卐 JAI MATA DI 卐

# DBMS → Interview Prep

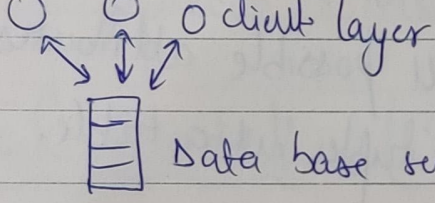
Data → fact that can be recorded (structured / unstructured)

Database → collection of related data

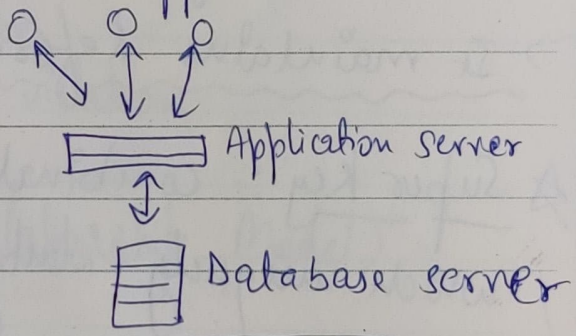
why? → Multiple people retrieving same data
↳ Large size / security / Redundancy
↳ Role based access control

## 2 Tier Architecture Vs 3-tier Architecture

2 layer → Easy Maintainency
But less scalable
& less secure



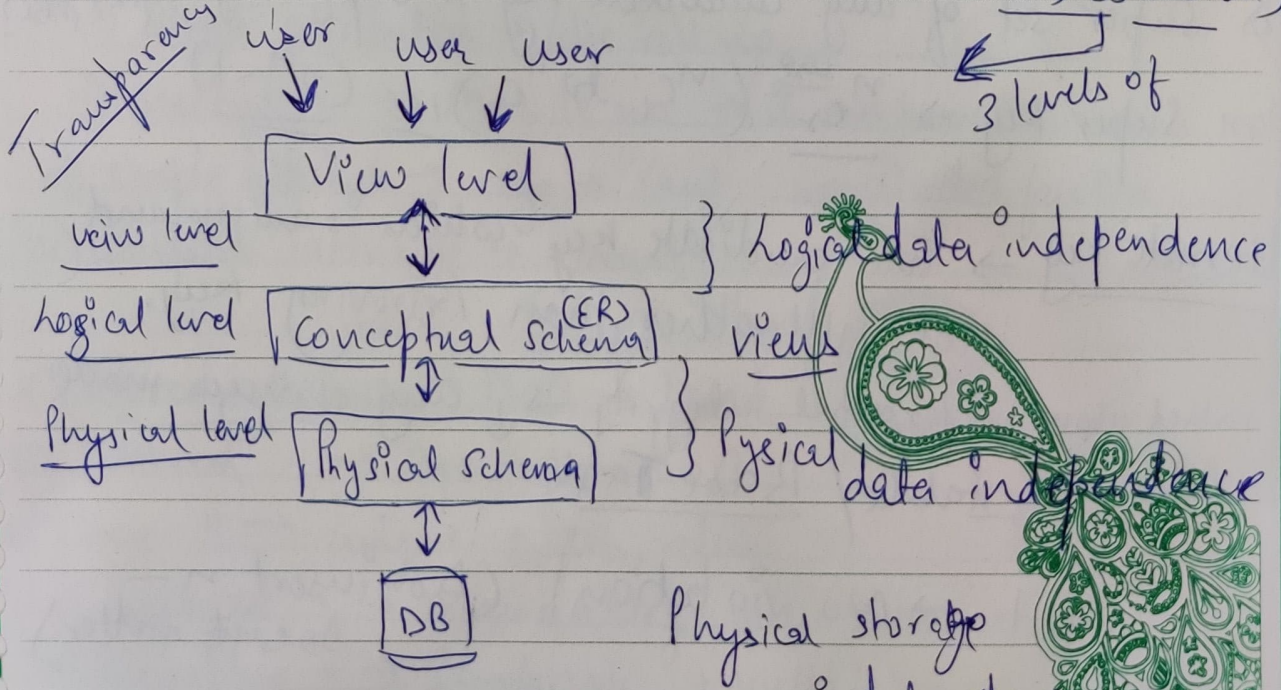

3layer → more users more scalable




Schema → Structure / Table

architecture → Data independence → user ≠ Data
(Data Abstraction)

3 levels of



Keys in DBMS → Attributes
↳use → uniquely identify

- Primary key → unique + not null
- Candidate key → ~~One~~ all who are eligible for being a primary key
  ↳ are unique but can be null
- Foreign key → Attribute or set of attributes that references to primary key of same or another table.
  ↳ It maintains Referential integrity.

