DBMS Notes

12 September 2023 08:28 PM

Data: raw and isolated facts about any subject or entity

Information: Processed meaningful data.

Database. collection of related data.

Unstructured DB: DBs with no undefined schemas that can store datas of various formats. (social media platforms etc)

Structured DB: DBs with rigid predefined schemas that can store data of a specific type only. (management systems)

Database Management System (DBMS)

→ Perform operations like delete, insert etc.

→ Manage database in an efficient way.

→ efficient and convenient environment to use.

Relational DBMS

→ DBMS designed for relational databases.

→ SQL, MySQL, Oracle are examples.

Relational Databases

→ Database that stores data in tables (relations)

File System

→ Manages and organises files

Database System.

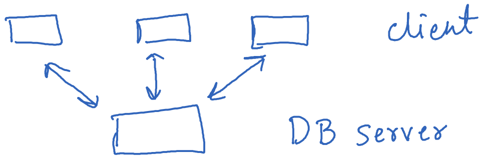
→ Manages and organises databases.

Advantages of DB over files.

- Ineffective utilization of memory and high input-output cost
  - **Large file transfer**. Inefficient memory and time utilization
  - In DBMS, only a train record will be retrieved using DBMS query. E.g., find a train record in IRCTC
- Difficulty in accessing data
  - **Need metadata**. E.g., need actual file location and name
  - In DBMS, simple query or API can be used to access the data without knowing location or other attributes. E.g., search train information
- Data redundancy
  - **Duplication of information in a file**, multiple same file with different formats, duplication of same information in different files
  - In DBMS, constraints such as primary key, foreign key are present
- Data inconsistency
  - Inconsistency can arise when we change just one part of redundant information present in file(s)
- Concurrent access by multiple users
  - Concurrent access needed for performance. E.g., in IRCTC lakhs of transactions are done in a day
  - Uncontrolled concurrent accesses can lead to **inconsistencies**
  - In DBMS, protocols exist to ensure concurrency
- Security problems
  - Unavailability of **role-based data access**
  - DBMS provide role-based security
    - Different role for different users such as student, faculty, dean role in university database

Data base Architecture

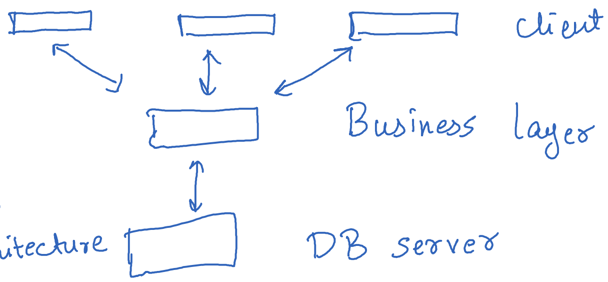
1) Two Tier



adv
→ simple
→ easy to maintain

disadv
→ security issues
→ overloading

2) Three Tier



Web services use this architecture

Adv
→ load decreased
→ security
→ scalability

Disadv
→ maintenance

Data Abstraction

[illegible] the

Data Abstraction

The process of hiding irrelevant details from the user to ease the user interaction with the database.

1) Physical level: user doesn't how a data is stored
location, name, indexing.

2) Logical level: user doesn't know how data is related

3) View level: Application program hide certain details of data.
A part of information is available to user.

View
↓
logical
|
physical

Schema

→ blueprint of the database.

→ Frame work to describe the structure of a database system.

→ to achieve data abstraction, Schema is used.

Levels:

External Schema: the view of data to users (like different interface for different users), security / access control

Logical Schema: the overall logical structure of schema.
Blueprint of the DB.

Physical Schema: The physical structure of the database
(like device used to store, pendrive etc)
hardware, indexing, organisation, access paths etc.

Instance

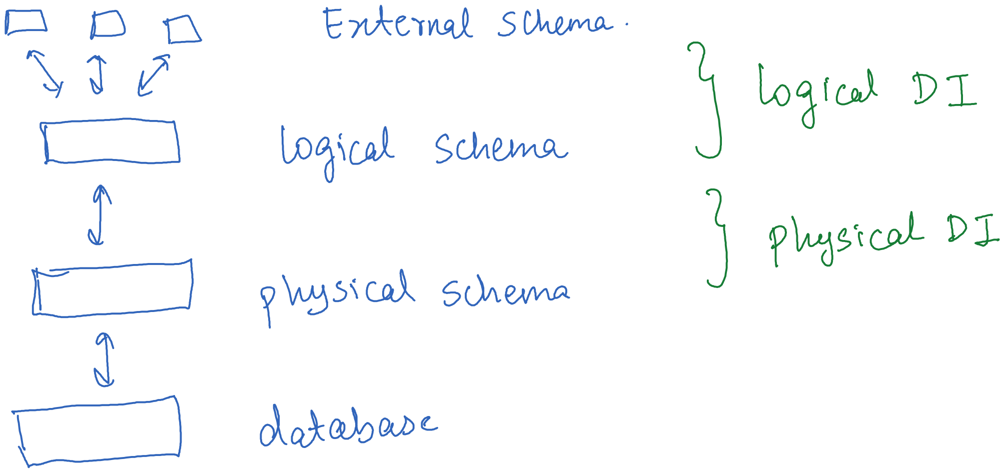
The actual content of the database at a particular point of time.

