Jai Shree Ram.

Name : Anurag Singhal

Standard : ..... Division : ..... Roll : .....

Subject : DBMS

INDEX

(Live SQL).

E.F. Codd → Father of DBMS

1. Database Management System (DBMS) :→

→ Basic Introduc° → 2 tier, 3 tier, 3 schema, (3 level of abstraction).
sup.
↓
(comes in Data Independence)

→ Various Data models :→
Network, Hierarchical,
(Relational, ER), object oriented.

(DBMS) or (RDBMS). — same.
↓
(Relational).

→ ER MODEL :→
conceptual. (Entity - Relationship)

Attributes & its types,
Relationship —" — sup.

sup.
→ Basics of keys :→
- primary key & its characteristic.
Candidate key —" —
Super key —" —

&
Foreign key —" —

→ Normalization →
closure Method → to find candidate key
functional dependencies
1st NF (normal form), 2NF,
3NF,
BCNF

→ Transaction Control & Concurrency →
ACID properties.
R-W problem (Read-write).
W-R "
W-W "
Conflict Serializability.
Recoverability.

Concurrency → locks.
2-PL, timeStamp. (2 phase lock).

→ SQL & Relational algebra.

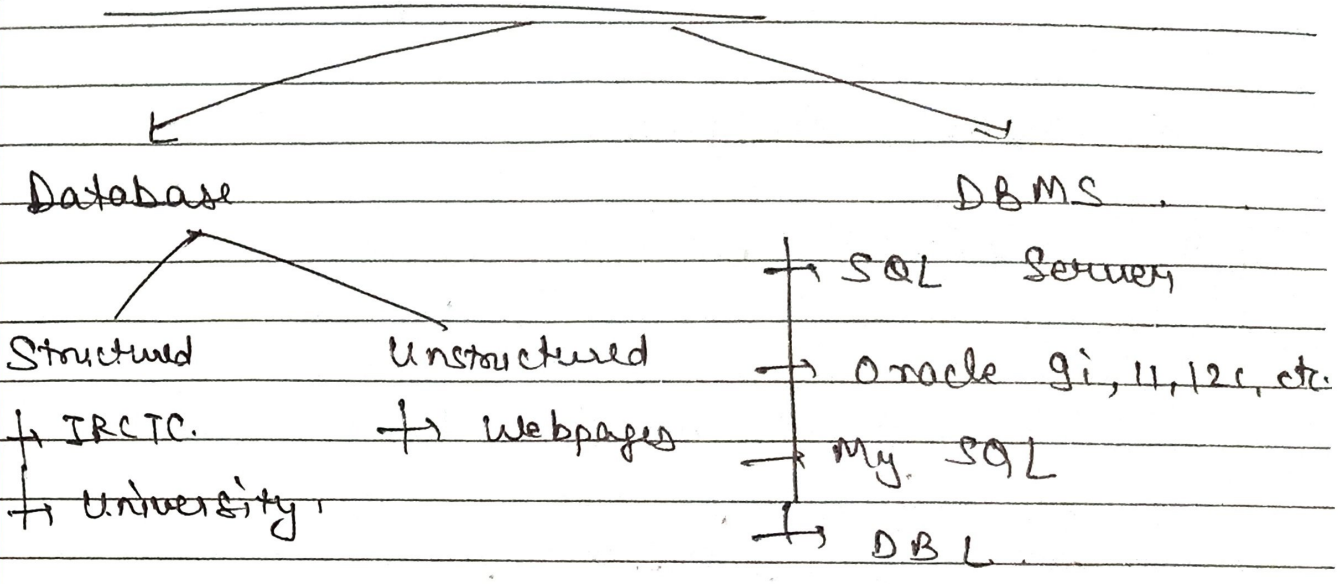
SQL → Structured Query language.
(It is a programming lang.).

DDL command. (Data-Definition).
DML (Data-Manipulation).
DCL
→ Constraint.
→ Aggregate func.
→ Joins
→ Nested Query.
→ In, Not in, any, all.

→ Indexing: → (Single level Indexing).
→ primary, cluster & Secondary Indexing.

→ B tree, B+ tree. (in Multi level).

(2) Database System :→

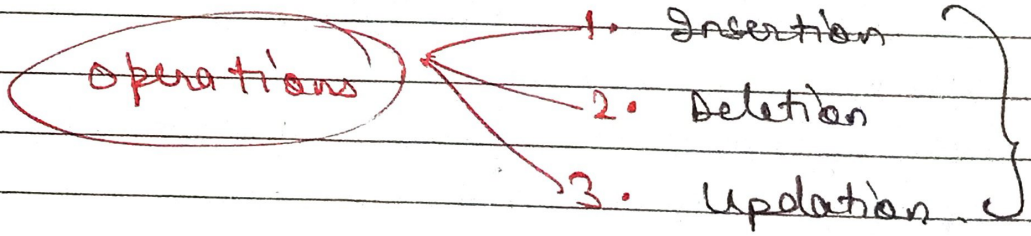


=) Database :→ Collec" of Related Data.

Ex :- Indian Railways & Indian passport have their different data.

=) Structured :→
RDBMS → Rela'al Database Management Sys"

=) DBMS :→ collection of operations



It provide easyness to perform opera's.

=) Diff. companies have made diff. DBMs.

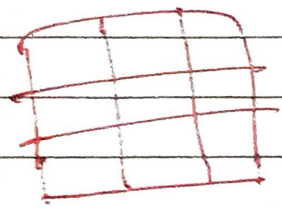
Ex :-
Microsoft Co. ⇒ SQL Server.
(Sequal Server).

⇒ Oracle ⇒. 9i, 11, 12 c, etc.
My SQL.

⇒ IBM ⇒. DBL.

(1) Structured Data! → RDBMS
↳ Relation.

Mean,
We stored in the table form.


→ (Table).

Now, Table is technically called as Rela".

→ Rela" is most usable form.

Now
→ Store Rela"s & access Rela"s is done by the Management System i.e. RDBMS.

⇒ When it is used for Relations, it is called as RDBMS.

Hence,
→ We perform Insert, Delete & updation opera"s on Rela"s.

(2) Unstructured Data! → There is not predefined Structure Ex! →
A webpage is a collec" of photos, videos, Chats, etc. i.e.
There is not a particular format in these.

⇒ Hence, RDBMS works only on the structured Data.

Note:→ 90% of data on this Earth is unstructured.

⇒ ∴ There is more technologies on unstructured data.

Ex:→ Bigdata, Hadoop, etc.

---

3. File System Vs DBMS :→

⇒ File System is used before DBMS.

⇒ User always manages its data in file form.

→ OS has inbuilt File System.

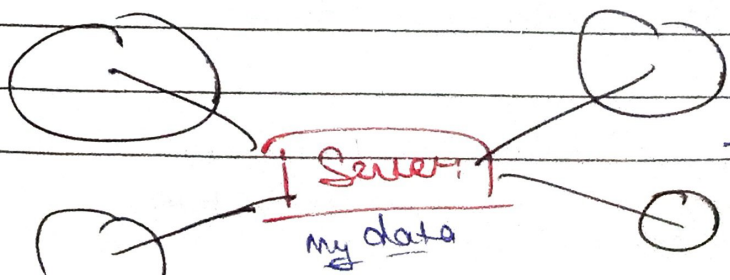
Ex:→ CIFS, NFS File Systems.

