Introduction to Hadoop and MapReduce Programming - Session 1

Course: Data Engineering - I

Lecture On: Introduction to Hadoop

Instructor: Vishwa Mohan

Segment - 01
Module Introduction

Module Introduction

Session 1

- Introduction to distributed systems
- GFS and MapReduce
- Introduction to Hadoop and components of HDFS
- Hadoop 2.x
- YARN and task processing
- Tools for Hadoop

Session 2

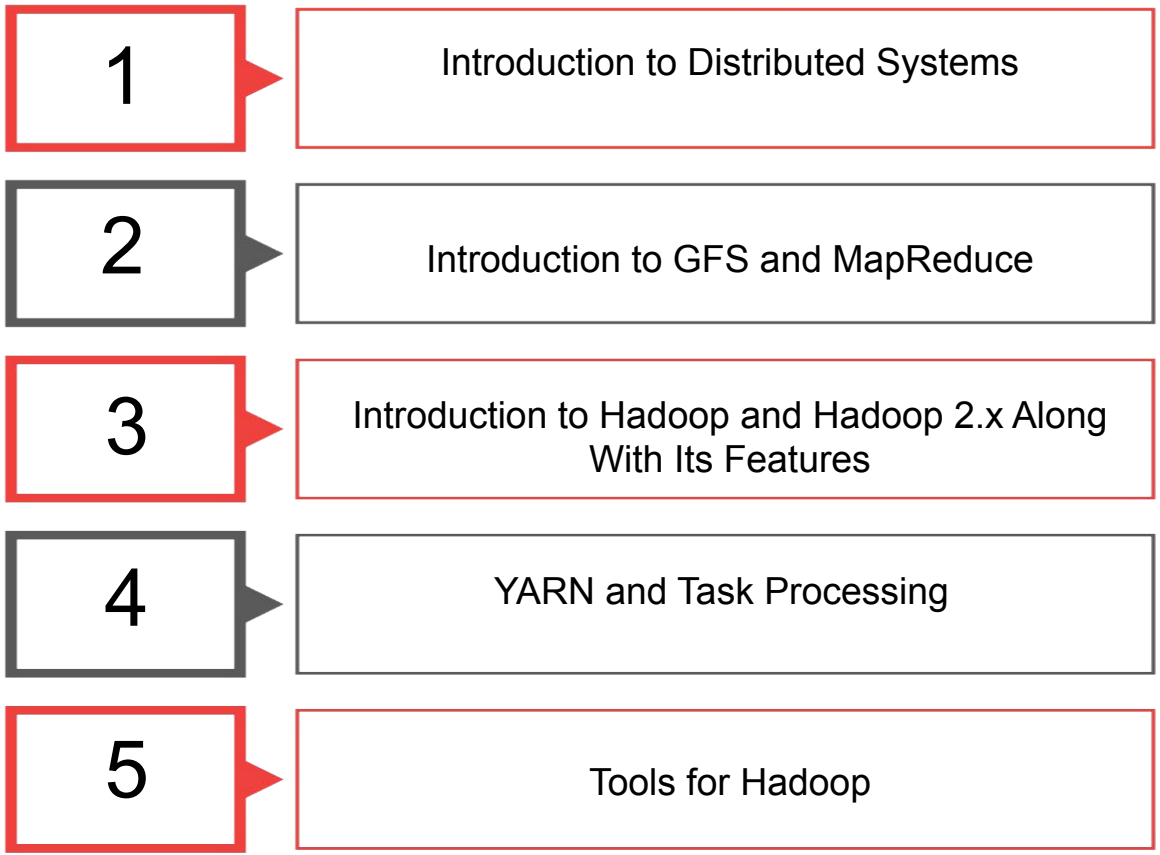
- File storage in HDFS
- Demonstration of basic commands in HDFS
- Write operation and rack awareness
- Read operation in HDFS
- Features and limitations of HDFS

Session 3

- Introduction to the MapReduce Framework
- Basic implementation of MapReduce using Python
- Hadoop streaming with demonstration
- The Combiner
- The Partitioner
- Job scheduling and fault tolerance in MapReduce

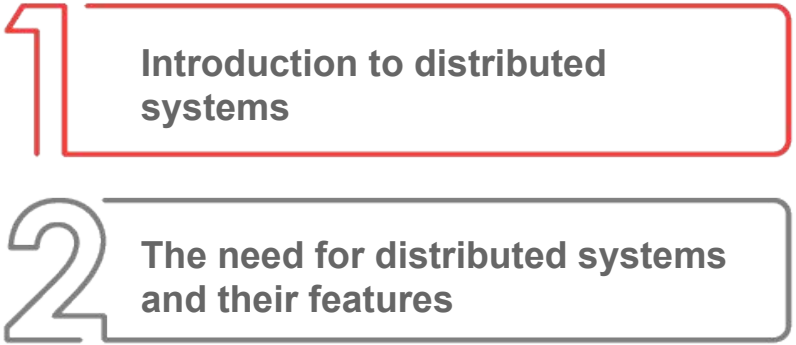
Segment - 02
Introduction: Introduction to Hadoop

Session Overview



Segment - 03
Introduction to Distributed Systems

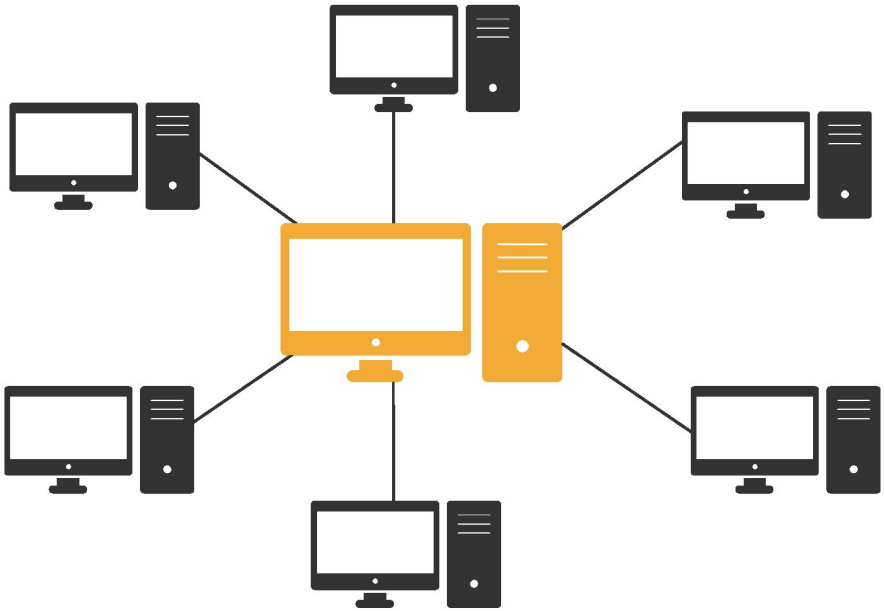
Segment Overview



Why are distributed systems needed?

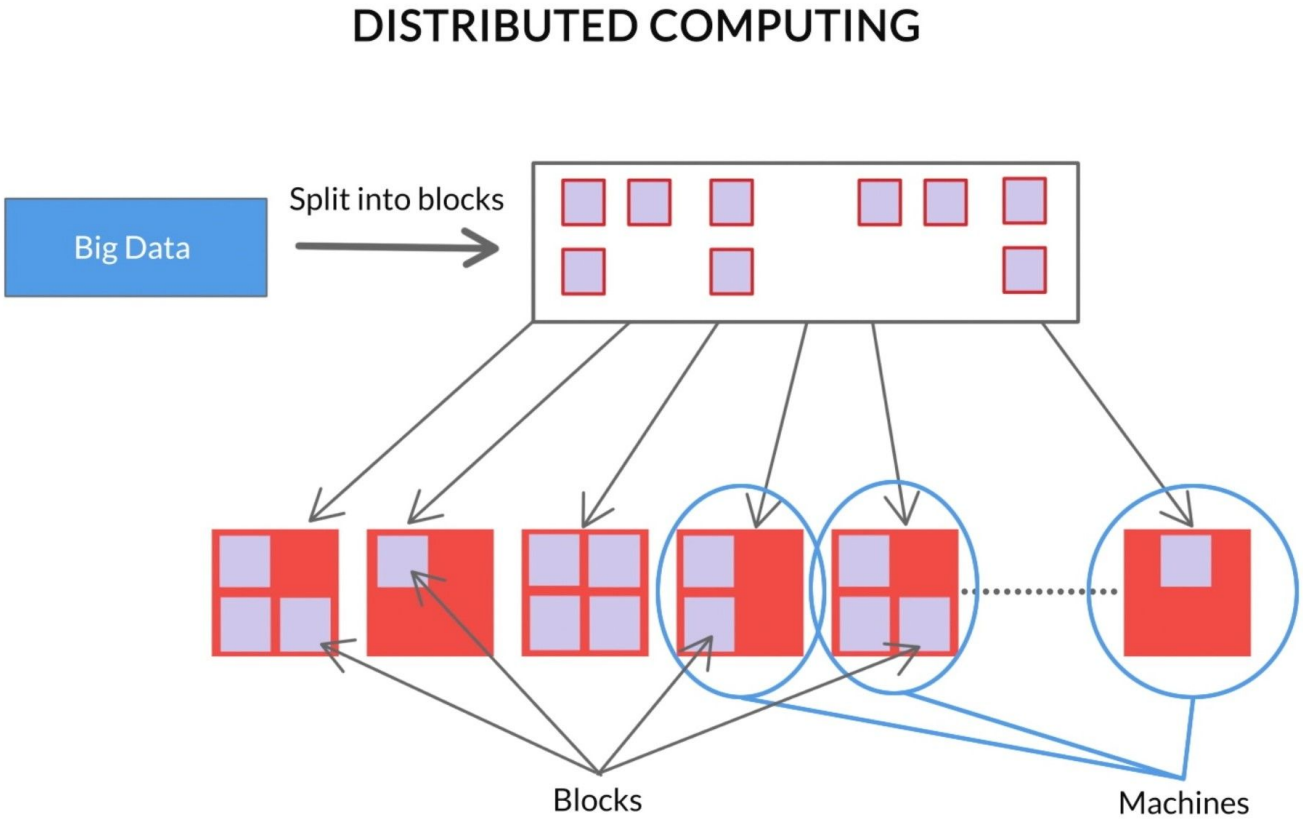
- A single store of Coffee-house may take 100s of order each day
- If suppose there are 10 stores in a small city, data grows more or less proportionately.
- At the global level, if Coffee-house has 10000 stores across the world, the amount of data generated and processed can’t be handled by a single machine.

Introduction to Distributed Systems



A **distributed system** is a system in which several independent computers are connected to each other over a network with middleware to look like a single machine.

Introduction to Distributed Systems

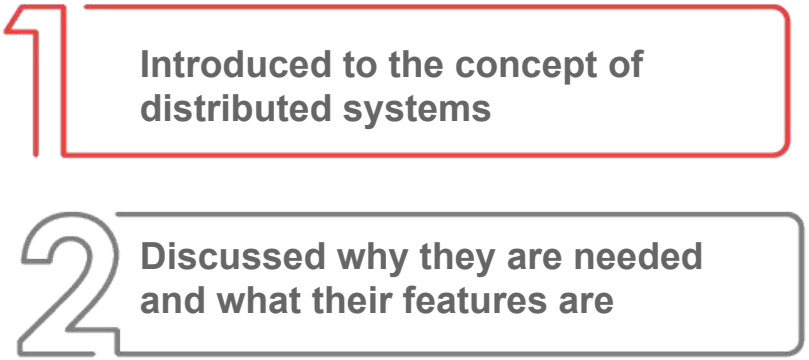


Introduction to Distributed Systems

Primary Challenges faced by Distributed Systems

- **Performance -** Also includes challenges like communication fault delay or computation fault delay.
- **Fault tolerance -** Tolerate faults and function normally.
- **Scalability -** System must remain effective even under heavy load.
- **Security -** Probable threats like information leakage, integrity violation, denial of services and illegitimate use
- **Concurrency -** Shared access to resources must be made available to the required processes.
- **Migration -** Tasks should be allowed to move within the system without affecting other operations.
- **Load balancing -** Load must be distributed among available resources for better performance.

Segment Summary

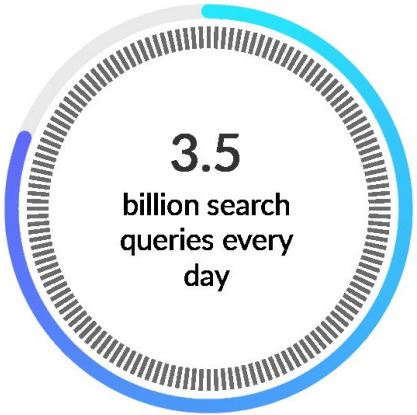


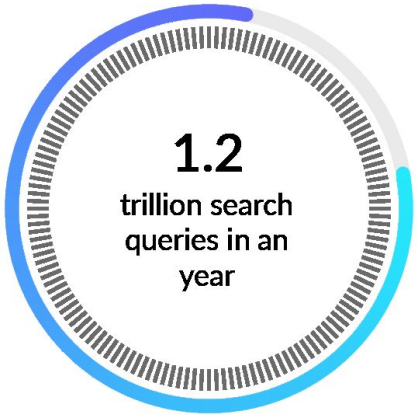
Segment - 04
Introduction to GFS and MapReduce

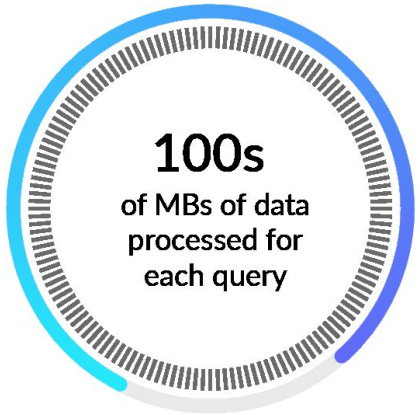
Segment Overview

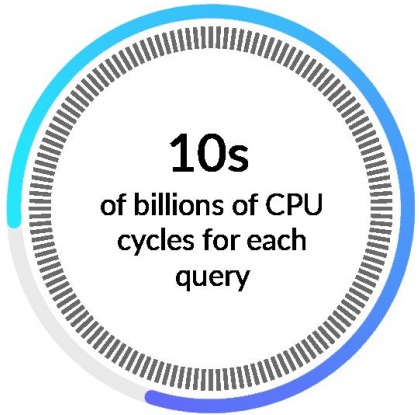
Introduction to GFS and MapReduce

Today, on average, Google handles...









