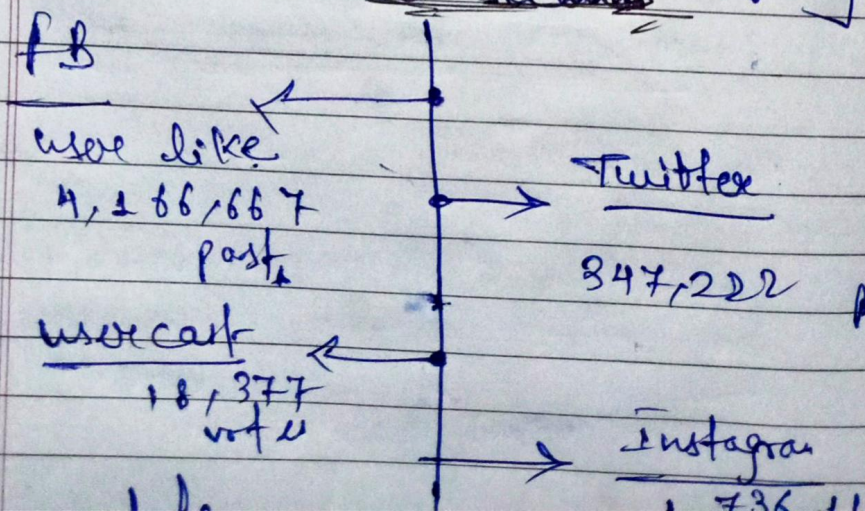
APACHE BIG DATA HADOOP

AGENDA →

- → Big Data Growth Drivers.
- → what is Big Data?
- → Hadoop Introduction
- → Hadoop Master/Slave Architecture
- → Hadoop Core components
- → HDFS Data Blocks
- → HDFS Read/Write Mechanism
- → What is MapReduce
- → MapReduce Program
- → MapReduce job Workflow
- → Hadoop Ecosystem
- → Hadoop Use Case: Analyzing Olympic Dataset.

→ BIG DATA GROWTH DRIVERS

[Data generated every 60 seconds]



Circa data

1 EB = 10^{18} bytes

2015 → 3.7 EB
2020 → 30.6 EB

Raising exponentially.

→ What is 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 management tools or traditional data processing applications".

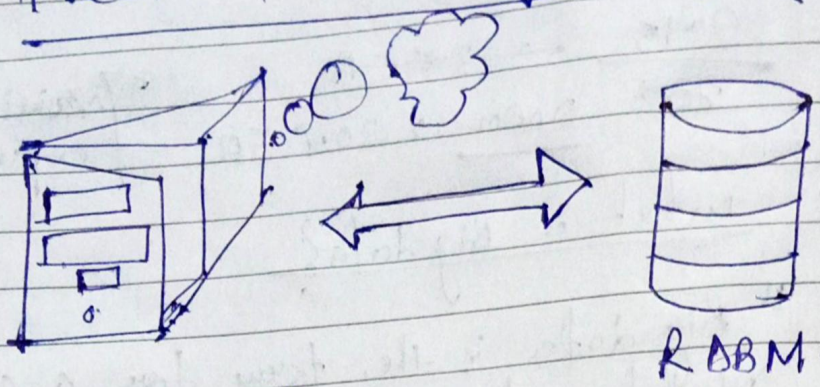
→ Big data is a problem because traditional systems are not able to store Big Data and process it.

So, In order to identify Big data problem, IBM has suggested 5 V's

There are:→

- Volume : (increasing huge data sets)
- Variety (Processing different type of data)
- Velocity (Data generated at alarming rate)
- Value (finding correct meaning out of the data)
- Voracity. (uncertainty and inconsistencies in data)

Traditional processing system



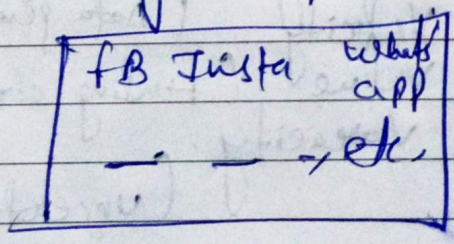
Traditional processing system.

