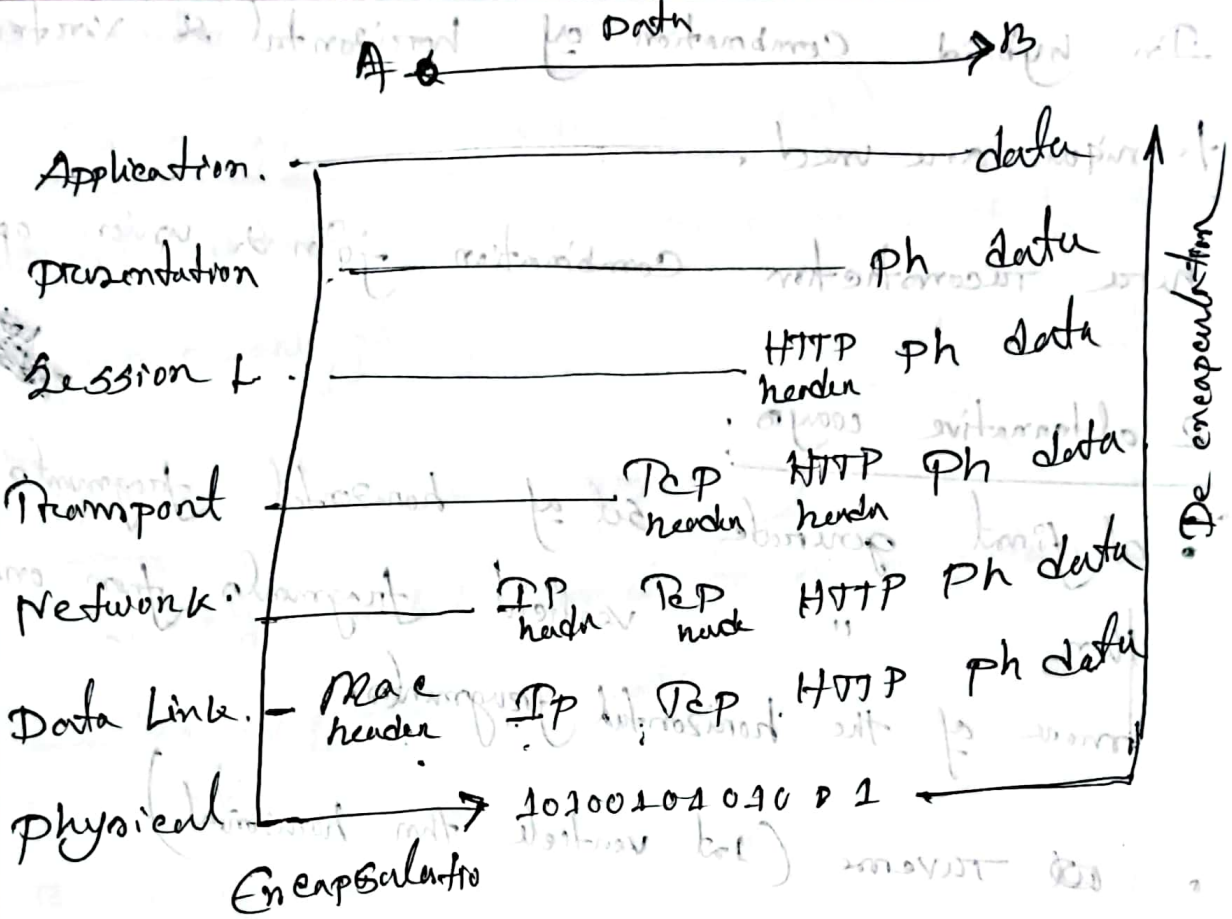
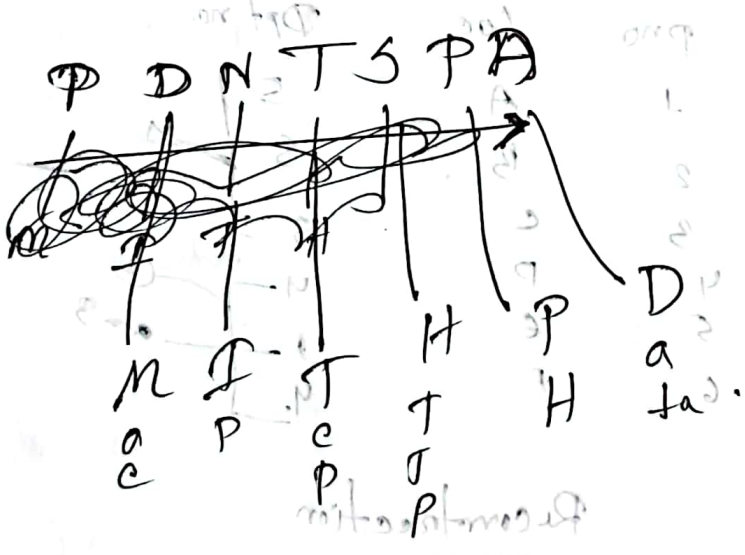
Sig1 Prerequisite

OSI model





leg 2 prerequisite

Ip address (Internet protocol)

Every device (phn, laptop) gets a unique ip address when it connects to network (unique number)

- send and recive data between my device & others.

Example : Let my ip 192.168.1.10. when go to google, lap top sends a request using this Ip and google replies it.

Mac Address : (Media Acess Control)

It is unique 12 digit hexadecimal number assigned to a devices network interface card (NIC)

- It works on hardware identifier.
- operates data Link and helps identify devices in local network

Example : when connect your devices to wifi router gives it an ip address so it can talk on internet router use your devices mac address so that understand which devie it is.

Port Address

It helps direct data to the correct application on a device.

Like IP identifies a device port number ensure data reach right services. (web browsing, email)

Example: Visit google.com, Computer sends a request to google server on port 443 (HTTPS) Server recognize the request and respond through same port.

Combined Example (hotel)

IP address → hotel address (where the building is)

Mac " → room number (which room u are)

port number → Service inside the room.

Chaak ip

public Ip → Assigned by Isp for Internet access.
(for find search "what is my Ip)

privite Ip : Assigned by router for local Network.
Communication.

ipconfig → [Ipv4 address → (192.168.1.100) [privte Ip
Defeult Gateway → router's Ip (192.168.1.1)

ipconfig /All → find mac address of my pc.

Chaak open ports :

port number helps to identify specific Network.
Services running on my laptop.

[Netstat -ano]

See the all active ports and proces Ip (PID

A Day in the life of a Web page Request.

- How many network protocols work together when we are open a simple webpage like www.google.com

for connecting Laptop to the internet and open a webpage:

1. Getting an IP : DHCP, UDP, IP, Ethernet.

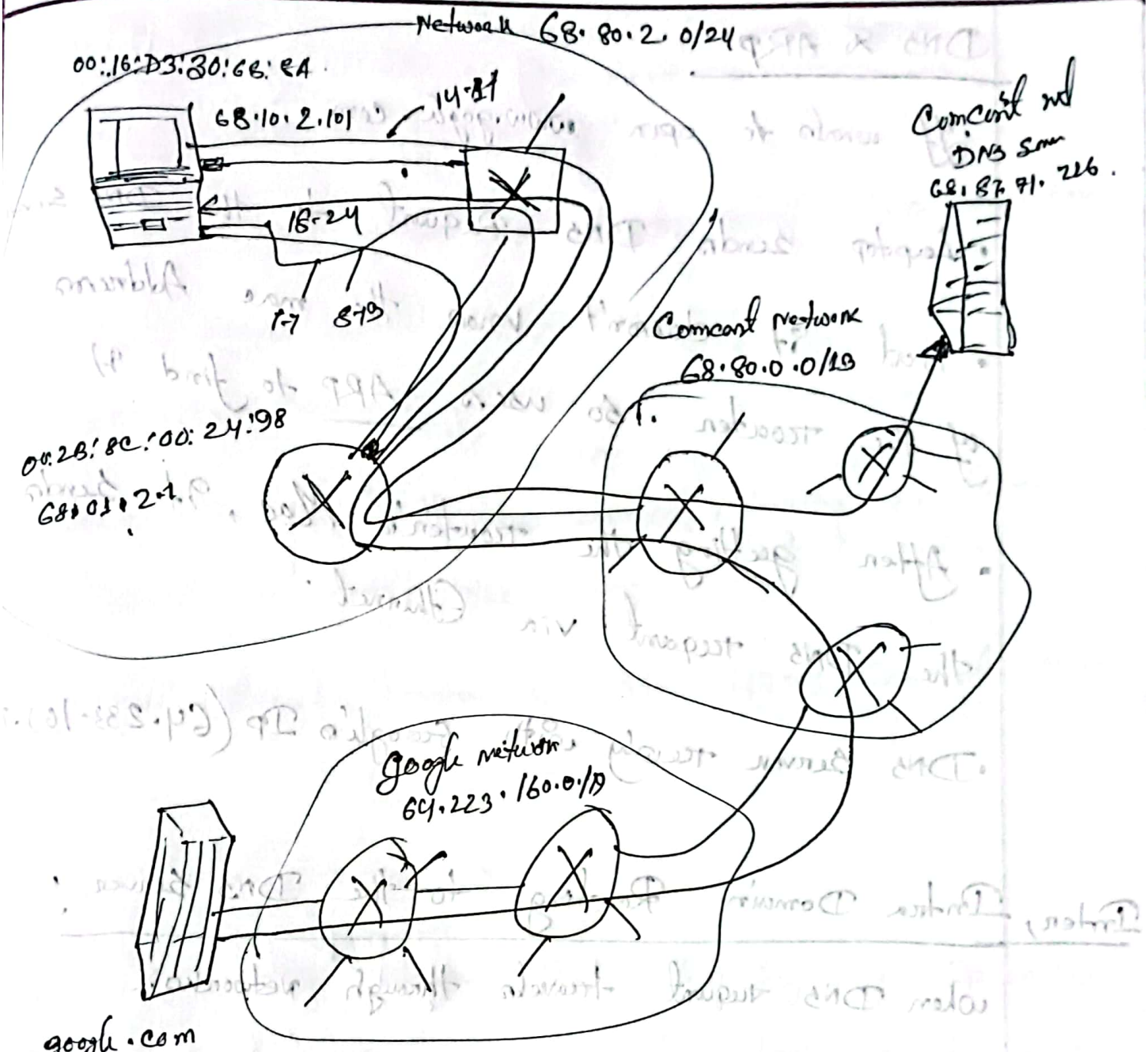
- first needs an Laptop IP to start (1st step)
- It uses DHCP to ask the router for
  - IP address.
  - Default gateway
  - DNS Server info.