→ File System:→ We stores are data in file form. & then into our drives.

→ **Why we use DBMS?**

Bcz, we are using the client-server architecture, means.

→ Our data is at a centralised loca^ & all over the world users can access that.



→ Hence, we can't use file system here. Now, DBMS comes into picture.

1.) If we have to search only for 1 KB, then in file system, complete file comes to us (approx of 25 GB). i.e., More memory usage. Now, DBMS :→ gives us only of 1 KB data from the Server.

(Searching is fast & Memory utilizn is Efficient).

2.) In file system, we require attributes to search data. i.e, Attributes (name, locn, permission, etc.)

Meta Data :→ Data about Data. (Data of a particular file).

In DBMS, no locn, it is totally independent. We don't require any attributes here. (Easeness provide).

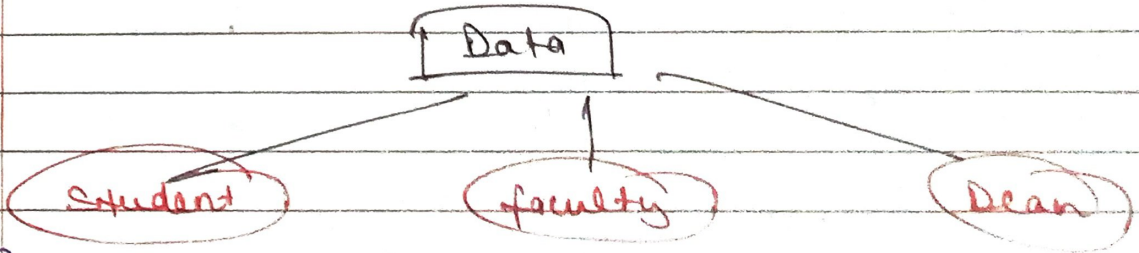
3.) Concurrency :→ means Concurrent Access → Multiple persons can access the data at the same time.

Ex:- IRCTC (Indian Railway System)

File system may show inconsistency when multiple users want access of a file at the same time.
DBMS, have proper protocol for concurrency.

RR
RW
WR
WW
(Read - write)

4.) Security :→ Role based security



(Role based Access Control.)

If we are users, we only get user data.
" - faculty, " - faculty
(O.S.)
File system, has no security for this.
No level-by-level. No Hierarchical.
DBMS has this role-based security.

5.) Data Redundancy :→ (Duplicacy).

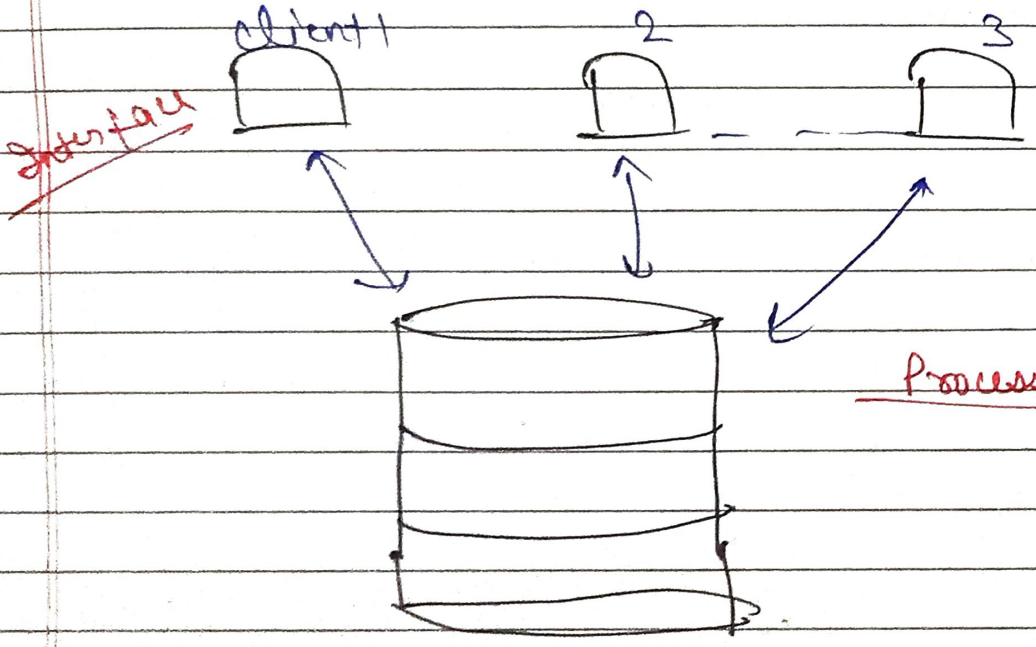
In file system, we can save same content by diff file names.
In DBMS, has many constraints to ensure unique data. Stops redundancy of data.

→ DBMS is at back end of every client server & web apps.

4. 2 tier & 3 tier architecture in DBMS :-

(i) 2 tier :- means 2 layers.

1. client (machine) layer.
2. Database Server. (Data layer).



→ 2 tier also called as client-Server Architecture.

Ex:- i) Indian Railway :- If we reserve ticket by going to railway Stn at ticket window.

ii) Bank :- When physically we draw or post some money.

(client → Request → Database Server → finds info)

→ Hence, limited clients & limited Database to which we access. i.e. why maintenance is very Easy.

But, the users are not limited. They are in big nos. Then, this 2 tier system fails. We call it Scalability.

Security :- → not gud, bcz clients are directly interact with the database.

Advantage :- Maintenance is Easy.

(#) 3-tier :- → (Now, mostly uses).

1.) client layer
2.) Business layer
3.) Data layer.

→ In 2-tier, Query is process in Data-base Server.

But,

→ here, Business layer supports interface. Our query is process in Business layer.

