Understanding of Hadoop

Agenda

- What is HADOOP
- Big Data and Hadoop
- Characteristics of Hadoop make it a unique platform
- Big Data Hadoop Architecture in detail.
- When to not to use Hadoop ?
- Understanding of Hadoop
- Four modules
- Hadoop Ecosystem: Hadoop Tools for Crunching Big Data
- There are five pillars to Hadoop that make it enterprise ready
- Internal Hadoop Working
- Problems with Traditional Approach
- What is Hadoop?: SQL Comparison
- When to use Hadoop ?
- When to not to use Hadoop ?

What is HADOOP

- **Apache Hadoop** is an **open source**, **Scalable**, and **Fault tolerant** framework written in **Java**. It efficiently processes large volumes of data on a cluster of commodity hardware. Hadoop is not only a storage system but is a platform for large data storage as well as processing.
- **Hadoop** is an open-source tool from the **ASF** – Apache Software Foundation. Open source project means it is freely available and we can even change its source code as per the requirements. If certain functionality does not fulfill your need then you can change it according to your need. Most of Hadoop code is written by *Yahoo, IBM, Facebook, Cloudera.*

What is HADOOP

- It provides an efficient framework for running jobs on multiple nodes of clusters. **Cluster** means a group of systems connected via LAN. Apache Hadoop provides parallel processing of data as it works on multiple machines simultaneously.
- By getting inspiration from **Google**, which has written a paper about the technologies. It is using technologies like **Map-Reduce** programming model as well as its file system (**GFS**).

What is HADOOP

- **Apache Hadoop** is an open source framework written in **Java**. The basic Hadoop programming language is Java, but this does not mean you can code only in Java. You can code in **C, C++, Perl, Python, ruby** etc. You can code the Hadoop framework in any language but it will be more good to code in java as you will have lower level control of the code.

Big Data and Hadoop

- Big Data and Hadoop efficiently processes large volumes of data on a cluster of commodity hardware. Hadoop is for processing huge volume of data. Commodity hardware is the low-end hardware, they are cheap devices which are very economical. Hence, Hadoop is very economic.
- Hadoop can be setup on a single machine (pseudo-distributed mode,but it shows its real power with a cluster of machines. We can scale it to thousand nodes on the fly ie, without any downtime. Therefore, we need not make any system down to add more systems in the cluster.

Understanding of Hadoop

- Apache Hadoop is a layered structure to process and store massive amounts of data
- Apache Hadoop is an open source framework for distributed storage and processing of large sets of data on commodity hardware. Hadoop enables businesses to quickly gain insight from massive amounts of structured and unstructured data. Numerous Apache Software Foundation projects make up the services required by an enterprise to deploy, integrate and work with Hadoop
- It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.
- Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures

Understanding of Hadoop

- Hadoop 2.0 is from the Hadoop 0.23 branch, with major components re-written to enable support for features like High Availability, and MapReduce 2.0 (YARN), and to enable Hadoop to scale out past 4,000 machines per cluster. Specifically, Hadoop 2.0 adds

Four modules

- The Apache Hadoop project is the core of an entire ecosystem of projects. It consists of four modules
- **Hadoop Common**: The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data. **–** It is the storage layer of Hadoop.
- **Hadoop YARN**: A framework for job scheduling and cluster resource management. It is the data processing layer of Hadoop.
- **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets. It is the data processing layer of Hadoop.

There are five pillars to Hadoop that make it enterprise ready:

1: Data Management – Store and process vast quantities of data in a storage layer that scales linearly. Hadoop Distributed File System (HDFS) is the core technology for the efficient scale out storage layer, and is designed to run across low-cost commodity hardware. **HDFS** – Hadoop Distributed File System (HDFS) is a Java-based file system that provides scalable and reliable data storage that is designed to span large clusters of commodity servers.

There are five pillars to Hadoop that make it enterprise ready:

2: Data Access – Interact with your data in a wide variety of ways – from batch to real-time.

- Apache Hive is the most widely adopted data access technology, though there are many specialized engines. For instance, Apache Pig provides scripting capabilities, Apache Storm offers real-time processing, Apache HBase offers columnar NoSQL storage and Apache Accumulo offers cell-level access control. All of these engines can work across one set of data and resources thanks to YARN and intermediate engines such as Apache Tez for interactive access and Apache Slider for long-running applications. YARN also provides flexibility for new and emerging data access methods, such as Apache Solr for search and programming frameworks such as Cascading.

