JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE	Jaipur Engineering College and Research Centre, Shri Ram ki Nangal, via Sitapura RIICO Jaipur- 302 022.	Academic year- 2020-2021

Vision & Mission of the Department

Vision of the Department

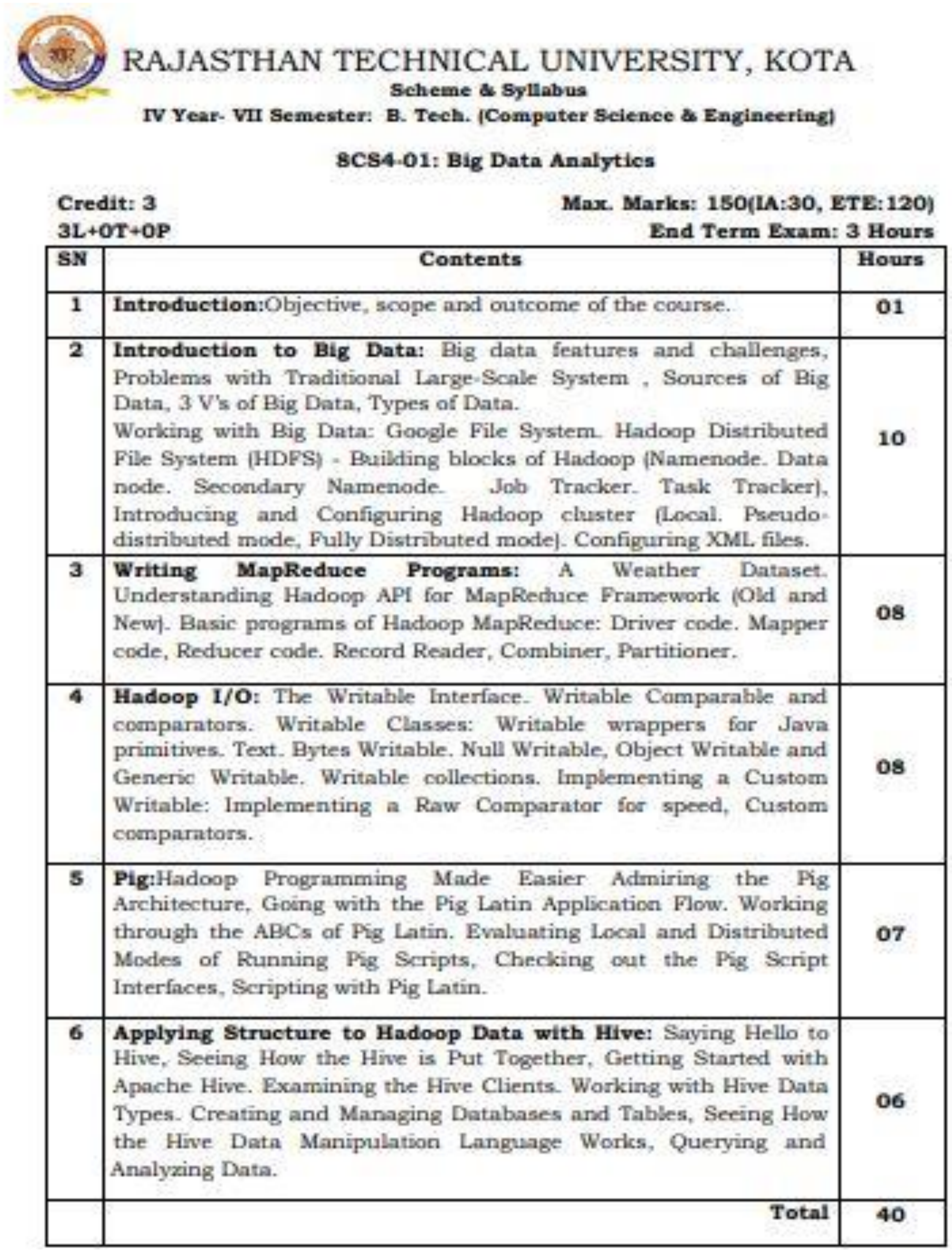
To become renowned Centre of excellence in computer science and engineering and make competent engineers & professionals with high ethical values prepared for lifelong learning.

Mission of the Department

M1: To impart outcome based education for emerging technologies in the field of computer science and engineering.
M2: To provide opportunities for interaction between academia and industry.
M3: To provide platform for lifelong learning by accepting the change in technologies
M4: To develop aptitude of fulfilling social responsibilities.

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE	Jaipur Engineering College and Research Centre, Shri Ram ki Nangal, via Sitapura RIICO Jaipur- 302 022.	Academic year- 2020-2021

SYLLABUS



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

Scheme & Syllabus

IV Year- VII Semester: B. Tech. (Computer Science & Engineering)

8CS4-01: Big Data Analytics

Credit: 3 **Max. Marks: 150(IA:30, ETE:120)**

3L+0T+0P **End Term Exam: 3 Hours**

SN	Contents	Hours
1	**Introduction:**Objective, scope and outcome of the course.	01
2	**Introduction to Big Data:** Big data features and challenges, Problems with Traditional Large-Scale System , Sources of Big Data, 3 V's of Big Data, Types of Data. Working with Big Data: Google File System. Hadoop Distributed File System (HDFS) - Building blocks of Hadoop (Namenode. Data node. Secondary Namenode. Job Tracker. Task Tracker), Introducing and Configuring Hadoop cluster (Local. Pseudo-distributed mode, Fully Distributed mode). Configuring XML files.	10
3	**Writing MapReduce Programs:** A Weather Dataset. Understanding Hadoop API for MapReduce Framework (Old and New). Basic programs of Hadoop MapReduce: Driver code. Mapper code, Reducer code. Record Reader, Combiner, Partitioner.	08
4	**Hadoop I/O:** The Writable Interface. Writable Comparable and comparators. Writable Classes: Writable wrappers for Java primitives. Text. Bytes Writable. Null Writable, Object Writable and Generic Writable. Writable collections. Implementing a Custom Writable: Implementing a Raw Comparator for speed, Custom comparators.	08
5	**Pig:**Hadoop Programming Made Easier Admiring the Pig Architecture, Going with the Pig Latin Application Flow. Working through the ABCs of Pig Latin. Evaluating Local and Distributed Modes of Running Pig Scripts, Checking out the Pig Script Interfaces, Scripting with Pig Latin.	07
6	**Applying Structure to Hadoop Data with Hive:** Saying Hello to Hive, Seeing How the Hive is Put Together, Getting Started with Apache Hive. Examining the Hive Clients. Working with Hive Data Types. Creating and Managing Databases and Tables, Seeing How the Hive Data Manipulation Language Works, Querying and Analyzing Data.	06
	Total	**40**

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE	Jaipur Engineering College and Research Centre, Shri Ram ki Nangal, via Sitapura RIICO Jaipur- 302 022.	Academic year- 2020-2021

PROGRAM OUTCOMES

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend

and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE	Jaipur Engineering College and Research Centre, Shri Ram ki Nangal, via Sitapura RIICO Jaipur- 302 022.	Academic year- 2020-2021

Jaipur Engineering College and Research Centre

Department of Computer Science & Engineering

Subject – Big Data Analytics **Subject code – 8CS4 - 01**

Semester - VIII **[L/T/P - 3/0/0]**

Course Outcome

CO1. To understand the features, file system and challenges of big data.

CO2. To learn and analyze big data analytics tools like Map Reduce, Hadoop.

CO3. To apply and evaluate Hadoop programming with respect to PIG architecture.

CO4. To create and analyze database with Hive and related tools.

CO- PO Mapping

H=3, M=2, L=1

Semester	Subject	Code	L/T/P	CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
VIII	**Big Data Analytics**	8CS4 - 01	L	CO1	3	2	2	2	1	1	1	1	1	1	2	3
			L	CO2	3	3	3	2	2	1	1	1	1	1	2	3
			L	CO3	3	3	3	2	2	1	1	1	1	2	2	3
			L	CO4	3	3	3	2	2	2	2	2	2	2	2	3

PROGRAM EDUCATIONAL OBJECTIVES:

1. To provide students with the fundamentals of Engineering Sciences with more emphasis in **Computer Science &Engineering** by way of analyzing and exploiting engineering challenges.
2. To train students with good scientific and engineering knowledge so as to comprehend, analyze, design, and create novel products and solutions for the real life problems.
3. To inculcate professional and ethical attitude, effective communication skills, teamwork skills, multidisciplinary approach, entrepreneurial thinking and an ability to relate engineering issues with social issues.
4. To provide students with an academic environment aware of excellence, leadership, written ethical codes and guidelines, and the self motivated life-long learning needed for a successful professional career.
5. To prepare students to excel in Industry and Higher education by Educating Students along with High moral values and Knowledge

JECRC Foundation

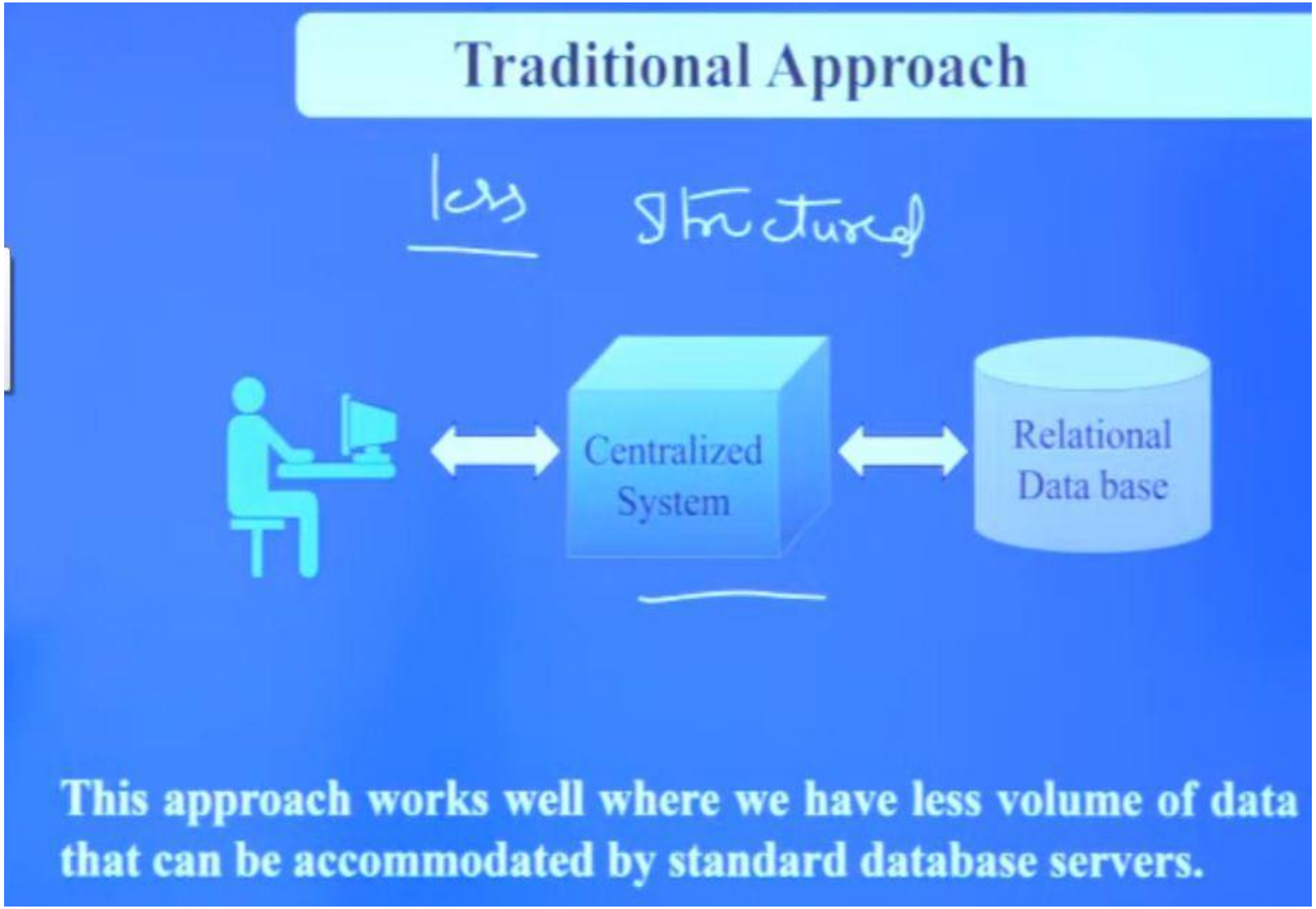
JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE

JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE

Year & Sem – IV year & VIII Sem
Subject – Big Data Analytics
Unit – I

Google File System(GFS)

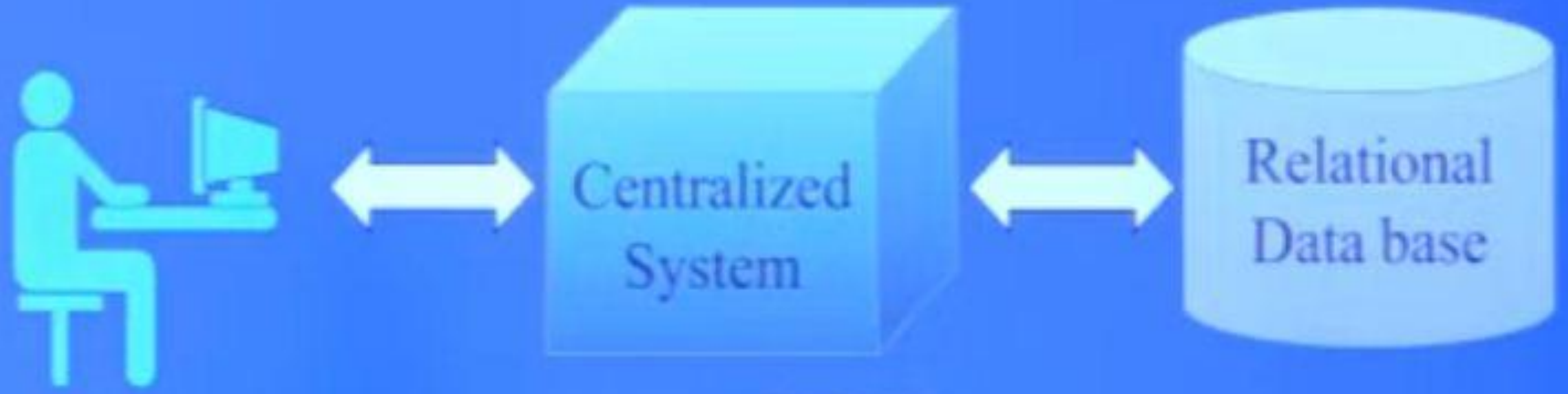
- Google File System is a proprietary distributed file system developed by Google to provide efficient, reliable access to data using large clusters of commodity hardware.
- A **distributed file system** (DFS) is a **file system** with data stored on a server.
- The DFS makes it convenient to share information and **files** among users on a network in a controlled and authorized way.



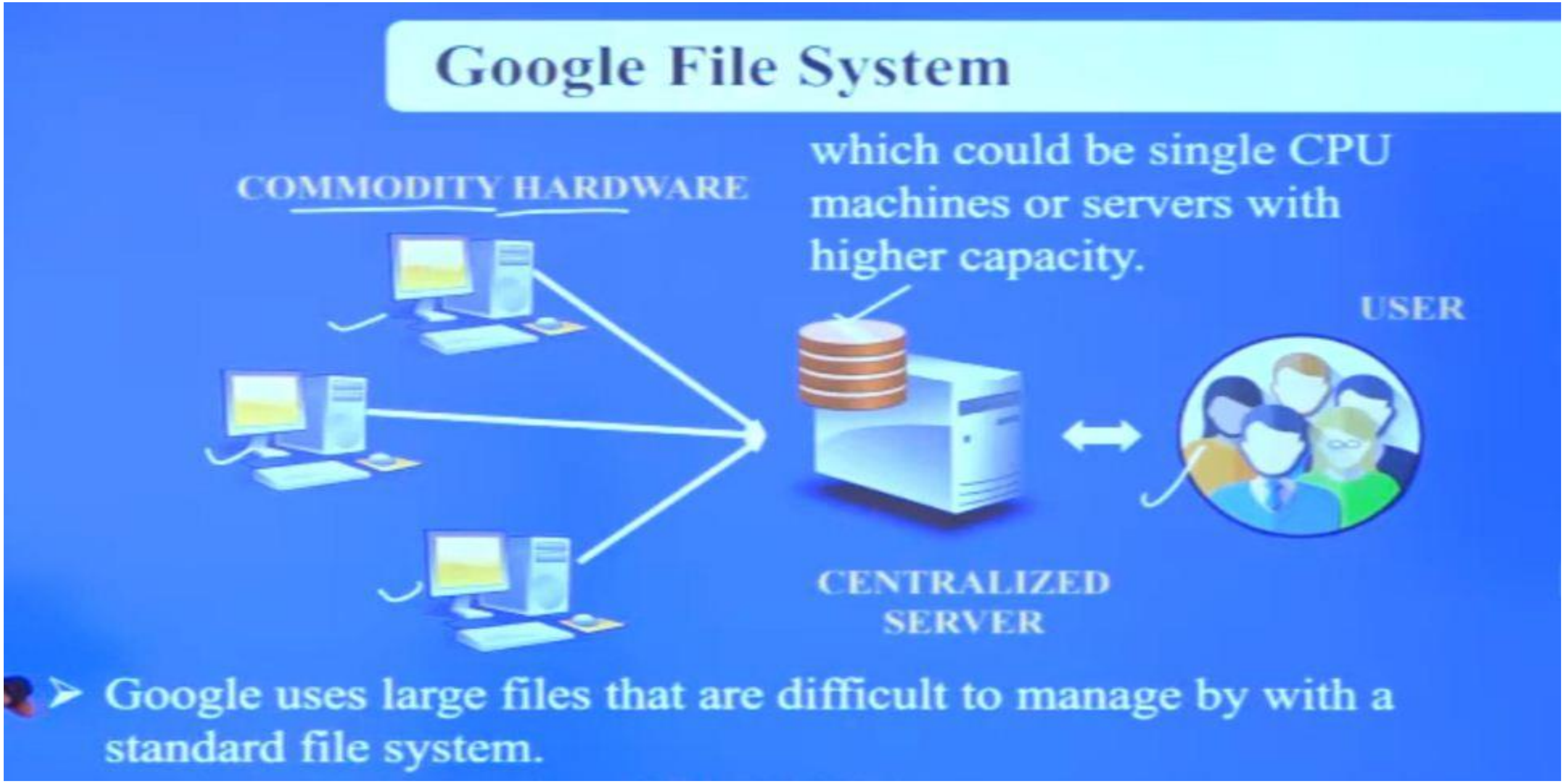
Traditional Approach

less

Structured

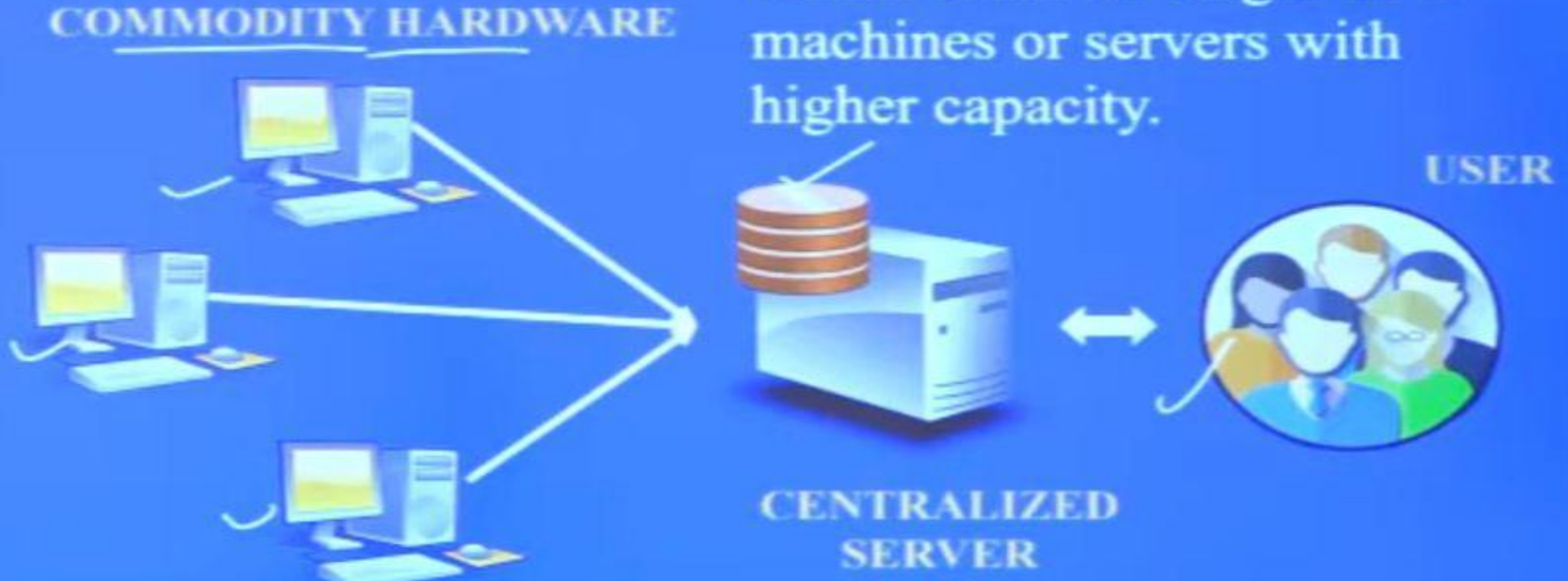


This approach works well where we have less volume of data that can be accommodated by standard database servers.

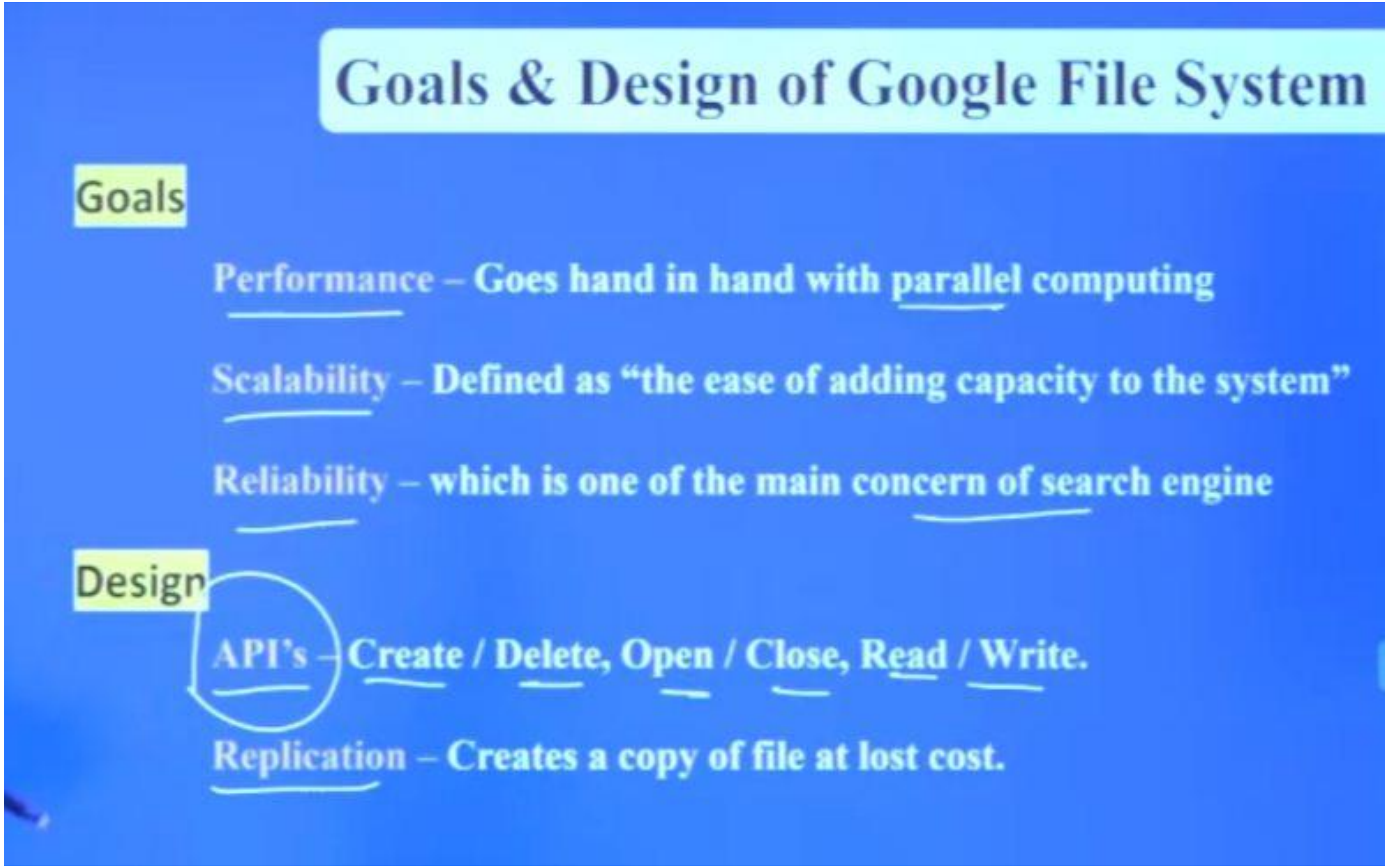


Google File System

which could be single CPU machines or servers with higher capacity.



