# EXPERIMENT NO 1

**AIM:** Study of Hadoop Ecosystem

### THEORY:

Big data is the term for a collection of structured and unstructured data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. 3Vs of Big Data are:

Volume - large amount of data

Velocity - needs to be analyzed quickly

Variety - structured and unstructured

Data is growing at exponential rates. Problems associated with it:

* Need to process data in real-time
* Hardware performance growth slowed:

**History of Hadoop:**

Hadoop is a free, Java-based programming framework that supports the processing of large data sets in a distributed computing environment. It is part of the Apache project sponsored by the Apache Software Foundation.

Hadoop was created by Doug Cutting [the creator of Apache Lucene] and Mike Cafarella in 2005.

Cutting, who was working at Yahoo! at the time, named it after his son’s toy elephant. It was originally developed to support distribution for the Nutch search engine project.During this time, another search engine project called Google was in progress. It was based on the same concept – storing and processing data in a distributed, automated way so that relevant web search results could be returned faster.

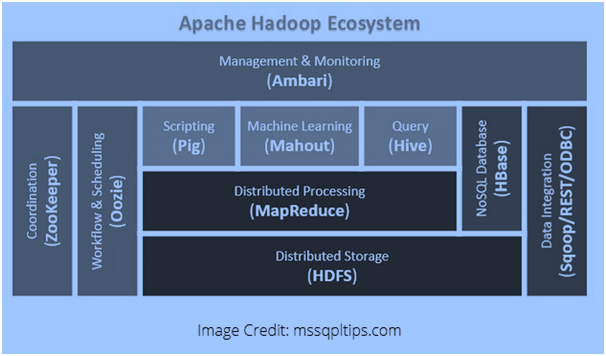
In 2006, Cutting joined Yahoo and took with him the Nutch project as well as ideas based on Google’s early work with automating distributed data storage and processing.

The Nutch project was divided – the web crawler portion remained as Nutch and the distributed computing and processing portion became Hadoop (named after Cutting’s son’s toy elephant).

In 2008, Yahoo released Hadoop as an open-source project. Today, Hadoop’s framework and ecosystem of technologies are managed and maintained by the non-profit Apache Software Foundation (ASF), a global community of software developers and contributors.

#### Hadoop Ecosystem:

The Hadoop ecosystem is diverse and grows by the day. It’s impossible to keep track of all of the various projects that interact with Hadoop in some form.



Hadoop ecosystem Hadoop Core consists of the following components:

#### HDFS Hadoop Distributed File System

Large files are split into blocks or partitions.

Default is to store 3 copies of each partition. Two local copies and 1 remote copy (rack or subnet aware).

Master-slave architecture.

Name Node (Master): manages files and blocks and runs on master node. Stores the metadata of the HDFS. One Name Node per cluster.

Data Node (Slave): stores blocks and runs on slave nodes. There can be thousands of those.

Hadoop 2 added Backup Node and Checkpoint Node (replaces Secondary Node).

Clients access HDFS via API or command line.

1. **Hadoop MapReduce:** An implementation of MapReduce programming model

Programming model for distributed processing.

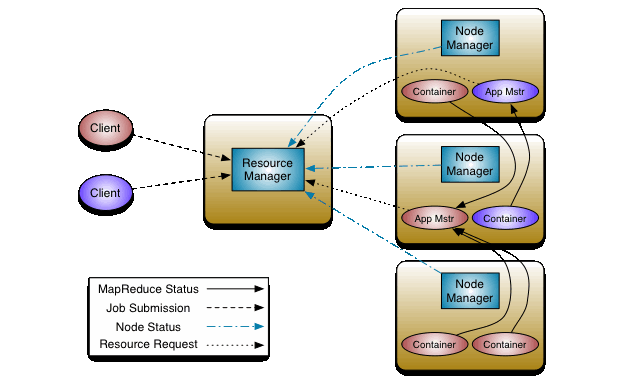
MapReduce engine consists of JobTracker and TaskTracker

JobTracker runs on master node, client submit jobs to it. It pushes work to TaskTracker. Rack aware, tries to assign tasks to nodes that contain the data or a node near it.

TaskTracker: runs on a slave node, spawns a JVM for each task, holds ‘available’ slots where tasks are pushed by the JobTracker. Speculative scheduling for slow running TaskTracker.

Scheduling: Default is FIFO. Fair and Capacity Scheduling was added as options.

