Data Base management

Data :- Data refers to a collection or raw facts or information that can be processed to derive meaning or knowledge. in easier term we can say any fact that can be stored
eg :- xyz, 1, nasik

Information :- processed form of raw data or data is called information.

ex

name	age
Abhishek	21

Data — collection of raw facts where clear meaning cannot be extracted

Information — Processed data from which meaning can be extracted.

Database
collection of interrelated data is called database
- it can be stored in table
- it can be of any size

eg multimedi DB or student DB

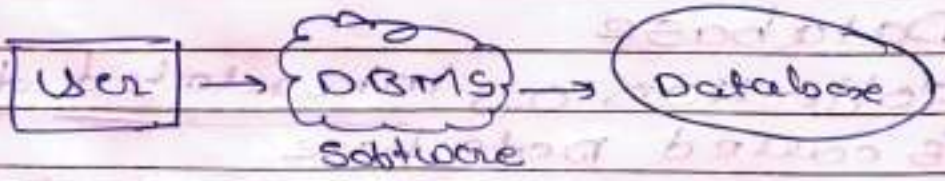
Page No.
Date

Database is controlled by a software called Database management system e.g MySQL, Oracle.

file system
it is an OS approach for organising and storing data on storage units like hard drives

major disadvantage
- data redundancy → repetation (DRY)
- Poor memory utilisation
- Data inconsistency → inconsistent
- Data security
- slow data retrival

DBMS
Users can access database and perform CROD Operation on the data with the help of a software which is called DBMS in a efficient and secure way



Page No.
Date

Applications
① Schools and colleges ② Banks ③ airlines.

Types of Data Bases

① Relational Data bases (RDBMS)
- most widly used
- collection of data item having relation between them
- in the form of Tables contain rows and colony
eg My SQL, ORACLE DB.
→ query performed using SQL.

② NON Relational Data bases (NO SQL)
- Stored in the form of key value pair
- eg mongoDB

③ Object orient Database
④ In memory Database
⑤ In-Memory Database
⑥ Time Series Database
⑦ Spatial Databases
⑧ Multi media Database.
⑨ columnar Database
⑩ XML Databases
⑪ New SQL Databases
⑫ Block chain Databases. → decentralise way of storing data.

MySQL = B Trees

Page No.
Date

#Need of DBMS

DBMS plays a vital role for all scales in effectively managing data ensuring data security and accuracy and supporting essential decision making process

Advantage

- Data security - DBMS implements security mechanism that regulate access to sensitive information.
- Data redundancy and inconsistency removal
- Data integrity
  DBMS guarantees data integrity by enforcing rules and constraints that prohibit the entry of incorrect or inconsistent data
- Database scalability
- Database Data Abstraction.

Disadvantage

- cost - managing and maintenance is costly

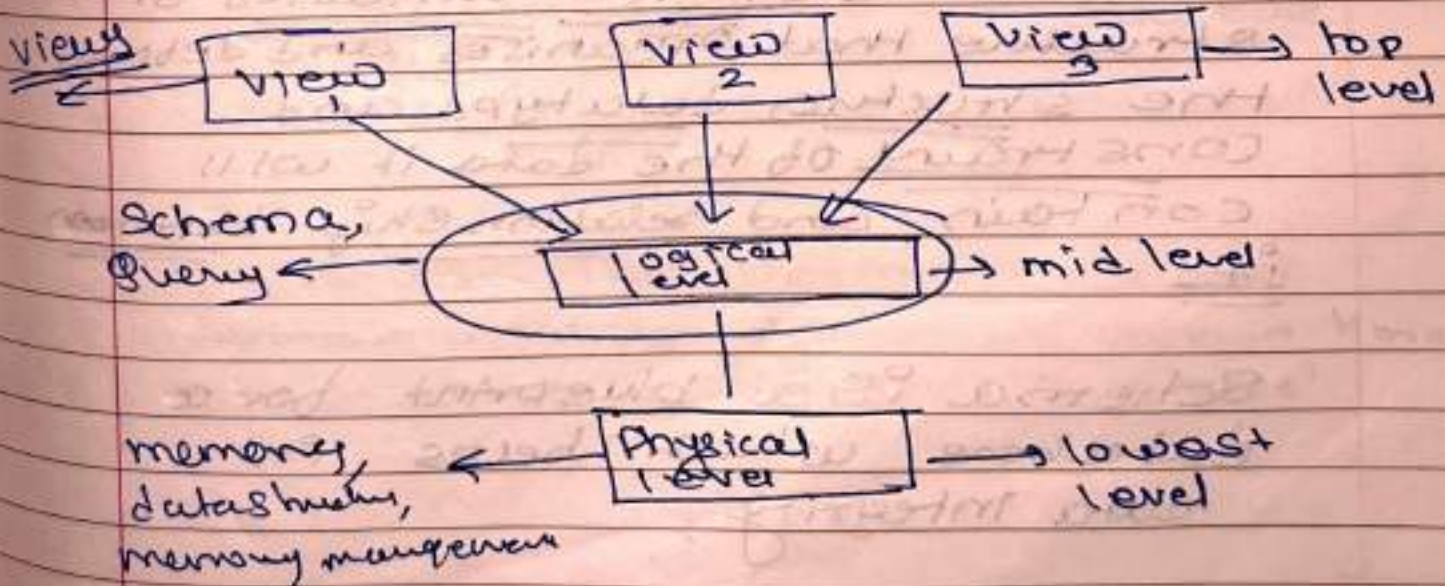
- ~~Scale project~~ Scale project - making right choice of DBMS for right project at right time.

- vendor - lock in -
once you have choosen a type of DBMS it is difficult to shift to another.

Data Abstraction.
Database systems are built with complex ways of ~~gathering~~ organising data. To make it easier for people to use the database. The creators hide the complicated stuff that they dont need to worry about. This hidding of unnecessary things from user is called data abstraction.

making sure that the user is interacting with data at higher level only

There are three levels of abstraction



→ How data is stored

Page No.
Date

① Physical level:- This is a lowest data abstraction it describes how data is actually stored in DB. You can get the complex data structure details at this level.

→ what data is stored

② logical level:- The middle level of 3 level data abstraction architecture. It describes what data is stored in data base

→ How data is viewed

③ view level:- Highest level of data abstraction this level describes the user interaction with data Base system

Schema AND Instance

A schema is a logical container or structure that organises and defines the structure, datatype and constraint of the data it will contain and relationship between it

Schema is a blueprint for a database which helps in data integrity,

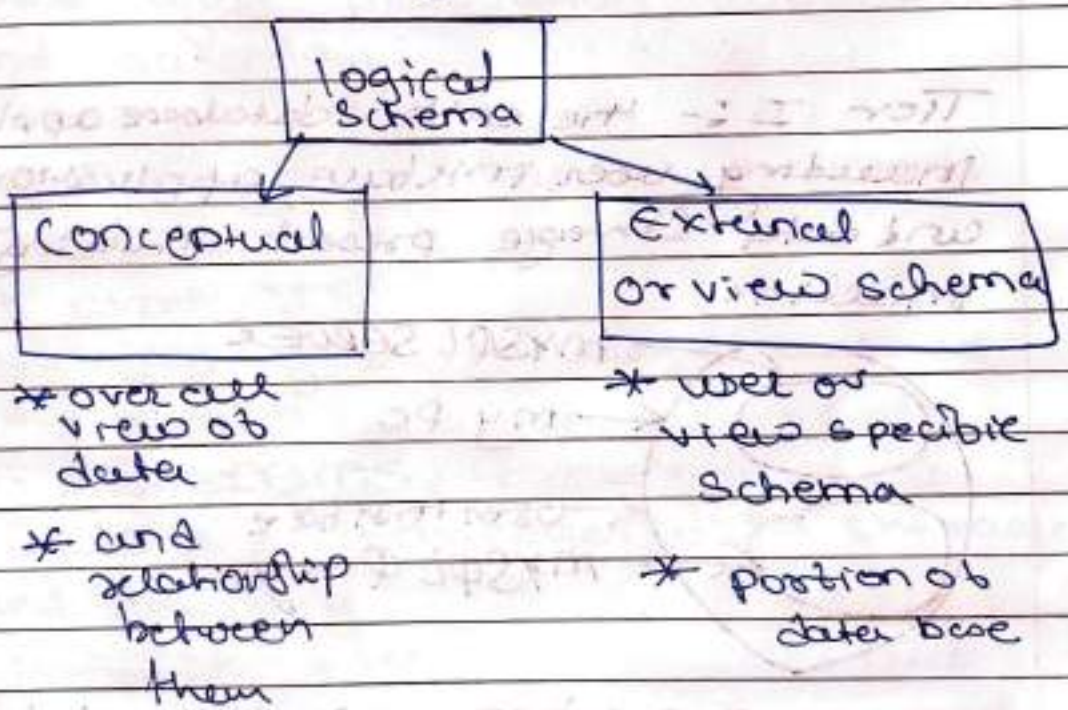
Page No.
Date

where is my data stored and How

1) Physical Schema :- How data is stored on the underlying hardware, including storage formats, file organisation, indexing etc.

what kind of data is stored:

2) Logical Schema :- defines the databases structure from a logical or conceptual perspective without considering how the data is physically stored.



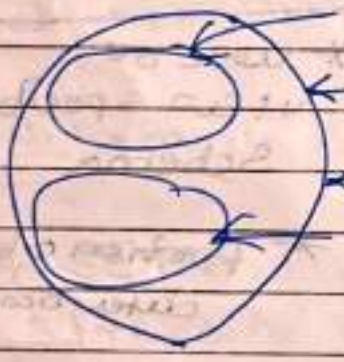
Instance :- The information present within a database at a specific point in time is referred to a databases "instance"

DBMS Architecture

It defines how the various components of the system work together to store, manage and retrive data efficiently

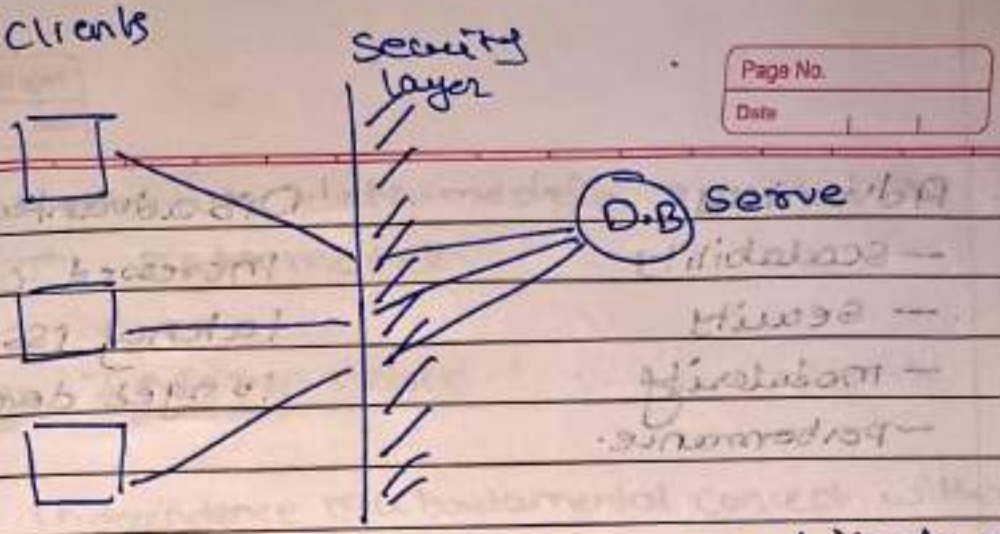
Types:-
Tier I
Tier II
Tier III
} depends on use case and specific need of application

Tier I :- the entire database application, including user interface, application logic and data storage present on a single machine



e.g when we install a database on our system and use it to machine SQL queries.

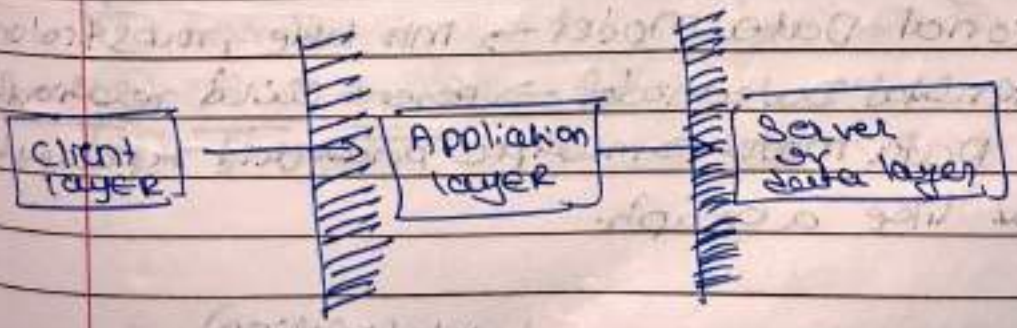
Tier II :- In Tier 2 the presentation layer / user interface runs on a client and data is stored on server



also called client server architecture.

Tier III :- most used
It separates the application into three layers presentation, application and data layer

presentation :- it handles the user interface
eg client, UI
application layer :- it manages business logic
eg ~~server~~ DBMS, logical rules
data layer :- it manages data storage and processing
eg database sever



Advantage	Disadvantage
- scalability	Increased complexity
- security	latency issues
- modularity	longer development time
- performance.	

DATA MODEL

A data mobodel within a DBMS serves as an abstract representation of how data gets structured and organised within a database

Basically a diagram or a high level view of database.

datamodel → is high level view of DB while Schema is a implementation of data model.

Types

Entity Relationship model (ER Model) — deals with entity and real world obj. and their relation ship
Relational Data Model → into table, rows & column
Hierarchical Data model → parent child relationship
Networ Data model → multiple parent child repulsionship
Structure like a graph.

table (relation)
row (tuple)
column (attribute)

Object oriented data model

NoSQL Data model

Data Independence

Data independence is a fundamental concept within database design and management, emphasizing the distinction between the logical and physical dimensions of data storage and administration in a database management system (DBMS). This principle yields various benefits, such as enhanced flexibility, heightened security and simplified maintenance.

if at any level if there is any change it won't affect the other levels

Essential components of Tables

1. Row / Tuple / record
2. cardinality ⇒ No of rows in table
3. column / Attributes
4. Degree ⇒ No of columns

Page No.
Date

Contrain - rules or conditions a data in the column must satisfy

Keys :- data identifier → primary key, foreign key

Views in DBMS

View is a virtual table that is derived from one or more underlying tables This means that it doesn't physically store data but rather provides a logical representation of data

It is a virtual table

Keys

Keys in DBMS make sure data integrity uniqueness and quick retrival of information Key is a attribute in a table

Types
Candidate Key
Primary Key
Foreign Key
Super Key

① Candidate key :- A candidate key refers to a group of attributes capable of uniquely identifying a record within a table. Among these one is selected to serve as the primary key

eg student-id, Adhaar-NO, ROLLNO.

② Primary key :- A primary key is a key which uniquely identifies each record in a table. It ensures that each tuple or record can be uniquely identified within the table. it is always Unique + NOT NULL

③ Foreign key :- A foreign key is a field in a table that refers to the primary key in another table. it establishes a relationship between two tables.

referenced table ← references ← referencing table

Base table → having PK

→ having FK

→ basically it means data is same across

referential integrity ensures that the relationships between tables remain accurate, consistent, and meaningful within a relational database.

data base and maintain the relationship.

will help referential integrity

Page No.
Date

on cascade delete
on CASCADE Update

or we can set null values.

insertion in referencing table may or may not cause violation

Deletion ~~in~~ in referencing table ~~will~~ will not cause violation

Updation also same like deletion until and unless foreign key is updated.

Integrity constraints in DBMS.

Integrity constraints help to ensure that data remains secure and meaningful throughout its life cycle.

