Date
Page

Data base - Storing data in efficient way and easy access of data.

Data - Raw data (Temperature of 2-3 days)
Information - Meaningful data.
Knowledge - If we study temperature data of 5 years and suppose if in last 2 years it has ↑ or ↓, then knowing why it has been occured, factors, etc. is knowledge. - is what we know

Issues of File Management
Redundancy, Inconsistency, Difficult to access data, Backup & recovery is not there, Unauthorized access of data

DBMS - System that manages database.
Components: Data definition (create table), Insertion, Updation, Retrieval of data, User administration (grant, revoke).

Database - is a collection of related data.
Rows (Tuple), Column (Attribute), Table (Relation) Entity

Relation Name	No. of col^n
Student	4
Course	4
Section	5
Grade	3

— Database

Col^n Name	Datatype	Rel^n Name

DRAWBACKS OF FILE MANAGEMENT

- Data redundancy and inconsistency
- Difficulty in accessing data
- Data isolation Files with diff formats
- Data integrity changing/ Adding constraints later is very difficult
- Atomicity of updates - Changes made should be in all entries is Atomicity. (Suppose system crashes)
- Concurrent access by multiple users
- Security issues We can't give particular access to particular user. (Student-teacher attendance)

We can't physically loose data - means the data is there, we loose the access to the data

User
↓
DBMS
DB
Data

DBMS → DB + set of programs
collection of interrelated data

ACID Property (Atomicity Consistency Isolation Durability

DRAWBACKS OF DBMS

- Cost of H/w & S/w
- Size (storage size, accomodation of servers.
- Complex
- Higher impact of failure.

1-tier architecture.

User
↕
DBMS

Disadv:- changes can be made by user directly to DB.

Scanned by CamScanner

Atomicity - Transaction in DB would be either fully completed or fully aborted

Date
Page

2-tier architecture (Client-server)

Client can directly communicate with server.

(View) User ⟷ Applicⁿ ⟷ DB (Physical)
(Conceptual)

3-tier architecture.

Client can't directly communicate with server

User ⟷ Applicⁿ Client ⟷ Applicⁿ Server ⟷ DB

3 levels Of Data Abstraction.

1. Physical / Internal level - Actual storage of DB
2. Logical / Conceptual - Structure & contents of DB
3. ~~Phys~~ External / View / User level - describes various user views.

Schema - Logical structure of the DB

Instance - Content of DB at a particular time

Advantages of 3-tier

1. Maintainability 3 tiers are independent so easy to maintain
2. Scalability We can increase physical schema
3. Flexibility
4. Availability

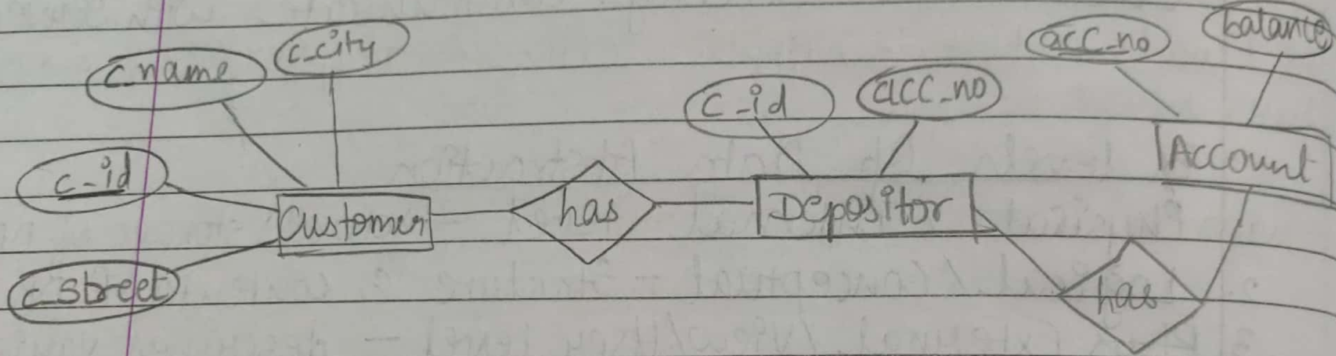
Process of data creation - Data Modeling

Data Independence - changing in the 1 layer without affecting its higher level.

1. Physical data independence - changes in physical layer doesn't affect logical & view level
2. Logical data independence - changes in logical layer doesn't affect view level, but changes physical layer. Difficult to maintain

Database Architecture

1. Centralized DB
2. Client Server DB
3. Parallel DB
4. Distributed DB



Data Models

✓ ER Model
✓ Relational Model
Object Oriented, Network, Hierarchical, etc..

DDL (Data Definition Language)

Create, Alter (add, modify, drop), Drop, Truncate

DML (Data Manipulation Language)

Insert, Update, Delete

DQL (Data Query Language)

select

DCL (Data Control Language)

Grant, Revoke

TCL (Transaction Control Language)

Commit, Rollback, Savepoint

KEYS

All probable attributes that can be a key
- Super key / Key
Eg: Roll No, Aadhar, PAN, contact no, etc.

