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.

Data - Data is meaningful known raw facts that can be processed and stored as information.

Database - Database is a collection of interrelated and organized data. In general, it is a collection of files (tables).

DBMS - Database Management System (DBMS) is a collection of interrelated data (usually called database) and a set of programs to access, update and manage those data (which form part of management system) **OR** It is a software package to facilitate creation and maintenance of computerized database.

Importance: Database systems have become an essential component of life in modern society, in that many frequently occurring events trigger the accessing of at least one database: bibliographic library searches, bank transactions, hotel/airline reservations, grocery store purchases, online (Web) purchases, etc., etc.

Traditional vs. more recent applications of databases: The applications mentioned above are all "traditional" ones for which the use of rigidly-structured textual and numeric data suffices. Recent advances have led to the application of database technology to a wider class of data. Examples include **multimedia** databases (involving pictures, video clips, and sound messages) and **geographic** databases (involving maps, satellite images). Also, database search techniques are applied by some WWW search engines.

Definition: A **database management system** (DBMS) is a collection of programs enabling users to create and maintain a database. More specifically, a DBMS is a general purpose software system facilitating each of the following (with respect to a database):

a) **Data Definition:** specifying data types (and other constraints to which the data must conform) and data organization.

b) **Construction:** the process of storing the data on some medium (e.g., magnetic disk) that is controlled by the DBMS

c) **Manipulation:** querying, updating, report generation.

d) **Sharing:** allowing multiple users and programs to access the database "simultaneously"

e) **System protection:** preventing database from becoming corrupted when hardware or software failures occur

f) **Security protection:** preventing unauthorized or malicious access to database.

Given all its responsibilities, it is not surprising that a typical DBMS is a complex piece of software. A database together with the DBMS software is referred to as a **database system**.

Ex:- oracle, Mysql, DBq, etc

Main Characteristics of database approach:

1. **Self-Description:** A database system includes —in addition to the data stored that is of relevance to the organization— a complete definition/description of the database's structure and constraints. This **meta-data** (i.e., data about data) is stored in the so-called **system catalog**, which contains a description of the structure of each file, the type and storage format of each field, and the various constraints on the data (i.e., conditions that the data must satisfy).

2. **Insulation between Programs and Data :** DBMS provides a conceptual or logical view of the data to application programs, so that the underlying implementation may be changed without the programs being modified. (This is referred to as program-data independence.)

3. **Data abstraction:-** Abstraction is nothing but hiding complex details and exposing the essential things. Database developers hide internal complexity through several levels of abstraction.

4. **Data Sharing and Multi-user Transaction Processing:** As you learned about (or will) in the OS course, the simultaneous access of computer resources by multiple users/processes is a major source of complexity. The same is true for multi-user DBMS's.

Draw backs of file systems (Advantages of DBMS):-

1. Data redundancy and inconsistency:- Redundancy means duplication of data. Some data present in different files in the computer system may cause data redundancy. During updates if we forget the updation of a file, while retrieving the data we may get inconsistent results. This problem can be easily reduced in DBMS.

2. Difficulty in accessing data:- In conventional file systems to access the data, we have to write so many number of application programs (written in c, c++, java, etc) for storing retrieving and updating the data effectively.

3. Data isolation:-In file systems, the data will be stored in different file formats (.txt, .html, .war, etc). so we have to write application programs to retrieve data from each file format. But in databases the data is stored at one place, so we can access data easily.

4. Integrity constraints problems:- Constraint means defining a condition on data. Handling the constraint at application programming level is very difficult. Maintaining integrity constraints in DBMS is very easy.

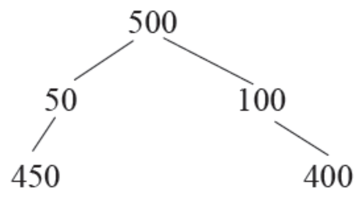
Ex:- to define a condition that marks of a student must be less than 100. We have to write application program in file system but in DBMS easily we can implement it.

