

III/I (R16)

DATABASE MANAGEMENT SYSTEMS

UNIT -1

SYLLABUS

UNIT-I: An Overview of Database Management,

- Introduction
- What is Database System
- What is Database
- Why Database
- Data Independence
- Relation Systems and Others Summary

Database system architecture, Introduction

- The Three Levels of Architecture
 - The External Level
 - The Conceptual Level
 - The Internal Level
- Mapping
- The Database Administrator
- The Database Management systems
- Client/Server Architecture.

INTRODUCTION:-

Evolution of databases

Databases have evolved since their inception in the 1960s, beginning with hierarchical and network databases, through the 1980s with [object-oriented databases](#), and today with SQL and NoSQL databases and [cloud databases](#).

In one view, databases can be classified according to content type: bibliographic, full text, numeric and images. In computing, databases are sometimes classified according to their organizational approach. There are many different kinds of databases, ranging from the most prevalent approach, the relational database, to a [distributed database](#), [cloud database](#) or NoSQL database

A database is a collection of [information](#) that is organized so that it can be easily accessed, managed and updated.

Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information. Data gets updated, expanded and deleted as new information is added. Databases process workloads to create and update themselves, querying the data they contain and running applications against it.

What is Database System?

Database System in short refers to the technology of storing and retrieving users' data with utmost efficiency along with appropriate security measures. I

What is Data?

In simple words data can be facts related to any object in consideration. For example your name, age, height, weight, etc are some data related to you. A picture , image , file , pdf etc can also be considered data.

What is a Database?

Database is a systematic collection of data. Databases support storage and manipulation of data. Databases make data management easy. Let's discuss few examples.

An online telephone directory would definitely use database to store data pertaining to people, phone numbers, other contact details, etc.

Your electricity service provider is obviously using a database to manage billing, client related issues, to handle fault data, etc.

Let's also consider the facebook. It needs to store, manipulate and present data related to members, their friends, member activities, messages, advertisements and lot more.

Why Database?

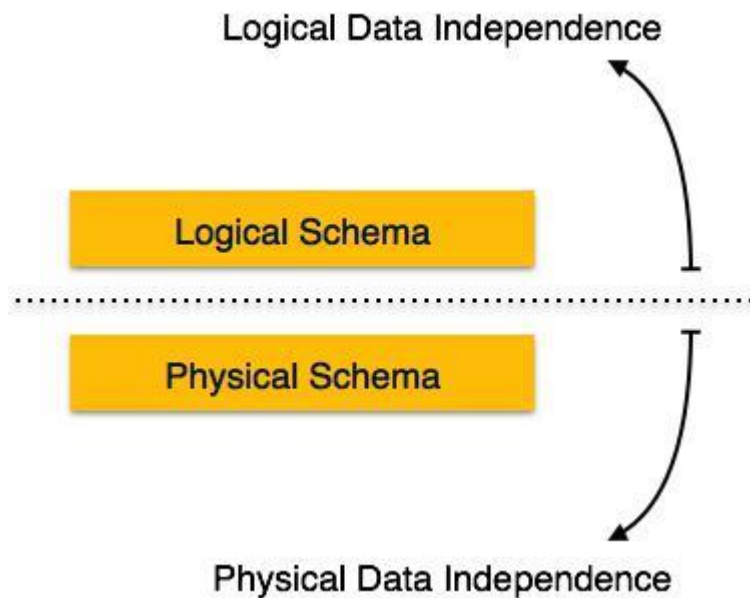
A database is an organized collection of various forms of data. It is also known as a structured set of data that is accessible in many ways through your computer.

The various reasons for which we require databases are:

- To manage large chunks of data: Yes, you can store data into a spreadsheet, but if you add large chunks of data into the sheet, it will simply not work. For instance: if your size of data increases into thousands of records, it will simply create a problem of speed.
- Accuracy: When doing data entry files in a spreadsheet, it becomes difficult to manage the accuracy as there are no validations present in it.
- Ease of updating data: With the database, you can flexibly update the data according to your convenience. Moreover, multiple people can also edit data at same time.
- Security of data: There is no denying the fact that your data is less secure in spreadsheets. Anyone can easily get access to file and can make changes to it. With databases you have security groups and privileges you set to restrict access.
- Data integrity: Data integrity also becomes a question when storing data in spreadsheets. In databases, you can be assured of accuracy and consistency of data due to the built in integrity checks and access controls.