Traditional Scenario:
Data is generated at a steady rate and is structured in nature.

→ very easy for traditional system to process it.

Now, in other scenario:
i.e.
↓
Big Data Scenario:

Heterogenous data is being generated at an alarming rate by multiple sources.



So, our traditional processing system are not capable of solving the problems of Big data i.e 5V's.

So, Solution?

Let's discuss case of Bob.

→ offline case (1 cheff, 1 waiter, 1 owner (Bob))

2 orders per hour. → can serve by single cheff.

Bob is happy.

Traditional processing system.

→ Online case

currently (1 cheff, 1 waiter).

→ 10 orders per hour → can't serve by single waiter.

Bob not happy.

This fails.

→ so → (Bob should hire more cheffs).

Now, we think that work will gets done now. But, no because, the

items to cook food are only 1, there could be a new chefs and bob doesn't wants them.

Bob still unhappy.

→ Last case

(multiple cook, multiple items to cook food).

Bob happy → Here, work is done successfully.

Conclusion

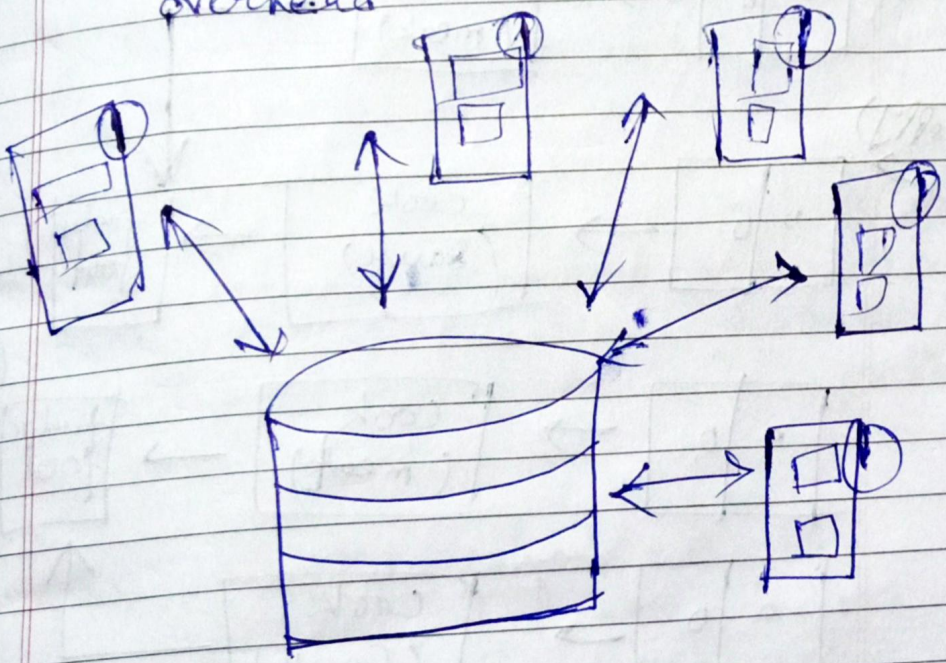
→ Multiple chef
→ Multiple items to cook food.

Scenario →

Multiple processing unit for data processing

Issue →

Bringing data to processing generated lots of network overhead.