There are five pillars to Hadoop that make it enterprise ready:

3: **Data Governance and Integration** – Quickly and easily load data, and manage according to policy. Workflow Manager provides workflows for data governance, while Apache Flume and Sqoop enable easy data ingestion, as do the NFS and WebHDFS interfaces to HDFS.

- **Workflow Management** – Workflow Manager allows you to easily create and schedule workflows and monitor workflow jobs. It is based on the Apache Oozie workflow engine that allows users to connect and automate the execution of big data processing tasks into a defined workflow.
- **Apache Flume** – Flume allows you to efficiently aggregate and move large amounts of log data from many different sources to Hadoop.
- **Apache Sqoop** – Sqoop is a tool that speeds and eases movement of data in and out of Hadoop. It provides a reliable parallel load for various, popular enterprise data sources.

There are five pillars to Hadoop that make it enterprise ready:

4: Security – Address requirements of Authentication, Authorization, Accounting and Data Protection. Security is provided at every layer of the Hadoop stack from HDFS and YARN to Hive and the other Data Access components on up through the entire perimeter of the cluster via Apache Knox.

- **Apache Knox** – The Knox Gateway ("Knox") provides a single point of authentication and access for Apache Hadoop services in a cluster. The goal of the project is to simplify Hadoop security for users who access the cluster data and execute jobs, and for operators who control access to the cluster.
- **Apache Ranger** – Apache Ranger delivers a comprehensive approach to security for a Hadoop cluster. It provides central security policy administration across the core enterprise security requirements of authorization, accounting and data protection.

There are five pillars to Hadoop that make it enterprise ready:

5: Operations – Provision, manage, monitor and operate Hadoop clusters at scale.

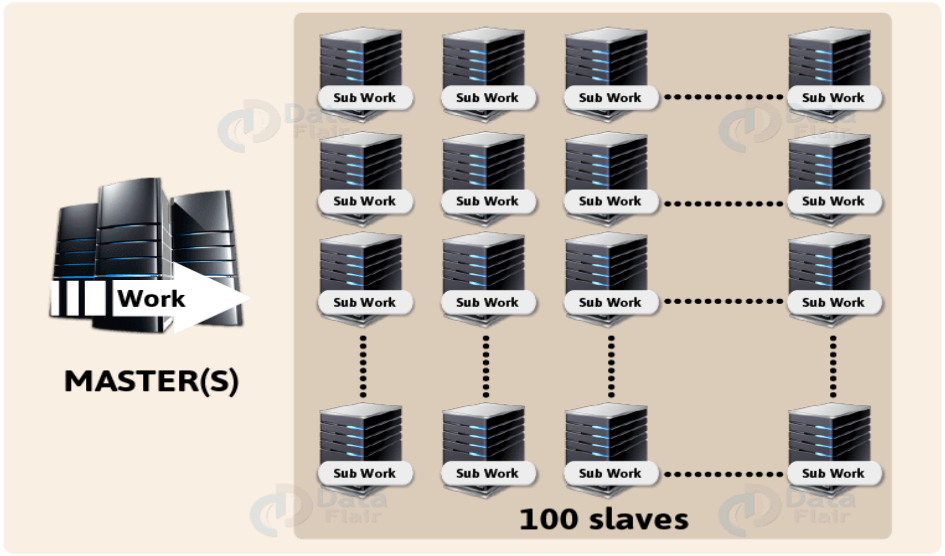
- **Apache Ambari** – An open source installation lifecycle management, administration and monitoring system for Apache Hadoop clusters.
- **Apache Oozie** – Oozie Java Web application used to schedule Apache Hadoop jobs. Oozie combines multiple jobs sequentially into one logical unit of work.
- **Apache ZooKeeper** – A highly available system for coordinating distributed processes. Distributed applications use ZooKeeper to store and mediate updates to important configuration information.

