## * Hadoop :-

- Open Source software programming framework for storing a large amount of data & performing the computation.
- Its framework is based on Java programming with some native C code.
- It is used for some adv. level of analytics, which includes ML & Data mining.
- Most well known tech. used for Big Data.
- It is a large scale batch data processing system.

## * Need for Hadoop :-

- Distributed clusture system.
- Platform for massive scalable applications
- Enables parallel data processing.
- Programmers no need to worry about → where file is located, How to handle failure & data lose, How to divide computation, How to program for scaling.

## * Advantages :-

- Ability to store a large amount of data.
- High flexibility
- Cost effective
- High computation power
- Tasks are independent
- Linear scaling

* Disadvantages :-

- Not very effective for small data.
- Hard clusture mangement
- Has Stability issues
- Security concerns

* Hadoop Architecture :-

① Hadoop Common :- These are Java libraries & utilities required by other • hadoop modules. These libraries provides file system & OS level abstraction & contains the necessary Java files & scribts to start Hadoop.

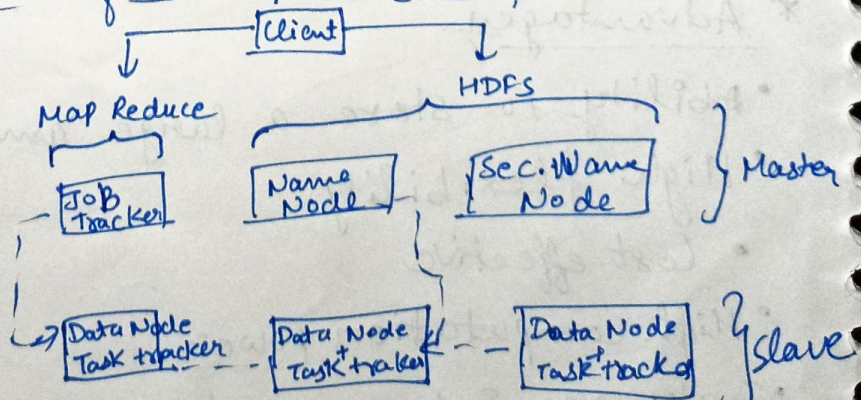
② Hadoop YARN :- This is a framework for Job scheduling & clusture resource managenet

③ HDFS :- A distributed file system that provides high-throughput access to application data.

④ Hadoop MapReduce :- This is YARN-based system for parallel processing of large data sets.

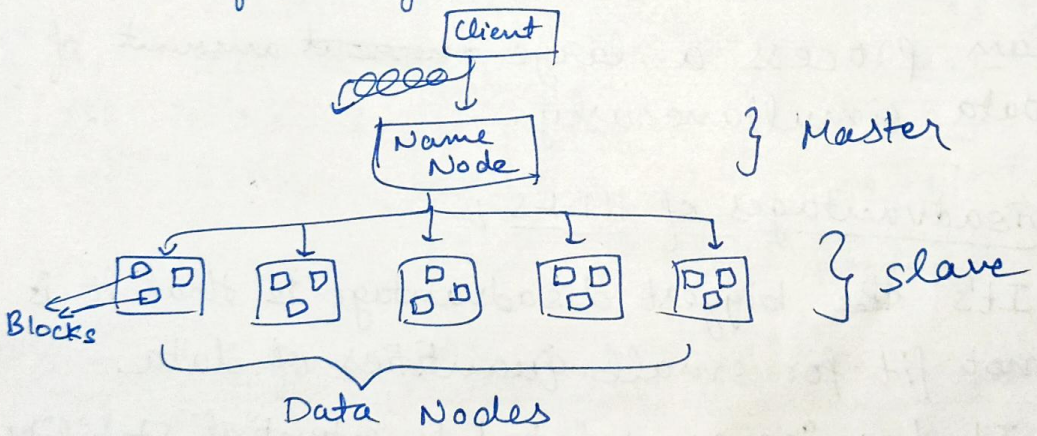
* Core components of Hadoop Clusture :-

① Client
② Master
③ Slave



## *HDFS :-

- Hadoop Distributed File System
- A scalable, fault tolerant, High performance distributed file system.
- Asynchronous replication
- Write-once & read-many
- 3 minimum Data Nodes. cluster
- Data divided into 64 MB or 128 MB blocks. each block replicated 3 times.
- These replicated blocks are stored in diff. Data nodes.
- HDFS consist of two core components → Name Node, Data Node.
- HDFS maintains all the coordination b/w the clusters & hardware, thus working at the heart of the system.