Key is something throughwhich we can uniquely identify each row.
It should be Unique, Not null and identify all other attributes (by closure)

Consider $A \to B$ and $B \to CD$
For being a key A, B, C, D should have closure: $R\{A, B, C, D\}$

$\{A\}^+ = \{AB, BCD\} = \{ABCD\}$ ✓ can be a key
$\{B\}^+ = \{BCD\} = \{BCD\}$ X can't be a key

For set of Super keys $\{A, B, C, AB, CD, BC\}$
Candidate key - optimum (minimum) set of Super keys
The key selected as 'C key' should not (include) have any super key.
So if A is 'C key', AB can't be C key.
if B is 'C key', BC and AB can't be C key
if C is 'C key', BC and CD can't be C key
So Candidate keys $= \{A, B, C\}$

For Super keys $\{A, B, CD, AB, BC\}$
So candidate keys $= \{A, B, CD\}$

Subset

Primary Key - is a key selected by DBA from the set of Candidate Keys.

$\Rightarrow$ R(ABCD)

$ABC \to D$, $AB \to CD$, $A \to BCD$

Find Candidate Keys.

R(A B C D)

Here check ke kona ma incoming arrow nathi. Here A

So A - is Essential attribute (i.e A will be present in all keys)

$\{A\}^+ = \{ABCD\}$ ✓

If all have incoming arrow, then find closure of all Attributes.

- Foreign key - Ek table ni primary key agar bija table ma hoi present, to bija table ma e Foreign key bane!

Refered / Base Table $\to$ Primary Key

Referencing Table $\to$ Foreign Key.

Referential Integrity Constraint

Constraints while Insert, Update and Delete from base and referencing tables.

Constraint

Base Table		Referencing Table	
Insert	×	Insert	✓ (Maybe)
Delete	✓ (Maybe)	Delete	×
Update	✓ (Maybe)	Update	✓ (Maybe)

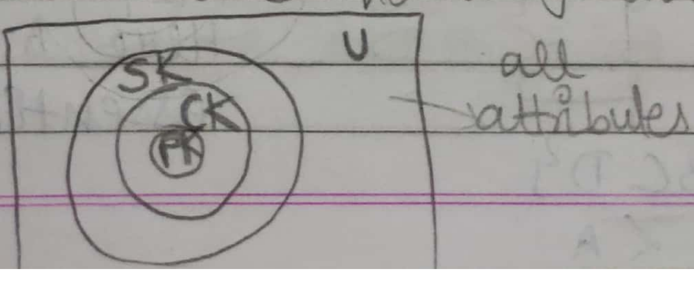
While deletion in Base Table

1) Delete from base - without thinking (which is not correct)

2) Use the clause 'on delete NULL' so the entry of that column in referencing table will be NULL.
This will not work - when the attribute which is Foreign key is also Primary key with pair, we can't have NULL in Primary key. × | M | | | | A | BC | D | ~~M~~ | → NULL

3) Use the clause 'on delete cascade'
By this on deletion the entire row in the referencing table will be deleted.
× | M | | | | | | | M | ×

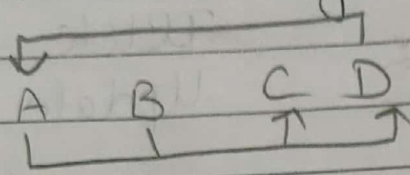
★ For foreign key - it is not compulsorily that name of Base table primary key attribute be same as Referencing table foreign key attribute.
(Name same hovu jaruri nathi)



Foreign key can have NULL and Repetitive values

Date
Page

⇒ R(ABCD) AB→CD, D→A, Find the candidate key.



Here there is no incoming arrow in B. So B is the essential attribute.

$\{B\}^+ = \{B\}$

~~R(ABCD) A→B, B→C, C→D~~

$AB^+ = \{ABCD\}$ ✓

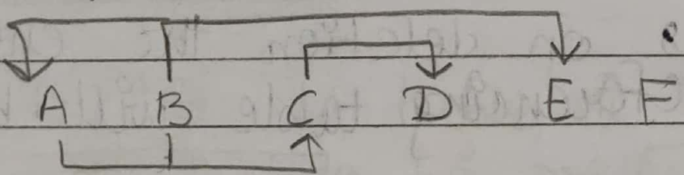
$BC^+ = \{BC\}$

$BD^+ = \{BDAC\}$ ✓

So AB and BD are the candidate key.

Other keys can be: ABC, ABD, BCD, ...

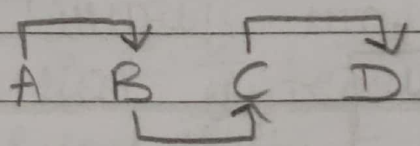
⇒ R(ABCDEF) AB→C, C→D, B→AE



Here FB is the essential attribute

$BF^+ = \{BFAECD\}$

⇒ R(ABCD) A→B, B→C, C→D

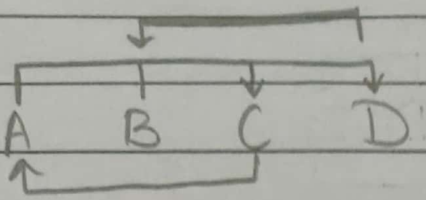


Here A is the essential attribute

$A^+ = \{ABCD\}$

So CK = {A

$\Rightarrow$ R(ABCD) $AB \rightarrow CD$, $C \rightarrow A$, $D \rightarrow B$



Here we don't get any essential attribute.
So find closure of A, B, C, D, AB, AC, AD, ...

