Big Data

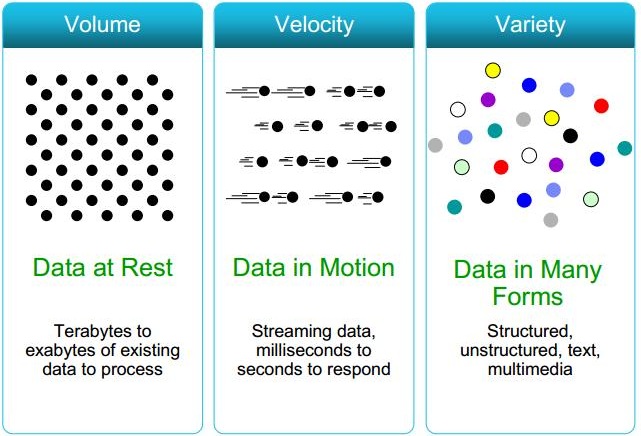
**What is Bigdata:**

The term big data is used to describe the collection of complex and large data sets. Hence it is difficult to process, store, search and analyze this kind of data using traditional data base management systems.

**Why big data is so important to any organization:**

1. More data leads to more accurate analysis
2. More accurate analysis leads to better decision making.
3. Better decision means greater operational efficiencies, cost reductions and reduced risk.

**Concept of 3 ‘Vs :**



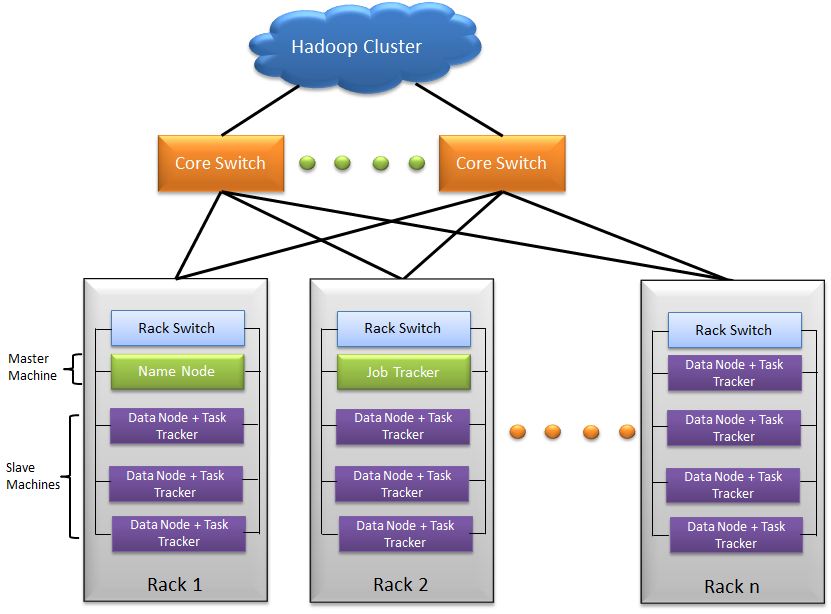
**Daemon /Process/Pillar of Hadoop :**

1. Data node
2. Name node
3. Secondary name node
4. Job tracker
5. Task tracker

**Name Node:**

It is heart of Hadoop eco system which maintains the meta data information of all the data nodes in the cluster.

Only one node can be set up per cluster.

****

**Heart Beat :**

Every 3 second data node sends acknowledgement response to name node. Its called heart beat.

**Block report:**

10th Heart beat is called as block report.

Name node 1 per cluster.

**How to read file from HDFS :**

1. Client requests for sales data.
2. Name node will check meta data information that, which data node keeps sales data information.
3. Name node sends index report to client.
4. Client directly get data from data node after getting the meta data information from name node.



**How to write a file to HDFS :**

Rack awareness policy : While writing the files, Hadoop writes data two of one rack and one of another rack. (2+1). This is called rack awareness policy.



1000 MB Data:



**What if Name Node fails :**

If name node fails secondary name node comes into picture.

Secondary name node :

It is not same as primary name node. It just follows the primary name node and maintain the log information of the job which is running on the primary name node.

**If the job is 60 % finished** 🡺 and then name node fails 🡺 Secondary name node checks the log and completes the rest of 40% job.