*Name Node :- It is the prime node which contains meta data like,
list of files, list of blocks in each file,
List of Data Nodes for each block, Creation time

*~~Data Node :-~~

* Data Node :-

- Stores the actual data.
- Block server stores data in the local file system.
- Periodically sends a report of all existing blocks to the name node.
- Periodic validation of checksum.
- It serves read, write request, performs block replication, deletion & creation upon instruction from Name node.

* Advantages of HDFS :-

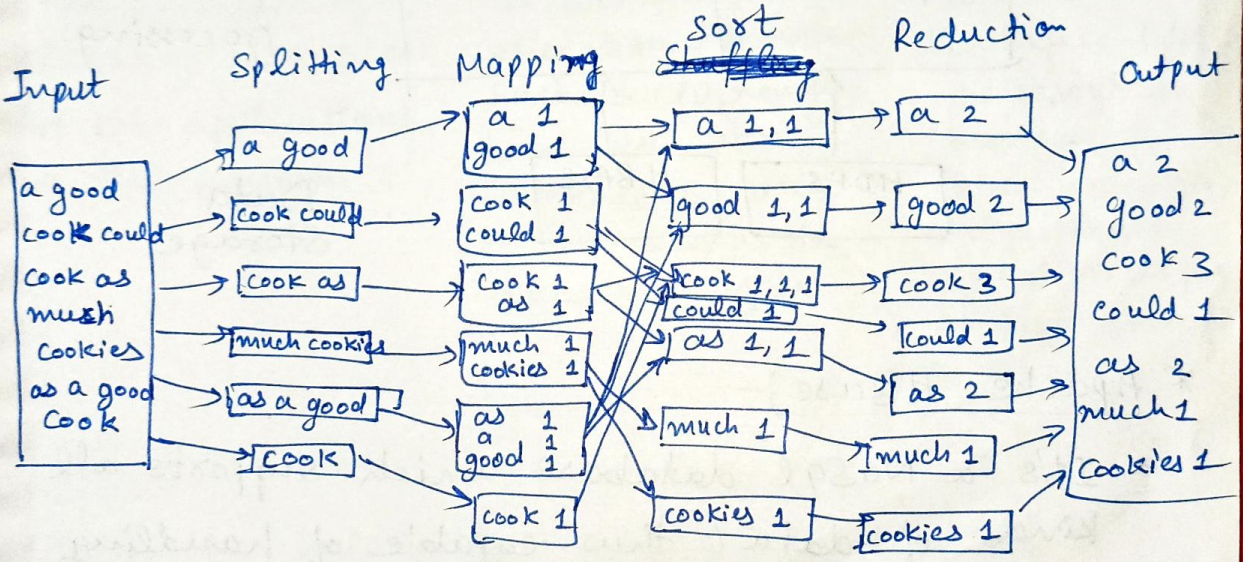
- It is inexpensive, immutable in nature, stores data reliably, ability to tolerate faults, Scalable, block structured.
- Can process a large amount of data simultaneously.