$A^+ = \{A\}$ X, $B^+ = \{B\}$ X, $C^+ = \{CA\}$ X, $D^+ = \{DB\}$ X

A B C D AB AC AD BC BD CD
ABC BCD ABD ACD ABCD

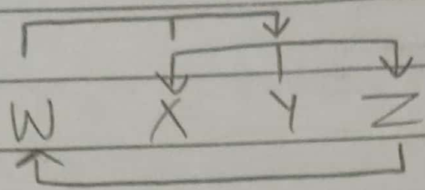
If A mali jai as key we still need to check for B, C, D. Suppose if noone them is key and A is a key, so we don't need to check for AB, AC and AD as they are superset of A. and So on.

Here none of A, B, C, D is a key.
So checking further

$AB^+ = \{ABCD\}$ ✓
$AC^+ = \{CA\}$
$AD^+ = \{ADBC\}$ ✓
$BC^+ = \{BCAD\}$ ✓
$BD^+ = \{BD\}$
$CD^+ = \{CDAB\}$ ✓

So CK = {AB, AD, BC, CD}

⇒ $R(WXYZ)$ $Z \to W$, $Y \to XZ$, $XW \to Y$



Here there is no essential attribute

$W^+ = \{W\}$ $X^+ = \{X\}$

$Y^+ = \{YXZW\}$ ✓ $Z^+ = \{ZW\}$

Now Y is the key so no need to check for XY, YZ, YW

WX, ~~WY~~ WZ, ~~XY~~, XZ, ~~YZ~~

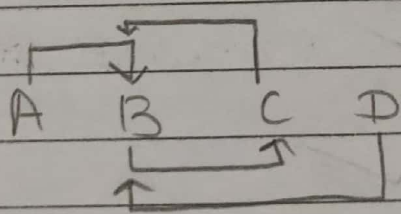
$XW^+ = \{XWYZ\}$ ✓

$WZ^+ = \{WZ\}$

$XZ^+ = \{XZWY\}$ ✓

So CK = $\{Y, XW, XZ\}$

⇒ $R(ABCD)$ $A \to B$, $B \to C$, $C \to B$, $D \to B$



Here AD is the essential attribute

$AD^+ = \{ADBC\}$ ✓

So CK = $\{AD\}$

RELATIONAL ALGEBRA OPREATORS

R(id, name, dept, sem) $(\sigma, \pi, \cup, -, \times, \bowtie)$
tuples - rows
attributes - columns.

Main 2 operations (selection & Projection)

Selection
R(AB, CD)
$\sigma_{A = B \wedge D > 5}(r)$ (Rows)
(gives all the rows having A=B and D>5

Projection
R(A, B, C)
$\pi_{A,C}(r)$ (Columns)
(gives all the rows of column A and C.

Union
Relations : r, s
Union $r \cup s$

Set Difference $r - s$

Intersection
$r - (r - s)$ (Derived opn)
$\therefore r \cap s = r - (r - s)$

JOINS
- Cartesian join product.
- Natural join

Cartesian product: $r \times s$
Occupies more space.

Date
Page

Composition of Operation.
r(A, B) , s(C, D, E)
$\sigma_{A=C}(r \times s)$
Gives the rows from cartesian product of r & s and then from that cartesian we get rows where A=C.

Natural Join $r \bowtie s$
r(A, B, C, D) s(B, D, E)
$\Pi_{A, r.B, C, r.D, E}(\sigma_{r.B=s.B \wedge r.D=s.D}(r \times s))$

Selection $\sigma_p(r)$, p - is the selection predicate
r - is the relation

Theta join - combine a select opⁿ and cartesian product into a single operation.
$r \bowtie_\theta s = \sigma_\theta(r \times s)$

Left Outer join (⟕)
For r ⟕ s, r ni badhi rows, and s ni je common hse and bija ma 'null'

Right Outer join (⟖)
For r ⟖ s, s ni badhi rows, and r ni je common hse e.

Full Outer join (⟗)
Left Outer join + Right Outer join.

... where e.ename = w.ename

Date ____
Page ____

⇒ 1 relation has 2000 rows and other has 2500. On natural join, how many maximum rows will be in natural join.
<u>Answer</u>: 2000 (Best Case)
Left side par je table hoi eni rows karta vadhare nai ave kyrey.
(Worst case): 0.

