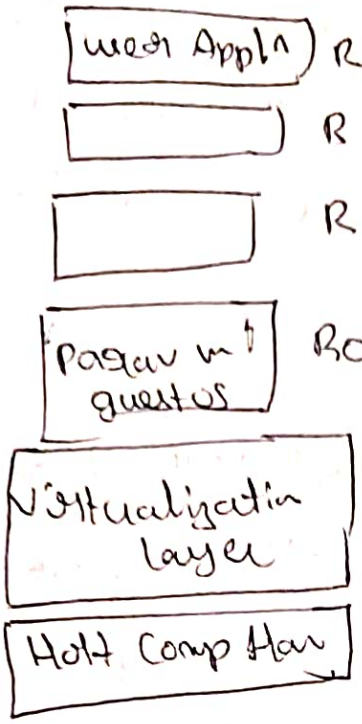
06/09/2024

Cloud Computing :-

→ Virtualization :- . . . (Continuation).

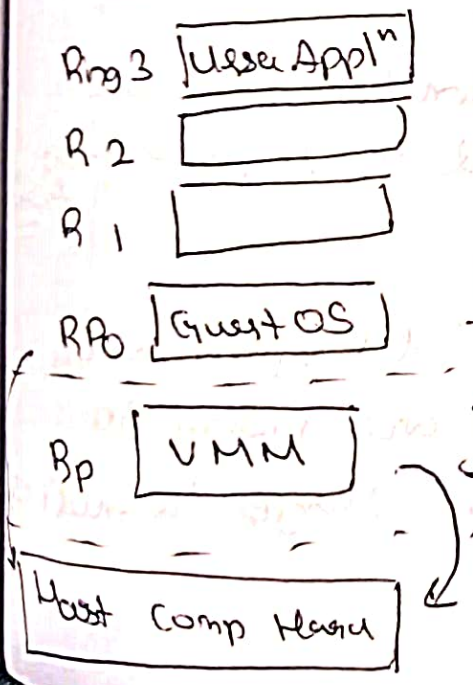
Tech 02 :- OS-Assisted Virtualization (para Virtualization)

- modification of guest OS Happens



* Para Virtualization involves modifying the OS kernel to replace non-virtualizable instructions with hypercalls that communicate directly with hypervisor using hypercall interface.
* The performance is improved by establishing a comm. b/w the guest OS and the hypervisor.

Tech 03 :- Hardware Assisted Virtualization.



* Intel implement virtualization by a new mode for -that Allows VMM to run in a new mode below "ring 0_R" privileges. and guest OS run in "ring 0" referred to as "ring 0_D" called as de-privileged mode

* Privileged Instructions Attract Automatically to the Hypervisor and is handled by the hardware eliminating the need for para-virtualization and binary translation.

→ use-case:-

a. A provider maintains a data center with limited resources Compute & Storage. The providers Comp manages it's customers and employees Across 5 regions in India. What kind of a virtualization technique is suggested for the situations

Sol:- Para-virtualization X.

- since resources are limited. the guest OS can be modified to support the virtualization process.
- Since only the customer & Employees of that particular company are going to Access the cloud. → we can change the guest OS

note:
limited no-of resource => full virtualization
=> maximize the virtualized resource => due to there being.)
=> full will be better than para.

note:
para => "on-demand" resources will be virtualized.

Assignment:
- create an EC2 instance.

note: down "steps" <=> "one-on-one" interaction.

- deploy Java program into "google" colab.

06/09/2026 Database System:-

→ Normalization:-
- refinement process / distill process
- layers:- "1NF, 2NF, 3NF, BCNF" |NF. normal form|.

focus
- *1NF :- table should have only "Atomic value".
  - ~~multi-valued~~ every cell in table => only "one value"
- *2NF :- functional dependency $|\alpha \to \beta|$
- *3NF :- Transitive dependency $|\alpha \to \beta, \beta \to \gamma \Rightarrow \alpha \to \gamma|$.
- *BCNF :-

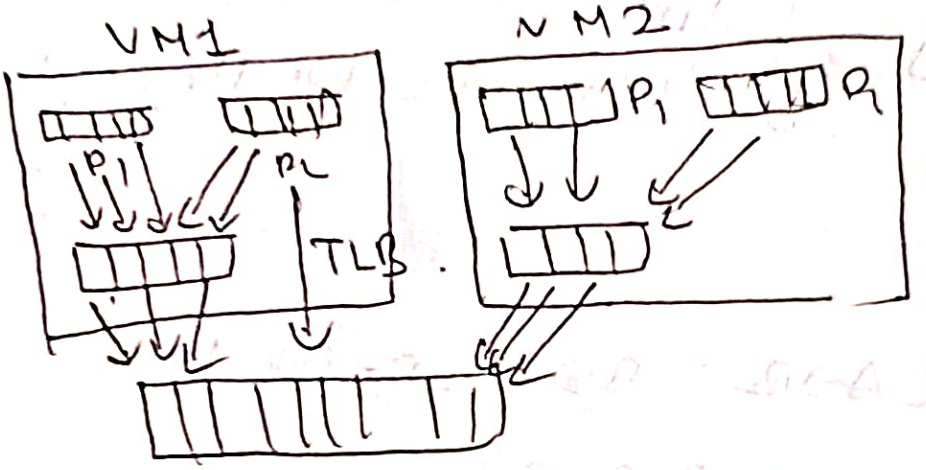
11/9/24 Cloud. Computing

→ Memory virtualisation :

