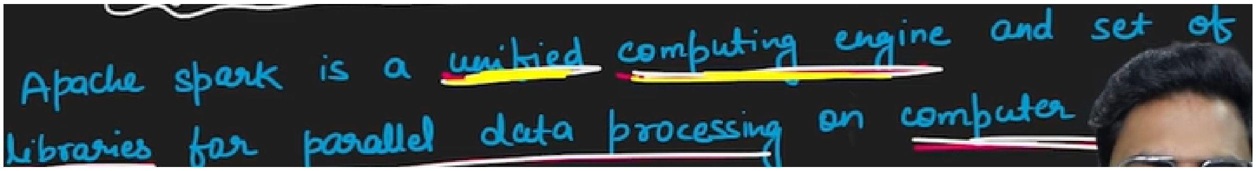
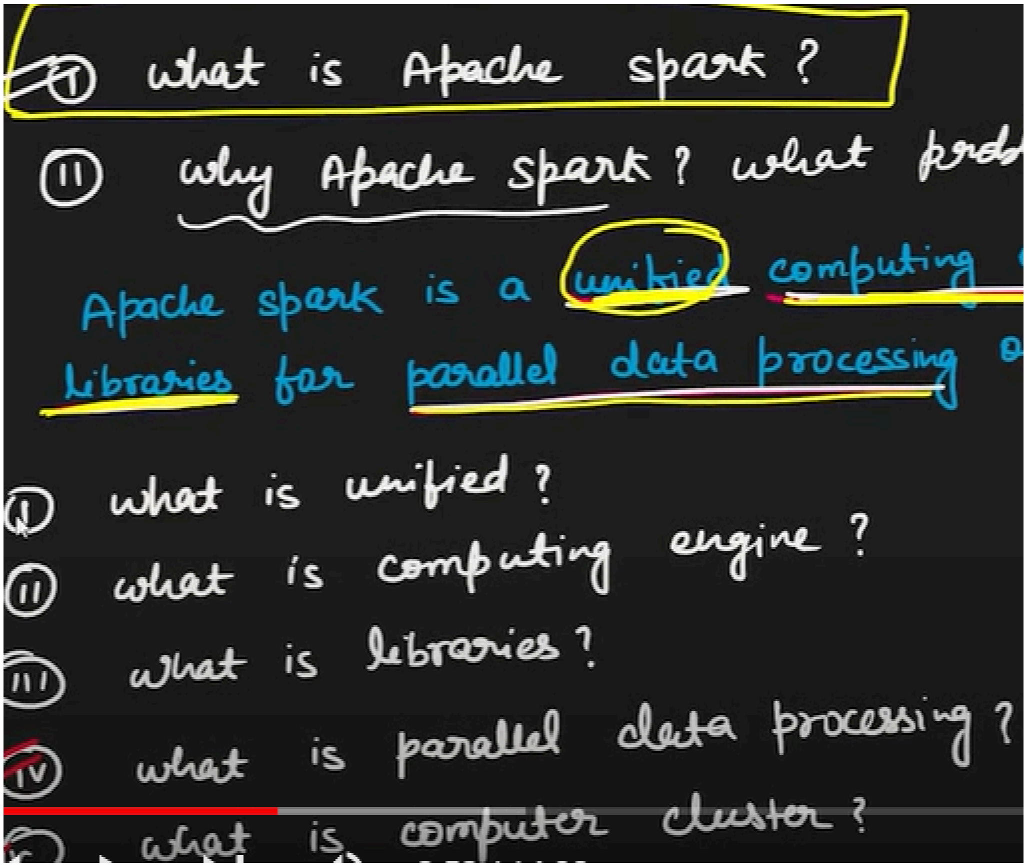
Spark Overview

1. What is a Apache spark?
2. 

   Apache spark is a unified computing engine and set of libraries for parallel data processing on computer

   on computer cluster
3. So we'll learn this terms now

   (I) what is Apache spark ?

   (II) why Apache spark ? what prob

   Apache spark is a unified computing

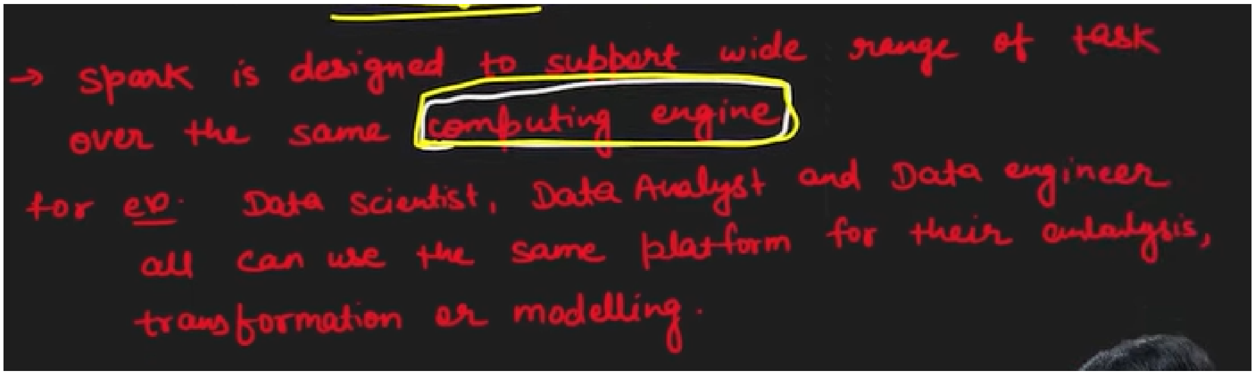
   libraries for parallel data processing o

   (i) what is unified ?

   (ii) what is computing engine ?

   (iii) what is libraries ?

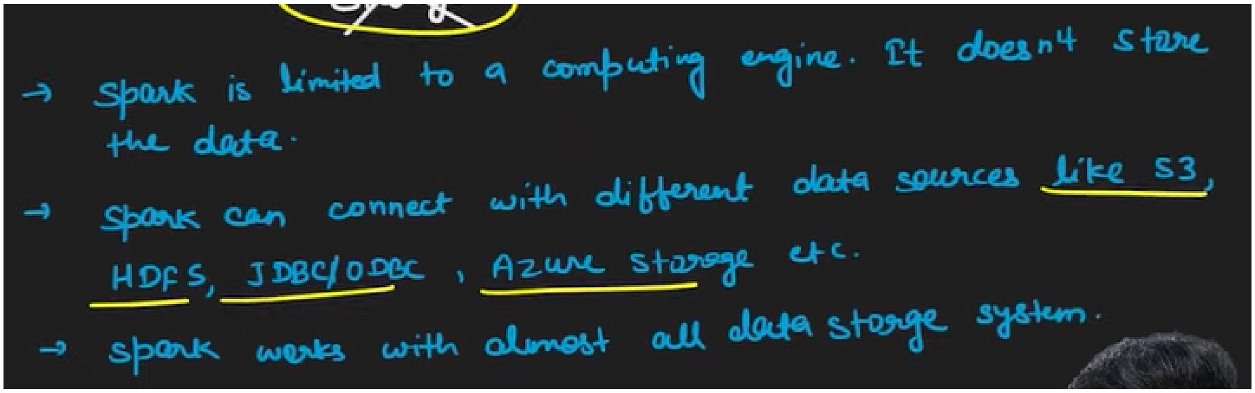
   (iv) what is parallel data processing ?

   (v) what is computer cluster ?
4. So what is unified? Spark is used by all(DA's,DE's) see below
5. 

   → spark is designed to support wide range of task over the same computing engine

   for eg. Data scientist, Data Analyst and Data engineer all can use the same platform for their analysis, transformation or modelling.
6. So what is a computing engine?

