**Hadoop Learning: -**

The main problem with the data is the storage and processing.

**File System: -**

Initially data was stored in files which was not structured.

Disadvantages of File System data management: -

Data Redundancy: -

Since flat file databases rely on files that contain records as text without any structural data, they cannot relate data from one file to another. For example, if one file contains an address record of Mr. Johnson, another file that uses address information of Mr. Johnson has to recreate that data. The second file must duplicate the data. This means that the address data on Mr. Johnson exists on two files at once. On large scales this leads to data redundancy that can quickly take up space in the database and prove cost in-efficient.

Limited User Access: -

Flat-file systems usually do not support access for multiple users. This means that multiple users at different workstations cannot access the same data simultaneously limiting access to important data if multiple users search for the same data at the same time.

**RDMS: -**

Then we stored the data in RDMS (Relational Database Management System). RDMS are widely used in many industries to store financial records, keep track of inventory and to keep records on employees. In a relational database, information is stored in tables (often called relations) which help organize and structure data. Even though they are widely used relational databases have some drawbacks.

Disadvantages of Relational Database: -

Cost: - Setting up and maintaining and RDMS can be an expensive undertaking, often beyond the budget of a small business. To begin with, you need to purchase the software and, in many cases hire a professional database administrator or programmer experienced in SQL to set it up. Once the database is up, you need to enter information into the database or import from existing records which can be time consuming. If your database is going to contain sensitive information you will need to ensure that the information is secured against unauthorized which adds another layer of cost to the implementation.

Limitations in structure: - Many relational database systems impose limits on the lengths of

the data fields. If you enter more information into a field than it can accommodate, the information will be lost.

Isolated information: - Because relational databases can use a large number of tables, there is always the risk that some information may be lost or forgotten, particularly when it is being transferred from one system to another. This is usually more of a problem for large organizations, particularly when they are using different database systems.

**Main Sources of data: -**

Business Process – OLTP (Online Transaction Processing). This mainly includes the bank transactions and the business sales. This is normally a structured data with not too much memory.

Human created data – Email, documents, social media

Machine data – CCTV, Sensors, Satellite

**Evolution of Hadoop: -**

Hadoop is an open source framework overseen by Apache Software Foundation which is written in Java for storing and processing of huge data sets with the cluster of commodity hardware. There are mainly two problems with big data. First one is to store such a huge amount of data and the second one is to process that data. The traditional approach like RDBMS is not sufficient due to the heterogeneity of the data. So hadoop comes as the solution to the problem of big data i.e. storing and processing the big data with some extra capabilities. There are mainly two components of Hadoop which are Hadoop Distributed File System (HDFS) and Yet Another Resource Negotiator (YARN).

Hadoop History: -

Hadoop was started by Doug Cutting and Mike Cafarella in the year 2002 when they both started to work on Apache Nutch project. Apache Nutch project was the process of building a search engine system that can index 1 billion pages. After a lot of research, they concluded that such a system will cost around half a million dollars in hardware and along with a monthly running cost of $30000 dollars which is very expensive. So, they realized that their project architecture will not be capable enough to the workaround with billions of pages on the web. So, they were looking for a feasible solution which can reduce the implementation cost as well as the problem of storing and processing of large datasets.

In 2003 they came across a paper that described the architecture of Google distributed File System called GFS which was published by Google to store large data sets. Now they realize that this paper can solve their problem of storing very large files which were being generated because of web crawling and indexing processes. But this paper was just the half solution to their problem.

In 2004 Google published one more paper on the technique of the technique MapReduce, which was the solution of processing those large datasets. Now this paper was another half solution for Doug Cutting and Mike Cafarella for their Nutch project. These both techniques (GFS and MapReduce) were just on white paper at Google. Google didn’t implement these two techniques. Doug Cutting knew from his work on Apache Lucene (It is a free and open source project) that open source is a great way to spread technology to more people. So together with Mike Cafarella he started implementing Google’s technique as open source in the Apache Nutch project.

