***INTRODUCTION TO HADOOP***



What is Big Data?

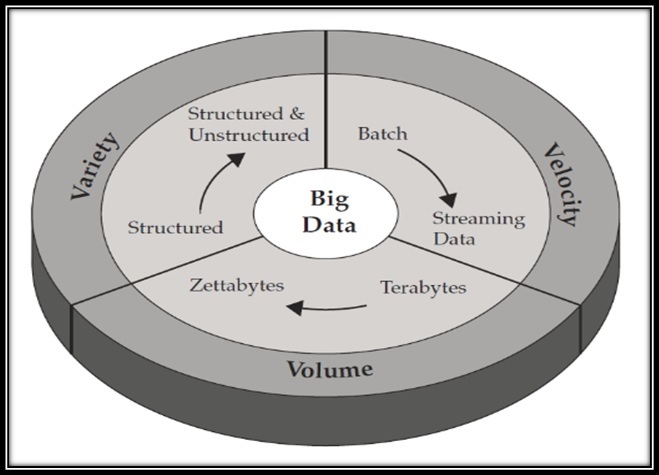
‘Big-data’ is similar to ‘Small-data’, but bigger. But having data bigger consequently requires different approaches techniques, tools & architectures .to solve. New problem and old problems in a better way.

Why Big Data?

Key enablers for the growth of “Big Data” are:

1. Increase of storage capacities
2. Increase of processing power
3. Availability of data

Characterization of Big Data:



We have all heard of the the 3Vs of big data which are Volume, Variety and Velocity. Yet, Inderpal Bhandar, Chief Data Officer at Express Scripts noted in his presentation at the [Big Data Innovation Summit in Boston](https://theinnovationenterprise.com/summits/big-data-innovation-boston) that there are additional Vs that IT, business and data scientists need to be concerned with, most notably big data Veracity. Other big data V’s getting attention at the summit are: validity and volatility. Here is an overview the 6V’s of big data.

**Volume**

Big data implies enormous volumes of data. It used to be employees created data. Now that data is generated by machines, networks and human interaction on systems like social media the volume of data to be analyzed is massive. Yet, Inderpal states that the volume of data is not as much the problem as other V’s like veracity.

**Variety**

Variety refers to the many sources and types of data both structured and unstructured. We used to store data from sources like spreadsheets and databases. Now data comes in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. This variety of unstructured data creates problems for storage, mining and analyzing data. Jeff Veis, VP Solutions at [HP Autonomy](http://www.autonomy.com/) presented how HP is helping organizations deal with big challenges including data variety.

**Velocity**

Big Data Velocity deals with the pace at which data flows in from sources like business processes, machines, networks and human interaction with things like social media sites, mobile devices, etc. The flow of data is massive and continuous. This real-time data can help researchers and businesses make valuable decisions that provide strategic competitive advantages and ROI if you are able to handle the velocity. Inderpal suggest that sampling data can help deal with issues like volume and velocity.

**Veracity**

Big Data Veracity refers to the biases, noise and abnormality in data. Is the data that is being stored, and mined meaningful to the problem being analyzed. Inderpal feel veracity in data analysis is the biggest challenge when compares to things like volume and velocity. In scoping out your big data strategy you need to have your team and partners work to help keep your data clean and processes to keep ‘dirty data’ from accumulating in your systems.

**Validity**

Like big data veracity is the issue of validity meaning is the data correct and accurate for the intended use. Clearly valid data is key to making the right decisions. Phil Francisco, VP of Product Management from [IBM](http://www-01.ibm.com/software/data/bigdata/)spoke about IBM’s big data strategy and tools they offer to help with data veracity and validity.

**Volatility**

Big data volatility refers to how long is data valid and how long should it be stored. In this world of real time data you need to determine at what point is data no longer relevant to the current analysis.

Big data clearly deals with issues beyond volume, variety and velocity to other concerns like veracity, validity and volatility.

LET US SEE SOME SUCCESS STORIES

The insights needn't be deep to be valuable. A number of wireless carriers have found ways of using big data analysis to identify likely defectors among their customers and used this to dramatically reduce churn rates.

**1. Amazon**

The online retail giant has access to a massive amount of data on its customers; names, addresses, payments and search histories are all filed away in its data bank.

While this information is obviously put to use in advertising algorithms, Amazon also uses the information to improve customer relations, an area that many big data users overlook.

The next time you contact the Amazon help desk with a query, don't be surprised when the employee on the other end already has most of the pertinent information about you on hand. This allows for a faster, more efficient customer service experience that doesn't include having to spell out your name three times.

**2. American Express**

The American Express Company is using big data to analyse and predict consumer behaviour.

