Syllabus

Content

1. Theory → Into to BD, Hadoop, YARN, &
2. Lab → Linux lab Activity
   → HDFS Lab Activity
   → Spark RDD
   → Spark SQL & Data frame
   → Spark Streaming analytics
   → Hive Activity
   → Spark ML Activity.

Drawbacks of Traditional approach:-

(1) Storage is less
(2) Types of data (it holds only structured data)
(3) Processing (one machine cannot handle all the Unit's data)

Components of Hadoop

HDFS will handle.
+
Mapreduce will process.
=
Hadoop.

Data Base → Collection of the data.

Structured (Rows & Column) Data.
≤ 20%

Semistructured & unstructured. (Jason file, Audio, Images, etc)
> 80% of data.

For this kind of data we need Big Data Concept.

(at) Hadoop is the tool to handle this unstructured Big data

Linux Activity :-

Typically in a project we work in a different environment.

Project life cycle

1. Client (Gives its requirement of project → ML / DE)
2. Understanding the project and work with team

For ex:- To built the machine learning Application.

Frontend (web UI) → Server → returns the output for some prediction

3. Deploy Application (software Dev phase) :-
   - Development phase → In this phase diff engineers work on different components of the entire application. (*) After the diff codes are integrated.
   - Testing phase
   - Production phase

Here in this phase, Machines used are the Linux OS

Linux OS :-

1991 — Linus Travolds built it

(*) Advantages :-

- Open Source (Base Kernal was in C & C++)
- Security is high
- very light & high stability
- Dont need quick restart.
- High Performance.
- Preffered in Embedded Systems
- Many Linux distributions (Ubuntu, Fedora, etc)
- It provides Shell scripting
- Huge Community Support
- used in Networking

(*) Disadvantages :-

- CLI, Commands
- No. Standard edition of Linux.
- Learning curve is very high.

We use Command Line Interface (CLI) to user with Linux.

→ Different users in Linux :-

(1) Root user :- Created by default in OS.
→ used to make changes in administration in OS

(2) Regular user :- * used for many functions
* It has username & password

(3) Service Account :- * used for some installation etc.

* We use Regular user on the Jupyterhub (Terminal)

(1) pwd → present working directory.

(2) Uname -a → Displays OS we use and time of server, Linux distribu

$ → promt

(3) whoami → name of user

(4) ls -l / → (Root directory) List format
Linux file system
→ listing the items in current directory

Linux file System

/ → Root directory. (Main folder)

diff folders (directory) :-

1. Bin → user Binaries
2. boot → boot loder files
3. Cdrom → Cd disk. that has to be played
4. dev → device files
5. etc → for Configuration files
6. Home → various users
7. lib → various libraries that could be installed
8. media → removable devices
9. mnt → mount directory
10. opt → Optional add-on directory
11. proc → Some kind of informations of proc
12. Sbin → System binaries
13. Srv → Services data
14. tmp → temporary files are stored
15. usr → user's utilities and applications
16. var → variable files are stored

(*) How to create a directory (folder) in Linux.

↳ mkdir foldername

ex :- xyz@fractal :~$ mkdir 22jan.

(*) How to enter the directory.

↳ Cd. 22jan. or Cd/home/name/22jan

(*) How to enter home folder. (root directory)

↳ Cd.

(*) Cd - → Take us to previous folder

(*) Cd .. → Take to parent directory of folder

Permissions :-

d → directory

– → file

r → read (4 → 2^2)

w → write (2 → 2^1)

x → execute (1 → 2^0)

rw → (4+2) = 6

rx → (4+1) = 5

wx → (2+1) = 3

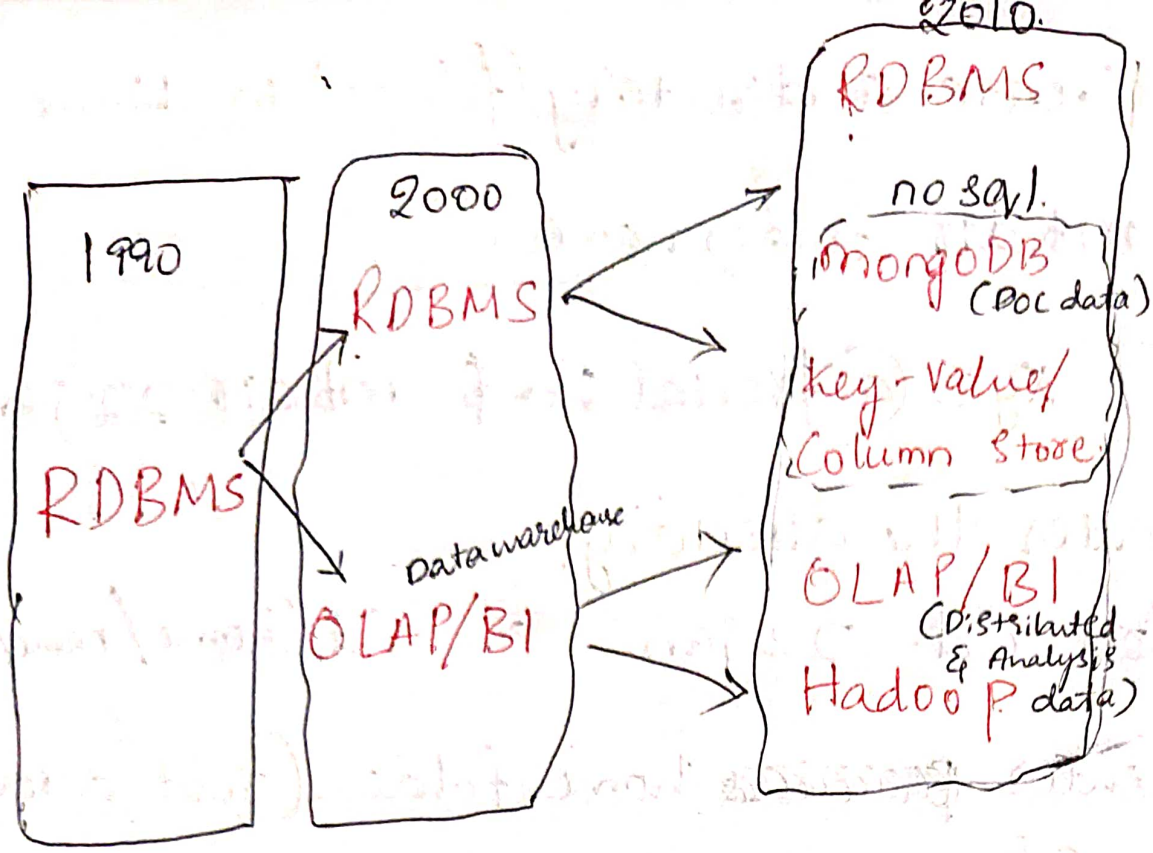
rwx → (4+2+1) = 7

} values

7 7 7

drwx rwx rwx

user's only | group | other



Transition from databases to data warehouse to datalakes

What is Big Data?

