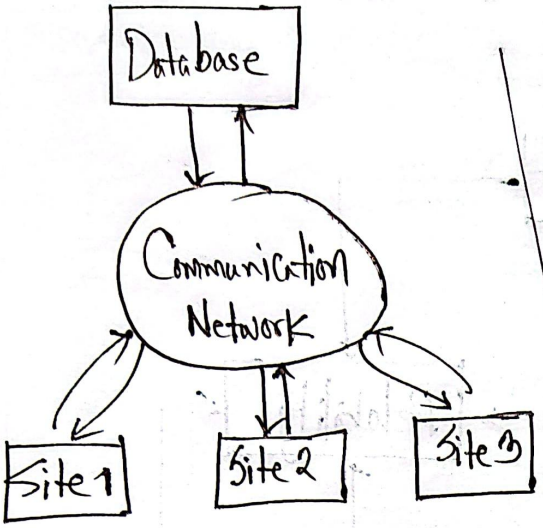
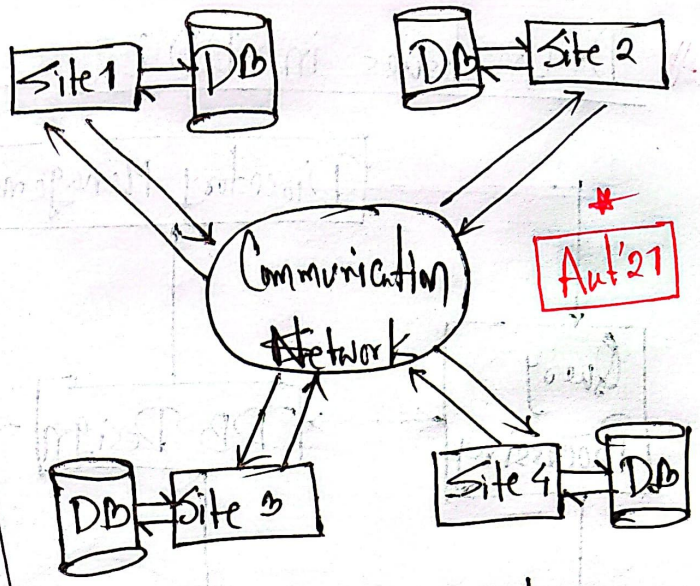
"Distributed Database"

Segment - (1)



Normal DB System



Aut'21

Distributed DB System

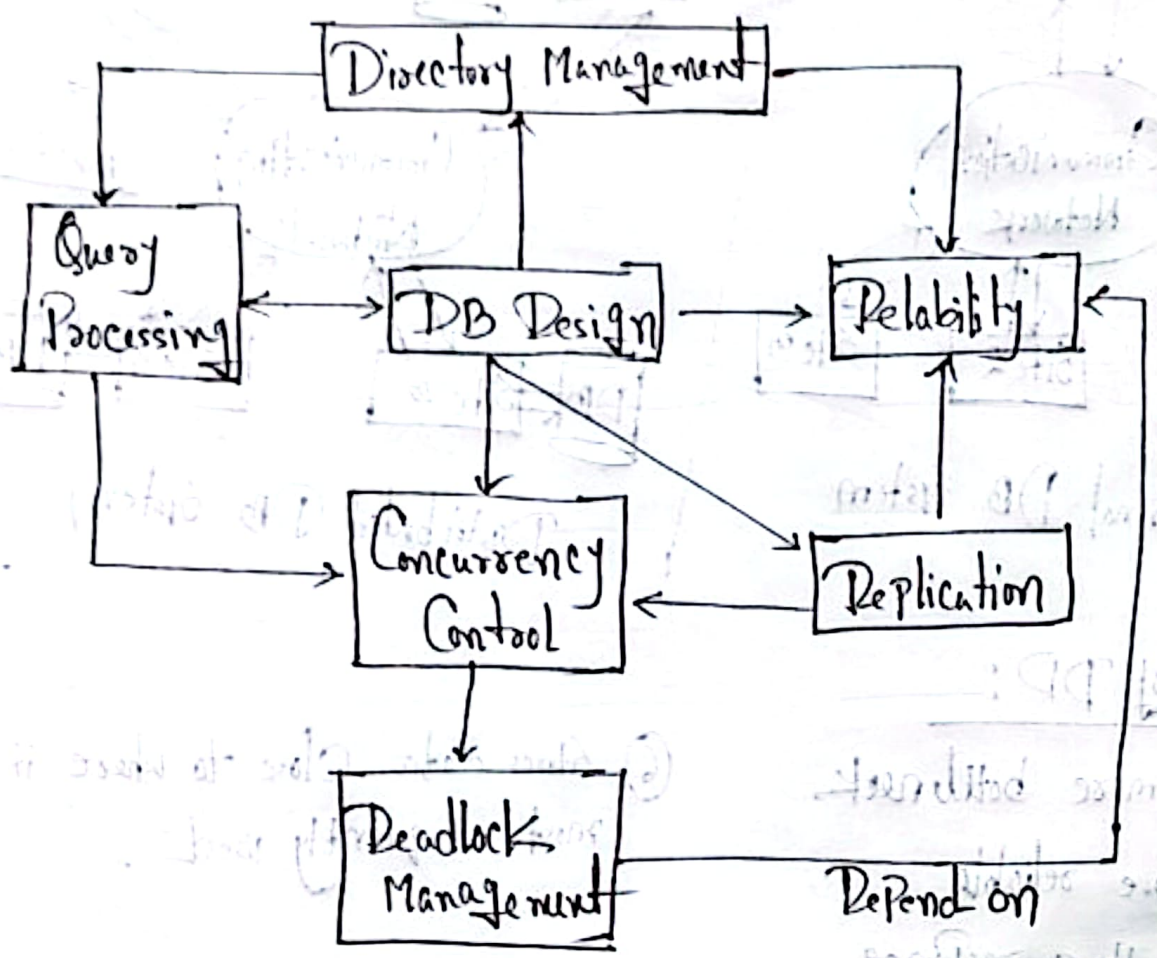
* Adv. of DD:

1. No more bottleneck
2. More reliable
3. Better response
4. Lower Communication Cost
5. Can easily expandable
6. Stores data close to where it is most frequently used.

* Defination:

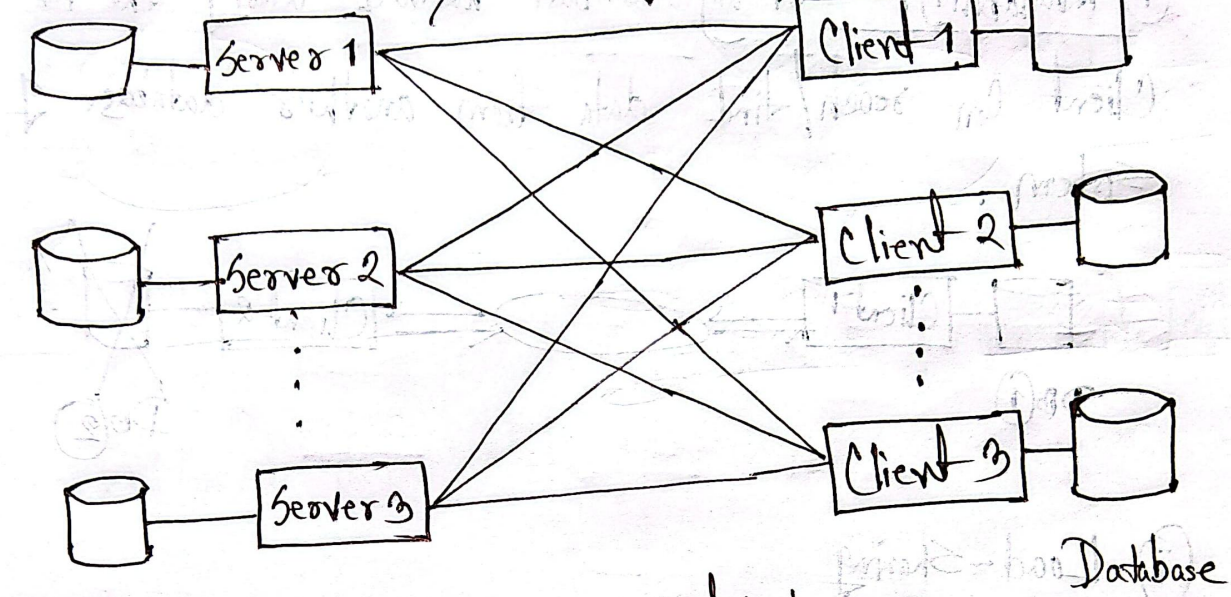
Collection of multiple database, logically related database distributed over a network in Distributed DB.

Design Issues in DD:

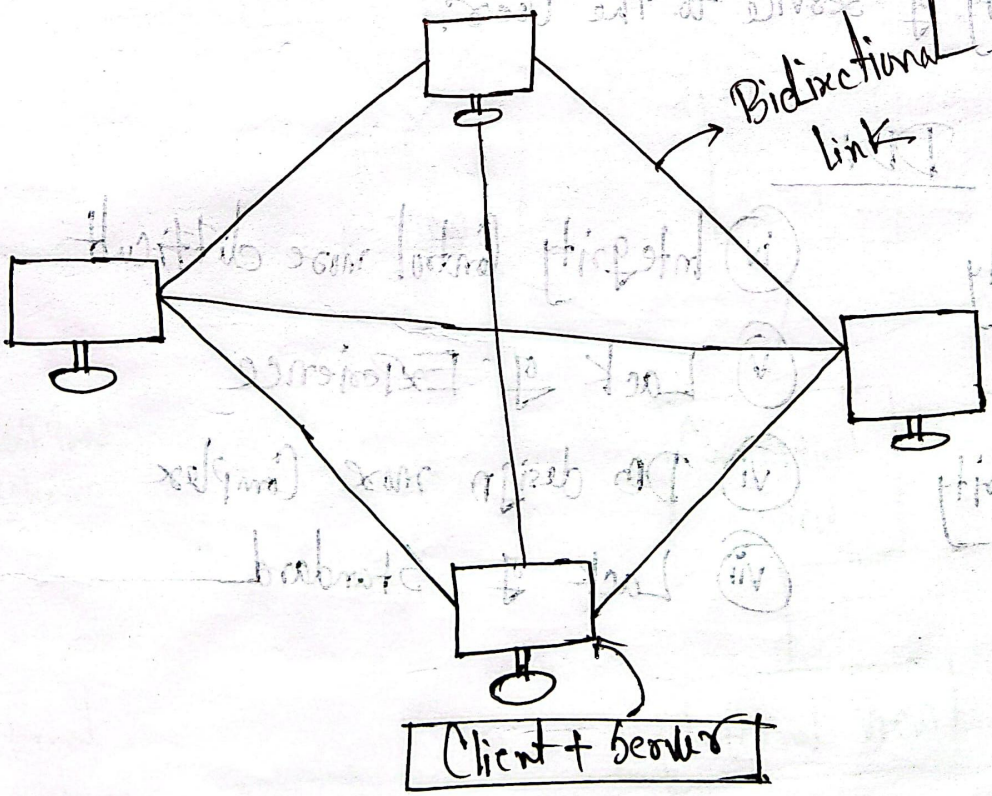


Architectures of DD :

