1.what is big data

2.File systems(FAT32,NTFS,EXT3)

3.HDFS File system

4. What is Map reduce and Map reduce Programming Model

5. Hadoop Architecture

MRV1 Architecture

YARN Architecture

6.

More On --> Namespace, edit log file, FS Image path

Vertically Scalable:

Increasing the resources like RAM, Storage Capacity and CPU Power to existing Machine (Server/CPU) to increase the computational power.

Vertical-scaling is often limited to the capacity of a single machine, scaling beyond that capacity often involves downtime and comes with an upper limit.

Horizontally Scalable:

Adding more Machines/Nodes dynamically to pool of resources (Cluster) will increase the computational power w.r.t Load.

Data&Software Programme will be distributed across multiple nodes to achieve computational Power.

What is Big data:

Big data means collection of large data sets that cannot be processed using traditional computing techniques.

Big Data Classified by IBM (Volume, Velocity and Variety)

Volume --> If the data that is to be processed beyond Terabytes.

Velocity--> Amount of Data that is being created increases rapidly with respect time.

Variety--> Data in different formats (Structured- Table Format (RDBMS), Semi Structured (XML, CSV Files) & UN Structured (Health records in PDF Files, Emails, Videos and Images))

Why Big Data?

To analyze the hidden information in large datasets which will help organizations for better decision making which in turn leads to smarter business moves, more efficient operations, higher profits and happier customers.

Big data Challenges:

* Storage
* Computational Power
* Data Loss
* Cost

Hadoop is a solution for the above Big data problems in a cost effective manner.

What is Hadoop: Hadoop is an open-source framework developed in java by Doug Cutting.

It allows to store and process big datasets in a distributed environment across clusters of computers using simple programming model called map reduce.

Key Features:

* Storing and processing the huge volumes of data.
* Supports distributed Computing.
* Highly Available (Less Downtime).
* Horizontally Scalable
* Cost Effective Hardware
* Easy for Programmers and Non-Programmers.

Is hadoop replacement for Databases?

No, hadoop is not a replacement for Databases and hadoop is not recommended for small data sets.

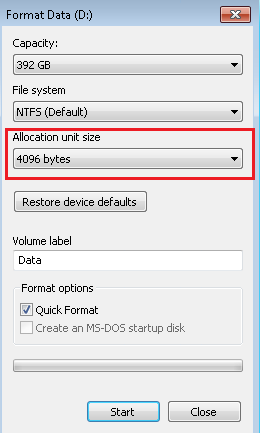
|  |  |
| --- | --- |
| Hadoop | RDBMS |
| Horizontally Scalable | Vertically Scalable |
| Distributed Computing Frame work.(OLAP) | OLTP |
| Read/Write based on (Key, Value Pair) | Read/Write based on Record |
| Structured, Semi Structured and Un Structured | Only Structured data |
| Batch Processing | Interactive Processing |
| No Relations across the data sets  (PK, FK ...) | Relations exists across Tables (PK, FK ...) |
| Processing will be done as a task at each data node of the cluster. | Data has to be Read from SAN and brought to RDBMS server for processing. |

File System:

* Internally file system is divided into blocks/units.
* Each block size will be of 4KB by default.
* However the size can be changed up on the requirement.
* 1 Block=8\* (1 sector)
* Window File system🡪NTFS and Unix 🡪 EXT3 with block sizes 4Kb.
* Check Sector value; Go to run🡪 msinfo32🡪Components🡪Storage🡪Drives

Second Way of checking Block Size:

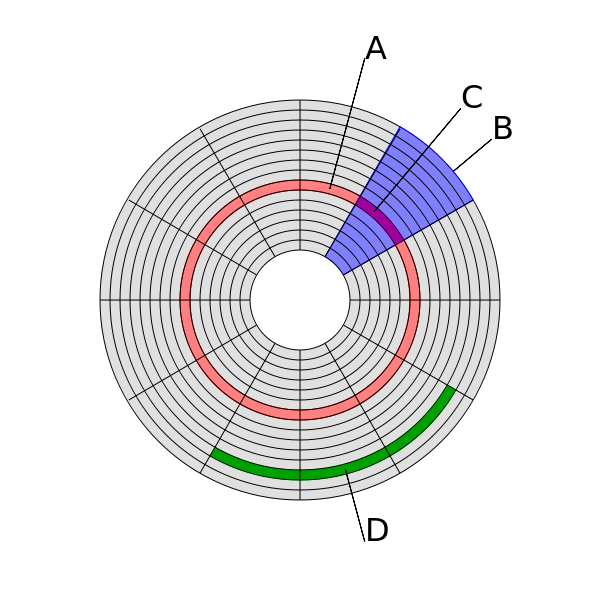
Select drive and format, it will show default Allocation unit size (Block).



File System size in Units:

|  |  |
| --- | --- |
| **Memory** | **Size** |
| 8 Bits | 1 Byte |
| 1 Kilobyte - KB | 1024 Bytes |
| 1 Megabyte - MB | 1024 KBs |
| 1 Gigabyte - GB | 1024 MBs |
| 1 Terabyte - TB | 1024 GBs |
| 1 Petabyte - PB | 1024 TBs |
| 1 Exabyte - EB | 1024 PBs |
| 1 Zettabyte - ZB | 1024 EBs |
| 1 Yottabyte - YB | 1024 ZBs |

Architecture of Data Cluster in Storage Box/HDD:



Disk structure:  
(A) Track  
(B) Sector  
(C) [Track sector](https://en.wikipedia.org/wiki/Disk_sector)  
(D) Cluster

* 1 Data Block=8 Sectors

Normal File System Vs HDFS:

|  |  |
| --- | --- |
| Local File System | HDFS |
| Block size is in Kb’s | Block size in MB |
| Default block size 4Kb | Default block size 128 MB |
| Left over space in block will get wasted | Left over block size will be utilized by other file. |
| No replication for blocks | Replication for blocks on multiple data nodes |
| High reliable hardware (Storage Disk) | Low cost hardware (Storage Disk) |

Why Block size is huge in HFDS:

When the block size is huge the OS will attempt to save the file by writing the blocks in contiguous locations. If the blocks are in contiguous locations both reads/writes will be faster and disk head can seek contiguously without positioning its head to seek randomly across the disk.

**Note:** OS will attempt to write the blocks in contiguous location however this is not guaranteed every time.

2. If the block size is small it will be a bottle neck for name node to hold the Meta data information of all the blocks of datanodes.

When read/ write a file from Hadoop cluster java program will be running behind the scenes but not the MapReduce code.

Ex: hadoop fs - copyFromLocal /usr/temp/test.txt /user/hadoop/testdir/

**What is streaming access?**

As HDFS works on the principle of ‘Write Once, Read Many‘, the feature of streaming access is extremely important in HDFS. HDFS focuses not so much on storing the data but how to retrieve it at the fastest possible speed, especially while analyzing logs. In HDFS, reading the complete data is more important than the time taken to fetch a single record from the data.

**Hadoop Architecture:**

Hadoop is a master/ slave architecture. The master being the namenode and slaves are datanodes. The namenode controls the access to the data by clients. The datanodes manage the storage of data on the nodes that are running on. Hadoop splits the file into one or more blocks and these blocks are stored in the datanodes. Each data block is replicated to 3 different datanodes to provide high availability of the Hadoop system. The block replication factor is configurable.