⇒ In a table there are X and Y attributes set to X=Y=1 initial. XMax and YMax are max. values of X and Y at any time among all records.
X = XMax + 1 , Y = 2YMax + 1 (given).
What is value (output) of following query
Select Y from tbl where X=7;
X = MX + 1 → (i) Y = 2MY + 1 → (ii)
Here M is just for confusing us. Consider M as redundant value.

x	1	2	3	4	5	6	7
y	1	3	7	15	31	63	127

∴ Y = 127 at X=7

⇒ employee (ename, street, city)
works (ename, cname, salary)
company (cname, city)

RELATIONAL CALCULUS

① Find name of all employees who live in city 'Miami'.
$\Pi_{ename}(\sigma_{city=\text{"Miami"}}(employee))$

② Find name of all employees salary > \$100,000.
$\Pi_{ename}(\sigma_{salary>100000}(works))$

Scanned by CamScanner

Date
Page

(3) Find name of all employees who live in 'Miami' and salary >$100,000

$\Pi_{ename}(\sigma_{city="Miami" \text{ and } salary>10000})$ (employee)

⇒ branch (b_name, b_city, assets)
customer (c_name, c_street, c_only)
account (acc_no, b_name, bal)
loan (loan_no, b_name, amt)
depositor (c_name, acc_no)
borrower (c_name, loan_no)

(1) Name of all branches in "Chicago".

$\Pi_{b\_name}(\sigma_{b\_city="Chicago"}(branch))$

(2) Name of all borrowers who have loan in branch "Downtown"

$\Pi_{c\_name}(\sigma_{b\_name="Downtown"}(borrower \bowtie loan))$.

⇒ emp (p_name, street, city)
works (p_name, company_name, salary)
company (company_name, city)
manages (p_name, manager_name)

(1) Find name of all employees who work for First Bank Corporation

$\Pi_{person\_name}(\sigma_{company\text{-}name="First Bank Corporation"}(works$

p-name

Date ______
Page ______

(2) Name and cities of residence of all employees who work for First Bank Corporation.

$\Pi_{p\text{-}name,\ city}(emp \bowtie (\sigma_{c\text{-}name = \text{"First Bank Corporation"}}(works)))$

$\Pi_{emp.p\text{-}name,\ city}(\sigma_{(c\text{-}name = \text{"\_\_\_"}) \wedge (emp.p\text{-}name = works.p\text{-}name)}(emp\ \cancel{\bowtie}\ \times\ works))$

(3) Find names, street address and cities of residence of all employees who work for 'FBC' and earn more than \$10,000 per annum.

$\Pi_{emp.p\text{-}name,\ street,\ city}(\sigma_{(c\text{-}name = \text{"FBC"}) \wedge (salary > 10000) \wedge (emp.p\text{-}name = works.p\text{-}name)}(emp\ \cancel{\bowtie}\ \times\ works))$

(4) Name of all employees in this database who live in same city as the company for which they work.

$\Pi_{emp.p\text{-}name}(\sigma_{(company.cname = works.cname) \wedge (company.city = emp.city) \wedge (emp.p\text{-}name = person.p\text{-}name)}(works \times emp \times company))$

DIVISION OPERATOR takes as input 2 relations called dividend and divisor such that

When 2 tables have same column name & both are primary keys, on Cartesian product

Cartesian Product = Natural Join

⑤ Persons who do not work for City Bank.

$\Pi_{pname}(\sigma_{company\_name <> \text{"CityBank"}}(works))$

⇒ Suppliers (sid, sname, addr)
Parts (pid, pname, color)
Catalog (sid, pid, cost)

① Name of suppliers who supply some red part

$\Pi_{sname}((\sigma_{color = \text{'red'}}(Parts) \bowtie Catalog) \bowtie Suppliers$

② IDs of suppliers who supply some red or green part

$\Pi_{sid}(\sigma_{color = \text{'red'} \lor color = \text{'green'}}(Parts) \bowtie Catalog)$

③ ID's of suppliers who supply some red part or are based at 21 George Street.

$\Pi_{sid}(\sigma_{color = \text{'red'}}(Parts) \bowtie Catalog) \cup \Pi_{sid}(\sigma_{addr = \text{"21 George"}}(Suppliers))$

④ Name of suppliers who supply some red part

Date
Page

5) IDs of suppliers who supply only red part.

$\Pi_{sid}(Supplier) / \Pi_{sid}(Catalog \bowtie \sigma_{color \neq 'red'}(Part))$

Division Operator

r		s	r / s
Fred	Db1	Db1	Fred
Fred	Db2	Db2	Sara
Mike	Db1		
Sara	Db1		
Sara	Db2		
Tina	Db2		

Db1, Db2 bana Fred and Sara mache

Tuple and Domain Relational Calculus

Query in tuple relational calculus is expressed as: $\{t \mid P(t)\}$

set of all tuples t such that Predicate P is true for t.

Quantifiers

$\exists t \in r(Q(t))$ There exists a tuple t in relation r such that Q(t) is true for t.

$\forall t \in r(Q(t))$ For all tuples t in relation r

Not p	$\neg p$
p or q	$p \vee q$
p and q	$p \wedge q$

$\{t[b\text{-}name], t[loan\text{-}no], t[amt] \mid t \in loan \wedge t[amt] > 120$

$\{t \mid t \in loan \wedge t[amt] > 12000\}$

Date ____
Page ____

⇒ Find employees whose salary > 1000.

$\{t \mid P(t)\} = \{t \mid t \in emp \wedge t[salary] > 1000\}$

or

$\{t \mid P(t)\} = \{t \mid t \in emp \wedge t.salary > 1000\}$

If only name attribute

$\{t[name] \mid P(t)\} = \{t[name] \mid t \in emp \wedge t[salary] > 1000\}$

AGGREGATE FUNCTIONS

sum, count, min, max, avg

a	b	c

O/P

C

$g_{sum(c)}(r)$

$branch\text{-}name \; g_{sum(amt)}(account)$

branch-name	sum(amt)

Takes a collection of values and returns a single value as a result.