➢ Google uses large files that are difficult to manage by with a standard file system.



Goals & Design of Google File System

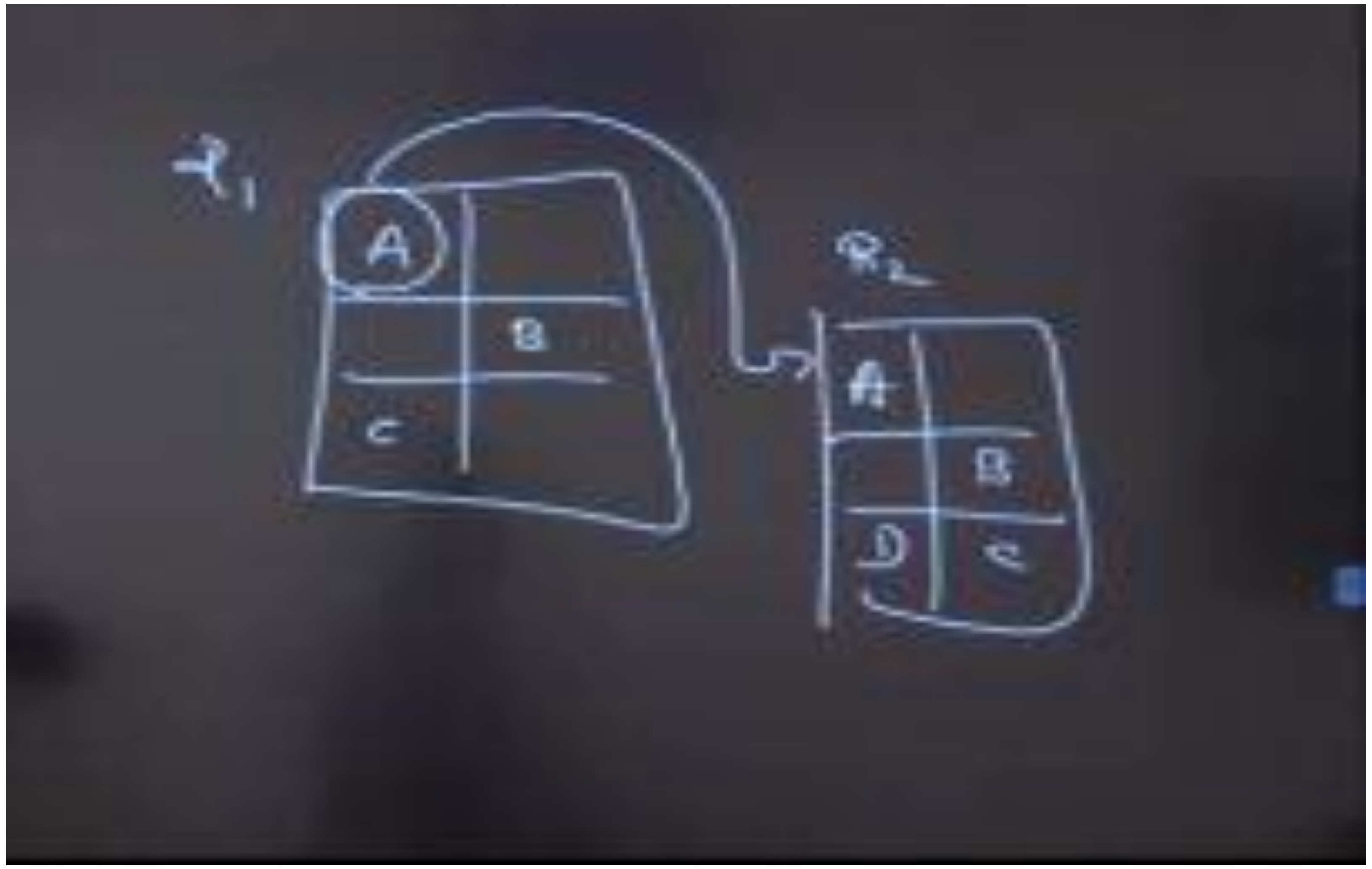
Goals

- Performance – Goes hand in hand with parallel computing
- Scalability – Defined as "the ease of adding capacity to the system"
- Reliability – which is one of the main concern of search engine

Design

- API's – Create / Delete, Open / Close, Read / Write.
- Replication – Creates a copy of file at lost cost.

Replication



Advantages of GFS

- The Google File System (GFS) is a scalable distributed file system for large distributed data intensive applications.
- It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients.
- GFSsupport the usual operations such as create, delete, open, close, read, and write files.
- allows multiple clients to append data to the same file concurrently while guaranteeing the atomicity of each individual clients
- A GFS cluster consists of a single **master** and multiple **chunk servers** and is accessed by Multiple clients.
- Files are divided into fixed-size chunks. Each chunk is identified by a fixed and globally unique **64-bit chunk** handle assigned by the master at the time of chunk creation.

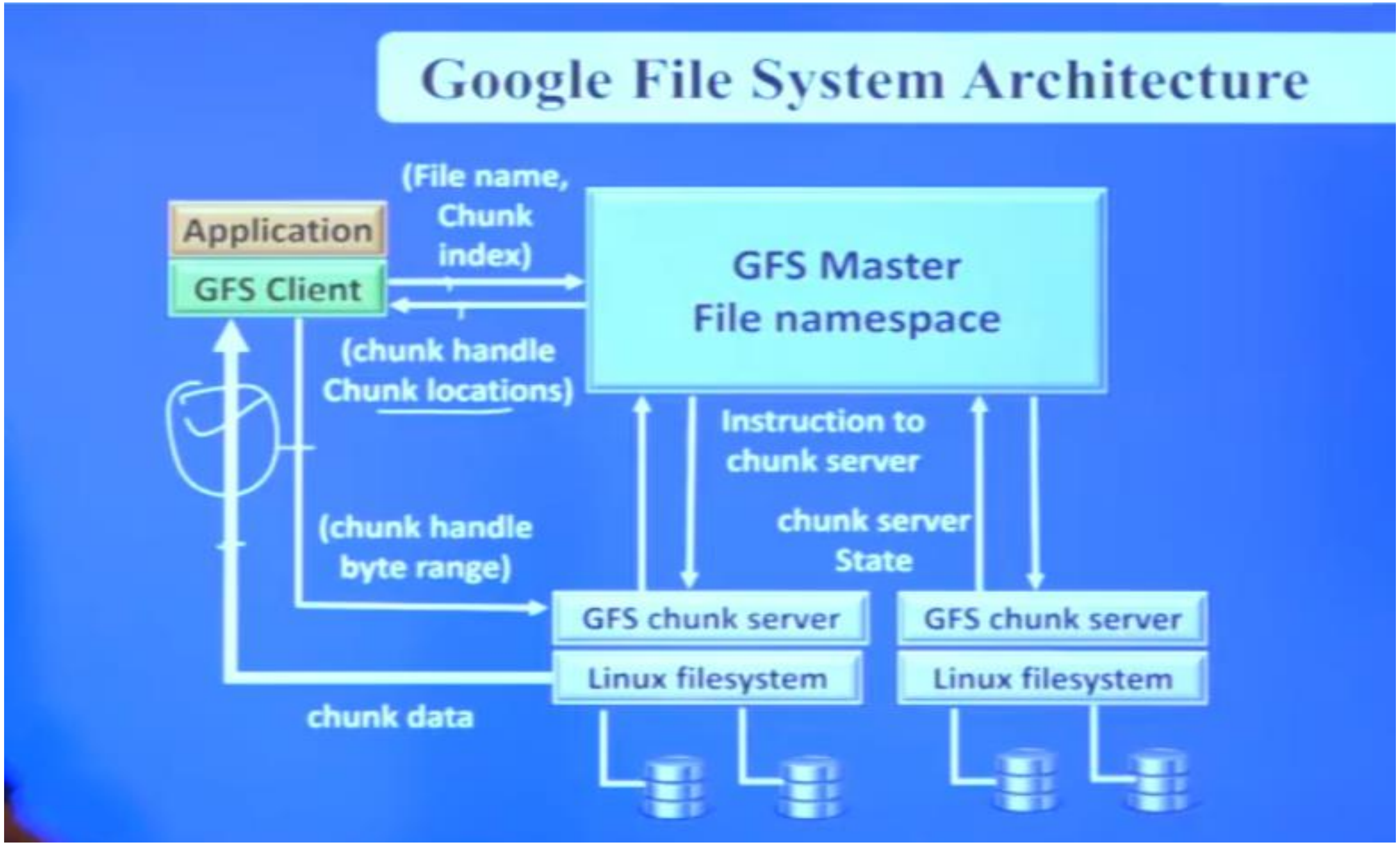
- For reliability, each chunk is replicated on multiple chunk servers. By default, there will three replicas and this value can be changed by user
- The **master** maintains all file system metadata. This includes the namespace, access control information, the mapping from files to chunks, and the current locations of chunks.
- The master periodically communicates with each chunk server in **Heart Beat** messages to give it instructions and collect its state.

GFS ARCHITECTURE

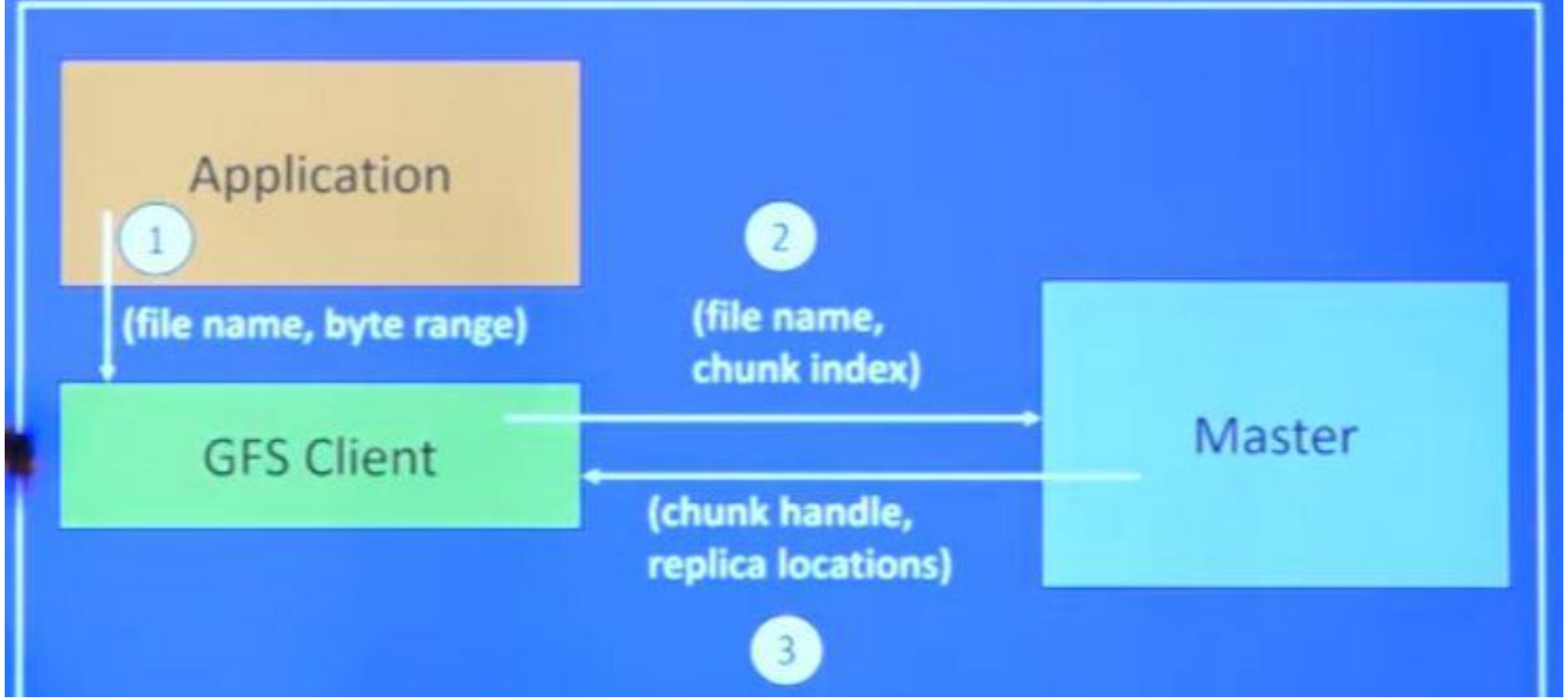
- Google organized the GFS into **clusters** of computers. A cluster is simply a network of computers. Each cluster might contain hundreds or even thousands of machines. Within GFS clusters there are three kinds of entities: **clients**, **master servers** and **chunk servers**.
- "client" refers to any entity that makes a file request. Requests can range from retrieving and manipulating existing files to creating new files on the system. Clients can be other computers or computer applications.

- The master server acts as the coordinator for the cluster.
- The master server also keeps track of **metadata**, which is the information that describes chunks.
- Chunk servers are responsible for storing the 64-MB file chunks.
- By default, the GFS makes three replicas per chunk

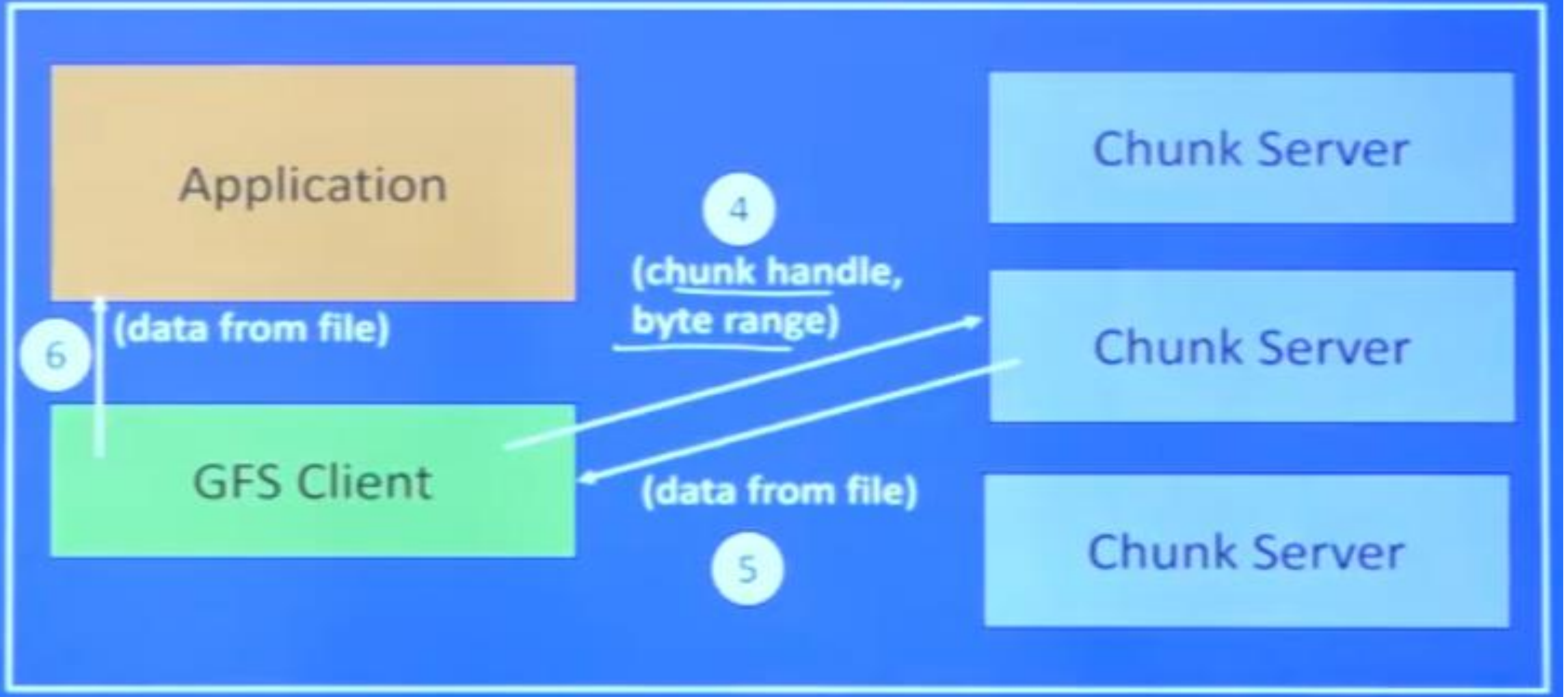
GFS Architecture



Google File System READ Operation Execution Flow



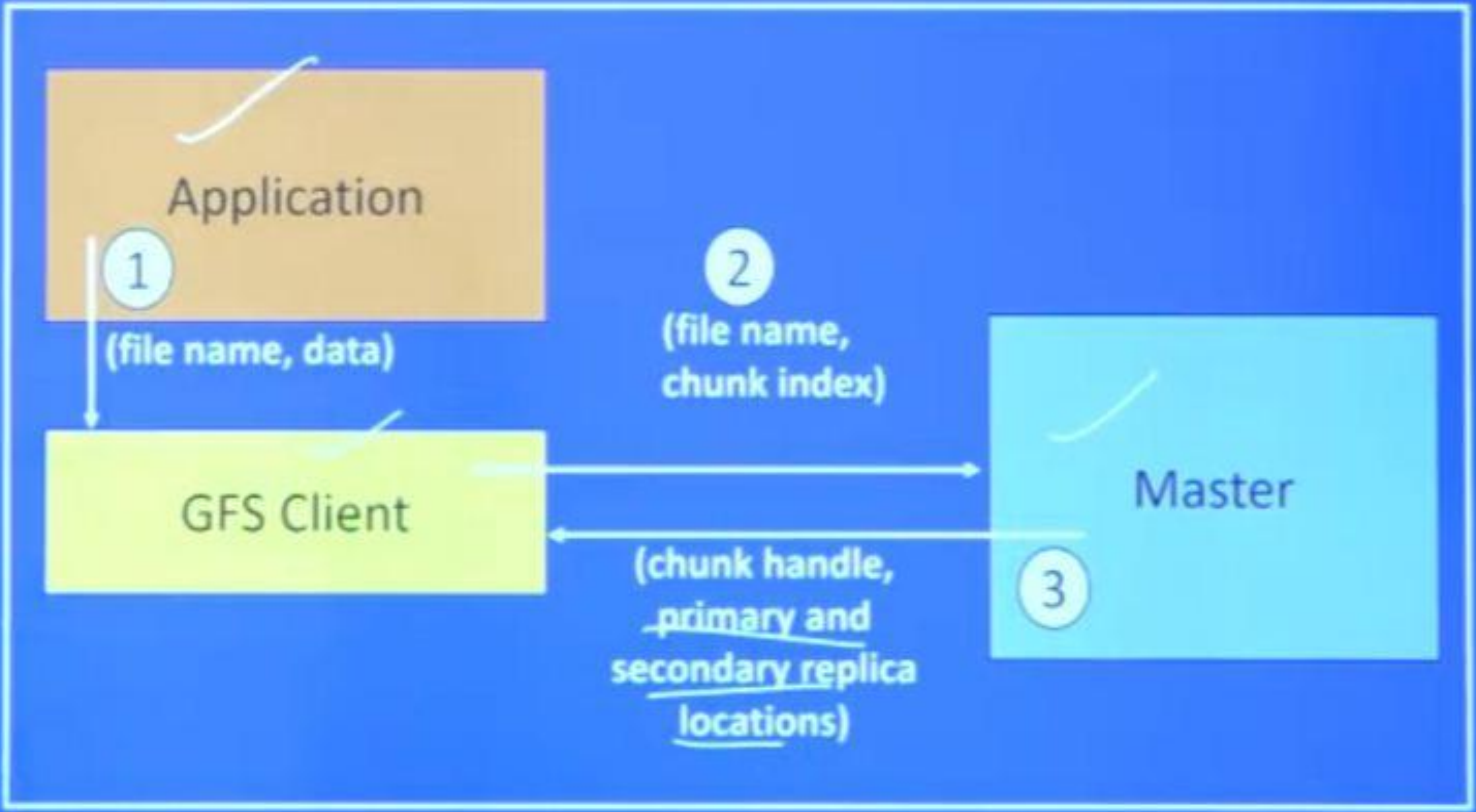
Google File System READ Operation Execution Flow



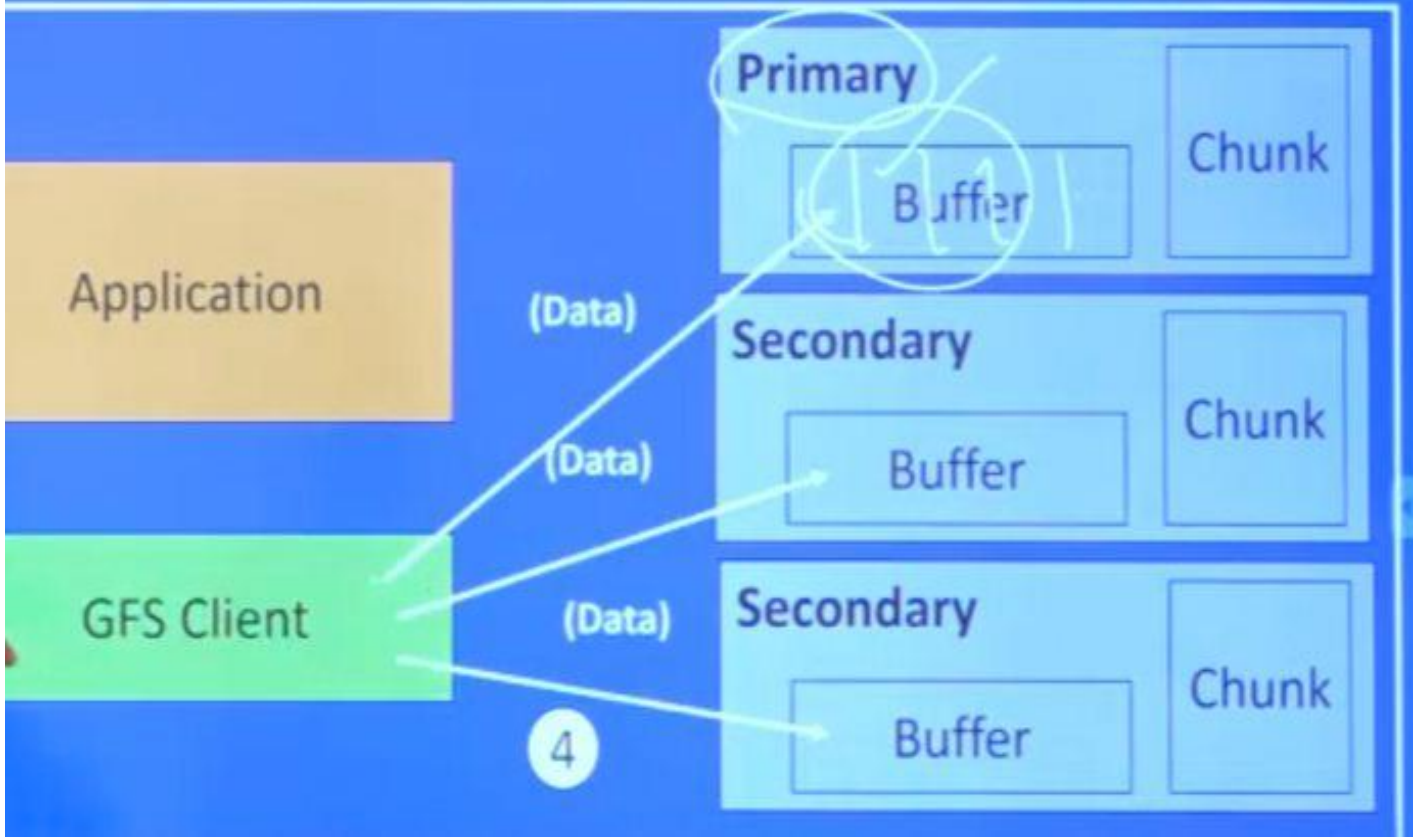
Google File System READ Algorithm

1. Application originates the read request.
2. GFS client translates the request from (filename, byte range) -> (filename, chunk index), and sends it to master.
3. Master responds with chunk handle and replica locations (i.e. chunk servers where the replicas are stored).
4. Client picks a location and sends the (chunk handle, byte range) request to that location.
5. Chunk server sends requested data to the client.
6. Client forwards the data to the application.