****

**Hadoop Components:**

The major components of hadoop are: 

* Hadoop Distributed File System: HDFS is designed to run on commodity machines which are of low cost hardware. The distributed data is stored in the HDFS file system. HDFS is highly fault tolerant and provides high throughput access to the applications that require big data.
* **Name node:** Namenode is the heart of the Hadoop system. The namenode manages the file system namespace. It stores the metadata information of the data blocks. This metadata is stored permanently on to local disk in the form of namespace image and edit log file. The namenode also knows the location of the data blocks on the data node. However the namenode does not store this information persistently. The namenode creates the block to datanode mapping when it is restarted. If the namenode crashes, then the entire hadoop system goes down.
* Secondary Namenode: The responsibility of secondary name node is to periodically copy and merge the namespace image and edit log. In case if the name node crashes, then the namespace image stored in secondary namenode can be used to restart the namenode.
* DataNode: It stores the blocks of data and retrieves them. The datanodes also reports the blocks information to the namenode periodically.
* JobTracker: JobTracker responsibility is to schedule the clients jobs. Job tracker creates map and reduce tasks and schedules them to run on the datanodes (tasktrackers). Job Tracker also checks for any failed tasks and reschedules the failed tasks on another datanode. Jobtracker can be run on the namenode or a separate node.
* TaskTracker: Tasktracker runs on the datanodes. Task trackers responsibility is to run the the map or reduce tasks assigned by the namenode and to report the status of the tasks to the namenode.

<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

**Read & Write strategy of Hadoop:**

<http://www.devinline.com/2015/03/read-and-write-operation-in-hadoop.html>

Name Node:

Secondary Name Node:

Data Node:

HDFS Federation: http://hortonworks.com/blog/an-introduction-to-hdfs-federation/

Block cache: Normally a datanode reads blocks from disk, but for frequently accessed files the blocks may be explicitly cached in the data node’s memory, in an off-heap block cache.

Job Tracker:

Task Tracker:

**What is a heartbeat in HDFS?**

A heartbeat is a signal indicating that it is alive. A datanode sends heartbeat to Namenode and task tracker will send its heart beat to job tracker. If the Namenode or job tracker does not receive heart beat then they will decide that there is some problem in datanode or task tracker is unable to perform the assigned task

Space Reclamation:

edits File:

FS image File:

Each *fsimage* file contains a serialized form of all the directory and file

inodes in the filesystem. Each inode is an internal representation of a

file or directory’s metadata and contains such information as the file’s

replication level, modification and access times, access permissions,

block size, and the blocks the file is made up of

FSIMAGE: has metadata information about HDFS. File owner, permissions, folders and block information (not block location) for files.

EDITS is a log of information that happens in real time to HDFS. The information is very similar to what you will see in FSIMAGE and it will be merged to FSIMAGE periodically - check Secondary Namenode lecture under architecture.

Edits file and FS image file will be in the same directory. Opened with only sudo privilege.

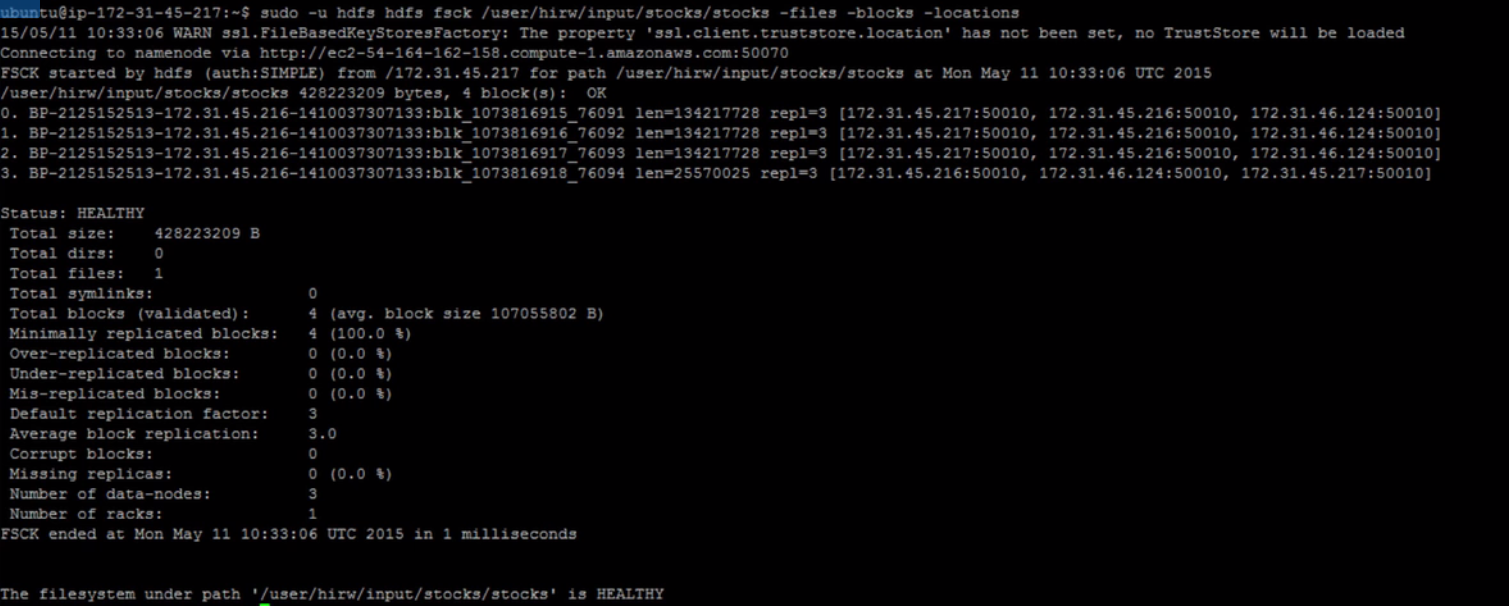
/var/lib/hadoop-hdfs/cache/hdfs/dfs/name/current/

