**BIG DATA**

**IOT-** Any physical devices connect through internet and help in smart works.

Big Data - Big data is the term for a collection of data sets so LARGE and COMPLEX that it becomes difficult to process using on-hand database system tools or traditional data processing applications.

**5 v’s of big data**

1. Volume - Amount of data been added from source to warehouse.
2. Variety (Structured[Table], Semi-structured[json, csv, email, xml], and unstructured[log, audio, video, image)
3. Velocity - At what frequently the data is added from source to data warehouse.
4. Value - After processing, data should have value.
5. Veracity[accuracy](uncertainty and inconsistencies in the data)

**Problems with big data:**

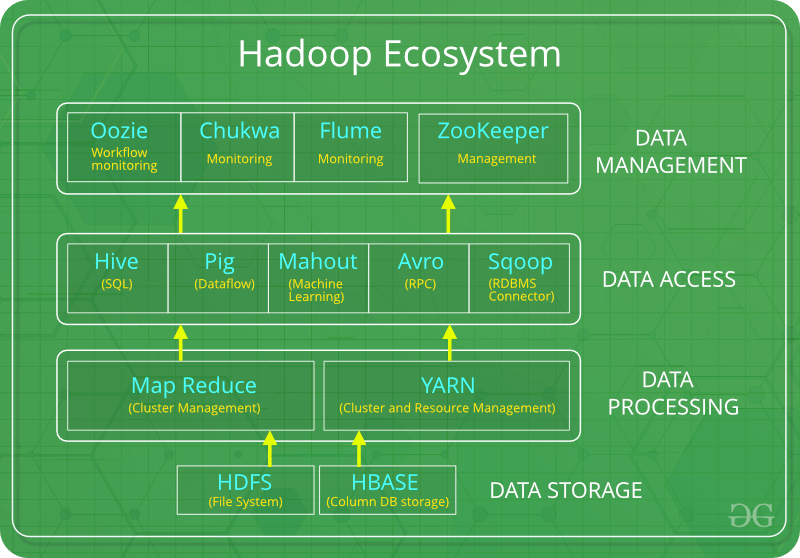
Storing exponentially growing huge dataset,

Processing data having complex structured,

Processing data faster.

**So tools like Hadoop, spark, Hbase, and MongoDB came to picture to solve and process complex data.**

**Hadoop**

****

Hadoop is a framework that allows a user to store and process large data sets in distributed and parallel fashion.

Hadoop helps in horizontal scaling up. If there is 4TB of data and there are 10 clusters, the data is stored into a cluster.

**Hadoop has mainly two core components in 1.0 version:**

1: HDFS - To address the storage wise issue.

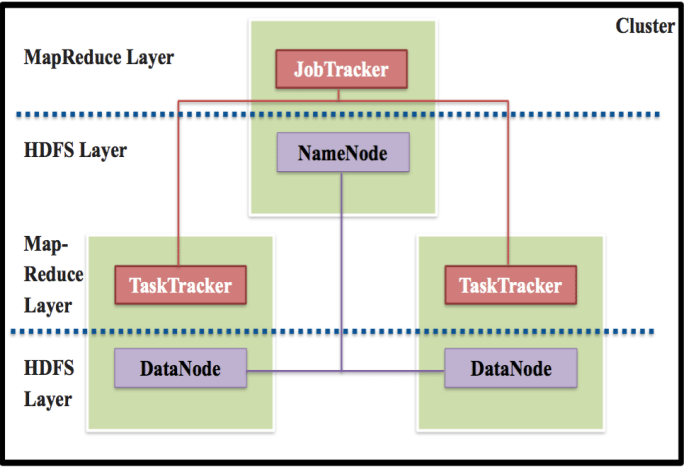
2: Map Reduce- To address the processing related issue.

**Hadoop works in Master salve architecture. Like under team lead, there will be 5 people(cluster) working in parallelly to complete the work faster. The master node will have metadata about all the data stored in clusters(in which and all the data is stored). While processing based on metadata in master node, it is going to send the logic where the data is(if the cluster not have data then no need to work on logic)**

**Hadoop Eco System**

**HDFS**

**Mapreduce - you submit a job to a job\_tracker[Allocating Resource, Managing task] -> The role of the job tracker is, it will tell each node that this much of resource(cpu, memory) you need for the computation/processing. The job of task tracker to report status of the node to job tracker.**

****

**Pig -** It is a scripting language (Latin script - different from sql script) which is alternative to MapReuce. It is a tool/platform which is used to analyze larger sets of data, representing them as data flows.

**Hive** - Same as SQL, Hive allows users to read, write, and manage petabytes of data using SQL. Hive is built on top of Apache Hadoop, which is an open-source framework used to efficiently store and process large datasets.

**Hbase** - HBase is a column-oriented non-relational database management system that runs on top of Hadoop Distributed File System (HDFS).