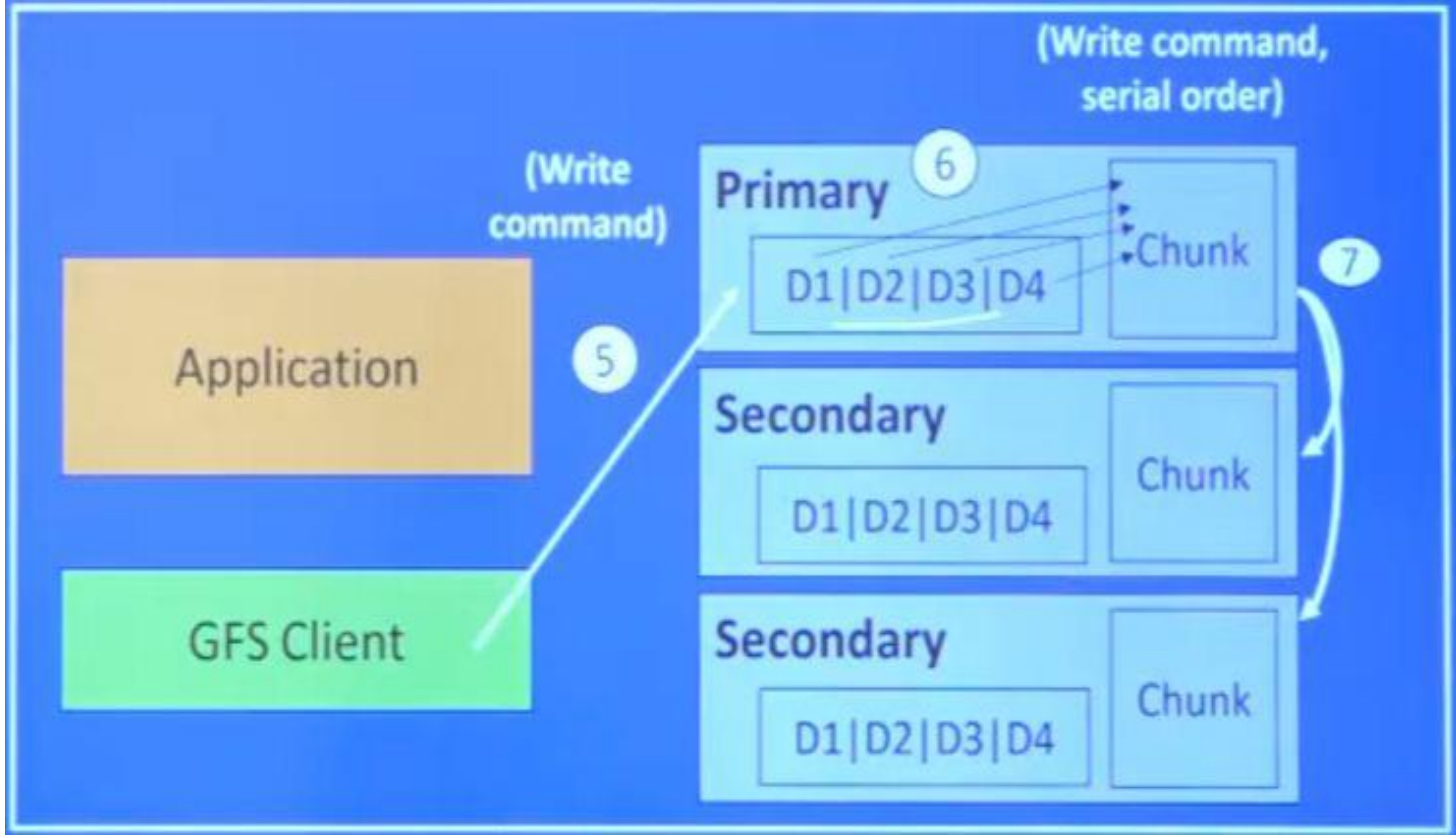
Google File System WRITE Operation Execution Flow



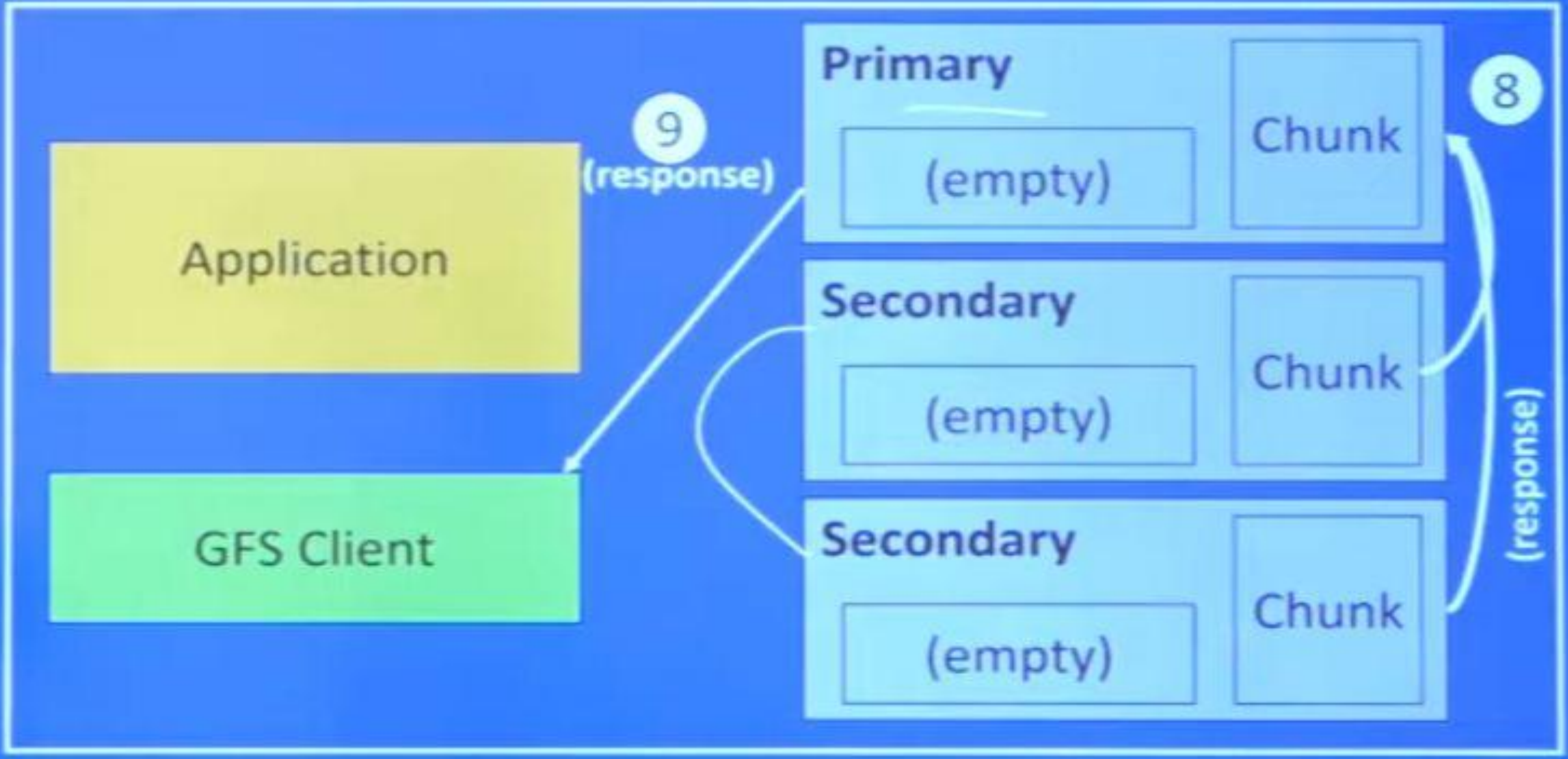
Google File System WRITE Operation Execution Flow

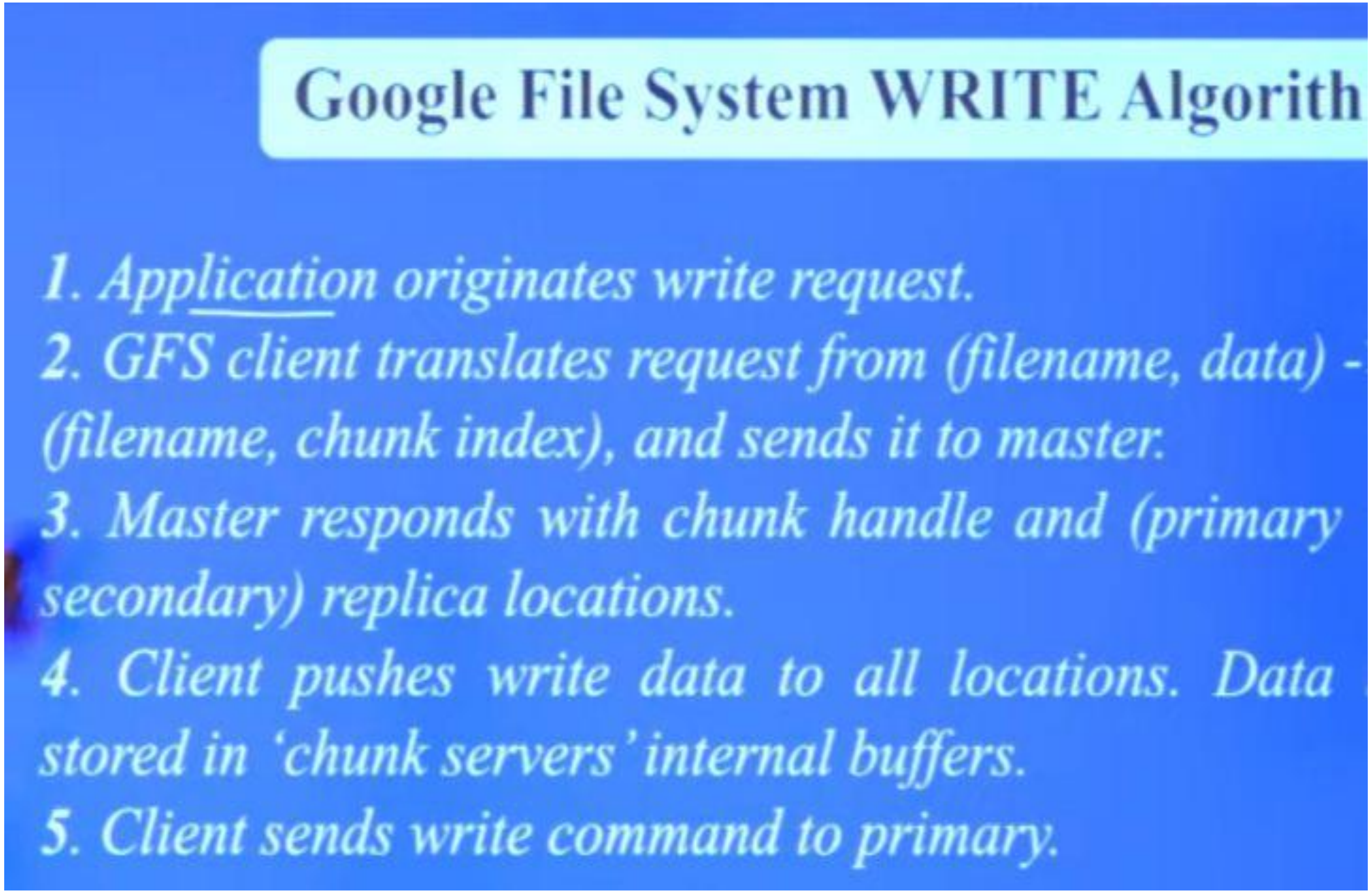


Google File System WRITE Operation Execution Flow



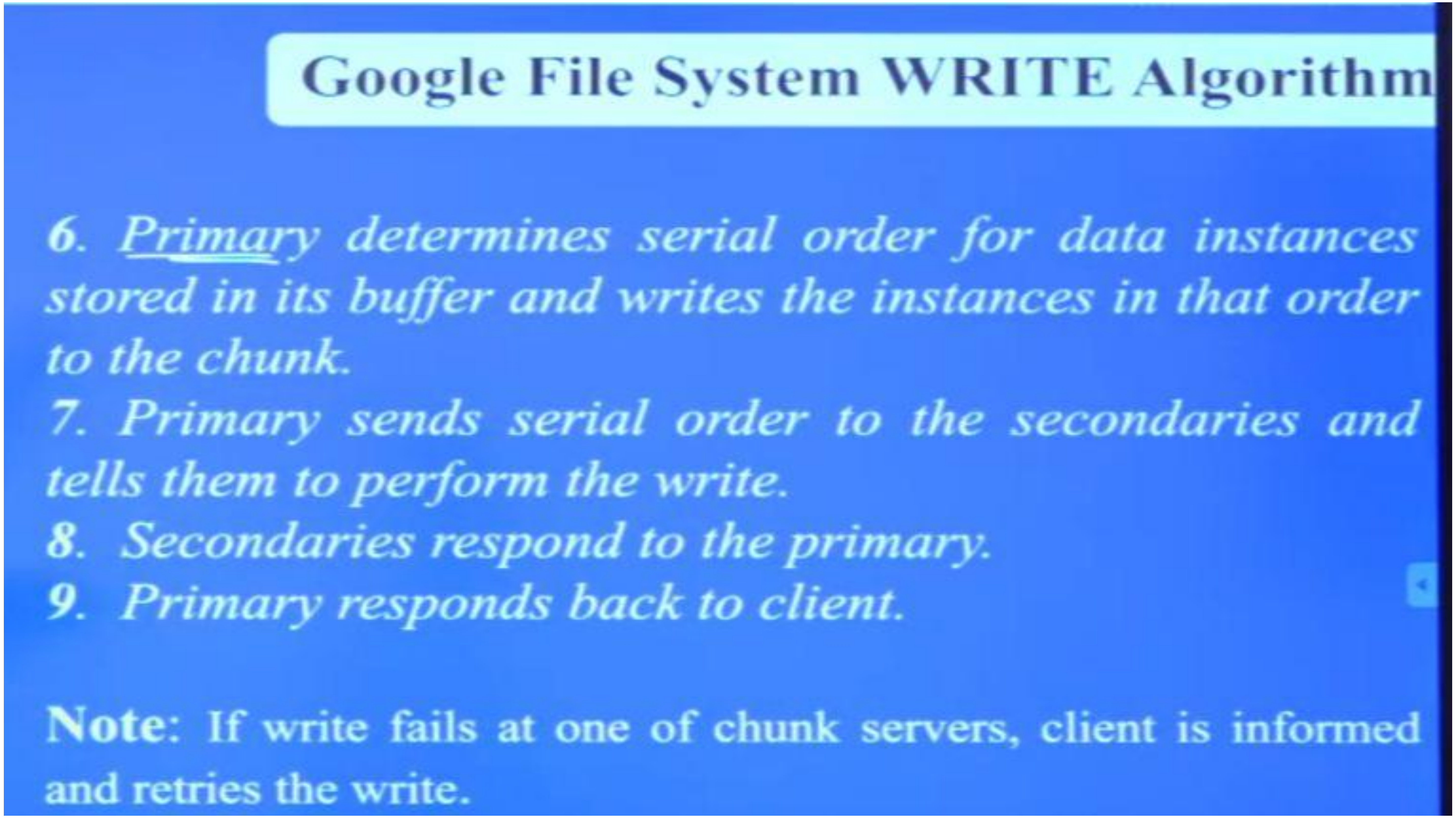
Google File System WRITE Operation Execution Flow





Google File System WRITE Algorith

1. Application originates write request.
2. GFS client translates request from (filename, data) -
(filename, chunk index), and sends it to master.
3. Master responds with chunk handle and (primary
secondary) replica locations.
4. Client pushes write data to all locations. Data
stored in 'chunk servers' internal buffers.
5. Client sends write command to primary.



Google File System WRITE Algorithm

6. Primary determines serial order for data instances stored in its buffer and writes the instances in that order to the chunk.

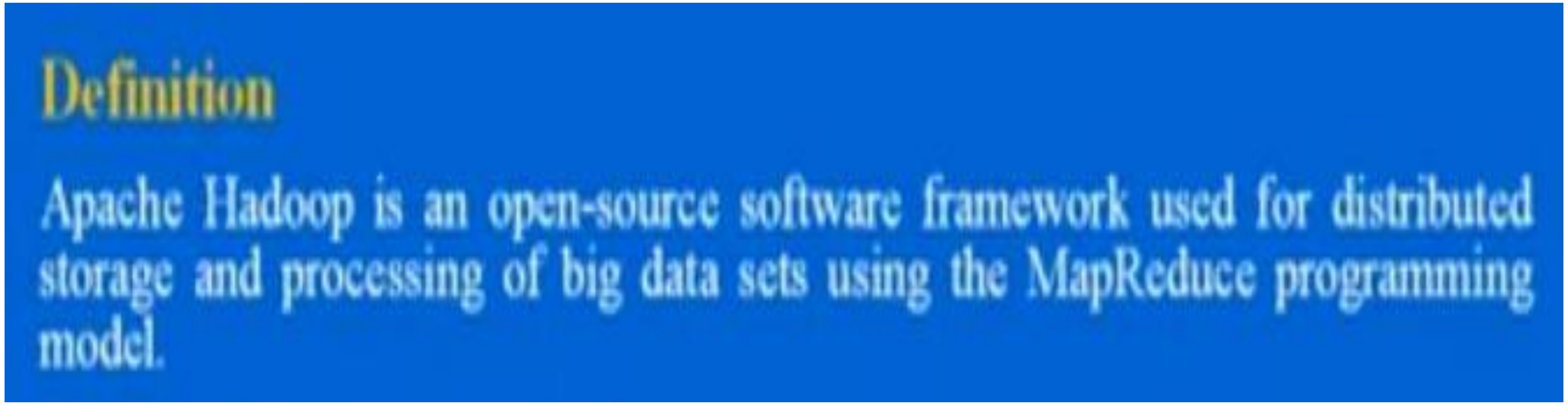
7. Primary sends serial order to the secondaries and tells them to perform the write.

8. Secondaries respond to the primary.

9. Primary responds back to client.

Note: If write fails at one of chunk servers, client is informed and retries the write.

HDFS(Hadoop Distributed File System)



Definition

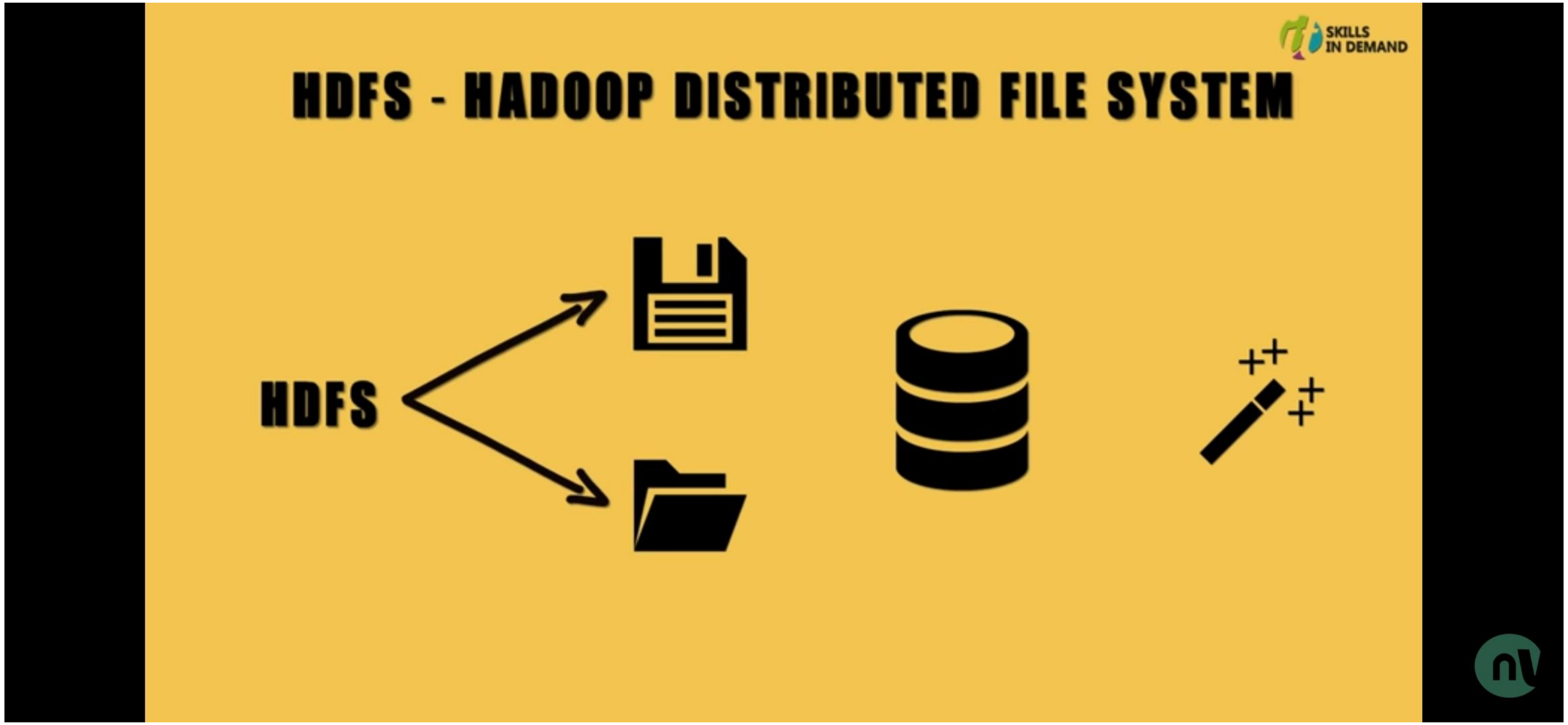
Apache Hadoop is an open-source software framework used for distributed storage and processing of big data sets using the MapReduce programming model.

GFS vs HDFS

HDFS	GFS
Cross Platform	Linux
Developed in Java environment	Developed in c,c++
It has Name node and Data Node	It has Master-node and Chunk server
Name node receive heartbeat from Data node	Master node receive heartbeat from Chunk server
Commodities hardware were used	Commodities hardware werused
WORM – Write Once and Read Many times	Multiple writer , multiple reader model
Deleted files are renamed into particular folder and then it will removed via garbage	Deleted files are not reclaimed immediately and are renamed in hidden name space and it will deleted after three days if it’s not in use

- **The base Apache Hadoop framework is composed of the following modules:**
- *Hadoop Common* – contains libraries and utilities needed by other Hadoop modules;
- *Hadoop Distributed File System (HDFS)* – a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster;
- *Hadoop YARN* – (introduced in 2012) a platform responsible for managing computing resources in clusters and using them for scheduling users' applications
- *Hadoop MapReduce* – an implementation of the MapReduce programming model for large-scale data processing.
- *Hadoop Ozone* – (introduced in 2020) An object store for Hadoop

HDFS



PROS

HDFS
HDFS
GB/TB
HDFS
LOG
HDFS

CONS

HDFS

HDFS

HDFS Architecture:

- HDFS is responsible for storing data on the cluster in Hadoop.
- Files in HDFS are split into blocks before they are stored on cluster of size 64MB or 128MB.
- On a fully configured cluster, —running Hadoop‖ means running a set of daemons, or resident programs, on the different servers in your network.
- Programs which reside permanently in memory are called —Resident Programs.
- Daemon is a thread in Java, which runs in background and mostly created by JVM for performing background task like Garbage collection.
- Each daemon runs separately in its own JVM.
- These daemons have specific roles; some exist only on one server, some exist across multiple servers.

Cont..

The daemons include :

- NameNode
- DataNode
- Secondary NameNode
- JobTracker
- TaskTracker

The above daemons are called as —Building Blocks of Hadoop

NameNode .

- Hadoop employs a master/slave architecture for both distributed storage and distributed computation.
- The distributed storage system is called the Hadoop File System , or HDFS.
- The NameNode is the master of HDFS that directs the slave DataNode daemons to perform the low-level I/O tasks.
- The NameNode is the bookkeeper of HDFS; it keeps track of how your files are broken down into file blocks, which nodes store those blocks, and the overall health of the distributed filesystem.
- The function of the NameNode is memory and I/O intensive. As such, the server hosting the NameNode typically doesn‘t store any user data or perform any computations for a MapReduce program to lower the workload on the machine.
- This means that the NameNode server doesn‘t double as a DataNode or a TaskTracker.
- There is unfortunately a negative aspect to the importance of the NameNode—it‘s a single point of failure of your Hadoop cluster.
- For any of the other daemons, if their host nodes fail for software or hardware reasons, the Hadoop cluster will likely continue to function smoothly or you can quickly restart it. Not so for the NameNode.

DataNode:

- Each slave machine in your cluster will host a DataNode daemon to perform the grunt (thankless and menial) work of the distributed filesystem—reading and writing HDFS blocks to actual files on the local filesystem.
- When you want to read or write a HDFS file, the file is broken into blocks and the NameNode will tell your client which DataNode each block resides in.
- Your client communicates directly with the DataNode daemons to process the local files corresponding to the blocks.
- Furthermore, a DataNode may communicate with other DataNodes to replicate its data blocks for redundancy.

Data Node Cont..