(*) V's of BD → Volume, Velocity, Variety, Veracity

Speed at which data is generated

speed at which it has to be analysed for actionable insights

Variety → Structured (in rows & columns), Semi-structured (in b/w), Unstructured (email, Photo, etc)

Veracity → Untrusted, Uncleansed

Big data is high-volume, high velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, & process automation

Volume, velocity, variety, & veracity

New Type of Data :-

Sentiment	Clickstream.	Sensors	Geographic
understand How.	Capture and analyse (website visitors data trails Ex optimize your web)	Discovers the patterns	Analyse locations based data

Server logs	unstructured
Research logs	web pages, emails, docs, etc

Big Data. - model building Issues. :-

(*)

⊛ Hadoop :- (OS)

Hadoop is essentially something, where we can learn software on diff machines. Since we have huge amount of data and we have to process it.

⊛ Hadoop is an open source framework, capable of processing large amounts of heterogeneous data sets in a distributed fashion across cluster of commodity computers & hardware using a simplified programming model.

→ Hadoop framework is based on following principle :-

⊛ In pioneer days they used oxen for heavy pulling, when the load got heavier, they did not grow a larger ox, we should be trying for bigger computers, but for more systems of computers.

HDFS and MapReduce

HDFS → Reliable Shared Storage.
(Hadoop distributed File System)

Distributed Storage
↓
All systems are interconnected to each other

+

MapReduce → Distributed computation.

Parallel processing
↓
All process happening in same time.

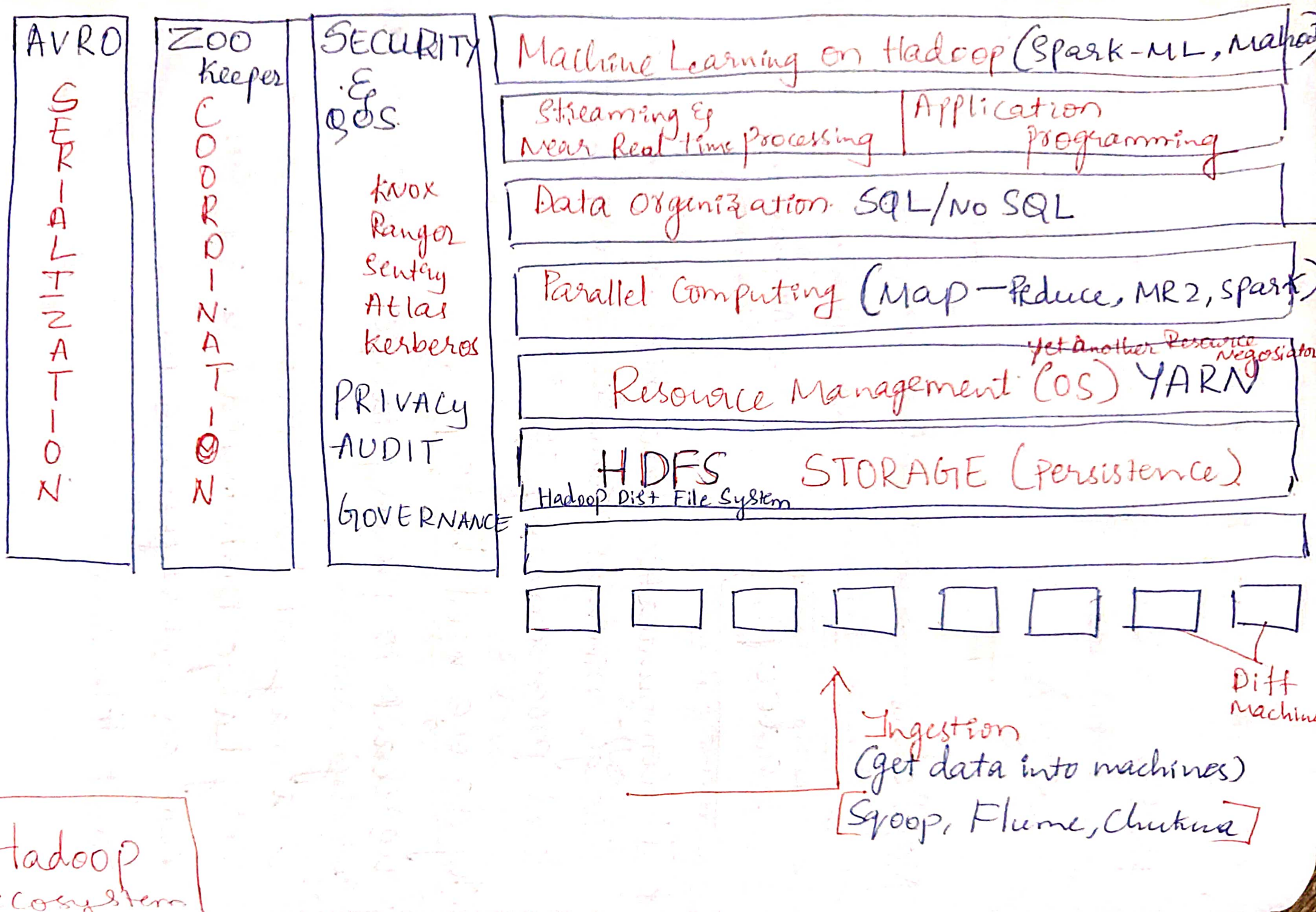
=

Hadoop

* It communicates b/w diff nodes (machines) through minimal transfer of data and high level programming is done.

Why do we need hadoop :-

* We want to process petabyte of datasets very quickly
* Data may not have strict schema.
* Very Large Distributed File System.
* Runs on heterogeneous OS.
* Fault tolerant.


AVRO
SERIALIZATION
ZOO Keeper
COORDINATION
SECURITY & QOS
Knox
Ranger
Sentry
Atlas
kerberos
PRIVACY
AUDIT
GOVERNANCE
Machine Learning on Hadoop (Spark-ML, Mahout)
Streaming & Near Real Time Processing
Application programming
Data Organization SQL/No SQL
Parallel Computing (Map-Reduce, MR2, Spark)
Yet Another Resource Negosiator
Resource Management (OS) YARN
HDFS
Hadoop Dist File System
STORAGE (Persistence)
Diff Machine
Ingestion
(Get data into machines)
Sqoop, Flume, Chukwa
Hadoop Ecosystem