for IP

way.

IP Link

- DHCP message are sent using -
  - UDP inside IP datagram, inside Ethernet frames.
- The Router replies with an IP (68.52.2.101) and DNS server (68.87.71.226)


Network 68.80.2.0/24
00:16:D3:30:C6:EA
68.10.2.101
14.22
18.24
1.7
8.19
00:2B:8C:00:24:98
68.01.2.1
Comcast network
68.80.0.0/13
Comcast nd
DNS Serv
68.87.71.226
Google network
64.223.160.0/19
google.com
web Server
64.223.169.105

DNS & ARP

- If wants to open www.gogle.com :
- Laptop sends DNS Request to the DNS server
- But it doesn't know the mac Address of the router .. So uses ARP to find it.
- After getting the router's Mac, it sends the DNS request via Ethernet.
- DNS Server reply with Google's IP (64.233.169.205)

Inter, Intra Domain Routing to the DNS Server :

when DNS request travels through networks :

- Routers use Intra-Domin routing like OSPF (inside the same isp)
- Across ISP and global network, inter domin routing like BGP is used.

Web Client Server Interaction : Tcp & HTTP

Connecting web server Tcp & HTTP.

- Now we know google IP, laptop start Tcp connection.
- uses 3 ways handshake.
  (SYN - SYN-ACK - ACK)
- Once connection is established, sends an HTTP get request to ~~fetch~~ get the webpage.
- google server responds with HTML content over Tcp

Display the page.

- webpage data broken into Tcp segments. IP packet and frame, packets are routed back to Internet to user.
- HTML content shown in google homepage

DHCP - for ip config	UDP/IP → delivery msg	ARP - find Mac
DNS - Domain name to ip	Tcp - reliable connection	
HTTP - web data transfer	routing protocol → move data between	

Distributed Database

Lec 1

- It is a database where data is stored across multiple computers or location, but it behaves like a single system to the user.

Design Considerations for Distributed Database

Data partitioning >> Replication >> Consistency & Concurrency control >> Network Communication & Latency >> Security & Privacy.

Data partitioning

- It means splitting a big database into smaller parts. and storing them on different computers.

- two types.

Horizontal fragmentation: Rows (tuples) of a table divided into groups.

faster performance became work done parallel.

Distributed Database

Lec 1

- It is a database where data is stored across multiple computers or location, but it behaves like a single system to the user.

Design Considerations for Distributed Database

Data partitioning >> Replication >> Consistency & Concurrency control >> Network Communication & Latency >> Security & privacy.

Data partitioning

- It means splitting a big database into smaller parts. and storing them on different computers.

- two types.

Horizontal fragmentation: Rows (tuples) of a table divided into groups.

faster performance became work done parallel.

Distributed Database

Lec 1

- It is a database where data is stored across multiple computers or location, but it behaves like a single system to the user.

Design Considerations for Distributed Database

Data partitioning >> Replication >> Consistency & concurrency control >> Network Communication & Latency >> Security & privacy.

Data partitioning

- It means splitting a big database into smaller parts. and storing them on different computers.

Two types.

Horizontal fragmentation: Rows (tuples) of a table divided into groups.

faster performance became work done parallel.

Distributed Database

Lec 1

- It is a database where data is stored across multiple computers or location, but it behaves like a single system to the user.

Design Considerations for Distributed Database

Data partitioning >> Replication >> Consistency & Concurrency control >> Network Communication & Latency >> Security & Privacy.

Data partitioning

- It means splitting a big database into smaller parts and storing them on different computers.

Two types.

Horizontal fragmentation: Rows (tuples) of a table divided into groups.

faster performance became work done parallel.

Vertical Fragmentation:

- The columns (attribute) split into smaller parts.
- If table have a column key so that we can join it/med.

- Different users access different parts of Data.
- keep sensitive data.

Replication: It means storing copies of the same data on multiple nodes in distributed database.
It improves - Performance, fault tolerance, Data availability.

Types of Replication

full Replication: Complete copy of entire database.
- ensure data availability & Reliability.
- require lot of storage & update all in slow, costly

Partial Replication: Only selected parts database are copied.
- Save storage and reduce cost.
- based on how used & where used

Multi Master Replication : Multiple nodes can handle both reading & writing operations.

- Update can be made different loc so as to improve fault tolerance & system speed.
- Complex to manage & keep data constant bez.

Consistency and Concurrency control : In dd keeping data constant across multiple nodes is difficult. when many transaction occour then conflicts may occour. Manage. this techniques like Locking, timestamps, optimistic. control are used.

Network communication & Latency : To get best performance and availability, low latency networks and effective communication necessary.

Security & privacy : DD need strong security since data is spread across many loc. Important measures me/de encryption, accen control, privacy protection to keep data safe.

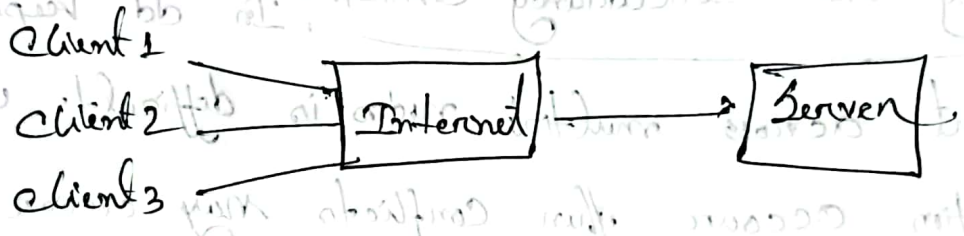
Distributed Database Architecture.

- Client - Server

. Clients request data or services.

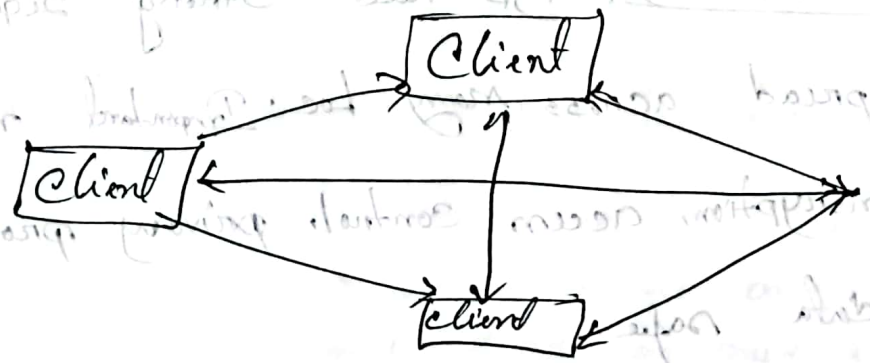
. A central server manage and process these request.

The server is in charge of maintaining data, storage, controlling access & organizing transactions.