5. Atomicity problems:- During the transactions, there may be the possibility of failing of computer system. In that case the transactions should not be happen, they should be rolled back. Such type of mechanism not there in file systems. This will be effectively done in DBMS.

Ex:-Suppose an amount of 100rs transferring from account A to B in an application program. If suppose power gone after deducting 100 from A. then that 100rs will not be added to B. so that will cause inconsistency. To avoid this, the transaction should be executed fully or should not be executed at all. i.e. no partial executions of the transactions. This is not maintained strictly in DBMS.

6. Concurrent-Access problems:- when multiple users wanted to update data simultaneously, we may get inconsistent data in file systems.

Ex:- consider an account A=500rs. If two customers want to withdraw 50rs and 100rs simultaneously. Two application programs for two with drawls will give the following.



We got 450rs and 400rs but the actual result is 350rs to maintain constraint results we got for DBMS.

7. Security problems:- All the data should not be accessed by all the users. One user must access his permitted data. Maintaining these restrictions in file system is very difficult. We can maintain better security using DBMS.

8. Representing Complex Relationships Among Data: A DBMS should have the capability to represent such relationships and to retrieve related data quickly.

9. **Providing Backup and Recovery:** The subsystem having this responsibility ensures that recovery is possible in the case of a system crash during execution of one or more transactions.

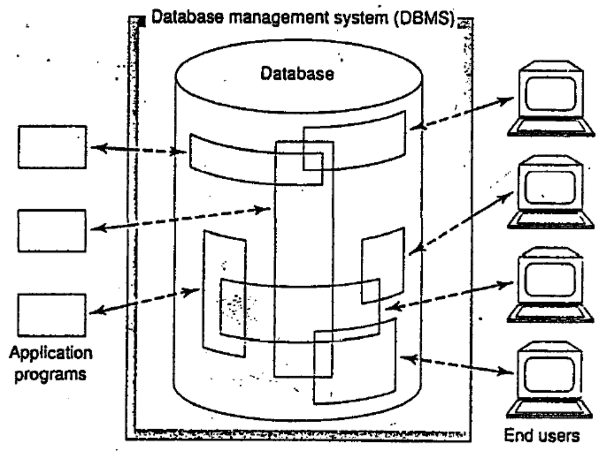


Fig. 1.4 Simplified picture of a database system

A Brief History of Database Applications

1) Early Database Applications:

   a. The Hierarchical and Network Models were introduced in mid 1960s and dominated during the seventies.
   b. A bulk of the worldwide database processing still occurs using these models.

2) Relational Model based Systems:
   a. Relational model was originally introduced in 1970, was heavily researched and experimented with in IBM Research and several universities.

3) Object-oriented and emerging applications:

   a. Object-Oriented Database Management Systems (OODBMSs) were introduced in late 1980s and early 1990s to cater to the need of complex data processing in CAD and other applications. Their use has not taken off much.

Many relational DBMSs have incorporated object database concepts, leading to a new category called object-relational DBMSs (ORDBMSs)

Database system architecture (3-tier architecture of DBMS)

Data abstraction:- Abstraction is nothing but hiding complex details and exposing the essential things. Database developers hide internal complexity through several levels of abstraction.

Data abstraction is used for following purposes:
1. To provide abstract view of data.
2. To hide complexity from user.
3. To simplify user interaction with DBMS.