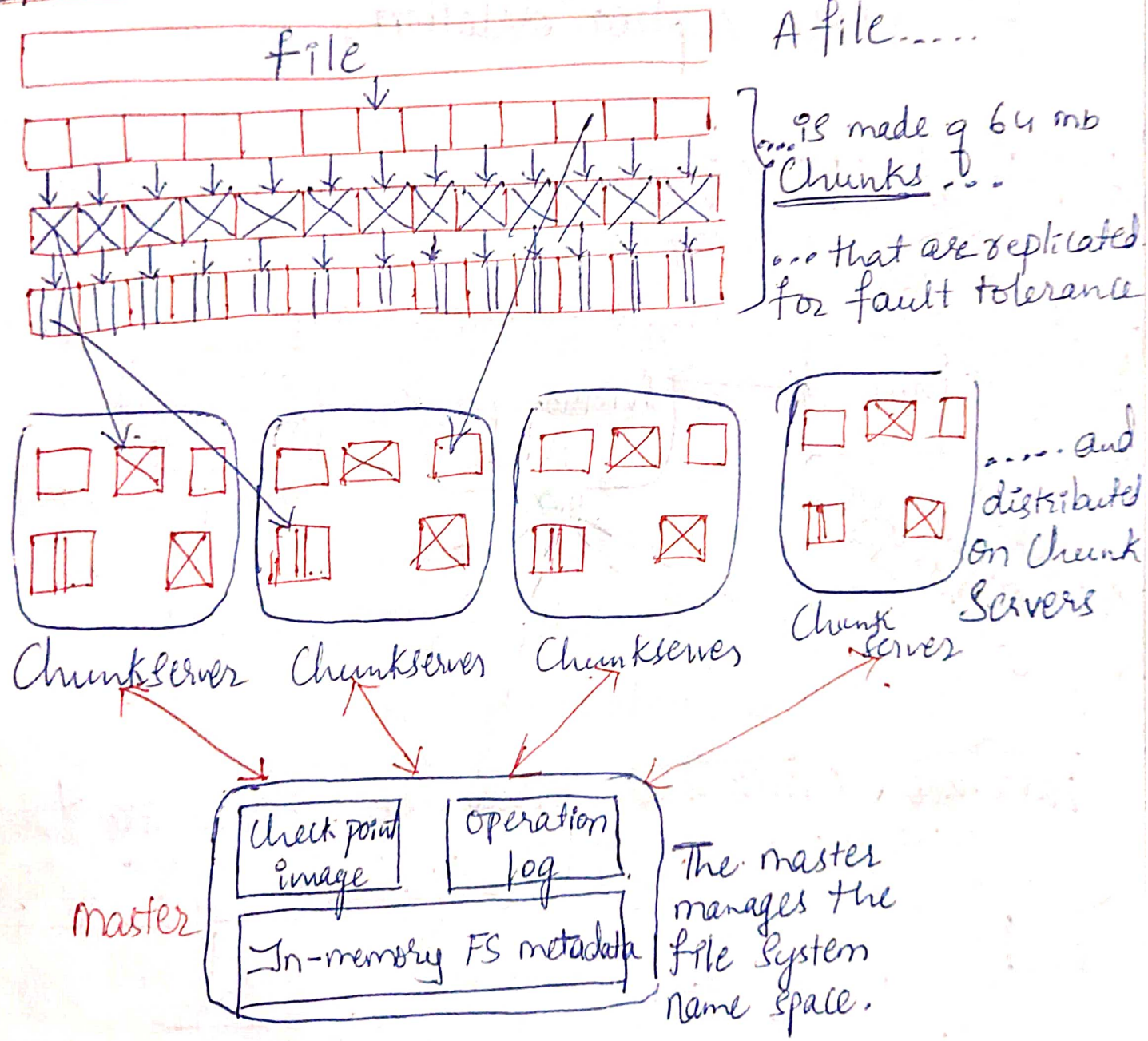
Major components of Hadoop :-

(1) The Storage Layer

(GFS → Google File System)

HDFS → Hadoop Distributed File System)

GFS

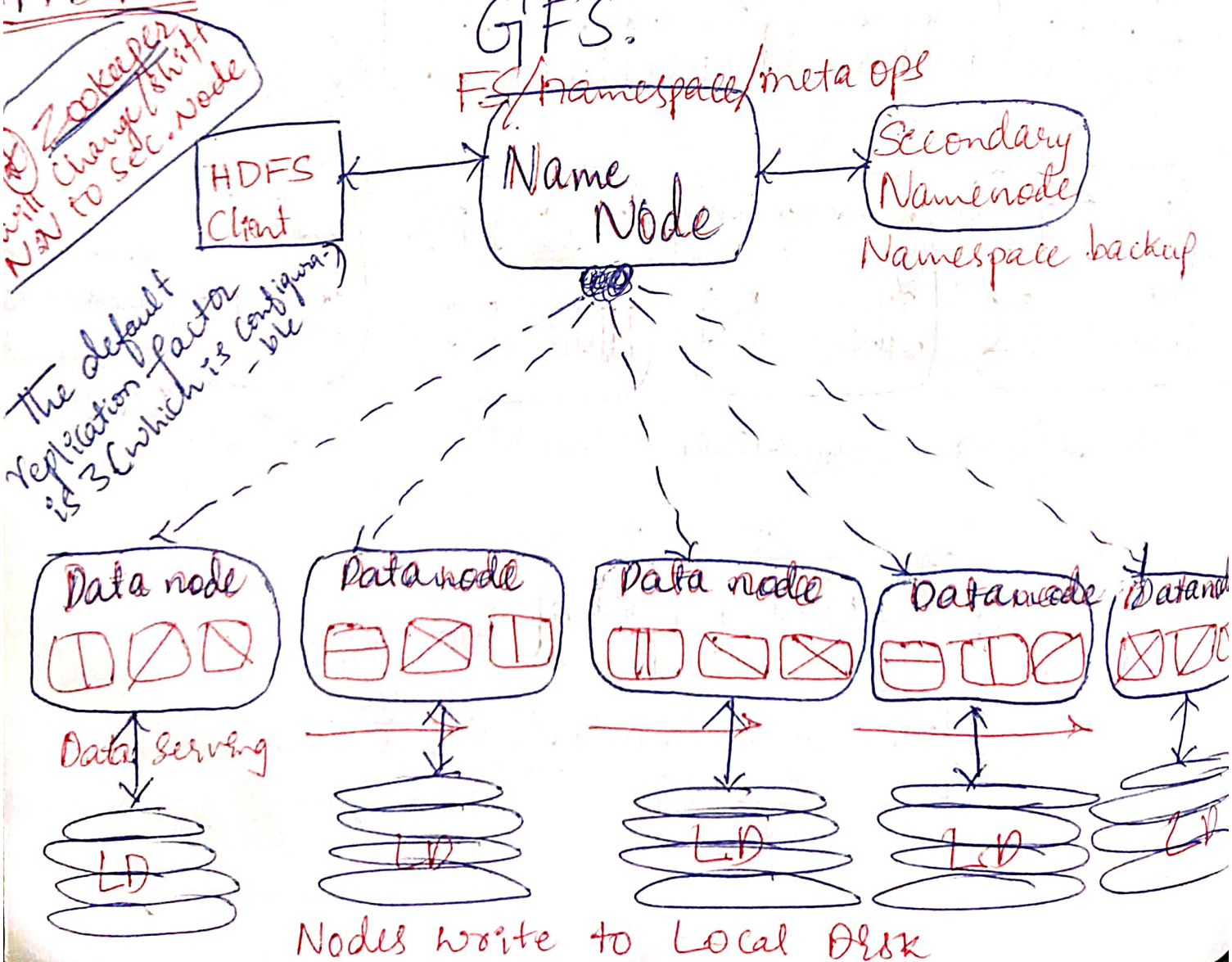


GFS Master Responsibilities

- Metadata Storage (Information of the data in Datanode)
- Namespace management/locking
- Periodic communication with Chunk Servers
- Chunk creation, re-replication, rebalancing
- Garbage collection.
- "Stale replica" deletion.