7. Spark never stores the data..but it gives flexibility of connecting with our storage services(Cloud, On Perm storage etc)
8. So spark is mainly used for computing…so spark has set of computer’s and ram which is used for computing purpose

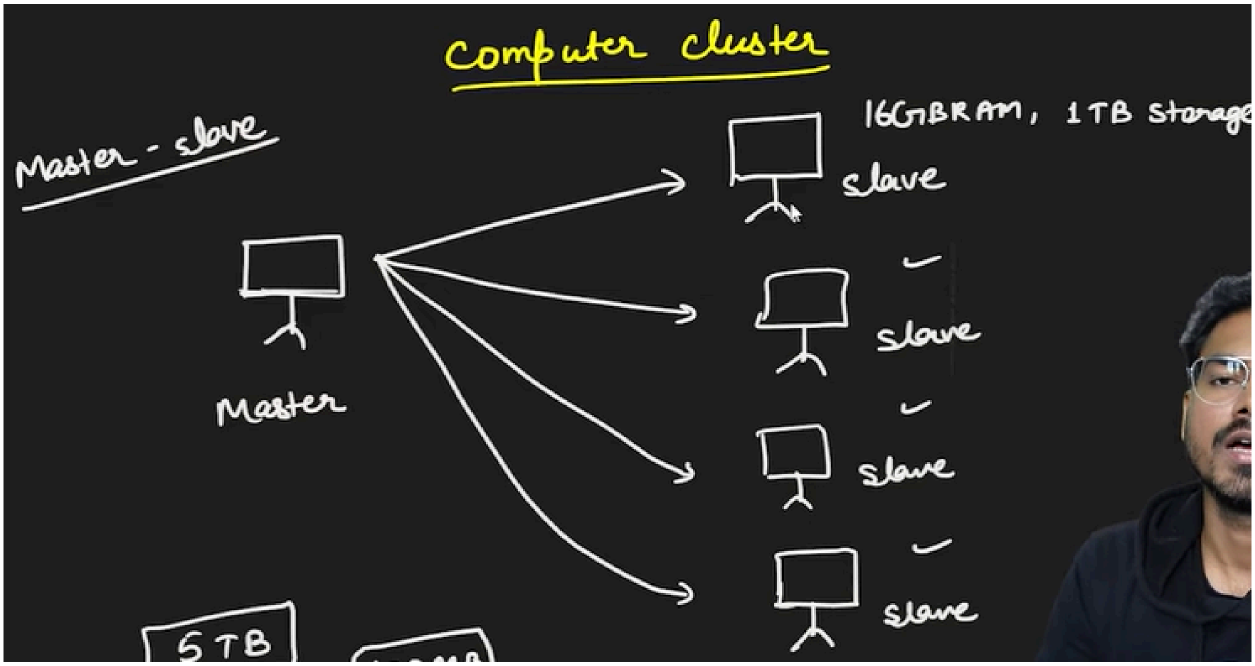


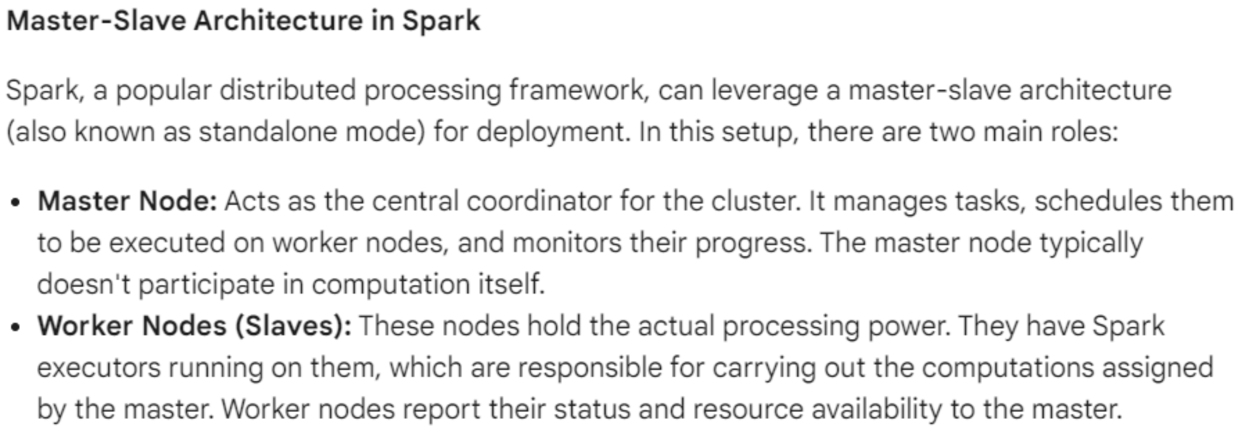
→ spark is limited to a computing engine. It doesn't store the data.

→ spark can connect with different data sources like S3, HDFS, JDBC/ODBC, Azure storage etc.

→ spark works with almost all data storge system.

9. Now what is computer cluster?

10. 



Master-Slave Architecture in Spark

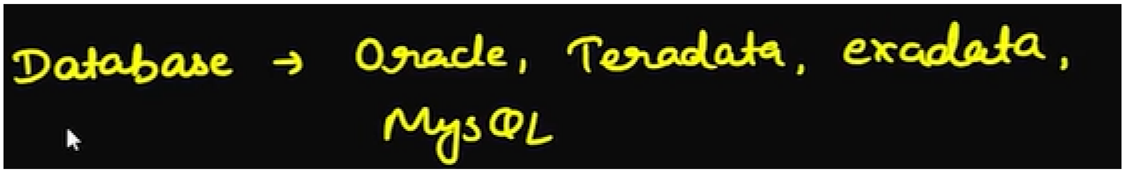
Spark, a popular distributed processing framework, can leverage a master-slave architecture (also known as standalone mode) for deployment. In this setup, there are two main roles:

- **Master Node:** Acts as the central coordinator for the cluster. It manages tasks, schedules them to be executed on worker nodes, and monitors their progress. The master node typically doesn't participate in computation itself.
- **Worker Nodes (Slaves):** These nodes hold the actual processing power. They have Spark executors running on them, which are responsible for carrying out the computations assigned by the master. Worker nodes report their status and resource availability to the master.

11. From this we've came known what actually spark does

Why Apache spark?

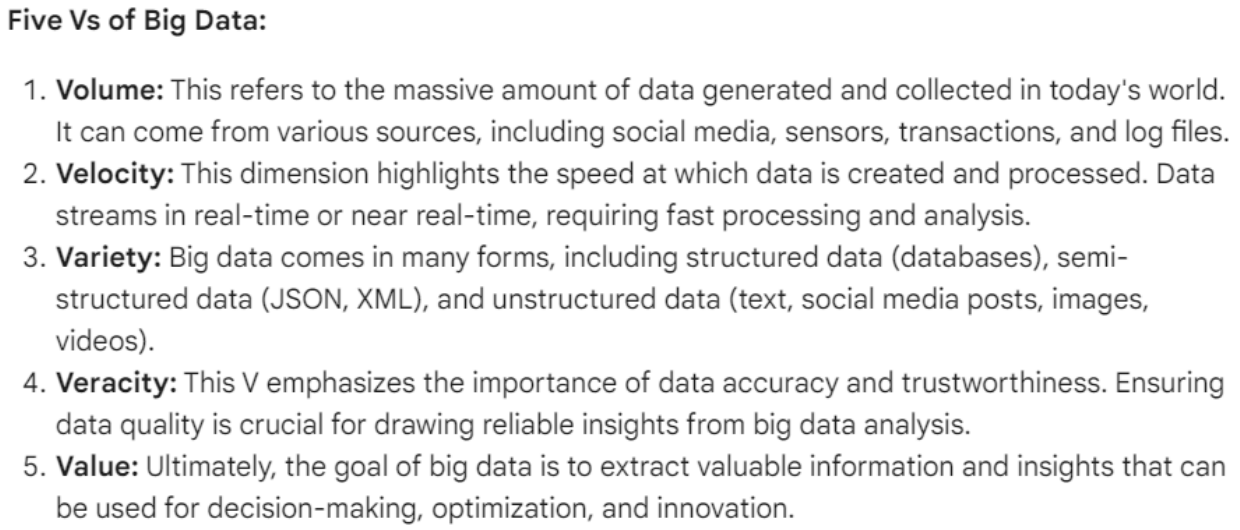
1. So initially we have multiple databases like