1. **Apache Hadoop YARN**

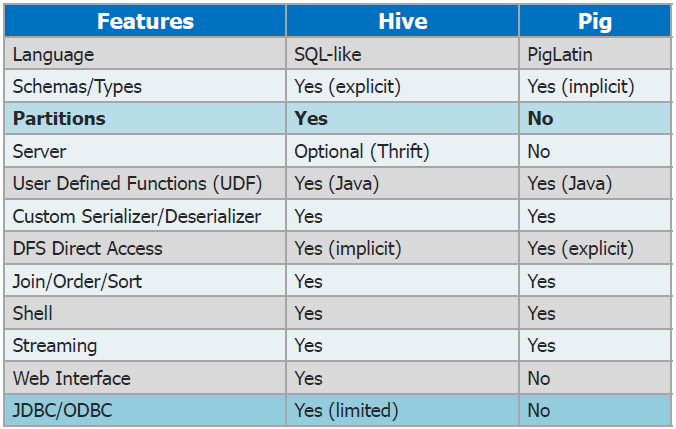


The fundamental idea of YARN is to split up the functionalities of resource management and job scheduling/monitoring into separate daemons. The idea is to have a global ResourceManager (RM) and per-application ApplicationMaster (AM). An application is either a single job or a DAG of jobs.

The ResourceManager and the NodeManager form the data-computation framework. The ResourceManager is the ultimate authority that arbitrates resources among all the applications in the system. The NodeManager is the per-machine framework agent who is responsible for containers, monitoring their resource usage (cpu, memory, disk, network) and reporting the same to the ResourceManager/Scheduler.

The per-application ApplicationMaster is, in effect, a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks.

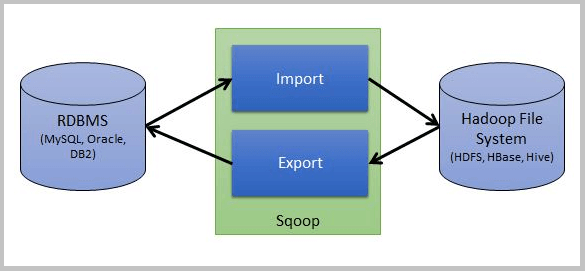
#### Difference between [Hive](https://www.dezyre.com/Big-Data-and-Hadoop/19) and Pig:



**HBase:** HBase is a data model that is similar to Google’s big table designed to provide quick random access to huge amounts of structured data. This tutorial provides an introduction to HBase, the procedures to set up HBase on Hadoop File Systems, and ways to interact with HBase shell. It also describes how to connect to HBase using java, and how to perform basic operations on HBase using java.

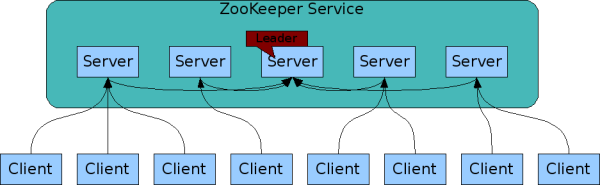
**Flume:** HBase is a non-relational database that allows for low-latency, quick lookups in Hadoop. It adds transactional capabilities to Hadoop, allowing users to conduct updates,inserts and deletes. EBay and Facebook use HBase heavily. Flume helps to collect data from a variety of sources, like logs, jms, Directory etc. Multiple flume agents can be configured to collect high volume of data.

**Sqoop:** A tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases (sqoop.apache.org). It is capable of transferring the data in both directions. It supports incremental imports. It supports any database that supports JDBC and allows for custom connectors. It is free but not open source.



**Oozie:** It is the workflow scheduler for Hadoop jobs. It is a Scalable, Reliable and Extensible system designed for complex search engine workflows. It directed Acyclic Graphs (DAG) of actions. It is a "control dependency" from one action to another. In this the second action can't run until the first action has completed. It was developed by Yahoo and Licensed by Apache and its latest Release is Oozie4.0.0 95 MSME

**Zookeeper:** ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. All of these kinds of services are used in some form or another by distributed applications. Each time they are implemented there is a lot of work that goes into fixing the bugs and race conditions that are inevitable. Because of the difficulty of implementing these kinds of services, applications initially usually skimp on them, which make them brittle in the presence of change and difficult to manage. Even when done correctly, different implementations of these services lead to management complexity when the applications are deployed.



**NOSQL:**

Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontally scalable.The original intention has been modern web-scale databases.

Features:

a. Simple data model using key-value pairs with secondary indexes

b. Simple programming model with ACID transactions, tabular data models, and JSON support

c. Application security with authentication and session-level SSL encryption

d. Integrated with Oracle Database, Oracle Wallet, and Hadoop

e. Geo-distributed data with support for multiple data centers

f. High availability with local and remote failover and synchronization

g. Scalable throughput and bounded latency

**Mahout:**

Apache Mahout is a library of scalable machine-learning algorithms, implemented on top of Apache Hadoop and using the MapReduce paradigm. Machine learning is a discipline of artificial intelligence focused on enabling machines to learn without being explicitly programmed, and it is commonly used to improve future performance based on previous outcomes.