HDFS

COH3: Open-source reimplementation of GFS.



(*) Namenode manages overal file System meta data
↳ It can be a single point of failure in the Systems

(*) Datanodes (one per machine) stores and retrieves actual data
↳ reports to ~~datanode~~ namenode

(*) Both datanode & Namenode include a heBserver, so node status can be easily checked.

(*) Secondary namenode: does housekeeping & acts as a memory backup.

(*) Namenode is "rackaware" → Knows how machines are arranged.

(*) Namenode doesnot directly read & write data.
↳ Clients gets data location from namenode
↳ Clients interacts directly with datanode to read/write data.

(*) Namenode keeps all block metadata in fast memory.

(*) For every 3 seconds every datanode sends the Signal to namenode with the data init, this is called Heartbeat

Rack Awareness :-

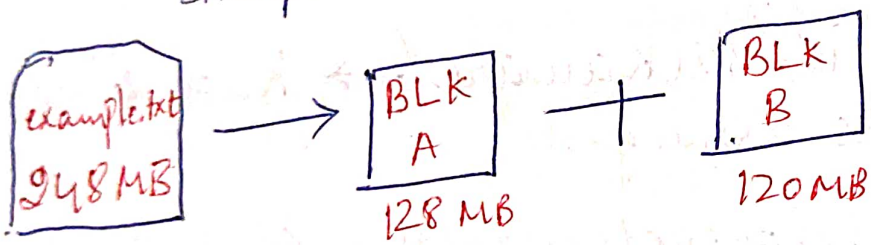
(in-built Rack Awareness Algorithm.)

* Hadoop has the Rack Awareness Alogrithm in it.
  - ↳ It improves the network performance
  - ↳ To prevent loss of Data

HDFS write Architecture :-

* write once, read multiple times

Storing the data



* At first, the HDFS Client will reach out to the name node for a write request against the two blocks, says Block A & Block B
* The name node will then grant the client the write Permission and will provide the IP adresses of the DataNodes where the file blocks will be copied eventually.

For Block A, list A = {IP of datanode 1, IP of DN 4, IP of DN 6}
For Block B, set B = {IP of dn 3, IP of DN 7, IP of DN 9}

There are following steps to undergo this:

1. Setup of Pipeline. (Path)
2. Data streaming and replication
3. Shutdown of pipeline. (Acknowledge new storage)

(1) Setup of pipeline means the Name-node will give the Ip address of the datanodes to the Client for the Storage

(2) Data streaming and replication includes the writing of the data in the specified datanodes.

(3) Shutdown of pipeline means intimating the namenode that the writing process has been completed and then the namenode stores it in metadata

HDFS Read Architecture :- It is very simple, the Client must specify the blocks (data) to be read, the name node will find the location of the blocks (if they are in one Datanode or not), then sends the IP address of the datanotes to read the data. to. Clients.

MapReduce :-

(*) MapReduce is used for batch Proc (Historical data)

Drawback

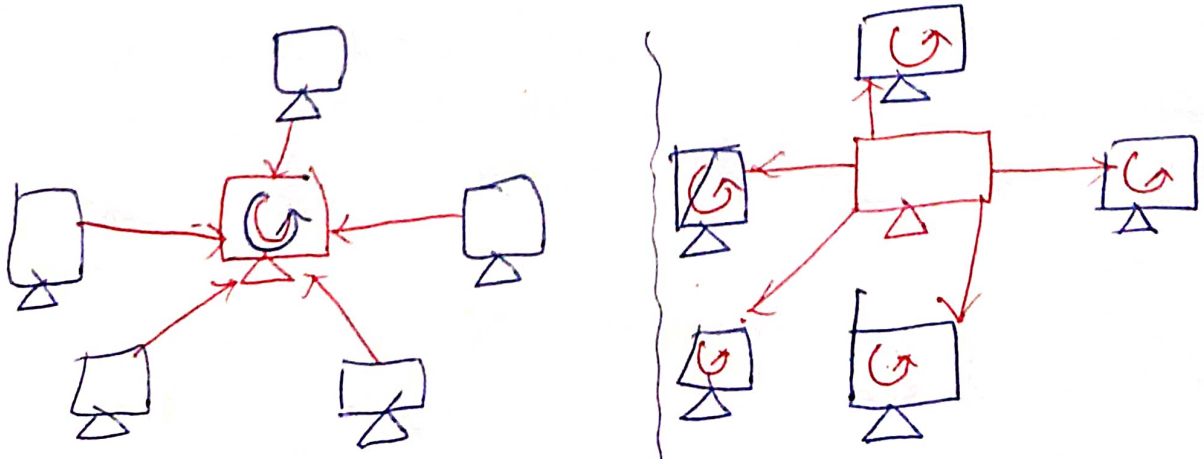
(*) MapReduce is a programming model for Processing large data sets with a parallel and distributed algorithm on a cluster

(*) MapReduce when coupled with HDFS can be used to handle big data. The fundamen -als of this HDFS-MapReduce System, which is commonly referred as Hadoop

(*) The basic unit of information, used in MapReduce is a (key, value) pair. All type of structured and unstructured data need to be translated to this basic unit, before feeding the data to MapReduce model