Data Independence

A database system normally contains a lot of data in addition to users' data. For example, it stores data about data, known as metadata, to locate and retrieve data easily. It is rather difficult to modify or update a set of metadata once it is stored in the database. But as a DBMS expands, it needs to change over time to satisfy the requirements of the users. If the entire data is dependent, it would become a tedious and highly complex job.



Metadata itself follows a layered architecture, so that when we change data at one layer, it does not affect the data at another level. This data is independent but mapped to each other.

Logical Data Independence

Logical data is data about database, that is, it stores information about how data is managed inside. For example, a table (relation) stored in the database and all its constraints, applied on that relation.

Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

Physical Data Independence

All the schemas are logical, and the actual data is stored in bit format on the disk. Physical data independence is the power to change the physical data without impacting the schema or logical data.

For example, in case we want to change or upgrade the storage system itself – suppose we want to replace hard-disks with SSD – it should not have any impact on the logical data or schemas.

REALTIONAL SYSTEMS AND OTHERS:

Relational data model is the primary data model in the Relational Systems, which is used widely around the world for data storage and processing. This model is simple and it has all the properties and capabilities required to process data with storage efficiency.

Concepts

Tables – In relational data model, relations are saved in the format of Tables. This format stores the relation among entities. A table has rows and columns, where rows represents records and columns represent the attributes.

Tuple – A single row of a table, which contains a single record for that relation is called a tuple.

Relation instance – A finite set of tuples in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

Relation schema – relation schema describes the relation name (table name), attributes, and their names.

Relation key – Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

Attribute domain – Every attribute has some pre-defined value scope, known as attribute domain.

DATABASE SYSTEM ARCHITECTURE:-

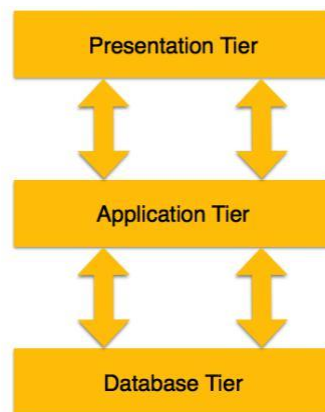
The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.

3-tier Architecture

3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.



- **Database (Data) Tier** – At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
- **Application (Middle) Tier** – At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer

sits in the middle and acts as a mediator between the end-user and the database.

- **User (Presentation) Tier** – End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.

The Three Levels of Architecture:-

Following are the three levels of database architecture,

1. Physical Level
2. Conceptual Level
3. External Level

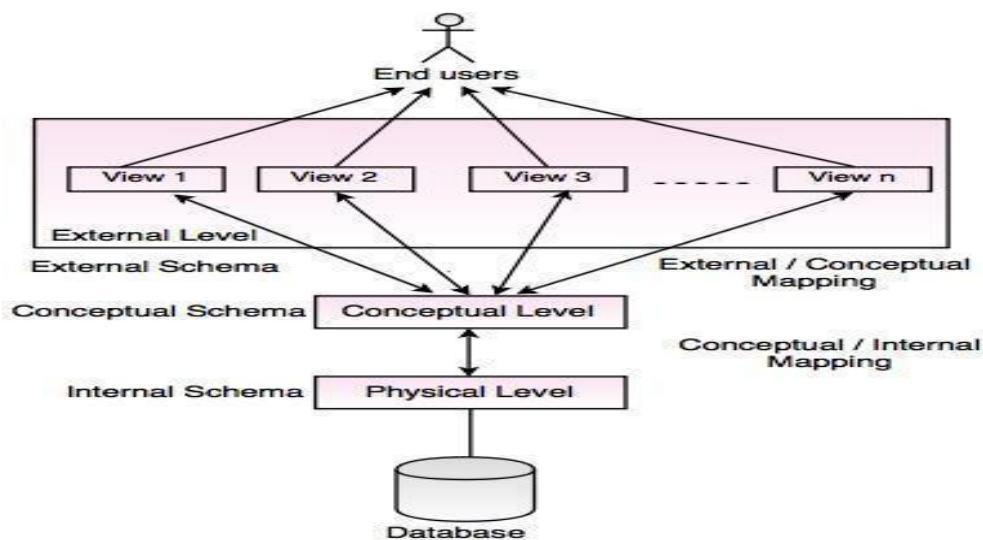


Fig. Three Level Architecture of DBMS

In the above diagram,

it shows the architecture of DBMS.