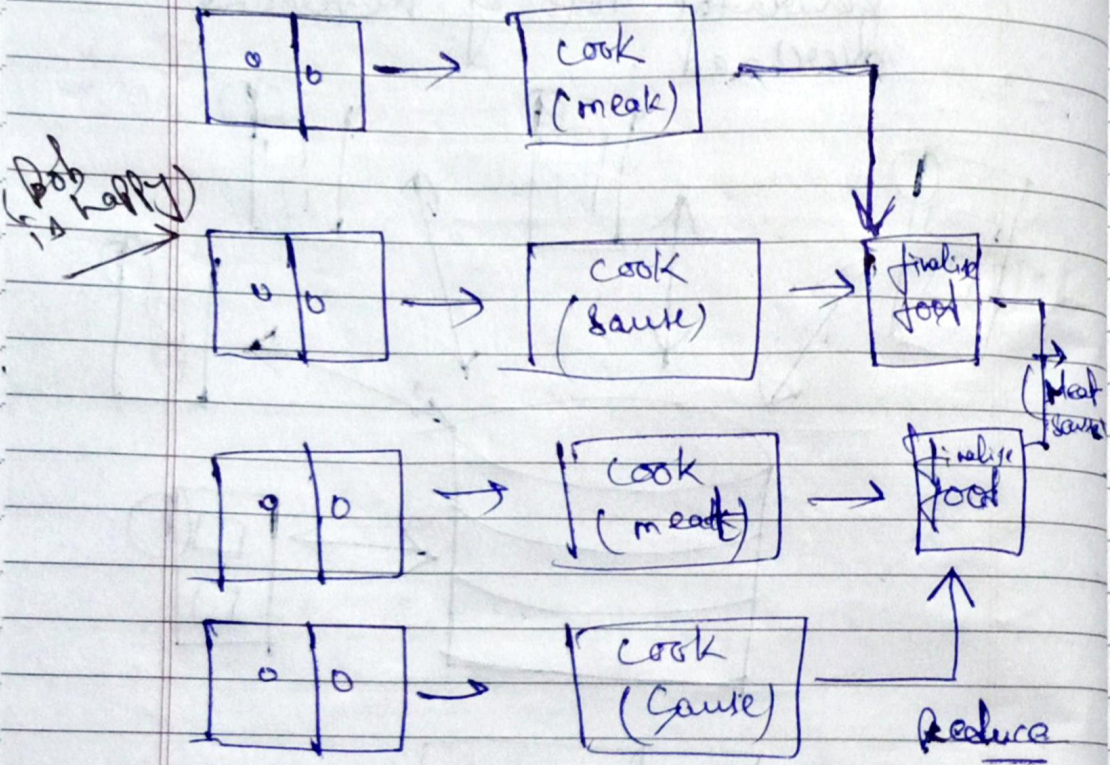
So, this solution also failed.

Now, what?

Effective Solution:→

" divide tasks to each chef then give it to head chef to finalise the tasks. "

Note:→ keep backups also.



Note:→ each cook can go to any shelf, if damage occurs.

Here,

In Hadoop terms

→ Each of the chef has got it's own shelf.

↓

in hadoop (data locality)

↓

means data is locally available into processing unit.

Note:→ This system is scalable for Bob, in case of heavy orders also, he can hire, more chef headchefs to take the problem.

Now, Bob has solved all the problems, but now, do we have the frameworks like that to solve all the problems like Bob (:)

↓

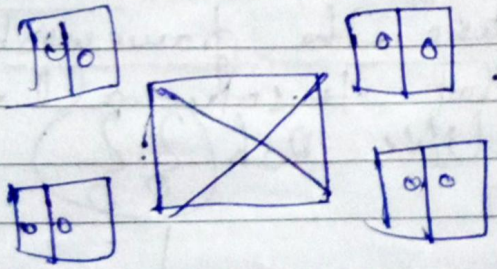
Yes, we have something called Apache Hadoop (framework to process BIG DATA).

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

Two major problem of Big data are:→
1) Storage
2) Processing.

Q. In order to solve storage problem of Big Data, we have got HDFS (Hadoop distributed file system).

↓
just like bob solves it's problem by distributing among friends.



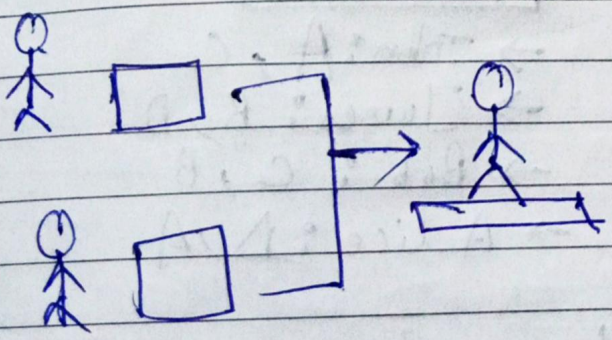
Storage: Distributed file system

(Hadoop runs → it is called hadoop cluster).