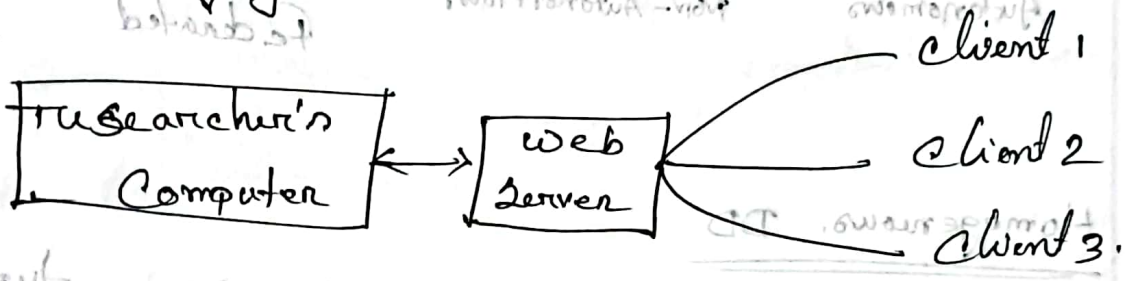
Peer to peer Architecture:

P2P each node in DD acts as both client and server. All node connected and work together to store & procsin data. Each node handale. with own data handle communication with other nodes. (ined Blockchain, folarud System)



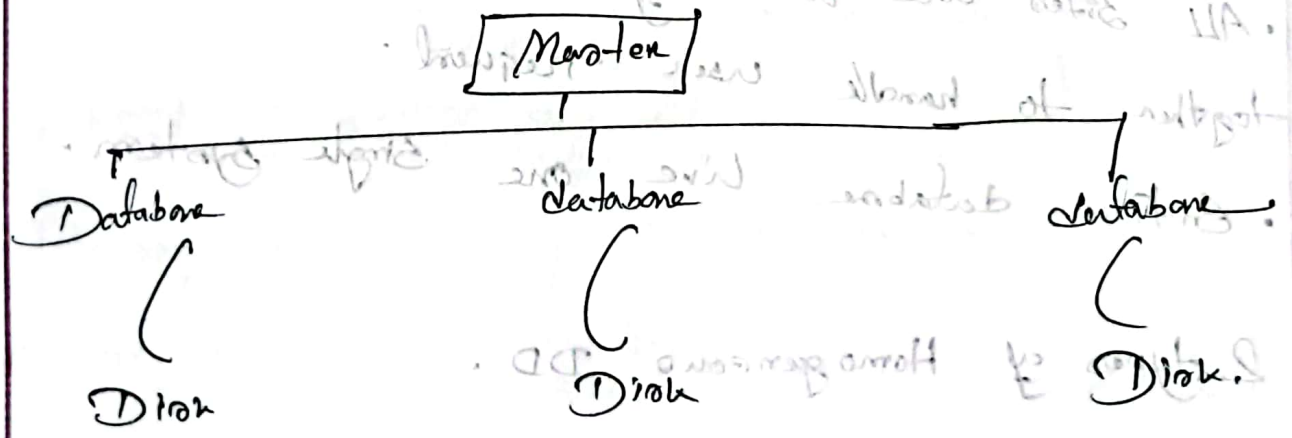
Federated Architecture: (Heterogeneous)

It connects multiple independent database into a single system using a unified interface. Each site keeps its own database, while virtual manager handle data request. useful for accessing different or legacy system without changing them.

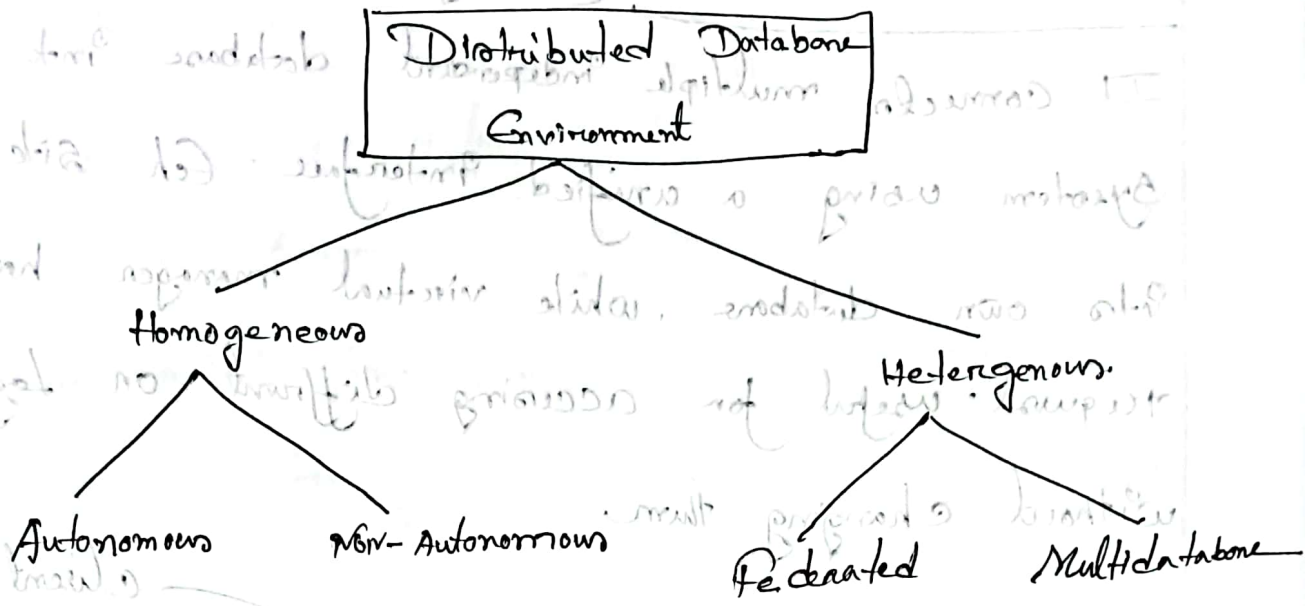


Shared - Nothing Architecture:

here data split across multiple nodes, with each node managing its own portion independently. Nodes dont share resources. useful for data warehouse



Types of Distributed Database.



Homogeneous. DD

here all (computer & Server) use the same type of database software and same operating system

Key features

- All sites use similar software and DBMS.
- ALL sites are aware of each other and work together to handle user request.
- Entire database like one single system.

2 types of Homogeneous DD.

Autonomous

- Each site works independently.
- They are connected using central application that helps them share data through messages.

Non-Autonomous:

- Data is shared across all sites.
- A central DBMS control & coordinate data update between the sites.

⊛ Heterogeneous DD

Here (Computer & Server) use different types of database software, data models, & os.

Features - Various DBMS - Relational, Network, Hierarchical, object oriented.

Query processing difficult.

Transaction processing also complex.

Cooperation is Limited.

Types — Federated

- Database system works independently.
- All connected in a way appear one system

un- Federated

- A Central controller is used to access manage all database.
- Less independance for individual database

Architecture.

DDB depends on 3 parameter.

- Distribution: It states the physical distribution of data across the different sites.

- Autonomy: This means how much control each part of a database system has. If a part works on its own without needing help, its high autonomy

Heterogeneity: It refers the dissimilarity of the data models, system components & database.

Architectural Models in DDBMS (management system)

client Server Architecture

Servers handle

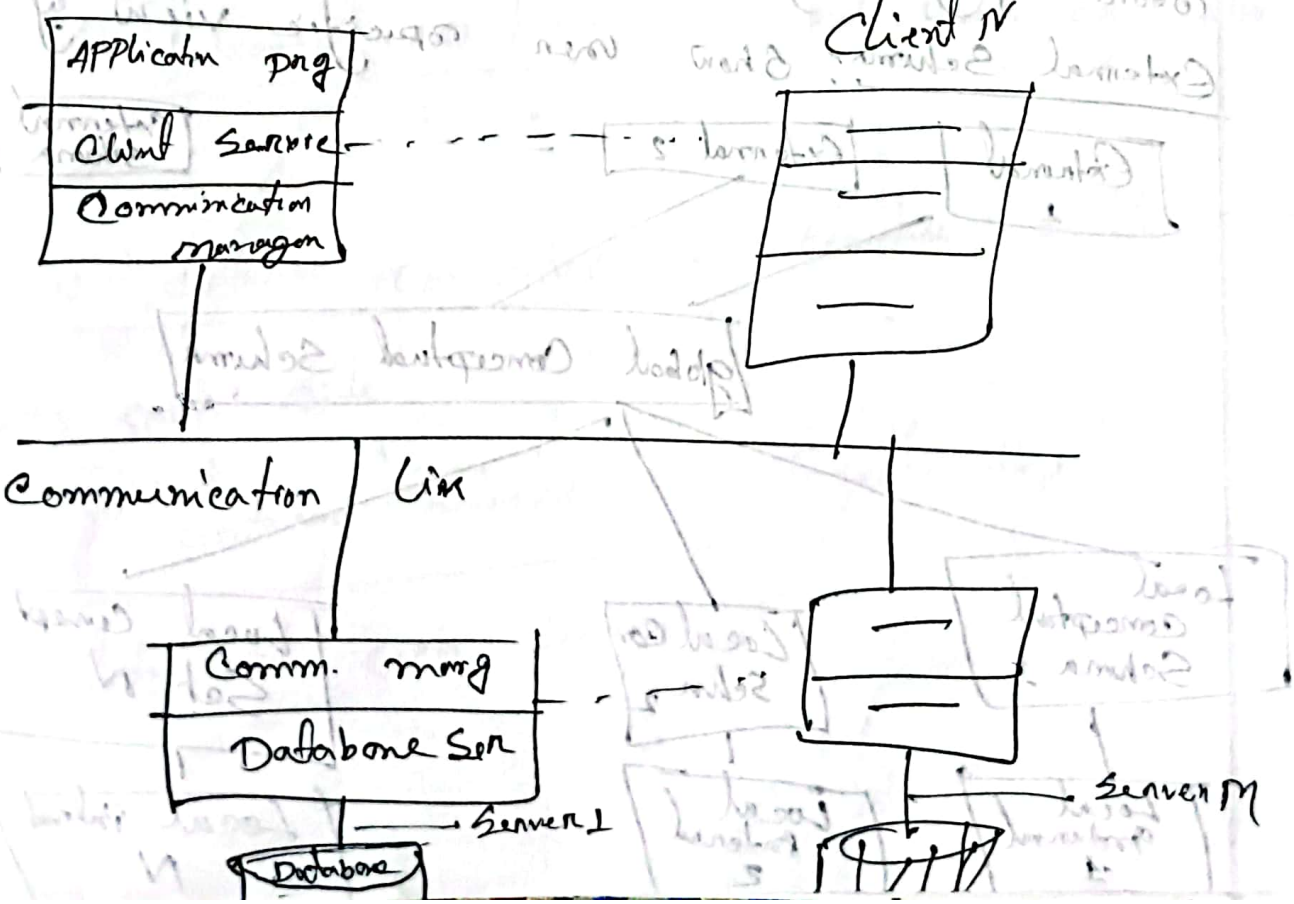
— Data management, Query processing, optimization
Transaction management.