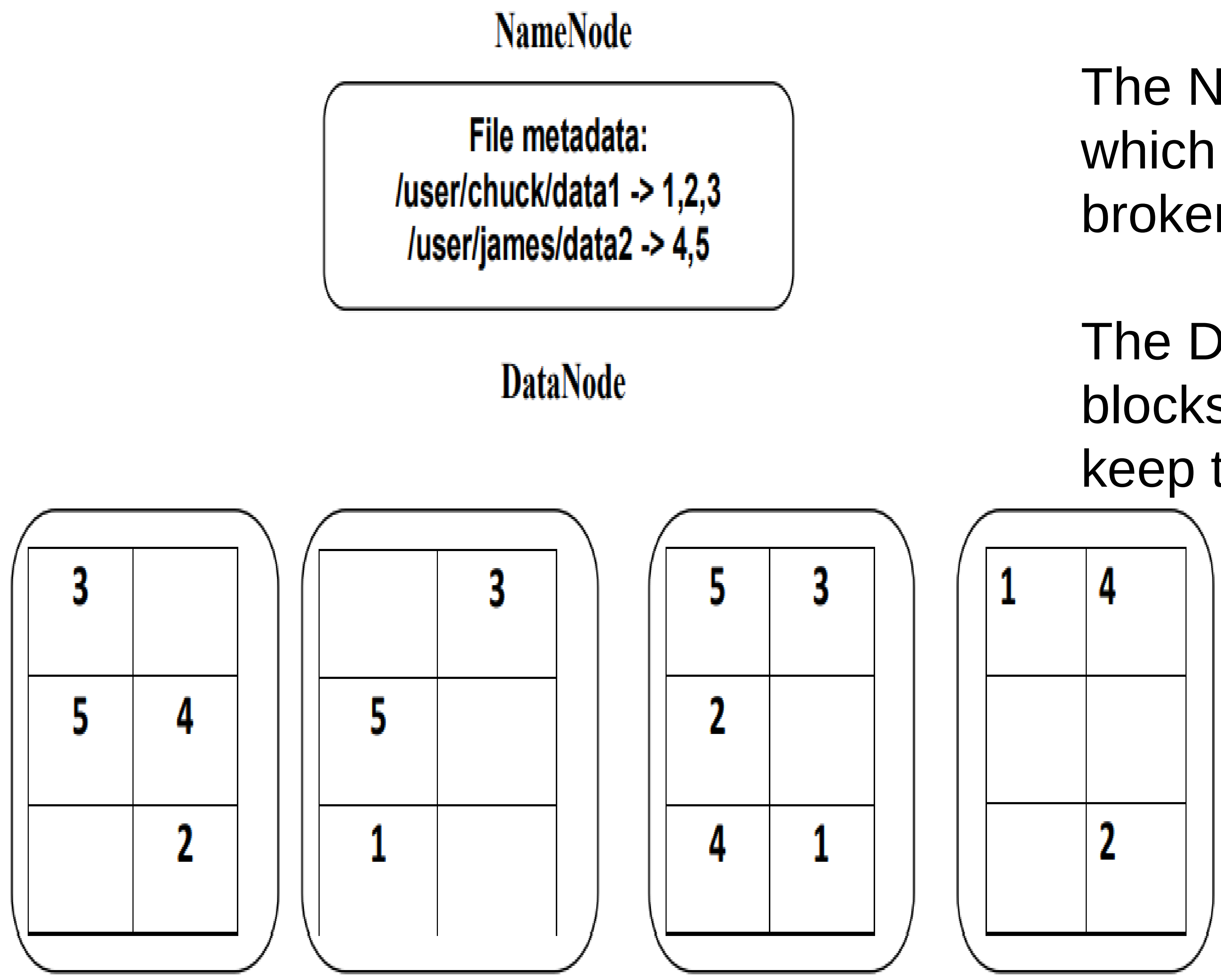
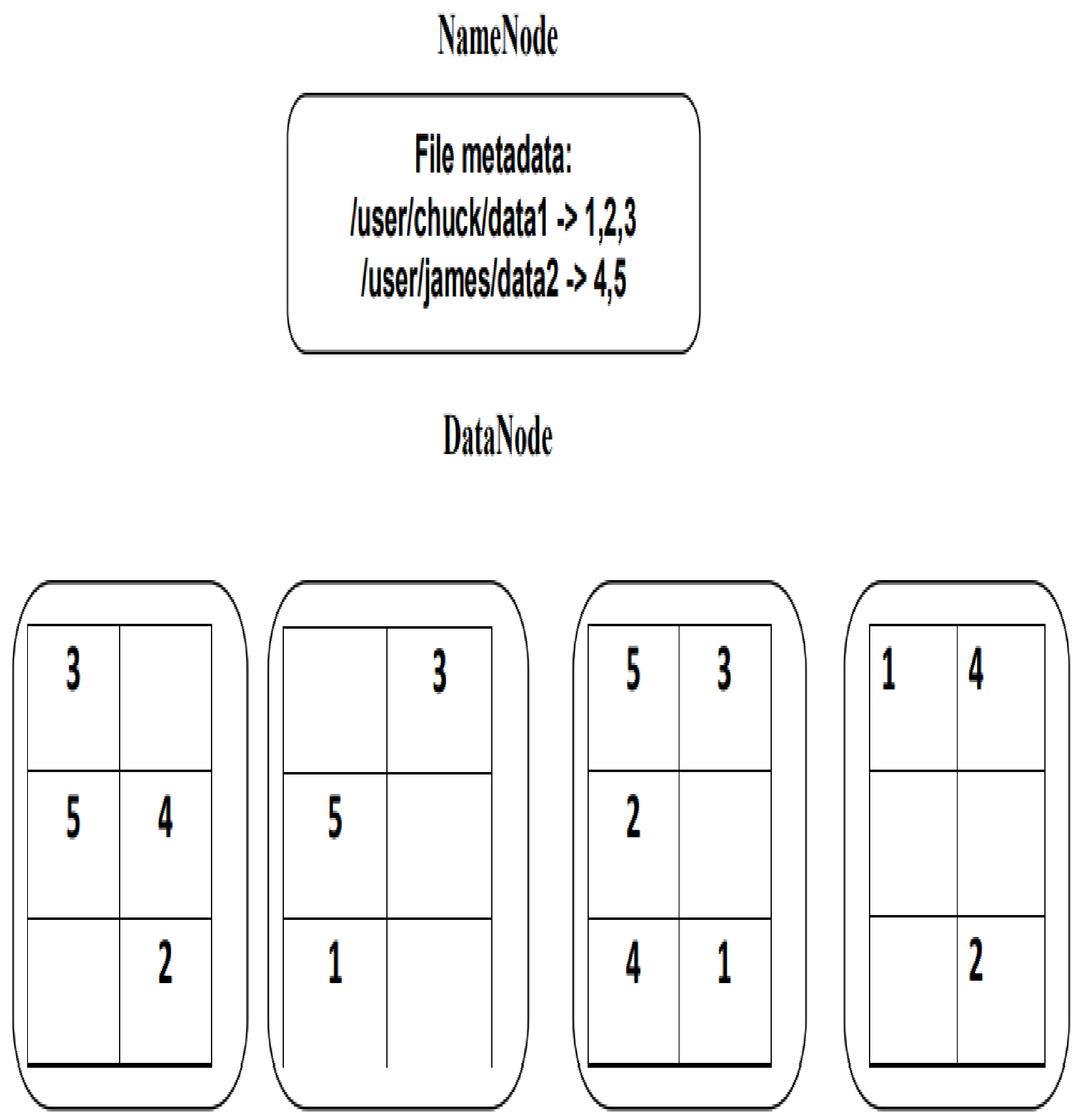


Figure : NameNode /DataNode interaction in HDFS.

The NameNode keeps track of the file metadata—which files are in the system and how each file is broken down into blocks.

The DataNodes provide backup store of the blocks and constantly report to the NameNode to keep the metadata current.

Data Node Cont..



The figure illustrates the roles of the NameNode and DataNodes.
In this figure, we show two data files, one at /user/chuck/data1 and another at /user/james/data2.
The data1 file takes up three blocks, which we denote 1, 2, and 3, and the data2 file consists of blocks 4 and 5. The content of the files are distributed among the DataNodes.

In this illustration, each block has three replicas. For example, block 1 (used for data1) is replicated over the three rightmost DataNodes. This ensures that if any one DataNode crashes or becomes inaccessible over the network, you‘ll still be able to read the files. DataNodes are constantly reporting to the NameNode.

Upon initialization, each of the DataNodes informs the NameNode of the blocks it‘s currently storing. After this mapping is complete, the DataNodes continually poll the NameNode to provide information regarding local changes as well as receive instructions to create, move, or delete blocks from the local disk.

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. Nevertheless, a NameNode failure requires human intervention to reconfigure the cluster to use the SNN as the primary NameNode

JobTracker:

- The JobTracker daemon is the liaison (communication/cooperation which facilitates a close working) between your application and Hadoop.
- Once you submit your code to your cluster, the JobTracker determines the execution plan by determining which files to process, assigns nodes to different tasks, and monitors all tasks as they‘re running.
- Should a task fail, the JobTracker will automatically relaunch the task, possibly on a different node, up to a predefined limit of retries.
- There is only one JobTracker daemon per Hadoop cluster. It‘s typically run on a server as a master node of the cluster

TaskTracker:

- As with the storage daemons, the computing daemons also follow a master/slave architecture: the JobTracker is the master overseeing the overall execution of a MapReduce job and the TaskTracker manage the execution of individual tasks on each slave node.

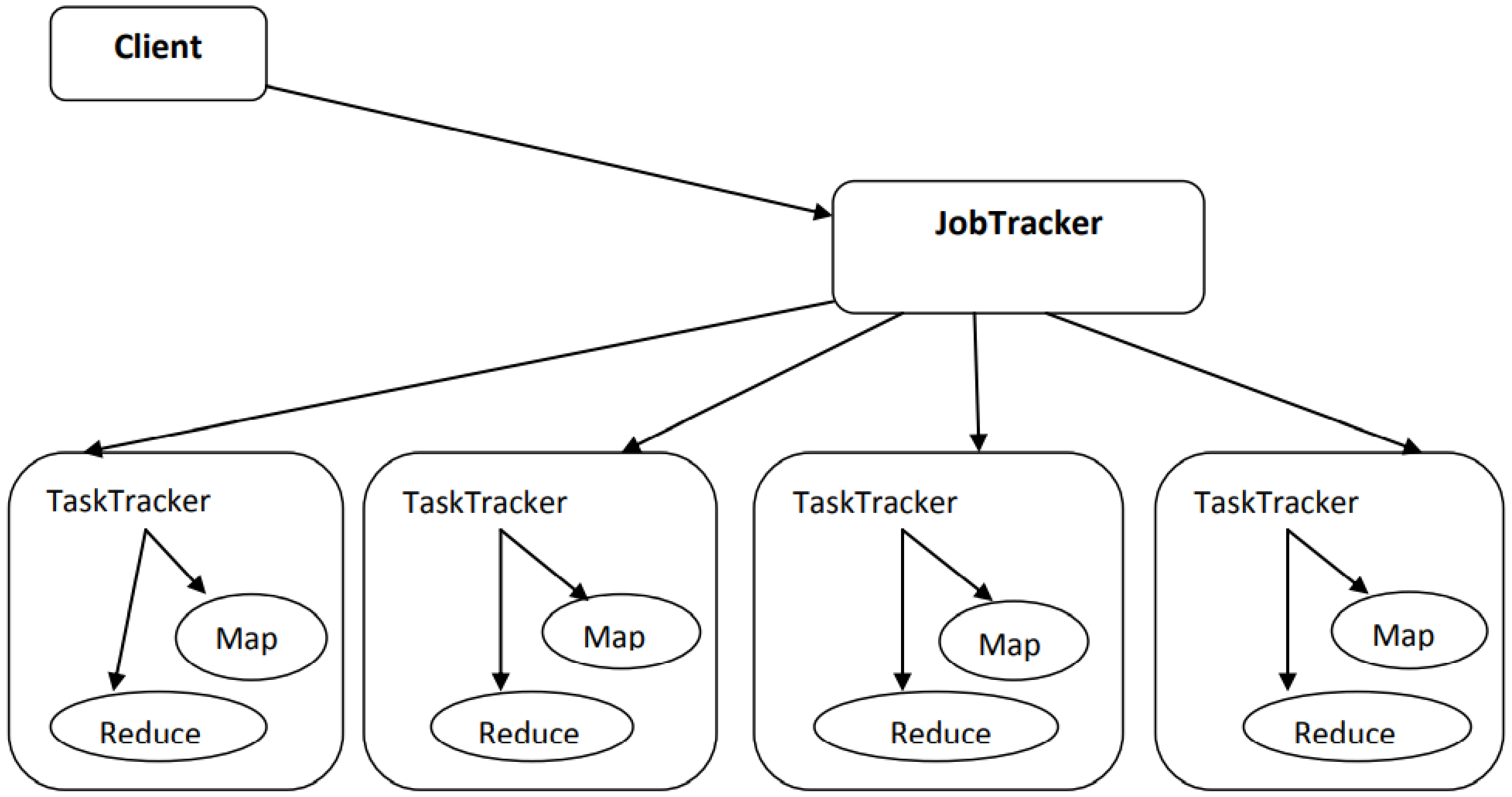


Figure: JobTracker and TaskTracker interaction. After a client calls the JobTracker to begin a data processing job, the JobTracker partitions the work and assigns different map and reduce tasks to each TaskTracker in the cluster.

TaskTracker:

- The above figure illustrates this interaction.
- Each TaskTracker is responsible for executing the individual tasks that the JobTracker assigns.
- Although there is a single TaskTracker per slave node, each TaskTracker can spawn multiple JVMs to handle many map or reduce tasks in parallel.
- One responsibility of the TaskTracker is to constantly communicate with the JobTracker.
- If the JobTracker fails to receive a heartbeat from a TaskTracker within a specified amount of time, it will assume the TaskTracker has crashed and will resubmit the corresponding tasks to other nodes in the cluster. .

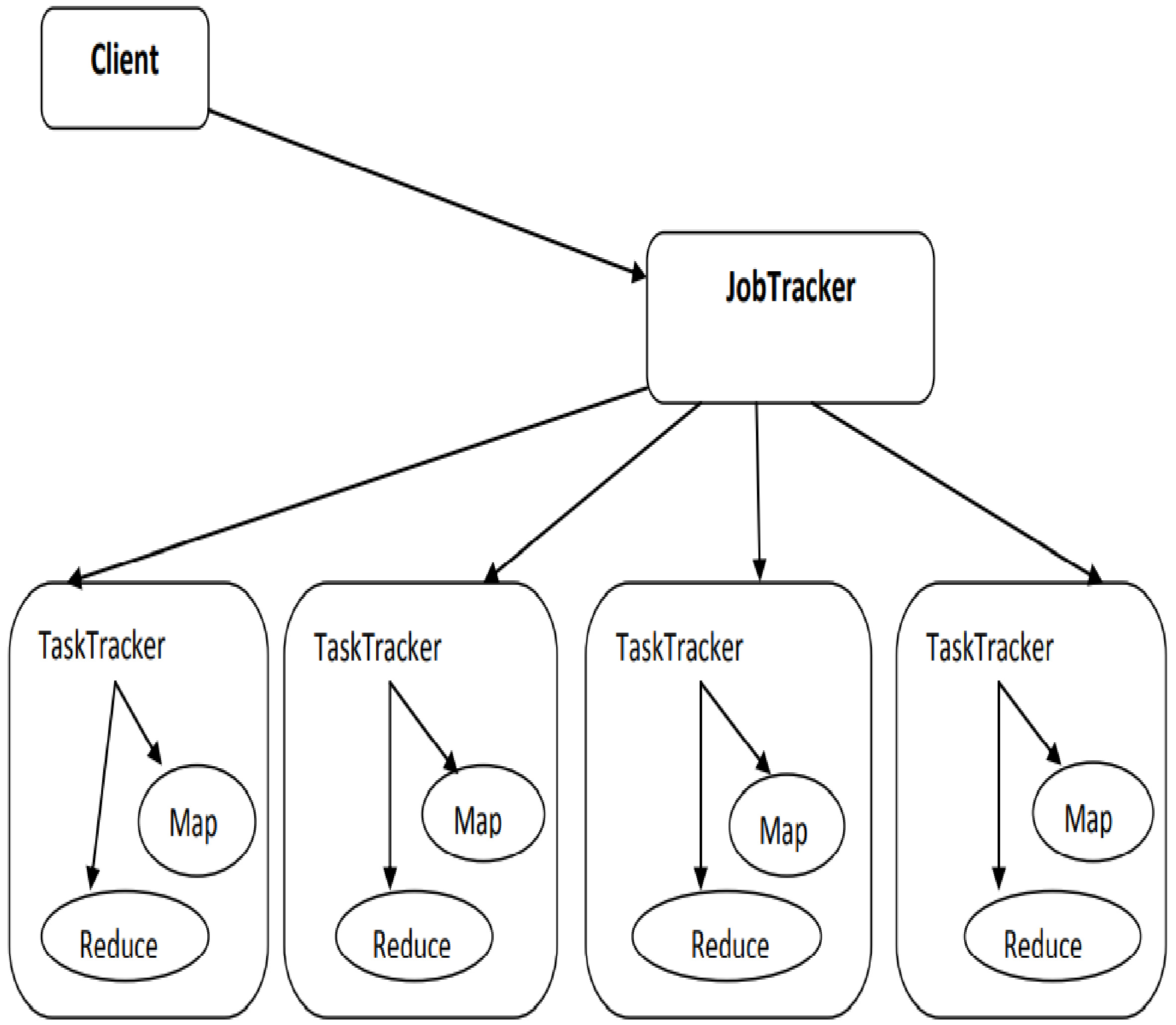


Figure: JobTracker and TaskTracker interaction. After a client calls the JobTracker to begin a data processing job, the JobTracker partitions the work and assigns different map and reduce tasks to each TaskTracker in the cluster.

Advantages of HDFS

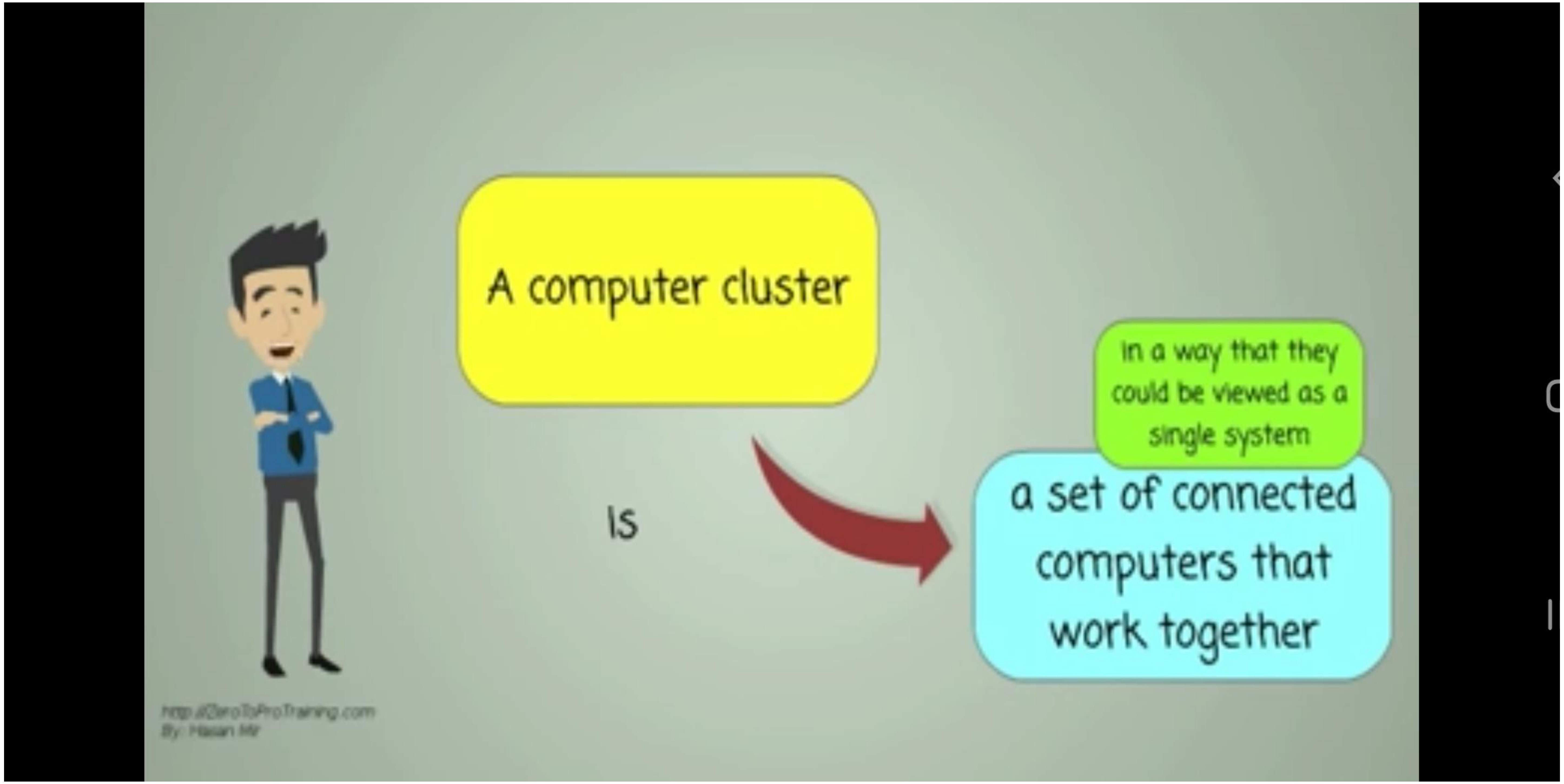
- Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores.
- Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer.
- Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption.
- Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

Cont..

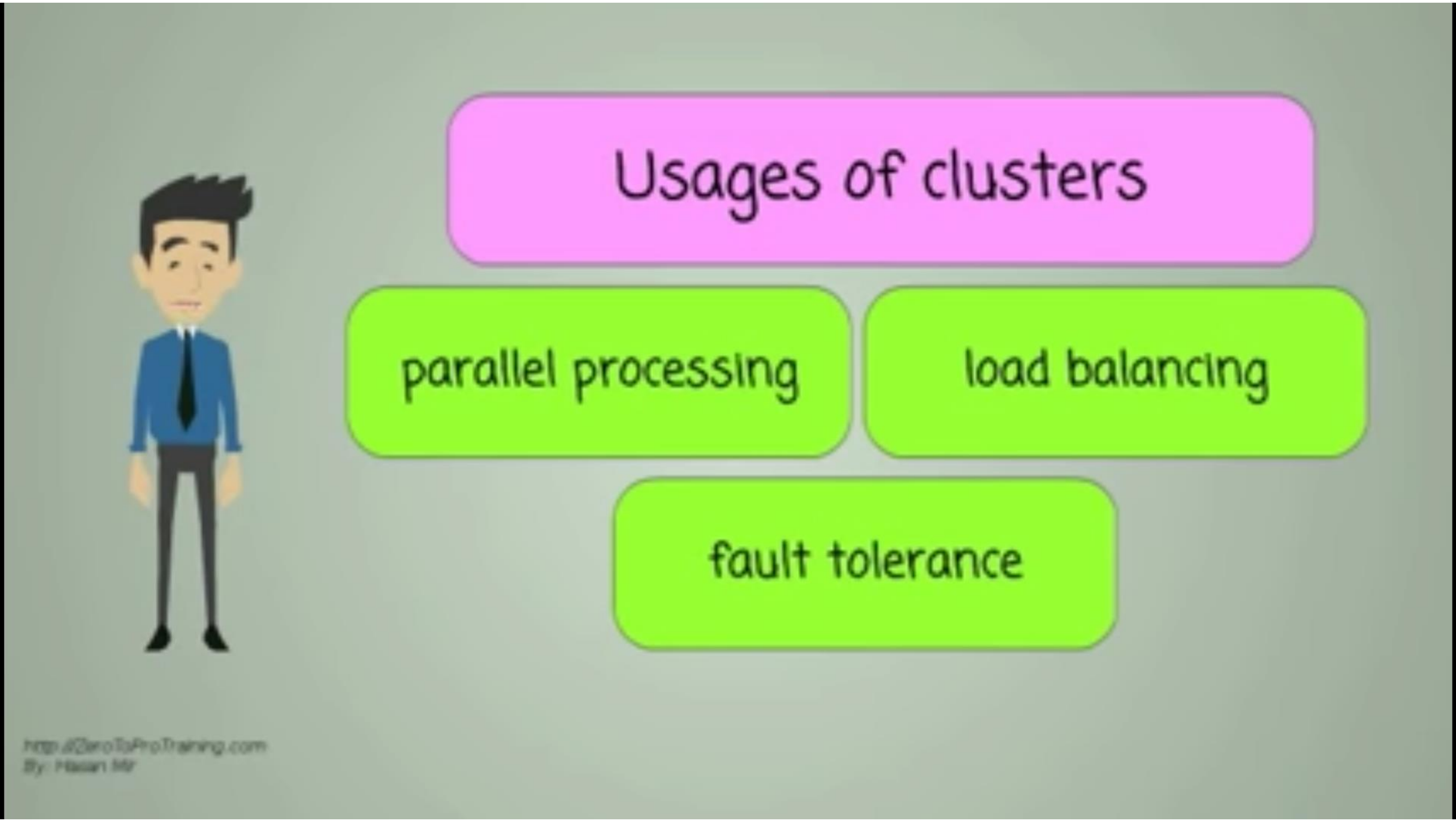
- Hadoop File System was developed using distributed file system design.
- It is run on commodity hardware.
- Unlike other distributed systems, HDFS is highly faulttolerant and designed using low-cost hardware.
- HDFS holds very large amount of data and provides easier access.
- To store such huge data, the files are stored across multiple machines.
- These files are stored in redundant fashion to rescue the system from possible data losses in case of failure.
- HDFS also makes applications available to parallel processing