* Disadvantages of HDFS :-

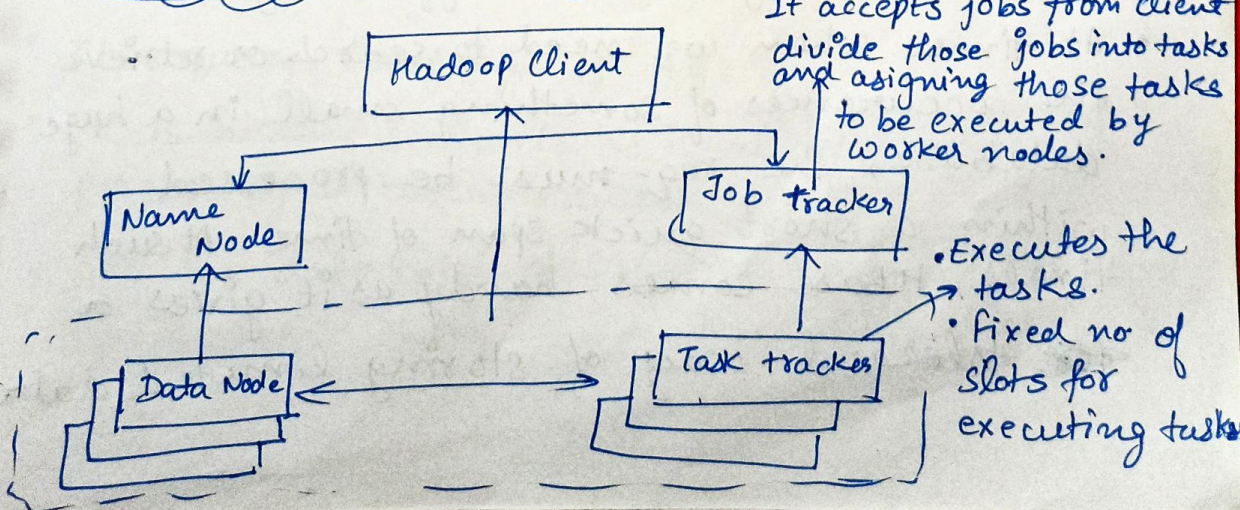
- It's biggest disadvantage is that it is not fit for small quantities of data.
- It has issues related to potential stability, restrictive & rough in nature.

# Map Reduce :-

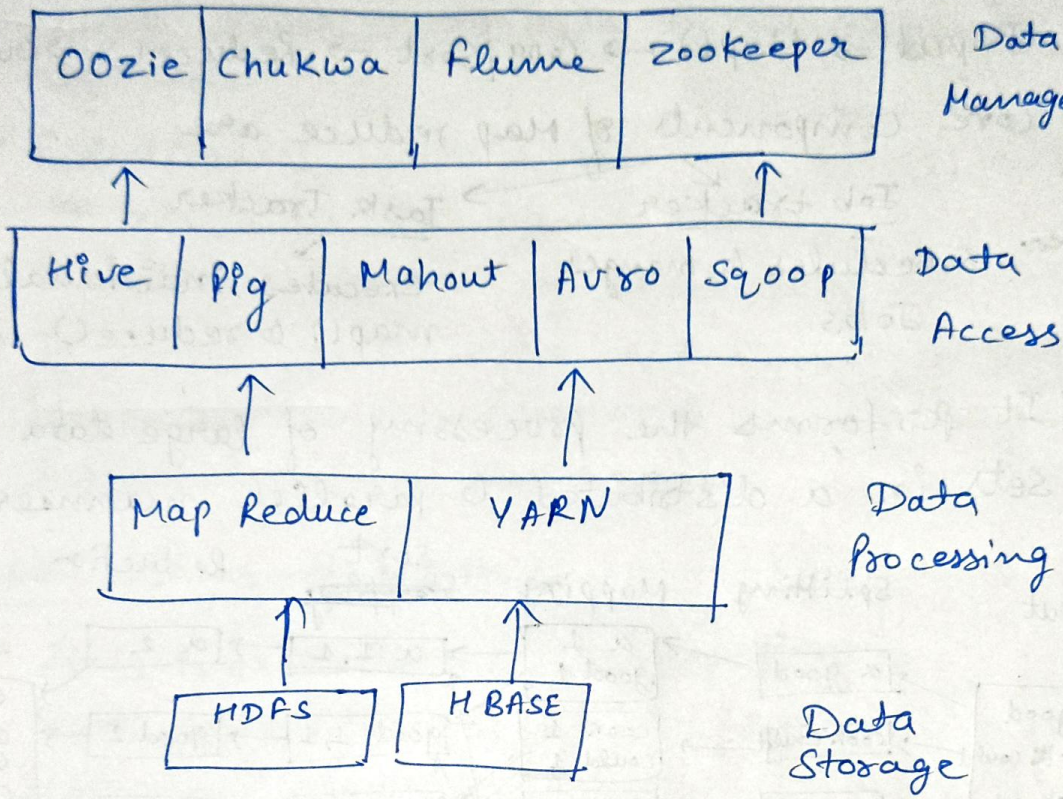
- Software framework for distributed computation
- Input → Map() → Copy/Sort → Reduce() → Output
- Core components of Map reduce are
  - Job tracker → schedules & manages Jobs
  - Task Tracker → executes individual map() & reduce()
- It performs the processing of large data sets in a distributed & parallel manner.