Clients handle:

— user interface, Constancy checking.
transaction support.

2 types — Single Server, multiple client.

— Multiple Server, multiple client



Peer-to-peer Architecture

Each peer acts both as client and a server

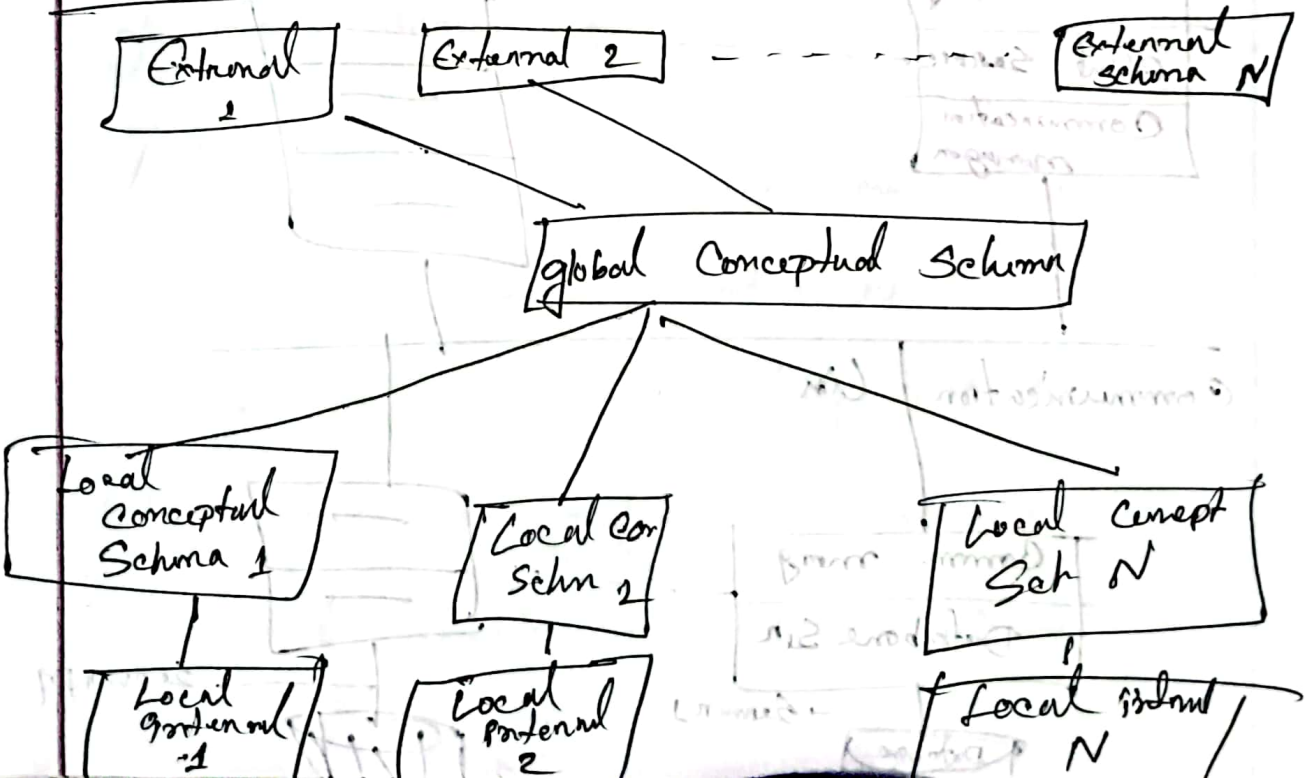
It has 4 Levels:

Global Conceptual Schema: Shows the overall Logical Structure of the data.

Local conceptual Schema: Shows how data is organized Logically at each site.

Local internal Schema: Shows how data is stored physically at each site.

External Schema: Show user specific view of data



3. Multi DBMS system

This involves integrating two or more autonomous database into one system.

6 level schemas.

Multi-database view level: Shows what different users see user specific views from the combined database

Multi-database Conceptual lvl: overall logical design of all connected database

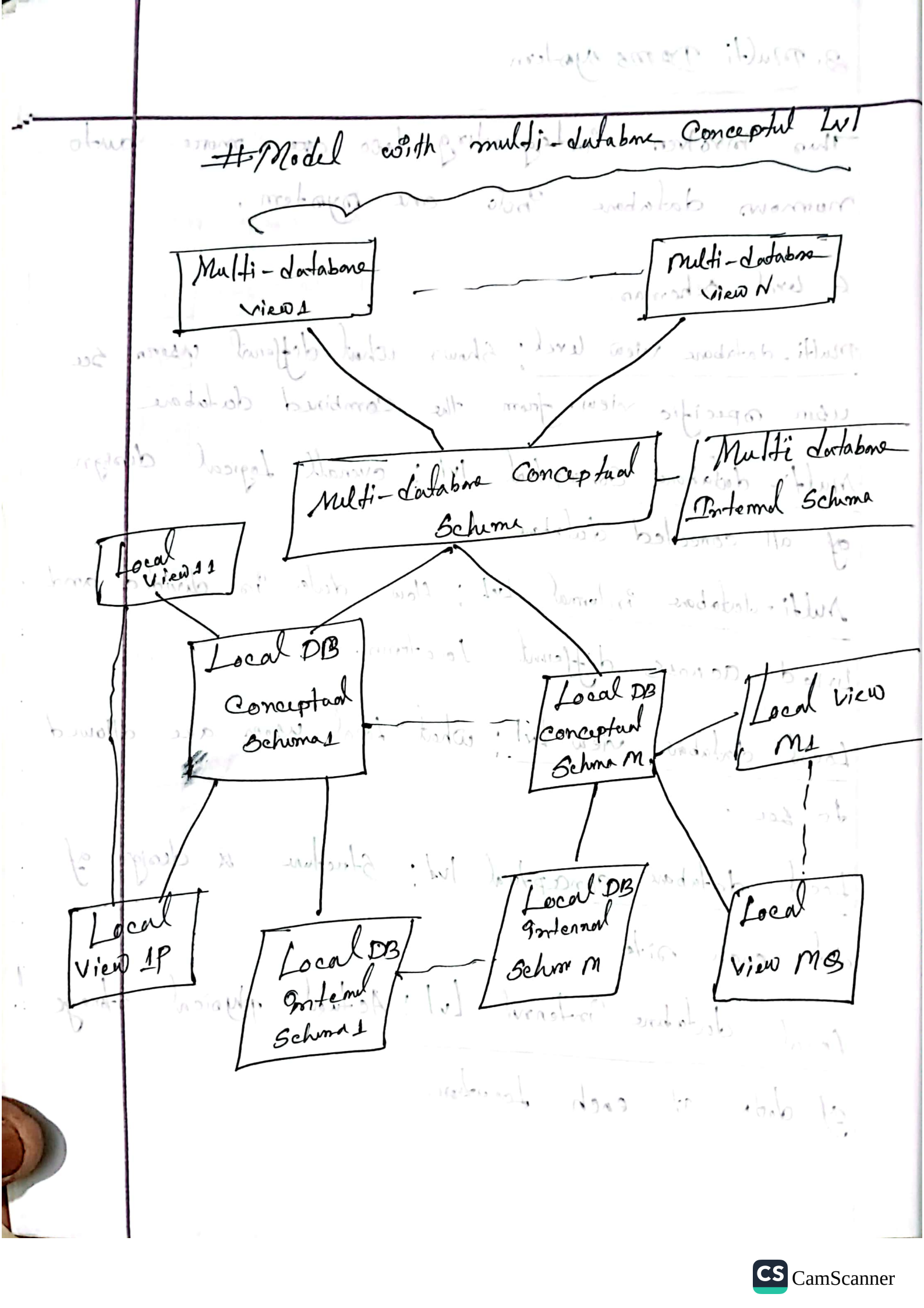
Multi-database internal LvL: How data is divided and linked across different Locations.