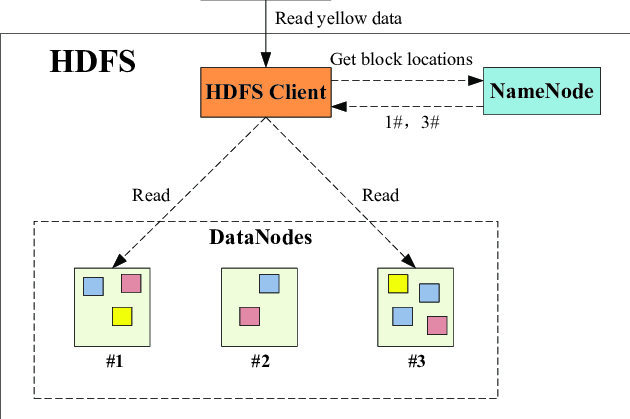
**Sqoop** - Apache Sqoop is a big data tool for transferring data between Hadoop and relational database servers.

**Flume** - Apache Flume is a tool for data ingestion mechanism for collecting aggregating and transporting large amounts of streaming data such as log data, events (etc...) from various web serves to a centralized data store.

**Oozie** - Apache Oozie is **a Java Web application used to schedule Apache Hadoop jobs.**

**Yarn -** YARN is responsible for allocating system resources to the various applications running in a Hadoop cluster and scheduling tasks to be executed on different cluster nodes.

**HDFS(Hadoop distributed file system)**

****

If I have temp.txt file, and the file need to be dividend. The division in Hadoop cluster happen at the file level, not cluster. A user can decide how the data can be divided based on block size in hadoop.

**Block size- It is a parameter using which we are going to divide the whole file into chunks of files. Default block size 64MB(version 1), 128(version 2).**

**Min[1 MB] - Max[256 MB] block size.**

**Name node -** Maintain the metadata of each and every file to track and search them easy.

Metadata is very important

*Daemon - process/program that will always be running in the background.*

Hadoop Daemons - NameNode, DataNode, Secondary Namenode

**Secondary Namenode** - This node will take the backup of metadata of the NameNode(master node).

**Job Tracker/Resource manager** - This will take care of all the jobs. Initially, JobTracker will interact with NameNode and get the details about the data storage(in which all the data node has files). Then JobTracker interact with **Task Tracker** which is inside data node. JobTracker tell Task Tracker this is the logic, process this logic on the data files.

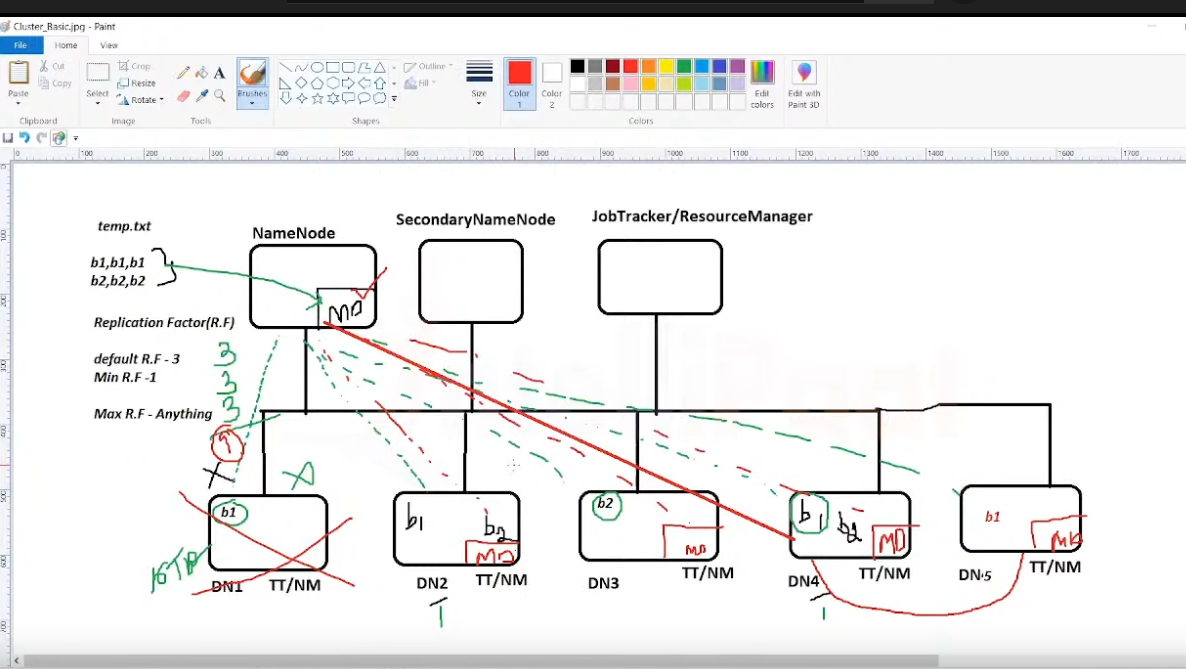
**Replication Factor -** Its a fault tolerance mechanism, The replication factor represents number of copies of a block(like 54 MB data of a file) that must be there in the cluster, default replication factor 3.