In 2005 Cutting found that Nutch is limited to only 20 to 40 node clusters. He soon realized two problems.

1. Nutch won’t achieve its potential until it reliably ran on large clusters.
2. And that was looking impossible with two people.

The engineering task in Nutch project was much bigger than he realized. So, he started to find a job with a company who is interested in their efforts and he found Yahoo. Yahoo had a large team of engineers that was eager to work on their project.

So, in 2006, Doug Cutting joined Yahoo along with Nutch project. He wanted to provide the world with an open source, reliable, scalable computing framework, with the help of Yahoo.

So, at Yahoo first he separates the distributed computing parts from Nutch and formed a new project called Hadoop (He gave name Hadoop it was the name of a yellow toy elephant which was owned by the Doug Cutting’s son. and it was easy to pronounce and was the unique word). Now he wanted to make Hadoop in such a way that it can work well on thousands of nodes. So with GFS and MapReduce, he started to work on Hadoop.

In 2007, Yahoo successfully tested Hadoop on a 1000 node cluster and start using it.

In January of 2008, Yahoo released Hadoop as an open source project to ASF(Apache Software Foundation). And in July of 2008, Apache Software Foundation successfully tested a 4000-node cluster with Hadoop.

In 2009, Hadoop was successfully tested to sort a PB (Petabyte) of data in less than 17 hours for handling billions of searches and indexing millions of web pages. And Doug Cutting left the Yahoo and joined Cloudera to fulfil the challenge of spreading Hadoop to other industries.

In December of 2011, Apache Software Foundation released Apache Hadoop version 1.0.

And later in Aug 2013, Version 2.0.6 was available.

And currently, we have Apache Hadoop version 3.0 which released in December 2017.

**The 5v’s of big data: -**

Data is being produced at astronomical rates. In fact, 90% of the data in the world today was created in the last 2 years. The term big data can be defined as the data that becomes so large that it cannot be processed using conventional methods. The size of the data which can be considered to be big data is a constantly varying factor and newer tools are continuously being developed to handle this big data. It is changing our world completely and shows no signs of being a passing fad that will disappear anytime in near future.

Velocity: - Obviously, velocity refers to the speed at which vast amounts of data are being generated, collected and analysed.  Every day the number of emails, twitter messages, photos, video clips, etc. increases at lighting speeds around the world. Every second of every day data is increasing.  Not only must it be analysed, but the speed of transmission, and access to the data must also remain instantaneous to allow for real-time access to website, credit card verification and instant messaging.  Big data technology allows us now to analyse the data while it is being generated, without ever putting it into databases.

Volume: - Volume refers to the incredible amounts of data generated each second from social media, cell phones, cars, credit cards, M2M sensors, photographs, video, etc. The vast amounts of data have become so large in fact that we can no longer store and analyse data using traditional database technology.  We now use distributed systems, where parts of the data is stored in different locations and brought together by software.  With just Facebook alone there are 10 billion messages, 4.5 billion times that the “like” button is pressed, and over 350 million new pictures are uploaded every day.  Collecting and analysing this data is clearly an engineering challenge of immensely vast proportions.