* Done throug virtual memory [logical Addrs]
  - Mapping ob logigical & physical Addrs is done through MMU [through page table]

* Two stage mapping prob virtual memory to the physical memory of the guest OS and 2nd mapping will be from physical memory to machine memory [virtual machine]

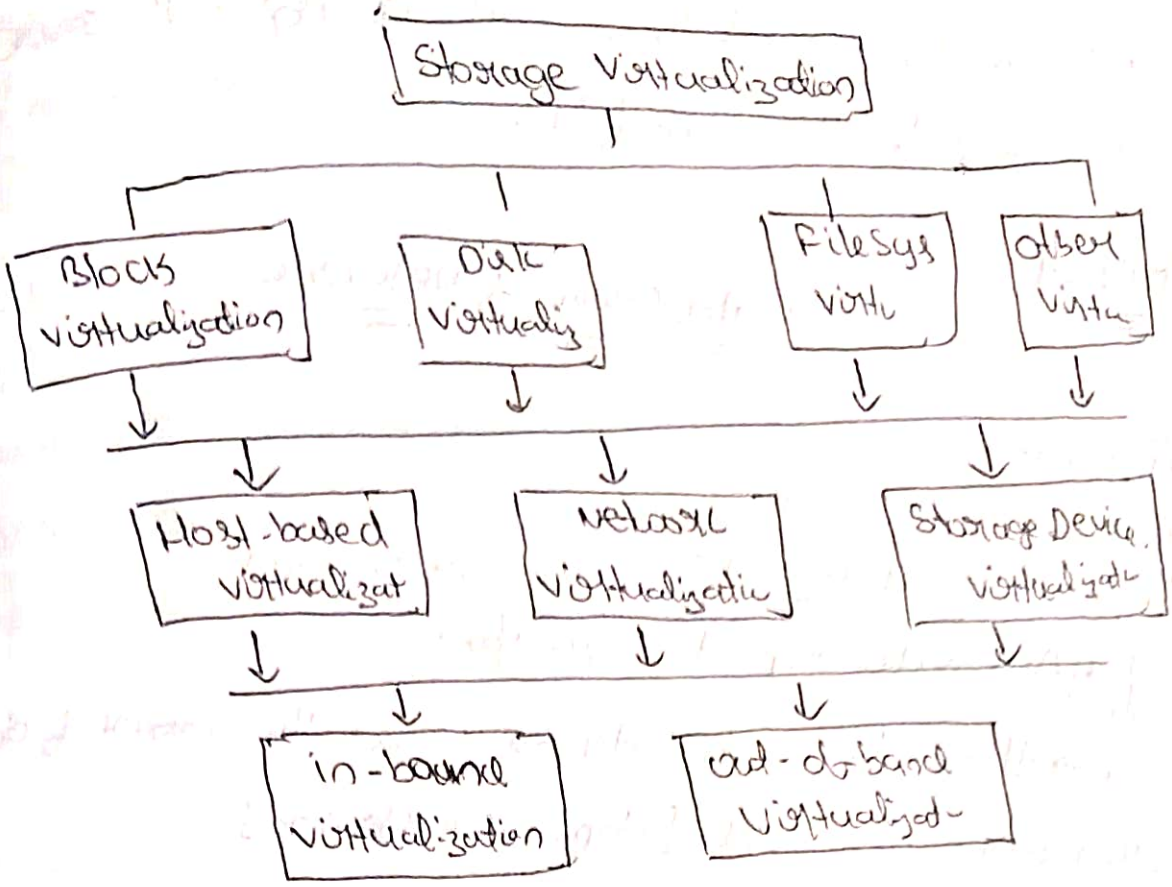
Two level memory mapping :-



Shadow page table : page tables used in the VMM

These page tables can be of diff. levels as well can be nested

Storage virtualization



* implementation of storage virtualization when it is host based it manages the path, volume and takes care of the replication backup
* when it is network path it takes care of path redirection, load balancing and Access Control.
* virtualization implemented at Storage device level taken care of volume managed Access Application and RAID (Redundant Array of Independent disk) Replication

javax.Servlet.*

GenericServlet
[no. diff. b/w get(), post()]

HttpServlet
[has get(), post() methods].

note:- install :- "netBeans & ApacheTomcat"

JavaLab

a. Define the class Student Containing the necessary variables for ID, Name, Marks[5 sub] "total" and "Avg": Capture necessary input from the user Can serialize the content into the file "student.txt" deserialize and Compute "total & Avg".