If the data node 1, data node 3 have same copy of blocks, which ever has more bandwidth will be located and picked.

**Heartbeat** - Every DataNode has to ping back to NameNode on every 3 second saying I am alive. If anyone one of the data node is dead, the data should be recovered. When any DataNode is dead, Namenode tell other DataNode(which ever have the higher bandwidth) to copy the data from dead DataNode to another DataNode.

**Block report** - Every DataNode will also have MetaData which is called BlockReport, and it is share with NameNode along with heartbeat.

The dead DataNode heals themselves after 9 seconds, this inbuilt hadoop feature.

****

**The above multi node cluster is in production level, In development level we use only single node.**

**Hadoop Commands**

To check the hadoop version - hadoop version

To start hadoop cluster/service →**start-dfs.sh**

To start the YARN service → **start-yarn.sh**

To check different services which are currently running **- jps**

To stop the cluster - **stop-dfs.sh**

To create new directory - hadoop fs -mkdir /path/directory\_name

To list the files and directories in HDFS - hadoop fs -ls

To copy file from local directory to hadoop directory - hadoop fs -copyFromLocal <local file> <hdfs location>

**PIG**

Pig is a scripting language used for exploring large data sets. **Pig Latin** is a hadoop extension that simplifies hadoop programming by giving a high level data processing language.

Components of PIG are

**Pig Latin** - A language used to express data flows.

**Pig Engine** - An engine on top of hadoop.

Pig Latin is data flow language,

provides support for long, float, chararray, schemas, and functions data types.

It operates on files in HDFS.

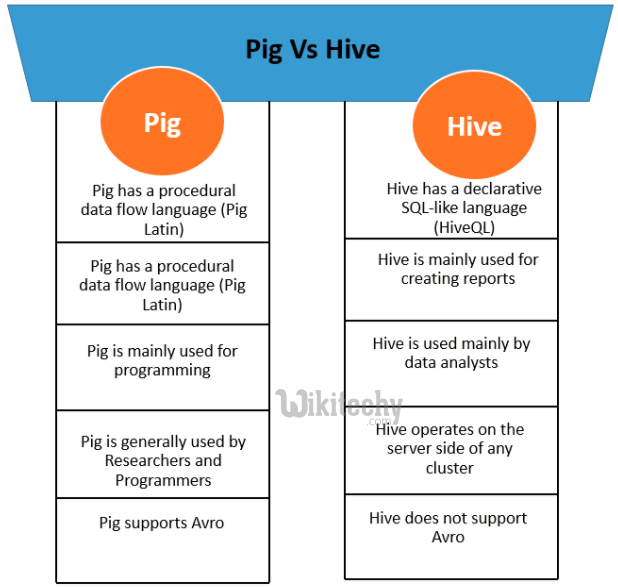
It also provides operations like JOIN, GROUP, FILTER, SORT.

Metadata is not required for pig.

Best suited for structured and semi-structured data.

Pig supports complex data type.

Usage: Web log processing, data processing.



----------------------------------------------------------------------------------------------------------------------------

What is client-side and server-side with example?

For example, developers can use JavaScript, a client-side programming language, to create forms that collect user input. Server-side development can process this input. For example, developers can use PHP to connect a database to a website and send user-inputted data to the database.

----------------------------------------------------------------------------------------------------------------------------

Pig can be run in Local(for testing purpose) and MapReduce mode,

In MapReduce mode, Pig will read the data from HDF file system, transform it and store it back to HDFS file system.

We have a file named NYSE\_dividends

**dividends = load ‘NYSE\_dividends’ as (exchange,symbol, dividend, colum4,colum5..);**

**grouped = group dividends by symbol;**

**avg = foreach grouped generate group, AVG(dividends.dividend);**

**Command to execute in pig -> dump avg;**

**Store avg into ‘xyz’; //store the file**

Or you can also create a file and paste same commands and run . Till now we did in local you can also do this in hadoop environment. **In pig to access columns, we use $ symbol for that particular column and it start from zero.**

Operators in PIG

LOAD - Default tab delimited inputs. Else have to specify.

Var = load ‘NYSE\_dividends’ using PigStorage(‘,’) as (colum1, column2);

STORE - Default stores the result as tab separated values. Also can specify(‘, ‘)

Store processes into ‘processed’ using PigStorage(‘ ,’ );

DUMP

**Relational operators in Pig**

FOREACH

FILTER

GROUP - the results will be key value pair.

ORDER BY

DISTINCT

JOIN

LIMIT

**HIVE**

Hive doesn’t have any storage. Hive is a client side application which is installed in our laptop, then which is connected to hadoop.

**Hive provides a SQL like interface and a (query language) HiveQL to query the data stored in databases and file systems that are integrated with hadoop.**

Hive uses map reduce and hdfs for data processing and storing.

Hive --> Jar file → Hadoop cluster.

Apache Hive is a data warehouse toll built on top of apache hadoop.

To create a file folder -- hadoop fs -mkdir folder\_name

To put file in to any folder -- hadoop fs -put file\_name folder\_name

**To create a table**

Create table fb(column1 int, column2 str, column3 int..)