# Some Common Frameworks of Hadoop



# Hadoop Ecosystem :-




## * Apache HBase :-

- It's a NoSql database which supports all kinds of data & thus capable of handling anything of Hadoop Database & hence can work with Big data set effectively.
- At times when we need to search or retrieve the occurences of something small in a huge database, the req must be processed within a short quick span of time. At such times, HBase comes handy as it gives a ~~tole~~ tolerant way of storing limited data

## * YARN :-

- Yet another resource Negotiator,
- It helps to manage resources across the clusters. In short, it performs scheduling & resource allocation.
- Three major components

**Resource Manager**
↓
has privilege of allocating resources for the applications in the system.

**Node Manager**
↓
It works on the allocation of resources such as CPU, memory, bandwidth per machine & later on acknowledges the resource mangger.

**Application Manager.**
↓
Works as a interface b/w the resource manager & node manager and performs negotiation as per the requirements of two.

## * HIVE :-

- With the help of SQL methadology, interface, HIVE performs reading & writing of large data set.
- Its query language is HQL.
- It is highly scalable as it allows real-time processing & batch processing both.
- All the data types of SQL are supported by HIVE hence making the query processing easier.

* PIG :-

- Developed by Yahoo which works on pig latin language & it is similiar to SQL.
- Platform for structuring the data flow, processing & analyzing huge data set.
- Pig does the work of executing programs in the background, all the activities of MapReduce are taken care of. After processing, Pig Stores the result in HDFS.
- It runs on Pig Runtime, just like Java runs on JVM.
- It helps to achieve ease of programming & optimization & hence it is a major segment of Hadoop.

* Sqoop :-

- Sqoop is a tool designed to help users to ~~large data~~ import large data existing relational databases into their Hadoop clusters
- Automatic Data import
- SQL to Hadoop.
- Easy to import data.
- Generates code for use in MapReduce application
- Integration with HIVE.

## * Zookeeper :-

- A high-performance coordination service for distributed application
- There was a huge issue of management of coordination & synchronization among the resources or the components of Hadoop which resulted in inconsistency, hence zookeeper overcomes all the problems by performing Synchronization, inter-component based communication, grouping & maintainance.

## * Avro :-

- A data serialization system that provides dynamic integration with scripting lang.
- Avro Data :-
  - → Expressive
  - → Small and faster
  - → Dynamic (API permits reading & creating)
  - → Includes a file system & a textual encoding.