2 Types
- Multiple Client / single Server
- Multiple Client / Multiple Server



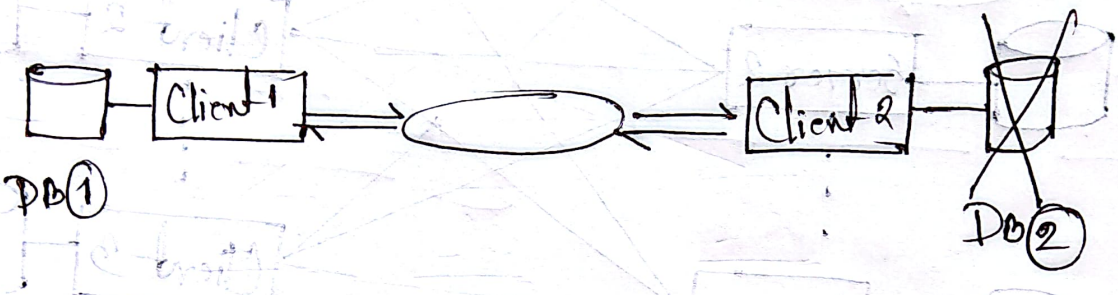
⇒ Client-Server Architecture



⇒ Peer-To-Peer Architecture

DD Functional Goal: (1(b) | Spring'22)

(i) **Availability:** If any database harmed among the DD system Client can reach/find data from another database of the system.



(ii) Load-Sharing

(iii) Resource Sharing

(iv) Quality of service to the User.

DisAdv. of DD:

(i) Complexity

(ii) Cost

(iii) Security

(iv) Integrity Control more difficult

(v) Lack of Experience

(vi) DB design more Complex

(vii) Lack of Standard

Types of DD:

(i) Homogeneous:

→ All sites use same DBMS Product [e.g: Oracle]

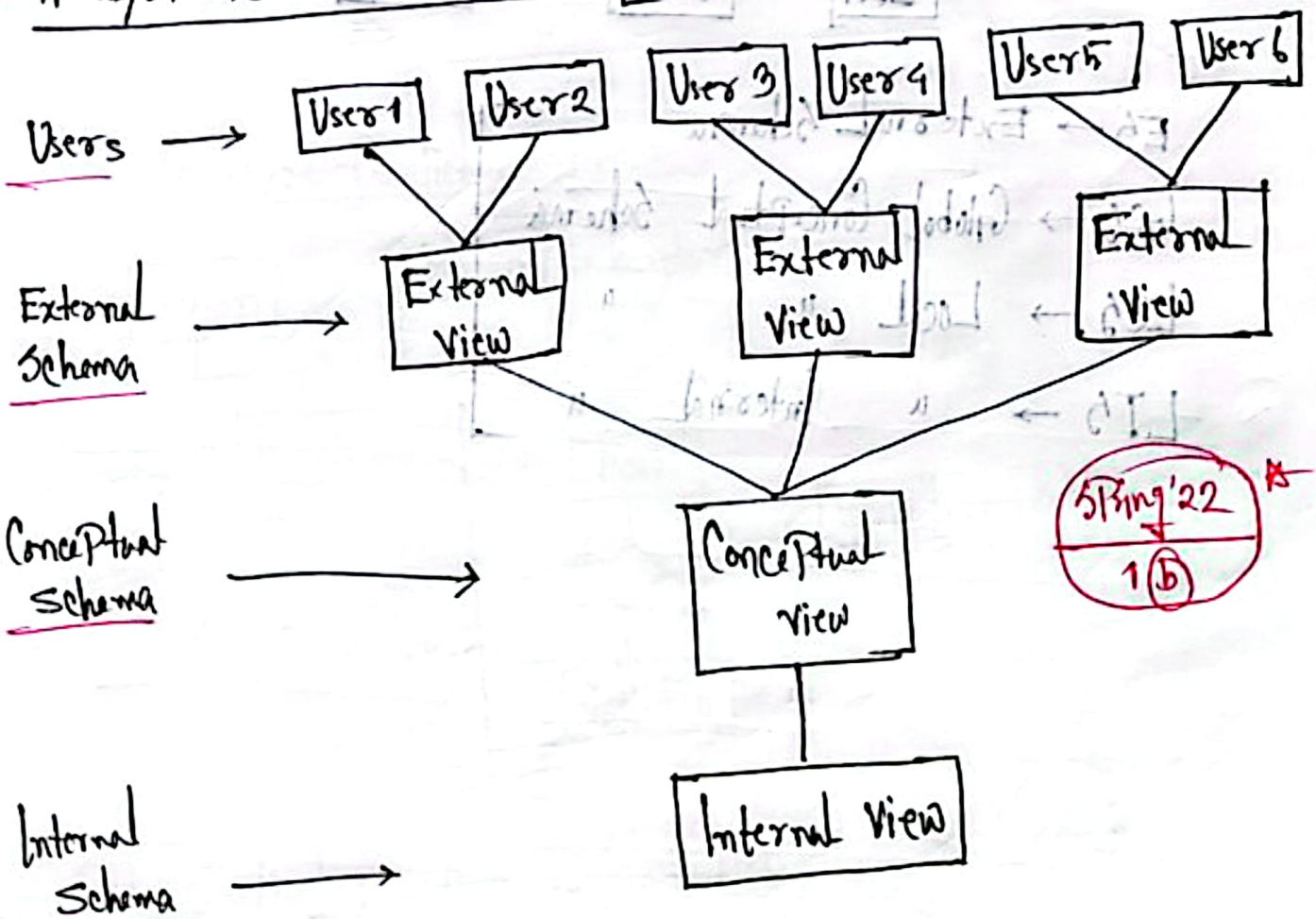
→ Fairly easy to design and manage.

