

Jai Shree Ram

Name : Anuag Singh

Standard : Division : Roll :

Subject : DBMS

INDEX

S. No.	Date	Title	Page No.	Teacher's Sign/Remarks
1.		DBMS intro	1 - 2	
2.		Database System & its types		3 - 5.
3.		file system VS DBMS		5 - 7 .
4.		2 tier & 3 tier architecture		8 - 10 .
5.		Schema		10 - 11
6.		3 level of Abstraction (or) 3 schema Arch.		11 - 13
7.		Data Independence		13 - 15.
8.		Candidate key & Primary key		15 - 16
9.		Primary key		16 - 18
10.		foreign key		18 - 19
11.		foreign key (Referential Integrity)		19 - 21
12.		Ques" of Referential Integrity		22
13.		Super key in DBMS		22 - 25
14.		E-R Model		25 - 26
15.		Types of attributes in ER Model		27 - 28
16.		Degree of Relationship (Cardinality)		28 - 30
17.		One to One Relationship		31 - 32
18.		Many to many Relationship		32 - 33 .
19.		Ques" practice		33 - 34
20.		Normalization		34 - 39
21.		First normal form (1NF)		39 - 40
22.		Closure Method		41 - 44
23.		Functional Dependency		44 - 47
24.		2nd Normal form (2NF)		47 - 50 .
25.		3rd Normal form (3NF)		50 - 52.
26.		BCNF (Boyce Codd Normal Form)		52 - 54
27.		Lossless & Lossy Join Decomposition		54 - 57.
28.		All normal forms with Real Life Examples		57 - 58.

(LIVE SOL)

S. No.	Date	Title	Page No.	Teacher's Sign/Remarks
29.		Minimal cover	38 - 60.	
30.		Question on Normalization	61 - 64	
31.		Q1 - Find normal form of a Rel^n	65 - 69	
32.		Normalization Questions	69 - 70.	
33.		Ques^ Explained on Normalisation	70 - 74	
34.		Covers & Equivalence of F.D.	74 - 76.	
35.		Dependency preserving Decomposit^	76 - 79	
36.		— " — Example	79 - 80.	
37.		Joins & Its types	81 - 82.	
38.		Natural Join	82 - 84	
39.		Self Join	84 - 86	
40.		EQUI-Join	86 - 87.	
41.		Left Outer Join	88 - 89	
42.		Right Outer Join	89 - 90	
43.		Relational Algebra (R.A.)	90 - 91	
44.		projection in Relational Algebra	91 - 92	
45.		Selection in — " —	92 - 93	
46.		Cross/ Cartesian product in — " — " —	93 - 94	
47.		Set Diff. in R.A.	94 - 95	
48.		Union oper^ in R.A.	96 - 97	
49.		Division oper^ — " —	97 - 99	
50.		Tuple Calculus in DBMS	99 - 103	
51.		Intro to SQL	104 - 106	
52.		All types of SQL Commands	107 - 108.	
53.		Create Table in SQL with execution	108 - 109.	
54.		ALTER Command (DDL) in SQL	109 - 110.	
55.		D/LW Alter & update	111 -	
56.		D/LW Delete, Drop & Truncate	112 - 113.	
57.		Constraints in SQL	113 - 115.	
58.		SQL Queries & Sub-Queries (i) & (ii)	115 - 116.	
59.		(iii) & (iv) part.	116 - 117.	
60.		(v) part	117 - 118.	
61.		(vi) part	119	
62.		(vii) part	120.	