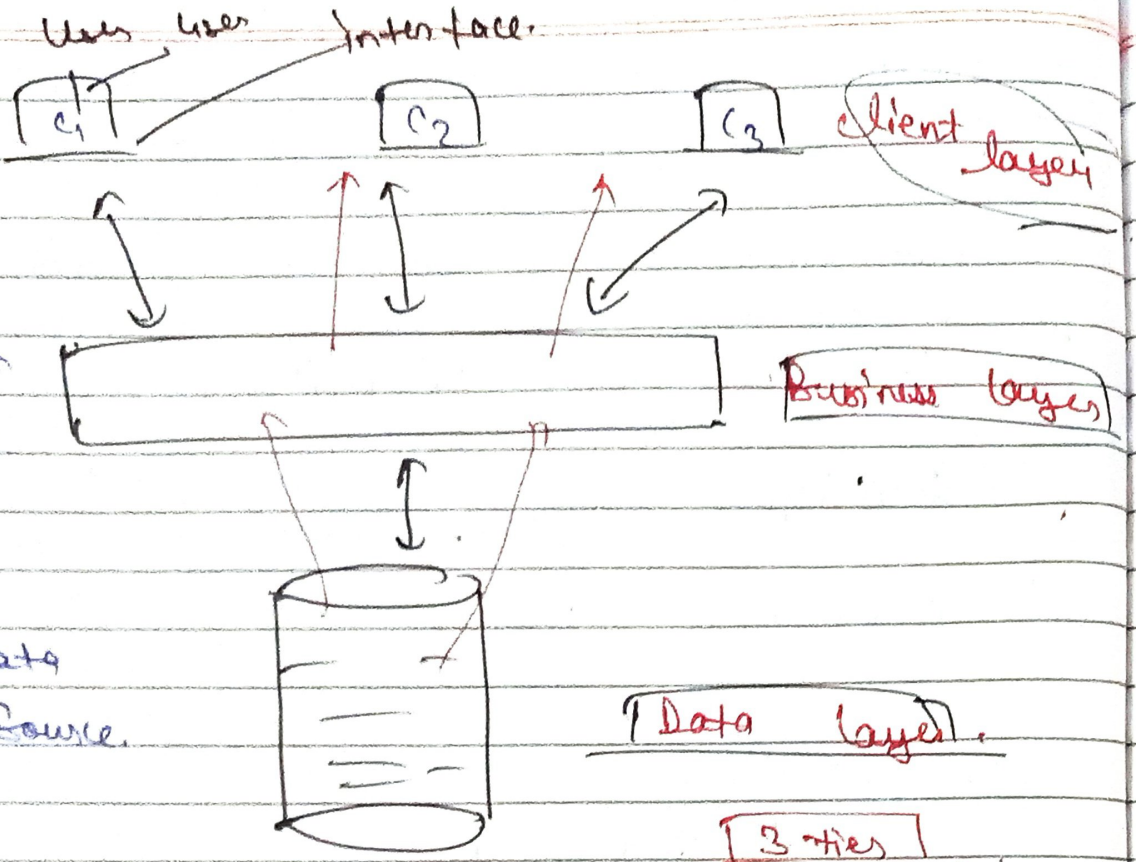
Hence,

We don't give load to Database Server here.

Hence, Business layer acts as Intermediate.

Q:- IRCTC & banking app & Gmail app

Advantage :- 1) Scalability.
2) Security. (no direct interact of data & user)



Here, Maintence is not Easy bcz, it is Complex.

Ex: Web Applica's (APPS) are kind of 3-tier Architecture.

If we go physically to any bank or Railway Sta", then it is 2-tier Architecture.

Schema → Logical Representa" of a database.

⇒ Ex: In RDBMS, data is stored in the form of Tables. (Rela's).

DBMS access & manages the data in this Schema (table) form. It is not actually stored in this table form in drives.

Ex:(i) Schema of a Student :- (Entity) → E-R

Roll. No.	name	addresses

↓ Relationships.

(ii) Course :- (Schema, Entity).

C. ID	name	Dura^n

↓ (Relation ship)

↳ logical Representation (structure).

→ But, we implement it by SQL. (Sequel).
(Integers, characters - - -).
Data Defi^n language (DDL) → to implement Schema. (Or to design)

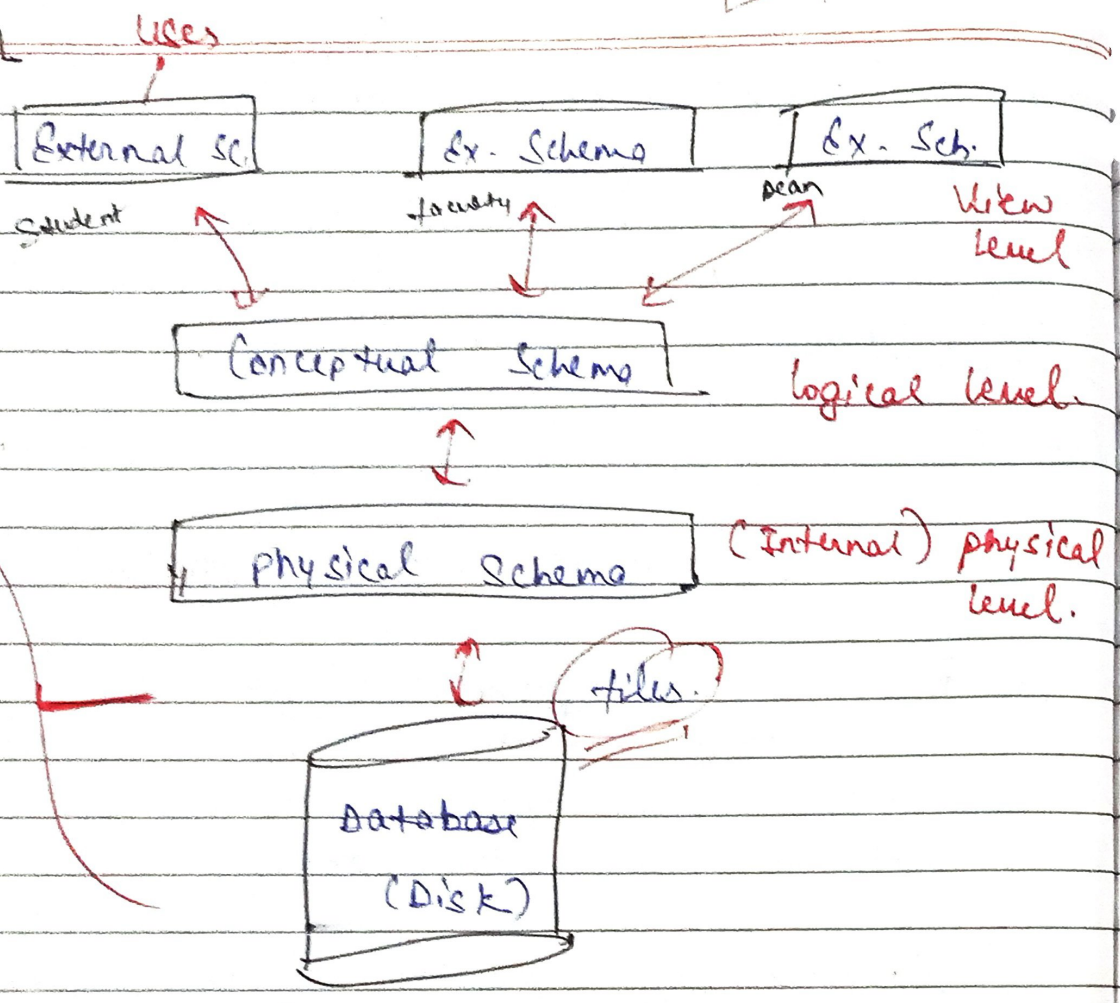
Schema is simply a structure (table form).

---

6. (Three Level of Abstraction) (or) (Three Schema Arch^n.)

→ (We hide the loca^n of where the data is stored, from the user)

Ex:- Our Mails, we don't know physically (exact loca^n) where our mails are stored. Ex- Delhi, UK, US, - - - etc.



External Sch: (View Level) :→ View that will be given to the User.

{ Students have their View.
Faculty have —"— . }.

Ex:→ After Login, which View comes to us is External Schema.

Conceptual Sch: E-R Model

Ex: Student (Roll No, age, addr, ----)

⇒ Info^n of all tables that we use, & their Relo^ship. A type of Blueprint is this.

physical Schema :→ where the data is actually physically present. The loca^ of data.

