**EXPERIMENT 01**

**Part 1**

CLASS: BE CMPN A ROLL NO. : 19

NAME: REBECCA DIAS PID: 182027

Aim:- (i)To study the fundamentals of HDFS Basics and Hadoop Ecosystem.

(ii) To get acquainted with the installation of Hadoop

Theory:-

**A.** **HDFS Basics: Introduction of BigData**

HDFS is a distributed file system that handles large data sets running on commodity hardware. It is used to scale a single Apache Hadoop cluster to hundreds (and even thousands) of nodes. HDFS is one of the major components of [Apache Hadoop](https://www.ibm.com/analytics/hadoop), the others being [MapReduce](https://www.ibm.com/analytics/hadoop/mapreduce) and [YARN](https://www.ibm.com/developerworks/library/bd-yarn-intro). HDFS should not be confused with or replaced by [Apache HBase](https://www.ibm.com/analytics/hadoop/hbase), which is a column-oriented non-relational database management system that sits on top of HDFS and can better support real-time data needs with its in-memory processing engine.

Goals of HDFS:

1.Fast recovery from hardware failures.

2.Access to streaming data

3. Accommodation of large data sets

4.Portability

HDFS has two core components, i.e. NameNode and DataNode.

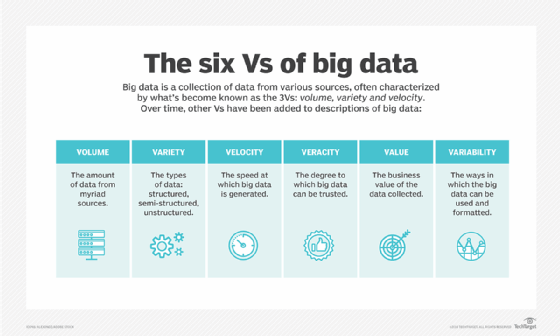
1. The NameNode is the main node and it doesn’t store the actual data. It contains metadata, just like a log file or you can say as a table of content. Therefore, it requires less storage and high computational resources.

2. On the other hand, all your data is stored on the DataNodes and hence it requires more storage resources. These DataNodes are commodity hardware (like your laptops and desktops) in the distributed environment. That’s the reason, why Hadoop solutions are very cost effective.

3. You always communicate to the NameNode while writing the data. Then, it internally sends a request to the client to store and replicate data on various DataNodes.

Big Data refers to the datasets too large and complex for traditional systems to store and process. The major problems faced by Big Data majorly falls under Six Vs. They are volume, velocity, and variety ,value, variability & veracity.

Companies [use big data in their systems](https://searchbusinessanalytics.techtarget.com/feature/8-big-data-use-cases-for-businesses-and-industry-examples) to improve operations, provide better customer service, create personalized marketing campaigns and take other actions that, ultimately, can increase revenue and profits. Businesses that use it effectively hold a potential competitive advantage over those that don't because they're able to make faster and more informed business decisions.



1. **Introduction to Hadoop**

#### Hadoop was invented to store large datasets, handling data in different formats, data getting generated with high speed.Apache Hadoop is an open source software framework used to develop data processing applications which are executed in a distributed computing environment.

Similar to data residing in a local file system of a personal computer system, in Hadoop, data resides in a distributed file system which is called as a Hadoop Distributed File system. The processing model is based on 'Data Locality' concept wherein computational logic is sent to cluster nodes(server) containing data. This computational logic is nothing, but a compiled version of a program written in a high-level language such as Java. Such a program, processes data stored in Hadoop HDFS.

#### 

1. **History of Hadoop**

In 2002, Doug Cutting and Mike Cafarella were working on Apache Nutch Project that aimed at building a web search engine that would crawl and index websites.

After a lot of research, Mike Cafarella and Doug Cutting estimated that it would cost around $500,000 in hardware.This project proved to be too expensive and thus found infeasible for indexing billions of webpages. So they were looking for a feasible solution that would reduce the cost.

In 2004, Nutch’s developers set about writing an open-source implementation, the Nutch Distributed File System (NDFS).In 2004, Google introduced MapReduce to the world by releasing a paper on MapReduce. This paper provided the solution for processing those large datasets. It gave a full solution to the Nutch developers.

Google provided the idea for distributed storage and MapReduce.