3/09/2024 Cloud Computing :-

... continuation:-

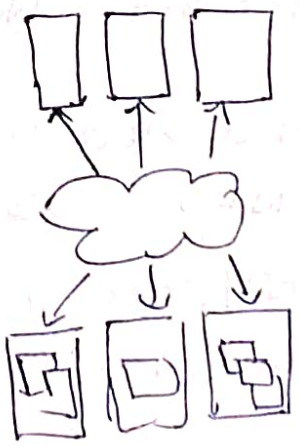
→ challenges of Storage virtualization:-

→ Block level Storage Virtualization:-

* multiple independent Storage arrays → presented to host as a single storage device.

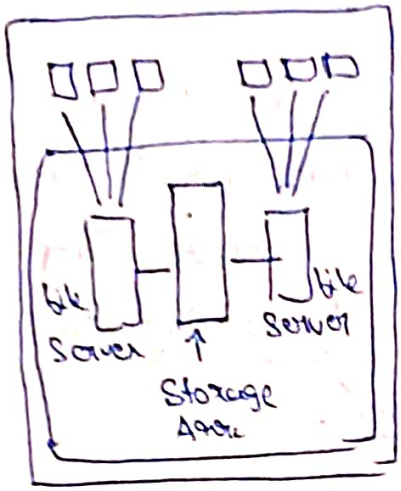
LUN: logical unit number

- mapping is done to redirect IO on the device to underlying physical array

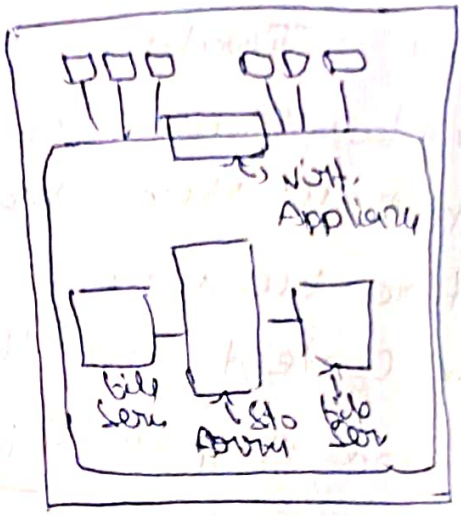


* It is deployed in a SAN environment to ensure non-disruptive data mobility and migration.
* It optimizes the resource utilization & cost.

-> File level virtualization:-
- File System:- organized way/method of storing data



before



After.

-> Network virtualization:-
* Physical network is presented as a logical layer.
* NV = Software Define Net. work + Network fu
Note:- If you have a limited no. of consumer with either. /Software Defined Network
SDN
- SDN is used to Datacenter virtualization at NFV is used for "WAN".
- The difference b/w conventional model & the technology is the use of "s/w" to manage the

18/09/24

Cloud Computing

→ Network virtualization:

L_1, L_3, L_2 over L_3 (overlay on top of physical layer)

* Virtualization happens over the overlay.
* NV is a process that involves abstracting network resources that was delivered through h/w originally is converted into a s/w based.
* NV is the process of grouping multiple phy networks into a single or multiple logical networks.
* NV Components:-

1. Network Hypervisor:- responsible for Abstracting network services and creating a virtual layer (the operators on the physical infrastructure.
   - VLayers consists of switches, router, load balancers etc, setting up firewalls.
2. Controller s/w :- helps in seamless comm. b/w the VMs & the networks.
3. Host protocols & virtual switches:- responsible for Implementing the host protocols for Virtual LANs and VXLAN (X ⇒ extensible, VL ⇒ virtual) that

manages the workload in virtual machines.

* Management tools: help in monitoring & controlling the network functions. & physically.

Types of Network virtualization:

1 External Virtual network virtualization: It involves network switching hardware & soln like VLANs and creates isolated segments within the physical network.

Suppose, two networks BSNL.

* It creates isolated segments of the physical network and tries to abstract the Appln that is running.

2 Internal Network virtualization:

Network Address Translation "NAT" NAT table

log no.	physic

* It "emulates" network within an OS partitions [Image/copy].

* It uses virtual Network Interfaces, shared interface, & NAT to establish a "Conn" b/w guest & host.

* To modern Network virtual 'GRE' is used.

(Generic Routing Encapsulation)

10/01/24

Cloud computing:-

→ Virtual machine:- [only a digital object]

* VM migration:- [- also cold, hot(live)]
  - cold: VM is shutdown
  - hot: VM is running.