- It is suitable for the distributed storage and processing.
- Hadoop provides a command interface to interact with HDFS.
- The built-in servers of NameNode and DataNode help users to easily check the status of cluster.
- Streaming access to file system data. HDFS provides file permissions and authentication

CLUSTER


A computer cluster
is
a set of connected computers that work together
in a way that they could be viewed as a single system
http://ZeroToProTraining.com
By: Hasan Mir

It relatively easy to add more power to the cluster



Usages of clusters

parallel processing

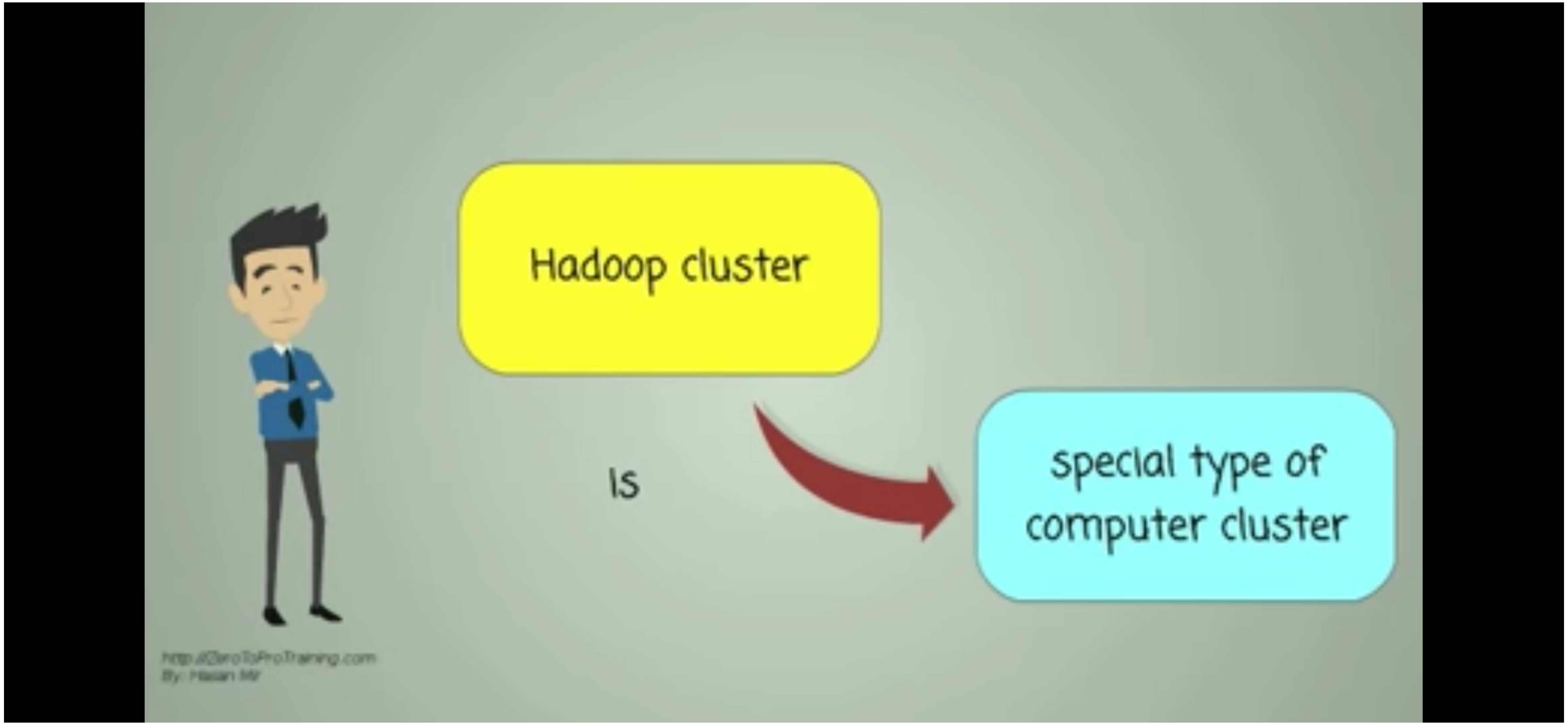
load balancing

fault tolerance

http://ZeroToProTraining.com
By: Hasan Mir

What is Hadoop cluster?