Database → Oracle, Teradata, exadata, MysQL

and they used to store only relational data.
2. But these days..we are getting data like text,CSV,Image and in many other formats
3. And also volume of data got increased

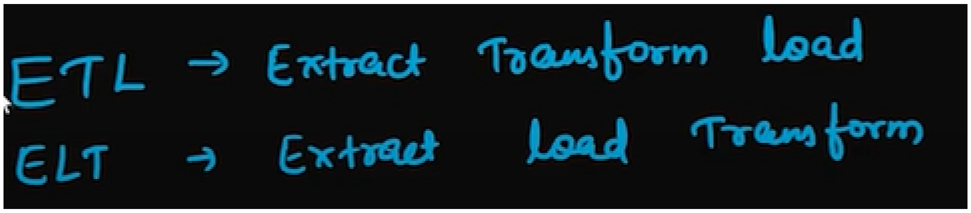
4. The five V’s of data



Five Vs of Big Data:

1. **Volume:** This refers to the massive amount of data generated and collected in today's world. It can come from various sources, including social media, sensors, transactions, and log files.
2. **Velocity:** This dimension highlights the speed at which data is created and processed. Data streams in real-time or near real-time, requiring fast processing and analysis.
3. **Variety:** Big data comes in many forms, including structured data (databases), semi-structured data (JSON, XML), and unstructured data (text, social media posts, images, videos).
4. **Veracity:** This V emphasizes the importance of data accuracy and trustworthiness. Ensuring data quality is crucial for drawing reliable insights from big data analysis.
5. **Value:** Ultimately, the goal of big data is to extract valuable information and insights that can be used for decision-making, optimization, and innovation.

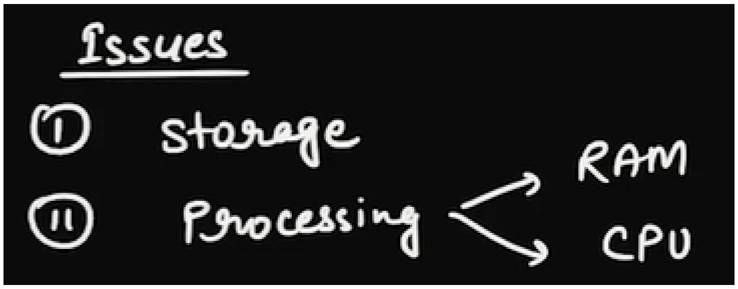
5. So one of the solution to handle this big data is spark
6. We used have ETL before there was data lake
7. Now as the data generated is very high..we are doing ELT



ETL → Extract Transform load

ELT → Extract load Transform

8. So now how do we store and transform this large volume of data?

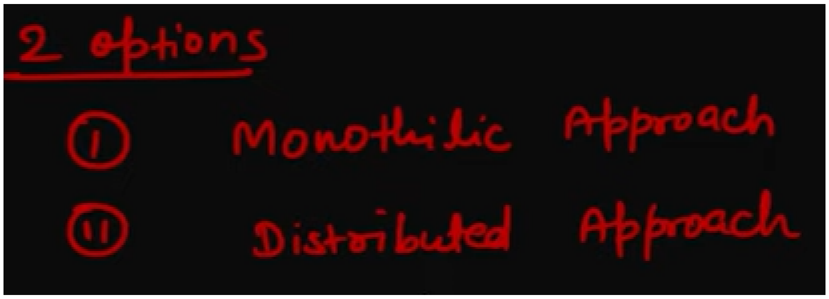


Issues

(i) storage

(ii) Processing

9. Issues
10. So we have 2 options to handle this data

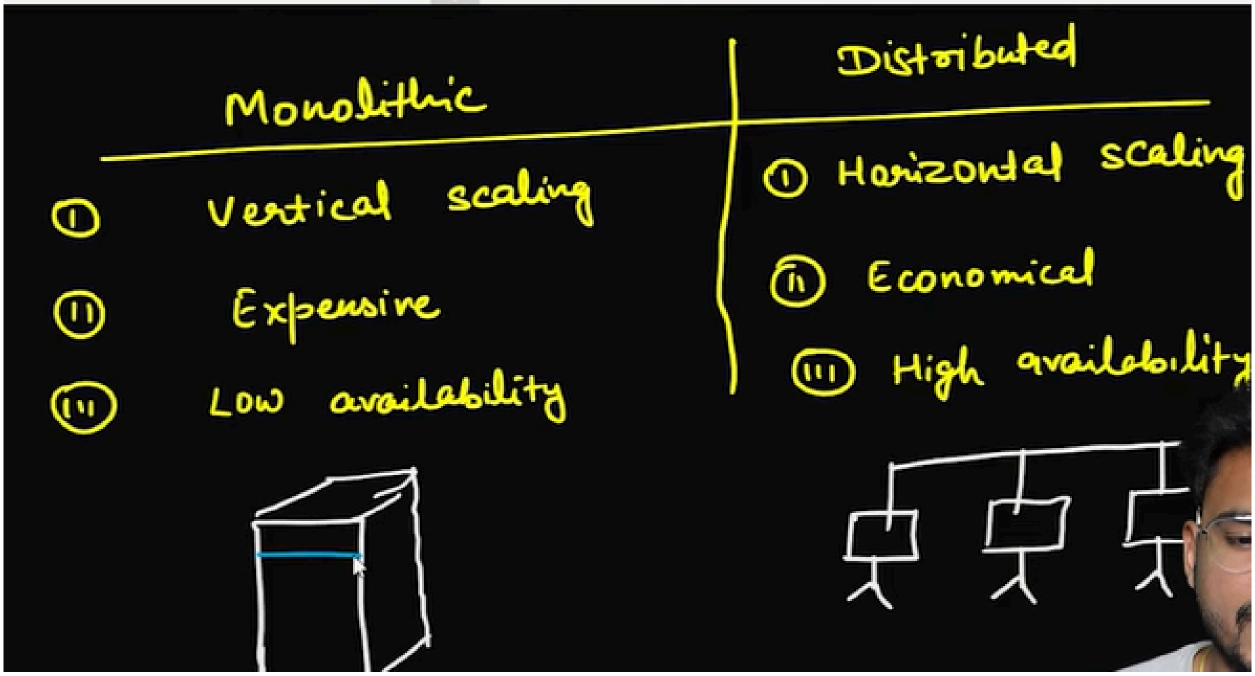


2 options

(i) Monothilic Approach

(ii) Distributed Approach

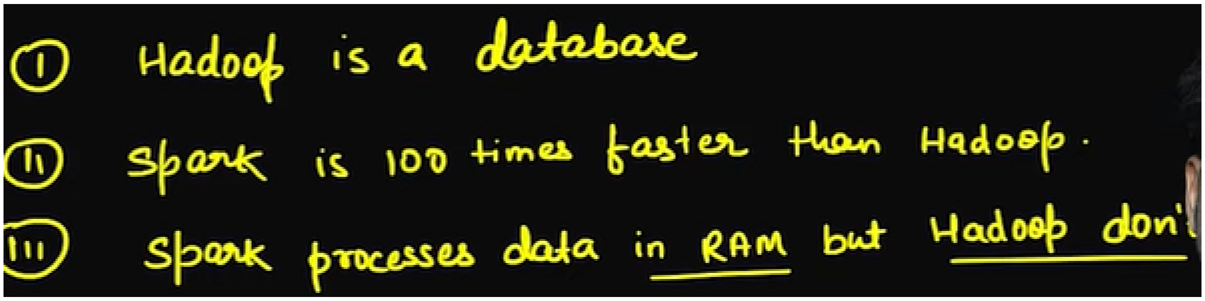
11. Monolithic approach is just increasing the capacity of one system(like increasing storage and ram)..but there will be a limit where we can scale this too

12. 