By looking at historical transactions and incorporating more than 100 variables, the company employs sophisticated predictive models in place of traditional business intelligence-based hindsight reporting.

This allows a more accurate forecast of potential churn and customer loyalty. In fact, American Express has claimed that, in their Australian market, they are able to predict 24% of accounts that will close within four months.

**3. BDO**

National accounting and audit firm BDO puts big data analytics to use in identifying risk and fraud during audits.

Where, in the past, finding the source of a discrepancy would involve numerous interviews and hours of manpower, consulting internal data first allows for a significantly narrowed field and streamlined process.

In one case, BDO Consulting Director Katie Tiernan noted, they were able to cut a list of thousands of vendors down to a dozen and, from there, review data individually for inconsistencies. A specific source was identified relatively quickly.

**4. Capital One**

Marketing is one of the most common uses for big data and Capital One are at the top of the game, utilising big data management to help them ensure the success of all customer offerings.

Through analysis of the demographics and spending habits of customers, Capital One determines the optimal times to present various offers to clients, thus increasing the conversion rates from their communications.

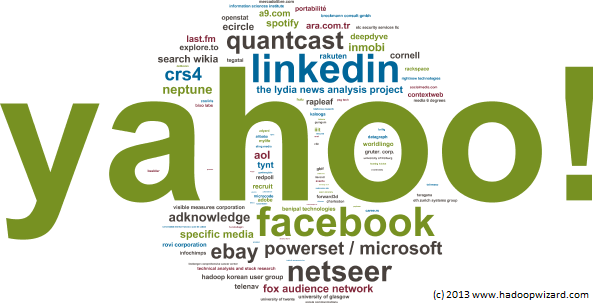
Not only does this result in better uptake but marketing strategies become far more targeted and relevant, therefore improving budget allocation.

**5. General Electric (GE)**

GE is using the data from sensors on machinery like gas turbines and jet engines to identify ways to improve working processes and reliability.

The resultant reports are then passed to GE's analytics team to develop tools and improvements for increased efficiency.

The company has estimated that data could boost productivity in the US by 1.5%, which, over a 20-year period, could save enough cash to raise average national incomes by as much as 30%.



**6. Miniclip**

Miniclip, who develop, publish and distribute digital games globally, use big data to monitor and improve user experience.

Due to the nature of the company and sector, customer retention is a priority for Miniclip in order to make games more profitable and, therefore, to support business growth.

Big data reporting, analysis, experimentation and machine learning data products allow the company to measure the successful elements of their products and implement them in future ventures, while also eliminating or improving the problematic components.

**7. Netflix**

The entertainment streaming service has a wealth of data and analytics providing insight into the viewing habits of millions of international consumers.

Netflix uses this data to commission original programming content that appeals globally as well as purchasing the rights to films and series boxsets that they know will perform well with certain audiences.

For example, Adam Sandler has proven unpopular in the US and UK markets in recent years but Netflix green-lighted four new films with the actor in 2015, armed with the knowledge that his previous work had been successful in Latin America.

**8. Next Big Sound**

Next Big Sound (NBS) has figured out how to use the data from Spotify streams, iTunes sales, SoundCloud plays, Facebook likes, Wikipedia page views, YouTube hits and Twitter mentions to predict the next big thing in music.

The company's analytics provide insight into social media popularity, the impact of TV appearances and many other nuggets of information that are invaluable to the music industry. Artists can also use the data for their own promotion, thanks to a partnership between NBS and Spotify.

Billboard now publishes two charts based exclusively on NBS’s data and they have worked with companies such as Pepsi and American Express to help steer billions being spent brands on music-related marketing and sponsorships.

**9. Starbucks**

Have you ever wondered how Starbucks can open three branches on the same street and not have their business suffer?

The coffeehouse behemoth uses big data to determine the potential success of each new location, taking information on location, traffic, area demographic and customer behaviour into account.

Making this kind of assessment before opening a store means Starbucks can make a fairly accurate estimation of what the success rate will be and choose locations based on the propensity toward revenue growth.

**10. T-Mobile**

The mobile network, like American Express, is combining customer transaction and interactions data to predict customer fluctuations.

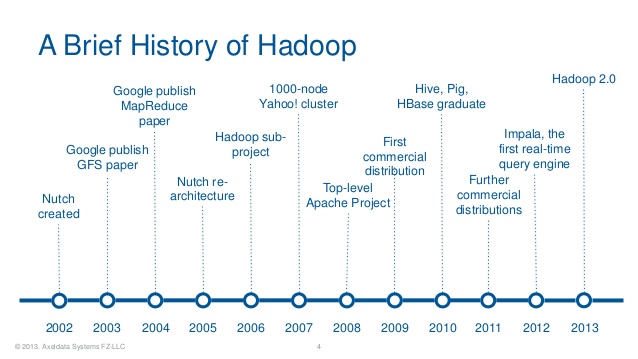
By utilising internal information on billing and customer relations management along with data on social media usage, T-Moblie USA claims they halved customer defections within a single quarter.