http://ZeroToProTraining.com
By: Hasan Mir



Hadoop cluster

is

special type of computer cluster

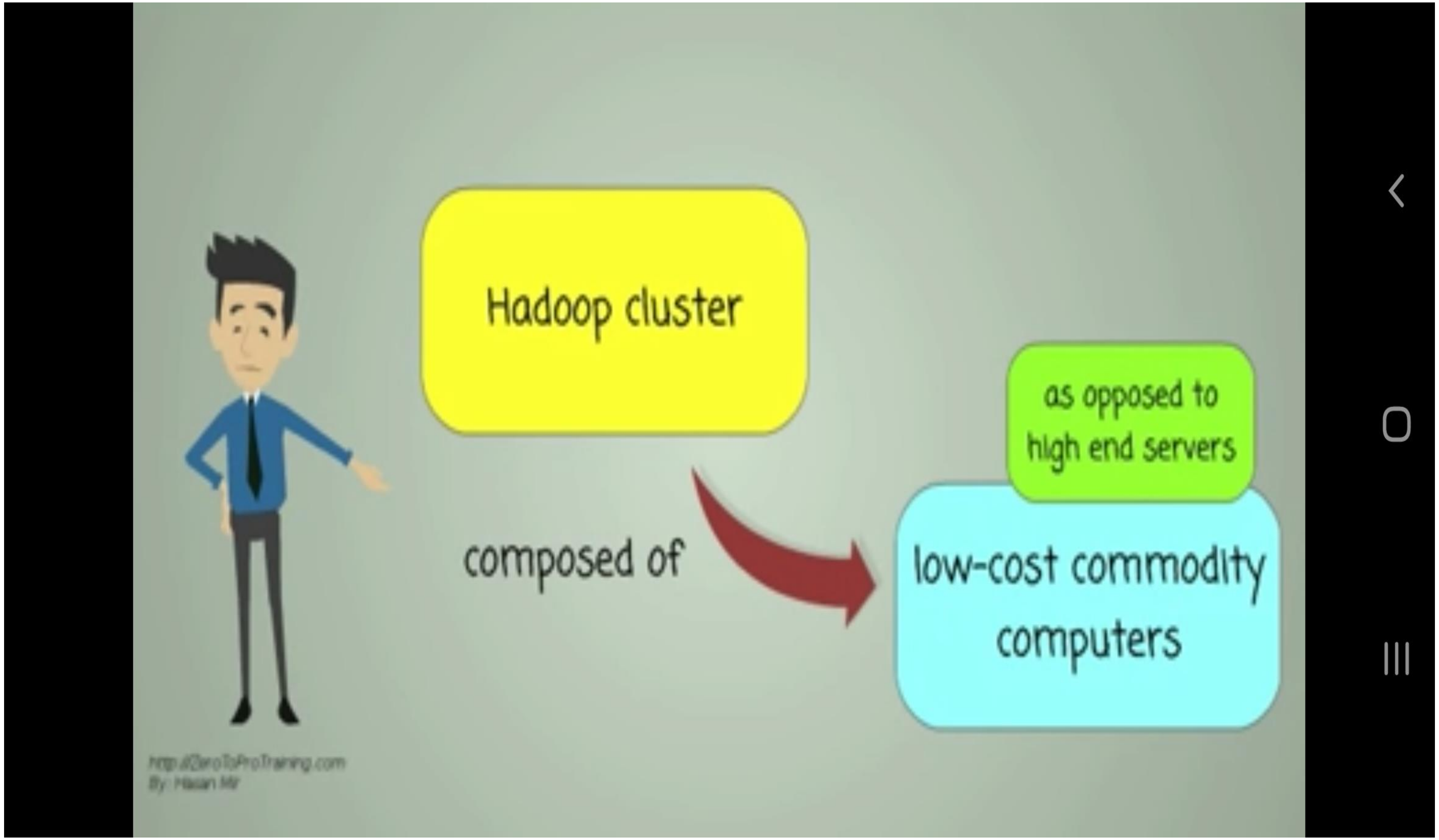
to the network

Add a new computer

In order to

add power to the cluster

http://ZeroToProTraining.com
By: Hasan Mir

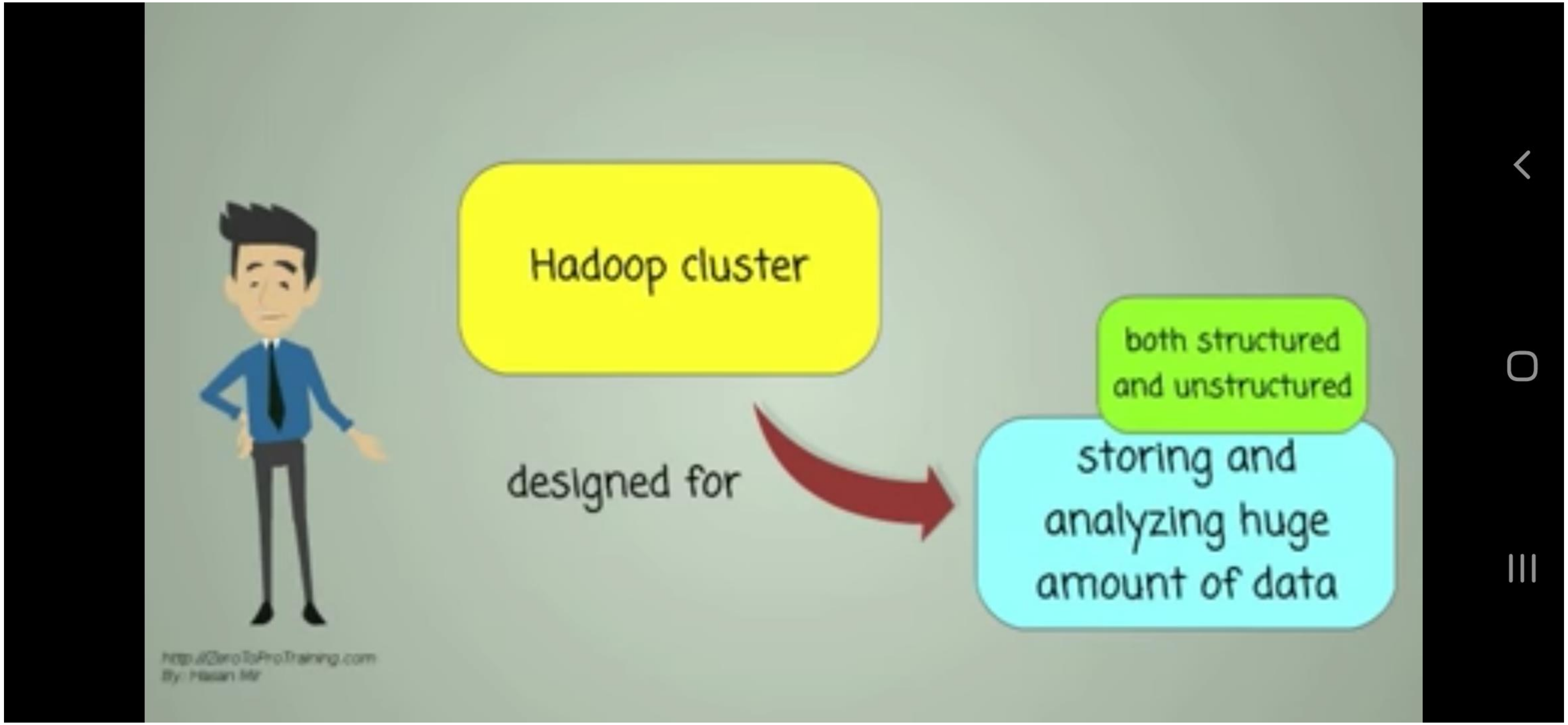

Hadoop cluster

composed of

low-cost commodity computers

as opposed to high end servers

http://ZeroToProTraining.com
By: Hasan Mir



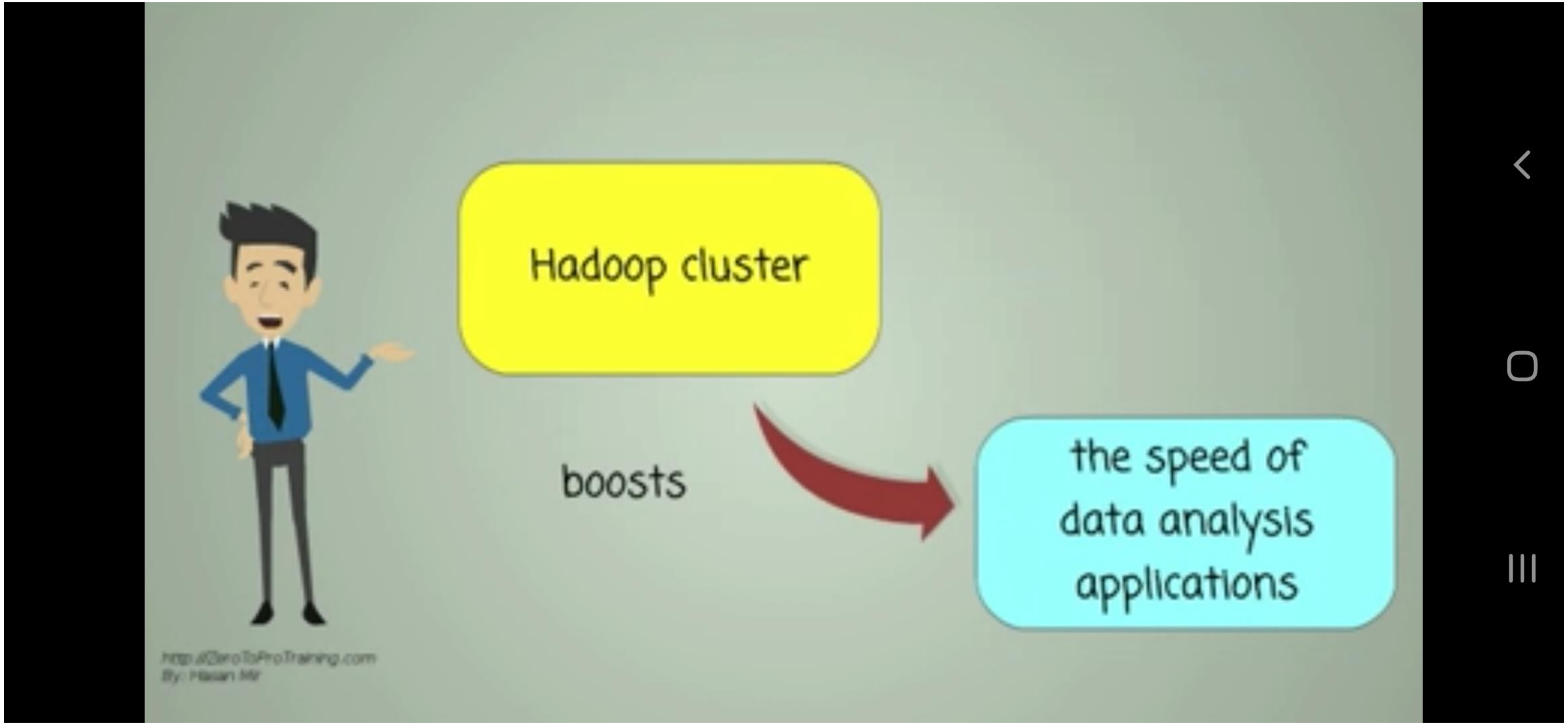
Hadoop cluster

designed for

both structured and unstructured

storing and analyzing huge amount of data

http://ZeroToProTraining.com
By: Hasan Mir

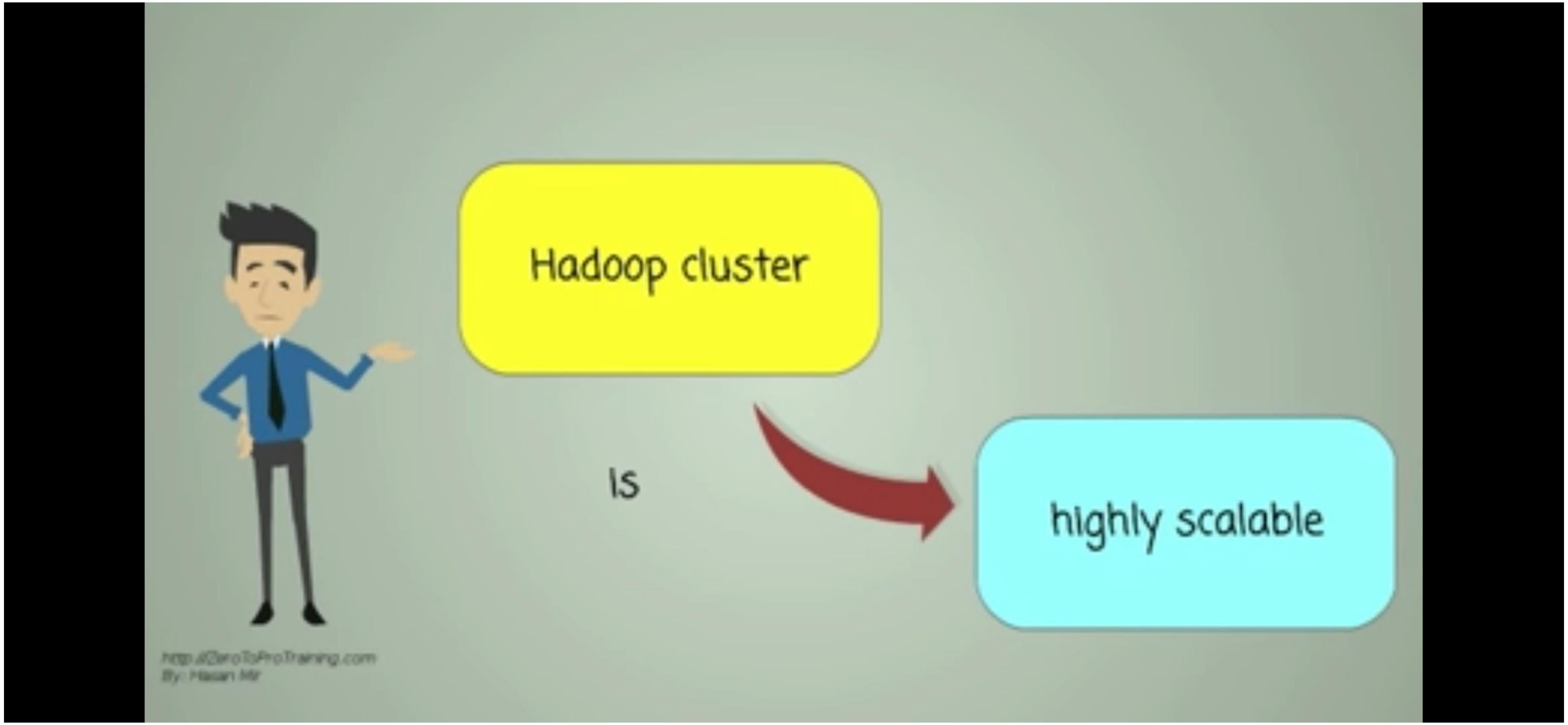


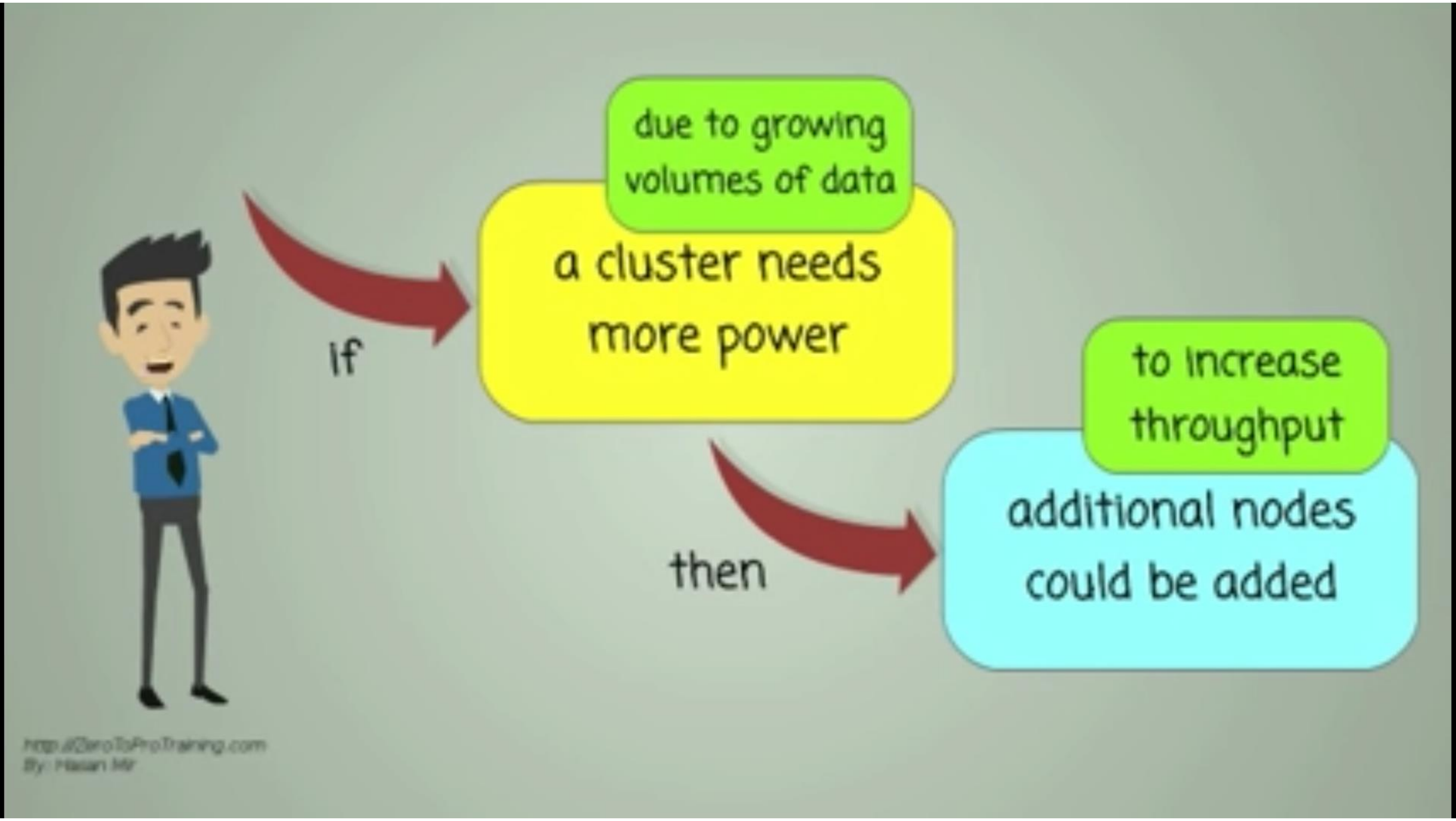
Hadoop cluster

boosts

the speed of data analysis applications

http://ZeroToProTraining.com
By: Hasan Mir


Hadoop cluster
is
highly scalable
http://ZeroToProTraining.com
By: Hasan Mir



due to growing volumes of data

if

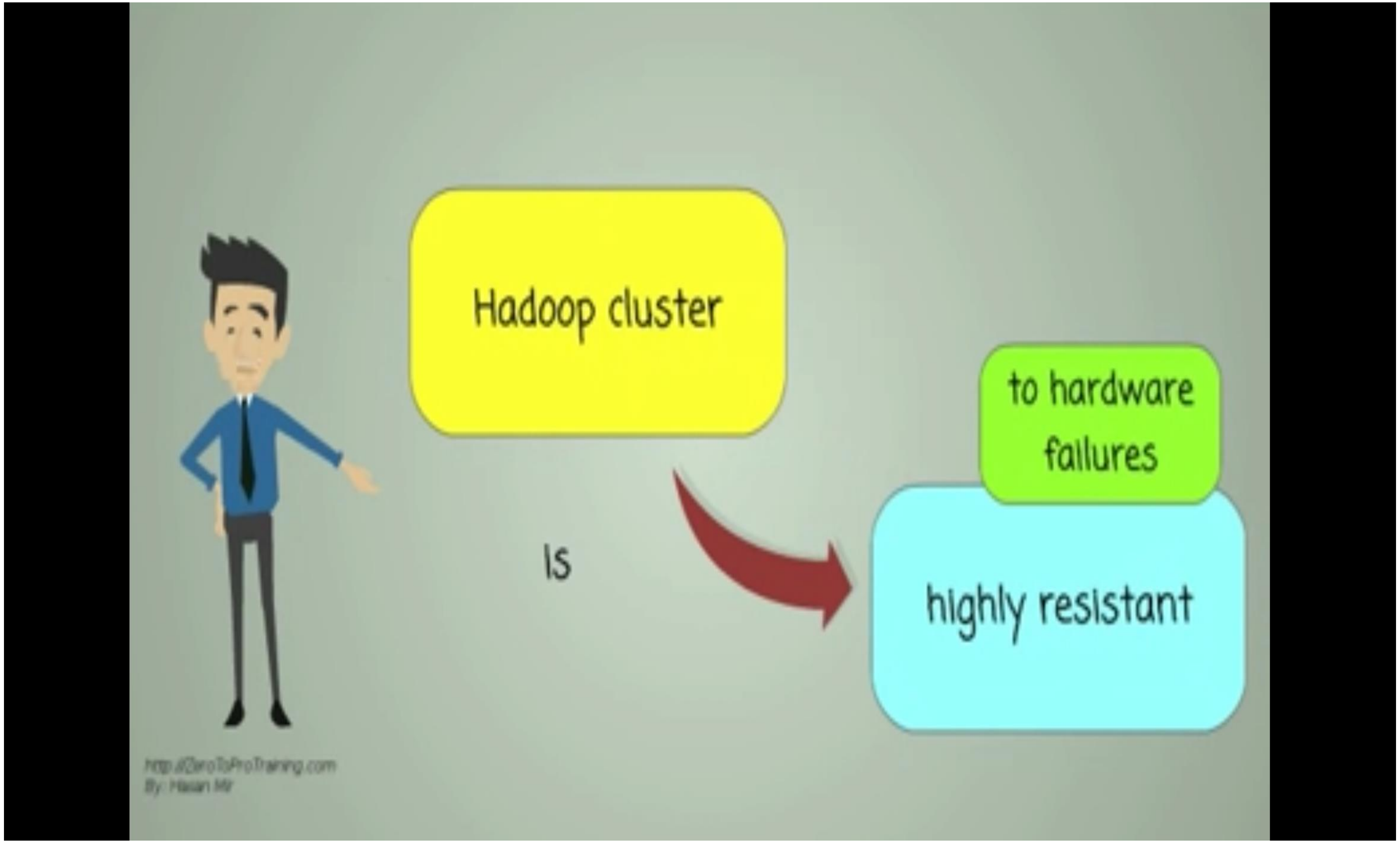
a cluster needs more power

then

additional nodes could be added

to increase throughput

http://ZeroToProTraining.com
By: Hasan Mir



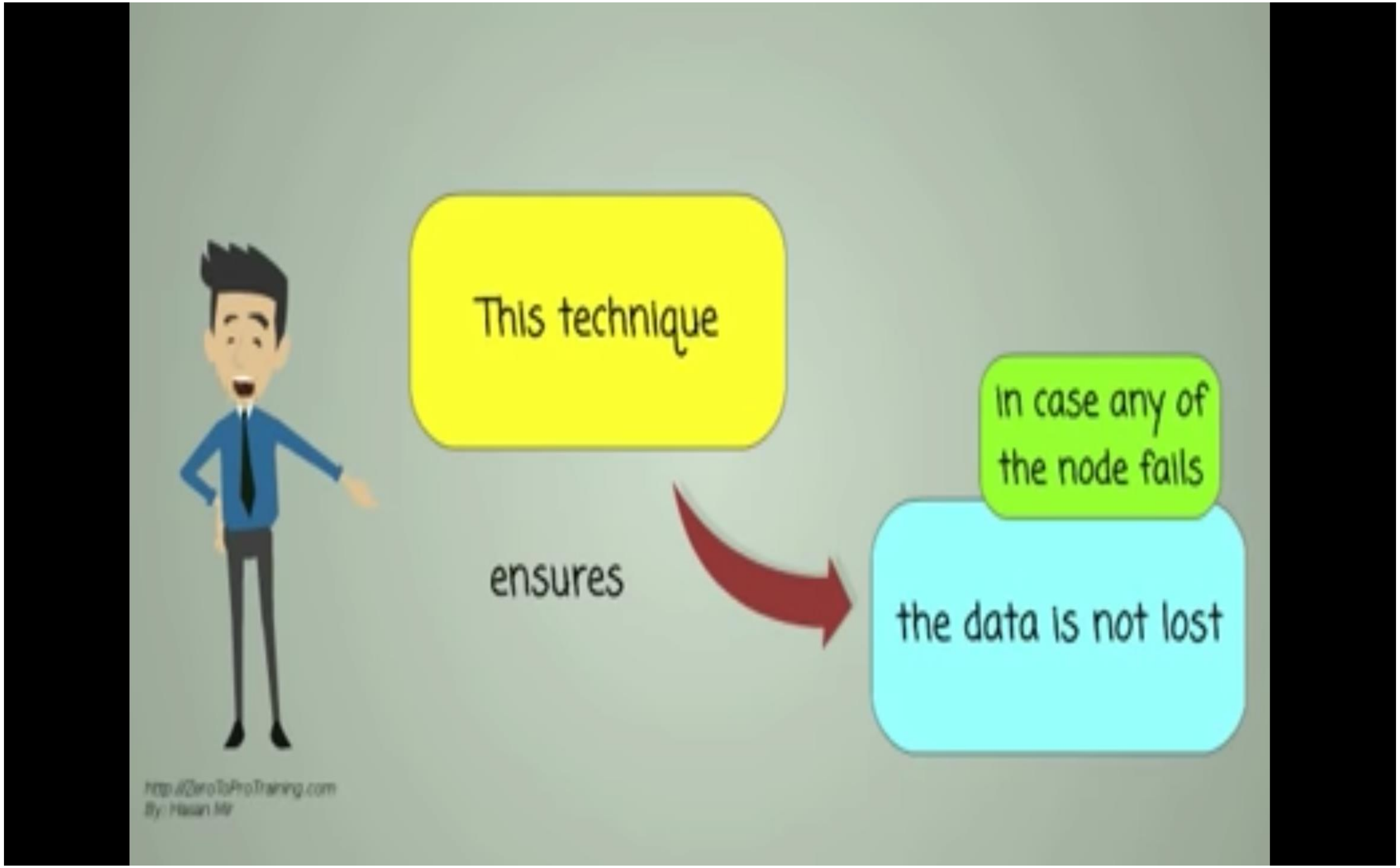
Hadoop cluster

is

highly resistant

to hardware failures

http://ZeroToProTraining.com
By: Hasan Mir



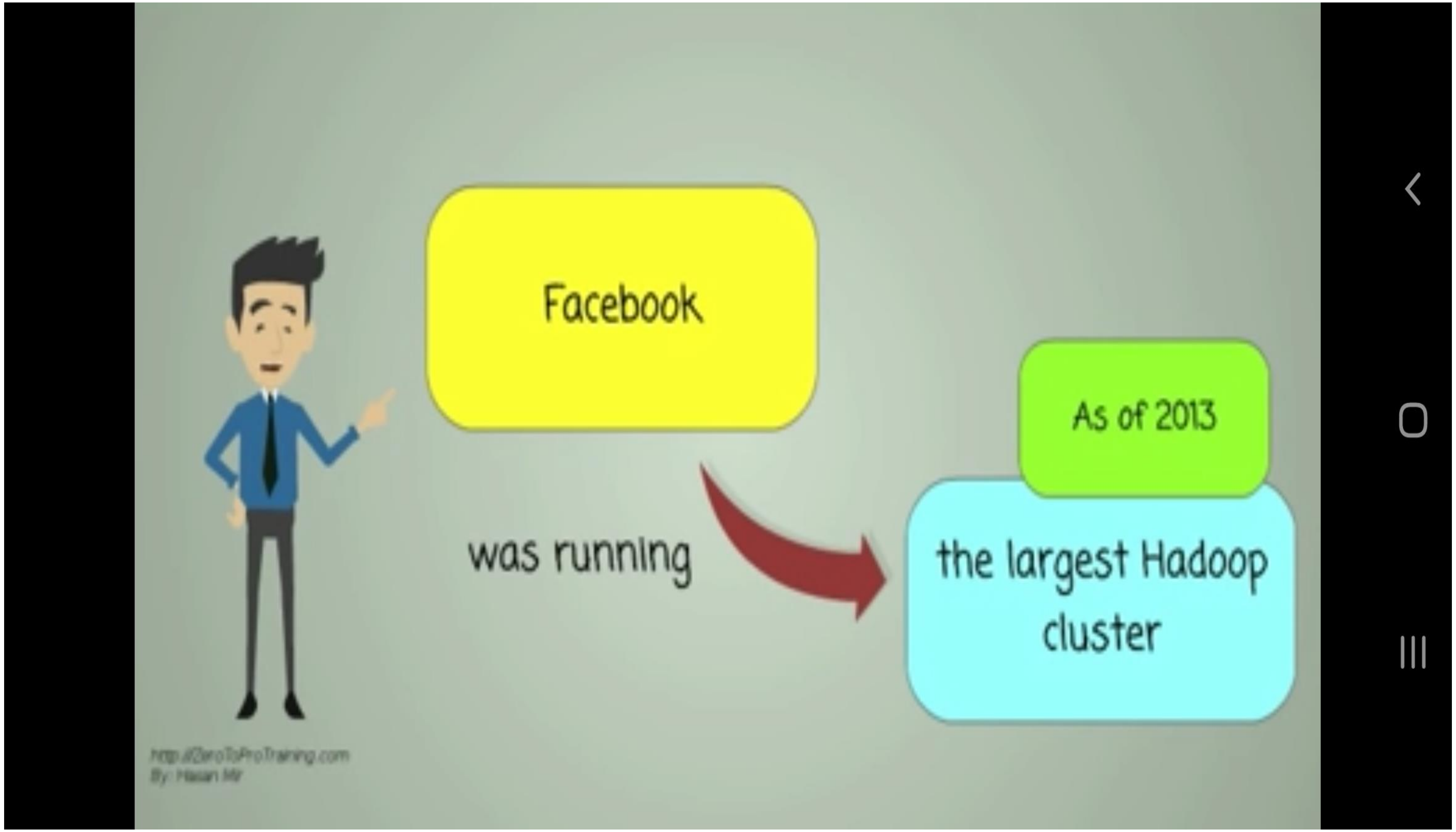
This technique

ensures

the data is not lost

in case any of the node fails

http://ZeroToProTraining.com
By: Hasan Mir

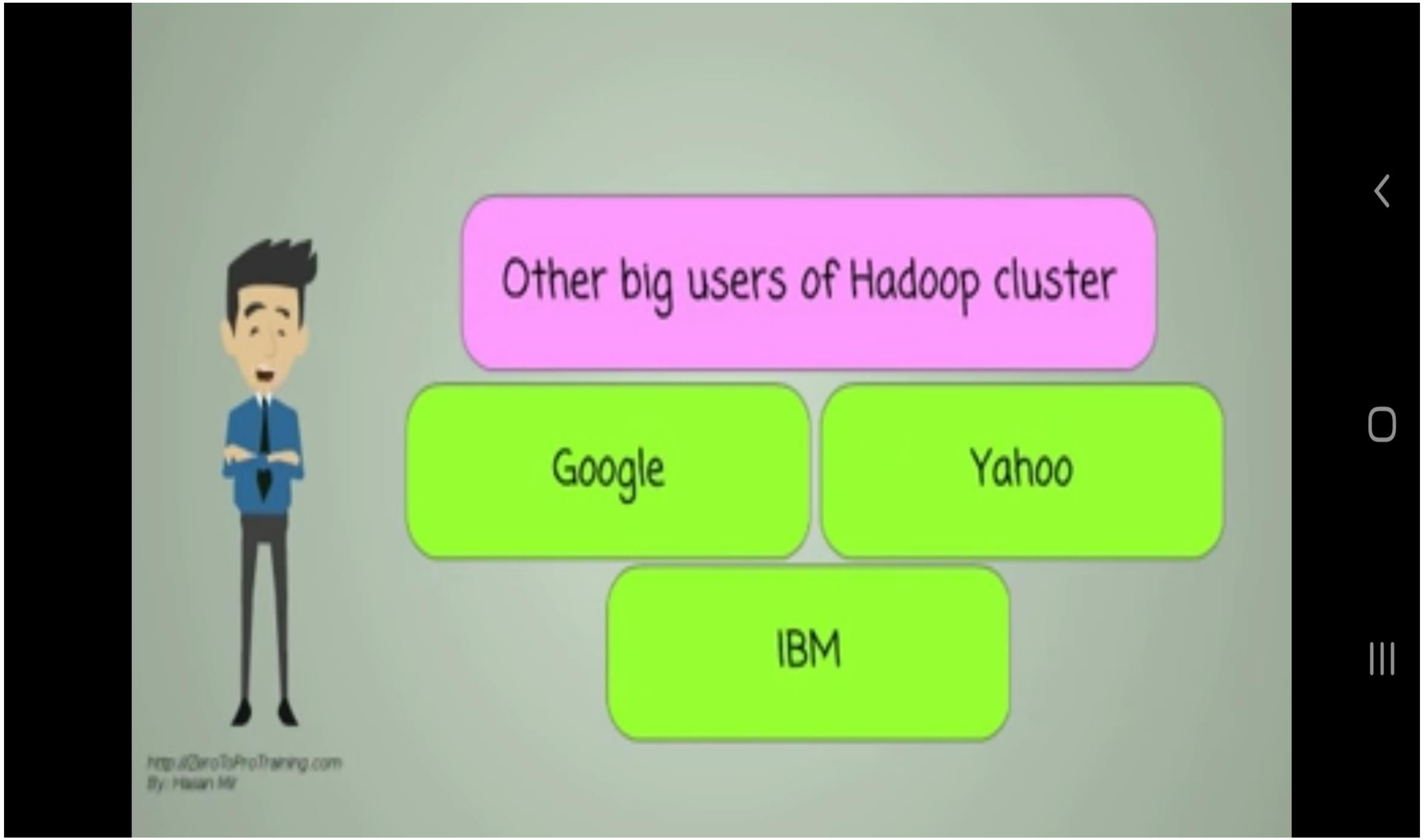


Facebook

was running

As of 2013

the largest Hadoop cluster



Other big users of Hadoop cluster

Google

Yahoo

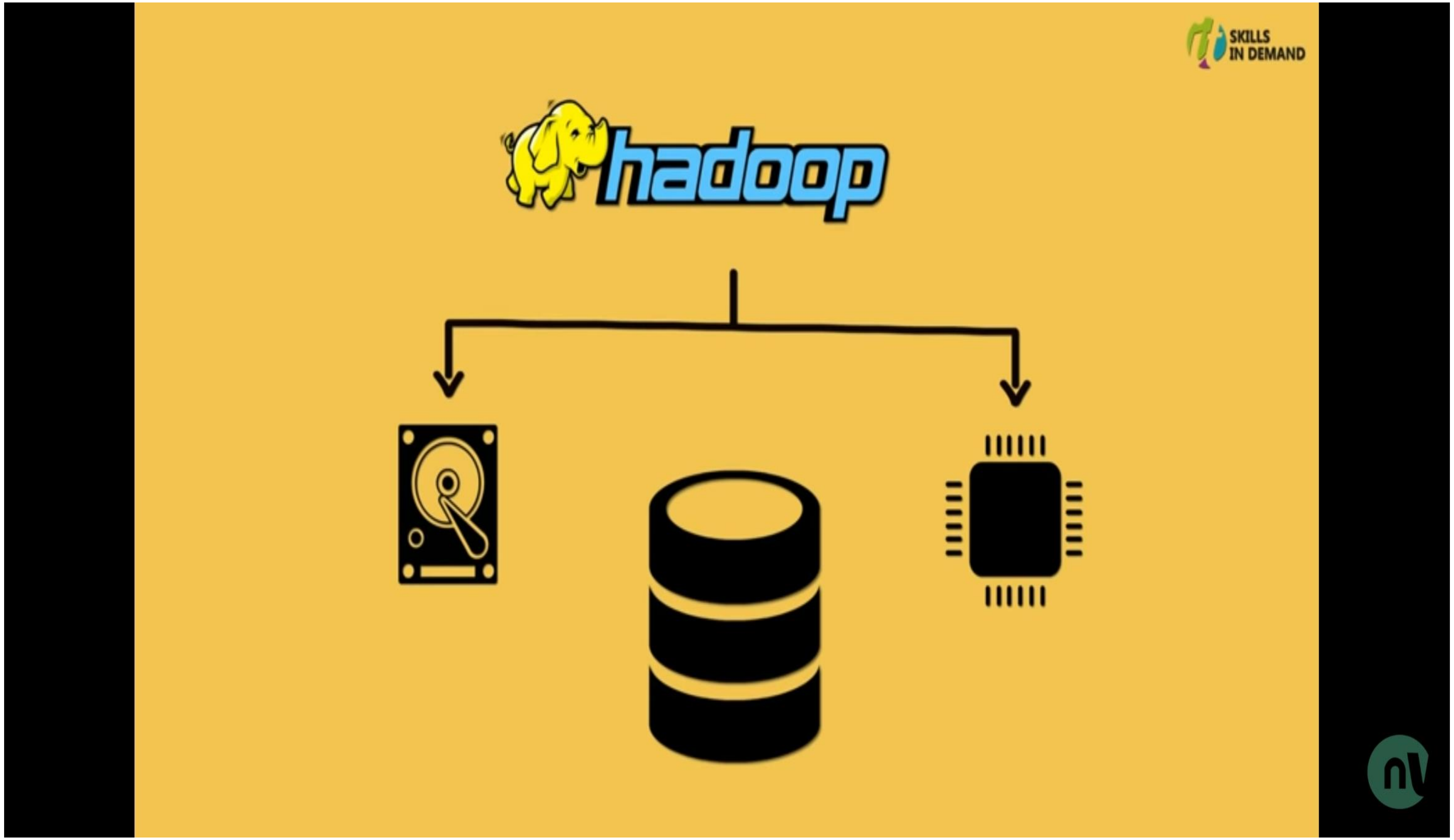
IBM

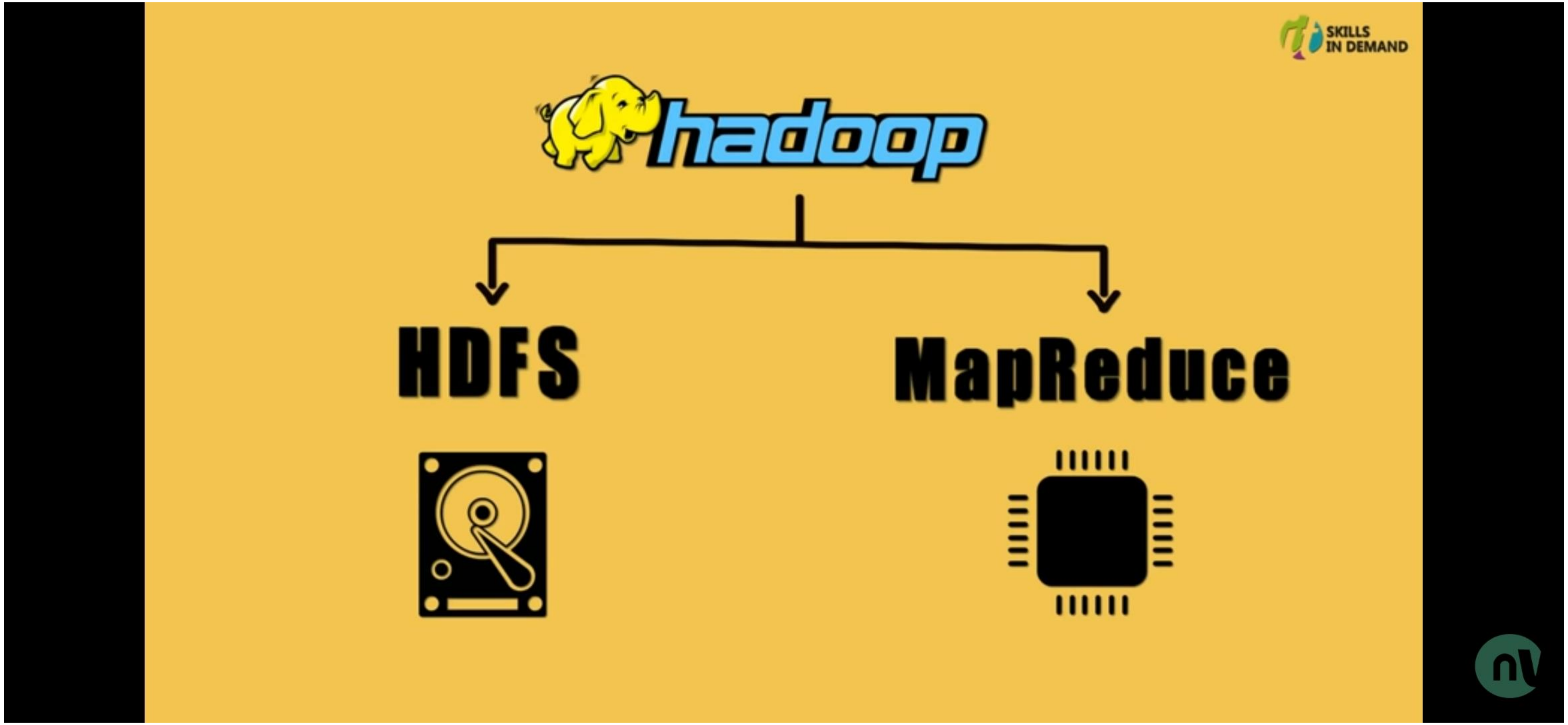
SKILLS IN DEMAND

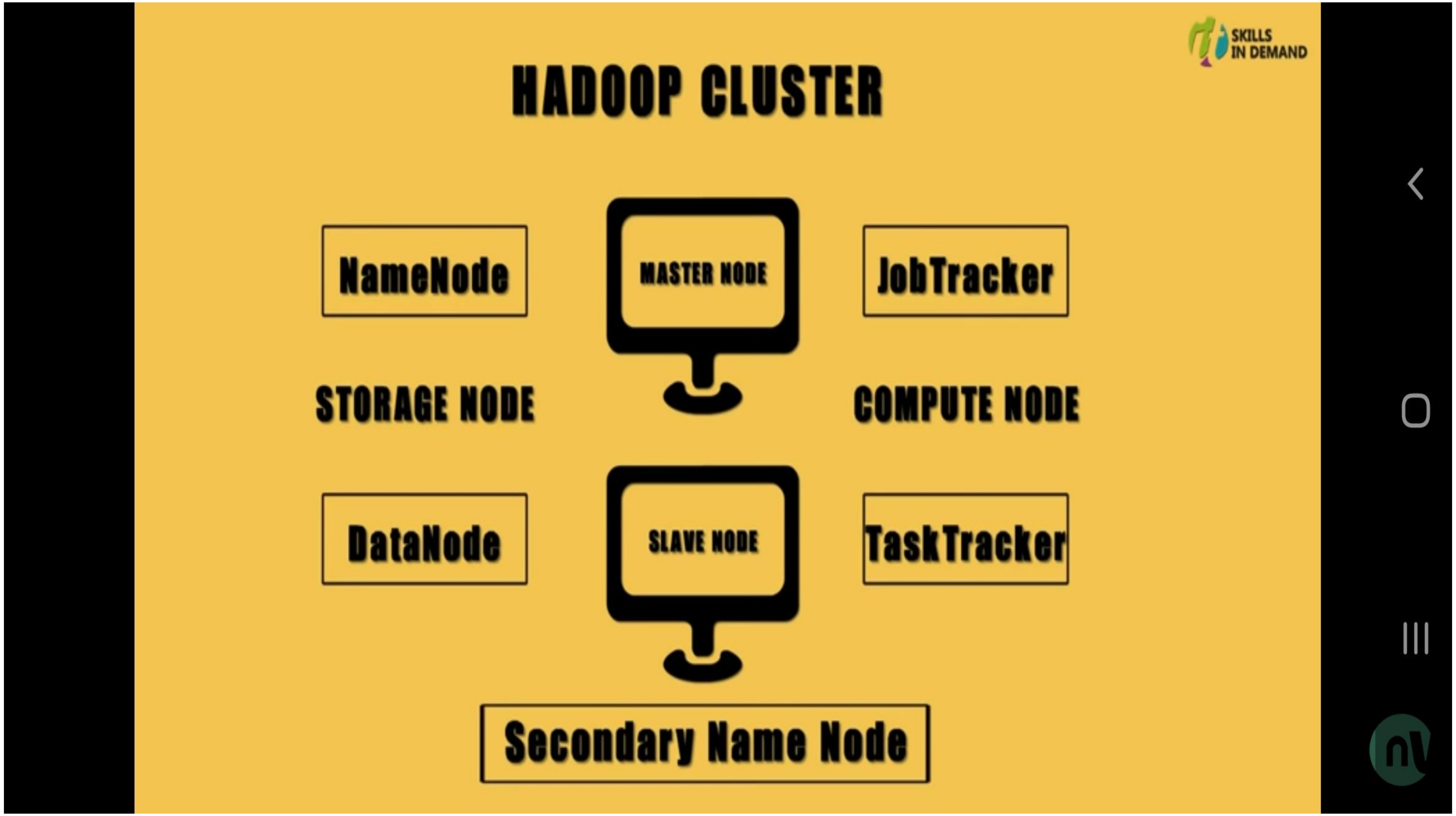
hadoop

open source™

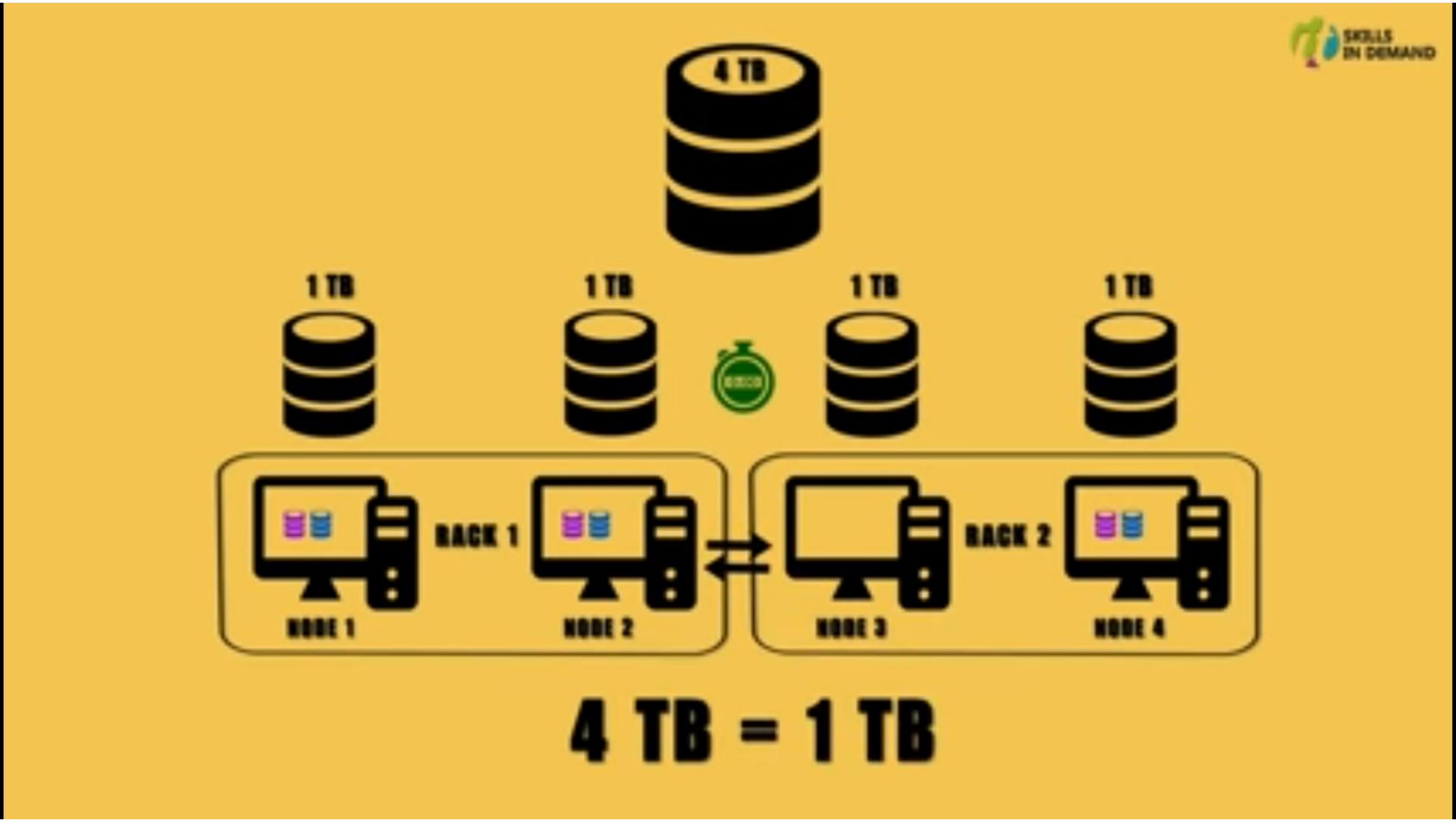
The Apache Software Foundation
http://www.apache.org/


