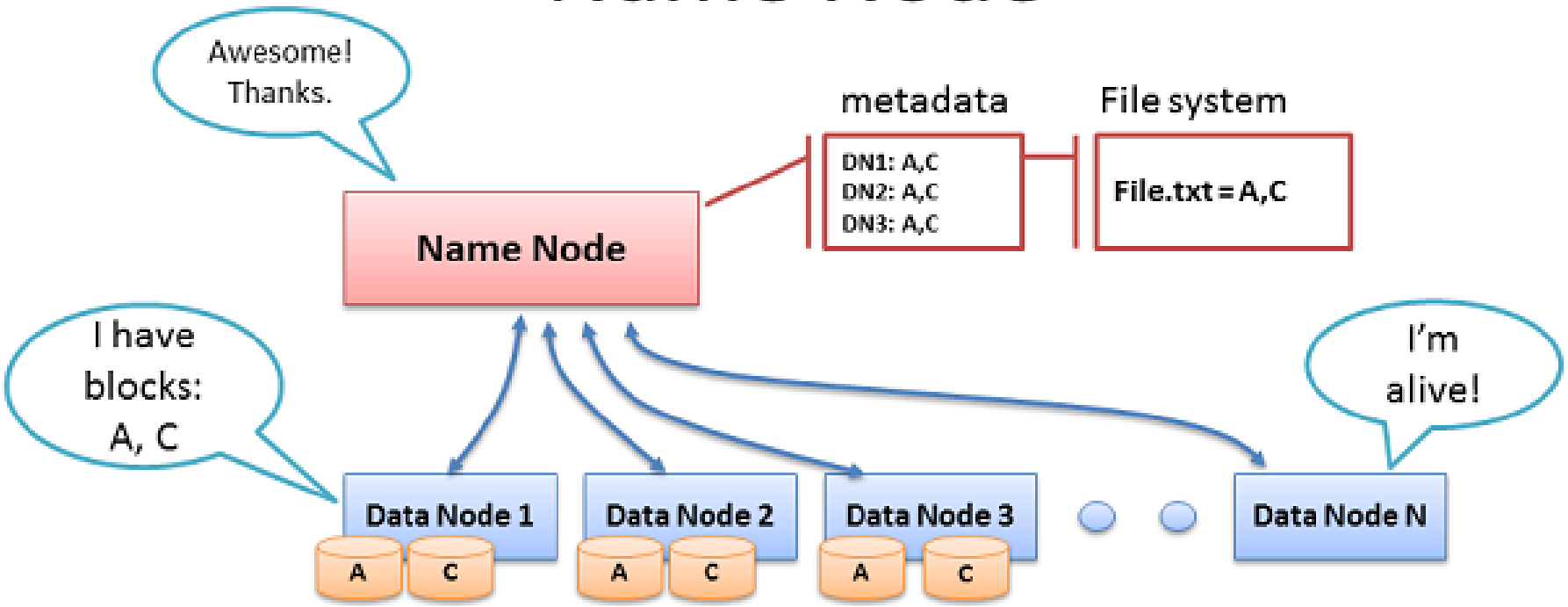
Map Reduce


Big Data
Split 1
Split 2
Split 3
Split 4
Split 5
Split 6
Split 7
Split 8
Split 9
Split 10
Split N
Map 1
Map 2
Map 3
Map 4
Map 5
Map 6
Map 7
Map 8
Map 9
Map 10
Map N
Reduce 1
Reduce 2
Reduce N
Output 1
Output 2
Output N