Introduction to GFS and MapReduce

Google File System (GFS)

GFS is a proprietary distributed file system made by Google for storing and processing large amounts of data in multiple commodity machines in large clusters, which ensures reliability and scalability and also maintains efficiency.

Considerations made while developing GFS and their effects

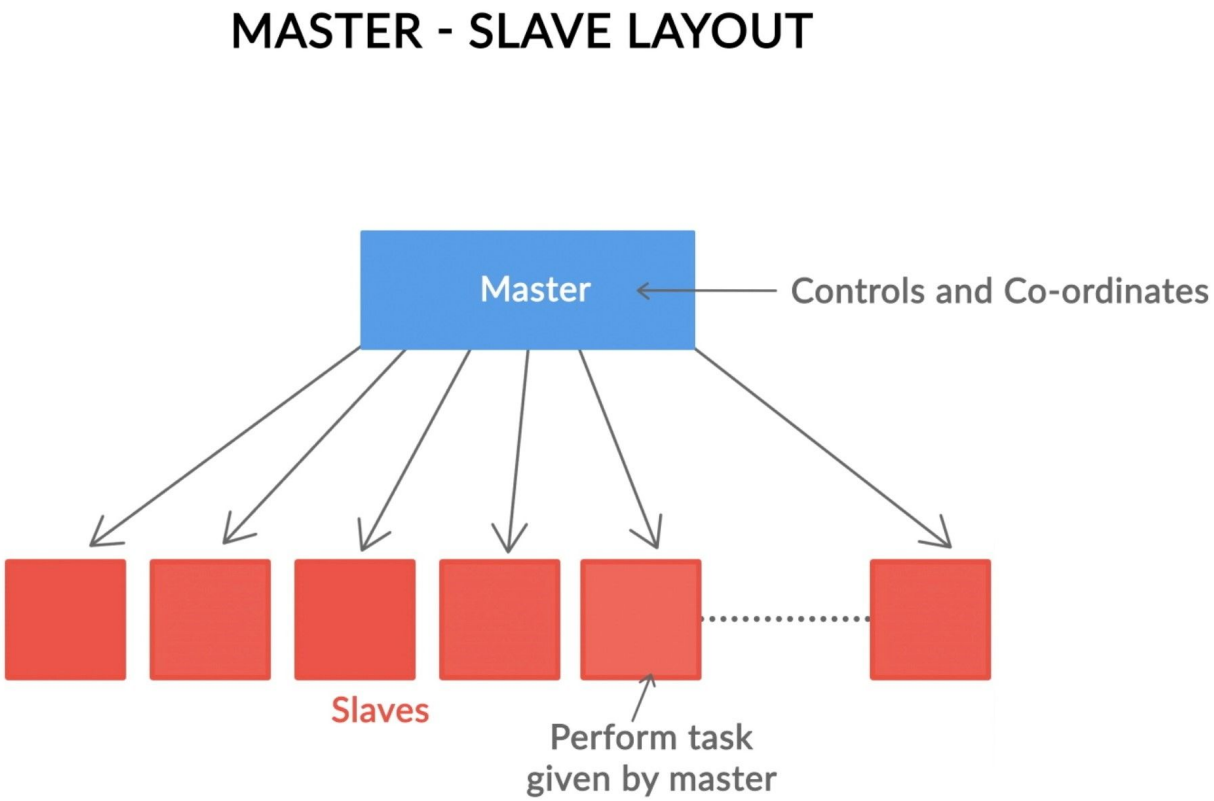
1. Uses commodity hardware
2. Can be easily scaled horizontally
3. Commodity hardware always fails
4. Fault tolerance and automatic error recovery are crucial

Introduction to GFS and MapReduce

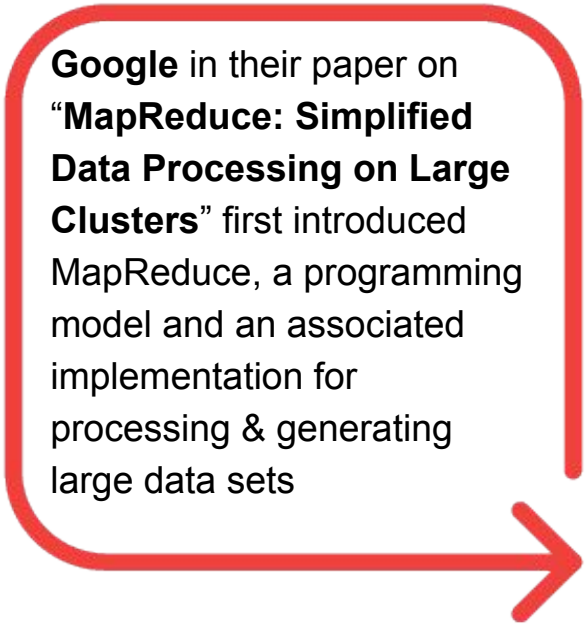
How GFS solved the problems of traditional Distributed systems?

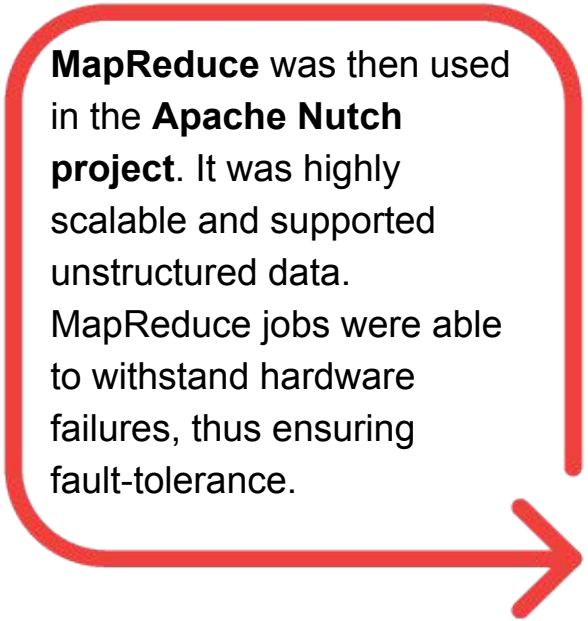
- Very **high availability** and **fault tolerance** maintained through **replication**
- **Automatic and efficient data recovery**
- **High aggregate throughput**
- **Data integrity** is ensured as each chunkserver verifies integrity of their own copies using checksums
- Each chunk server is constantly monitored by GFS master
- **Modularity** allows GFS to easily expand to account for increased loads
- Master Node ensures **load balancing** is maintained

Introduction to GFS and MapReduce

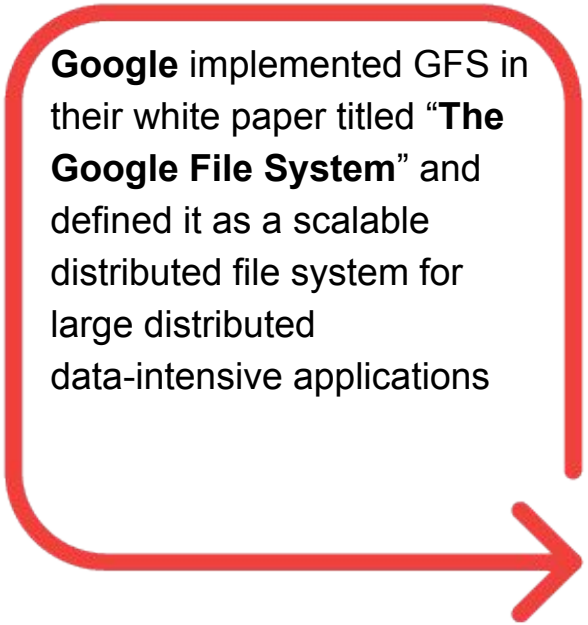


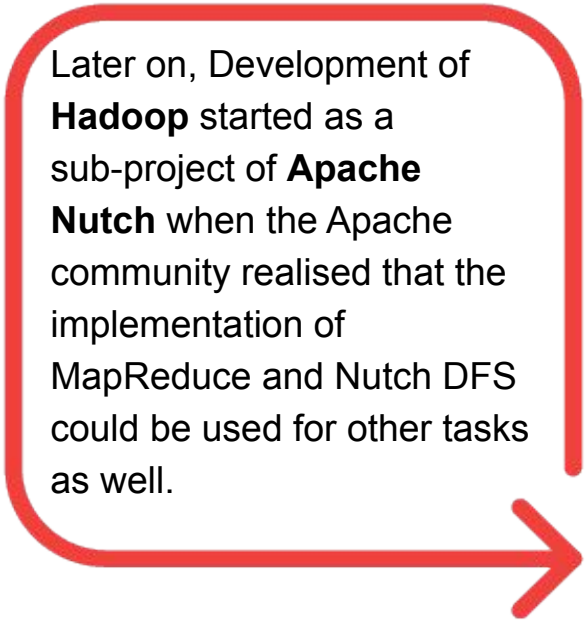
Introduction to GFS and MapReduce



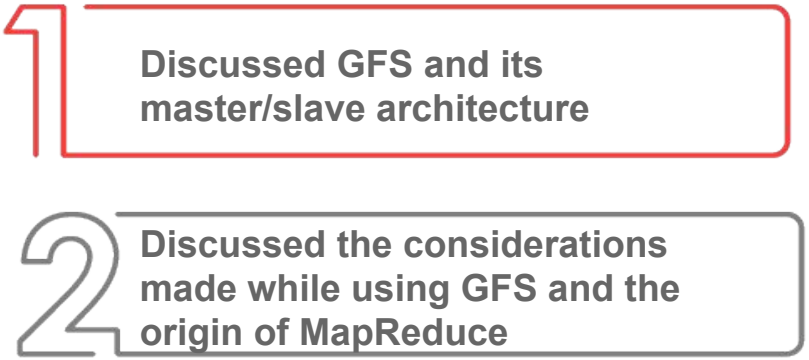


Introduction to GFS and MapReduce



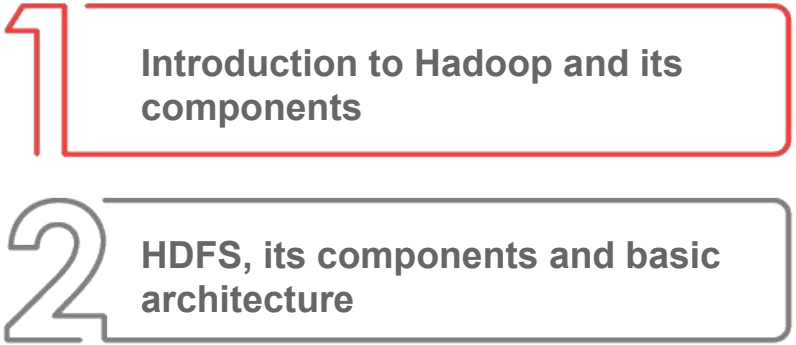


Segment Summary



Segment - 05
Introduction to Hadoop

Segment Overview

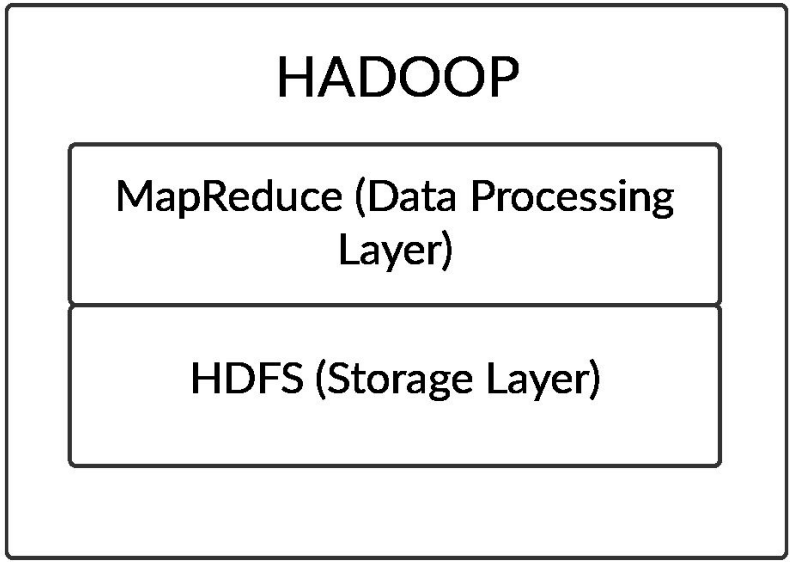


Introduction to Hadoop

Hadoop is an open source framework that is used for storing and processing big data on the clusters of commodity hardware.