→ A Normal Data Base Designer is working at level of Conceptual Schema.

→ front End or Interface Developers are at level of Ext. Schema.

→ Database Administrater (who has all control of data) is at physical Schema.

⇒ Centralised — Data at one place.
Multiple — Data all over the World.

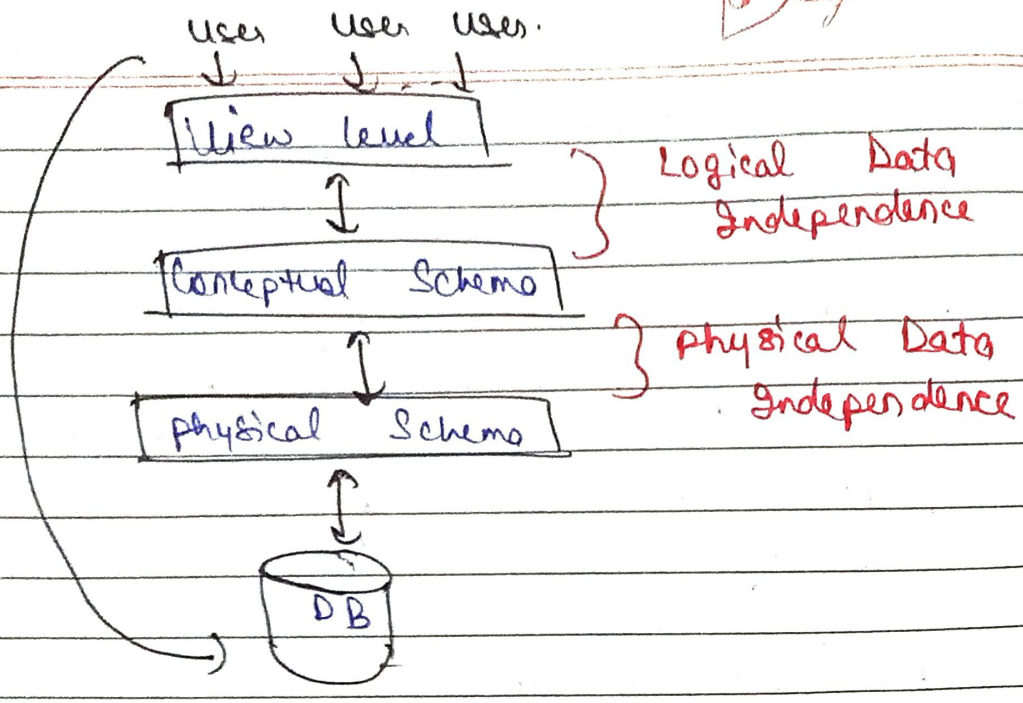
Note:) When we see the data as a user, then we see it in Table form. But, Actually in hard disk, data is stored as files and we apply the layer of DBMS on it.

There are 3 layers b/w the user & the data

7. Data INDEPENDENCE :→

→ Make user independence of data. Hide its loca^ & etc.

→ Conceptual Schema :→ which table we use, how many tables are there, how many attributes, relationship b/w them.



(#) Logical Data Independence! →

Let

→ Student Table. → add by user 1.

Name	Age	Mob. No.

It will not affect the Applica^n program.

→ We don't need to write the App^n pgm again. What user 1 changes want, we give him. But, It will not affect the actual logical structure.

mean, User 2 still can see only col 1 & col 2. Mob No. Colu^m is only ∀ user 1.

→ We use this concept, by Views. (Virtual Table).

→ In actual Table, may be we have 50 col's but user can only see 5-7 col's. So, user think there is only 5-7 col's. But, there are many.

→ Hence, View level don't change by users changes in Conceptual Schema.

Ex:- UMS (University Management System), Shopping website → They can add or delete any col's.

Hence,
It is Logical Data Independence.

physical Data Independence :→ Any change in physical ~~Data Independence~~ Schema won't affect our Conceptual Schema.

Ex:- If we take data from Hard Disk 1 to 2, then data not changes. Tables & Structures remains the same.

Any change in Back End, won't affect the user. It is Data Independence.

---

8) Candidate key & Primary key :→

⇒ key :→ It is one of the attribute in the table.

Use of key: To uniquely identify any 2 tuples in the table.

Roll.No.	S.name	City	Age
1	Reddy	Shamli	20
2	Anurag	Kanpur	21
3	Reddy	Shamli	20

1.) Same 1 student repeats or 2 diff. students with same attri.

2.) 2

4 To identify this, we must have a key.

Ex:) Student Table

1. Aadhar Card
2. Roll No.
3. Reg. No.
4. License No.
5. Voter Id
6. Phone No.
7. Email - Id.

These all attributes can uniquely identify any 2 rows.

The set of all 1-7 values. If we make a set of all of them. Then we call it Candidate key.

{Aadhar Card, Roll No, Reg No ——— Email-Id}.

Now, from above Candidate key set, we choose the one most appropriate & call it Primary key & rest all keys known as Alternative keys.

---

9. PRIMARY KEY :→

(Every lock has its unique key.)

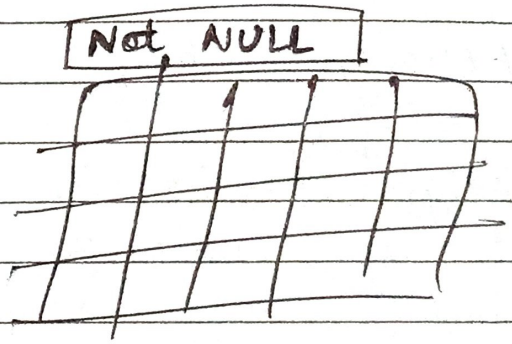
i.e.
We uniquely identify 2 things.

Ex:) Any 2 student can have same name, age, D.O.B. but, some attributes like phone No., Aadhar Card —— are always unique

⇒ Candidate keys :- These are Unique.

→ phone no
→ aadhar card
→ Pan
→ Reg no.
→ Roll no.

} Candidate Keys.



→ phone no, aadhar card vagera को NULL नहीं रख सकते। भरना ही पड़ेगा।

Case 1: We fill wrong aadhar card no.
Case 2: We don't take admn without aadhar card no.

In student case, most appropriate key is
Reg no.
Roll no.

→ PRimary key = { unique + NOT NULL }

→ We don't give primary key to them, they give to us.

Ex :- university give us Reg. No.
passport office give us passport No.
License office give us License No.

- We have only 1 primary key in a Database. Software don't allow us this.

(why need 2 or more, when we only need 1)

---

10 **Foreign key in DBMS** :→

(F.k.)

- **Foreign key** :→ It is an attribute or set of attributes that references to primary key of same table or another table (relation).

→ It Maintains referential Integrity.

Ex:- Student →

P.K (primary key).	Roll no	name	Add.
	1	A	Delhi
	2	B	Shamli
	3	A	Mumbai

Course →

(F.k.)

C. ID	C. name	Roll. No.
C_1	DBMS	1
C_2	networks	2

→ Roll. No is same b/w the student & course. It shows Rela^ship b/w them.