So, now, all the Big amount of data that we are dumping gets distributed over different machines and this machines are interconnected on ~~in the system~~ which our data is getting distributed, and in ~~the~~ hadoop terms, it is called Hadoop cluster.

Now, In order to solve processing problem of BigData, we have got MapReduce.

Just like, Bob has managed to divide a task among his chefs and meet the requirement quite quicker. Similarly, in order to process data,

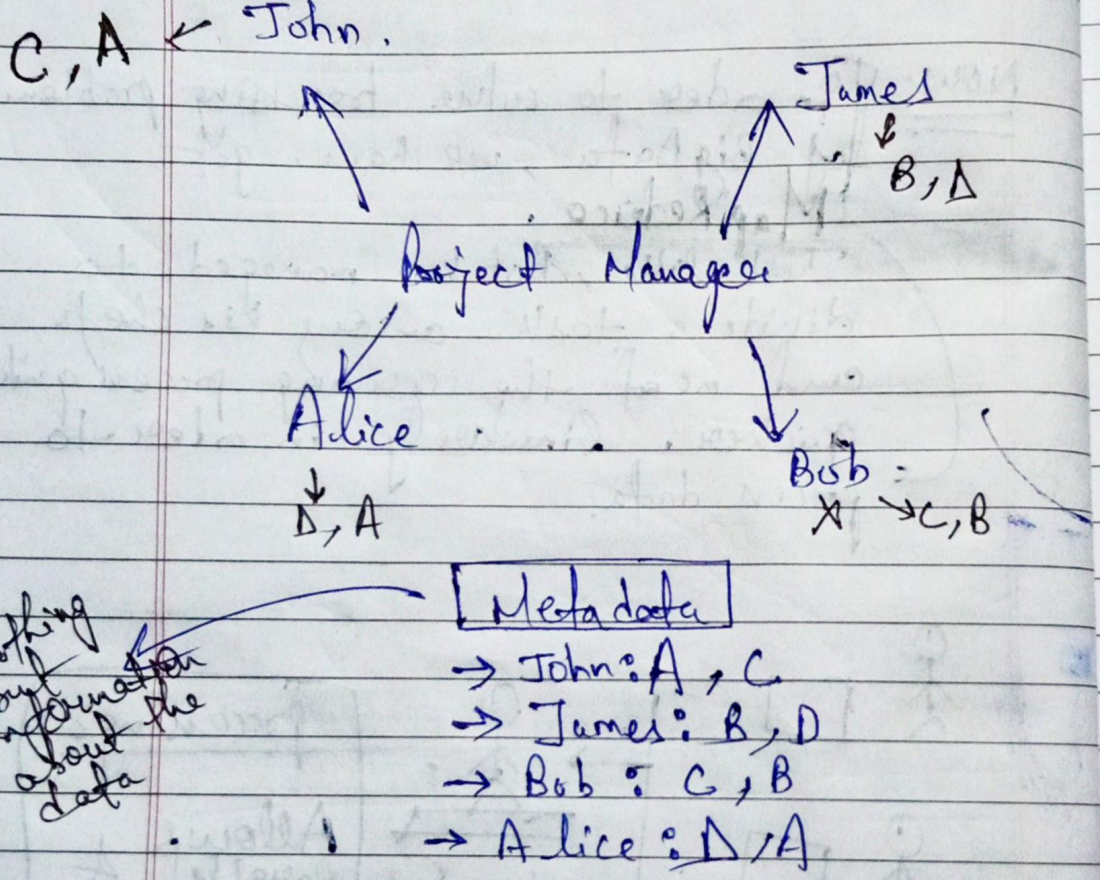


Processing:

Allows parallel & distributed processing

Hadoop: Master/Slave Architecture

Scenario:
A project Manager managing a team of four employees. He assigns project to each of them and tracks the process.



Project manager now got the backup.

