__/__/____

Database Management System

Data doesn't having meaning unless processed.

1. what is Data ?

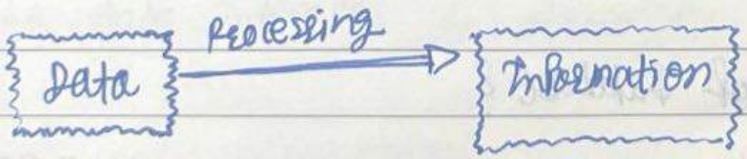
→ Data is a collection of raw, unorganized facts & details like text, observation, figures, symbols and description of things etc.

In other word, data does not carry any specific purpose and has no significance by itself.

Data is measured in terms of bits and bytes which are the basic units of information in the context of computer storage & processing.

2. Type of Data
   a. Quantitative → Numerical
   b. Qualitative → Descriptive but not numerical

3. what is Information ?
   a. Information is processed, organized & structured data.
   b. It provide context of data to enables decision making.
   c. Processed data that make sense us.



Information depends on data, data doesn't depends on Information.

e.g. you have the data peep from your locality some info. is/are there are 100 Males, females, etc.

__/__/____

4. Data vs Information

→ a. Data is collection of facts, while Information puts those facts into context.

b. while data is raw, unorganized, information is organized.

c. Data points are individual and sometimes unrelated, Information maps out that data to provide a big picture view that how data are fits together.

d. Data comes in the form of graph, numbers, etc while Information is typically presented through words.

e. Data alone is not efficient to decision-making, make decision based on Information.

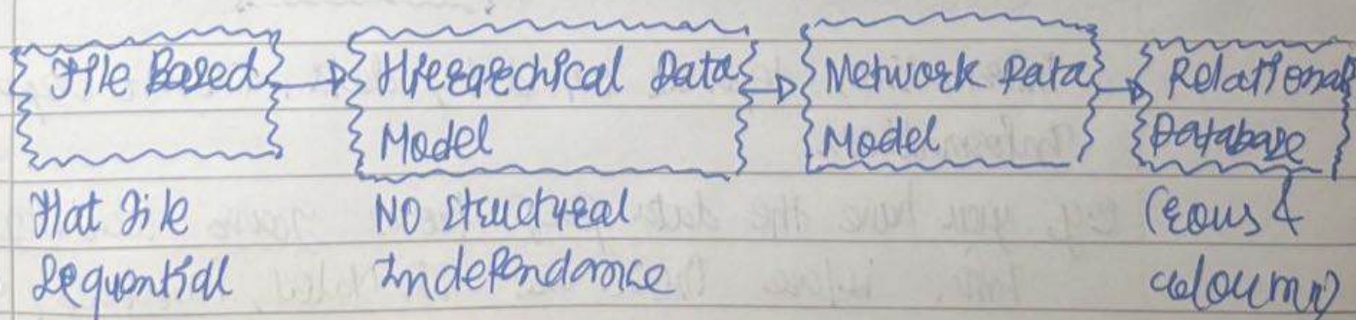
5. what is Database?

→ a. Database is an electronic place/system where data is stored in a way that it can be easily accessed, managed & updated.

b. Main Purpose of database to operate large amount data

6. Evolution of Database?

→

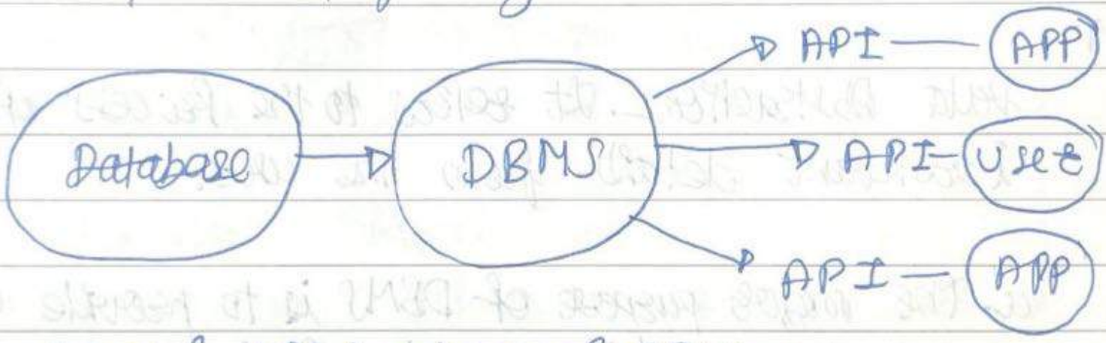


7. What is DBMS?

→ a. Database Management System (DBMS) is a collection of interrelated data and a set of program to access those data.

b. Primary goal of a DBMS is to provide a way to store & retrieve database information that is both convenient & efficient.

c. A DBMS is the database itself, along with all the software & functionality.
It is used to perform different operations like addition, access, updating and deletion of the data.



8. Advantages & Disadvantages of DBMS?

→ Advantage

a. Uniform administration procedure of Data.
b. Control Redundancy as all the data store in single database file.
c. Data Sharing, an authorized user can share data.
d. It Provides Backup & Recovery System.
e. Offers data integrity & Security.

Disadvantage

a. Cost is High, coz require High speed processor & memory.
b. DBMS can't perform sophisticated Calculation.
c. Using of the same program at a time by multiple user can lead to data-Loss.

9. DBMS vs File System

→ a. File Processing System has Major dis-advantage

i. Data Redundancy & Inconsistency. [NO Duplicate data]

ii. Difficulty in accessing data. [Accessing Problem]

iii. Data Isolation. [Data store in diff File Format, retrieve Prob]

iv. Integrity ~~Prob~~ Problem. [check condition, Mini. 1000₹ in ac]

v. Atomicity Problem

vi. Concurrent access anomalies.

vii. Security Problem.

Basics	File System	DBMS
Structure	The file system is a way of arranging the files in a storage medium within a computer.	DBMS is software for managing the database.
Data Redundancy	Redundant data can be present in a file system.	In DBMS there is no redundant data.
Backup and Recovery	It doesn't provide Inbuilt mechanism for backup and recovery of data if it is lost.	It provides in house tools for backup and recovery of data even if it is lost.
Query processing	There is no efficient query processing in the file system.	Efficient query processing is there in DBMS.
Consistency	There is less data consistency in the file system.	There is more data consistency because of the process of normalization.
Complexity	It is less complex as compared to DBMS.	It has more complexity in handling as compared to the file system.
Security Constraints	File systems provide less security in comparison to DBMS.	DBMS has more security mechanisms as compared to file systems.
Cost	It is less expensive than DBMS.	It has a comparatively higher cost than a file system.
Data Independence	There is no data independence.	In DBMS data independence exists, mainly of two types: 1) Logical Data Independence. 2)Physical Data Independence.
User Access	Only one user can access data at a time.	Multiple users can access data at a time.
Meaning	The users are not required to write procedures.	The user has to write procedures for managing databases
Sharing	Data is distributed in many files. So, it is not easy to share data.	Due to centralized nature data sharing is easy
Data Abstraction	It give details of storage and representation of data	It hides the internal details of Database
Integrity Constraints	Integrity Constraints are difficult to implement	Integrity constraints are easy to implement
Attributes	To access data in a file , user requires attributes such as file name, file location.	No such attributes are required.
Example	Cobol, C++	Oracle, SQL Server