Features:

a. Collaborative filtering: mines user behavior and makes product recommendations (e.g. Amazon recommendations)

b. Clustering: Takes items in a particular class (such as web pages or newspaper articles) and organizes them into naturally occurring groups, such that items belonging to the same group are similar to each other

c. Classification: learns from existing categorizations and then assigns unclassified items to the best category

**Lucene/Solr:**

There is but one tool for indexing large blocks of unstructured text, and it’s a natural partner for Hadoop. Written in Java, Lucene integrates easily with Hadoop, creating one big tool for distributed text management. Lucene handles the indexing; Hadoop distributes queries across the cluster.

Features:

a. Advanced Full-Text Search Capabilities

b. Optimized for High Volume Web Traffic

c. Standards Based Open Interfaces – XML, JSON and HTTP

d. Comprehensive HTML Administration Interfaces

e. Server statistics exposed over JMX for monitoring

f. Linearly scalable, auto index replication, auto failover and recovery

**Avro:**

Avro provides a convenient way to represent complex data structures within a Hadoop MapReduce job. Avro data can be used as both input to and output from a MapReduce job, as well as the intermediate format. The example in this guide uses Avro data for all three, but it’s possible to mix and match; for instance, MapReduce can be used to aggregate a particular field in an Avro record.

Features:

a. Near Real-time indexing

b. Flexible and Adaptable with XML configuration

c. Extensible Plugin Architecture

d. Frequent itemset mining – analyzes items in a group (e.g. items in a shopping cart or terms in a query session) and then identifies which items typically appear together

**GIS tools:**

The world is a big place and working with geographic maps is a big job for clusters running Hadoop. The GIS (Geographic Information Systems) tools for Hadoop project has adapted some of the best Java-based tools for understanding geographic information to run with Hadoop. Your databases can handle geographic queries using coordinates instead of strings. Your code can deploy the GIS tools to calculate in three dimensions. The trickiest part is figuring out when the word “map” refers to a flat

thing that represents the world and when “map” refers to the first step in a Hadoop job

Features:

a. Run Filter and aggregate operations on billions of spatial data records inside Hadoop based on spatial criteria.

b. Define new areas represented as polygons, and run Point in Polygon analysis on billions of spatial data records inside Hadoop.

c. Visualize analysis results on a map with rich styling capabilities, and a rich set of base maps.

d. Integrate your maps in reports, or publish them as map applications online.

**Spark:**

Apache Spark is an open-source data analytics cluster computing framework originally developed in the AMPLab at UC Berkeley. Spark fits into the Hadoop open-source community, building on top of the Hadoop Distributed File System (HDFS). However, Spark is not tied to the two-stage MapReduce paradigm, and promises performance up to 100 times faster than Hadoop MapReduce for certain applications.Spark provides primitives for in-memory cluster computing that allows user programs to load data into a cluster’s memory and query it repeatedly, making it well suited to machine learning algorithms

Features:

a. Proven scalability to 100 nodes in the research lab and 80 nodes in production at Yahoo

b. Ability to cache datasets in memory for interactive data analysis: extract a working set, cache it, query it repeatedly.

c. Interactive command line interface (in Scala or Python) for low-latency data exploration at scale.

d. Higher level library for stream processing, through Spark Streaming.

e. Higher level libraries for machine learning and graph processing that because of the distributed memory-based Spark architecture are ten times as fast as Hadoop disk-based Apache Mahout and even scale better than Vowpal Wabbit.

**Ambari:**

The Apache Ambari project is aimed at making Hadoop management simpler by developing software for provisioning, managing, and monitoring Apache Hadoop clusters. Ambari provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.

Features:

a. Ambari provides a dashboard for monitoring health and status of the Hadoop cluster.

b. Ambari leverages Ganglia for metrics collection.

c. Ambari leverages Nagios for system alerting and will send emails when your attention is needed (e.g., a node goes down, remaining disk space is low, etc)

**Advantages of Hadoop:**

* Distributed (runs on master-slave cluster)
* Scalable (1000’s of nodes)
* Fault-Tolerant (built-in redundancy)
* Cost-efficient (commodity hardware)
* Open-Source (Apache projects)
* Free (Apache License 2.0)
* Used and supported by major corporations (Google, Yahoo!, IBM, ebay, facebook, etc.)
* Commercial distributions from companies like Cloudera and Hortonworks

**CONCLUSION:** The Hadoop ecosystem and its core components were studied successfully in this experiment.