**If the job is 0% completed** 🡺 and name node fails 🡺SNN will not do anything because the log doesnot contain job log as the job is not started yet.

**Process to NWT :**

1. Enters into safe mode 🡺 No optimisation in HDFS.
2. Check the responsibility of fsimage and editlogs.
3. Come out of safe mode.

**Fs image : file system image**

**Edit logs : transaction logs**

**If name node fails frequently:**

Check point node : it is the back up of name node.

**Balancer:**

It balances the cluster node.

If the space of data node is getting filled up. Name node balances it by adding more space.

Name node checks the block report and balances the data nodes.

**Modes in Hadoop :**

1. Local mode /standalone mode
2. Pseudo mode 🡺 multi user,1 system all components
3. Distributed mode 🡺 distributed cluster

Yahoo: 4000 nodes of cluster

FB : 2320 nodes of cluster

**Federation in HDFS :**

This is a new concept where we can have n number of name nodes

**Horizontal scaling:**

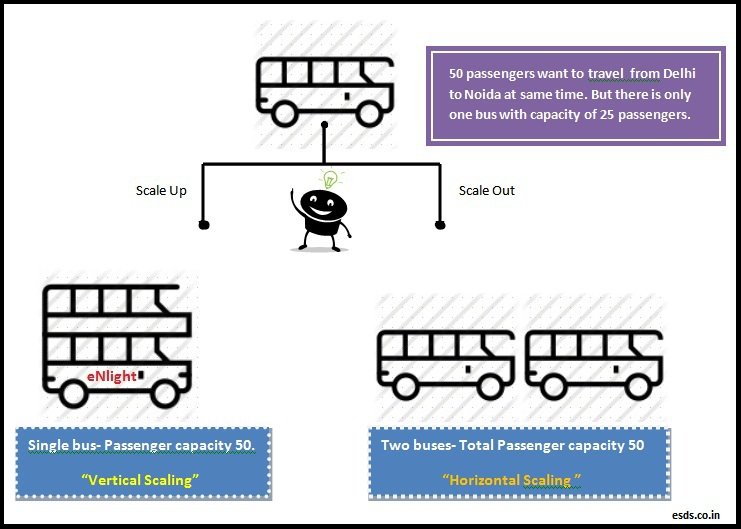
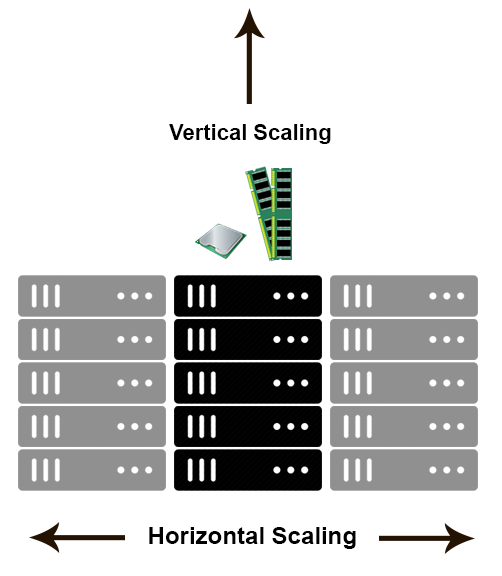
As mentioned in above website example, when your business grows at the same time hits also grows so the responsibility of your server/node grows.  So to reduce this responsibility what we can do is we can add one more server with same capacity along with existing server. Now these two servers can handle the traffic effectively. This is what called horizontal scaling. We have not changed capacity of individual server but we decreased the load on server.

Horizontal scaling means enhancing the performance of server /node by adding more instances of server to your pool of servers so that load can be spread.

Horizontal scaling means scaling out. Horizontal scalability can be achieved with the help of clustering, distributed file system, load – balancing.

To address performance issues you can use either vertical scaling or horizontal scaling or both in cloud environments.

In cloud market some **auto scalable** models are present which are smarter than traditional scaling models and gives best performance & no down time.



**Vertical scaling:**

To increase the capacity if we increase resources in same logical unit/server then it is vertical scaling.  E.g. Add more CPUs in existing sever. If system is not handled by one CPU then increase the CPU to 3 or 4. Another example is server is having 8 GB RAM then scale it to 16 GB. Same applicable to storages also.