10. View of Data (3-Schema Architecture)

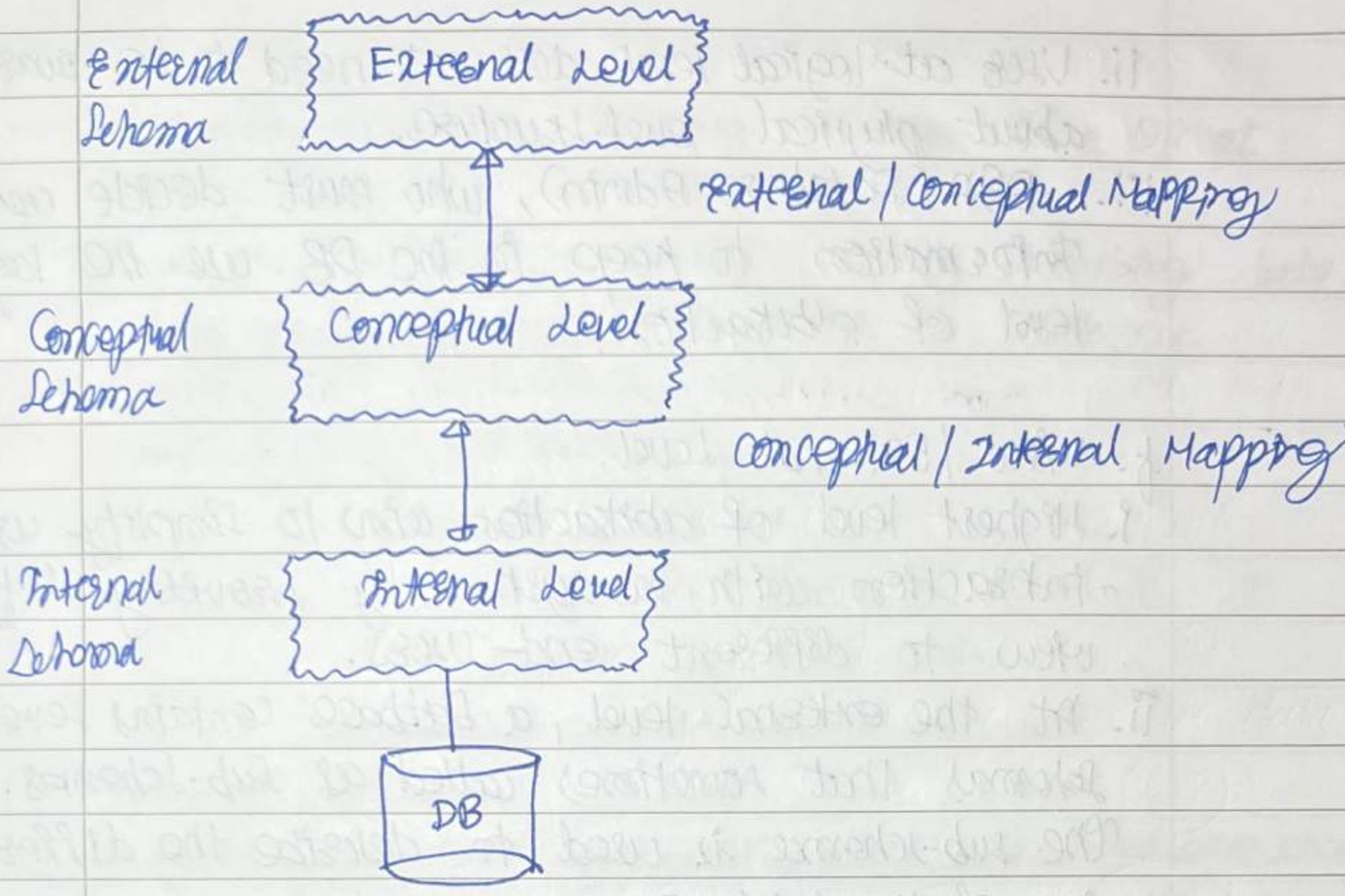
→

Data Abstraction - It refers to the process of hiding irrelevant details from the user.

a. The major purpose of DBMS is to provide users with an abstract view of the data. That is system hides certain details of how the data is stored & ~~maintained~~ maintains

b. The interaction with the System make simple, abstraction is applied through several levels of abstraction.

c. The main objective of 3 level architecture is to enable multiple users to access the same data with a personalized view while storing underlying data only once.



d. Physical Level / Internal Level

i. The lowest level of abstraction describes how the data are stored.

ii. It has physical schema which describe physical storage structure of DB.

iii. Generally ~~talks~~ talk about storage allocation, encryption, de---.

iv. Godl, we must define Data Structure for algorithm that allow efficient access of data.

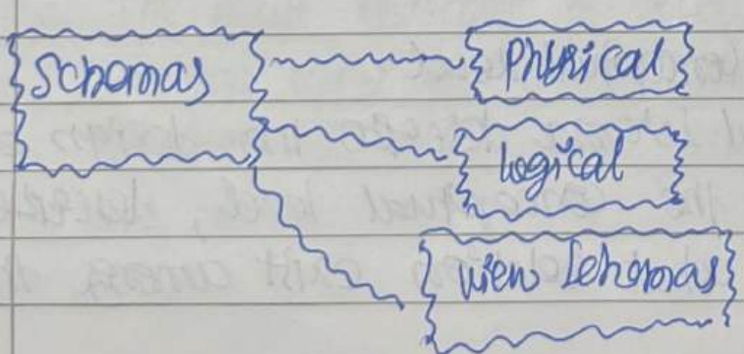
e. Logical Level / Conceptual Level

i. The Conceptual Schema describe the design of a database at the conceptual level, describe what data stored, what relation exist among that data.

ii. Users at logical level does not need to be aware about physical level structure.
iii. DBA (Database Admin), who must decide what information to keep in the DB use the logical level of abstraction.

f. View / External Level
i. Highest level of abstraction aims to simplify users interaction with the system by providing different view to different end-users.
ii. At the external level, a Database contains several schemas that sometimes called as sub-schemas. The sub-schema is used to describe the different view of the database.
iii. At views also provides a security mechanism to prevent users from accessing unauthorized part of DB.

II. Instances and Schemas
a. The collection of information stored in DB at a particular moment is called an instance of DB.
b. The overall design of DB is called DB schema.
c. Schema is a structural representation of data.



12. Data Models

a. Provide a way to describe the design of a DB at logical level.

b. A collection of conceptual tools for describing data, data relationship, data semantics & consistency constraints.

c. Example → ER model,

Relational model,
Object-oriented model
Object-Relation model

13. DataBase Languages

a. Data Definition language to specify the Database schema.

b. Data Manipulation Language (DML) to express Database Queries and Updates.

c. Both language features are present in single DB language, e.g. SQL Language.

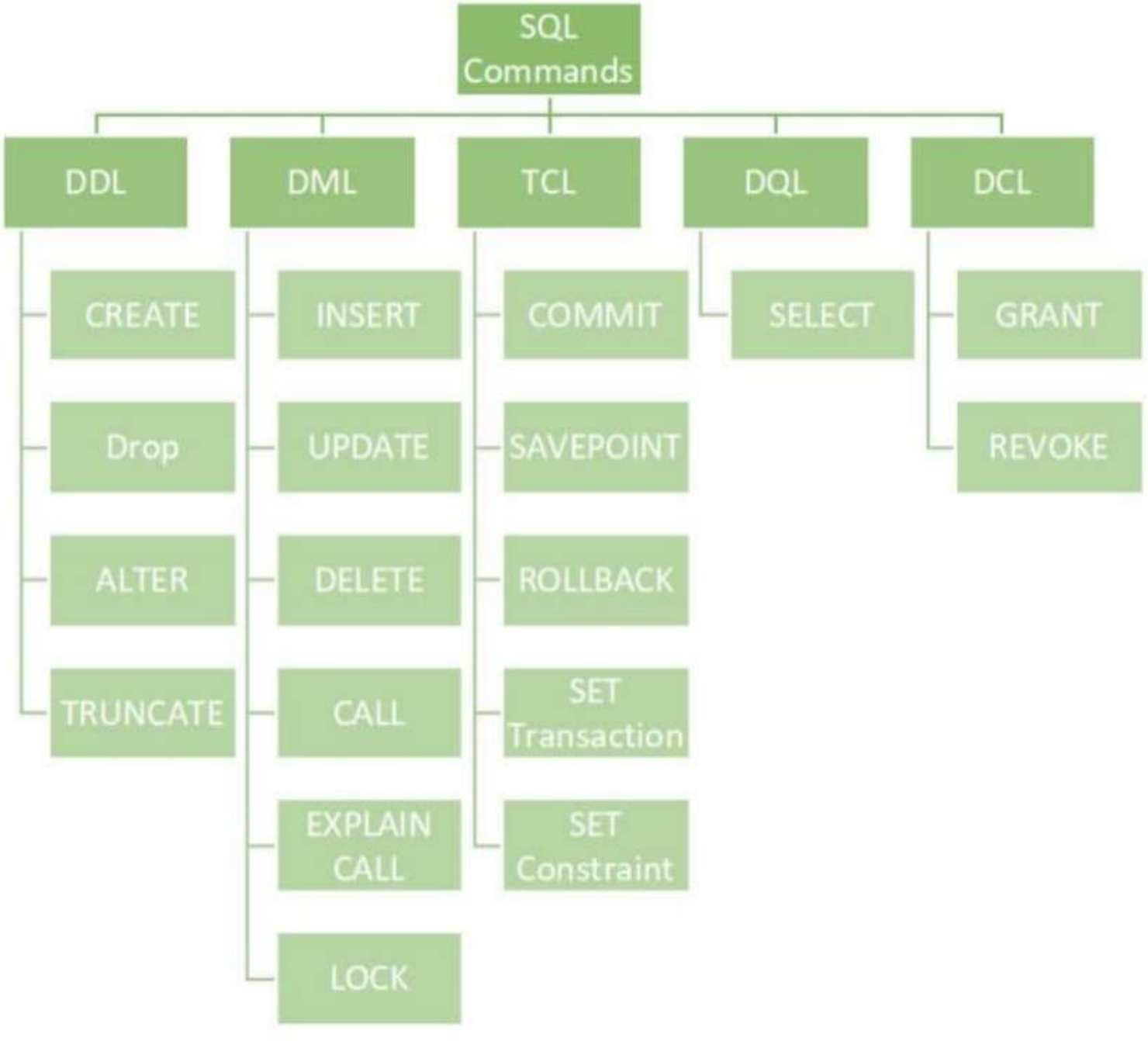
DDL → Specify consistency constraint, which must be checked every time DB is updated.