→ In Table 2, Roll No. col^n value ^ references from Roll no. attribute (primary key) in Table 1. take

Q! Can I write Roll. No. '10' in Foreign key (F.k.)?

→ No, Sir, It is not in Table 1 (Roll. No) Until now.

(with f.k.)

⇒ Table 2 is called Referencing Table.

⇒ Table 1 is called Referenced Table. (with p.k.). or Base Table.

* Create table Course
(
Course_id varchar (10),
Course_name varchar (20),
Rollno int references
Student (Roll no).
);

⇒ Code, Ex.

↳ Now, how to write a query after the table is created.

⇒.
Alter table Course
ADD constraint f.k.
foreign key (Roll no)
references Student (Roll no);

Note: (1.) We don't have to keep the name 'Roll. No.' same of both the attributes necessarily, we can also take diff. names.
(2.) In a table, there can be more than 1 foreign key.

(11.) Foreign key :- (Referential Integrity) :-

⇒ Integrity means same value for the database

Ex 1 (1)

→ In mobile phones, diff. price at Flipkart, Amazon & Store so, not Integrity.

(2.) In university, a Student Reg no. to same at every office in the university, in library, in dean office. so, Integrity present here.

Ex 2

Student (Base Table or Referenced Table)

Roll No.	name	add.
1	A	Delhi
2	B	Mumbai
3	A	Chd.
4	C	Chd.

P.K. ✓ — Deletion

C. Id	C. name	Roll. No
C_1	DBMS	1
C_2	networks	2
C_3	Ckt	7

F.K.

Course (Referencing table).

(#) Referenced Table →

1.) Insert :→ No violation (We can easily add).

2.) Delete :→ May cause violation (bcz, if we delete a row & with same roll. no. a row is in referencing table, Then, it's not possible).

Sol^{n} :→ 1.) On delete Cascade
(Also delete from every table).

2.) On delete set Null
(Insert NULL on other tables).

~	~	Roll. No
		NULL

f.k. can, take reference from P.K. of same table, also.

But, if same col^n is also a primary key in that other table. Then, we can't use this sol^n. bcz,

primary key cannot be NULL.
<unique, not NULL>

3.) on delete No Action.
(in this sol^n, our row dont get deleted, first we have to delete from other tables, than we can delete in our table).

3.) updation: → may cause viola^n (bcz, if we make '20' of '2', then now how we can get Roll No '2' in referencing table).
Sol^n !: →
1.) On update Cascade
2.) On update Set NULL.
3.) On update No Action.

(#). Referencing table: →

1.) Insert: → May cause viola^n (bcz, it '1'le not base table, then how it in Referencing table)

2.) Delete: → No violation

3.) updation: → May cause violation. (bcz, we cant make '20' of 2 if '20' is not in base table).

Note: 1.) Same key can be foreign & primary key in a table.
2.) Many table can have a foreign key from a single table by take reference of its P.K.

(12) Q:-

Let $R_1(a, b, c)$ and $R_2(x, y, z)$ be 2 relations in which 'a' is foreign key in R_1 that refers to primary key of R_2. Consider, 4 options →

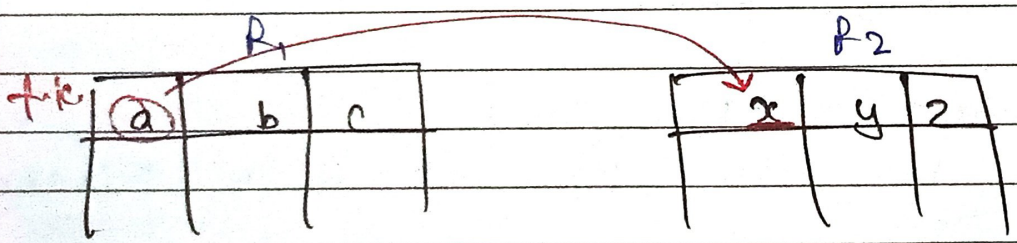
a) Insert into R_1 — ✗
b) Insert into R_2 — ✓
c) Delete from R_1 — ✓
d) Delete from R_2 — ✗

Which is correct regarding referential Integrity ?

1.) option a & b cause violation
2.) option b & c will cause violation
3.) option c & d — " —
4.) option d & a — " — → (Ans)

Let x be p.k. in R_2.

Sol^n :-



Referencing Table. Base Table (or) Referenced Table.

Note :- If f.k. is not there, then we can do anything (insert, delete, update) without any violation.

(13) SUPER KEY in DBMS :→

2). Super key is a combination of all possible attributes which can uniquely identify 2 tuples (row) in a table.

→ Candidate key is minimal.

Eg:-

Candidate key (C.K.) = Roll No. (also Super key)

Roll no.	name	age

- Roll no, name
- Roll no, age
- Roll no, name, age

all are Super key.

Candidate key (C.K.) के साथ कुछ भी add कर दो, तो super key हो जाएगी।

2). name, age × (not Super key).

→ Super set of any Candidate key is Super key.

Q:- R(A_1, A_2, ... A_n) then how many Super keys are possible.

If → A_1 is Candidate key.
→ A_1, A_2 are candidate keys.

Ans:- power Set :→ how many Subsets can be possible of given set.

Eg:- → A_1 A_2 A_3 (Either take or not take So, 2 possibility)

$2 \times 2 \times 2$ → 8

→ 2^3.

Now, Sol^n → R ($A_1, A_2, \ldots A_n$).

i) ⇒ (इसमें A_1 तो आएगा ही).

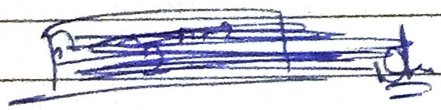
A_1
A_1, A_2
A_1, A_3
A_1, A_2, A_3, A_4 .

R ($A_1, A_2, A_3, \ldots A_n$)
↙ $n-1$
Compulsory.

$1 \times \underbrace{2 \times 2 \times \ldots 2}_{n-1}$.

$\therefore$ 2^{n-1} Ans.

ii) R ($A_1, A_2, A_3, \ldots A_n$)
$n-2$



→ A_1, A_2 both C.K.

when A_1 →	when A_2 →
A_1	A_2
A_1, A_2	A_2, A_1
A_1, A_2, A_3	A_2, A_1, A_3
2^{n-1}	2^{n-1}

But, some sets are common (bcz $A_1A_2 = A_2A_1$).

$\therefore$ Common are those who has both A_1, A_2.

$\therefore$ those are 2^{n-2}.

Now,

$2^{n-1} + 2^{n-1} - 2^{n-2}$ ✓

(or)

$2^{n-1} - 2^{n-2}$.

iii) $A_1 \cdot A_2$ combined is C.K.

then,

2^{n-2} ✓.

iv) $A_1 A_2$, $A_3 A_4$ are C.K.

2^{n-2} 2^{n-2}

Now,

$A_1 A_2 A_3 A_4$	$A_1 A_2 A_3 A_4$
$A_1 A_2 A_3 A_4 - -$	$A_3 A_4 A_1 A_2 - -$

To remove these common Elements.

$A_1 A_2 A_3 A_4 - - - - A_n$

2^{n-4}

$2^{n-2} + 2^{n-2} - 2^{n-4}$ ✓

$2^{n-1} - 2^{n-4}$.