Map Reduce

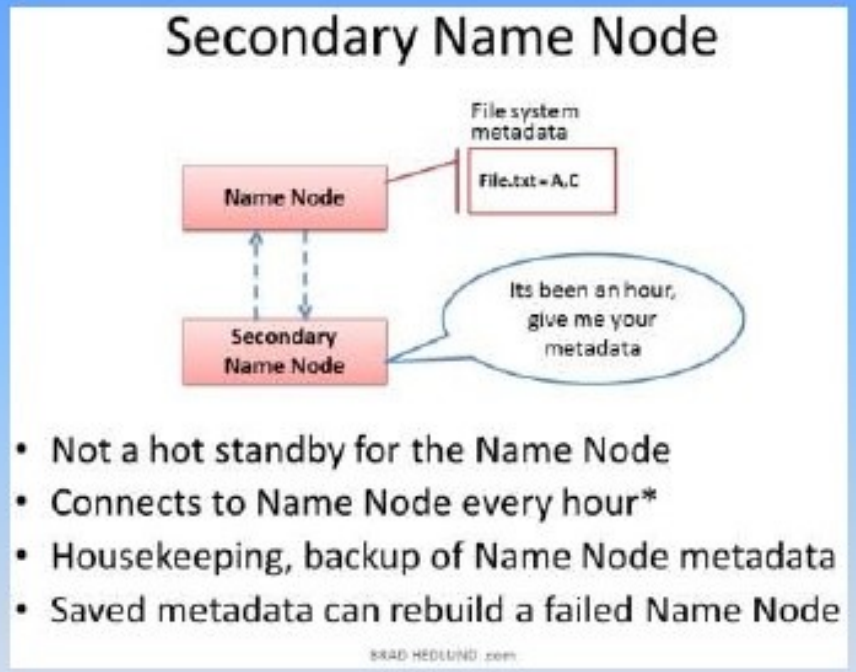

MapReduce – Word Count Example Flow
input
split
map
[combine]
Shuffle & sort
reduce
output
SQL DW SQL
SQL SSIS SSRS
SQL SSAS SSRS
DW BI SQL
SQL DW SQL
SQL SSIS SSRS
SQL SSAS SSRS
DW BI SQL
(SQL, 1) (DW, 1) (SQL, 1)
(SQL, 1) (SSIS, 1) (SSRS, 1)
(SQL, 1) (SSAS, 1) (SSRS, 1)
(DW, 1) (BI, 1) (SQL, 1)
(SQL, 2) (DW, 1)
(SQL, 1) (SSIS, 1) (SSRS, 1)
(SQL, 1) (SSAS, 1) (SSRS, 1)
(DW, 1) (BI, 1) (SQL, 1)
(BI, 1)
(DW, 1) (DW, 1)
(SQL, 2) (SQL, 1) (SQL, 1) (SQL, 1)
(SSAS, 1)
(SSIS, 1)
(SSRS, 1) (SSRS, 1)
(BI, 1)
(DW, 2)
(SQL, 5)
(SSAS, 1)
(SSIS, 1)
(SSRS, 2)
BI, 1
DW, 2
SQL, 5
SSAS, 1
SSIS, 1
SSRS, 2

Name Node



- Data Node sends Heartbeats
- Every 10th heartbeat is a Block report
- Name Node builds metadata from Block reports
- TCP – every 3 seconds
- If Name Node is down, HDFS is down

Secondary NameNode



The Secondary NameNode (SNN) is an assistant daemon for monitoring the state of the cluster HDFS. Like the NameNode, each cluster has one SNN, and it typically resides on its own machine as well. No other DataNode or TaskTracker daemons run on the same server.

The SNN differs from the NameNode in that this process doesn't receive or record any real-time changes to HDFS. Instead, it communicates with the NameNode to take snapshots of the HDFS metadata at intervals defined by the cluster configuration.

As mentioned earlier, the NameNode is a single point of failure for a Hadoop cluster, and the SNN snapshots help minimize the downtime and loss of data

Secondary NameNode

Copies FsImage and Transaction Log from Namenode to a temporary directory.

Merges FSImage and Transaction Log into a new FSImage in temporary directory.

Uploads new FSImage to the NameNode

- Transaction Log on NameNode is purged.

DataNode

A Block Server

- Stores data in the local file system (e.g. ext3)
- Stores metadata of a block (e.g. CRC)
- Serves data and metadata to Clients