DML → Data Manipulation involve, Retrieval, Insertion, Deletion, & Updation.

14. DCL → Data Control Language, used to deal with rights, Permission, etc.

DQL → Data Query Language, used to perform queries on data.

TCL → Transaction control Language, used to control execution of Transaction.


SQL Commands
DDL
DML
TCL
DQL
DCL
CREATE
Drop
ALTER
TRUNCATE
INSERT
UPDATE
DELETE
CALL
EXPLAIN CALL
LOCK
COMMIT
SAVEPOINT
ROLLBACK
SET Transaction
SET Constraint
SELECT
GRANT
REVOKE

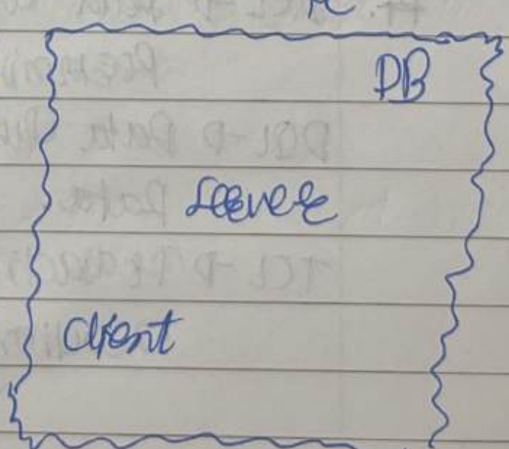
__/__/____

14. How Database is accessed from application Program?
a. Apps written in host languages, C, C++, Java interacts with DB.
b. API is provided to send DML/DDL statements to DB to retrieve information.
i. Open Database Connectivity (ODBC) Microsoft, "C".
ii. Java Database Connectivity (JDBC), Java.

15. Database Administrator (DBA)
a. A Person who has central control of both the data and the Programs that access those data.
b. Functions of DBA
- Schema Definition
- Storage Structure & access Method methods
- Schema & physical organization Modification.
- Authorization Control
- Routine Maintenance. {Any upgradation needed or not}

16. DBMS Application Architecture
Client Machine on which remote DB users work
Server Machine on which DB system runs.

a. TI Architecture
The Client, Server & DB all present on same Machine



b. T2 Architecture

i. Client Machine, which invokes DB system functionality at server end through query language statement.

ii. API standards like ODBC & JDBC are used to interact between client & server.

c. T3 Architecture

i. APP is partitioned into 3 logical components.

ii. Client Machine is just a frontend and doesn't contain any direct DB calls.

iii. Client machine communicates with app server, & app server communicated with DB system.

iv. T3 Architecture are best for WWW Applications.

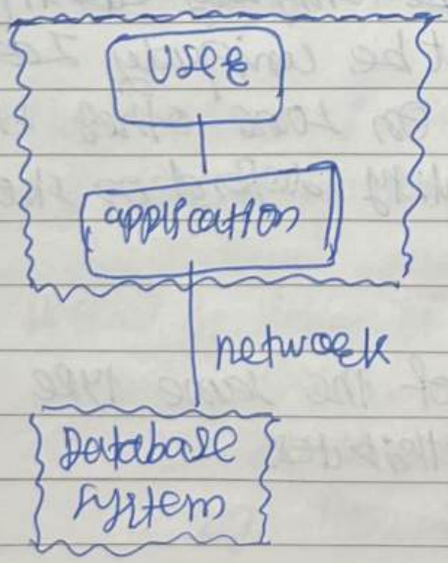
v. Advantage

① Scalability, due to distributed app. server.

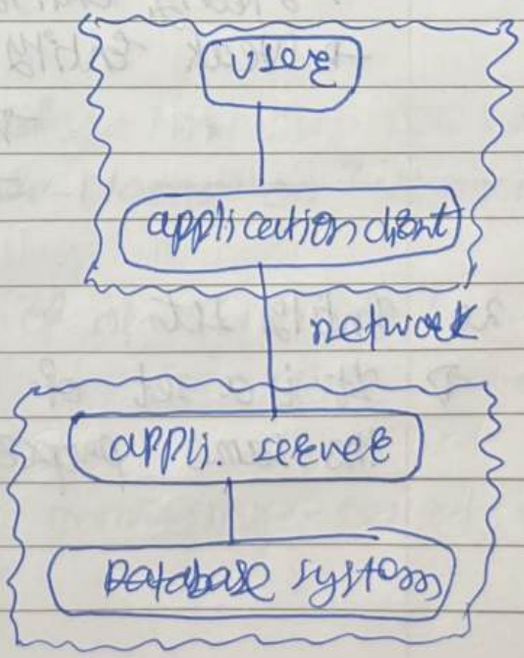
② Data Integrity, APP server acts as middle man between client & DB, which minimize chance of data corruption.

③ Security.

2-Tier Archi.



3-Tier Architecture



17. Data Model

→ Collection of conceptual tools for describing data, data relationships, data semantics, & consistency constraints.

18. ER Model

a. It is a high level data model based on a perception of a real world that consists of a collection of basic objects, called entities & of relationships among these objects.

b. Graphical representation of ER Model is ER diagram, which acts as a blueprint of DB.

19. Entity

→ An entity is a thing or object in the real world that is distinguishable from all other objects.

→ It has physical existence

→ Each student in a college is an entity.

→ Entity can be uniquely identified. [Primary key],

→ Strong Entity → can be uniquely identified.

→ Weak Entity → can not be uniquely Identified,

→ Depends on some other strong entity.

→ weak Entity depend on strong entity.

20. Entity Set

→ It is a set of entities of the same type that share the same properties, or attributes.

21. Attribute

→ An entity is represented by a set of attribute Each entity has a value for each of its attributes.

Student entity has following attributes,

- Student_Id
- Roll NO
- Address, so on.

Types of Attributes

1. Simple :- Attributes which can't be divided further
   e.g Customers account Number.

2. Composite :- Can be divided into subparts.
   e.g Name of a person, First Name, Second & Last Name.
   Address can be also divided.

3. Single-valued :- Only one value attributes
   e.g Student_ID, loan_Number.

4. Multi-valued :- Attribute having More than one value.
   e.g phone Number, Nominee-Name on Insurance, etc.

5 Derived :- Value of this type of attributes can be derived from the value of other related attributes
   e.g. Age, loan-age, membership-period, etc.

6. NULL : An attribute takes a NULL value when an entity does not have a value for it.

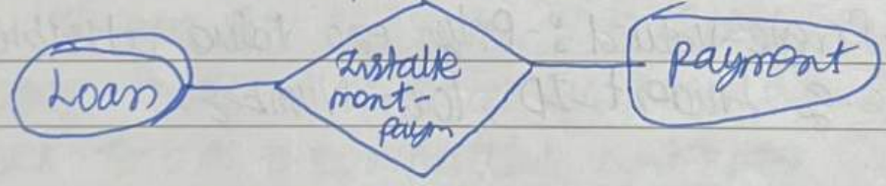
- It may indicate Not Applicable.
  e.g Person having no Middle Name.
- It May Indicate Unknown
  - Missing Entry
  - Not known (Salary Attribute) yet

22. Relationships

→ a. Association among 2 or More entities.
e.g Person has vehicle,
Parent has Child,
Customer borrow loan.

b. Strong Relationship, Between two Independent entity.

c. Weak Relationship, Between weak & Strong entity.



d. Degree of Relationship
Number of entities participating in a relationship.
Unary, only one entity participates, e.g.
Employee manages employees.

Binary, two entities Participates.
e.g Student takes Course.

Ternary, three entities participates,
eg Employee work-on branch, employee works-on job.

23. Relationship Constraints

1. Mapping Cardinality
Number of entities to which another entity can be associated via a relationship.

2. One to one, entity in A associates with at most one entity in B, where A & B are entity sets. And an entity of B is associated with at most one entity of A.
e.g Citizen has Aadhar Card.

3. One to Many, Entity in A associates with N entity in B. while entity in B is associates with N entity in A.
e.g citizen has Vehicle.

4. Many to one, Entity in A associates with at most one entity in B. while entity in B can be associated with N entity in A.
e.g Courses taken by Professors.