(ii) Heterogeneous:

→ Sites may use different DBMS Products [e.g: Oracle/sql]

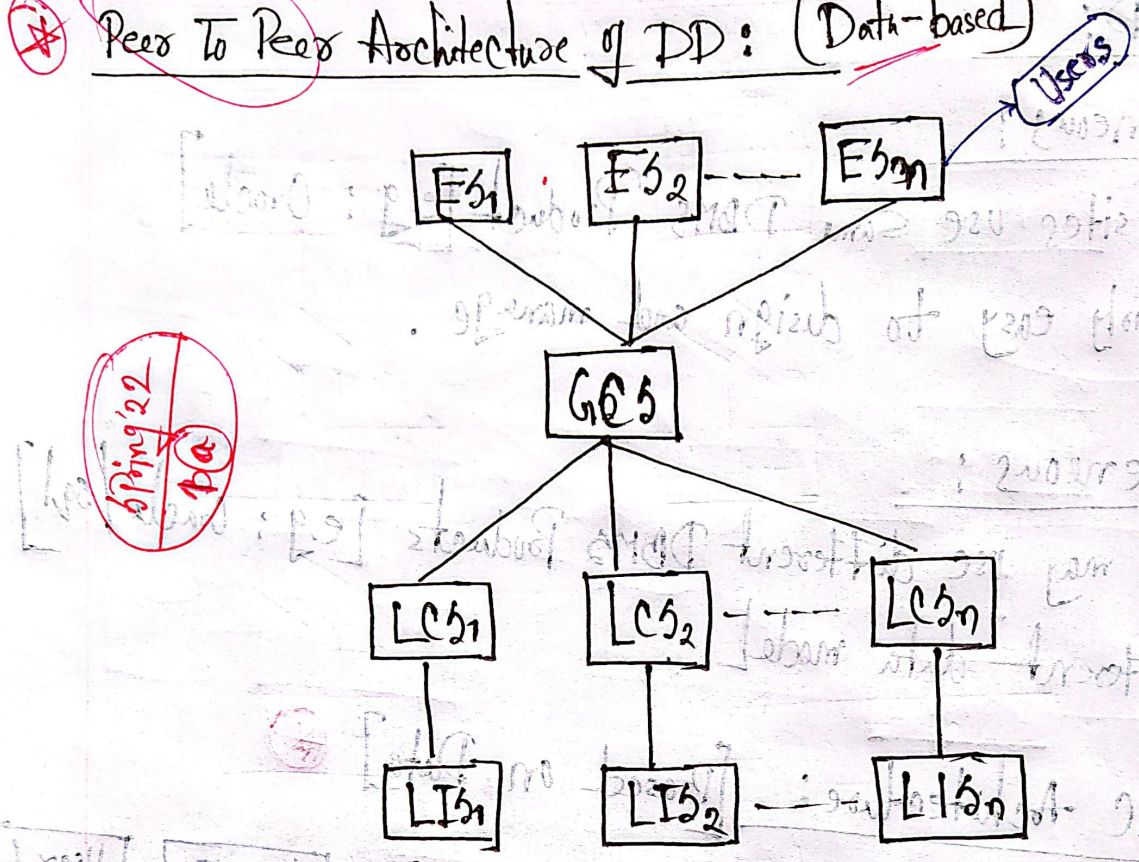
→ Different data model.

ANSI/SPARC Architecture: [Based on Data] ★



Spring '22
1 (b)

Peer To Peer Architecture of DD: (Data-based)



ES → External Schema

GCS → Global Conceptual Schema

LCS → Local " "

LIS → " Internal "

Segment - (2)

(iv) Conflict Analysis : SP'22 2(b)

(i) Name Conflicts :

→ Homonyms e.g : Product → Price (Buy Price)

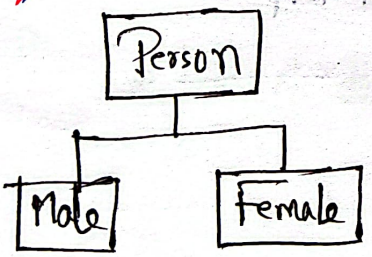
Product → Price (Sell Price)