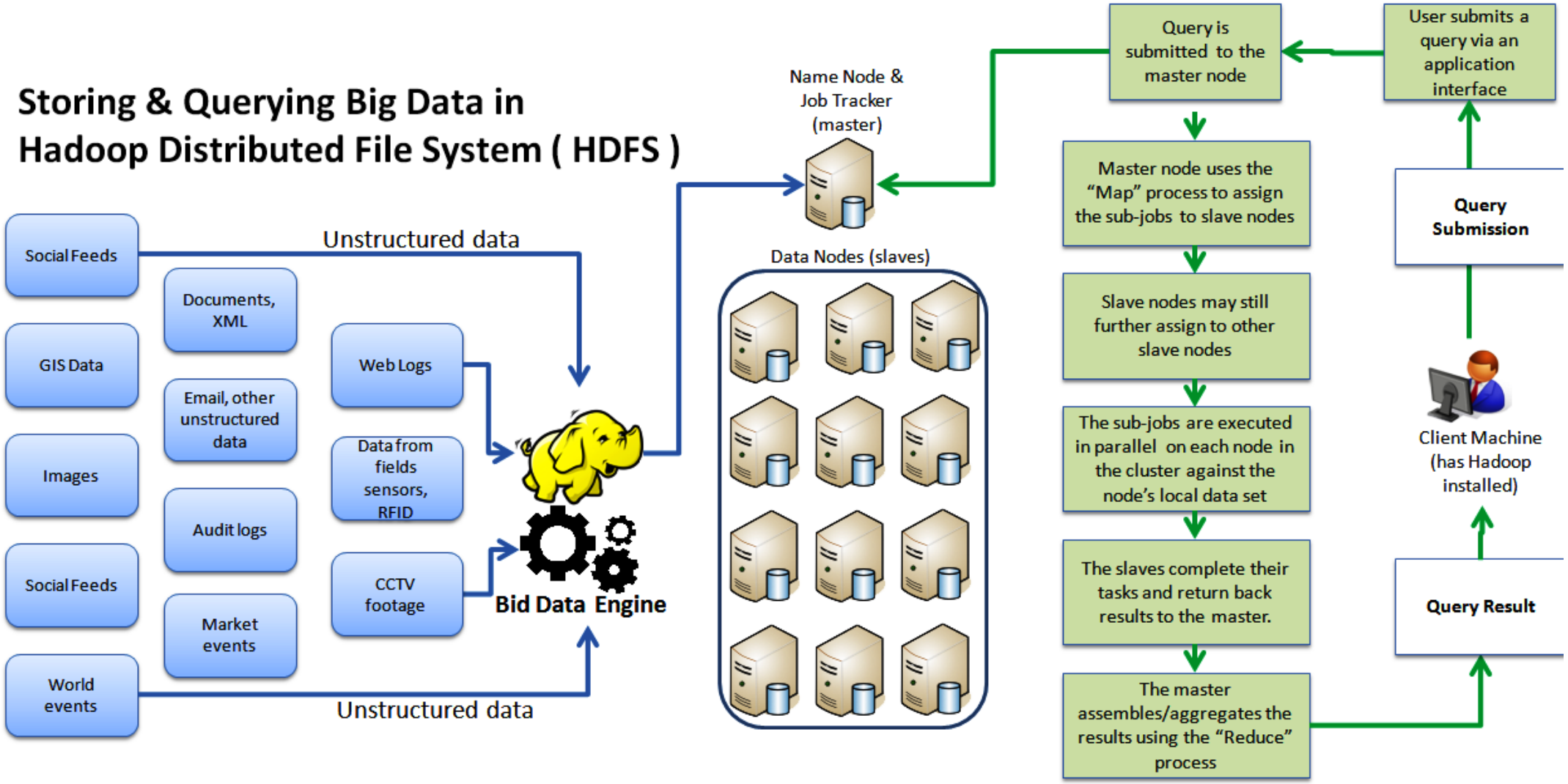
Block Report

- Periodically sends a report of all existing blocks to the NameNode

Facilitates Pipelining of Data

- Forwards data to other specified DataNodes

Storing & Querying Big Data in Hadoop Distributed File System (HDFS)



Data is chopped and stored on the HDFS – Hadoop Distributed File System

Data in the HDFS is scattered over numerous nodes for built in fault tolerance

HDFS has one master/name node and numerous slave/data nodes

Name node stores meta data and data nodes store data blocks

Name nodes and data Nodes reside on commodity servers i.e. x86

Each node/server offers local storage and computation

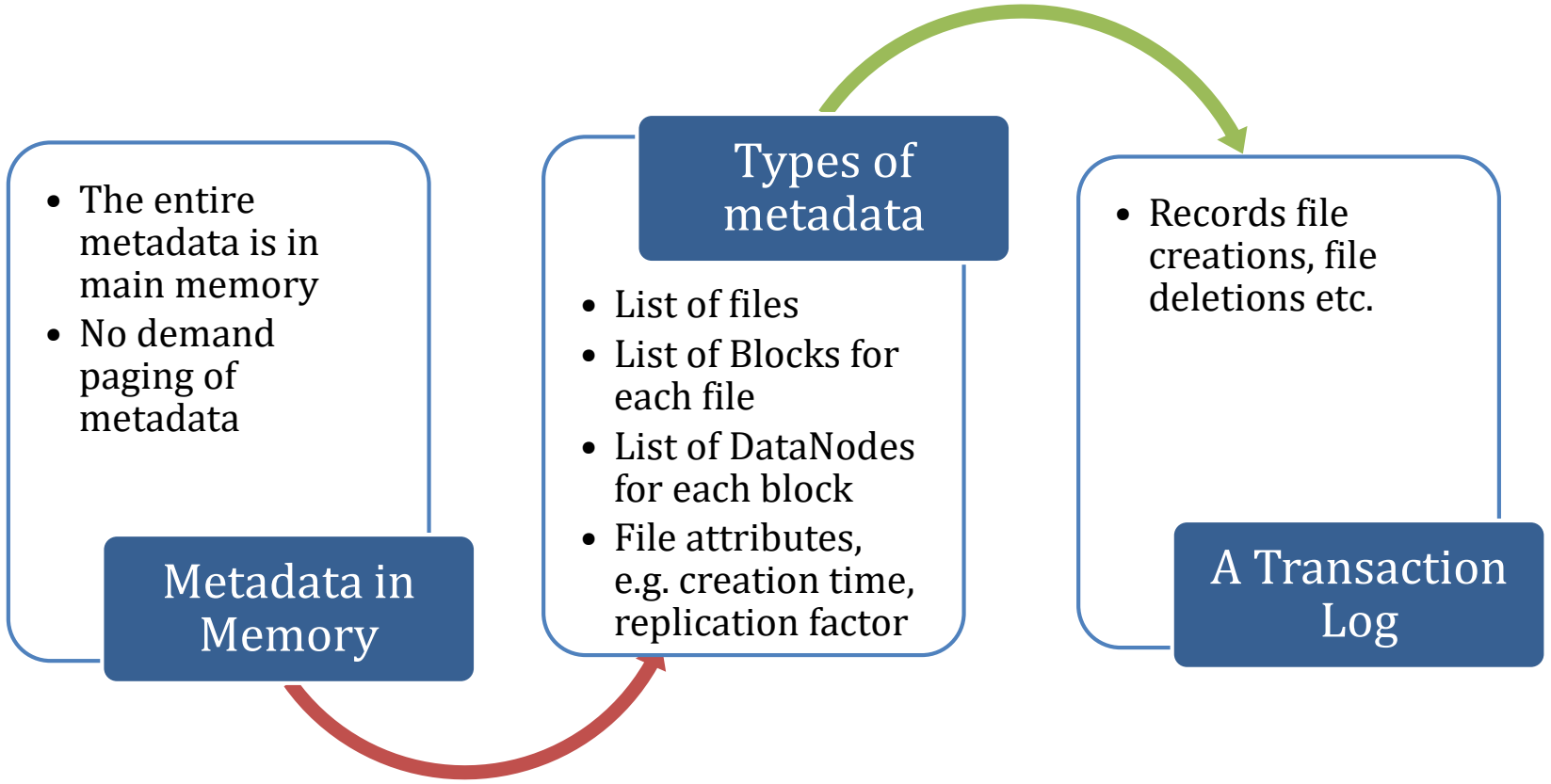
Data Retrieval

When a client wants to retrieve data:

Communicates with the NameNode to determine which blocks make up a file and on which data nodes those blocks are stored.