5. Many to Many.
eg → Customers buys product.

6. Participation Constraints / Minimum Cardinality Constraint
Partial & Total Participation.

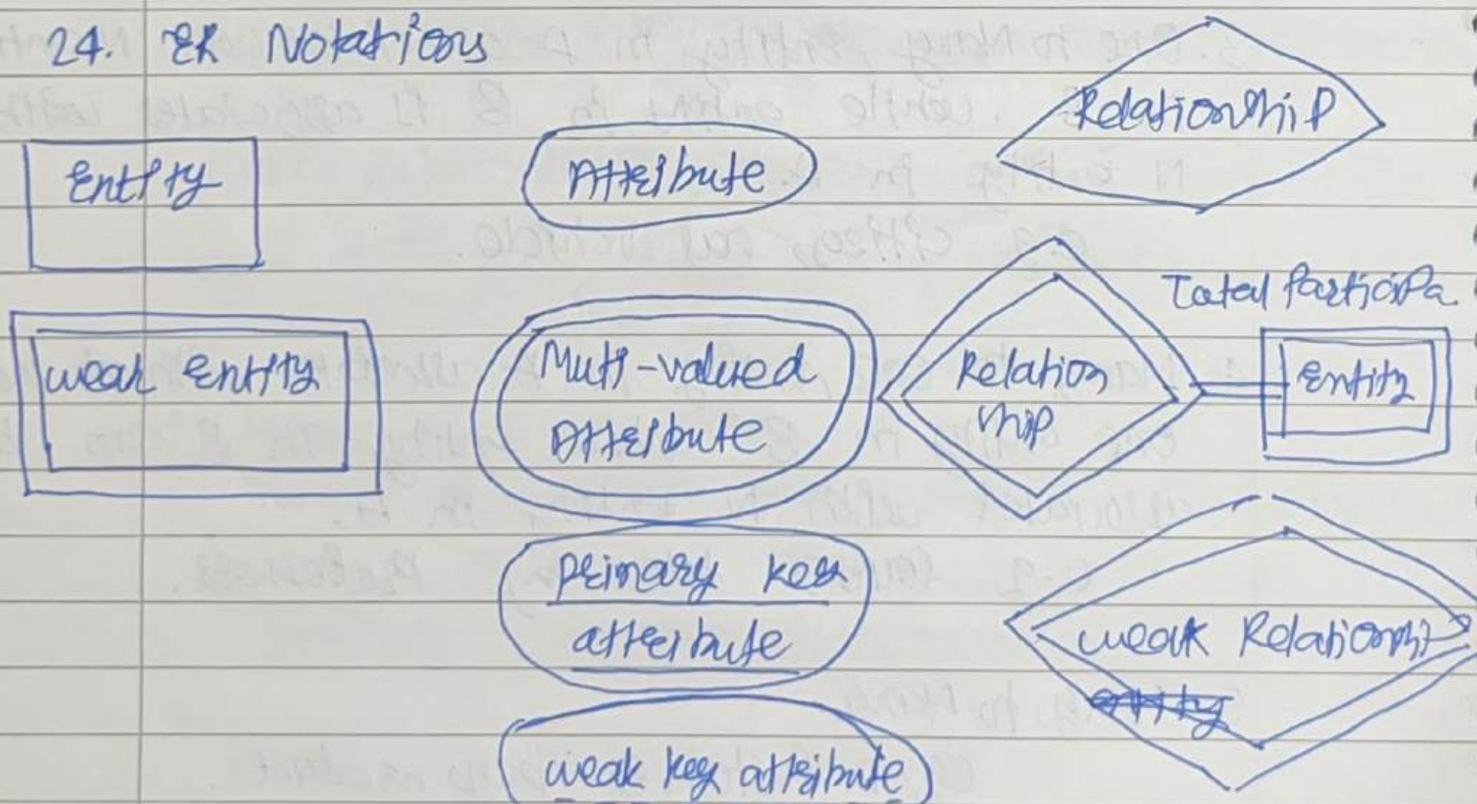
a. Partial Participation, not all entities are involved in the relationship instance

b. Total Participation, each entity must be involved in at least one relationship instance.

e.g.
→ Customer borrow loan, loan has total Participati as it can't exist without customer entity and customer has partial participation.

c. weak entity has total participation constraint, but strong may not have total.

24. ER Notations



25. Extended-ER Features

a. Basic ER features studied, can be used to model most DB features but when Complexity increases, it is better to use some extended ER features to model the DB schema.

b. Specialisation

① In ER model, we may require to subgroup an entity set into other entity sets that are distinct in some way with other entity sets.

② Specialization is splitting up the entity set into further sub-entity sets on the basis of their functionalities, specialities & features.

③ It is a TOP-Down approach.

④ why specialisation?

i. Certain attributes may only be applicable to a few entities of the Parent entity set.

ii. DB designer can show the distinctive features of sub-entities.

iii. To group such entities we apply specialization, to overall refine the DB blueprint.

c. Generalisation

① It is just Reverse of specialization.

② It is Bottom-UP approach.

③ DB Designer may encounter certain Properties of 2 entities are overlapping, Designer may consider to make a new generalised entity set.

④ why Generalization?
i. Makes DB more refined and simpler.
ii. Common attributes are not Repeated.

d. Attribute Inheritance
① Both Specialization & Generalization has attribute Inheritance.

② The attribute ~~case~~ of Higher entity sets are inherited by lower ones.

e. Participation Inheritance
If a Parent entity set participates in a relationship then its child entity sets will also Participate in that relationship.

f. Aggregation

① How to show relationships among relationship? Aggregation is the technique.

② Abstraction is applied to treat relationship as Higher level entities

③ Avoid Redudancy by aggregation relationship

26. Step to Make ER Diagram

① Identify Entity Sets
② Identify Attributes & their types.
③ Identify Relationship & Constraints
(Mapping, Participation)

ER Model of Banking System

Requirements :-
① Branches.
② Customers.
③ Custom have account & they take loans.
④ Customer associated with some banker (loan Mangee, ok)
⑤ Bank has employees.
⑥ Accounts saving, current.
⑦ Loan designated by branch
loan ≥ 1 customers
loan Payment schedules.

Step 1: Identifying Entity Sets
① Branch
② Customer
③ Employee
④ Current / Saving account
⑤ Loan
⑥ Payment (Loan) (weak entity)

__/__/____

Step 2: Identifying the attributes & their types.

① Branch :- name, city, assests, liabilities
② Customer :- cust_id, name, address, contact no, DOB, age
③ Employee :- emp_id, name, contact-no, dependent name, year_of_service, start-date.
④ Saving account :- acc_number, balance, Intsert-Rate, daily withdrawal limit,
⑤ Current account :- acc_number, for-traaction charges, overdraft_amount.
⑥ Generalized Entity :- Account ▷ acc_no, balance
⑦ loan :- loan_number, amount
⑧ weak Entity Payment :- Payment_no, date, amount.

Step 3: Relationship

① Customer borrows loan.
M : N

② loan originated by branch
N : 1

③ Loan — loan-Payment — Payment.
1 : N

④ Customer deposit account
M : N

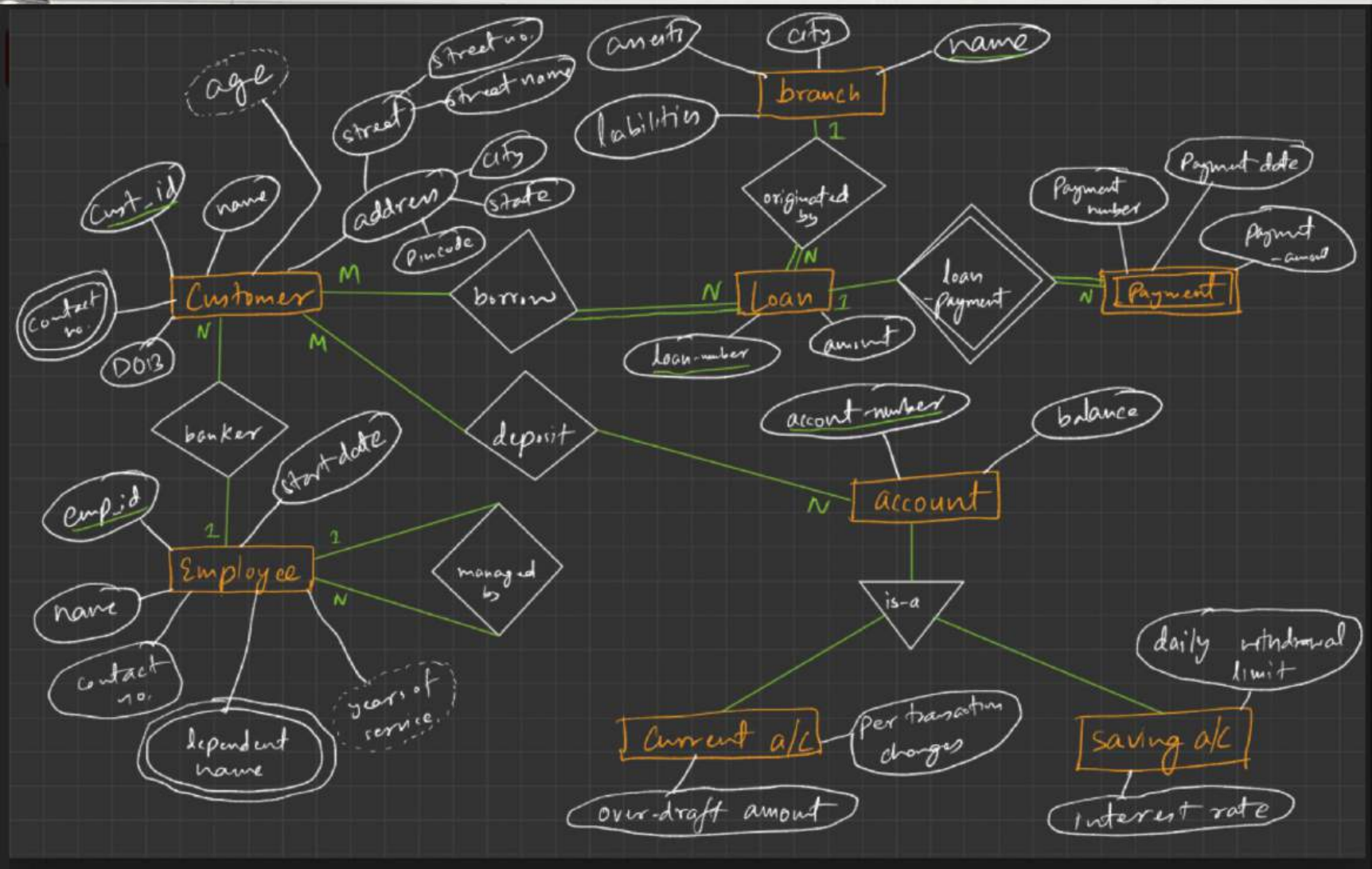
⑤ Customer banker Employee

N : 1.