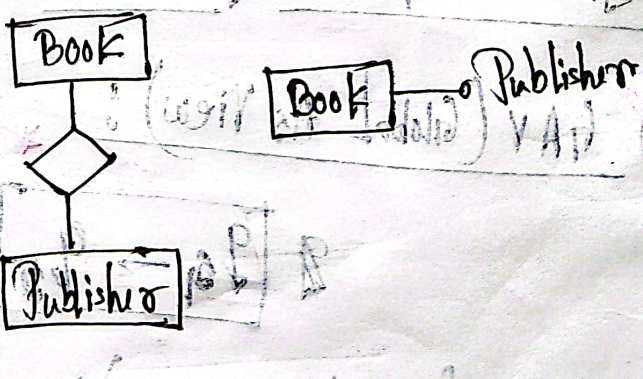
→ Synonims e.g : Department, Division

same meaning

(ii) Structure Conflicts :

Person → Gender

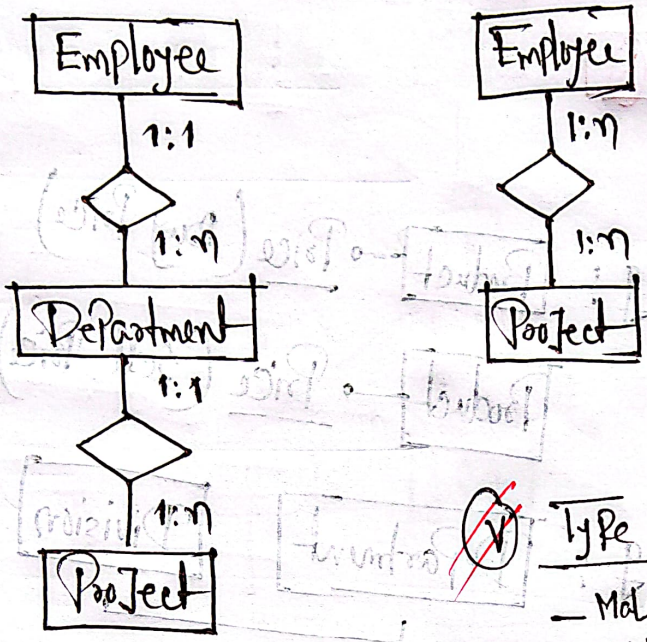




(vi) Data Conflict : - Currency (EURO, USD)

- Scale (Kilo/Pound)

(iii) Dependency Conflict:

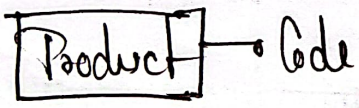


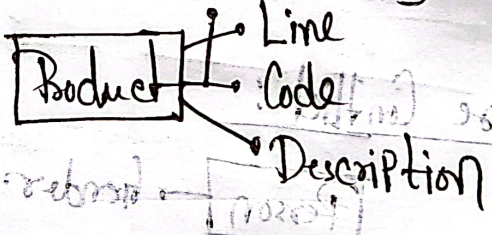
(v) Type Conflict:

- Male/female
- M/F
- 1/0

} For Gender attribute

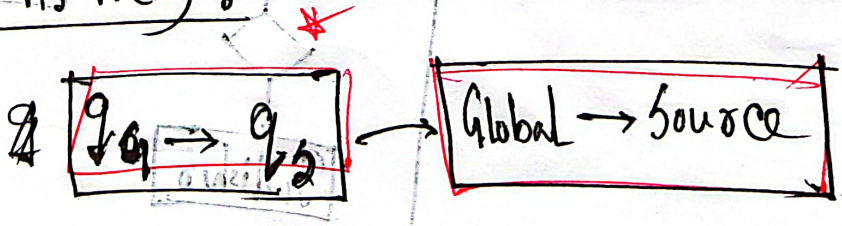
(iv) Key Conflict:



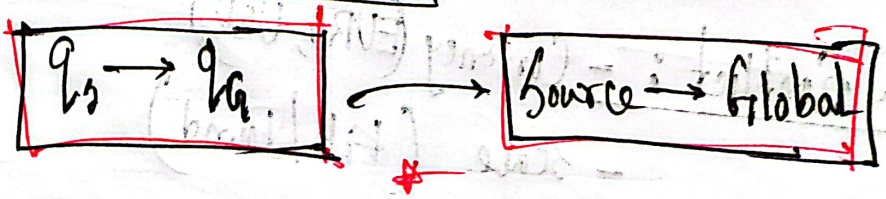


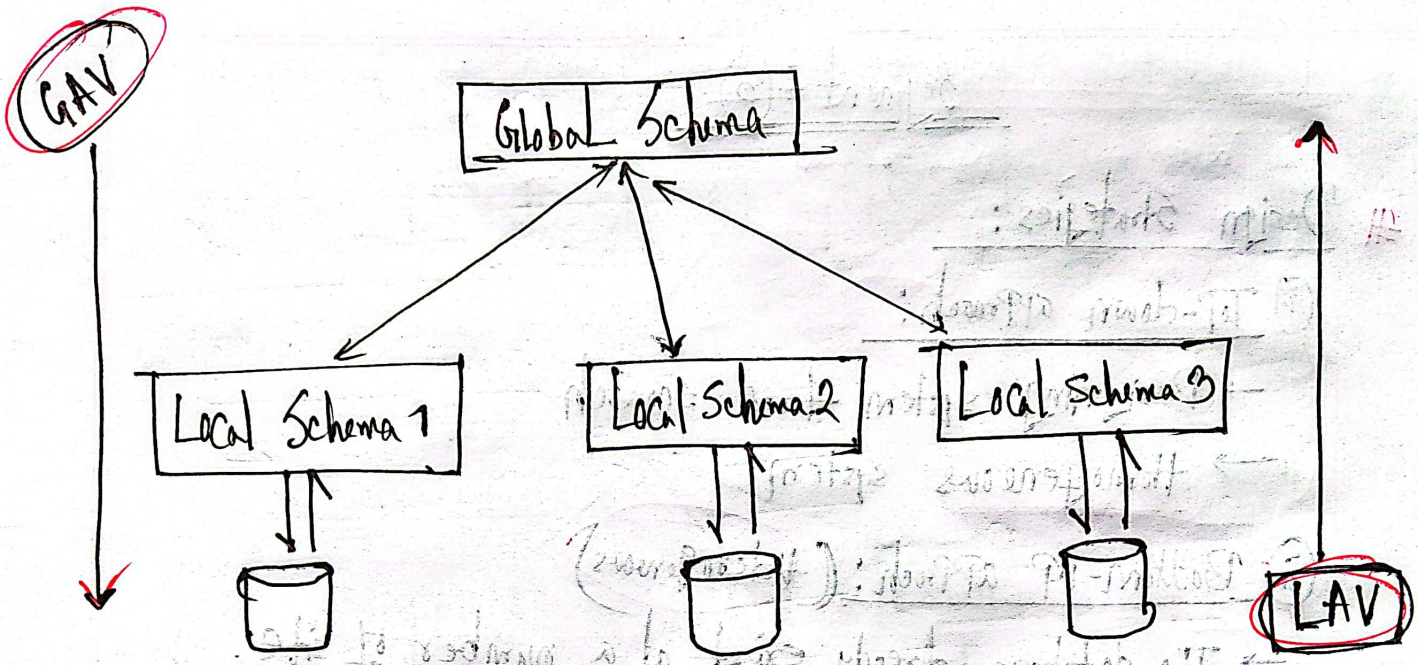
(vi) Logical View/Mapping:

(i) GAV (Global As View):



(ii) LAV (Local As View):





Characteristics of DD: 3P'22 | 1(b)

(i) Make data accessible by all units.

(ii) Store data close to where it is most frequently used.

(iii) Data are stored at several location.

(iv) Location of data is unknown to client

Segment - (3)

Design Strategies:

(i) Top-down approach:

→ Designing system from scratch

→ Homogeneous system

(ii) Bottom-up approch: (Heterogeneous)

→ The database already exist at a number of site.

→ The database should be connected to solve tasks.

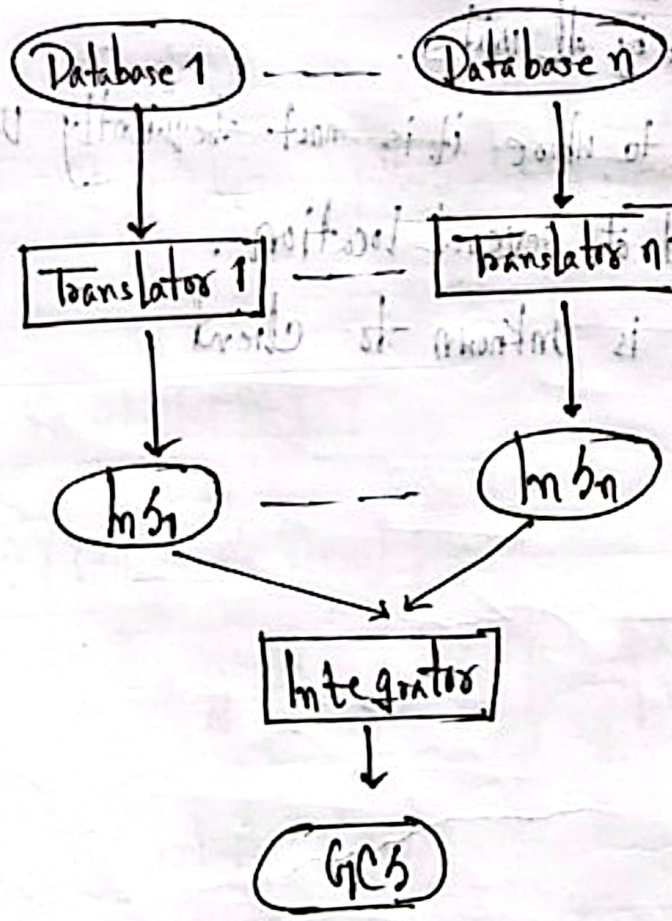
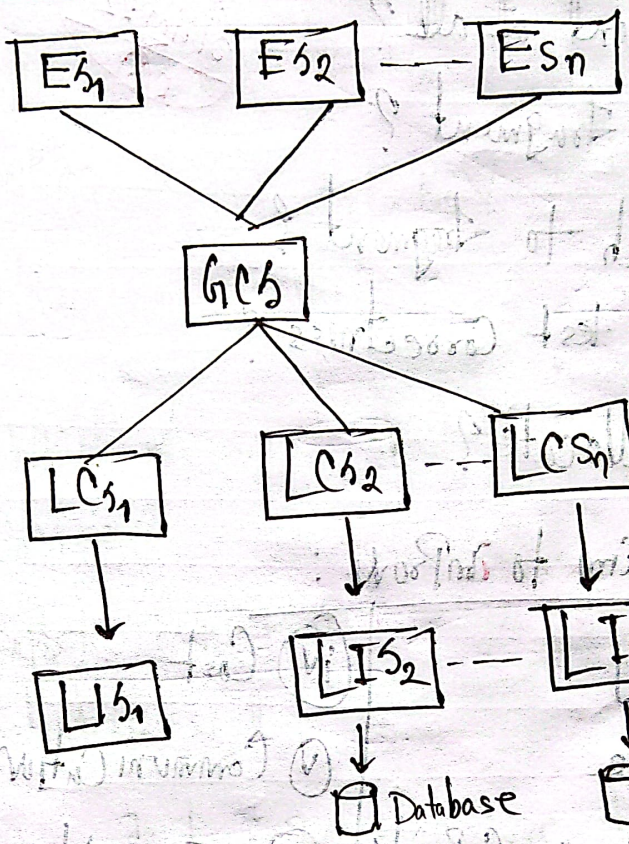


fig: Bottom-up approach

Top-Down : (Homogeneous)