Steps to "Migration" [Imp]*

Stage 0: Pre-migration.
- Activate VM on Host A
- Alternate VM on Host A physical host may be preselected for migration. Block devices mirrored and free resources maintained.

Stage 1: Reservation:
- Initialize a container on the target host

Stage 2: Iterative pre-Copy
- Enable shadow paging
- Copy dirty pages in successive rounds.

Stage 3: Stop & Copy
- Suspend VM on host A
- Generate ARP to redirect traffic to Host
- Synchronize all remaining VM state to Host

stage 4: Commitment
VM state on Host A is cleared

stage 5: Activation
VM starts on Host B
connects to local devices
Resumes normal operation

⇒ Hadoop and Map Reduce-1

⇒ Programming Model : [new Module]

* Parallel & Distributed Paradigm

a. find out the sale value of the "product" in a location buy a customer through a sales person from a warehouse during the festival:

(a)

*characteristics of Big data:-
-Volume, velocity, variety, value, voracity.
↓
uncertainty

*OLAP
↓
Analytics (Historic Data, Duplicate. ⇒ Agreed, de-normal

*OLTP
↓
Transactions [normalization]

CAT-II :- Heuristic Optimization & Relation Algebra + SQL + [illegible]

↓

• 10 Marks

⇒ Syllabus :- Normalization, Heuristic Optimation, Distributed DB, MongoDB.

23/09/2024

Cloud Computing

→ Hadoop :-

- open source Sto framework clusters of commodity hard ⇒ 'storage'.
- Hadoop 1.0 ⇒ Scalability issue

  Hadoop 2.0 ⇒ introduce YARN,

App^n :- i) Data intensive text process etc.,

features :- (i) suitable for Big data Analysis.

(ii) scalability

(iii) fault tolerance. [System perform independent on fault]

Hadoop Ecosystem :-

1. Hadoop Common
2. HDFS (Hadoop distributed filesystem).
3. YARN (Yet Another Resource Navigator)
4. MapReduce.

* HDFS:-

- based on "google file system."
- MasterNode, Slave Node [Master-Slave Architecture]

MasterNode ⇓ NameNode ⇓ [Meta-Data]

Slave Node ⇓ DataNode ⇓ [Store data]

provide fault tolerance ⇑ [replication factor] ⇑ 3 by default

- Block data [Fragmentation - 4 replicated]

⇓ Smallest unit of storage ⇒ contiguous storage, default ⇒

→ NameNode:-

Meta-Data :- Name, replicas, .. /home/fool/data, 3, ...

(Name ↓ DataNode name; replicas ↑ no. of; /home/fool/data = path; 3 ↑ rep. fac)

- Assume 3 different Dataset of "Employee, project. Details, Exp. project_emp", with sizes 1200 mb, 1000 mb, 3000 mb respective. define the metadata for these three dataset to be distributed in a HDFS.

(project. Details = 3; Exp. project_emp ⇓ 4 or 5th)

$\frac{1200}{256}$ [5 block]