Types of Integrity constraints

- Domain Integrity constraints
- Entity Integrity constraint
- Referential integrity constrain
- Key constraint
- Null constraint
- Unique constraint
- Default Constraint

Super key

It is a set of one or more attributes (columns) that can be uniquely identify a tuple (row) in relation table

Superset of any candidate key

A super key becomes a candidate key if it is minimal (No proper subset of it can uniquely identify a tuple)

Student → { Sid, Sname, Sphn, Sadhaar }

C.K → { Sid, Sph, Sadhaar }

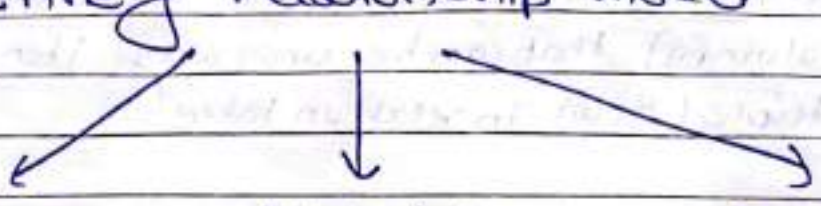
S.K → atleast one C.K + optional 1 extra attribute

P.K → { Sid }.

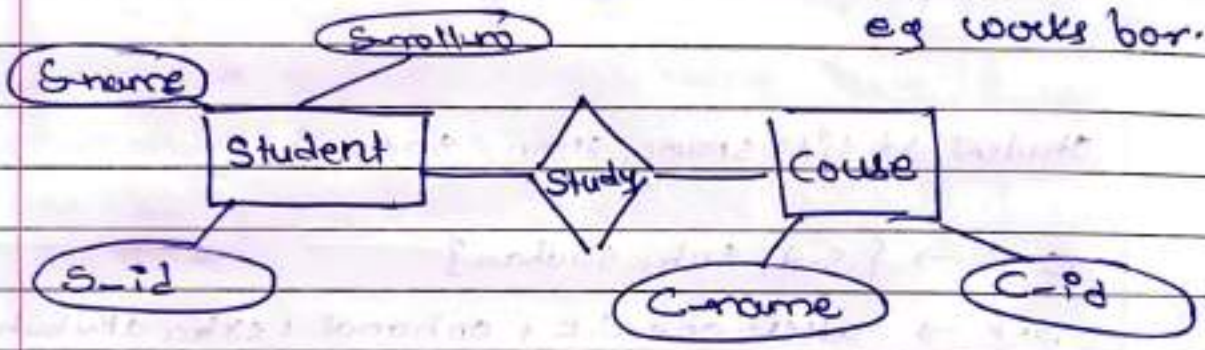
minimal superkey is a P.K

ER MODEL

⇒ Entity Relationship model



Entity	Attributes	Relationship
Things/objects	Properties of entity	association among entities
eg Person	eg age, name	eg works for.



The primary role is to offer a visual representation of a database architecture by illustrating the entities, their respective attributes and interconnections between them

ER MODEL

Entities	Attribute	Relationship
(1) Strong Entity	(1) Simple Attribute	(1) One to one
(2) Weak Entity	(2) Composite Attribute	(2) One to many
	(3) Single valued Attribute	(3) many to one
	(4) multivalued Attribute	(4) many to many
	(5) Stored Attribute	
	(6) Derived Attribute	
	(7) Composite Attribute	

Page No.
Date

Symbols used in ER model

Rectangle — entity

Ellipse — Attribute

Diamond — relation

Line — relation from attr to entity

Double ellipse — multivalued attribute

Double rectangle — weak entity

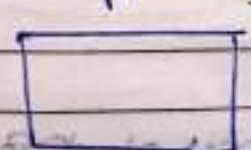
Entity

An entity is something from real world like student or university

Types

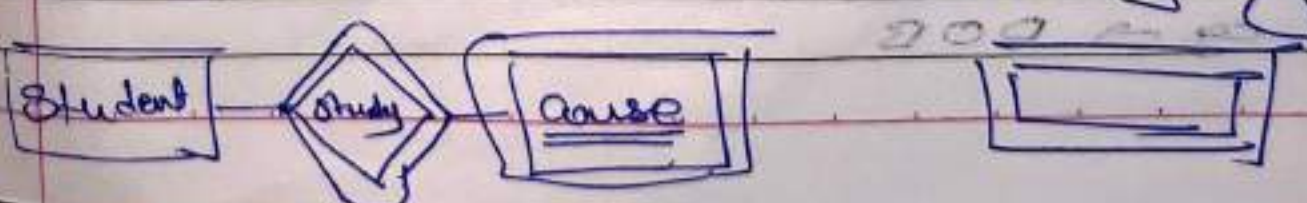
Strong

- independent
- exists alone
- has primary key



Weak

- dependent on strong independent
- does not exist alone
- does not have primary key



Attributes → columns or characteristics

① Simple Attribute

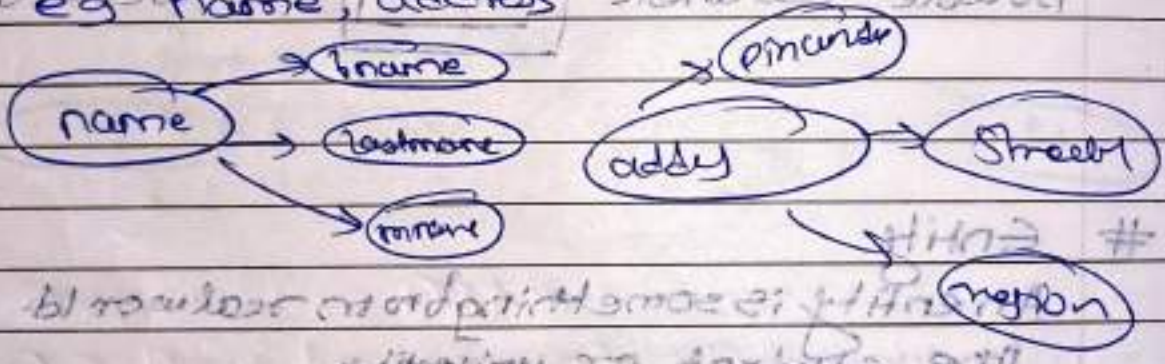
A simple attribute is a atomic and cannot be divide any further.

eg first Name, age

② Composite Attribute

A composite attribute is made up of several parts, where each part represents a part of whole attribute

eg name, address



③ Single valued Attribute

A single value attribute is an attribute that holds a single value

e.g Age

④ Multi valued Attribute

e.g phnumber, Siblings

⑤ Stored Attribute

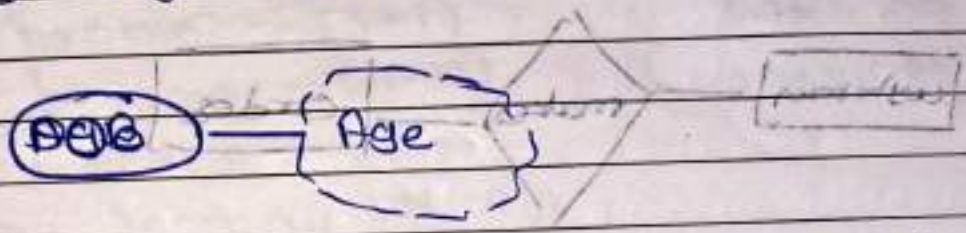
Attribute that is stored as a part of DB record

eg → DOB

⑥ Derived Attribute

An attribute that is derived from other attribute is a formula

e.g age from DOB



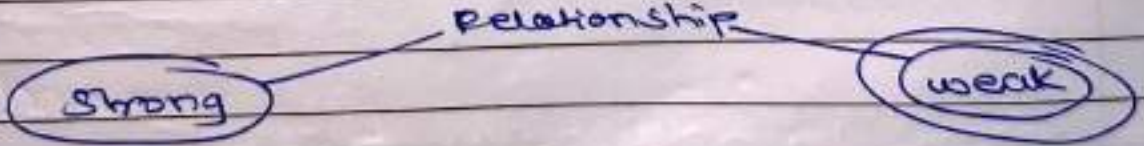
⑦ Complex Attribute

A complex attribute is an attribute that is made up of multiple smaller attributes

eg Name(composite) → fname
middlename
lastname

Relationships

Relationship in ER MODEL is the connection between entity based on related data



A strong relationship exists when two entities are highly dependent on each other and one entity cannot exist without other

A weak relationship on the other hand exists when two entities are related but one entity can exists without other

Degree in DBMS

→ refers to num of attributes or columns that a relation or table has

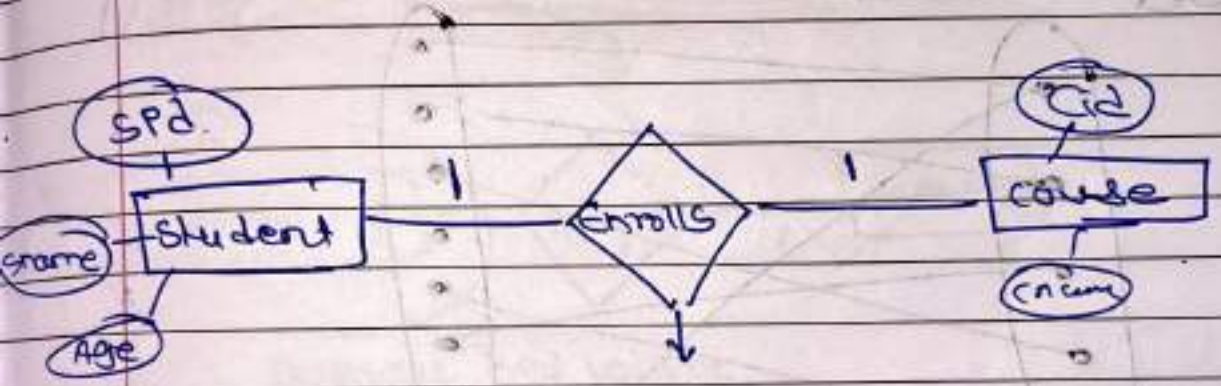
cardinality

no of rows

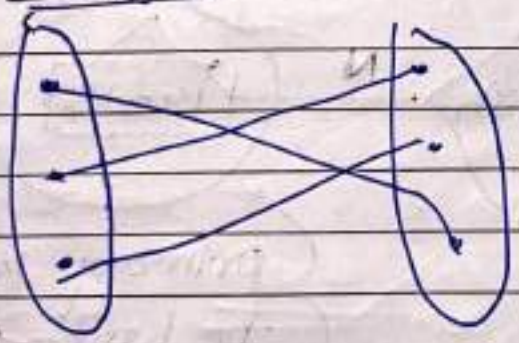
There are 4 types of relation ship

on the basis of cardinality

1. one to one (1-1)
2. one to many (1-N)
3. many to one (N-1)
4. many to many (N-N)

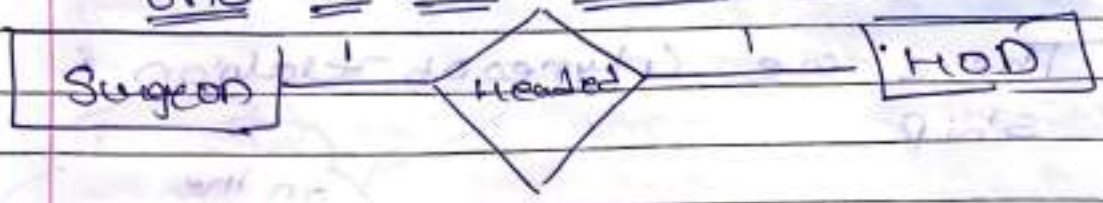


One entity form one entity set can form relation to one entity in another entity set



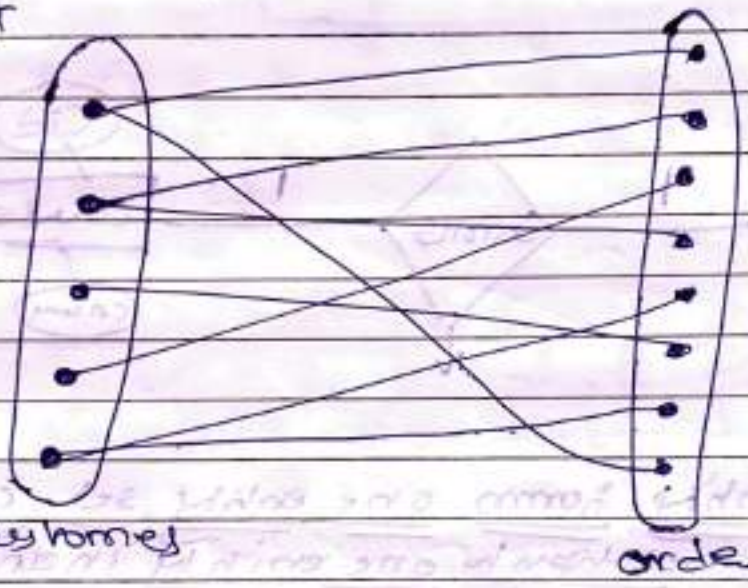
Page No.
Date

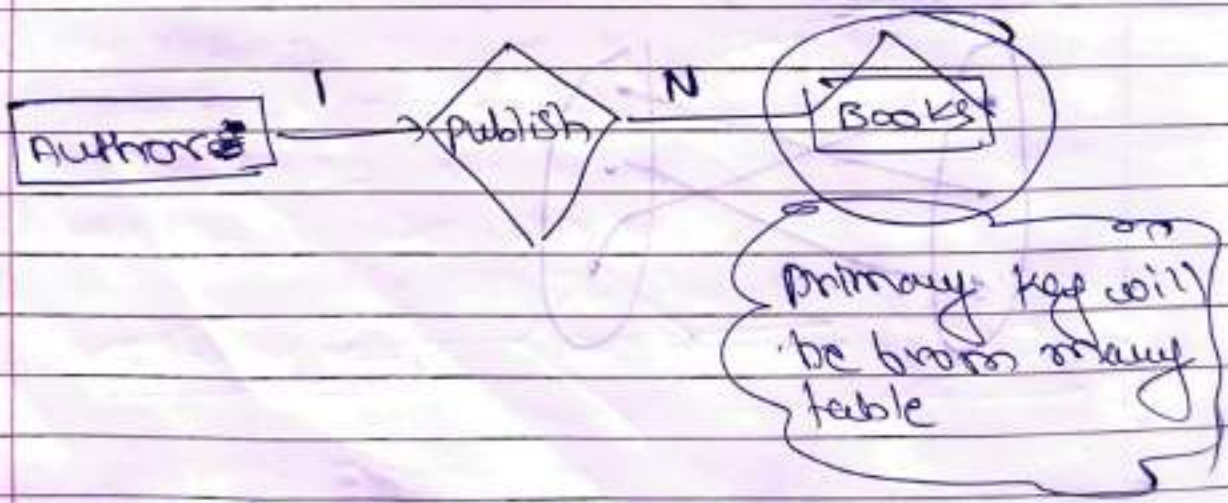
one to one relationship



one to many relationship

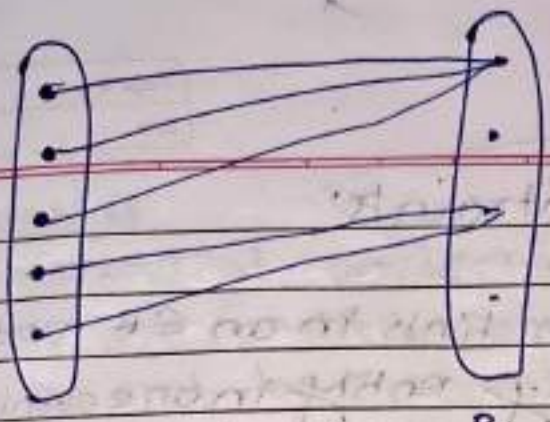
A database model where one entity from one entity set is associated with multiple entities in another entity set



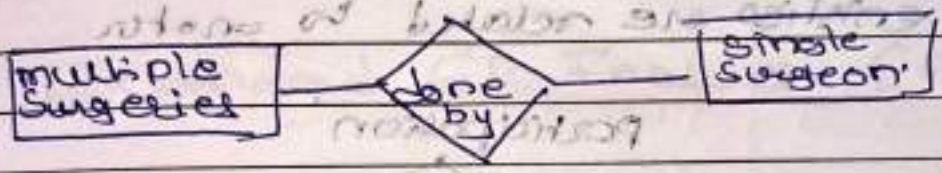


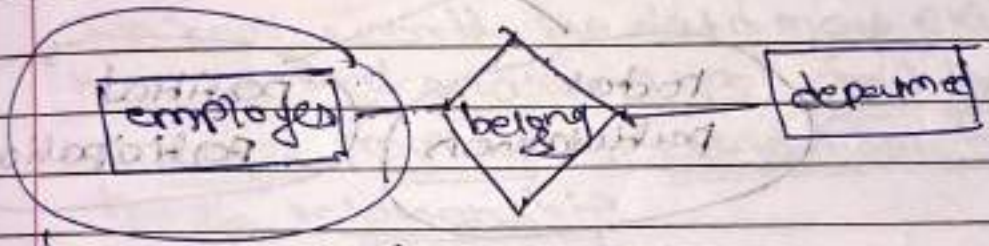
many to one relationship.

when many entities from an entity set belong or are associated with one single entity in another entity set is called many to one relationship



surgeons Department

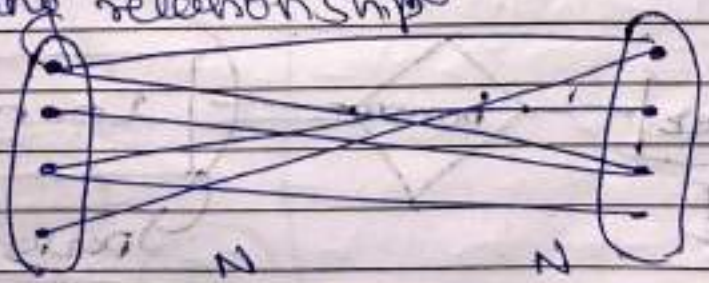




↳ primary key will be taken from any table

Many to many relationship

When multiple entities from one entity set are associated with multiple entities from another entity set are called many to many relationship



students — enrolls — course.