$G_1, G_2, \ldots, G_n \; g_{F_1(A_1), F_2(A_2) \ldots F_n(A_n)}(E)$

E - relation

$G_1, G_2, \ldots$ list of attributes on which to group

F_i is aggregate fn

A_i is attribute name

$\{t \mid t \in loan \wedge t[amt] > 1200\}$

⟹ Find employees whose salary > 1000.

$\{t \mid P(t)\} = \{t \mid t \in emp \wedge t[salary] > 1000\}$

or

$\{t \mid P(t)\} = \{t \mid t \in emp \wedge t.salary > 1000\}$

If only name attribute

$\{t[name] \mid P(t)\} = \{t[name] \mid t \in emp \wedge t[salary] > 1000\}$

AGGREGATE FUNCTIONS

sum, count, min, max, avg

a	b	c

o/p

c

$g_{sum(c)}(r)$

$branch\text{-}name \; g_{sum(amt)}(account)$

branch-name	sum(amt)

Takes a collection of values and returns a single value as a result.

$G_1, G_2, \ldots, G_n \; g_{F_1(A_1), F_2(A_2), \ldots, F_n(A_n)}(E)$

E - relation

$G_1, G_2, \ldots$ list of attributes on which to group

F_i is aggregate fn

A_i is attribute name

In Relational Algebra while performing Union and Intersection, that both relations should have same number of rows and columns.

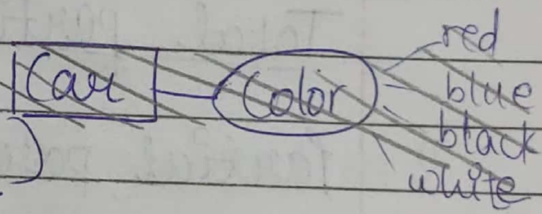
LOGICAL DESIGN

1. E-R Model - (Entity - Relationship Model)
Consists of Entity, Attributes & Relations.

Composite attribute

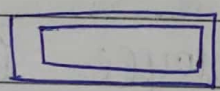
Name
fname mname lname

Multivalued attribute
(Contact No)



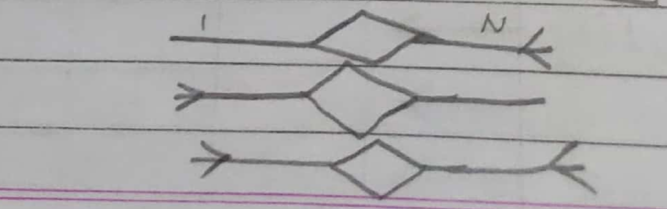
Entity - is something which exists and are distinguishable from others.

Weak entity -



Entity not have any attributes as Primary Key.
It will depend on some other existing entity.

Cardinalities
One-to-One
One to Many
Many to One
Many to Many.



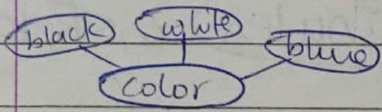
Derived attribute - Dob → age

Simple, Composite, Single valued, Multi valued, Derived

attribute

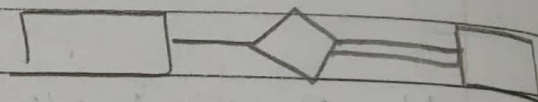
key attribute

derived attribute

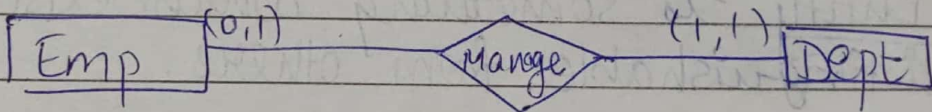


composite

multivalued

Total participation 

Partial participation



(0,1) - Every employee is not a manager

Every employee has department

(1,1) → ~~Every department empl~~ "

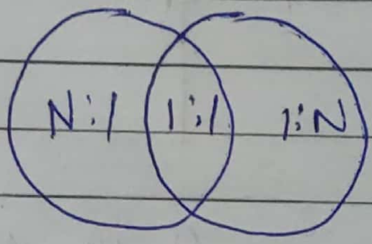
Every department has manager

(single row)

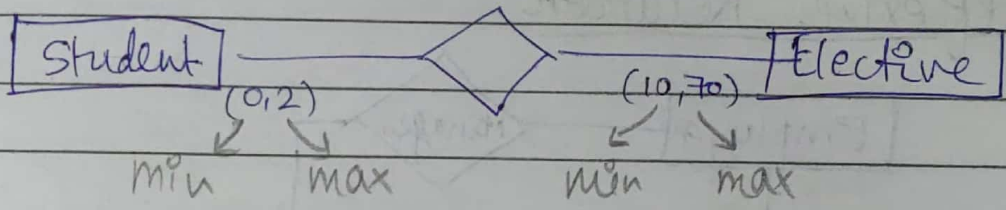
	Name	Roll No	Std
entity →	--	--	--
" →	--	--	--
" →	--	--	--

} entity set - (all rows)