The company has integrated the data gathering tools across its' IT systems.

HISTORY OF HADOOP

Following are the major events that led to the creation of the stable version of Hadoop that's available.

* 2003 - Google launches project Nutch to handle billions of searches and indexing millions of web pages.
* Oct 2003 - Google releases papers with GFS (Google File System)
* Dec 2004 - Google releases papers with MapReduce
* 2005 - Nutch used GFS and MapReduce to perform operations
* 2006 - Yahoo! created Hadoop based on GFS and MapReduce (with Doug Cutting and team)
* 2007 - Yahoo started using Hadoop on a 1000 node cluster
* Jan 2008 - Apache took over Hadoop



* Jul 2008 - Tested a 4000 node cluster with Hadoop successfully
* 2009 - Hadoop successfully sorted a petabyte of data in less than 17 hours to handle billions of searches and indexing millions of web pages.
* Dec 2011 - Hadoop releases version 1.0
* Aug 2013 - Version 2.0.6 is available

**Hadoop** is an open-source framework given by Apache Software Foundation for storing huge datasets and processing huge datasets with cluster of commodity hardware. It has two main components:

**1**. **HDFS**- Hadoop distributed file-system is a framework used for storing our data.

**2. MapReduce-** MapReduce is a framework which is used to process the data which we are storing in HDFS.

1. Hadoop Distributed File System(HDFS):-

Hadoop distributed file system is a specially designed file-system for storing large datasets with cluster of commodity hardware and with streaming access pattern. Here cluster is a group of data nodes or network of machines or processors where each of them having their own primary storage(ex-RAM) and secondary storage(ex-harddisk). The hardware used in hadoop file system will be at low cost i.e., we don’t require costly machines for storing and processing our data. Hadoop need not to be worked on highly reliable hardware. It can be worked on simple cheap hardware.

Suppose if we have 500TB of data. It will take 4-5crores of money to maintain that much data on servers whereas 1-1.5crores will be sufficient to maintain on normal machines which we use daily like PCs, laptops etc.

Let us see an example how the files are stored in normal file system and HDFS.

We have a harddisk capacity 500GB where each block consists of 4Kb size. If we have a file size of 2Kb then remaining 2Kb space will be wasted. It is left blank or as free space. This is done in our normal file system.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | Each block 4kb in size |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

But in our HDFS, by default the block is given 128Mb. We install hadoop(HDFS) on top our hard-disk where the memory will not wasted.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | Each block 128Mb in size |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

If we have a file size of 35Mb, it is first divided into blocks and stored in one of the 128Mb block where after storing 93Mb will be remaining as free space which will be released for some other file storing. As hadoop stores large data sets that’s the reason why the block size is default 128Mb.

Features of HDFS:-

Hadoop distributed file system has four important features which makes it idle to use for big data analytics.

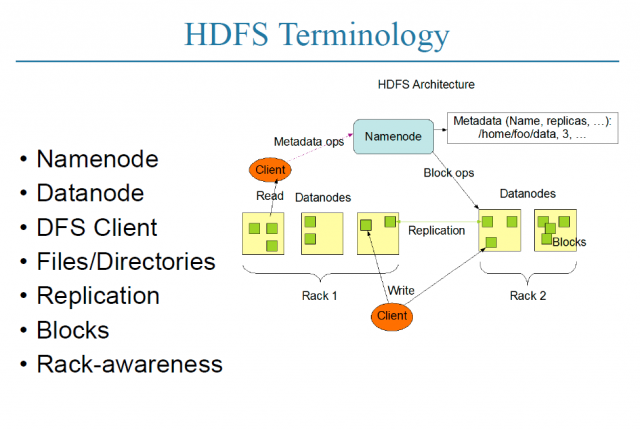
1. Scalable

2. Reliable

3. Economical

4. Flexible





MASTER-SLAVE ARCHIETCTURE:

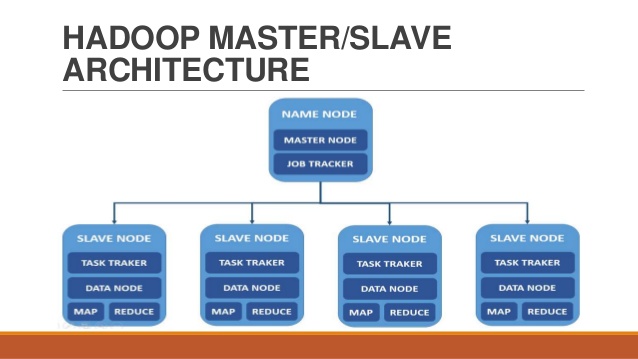
HDFS has two core components:

Master: NameNode

Slave: {Datanode}…..{Datanode}

–     The Master (NameNode) manages the file system namespace operations like opening, closing, and renaming files and directories and determines the mapping of blocks to DataNodes along with regulating access to files by clients

–     Slaves (DataNodes) are responsible for serving read and write requests from the file system’s clients along with perform block creation, deletion, and replication upon instruction from the Master (NameNode).