Monolithic	Distributed
(i) Vertical scaling	(i) Horizontal scaling
(ii) Expensive	(ii) Economical
(iii) Low availability	(iii) High availability

13. So first hadoop was came into scenario to solve this issue..but later spark took the throne

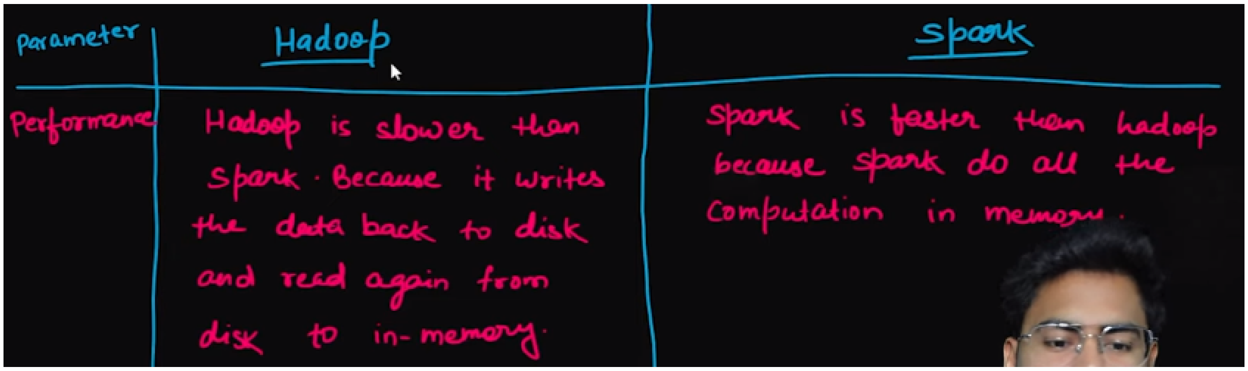
Hadoop vs Spark

1. First lets learn some common misconceptions
2. 

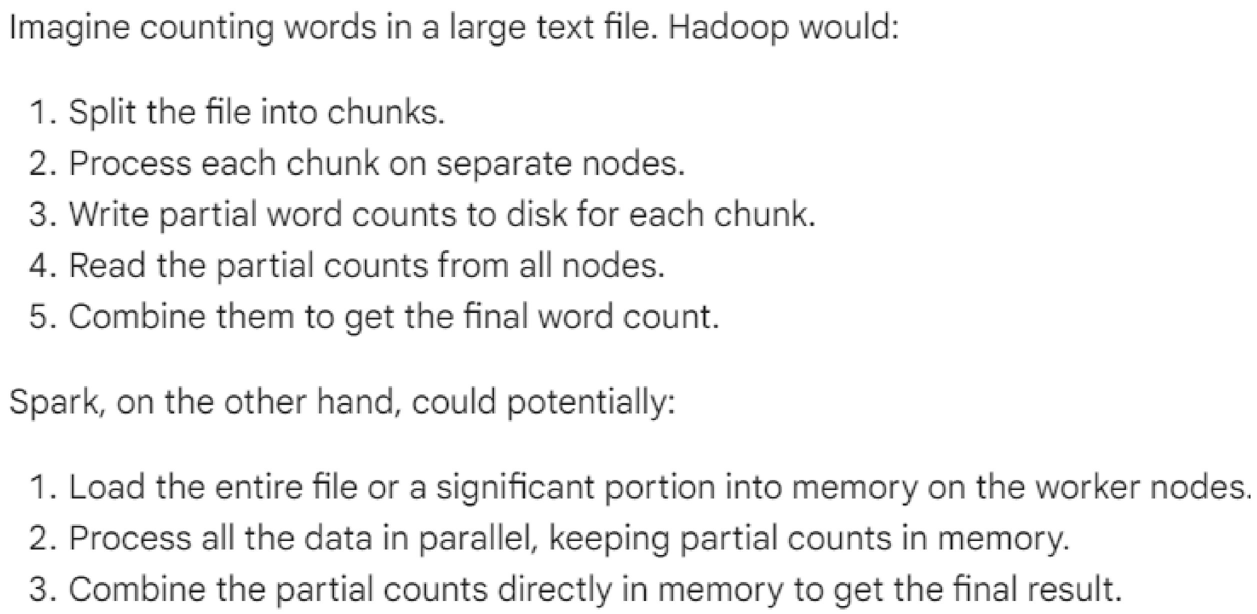
(i) Hadoop is a database

(ii) spark is 100 times faster than Hadoop.

(iii) spark processes data in RAM but Hadoop don'

3. Spark is upto 100 times faster than hadoop..but not 100 times
4. 

Parameter	Hadoop	Spark
Performance	Hadoop is slower than spark. Because it writes the data back to disk and read again from disk to in-memory.	Spark is faster than hadoop because spark do all the computation in memory.



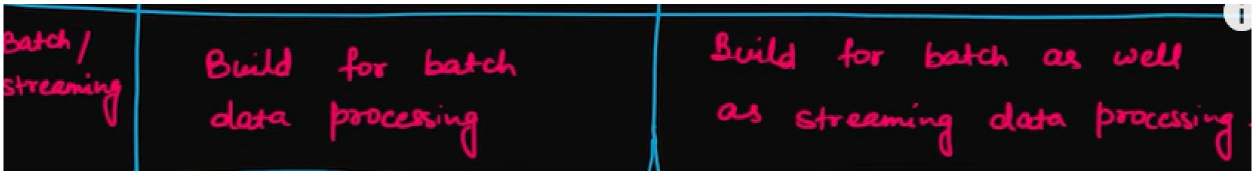
Imagine counting words in a large text file. Hadoop would:

1. Split the file into chunks.
2. Process each chunk on separate nodes.
3. Write partial word counts to disk for each chunk.
4. Read the partial counts from all nodes.
5. Combine them to get the final word count.

Spark, on the other hand, could potentially:

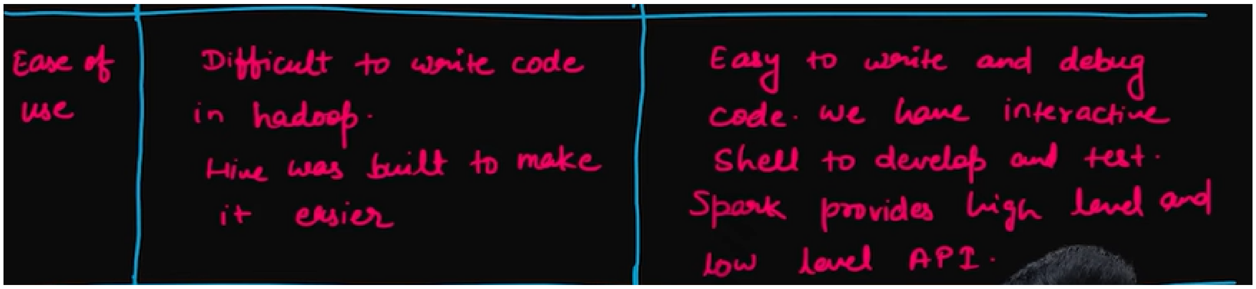
1. Load the entire file or a significant portion into memory on the worker nodes.
2. Process all the data in parallel, keeping partial counts in memory.
3. Combine the partial counts directly in memory to get the final result.