Consider you have a business website. As business grows you website gets more hits. Due to increase in  hits your  server   performance starts degrading. To handle the load you need to scale the resources by adding CPUs(Processors),RAM, disk capacity etc. So in this case if you are using vertical scaling strategy then you need to enhance the capabilities of same server/node which will handle the load properly .Vertical scaling means boosting the power of individual server.

The example in image shows same concept. To serve more passengers instead of adding one more bus we are just increasing the capacity of same bus by adding one floor to bus to accommodate 50 passengers.

Vertical scaling is also called **Scale up** approach.

**Single node cluster:**

**Multi node cluster:**

Sharing of cluster is possible but we should not share the same.

**Hadoop Utilities**

hadoop namenode –format 🡺 format the Hadoop file system

hadoop fs start –dfs.sh 🡺 start the name node

hadoop fs –setrep 5 dir/emp.txt

It will sow the status of the safe mode 🡺 hadoop fsadmin –safemode get

Safe mode 🡺 sudo –u hdfs dfsadmin –safemode enter

Exit safe mode 🡺 sudo –u hdfs dfsadmin –safemode leave

hadoop fs -ls <path> 🡺 list out the files present in the specified path

hadoop fs -lsr <path> 🡺 list out the files recursively.

hadoop fs -du <path> 🡺displays the disk usage in bytes of all files present in the specified path.

hadoop fs -dus <path> 🡺same as du + print the summary of disk usage of all files in the specified path.

hadoop fs -mv <src> <destination> 🡺 move or rename the file

hadoop fs -cp <src> <dest> 🡺 copy the file

hadoop fs -rm <path> 🡺 remove the file or empty directory identified by path

hadoop fs -rmr <path> 🡺 removes file or directories.

hadoop fs- put <localsrc> <destination> 🡺 copy files from local file system to Hadoop file system.

hadoop fs -copyFromLocal <localSrc> <dest> 🡺 same as put

hadoop fs -moveFromLocal <localSrc> <dest> 🡺 move the files from local file system to Hadoop file system.

hadoop fs -get [-crc] <src> <localDest> 🡺 copies the file from Hadoop file system to local file system.

hadoop fs -getmerge <src> <localDest> 🡺 Retrives all files from Hadoop file system and merge to a single file in local file system

hadoop fs -cat file\_name 🡺 read the file

hadoop fs -copyToLocal <src> <localDest> 🡺same as get

hadoop fs -moveToLocal <src> <localDest> 🡺 move the file from hdfs to local system

hadoop fs -mkdir 🡺 create a directory

hadoop fs -touchz <path> 🡺 create an empty file in the specified path with current time stamp. It wont create a file if a file is already present in the path and sizes more than zero bytes.

hadoop fs -test -[ezd] <path> 🡺 Returns 1 if path exists; has zero length; or is a directory or 0 otherwise.

hadoop fs -stat [format] <path> 🡺 Prints information about path. Format is a string which accepts file size in blocks (%b), filename (%n), block size (%o), replication (%r), and modification date (%y, %Y).

hadoop fs -tail [-f] <file2name> 🡺 Shows the last 1KB of file on stdout.

**Map Reduce**

Map reduce is a technique by which we process a large amount of data by dividing data in smaller chunks.

Jobtracker

tasktracker

tasktracker

tasktracker

Input Job (mapper, reducer, input)

Data transfer

Assign tasks

1. Client submits 100 job to job tracker.
2. Job tracker asks name node about the data nodes where the jobs will process.
3. Job tracker assigns 100 jobs to 100 task trackers.
4. Every 60 secs task tracker sends acknowledgment status to job tracker.



Map reduce has 2 phases:

1. Mapper phase : split the data
2. Reducer phase : segregate the data

Input file : It should be in HDFS.

Input format : 3 types of format.

|  |  |  |
| --- | --- | --- |
| Format | Key | Value |
| Text Input format | Byte offset | Complete content |
| Key value pair | Till first tab | Rest of the content |
| Sequential file format | User defined | User defined |
|  |  |  |

Hi How are you ? 🡺 here byte offset of this content is Long. Hence key is long and value is complete content.

