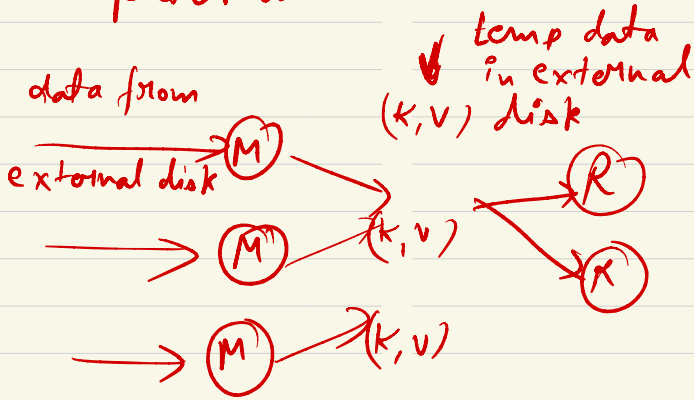
** Apache Spark (Very Imp)

⇒ Hadoop vs Spark

*) Hadoop's Drawback

- It was only made for batch data processing
- Slow processing because no in-memory Computation.



- Not fault tolerant in terms of computation
- No mechanism for caching & persist

(Hadoop) → Map-Reduce
→ HDFS
→ Yarn (Resource Manager)

Apache Spark

- It is a distributed computation framework or engine.
- It doesn't have any separate storage or file system.

Features of Spark

1. **Speed** → Spark is 100 times faster than hadoop because it does in memory processing.

2. **Powerful Caching** → Capability to persist data in memory.

3. **Deployment** → Spark can work with different resource managers.

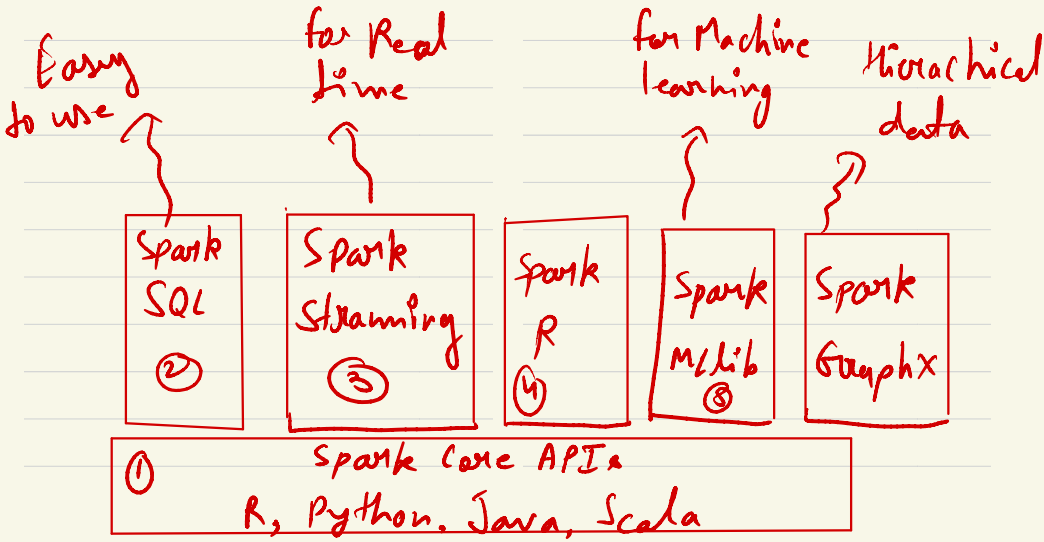
   → YARN
   → Mesos
   → Kubernetes
   → Spark own cluster Manager

④ Good fit for Batch & Real time data processing.

⑤ Polyglot: Spark provides high level api or libraries for different language:

- Python (PySpark)
- Java
- Scala
- R

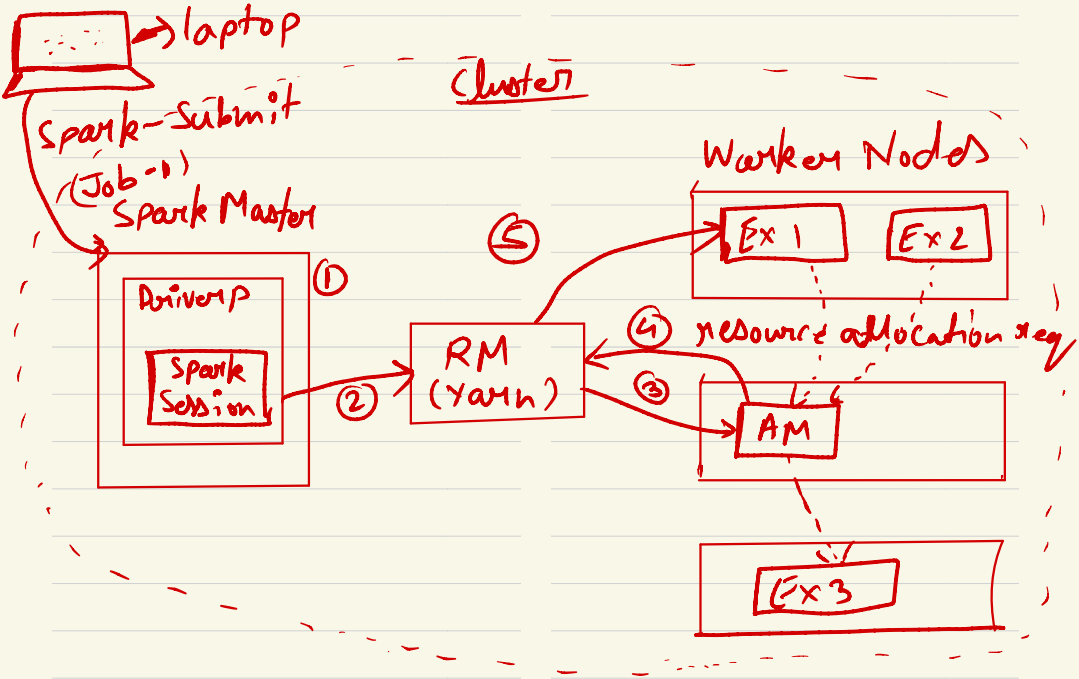
Apache Spark Ecosystem



(somedata.groupBy(Col1))

Spark Architecture

- Spark follows Master-Slave Architecture
- Spark Master, Spark Worker Node



① Driver Program

- It acts like a main method which will create spark Context.
- The first method which will be called.
- It does all management related things for a job like; lineage

preparation, scheduling the tasks, data distribution, communicate with RM

Spark Context or Spark Session

- It is the entry point to spark core of spark application.
- It will connect spark application with execution environment or cluster.
- befor spark 2.0 it was spark Context & after 2.0 it is spark Session

Resource manger

- It will create the very first Container & start Application master service in it.
- Application master will request to RM for Complete resource allocation or to start required number of executors.

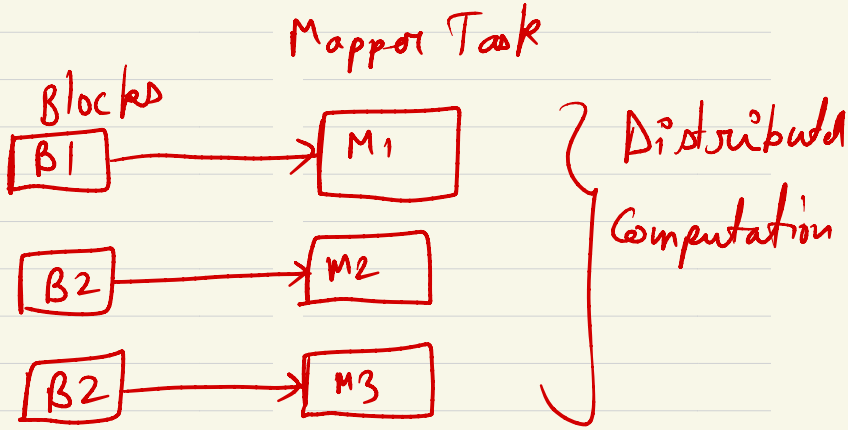
- **Executors**: It does the actual

Computation. They are lik virtual Containers on worked node which has CPU cores, RAM, Network Bandwidth

→ RM will create required number of executors

→ Executors will start interacting with AM to send the updates for Job progress etc.

in Hadoop



1 Core will process 1 Mapper Task

Distributed Computation in Spark

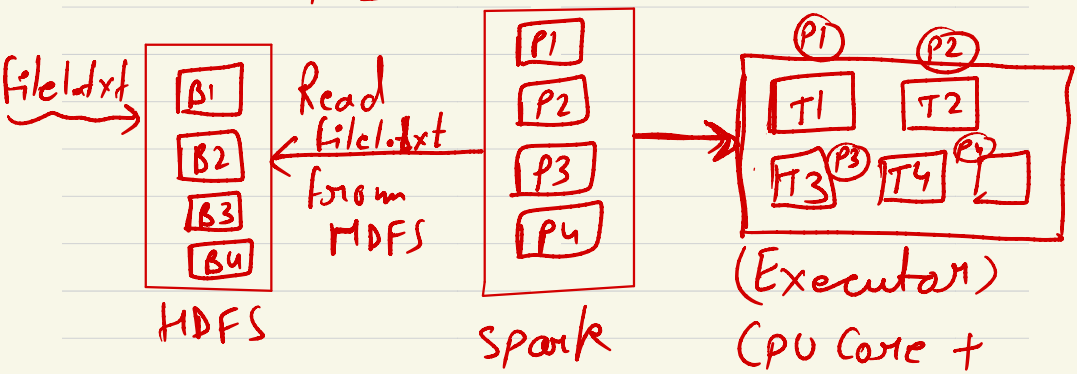
→ In spark, data will be divided in number of Partitions

↳ How Spark does parallel processing?