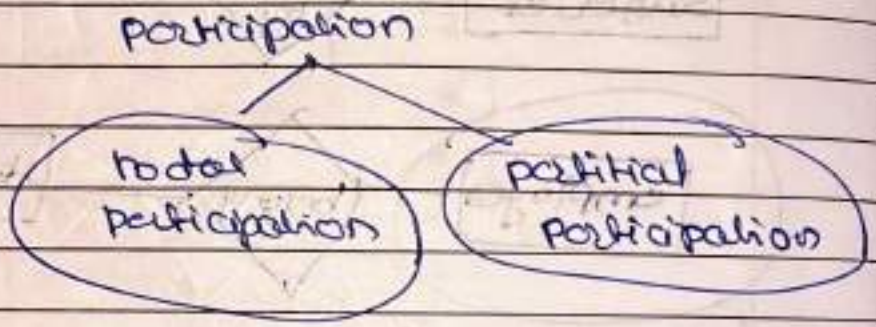
primary key (sid, cid)

Here reduction of tables is not possible

Participation Constraints

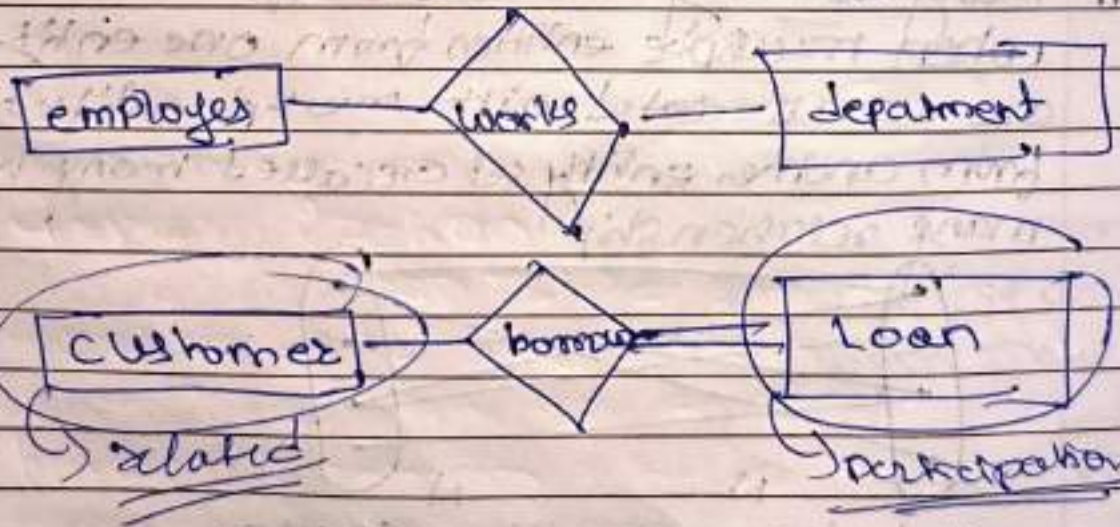
Participation constraints in an ER model define wether every entity in one group must be connected with at least one entity in another group.

→ how entities are related to another



total

In a total participation constraint each entity in a participation set must be associated with at least one entity in related entity set



should always participation

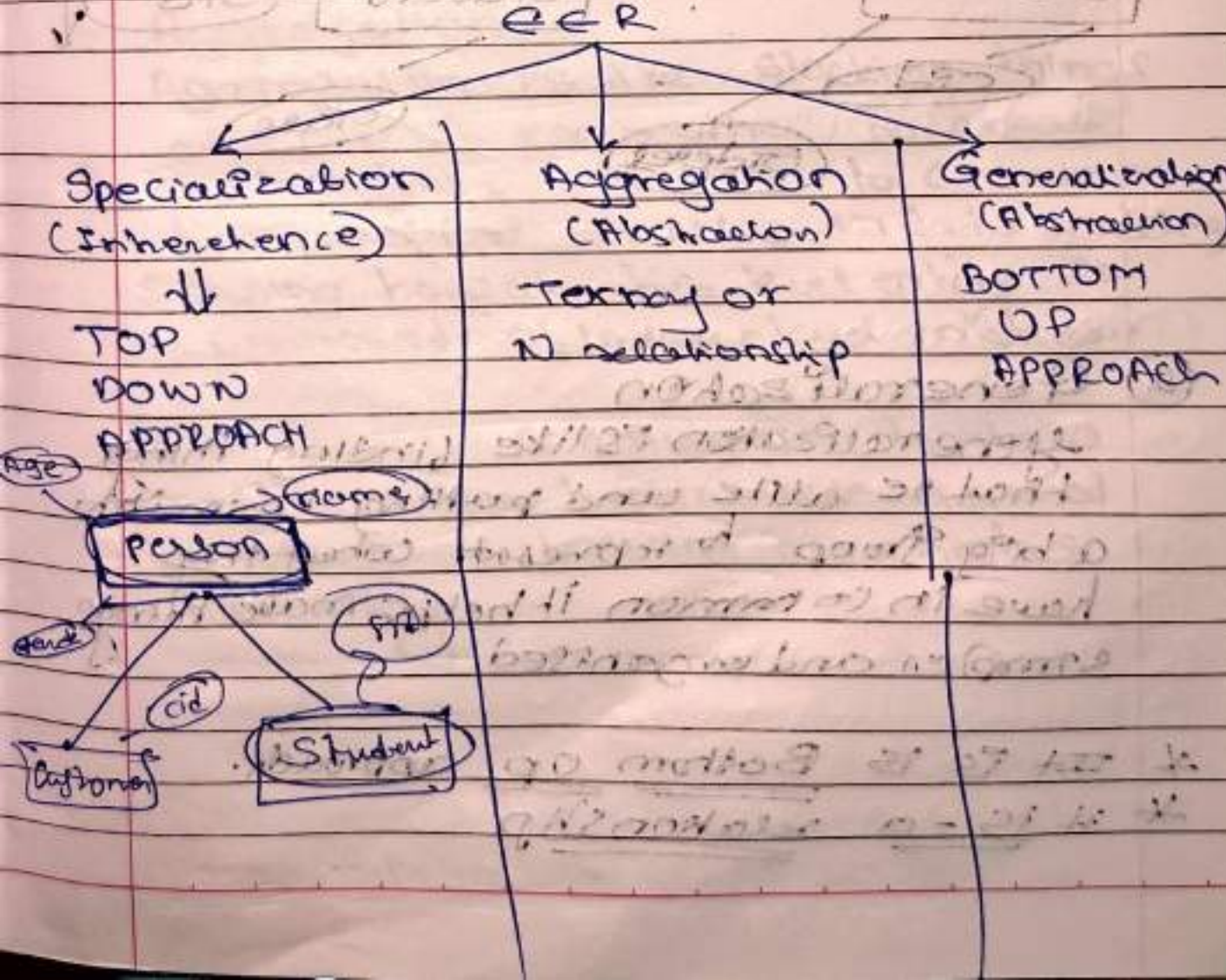
Page No.
Date

Partial

In a partial participation constraint, entities in the participating entity set may or may not be associated with entities in the related entity set

Extended ER Feature. (EER)

In real world the data may exibhit some hierarchical relationships and the EER model provides mechanism to represent these relationship

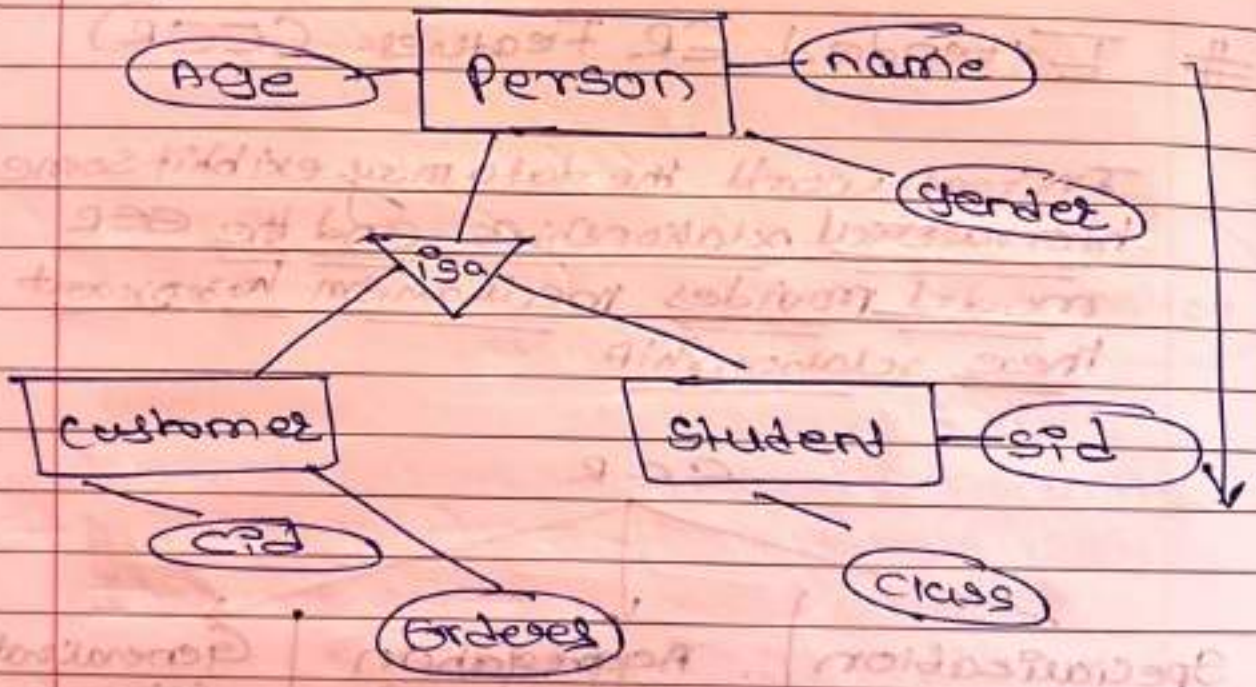


Page No.
Date

(1) Specialization

Specialization in the ER model is like categorizing entities based on common features

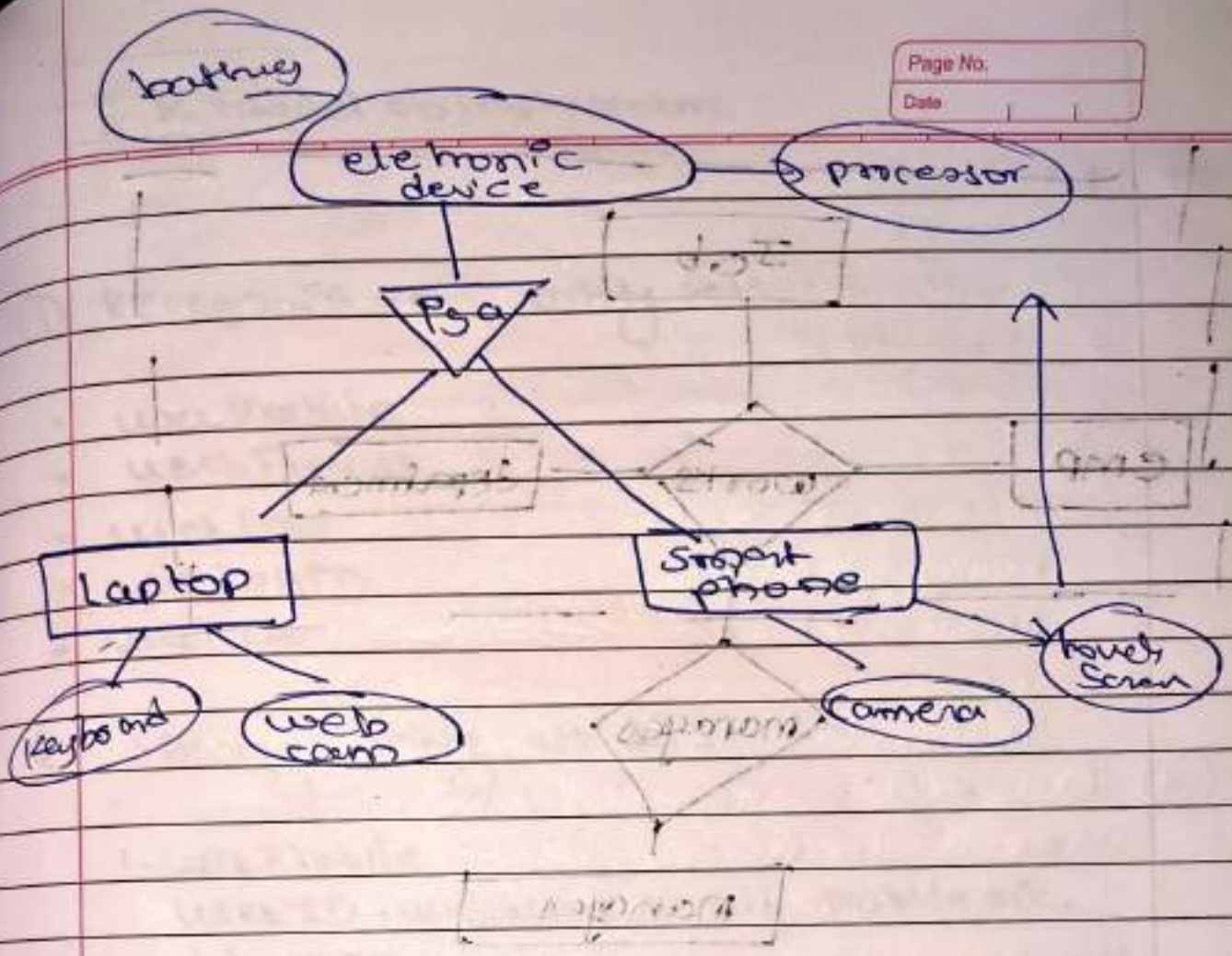
* It is a TOP DOWN APPROACH
* IS-A relation ship



(2) Generalization

Generalization is like binding things that are alike and putting them into a big group to represent. Coheyt they have in common it helps make things simpler and organised