S. No.	Topic Name	page. No.
63.	Use of IN and Not IN	121 - 122
64.	Use of IN and Not IN in Subquery	122 - 123
65.	Exist & Not Exist Subqueries	123 - 124
66.	Aggregate Func's in SQL	124 - 126
67.	Co-related Subquery in SQL	126 - 127
68.	DB Joint, Nested Subquery & Co-related Subquery	127 - 129
69.	Find N th Highest Salary using SQL	129 - 132
70.	3 Imp Questions on SQL Basic Concepts	133 - 134
71.	PL- SQL	135
72.	Transac ⁿ Concurrency	136 - 137
73.	ACID properties of a Transac ⁿ	137 - 139
74.	Transaction States	139 - 141
75.	SCHEDULE (Serial Vs Parallel Schedule)	141 - 142
76.	Types of problems in Concurrency	142 - 144
77.	Read-Write Conflict (OR) Unrepeatable Read probm	144 - 145
78.	Irrecoverable Vs Recoverable Schedule In Trans.	146 -
79.	Cascading VS Cascadless Schedule	147 - 149
80.	SERIALIZABILITY	150 - 152
81.	Conflict Equivalent Schedules	152 - 154

E.F.Codd → Father of DBMS



Date: _____
Page: _____

①

(1)

Database Management System (DBMS) :-

→ Basic Introduct' → 2 tier, 3 tier, 3 schema,
(3 level of abstraction), ^{graph.}

↓ (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 — " — . ^{graph.}

→ 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.

(SQL is a programming lang.).

D D L command. (Data - Definition).

D M L (Data - Manipulation).

D C L

→ 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

DBMS

Structured

Unstructured

↳ IRCTC.

↳ University

→ SQL Server

→ webpages

→ Oracle 9i, 11, 12c, etc.

→ MySQL

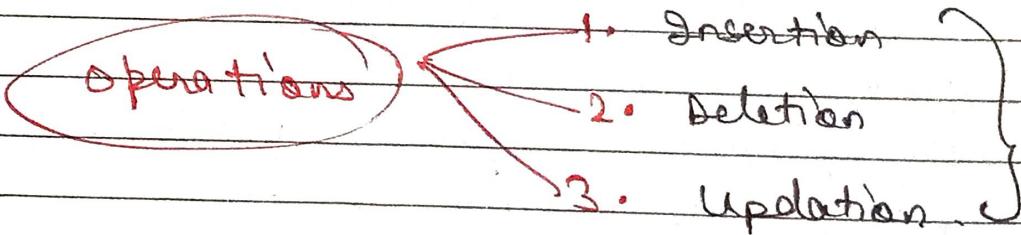
→ DBL

⇒ Database : → "Collector" of Related Data.

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

⇒ Structured : →

RDBMS → Relational Database Management Sys.

⇒ DBMS : → collection of operations

It provide easiness to perform opera's.

⇒ Diff. Companies have made diff. DBMS.

Ex:-

Microsoft Co. ⇒ SQL Server.

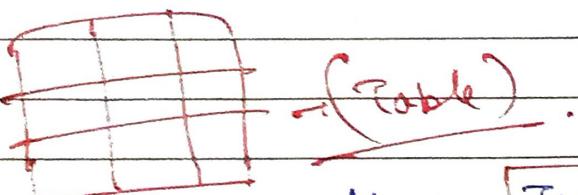
(Sybase Server).

⇒ Oracle → 9i, 11, 12 c, etc.
My SQL.

⇒ IBM → DB2.

(4) Structured Data → RDBMS
↳ Relation

Mean,
we stored in the table form.



Now, Table is technically called as Relaⁿ.

→ Relaⁿ is most usable form.

Now

→ Store Relaⁿ's & Access Relaⁿ's is done by the Management System i.e. DBMS.

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

Hence,

→ We perform Insert, Delete & update op's on Relaⁿ's.

(5) Unstructured Data: There is not predefined structure ↳
A webpage is a collection of photos, videos, chats, etc. i.e.,
There is not a particular format in these.

- Hence, RDBMS works only on the structured Data.

Note: Govt. of data on this Earth is unstructured.

- i.e., There are more technologies on unstructured data.
Ex:- Bigdata, Hadoop, etc.



file System Vs DBMS

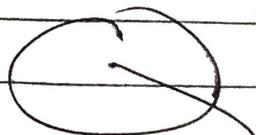
- file system is used before DBMS.
→ user always manages its data in file form
→ OS has inbuilt file system.

Ex:- CIFS, NFS file systems.

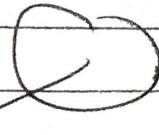
A file system like stores our data in file form & then into our drives.