06.

14. E-R Model → (Entity Relationship Model)

→ used for logical representation.

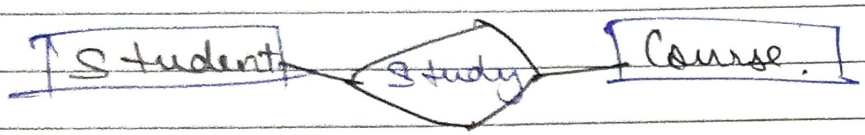
→ To see logical structure before implementation (Design).

→ It does the job of database design.

Entity - Any object which has physical existence is Entity.

Ex:- Student (Roll no, age, address)
(Entity) attributes

→ Relationship → Relationship b/w 2 or more Entities



Relationship b/w student & course is of Study.

⇒ Student (Roll no, age, address)
Entity type (Schema).

⇒ We implement these by using SQL.
(Structured Query lang.).

→ Entity.
→ Attributes - Characteristics of Entity.
↳ types.
→ Relationship - ↳ types → 1 to 1
1 to many
many to 1
many to many } 4 types

Entity → Student → represented by rectangle
Attribute →
Relationship →

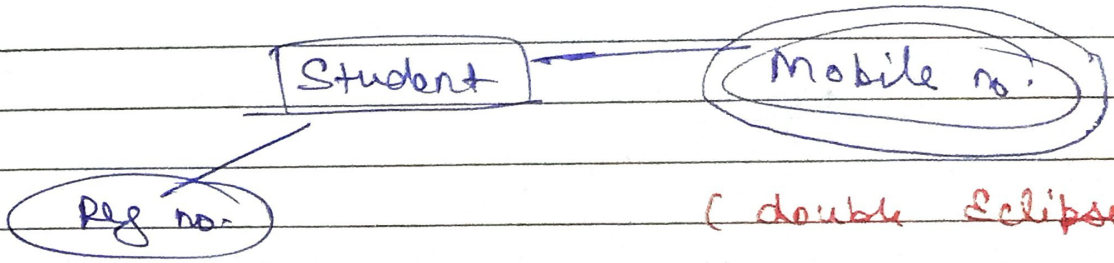
(15.) Types of Attributes in ER Model :-

[Student]

1.) Single vs Multivalued attributes :→

Single → Reg no.

Multivalued → Mobile no. or address. (May be more than 1 C.No. of a student)



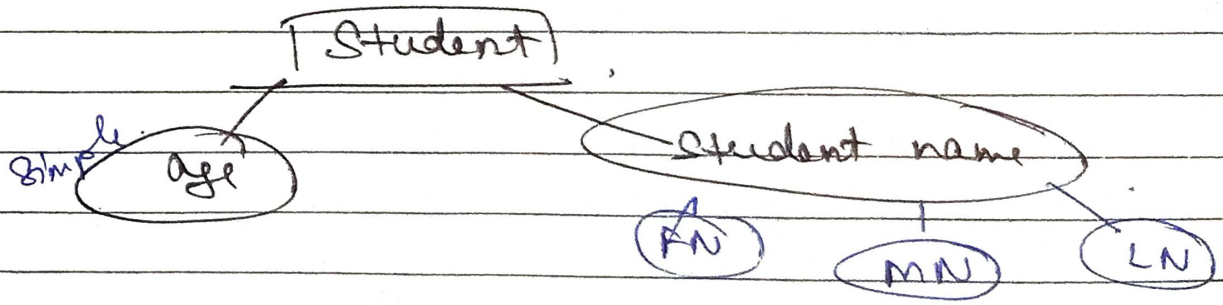
(double Eclipse).

2.) Simple vs Composite Attributes.

Simple - can't be further divided.

Composed - composed of more than 1 value.

Ex:- name (first name, middle name, last name)



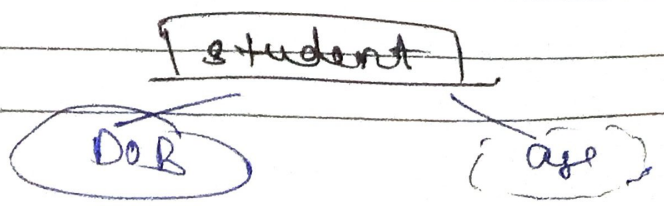
3.) Stored & Derived Attributes :→

Stored → These are stored & can't be derived.

Ex - DOB

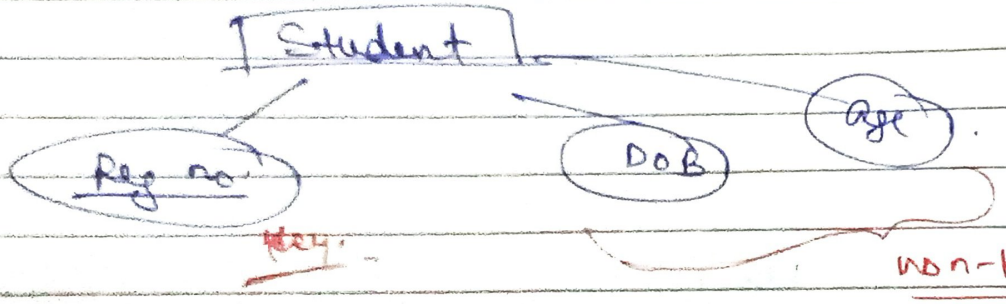
Derived → Ex - Age. (derived from D.O.B.)

dotted Eclipse.



4.) Key vs Non-key Attributes :- →
Key - used to uniquely identified.
Unique (No Repitition)

Ex:- Reg No is always unique for a student.
Represent with underline (___).

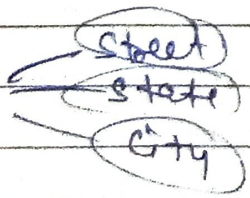


5.) Required vs optional Attributes :- →
Required → These are Mandatory (*) Name
optional → can also be leave. D.oB,
Add.,

6.) Complex Attribute :- →
(Composite + Multivalued)

Eg:- If a student have 2 Residential Add, & in each Add., he have 2 phone no.
Multivalued.