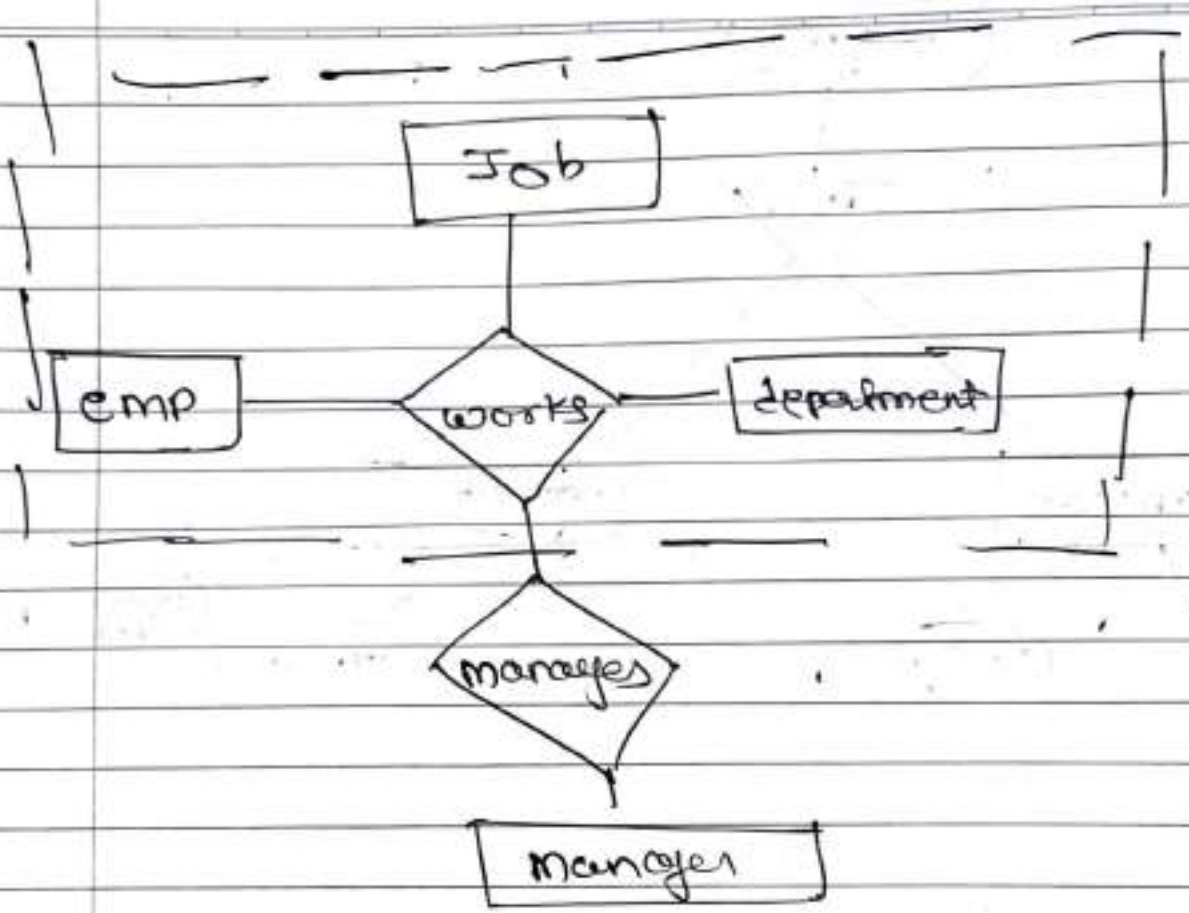
* It is is Bottom up approach.
* It IS-A relationship



Aggregation

Aggregation is like stacking things on top of each other to create a structure. it is used to create a hierarchical structure in data modeling showing how a higher level entity is composed of lower-level entities.

Abstraction is employed to view relationships from a more general perspective focusing on a highest level entity.



ER MODEL

Steps

1. Recognize entities
2. Specify entity characteristics / attributes
3. Discover connections / relationship (also constraints like mapping)
4. Define constraint type / cardinality
5. Construct ER D
6. Annotate relationships
7. Review and refine the model
8. Document model
9. Validate with stakeholder
10. Implement the database schema

ER model of instagram

Page No.
Date

① Recognise entity sets

- UserProbile
- UserFriends
- UserPost
- Userlogin
- Userlikes

② Specify entity attributes

1. UserProbile
   UserID, Username, email, mobile pic, dob, age
   UserID - PK
   Username - composite attribute
   email - single value attribute
   mobilepic - single value
   dob - single valued
   age - derived attribute

2. UserFriends
   FollowerID → PK
   Followername - single value
   UserID - single valued attribute → FK

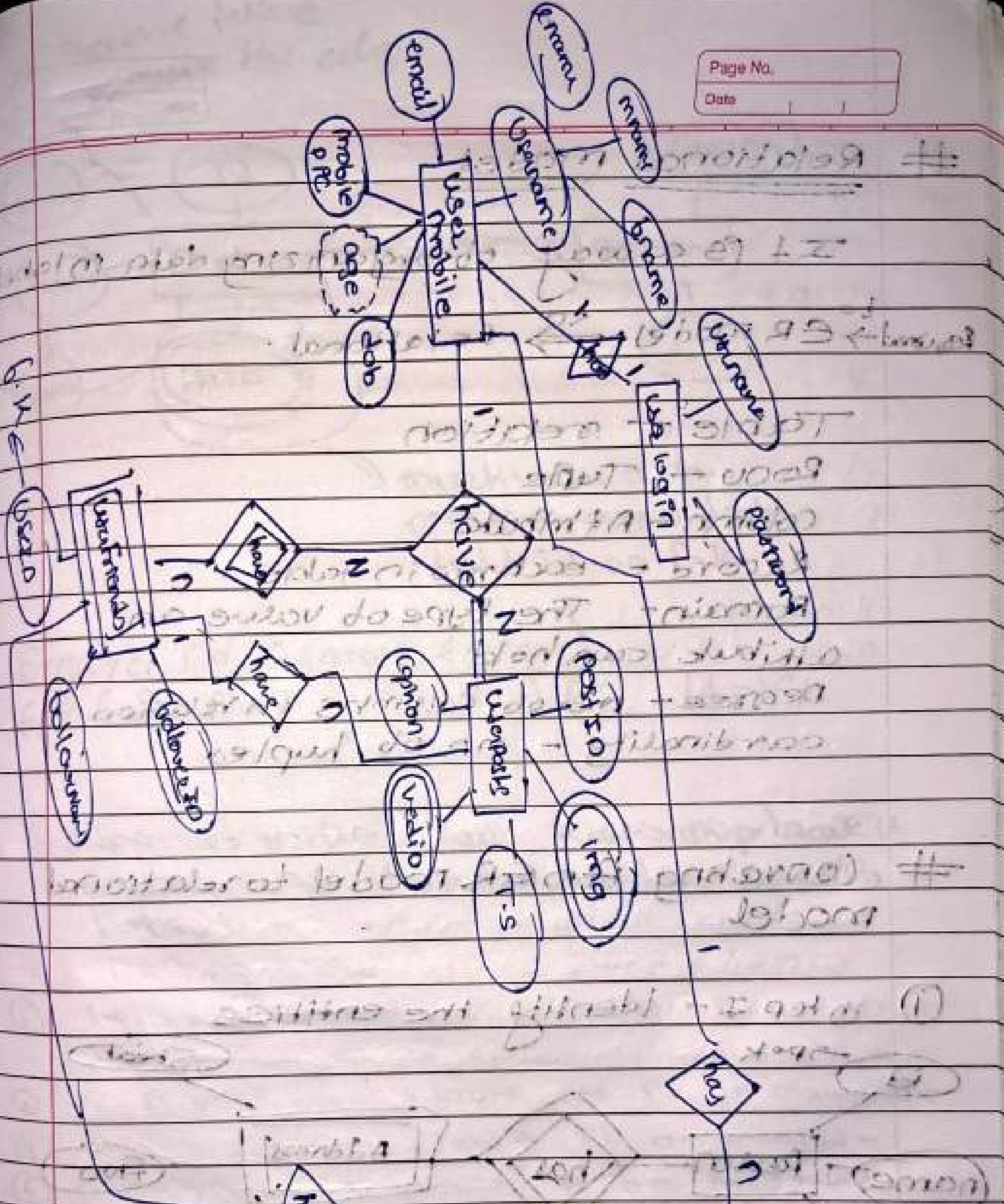
3. UserPost
   PostID → PK
   caption - single valued
   Image - multivalued
   video - multivalued
   likescount - single valued
   timestamp - single valued attribute

4. userlogin
loginID – PK
loginUserName – S.V.
loginPass → S.V

5 User likes
PostID – PK
UID – sigvat

(3) Discuss connections, participation and cardinality

UserProbile have userFriends (M:M)
UserProbile have userPost (1:M) userPost will always be in total participation with userProbile (=)
UserProbile has userLogin (1:1)
UserProbile has userLikes (1:M)
UserLike will always have total participation
UserFriends have userPost (1:M) (=)
UserFriends have login (1:1)
UserFriends has userLikes (1:M) (=)


email
fname
mname
lname
Username
Mobile p.p.
User Mobile
age
dob
has
User login
Username
password
have
N
N
WebPosts
PostID
opinion
vedio
img
T.S
has
has
UserFriends
follower
friendsID
has
userliked
PostID
UserID

Relational Model

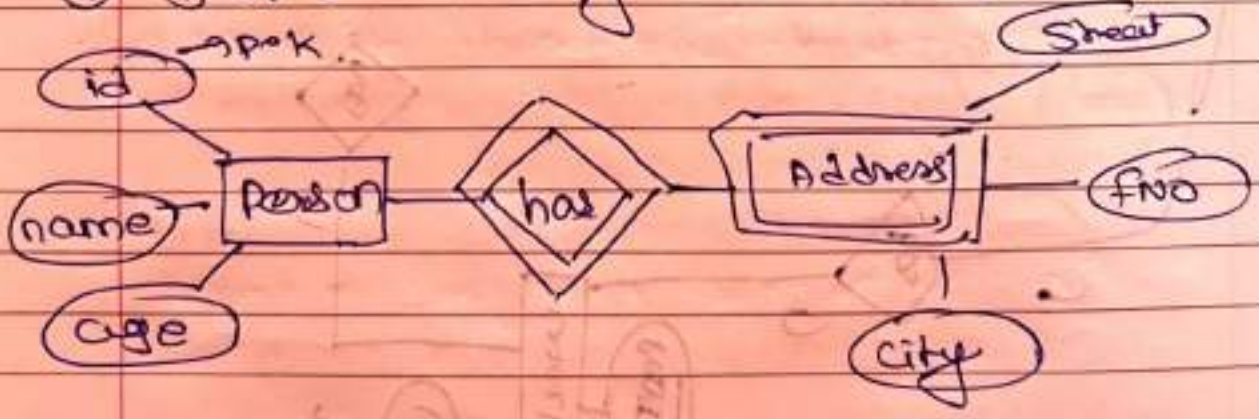
It is a way of organizing data in tables

Required to → ER model ⟹ to Relational.

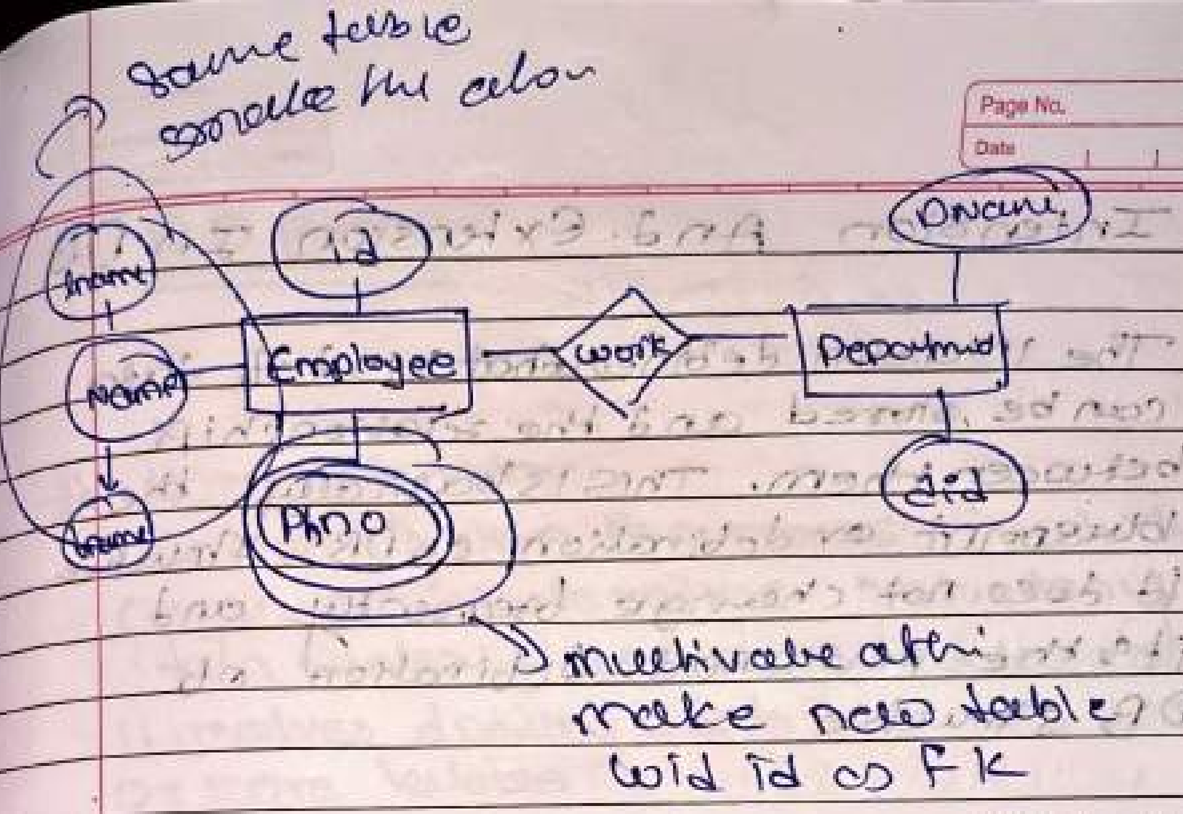
Table — relation
Row — Tuple
Column - Attribute
Record - each row in table
Domain - The type of value an attribute can hold
Degree - no of columns in relation
cardinality - no of tuples

Converting An ER model to relational model

① Step I - Identify the entities



Person (id, name, age) → id (P.K)
Address (id, blatNO, Street, City) → id (FK)
id (FK)
(PK)



Emplye(id, Efname, Elname)
Eph (id, phNO) (id + pho) PK   id => FK

Step 4 - If entities have relationship break it down, and then reduce the tables if possible.

1. 1-1 relationship → 2 tables P.K can be on anside
2. 1-N relation → 2 table ~~or~~ PK on many side
3. N-1 relation → 2 tables PK on many side
4. N-N relation → 3 tables PK on new table

Intension And Extension In DB

The intension defines what kind of data can be stored and the relationships between them. This is basically the blueprint or definition of DB structure. It does not change frequently and it's the permanent definition of DB structure.

It includes
Table definations, constraints, relationships between tables.

The extension is the actual data stored in the DB at a given instance in time. Basically the data which is stored in tuples/rows at a given instance of time. When there are more tuples added the data can change

RDBMS

Normalisation

Normalisation is a process in which we organize data to reduce redundancy (data duplicacy) and improve data consistency. It involves dividing a database into two or more tables

What is data redundancy and consistency and why it's important

When there is same set of data repeated each and every time it results in duplicacy of data (either in row or column) Now row level duplicacy can be removed by using primary key for unique values

Now when we have same data for some set of columns, it leads to different anomalies

1) Insertion Anomalie
2) Updation Anomalie
3) Deletion Anomalie

Redundant data also results in increase in size of DB

Page No.
Date

① Insertion Anomalie
It occurs when it is difficult to insert data into the database due to the absence of other required data.
Consider you want to add a new department but there is no employee in that dept yet

② Deletion Anomaly
It occurs when deleting data removes other valuable data
consider if you delete all the record in the table you will loose the track of dept, their managers and salaries

③ Updation Anomalie
It occurs when changes to data require multiple updates
Consider you want to change the salary for people working in HR dept you need to update it at 3 place

Normalisation is a solution to Anamolies.