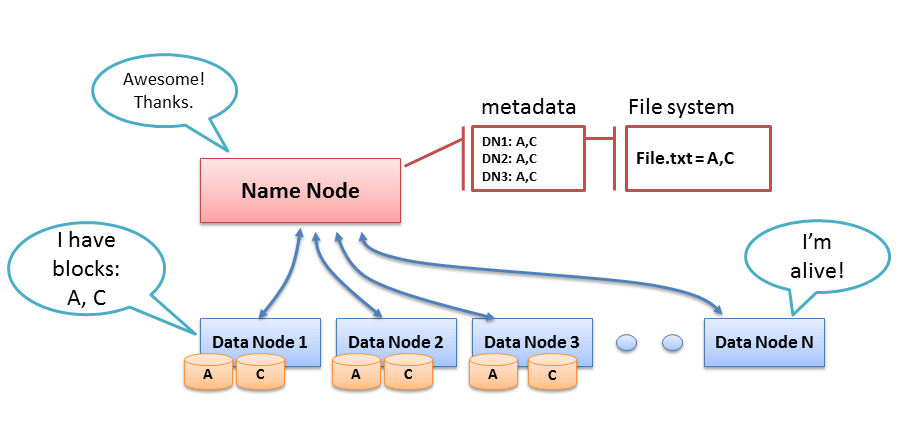


NAME NODE RESPONSIBILITIES:

The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files in the file system, and tracks where across the cluster the file data is kept. It does not store the data of these files itself.

Client applications talk to the NameNode whenever they wish to locate a file, or when they want to add/copy/move/delete a file. The NameNode responds the successful requests by returning a list of relevant [DataNode](https://wiki.apache.org/hadoop/DataNode) servers where the data lives.

The NameNode is a [Single Point of Failure](https://wiki.apache.org/hadoop/Single%20Point%20of%20Failure) for the HDFS Cluster. HDFS is not currently a High Availability system.

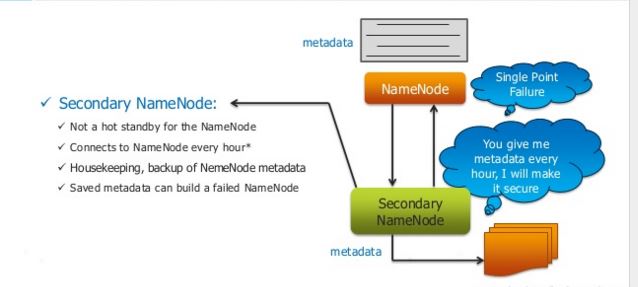


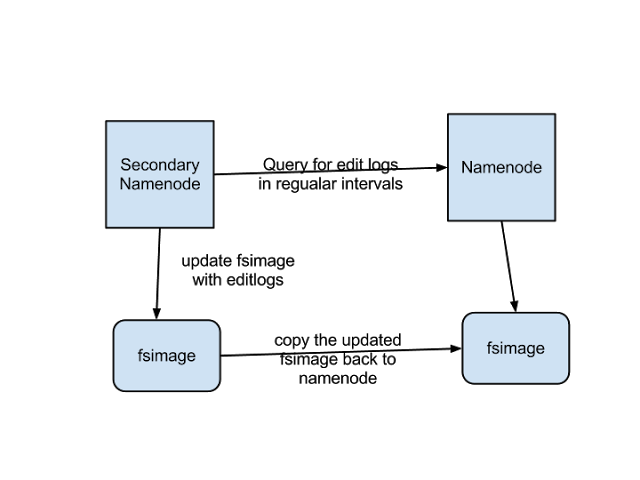
When the NameNode goes down, the file system goes offline. There is an optional [SecondaryNameNode](https://wiki.apache.org/hadoop/SecondaryNameNode) that can be hosted on a separate machine. It only creates checkpoints of the namespace by merging the edits file into the fsimage file and does not provide any real redundancy. Hadoop 0.21+ has a [BackupNameNode](https://wiki.apache.org/hadoop/BackupNameNode) that is part of a plan to have an HA name service, but it needs active contributions from the people who want it (i.e. you) to make it Highly Available.