5.
6. The next diff is batch and streaming



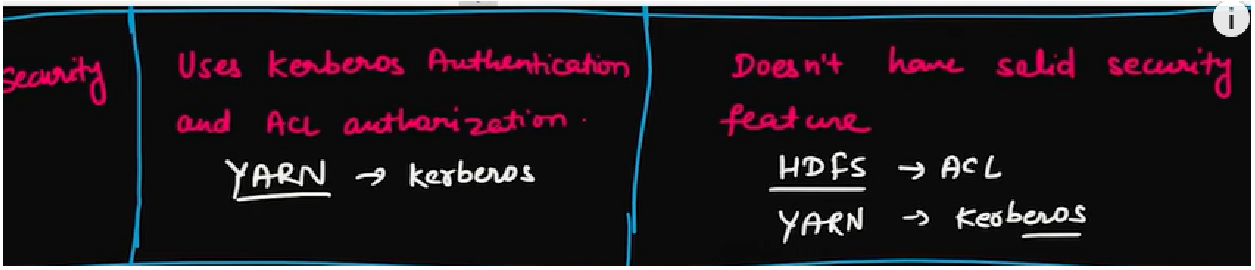
Batch / streaming	Build for batch data processing	Build for batch as well as streaming data processing.

7. Ease of use

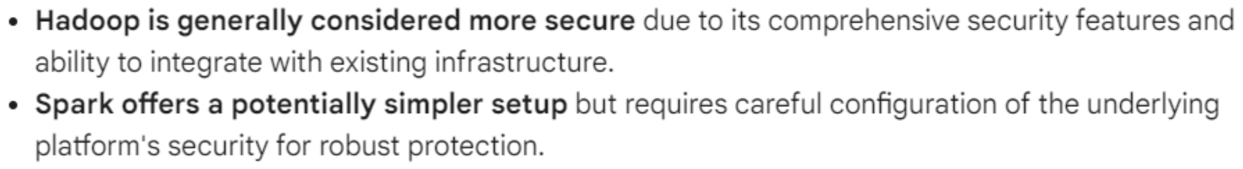


Ease of use	Difficult to write code in hadoop. Hive was built to make it easier	Easy to write and debug code. we have interactive Shell to develop and test. Spark provides high level and low level API.

8. security



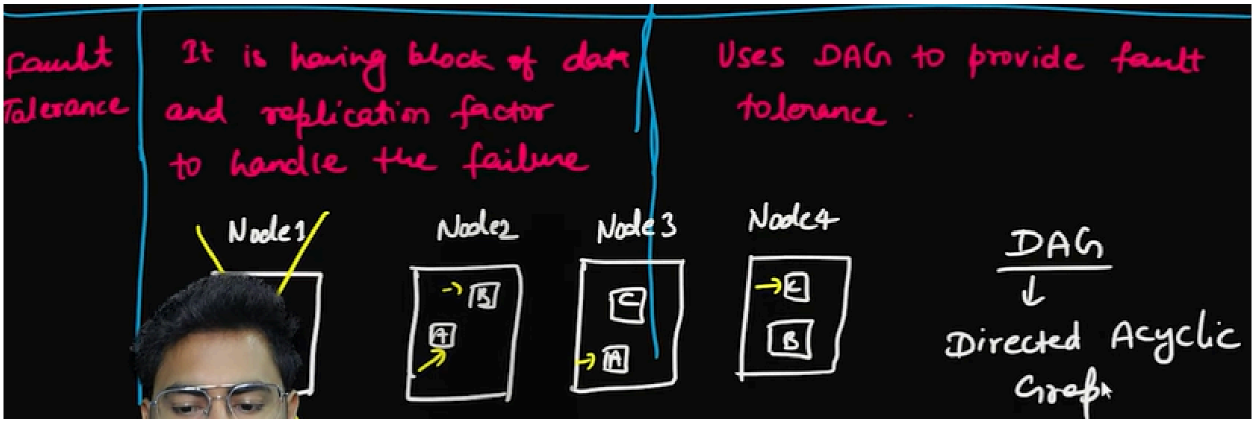
Security	Uses kerberos Authentication and ACL authorization. YARN → kerberos	Doesn't have solid security feature HDFS → ACL YARN → Kerberos



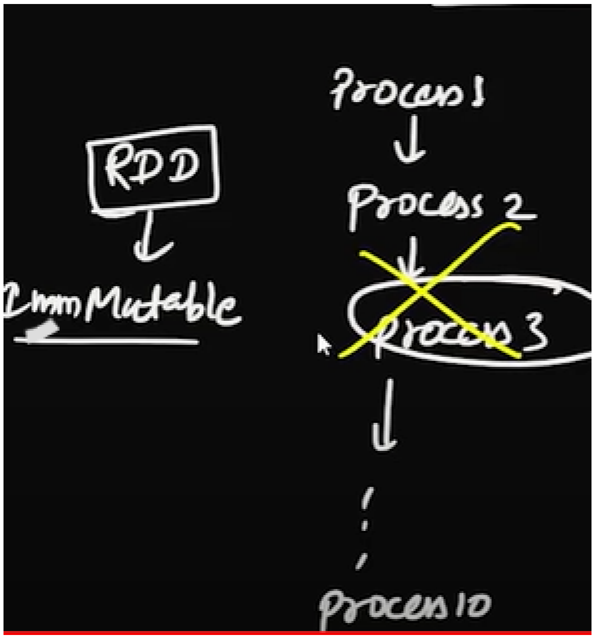
- **Hadoop is generally considered more secure** due to its comprehensive security features and ability to integrate with existing infrastructure.
- **Spark offers a potentially simpler setup** but requires careful configuration of the underlying platform's security for robust protection.

9.

