pyspark

* components :-

1. Spark SQL
2. Spark Streaming
3. Spark MLIB
4. Spark Graphx
5. Spark R
6. Spark core

* what is Apache Spark

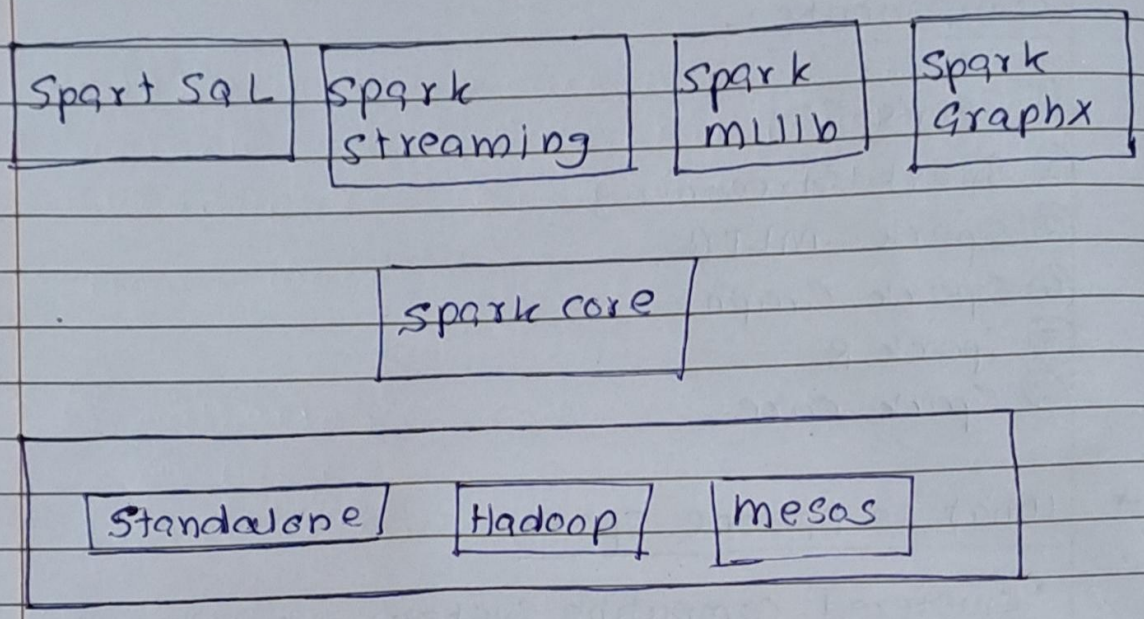
- clustered computing system
- provide high level API
- Support general execution graph
- High level tools for structured data processing
- can run sperate or with exising cluster manger.