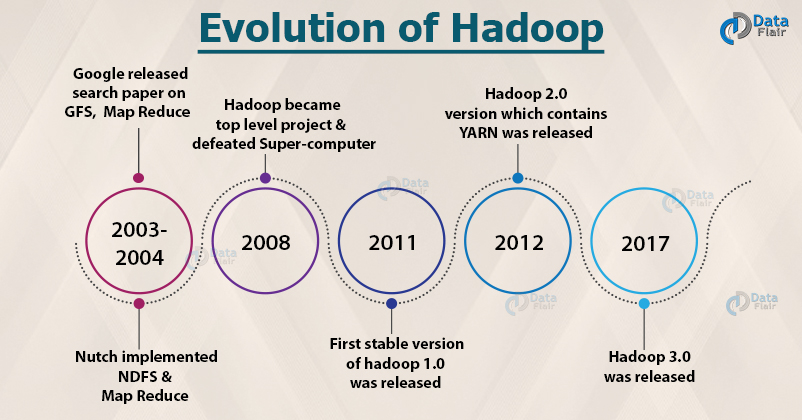
The Apache community realized that the implementation of MapReduce and NDFS could be used for other tasks as well. In February 2006, they came out of Nutch and formed an independent subproject of Lucene called “**Hadoop**” (which is the name of Doug’s kid’s yellow elephant).

In January 2008, Hadoop confirmed its success by becoming the top-level project at Apache.By this time, many other companies like Last.fm, Facebook, and the New York Times started using Hadoop.

In April 2008, Hadoop defeated supercomputers and became the fastest system on the planet by sorting an entire terabyte of data.

In November 2008, Google reported that its Mapreduce implementation sorted 1 terabyte in 68 seconds.

In April 2009, a team at Yahoo used Hadoop to sort 1 terabyte in 62 seconds, beaten Google MapReduce implementation.

On 6 April 2018, Hadoop release 3.1.0 came that contains improvements.

**D.Map Reduce**

In today’s data-driven market, algorithms and applications are collecting data 24/7 about people, processes, systems, and organizations, resulting in huge volumes of data. The challenge, though, is how to process this massive amount of data with speed and efficiency, and without sacrificing meaningful insights.This is where the MapReduce programming model comes to rescue.

MapReduce is a programming model or pattern within the Hadoop framework that is used to access big data stored in the Hadoop File System (HDFS). It is a core component, integral to the functioning of the Hadoop framework.

MapReduce facilitates concurrent processing by splitting petabytes of data into smaller chunks, and processing them in parallel on Hadoop commodity servers. In the end, it aggregates all the data from multiple servers to return a consolidated output back to the application.

For example, a Hadoop cluster with 20,000 inexpensive commodity servers and 256MB block of data in each, can process around 5TB of data at the same time. This reduces the processing time as compared to sequential processing of such a large data set.

With MapReduce, rather than sending data to where the application or logic resides, the logic is executed on the server where the data already resides, to expedite processing. Data access and storage is disk-based—the input is usually stored as files containing structured, semi-structured, or unstructured data, and the output is also stored in files

The Map function takes input from the disk as <key,value> pairs, processes them, and produces another set of intermediate <key,value> pairs as output.

The Reduce function also takes inputs as <key,value> pairs, and produces <key,value> pairs as output.

**E.Shortcomings of Hadoop 1.0**

**1. Issue With Small Files**

Hadoop is suitable for a small number of large files but when it comes to the application which deals with a large number of small files, Hadoop fails here. A small file is nothing but a file which is significantly smaller than Hadoop’s block size which can be either 128MB or 256MB by default. These large number of small files overload the Namenode as it stores namespace for the system and makes it difficult for Hadoop to function.

**2. Vulnerable By Nature**

Hadoop is written in Java which is a widely used programming language hence it is easily exploited by cyber criminals which makes Hadoop vulnerable to security breaches.

**3. Processing Overhead**

In Hadoop, the data is read from the disk and written to the disk which makes read/write operations very expensive when we are dealing with tera and petabytes of data. Hadoop cannot do in-memory calculations hence it incurs processing overhead.

**4. Supports Only Batch Processing**

At the core, Hadoop has a batch processing engine which is not efficient in stream processing. It cannot produce output in real-time with low latency. It only works on data which we collect and store in a file in advance before processing.