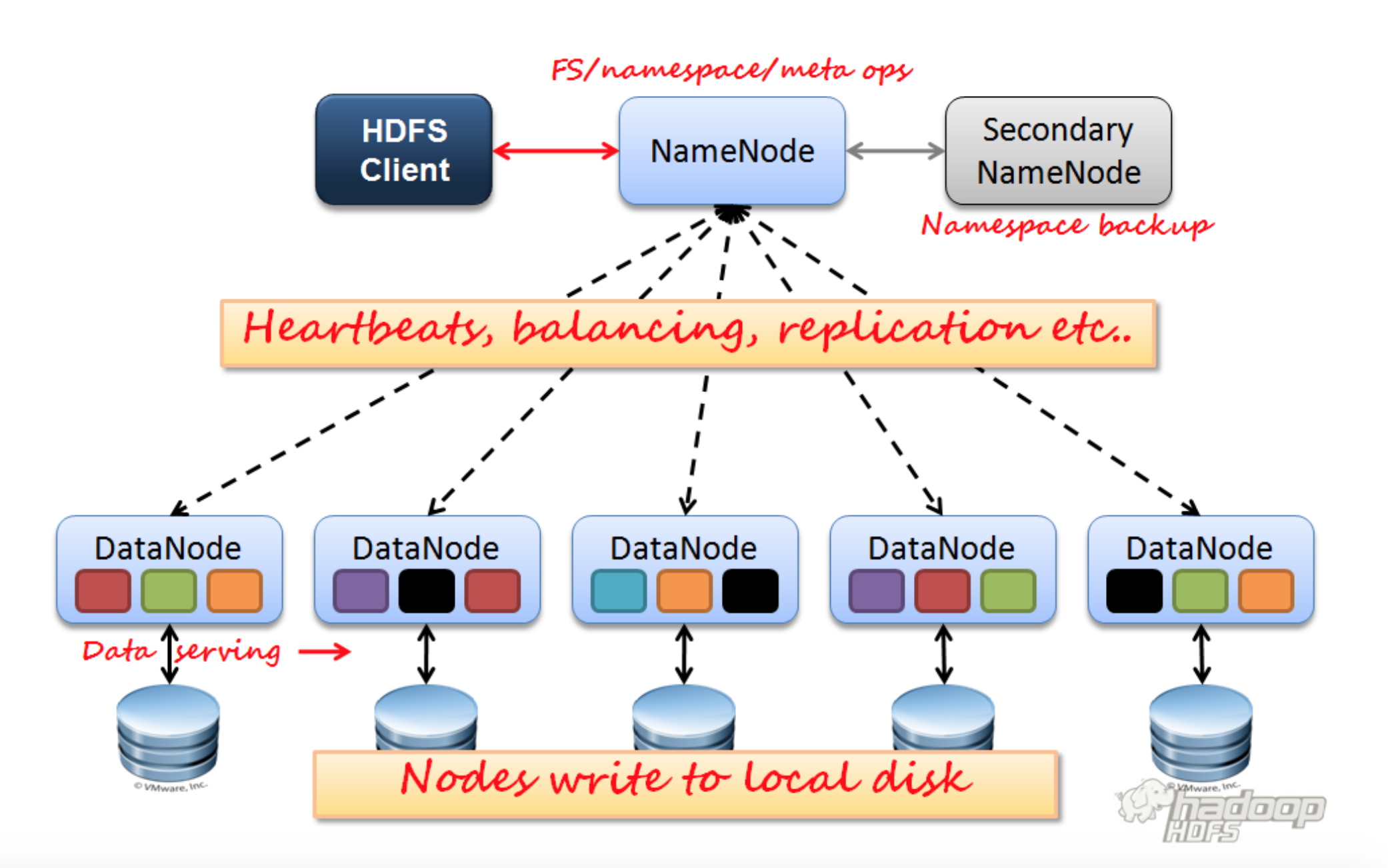
Value: - When we talk about value, we’re referring to the worth of the data being extracted.  Having endless amounts of data is one thing, but unless it can be turned into value it is useless.  While there is a clear link between data and insights, this does not always mean there is value in big data. The most important part of embarking on a big data initiative is to understand the costs and benefits of collecting and analysing the data to ensure that ultimately the data that is reaped can be monetized.

Variety: - Variety is defined as the different types of data we can now use.  Data today looks very different than data from the past.  We no longer just have structured data (name, phone number, address, financials, etc) that fits nice and neatly into a data table.  Today’s data is unstructured.  In fact, 80% of all the world’s data fits into this category, including photos, video sequences, social media updates, etc.  New and innovative big data technology is now allowing structured and unstructured data to be harvested, stored, and used simultaneously.