Advantages of MapReduce :-

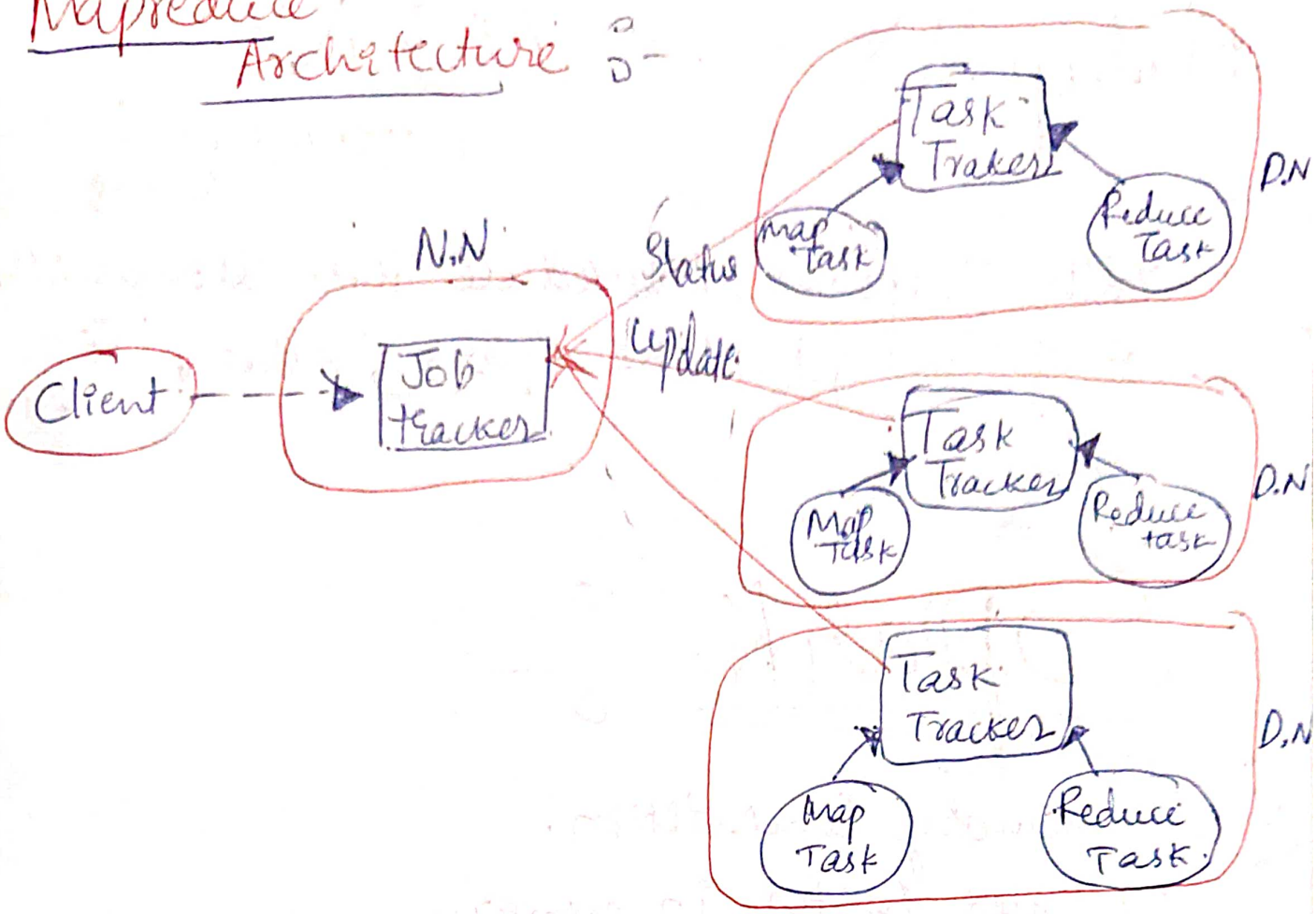
(1) Parallel Processing :-



Moving data to Processing unit (Traditional Approach)

Moving Processing unit to the data (MapReduce Approach)

Mapreduce Architecture :-



Drawbacks :-

1. Batch processing
2. not suitable for Real time Data / Data streaming
3. It supports upto 4000 Nodes per Cluster
4. It has a Single component : Job traker to perfo many activities like Resource Management, Job Scheduling, Job Monitoring, Re-scheduling Jobs etc
5. Jobtraker is Single point of Failure(N
6. It runs only Map/Reduce jobs
7. It follow slots Concept in HDFS to allocate resources.

Hadoop 2.X → YARN

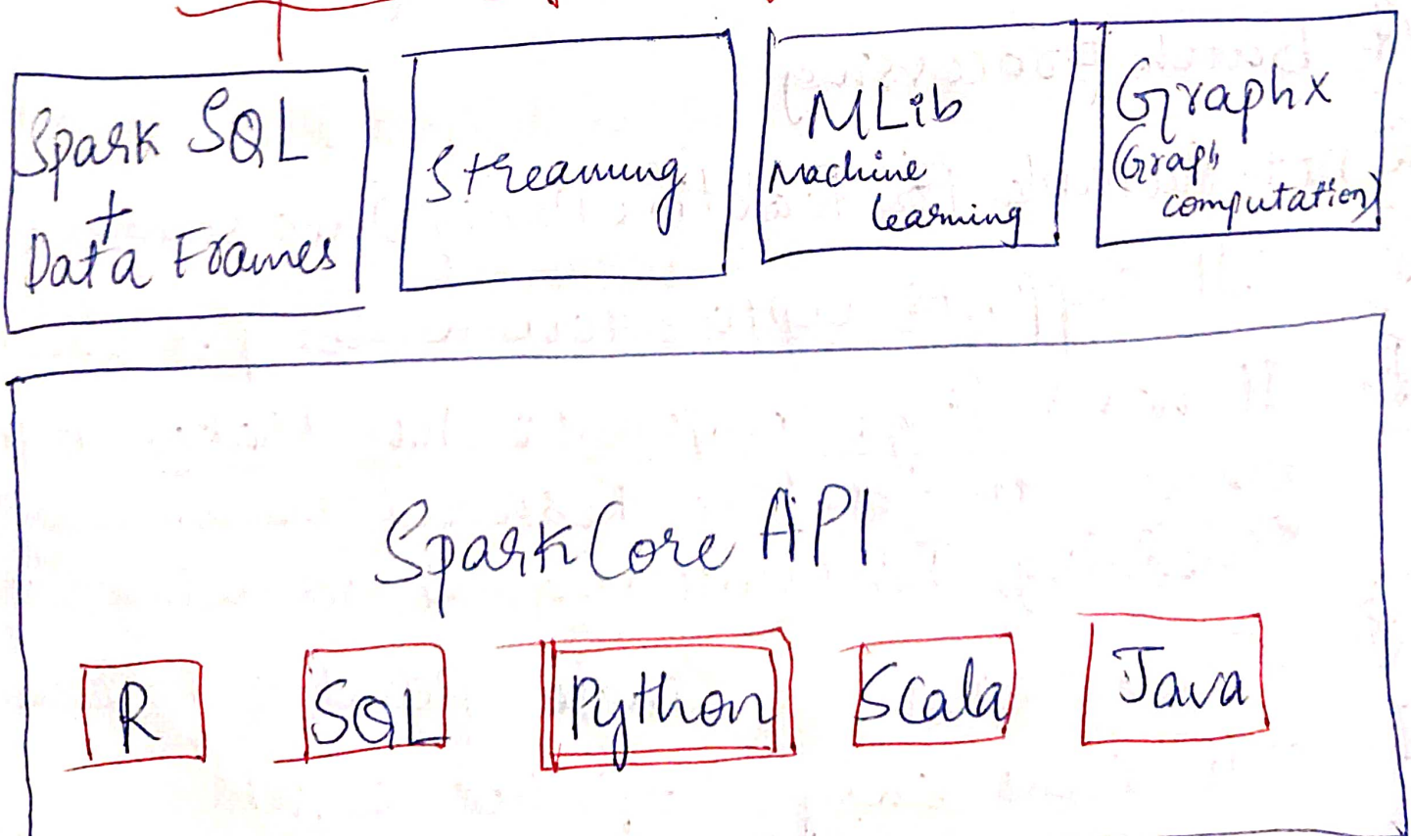
(Yet Another Resource Negotiator)