Why use DBMS?

- bcz, we are using the client-server architecture, means,
→ Our data is at a centralised locaⁿ & all over the world users can access that.



I Server
my data



client (users)

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).

∴

More memory usage. Now,

DBMS! → gives us only of 1 KB data from the Server.

(Searching is fast & Memory utilization is efficient).

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

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

In DBMS, no locaⁿ, it is totally ~~completely~~ independent. We don't require any attributes here.

(Easiness provide).

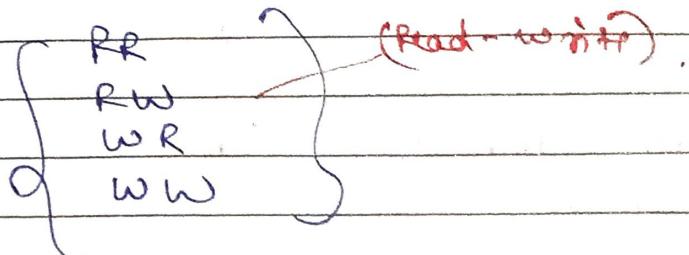
3.) Concurrency: → means concurrent Access.

Multiple persons can access the data at the same time.

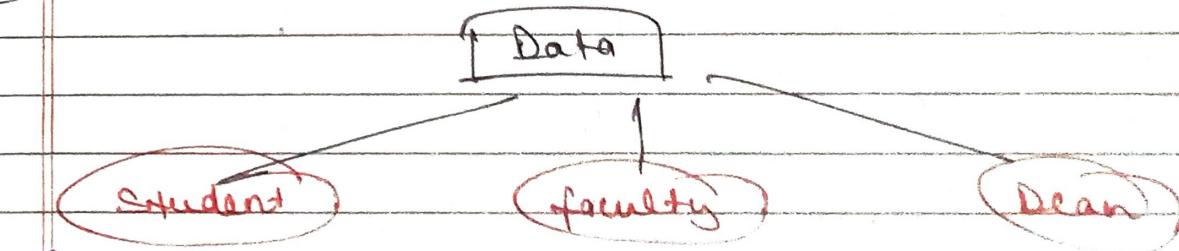
Ex: IRCTC (Indian Railway) System

File system may show inconsistency when multiple users want access of a file at the same time.

DBMS, have proper protocol for concurrency.



4.) Security: → Role based security.



(Role based Access Control).

If we are user, 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 system

5.) Data Redundancy & (Duplication).

In File System, we can save same content by diff file names.

In DBMS, has many constraints to ensure unique data. Steps redundancy of data

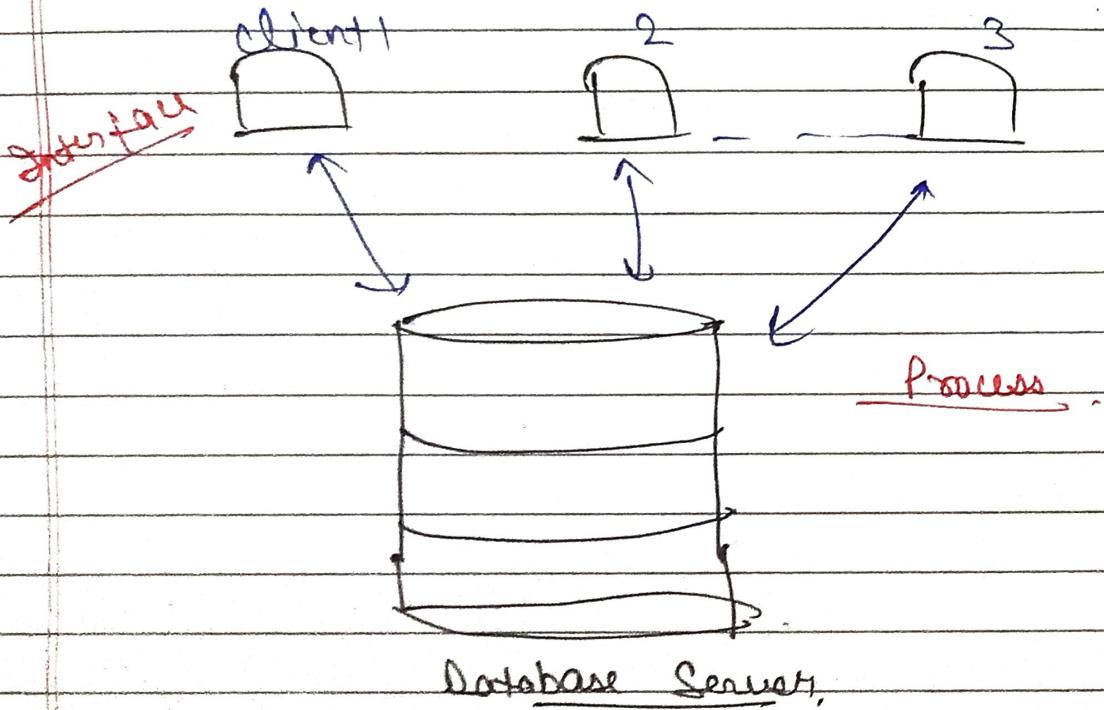
• DBMS is at back end of every Client Server & Web Applic.