Note → cluster computing system
↓
Group of virtual machines

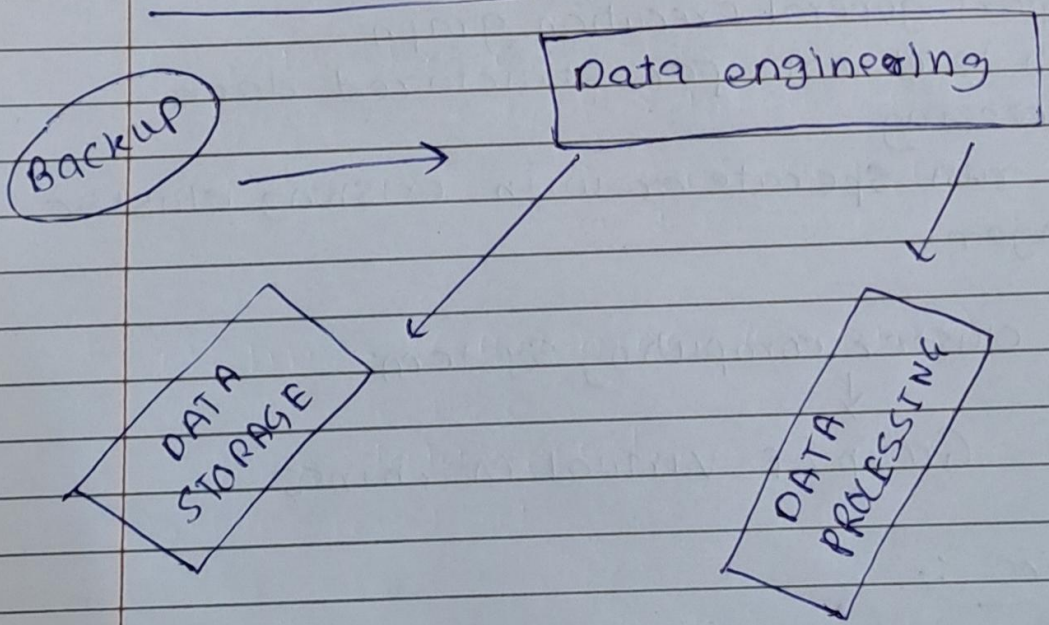
* features :-

- Developed using Scala language
- Runs on JVM
- API → Scala, Java, python
- Shell → Scala, python
- Datasources → SQL, NO-SQL, Local files.

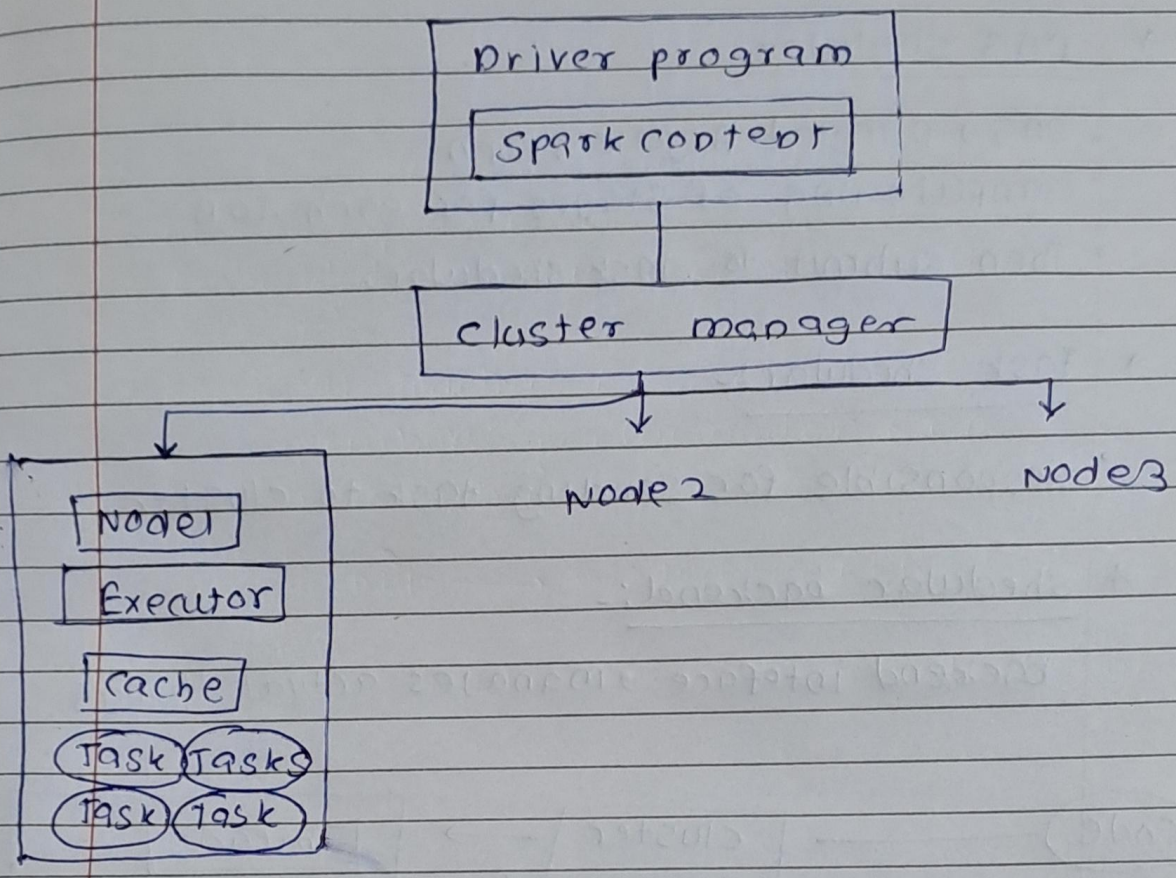
* Component diagram:-



* data E. concept :-



Parallel and distributed



- JOB
- stages → map, reduce
- DAG → Directed Acyclic graph
- Executor

* Spark content :-

- connection to spark cluster
- can be use to create RDD.
- accumulator and brodcast variables on that cluster

PAGE NO.:

* Streaming :-

- Add on to spark core API which allows scalable, high throughput fault tolerant stream processing of live data stream.