Citizen ⟷ ◇ ⟷ Aadhaar — $\boxed{\text{One}}$ to One
Citizen ⟵ ◇ — MobileNo — One to Many
Student — ◇ ⟶ Dept — Many to One
Faculty — ◇ — Student — Many to Many
M : N



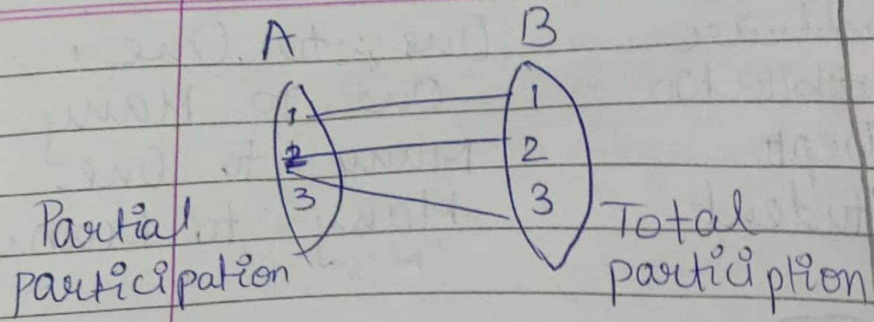
PARTICIPATION CONSTRAINT

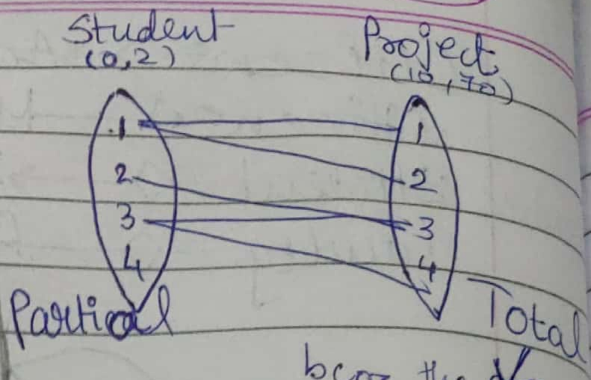


Scenario- A student can select 0 or 1 or 2 elective course. And elective course should have min 10 Students & max 70 students to run

A relationship can have attributes.
It is advised to not have or lessen the number of attributes attached to the relationship.
M : N difficult to remove. So don't do it
1 : 1 attach to any of the 2 entity.
1 : N attach to N side.

because if one 1 side we will need seperate cols for each entry in N side entity. If on many(N) side we just need 1 col.

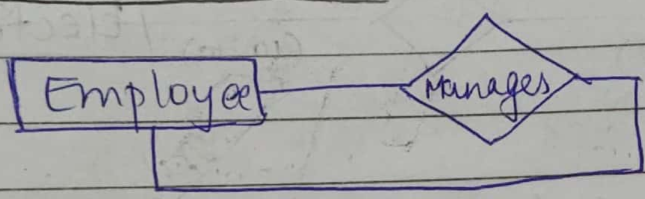




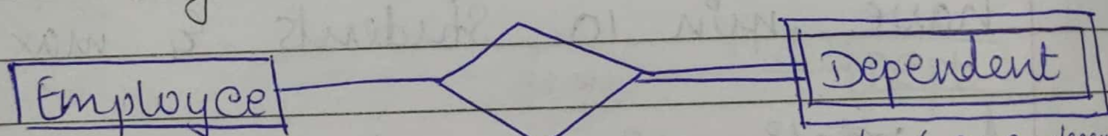
bcoz the course come only if there min 10 students

Wherever there is 0 in min, then partial participation will be there

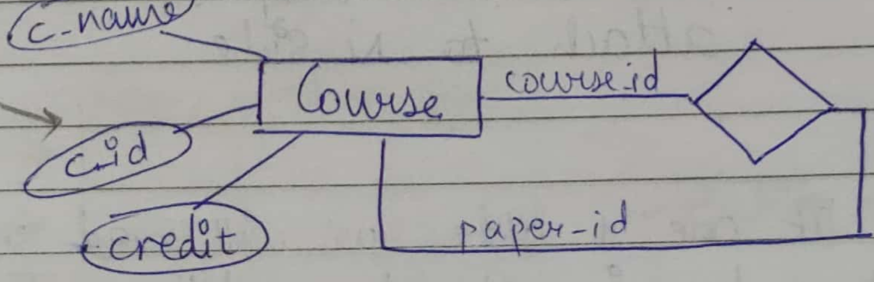
Reflexive Relation:



Weak entity and Total participation

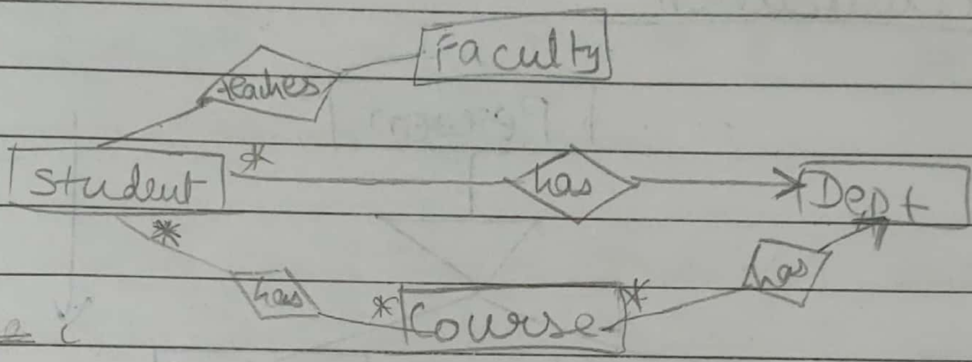


if there is emp he may or may not have dependent
if it is in dependent, he must be in employee.
So, with weak entity there will always be total participation.

