Advance Database management

Week-2 (class-2)

Continuation from first ppt

- Bigdata - is the main focus of this course.

Week-3 will be more Hadoop.

Highly scalable is mostly consist of commodity Hardware

SQL - two Application OLAP, OLTP

Big data Application

OLTP - strictly follows ACID properties

but for Big data we usually use OLAP as ACID is not that important

rather performance is important

Big Data Technologies

- Goals
  - Support of non-structural or semi-structural data
  - Highly Scalable
- Two major technologies
  - Hadoop ecosystem
  - NoSQL DBMS
- Support of
  - *Web-based Real-time Application*
  - *OLAP*
  - *Online Streaming Processing*

Historical data - This data is processed in batches

Online streaming process - is Basically real-time OLAP

Hadoop Ecosystem

Hadoop Ecosystem
- History -

- Hadoop Distributed File System/MapReduce
  - Initially developed at Google for a project named Nutch (2003) to crawl and index millions o web pages
  - The *storage engine* and *processing* parts which Nutch packaged as the core of **Hadoop** product at Yahoo
  - First Apache open-source release in 2006
- Pig
  - Initially developed at Yahoo
  - Component built on top of MapReduce
  - Provides applications with SQL-like interface to access data in HDFS
  - First Apache open-source release in 2008
- Hive
  - Initially developed at Facebook
  - Component built on top of MapReduce
  - Provides applications with SQL-like interface, DDL, JDBC/ODBC inteface to access data in HDFS
  - First Apache open-source release in 2010

Map Reduce.

dies

Hadoop developed at google then it was open sourced

Hadoop distributed file system (HDFS) = JAVA pro

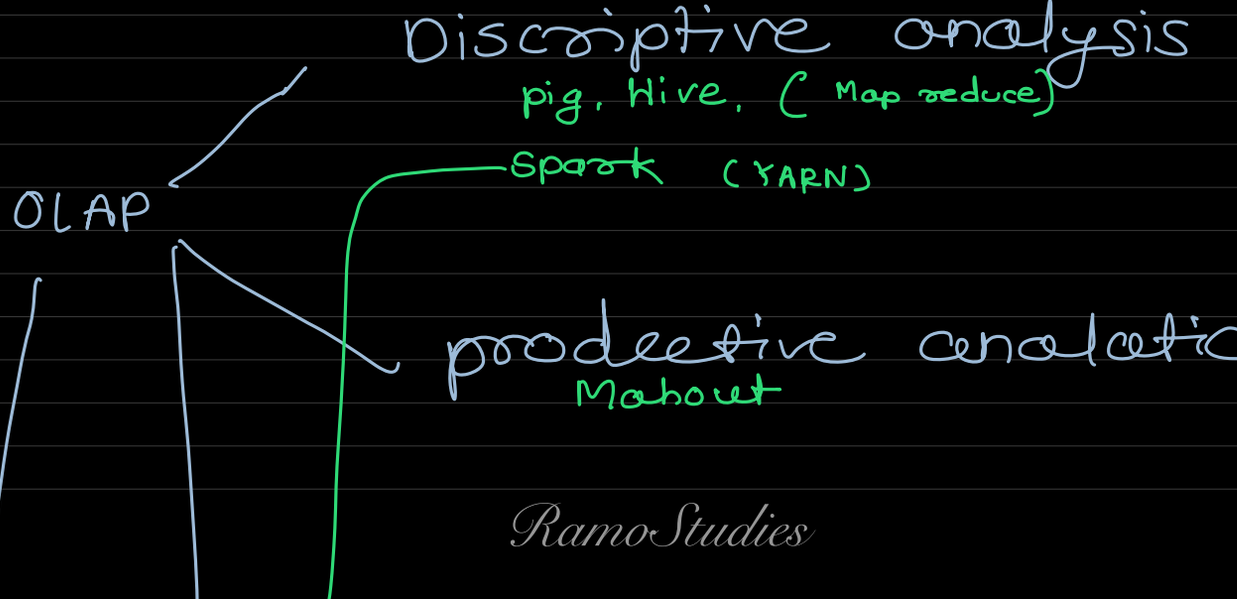
In case you don't wanna use Java then we can use SQL using pig & Hive.

YARN - Yet another resource negotiator

Generic distributed processing distribution.