Design Strategies :

Two main aspects —

(i) Fragmentation

→ Relation maybe devided into a number of sub-relations.

(ii) Allocation & Replication : *

→ Each fragment is stored at site.

→ Copy of fragment may be maintained at several sites.

Distributed Design issue: (4)

(i) Why fragment at all?
(ii) How to fragment?
(iii) How much to fragment?
(iv) How to test Correctness?
(v) How to allocate?

Fragmentation aim to improve:

(i) Reliability
(ii) Performance
(iii) Balance Storage Capacity
(iv) Cost
(v) Communication Cost
(vi) Security

Types of fragmentation: (2)

(i) Horizontal
(ii) Vertical
(iii) Mixed / Hybrid

Correctness Rules of Fragmentation:

(i) Completeness → $R = R_1, R_2 \dots R_n$

(ii) Reconstruction → $R = R_1 \nabla, R_2 \nabla \dots \nabla R_n$

(iii) Disjointness → ~~$R \cap R$~~

Replication: Which fragment shall be stored as multiple copies?

→ Complete replication

→ Selective "

Spring '22 2(a)

Allocation: On which site to store the various fragments?

5P'22|4b

Reference Model: Find out / sort data depends on different [reference / Criteria / entity / Condition] :

Aut '21 | 1(a)

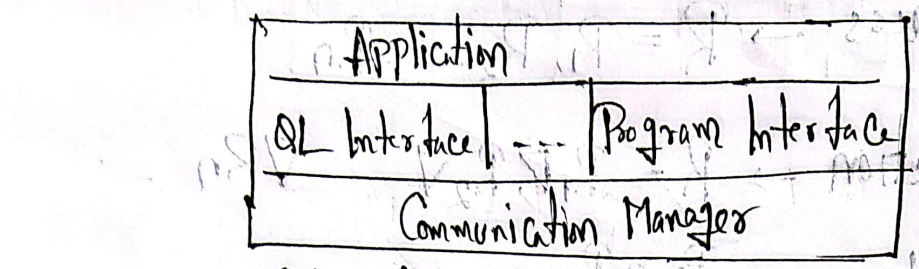
3 different approach of Reference Model:

(i) Component-based

(ii) Function-based [ISO/OSI Arc.]

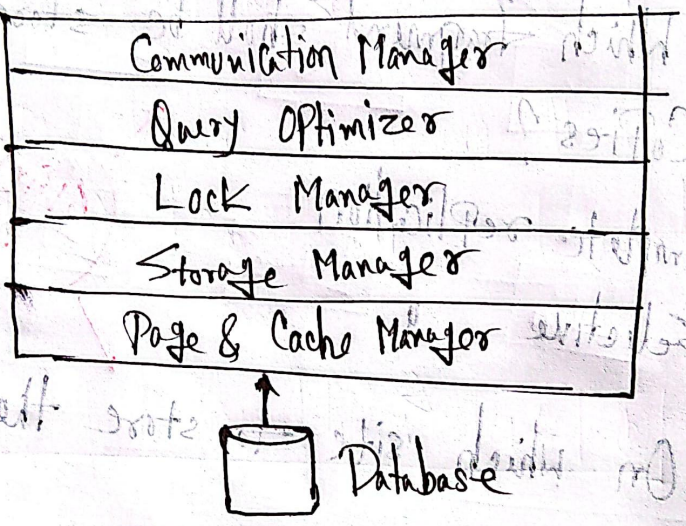
(iii) Data-based [e.g: ANSI/SPARC Arc]

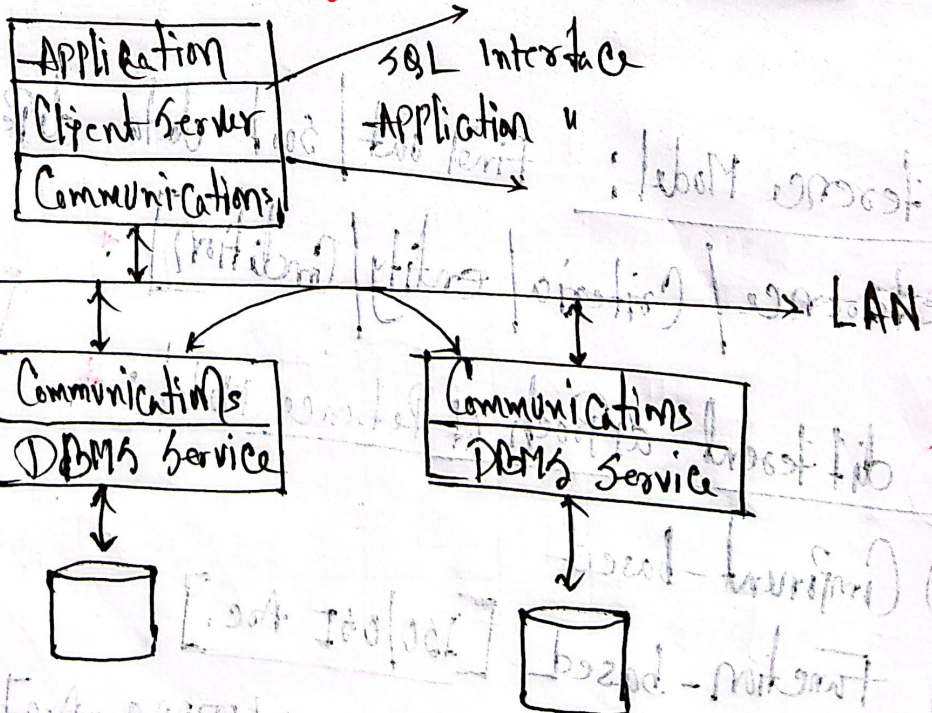
Client-Server Arc: (i) Many client/Many server (ii) Single server/Many Client.