Data Independence (DI)

change DB schema at one level of DBMS without changing the DB schema at the next higher level.

Logical DI : ability to change the logical schema without changing the external schema

Physical DI : ability to modify the physical schema without changing the logical schema.



Data Models

Tools for describing
→ Data
→ Data relationships
→ Data constraints.
→ Data sematics

Types of Models
→ Hiearchial
→ Network
→ Relational
→ XML
→ E R model.

Relational Data Model.

-> managed using Data Description language (DDL) and data manipulation language (DML)

-> Data stored in tables

-> columns are attributes of data.

Database Design.

Logical design: deciding the schema

Business decision: what to store in DB

Logical decision: what relations must be implemented and how attributes are distributed

Physical Design: deciding the physical layout of the DB.

Online Processing System.

-> operate data of a business environment.

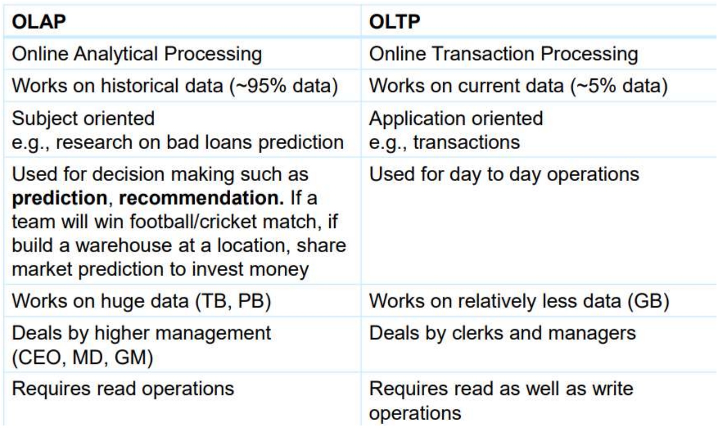
-> Analyses aggregated stored data.

-> captures, stores and processes data from transaction in real time.

Types: OLAP, OLTP

Why types: low performance due to huge data

Access time depends on data size.



OLAP	OLTP
Online Analytical Processing	Online Transaction Processing
Works on historical data (~95% data)	Works on current data (~5% data)
Subject oriented e.g., research on bad loans prediction	Application oriented e.g., transactions
Used for decision making such as **prediction**, **recommendation.** If a team will win football/cricket match, if build a warehouse at a location, share market prediction to invest money	Used for day to day operations
Works on huge data (TB, PB)	Works on relatively less data (GB)
Deals by higher management (CEO, MD, GM)	Deals by clerks and managers
Requires read operations	Requires read as well as write operations

Relations

→ tables with attributes (columns) and records (rows)

- Columns are attributes
- Rows are tuples or records
- The set of allowed values for each attribute is called the **domain** of the attribute
- Attribute values are (normally) required to be **atomic**; that is, indivisible

Candidate Key

→ uniquely identify any two tupples in the table

→ can be more than one.

Primary Key

→ Candidate Key

→ unique + not null

→ not more than one

primary key (attribute)

Alternate Key

→ set of candidate Keys which are not primary key

Super Key

→ combinate of all possible attributes that can identify tupples.

→ super set of candidate key

Foreign Key

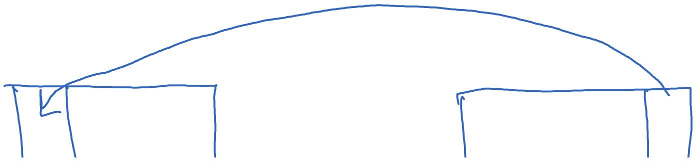
→ attributes which reference to the primary key of same or other table

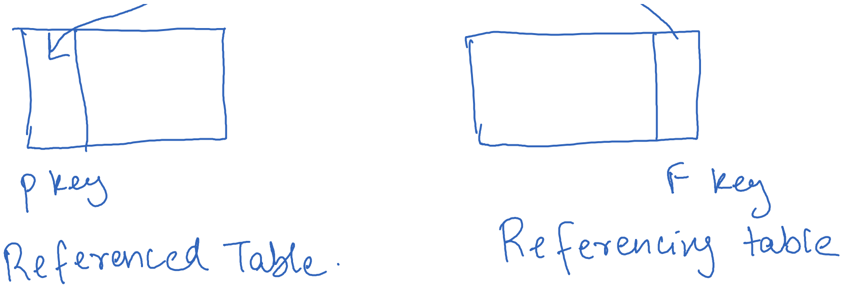
→ maintains referential integrity

→ can be multiple.