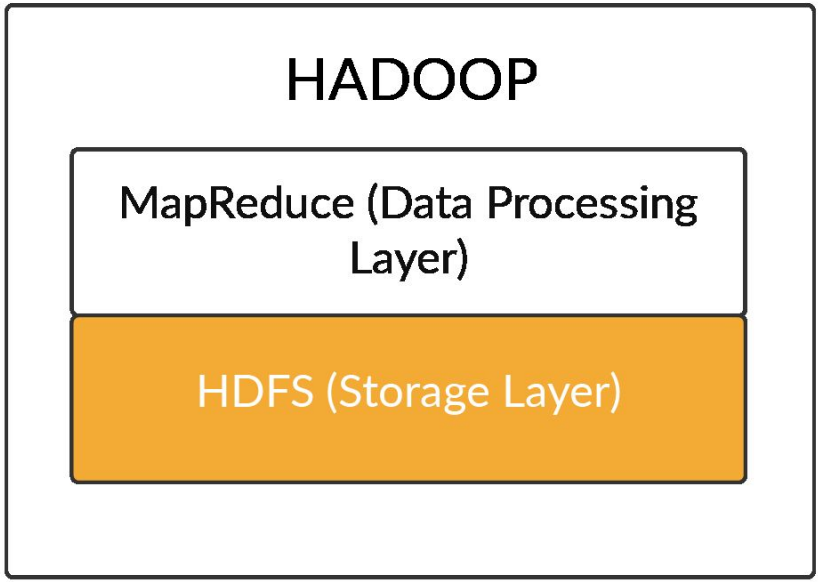
Introduction to Hadoop

Hadoop contains two main components: The Hadoop Distributed File System (HDFS) and MapReduce



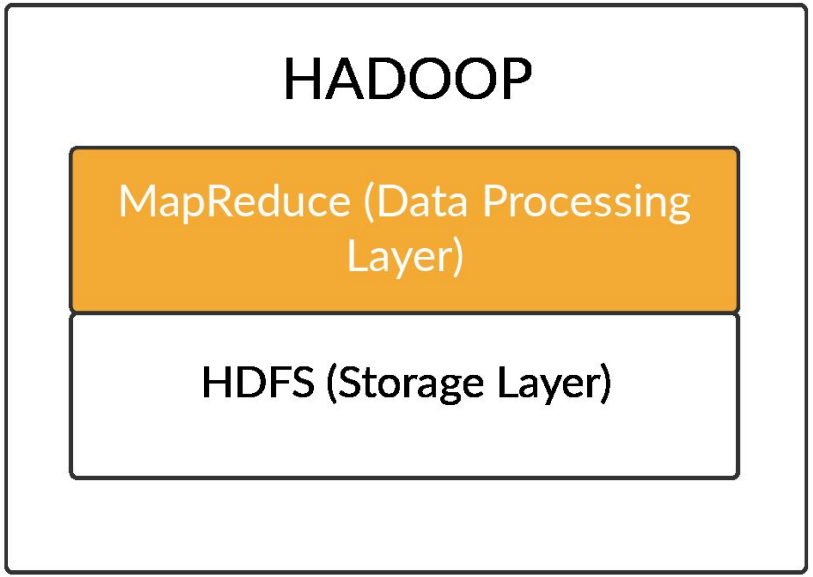
Introduction to Hadoop

Hadoop contains two main components: The Hadoop Distributed File System (HDFS) and MapReduce



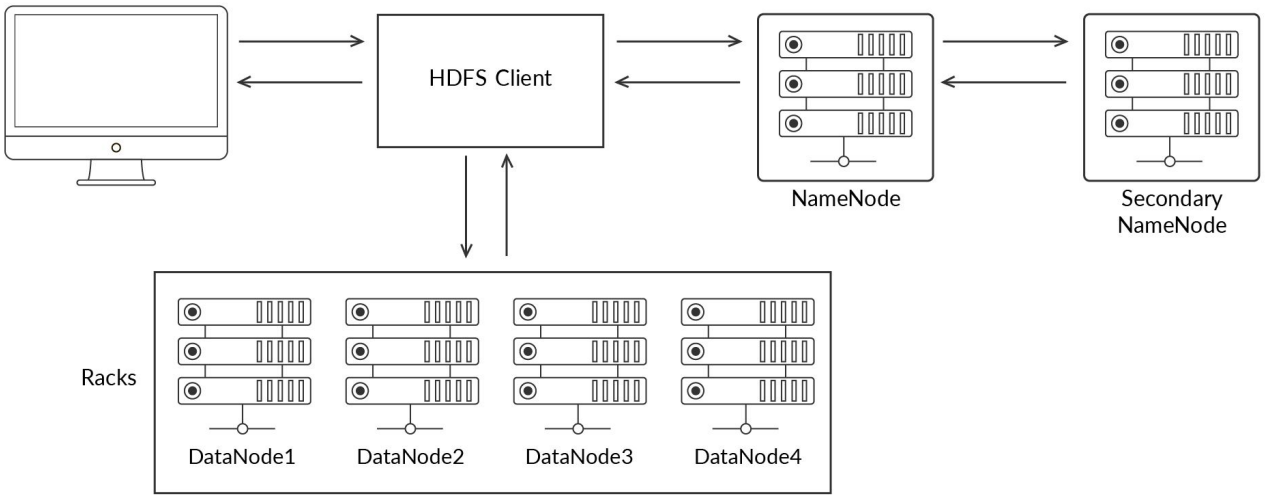
Introduction to Hadoop

Hadoop contains two main components: The Hadoop Distributed File System (HDFS) and MapReduce



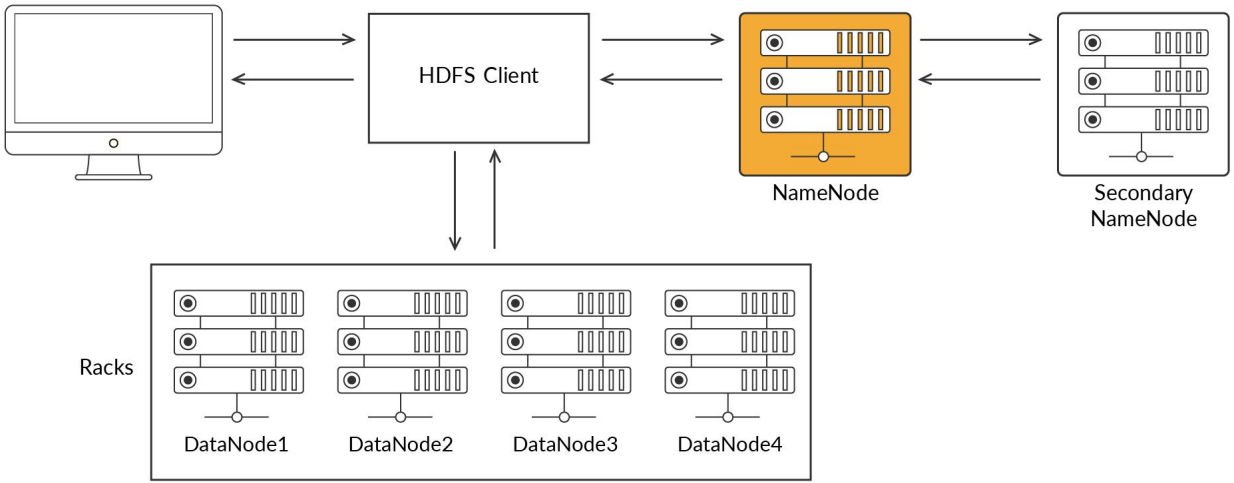
Introduction to Hadoop

HDFS has three main components: The NameNode, the Secondary NameNode and DataNodes



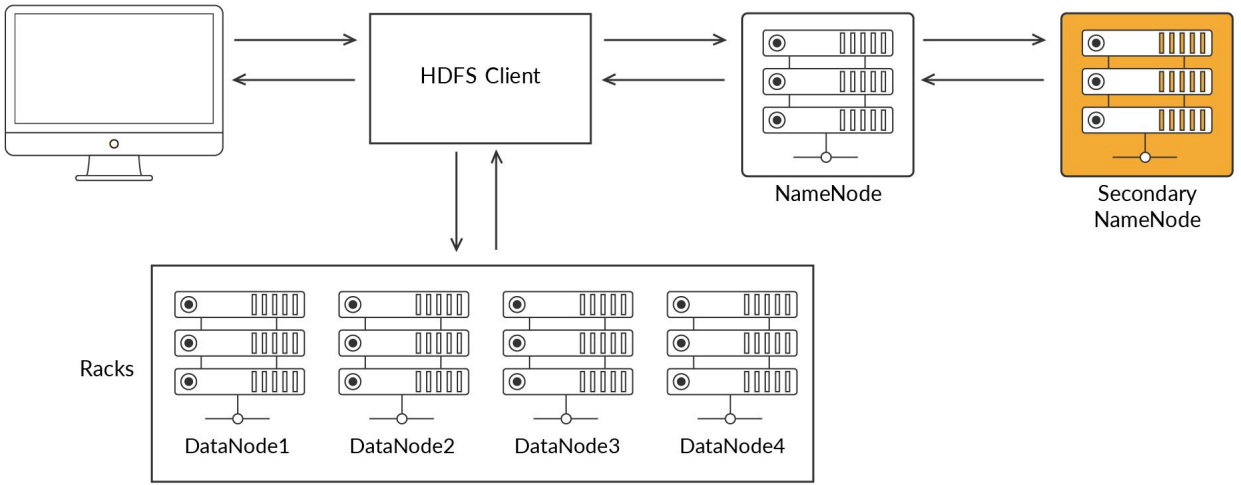
Introduction to Hadoop

HDFS has three main components: The NameNode, the Secondary NameNode and DataNodes



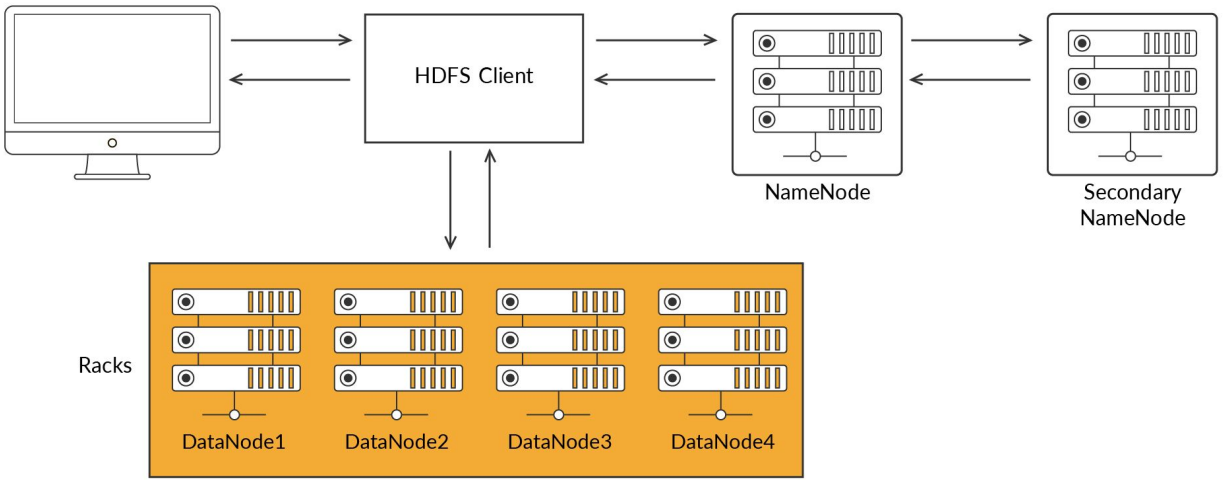
Introduction to Hadoop

HDFS has three main components: The NameNode, the Secondary NameNode and DataNodes

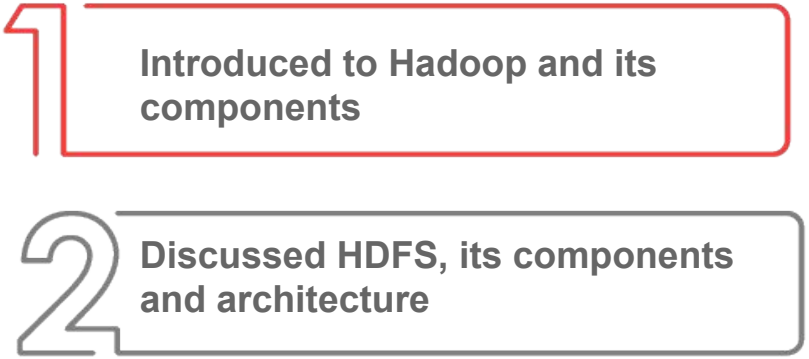


Introduction to Hadoop

HDFS has three main components: The NameNode, the Secondary NameNode and DataNodes