- Apache Hadoop can be useful across a range of use cases spanning virtually every vertical industry. It is becoming popular anywhere that you need to store, process, and analyze large volumes of data. Examples include digital marketing automation, fraud detection and prevention, social network and relationship analysis, predictive modeling for new drugs, retail in-store behavior analysis, and mobile device location-based marketing. To learn more about Apache Hadoop, watch the Video by following:
- https://youtu.be/6UtD53BzDNk

Characteristics of Hadoop make it a unique platform

- Flexibility to store and mine any type of data whether it is structured, semi-structured or unstructured. It is not bounded by a single schema.
- Excels at processing data of complex nature. Its scale-out architecture divides workloads across many nodes. Another added advantage is that its flexible file-system eliminates ETL bottlenecks.
- Scales economically, as discussed it can deploy on commodity hardware. Apart from this its open-source nature guards against vend

Big Data Hadoop Architecture in detail.

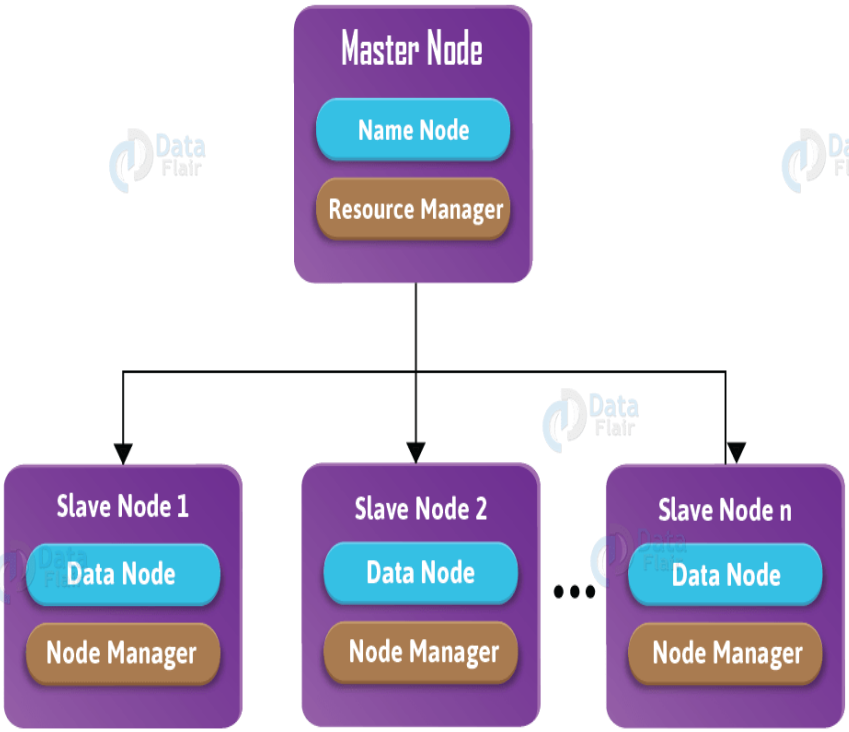


Hadoop works in **master-slave** fashion. There is a master node and there are n numbers of slave nodes where n can be 1000s. Master manages, maintains and monitors the slaves while slaves are the actual worker nodes. In Hadoop architecture the Master should deploy on good configuration hardware, not just commodity hardware. As it is the centerpiece of **Hadoop cluster**.

Master stores the metadata (data about data) while slaves are the nodes which store the data. Distributedly data stores in the cluster. The client connects with master node to perform any task.

Hadoop Daemons

- Daemons are the processes that run in the background. There are mainly 4 daemons which run for Hadoop.



Namenode – It runs on master node for HDFS.
Datanode – It runs on slave nodes for HDFS.
ResourceManager – It runs on master node for Yarn.
NodeManager – It runs on slave node for Yarn.
These 4 demons run for Hadoop to be functional. Apart from this, there can be secondary NameNode, standby NameNode, Job HistoryServer, etc.

Big Data Hadoop Architecture in detail.

- Master stores the metadata (data about data) while slaves are the nodes which store the data. Distributedly data stores in the cluster. The client connects with master node to perform any task.

Internal Hadoop Working

- Before learning how hadoop works, let us brush our **Hadoop Skills.**
- There are 2 layers in Hadoop – **HDFS** layer and **Map-Reduce** layer and 5 daemons which run on Hadoop in these 2 layers. Daemons are the processes that run in the background. The Hadoop Daemons are:-
- **a) Namenode** – It runs on master node for HDFS.
- **b) Datanode** – It runs on slave nodes for HDFS.
- **c) Resource Manager** – It runs on **YARN** master node for MapReduce.
- **d) Node Manager** – It runs on YARN slave node for MapReduce.
- **e) Secondary Namenode** – It is backup for namenode and runs on a different system (other than master and slave nodes. One can also configure it on the slave node.)
- These 5 daemons run for Hadoop to be functional.
- **HDFS** provides the storage layer and MapReduce provides the computation layer in Hadoop. There are 1 namenode and several datanodes on storage layer ie HDFS. Similarly there is a resource manager and several node managers on computation layer ie MapReduce.
- Namenode (HDFS) and resource manager (Map-Reduce) run on master while datanodes (HDFS) and node manager (Map-Reduce) run on slaves.