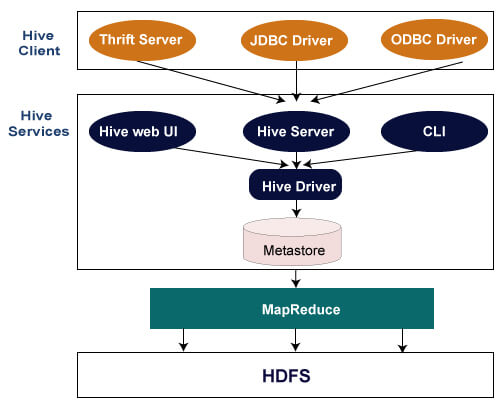
row format delimited fields terminated by ‘ , ‘

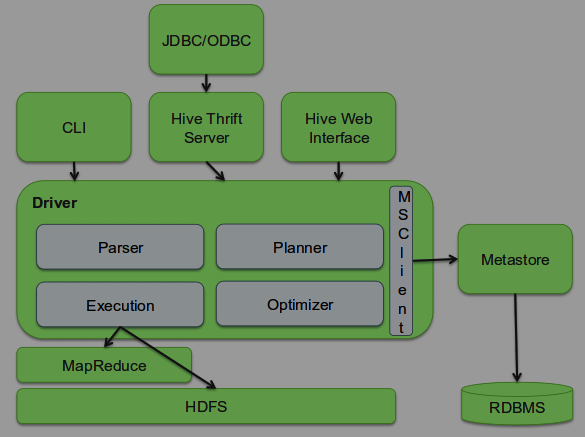
stored as textfile location ‘/user/path to file data/’ ;

And queries are very similar to sql. There are also HiveServer 1, HiveServer 2

Hive Architecture

Metastore is **the central repository of Apache Hive metadata**. It stores metadata for Hive tables (like their schema and location) and partitions in a relational database. It provides client access to this information by using metastore service API.





Command line interface

Create a table

>Create table customer(custno string, first\_name string, lastname)name string, age int)

> row format delimeited

> fields terminated by ‘ , ‘;

To load the data

>Load data inpath ‘/home/cloudera/Desktop/customer’ into table customer

>select count(\*) from transaction\_records;

**PARTITIONS IN HIVE**

Partition means we tell hive to divide that data based on **column.** Like if there are people from us and India based on country, the data ais partitioned.

Static partitioning - we have to do it.

CREATE TABLE table\_name(f\_name varchar(64), l\_name(64)

**PARTITIONED** by (country varchar(64), state varchar(64))

Row format delimited fields terminated by ‘ ,’

Dynamic partitioning - hive will do itself.

**YARN(yet another resource negotiator) Architecture**

In hadoop 2 YARN is only made responsible for **allocating resources**. It is scalable with other technologies.

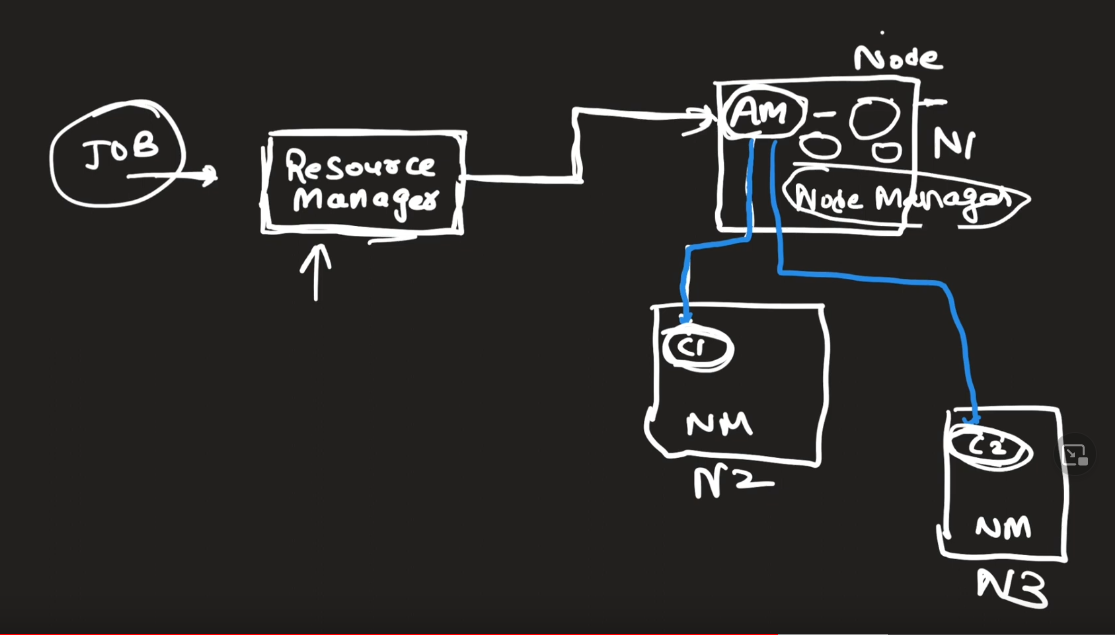
JOB → Resource manager → [Node 1, Node manager ]Application Master