⑥ Employee Manage by Employee

N : 1

ER Diagram :-



27. FaceBook (FB) ER Diagram.

Requirements :-

① People → user_profile → friends
② User can Post
③ Post → contains → text content, images, videos.
④ Post → like, Comment.

Step1 : Identify Entity sets.

① User People
② User Post
③ Post Comment
④ Post like

Step2 : Attributes

① User_People :- Name, UserName, Email, Contact, DOB, age.
② User_Post :- Post_id, text_content, image, video, created_timestamp, modified_time stamp.

③ Post_Comment :- Post_Comment_id, text content, timestamp.

④ Post_like :- Post_like_id, timestamp.

Step3 :- Relationship

① User_Profile — N — friendship — : — N — user_profile.

② User_profile — 1 — Posts — : ——— N — User_post.

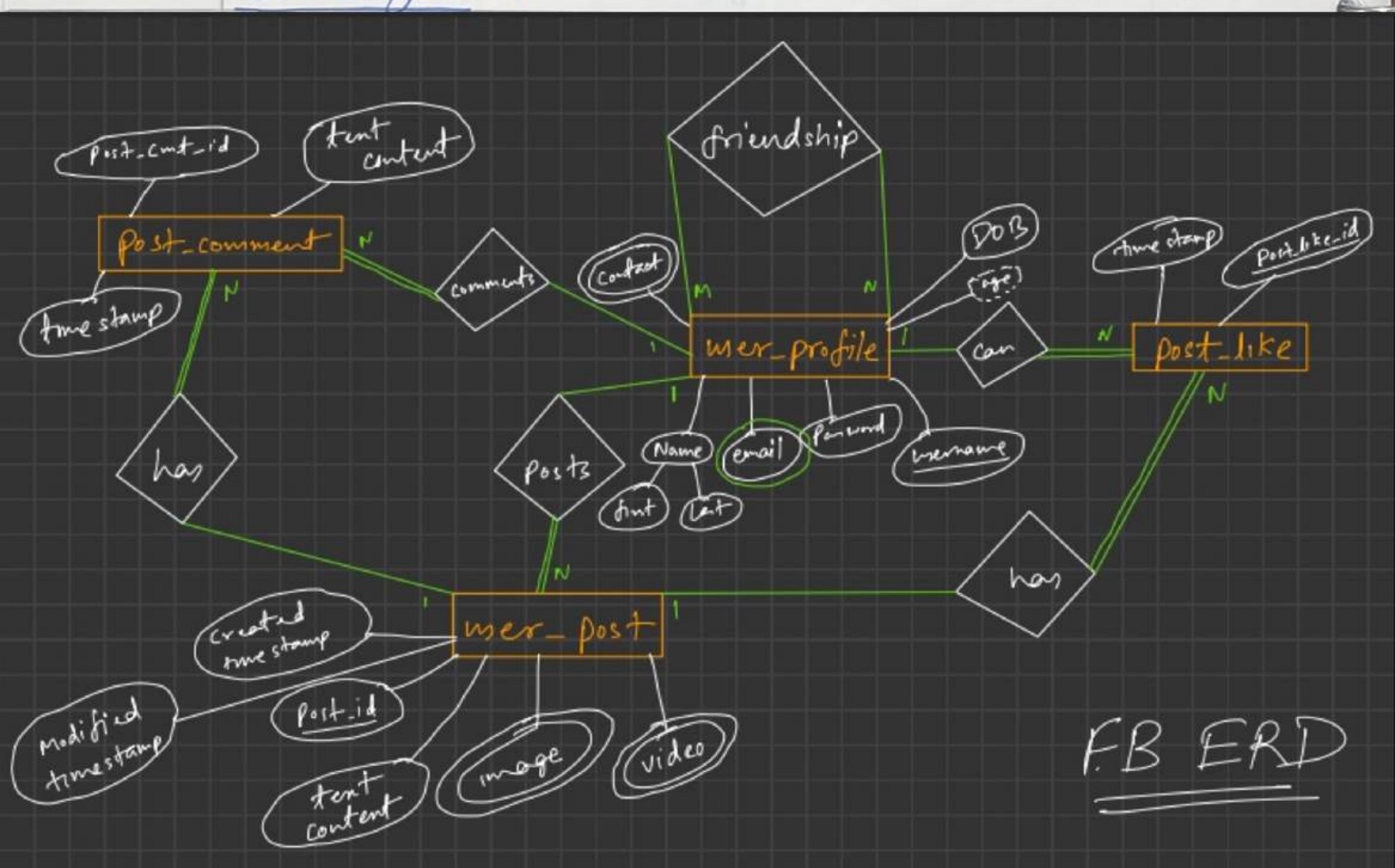
③ User profile — 1 — can — : ——— N — post_like

④ User Profile — 1 — comments — : —— N — Post_comments

⑤ User Post — 1 — has — : ——— N — post_comment.

⑥ User_post — 1 — has — : —— N — post_like.

ER diagram :-



28. Relational Model

a. Relational Model organize the data in the form of relations (tables).

b. A relational DB consist of collection of tables, each of which is assigned a unique name.

c. A row in a table represents a relationship among a set of values, & table is collection of such relationship.

d. Tuple :- A single row of table represent a single data point / a unique record.

e. Columns :- Represent attribute of relationship

f. Relation Schema :- Defines the design & structure of relation.

Common RDBMS are Oracle, IBM, MySQL, MS Access.

g. Degree of Table :- Number of attributes in given Table.

h. Cardinality :- Total no. of Tuple in Relation.

i. Relational key :- Set of attribute which can uniquely identify each tuple.

Properties of Table in Relational Model

① The name of Relation distinct among all other.

② The values have to be atomic, can't broken further.

③ The name of each attribute/ column must be unique.

④ Each tuple Must be unique in a table.

⑤ The sequence of Row & column has no significance.

⑥ Table Must follow Integrity Constraints

It helps to maintain data integrity / consistency.

Relational Model Keys

① Super key :- (SK) Any P.C of attribute present in a table which can uniquely identify each tuple.

② Candidate Key (CK) :- Minimum subset of super keys, which can uniquely identify each tuple.
CK value shouldn't be NULL & contain no redundant value.

③ Primary key (PK) :- Selected out of CK set, has the least no. of attributes.

④ Alternate key (AK) :- All CK except PK.

⑤ Foreign key (FK) :- It create relation between 2 tables. A relation (R1) may include among its attribute the PK of an other relation (R2). This attribute called FK from (R1) refering (R2).

⑥ Composite key :- PK formed using at least 2 attribute.

⑦ Compound key :- PK which is formed using 2 FK

⑧ Surrogate key :- Synthetic key, May be used as PK, Generated automatically by DB.

Integrity Constraints

① CRUD operation must be done by some integrity policy.

② Introduced, so DB doesn't corrupt & consistent.

③ Domain Constraint
   a. Restrict the value in attribute of Relation.
   b. Restrict the Data Types of every attribute.

④ Entity Constraint
   every Relation must have PK, PK != NULL.

Referential Constraint

① Specified Between 2 Relation & helps maintain Consistency among tuples of two Relations.

② It requires that the value appearing in specified attribute of any tuple in referring relation is also appear in the specified attribute of at least one tuple in the referenced relation.

③ If FK is referring referencing table refers to PK of referenced table then every value of the FK in referencing table must be NULL.

ON Delete CASCADE

④ FK must have the matching PK of its each value in the Parent Table.

Key Constraint:-

① NOT NULL : This Constraint will restrict the user from not having a NULL value.

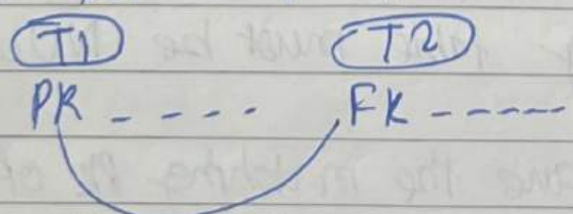
② UNIQUE : It helps us to ensure that all the values consisting in a column are different from each other.