Normalisation does not guarenty to completely remove the redundancy or integrity issue it tends to lower it down

Denormalisation

This opposite of Normalisation.
It involves intentionally introducing some redudancy into well-normalized database schema to improve query Performance

* Benifits
1) basic Queries
2) simplified Queries as no Joins.

* Disadvantage
1) Data redudancy
2) less Data consistency
3) less flexibility

Functional dependency

Function dependency describes the relationship between attributes in a relation

A FD is a constraint between two sets of attributes in a relation form a database

FD :- X (determinant) → Y (dependent)

$y = f(x)$

Attribute closure / closure set

Attribute - closure helps us for identifying candidate keys checking for functional dependencies and in normalisation

First Normal Form

First Normal form is the first step in the normalisation process which help us to reduce data redundancy.
- Every table should have atomic values.
ie there shouldn't be any multivalued attribute

It ensures the following set of rules is followed in table
- Atomicity (Attribute should have single value)
- uniqueness of rows (each rows should be)

ID	pName	order
1	Raj	mathsbiology
2	Rik	maths
3	Rahul	Soyen, Egg

M.V

3 ways to convert to 1NF

(1) Repeat the values in id and pname column twice to store single value of m.v attribute order

ID	pname	order
1	Raj	maths
2	Ras	Soya
	Rik	maths
3	Rahul	Soya
	Rahul	Egg

Pk = (order, ID)

e) make new column for each M.V

ID	PName	Order1	order2
1	Raj	mubbin	Sogo
2	Rith	mubbin	null
3	Rahul	Sagar	Egg

PK → ID

Best approach

5) make new table for m.v. Divide the table into student (base) and order (referencing) table based on the M.V attribute order.

ID	PName
1	Raj
2	Rith
3	Rahul

FK

ID	order
1	mubbin
1	Sogo
2	mubbin
3	Soga
3	egg

Second Normal form

A relation is in 2NF if it satisfies the following

1) it is in a first Normal Form (1NF)
2) it has no partial dependency which means no non prime attribute is dependent on a part of any candidate key

Page No.
Date

* when P.D is there in a table?
(LHS is a proper subset of candidate key AND RHS is a non prime attribute)

candidate key : CustomerID + OrderID
Prime Attribute : {CustomerID, OrderID}
Non prime attribute : {OrderName}.

in this relation orderName is dependent on orderID only according to OrderID we provide we provide the ordername

OrderName is determined by only orderID

CustomerID	OrderID	OrderName
1	1	mubbin
2	1	mubbin
1	2	Soga
4	2	Sogar

2NF X

CustomerID	ID
1	1
2	1
1	2
4	2

OrderID	OrderName
1	mubbin
1	mubbin
2	Sugga
2	Sugga

2NF

Page No.
Date

Keys in DBMS

* Why need key
when we need to access any record, row or a tuple we need some identifier uniquely that's why we need a key

* No two tuples can be exactly same in RDBMS

* A key is an attribute or a set of an attribute that can uniquely identify the tuple uniquely

Super key is same as key
{ Super key == key }
same in defination

e.g. sid → SK
{sid, sname} → SK
{std, marks} → SK

if R = (sid, name, marks, dept, course)
then max no of Skey

$^5C_1 + {}^5C_2 + {}^5C_3 + {}^5C_4 + {}^5C_5$

$5 + 10 + 10 + 5 + 1 = 31$

or $2^n - 1 = 2^5 - 1 = 32 - 1 = 31$

n = no of attributes in a relation

If R (A, B, C, D)

max no of SK = $2^n - 1 = 2^4 - 1 = 16 - 1 = 15$

super key is a theoretical concept and we do not implement it in real life.

candidate key

it is a SK whose proper subset is not a SK

$S_1 = \{1, 2, 3\}$ $S_2 = \{1, 2\}$

~~$S_2 \subset S_1$~~ $S_2 \subset S_1$ ✓ proper subset

$S_1 = \{1, 2, 3\}$ $S_2 = \{1, 2, 3\}$

$S_2 \subseteq S_1$ → improper subset

R (A, B, C) SK → A, AB, AC, ABC, BC

				SK	CK
1	1	1	ABC	✓	X
2	1	2	AC	✓	X
3	2	1	A	✓	✓
4	2	2	BC	✓	✓

every

empty set cannot be SK

primary key

one C.O column = no null value
is primary key
and remaining C.O are called
alternate key or secondry keys.

* PK can have single attribute or combination of attribute
* PK should only be one
CO or SK can be more than one.

Functional Dependency

FD : $x \rightarrow y$

determinant dependent

$y = b(x) = x^2 - 4x$

x	y
1	1
2	1
3	2
4	3
2	5

(1) if $t_1.x = = t_2.x$
then $t_1.y = = t_2.y$

Page No.
Date

Student

RNO	Name	marks	Dept	Cour
1	a	70	CS	C1
2	b	60	CC	C1
3	a	70	CS	C2
4	b	60	CE	C3
5	c	80	IT	C3
6	d	80	EC	C2

R.NO → Name

FD : $x \rightarrow y$
if $t_1.x = t_2.x$
then $t_1.y = t_2.y$

Types of Functional Dependencies

① Trivial
$X \longrightarrow Y$ or $X \rightarrow X$
if $Y \subseteq X$

RNO → RNO T.D
(R.NO, Name) → (Name) T.D
(R.NO, Name) → (RNO) T.D
T.D are always valid.

② Non Trivial
$X \rightarrow Y$
but $X \cap Y = \phi$
R.NO → Name
may or maynot be related ali

Page No.
Date

Amstrongs Axioms / inference rules

① Reflexivity (T.D)

$X \to X$

$X \to Y \quad Y \subseteq X$

② Transitivity

If $(X \to Y \ \& \ Y \to Z)$ then $X \to Z$

but if $(X \not\to Y \ || \ Y \not\to Z)$ then it is not always $X \not\to Z$ we can not relate

③ Aggomentation

If $X \to Y$

then

$XA \to YA$

If RNO → Name

(RNO, marks) → (Name, marks)

It is not true that only a primary key can determine & only non primary key can also determine an attribute

④ If $X \to Y$ & $X \to Z$

Then $X \to YZ$

R.NO → Name & RNO → marks

RNO → (Name, marks)

Page No.
Date

⑤ $X \to YZ$ then $X \to X$ & $X \to Z$ ✓

ib $XY \to Z$ then $X \to Z$, $Y \to Z$ ✗

⑥ ib ($X \to X$ & $YZ \to A$)
then $XZ \to A$

⑦ ~~X A~~ ib ($X \to X$ & $A \to B$)
then $XA \to XB$

Attribute closure / closure set

R(A, B, C, D, E)
FD{ $A \to B$, $B \to C$, $C \to D$, $D \to E$ }

$A \to C$ $B \to D$ $C \to E$
$A \to A$ $B \to E$ $C \to C$
$A \to D$ ~~$B \to BE$~~ $C \to CD$
$A \to E$ $B \to B$ $D \to E$
$A \to ABCDE$ $B \to BEDC$

$A^+ = \{A, B, C, D, E\}$
$BD^+ = \{BD, B, D, AD\}$ {A, D, B, C, E
$B^+ = \{B, C, D, E\}$
$D^+ = \{C, D, E\}$

Page No.
Date

Sk is a set of attribute whose closure contains all the attributes of given ~~datas~~ relation

eg A, and AD

finding all C.K of a relation

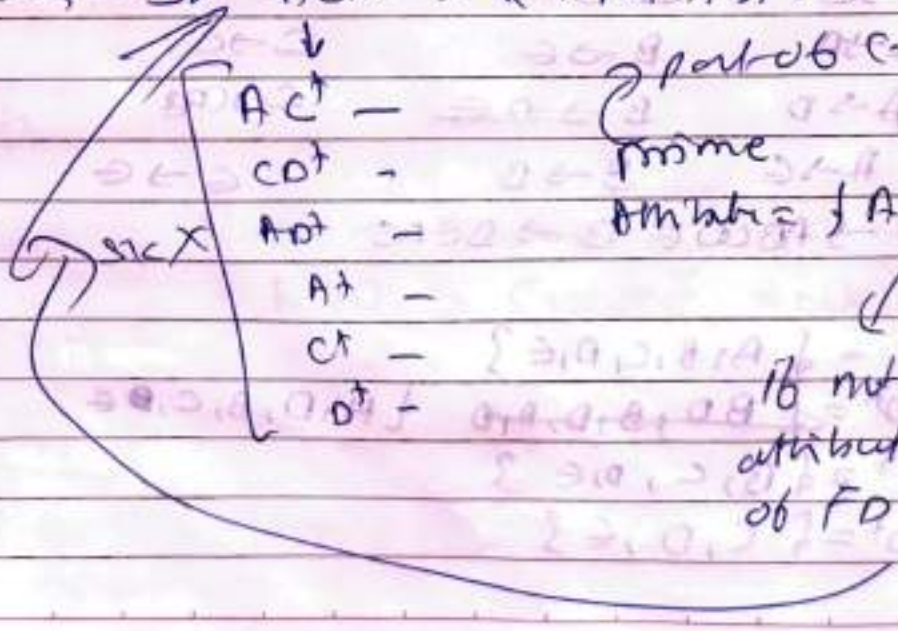
R(A, B, C, D, E)
FD :- { A → B, D → E }

Sk $ABCDE^+ = \{A, B, C, D, E\}$

Sk $ACDE^+ = \{ABCDE\}$

Sk $ACD = \{A, C, D, B, E\}$

C.K ← Sk $ACD^+ = \{A, C, D, B, E\}$



R(A, B, C, D)

FD : { A → B, B → C, C → A }

SK ⇒ ABCD+ { A, B, C, D }

SK ⇒ ACD+ { A, B, C, D }

SK ⇒ AD+ { A B C D }

↓

{ A, D, [redacted] }

D·A = { A, D }

C·K A+ = { A, B, C } D·A = { A, D, C, B, [redacted] }

D+ = { D }

C·K AD

↓

C·K CD → SK

↓

B·K BD → SK

↓

C·K [redacted]

R (A , B , C , D , E , F)

FD = { AB → C , C → DE , E → F , D → A , C → B }

~~ABCDEF~~

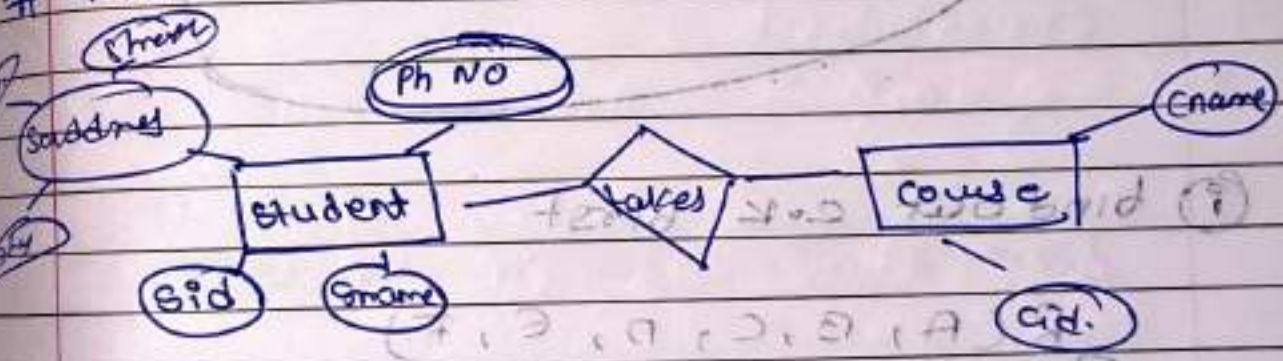
R(A, B, C, D)

FD = {AB → CD, D → B, C → A}

Normalisation.

~~Normalisation is or no~~

First Normalisation.



Student

Sid	Sname	Sadd	PNO
1	Jenny	HARYANA, INDIA	P1, P2
2	Jiya	PUNJAB, INDIA	P3
3	Payal	RAJ, INDIA	P4, P5
4	shavi	HRAYANA, INDIA	P2.

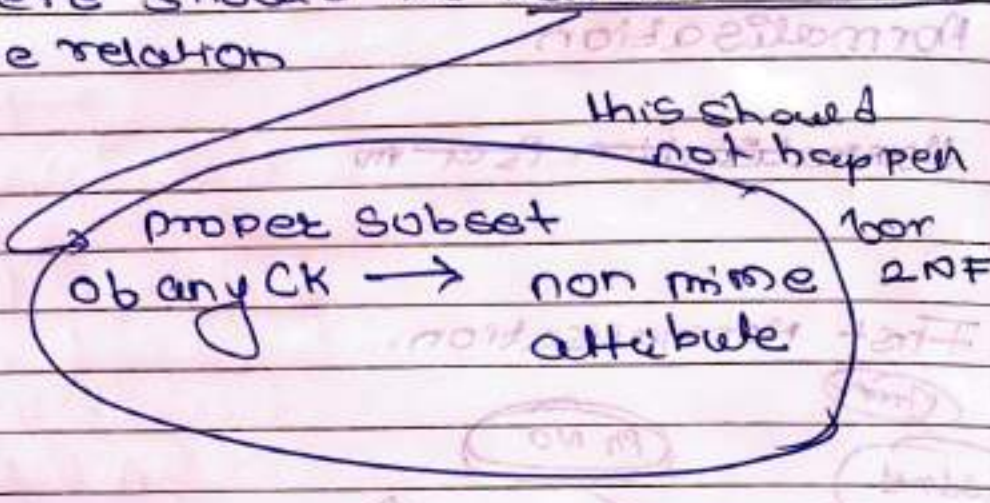
every attribute should have atomic values

unqiuness in rows.

Second Normal form.

rules

1. relation should be in a 1st NF
2. There should NO Partial dependency in the relation



(1) find out C.K first

R(A, B, C, D, E, F)

F.D ⇒ {A → B, B → C, C → D, D → E}

A^+ = {A, B, C, D, E, F}

$(AF)^+$ = {A, F}

A^+ = {A, B, C, D, E} Not SK

F^+ = {F} Not SK

P.A = {A, F}

C.K.

AF → only CK as no PA on R of Relation

PA = {A, F}
NPA = {B, C, D, E}

CK ⇒ AF
proper subset of AF = {A, F}
A determines B
which is a non prime A

∴ This relation is not in 2NF.

# R (A, B, C, D)
FD = {AB → CD, C → A, D → B}

ABCD⁺ = {A, B, C, D

AB⁺ = {A, B, C, D}

A⁺ = {A, } → Not SK

B⁺ = {B, } → Not SK

→ C.K

PA = {C, A, D, B}

PA = {C, A, D, B}

NPA = {~~ABCD~~}

AB
↓ ↘
C B AD

~~AB~~

ABC {A, B}
CB C {C B}
AB } {A B}

This is in 2NF

Page No.
Date

Attribute

when all the ~~non~~ of relation are P.A
then the relation is a 2NF

Q $R(A, B, C, D)$
$FD \rightarrow \{A \rightarrow B;\ B \rightarrow C;\ C \rightarrow D\}$

$ABCD^+ = \{A, B, C, D\}$

$A^+ = \{ABCD\}$

↓ proper subset

$\{\phi\}$
C.K

A

only CK

P.A $= \{A\}$

NPA $= \{B, C, D\}$

~~PA can determine NPA so it is not in 2NF~~

proper subset of C.K

$A = \{\phi\}$

cannot determine any NPA hence it is ~~not~~ in 2NF

~~when one relation has a single C/K then~~

* for converting a non 2NF form relation into a 2NF form relation we need to perform lossless decomposition

Third normal form

Student

Sid	Sname	DOB	State	Country	PIN	T.C
1	A	–	HR	IN	1234	–
2	B	–	HR	IN	4567	–
3	C	–	HR	IN	8910	–
4	D	–	PN	IN	111213	–

rules.