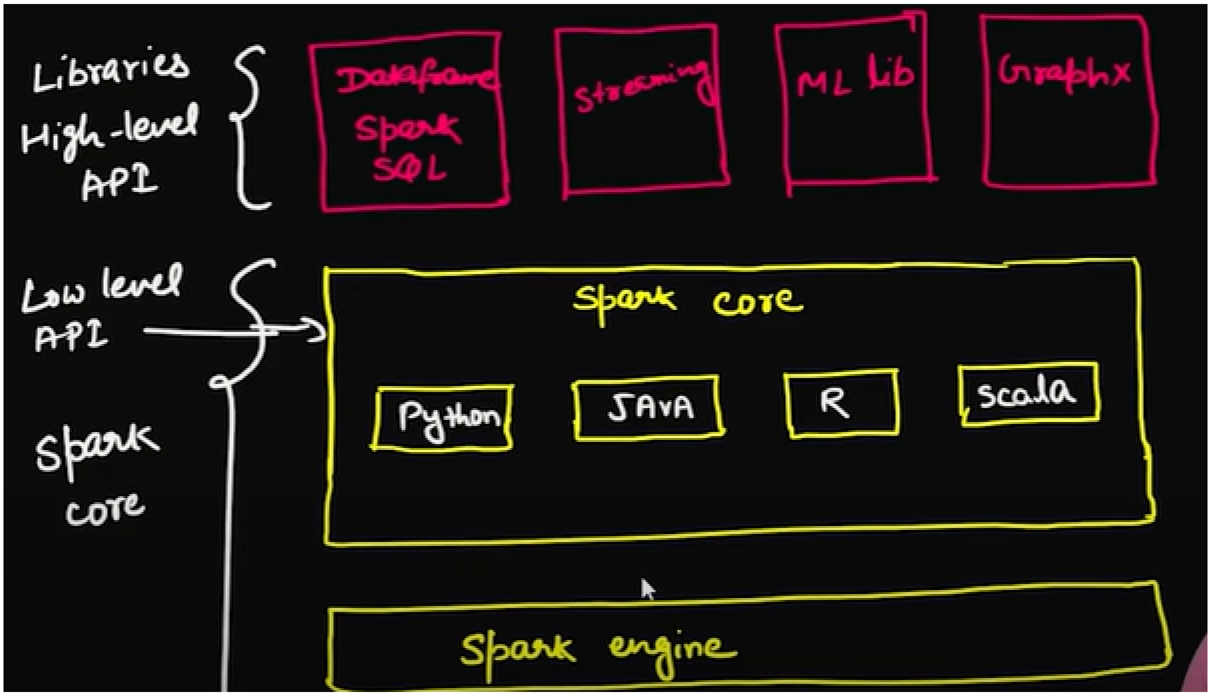
10. Fault tolerance

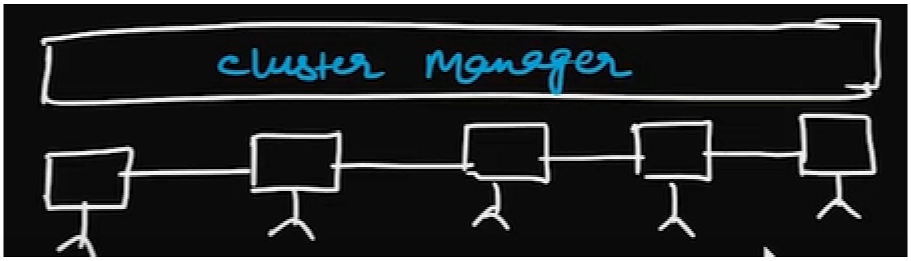


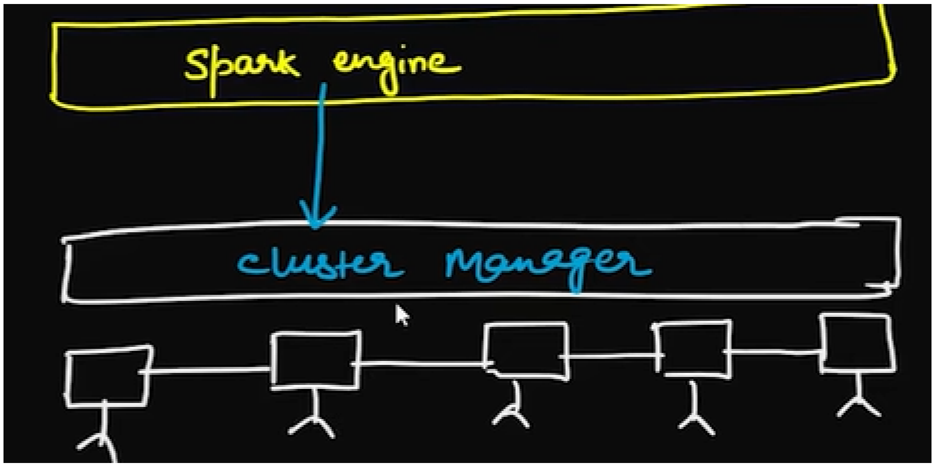
Fault Tolerance	It is having block of data and replication factor to handle the failure	Uses DAG to provide fault tolerance.



Spark EcoSystem

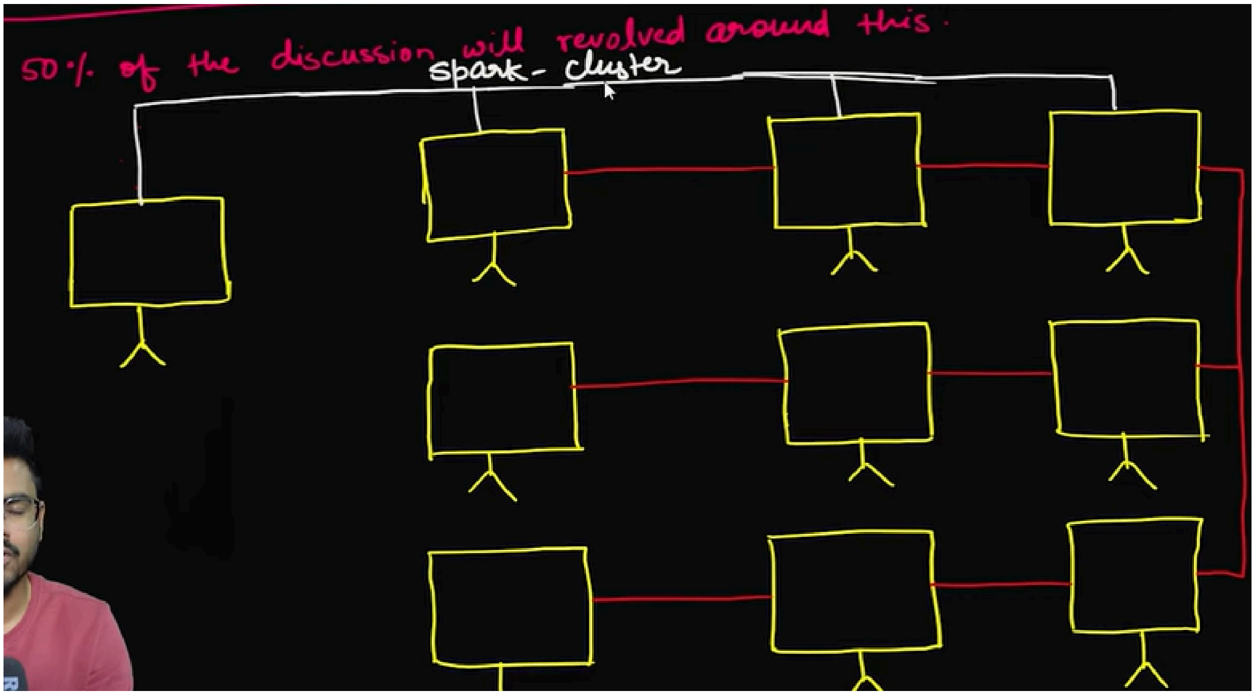
1. 

2. In high level API ..spark will be creating dataframes and datasets
3. In low level API ..it creates RDD’s
4. Eventually..the code which is written in highlevel API..will comes to low-level API and to spark engine
5. Similarly if we write low_level..then it comes to spark engine
6. 

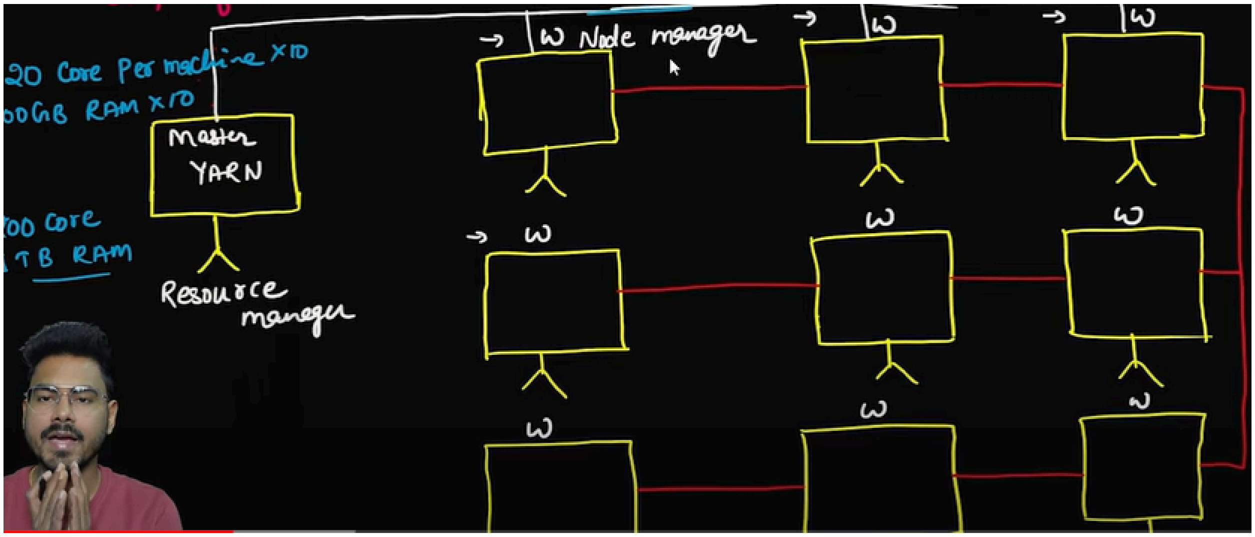
7. Here we have cluster manager ..which is used to assign resources needed for spark engine
8. 