Segment Summary

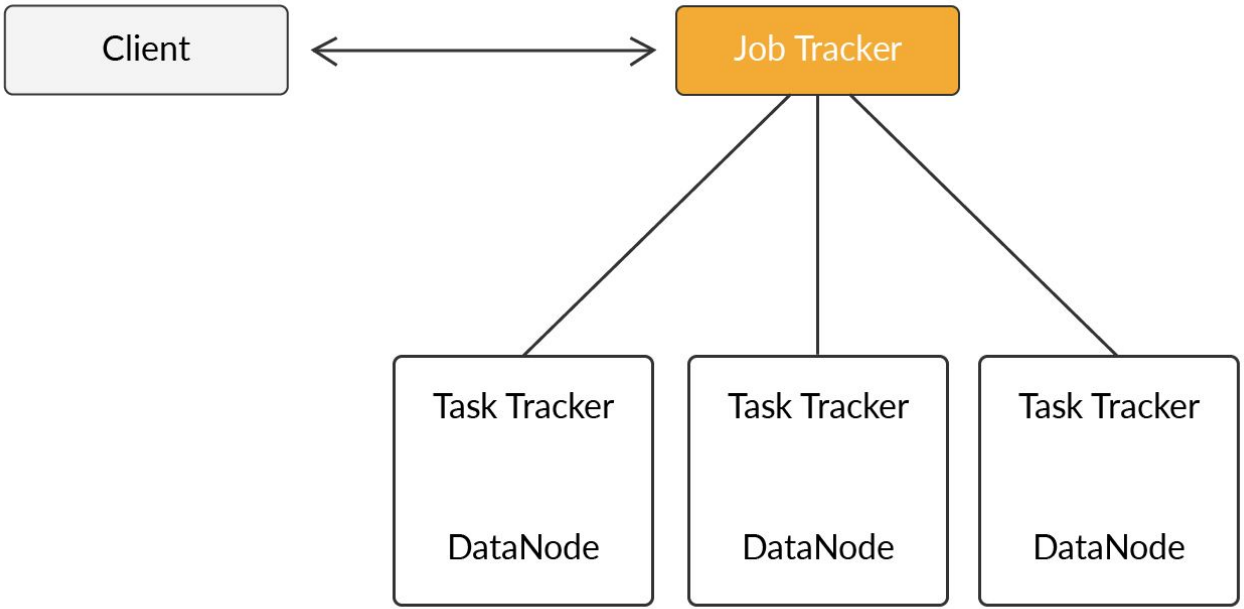


Segment - 06
Hadoop 2.x

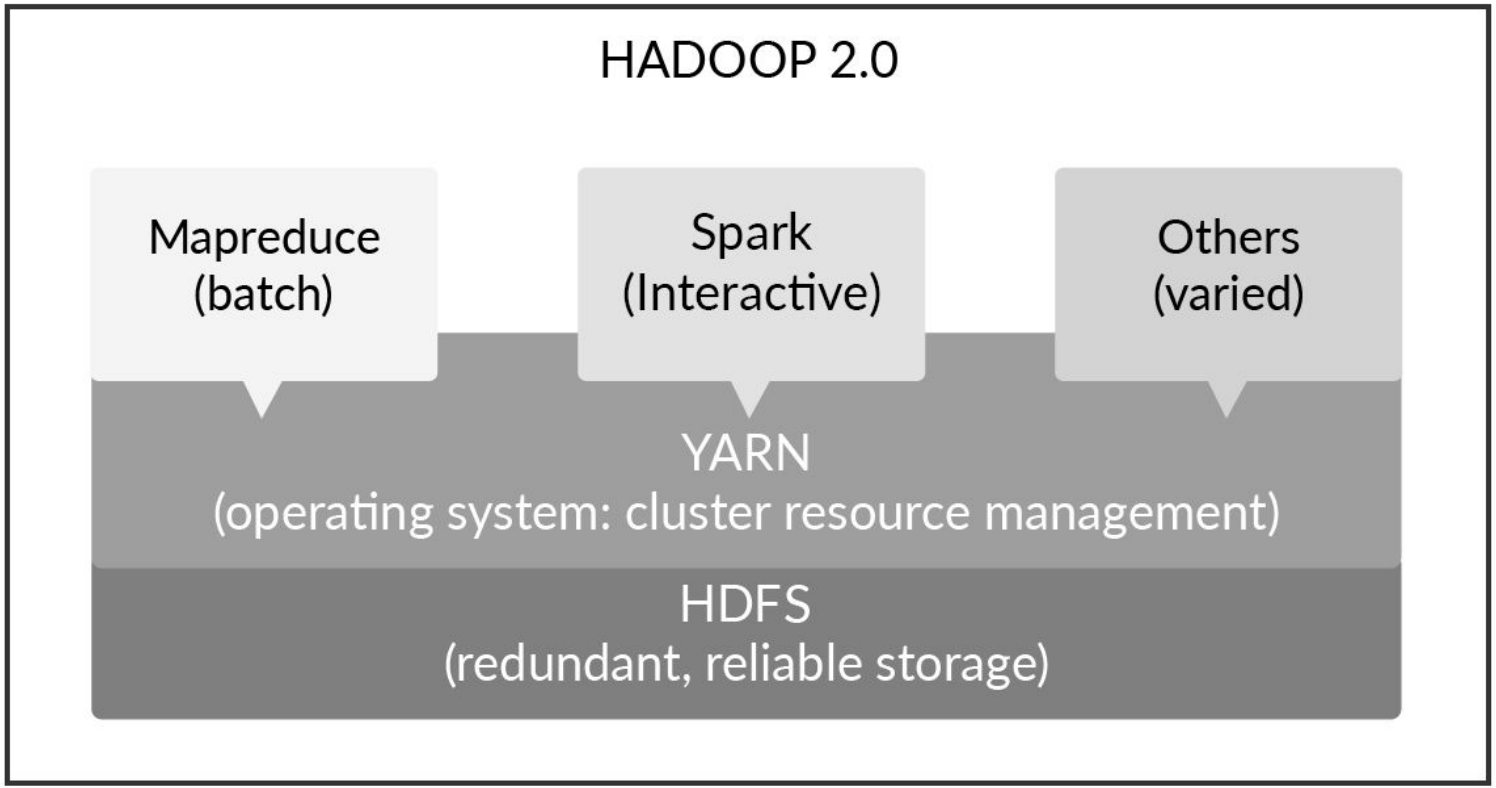
Segment Overview

Hadoop 2.x

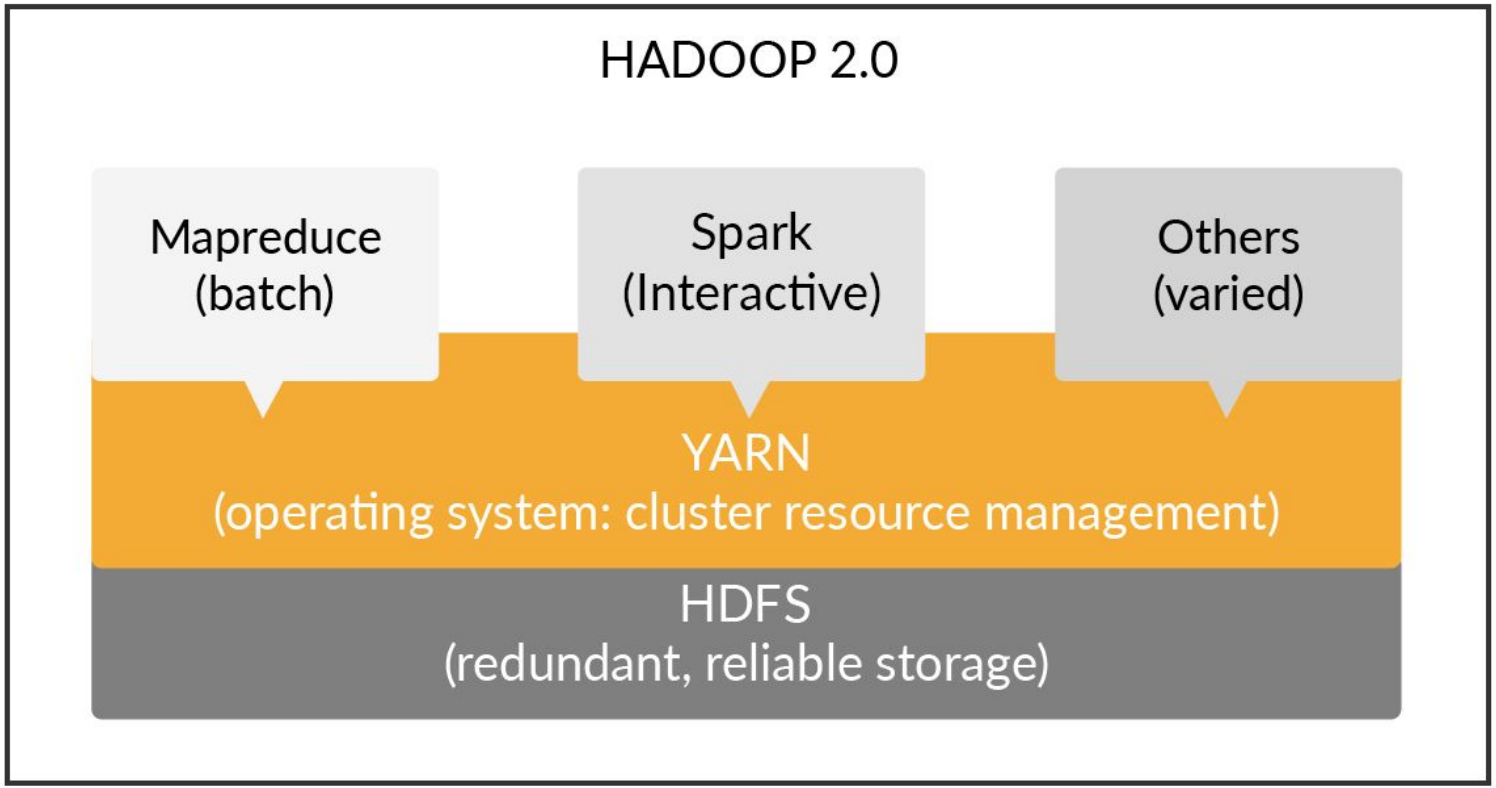
Main motivation for developing Hadoop 2.x



Hadoop 2.x


HADOOP 2.0
Mapreduce
(batch)
Spark
(Interactive)
Others
(varied)
YARN
(operating system: cluster resource management)
HDFS
(redundant, reliable storage)

Hadoop 2.x

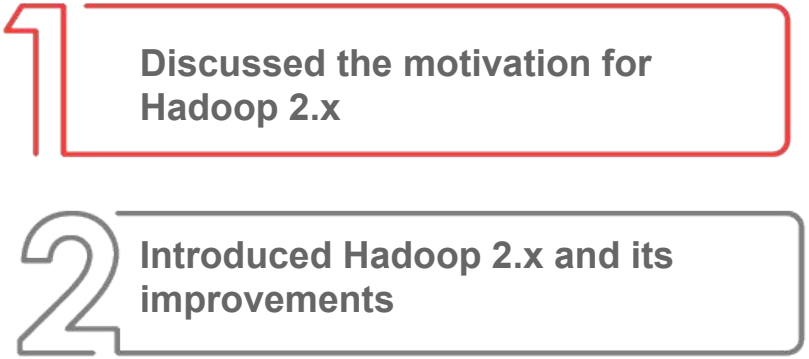

HADOOP 2.0
Mapreduce (batch)
Spark (Interactive)
Others (varied)
YARN
(operating system: cluster resource management)
HDFS
(redundant, reliable storage)

Hadoop 2.x

Improvements in Hadoop 3.x

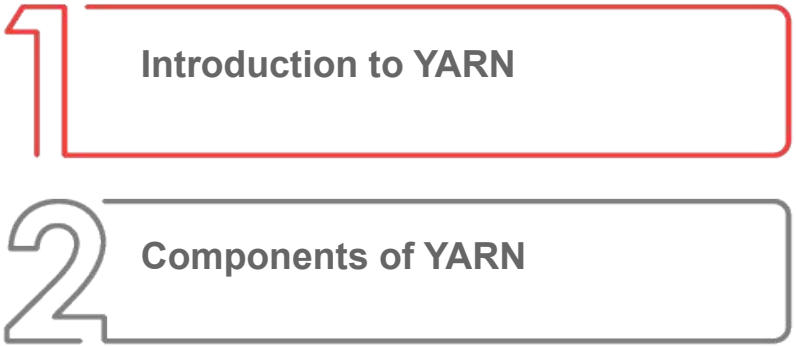
- The runtime version requirement of Java 8
- Erasure coding support
- Improved YARN Timeline Service v.2
- Support for more than two NameNodes
- Support for GPUs in YARN
- Intra-node disk balancing.

Segment Summary



Segment - 07
YARN

Segment Overview



YARN

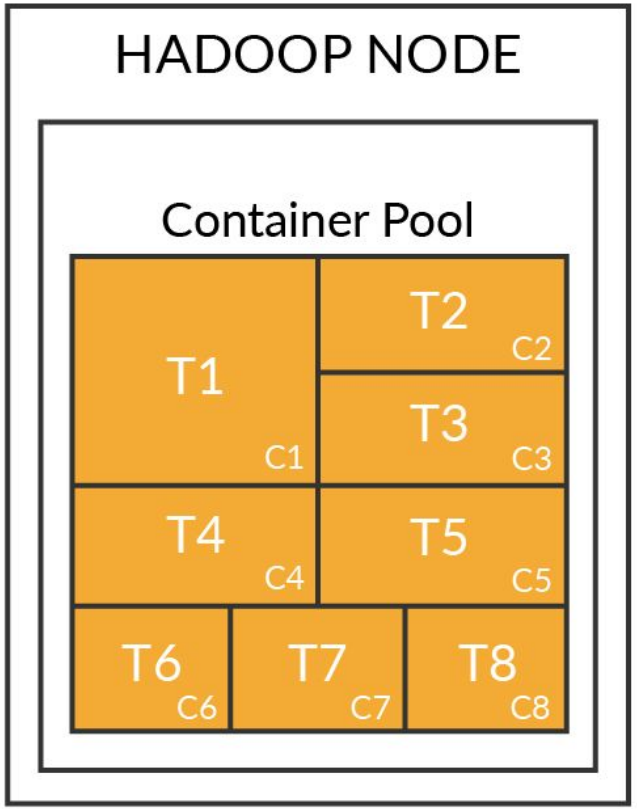
YARN was introduced in Hadoop 2.x mainly to split the responsibilities of the JobTracker in Hadoop 1.x into separate components, thus improving the scalability and reliability of the whole system.