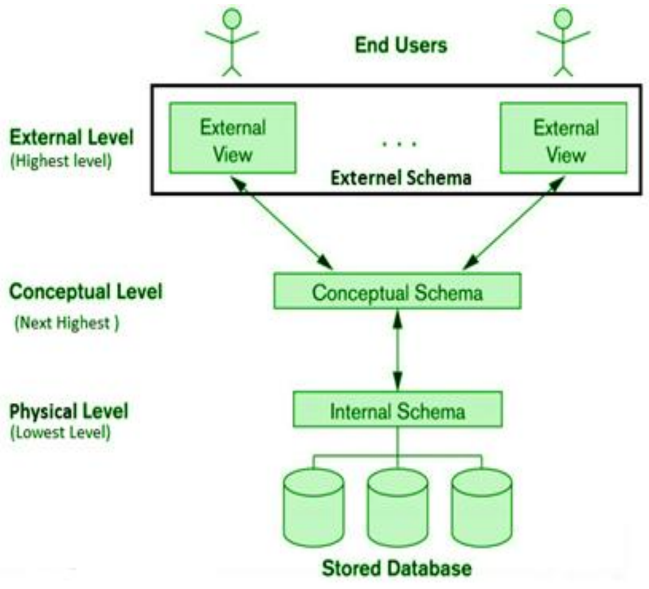
Levels of data abstraction:
There are three levels of data abstraction.

1. Physical level: It describes how a record (i.e. ., customer) is stored.
Features:
- a) Lowest level of abstraction.
- b) It describes how data are actually stored.
- c) It describes low-level complex data structures in detail.
- d) At this level, efficient algorithms to access data are defined.

2. Logical level: It describes what data stored in database, and the relationships between them are defined.
Features:
- a) It is next-higher level of abstraction. Here whole Database is divided into small simple structures.
- b) Users at this level need not be aware of the physical-level complexity used to implement the simple structures.
- c) Here the aim is ease of use.
- d) Generally, database administrators (DBAs) work at logical level of abstraction.

3. View level: Application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes.

Features:
a) It is the highest level of abstraction.
b) It describes only a part of the whole Database for particular group of users.
c) This view hides all complexity.
d) It exists only to simplify user interaction with system.
e) The system may provide many views for the whole system.



Data independence:- The ability of DBMS to modify its own schema definition at one level, without affecting the schema definition of next higher level is known as data independence.

There are two levels of data independence

Physical data independence:- It is ability of DBMS to modify the physical schema without causing any changes in the schema at logical level and at the view level. Generally modifications to the physical level are done to improve system performance.

Logical data independence:- It is the ability of DBMS to modify the logical schema without causing any changes in the application programs in the view level. Achieving logical data independency is much more difficult than physical data independence because the application programs highly dependent on the logical structure of data.

Instance:- The collection of information stored in the database at a particular point of time is called instance.

Schema:- The design of database at different levels is called schema.

Database users:-
For a small personal database one person defines, constructs and manipulates the database and there is no sharing. In large organizations many people involved in design, use and maintenance of a large database with hundred of users.
We have two kinds of database users
1. **Actors on the scene**
2. **Workers behind the scene**

Actors on the scene:- These people involves the day-to-day use of a large database.
We have four categories:-
1. Database administrators (DBA):- The responsibility of
DBA is to administrate the resource of databases. He is responsible for authorizing access the database. He will coordinate and monitor the database and acquire software and hardware resources needed. He is responsible for maintaining database security.
2. Database designers:- These are responsible for identifying the data is to be stored in databases and choosing appropriate structures to represent and store this data. These are responsible to communicate database users to know their requirements.
3. End uses:- These are the people who need access to database for querying, updating and generating reports. The database exists primarily for their use only. There are four categories of end users

- a) Casual end user
- b) Naive or parametric end user
- c) Sophisticated end user
- d) Standalone end user

4. System analyst and application programmers:- system analyst determine the requirements or specifications of end users especially naive or parametric end users. Application programmers implement these specifications as programs. Then they test, debug, document and update these programs.

2. Workers behind the scene:- These are the people who are not interested in the database content itself.
There are three categories:-

1. Database system designers and implementers:- These people will design and implement the DBMS modules and interfaces as a software package. These will implement the modules including the implementation of catalog, query processing, interface processing, buffering data, controlling concurrency, recovery and security.

2. Tool developers:- These people will implement the tools that facilitate database modeling and design, database system design, and improved performance. Tools are optional packages that are often purchased separately.