9. Here we have clusters managers like YARN,MESOS,Kubernetes etc

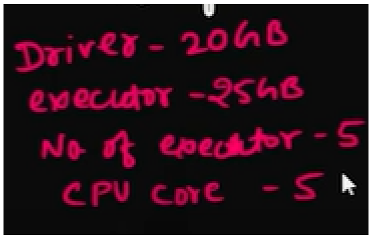
Spark Architecture

1. 50% of spark interview will revolve around this
2. 

3. Here we have a cluster setup of 20 core and 100GB RAM per machine ..so in total we have 200 core and 1 TB of storage in this cluster setup
4. The left most one is master node and 9 other nodes are workers nodes
5. In the masters node ..YARN will be pre installed ..which is used to manage the resources
6. And for worker nodes..there will be node manager

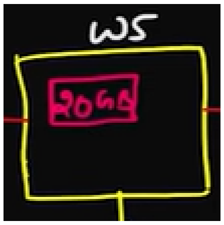


7. Now lets imagine a developer has written a spark code and wants to execute it and to execute it ..he needs this resources

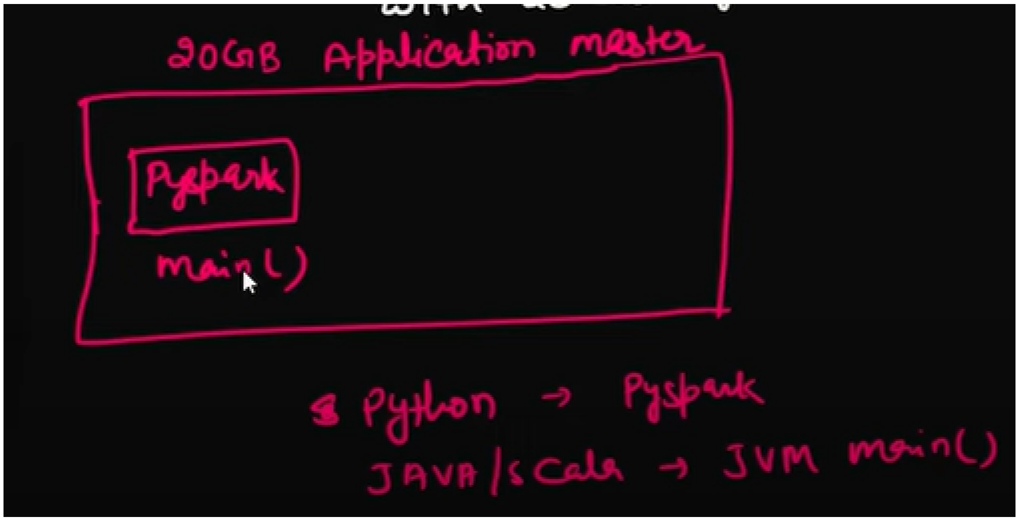


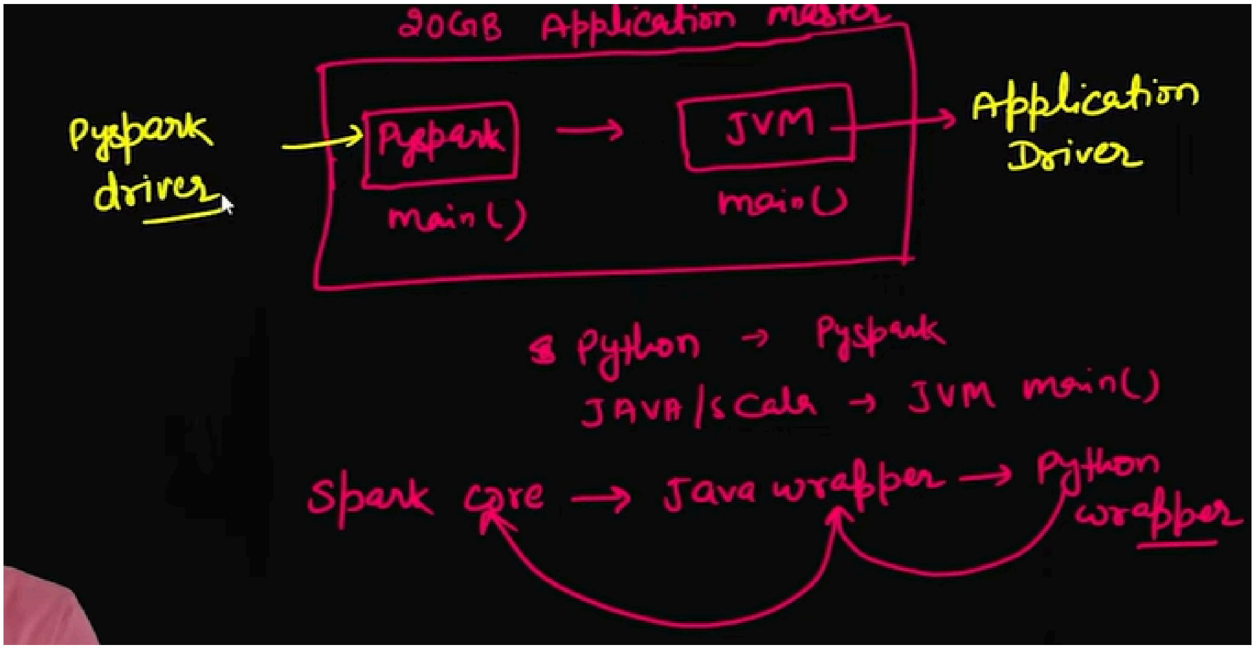
Driver - 20GB
executor - 25GB
No of executor - 5
CPU core - 5

8. So with these resources developer will approach the resource manager(YARN)
9. Now YARN will randomly picks worker node and requests this worker node to create 20gb of RAM