* Yarn performs Mapreduce jobs along with many other jobs like Spark, HBase, Tez etc

Spark :-

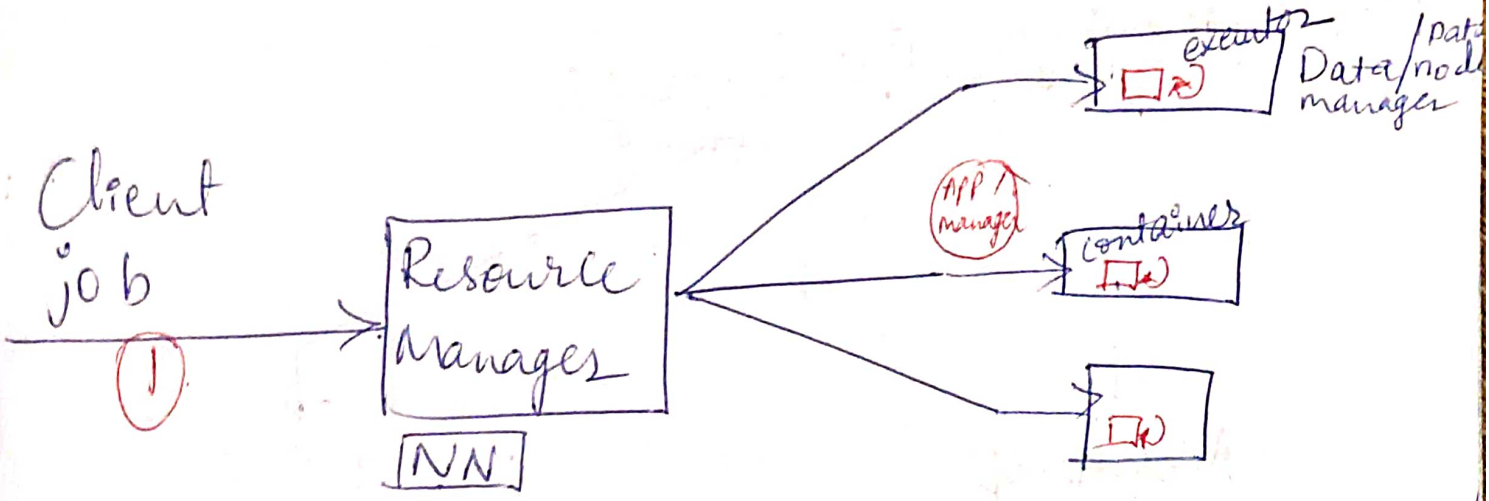
* In-memory Computation.
* 100 times faster in memory
* Supports → ML, SQL, Streaming data, Graph algorithms

Spark Ecosystem



Spark & YARN

Architecture.



Steps

1. Client job
2. Creates application master
3. R.M Creates Container (memory location)
4. Application master sends the job to Container & executes the job
5. And the job is sent back to. Client

Yarn	Spark.
* It will handle resource managers	* Spark says "what job to be done."
* Allocates memory location, Application manager etc.	* It is in memory computation.
	* It executes the job in Container

Spark

Parallel processing → Mapreduce
Parallel processing → Spark

Spark → Storage

Storage → Memory (RAM)
Storage → HDD (ROM)

Spark (In-memory computation)

Mapreduce → read & write.

Spark-execution :-

Spark → RDD (stores it to) → DF → SQL.

There are 2 operation (i) Transformation.
(ii) Action.

Action → This submits the job to YARN.

YARN → Resource manager layer

* It will assign executo to solve this job using applicat^{n} manager.

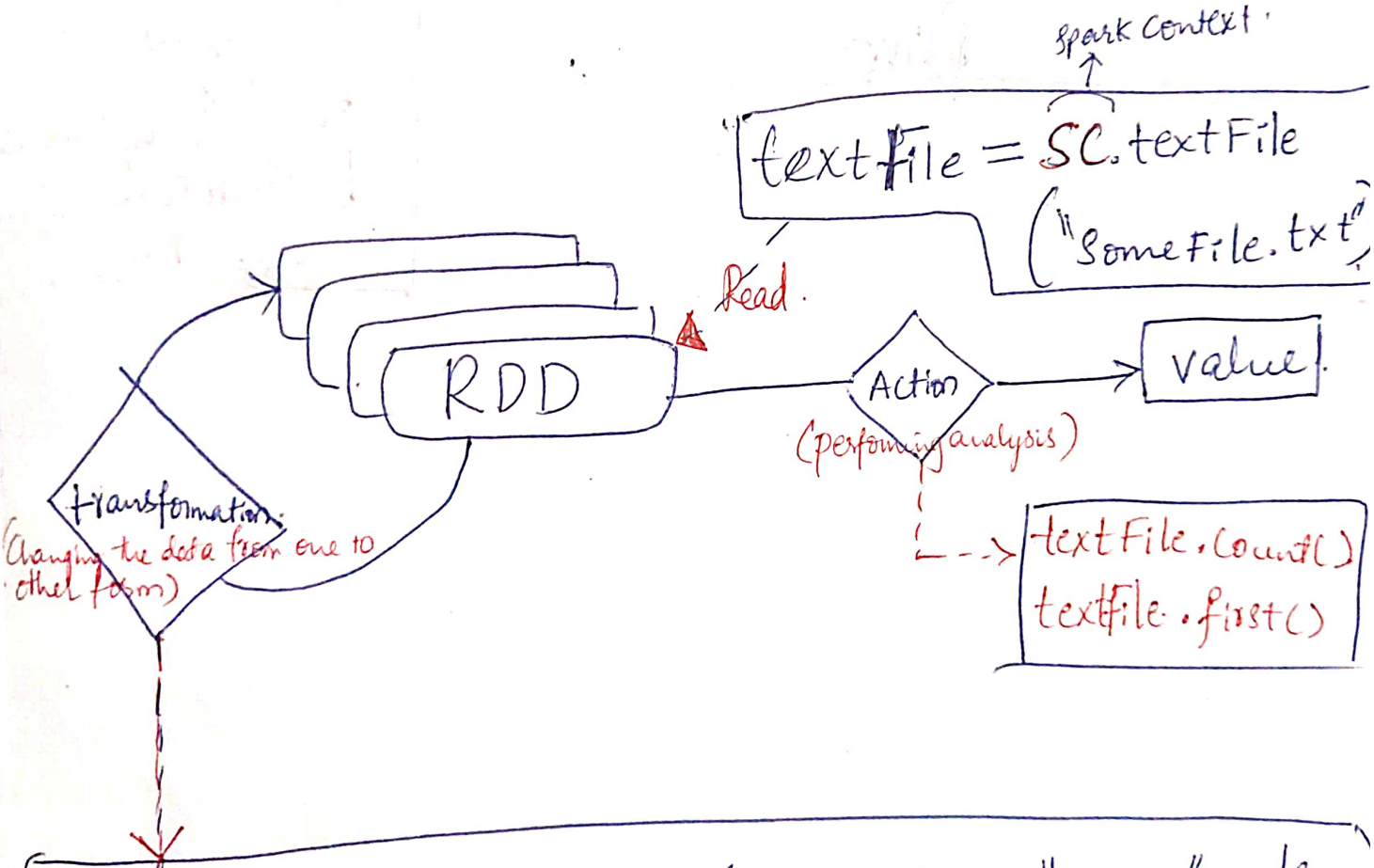
Resilient Distributed Dataset (RDD) - Key Spark Construct

↳ RDDs represent data or transformations on data.