How Hadoop Works?

- Hadoop does distributed processing for huge data sets across the cluster of commodity servers and works on multiple machines simultaneously. To process any data, the client submits data and program to Hadoop. **HDFS** stores the data while **Mapreduce** process the data.
- As we know, **HDFS** is the storing element of Hadoop. There are 2 daemons that run for HDFS:
- **Namenode** runs on the master node.
- **Datanode** runs on slaves.

How Hadoop Works? (countin..)

- Namenode daemon stores the meta data while datanode daemons store the actual data.
- The data is broken into small chunks called as **blocks** and these blocks are stored distributedly on different nodes in the cluster. Each block is replicated as per the replication factor (By default 3).
- Namenode daemon stores the meta data while datanode daemons store the actual data.
- The data is broken into small chunks called as **blocks** and these blocks are stored distributedly on different nodes in the cluster. Each block is replicated as per the replication factor (By default 3).
- Let us now understand how data is processed in Hadoop.

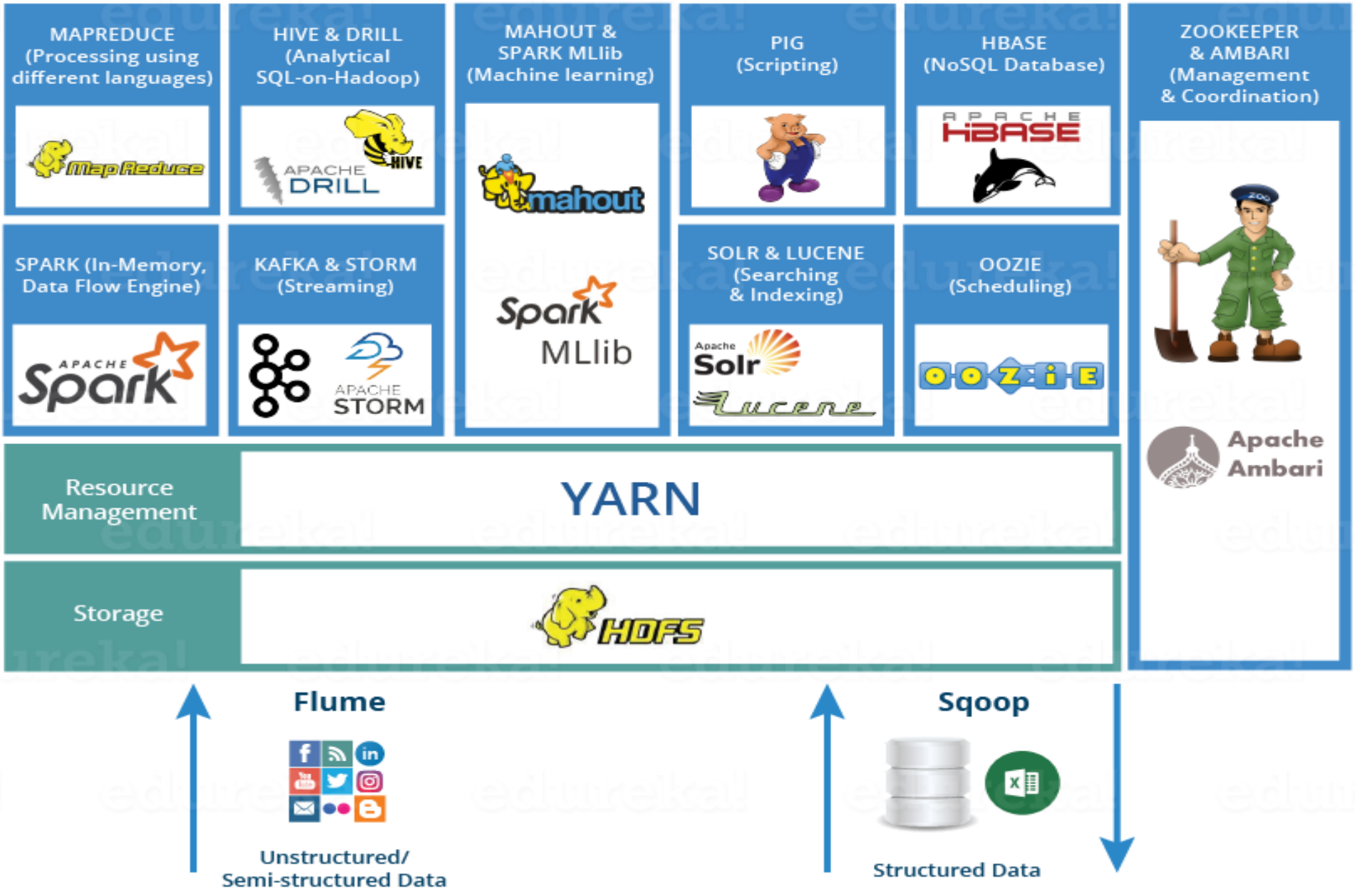
How Hadoop Works? (countin..)

