⇒ BPEL is stateful & needs to preserve the state of variables which are stored in variables.
When we declare a variable, we must specify the variable name & type.

⇒ Create Instance specifies whether an instance of the BPEL process should be created or not when <Receive> is encountered.

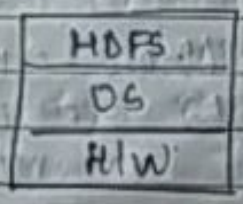
⇒ <assign name = "Assign 1"> || when multiple variables are passed.
<copy>
<from> —— </from>
<to> —— </to>
</copy>
<copy>
<from> —— </from>
<to> —— </to>
</copy>
</assign>

27/8/19 Hadoop
- HDFS (Hadoop Distributed File System)
- Map-Reduce framework.

Characteristics: (Of BigData)
i) Volume (in terms of Petabytes)
ii) Velocity - Rate at which data gets generated.
iii) Variety - The types of data (text, pictures, video etc.)

HDFS is useful for storage of the data & Map-Reduce framework assists preprocessing of the data.

Install HDFS:



File
- Data (stored in blocks)
- Metadata (i-node in case of Linux) (stored in blocks)

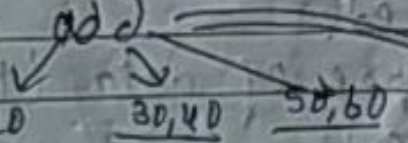
Default block size (in Windows): 4 KB

If it is 840 MB then, even to store a text file containing 'Hello World' will take up 840 MB of block (too much wastage of block size).

On the other hand, meta data info is less when blocks are of 840 MB

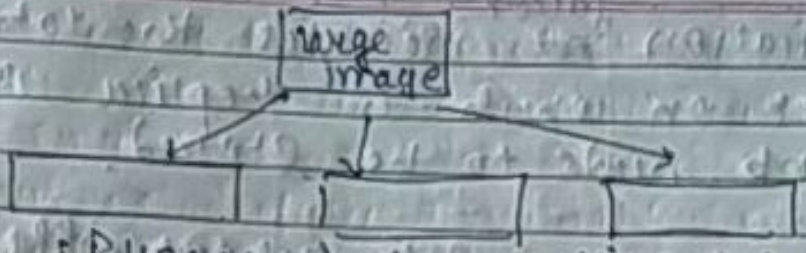
Thus, to store petabytes of info, it is better to have larger block size. For this reason, HDFS lies on top of normal OS.

SIMD (Single ins. multiple Data).



Single ins. is used on multiple data.

HDFS works in a similar way.



(Processed by multiple replicas on different machines)

11/9/19 Hadoop → HDFS (Storage of data)
→ MapReduce (Processing of data)

Big Data - Collection of data sets so large & complex that it becomes difficult to process using on-hand database management tools.

⇒ Key enablers for growth of Big Data (BD):
- increase in storage capacities
- processing power
- availability of data

⇒ How to handle (BD):
- Concept of torrents
  Reduce time to read by reading it from multiple sources simultaneously

Questions which arise:
(i) How to handle system ups & downs?
(ii) How to combine the data from all systems?

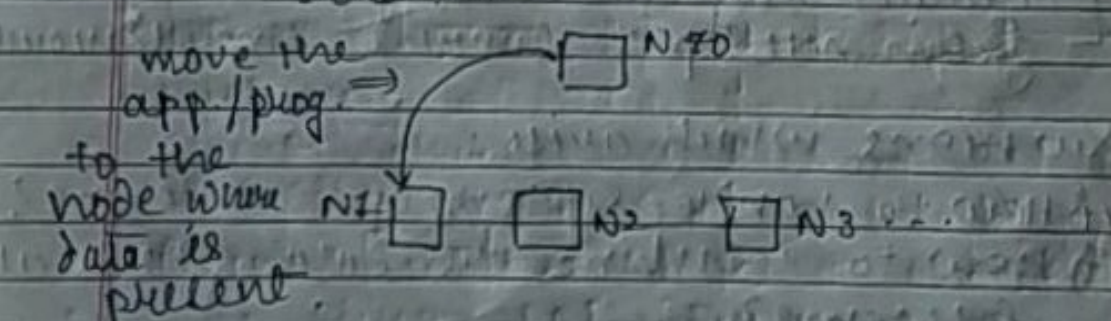
(i): • commodity H/w (Desktop (HDD - cheap H/w))
• Chances of failure high
• Hence, maintain multiple replicas
Ex - Google File System (GFS)

⇒ Replication factor indicates the total no. of nodes on which a replica of a chunk needs to be stored.