So, In HADOOP, what happens is that

(A, C)

(B, D)

Slave Node

Slave Node

Master Node

Slave Node

Slave Node

(D, A)

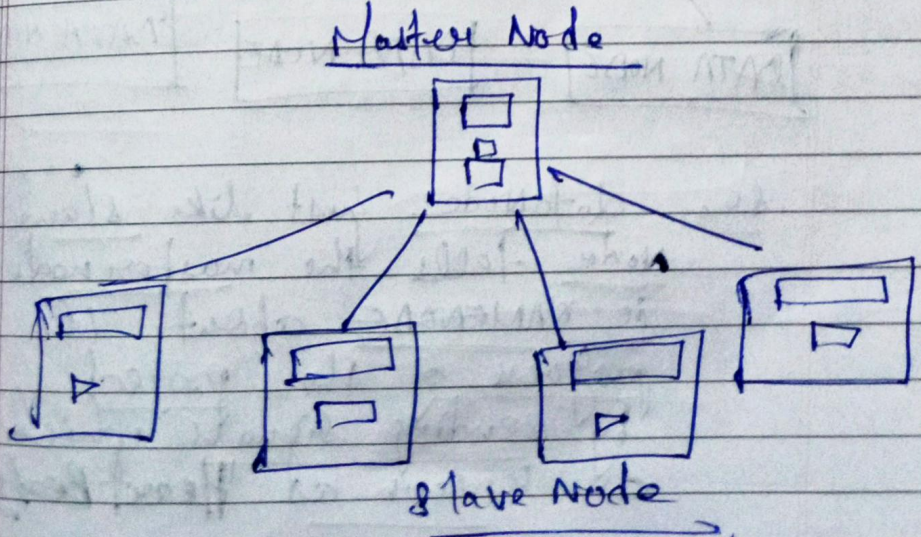
(C, B)

→ Master Node keeps a track report of all the processing going on the in the slave Node!

and, in case of disaster, i.e., if any of the slave Nodes goes down, the master Node has always got a backup.

→ (compare this structure with hadoop cluster).

↓ it looks like



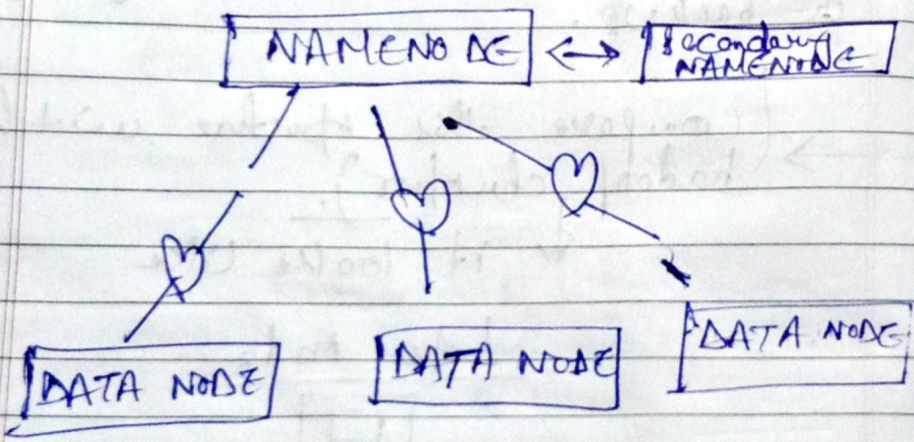
*) HDFS → (distributed file system in Hadoop).

(Core component) → NameNode
→ DataNode
→ Secondary Node

So, here,

MasterNode → NameNode

Slave Node → DataNode.



So, dataNode just like slave Node tells the master node, ie. NAMENODE about the progress of the project by sending signals which are known as Heart Beats.

Master daemon → NAME NODE

Now, this is just a signal to tell the name node that the data node is alive and working fine.

dataNode: → This is where, your actual data is stored.

dataNode → responsible → for ~~all~~ managing → all data → across → dataBlocks.