Then communicated directly with the data nodes to read the data.

NameNode Metadata



Heartbeats

DataNodes send hearbeat to the NameNode

Once every 3 seconds.

NameNode uses heartbeats to detect DataNode failure.

Hadoop 1.0: Limitations

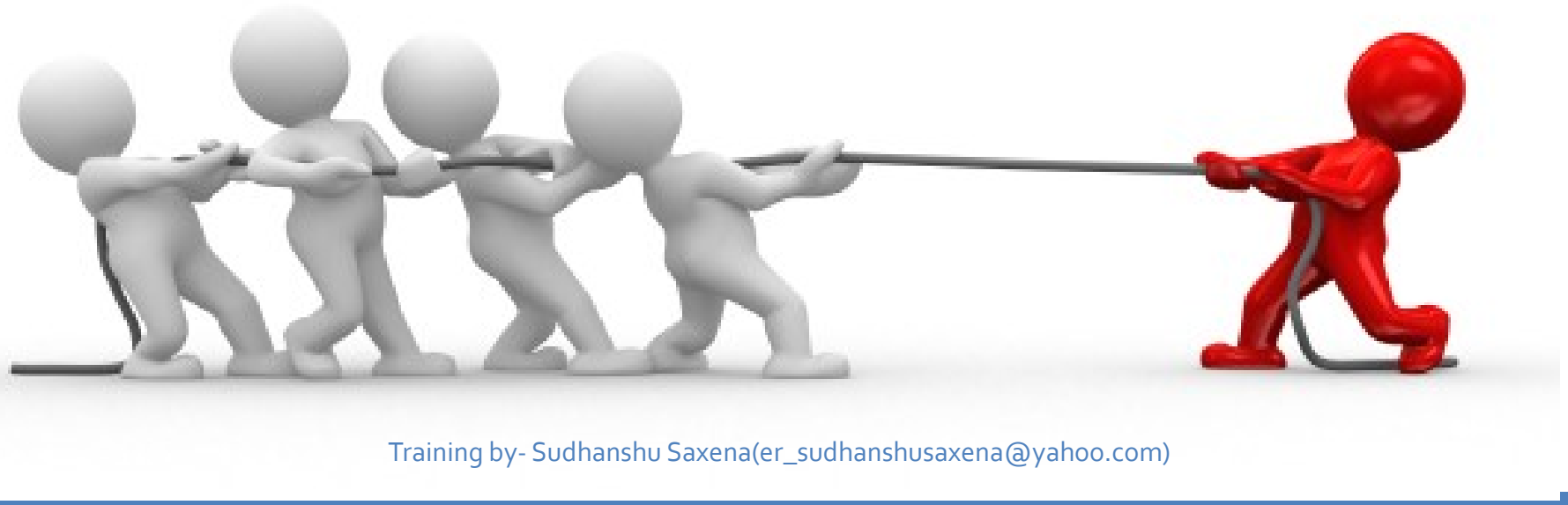
- **Only supports MapReduce, no other paradigms**
  - Everything needs to be cast to MapReduce
  - Iterative applications are slow
- **Scalability issues**
  - Max cluster size roughy 4,000 nodes
  - Max concurrent tasks, roughly 40,000 tasks
- **Availability**
  - System failures destroy running and queued jobs
- **Resource utilization**
  - Hard, static partitioning of resources in Map or Reduce slots
  - Non-optimal resource utilization

NameNode Failure

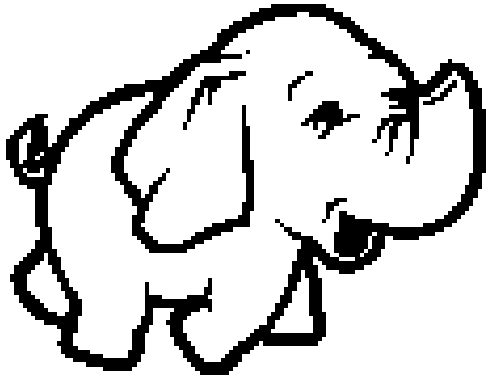
- A single point of failure

- Transaction Log stored in multiple directories
  - A directory on the local file system
  - A directory on a remote file system (NFS/CIFS)