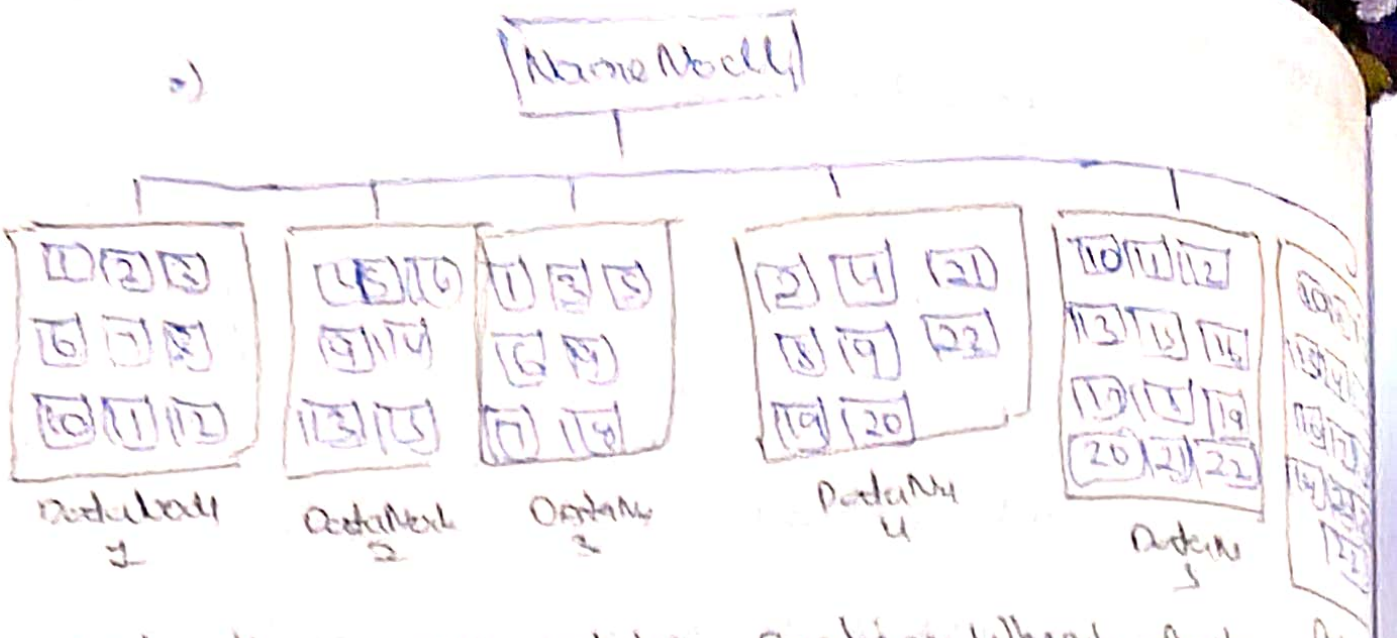
metadata:-

1. Employee, Name1, R:2, BL[1, ..5], user1/data/emp1.
2. Proj-Detail, R:3, BL[2,4,5,6] [6, 7, 8, 9] user2/data/proj-1.

$\frac{1000}{256}$ ⇒ 4 block.

3. EmpExp-projec, R: 3, BL[10-22] user3/dat/..

$\frac{3000}{256}$ = 12 block.



Note:- there will not be caching when data is large.

23/9/24 Data Comm / Network.

* Controlled Access :-

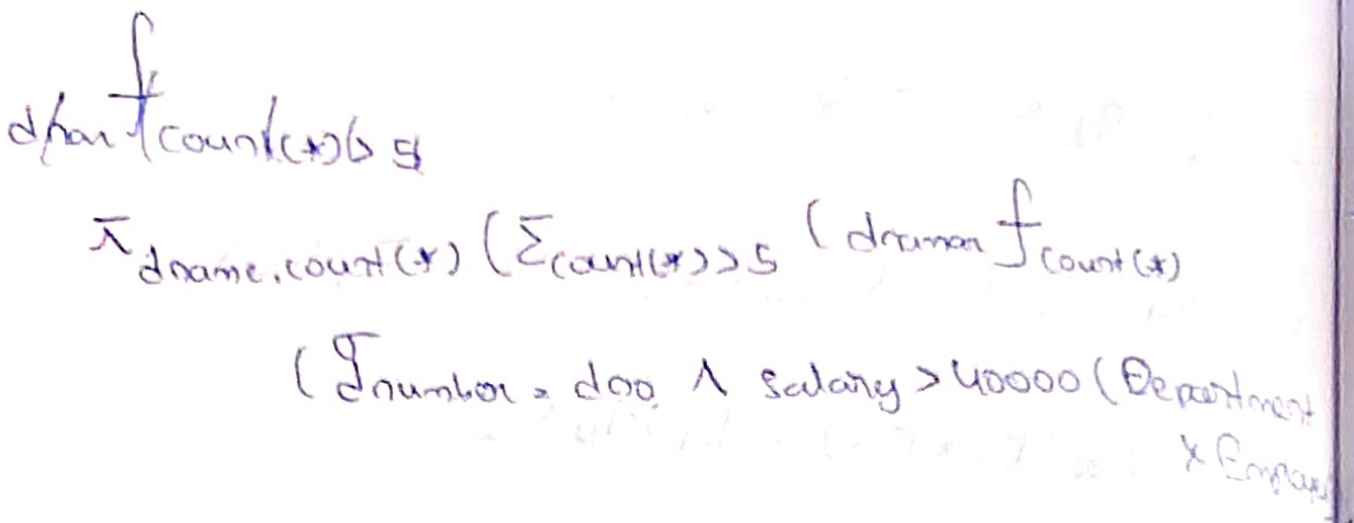
Token passing - Network token frame moving around the - who want's to transmit - fetch token - then transmit.

→ Channelization :- (Datalink layer)

1. FDMA
2. TDMA
3. CDMA

→ Heuristic optimization Algorithm :

Select Dname, count(*) from Department, Employee where Dnumber = Dno AND Salary ≥ 4000 group by Dname. Having count(*) > 5;



25/09/24

Cloud Computing

two imp processes:

1. heartbeat :- [send by datanode → namenode] check liveness
2. BlockReport :- [send by data node → to namenode]

NameNode ⇒ Failure

⇓

secondary Namenode

[content of the NameNode will be saved periodically in the secondary]

[It cannot take over the role of the Namenode]

* Communication Protocols:
  - HDFS protocols run on top of the TCP/IP.
  - Exclusive => Client, DataNode Protocol.