3. Operators and maintenance personnel:- These are responsible for actual running and maintenance of the hardware and software environment for the database system.

Applications of DBMS:-

1. Banking:- In banking systems DBMS is used for storing customer information, tracking day-to-day credit and debit transactions and generating bank statements etc……

2. Online shopping:-Online websites like amazon, flip kart, use DBMS to store product information, addressing and preferences credit details and provide us for the list of products based on our query.

3. Airline:- In airline reservation system, DBMS is used to keep records of flights arrival, departure and delay status.

4. Manufacturing:-In manufacturing, DBMS is used to keep track of records of all the details about the products like quantity, bills, purchase, supply management etc.

5. Human resource management:- In big organizations there are many workers working. Human resource management keep records of each employee's salary, tax and work through DBMS.

6. Telecom:-In telecom there is a need of DBMS to keep track of information regarding calls mode, network usage, customer details, etc

7. Education system:-Database systems are used frequently in schools and colleges to store and retrieve the data regarding student details, staff details, course details, exam details, fee details, etc
8. Railways:- Database is required in railways to keep record of ticket booking, train departure and arrival status and keep track of trains that are getting late etc.

9. Library management system:-These are thousands of books in a library and it is very difficult to keep record of all the books information"s in a register. Using DBMS we can maintain all the information related issue dates, names of books, author, availability, of the book etc.

10. Social media:-Daily thousands, millions of users signed up for the social media sites like face book, twitter, etc. All the information of these users are stored and maintained by using databases only.

11. Military:- Military keeps the records of millions of solders information and millions of files using DBMS keeping the data safe and secure.

Data Models

The Evolution of Database systems are as follows:

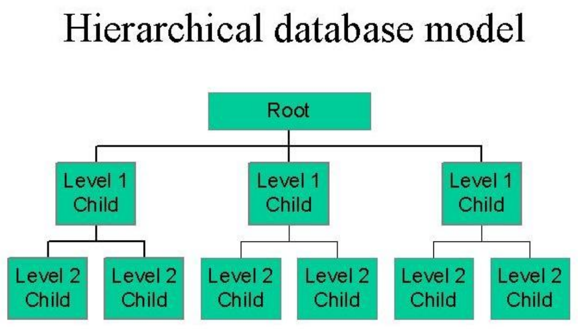
1. File Management System
2. Hierarchical database System
3. Network Database System
4. Relational Database System

File Management System:

The file management system also called as FMS in short is one in which all data is stored on a single large file. The main disadvantage in this system is searching a record or data takes a long time. This lead to the introduction of the concept of indexing in this system. Then also the FMS system had lot of drawbacks to name a few like updating or modifications to the data cannot be handled easily, sorting the records took long time and so on. All these drawbacks led to the introduction of the Hierarchical Database System.

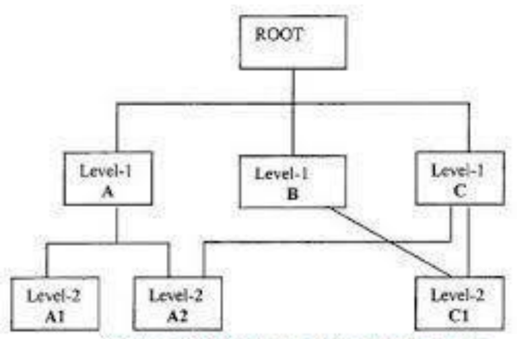
Hierarchical Database System:

The previous system FMS drawback of accessing records and sorting records which took a long time was removed in this by the introduction of parent-child relationship between records in database. The origin of the data is called the root from which several branches have data at different levels and the last level is called the leaf. The main drawback in this was if there is any modification or addition made to the structure then the whole structure needed alteration which made the task a tedious one. In order to avoid this next system took its origin which is called as the Network Database System.



Network Database System:

In this the main concept of many-many relationships got introduced. But this also followed the same technology of pointers to define relationships with a difference in this made in the introduction if grouping of data items as sets.