1. Relation should be in a 2NF
2. NO NPA should be able to determine ANPA or there should not be any transitive dependency

A table is in BNF if and only if for each of its non trivial functional dependency at least one of the following conditions holds

(1) LHS is SK
(2) RHS is prime attribute

Q R(A, B, C, D)
FD → { A → B, B → C, C → D }

A⁺ = { A, B, C, D } ⟹ S.K

A → B → C → D

P.S.S

{ ϕ }

C.K

PA = { A }

ON C.K

PA = { A } NPA = { B, C, D }

FD A → B, B → C, C → D

NPA NPA

This relation is in 2NF but not in 3NF

Page No.
Date

R(A,B,C,D,E,F)
FD → { AB → CDEF, BD → F }

ABCDEF+ = { ——— }

SK → AB+ = { ——— }

↳ A+ = {A} X
B+ = {B} X

CK = {AB}
PA = {A,B} NPA = {CDEF}

AB → CDEF (BD → F)
X transitive ✓ transitive

BCNF (Boyce-Codd Normal form)
↳ advanced version of 3NF

Rules
① Relation is in 3NF
② For each non trivial FD X → Y
X must be a super key

R(A,B,C)
FD :- { A → B, B → C, C → A }
SK = ABC+ = {ABC}
SK = A+ = {ABC}
CK = A, B, C
PA = {A, C, B}
NPA = {ϕ}

Page No.
Date

Rule is following BCNF

① Trivial FD

$A \rightarrow B$ where B is subset of A

e.g {EmpID, Name} → Name

but

{EmpID} → EmpID

$AB \rightarrow B$ or $A \rightarrow A \Longrightarrow$ T.D.

② Non Trivial FD

$AB \rightarrow C$

{EmpID, Name} → {empAddress}

$C \nsubseteq AB$, $C \cap AB = \phi$

Then NT FD.

Amstrongs rules

① reflexive $X = \{a, b, c, d, e\}$

$Y = \{a, b, c\}$

If Y is a subset of X

then $X \rightarrow Y$

Page No.
Date

(2) Aggomentation
if $A \rightarrow B$
then
$AZ \rightarrow BZ$

(3) transitivity
if $(A \rightarrow B \text{ & } B \rightarrow Z)$
then
$A \rightarrow Z$

why Normalisation
→ reduce redundancy
→ improve query performance.
→ data consistency

if not Normalise
(1) Delete, update, insert anomalies.
(2) data redundancy, and space consomption.

1NF
(1) No multivalue attribute only atomic values.
(2) unique tuple identifier

emp

ID	Name	Phone
1	A	93, 82
2	B	81
3	C	95, 98

emp

ID	Name
1	A
2	B
3	C

~~Phone~~ contact

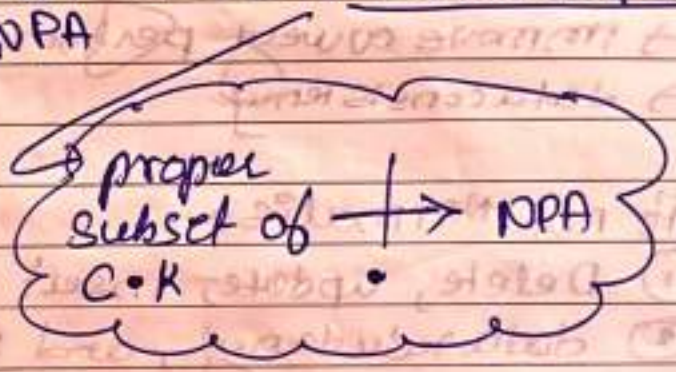
ID	ph	EID
1	93	1
2	82	1
3	81	2
4	98	3
5	92	3

2NF

① Should be in 1NF form

② Should not have partial dependency

PA ⟶ NPA



3NF

① Should be in 2NF form

② Should not have transitivity

NPA ⟶ NPA

BCNF

① Should be in 3NF

② for each non trivial X ⟶ Y

X is a super key

2NF

Sid	ProjID	Sname	Proj name
589	P09	amia	Geo
576	P07	Jcob	MERN
586	P03	Ava	IOT
592	P05	Alex	Cloud

P.K = { Sid, ProjID }

PD → Sid → Sname
ProjID → PName } PoD.

Solution for 2NF:

PK

Student

Sid	ProjID	Sname
589	P09	amia
576	P07	Jcob
586	P03	ava
592	P05	alex

Project

PID	Pname
P09	
P07	
P03	
P05	

PK

3 NF

PA → NPA

A	B	C
a	1	x
b	1	y
c	1	x
d	2	y
e	2	y
b	3	z
g	3	z

→ Not in 3NF

⇓

R1

A	B
a	1
b	1
c	1
d	2
e	2
b	3
g	3

R2

B	C
1	x
2	y
3	z

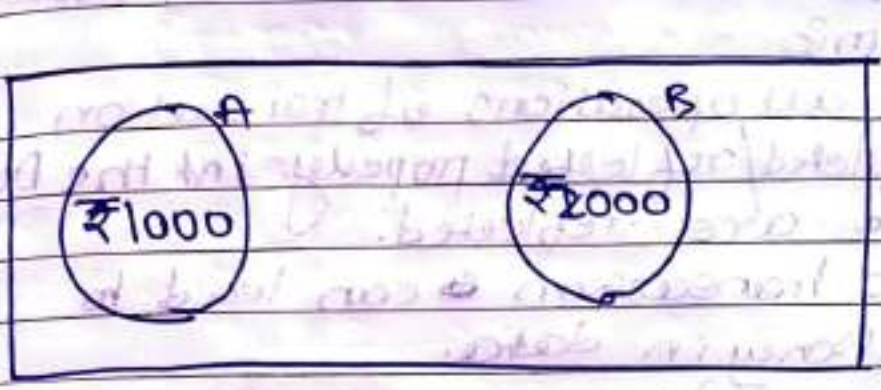
BCNF

when a prime attribute will again determine a prime Attribute then table is not in BCNF

$X \longrightarrow Y$

'X should be a super key.

ACID Properties And Transactions.



task 1 → ₹50 from Bank A/C A to A/C B

steps:-

① read (A)
② A = A - 50;
③ write (A)
④ read (B)
⑤ B = B + 50
⑥ write (B)

6 steps happening in a sequence can be called a Transaction.

Transactions need to be Atomic {either complete or not complete}

* Transaction is a unit of work done against DB in a logical sequence of SQL queries
* The transaction will either will be permanent on success, but will roll back to initial state on any errors.

ACID :- To ensure integrity of data we require that DB system maintain the following properties

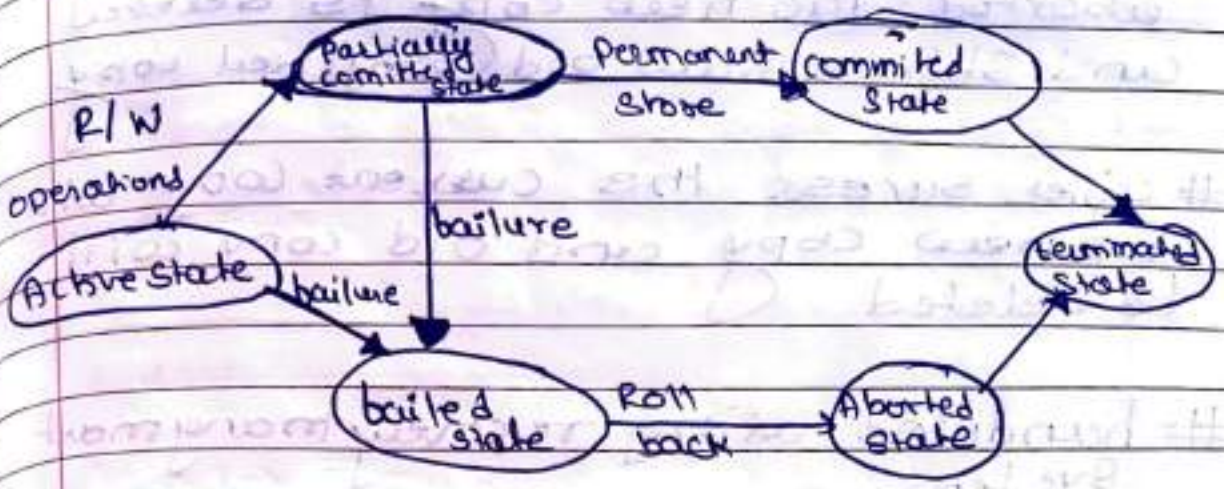
(1) A ⇒ Atomic
Either all operations of transaction are completed/reflected properly in the DB or none are reflected.
Non Atomic transaction can lead to inconsistency in data.

(2) Consistency (C) :-
* integrity constraints must be maintained before and after transaction
* DB must be consistent after completion of transaction.

(3) Isolation
* Even though multiple transactions may execute concurrently the system guarantees that for every pair of transactions T_i and T_j appears
• it appears to T_i that T_j finished execution before T_i started or T_j started execution after T_i finished. Thus each transaction is unaware of other transactions executing concurrently in the system
• multiple transactions can happen in the system in isolation without interfering each other

(4) Durability — done by recovery management system.
* After transaction completes successfully the changes it made to the database persist even if there are system failures.

Transaction States (life cycle)

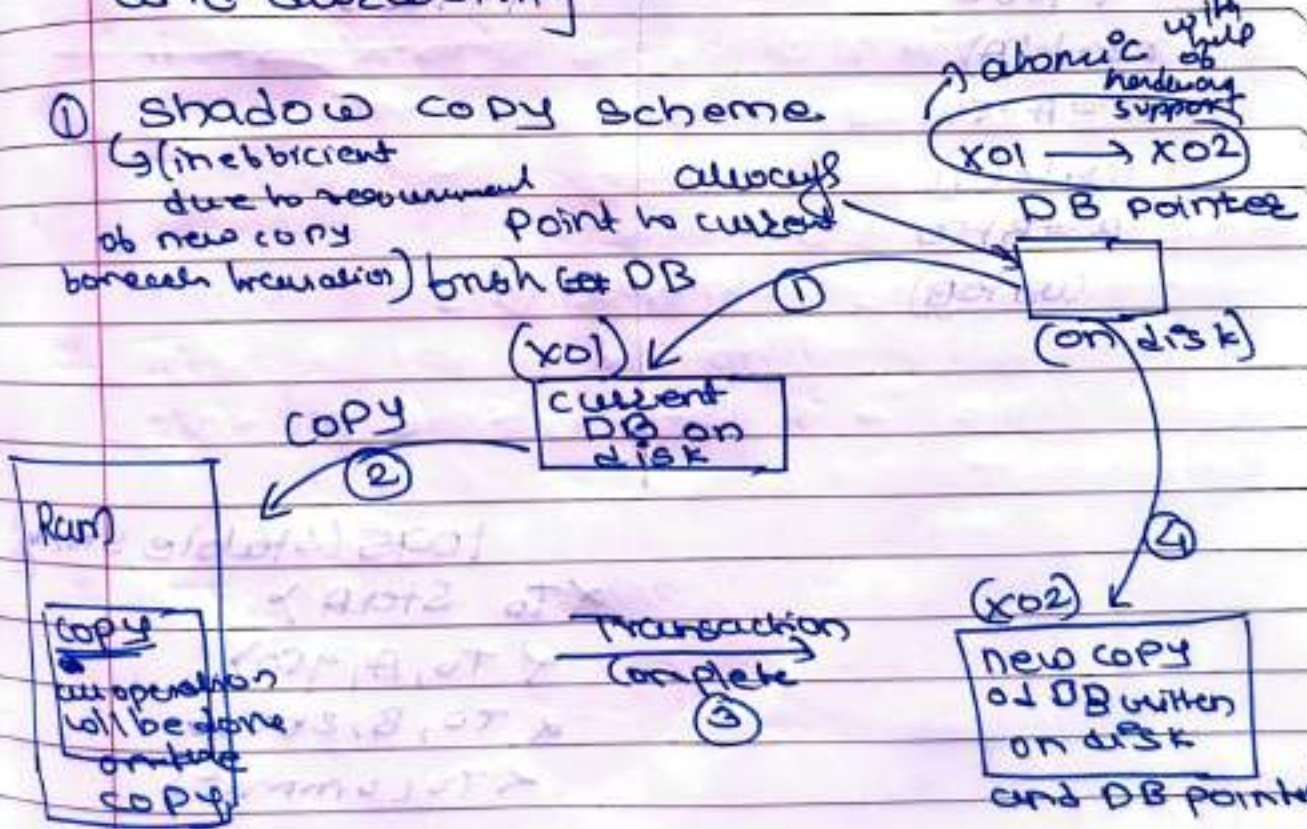


How to implement

How to implement Atomicity and durability

① Shadow copy scheme

(inefficient due to requirement of new copy before each transaction)



#transaction is not considered as completed until DB pointer does not point to new location

if at any case the transaction is aborted the new copy is deleted and still we have old original copy

after success the current copy will be new copy and old copy will be deleted

handeled using recovery mangment system

(2) log based recovery system.
→ generation of a log for every completion of step in a stable storage

A	B
₹1000	₹2000
read (A)	
A = A - 50	
write (A)	
B = B + 50	
write (B)	

logs (stable storage)
< To Start >
< To, A, 950 >
< To, B, 2050 >
< To commit >

Page No.
Date

* The log is a sequence of records log of each transaction is maintained in some stable storage so that if any failure occurs then it can be recovered from there.

* First log is written then record is written

(a) debered DB modification.
Keep writing logs and actual modification or transaction are reflected after completion of logs

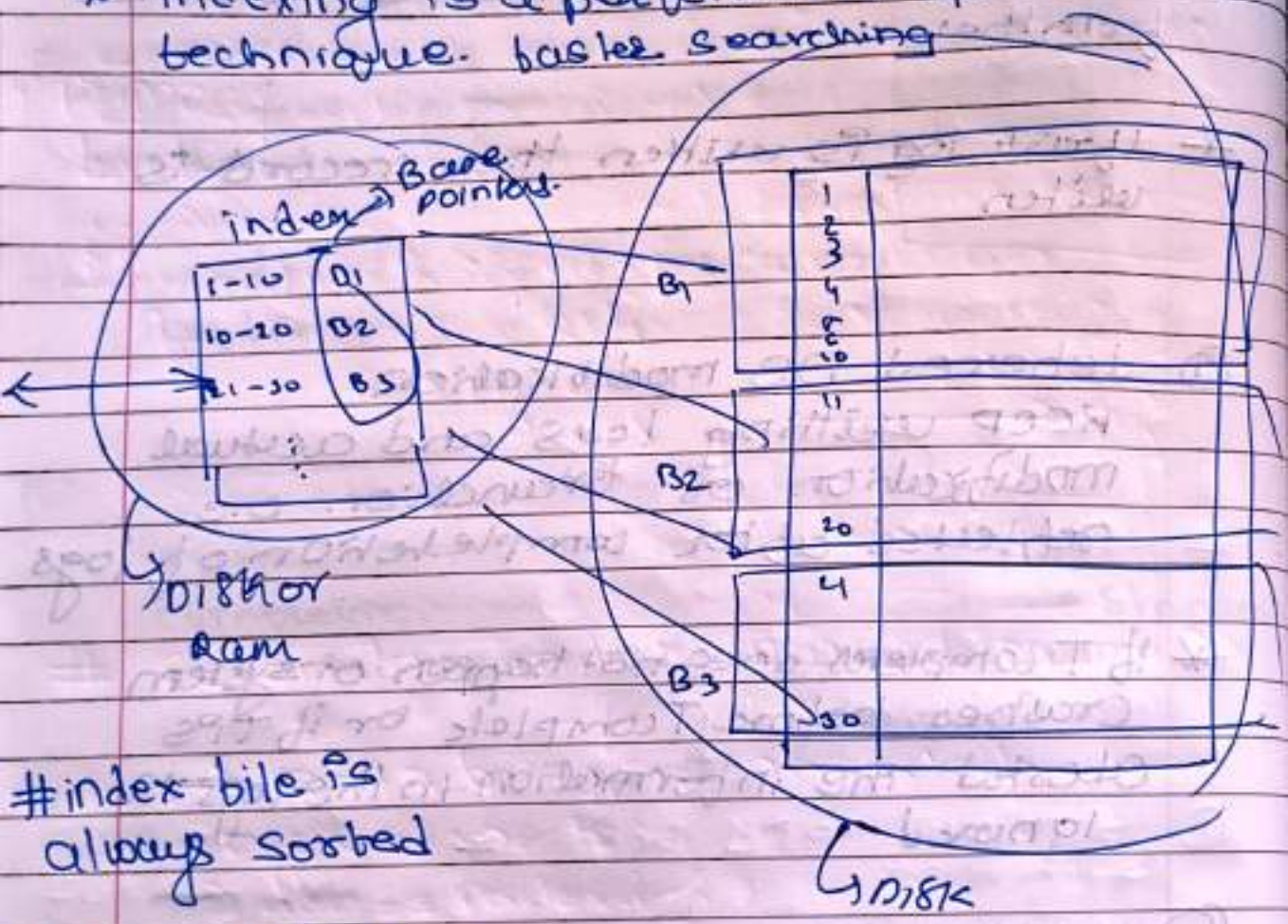
* If T completes does not happen or system crashes before T complete or if it is aborted the information in the logs is ignored.