Hi How are you 🡺 key: Hi Value: How are you

Sequential format is multiple files zipped together.

1. Select the file to be used.
2. Based on the format write the definition of input split.
3. Create the factor of record data.

Input split: split(delimeter)

Split (“ ”)

Mapper phase :

1st phase to interact with

Java code

Split data

1. Import all required packages to the mapper class.
2. Define the mapper class with 4 parameters

.input key

.input value

.output key

.output value

1. Map method with 4 parameters : In map method we write the logic to split data.
   1. Input key
   2. Input value
   3. Output connector
   4. reporter
2. Output collector(OK,OV)

Intermediate phase:

Shuffle🡺

(1,x) (1,y) (1,z) (2,I) (2,q)

All the output of mapper will talk to each other and create the list of values having same key.

(1,x,y,z) (2.I.q)

Partitioner 🡺

All the values associated with the same key should go to same reducer.

Reducer 1 : (1,x,y,z)

Redcuer 2: (2,I,q)

Sort 🡺

Reducer phase :

Java code

Data aggregation (sum,max,min,avg,count)

1. Import all required packages
2. Define reducer class with 4 parameters
   1. Input key : that is out put key of mapper
   2. Input value : Out put value of mapper
   3. Out put key
   4. Output value
3. Reducer method with 4 parameters
   1. Input key
   2. Input value
   3. Out put key
   4. Reporter
4. Output collector (Output key, output value)

Reducer method will call every input of values.

|  |  |
| --- | --- |
| (1,X,Y,Z)  (2,P)  (3,Q) | 3 times |

Combiner Class:

It is an optional phase in map reduce which is used to increase the performance of a job.

Combiner comes after intermediate phase before reducer.

Ex:

(Cat, 1) (Cat, 1) (Cat, 1)

Normal : (Cat, 1, 1, 1) 🡺 3 times called

Combiner class : (Cat, 3) 🡺 1 time called

I/p🡺 Mapper 🡺Intermediate phase🡺Combiner class🡺Reducer🡺 O/p

Speculative Execution :

If all jobs completed and awaiting for one job which is not completed for any reason. In this case speculative execution will create a duplicate job in different system and execute it.If the duplicate job executes faster. It completed the process.

Hadoop Streaming :

Code written in any programming language can be used as Hadoop streaming.

Stdin

Stdout

Word count program

Hi How are you

Hi How are you bye

2 lines so 2 mapper

|  |  |  |
| --- | --- | --- |
| Map-1    Hi,1  How,1  Are,1  You,1 | IK : longWritable  IV : Text  OK : Text  OV: Int writable | Map-2  Hi,1  How,1  Are,1  You,1  Bye,1 |

Intermediate phase :

Hi,1,1

How,1,1

Are,1,1

You,1,1

Bye,1,1

Reducer : IK: text, IV: IntWritable, OK: Text . OV : IntWritable

Hi,2

How,2

Are,2

You,2

Bye,1

HIVE

Hive deals with structured and semi structured data.

Only OLAP operation. That is DDL (create,alter,drop).

Recent versions of Hive supports OLTP but nobody uses it as it is not yet stable.

Hive architecture 🡺



Data model of Hive 🡺

Ware house: collection of data base

Metastore: Every database has its own meta store. Collection of schema.

Database:Collection of table

Partitions:

Bucketing:

We should not do repartitioning as it will decrease the performace.



Sub portioning creates more files. Bucketing does not create more file.

Data types 🡺

1. Int
2. Float
3. Double
4. String/varchar

To go to hive shell. Type hive from terminal.

Show databases; it will show data bases present in hive.

create database db\_name;

create database dxpo; create database in hive.

Default database is derby.You have to mention the default database to use the table.

create database db\_name with dbproperties(‘creator’=’Nihar’,’created’=’Dec’);

desc database db\_name;

describe database extended db\_name;

describe database extended dxpo;

alter databse db\_name set dbproperties(‘creator’=’Sudhir’,’created’=’Jan’);

drop db\_name; DB should be empty before deletion.

use db\_name;

Load Data from local file system:

load data local inpath ‘home/cloudera/…’ into table table\_name

Load Data from HDFS :

load data inpath ‘/user/cloudera/Test/dept.txt’ into table table\_name