Network Database Model Diagram

Relational Database System:

In order to overcome all the drawbacks of the previous systems, the Relational Database System got introduced in which data get organized as tables and each record forms a row with many fields or attributes in it. Relationships between tables are also formed in this system.

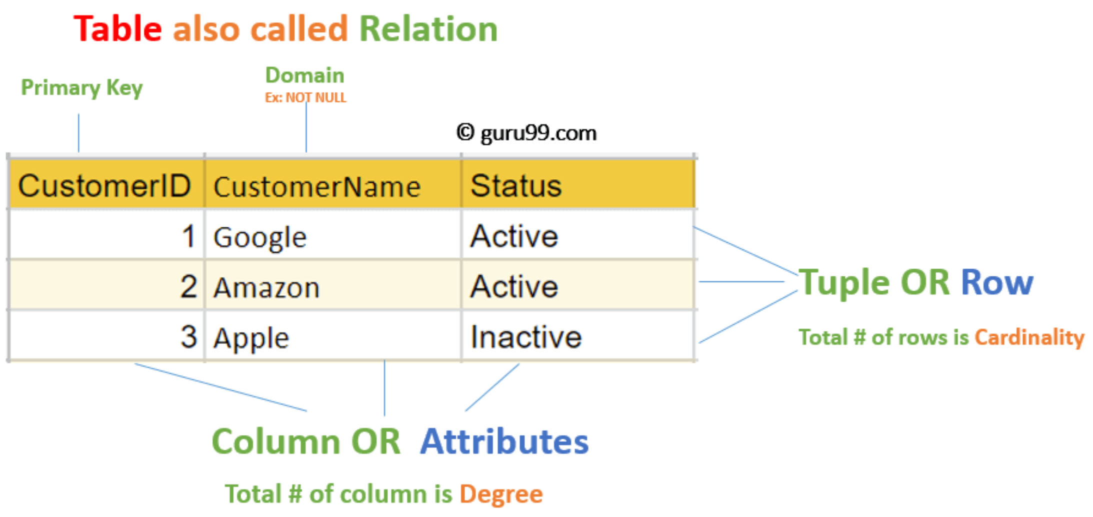


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
© guru99.com
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Mapping

Mapping in DBMS is the no. of values of one entity connected to no. of values of another entity.

- A mapping constraint is a data constraint that expresses the number of entities to which another entity can be related via a relationship set.
- It is most useful in describing the relationship sets that involve more than two entity sets.
- For binary relationship set R on an entity set A and B, there are four possible mapping cardinalities. These are as follows:
    1. One to one (1:1)
    2. One to many (1:M)
    3. Many to one (M:1)
    4. Many to many (M:M)

Database Administrator (DBA)

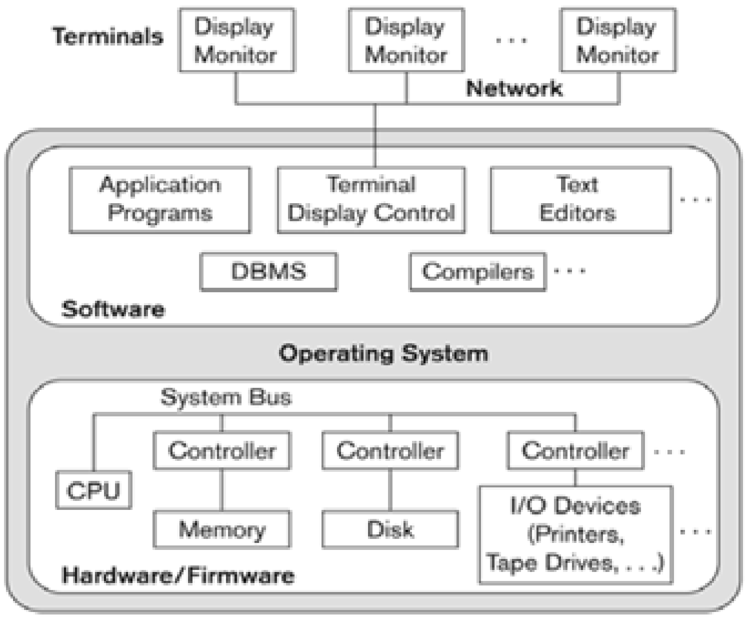
A database administrator (DBA) is a specialized computer systems administrator who maintains a successful database environment by directing or performing all related activities to keep the data secure. The top responsibility of a DBA professional is to maintain data integrity.