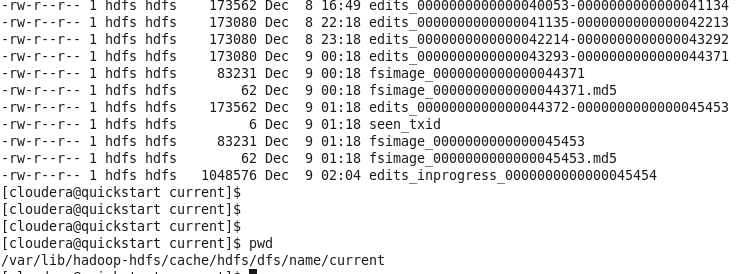
Command to view fsimage file:

sudo hdfs -oiv fsimage\_0000000000000049229 kiranfsimage.txt

oiv- offline image viewer.

sudo hdfs fsck /kiran.txt -files -blocks –locations





Here edit files are created in batches each holding some set of transactions in it and all this batch files will be consolidated and created into one FSimage file.

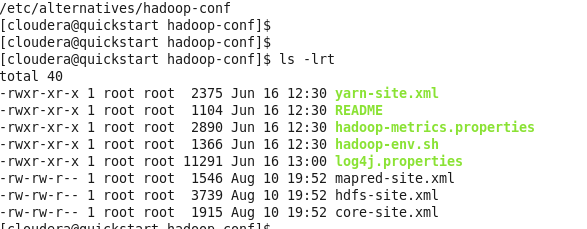
**Edits File Format:** edits\_00000StartOfTxn-0000EndOfTxn

**Fsimage Format:** fsimage\_00LastTransactionId (Till this point snapshot has been created)

Configuration files in Hadoop:

The xml files in the conf directory are some of the important configuration files in Hadoop system.

**conf directory** - /etc/hadoop/conf(Local file system)



**mapred-site.xml:**

On which node job tracker is configured to run.

* mapred.job.tracker=localhost:8021

What kind of frame work is been used to read/write from HDFS

* mapreduce.framework.name=yarn/classic/local
* Yarn=yarn framework (MRV2)
* Classic= map reduce frame work (MRV1)
* Local= can be run from IDS (Not sure)

Mapreduce temp directory is defined.

* mapreduce.task.tmp.dir=/var/lib/hadoop-hdfs/cache/${user.name}

**core-site.xml:**

Will have the information like where the Hadoop file system has been installed.

* fs.defaultFS=hdfs://quickstart.cloudera:8020 (Usually Name node)

**hdfs-site.xml:**

Replication factor is defined in this file.