Slave daemons

Secondary NAMENODE: →

Metadata is maintained by HDFS using two file :→

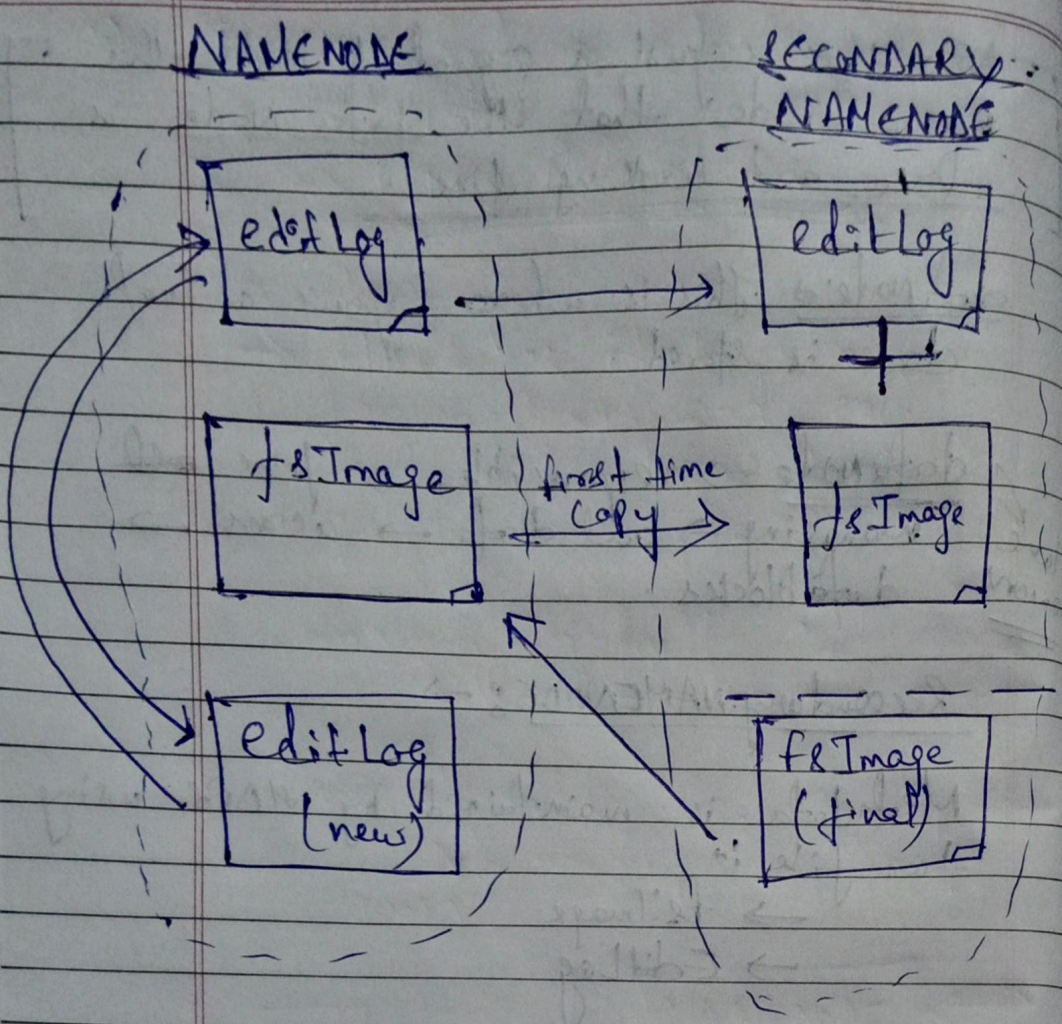
→ fsImage
→ EditLog

- fsImage → This contains all the modification that have been made across your hadoop cluster ever since the NAMENODE was started.

Store fsImage on disk

- EditLog → It only contains most recent changes about metadata.

(This file resides in the root of namenode machine)



⇒ We have this secondary namenode here, which performs the task known as checkpointing.

Checkpoint :-

It is the process of combining editLog with the fsImage.

Secondary Node here

↓

have copy of

↓

EditLog and fsImage from the NAMENODE

and, then,

if add's them.