(iii): • Analyze data across different machines.
• Data has to travel across network. Only then merging is possible
• To achieve this, MapReduce is used.

⇒ Hadoop is an Open-source S/W framework that supports data-intensive distributed appl^n

Goals:
a) Facilitate storage & processing of large & rapidly growing datasets
b) Highly scalable (in terms of no. of nodes) and available
c) use commodity H/W with redundancy
d) Move computation rather than data



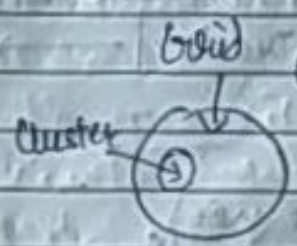
When to use Hadoop?
→ Processing of unstructured data

→ Processing can easily be made parallel
→ Running batch jobs possible
→ Lots of commodity H/w available.

Not a good choice if
→ Calculations with little or no data
→ Processing cannot be made parallel. (Sequential processing req.)
→ Interactive system is to be built.

Core Hadoop Concepts
a) Appln are written in high-level code
- N/w prog. ~~not~~ is not a concern
b) Nodes talk to each other as little as possible
- 'Shared nothing' archi.
c) Data is spread among m/c in advance
- computation occurs on m/c where data is already present.

Hadoop Components
- Main Compo.
  → HDFS
  → MapReduce
- Hadoop Eco System
  - Proj. developed around core Hadoop
  - Ex - Hive, Flume
- Hadoop Cluster
  - min. 1 node and max. as many as thousand nodes.



Cluster: 10-15 m/c (All same configure)
Massive shared archi.
LAN env.

Cloud: concept of virtualization (Virtual Box)

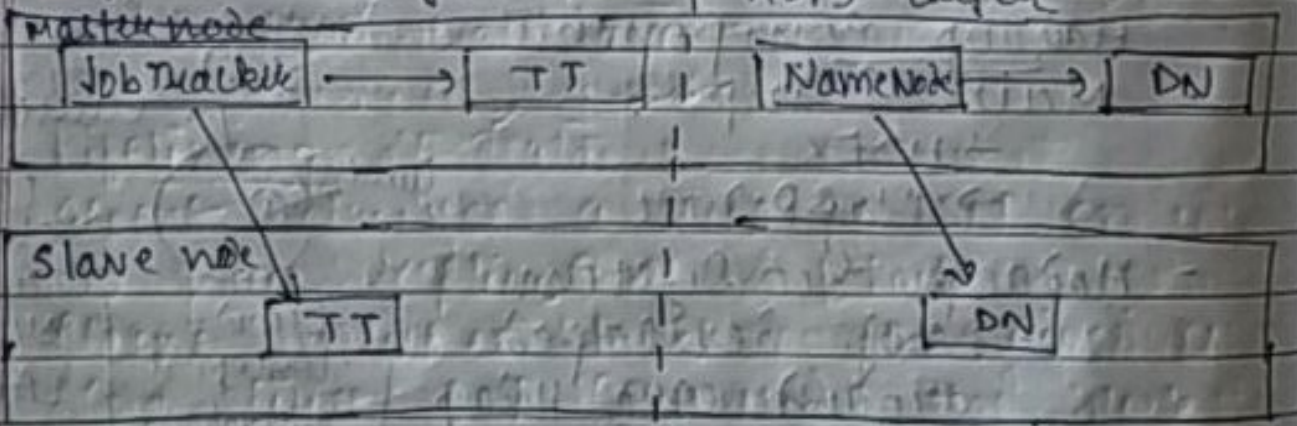
Grid: Heterogenous archi. Master-slave archi. LAN/WAN env.