With YARN - you can build Application on top that's another use of YARN with neglected Map redcue wich speed up the process

So YARN is key



Unified Engine
it's product it can do
discriptive analysis, predictive analysis
& also, online streaming process

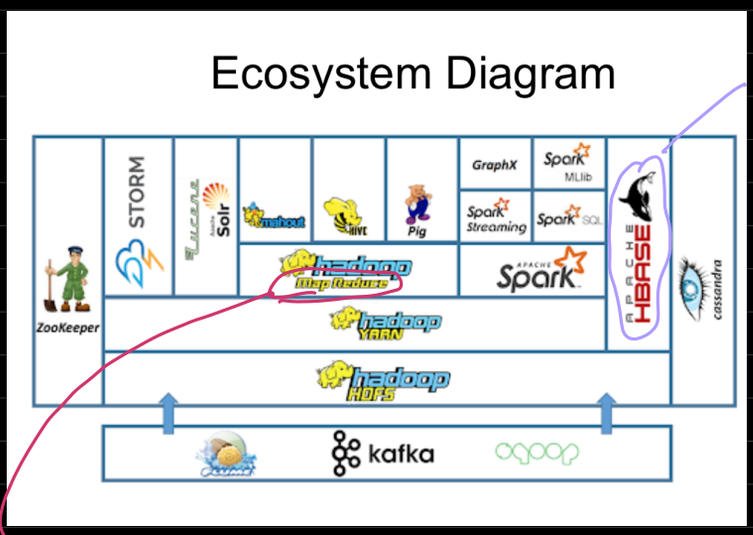
Online streaming processing

storm
Spark → YARN

Web-real time Application

HBase- NOSQL DBMS.

One of the focus of class
Map reduce



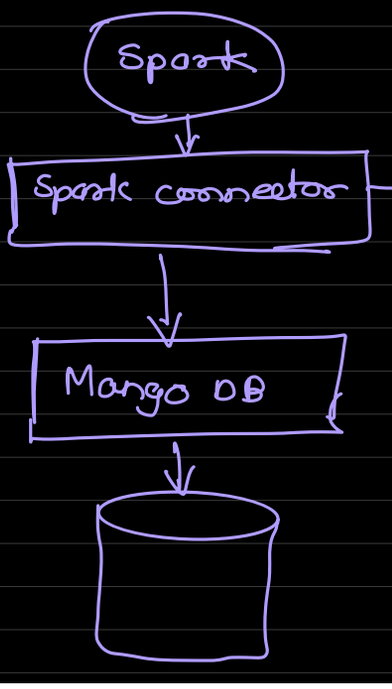
Its itself is DBMS

According to book mapreduce should be below YARN

What's the diff bet' DBMS & file system

logical data view
(doesn't care where file is stored)
High level view to manage data