↳ Either Spark read data from

① distributed file system, databases

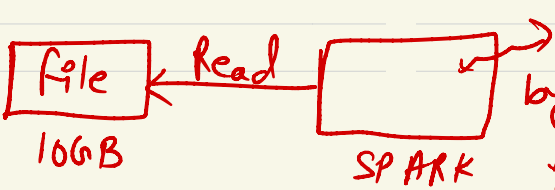
$P \cong B$



(Executor)
CPU Core +
RAM +
Network Band
(1 Ex = 5 CPU Core)

in Spark we will have a controle to decide the number of executors, it's cpu core, memory etc

↳ ② OR divide data in partitions explicitly in Spark



raw-dat = read.partition(n)

by default Spark will divide in 250 Partitions

⊗ Tasks

- Smallest unit of execution is known as Task
- Each task will process one data partition
- One task will get executed on one CPU Core
- One executor can run multiple tasks and equal to number of CPU Cores.

Q.) Parameters in Spark Submit Command?

- num of executors
- driver memory
- executor memory
- executor cores

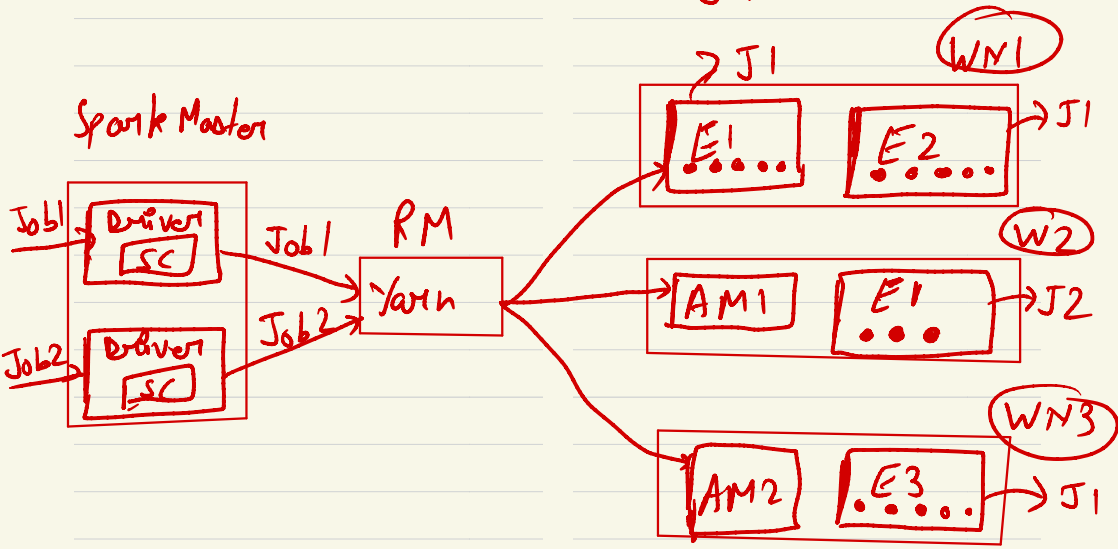
Spark with YARN as RM

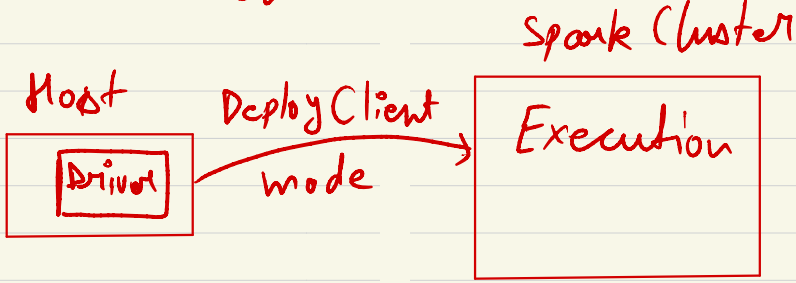
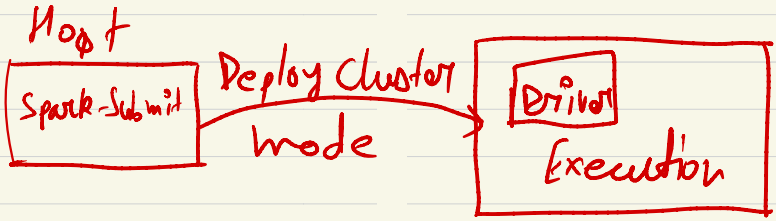
Ex: Workernode ⇒ 256 GB, 32 CPU Cores

We want to execute 2 Jobs, Job1 & Job2?

What we need for Job1 ⇒ 3 executors,
5 CPU cores each
32 GB Memory each

What we need for Job2 ⇒ 1 executor,
3 CPU Cores
16 GB



Deployment mode in Spark

- Client mode
- Cluster mode