Components of YARN

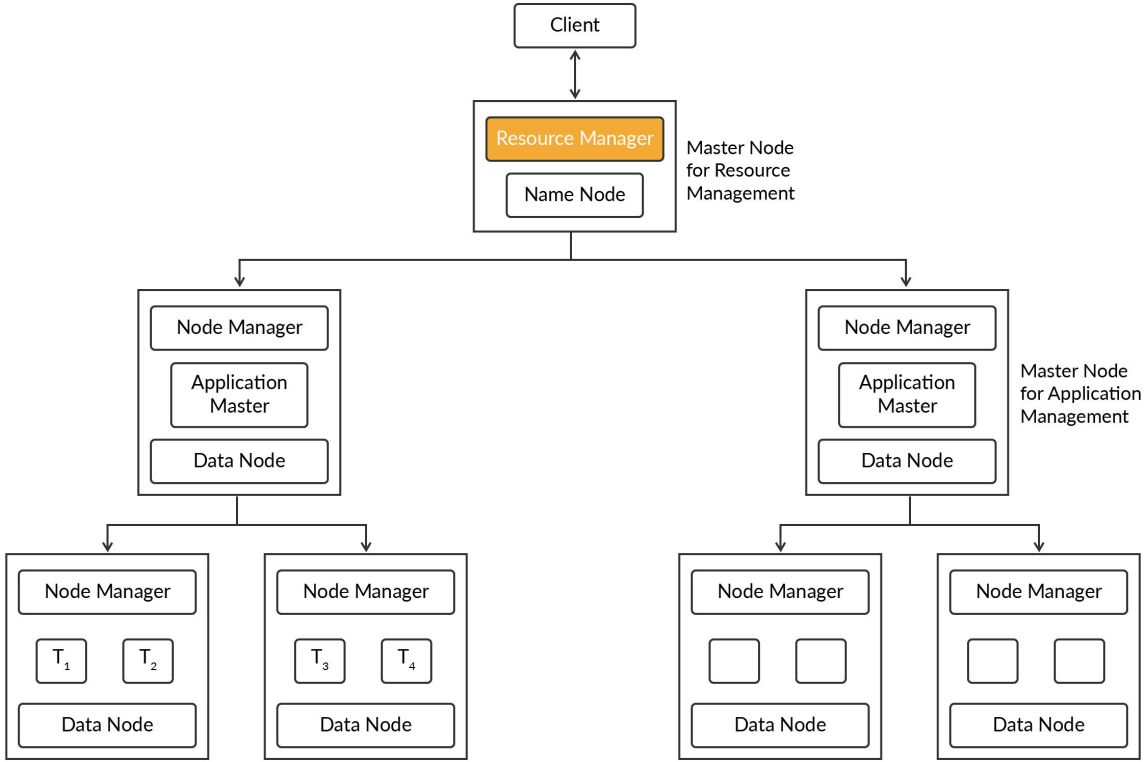
1. Resource Manager
2. Node Manager
3. Application Master
4. Containers

YARN

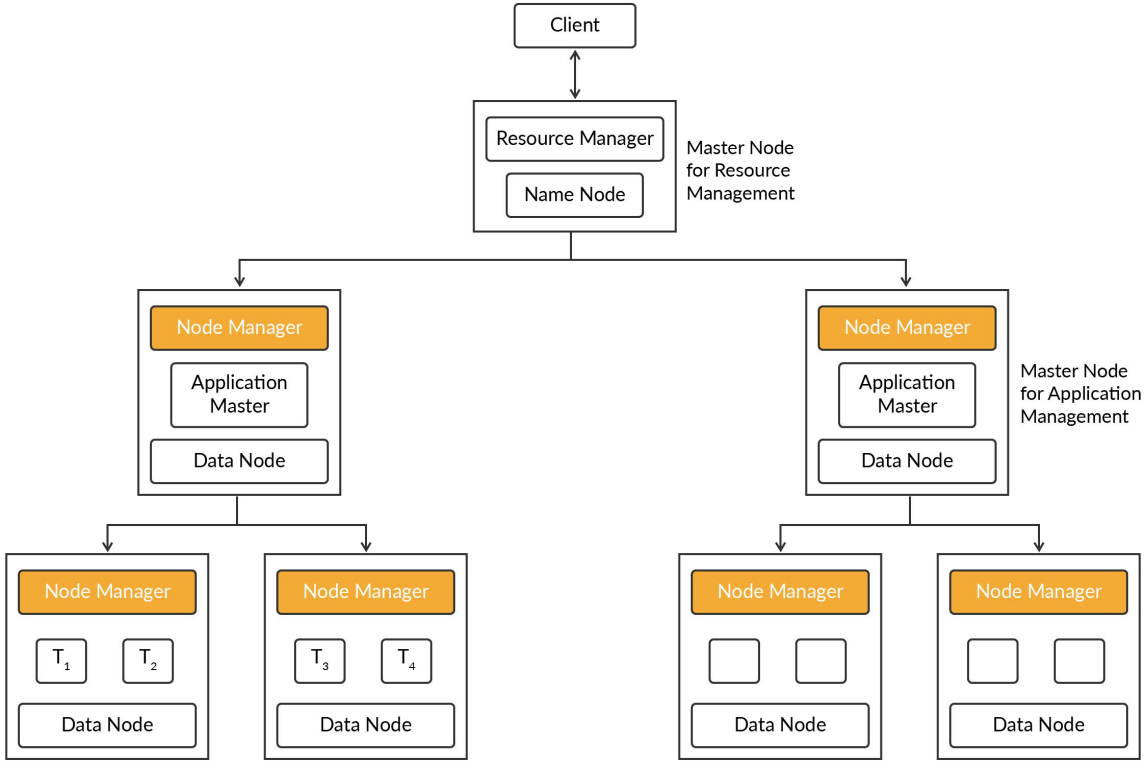
Containers



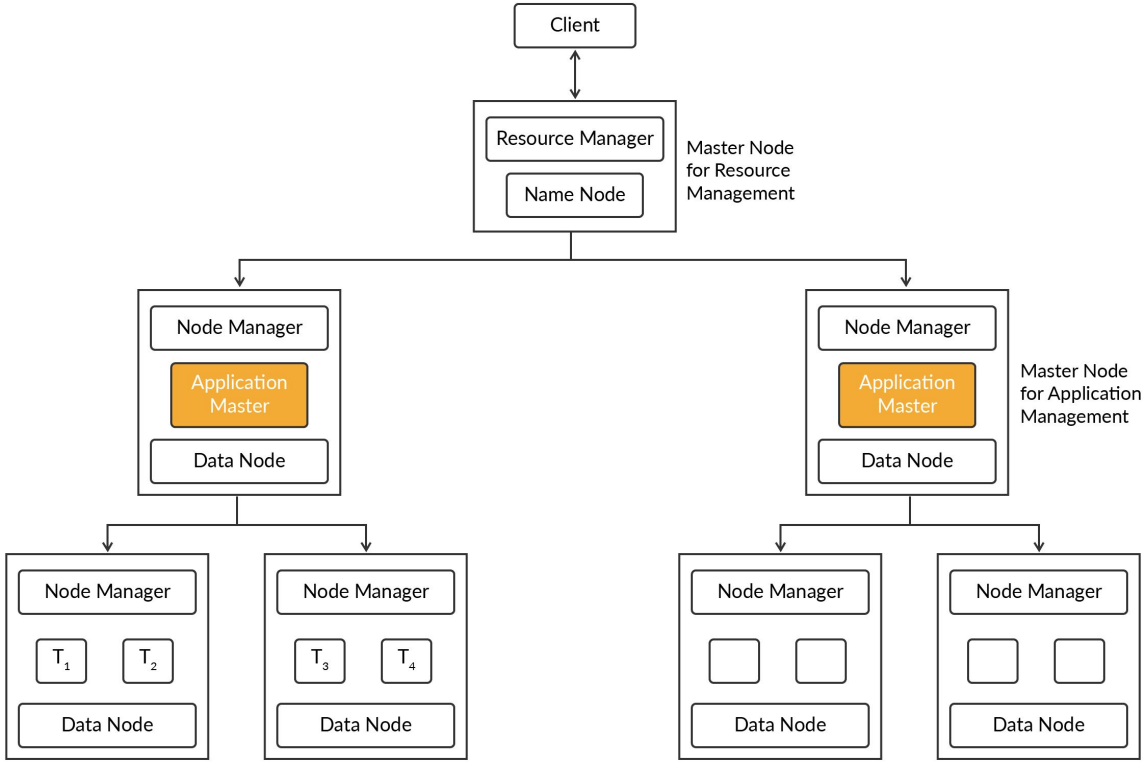
Resource Manager



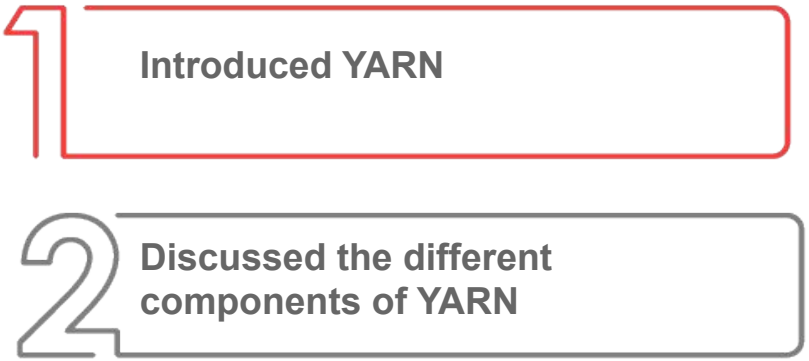
Node Manager



Application Master

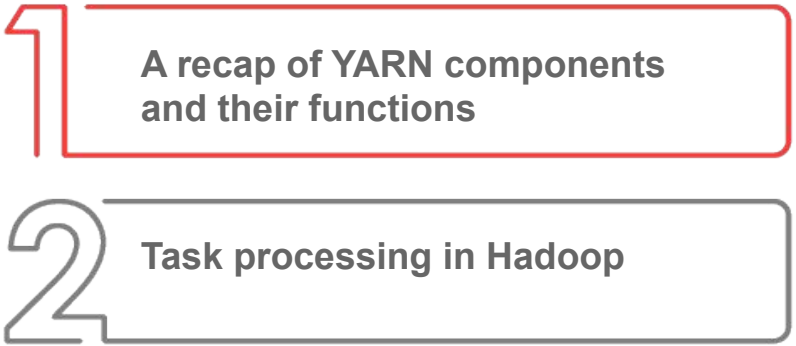


Segment Summary



Segment - 08
Task Processing in Hadoop

Segment Overview



Task Processing in Hadoop

A quick recap of the components of YARN

Resource Manager

- One per cluster
- Runs in the master node

Application Master

- Installed in slave nodes
- Monitors jobs and requests containers from the Scheduler