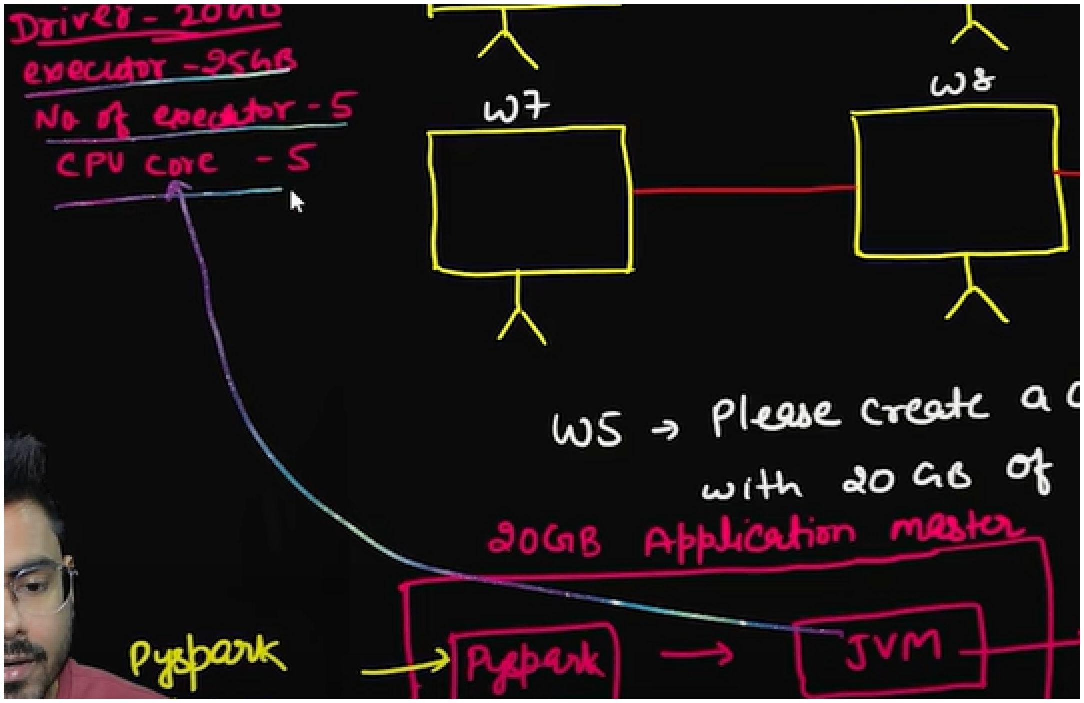


10. Now lets go inside of w5(container)..now this container/worknode will be called as application master



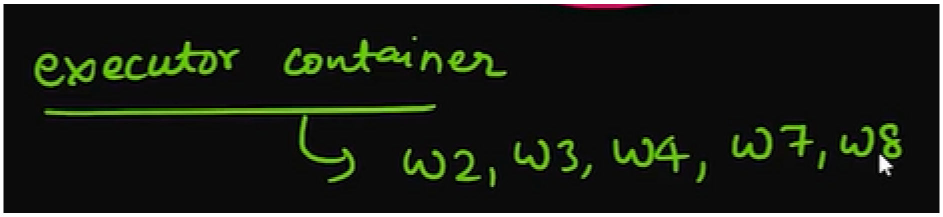
11. 

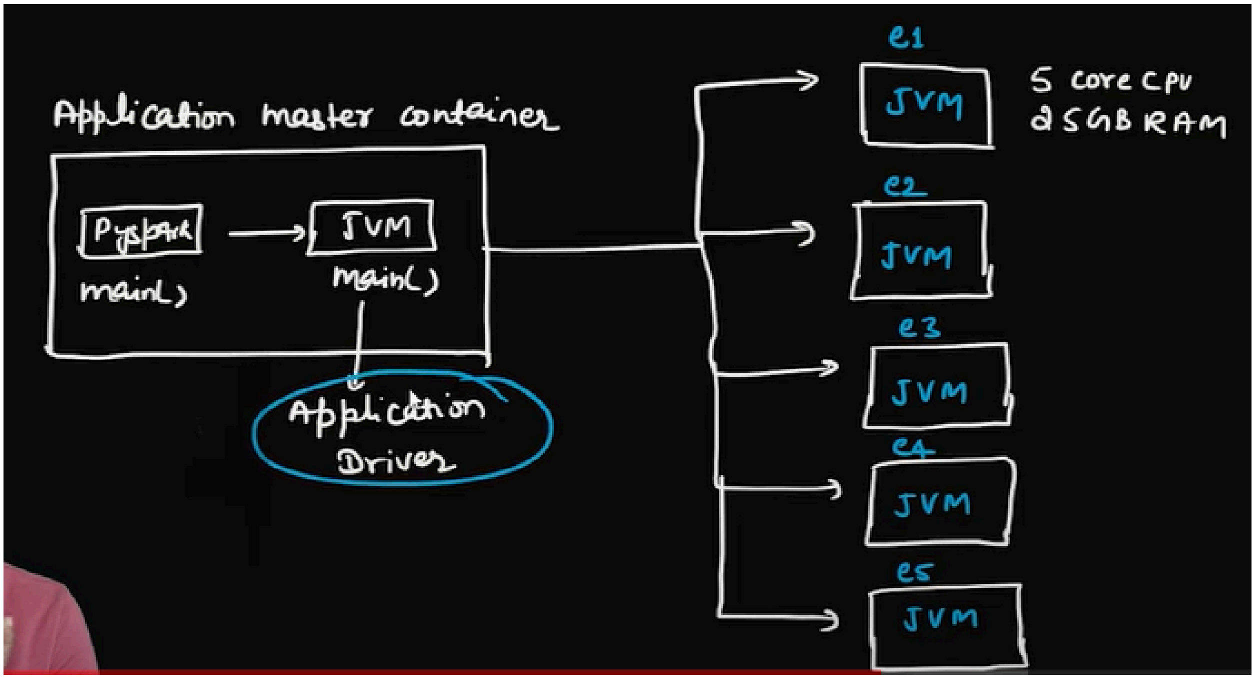
12. Now after successfully creating the JVM…now JVM goes back to the requiremtns

13. 

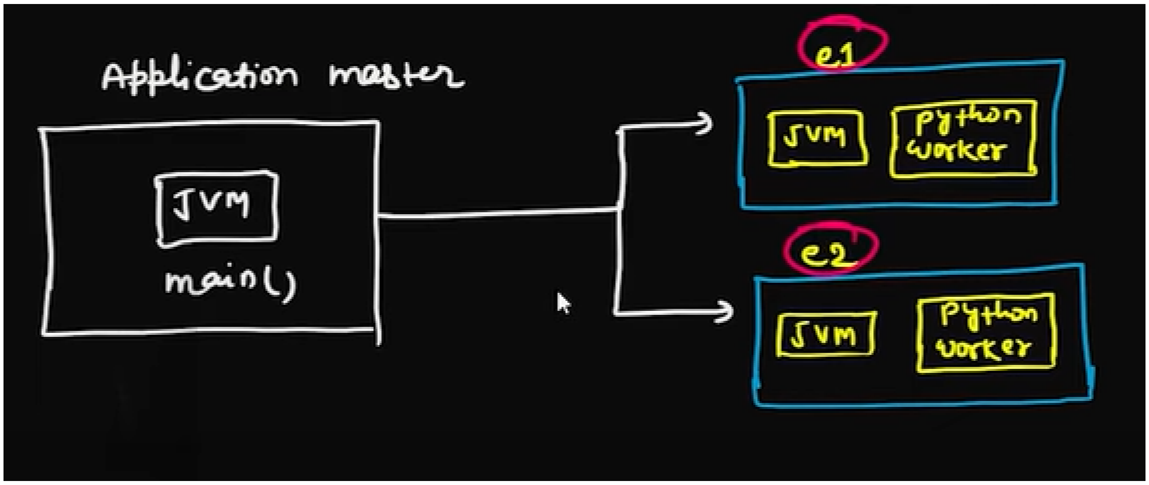
14. Now with this request..we will go back to resource manager
15. Now as it needs 5 executors…the YARN will assign 5 worker nodes

16. We have executors containers..which are given by YARN as we requested



17. 

18. Now what if write a python UDF? Our executors will not understand that
19. Lets take a small example and understand



20. So if want to execute a UDF in executors ..then we need to have python worker inside the executor
21. But using python worker inside a executore will degrade the performance of executor
22. Inshort explanation start from @18:00 in the video