(transformations → Sum, percentage or Avg, etc)

↳ Actions can be applied to RDD.

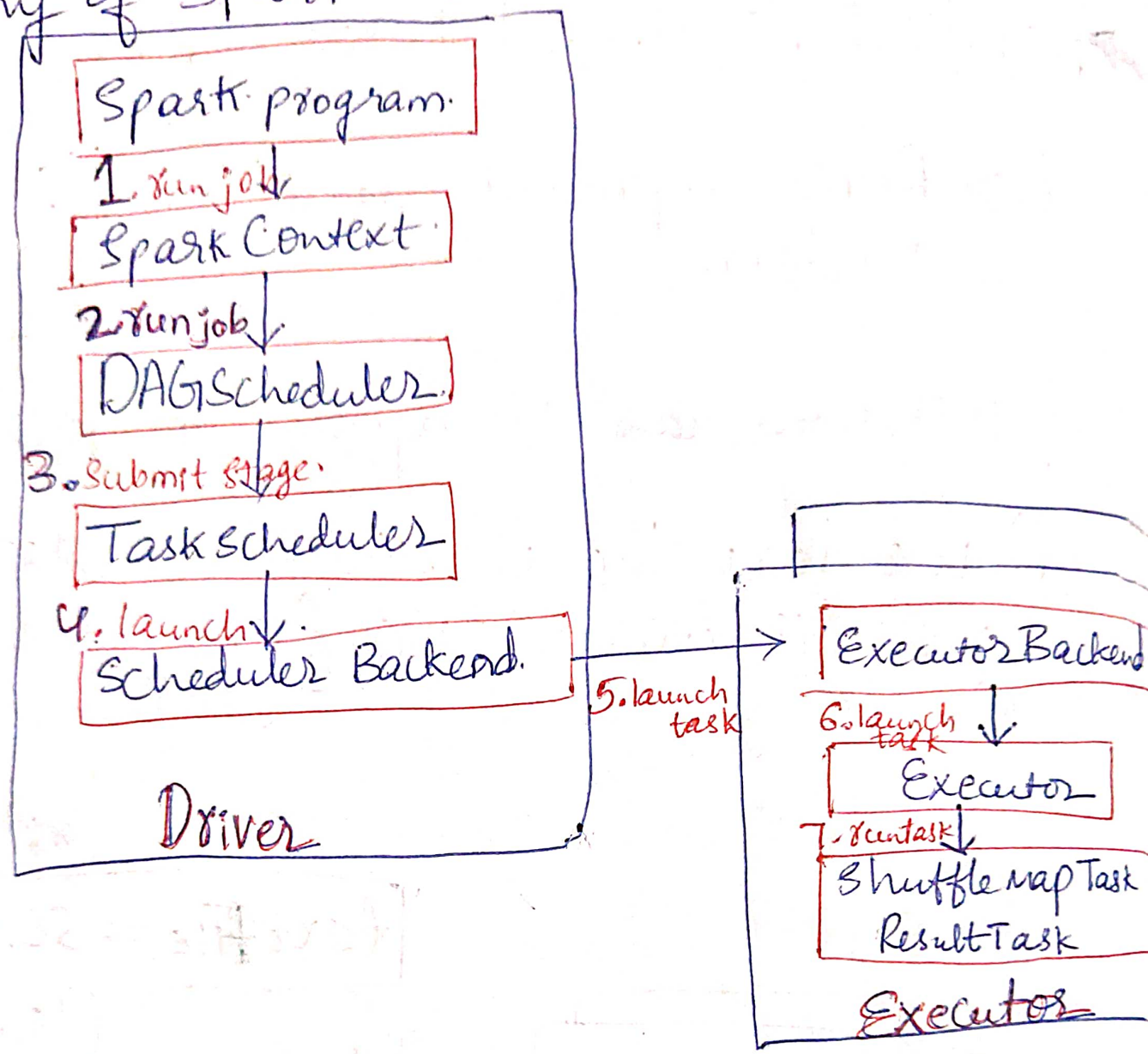
* we load the unstructured data and get structured data (pre-processing)



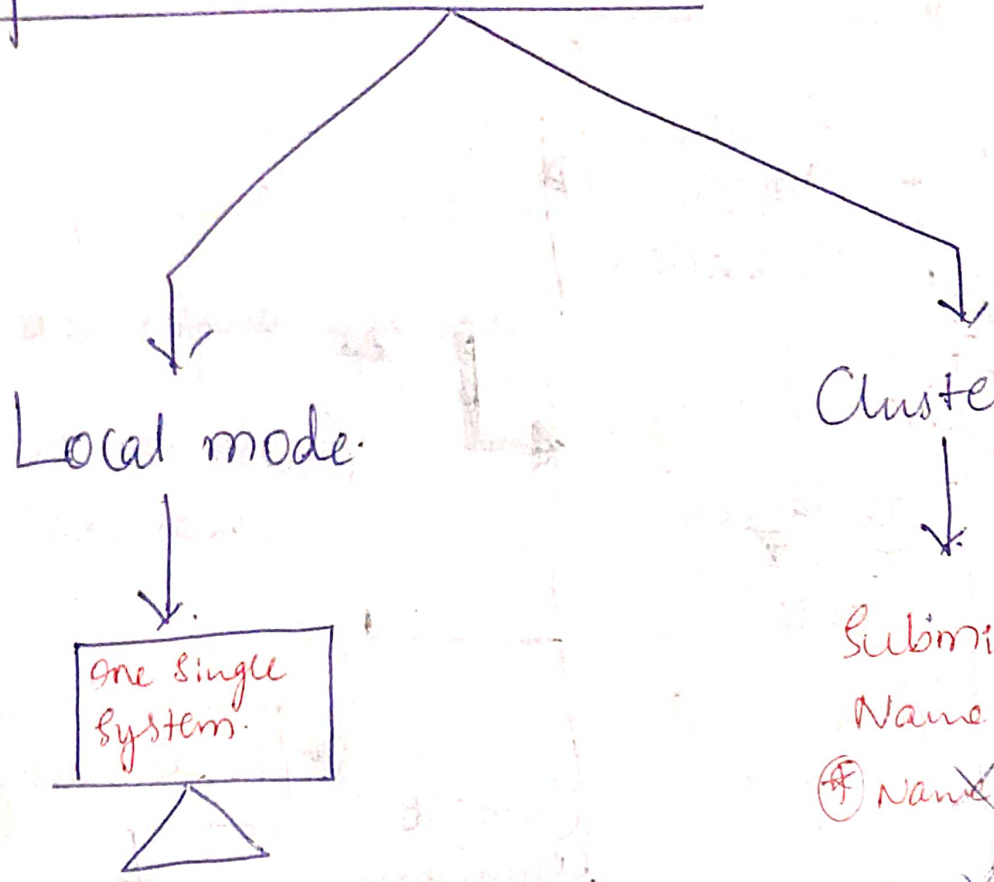
linesWithSpark = textFile.filter(lambda line: "Spark" in line)

* we can modify RDD, therefore for any change we do transformation and store as new RDD.