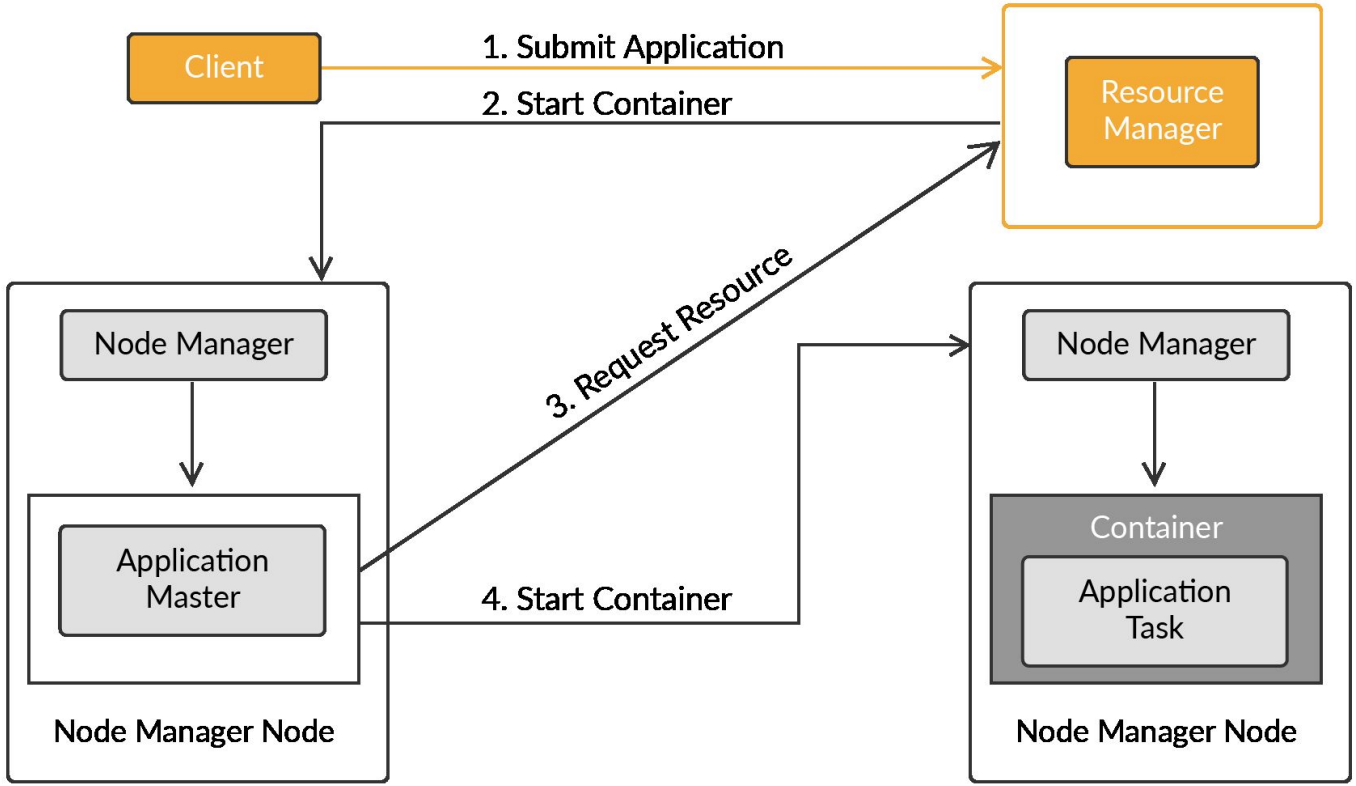
Node Manager

- Installed in slave nodes
- Launches and monitors containers

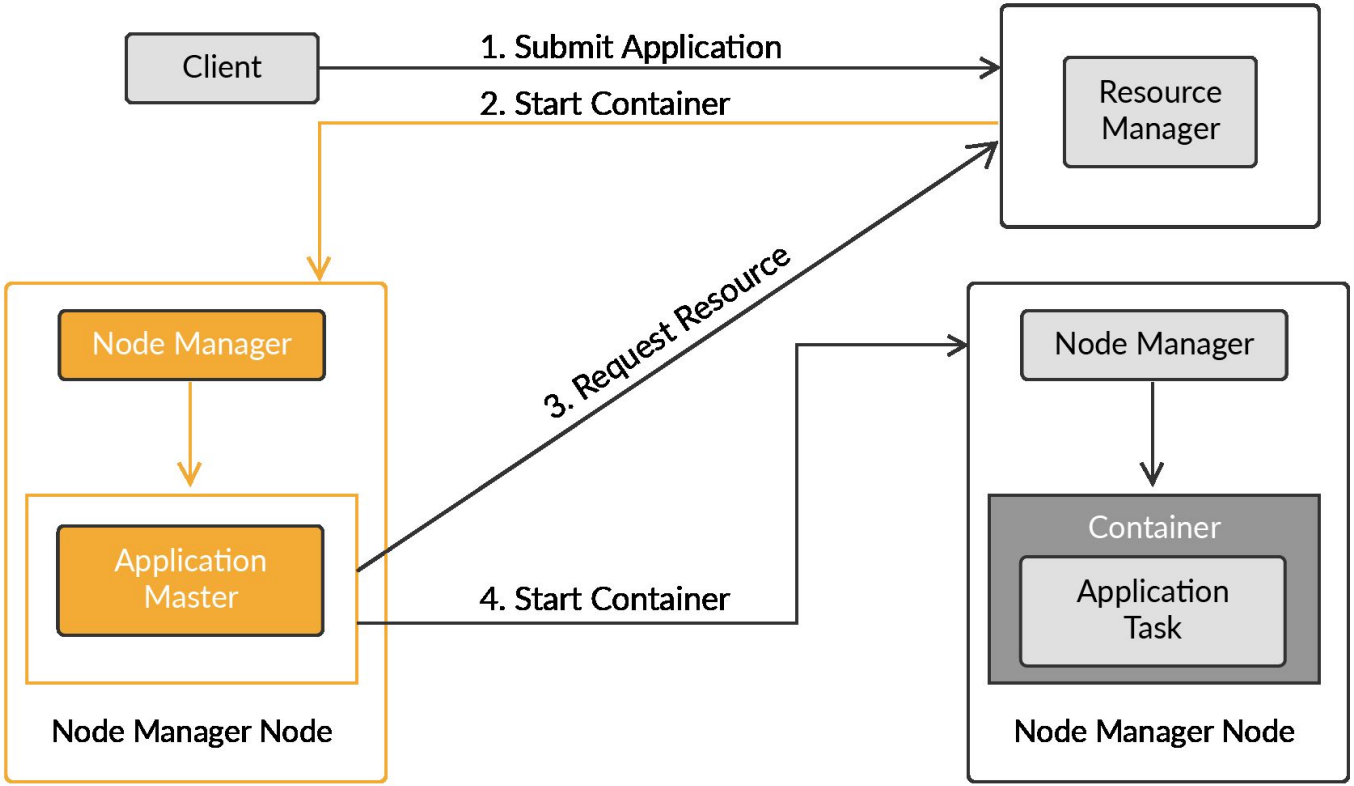
Containers

- Consist of processors and memory hardware required for processing tasks

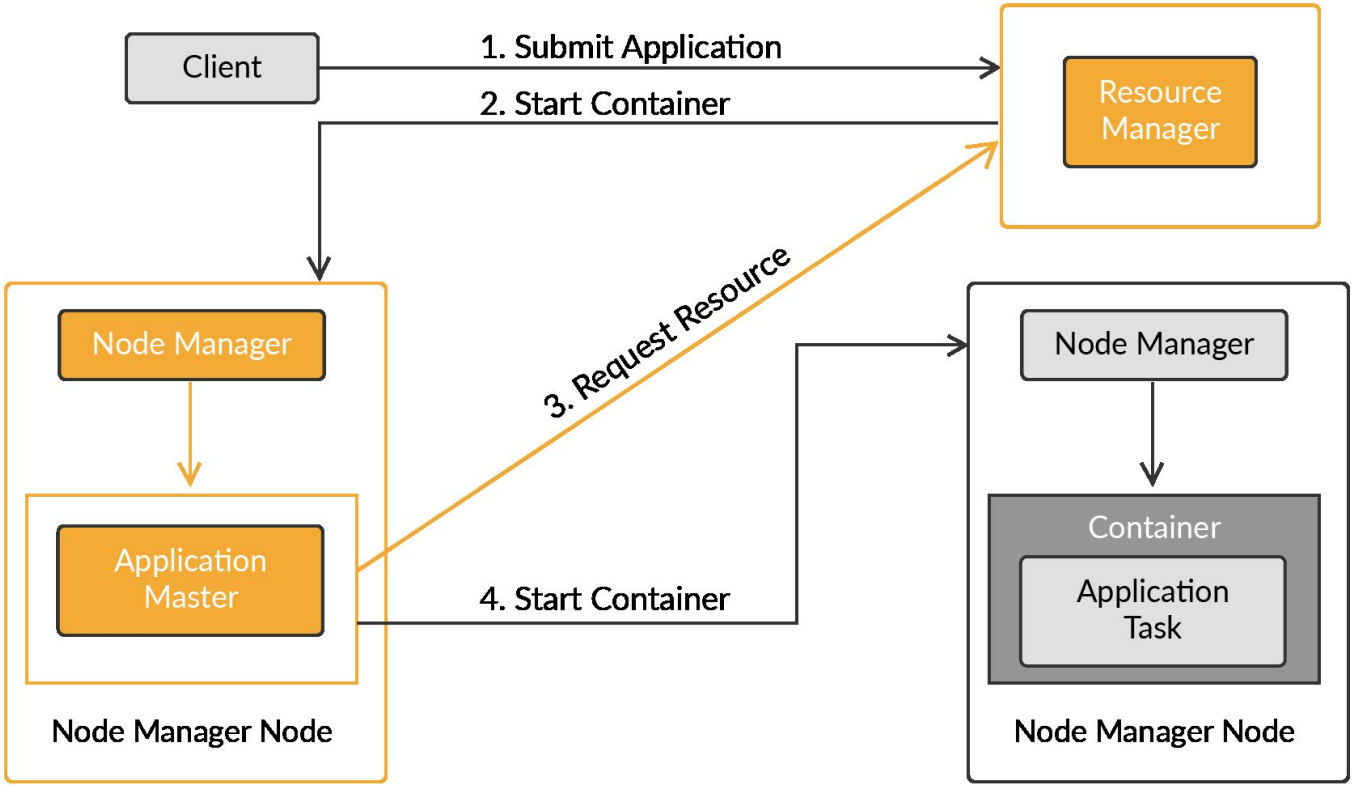
Task Processing in Hadoop



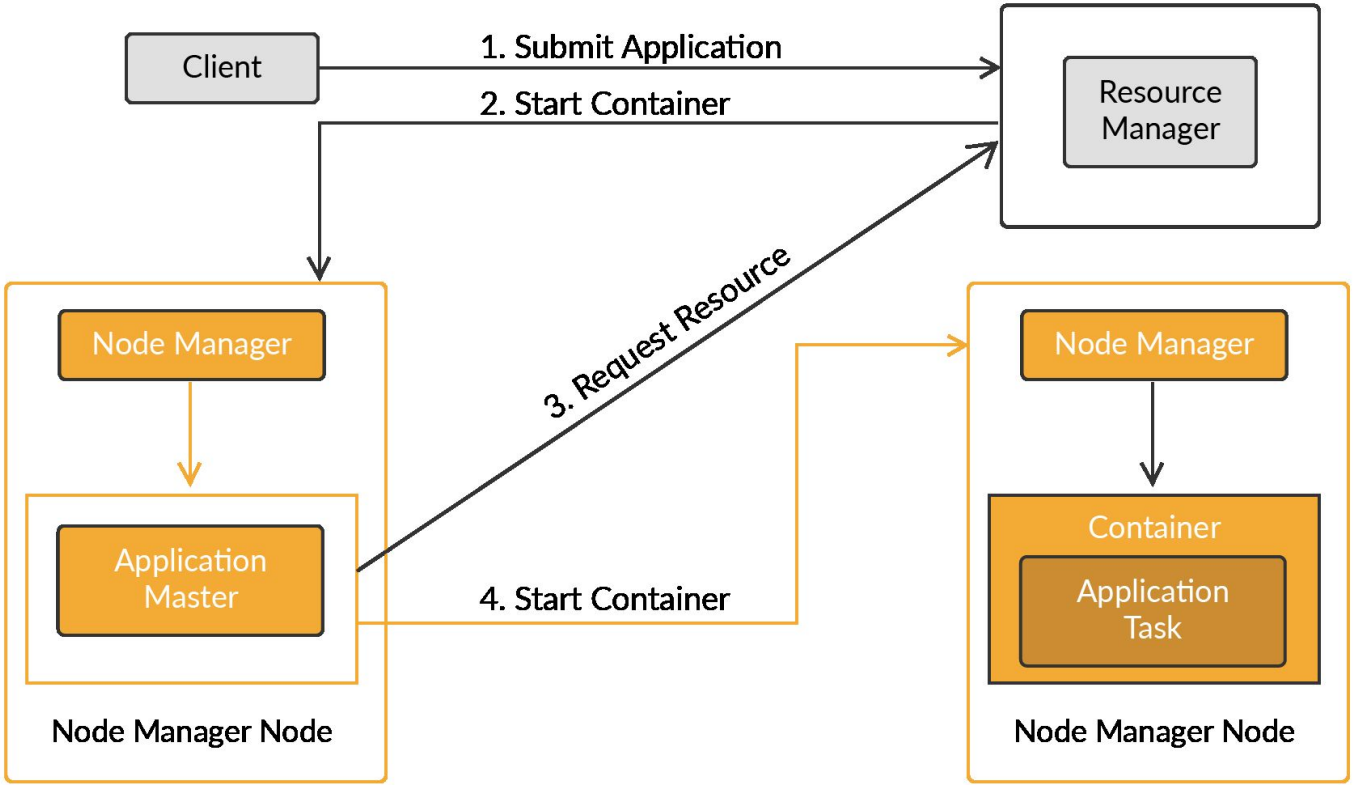
Task Processing in Hadoop