Responsibilities of DBA (Data Base Administrator):-

1. Creates and maintain all databases required for development and testing.
2. Performs ongoing tuning of the database instances.
3. Install new versions of the oracle DBMS and its tools and any other tools that access the oracle database.
4. Planes and implements backup and recovery the oracle database.
5. Controls migrations of programs, database changes, reference data changes through the development life cycle.
6. Implement and enforces security for all the oracle databases.
7. Performs databases reorganizations to assist performance.
8. Evaluates releases of oracle and its tools.
9. Provides technical support to application development team.
10. Enforces and maintains database constraints to ensure integrity of the database.
11. Administrators all database objects, including tables, views, indexes, clusters, packages, and procedures.
12. Troubleshoots with problem regarding the databases, application and development tools.
13. Create new databases users as required.
14. Manage sharing of resources amongst applications.
15. DBA has ultimate responsibility for the physical database design.

Client/Server Architecture

Client-server architecture, architecture of a computer network in which many clients (remote processors) request and receive service from a centralized server (host computer). Client computers provide an interface to allow a computer user to request services of the server and to display the results the server returns.



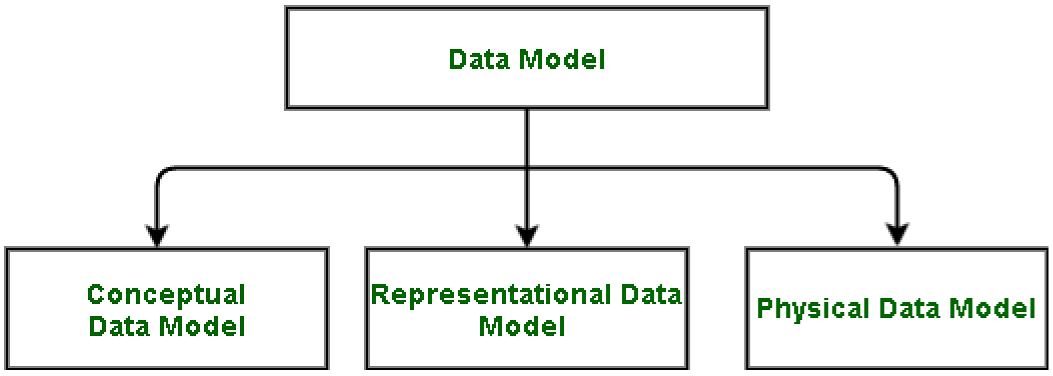
<u>Important questions from old question papers</u>

1. Compare and contrast various Data Models.
2. Demonstrate data abstraction implementation in DBMS. (Or)
   Explain the architecture of DBMS with a neat sketch.
3. List out the functionalities of DBA.
4. Define data independence? How do you implement data independence in DBMS? Explain
5. Differentiate File systems from DBMS.
6. Write any three data base applications with their functionalities.

Data Models

A **Data Model** in Database Management System (DBMS), is the concept of tools that are developed to summarize the description of the database.

It is classified into 3 types:



1.Conceptual Model

2. Representation Model

3.Physical Model

1. **Conceptual Data Model:**

Conceptual data model, describes the database at a very high level and is useful to understand the needs or requirements of the database. It is this model, that is used in the requirement gathering process i.e., before the Database Designers start making a particular database. In this model we identify what are the Entities and attributes and their relationships between the entities by ER Model.