- **Map Reduce** is the processing layer of Hadoop. It has 2 daemons:
- **Resource manager** that splits the job submitted by the client into small tasks.
- **Node manager** that actually do the tasks in parallel in a distributed manner on data stored in datanodes.
- To process the data, the client needs to submit the algorithm to the master node. Hadoop works on the principle of **data locality** ie. Instead of moving data to the algorithm, the algorithm is moved to datanodes where data is stored.

Let us summarize how Hadoop works step by step:

- Input data is broken into blocks of size **128 Mb** and then blocks are moved to different nodes.
- Once all the blocks of the data are stored on data-nodes, the user can process the data.
- Resource Manager then schedules the program (submitted by the user) on individual nodes.
- Once all the nodes process the data, the output is written back to HDFS

Hadoop Ecosystem: Hadoop Tools for Crunching Big Data



Hadoop Ecosystem Components

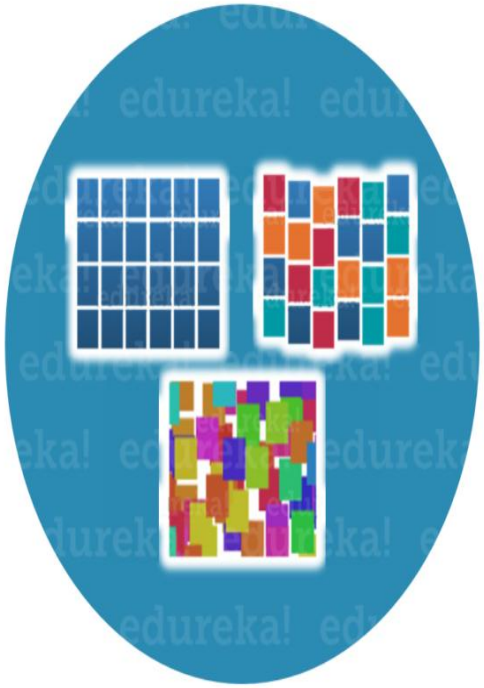
- **Hadoop HDFS** – Distributed storage layer for Hadoop.
- **Yarn Hadoop** – Resource management layer introduced in Hadoop 2.x.
- **Hadoop Map-Reduce** – Parallel processing layer for Hadoop.
- **HBase** – It is a column-oriented database that runs on top of HDFS. It is a NoSQL database which does not understand the structured query. For sparse data set, it suits well.
- **Hive** – Apache Hive is a data warehousing infrastructure based on Hadoop and it enables easy data summarization, using SQL queries.
- **Pig** – It is a top-level scripting language. As we use it with Hadoop. Pig enables writing complex data processing without Java programming.
- **Flume** – It is a reliable system for efficiently collecting large amounts of log data from many different sources in real-time.
- **Sqoop** – It is a tool design to transport huge volumes of data between Hadoop and RDBMS.
- **Oozie** – It is a Java Web application uses to schedule Apache Hadoop jobs. It combines multiple jobs sequentially into one logical unit of work.
- **Zookeeper** – A centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.
- **Mahout** – A library of scalable machine-learning algorithms, implemented on top of Apache Hadoop and using the MapReduce paradigm.