## * Oozie :-

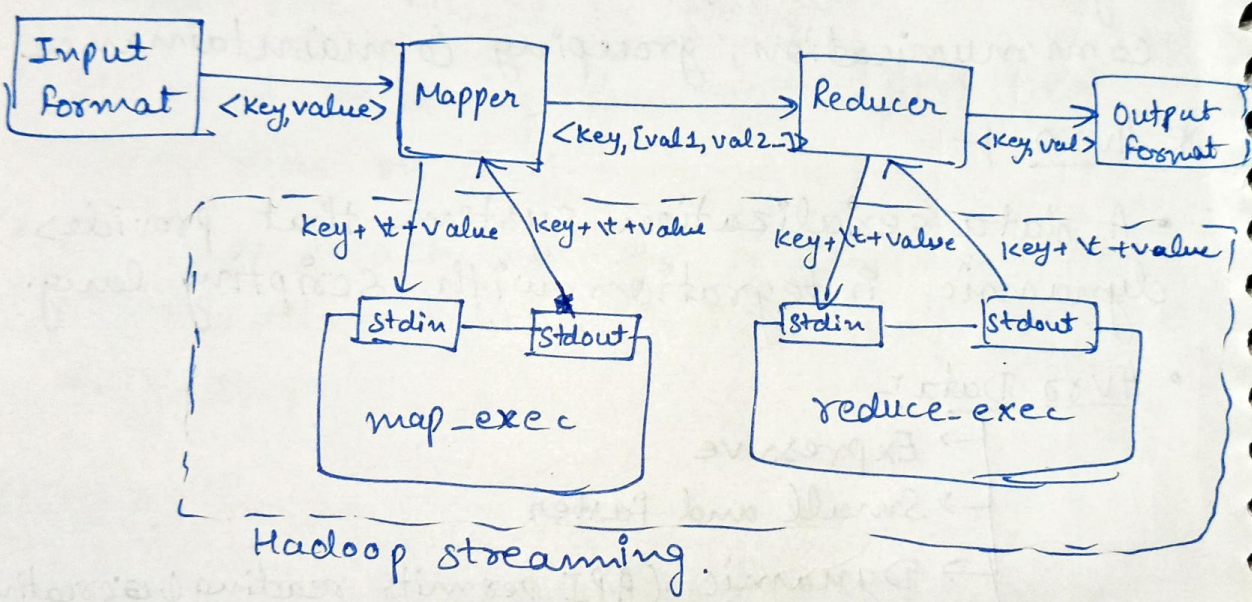
- It performs the task of scheduler, thus scheduling jobs & binding them together as a single unit.
- There are two kinds of Jobs i.e.

(i) Oozie work flow

(ii) Oozie ~~coordinator~~ Coordinator ~~Job~~.

- Oozie Workflow is the jobs that need to be executed in a sequential ordered manner
- Oozie Coordinator jobs are those that are triggered when some data or external stimulus is given to it.

## * Hadoop pipes :-




## * Hadoop Streaming :-

- Feature comes with hadoop distribution that allows developer to write map-reduce prog. in diff. languages like Python, C++, Ruby, etc.
- We can use any prog. lang. which can read from the standard input (STDIN) like keyboard & all and write using standard output (STDOUT)

- Main advantage of Hadoop streaming utilities is that it allows Java as well as non-Java programmed MapReduce jobs to be executed over hadoop clusters.
- It translates the application logic into the ~~map~~ mapper & reducer sections with the key & value output elements.
- Three main components :- Mapper, Reducer, Driver.