(b) immidiate DB modification.
write logs along with transaction and DB modification are done with logs.
log → update
log → update
log → update

debered	immidiate
<To, A>	<To, A, 1000, 950>
<To, A, 950>	<To, B, 2000, 2050>
<To, B, 2000, 2050>	<To commit>
<commit>.	

Indexing

* indexing is a performance optimisation technique. faster searching



#index file is always sorted

#we try to reduce search space in index file.

#idex table is a D.S where mapping is present

Dense index > Spare dense index values values

Page No.
Date

Types of indexing

① Primary indexing (clustering indexing)
→ used for database which is sorted on any one attribute that attribute of tuple may or may not be P.K

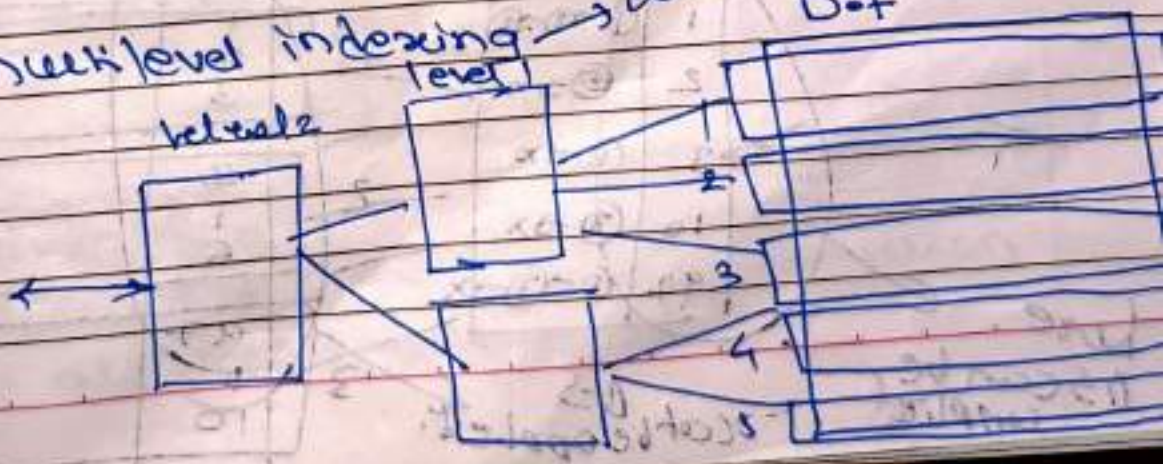
Types of indices

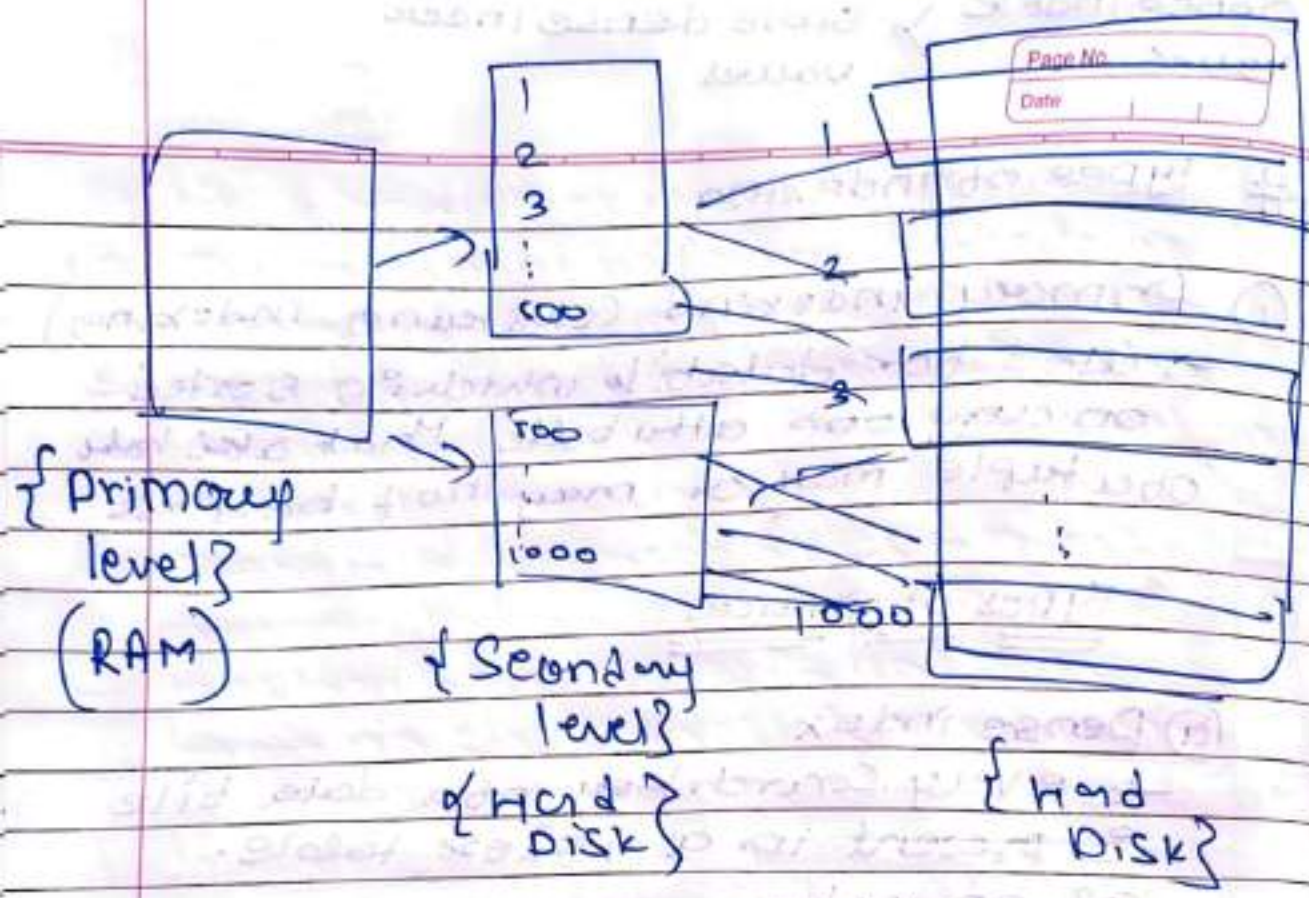
ⓐ Dense index
→ every search key of a data file is present in a index table as an entry

ⓑ Sparse index → group by
→ every search key of data file is not present in a index table only few are present which represent bock of chunck of block

* when sorting of D.b is done on base of P.K attribute we will need sparse indexing and when non P.K is used we will require dense indexing

② Multilevel indexing → used in huge data

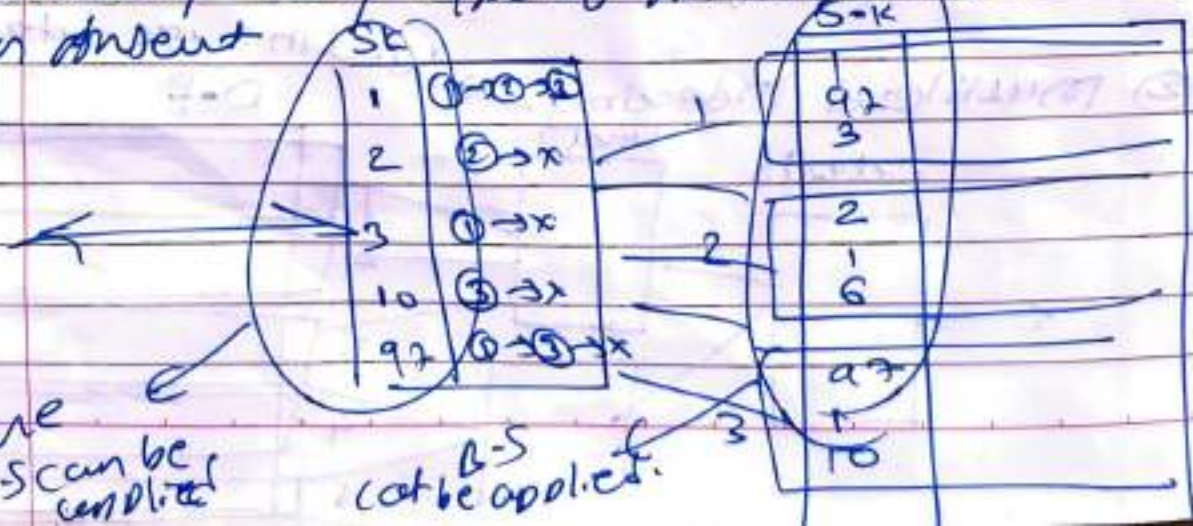




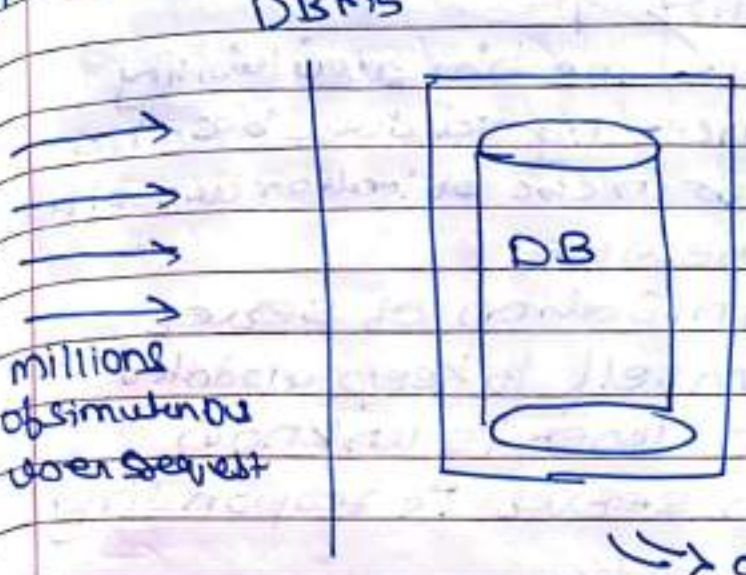
Secondary index (nonclustering index)

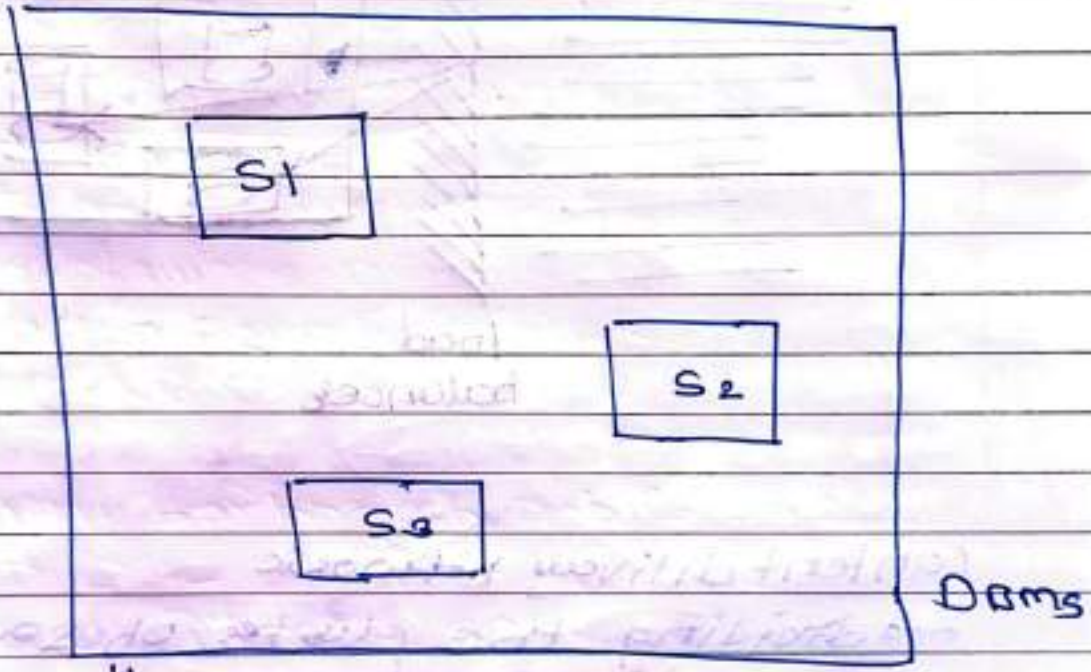
→ Data files are unsorted in all aspects
→ dense indexing
→ linear search.

Secondary indexing is applied on the non sorted attributes of a tuple where already any type of primary index is present



Clustering / Replica sets.