- Need to develop a real HA solution



Hadoop 1.0 Drawbacks - NameNode

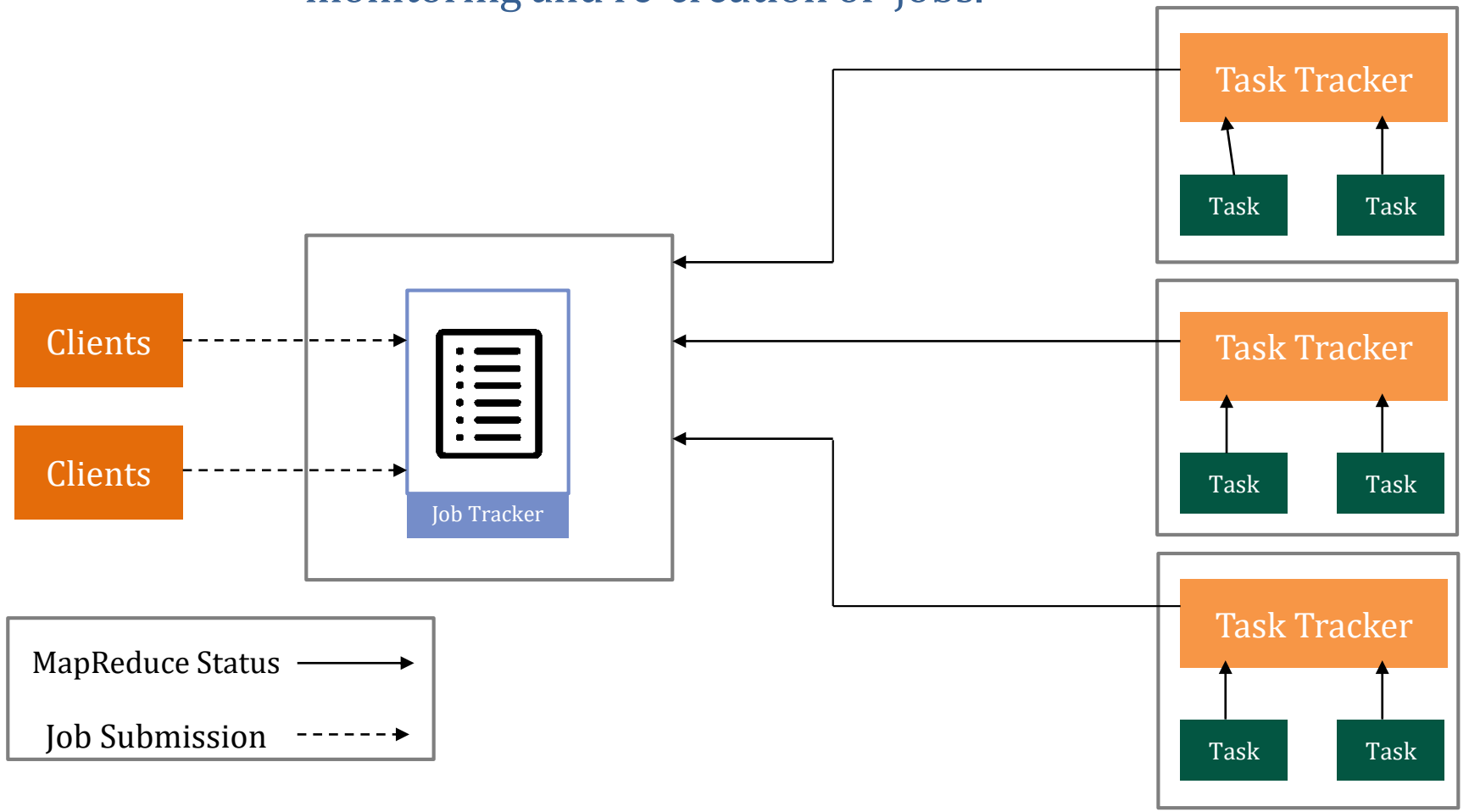


Hadoop 1.0 - Drawbacks

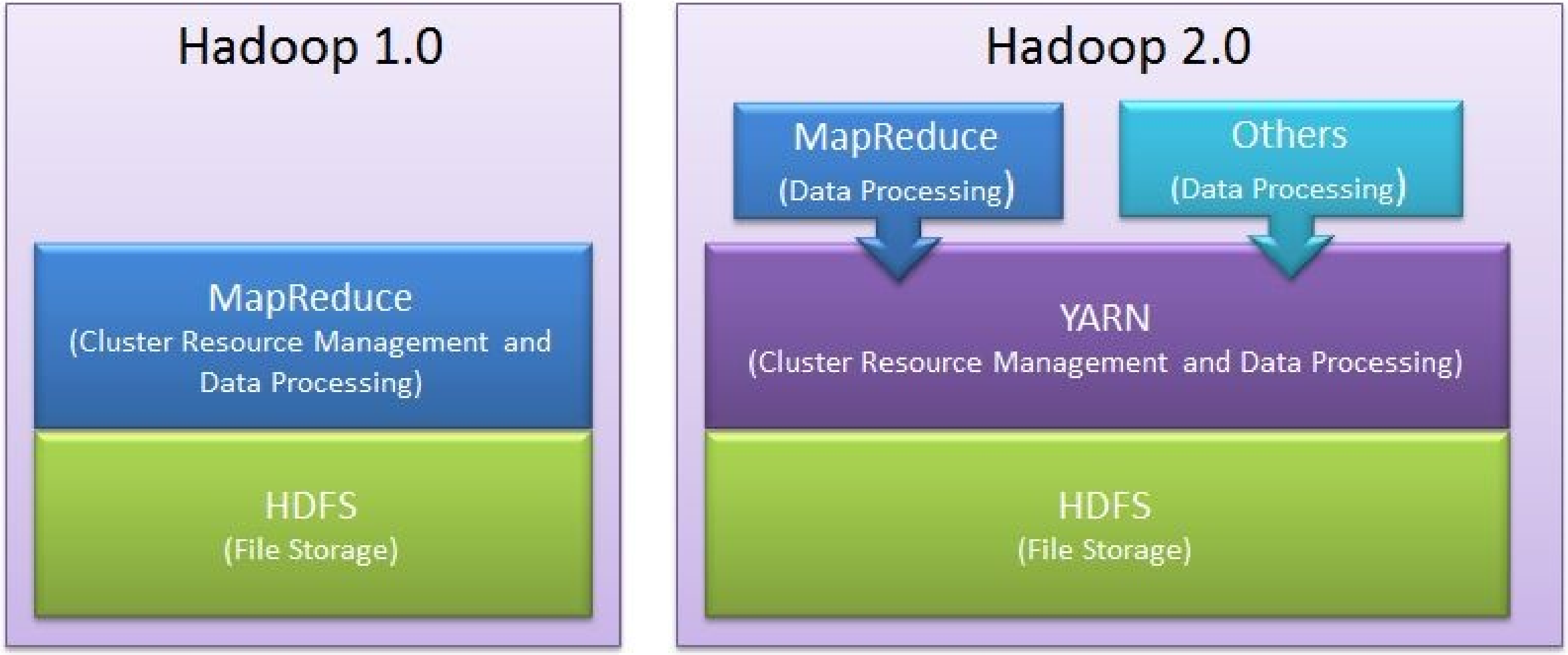
- Namenode is not at all backed up by any hot standby systems.
- Secondary Namenode is not a hot standby and has only hourly backup data.
- In case of primary Namenode failure, the entire cluster becomes inaccessible.
- The namespaces are limited to one, with no horizontal scalability.