physical location

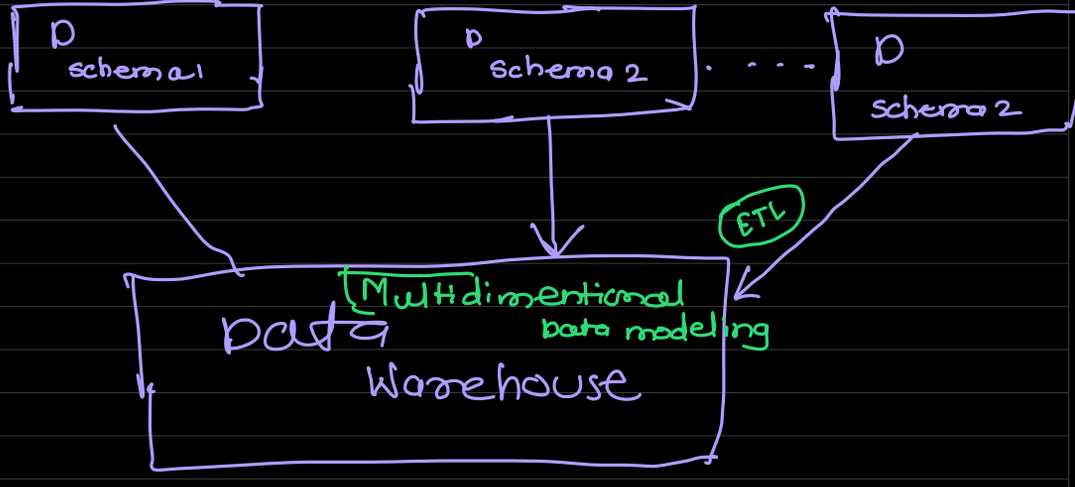


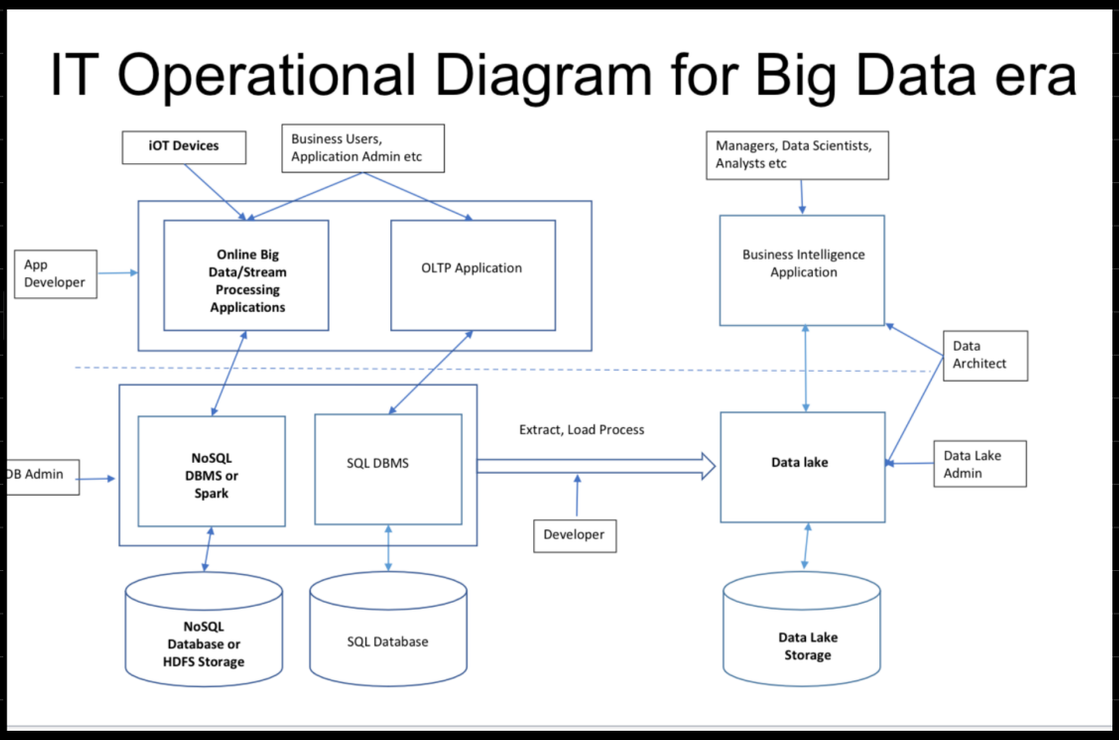
Spark uses connector to connect with mongoDB.

New Big Data Concepts

On-line streaming processing
- Real-time OLAP

Data Lake (vs Data Warehouse)
- Data warehouse store data in a multidimensional model
- Data lake store data as it is — need thing





final data - project | Data lake
NOSQL
Big data

Course Roadmap

- Distributed Data Processing
- Hadoop Ecosystem
  - Hadoop Distributed File System
  - YARN
  - MapReduce Programming Model
  - Spark
  - HBase
- NoSQL DBMS
  - Data Sharding
  - Data Availability
  - Data Consistency
  - Data Backup & Recovery