4.

2 tier \rightarrow 3 tier architecture in DBMS !

(+) 2 tier means 2 layers.

1. Client (Machine) layer.
2. Database Server, (Data layer).



→ 2 tier also called as Client-Server architecture.

~~Ex:~~ ~~Ex:~~ Indian Railway → If we desire ticket by going to Railway Sta' at ticket window.

~~Ex:~~ Bank → When physically we draw or post some money.

(client → request → Database Server).

↑ gives info

→ Hence, limited clients & limited Database to which we access. i.e., 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 gd, bcs, clients are directly interact with the database.

of Advantage: Maintenance is easy.

3-tier: → (Now, mostly used).

- 1.) Client layer
- 2.) Business layer
- 3.) Data layer.

→ In 2-tier, query is process in Database Server
But,

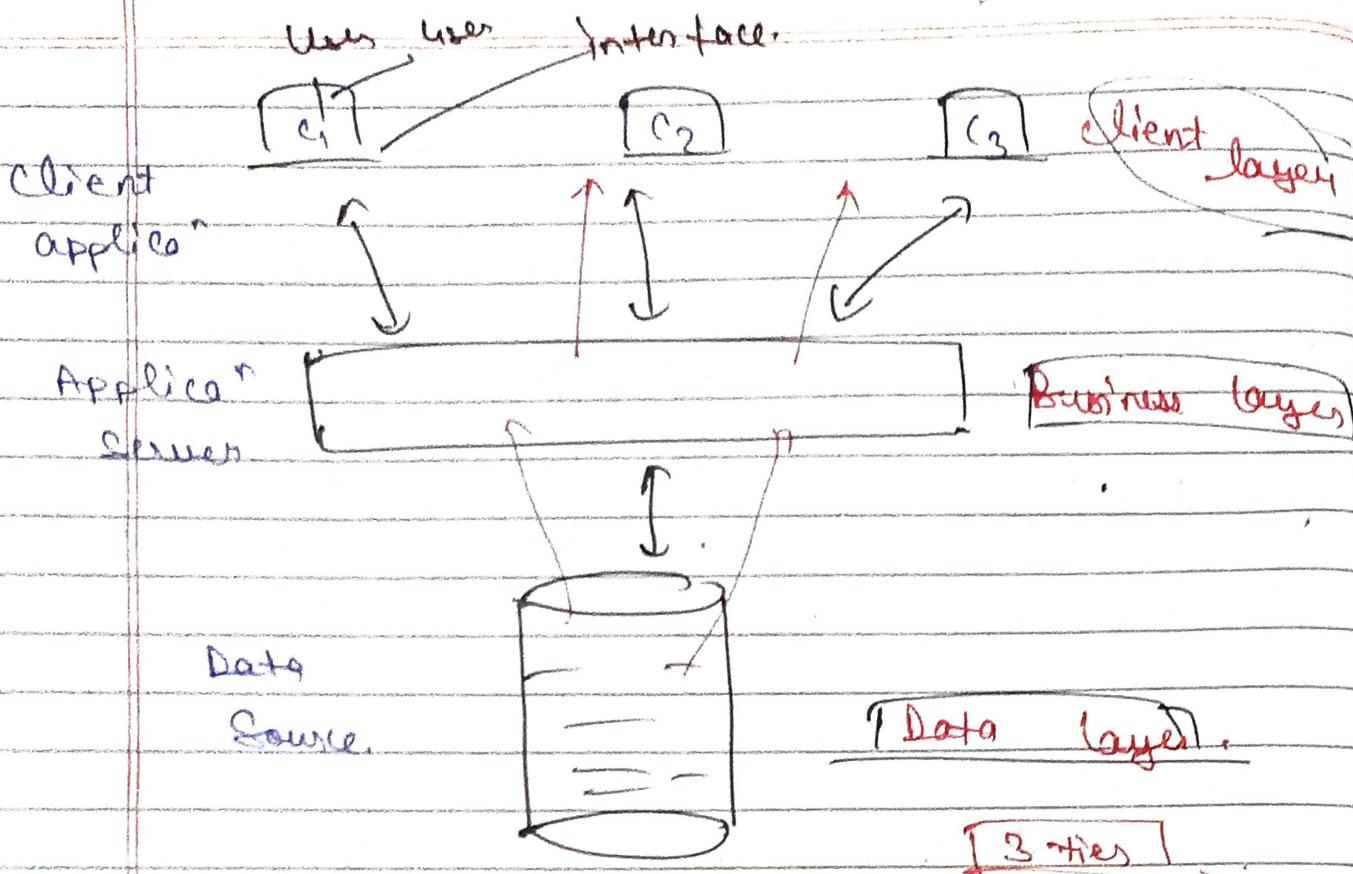
→ here, Business layer supports interface.
Our query is process in business layer
Hence,
we don't give load to Database Server here.