Add. is Composite

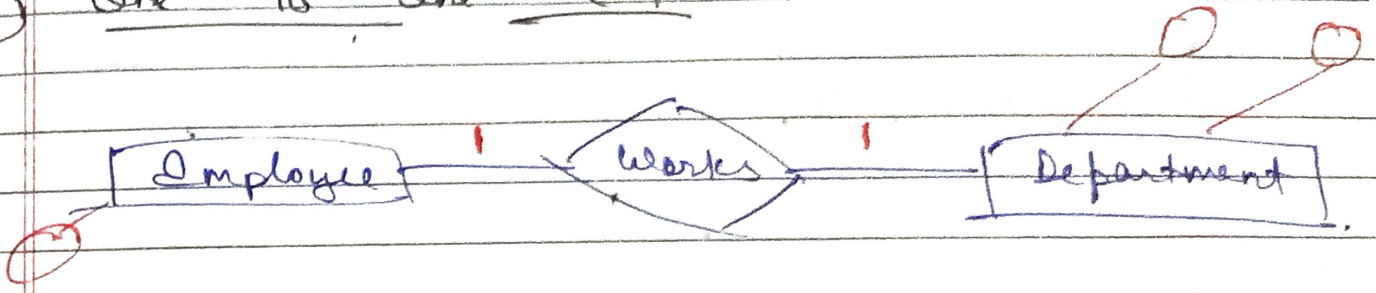


16. Degree of Relationship :- → (Cardinality).
→ how the Entities are connected with each other.

4 types! →

1) 1 - 1 — One to one
2) 1 - M
3) M - 1
4) M - M (M - N) — Many to many

(1) One - to - One (1 - 1) : →



Convert Entity into Table : →

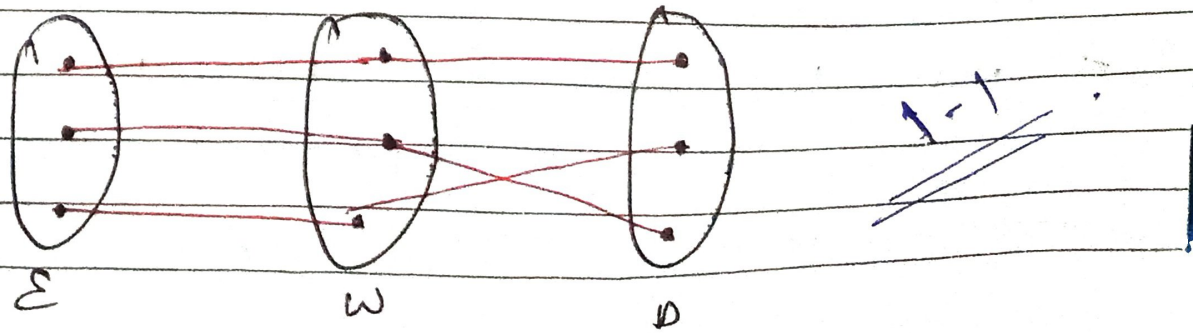
(Relo'ship का भी Table बनेगा, तो हमें Employee & Department का Rel^ बनाना है।)

Rel'ship Table! → Attributes? → 2 always (primary keys of both the table)

E.ID & D.ID

These, E.ID & D.Id works as a foreign key (F.K)

→ When we have to enter data in this Relo^n ship table, then we have to see Rel.ship (1-1, 1-M, ---).



→ P.K. = Either E_Id or D_Id.
(primary key)

E_Id	E name	age
E_1	A	20
E_2	B	25
E_3	C	28
E_4	A	24
E_5	B	25

f.k. f.k.

EID	D_ID
E_1	D_1
E_3	D_2
E_2	D_3

D_ID	Dname	Loc
D_1	IT	Banglore
D_2	Prod.	Delhi
D_3	HR	Delhi

Both

PK = E_ID

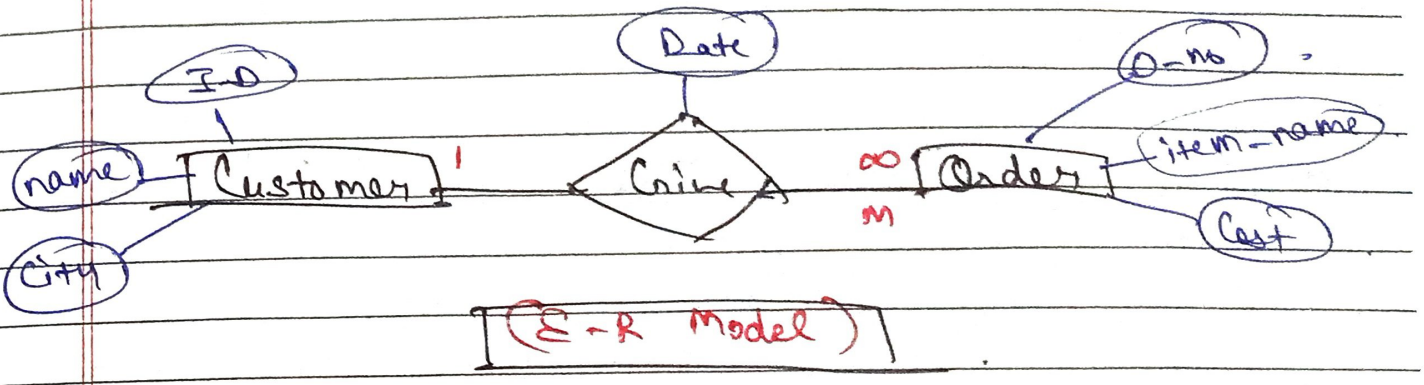
* Can we Merge?
Now, In Table 1 & Table 2, E_ID is the primary key.
Hence, we can merge Table 1 & 2.

E_ID	E name	age	D_ID
E_1	A	20	D_1
E_2	B	25	D_3
E_3	C	28	D_2
E_4	A	24	—
E_5	B	25	—

→ Now, we have 2 Table at final !!.
(Merge Table & Department Table)

Every Table must have its primary key.

(17) One to Many Rela^n ship :→ (1-M).

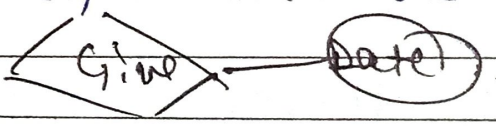


(E-R Model)

→ When we physically implement the E-R Model, then we need Relational Model & we use Tables in Rela'al Model.

2) Rela'ship may have its attribute.

(Ex-1)



& we call it Descriptive Attribute.

f.k. f.k.

Id	Name	City
C_1	A	Tal.
C_2	B	Delhi
C_3	C	Mumb
C_4	A	Mumb

I-D	O-no	Date
C_1	O_1	—
C_1	O_2	—
C_2	O_3	—
C_2	O_4	—

O-no	Item name	Cost
O_1	Pizza	100
O_2	Burger	200
O_3	pasta	300
O_4	Cold-Drink	400

Here O-no is always diff. & so unique.

P.K. = (O_no).

Note:- Always P.K. of the many side in (1-m) is also the p.k. of the relo"ship Table.

Can we Merge Tables? (many diff side merge).

Yes.

By Relo"ship & Order Table.

(bcz both have same P.K.).

=>

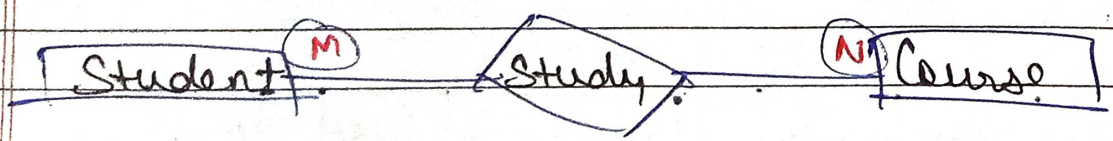
ID	O_no	item name	Cost	Date
~	~	~	~	~

Now, 2 Tables.

(M-1) is also same like this.

18. Many-to-Many Relo"ship :→

(M-N)



f.k. f.k.

Roll no	name	age
1	A	16
2	B	17
3	A	16
4	C	17
5	D	15

Base Table

Rollno	C-id
1	C1
2	C2
1	C2
2	C4
3	C3

Referencing Table

C-id	name	Credit
C1	Maths	4
C2	phy.	4
C3	Chem.	4
C4	Hindi	4

Base Table

Many - Many.