* integrity : maintains same value in DB

foreign key (attribute) references tablename (attribute)





Deletion in Referenced Table

- On Delete Cascade
  - If referential integrity violation occurs, delete corresponding record from both the tables
- On Delete No Action
  - Referenced attribute value deletion is restricted, if this attribute is referred by foreign key of referencing table
    - If it is essential to delete, delete in referencing table then in referenced table
- On Delete Set Null
  - If referential integrity violation occurs, place NULL in the corresponding foreign key attribute

ER Model

→ used in high level RDBMS

→ has an associated diagrammatic representation ER diagram

→ Logical representation of database.

→ components : attribute, entity, relationship.

Entity

- An object in real world
  - Example: Student, course, faculty

Attribute

- Characteristics of an entity
  - Example: {roll no, age, address} can be attribute of Student entity

Relationship

- Connection or association between entities

An **entity** is an object that exists and is distinguishable from other objects

- Example: specific person, company, event, plant

An **entity set** is a set of entities of the same type that share the same properties

- Example: set of all persons, companies, trees, holidays

An entity is represented by a set of attributes; i.e., descriptive properties possessed by all members of an entity set

- Example:

  instructor = (*ID*, *name*, *salary*)
  course= (*course_id*, *title*, *credits*)

A subset of the attributes form a **primary key** of the entity set

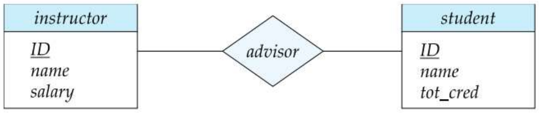
Representation

Rectangles represent entity set

ellipse represent attribute

underline represent P Key.

Relationship.



- A **relationship** is an association among several entities

  Example:

44553 (Peltier)	*advisor*	22222 (Einstein)
student entity	relationship set	*instructor* entity

- A **relationship set** is a mathematical relation among *n* ≥ 2 entities, each taken from entity sets

(represented by diamond)

types of relationship

Binary relationship

- Involve two entity sets (or degree two)
- Most relationship sets in a database system are binary

Relationships between more than two entity sets are rare. Most relationships are binary

- Example: *students* work on research *projects* under the guidance of an *instructor*.
- Relationship *proj_guide* is a ternary relationship between *instructor, student,* and *project*

Attributes

types :

- **Simple** and **composite** attributes
  - Simple Attribute cannot be divided further
    - Example: student age
  - Composite attribute can be divide further
    - Example: student name (first name, middle name, last name), student address
- **Single-valued** and **Multi-valued** attributes
  - Single-valued attribute has only one value
    - Example: Student registration_number
  - Multi-valued attribute has more than one vlaue
    - Example: Student phone_numbers, address
- Complex attributes
  - Composite + Multi-valued
    - Example: Two address of a student

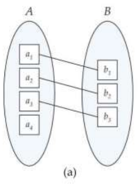
- Attribute types
  - Stored attributes
    - Cannot be derived
    - Example: date_of_birth
  - Derived attributes
    - Can be computed from other attributes
    - Example: age given date_of_birth
  - Key and non-key attributes
    - Key is an unique attribute
    - Example: registration number is a key attribute in student entity
- **Domain** – the set of permitted values for each attribute
- Representation of different types of attributes
  - Derived: represented in dotted eclipse
  - Multivalued: represented in double eclipse
  - Key Attribute: underlined in eclipse

Mapping Cardinality

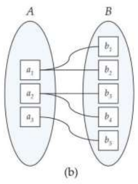
→ useful to know how two entities are associated with each other.

Types :

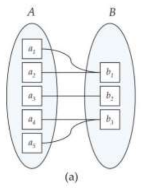
1) one - one
2) one - many
3) many - one
4) many - many



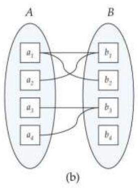
One to one



One to many



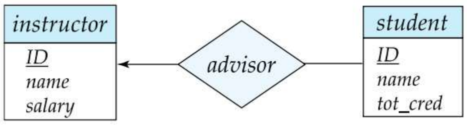
Many to one



Many to many

→ means one

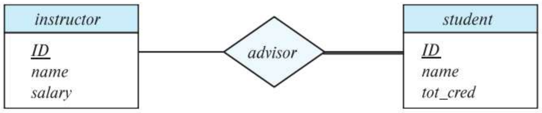
— means many



one to many

PARTICIPATIONS

Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set



- Participation of *student* in *advisor* relation is total
  - Every *student* must have an associated instructor

Partial participation: Some entities may not participate in any relationship in the relationship set

- Example: participation of *instructor* in *advisor* is partial

WEAK ENTITY SETS

→ entity set with no key

→ maybe because there is no attribute to differentiate.

→ Allowed in ER diagram

→ Not allowed in RDBMS

Self - Referenced Set

→ one entity of entity set is related to another entity in the same entity set.

Imp points

→ in one-to-one models, the pkey of relationship set shouldn't be combination of Pkeys of its connecting sets. only one of the pkeys should be made the pkey of relationship set.