③ DEFAULT : It is used to set the default value.

④ CHECK : It is one of the Integrity constraints in DBMS.
It keeps the check that Integrity of data is maintained before & after the completion of CRUD.

⑤ PRIMARY KEY : This is an attribute or set of attribute used 'identify the each entity in the entity set.

⑥ FOREIGN KEY : Whenever there is some relation Between 2 entities there must be some common attribute Between them.



This key will prevent every action which can result in data loss.

29. Transform ER model to Relation Model

① Both ER Model & Relational Model are abstract logical representation of real-world entities.

② ER diagram notations to relations:

a. Strong Entity

Become individual table with entity name, attributes becomes column of the relation.

Entity PK used as Relational (PK).

FK will added make Relationship Between 2.

b. Weak Entity

A table is formed with all the attribute of the entity.

PK of its corresponding strong entity will be added as FK.

c. Single Value attribute

Represent as Column directly in the tables.

d. Composite Attribute

① Handled by creating a seperate attribute itself in the original relation for each.

e. Multivalue Attribute

New tables are created for each multivalue attri. PK of entity is used as column FK, in the new table.

__/__/____

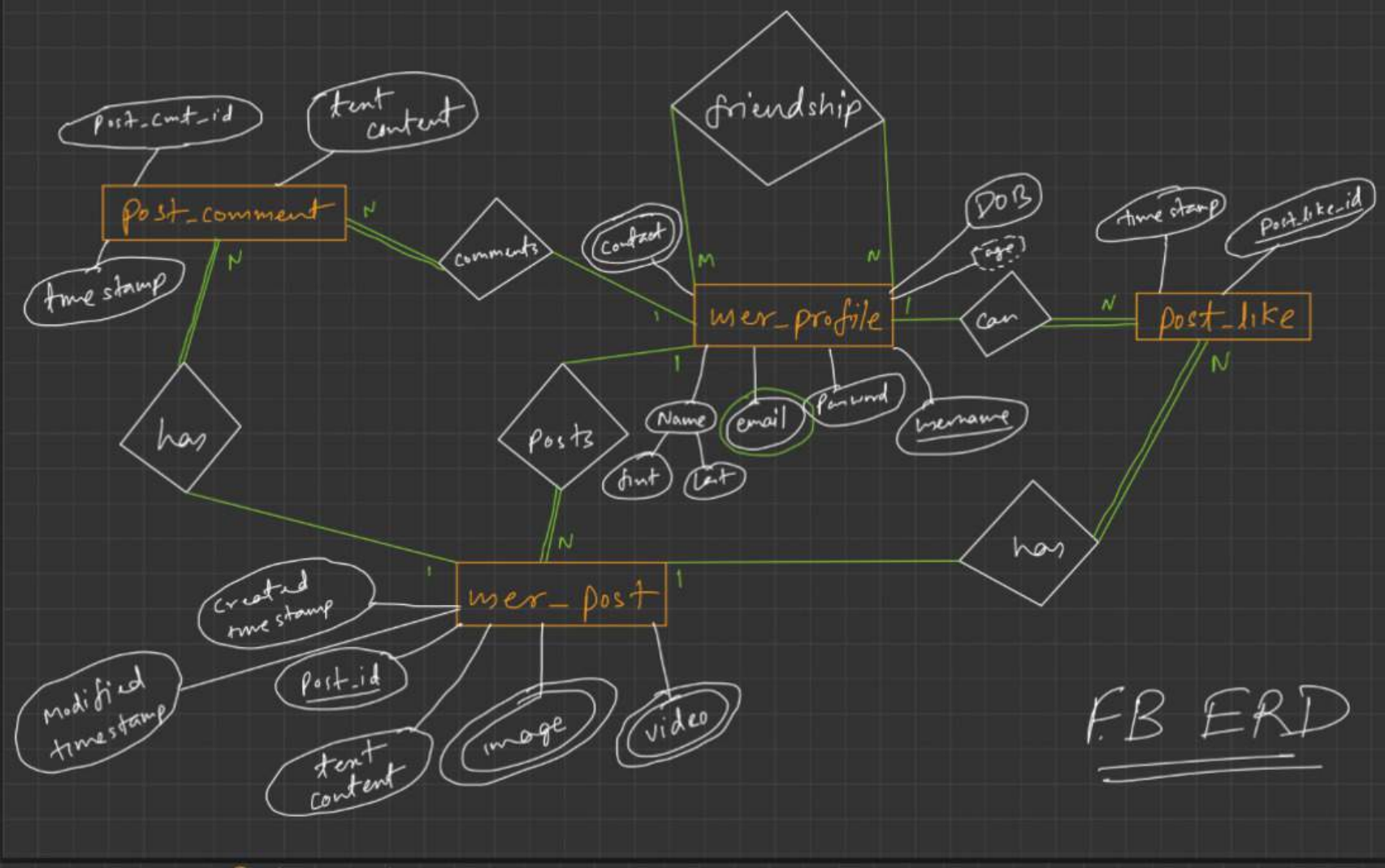
f. Derived Attribute
Not considered in table.

2. Generalisation

Method-I Create a table for the higher level entity set. For each lower entity set, create a table that include a column for each of the attribute of that entity set plus a column for each attribute of the primary key of the higher-level entity set.
For e.g Banking System Generalisation,
① Table 1: account (account-number, balance)
② Table 2: saving account (account number, interest-rate, daily-limit)
③ Table 3: current account (account number, overdraft-amount, per-transaction-charges)

Method-II An alternate representation is possible. Do not create a table for the higher-level entity set. Instead, for each lower-level entity set, create a table that includes a column for each of the attribute of that entity set plus a column for each attribute of higher-level entity sets.
① Table 1: saving-account (acc_number, balance, interest-rate, daily-withdrawal-limit).
② Table 2: (current_account (accomt-number, balance, overdraft-amount, per-transaction charges).

Drawbacks: If the second method were used for an overlapping generali. some values such as balance would be stored twice unnecessarily. If the generalization weren't complete if some account were neither nor current account these account could not be represented with second method.



* FB Relational Model

(1) user_profile (username, name-first, name-last, password, DOB)

(2) user-profile-email (username {F.K}, email)

(3) user_profile-contact (username {f.k}, contact-number)

(4) friendship (profile_req {f.k}, profile_accept {f.k}) ⟶ Compound key

(5) post-like (post_like_id, timestamp, post_id {f.k}, username {f.k})

(6) user-post (post-id, created-timestamp, modified-timestamp, text-content, username {f.k})

(7) user-post-image (post_id {f.k}, image_url)

(8) user-post-video (post-id {F.K}, video_url)

(9) post-comment (post-comment-id, text-content, timestamp, post-id {f.k}, username {f.k})

30. SQL (Structured Query Language)

① SQL used to access and manipulate data.
② SQL used CRUD operation to communicate with DB.

③ what is RDBMS?
- Software that enables u to implement designed relational model.
- eg MySQL, MS SQL, oracle, etc
- table relation is the simplest form of the data storage object in R-DB.
- MySQL is open-source RDBMS, and it uses SQL for all CRUD operation.

- MySQL used client-server model, where client is CLI that used service provided by MySQL server.

④ SQL vs MySQL
- SQL is structured Query language used to perform CRUD operation in R-DB while MySQL is a RDBMS used to store, manage DB.

SQL Data types :-

① INT

② VARCHAR (variable length data types)

③ Types of SQL commands:

a. DDL (Data definition language): defining relational schema.
- CREATE : create table, DB, view.
- ALTER : modifications in table.
- DROP : delete table, DB, view.
- TRUNCATE : remove all tuples from table.
- RENAME : rename DB name, table name.

b. DRL / DQL (Data Retrieval or Data query) lang: retrieve data from tables
- SELECT

c. DML (Data modification language: use to perform modifications in the DB
- INSERT : insert data into a relation.
- UPDATE : update relation data.
- DELETE : delete row from the relation.

d. DCL (Data Control language): grant or revoke authorities from user
- GRANT : access privileges to DB.
- REVOKE : revoke user access privileges

e. TCL (Transaction Control Language): to manage transactions done in the DB
- START TRANSACTION: Begin a transaction.
- COMMIT: apply all the changes & end transaction.
- ROLLBACK: discard changes and end transaction
- SAVEPOINT: checkout within the group of transactions in which to roll back.

MANAGING DB (DDL)
1. Creation of DB
- CREATE DATABASE IF NOT EXISTS name;
- USE DB-name;
- DROP Database if not exists db-name;
- SHOW Database;
- SHOW Tables;

Data Retrieval Language (DRL)
- Syntax: SELECT <set of column name> FROM table_name>;
- order of execution Right to left

- can select use keyword without FROM clause?
yes, using DUAL Tables.

e.g. SELECT 55+11;
SELECT now();

__/__/____

- WHERE
  Reduce rows based on given condn.