[Node 2, Node manager]

[Node n]

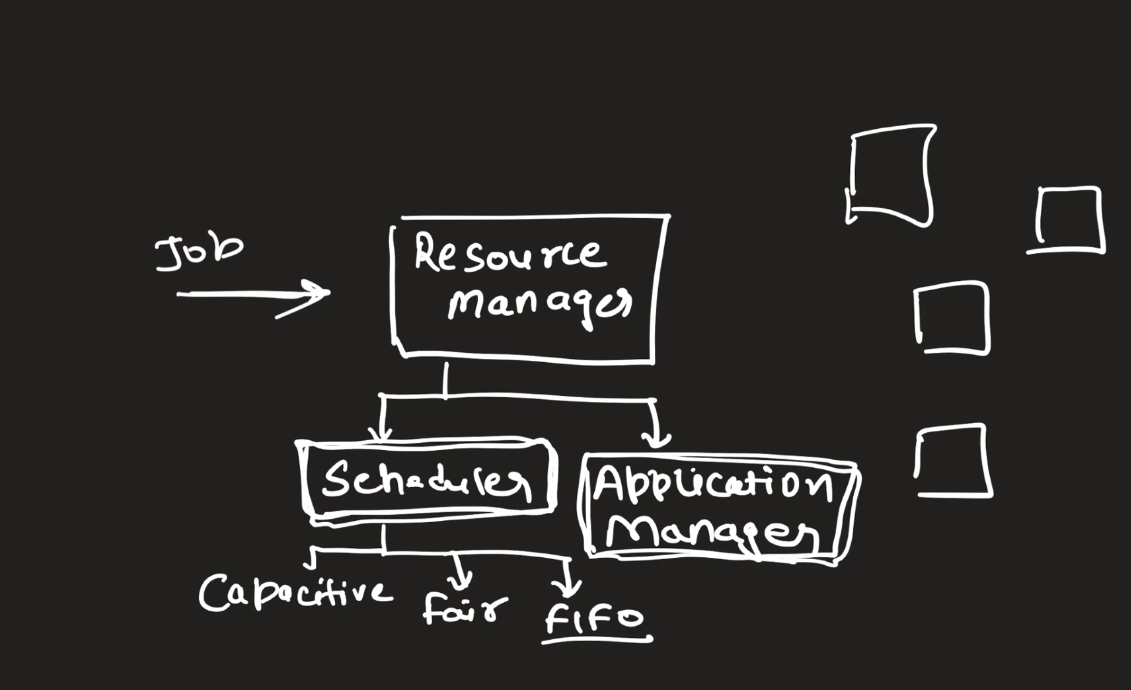
The resource manager is just doing resource management, Initially the resource manager communicates with Node Manger and negotiates for processing. After the resource negotiation, once the confirmation come back to resource manger saying resource are available. Then application manager is launched. Then application manager (of Node 1] manages the tasks.