It is essential to look after the NameNode. Here are some recommendations from production use

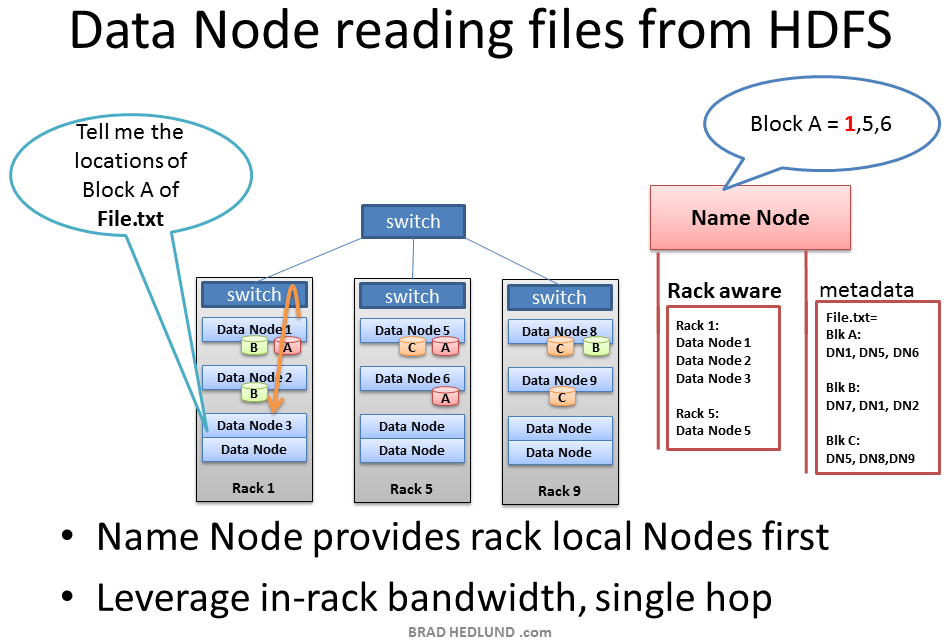
* Use a good server with lots of RAM. The more RAM you have, the bigger the file system, or the smaller the block size.
* Use ECC RAM.
* On Java6u15 or later, run the server VM with compressed pointers -XX:+UseCompressedOops to cut the JVM heap size down.
* List more than one name node directory in the configuration, so that multiple copies of the file system meta-data will be stored. As long as the directories are on separate disks, a single disk failure will not corrupt the meta-data.
* Configure the NameNode to store one set of transaction logs on a separate disk from the image.
* Configure the NameNode to store another set of transaction logs to a network mounted disk.
* Monitor the disk space available to the NameNode. If free space is getting low, add more storage.
* Do not host [DataNode](https://wiki.apache.org/hadoop/DataNode), [JobTracker](https://wiki.apache.org/hadoop/JobTracker) or [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) services on the same system.

If a NameNode does not start up, look at the [TroubleShooting](https://wiki.apache.org/hadoop/TroubleShooting) page.





DATA NODE:



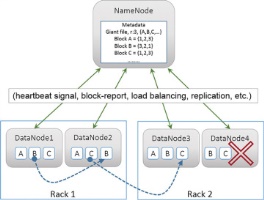
A DataNode stores data in the [[HadoopFileSystem](https://wiki.apache.org/hadoop/HadoopFileSystem)]. A functional filesystem has more than one DataNode, with data replicated across them.

On startup, a DataNode connects to the [NameNode](https://wiki.apache.org/hadoop/NameNode); spinning until that service comes up. It then responds to requests from the [NameNode](https://wiki.apache.org/hadoop/NameNode) for filesystem operations.

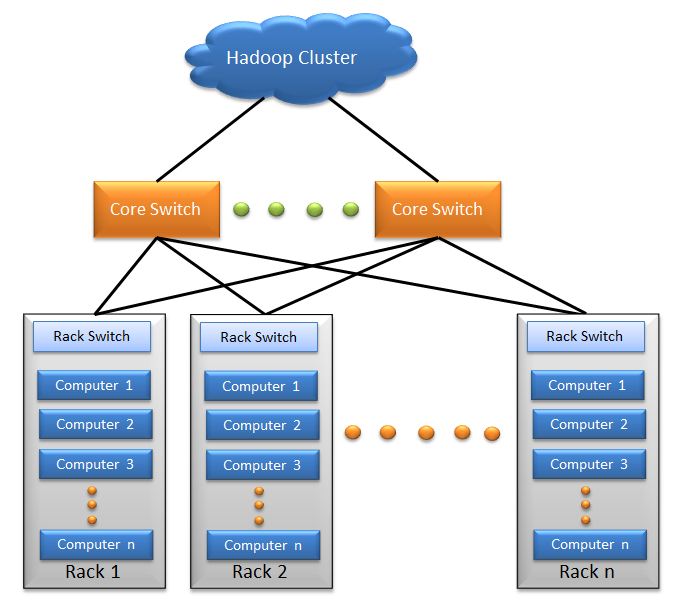
Client applications can talk directly to a DataNode, once the [NameNode](https://wiki.apache.org/hadoop/NameNode) has provided the location of the data. Similarly, [MapReduce](https://wiki.apache.org/hadoop/MapReduce) operations farmed out to [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) instances near a DataNode, talk directly to the DataNode to access the files. [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) instances can, indeed should, be deployed on the same servers that host DataNode instances, so that [MapReduce](https://wiki.apache.org/hadoop/MapReduce) operations are performed close to the data.