## * Apache Spark :-

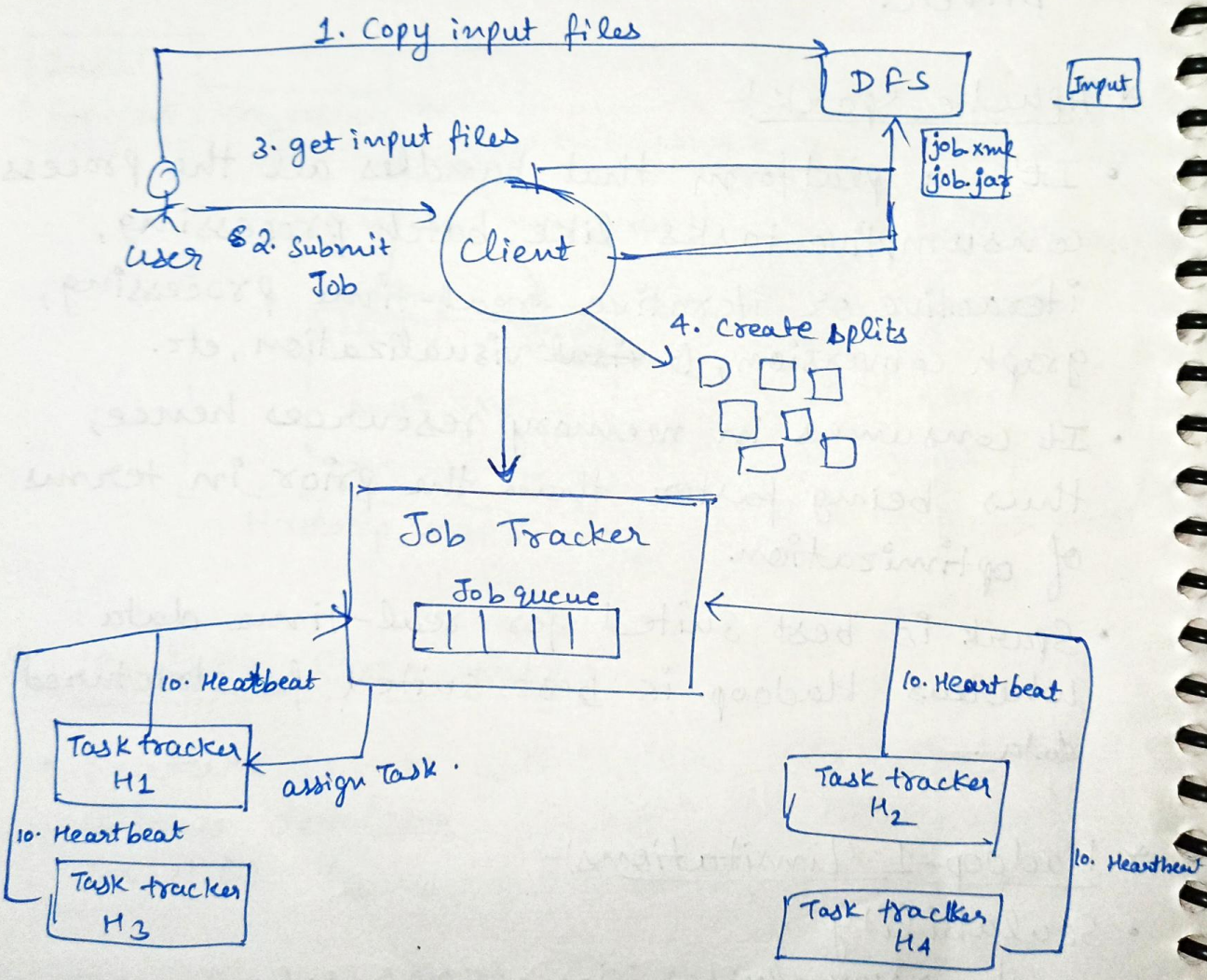
- It's a platform that handles all the process consumptive tasks like batch processing, iteractive or iterative real-time processing, graph conversion, & ~~visual~~ visualization, etc.
- It consumes in memory resources hence, thus being faster than the prior in terms of optimization.
- Spark is best suited for real-time data whereas Hadoop is best suited for structured data.

## * Hadoop-1 Limitations :-

- Scalability
  - → Max cluster size ~ 5000 nodes
  - → Max concurrent tasks ~ 40,000
- Availability :- Failure kills Queued & Running Jobs

- Hard partition of resources into map & reduce slots :-
  → Non-optimal Resource Utilization
- Lacks support for alternate paradigms & services
  → iterative application in map reduce ar 10x slower.

# Job tracker flow :-



# Developing a MapReduce Application :-

* Stage-1 :-

① Start by writing map & reduce function

② Then write a driver prog. to run a job, which used to run small subset of the data on your prog. to check

③ If it fails, use IDE debugger to find the source of problem/error.

④ with this, we can expand our unit test and test it again

* Stage-2 :-

① When the prog. runs expected we can run on full dataset, it may expose some more issues, which we can fix again by above steps.

② Debugging failing programs in the cluster is a challenge, so we look at some common techniques to make it easier.

* Stage-3 :-

① After the program is working, we can do some tuning like making map reduce function faster

② Profiling distribution programs is not easy, but Hadoop has hooks to aid the process.