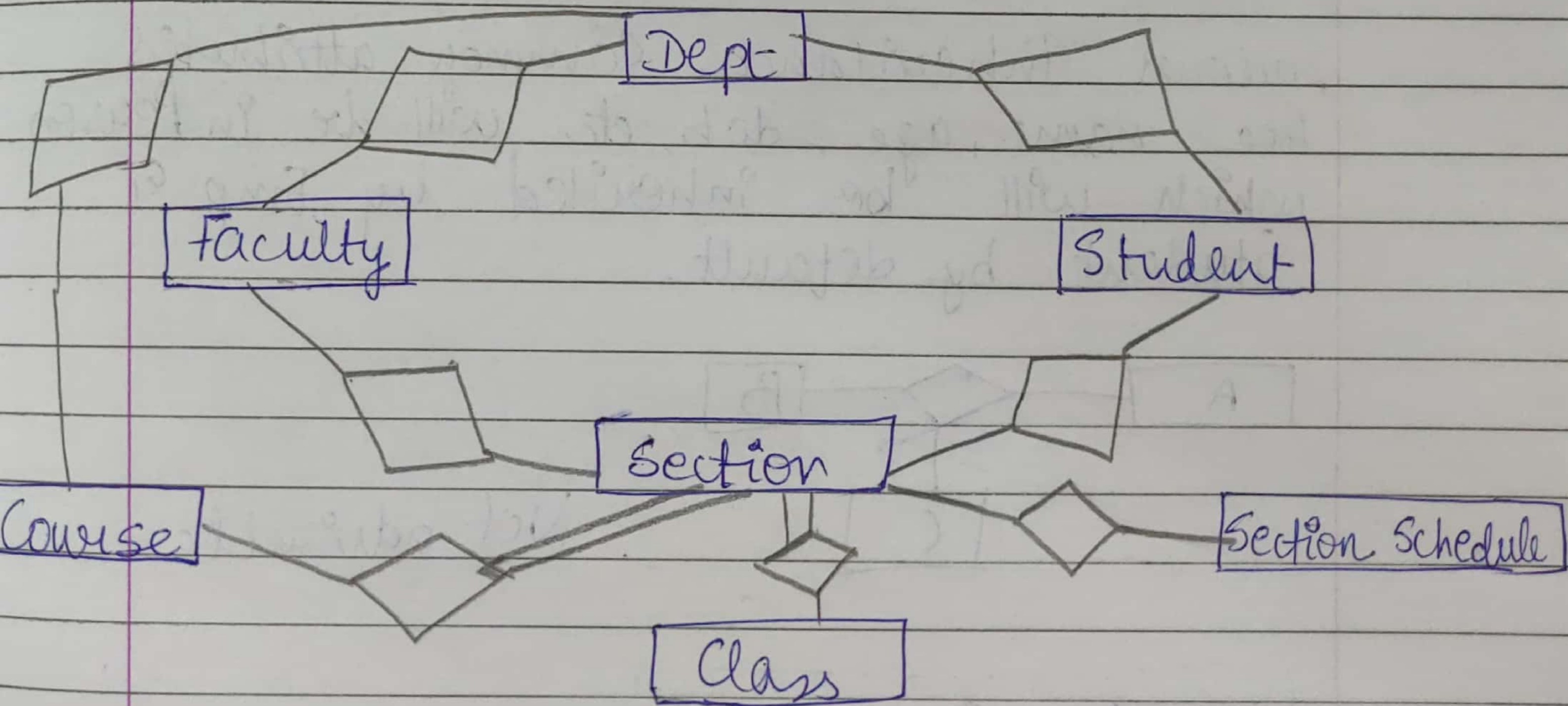


Scanned by CamScanner

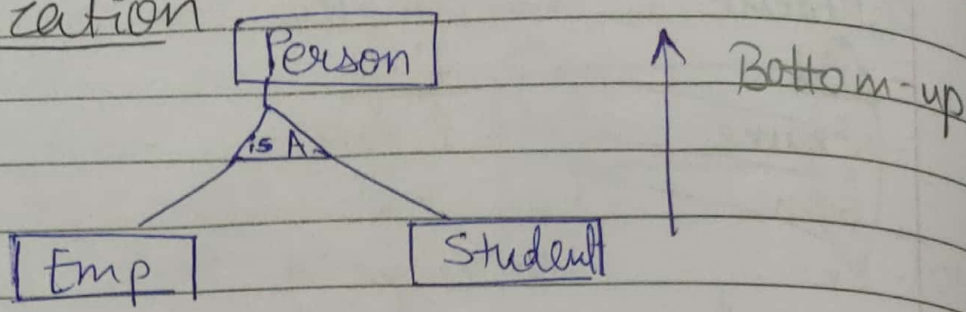
Student	faculty	dept	Course
- s_id	f_id	dept_id	course_id
- s name	f_name	deptname	sem_id
- sem	post		course_name
- cno	course_id		dept_name
F - dept_id	dept_id		
F - courses			