chapter - 3
Data processing

1. Multiple threading
Execution time = $T_p \times N$ Improving Execution time

with multithreading ex. 20
$= \frac{T_p \times N}{20}$

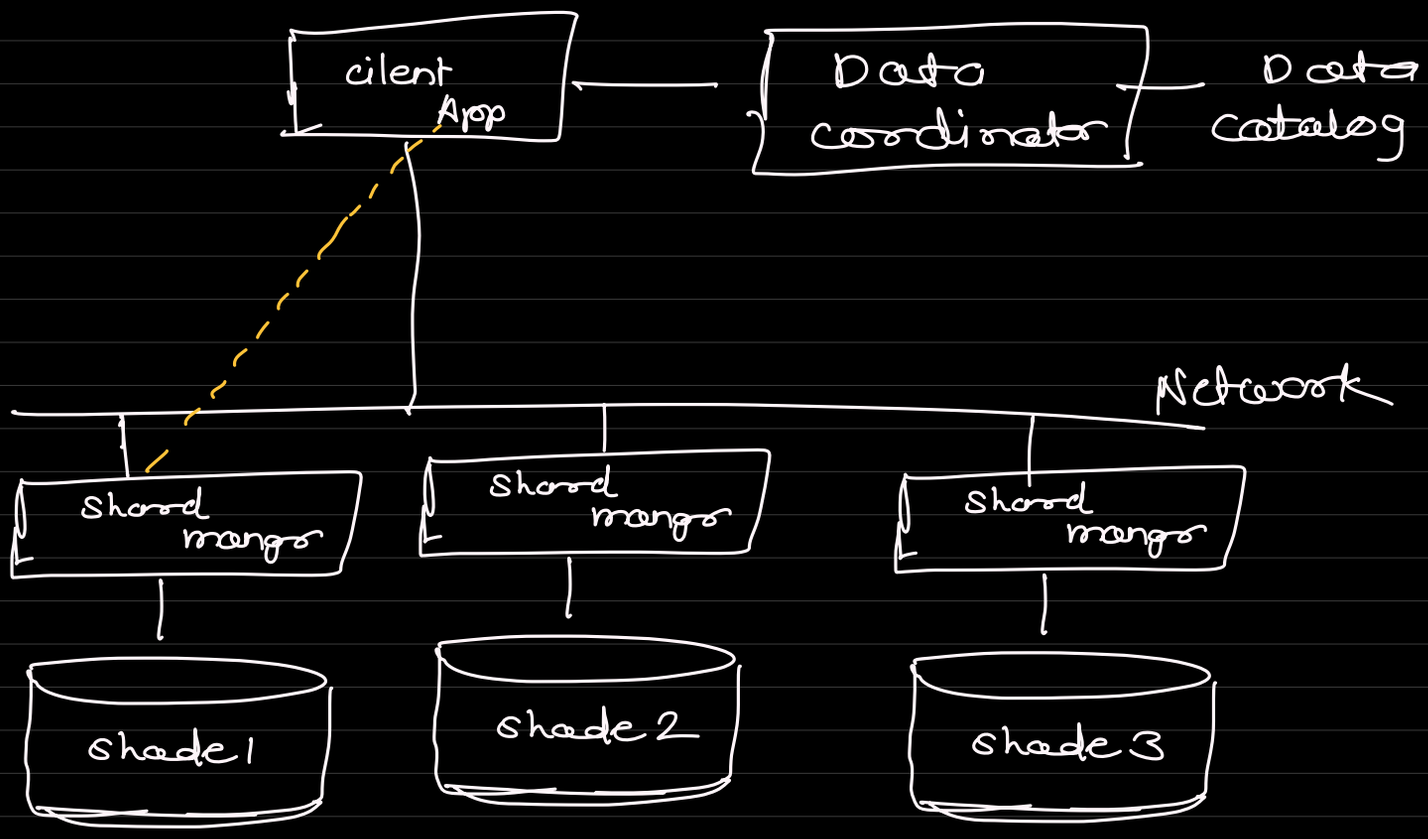
so we significantly reduce Execution time.

2. Multiple process multiple processors

3. Horizontal scaling. is more cost effective than vertical scaling

▲ Data sharding — Search more implement parallism into this
send data into blocks to diff machines

more group = more parallism



Distributed Algorithms
ex.
- Hadoop Mapreduce
- distributed query processing - send query to diff systems
- distributed ML algo. — spark already implemented it (i think)

framework for executing distributed algo
- Generic - Hadoop, Mesos

Distributed job processing

Task scheduling aka. DAG =

Distributed job should know following
- where job is located
- DAG
- Resource requirments to run each task

IPC - interprocess communication

Data locality ? = How close the data is to your task? = distance from data.

How much cost the overhead have to get to the data
if overhead is higher then locality is low

medium
2. if data is on diff machine data overhead is Higher.
3. if task is on another machine & data on another machine then that's the highest locality.

worst

Data Avability.

- usage of redudancy & proper failover will achieve high availability.

Disk array - Backup disk

- so when using shards to maintain the high availibility we keep backup of that data to another disk

---

Map reduce Application
Spark Application

Arch & implementation of Hadoop Ecosystem.

final project

2 Member team.