Hadoop 1.0 Drawbacks – Job Tracker

The Job Tracker becomes a bottleneck with all the initiation, monitoring and re-creation of jobs.



Difference between Hadoop 1.0 and 2.0



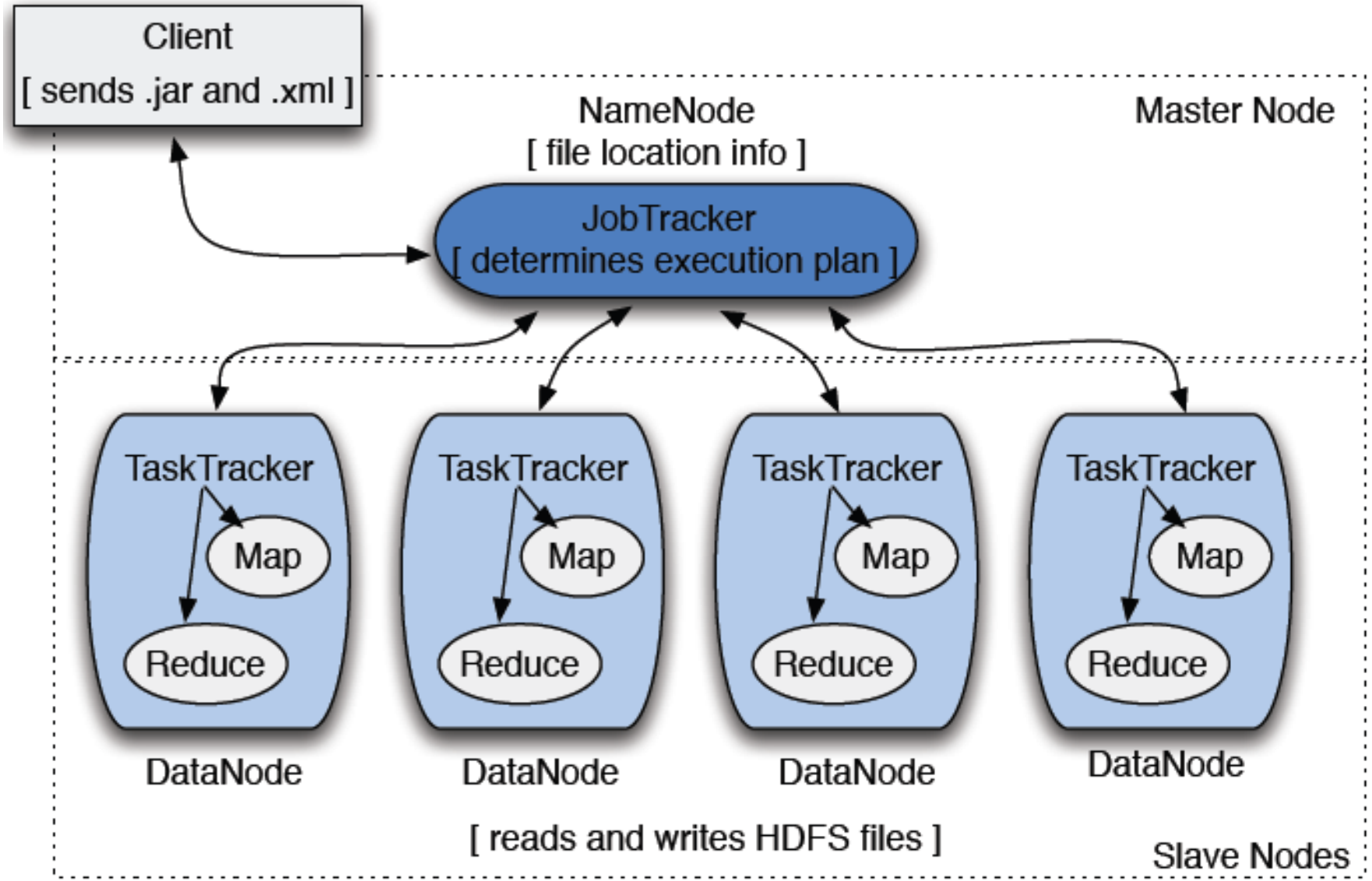
JobTracker and TaskTracker

Job Tracker –

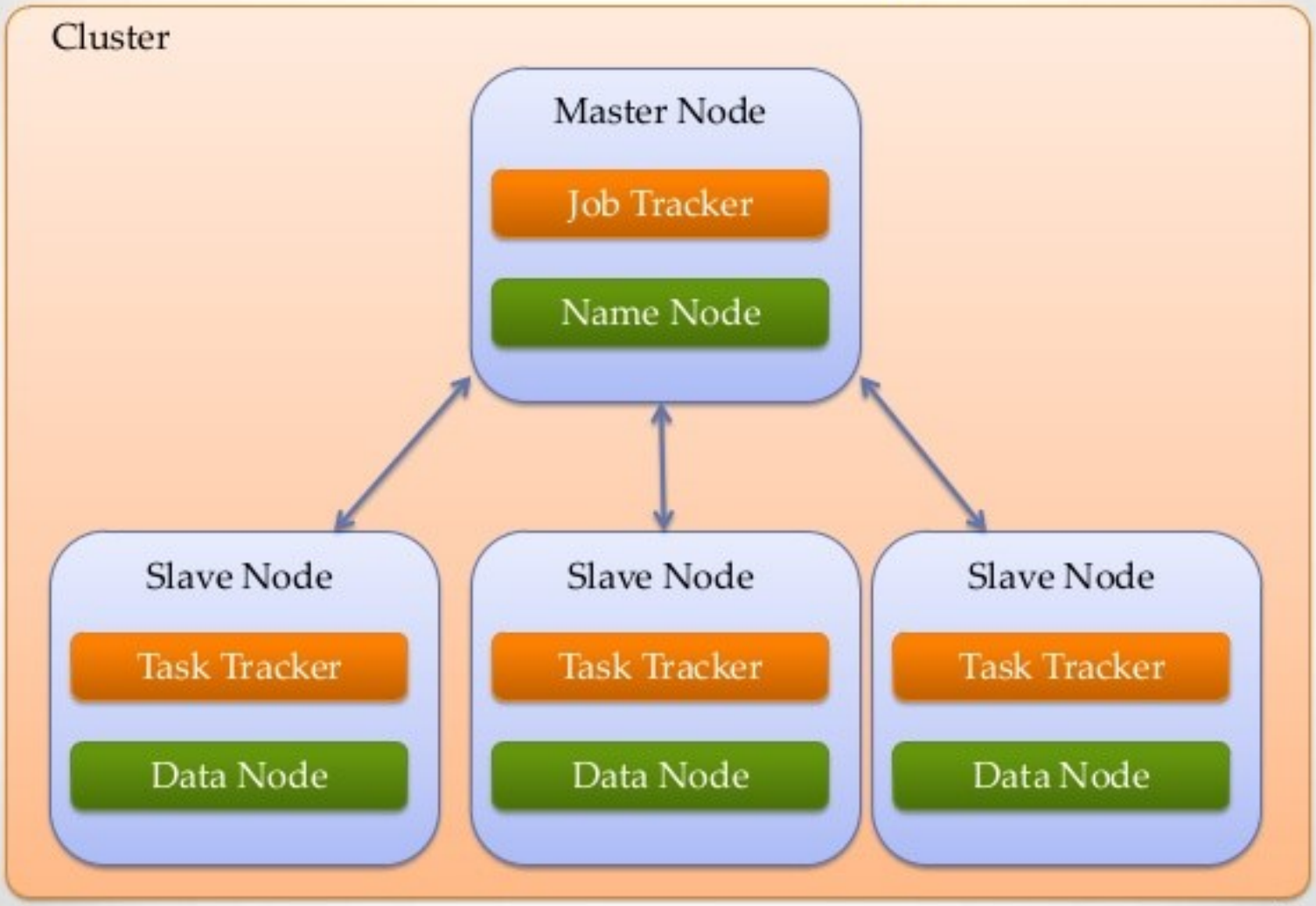
- JobTracker process runs on a separate node and not usually on a DataNode.
- JobTracker is an essential Daemon for MapReduce execution in MRv1. It is replaced by ResourceManager/ApplicationMaster in MRv2.
- JobTracker receives the requests for MapReduce execution from the client.
- JobTracker talks to the NameNode to determine the location of the data.
- JobTracker finds the best TaskTracker nodes to execute tasks based on the data locality (proximity of the data) and the available slots to execute a task on a given node.
- JobTracker monitors the individual TaskTrackers and the submits back the overall status of the job back to the client.
- JobTracker process is critical to the Hadoop cluster in terms of MapReduce execution.
- When the JobTracker is down, HDFS will still be functional but the MapReduce execution can not be started and the existing MapReduce jobs will be halted.