Assume 3 file do size 1000, 1200 and 3 GB,

typ: DataStructures: FsImage, and EditLog

25/09/24

Data Comm. & Networks

=> IP Addr:

* classful Addressing [IPv4]

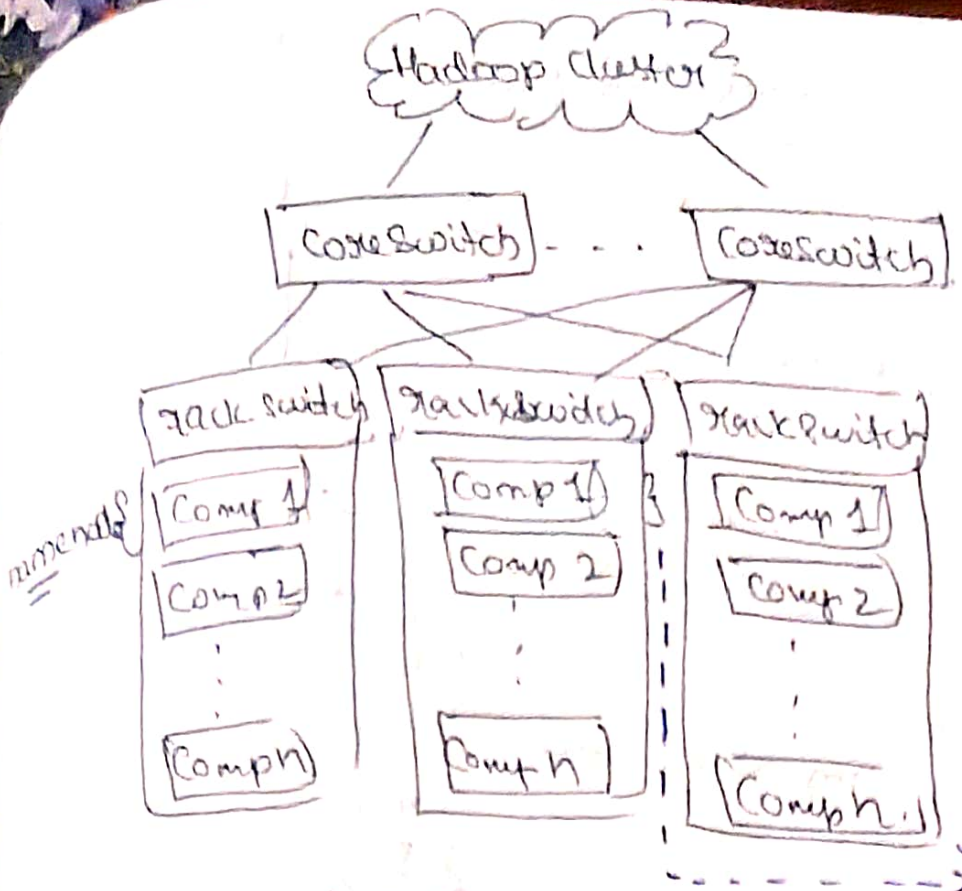
unique

- Any form i.e., Binary, Decimal, Hexa-decim
  ↓
  for convinence sake

128.11.15.1.

- 0 to 255

Class A: 2^{31}	=> 50%	0-127	2^7 byte 1-netID, 2^{24} 3 byte host
Class B: 2^{30}	=> 25%	128-191	2^{14} 2 byte - netID, 2^{16} 2 byte host
Class C: 2^{29}	=> 12.5%	192-223	2^{21} 3 byte - netID 2^8 1 byte host
Class D = 2^{28}	=> 6.25%	224-239	=> multicast
Class E = 2^{28}	=> 6.25%	240-255	=> Reserved for future use