SKILLS IN DEMAND
hadoop


SKILLS IN DEMAND
hadoop
HDFS
MapReduce



HADOOP CLUSTER


SKILLS IN DEMAND
4 TB
1 TB
1 TB
1 TB
1 TB
RACK 1
RACK 2
NODE 1
NODE 2
NODE 3
NODE 4

4 TB = 1 TB

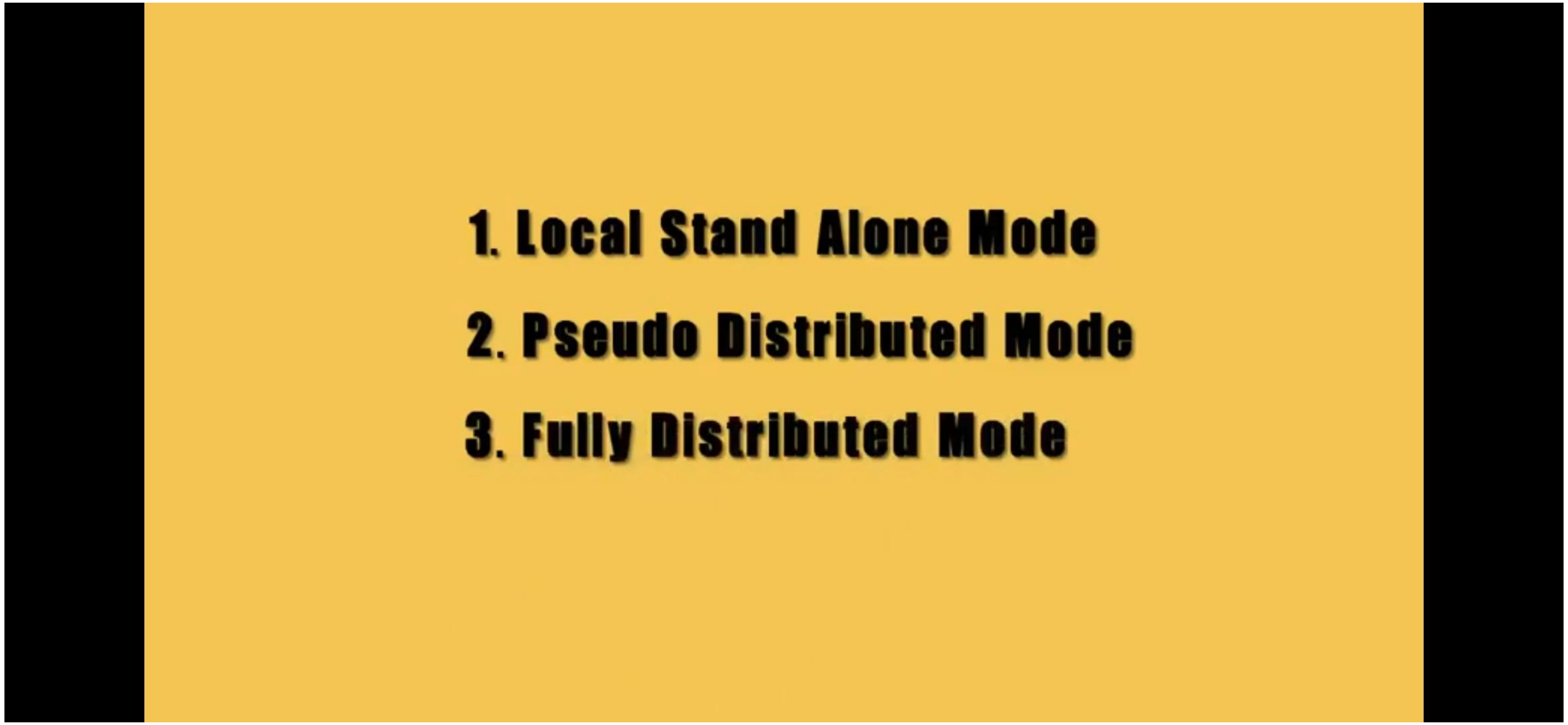
HADOOP CLUSTER MODES

- A Hadoop cluster is nothing but a group of computers connected together .
- We use it for storing and processing large data sets.
- Hadoop clusters have a number of commodity hardware connected together.
- They communicate with a high-end machine which acts as a master.
- These master and slaves implement distributed computing over distributed data storage.
- It runs open source software for providing distributed functionality.

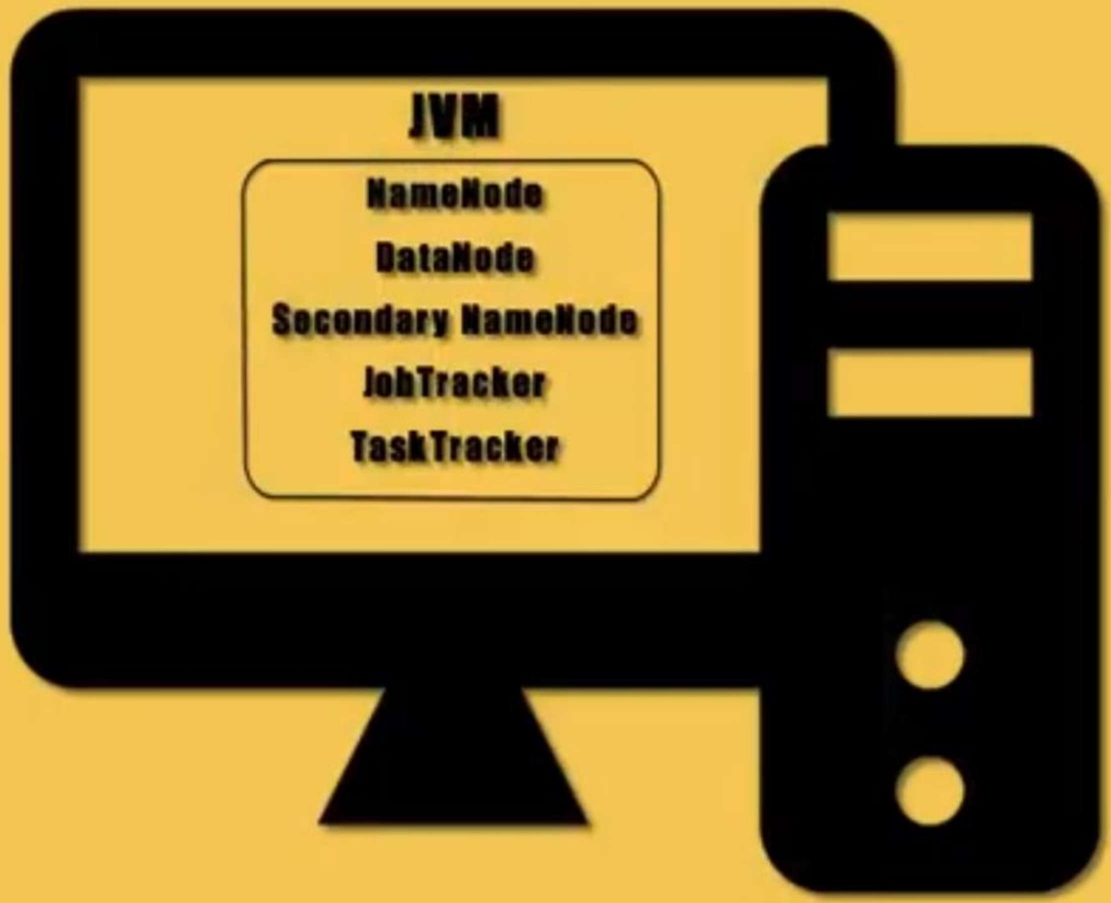
Single Node Cluster VS Multi-Node Cluster

- As the name suggests, single node cluster gets deployed over a *single machine*. And multi-node clusters gets deployed on *several machines*.
- In **single-node Hadoop clusters**, all the daemons like NameNode, DataNode run on the same machine. In a single node Hadoop cluster, all the processes run on one JVM instance. The user need not make any configuration setting. The Hadoop user only needs to set JAVA_HOME variable. The default factor for single node Hadoop cluster is one.
- In **multi-node Hadoop clusters**, the daemons run on separate host or machine. A multi-node Hadoop cluster has master-slave architecture. In this NameNode daemon run on the master machine. And DataNode daemon runs on the slave machines. In multi-node Hadoop cluster, the slave daemons like DataNode and NodeManager run on cheap machines. On the other hand, master daemons like NameNode and ResourceManager run on powerful servers. Ina multi-node Hadoop cluster, slave machines can be present in any location irrespective of the physical location of the master server.

Configuration Modes Of Hadoop


1. Local Stand Alone Mode
2. Pseudo Distributed Mode
3. Fully Distributed Mode

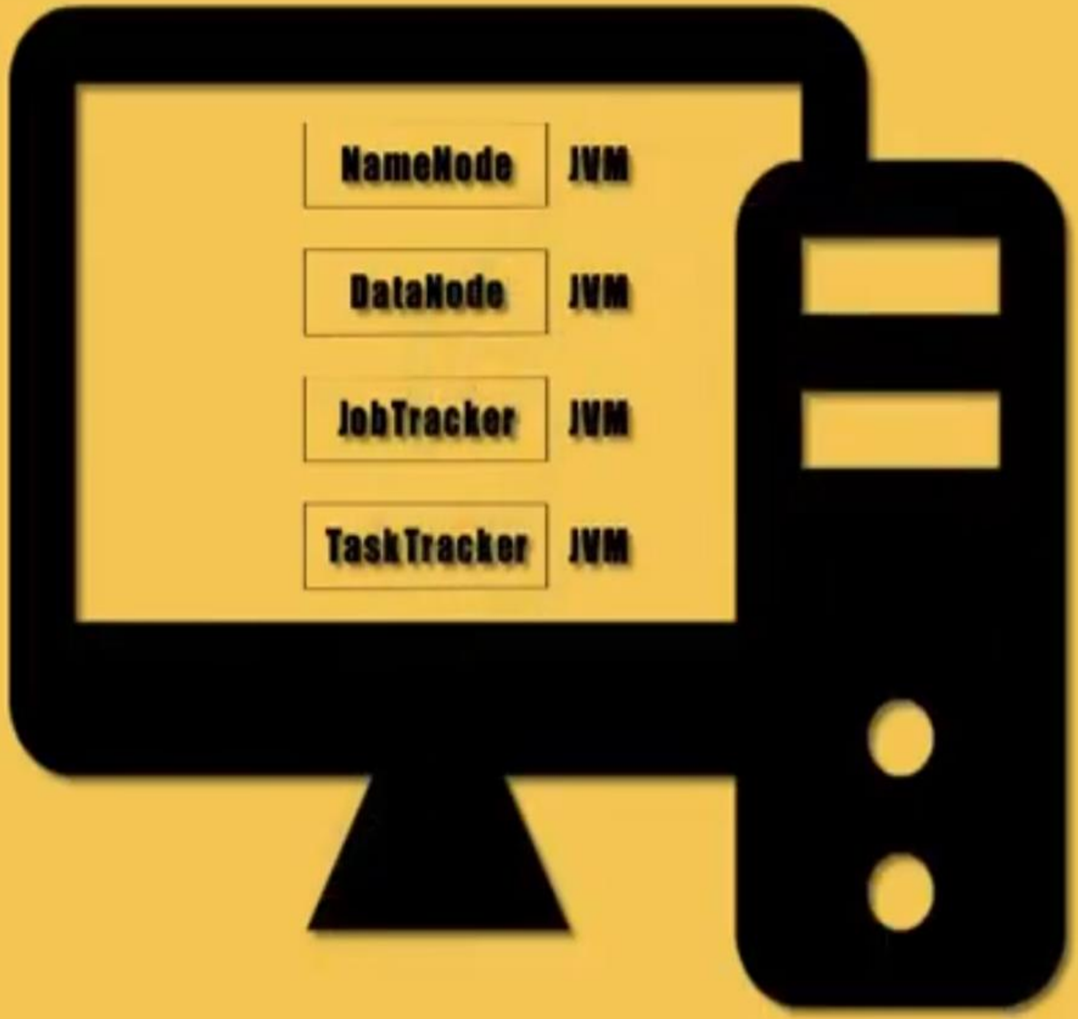
LOCAL STAND ALONE MODE



- **Local/ Standalone Mode**

- All the components of Hadoop such as Namenode, Datanode, Secondary Nanenode, TaskTracker, JobTracker are configured on the same machine.
- It is running within the single JVM.

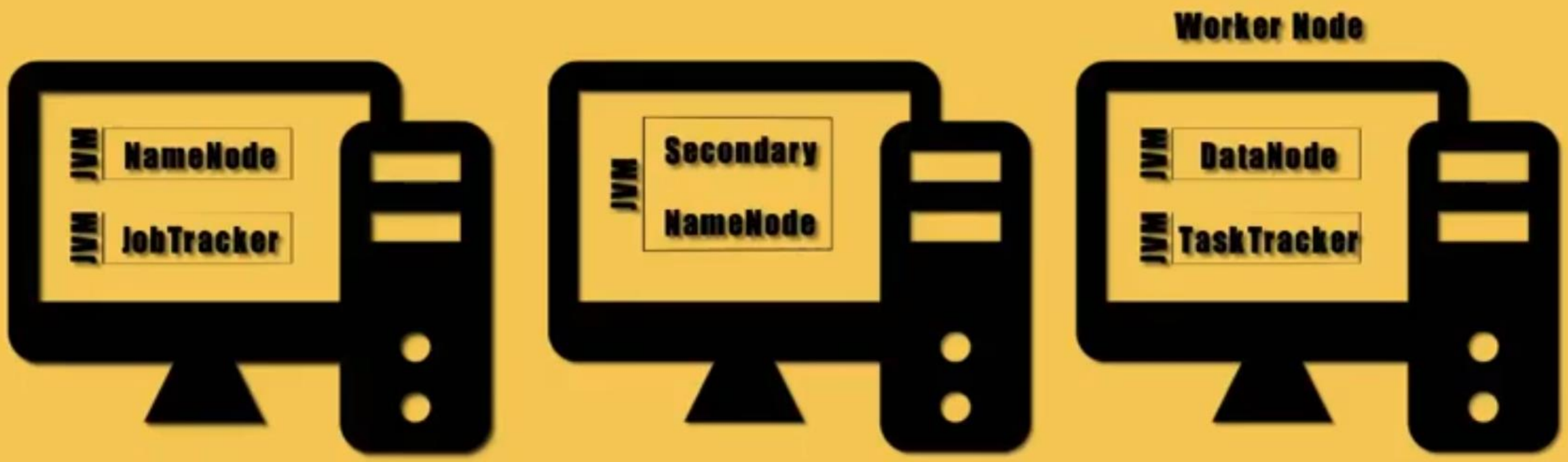
PSEUDO DISTRIBUTED MODE



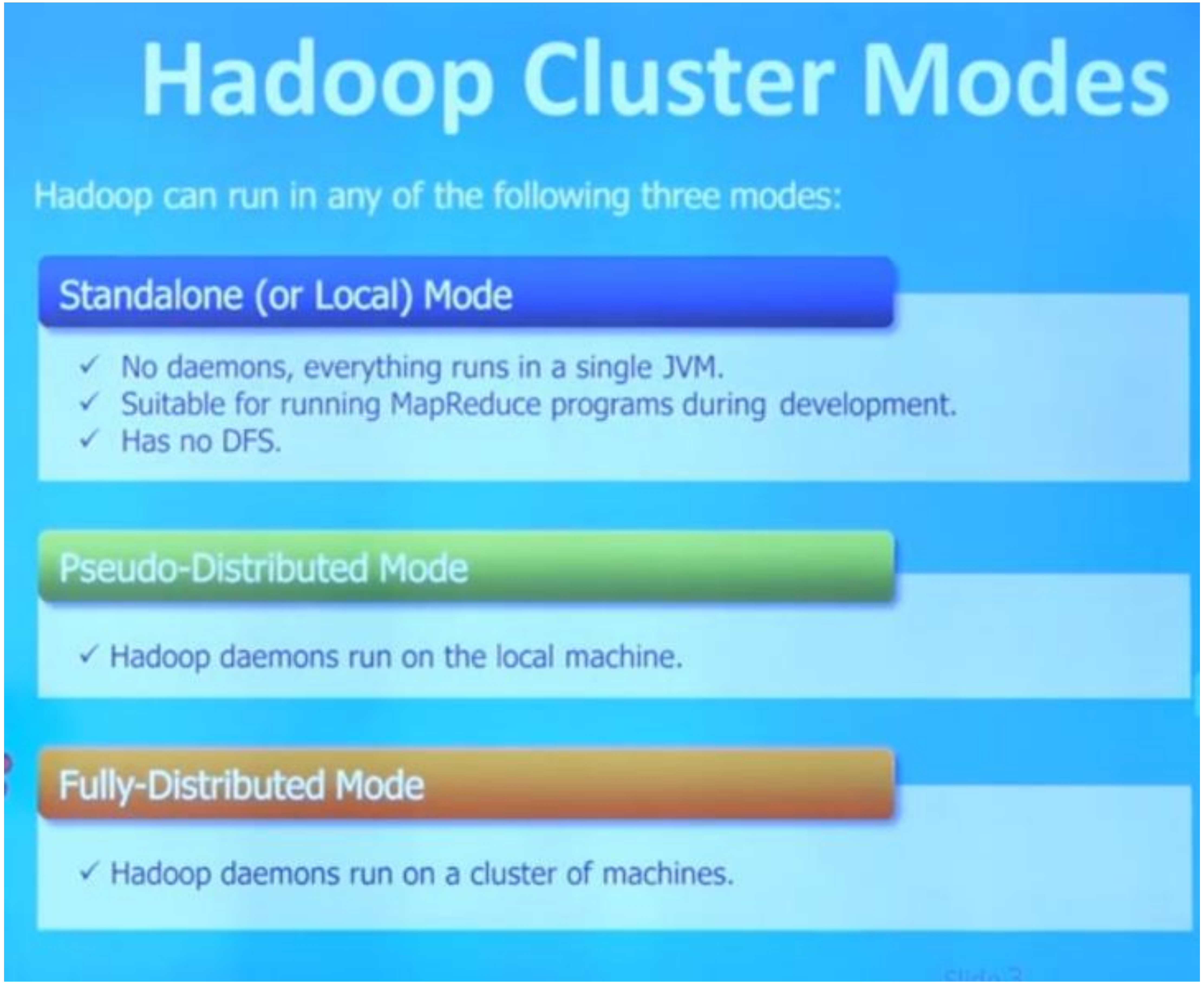
- **Pseudo Distributed Mode**

- All the components of Hadoop such as Namenode, Datanode, Secondary Nanenode, TaskTracker, JobTracker are configured on the same machine and each of these components running on a separate JVM.

FULLY DISTRIBUTED MODE



- **Fully Distributed Mode**
- All the components of Hadoop such as Namenode, Datanode, Secondary Nanenode, TaskTracker, JobTracker are configured on the different machine .
- Usually on a small cluster Namenode and Jobtracker are configured on a same machine with each component having its own JVM.
- Where as on a large cluster Namenode and Jobtracker are configured on a seperate machine.
- Datanode and the Tasktracker are commonly reffered as the worker Node.
- Each cluster would have plenty of Worker Nodes.



Hadoop Cluster Modes

Hadoop can run in any of the following three modes:

Standalone (or Local) Mode

- ✓ No daemons, everything runs in a single JVM.
- ✓ Suitable for running MapReduce programs during development.
- ✓ Has no DFS.

Pseudo-Distributed Mode

- ✓ Hadoop daemons run on the local machine.

Fully-Distributed Mode

- ✓ Hadoop daemons run on a cluster of machines.