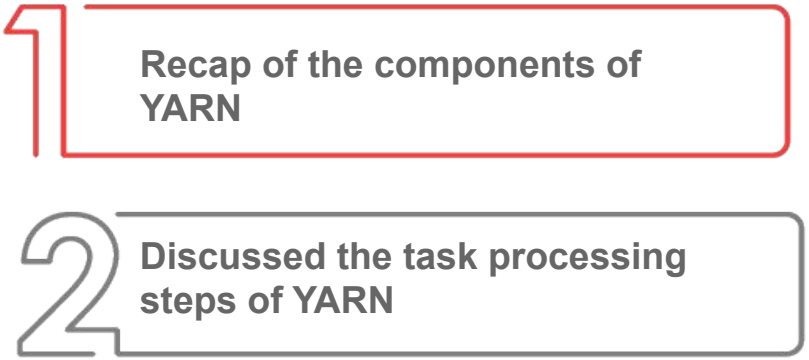
Task Processing in Hadoop



Task Processing in Hadoop



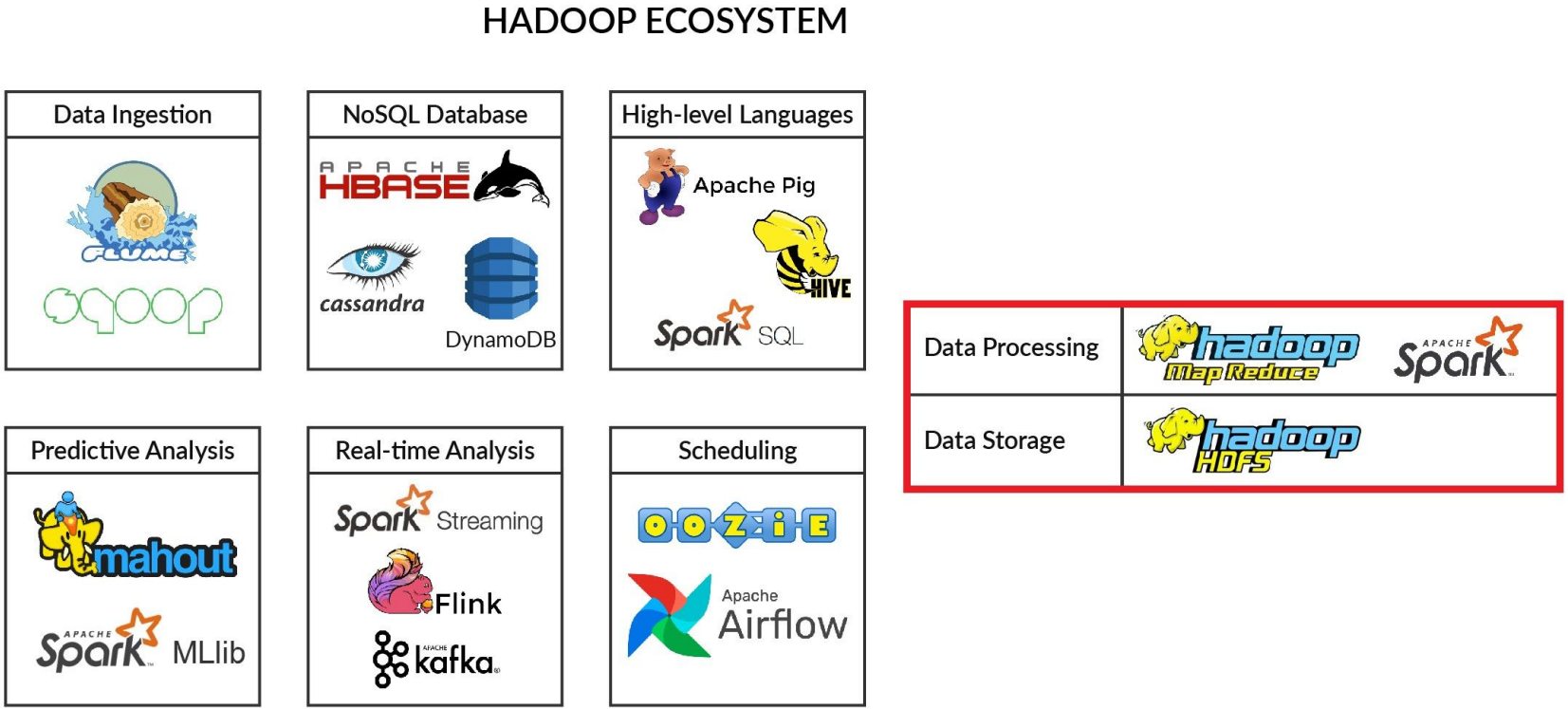
Segment Summary



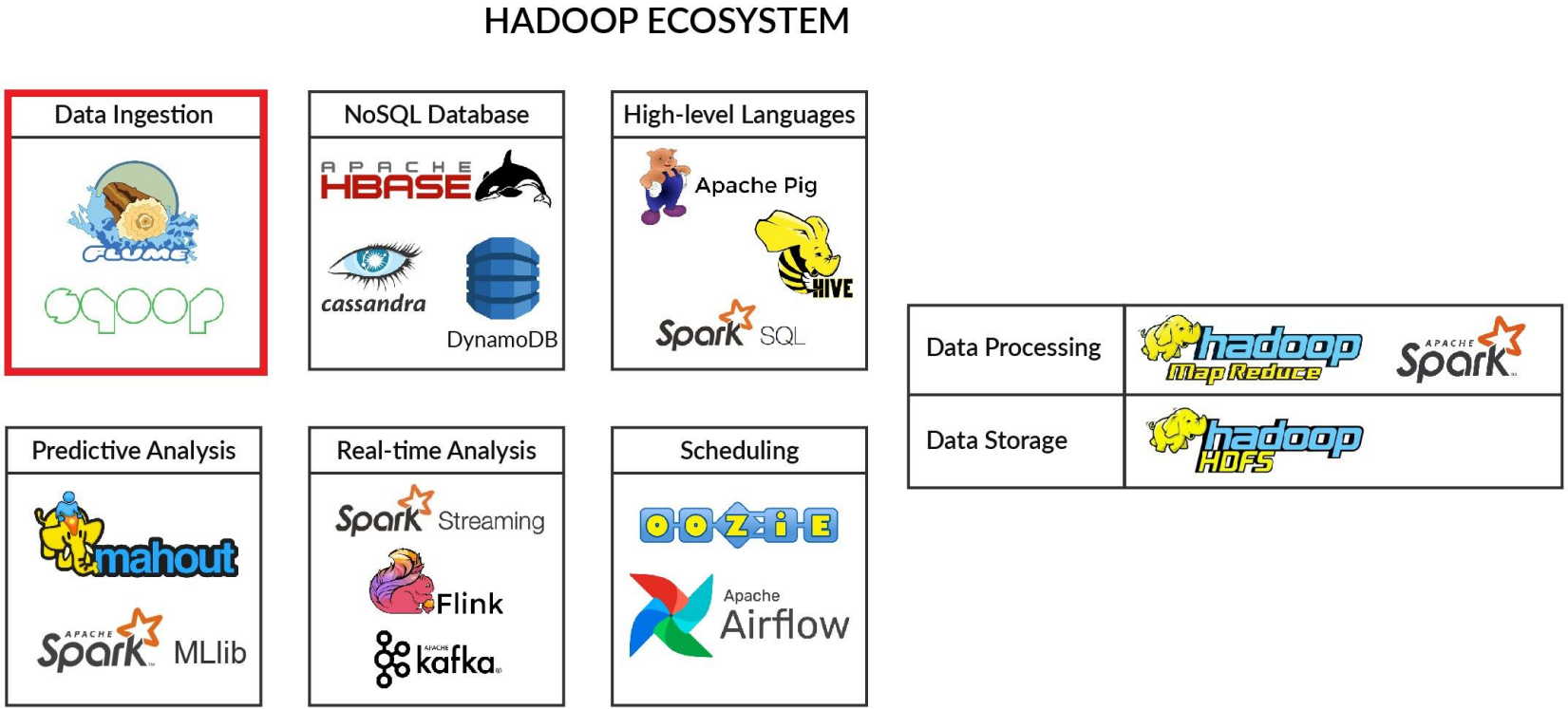
Segment - 09
Tools for Hadoop

Segment Overview

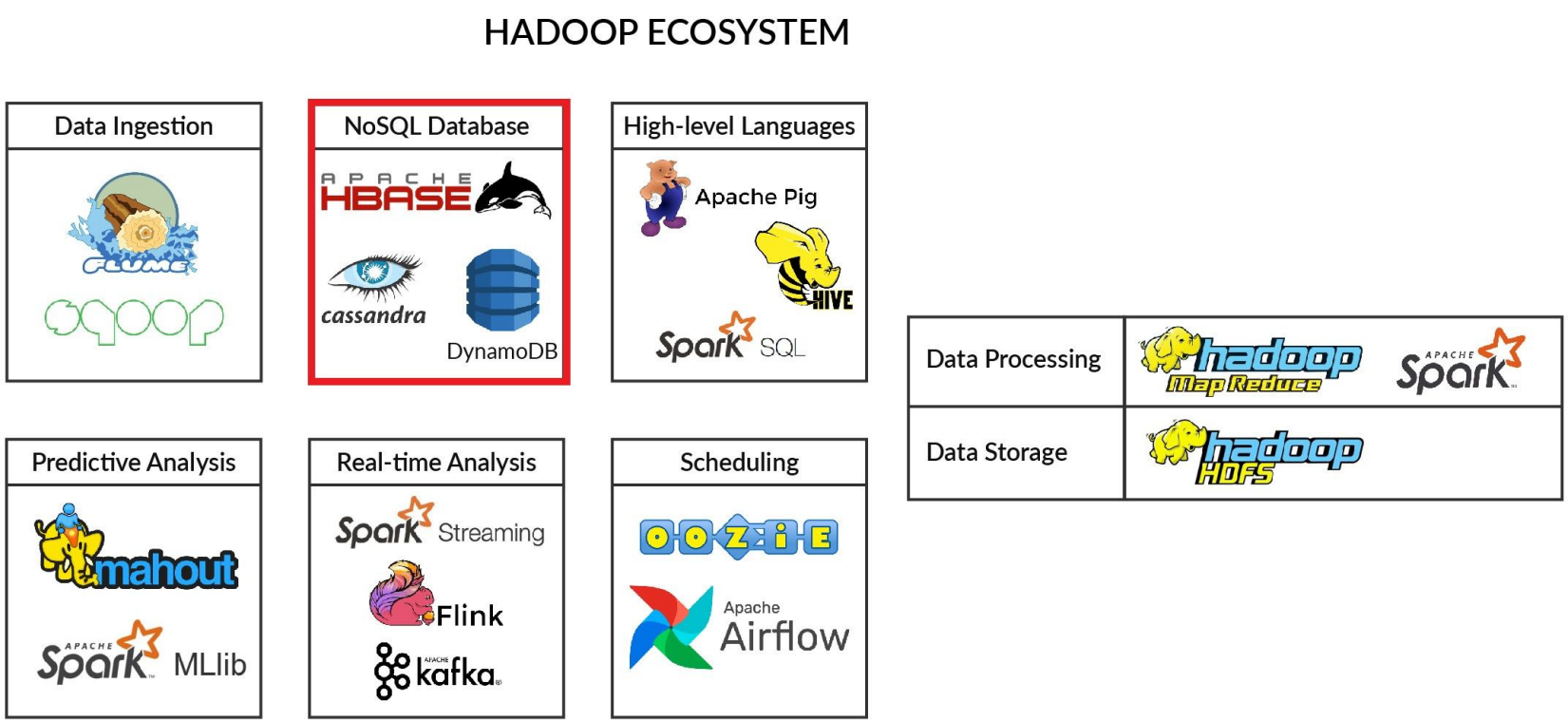
Tools for Hadoop



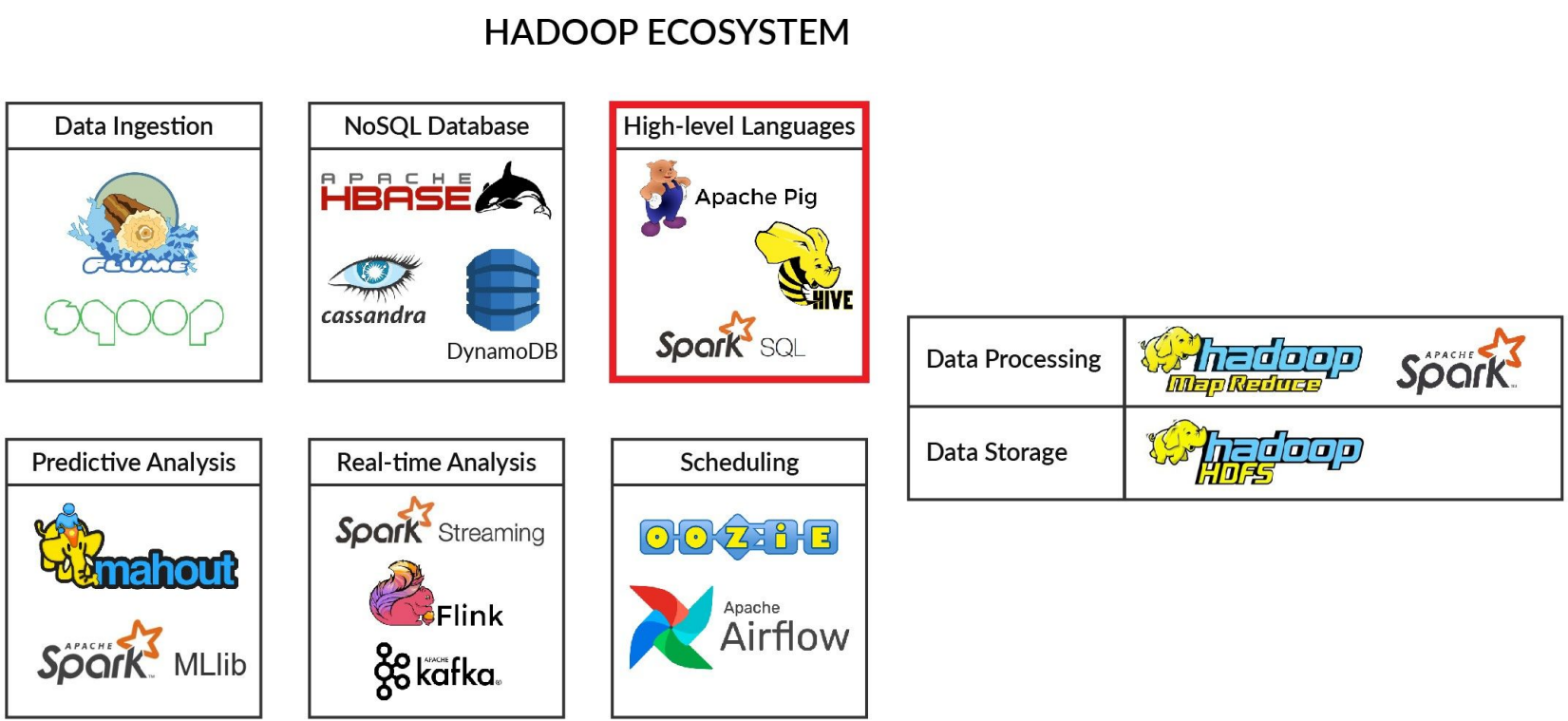
Tools for Hadoop