**5. Iterative Processing**

Hadoop cannot do iterative processing by itself. Machine learning or iterative processing has a cyclic data flow whereas Hadoop has data flowing in a chain of stages where output on one stage becomes the input of another stage.

**6. Security:**

For security, Hadoop uses Kerberos authentication which is hard to manage. It is missing encryption at storage and network levels which are a major point of concern.

**F.HDFS Features**

**1. Fault Tolerance**

The fault tolerance in Hadoop HDFS is the working strength of a system in unfavorable conditions. It is highly fault-tolerant. Hadoop framework divides data into blocks. After that creates multiple copies of blocks on different machines in the cluster.

So, when any machine in the cluster goes down, then a client can easily access their data from the other machine which contains the same copy of data blocks.

**2. High Availability**

Hadoop HDFS is a highly available file system. In HDFS, data gets replicated among the nodes in the Hadoop cluster by creating a replica of the blocks on the other slaves present in HDFS cluster. So, whenever a user wants to access this data, they can access their data from the slaves which contain its blocks.

At the time of unfavorable situations like a failure of a node, a user can easily access their data from the other nodes. Because duplicate copies of blocks are present on the other nodes in the HDFS cluster.

**3. High Reliability**

HDFS provides reliable data storage. It can store data in the range of 100s of petabytes. HDFS stores data reliably on a cluster. It divides the data into blocks. Hadoop framework stores these blocks on nodes present in HDFS cluster.

HDFS stores data reliably by creating a replica of each and every block present in the cluster. Hence provides fault tolerance facility. If the node in the cluster containing data goes down, then a user can easily access that data from the other nodes.

HDFS by default creates 3 replicas of each block containing data present in the nodes. So, data is quickly available to the users. Hence user does not face the problem of data loss. Thus, HDFS is highly reliable.

**4. Replication**

Data Replication is unique features of HDFS. Replication solves the problem of data loss in an unfavorable condition like hardware failure, crashing of nodes etc. HDFS maintain the process of replication at regular interval of time.

HDFS also keeps creating replicas of user data on different machine present in the cluster. So, when any node goes down, the user can access the data from other machines. Thus, there is no possibility of losing of user data.

**5. Scalability**

Hadoop HDFS stores data on multiple nodes in the cluster. So, whenever requirements increase you can scale the cluster. Two scalability mechanisms are available in HDFS: Vertical and Horizontal Scalability.