P.K. in Referencing Table (Relationship Table) :→

Roll. No repeats &
C-id also repeats.

So, Roll no. & C-id both make P.K. Combinely.

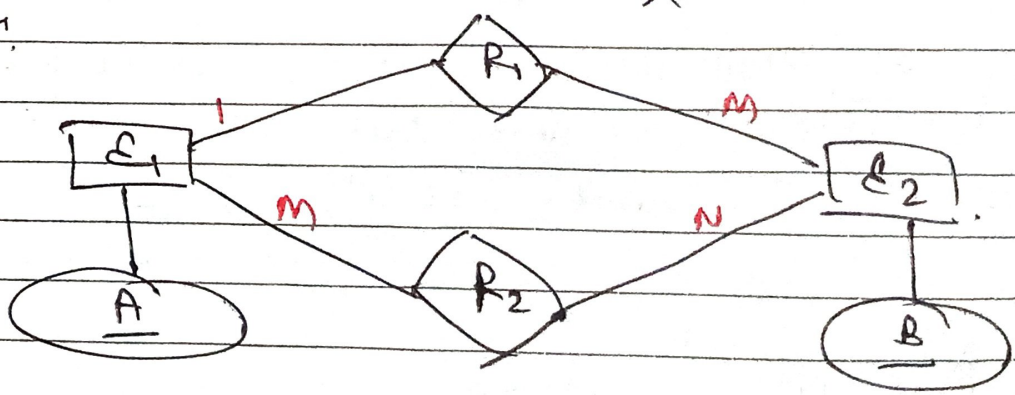
i.e, Composite key = Roll no. C-id

Can we Reduce Tables ?

→ No : bcz, p.k. is combined.

Note:1 p.k. in Relationship Table depends on Relationship (1-1, 1-M, ... M-M).

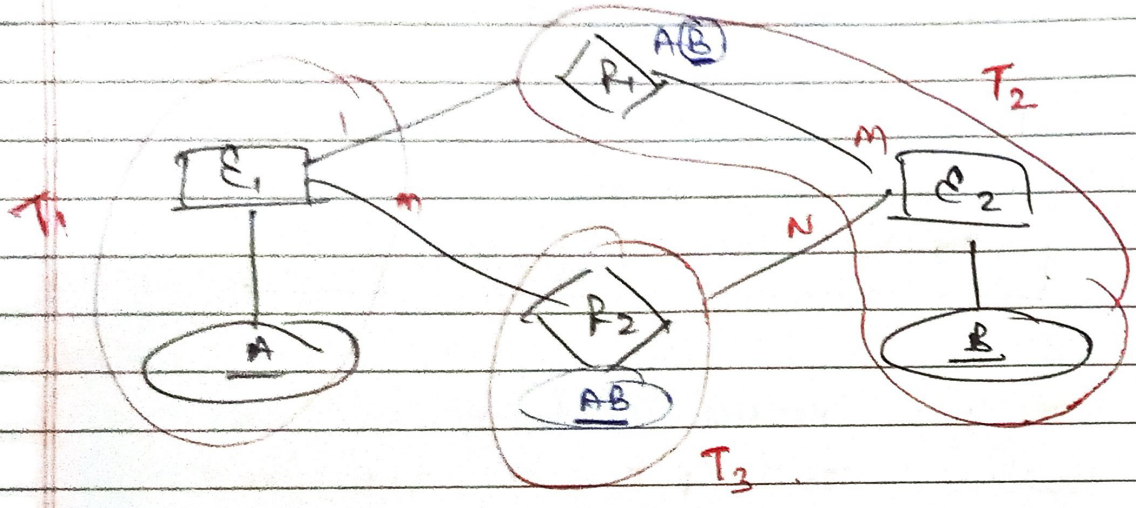
Q. Q:→



* What is the min^m no. of tables required to represent this E-R model into Relational Model ?

a) 2 b) 3
c) 4 d) 5

E_1
E_2
R_1
R_2
} 4 Tables ... But Minimum ?



Hence,

$T_1 = E_1$
$T_2 = R_1 E_2$
$T_3 = R_2$
} 3 Tables ✓

E_2 का सभी attributes $R_1 E_2$ के Combined Table में आ जाएगा। So Now, we also don't need Separate E_2 Table. $\boxed{\text{Min}^M = 3}$ ✓

---

(20) **Normalization** :→

It is a technique to remove or Reduce Redundancy (Duplicacy) from a Table.

→ There are 2 types of duplicacy in Database!→
1.) Row level
2.) Column "

(#) Row level!→

S-Id	S-name	Age
1	Ram	20
2	Varun	25
1	Ram	20

Same (Duplicate) =!

Row level.

→ We use the concept of primary key (P.K.)
We set a P.K. to any appropri. attribute.

Primary key (Unique + Not Null).

P.K. will take care of this duplicacy.

(#) Column-level!→

	Student		Course		Faculty		
P.K.	S-Id	S-name	C-Id	C-name	F-Id	F-name	Salary
	1	Ram	C_1	DBMS	f_1	John	30,000
	2	Ravi	C_2	Java	f_2	Bob	40,000
	3	Nitin	C_1	DBMS	f_1	John	30,000
	4	Anurag	C_1	DBMS	f_1	John	30,000
	5	Varun	C_{10}	MBBS			

⇒ 4 columns are same in many rows.

~~Anomaly~~

→ Insertion Anomaly
→ Deletion "
→ updation "

(→ Anomaly means problem, occurs on special occasion.)

Now,

S.Id	S.name	C.id	Cname	F.id	Fname	Salary
1	Ram	C_1	DBMS	f_1	John	~~30k~~ 40K
2	Rani	C_2	Java	f_2	Bob	40k
3	Nitin	C_1	DBMS	f_1	John	~~30k~~ 40K
4	Amritpal	C_1	DBMS	f_1	John	~~30k~~ 40K.
5	Varun	—	—	—	—	—
		C_{10}	MBBS			

1) Insertion Anomaly :→
→ We want to add data of a new st.
Let, Varun
Let
University starts a new Course,
C_{10} – MBBS
→ We can't insert this info. in table.
Even, we — " — the new faculty data bcz,
→ We only introduce the new Course C_{10}.
We don't talk about the student. and we don't have S. Id.
→ We also remain it (s.id) NULL - bcz it is a P.K.
→ So, we can't insert directly.

It is the our Insertion Anomaly.

2.) Deletion Anomaly :->

We have simple query = Remove the database of Roll. No. 1.

```
Delete from student
where S-id = 1.
```

:- It will delete the whole row.

We don't face any problm here.

Now,

We have to delete the data of Roll. No 2.

```
Delete from student
where S-id = 2
```

→ Row 2 is deleted fully from database.

Now,

Row 2 is blank there.

Now,

tell us who is teaching to Roll No. 2 & what was the course name of Roll. No 2.

likely be, It was only one student who was studying that particular course & that particular faculty is teaching that course.

⇒ We here, only delete the detail of student but, bcz of him, all the info. get deleted.

Course Info — lost
Faculty Info — lost.