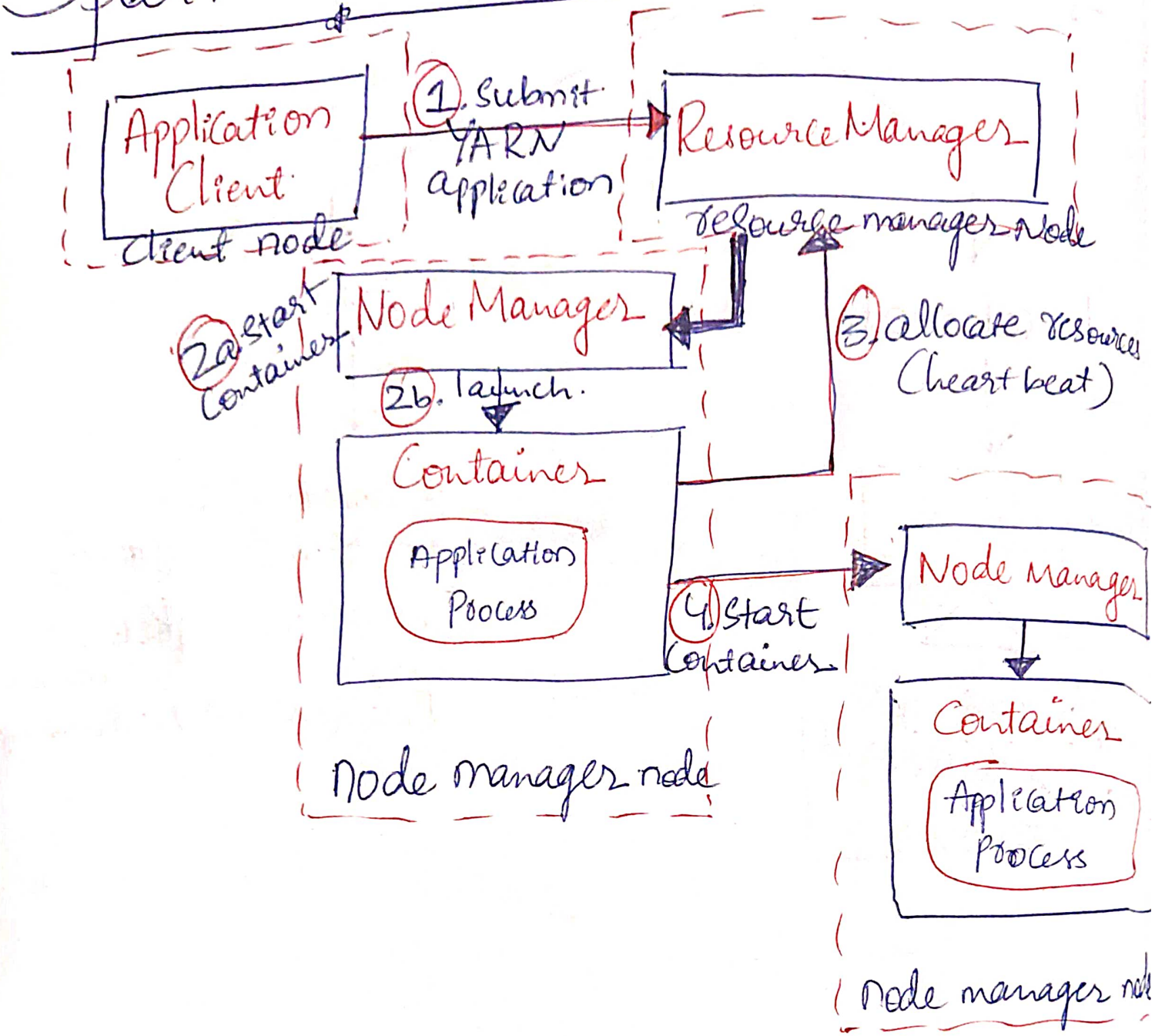
Anatomy of Spark Job Run :-

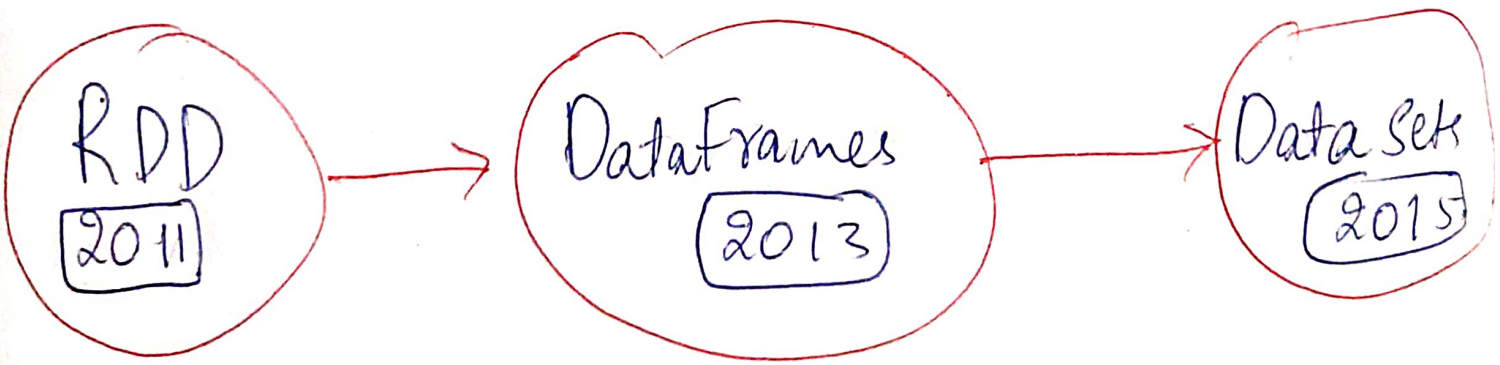


Spark Job Execution.



Spark on Yarn





Spark + ML :-

Dataset (Given)

↓

We need to build the model.

↓

① preprocessing (spark) + ML model.

∴ Spark ML.