One cluster
{ group of same DB server }

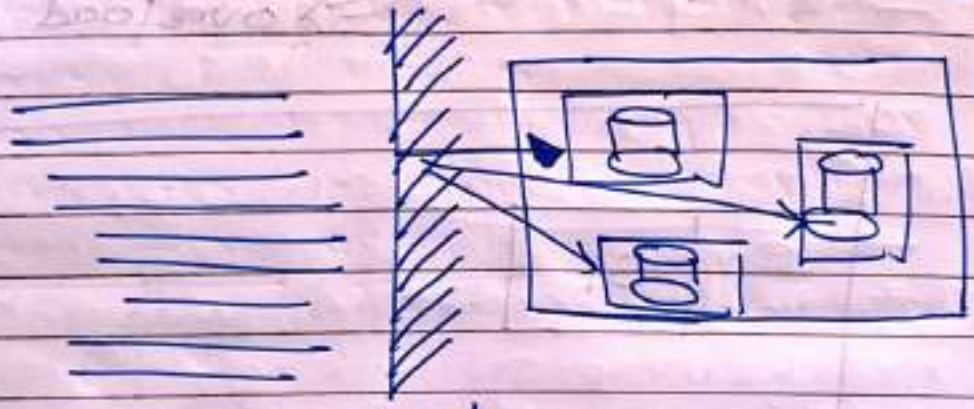
S1 = S2 = S3
set of server or replica of servers.

advantage

① Redundancy → same information in all replica server

this redudancy will help in

* availability ↑
* maintenence me bhi availability
* load balancing ja dude busy ho
* Backup in case of load or corruption
* Security measures
* ~~Sy~~ Synchronisation of server among themself to keep updated
* abstraction user is unknow from which server is responding



load balancer

Content dilivery network

→ dividing the cluster of server geographically to server the world wide request

Partition And Sharding

Data → store → DBMS

↓

Bohot jayda → saving massed

huge data comes with manageability issue

huge request

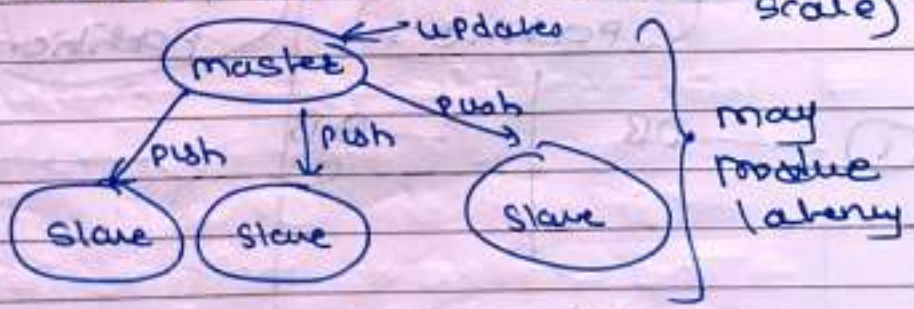
* when we have huge data & huge request we usually distribute data which result in distributed DB

⇓

solutions

① Scale up (top notch hardware)
↳ limited and costly

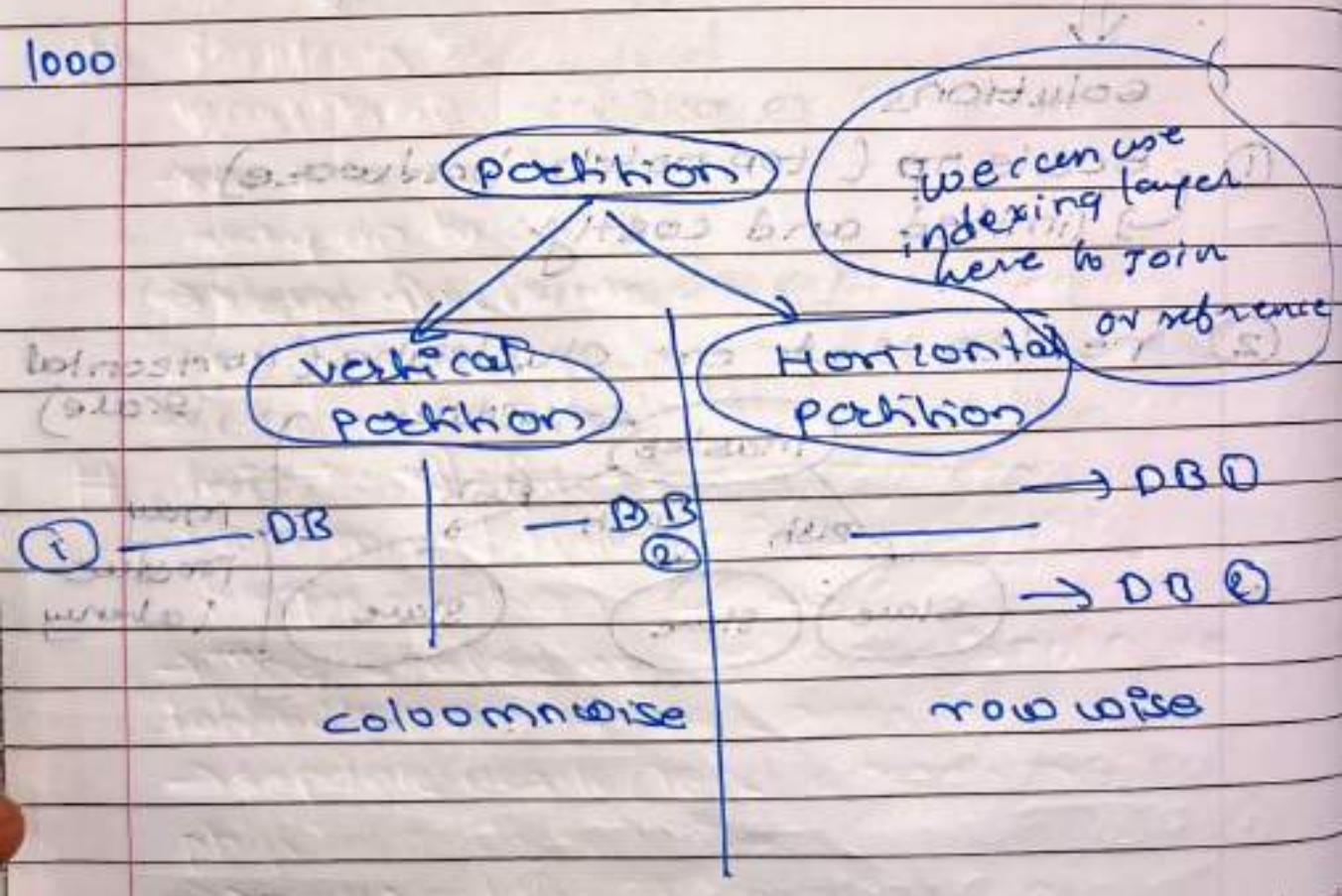
② replica set or clustering (Horizontal Scale)



(3) partition & Horizontal Scaling or Scale Out

Student Table

	Roll NO	Name	age	sex	DOB
1					
2					
3					
:					
:					
:					
1000					



Page No.
Date

advantages
→ less costly
→ high availability
→ high response time
→ manegability
→ parallelism.

Partition And sharding results on distributed database.

sharding
a technique to apply a horizontal partitioning.

* The fundamental idea of sharding is the idea that instead of having all the data set on one DB instance we split it up and introduce a routing layer so that we can forward a request to the right instance that actually contain data

Pros
→ availability, scalability

Cons
→ costly, complexity

difference btw SQL & NOSQL *
why and when to use SQL or NOSQL *

Page No.
Date

Database Scaling Pattern

advantage of scalability → reduce latency
→ query performance
→ availability
→ user experience.

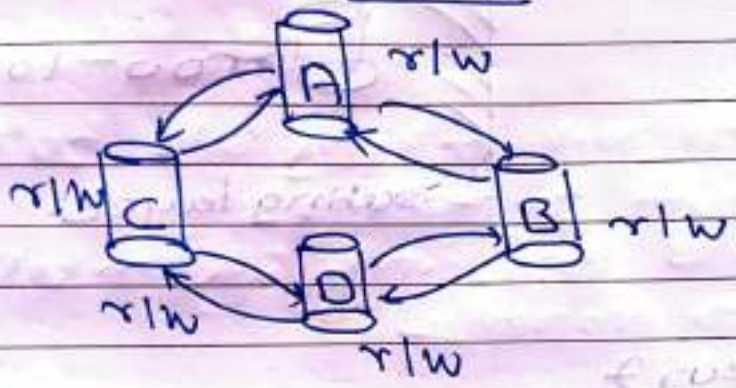
pattern 1
Query Optimisation & connection pool implementation.
→ to cache frequently used non dynamic data like; booking his, payment his, user mobiles
→ introduce data redudancy
→ use connection pool libraries to cache DB connections

Pattern 2
Vertical scaling or scale up
→ scale up is model friendly only up to a limit
→ more you scale up cost increases exponentially

~~Command Query Segregation~~
command Query Responsibility Segeregation (CQRS)
→ The scaled up machine is notable to handle all read/write requests
→ separate write/read operations physical machine wise
→ all read Query to replica and all write Query to primary machine.

Page No.
Date

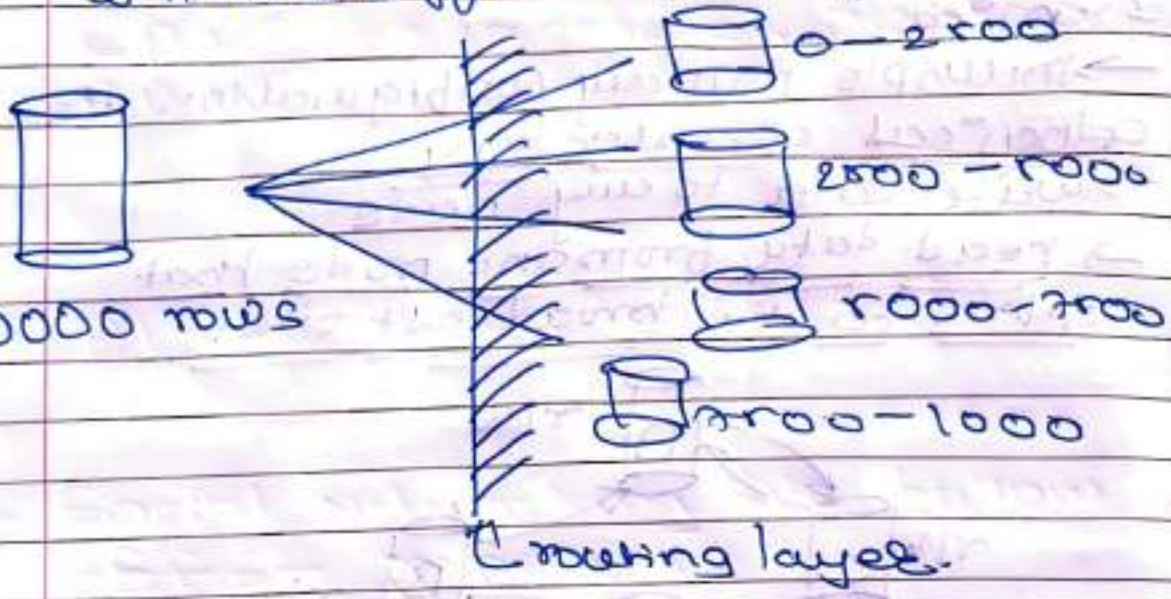
Pattern 4
Multiple primary replication.
→ why not distribute write request to replicas also
→ all machines can work as primary replica
→ multiple primary configuration is a logical circular ring
→ write data to any node
→ read data from any node that replies to the broadcast first



Partition of Data by functionality
→ what about separating the location tables in separate DB schema
→ what about putting that DB is separate machines with primary-replica or multi primary configuration
→ Different DB can host data categorised by different functionality
→ Backed or application layer is required to join the tables.

Pattern 6 → Horizontal scaling out

→ apply Sharding (horizontal scale) to each machine will have same logical schema but will have different part of data



Pattern 7

Establishing multiple Data center or Data centers wise partition.

→ requests traveling across continents are having high latency.
→ what about distributing traffic across data centers
→ data centers across continents
→ enable cross data center replication which help disaster recovery
→ This will always maintain availability.

The CAP Theorme — related to distributed system

C :- consistency.
A :- availability
P :- Partition tolerence
↳ communication ka break hona between multiple nodes

The cap theorme states that a distributed system can only provide two of three properties simultenously. consistency, availibility and partition tolerence. The theorme formalises the tradeoff between consistency and availability when there's a partition.

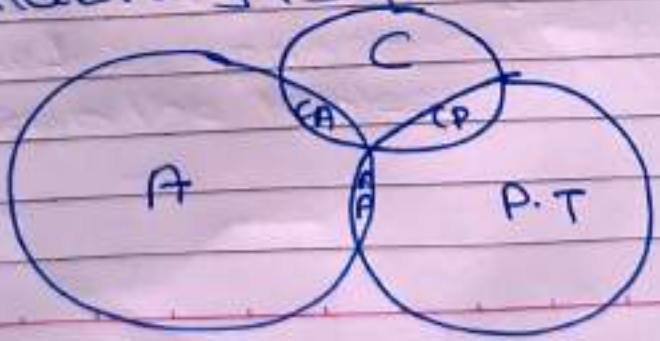
Incase of partition

AP or CP

only two possible

* When availability is choosen consistency is lost

* When consistent is choosen the availability is lost.



In Banking System availability is not imp consistency is so they use CP and Mongo DB uses CP

* All Three together are possibe only in hypothetical situation of a single node DB

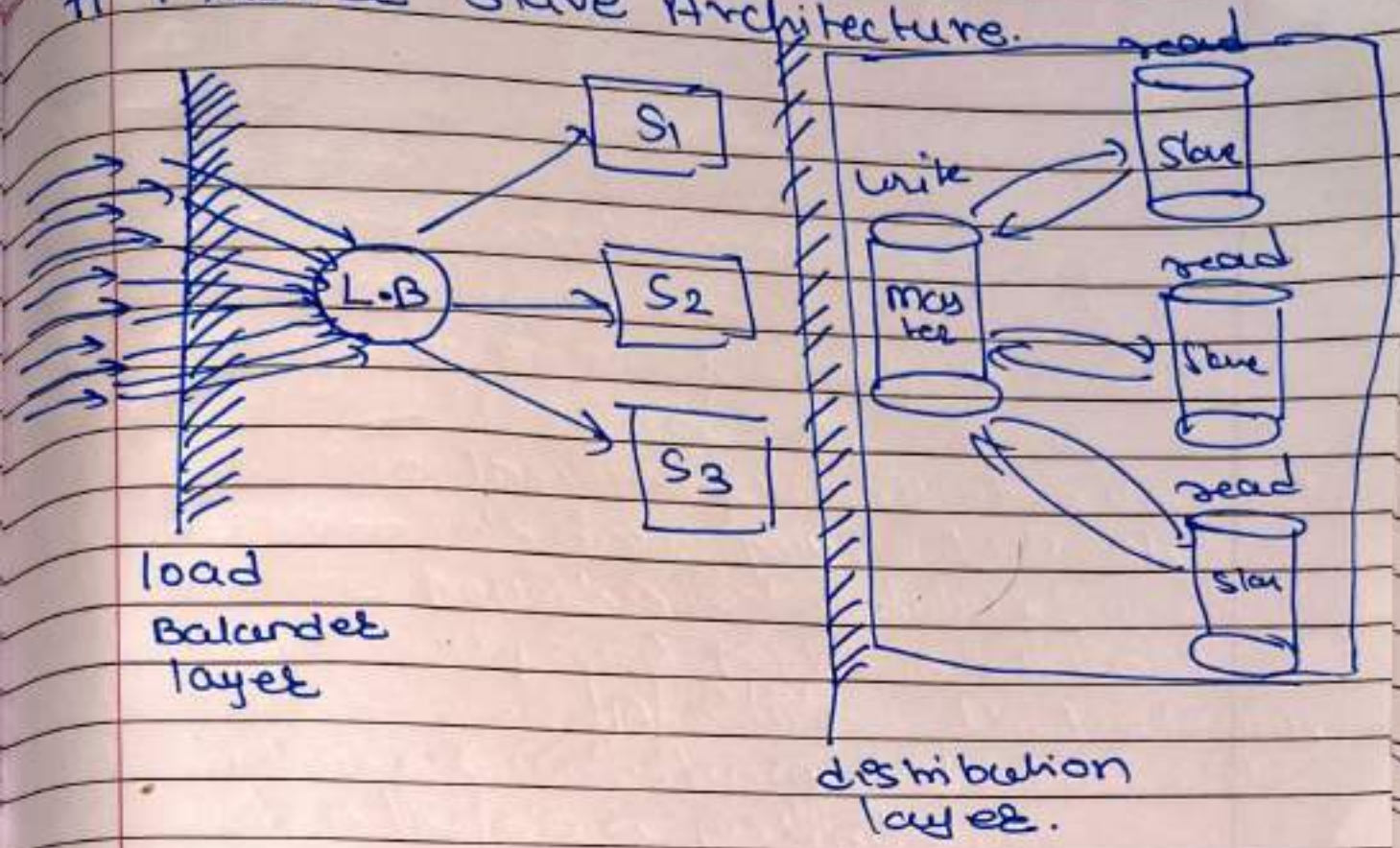
CA X → practical not possible.
CP ✓
AP ✓
} in case of partition

AP — are eventually consistent

* Social Networking site follow BASE properties
* while Banking System follow ACID property

Basically Available
Soft state
eventually consistent

Master Slave Architecture.



master slave will remove single point of failure

* master is always latest (original / primary DB and it handles write o/p

replication to solve can be both
Async and sync
↳ write then
↳ write
and after
sometime
of interval replication.
↳ each transaction
replicate