↓

fsImage + EditLog

↓

in order to get New fsImage.

Q. → why do we need new fsImage ?!

By default, checkpointing happens every hour (1hr)

~~After~~, by the time checkpoing happening -

(end of cycle) new things get into new editlog, until the next checkpoints happen, which which again contain all the recent changes since the last checkpoint.

How does data is actually stored in DataNodes?

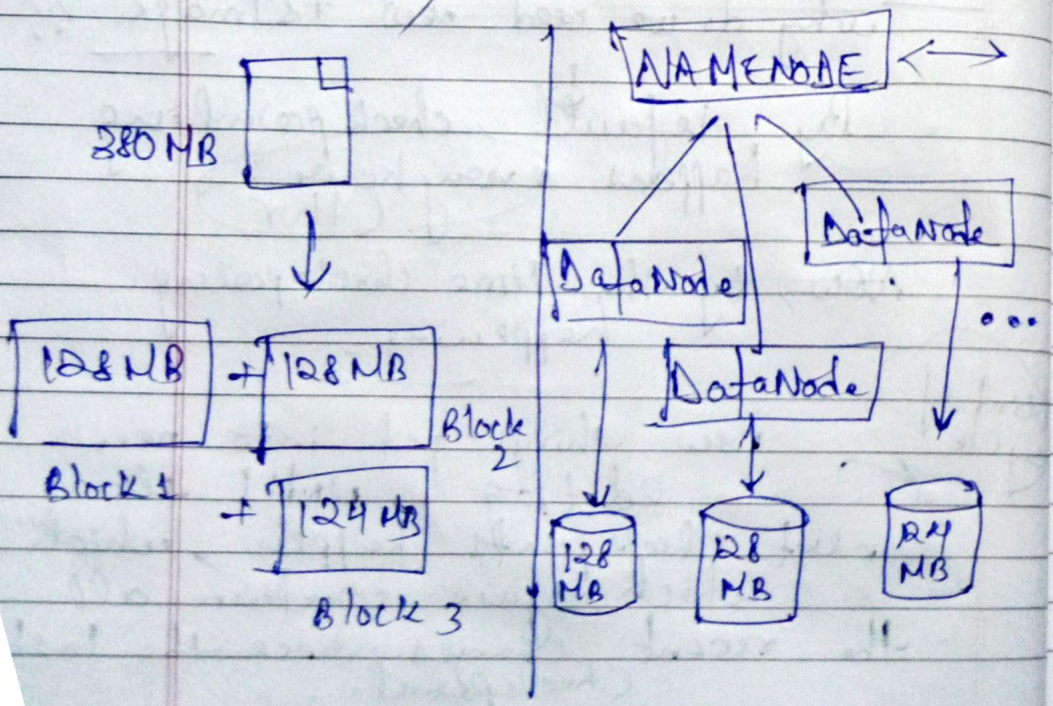
→ HDFS Data Blocks.

~~→ Block of~~

→ HDFS is Block structured file system. and each file is divided into a block of particular size.

and, by default that size is 128 MB.

(128 MB in Apache Hadoop 2x
(64 MB in Apache Hadoop 1x))



at last, datanodes we are saving 4MB, so, for huge data, we can save more storage with this idea.

Advantages of using distributed file system:- (HDFS)

- Suppose we have to store 3TB data and we are having datanodes of 1TB only.

  So, what we can do, we can store 1TB, in each of them. So, total we require 3-datanodes to store 3TB data.

and, if we see, this provides me with the abstraction of a single computer ie. is having capacity of 3TB.

So, that's the power of HDFS.

→ We can add new datanodes in order to store more data in the cluster. eg. 5TB instead of 3TB. So, requires 5 datanodes.

So, highly scalable.

→ Time management. (1 datanode → reading time 4 sec)

↓

parallel reading by distributing your file with the same single computer with the same capacity, same processing power.

Instead of 1, if we got 4 datanodes in cluster. it will take 1/4 times of initial.

1/4 of 4 = 1 sec only.