Mapping is the process of transforming request response between various database levels of architecture.

Mapping is not good for small database, because it takes more time.

In External / Conceptual mapping, DBMS transforms a request on an external schema against the conceptual schema.

In Conceptual / Internal mapping, it is necessary to transform the request from the conceptual to internal levels.

1. Physical Level

Physical level describes the physical storage structure of data in database.

It is also known as Internal Level.

This level is very close to physical storage of data.

At lowest level, it is stored in the form of bits with the physical addresses on the secondary storage device.

At highest level, it can be viewed in the form of files.

The internal schema defines the various stored data types. It uses a physical data model.

2. Conceptual Level

Conceptual level describes the structure of the whole database for a group of users.

It is also called as the data model.

Conceptual schema is a representation of the entire content of the database.

These schema contains all the information to build relevant external records.

It hides the internal details of physical storage.

3. External Level

External level is related to the data which is viewed by individual end users.

This level includes a no. of user views or external schemas.

This level is closest to the user.

External view describes the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

DATABASE ADMINISTRATOR:-

A database administrator (DBA) directs or performs all activities related to maintaining a successful [database](#) environment. Responsibilities include designing, implementing, and maintaining the database system; establishing policies and procedures pertaining to the management, [security](#), maintenance, and use of the [database management system](#); and training employees in database management and use.

ROLE OF DBA:-

- Actively participates in planning the installation of new organization-wide systems and applications. He also assists during the installation as per specific functions.
- Implements the work plan for the Department. He meets the staff to identify any problems, take measures, and resolves it.
- Monitors the efficiency and effectiveness of all database resources and thus, keep the flow of work uninterrupted owing to technology.

- Ensures maximum service through identification of opportunities for improvement and make new recommendations.
- Continuous review and evaluation of the software, hardware, service delivery, and updates as and when required.
- Troubleshoot the problems [if any]. It includes a quick understanding of the problem and its resolution, restoration of the data, rectify the issue and minimize the damage.

The Database Management Systems:-

A **database management system** stores data in such a way that it becomes easier to retrieve, manipulate, and produce information.

Characteristics:-

Traditionally, data was organized in file formats. DBMS was a new concept then, and all the research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics –

- **Real-world entity** – A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use students as an entity and their age as an attribute.
- **Relation-based tables** – DBMS allows entities and relations among them to form tables. user can understand the architecture of a database just by looking at the table names.
- **Isolation of data and application** – A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- **Less redundancy** – DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.
- **Consistency** – Consistency is a state where every relation in a database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state. A DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.

- **Query Language** – DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.
- **ACID Properties** – DBMS follows the concepts of **A**tomicity, **C**onsistency, **I**solation, and **D**urability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database stay healthy in multi-transactional environments and in case of failure.
- **Multiuser and Concurrent Access** – DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.
- **Multiple views** – DBMS offers multiple views for different users. A user who is in the Sales department will have a different view of database than a person working in the Production department. This feature enables the users to have a concentrate view of the database according to their requirements.
- **Security** – Features like multiple views offer security to some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. Additionally, it can also be managed how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

Client/Server Architecture:-

Client-server architecture (client/server) is a network **architecture** in which each computer or process on the network is either a **client** or a **server**. Servers are powerful computers or processes dedicated to managing disk drives (file servers), printers (print servers), or network traffic (network servers).

In client/server architecture, the database application and the database are separated into two parts: a front-end or client portion, and a back-end or server portion. The client executes the database application that accesses database information and interacts with a user through the keyboard, screen, and pointing device such as a mouse. The server executes the Oracle software and handles the functions required for concurrent, shared data access to an Oracle database.