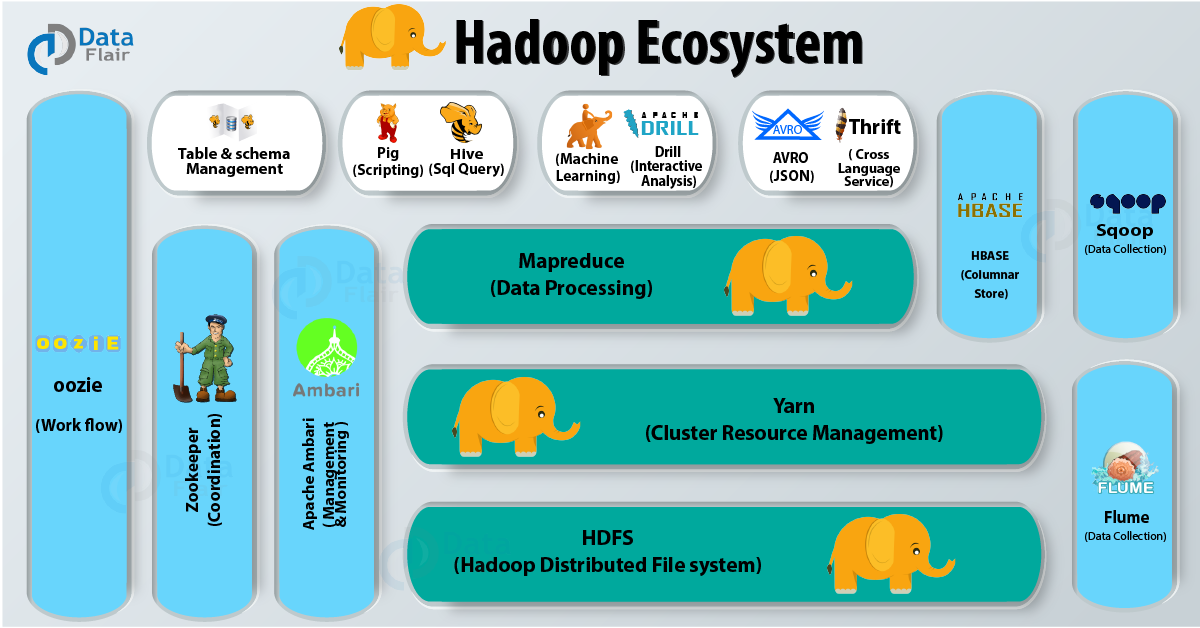
**6. Distributed Storage**

All the features in HDFS are achieved via distributed storage and replication. HDFS store data in a distributed manner across the nodes. In Hadoop, data is divided into blocks and stored on the nodes present in the HDFS cluster. 

After that HDFS create the replica of each and every block and store on other nodes. When a single machine in the cluster gets crashed we can easily access our data from the other nodes which contain its replica.

**G.Hadoop EcoSystem & Tools in Hadoop Ecosystem**

The Hadoop ecosystem components are actually different services deployed by the various enterprise.



**1. Hadoop Distributed File System**

Hadoop Ecosystem Component

HDFS is the foundation of Hadoop and hence is a very important component of the Hadoop ecosystem. It is Java software that provides many features like scalability, high availability, fault tolerance, cost effectiveness etc. It also provides robust distributed data storage for Hadoop. We can deploy many other software frameworks over HDFS.

**2. MapReduce**

MapReduce is the data processing component of Hadoop. It applies the computation on sets of data in parallel thereby improving the performance. MapReduce works in two phases –

Map Phase – This phase takes input as key-value pairs and produces output as key-value pairs. It can write custom business logic in this phase. Map phase processes the data and gives it to the next phase.

Reduce Phase – The MapReduce framework sorts the key-value pair before giving the data to this phase. This phase applies the summary type of calculations to the key-value pairs.

**3. Yarn**

Yarn which is short for Yet Another Resource Manager. It is like the operating system of Hadoop as it monitors and manages the resources. Yarn came into the picture with the launch of Hadoop 2.x in order to allow different workloads. It handles the workloads like stream processing, interactive processing, batch processing over a single platform. Yarn has two main components – Node Manager and Resource Manager.

But it does not monitor the status of the application. So in the event of failure of the task, it does not restart the same.

We have another concept called Container. It is nothing but a fraction of NodeManager capacity i.e. CPU, memory, disk, network etc.

**4. Hive**

Hive is a data warehouse project built on the top of Apache Hadoop which provides data query and analysis. It has got the language of its own call HQL or Hive Query Language. HQL automatically translates the queries into the corresponding map-reduce job.

**5. Pig**

Pig is a SQL like language used for querying and analyzing data stored in HDFS. Yahoo was the original creator of the Pig. It uses pig latin language. It loads the data, applies a filter to it and dumps the data in the required format. Pig also consists of JVM called Pig Runtime. Various features of Pig are as follows:-

**6. HBase**

HBase is a NoSQL database built on the top of HDFS. The various features of HBase are that it is open-source, non-relational, distributed database. It imitates Google’s Bigtable and written in Java. It provides real-time read/write access to large datasets.

**7. Mahout**

Mahout provides a platform for creating machine learning applications which are scalable.

**8. Zookeeper**

Zookeeper coordinates between various services in the Hadoop ecosystem. It saves the time required for synchronization, configuration maintenance, grouping, and naming.

**9. Oozie**

It is a workflow scheduler systems for managing Hadoop jobs. It supports Hadoop jobs for Map-Reduce, Pig, Hive, and Sqoop. Oozie combines multiple jobs into a single unit of work. It is scalable and can manage thousands of workflow in a Hadoop cluster. Oozie works by creating DAG i.e. Directed Acyclic Graph of the workflow. It is very much flexible as it can start, stop, suspend and rerun failed jobs.

Oozie is an open-source web-application written in Java. Oozie is scalable and can execute thousands of workflow containing dozens of Hadoop jobs.

**10. Sqoop**

Sqoop imports data from external sources into compatible Hadoop Ecosystem components like HDFS, Hive, HBase etc. It also transfers data from Hadoop to other external sources. It works with RDBMS like TeraData, Oracle, MySQL and so on. The major difference between Sqoop and Flume is that Flume does not work with structured data. But Sqoop can deal with structured as well as unstructured data.