DataNode instances can talk to each other, which is what they do when they are replicating data.

* There is usually no need to use RAID storage for DataNode data, because data is designed to be replicated across multiple servers, rather than multiple disks on the same server.
* An ideal configuration is for a server to have a DataNode, a [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker), and then physical disks one [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) slot per CPU. This will allow every [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) 100% of a CPU, and separate disks to read and write data.
* Avoid using [NFS](https://wiki.apache.org/hadoop/NFS) for data storage in production system.



HADOOP CLUSTER SETUP:



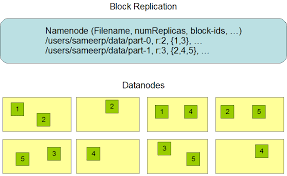
HDFS BLOCK REPLICATION:

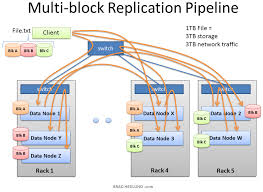
Hadoop Distributed File System (HDFS) is designed to store data on inexpensive, and more unreliable, hardware. *Inexpensive* has an attractive ring to it, but it does raise concerns about the reliability of the system as a whole, especially for ensuring the high availability of the data.

Planning ahead for disaster, the brains behind HDFS made the decision to set up the system so that it would store three (count ’em — three) copies of every data block.

HDFS assumes that every disk drive and every slave node is inherently unreliable, so, clearly, care must be taken in choosing where the three copies of the data blocks are stored.

The figure shows how data blocks from the earlier file are *striped* across the Hadoop cluster — meaning they are evenly distributed between the slave nodes so that a copy of the block will still be available regardless of disk, node, or rack failures.