Although the client application and Oracle can be executed on the same computer, it may be more efficient and effective when the client portion(s) and server portion are executed by different computers connected via a network.

The below figure shows the interaction between Client and |Server:-

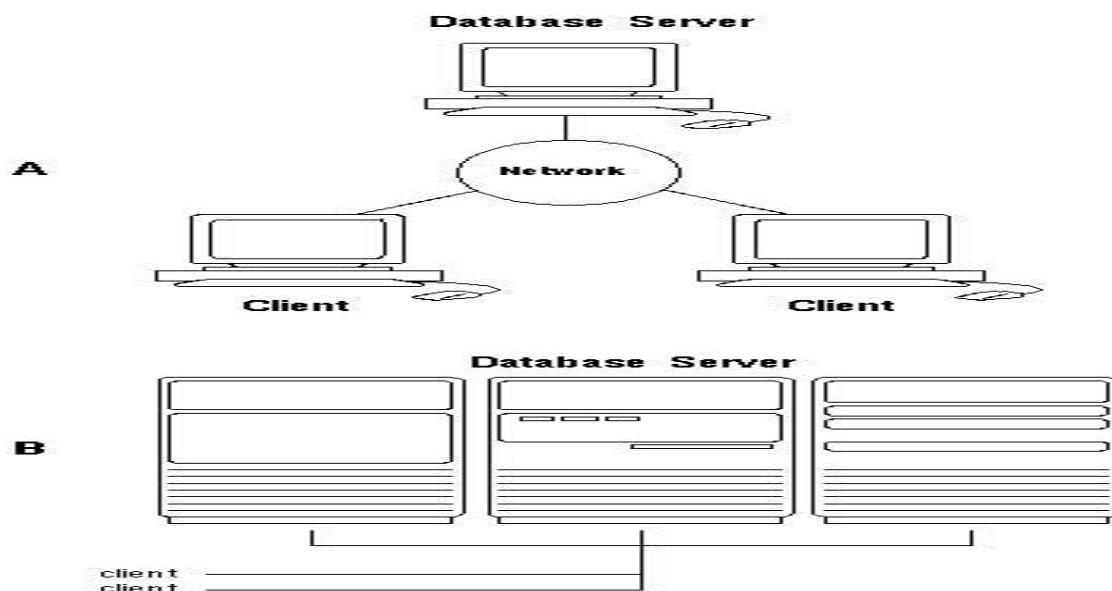


Figure:- **The Client/Server Architecture and Distributed Processing**

Benefits of the Oracle client/server architecture in a distributed processing environment include the following:

- Client applications are not responsible for performing any data processing. Client applications can concentrate on requesting input from users, requesting desired data from the server, and then analyzing and presenting this data using the display capabilities of the client workstation or the terminal (for example, using graphics or spreadsheets).
- Client applications can be designed with no dependence on the physical location of the data. If the data is moved or distributed to other database servers, the application continues to function with little or no modification.

- Oracle exploits the multitasking and shared-memory facilities of its underlying operating system. As a result, it delivers the highest possible degree of concurrency, data integrity, and performance to its client applications.
- Client workstations or terminals can be optimized for the presentation of data (for example, by providing graphics and mouse support) and the server can be optimized for the processing and storage of data (for example, by having large amounts of memory and disk space).
- If necessary, Oracle can be *scaled*. As your system grows, you can add multiple servers to distribute the database processing load throughout the network (*horizontally scaled*). Alternatively, you can replace Oracle on a less powerful computer, such as a microcomputer, with Oracle running on a minicomputer or mainframe, to take advantage of a larger system's performance (*vertically scaled*). In either case, all data and applications are maintained with little or no modification, since Oracle is portable between systems.
- In networked environments, shared data is stored on the servers, rather than on all computers in the system. This makes it easier and more efficient to manage concurrent access.
- In networked environments, inexpensive, low-end client workstations can be used to access the remote data of the server effectively.
- In networked environments, client applications submit database requests to the server using SQL statements. Once received, the SQL statement is processed by the server, and the results are returned to the client application. Network traffic is kept to a minimum because only the requests and the results are shipped over the network.