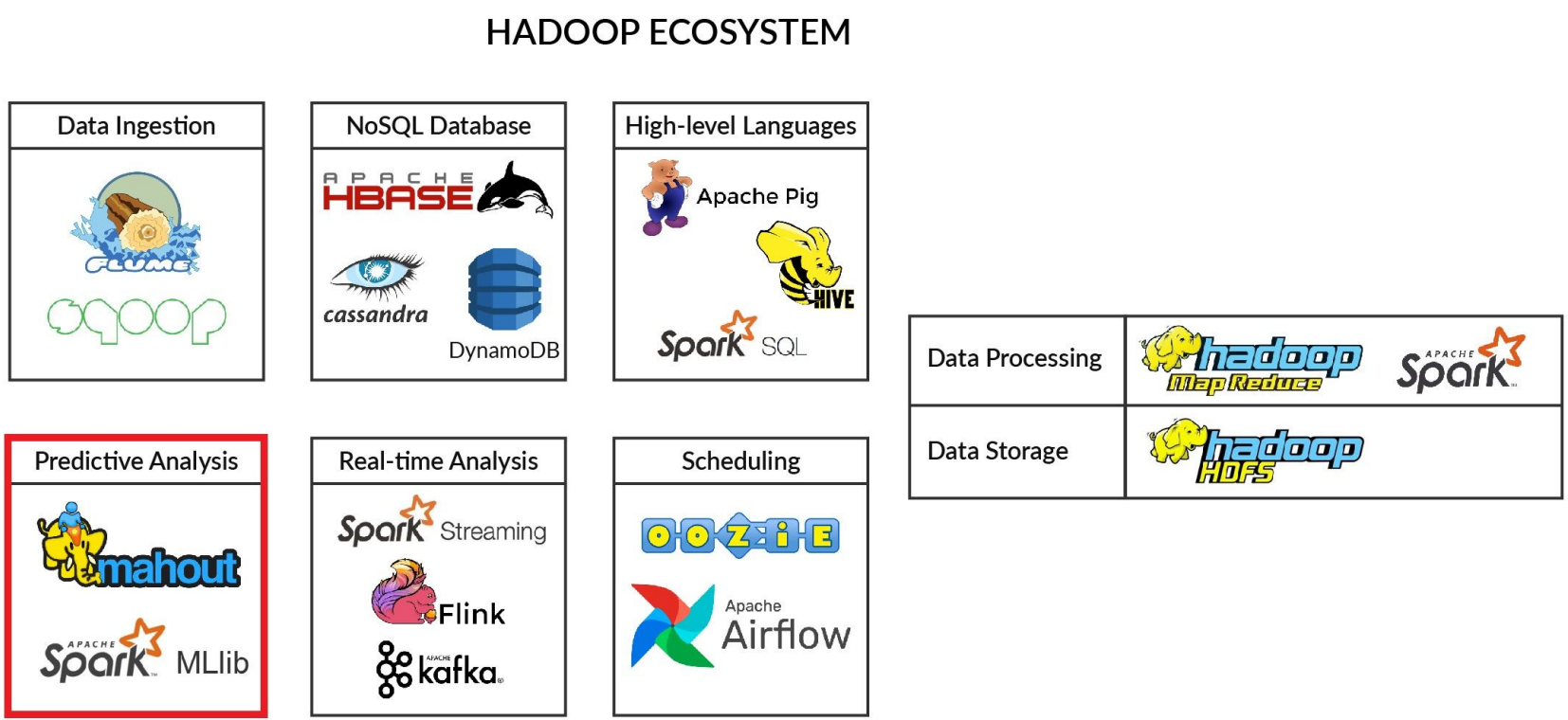
Tools for Hadoop



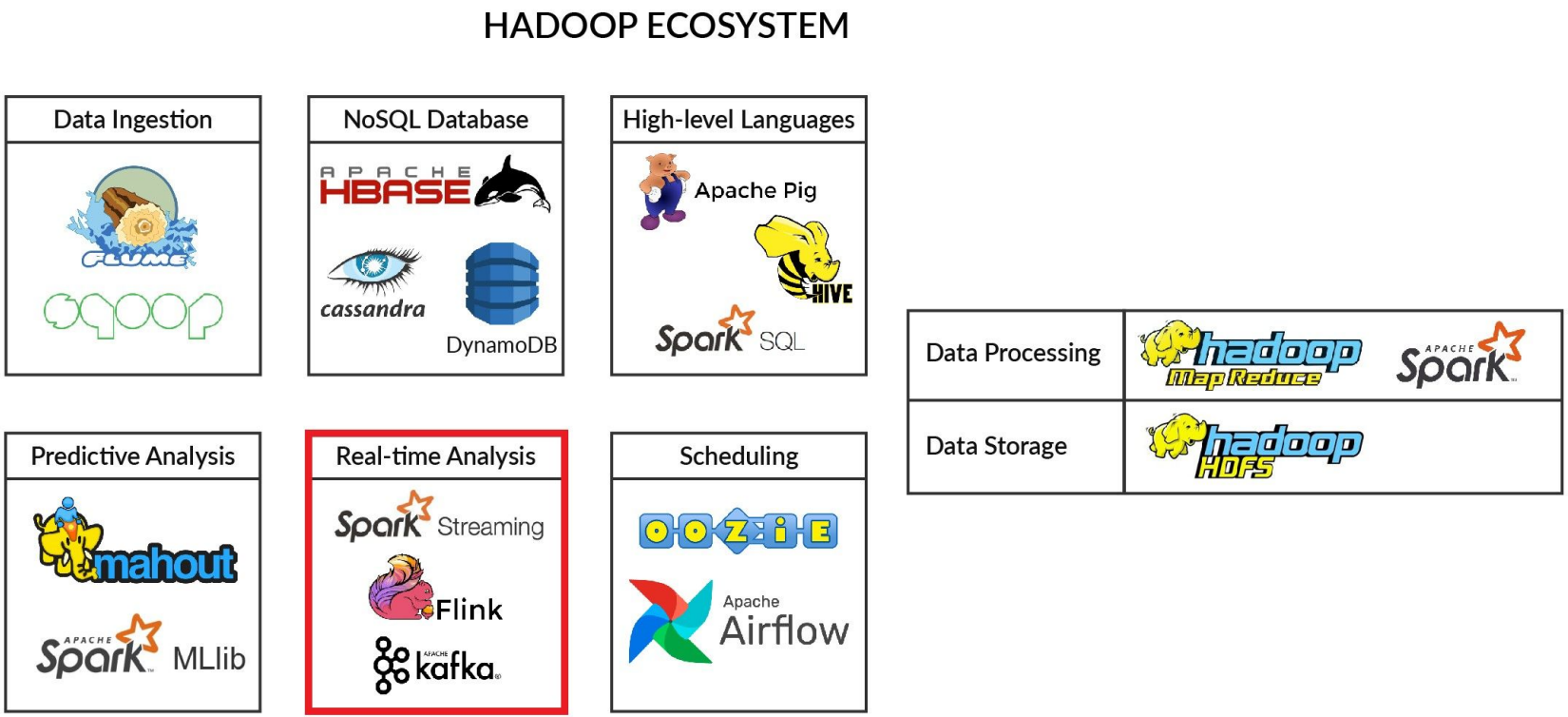
Tools for Hadoop



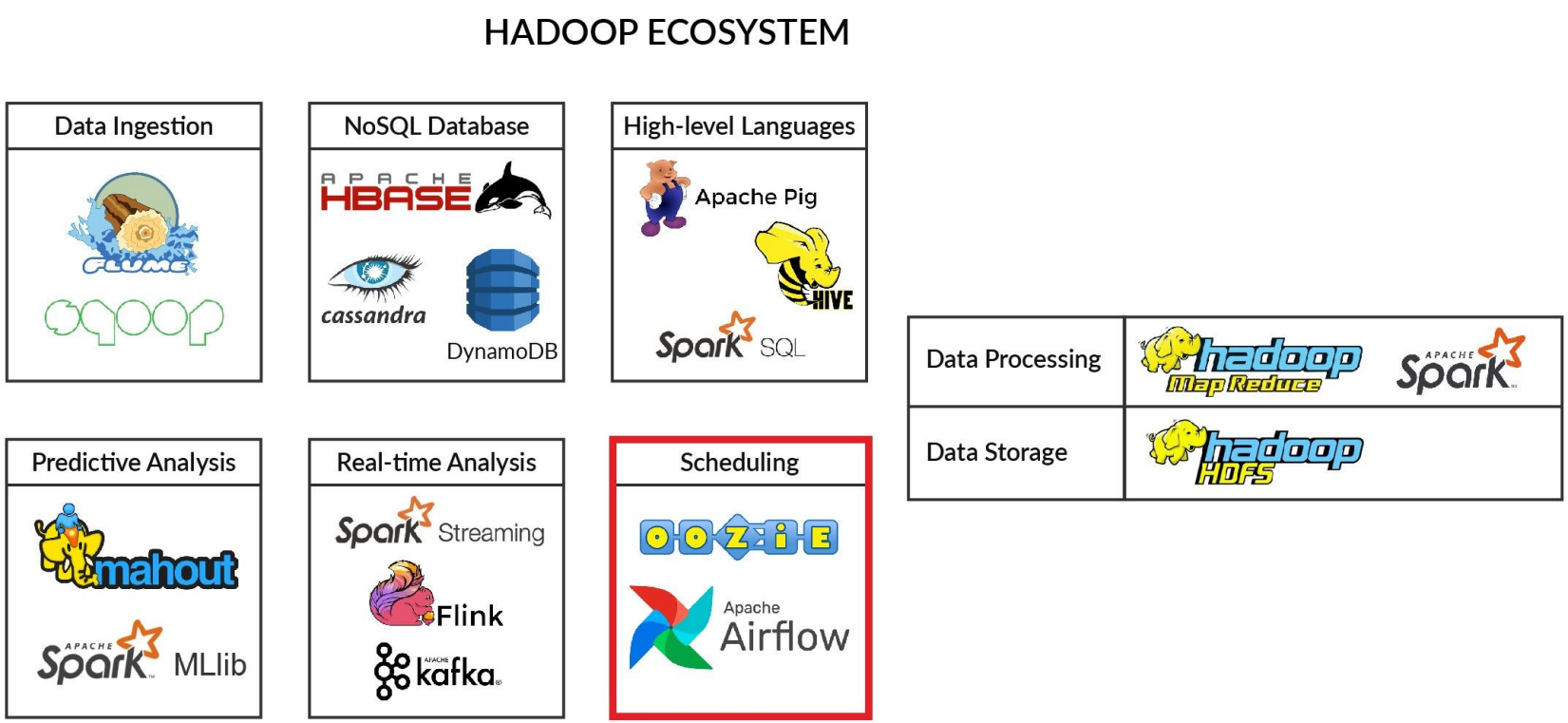
Tools for Hadoop



Tools for Hadoop



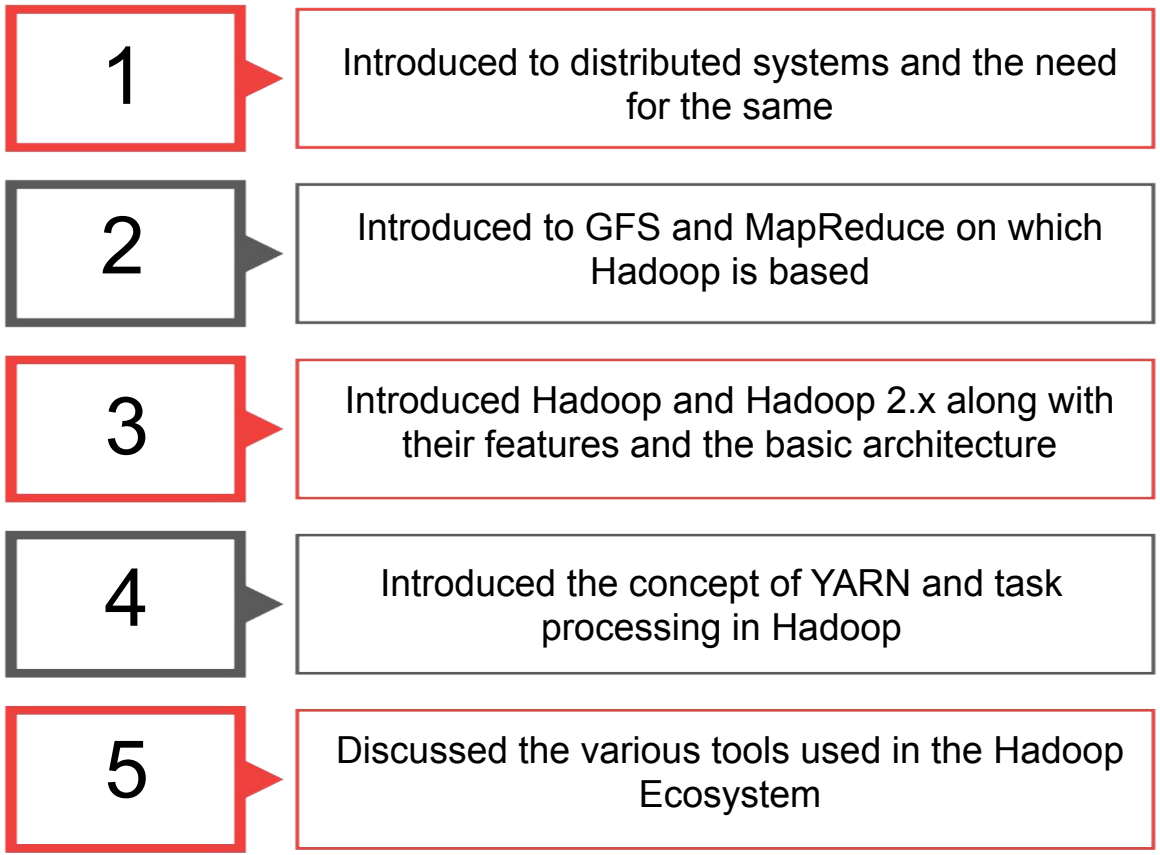
Tools for Hadoop



Segment Summary

1. Discussed the various tools used in the Hadoop Ecosystem

Session Summary



Thank you