* DBMS *

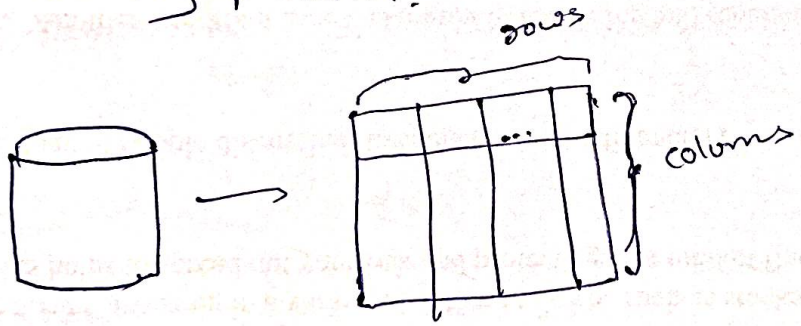
=> Database : Stores the user's data or collection of data

=> DBMS : Software / system to store and retrieve user's data

- It manipulate the database
- Accept the user request for data and instruct OS per data.

=> Why DBMS ? Difficulties of traditional file-processing systems supported by conventional OS.

1. Data redundancy & inconsistency
2. Difficult to access the data.
3. Data isolation - multiple file format
4. Integrity problem (inaccurate, inconsistent, unreliable)
5. Atomicity (Transaction as a single unit)
6. Concurrent access by multiple user
7. security problem.



Types of DBMS

OLTP → Transaction
OLAP → Analytical

=> Transcation : Fundamental unit of work that perform logical set of operation on the Database.

- Transcation follow ACID properties

1. Atomicity
2. Consistency
2. Isolation
4. Durability

- Solves the problem of
  1. Data integrity
  2. Concurrency control
  3. Recovery.

⇒ ER Diagram : Entity Relationship Diagram is visual represent. of data and its relationship in database. helps design logical structure of Database

- Component of ER Diagram

1. Entities : Real world entities, can be distinguished from other.

   Eg In Hospital system,

   Doctor, Nurse, Assistent.

   -  representation

2. Attributes : Describes an entity

   Eg Doctor

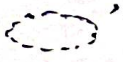
   Name, ID, Email, Address.

   -  connected to their entity

3. Relationship : Association between entities

   →

- Types of Attributes

Single	Name, ID
composite	Name -> (First name, Last name)
Derived	Age (From DOB) "  "
multivalued	ph. number "  "

- Types of Entities

Strong Entity	Book (has a key : ISBN)
Weak Entity	Library card (no key)

- Cardinality & Participation constraints : maximum no. of relationship instance in which an entity can participate

[set A] [set B]

1. One to one (1:1)
   member ↔ librarycard

2. one to many (1:N) (library has many books)
   library ↔ Books

3. many to many (Book can have multiple author)
   Book ↔ Author

- Participation

1. Total participation : Every instance must participate
2. Partial participation : Not all entities need to particip

=> Closure of Attribute set : set of all attributes that can be functionally determined from an attribute set

Eg R(A, B, C, D) and FDs. Find A^*

$A \rightarrow B$
$B \rightarrow C$
$A \rightarrow D$

=> $A^* = \{A\}$

- $A \rightarrow B$
  A is in A^* → add B in A^* → $A^+ = \{A, B\}$

- $B \rightarrow C$
  B is in A^* → add C in A^+ → $A^+ = \{A, B, C\}$

- $A \rightarrow D$
  A is in A^* → add D in A^* => $A^+ = \{A, B, C, D\}$

∴ $A^* = \{A, B, C, D\}$ contains all the attributes of A

Hence, A^+ is a superkey

- Eg R(P, Q, R, S) and FDs Find P^+

$P \rightarrow Q$

$Q \rightarrow R$

Ans:) $P^+ = \{P\}$

- $P \rightarrow Q$

  $P^+ = \{P, Q\}$

- $Q \rightarrow R$

  $P^+ = \{P, Q, R\}$

$\therefore P^+ = \{P, Q, R\}$ Not a superkey

=> Calculate the candidate key

Step 1: Identify attributes

- Attributes apper only on RHS (non-prime attribute) cannot be part of candidate key
- Attribute ~~apper~~ never appear of RHS (essential attri) mush have in candidate key

Step 2: List All possible attributes set

step 3: Test each set for superkey

step 4: Test for minimality

- If the attribute set is superkey, If any subset is also superkey

  If yes it's not minimal

  If NO it's a candidate key

Eg R(A, B, C, D)

A → B
B → C
A → D

=> 1. RHS = {B, C, D}
LHS = {A, B}

A is essential

2. list all possible keys {A} {~~A~~} {A, B} {A, C} {A, D}
{A, B, C} {A, B, D}

3. Test {A}

$A^+ = \{A\}$

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

A^+ is full relation, so {A} is a superkey

4) {A} is by default minimal

5) {A} is candidate key

=> Decomposition of a Relation / Table

- Breaking or dividing a single relation in two or more sub relations.

- Types of Decomposition

1. lossless : No info. or data is lost.
   - when join same as original
   - $R_1 \cap R_2$ must be key in R_1 or R_2
   - Data integrity preserved.

2. lossy join :
   - may add / remove rows (spurious data)
   - $R_1 \cap R_2$ is not key
   - Data integrity is lost

=> Normalization:

Process of making Database consistent by
- Reducing the duplications
- Data integrity of data through lossless decomposition
- Avoid anomalies (update, insert, delete)

Why we need Normalization?

Eg

Roll no.	Name	Course	Course Fee
1	Alex	math	500
2	Josh	math	500
3	Alex	physics	600

=> Problems:

1. Reduendancy:
   course fee for math is repeated
2. Update anamoly:
   If we update the fees for math, we need to change in every row with math course
3. Delete anamoly:
   Deleting an entry will also delete the course info.
4. Insert anamoly:
   New course info can't be added if no student is enrolled

=> Normalized forms.

1. First normal form (1NF)
2. second normal form (2NF)
3. Third normal form (3NF)
4. Boyce-codd normal form (BCNF)

1. 1NF.
   - Atomicity / single value in each column.
   - No duplicates

2. 2NF
   - It must be 1NF
   - No partial dependency

     A → B is called partial if A is subset of a composite key.

3. 3NF
   - It must be 2NF
   - No transitive dependency

     non prime → non prime (transitive FD)

4. BCNF
   - It must be 3NF
   - In all FD's LHS must be super key