Hence, business layer acts as Intermediate.

Ex: IRCTC & banking app & Gmail app

Advantage:

- 1.) Scalability.
- 2.) Security. (no direct interact of data b/w user)



(Here, Maintenance is not easy bcz,
it is complex)

Ex: Web Applicⁿs (APPS) are kind of
3-tier architecture

If we go physically to any bank or
Railway staⁿ, then it is 3-tier architecture.

(S) Schema → Logical Representaⁿ
of a database.

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

DBMS Access & manage the data in this
Schema (Table) form. It is not actually stored
in this Table form in drives.

Ex. Q1) Schema of a Student: → (Entity) E - R

Roll No.	name	address
----------	------	---------

✓ Relationship.

Q2) Course: (Schema, Entity).

C. ID	name	Dur ⁿ
-------	------	------------------

✓ (Relationship).

↳ Logical Representation (structure).

→ But, we implement it by SQL.
(integer, character, ...). (sequential).

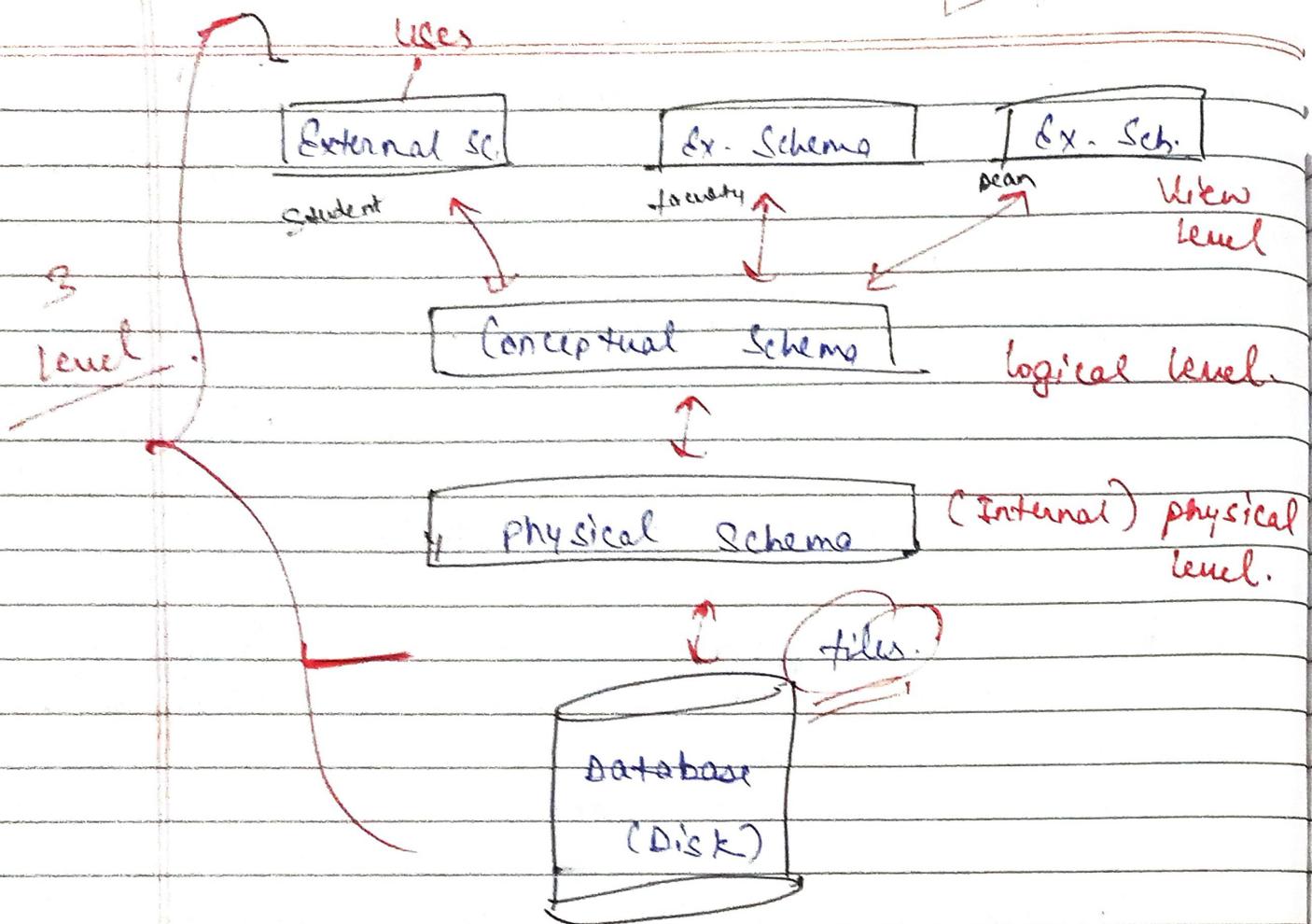
Data Defin' language (SQL) → to implement
Schema. (or to design).

Schema is simply a structure. (table form).

Q3) (Three Level of Abstraction) (or) Three Schema Archi.

→ Rule hide the "locⁿ" of where the data is stored, from the user).

Ex: Our Mails, we don't know physically (exact locⁿ) 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, add, — — —)

→ Informal of all tables that we use, & their relationship. A type of Blueprint is this.

* physical Schema: → where the data is actually physically present. The loc' of data.

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

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

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

→ Centralised — Data at one place.
Multiple — Data all over the world.

Ques:

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.