Veracity: - Last, but certainly not least there is veracity.  Veracity is the quality or trustworthiness of the data.  Just how accurate is all this data?  For example, think about all the Twitter posts with hash tags, abbreviations, typos, etc., and the reliability and accuracy of all that content.  Gleaning loads and loads of data is of no use if the quality or trustworthiness is not accurate.  Another good example of this relates to the use of GPS data.  Often the GPS will “drift” off course as you peruse through an urban area.  Satellite signals are lost as they bounce off tall buildings or other structures.  When this happens, location data has to be fused with another data source like road data, or data from an accelerometer to provide accurate data.

**HDFS Architecture: -**

HDFS is the storage part of Hadoop.



The HDFS architecture mainly consists of four components: -

1. Client – Edge node (get, put and read the data from the HDFS)
2. Name node
3. Data node
4. Secondary name node

HDFS is a master slave architecture.

Name node is the master node. The main tasks of the name node are the cluster administration and metadata management.

Steps to store data into the hadoop cluster: -

1. Client contacts the name node to store the data.
2. Name node will say to the client to store data in particular data nodes.
3. The client will then divide the data into chunks of 64 MB and put the data in the data nodes informed by the name node.
4. Then data node will inform name node that a chunk has been stored in it.
5. Then name node will ask the data node to replicate the chunk in different nodes to avoid fault tolerance. The default replication factor is 3.

For every 3 seconds data node will send heartbeat to the name node saying that it is alive. Heartbeat is a simple java program. If a data node is not sending heartbeat means if the data node is not responding, then name node will give a grace time of 10 minutes and wait for the data node to respond. If after 10 minutes also data node is not responding, then name node will consider the data node as dead.

Normally there will be 3 copies of a chunk available if the cluster is good. When any node goes down all the chunks in it will be lost making the chunk copies to 2. Then That chunk is said to be under replication. In this case name node will ask the data node which is having a copy of the chunk which is under replication to replicate it into a different data node so that the chunk copies become 3 which is the replication factor we set.

When any data node which is dead comes back and starts responding it causes the chunk copies to become 4 which is called over replication. In this case the name node asks the data node which came back to clear all the copies of the chunks in it as it had already replicated it.

If any data node seems to be more burdened and if we run a balancing program in the name node then name node will accordingly distribute all the chunks across all the data nodes avoiding any data node to be more burdened.

The maximum number of replications possible are 512.

Name node does the metadata management in the following way:

1. When the name node is first started there will be 2 files called fsImage and Edit log.
2. When any transaction occurs i.e. when any chunk is stored or deleted from the data node that metadata will be stored in both fsImage and Edit log. **Note:** fsImage data is not permanently saved means not stored in hard disk it will be in RAM only whereas Edit log data is continuously saved to hard disk for every 1 second.
3. After one hour the Edit log which is saved in the hard disk is sent to secondary name node which will do checkpointing with the old fsImage present in it. Checkpointing means merging the fsImage and Edit log.
4. Then the secondary name node will store a backup of the fsImage in its hard disk and sent the modified fsImage to the name node hard disk.
5. In this way the fsImage in the name node gets updated with recent data for one hour. We won’t transfer the fsImage directly to the secondary name node because it will create more burden on the name node as the fsImage file size is more and also, we want to make sure that name node is properly capturing the metadata in the fsImage.
6. When the name node gets restarted it will take the fsImage and Edit log from hard disk and do checkpointing because secondary name node won’t contain the edit log. Till the fsImage in the name node reaches to the RAM name node will not allow any transactions and the name node will be in a mode called safe mode.

This is Hadoop 1.0 architecture.