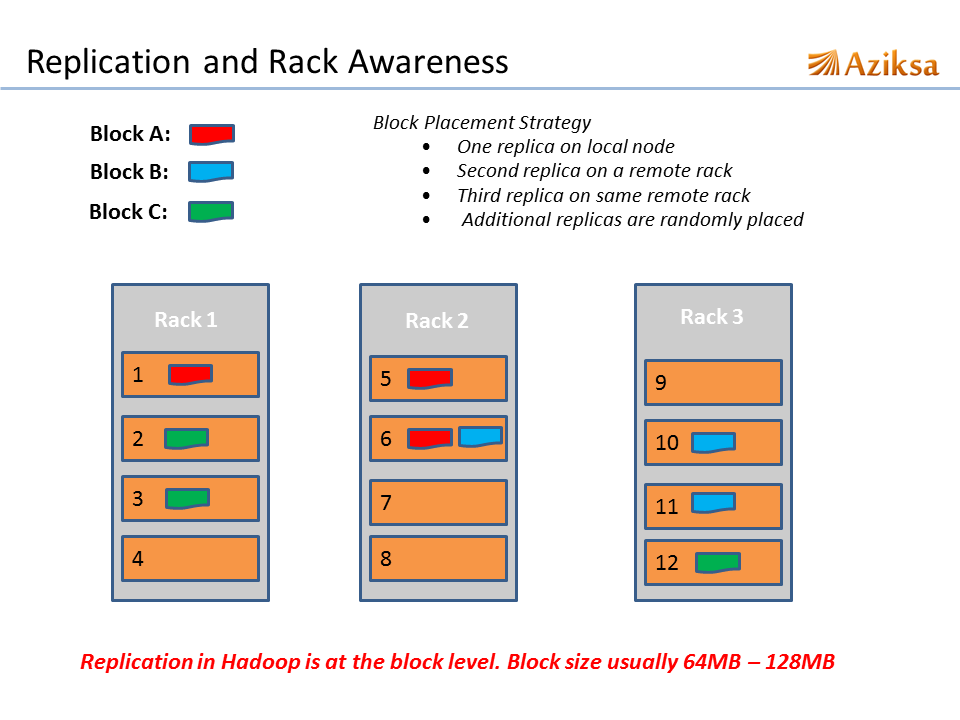




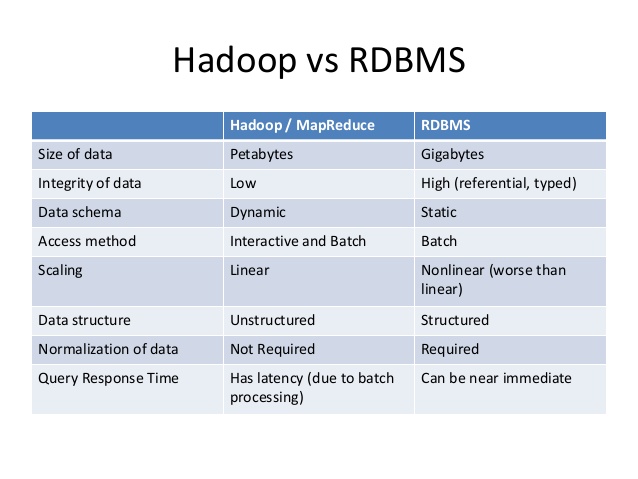
HDFS RACK AWARENESS:

Hadoop components are rack-aware. For example, HDFS block placement will use rack awareness for fault tolerance by placing one block replica on a different rack. This provides data availability in the event of a network switch failure or partition within the cluster.

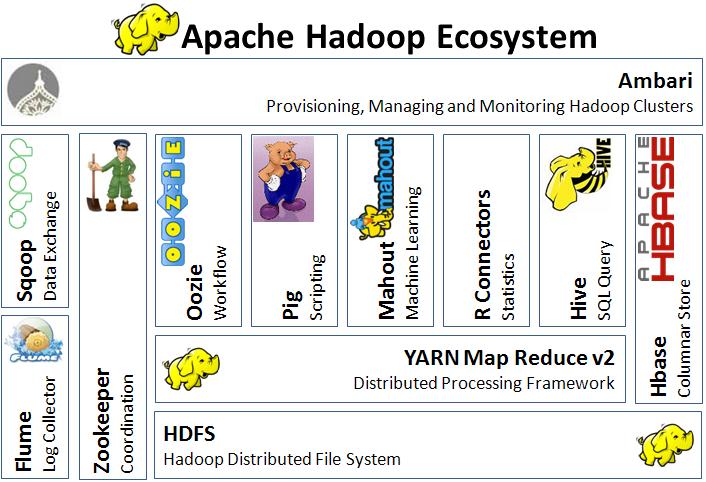
Hadoop master daemons obtain the rack id of the cluster slaves by invoking either an external script or java class as specified by configuration files. Using either the java class or external script for topology, output must adhere to the java







HADOOP ECOSYSTEM:

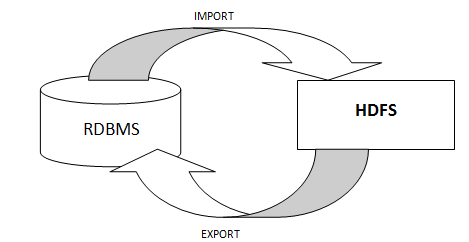


We all know Hadoop is a framework which deals with Big Data but unlike any other frame work it's not a simple framework, it has its own family for processing different thing which is tied up in one umbrella called as Hadoop Ecosystem. Before jumping directly to members of ecosystem let's have a understanding of classification of data. Data is mainly categorized in 3 types under Big Data platform.

* Structured Data - Data which has proper structure and which can be easily stored in tabular form in any relational databases like Mysql, Oracle etc is known as structured data.Example- Employee data .
* Semi-Structured Data - Data which has some structure but cannot be saved in a tabular form in relational databases is known as semi structured data. Example-XML data, email messages etc.
* Unstructured Data - Data which is not having any structure and cannot be saved in tabular form of relational databases is known as unstructured data. Example- Video files, Audio files, Text file etc.

Let's try to understand each component in detail

**SQOOP : SQL + HADOOP = SQOOP**

[](http://hadooptutorials.co.in/tutorials/hadoop/images/understanding-hadoop-ecosystem/sqoop-import-export.PNG)

When we import any structured data from table (RDBMS) to HDFS a file is created in HDFS which we can process by either Map Reduce program directly or by HIVE or PIG. Similarly after processing data in HDFS we can store the processed structured data back to another table in RDBMS by exporting through Sqoop.