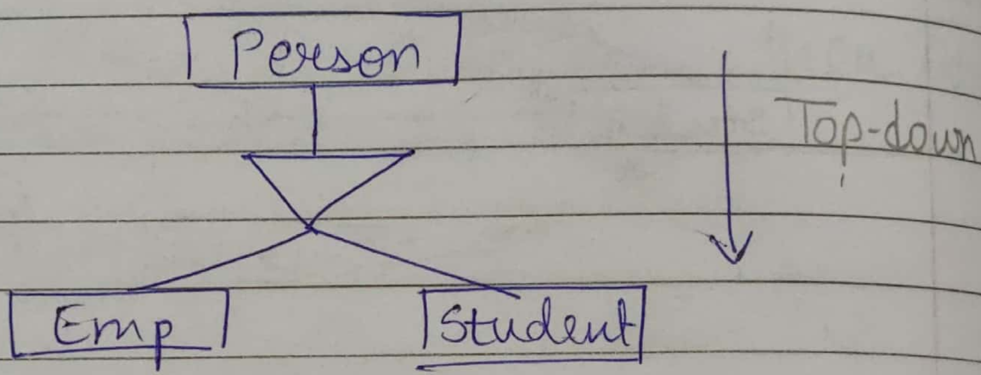
Course {
cid cname



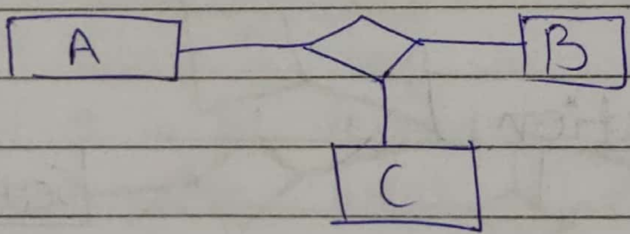
△ ↑ Generalization



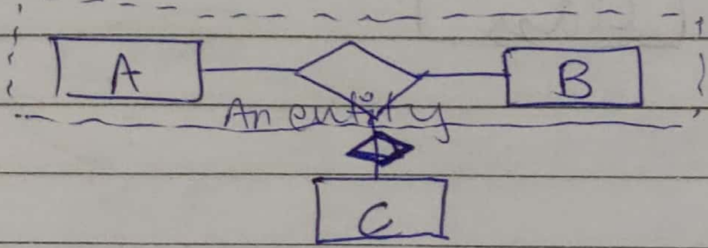
▽ ↓ Specialization



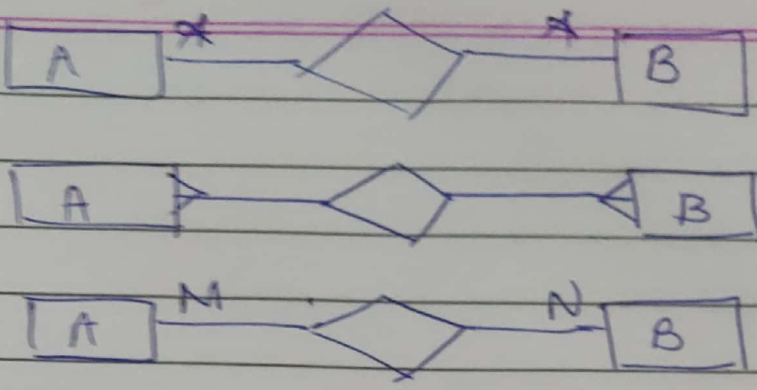
Follows inheritance. Common attributes like name, age, dob, etc will be in Person which will be inherited by Emp & Student by default.



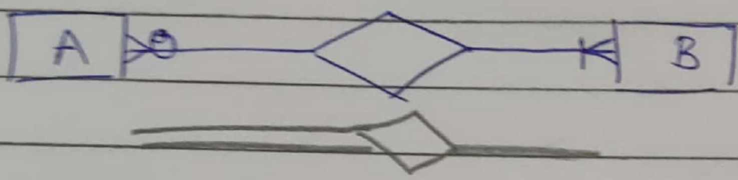
Not advisable



Aggregation

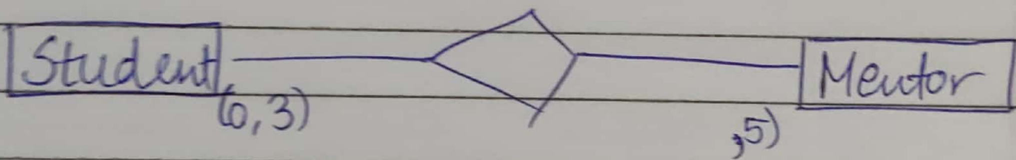


Many-to-Many



Total participation

⇒ A mentor can have atmost 5 students.
Student can have atmost 3 mentors.



Student can study on own. (So 0 min)