Local database view Lvl: what Local users are allowed to see.

Local database conceptual lvl: Structure & design of data each site

Local database internal Lvl: Actual physical storage of data at each Location.

Model with multi-database Conceptual Lvl



Design Alternative.

The distribution design alternative for tables are

NON-replicated & Non-fragmented

- Each table sorted one site.
- Best when few queries need data from multiple sites.
- Saves communication cost.

Fully replicated:

- Every site has a full copy of all tables
- fast for read queries ino delay
- update costly due to many copies.

partially replicated:

- Only some tables or parts of tables copied.
- Copies depend on how often & where accessed.
- Balance performance & storage

Fragmented

- tables are split and sorted many sites
- No extra copies (only one version)
- Increase speed and fault tolerance

& types fragmentation

- Horizontal (Rows are divided)
- Vertical (Columns are divided)
- Hybrid (mix of both)

Mixed distribution:

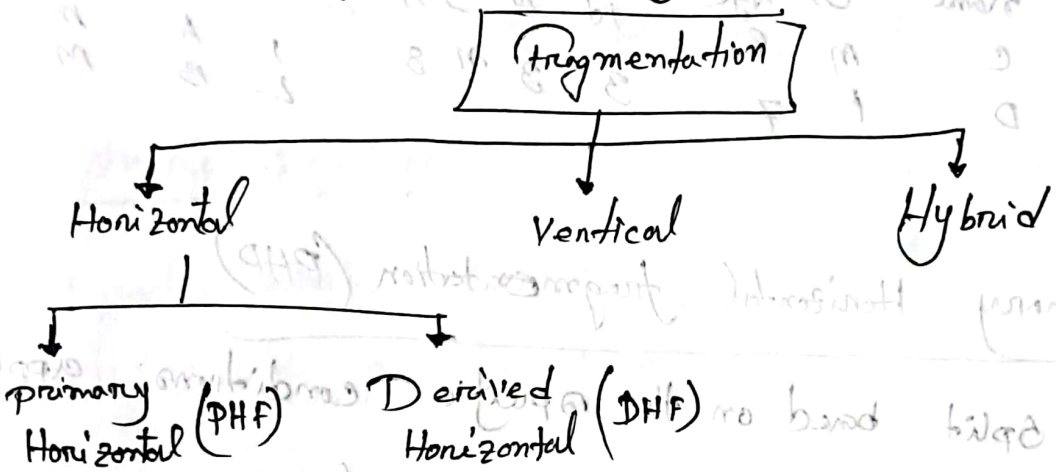
- Combines fragmentation & partial replication
- Fragments copies to sites where they accessed often
- Gives flexibility & better performance

Fragmentation

The process of dividing the database into smaller multiple parts is called fragmentation.

These fragment may be stored at different locations.

(*) Types of fragmentation. 3 types



Horizontal Fragmentation

Dividing a table into smaller parts based on the rows. Each part (fragment) has all the columns but only some of the rows from original table.

These parts are stored diffrent loc.

No row repeated in any fragment

id	name	Sex	Age
1	A	M	5
2	B	M	3
3	C	M	8
4	D	F	7

Site 1

id	name	Sx	Age
3	C	M	8
4	D	F	7

Site 2

id	N	S	A
3	C	M	8
4	D	F	7

Site 3

id	N	S	A
1	A	M	5
2	B	M	3

primary Horizontal fragmentation (PHP)

It Split based on the specific conditions apply in rows (tuples) [Str clm same but row same of some row added]

I	N	C	S
1	A	C	100
2	B	C	200
3	C	D	312
4	D	B	400

Based on City (C)

1	A	C	100
2	B	C	200

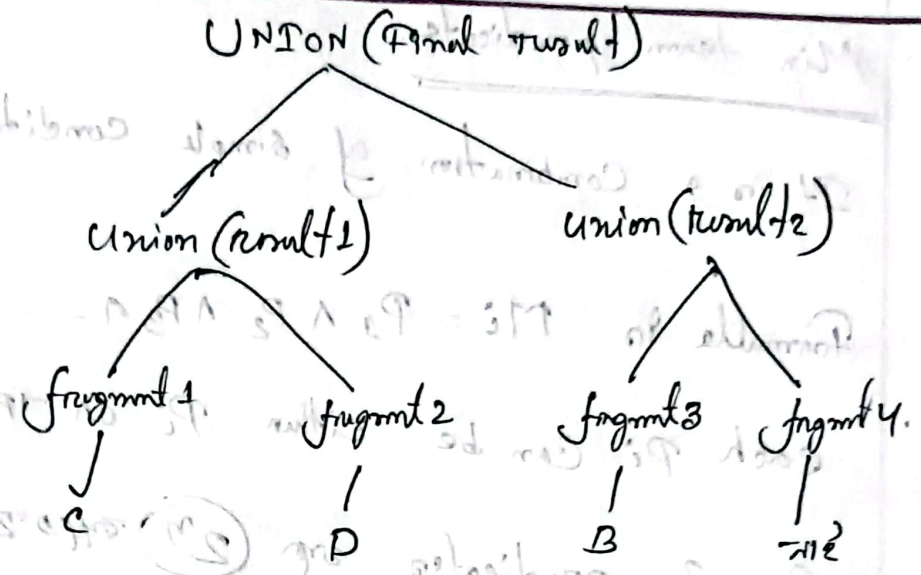
← fragment 1

3 C D 312 ↑ fragment 2

4 D B 400 ↑ fragment 3

Create table ... fragment as
Select * from emp
where city = 'pune'

Reconstruction



Simple predicate

It is a basic condition used to filter rows in a table based on the value of single attribute.

Pi : Ai Θ value → a constant

Attribute (Clm name)

Operator (<, >, ≤, ≥, =, ≠)

E	N	D	S
1	P	HR	25k
2	A	IT	32k
3	K	HR	28k
4	N	IT	35k

P = S < 30k, S > 30k

D = HR D = IT

S < 3k

HR

1 P HR 25

3 K HR 28

2 A IT 32k

4 N IT 35k

S > 30k

IT

Min term predicate

It is a Combination of simple condition (used And $\wedge$)

Formulla is $Mi = P_1 \wedge P_2 \wedge P_3 \wedge \dots \wedge P_n$

each P_i Can be either P_i or $\neg P_i$ (not)

যদি 2 টা predicates থাকে (2^n) সংখ্যক হবে

$P_1 = b < 10k$
$P_2 = B \geq 10k$ (2) So $2^n = 2^2 = 4$

All 4 min term predicates

$m_1 \rightarrow B < 10k \wedge B \geq 10k$ — invalid

$m_2 \rightarrow B < 10k \wedge ! B \geq 10k$ — valid

$m_3 = ! B < 10k \wedge B \geq 10k$ — valid

$m_4 = ! B < 10k \wedge ! B \geq 10k$

Sql for m_1

Select * from account
where b<10k AND B>=10k [invalid

↗ কোনটা একটা হবে

Same সব not হবে (!) (NOT)

It help dividing the data table horizontaly in DD

Correctness Rules for fragmentation

while Perform fragmentation process, outcomes expect.

- Should not loss data
- Should not get unusual data. & no data repeated

for check and

Completeness: no loss of data due fragmentation all records (row) from the orginal table must be present in at least one of the fragments.