Problems with Traditional Approach

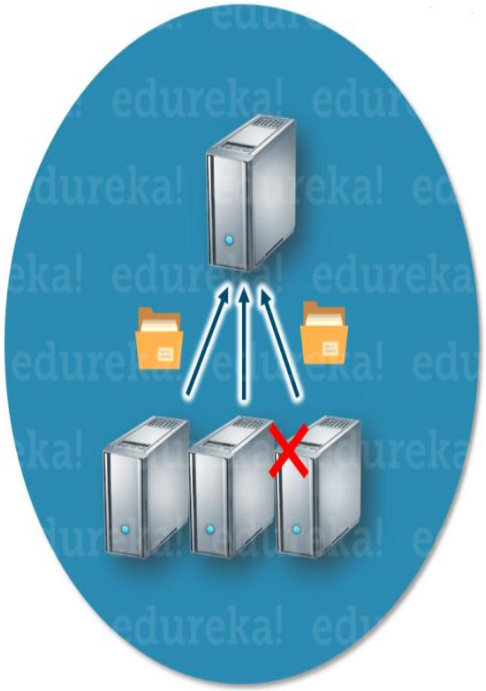
- In traditional approach, the main issue was handling the heterogeneity of data i.e. structured, semi-structured and unstructured. The RDBMS focuses mostly on structured data like banking transaction, operational data etc. and Hadoop specializes in semi-structured, unstructured data like text, videos, audios, Facebook posts, logs, etc.
- RDBMS technology is a proven, highly consistent, matured systems supported by many companies. While on the other hand, Hadoop system technology is developed and is in demand due to Big Data, which mostly consists of unstructured data in different formats.
- Now let us understand what are the major problems associated with Big Data. So that, moving ahead we can understand how Hadoop emerged as a solution.

Problems with Traditional Approach

Storing huge and exponentially growing datasets



Processing data having complex structure (structured, un-structured, semi-structured)



Bringing huge amount of data to computation unit becomes a bottleneck

Problems with Traditional Approach

- ***So, the first problem is storing the colossal amount of data.*** Storing this huge data in a traditional system is not possible. The reason is obvious the storage will be limited to one system and the data is increasing in tremendous rate.
- ***Second problem is storing heterogeneous data.*** Now we know storing is a problem, but let me tell you it is just one part of the problem. Since we discussed that the data is not only huge, but it is present in various formats as well like: Unstructured, Semi-structured and Structured. So, you need to make sure that you have a system to store these varieties of data, generated from various sources.
- ***Now, let's focus on third problem, which is accessing and processing speed***. The hard disk capacity is increasing but disk transfer speed or the access speed is not increasing at similar rate. Let me explain you this with an example: If you have only one 100mbps I/O channel and you are processing say 1TB of data, it will take around 2.91 hours. Now, if you have four machines with four I/O channel for the same amount of data, then it will take 43 minutes approx. Thus for me, accessing and processing speed is the bigger problem than storage of Big Data.
- Before understanding what is Hadoop, let us first look at the evolution of Hadoop over a period of time.

What is Hadoop?: SQL Comparison

- https://www.youtube.com/watch?v=MfF750YVDxM

When to use Hadoop ?

- *Search* – Yahoo, Amazon, Zvents
- *Log processing* – Facebook, Yahoo
- *Data Warehouse* – Facebook, AOL
- *Video and Image Analysis* – New York Times, Eyealike
- HealthCare
- Live Steaming
- Votes casting and ETC

When to not to use Hadoop ?

- Following are some of those scenarios :
- *Low Latency data access* : Quick access to small parts of data
- *Multiple data modification* : Hadoop is a better fit only if we are primarily concerned about reading data and not writing data.
- *Lots of small files* : Hadoop is a better fit in scenarios, where we have few but large files.

Conclusion

- we can say that Apache Hadoop is the most popular and powerful big data tool. Big Data stores huge amount of data in the distributed manner and processes the data in parallel on a cluster of nodes. It provides world's most reliable storage layer- HDFS. Batch processing engine MapReduce and Resource management layer- YARN. 4 daemons (NameNode, datanode, node manager, resource manager) run in Hadoop to ensure Hadoop functionality.

Reference

- https://www.youtube.com/watch?v=MfF750YVDxM
- https://www.youtube.com/watch?v=mafw2-CVYnA
- https://data-flair.training/blogs/big-data-healthcare-real-world-use-cases/