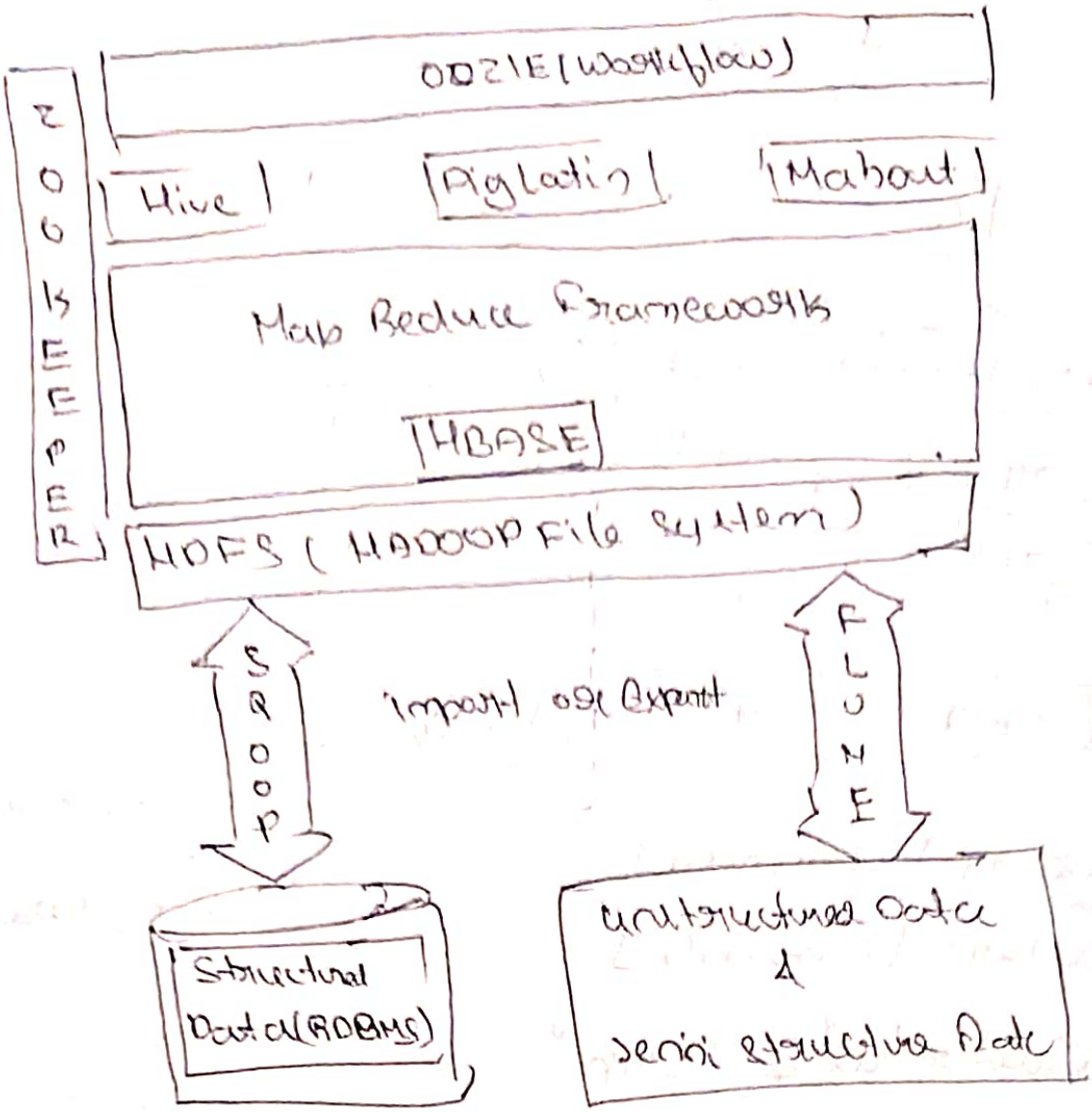
• TaskTracker :- [Datanode]

• JobTracker :- [namenode]

Hadoop Cluster has 3 Comp.

Hadoop Client :- It is neither master nor a slave but is responsible for loading data into the hadoop cluster, submit the mapreduce job, describes how data should be processed and retrieves the data, ~~to~~ looking the response after job completion.

Hadoop Ecosystem :-



→ HBASE [Support noSQL Database]

→ HIVE: [support SQL Data]

→ PIG: [Dataflow language, & scripting platform helps in dataflow. derived from Pig Latin]

→ Zookeeper: coordinating the services to the distributed Appln

FLUME, Spark, Storm :- handling data streams and near real-time data

SQOOP :- used for integrating databases and Data warehouses with hadoop.

OOZIE :- workflow scheduler & is responsible to represent workflows as directed Acyclic graph [DAG]

Ambari :- A web-based tool for provisioning, monitor hadoop by clusters.

AVRO :- is a data serialization

Chukwa :- collecting & managing large datasets

mahout :- Imp tool - include machine learning libraries.

Submarine :- AI platform

Pig :- Dataflow programing language - built on hadoop which can replace - the mapreduce engine.

Hadoop [mp]

Ex:- Consider a weather dataset with parameter temperature, humidity, pressure and wind speed. collected for "10 years" -> real-time data is also collected for these parameters on a daily basis. How can weather data be distributed in the

Cloud Computing

- MapReduce
- Data loss Prevention
- Data movement minimization } mapreduce does this to overcome "comm cost" "loss of data".

* MapReduce
  - Framework :- 2 functions [Map Function, Reduce Function]
  - Job :) "Full Program"
  - Task :) Job is reduced into "Tasks".

Map :- [key/value pair] intermediary.

$\{k_1/v_1\} \rightarrow \{k_2, v_2\}$ [group similar keys]

Reduce :- $\{k_2, list(v_2)\} \rightarrow$ (final key, final value).

biggest city in California key

Los Angeles is a great place

① ① ① ① ① ① (value)

California is a great place (key)

① ① ① ① (value)

grouping

California {1, 0}

low Angles {1}.