How it works ?

(a) Gathering :-
  (1) Basic
  (2) Advance.

(b) processing :-
  Gathered data processed.

(c) Data storage :-
  processed data → database
  pstream → continuous flow of data

* MLlib :- (machine learning)

- High data speed.
- High level algo's

PAGE NO.:

* Spark core

- All functionalities are build on top of Apache Spark core.
- Embedded with RDD (Resilient distributed dataset)
- RDD is abstraction of spark.
- two operations performed on RDD.
  (a) Transformation :-

      Old RDD ⟶ Create New RDD

  (b) Action :-
      use to work with Actual dataset.

* Spark SQL :-

- Act as SQL query engine.
- works with structured and semistructured data.
- Also use to access streaming and historical data.
- use for structured data processing.

* features :-

1. cost Based optimizer.
2. mid query fault tolerance.
3. compitible with hive.
4. Dataframe + SQL ⟶ way to access data.

* DAG shedular:-

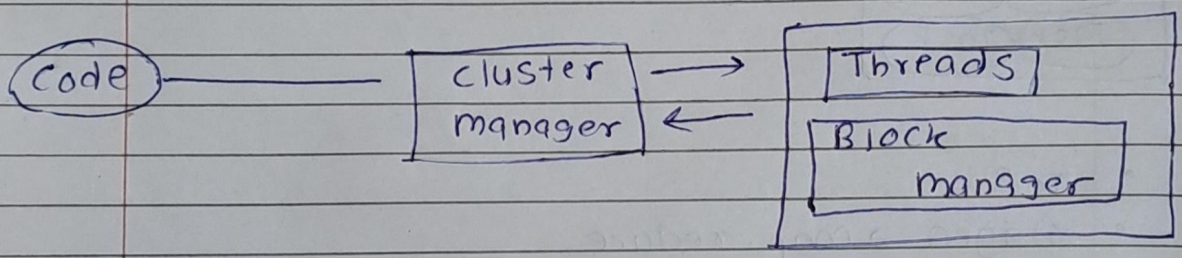
- DAG → Directed Acyclic Graph
- Compute DAG of stages for each JOB
- Then submit to task shedular.

* Task shedular:-

- Responsible for sending task to cluster.

* Shedular backend:-

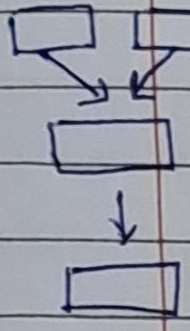
Backend inteface, manages actual things



* How it work?

- Small code base.
- divide in various layers
- 1st layer :- interpreter
  ↳ Scala interpreter.
- As we enter our code in spark console spark create operator graph.
- Graph submitted to DAG shedular
- DAG's optimised the graphs.
- then task manager send to stash to Cluster.

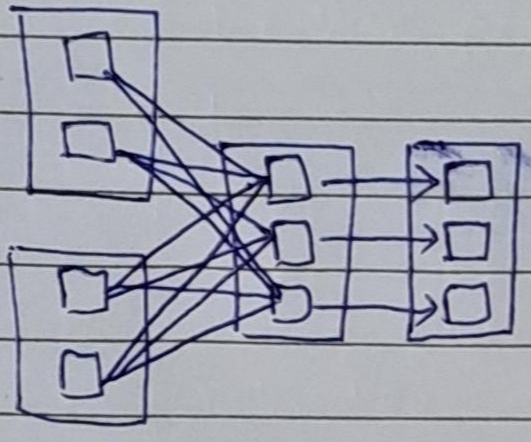
RDD obj



build operator

DAG

DAG Shedular



Graph splitted into stages of tasks

Task Shedular

Cluster manager

task

worker