**HDFS (Hadoop Distributed File System)**

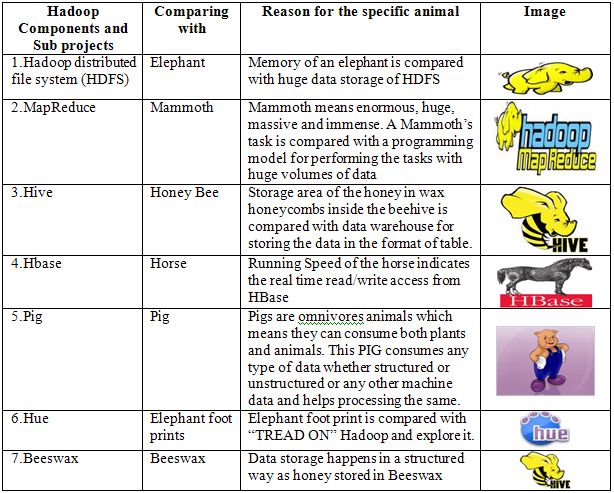
HDFS is a main component of Hadoop and a technique to store the data in distributed manner in order to compute fast. HDFS saves data in a block of 64MB(default) or 128 MB in size which is logical splitting of data in a Datanode (physical storage of data) in Hadoop cluster(formation of several Datanode which is a collection commodity hardware connected through single network). All information about data splits in data node known as metadata is captured in Namenode which is again a part of HDFS.

**MapReduce Framework**

It is another main component of Hadoop and a method of programming in a distributed data stored in a HDFS. We can write Map reduce program by using any language like JAVA, C++ PIPEs, PYTHON, RUBY etc. By name only Map Reduce gives its functionality Map will do mapping of logic into data (distributed in HDFS) and once computation is over reducer will collect the result of Map to generate final output result of MapReduce. MapReduce Program can be applied to any type of data whether Structured or Unstructured stored in HDFS. Example - word count using MapReduce

**HBASE**

Hadoop Database or HBASE is a non-relational (NoSQL) database that runs on top of HDFS. HBASE was created for large table which have billions of rows and millions of columns with fault tolerance capability and horizontal scalability and based on Google Big Table. Hadoop can perform only batch processing, and data will be accessed only in a sequential manner for random access of huge data HBASE is used.

****

**Hive**

Many programmers and analyst are more comfortable with Structured Query Language than Java or any other programming language for which Hive is created by Facebook and later donated to Apache foundation. Hive mainly deals with structured data which is stored in HDFS with a Query Language similar to SQL and known as HQL (Hive Query Language). Hive also run Map reduce program in a backend to process data in HDFS but here programmer has not worry about that backend MapReduce job it will look similar to SQL and result will be displayed on console.

**Pig**

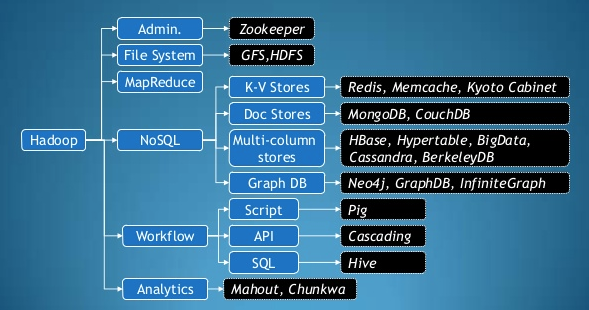
Similar to HIVE, PIG also deals with structured data using PIG LATIN language. PIG was originally developed at Yahoo to answer similar need to HIVE. It is an alternative provided to programmer who loves scripting and don't want to use Java/Python or SQL to process data. A Pig Latin program is made up of a series of operations, or transformations, that are applied to the input data which runs MapReduce program in backend to produce output.

**Mahout**

Mahout is an open source machine learning library from Apache written in java. The algorithms it implements fall under the broad umbrella of machine learning or collective intelligence. This can mean many things, but at the moment for Mahout it means primarily recommender engines (collaborative filtering), clustering, and classification. Mahout aims to be the machine learning tool of choice when the collection of data to be processed is very large, perhaps far too large for a single machine. In its current incarnation, these scalable machine learning implementations in Mahout are written in Java, and some portions are built upon Apache's Hadoop distributed computation project.

**Oozie**

It is a workflow scheduler system to manage hadoop jobs. It is a server-based Workflow Engine specialized in running workflow jobs with actions that run Hadoop MapReduce and Pig jobs. Oozie is implemented as a Java Web-Application that runs in a Java Servlet-Container. Hadoop basically deals with bigdata and when some programmer wants to run many job in a sequential manner like output of job A will be input to Job B and similarly output of job B is input to job C and final output will be output of job C. To automate this sequence we need a workflow and to execute same we need engine for which OOZIE is used.



**Zookeeper**

Writing distributed applications is difficult because of partial failure may occur between nodes to overcome this Apache Zookeper has been developed by maintaining an open-source server which enables highly reliable distributed coordination. ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services . In case of any partial failure clients can connect to any node and be assured that they will receive the correct, up-to-date information.