TaskTracker –

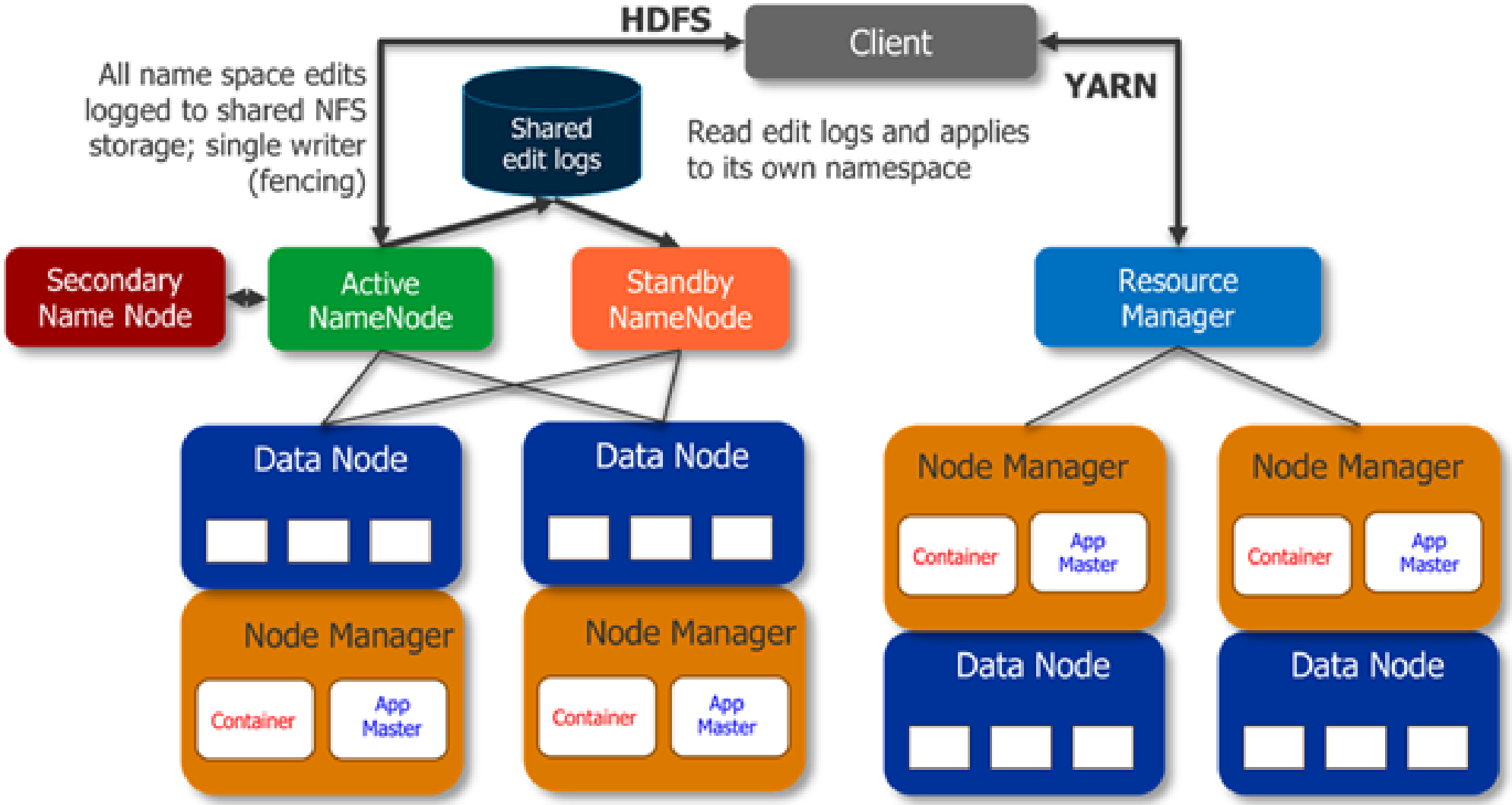
- TaskTracker runs on DataNode. Mostly on all DataNodes.
- TaskTracker is replaced by Node Manager in MRv2.
- Mapper and Reducer tasks are executed on DataNodes administered by TaskTrackers.
- TaskTrackers will be assigned Mapper and Reducer tasks to execute by JobTracker.
- TaskTracker will be in constant communication with the JobTracker signalling the progress of the task in execution.
- TaskTracker failure is not considered fatal. When a TaskTracker becomes unresponsive, JobTracker will assign the task executed by the TaskTracker to another node.


Client
[sends .jar and .xml]
NameNode
[file location info]
Master Node
JobTracker
[determines execution plan]
TaskTracker
Map
Reduce
DataNode
TaskTracker
Map
Reduce
DataNode
TaskTracker
Map
Reduce
DataNode
TaskTracker
Map
Reduce
DataNode
[reads and writes HDFS files]
Slave Nodes

How Does Hadoop Work?



Hadoop 2.0 architecture



Hadoop 1.0(MR+HDFS)	Hadoop 2.0 (YARN+HDFS)
Job tracker was responsible for job life cycle management and cluster resource management, turn out to be the bottleneck for scalability	Resource management and job life cycle management is split between Resource manager and Application master
Only map-reduce jobs are possible to run	Other jobs are also possible to run for example spark
Only FIFO scheduler is present	Capacity Scheduler, Fair scheduler and FIFO scheduler
Resources optimization is not good	Resources are used in a dynamic requirement basis which help to utilize cluster resources to the best
Less secure	Secure cluster from malicious client applications, using secret token managers and Kerberos for authentication
No High Availability	High Availability is present

Table: Comparison between RDBMS and Hadoop		
Characrteristics	**RDBMS**	**Hadoop**
Basic Description	Traditional row-column databases used for both transactional systems, reporting, and archiving.	An open-source approach to storing data in a file system across a range of commodity hardware and processing it utilizing parallelism (multiple systems at once)
Manufacturers	Sql Server, MySql, Oracle, etc	Hadoop implementations by CloudEra, Intel, Amazon, Hortonworks
Best for applications	Reads & Writes, "reasonable" data sets (< 1B rows)	Inexpensive storage of lots of data, structured & semi-structured
Strength, Weakness	Massive data volumes, unstructured & semi-structured data	Complex, code-based, incompatible approaches in market, writes (one at a time)
Scalability	Challenging to "scale-out"	Strong bias to the open-source community & Java

Thank You!!!

Lets Move to Practical.. ☺