	Orchestration Framework: Hbase, Flume
	Data Access Framework: Pig, Hive
	MapReduce (Data Processing Framework)
(File Sys. 2)	HDFS (Data Storage FW)
	JVM
(File Sys. 1)	OS (Linux) → (ext3 FS).
	Comp. H/c

Hadoop Cluster

- follows master-slave archi.
- Master node contains
  - Job Tracker node } MapReduce layer
  - Task Tracker node }
  - NameNode } HDFS layer
  - DataNode }
- Multiple slave nodes
  - Task Tracker (MapReduce layer)
  - Data node (HDFS layer)

⇒ MapReduce layer | HDFS layer



17/9/19

Reading a file

Client appln wants to read a file:

- It comm. with the NN to determine which blocks make up the file & which DN those blocks reside on
- It then comm. directly with the DN to read the data

Pipelining of Data

- Client retrieves a list of DN on which to place replicas of a block
- Client writes block to the first DN only
- The first DN forwards the block to next DN in pipeline
- After all replicas have been written, client moves onto next DN

Page No.
Date

Node failure
i) No heartbeats sent to NN
ii) Replication factor for that particular block decreases

MapReduce
→ It is a programming model & s/w framework
→ Aim at processing of vast amounts of data in parallel on large clusters of commodity hardware
→ Two phases: i) Map
ii) Reduce
→ MapReduce code is generally written in Java. (Others: Ruby, C++, Python)

Map step: Distribute work to all slave nodes and apply a prog. on each of them
Reduce step: Collect the results from the slave nodes & process further.

Executing Prog on Data

Wordcount example

200 MB file

HDFS default block size = 64 MB

∴ No. of blocks req = $\frac{200}{64} \approx 4$ blocks

→ No of mappers = no of input splits

→ Mappers & Reducers work with <key, value> pair

We need to convert a text line to <key, value> pair. It is done by RecordReader.

Every line → 1 record

key → <Byte offset, entire line> ← value

↑ no of chars including blank spaces

Three pairs depends on the formats:

i) TextInputFormat

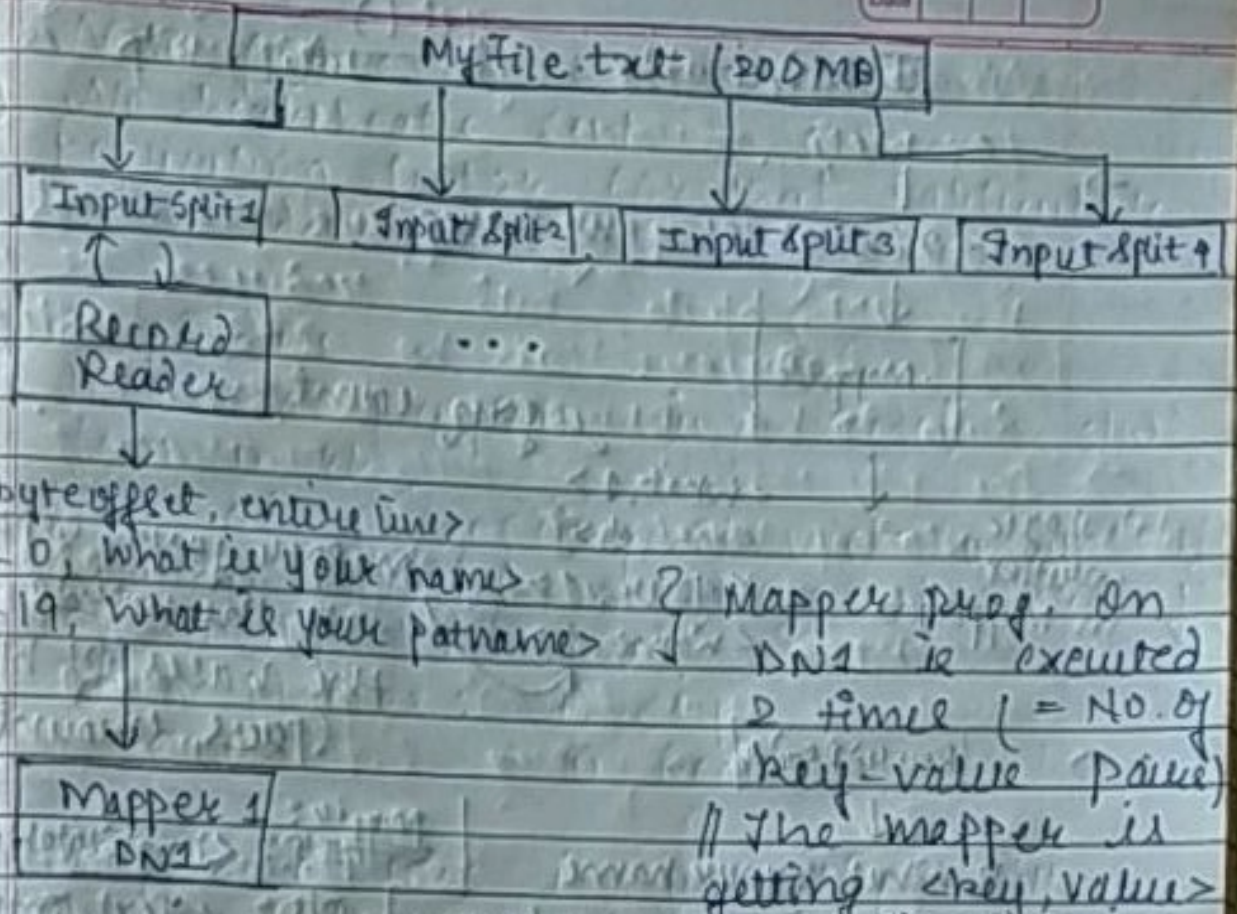
ii) KeyValueTextInputFormat

iii) SequenceFileInputFormat

iv) SequenceFileAsTextInputFormat

On what basis the RecordReader converts these records into <key, value> pairs depends on the format of the input file

18/4/



(Mapper)

Mapper prog. on DN1 is executed 2 times (= No. of key-value pairs)

// The mapper is getting <key, value> pairs. These pairs must be Objects.

→ Collection framework works only on Objects and thus, Hadoop has box classes which are similar to the wrapper classes in Java.

18/9/19

Box classes

Primitive DT	Java	Hadoop
int	Integer	IntWritable
float	Float	FloatWritable
long	Long	LongWritable
double	Double	DoubleWritable
String	String	Text
char	Characters	Text

int new IntWritable(int) IntWritable

IntWritable —get()→ int

→ <0, what is your name>

↓ Mapper

↓ <what, 1>
Logic written in Mapper.java <is, 1>
<your, 1>
<name, 1>

(seq. count)

Mapper.java separates all the words on the basis of blank spaces encountered.

→ RecordReader

0, what is your name
1, what is your patname

↓ Mapper

Mapper:
I/P: <LongWritable, Text>
O/P: <Text, IntWritable>

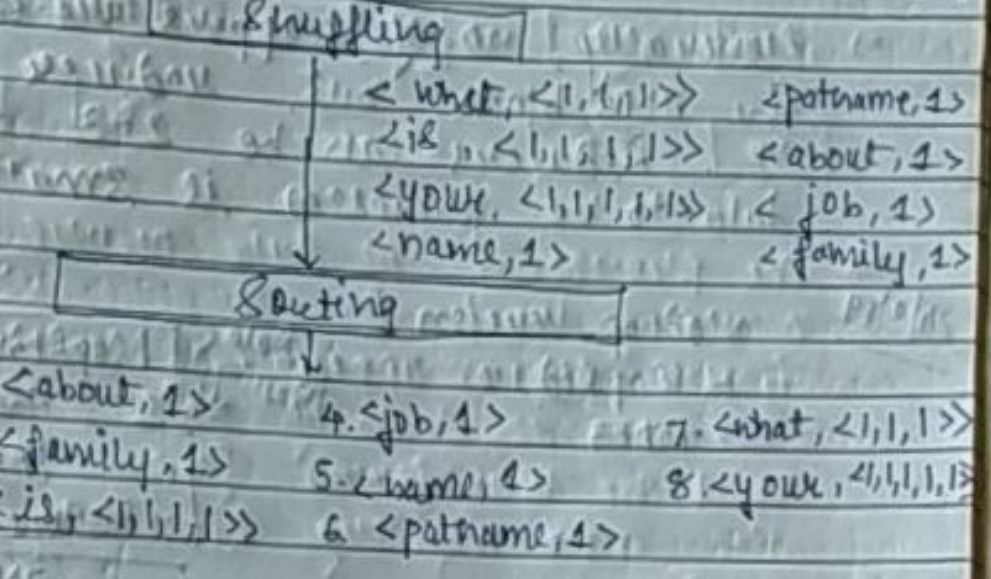
↓ mapper output. It also key, value pair. Data are called intermediate data.