- BETWEEN
  Select * from customer where age between 0 AND 30;

- IN
  Reduce OR condn
  select * from ___ where name IN(---, ---);

- AND, OR, NOT
  AND : where cond1 AND cond2
  OR : where cond1 OR cond2
  NOT : where col_name NOT IN(1, 2, 3, 4).

- IS NULL

- Pattern Searching / wildcard
  - % any number of characters
  - '_' only one character
  - select * from customer where name Like '%P_';

- order by
  select * from customer order by name desc, ASC;

distinct
  select distinct dept from table;

-GROUP BY

Group by clause is used to collect data from multiple records and group the result by one or more columns.

Select c1, c2, c3, c4 from tableName where cond
group by c1, c2, c3, c4;

used with aggregation function,
COUNT(), SUM(), AVG(), MIN(), MAX().

- Group by HAVING

out of the categories made by GROUP_BY, we could like to know only particular thing (condition).

select COUNT(cust_id), c2, c3 from table group by country HAVING COUNT(cust_id) > 50;

WHERE VS HAVING

- Both having same function of filtering the row
- WHERE clause is used to filter the row from table.
- HAVING clause is used to filter the rows from group based on specified condn.

CONSTRAINT

1. Primary key
2. Foreign key
3. Unique
4. Check
5. Default.

__/__/____

Alter Operation
Add
modify
change column
drop column
Rename

- Data Manipulation Language
Insert
Update → ON Update cascade
Delete → Delete Cascade, on delete Null.

- Replace
Primarily used for already present tuple in a table.

JOINS

~~(1) Primarily used for already present tuple.~~
(1) ~~All RDBMS are relational in nature, we prefer to other tables to get meaningful outcomes.~~

(2) ~~INNER JOIN~~

(3) ~~OUTER JOIN~~

JOINS

① All RDBMS are relational in nature, we refer to the other tables to get meaningful outcomes.

② PK are used to do reference to other table.

③ INNER JOIN

- Return a resultant table that has matching values from both the tables or all the tables.

- Alias in MySQL (as)
alias in mysql is used to give a temporary name to a table or a columns in a table for the purpose of a query.

④ OUTER JOIN

a. LEFT JOIN
This return a resulting table that all the data from left table and the matched data from the right table.

b. RIGHT JOIN
This returns a resulting table that contains all the data from right table and matched the data from left table.

c. FULL JOIN

This returns a resulting table that all the data when there is match on left or right table data.

⑤ D. CROSS JOIN

This return all the cartesian product of the data present in both tables.

All possible variation are reflected in the output.

⑥ SELF JOIN.

It is used to get the output from a particular table when the same table is joined to itself.

SET Operations

UNION

→ Combines two or more select statements.

INTERSECT

→ Return common values of the table.

MINUS

→ This operator returns the distinct row from the first table that does not occure in the second table.

Sub Queries

1. Outer query depends on inner query
2. Alternate to Joins.
3. Nested Queries

__/__/____

MySQL views

① A view is a database object that has no value. Its contents are based on the base tables. It contains rows & columns similar to the real table.

② In MySQL the view is a virtual table created by a query by joining one or more tables.

31. Normalization

→ Normalization is step towards DB optimization.

a. Functional Dependency (FD)

① It is a relationship between the primary key attribute, of the relation to that of the other attribute of the relation.

② $X \rightarrow Y$ (X: Determinant, Y: Dependent)

③ Type of FD,

i. Trivial FD

If $A \rightarrow B$ has trivial dependency if B is a subset of A.

$A \rightarrow A$, $B \rightarrow B$ are also Trivial FD.

ii. Non-Trivial FD

If $A \rightarrow B$ has ~~trivial~~ non-trivial functional dependency if B is not a subset of A.

($A \not\subseteq B$, $A \cap B = NULL$).

④ Rules of FD (Armstrong's axioms)

i. Reflexive

If A is a set of attribute, & B is a subset of A, then $A \rightarrow B$ holds.

($A \subseteq B$, $A \rightarrow B$)

ii. Augmentation

If B can be determined from A, then adding an

attribute to this functional dependency won't change anything.
$A \rightarrow B$, then $AX \rightarrow BX$ holds.

iii. Transitivity
if A determines B and B determines C, we can that A determine C
$A \rightarrow B, B \rightarrow C$ then $A \rightarrow C$.

b. Why Normalization ?
→ To avoid Redundancy in the DB.

c. What happens when DB have Redundant data ?
→ Insertion, Deletion, Updation anomalies arises.

d. Anomalies (Abnormalities)
→ Anomalies means abnormalities in the data.

i. Insertion
→ when a certain data attribute can't be inserted in the DB beoz of the dependent data attribute without knowing.

ii. Deletion
→ The delete anomaly refers to the situation where deletion of data result in the unintended loss of some other important data.

iii. Updation

→ The update anomaly is when an update of single data point require multiple rows of data to be updated.

And due to Updation at many place, Data Inconsistency occures.

e. what is Normalization ?

i. Normalization is used to minimize the redundancy from a relations. It is also used to eliminate undesirable characteristics like Insertion, Updation, Deletion.

ii. Normalization, divides the Composite attributes into individual attribute, simply larger table into smaller by using relationship among them.

f. Types of Normal forms

i. 1NF → Every Relation cell must have atomic value, Relation must not have multi-value attribute.

ii. 2NF → Relation must be in 1NF.
there should not be any partial dependency.
① All non-prime attribute must fully dependent on PK.
② Non-prime attribute can not depend on the part of the PK.

iii. 3NF → Relation must be in 2NF.
No transitivity dependency exists.
non-prime attribute should not find non-prime attribute.

iv. BCNF → (Boyce-Codd normal form)
Relation must be in 3NF.
FD, A → B, A must be super key.
we must not derive attribute from any prime or non-prime attribute.

2. Advantages of Normalization
- Minimize data Redundancy
- Maintain great data consistency.

32. Transaction
a. A unit of work done against DB in a logical sequence.
b. Sequence is very important in transaction.
c. It is a logical unit of work that contain one or more SQL statements. The result of all these statements in a transaction either get completed successfully (all the change made permanent to DB) or. if at any point any failure happens it gets rollbacked (all the changes being done or undone).

33. ACID Properties

To ensure integrity of the data, we require that the DB system maintain the following ACID Properties.

a. Atomicity :- Either all operation of transaction are reflected properly in the DB or none are.

b. Consistency :- ① Integrity constraints must be maintained before & after transaction.
② DB must be consistent after transa.

c. Isolation :-

① Even though multiple transaction may execute concurrently, the system guarantees that, for every pair transaction T_i & T_j, it appears to T_i that either T_j finished execution before T_i started or T_j started execution after T_i finished.

② Multiple transaction can happens in the system in Isolation, without interfering each other.

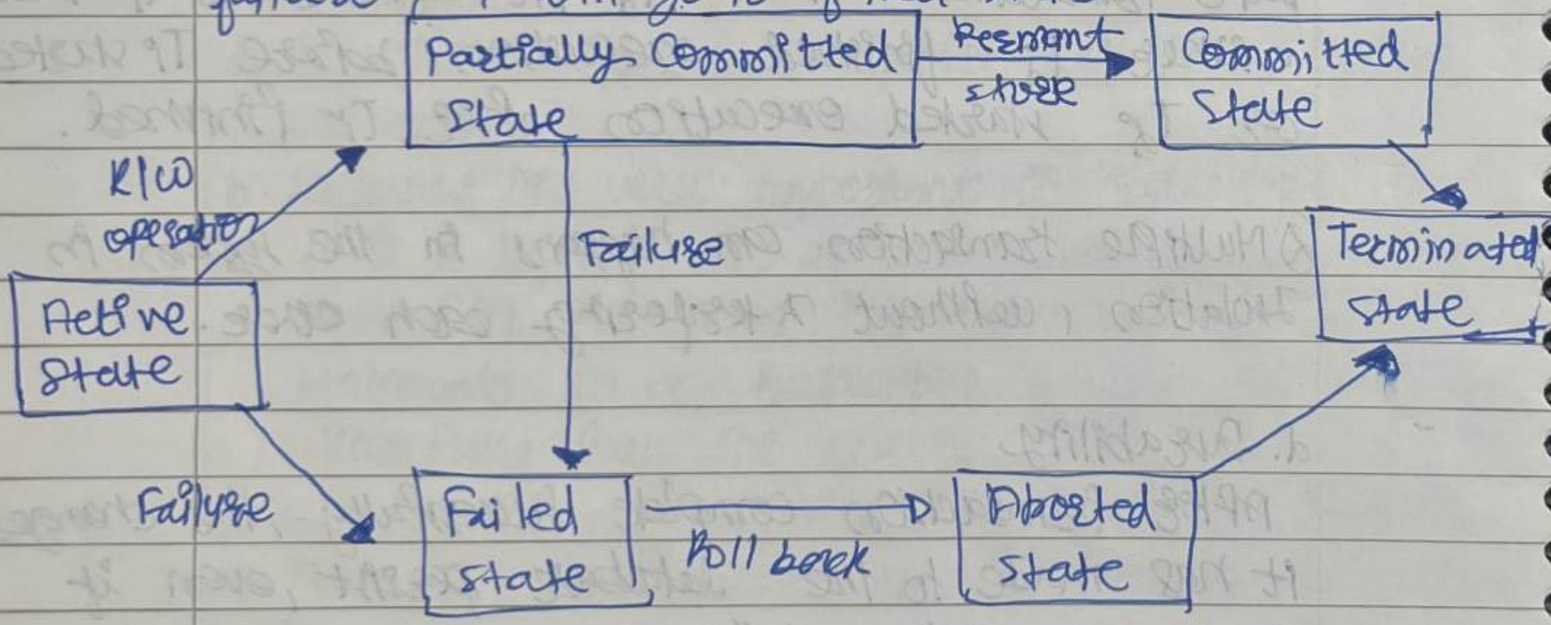
d. Durability

After transaction complete successfully, the changes it has made to the database persist, even if there are system failures.