2. **Representational Data Model:**

This type of data model is used to represent only the logical part of the database and does not represent the physical structure of the databases. In this we different types of data models like bellow.

1. Hierarchical Model
2. Network Model
3. Entity-Relationship Model
4. Relational Model
5. Object-Oriented Data Model

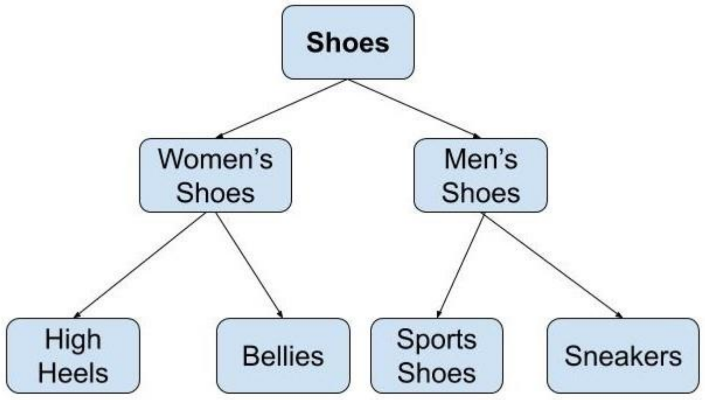
3. Physical Data Model:

Ultimately, all data in a database is stored physically on a secondary storage device such as discs and tapes. This is stored in the form of files, records and certain other data structures.

2.Representation Models:

1. Hierarchical Model

Hierarchical Model was the first DBMS model. This model organizes the data in the hierarchical tree structure. The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node. This model easily represents some of the real-world relationships like food recipes, sitemap of a website etc. ***Example:*** We can represent the relationship between the shoes present on a shopping website in the following way:



Hierarchical Model

Features of a Hierarchical Model :

1. ***One-to-many relationship:*** The data here is organized in a tree-like structure where the one-to-many relationship is between the data types. Also, there can be only one path from parent to any node. ***Example:*** In the above example, if we want to go to the node *sneakers* we only have one path to reach there i.e through men's shoes node.

2. ***Parent-Child Relationship:*** Each child node has a parent node but a parent node can have more than one child node. Multiple parents are not allowed.

3. ***Deletion Problem:*** If a parent node is deleted then the child node is automatically deleted.

4. ***Pointers:*** Pointers are used to link the parent node with the child node and are used to navigate between the stored data. *Example:* In the above example the '*shoes*' node points to the two other nodes '*women shoes*' node and '*men's shoes*' node.

Advantages of Hierarchical Model:

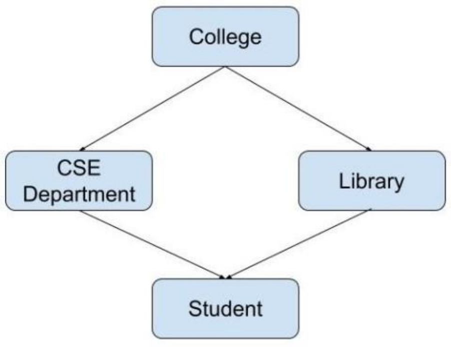
- It is very simple and fast to traverse through a tree-like structure.
- Any change in the parent node is automatically reflected in the child node so, the integrity of data is maintained.

Disadvantages of Hierarchical Model

- Complex relationships are not supported.
- As it does **not support more than one parent** of the child node so if we have some complex relationship where a child node needs to have two parent node then that can't be represented using this model.
- If **a parent node is deleted** then the child node is automatically deleted.

2. Network Model

This model is an extension of the hierarchical model. It was the most popular model before the relational model. This model is the same as the hierarchical model, the only difference is that a record can have more than one parent. It replaces the hierarchical tree with a graph. ***Example:*** In the example below, we can see that node **student has two parents** i.e. CSE Department and Library. This was earlier not possible in the hierarchical model.