<what, 1> <what, 1>
<is, 1> <is, 1>
<your, 1> <your, 1>
<name, 1> <patname, 1>

⇒ Intermediate data have duplicate keys (what, is etc). It can have duplicate values, but shouldn't have duplicate keys.

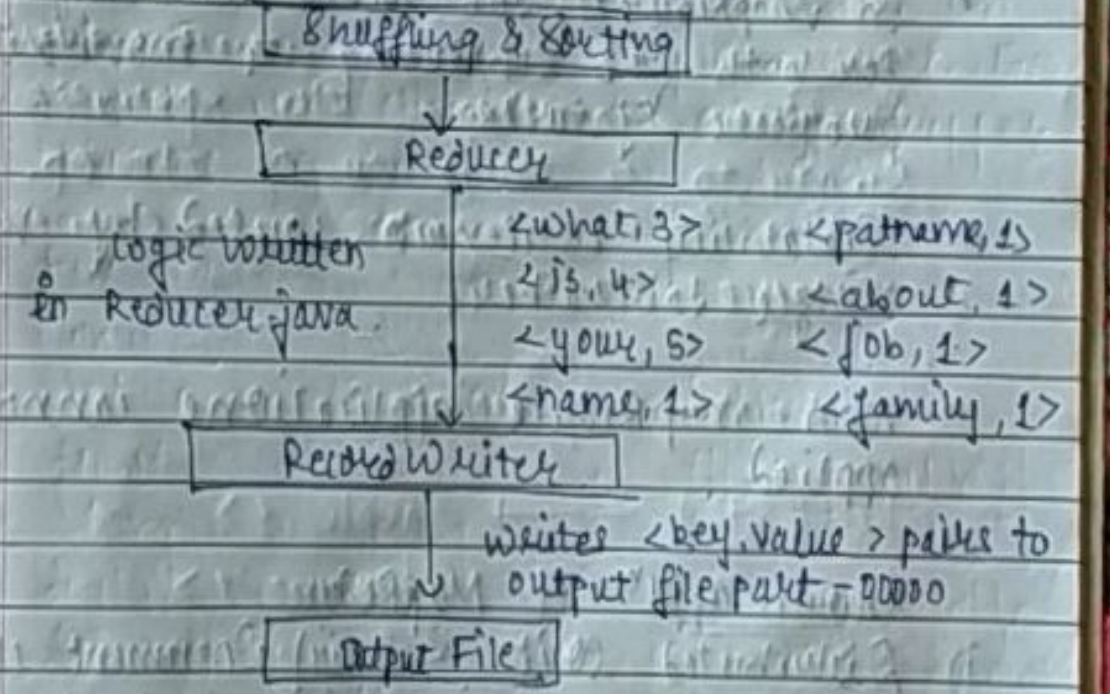
∴ Two more phases to be passed:
i) Shuffling
ii) Sorting

Shuffling: Combine all values associated to single (unique) identical keys



Reducer

It will combine the <key, value> pairs generated by different Mappers.



⇒ By default, Reducer in Hadoop framework provides sorting functionality only.