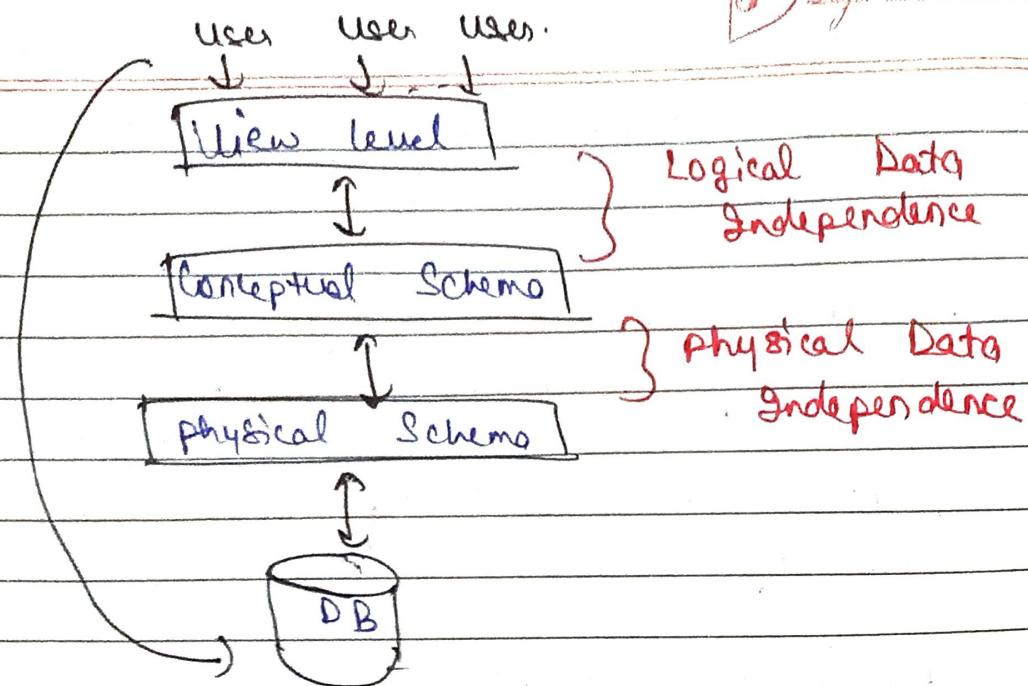
There are 3 layers b/w the user & the data

(F)

Data INDEPENDENCE : →

→ Make user independence of data. Aside its loc' & etc.

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



② Logical Data Independence! →

Let,

- Student Table. → add by user 1.
- | Name | Age | Mob. No. |
|------|-----|----------|
| | | |

It will not affect the application program.

- we don't need to write the App' 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. col is only for user 1.

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

- In Actual Table, may be we have so col's but user can only see S-T col's. So, user think there is only S-T col's. But, there are many.

→ Hence, view level don't change by user's changes by users in Conceptual Schema.

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

Hence,

It is logical Data Independence.

It physical Data Independence! → Schema
Any change in physical Data Independence 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 2 student repeats
or
2) 2 diff. students with same attr.

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.

all
These 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.

————— X ————— X —————

Q.)

PRIMARY KEY : →

(Every lock has six unique key.),
ie,

use uniquely identify 2 things.

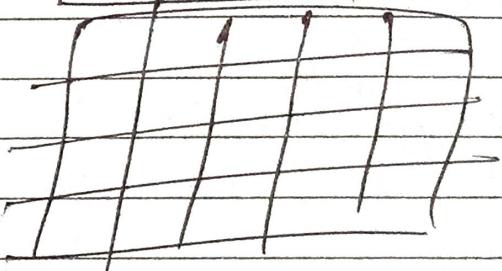
~~Ex:-~~ Any 2 student can have same name, age,
D.O.B. But, some attributes like
phone No., Aadhar Card —— are always unique.

→ Candidate keys! → These are Unique.

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

Candidate Keys.

Not NULL



→ phone no., aadhar card never be NULL अस्ति क्वाप्ति
लाला ! आरना की पड़ताल !

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
be 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 ~~can~~ 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.I.C.)

- Foreign key : → It be an attribute or set of attributes that references its primary key of ~~the~~ same table or another table (relation).
- It maintains referential integrity.

Ex:

Student →

Course →

(F.I.K.).

P.R.K (primary key).	Roll no.	name	Add.	C. ID	C. name	Roll. No.
	1	A	Delhi	C ₁	DBMS	1
	2	B	Chennai	C ₂	networks	2
	3	A	Mumbai			

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

→ In Table 2, Roll No' colⁿ value "refers" from Roll no' attribute (primary key) in Table 1.

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

→ No, b/c, If 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),

, Code,
 Ex.

Roll_no int references

Student (roll no).

);

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.