SQL Query ↓ ↑ Result Table

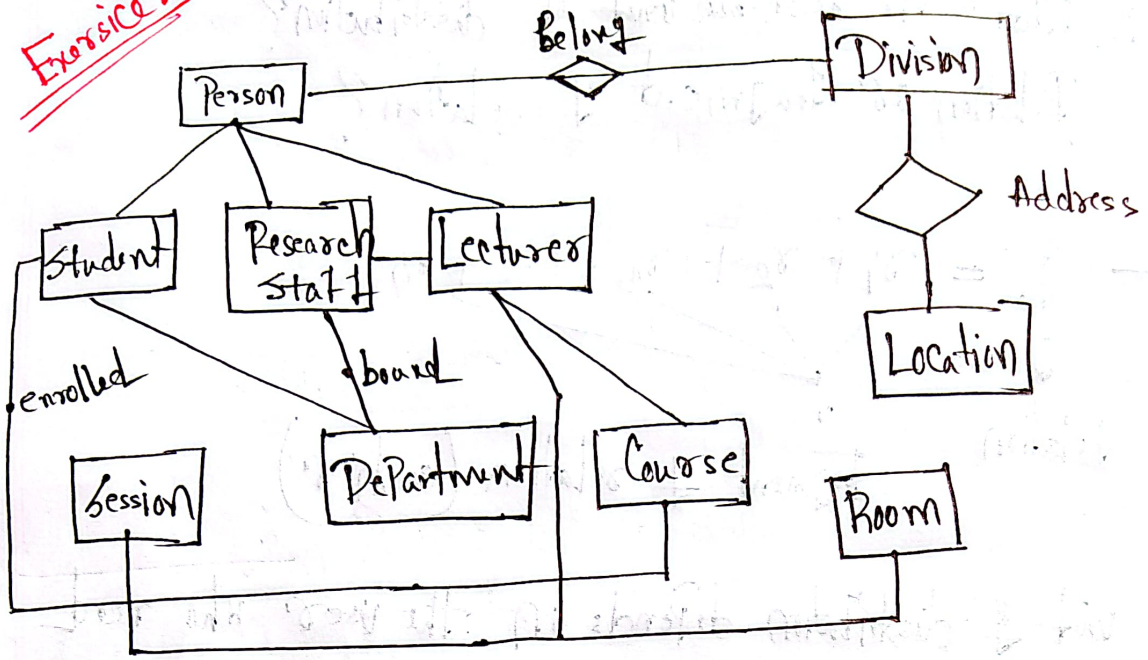
Data-based





Component-based

Exercise:



Reverse-Eng of DD:

[Logical Schema → ERD/Conceptual ERD]

Correctness Rules of Fragmentation:

$R = r_1 + r_2 + r_3 \dots r_n$

(i) যে কোনো relation (R) কে ছোট ছোট relation (r) এ ভাগ করা

(ii) Correction করা

(iii) পুনরায় জোড় লাগানো এবং R পাওয়া — [using DNF/CNF]

Fragmentation — Definition, Type → Definition, Example

SP22 | 2(a)

Question: What is the resonable unit of distribution? Relation or fragment of relation?

Answer:

$R = r_1 + r_2 + r_3 + \dots + r_n$

↓ Relation

Fragment of relation (each r_i)

Resonable unit of distribution depends on the users who need the data. It's can be a relation or fragment of relation.

Horizontal and Vertical fragmentation: SP22 | 2(b)

No	Name	Loc	Gender
1	A	Ctg	M
2	B	Dk	F
3	C	Syl	M
4	D	Ctg	F
5	E	Dk	M

No	Name	Loc	Gender
1	A	Ctg	M
2	B	Dk	F
3	C	Syl	M

No	Name	Loc	Gender
4	D	Ctg	F
5	E	Dk	M

Horizontal Fragmentation

NO	Name	Loc
1	A	Ctg
2	B	DK
3	C	Syl
4	D	Ctg
5	E	DK

NO	Gender
1	M
2	F
3	M
4	F
5	M

Vertical Fragmentation

Data Integration Problem: Problem while Joining two table for different types of Conflicts.

Fragmentation: It's a Process of dividing the whole or full database into various sub-tables or sub-relations so that the data Can be stored in different systems. The small Pieces or sub-relations or sub-tables are Called fragmentation.

FAHIM REZA | C211046
fahim.bdctg51@gmail.com