⇒ No reduce can begin until map is complete.

⇒ Generally, no. of mappers are equal to the HDFS block but, reducer is only 1. Reducers can be 2-3 unless data to be processed is small.

24/9/19

Hadoop Version 1 — V2

= MapReduce (Data Processing) | MapReduce | Other Frameworks

= HDFS

YARN

Resource Management

HDFS

Distributed Redundant storage

YARN: (Why needed?)

* In MapReduce1, JobTracker is responsible for both job scheduling & monitoring progress of tasks.

* Scalability bottleneck caused by having single JobTracker.

* Non-mapReduce applications cannot be applied.

Known appln of MapReduce

i) Distributed sort ii) Document clustering

iii) Analyzing & indexing textual info. iv) Analyzing similarity of users behavior

⇒ Big data analytics is the process of examining large data sets to uncover hidden patterns.

Two types: (a) Batch analytics
(b) Real-time analytics

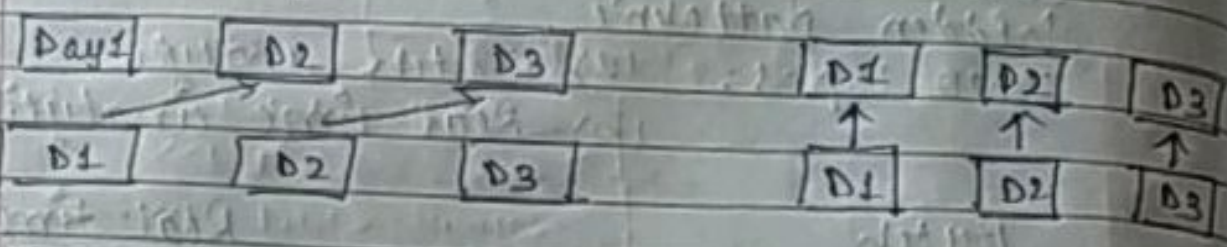
Batch	Real-time
* The collection & storage of data for processing at a scheduled time when a sufficient amount of data has been accumulated.	* The immediate processing of data after the transaction occurs with the DB being updated at the time of the event.
* Ex- Cheque clearing, generation of Bills,	* Ex- Ticket Reservation systems.
* Many transactions take place at once.	* The act of processing data is repetitive.
* Data takes time to be processed	* Data is processed immediately.

⇒ Apache Spark can be used for near real time processing of data as well as batch processing of data.
Otherwise most of the hadoop framework is used for batch processing.

Hadoop

Spark

Processing Data



⇒ The processing of data occurs at memory level.
First, the mapper code will run when it is run of DN1, the block 1 will be transferred from hard disk to the main memory. Similarly, for block on DN2.
After processing, result is again transferred from HD to MM while shuffling / sorting / reducing, all the results are collected onto one slave machine which again requires multiple I/O operations. Owing to a large no. of I/O op., the map-reduce programe are slower in nature.

⇒ Without map-reduce

(A, B, D) ⟶ <apple, 1>
(A, B, D) <banana, 1>
<orange, 1>
:
:

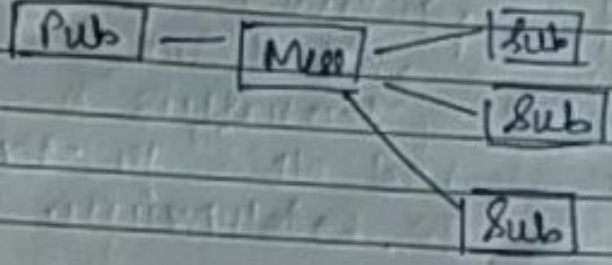
Now, when apple is encountered prog. searches for it in the previous pairs & if found, increases its freq count this is time consuming

Page No.
Date

10/19

JMS

1) Synchronous
2) Asynchronous
   - Topic
   - Queue



Admin console:
Username : admin
P : adminadmin

JDBC → JMS Resources
Connection factories (create new)
Destination Resource (create new)
↳ Select Topic / Queue. (here, Queue).

→ Microservice
→ JMS
→ Hadoop.