⁂ Super Key :- combination of all possible attributes which uniquely identify two tuples in a table.
↳ like Roll no, name → (Primary, candidate)
together → super key

⁂ Super set of any candidate key is super key → {A}, {AB} {AC} {ABC}
Max Super key → $^{n}C_{n}$ to $^{n}C_{n}$ i.e. $(2^n - 1)$

Alternate key → The candidate key which is unique and not null other than Primary key.

I talked about referential integrity → ↓ any changes made in Reference table / Base Table

i) Insert → No violation (Do insert n — Doesn't matter)

Functional dependency ↓ Problems of Anamoly

2) Delete → May cause violation
   - → on delete Cascade
   - → on delete set null
   - → on delete no action

3) UPDATION → May cause violation

Referencing table → the one who will have foreign key
- Insert → May cause violation
- Delete → No violation
- Updation → May cause violation

## ER Model (Entity Relationship Model)

Entity → real world object (noun)  attributes → set of entity

### Types of attributes

i) Single attributed → like roll no.
ii) Multiple " → Mobile no → I can have 2 nos
iii) simple attribute → age → Cant be divided further
iv) composite attribute → Name
   - first
   - middle
   - last

v) Stored attribute → DOB → fixed
vi) Derived " → Age → from DOB.
vii) Key " → unique → SID, roll no
viii) Required " → Mandatory value → Name
ix) complex " → (Composite + Multiple)

# Degree of relationship / cardinality

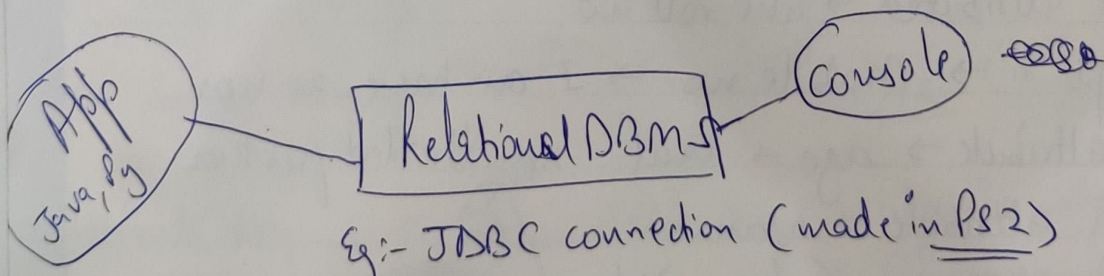
1. one-one (1:1) → Employee — works — Department
2. one-many (1:N) → Customer — order(s) — object (Laptop)
   ↳ he can order many
3. Many to many (M:N) → No deduced tables
   ↳ Students — enrolls — classes

## Types of Languages

SQL → standard by ISO

four sub languages

- → DDL → data definition language (create/drop/alter)
- → DQL → Data Query language (~~create~~ Select)
- → DML → Data manipulation (update/delete/Alter)
- → DCL → Data control (grant/revoke)




Eg:- JDBC connection (made in PS2)

SQL Server
MySQL
oracle

TCL → Transaction control → commit, rollback, savepoint

# Functional Dependency

- FD : $x \rightarrow y$ if $t_1.x = t_2.x$ then $t_1.y = t_2.y$
  typically exist b/w primary key & non-key attribute within a table.

Emp_Id → Emp_Name.

Types → Trivial & Non trivial → Z is not a ⊄ A.

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

Normalization → organizing data in DB

↓

- To min the redundancy

1 NF → atomic value → attribute can't hold multi values

- Kisi ke 2 phone hain → Ek hi row mai kra to its not in 1NF → Repeat the entries with diff phone → making it in 1NF

2NF → relation must be in 1NF

→ all non-key attributes functionally dependent on primary key.

Prime attributes :- attribute that are part of one of the candidate key

Non prime attribute :- not a part of any candidate key