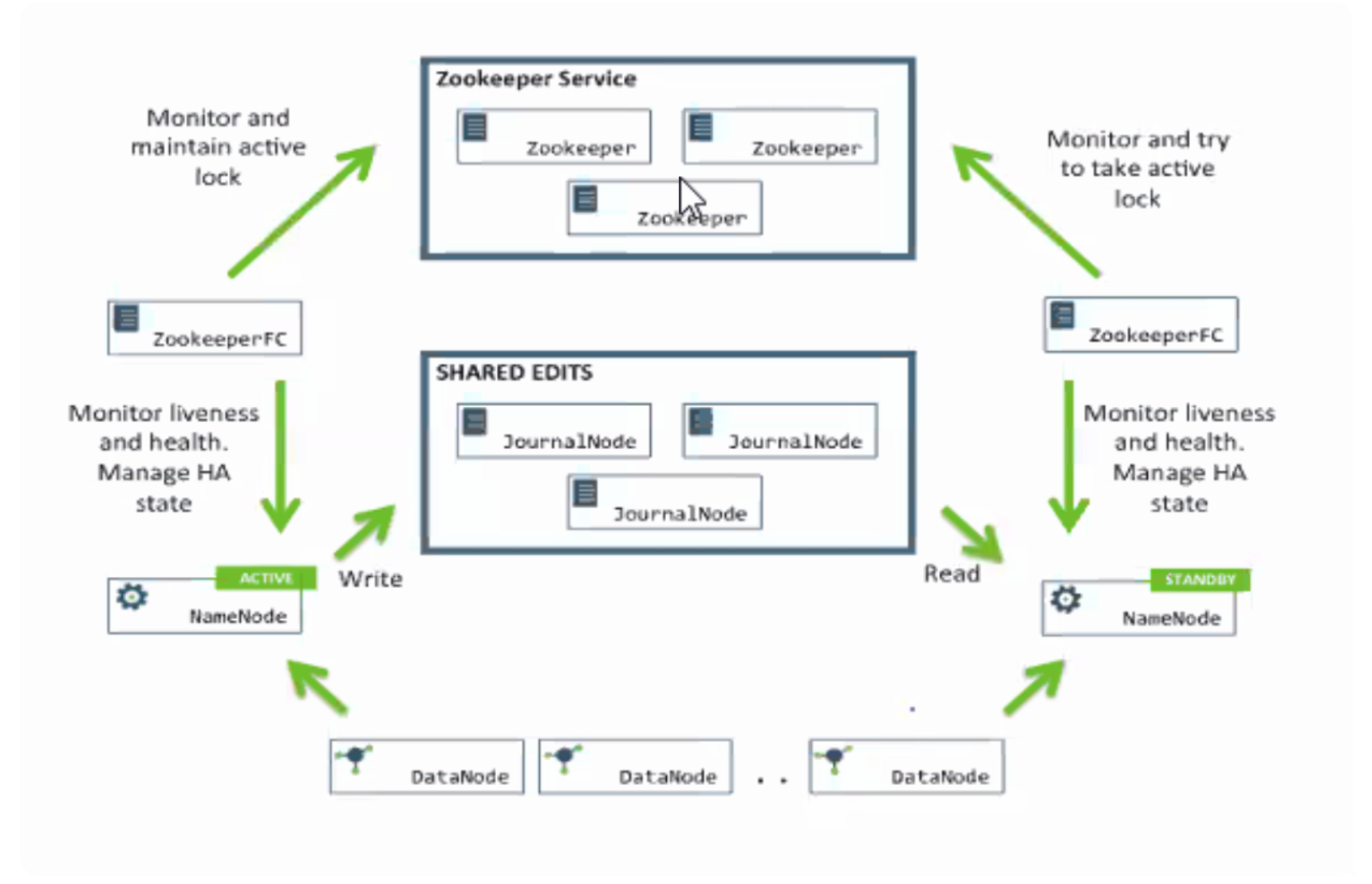
Drawbacks of Hadoop 1.0 architecture: -

1. Single point of failure: - When the name node goes down there won’t be any edit log to do checkpointing making the architecture a single point of failure.
2. Scalability: - The fsImage size will keep on increasing and eventually it will come to a stage where the name node can’t support it. For example, if the name node has 2 TB RAM one or the other day it cannot open the fsImage because the fsImage size is more than the RAM of the name node.
3. Block size is 68 MB which makes huge number of chunks to be created every time for a huge file.