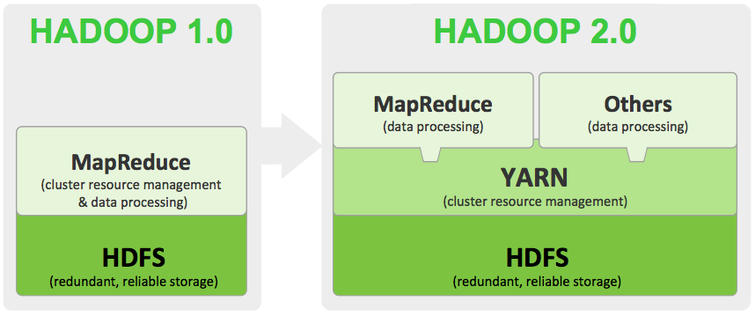
* dfs.replication=3 (Default)

Name Node and Data Node directories are defined.

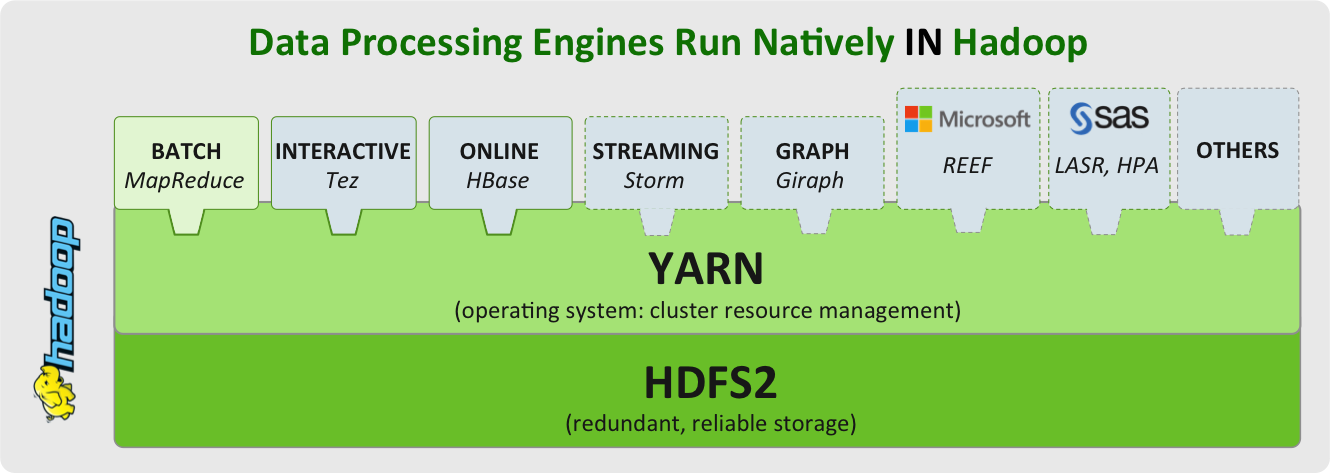
* dfs.namenode.name.dir= /var/lib/hadoop-hdfs/cache/hdfs/dfs/name
* dfs.datanode.data.dir= /var/lib/hadoop-hdfs/cache/hdfs/dfs/data
* fs.checkpoint.dir
* fs.checkpoint.edits.dir

Map Reduce Architecture:

YARN Architecture: Before Hadoop 2.0, Applications use to rely only on MapReduce programming model. After 2.0 version YARN has been introduced to interact with different workloads by running natively in Hadoop.



**Different workloads running natively on Hadoop 2.0**



List of Demons in MRV2 Architecture (YARN):

Hadoop 2.x supports two Name Nodes at a time one node is active and another is standby node

Active Name Node handles the client operations in the cluster

StandBy Name Node manages metadata same as Secondary Name Node in Hadoop 1.x

When Active Name Node is down, Standby Name Node takes over and will handle the client operations then after

HDFS HA can be configured by two ways

Using Shared NFS Directory

Using Quorum Journal Manager

References of YARN:

http://www.tomsitpro.com/articles/hadoop-2-vs-1,2-718.html