So the **application master does the task management.**

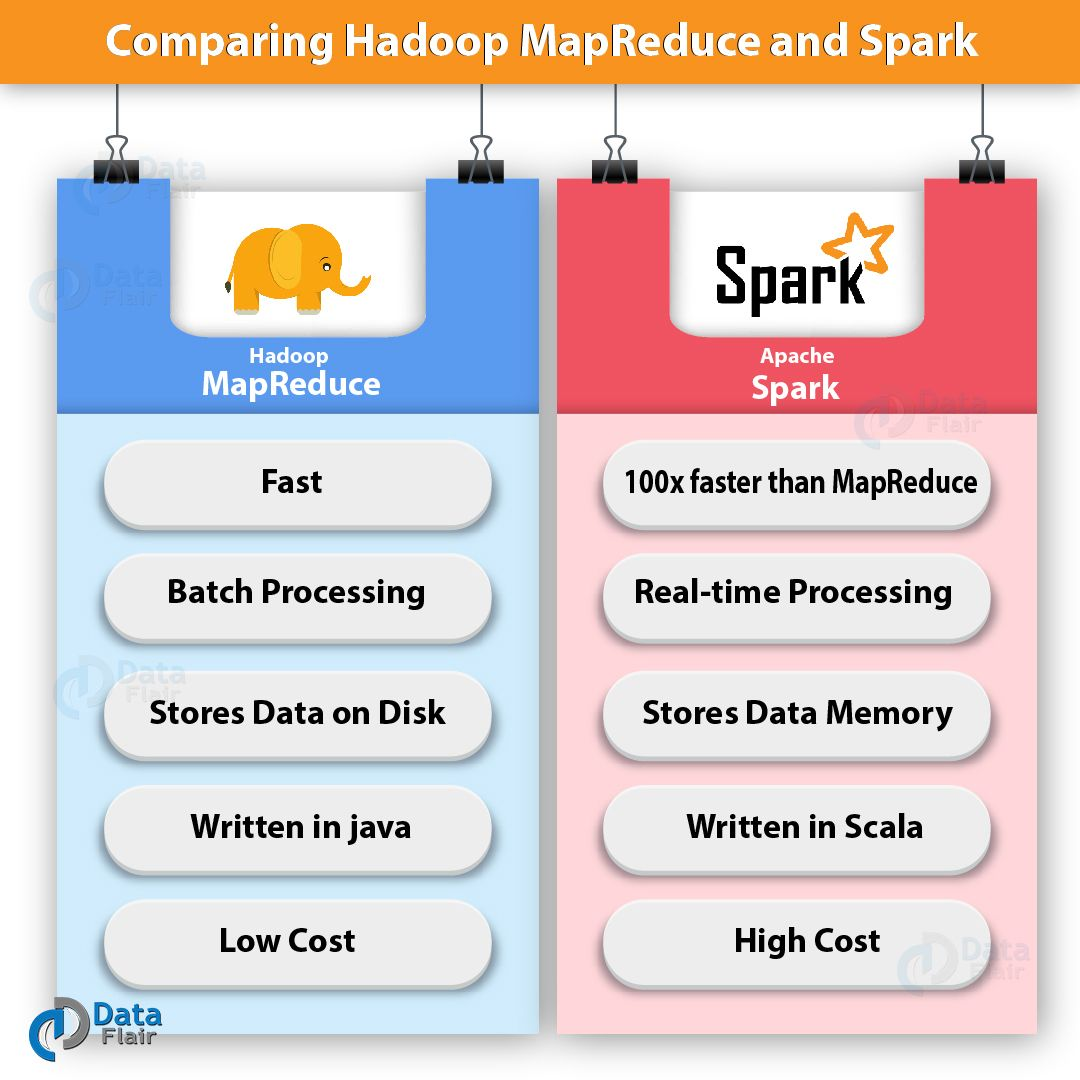


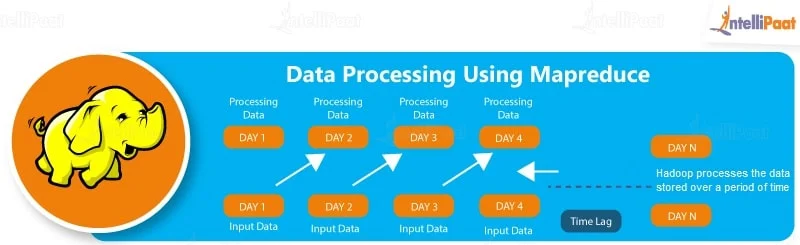
**Resource Manager** --Scheduler

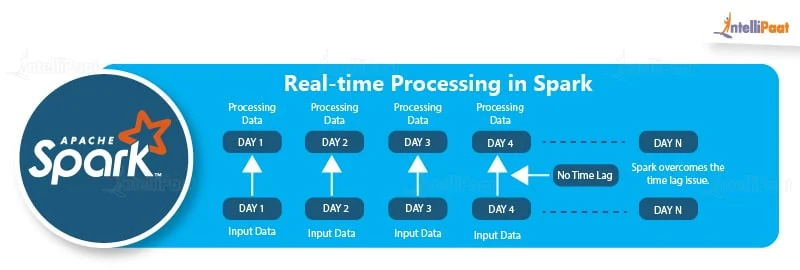
-- Application manager

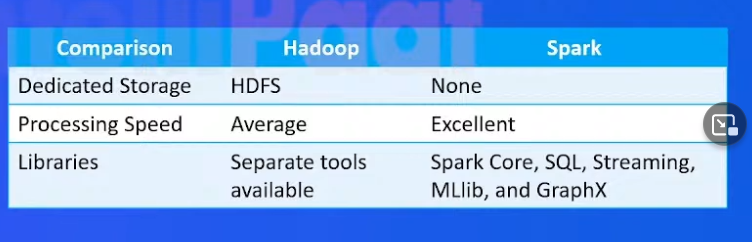


**Hadoop vs Spark**









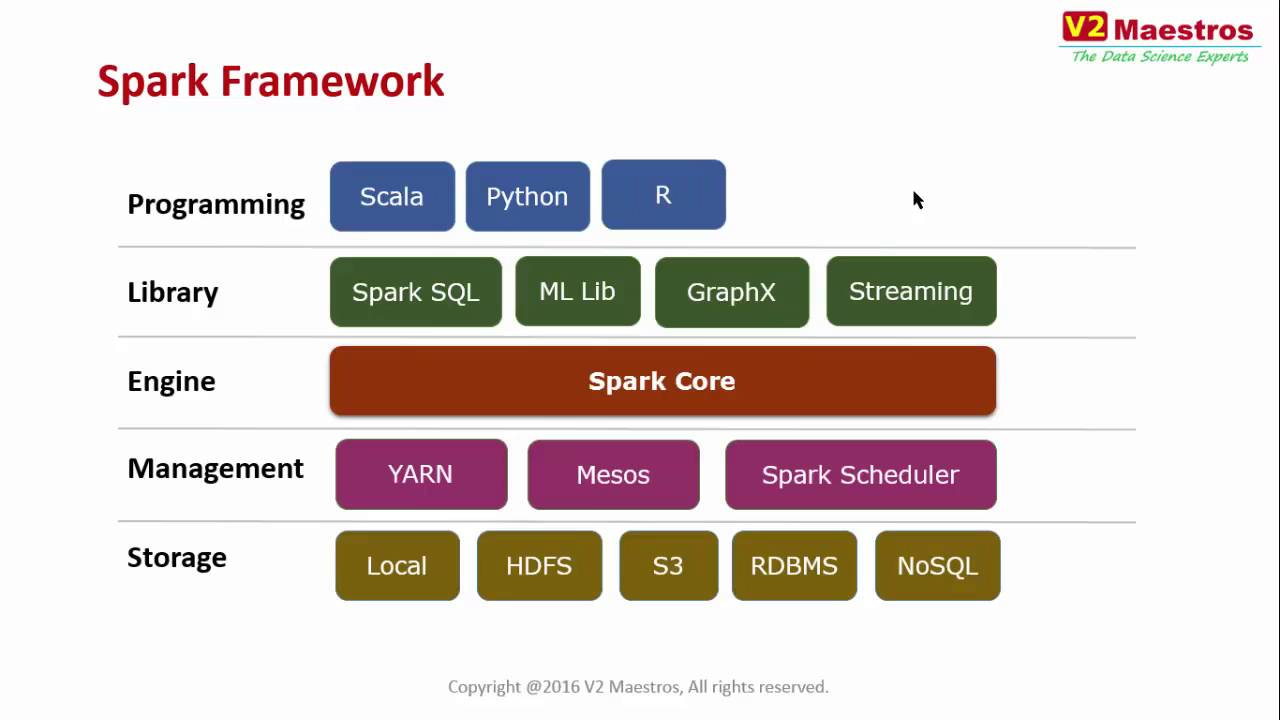
**Apache Spark**

Real time processing, low latency, Speed, Polyglot, powerful caching, deployment, Runs Everywhere(hadoop, Kubernetes)**. Spark comes with built-in libraries, with those you can do all the things that