Reconstruction: we should rebuild the orginal table from the fragments. | horizontal — union
vertical — joint

orginal table = fragm1 ∪ f2 ∪ f3

Disjointness: same data shouldn't in more than one fragment.

each row placed one fragment

$$R_1 \cap R_2 = \phi$$

Derived Horizontal fragmentation:

Split table (member) based on the values in another table (owner table).

owner rel^n → primary key

Member rel^n → foreign key.

owner (Campus)

C.Id	C.Name	Cots
1	A	DH
2	B	ch
3	C	Raj

Member table (Student)

Sid	Nm.	C.id
101	Rhim	1
102	Karim	2
103	Sumija	1
104	Rahib	3

So owner table fragmnt

HF1 = 1 A DH

HF2 = 2 B ch

HF3 = 3 C Raj

Drive fragmnt Member table (by forgin key)

101 — Rahim — A camps — Dhaka — (1) output

102 — Sumiya — B — Dhaka

output 2

102 Kamim B campm Tech

output 3

104 Rakib e campm Raj

Vertical Fragmentation

It is dividing by table into smaller fragments by columns [Break table vertically].

id	name	Sex	Age
1	A	m	3
2	B	m	5
3	c	m	8
4	D	F	6

id	Name
1	A
2	B
3	c
4	D

id	name	Sex
1	A	m
2	B	m
3	c	M
4	P	F

id	Age
1	3
2	5
3	8
4	6

[in each clm primary key added mf]

2 approchs

- Grouping (Bottom up approach)

start small → combine attributes step by step → from meaningful fragments.

Splitting (Top down approach)

start big - split based on acess pattern - get smaller, efficient fragments.

Reconstruction

here it is performed by using full outer join operation on fragments.

```
Select * from
frg1
NATURAL join
frag 2
```

Hybrid / mixed fragmentation

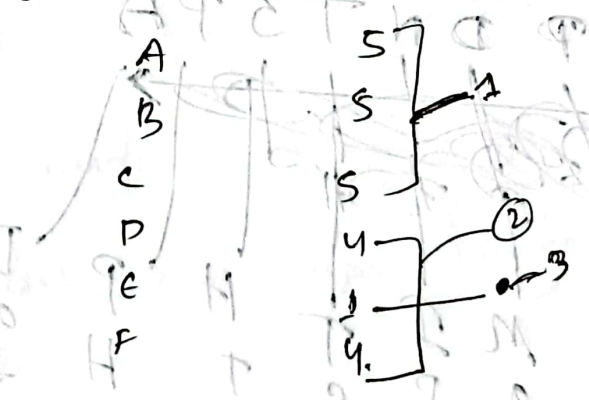
In hybrid combination of horizontal & vertical. techniques are used.

here reconstruction combination join & union operation.

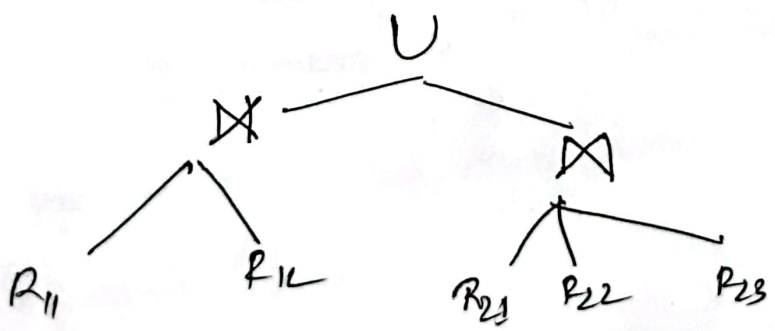
2 alternative ways:

- at first generate set of horizontal fragments then " vertical fragments from one or more of the horizontal fragments.
- traverse (1st vertical then horizontal)

product name	pno	loc	Dpt no.
P. X	1	A	5
P. Y	2	B	5
P. Z	3	C	5
com.	4	D	4
cueb	5	E	1
Bm.	6	F	4.



Reconstruction



Schema Integration

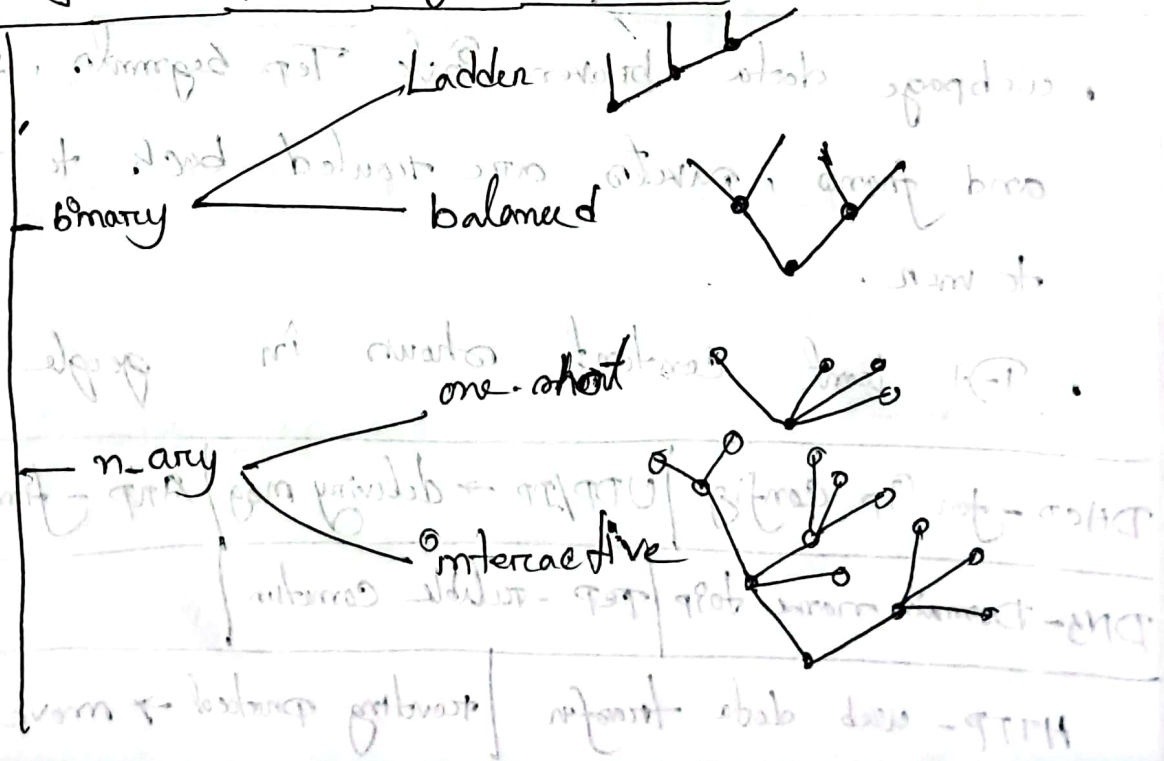
- It is a process of combining multiple Database schemas from different sources into one unified, global schema.

Why need

- Companies have many database
- Database may use different structures.
- Schema integration combines into one view

Benifits → Constant data, no duplicates, easy access.

Integration processing Strategies:



Steps of Integration process

Pre Integration: Decide strategy & find Schemas to Integrate.

Compare Schemas: find similarities & detect Conflicts.

Resolve Conflicts:

Conforming the Schemas: resolve conflicts to make merging possible. (usually need help from designer)

Merging: Combines & adjusted schemas to an Integrated version

Aid	Aname	mail
A1	Emma	01

Bid	Bookid	writer id	price
B1	Harry	A1	20

↓ pres (u Br)

Bookid	title	writer id	Price	Email	Anm
B1	Hm	A1	20	01	Emm

Conflicts

When integrating data from multiple heterogeneous sources, various conflicts arise due to diff erences in how data is structured, named or represented.

Types → Naming, Structure, Data type, Value, Key / Identify Constraint:

Naming Conflicts: Same things in diffrnt name.

Synonyms, homonyms.
E.g id vs st.id, postalcode vs pinnetCode

Structure Conflicts: Same data, diffrnt org.

Attributes vs entity, Aggregation Diff.
daily vs monthly sales.

addr vs Location

addr in dB, and another table

Data type conflicts: Same attribute has diff data types.

Phn no — Integer & string.

Value (Semantic Conflicts)

~~Different key Same entity.~~

Same field diff meaning on units.

price (USD vs EUR)

key Conflicts : Diff key Same entity (primary key)

Employee id vs Security no

Constraint conflicts : Diff constraint same data.

Phn no one require Not null, one require null.

VDI)

View Data Integration

View is a virtual table (not real or physical)

Create View, view name AS
Select Clm 1, Colm 2
from University (table name)
where CGPA < 3.

Features

Virtual: doesn't hold data itself.

up to date: See most current data from source table

Security & simplification: hide restricted access and show only certain clm & row to user.

[only (view) যায় move, copy করা যায় remove করা যায় interface থেকে]

A declarative mapping language: It is used to specify how data from diff sources maps to a integrated schema.