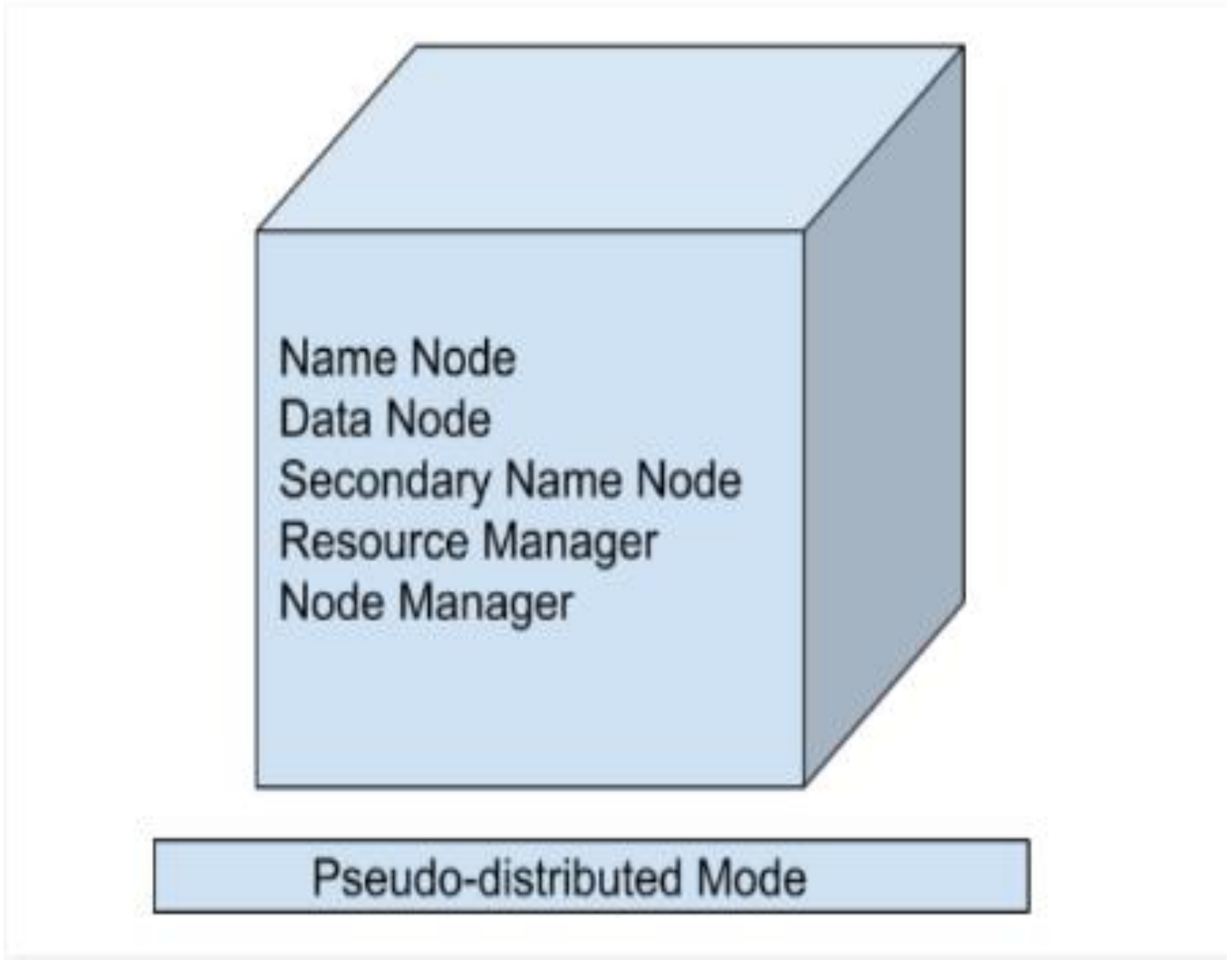
Local Mode / standalone Mode

- Standalone mode is the default mode in which Hadoop run.
- Standalone mode is mainly used for debugging where you don’t really use HDFS.
- You also don’t need to do any custom configuration in the files- mapred-site.xml, core-site.xml, hdfs-site.xml.
- Standalone mode is usually the fastest Hadoop modes as it uses the local file system for all the input and output.
- HDFS is not being used
- Uses local file system for input and output
- No need to change any configuration files
- Default Hadoop Modes

Pseudo Distributed Mode

- The pseudo-distribute mode is also known as a single-node cluster where both NameNode and DataNode will reside on the same machine.
- In pseudo-distributed mode, all the Hadoop daemons will be running on a single node.
- All the Master and Slave Daemons will be running on a single system.
- Replication Factor will be ONE for Block
- Changes in configuration files will be required for all the three files

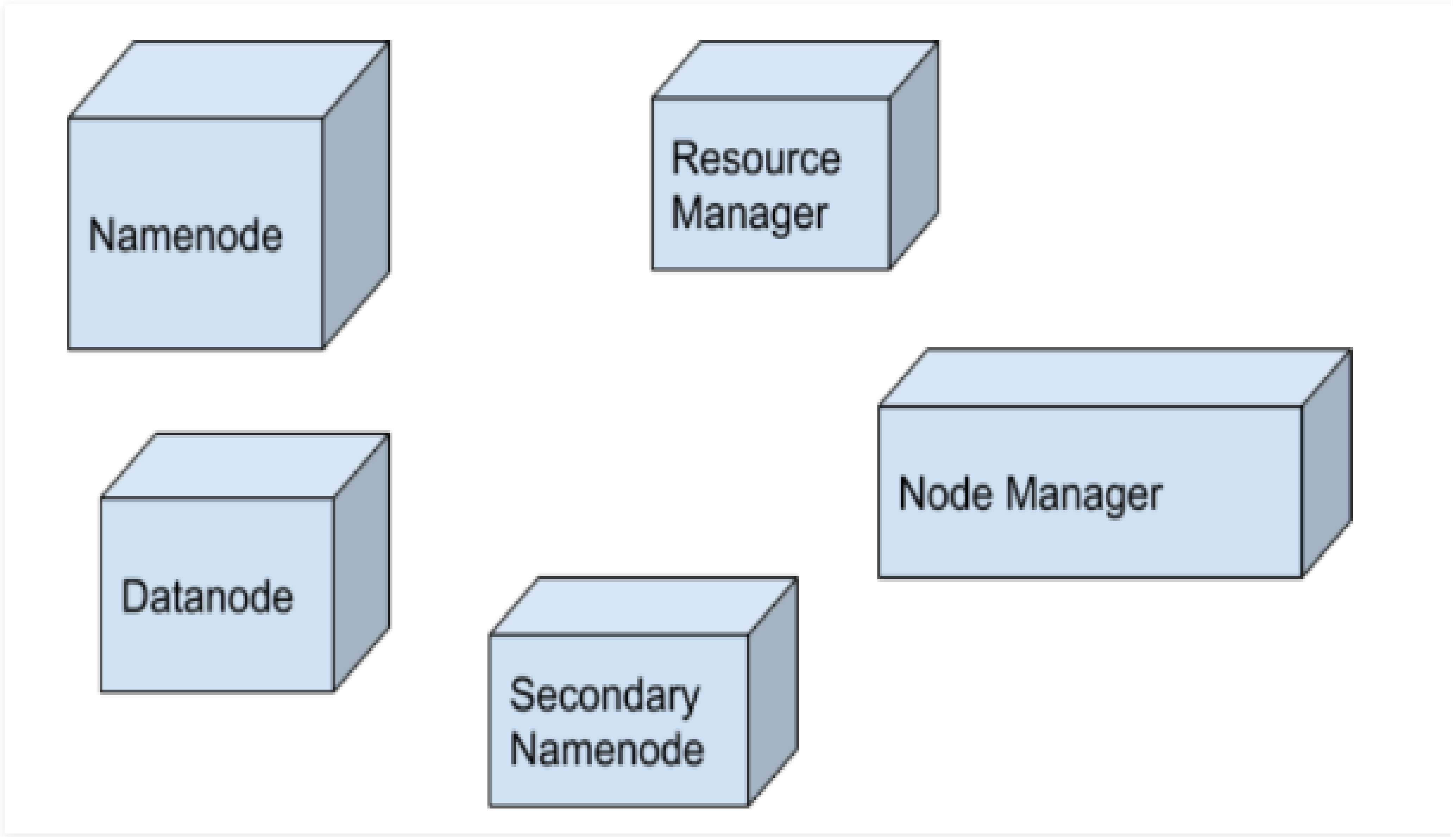
  mapred-site.xml, core-site.xml, hdfs-site.xml


Name Node
Data Node
Secondary Name Node
Resource Manager
Node Manager
Pseudo-distributed Mode

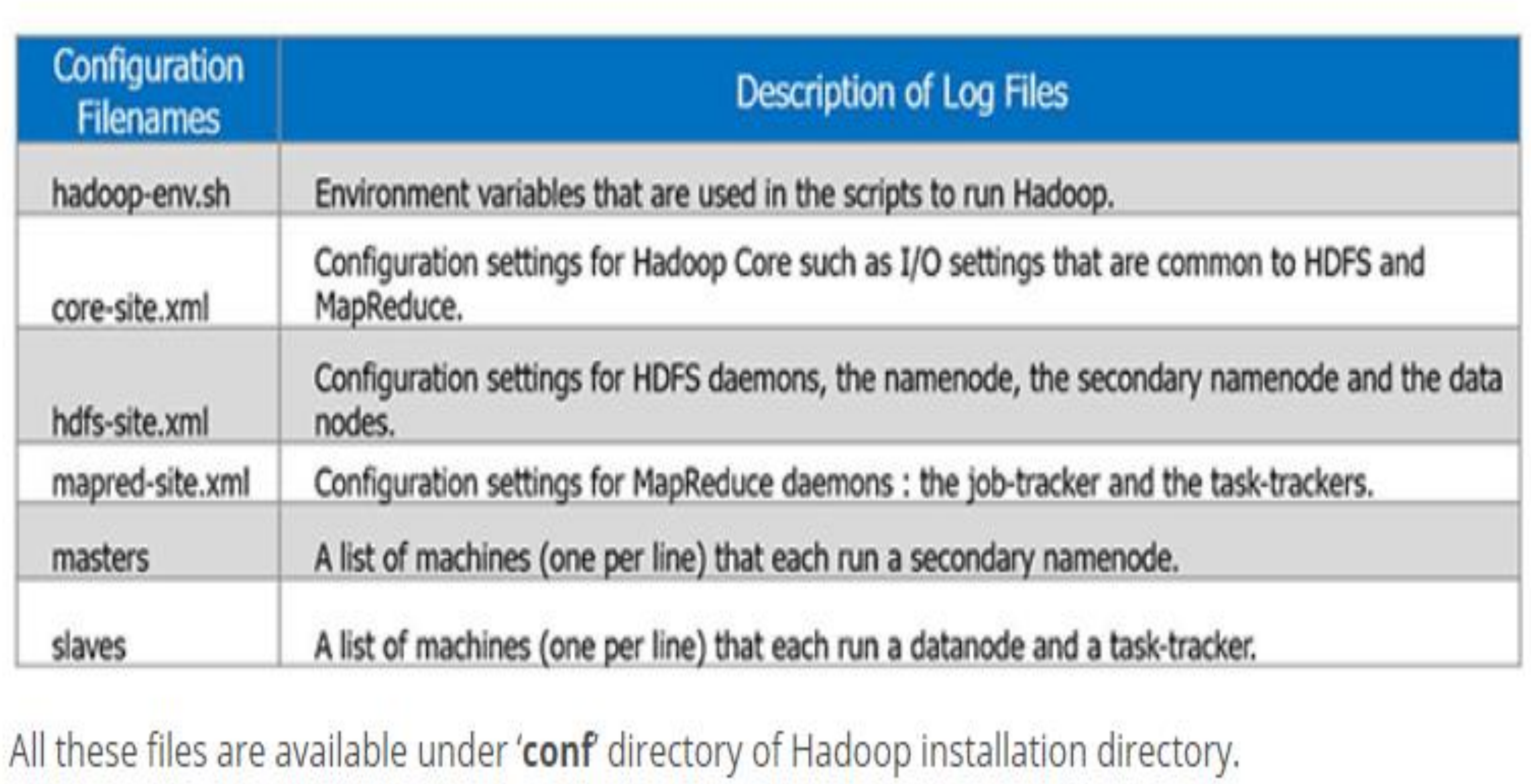
Fully Distributed Mode(Multi-Node Cluster)

- This is the production mode of Hadoop where multiple nodes will be running.
- Here data will be distributed across several nodes and processing will be done on each node.
- Master and Slave services will be running on the separate nodes in fully-distributed Hadoop Mode.

Fully Distributed Mode



Hadoop Cluster Configuration Files.

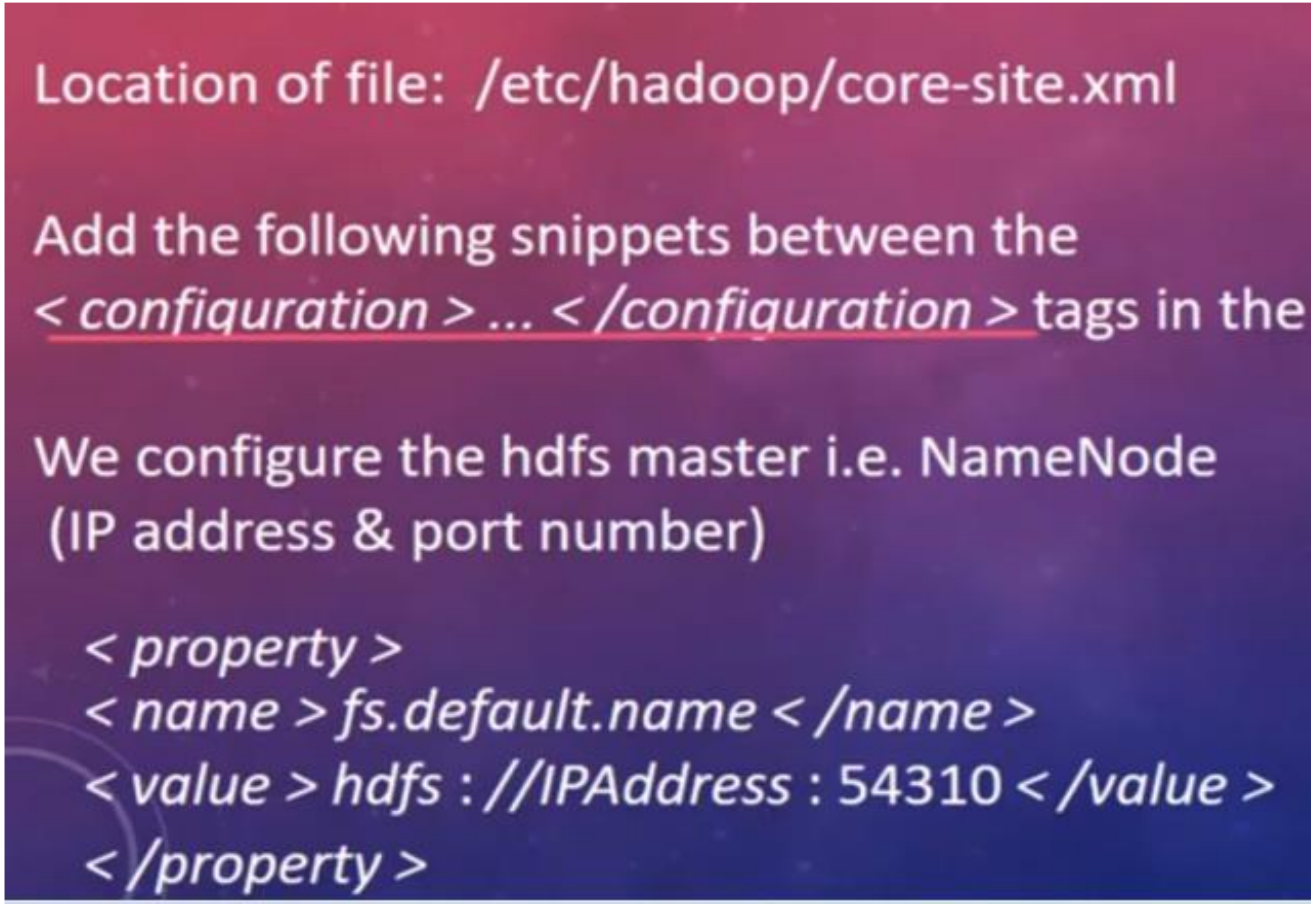


Configuration Filenames	Description of Log Files
hadoop-env.sh	Environment variables that are used in the scripts to run Hadoop.
core-site.xml	Configuration settings for Hadoop Core such as I/O settings that are common to HDFS and MapReduce.
hdfs-site.xml	Configuration settings for HDFS daemons, the namenode, the secondary namenode and the data nodes.
mapred-site.xml	Configuration settings for MapReduce daemons : the job-tracker and the task-trackers.
masters	A list of machines (one per line) that each run a secondary namenode.
slaves	A list of machines (one per line) that each run a datanode and a task-tracker.

All these files are available under '**conf**' directory of Hadoop installation directory.

- **1) HADOOP-ENV.sh**->>It specifies the environment variables that affect the JDK used by Hadoop Daemon (bin/hadoop). We know that Hadoop framework is wriiten in Java and uses JRE so one of the environment variable in Hadoop Daemons is $Java_Home in Hadoop-env.sh.
- **2) CORE-SITE.XML**->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop cluster.It informs Hadoop daemons where the NAMENODE runs in the cluster. It also informs the Name Node as to which IP and ports it should bind.
- **3) HDFS-SITE.XML**->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop. It contains the configuration settings for NAMENODE, DATANODE, SECONDARYNODE. It is used to specify default block replication. The actual number of replications can also be specified when the file is created,
- **4) MAPRED-SITE.XML**->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop. It contains the configuration settings for MapReduce . In this file, we specify a framework name for MapReduce, by setting the MapReduce.framework.name.
- **5) Masters**->>It is used to determine the master Nodes in Hadoop cluster. It will inform about the location of SECONDARY NAMENODE to Hadoop Daemon.
  The Mater File on Slave node is blank.
- **6) Slave**->>It is used to determine the slave Nodes in Hadoop cluster.
  The Slave file at Master Node contains a list of hosts, one per line.
  The Slave file at Slave server contains IP address of Slave nodes.

Core-site.xml



Location of file: /etc/hadoop/core-site.xml

Add the following snippets between the *< configuration > ... < /configuration >* tags in the

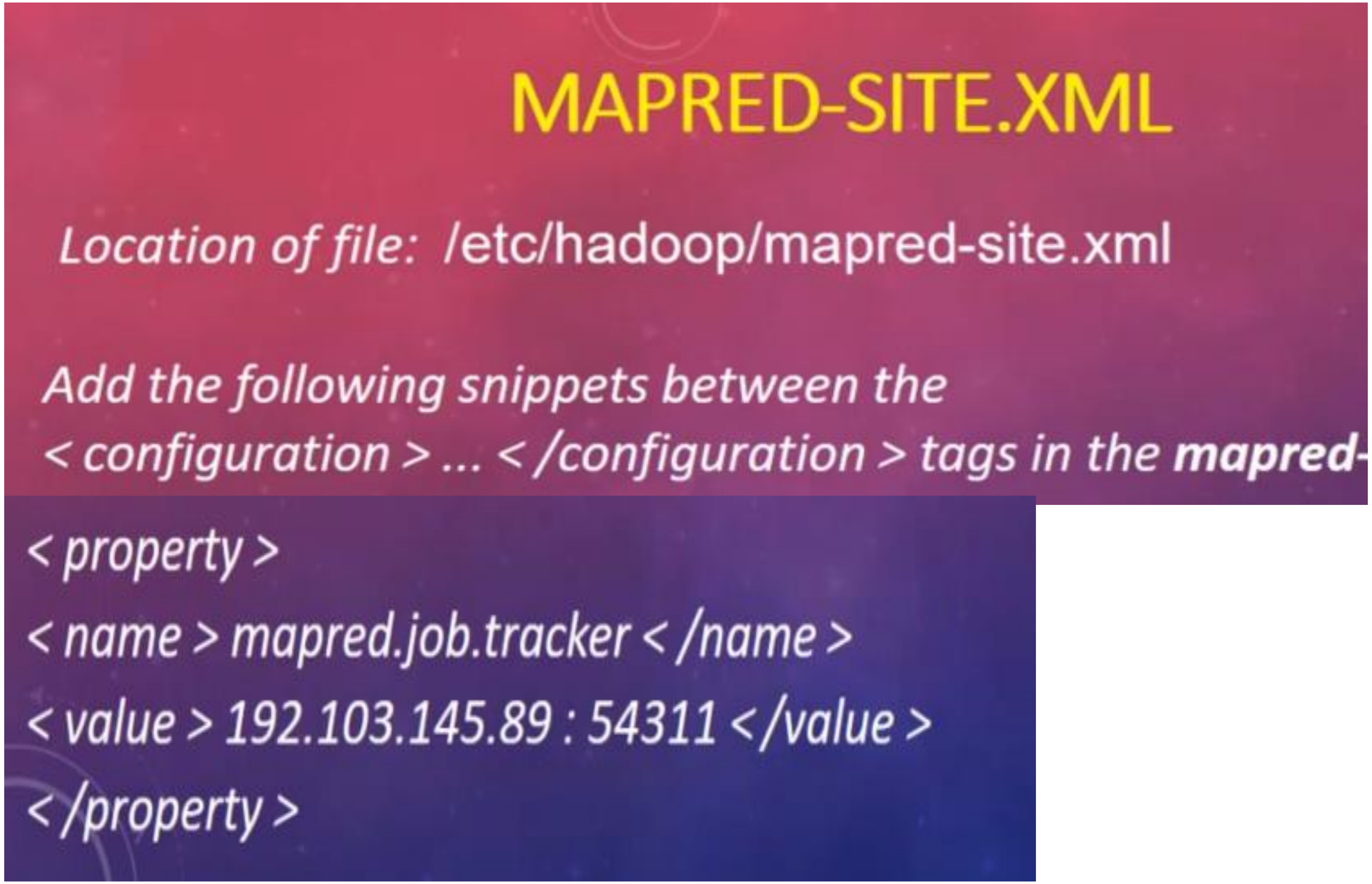
We configure the hdfs master i.e. NameNode (IP address & port number)

```
< property >
< name > fs.default.name < /name >
< value > hdfs : //IPAddress : 54310 < /value >
< /property >
```

Core-site.xml

- It is used to configure the HDFS Master(Namenode)
- In the configuration tag we can configure n number of property.
- All property are written inside the open and close tags
- In the Core-site.xml we can configure the IP Address of the Namenode and the port address.

Mapred-site.xml



MAPRED-SITE.XML

Location of file: /etc/hadoop/mapred-site.xml

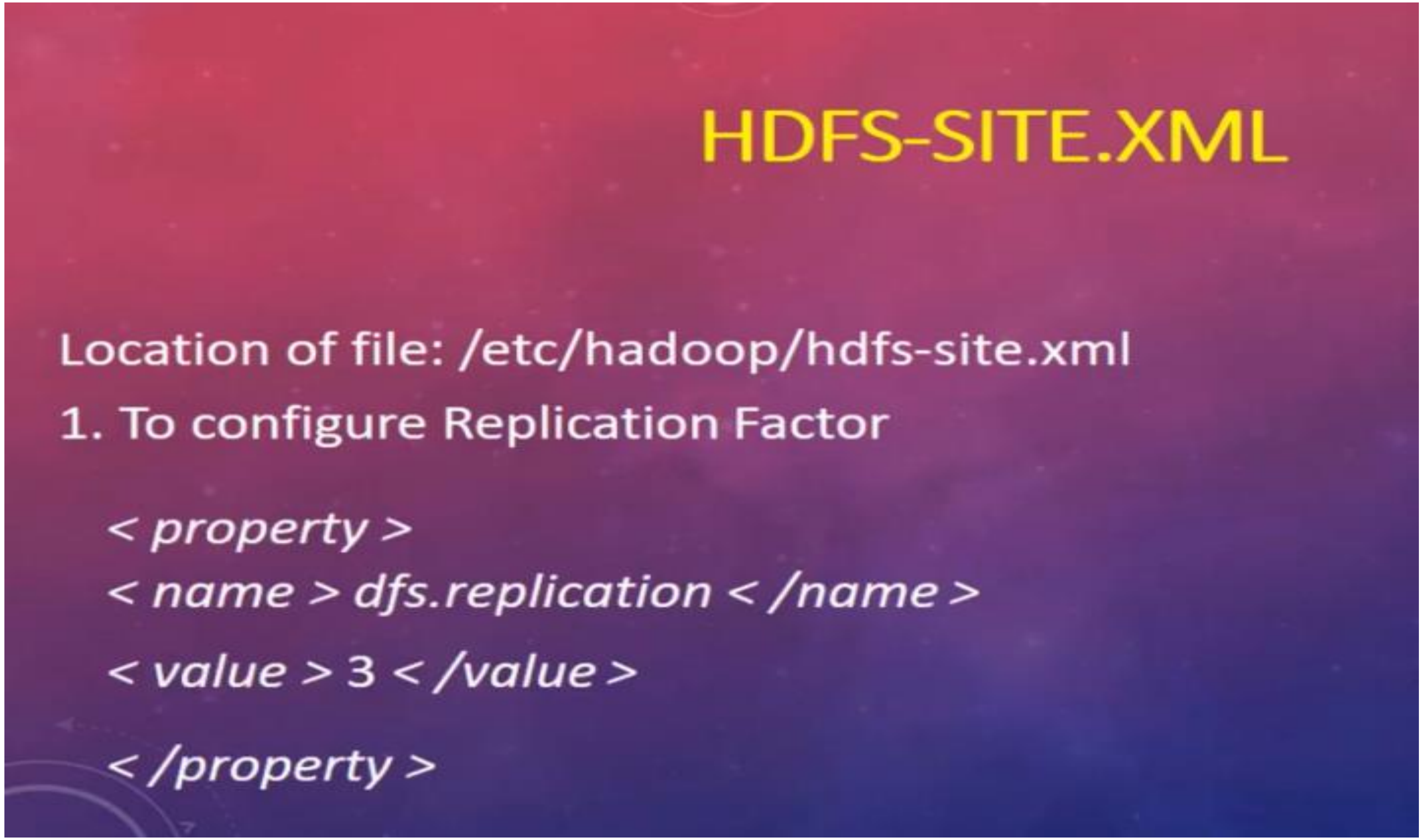
Add the following snippets between the < configuration > ... < /configuration > tags in the ***mapred-***

```
< property >
< name > mapred.job.tracker < /name >
< value > 192.103.145.89 : 54311 </value >
< /property >
```

Mapred-site.xml

- In the Mapred-site.xml we configure the master of Mapreduce that is the Jobtracker.
- We can locate the machine in which jobtracker is running by configuring name , IP, and Port

Hdfs-site.xml



HDFS-SITE.XML

Location of file: /etc/hadoop/hdfs-site.xml

1. To configure Replication Factor

```
< property >
< name > dfs.replication < /name >
< value > 3 < /value >
< /property >
```

2. To configure/ specify Namenode metadata location

```
< property>
< name> dfs.nanenode.name.dir</name>
< value > file:/user/local/hadoopdata/hdfs/namenode</value>
</property>
```

3. To configure/ specify Datanode data storage location

```
< property>
< name> dfs.datanode.name.dir</name>
< value > file:/user/local/hadoopdata/hdfs/datanode</value>
</property>
```

Master and Slave File

- Master: Contains the IP Addresses of the node where secondary Namenode is running.
- Slave: Contains all the IP Addresses of the Datanode and the tasktracker.

Hadoop-env.sh

- It is a Shell file.
- Hadoop require JRE Environment so where this JRE is located that path is set in environment varriable called javahome.