hadoop ecosystem tool does. Spark can do all the things alone compared to hadoop ecosystem. The issue with hadoop is after hadoop 1 war introduce there were lot of demand for different tolls(like Hive\_Sql, Mahout\_ML)**

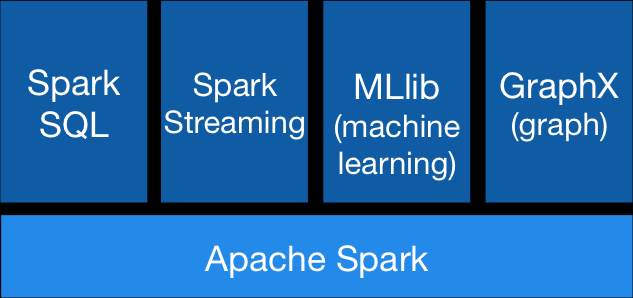
**Apache spark is like replacement for Map reduce. Spark is a processing engine. The best part of the spark is it can connect to most of the platform, Now people use spark as an ETL tool.**

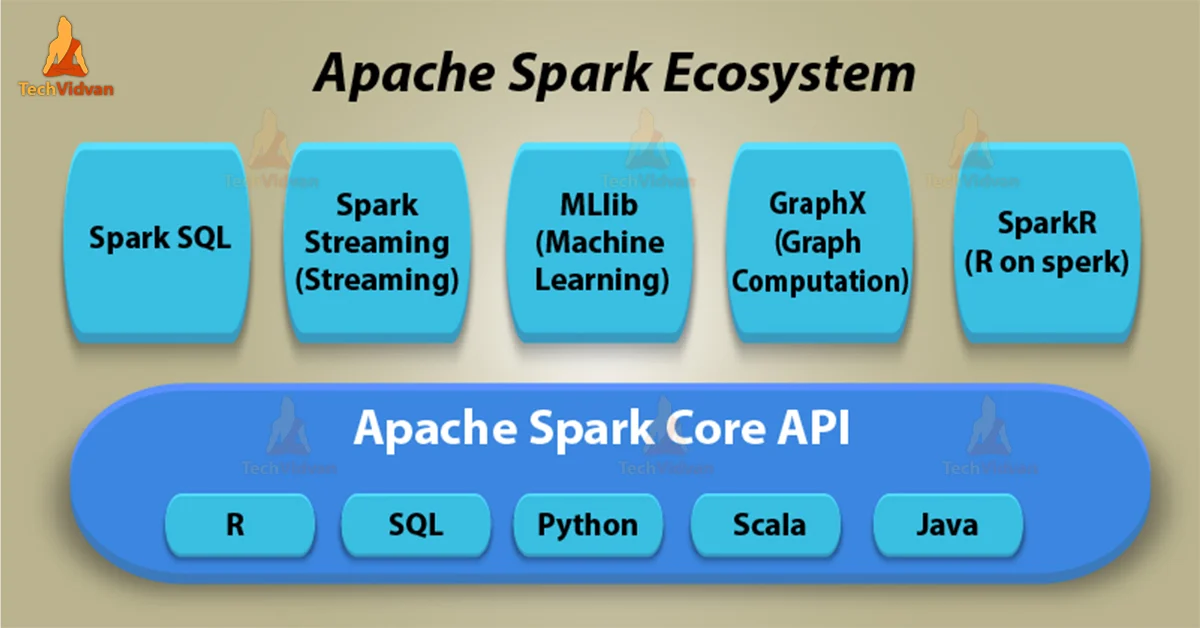
**Hdfs- primary storage, Yarn - cluster manager, spark - execution engine.**