To overcome the above drawbacks Hadoop 2.0 architecture came up.

1. To overcome single point of failure high availability was provided in 2.0.
2. To overcome scalability drawback Federation was introduced.
3. The block size is increased to 128 MB.

Hadoop 2.0 Architecture: -



The main components in Hadoop 2.0 architecture are: -

1. Active Namenode – To store metadata to fsImage and write the recent transactions to journal node.
2. Standby Namenode – To perform checkpointing and send the recent fsImage back to the Standby name node.
3. Journal node – To store the recent transactions of one hour.
4. Zookeeper – To monitor name node health

When any node wants to become a name node it should contain the contain fsImage and the edit log with it and also if the dead name node comes back there should be a mechanism to make only active name node at any point of time because both the name nodes should not write data to the journal node.

Name node metadata management in Hadoop 2.0: -

1. The name node will write the metadata to its fsImage and the recent transactions data is written to shared edits place inside journal nodes.
2. After one hour the standby name node will take the recent transactions from journal node and perform checkpointing and send the recent fsImage back to the active name node hard disk and also stores a copy of the fsImage in its hard disk.
3. The name node which contains the in\_use.lock file will be considered as the active name node.
4. Namenode will send heartbeat for every 3 seconds to the zookeeper indicating that is active. If the name node is not sending heartbeat zookeeper will give a grace time of 10 minutes even, then if the name node is not representing then it is considered as dead. Zookeeper monitors the health of the name node using FC which is known as Fail over Controller program.
5. If the active name node is dead, then immediately zookeeper will send in\_use.lock file to standby name node. Then standby name node will take the fsImage in its hard disk and the recent transactions from the journal node and perform checkpointing. Till the standby name node perform checkpointing no transactions are allowed making the standby name node to go the safe mode. In this way single point of failure is handled.
6. If the dead name node comes back and as it has in\_use.lock file in it, it will start writing the recent transactions to the journal node and as the standby name node is also having the in\_use.lock file in it, it also starts writing to the journal node. This scenario is called split brain scenario.
7. To avoid this scenario zookeeper will send a fencing program to the name node which is back to stop writing data to the journal node.
8. After this zookeeper will send a program to remove the in\_use.lock file from the name node which came back.
9. If there is no standby name node to read and perform checkpointing then the active name node only will perform checkpointing and save the fsImage to the hard disk.
10. In hadoop 2.0 the client connects to the zookeeper to store the data then zookeeper redirects the corresponding request to the active name node because zookeeper is the one which knows the active name node.

The default location in edge node in Cloudera is /home/cloudera.

The default location in hadoop is /user/cloudera.

Every hadoop command is either started with hadoop fs – (hadoop 1.0) or hdfs dfs – (hadoop 2.0). To store or retrieve data from hdfs we need to give absolute path every time.

Hadoop Commands: -

hadoop fs -ls – To list the contents of a directory.

hadoop fs -mkdir – To create a directory.

hadoop fs -cat – To display the content of a file on the console.

hadoop fs -put – To copy the file from local system to HDFS.

hadoop fs -get – To copy the file from HDFS to local system.

hadoop fs -appendToFile – To append to the contents of an existing file.

hadoop fs -cp – To copy the file from one directory to other in HDFS.

hadoop fs -mv – To move the files from one directory to other in HDFS.

hadoop fs -count – To know the number of directories and sub directories, number of files and total size of the files in the given directory.

hadoop fs -touchz – To create an empty file.

hadoop fs -put -f – To forcefully overwrite the file which is already created.