8 Approach

Global As view || local As view || Global & local As view

VIDS problem

Need data from multiple sources (heterogeneous)

this source diff from.

formates (text files, web page)

structures (diff schemas)

access method (web form, database clients)

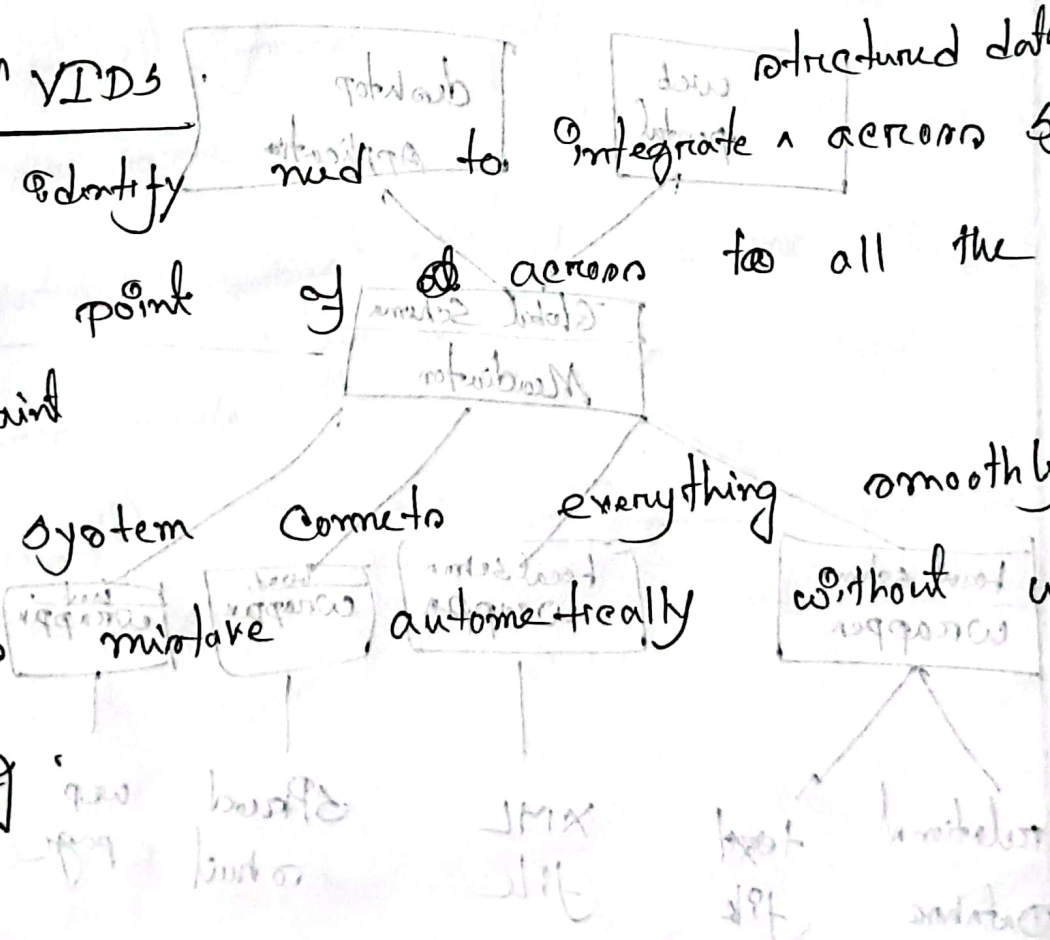
The 6th VIDS

① find identify need to integrate & access system structured data

② single point of access to all the Data point

* ~~From~~ system connects everything smoothly

& fixes mistake automatically without users noticing.



Data integration system

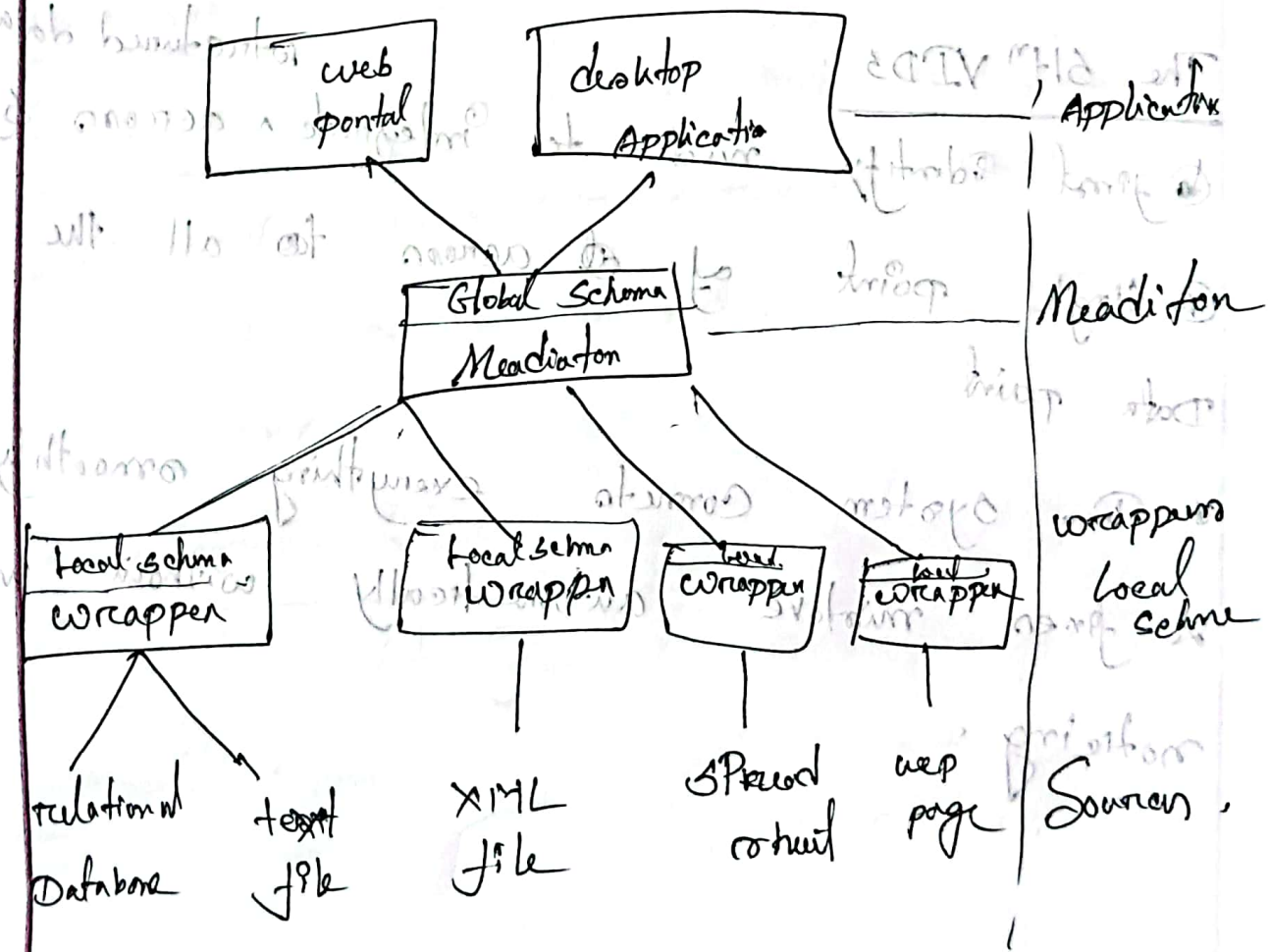
VDIS Architecture

Sources → Different database, files, web page

wrappers : Convert each source into a common format

mediator : uses mapping to combine data and present global schema

Application layer : Query global view without worrying about the data comes from;



Mediation process

① User sends a query to mediator without worrying about data in.

② Mediator understand & splits the query into sub-queries parts for each source

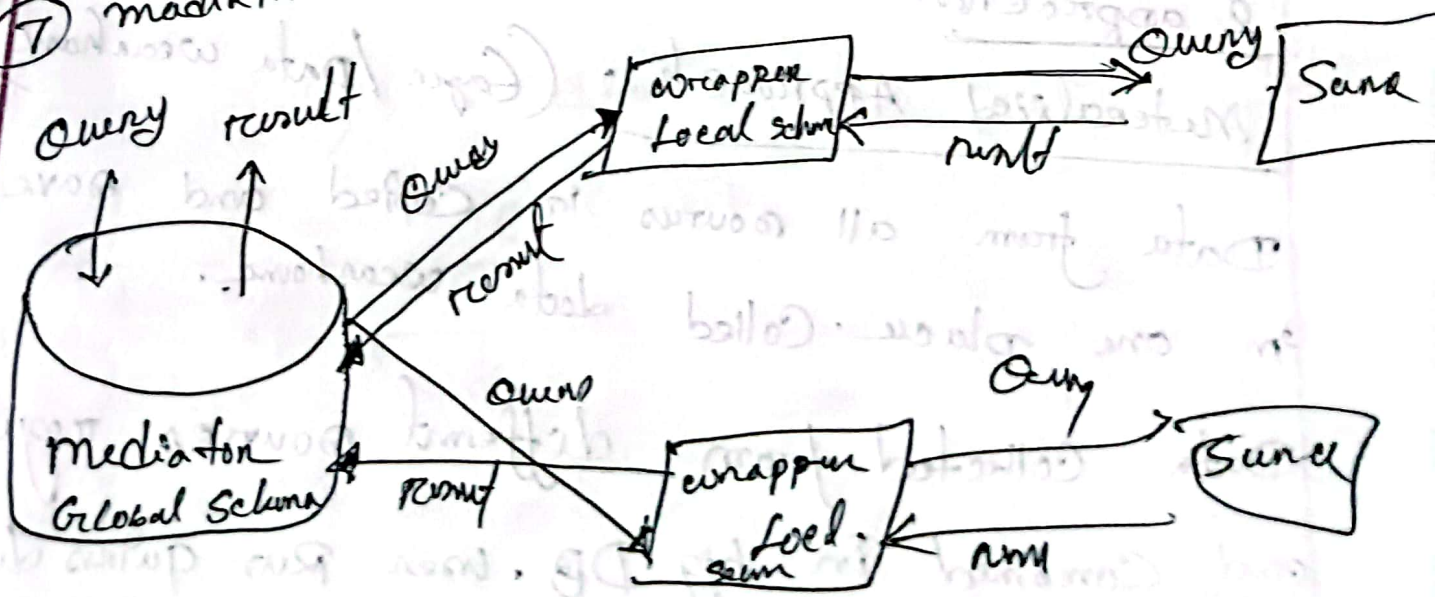
③ Mediator sends these sub queries to the right wrappers.