Spark require STORAGE, Resource manager. There are different ways you can make this schema. Like (HDFS, YARN, SPARK ], or [Local, Mesos, SPARK] )

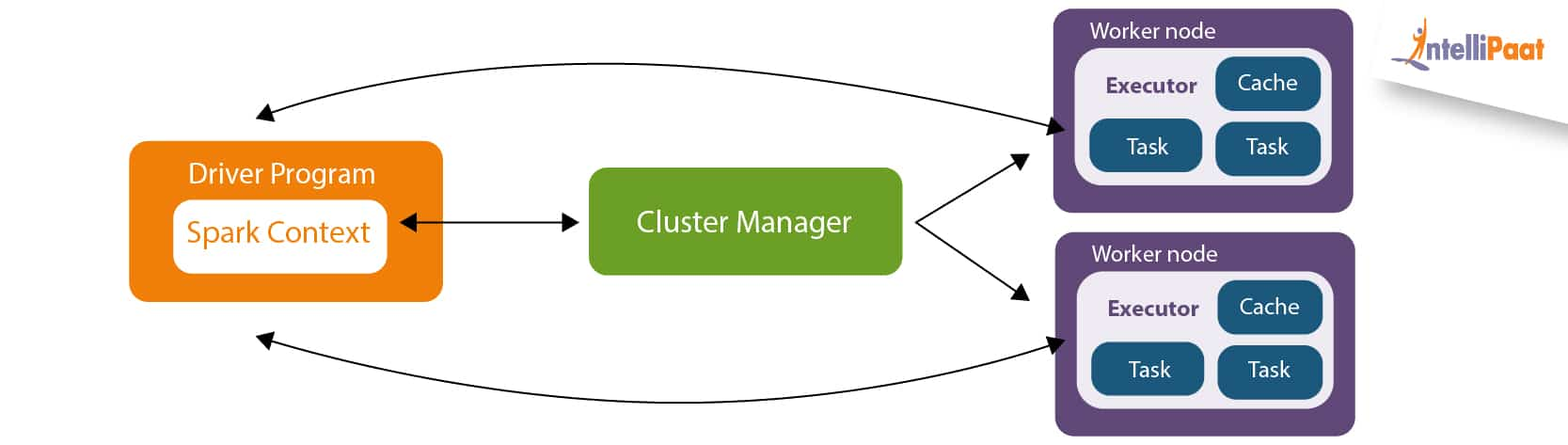


Spark Core and libraries/components





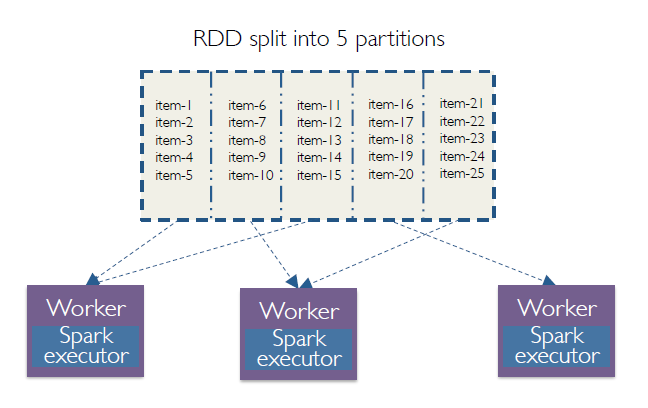
SPARK Architecture



**The driver program initialize the spark context, then driver program communicate with cluster manager(YARN), Then driver program negotiate the resource with the YARN(cluster manager).**

**YARN will launch the executors on Data\_Node. Then driver program push the business logic called task to the executors, and they execute parallelly. Once execution over the driver program get the results and executors are deleted.**

**PARTITION in SPARK**



**Practice**

In hadoop directory where you installed,

>>PySpark - will bring python interactive shell

>>spark-shell - will bring scala interactive shell

>>

=====================================================================

Scala was the first language which supported both object-oriented and functional programming(where you try to reduce umber of lines of coding). Scala was built like hybrid of Java(difficult but more and Python(easy, not optimized as java)

In Scala there are two types of data types,

1. Immutable Variable
2. Mutable Variable

=====================================================================