DBMS - Database Management System

- Basic Introduction

  Definition , 2&3 tier Architecture, 3 schema Architecture (Data Independence)

- E-R model

  Entity - Relationship Model, Different attributes, Types of Relationships

- Basics of Keys

  Primary key, Candidate & super keys, Foreign key

- Normalization

  Closure method, Functional dependencies, 1st 2nd 3rd Normal Form, BCNF

- Transaction control & concurrency

  ACID properties, R-W W-R W-W problems, Conflict Serializability, Recoverability ,2PL

- SQL and Relational Algebra

  DDL,DML,DCL commands, Aggregate functions, Joins, Nested query

- Indexing

  Primary, Cluster, Secondary, Btree, B+tree

Introduction

A Database System consists of two things one is Database and DBMS.

What is a Database? - It is a collection of Related data. Database is of **2** types: Structured Database and Unstructured Database.

Structured Database is where data is stored in a particular shape (table technically known as Relation) Ex: IRCTC, University ERP's.
Unstructured Database is where there is no particular shape/structure to store data. Ex: Webpages. 90% of the data in the world is unstructured.

Data is stored at backend (hard drives, servers) data gets inserted, deleted , updated constantly; in other words operations on data occur constantly.

DBMS is all about managing the data in the database by performing operations on Relation hence it is also known as **RDBMS (Relational Database Management System**).

File System - before DBMS

Using File system one can access his/her data only and is inbuilt inside the Operating Systems.Data is stored line by line without any proper structure. File System has more redundancy. When users exist at different locations, the file system can not help in accessing the data, hence we began using **client server architecture** - Since the data is centralized, It helps in using one's data by many users throughout the world.

Advantages of DBMS:

Fast searching, Efficient Memory usage, Attribute Independent, Concurrency, Security (Role based access control), Data Integrity.

2 tier and 3 tier Architecture

2 tier(layer) architecture/ client-server architecture:

There is no need for any direct connection b/w the Client layer(machines) and Server Layer(database). A request comes from the client layer to the server , the server processes (executes) the request and fetches the required data. Here, Server gets severally loaded. EX: IRCTC database at railway station, Bank database.

Advantages: Authorized clients, easy maintenance.
Disadvantages : Numerous users (Scalability) leads to database crashdown, Security - Client is having direct interaction with the database which is a vulnerability.

3 tier(layer) architecture :

It consists of 3 layers : client layer, Business layer (Application server), server layer.

Client layer is all about Graphical User Interface and the business layer supports the client layer. A requests comes from the client layer to the business layer , the business layer processes (executes) the request and fetches the required data. Here the server doesn't get loaded since the request is passed to the business layer. Business layer behaves as an intermediary between the client and server.

Advantages : Scalability, Security are maintained. The user never knows where the data actually exists.

Schema

Schema is a logical Representation of a database. Logical Representation means representation in tables(Relation) ; when users access the data it is represented in tables (Relation). The schema is implemented using SQL (Structured Query Language).

Three Schema Architecture

We know schema means structure; the main motto of three schema architecture is data independence.

User -> External Schema -> Conceptual Schema -> Physical Schema -> Database
User -> View level -> Logical Level-> Physical (internal) Level -> Database

External Schema is also known as view level; the job of view level is how to represent(show) the data to the user. The view will not be the same for all the users. It can restrict the data access according to the respective authorizations. In a bird's eye view it is the first view after we login to a certain portal.
Ex: In an ERP system a faculty has one view whereas the student has another view.

Conceptual Schema has the details of the relationship between the tables. It is the blueprint of how data is gonna be organized.

Physical Schema has the information of location of the data stored and the form in which it is stored (file form/ extension form…); it is the place where we get to know where the data is physically present.

Don’t get confused, the data stored in tables is wrt to the user but in actual it is stored in file format in the server(hard disk).

A database administrator works on Physical Schema and a database designer works on Conceptual Schema and a front-end designer works on External Schema. Data can be stored either in a centralized manner or in a distributed manner.

Data Independence/ Data Abstraction

A user always accesses the data with the help of a web application/ Interface. Data Independence is making data independent from the user.In other words, never letting the user know where & how the data is actually stored.

There are 2 type of data independence : Logical & Physical data independence

Logical Data Independence:

In the conceptual schema, any user made changes will not affect the other users i.e no change at the view level and the changes made by the user will be displayed only to him.

Physical Data Independence:

Similar to above, any changes made at the physical schema will not affect the conceptual schema which also implies no effect on the External schema of the web application.

Both these logical data Independence and Physical data Independence try to provide transparency b/w the user and database.

Candidate Key

Key is an attribute used to uniquely identify n two tuples in the table. Collection/set of all such keys is known as candidate key. All the keys in the candidate key are unique. There can be any number of keys in the candidate key.

Ex: For a student data, <Aadhar card, voter id, mail id, mole marks , phone no> form the candidate key.

Primary key

The most important key is known as the primary key and the rest are all known as alternate keys. The primary key cannot be null and is always unique thus , these two form the criteria to identify the primary key.

- Primary key value is never taken from the user and is always given to the user. Ex; Id no, Passport num, license num.
- There can be only one primary key in a table

Foreign Key

It is an attribute or set of attributes that references the primary key of the same table or other table(Relation).

- It maintains Referential Integrity.
- There can be more than one foreign key in a table

Two tables are said to be related if there exists any common attribute (column) between them; the name of the attribute need not be the same in the two tables.

The table having the foreign key which refers to the primary key is known as the **referencing table.**

The table having the primary key which is being referred by the foreign key is known as a **referenced/base table**.

A table can be both referencing and referenced if both foreign key and primary key exist in the same table.

RNO	Name	Place
1	A	Delhi
2	B	Mumbai
3	A	AP

Referenced Table

CNO	Course Name	RNO
112	C1	1
113	C2	2
114	C3	3

Referencing Table

Ex:

Referential Integrity

Integrity means having the same value throughout the database

Referenced Table :

Any **insertion** to the referenced table makes **no violation** to the referential integrity.

Any **deletion** from the referenced table **may cause a violation** to the referential integrity.

- The solution to this violation is **" on delete cascade"**; it means the moment a data gets deleted from the base table, the related data gets deleted automatically from all the referencing tables. It is the best method.

- Another solution is **"on delete set null"**; it means the foreign key value of the deleted data, becomes null in the referencing table keeping the remaining attributes unchanged. This solution is applicable only if the foreign key doesn't behave as a primary key to any other referencing table.

Any **updation** from the referenced table **may cause a violation** to the referential integrity.

- The solution to this violation is **" on update cascade"**; it means the moment a data gets updated from the base table, the related data gets updated automatically from all the referencing tables. It is the best method.

- Another solution is **"on update set null"**; it means the foreign key value of the updated data becomes null in the referencing table keeping the remaining attributes unchanged. This solution is applicable only if the foreign key doesn't behave as a primary key to any other referencing table.

Referencing Table:

Any **insertion** to the referencing table **may cause a violation** to the referential integrity.
Any **deletion** from the referencing table makes **no violation** to the referential integrity.
Any **updation** from the referencing table **may cause a violation** to the referential integrity.

Super Key