④ Wrappers translate and send queries to their Local Sources, Local Source return result to wrappers

⑤ wrappers format result and send back to the mediator.

⑥ Mediator Combine result into one clear ful Ans.

⑦ mediator sends the Ans to the user.



VDBS Categories

① Common data model & Query Language

- When different system share data, need a common way to understand & called data model.
- wrappers and mediator use the same data model & query language to talk each other.

Common data model - Relational, XML, OPP

② Mapping language

- Global as view || Local as view || Global & Local as view

③ Data storage method

where data actually stored.

2 approaches.

Materialized Approach: (Eager / Data warehouse)

Data from all sources is copied and saved on one place. Called data warehouse.

Data collected from different sources regularly and combined in big DB. User Run queries direct

Advantage

Good query result

No delay: doesn't fetch data from other source each time.

Dis Advantage

- Data can be outdated, Not update all time
- Need more storage merge data.

Virtual Approach

Data stays in the original sources. global data base is virtual, meaning there's no central storage of data.

Working: - user ask query split into smaller parts
- These part go diff data source real time
- System collect and combine answer and give the final result.

Advantage	Dis Advantg.
- Data up to date.	- Slower query due to network delay.
- No need Extra storage	- Source down then query are failed on daily

Materialize vs Virtual

Materialize	Virtual
Store centrally (warehouse)	Stays in original sources
Might be old	up to date
needs Extra Space	No - Extra need
works if Sources offline	Must be online Sources for work
Eg- Analytics, reporting font	• real time monitor, up to date dashboard, Slower

Mapping Global As View (GAV)

Global Schema is like a large structure or view that combines data. Global schema created based on local schema

the GAV mapping occur

$$V_i = I(R_i)$$

V_i — Local view

I — Global relation

R_i

vi gets data from Source provided Ri

Example $M_1 = V_1$ (Book)

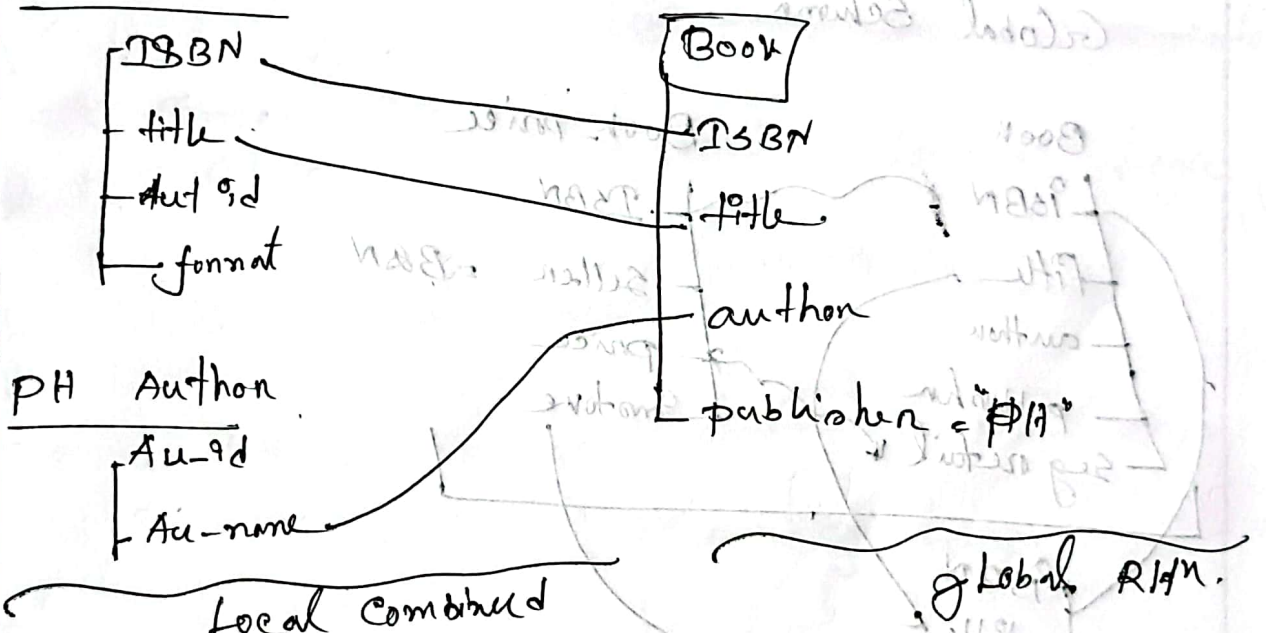
$M_2 = V_2$ (Book price)

V1 (Book) => ISBN, title, Author_name, 'PH')

PHBook (ISBN, aut-id, format)

PH author (Aut_id, Aut_name)

PH Book.



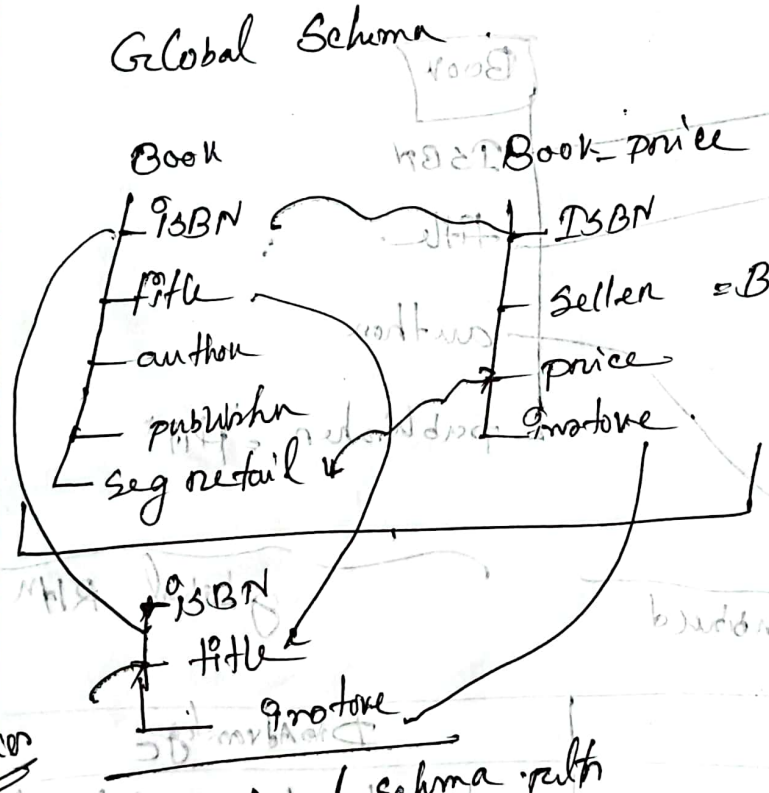
Advantage	DisAdvantage
Easy query writing fast query performance use real world terms.	- hard to update bez tightly linked with source - Limited flexibility only show what's in source.

⊛ Local As view (mapping)

- Local Schema is described in terms of the global schema. identify query.

$$\Pi(Ri) = Ui$$

Local relⁿ Ri is mapped to a query Ui over the global schema.



BRN deliverables

Advantage	Dis advantage
Easy to add new data Sources: Each Sources works on Its own.	write correct query tough. Missing Some data

Global-Local As view (Hybrid)

Generalization Both GAV & LAV.

Mapping $\Rightarrow V_i \rightarrow U_i$

V_i : Query Local.

U_i : Query global.

Adv	Dis Adv.
Flexible & simple each Sources can register Easely Can handle Complex mapping Mix the best LAV & GAV	~~Each Sources can register.~~ needs Special logic for answering (Complex) require smart query rewriting tools.

Some LAV Example here