.

.

.

@ State 1:-

Prompt engineering is the process of designing input or prompts to guide AI models to generate the desired output. It's a way to provide AI models with context, instructions and examples so they can understand the users intent and respond Appropriately.

* Mapping:-

Prompt: 1, engineering: 1 is: 1 the: 1

process: 1 of: 1 designing: 1 inputs: 1 or: 1

prompts: 1 to: 1 guide: 1 AI: 1 models: 1

to: 1 generate: 1 the: 1 desired: 1 output: 1

: 1 a : 1 way : 1 to : 1 Provide : 1
: 1 model : 1 with : 1 context : 1 , intand : 1
and : 1 example : 1 so : 1 they : 1 can : 1 , under-
stan : 1 the : 1 , users : 1 intent : 1 and : 1
respond : 1 Appropriately : 1.

grouping :-

prompt : {1}

engineering : {1}

is : {1}

the {1, 1, 1}

process : {1}

of : {1}

designing : {1}

Inputs : {1}

on : {1}

Prompts : {1}

to : {1, 1, 1}

guide : {1}

AI : {1, 1}

Models {1, 1}

generate : {1}

desired : {1}

outputs : {1}

this : {1}

a : {1}

way : {1}

provide : {1}

with : {1}

Context : {1}

instruction : {1}

and : {1, 1}

Examples : {1}

so : {1}

they : {1}

can : {1}

understan : {1.}

users: {1}

intent: {1}

respond: {1}

Appropriately {1}

Reduce:

promt: {1}, engineering: {1}, u: {1}, -lbe: {3}

proce: {1}, input {1}, or {1}, prompts: {1}

to: {3}, guide: {1}, AI: {2}, models: {2}

generate: {1}, desired: {1} output: {1}

As: {2}, a: {1} copy: {1}, provide: {2}

with: {2}, context: {1}, instructions: {1}

and: {2}, examples: {1}, to: {1} -key: {1}

Can: {1}, understand: {1}, users: {1}

intent: {1} respond: {1}, Appropriately: {1}.

db.people.find().pretty() :- Beautify the result

* Create:

db.createCollection('collection_name')

* Insert: default PK :- object ID is created.

display :- {

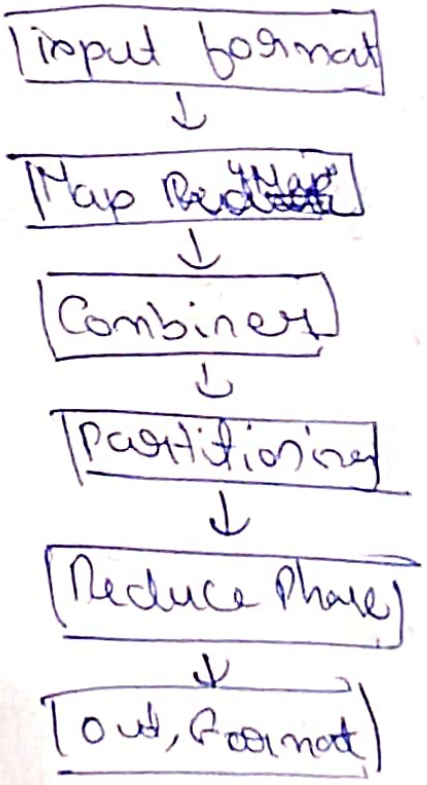
Colm : 'value'

Colm : 'value'

}

04/00/24 Cloud Computing

→ Dataflow in MapReduce



Partitioner :- It is to segment the 'key pair' based on the user defined 'function'. The no. of partitioners will be equal to the no. of reducers.

Id	Name	Age	Gender	Salary
1201	Gopal	45	Male	50000
1202	manisha	40	Female	51000
1203	vishnu	51	Male	51000
1204	Bikash	61	Female	10001
1205	yash	21	None	11001

Map Table :-

For Above, The map task accepts the key value pair as input and the key would be any key plus the file name + the line number.

Step 1:- Read value of the records, which is accessed from the input split and stored as a list of arguments.

Step 2:- Using the split function the gender value is separated and stored as a String.

Step 3:- The gender info. & the record data value are output as key value pair.

from "map task" to "partition task"

Step 4:- Repeat these steps for all the records.

Step 5:- partition task accepts the key values from map task & in this eka, the key will be gender & value = "The values for that particular gender.

Step 6:- read the gender value from input to value pair and store it as a String. The output of the partitioner will have two collections of "key value - pairs".

Step 7:- The reducer executes twice for two collections of gender & writes it on the output file.

07/10/24

Cisco Comm. & Network

→ Network Addr. Translation & Before the IPv6 term

* Private Addr :- 10.0.0.0 . . .

172.16.0.0 . .

192.168.0.0 . 70 } Can't access "Internet"

* public IP Addr :- Access Internet.