34. Transaction States

① Active state :- The very first state of the life cycle of the transaction, all the read & write operations are being performed. If they execute without any error the T comes to Partially committed state.
Although if any error occure then it leads to a failed state.

② Partially Committed State :- After transaction is executed the changes are saved in the buffer in the main memory. If the changes made are permanent on the DB then the state will transfer to the committed state and if there is any failure, T will go to failed state.



Transaction State in DBMS

③ Committed state :- when updates are made permanent on the DB. Then the T is said to be in the committed state. Rollback can not be done from the committed states. Now, consistent state is achieved at this stages.

④ Failed state :- when T is being executed & some failure occurs. Due to this it is impossible to continue the execution of T.

⑤ Aborted state :- when t reaches the failed state all the changes made in the buffer are reversed. After that the T roll back completely. T reaches abort state after rollback

⑥ Terminate state :- A transaction is said to have terminated if has either commited or aborted.

35. How to implement Atomicity & Durability in transactions

a. Recovery Mechanism Component of DBMS support atomicity & Durability.

① ~~Shadow~~

⊕ Shadow Copy Scheme

i) Based on Making Copies of DB. (aka shadow copy)

ii) Assumption only one transaction active at each time.

iii) A Pointer called db Pointer maintained on the disk, which at any instant points to current copy of DB.

iv) T, that wants to update DB first create a complete copy of DB.

v) All further updates ~~DB~~ are done on new DB copy leaving the original copy (shadow copy) untouched.

vi) If at any point the T has to be aborted the System deletes the new copy. And the old copy is not affected.

vii) If T success, it is committed as,

① OS makes sure all the pages of the new copy of DB written on the disk.

② DB System updates the dp-pointer to point to the new copy of DB.

③ New copy is now current copy of DB.

④ The old copy is deleted.

⑤ The T is said to have been COMMITED at the point where the updated dp-pointer is written to disk.

viii) Atomicity,

① if T fails at any Time before dp-pointer is updated, the old content of DB are n't affected.

② T abort can be done by just deleting the new copy of DB.
③ Hence, either all updates are reflected or none.

ix) Durability
① Suppose, system fails are any time before the updated db-pointer is written to disk.
② When the system restarts it will be read db-pointer & will thus, see the original content of DB & none of the effects of T will be visible.
③ T is assumed to be successful only db-pointer is updated.
④ If system fails after db-pointer has been updated. Before that all the pages of the new copy were written to disk. Hence when system restarts, it will read new DB copy.

x) The implementation is dependent on write to the db-pointer being atomic.

Inefficient, as entire DB is copied for every T.

c. Log Based Recovery Method
i) Log is a sequence of records. Log of each T is maintained in some stable storage so that if any failure occurs then, it can be recovered from there.

ii) If any operation is performed on the database, then it will be recorded in the log.

iii) But the process of storing the logs should be done before the actual transaction is applied in the database

iv) Stable storage is a classification of computer data storage technology that guarantees atomicity for any given write operation & allows software to be written that is robust against some hardware & power failure.

v) Deferred DB modification

① Ensuring atomicity by recording all the DB modifications in the log but deferring the execution of all the write operation until the final action of the T has been executed.

② Log used information is used to execute deferred writes when T is completed.

③ If system crashed before the T completes, or if T is aborted, the information in the logs are ignored

④ If T completes, the records associated to it in the log file are used in executing the deferred writes.

⑤ If failure occure while this updating is taking place, we perform Redo.

vi) Immediate DB Modification

① DB modification to be output to the DB while ~~modification in the log~~ the T is still active.

② DB modification written by active T are called uncommitted modification.

③ In the event of crash or T failure, system uses old value field of the log records to restore modified value.

④ Update take place only after log records in a stable storage.

⑤ Failure Handling

i) System failure before T completes, or if T aborted then old value field is used to undo the T.

ii) If T completes and system crashes, then new value field is used to redo T having commit logs in the logs.

36. Indexing in DBMS

a. Indexing is used to optimize the Performance of a database by minimizing the number of disk accesses required when a query is processed.

b. The index is a type of data structure. It is used to locate & access the data in a database table quickly.

c. Speed up operation with read operations like SELECT queries, WHERE clause, etc.

d. Search Key: Contains copy of primary key or candidate key of the table or something like.

e. Data Reference: Pointer holding the address of disk block where the value of the corresponding key is stored.

f. Indexing is optional, but increases access speed. It is not the primary mean to access the tuple It is secondary mean.

2. Index file is always sorted.

3. Indexing Methods

i) Primary Index (Clustering Index)

① A file may have several indices, on different search keys. If the data file containing the records is sequentially ordered, a primary index is an index whose search key also defines the sequential order of the file.

② NOTE: The term primary index is sometimes used to mean an index on primary key. However, such usage is nonstandard & should be avoided.

③ All files are ordered sequentially on some search key. It could be primary key or non-primary key.

(4) Dense And Sparse Indices

(i) Dense Index

- The dense index contains an index record for every search key value in the data field file.
- The index record contains the search key value & a pointer to the first data records with that search key value. The rest of the records with the same search key value could be stored sequentially after the first record.
- It needs more space to store index record itself. The index records have the search key & a pointer to the actual record on the disk.

(ii) Sparse Index

- An index record appears for only some of the search-key values.
- Sparse index helps you to resolve the issues of dense indexing in DBMS. In this method of indexing techniques, a range of index columns stores the same data block address, and when data need to be retrieved, the block address will be fetched.

(5) Primary indexing can be based on data file is sorted with respective primary key attribute or non-key attributes.

(6) Based on key attribute
(i) Data file is sorted w.r.t primary key attrib.
(ii) PK will be used as search-key in index.
(iii) Sparse Index will be formed i.e no. of entries in the index file = no. of blocks in datafile.

(7) Based on Non-key attribute
(i) Data file is sorted w.r.t non-key attribute.
(ii) No. of entries in index = Unique non-key attri values in the data file.

(iii) This is dense index as, all the unique values have an entry in the index file.

(iv) E.g Let's assume that a company recruited many employees in various departments.

(8) Multi-Level Index
(i) Index with 2 or more levels.
(ii) If the single level index become enough large that the binary search itself would take much time, we can break down Indexing into multiple levels.

ii) Secondary Index (Non-Clustering Index).

a. Datafile is unsorted. Hence, Primary Indexing is not possible.
b. can be done on key or non-key attributes.
c. called secondary Indexing because normally Primary Indexing already done.
d. No. of entries in the index file = No. of records in the data file.
e. It is example dense index.

i. Advantages of Indexing
   (1) Faster Access & retrieval of data.
   (2) IO is less.

j. Limitation of Indexing
   (1) Additional space to store Index table.
   (2) Indexing decrease performance of INSERT, DELETE & UPDATE query.

37. NOSQL (Not only SQL)

① NoSQL are non tabular databases and store data differently than Relational tables.
They provide flexible schemas and scale easily with large amount of data & high user loads.
- They are schema free.
- Data structure used are not tabular, they are more flexible, has the ability to adjust dynamically.
- Can handle huge amount of data.
- Most of the NOSQL are open source and has capability of horizontal scaling.
- It just stores data in some format other than Relational.

② History
- NOSQL database emerged in the late 2000s as the cost of storage dramatically decreased. Gone were the days of needing to create a complex, difficult to manage data model in order to avoid data duplication. So, NOSQL databases optimized for developers productivity.

- Data becoming unstructured more, hence structuring (define schema in advance) them had becoming costly.

- NOSQL database allow developers to store huge amount of unstructured data, giving them a lot of flexibility.

③ NOSQL Database Advantages

A. Flexible schema

- RDBMS has predefined schema, which become an issue when we do not have all the data with us or we need to change the schema. Its huge task to change schema on the go.

B. Horizontal Scaling

- Horizontal scaling also known as scale out, refers to bringing on additional nodes to share the load. This is difficult with relational database due to the difficulty in spreading out related data across nodes. With non-relational databases, this is made simpler since collection are self-contained and not coupled relationally.

C. High Availability.

D. Easy Insert and Read Operation.