Network Model

Features of a Network Model

1. ***Ability to Merge more Relationships:*** In this model, as there are more relationships so data is more related. This model has the ability to manage **one-to-one relationships** as well as **many-to-many relationships**.
2. ***Many paths:*** As there are more relationships so there can be more than one path to the same record. This makes data access fast and simple.
3. ***Circular Linked List:*** The operations on the network model are done with the help of the circular linked list. The current position is maintained with the help of a program and this position navigates through the records according to the relationship.

Advantages of Network Model

- The data can be accessed faster as compared to the hierarchical model. This is because the data is more related in the network model and there can be more than one path to reach a particular node. So the data can be accessed in many ways.
- As there is a parent-child relationship so data integrity is present. Any change in parent record is reflected in the child record.

Disadvantages of Network Model

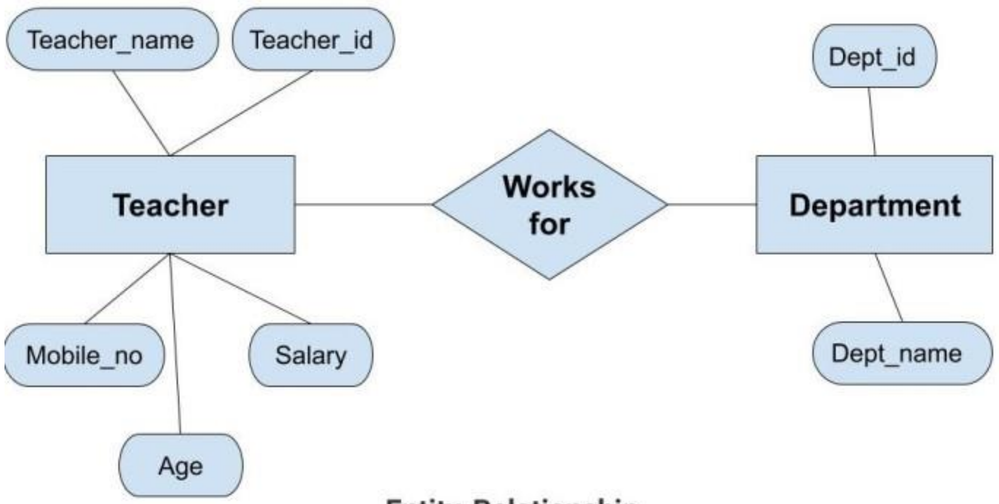
- As more and more relationships need to be handled the system might get complex. So, a user must be having detailed knowledge of the model to work with the model.
- Any change like updation, deletion, insertion is very complex.

3. Entity-Relationship Model

Entity-Relationship Model or simply ER Model is a high-level data model diagram. In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand. It is also very easy for the developers to understand the system by just looking at the ER diagram. We use the ER diagram as a visual tool to represent an ER Model. ER diagram has the following three components:

- ***Entities:*** Entity is a real-world thing. It can be a person, place, or even a concept. *Example:* Teachers, Students, Course, Building, Department, etc are some of the entities of a School Management System.
- ***Attributes:*** An entity contains a real-world property called attribute. This is the characteristics of that attribute. *Example:* The entity teacher has the property like teacher id, salary, age, etc.
- ***Relationship:*** Relationship tells how two attributes are related. *Example:* Teacher works for a department.

Example:



Entity-Relationship Model

In the above diagram, the entities are Teacher and Department. The attributes of ***Teacher*** entity are Teacher_Name, Teacher_id, Age, Salary, Mobile_Number. The attributes of entity ***Department*** entity are Dept_id, Dept_name. The two entities are connected using the relationship. Here, each teacher works for a department.

Features of ER Model

- ***Graphical Representation for Better Understanding:*** It is very easy and simple to understand so it can be used by the developers to communicate with the stakeholders.
- ***ER Diagram:*** ER diagram is used as a visual tool for representing the model.
- ***Database Design:*** This model helps the database designers to build the database and is widely used in database design.

Advantages of ER Model

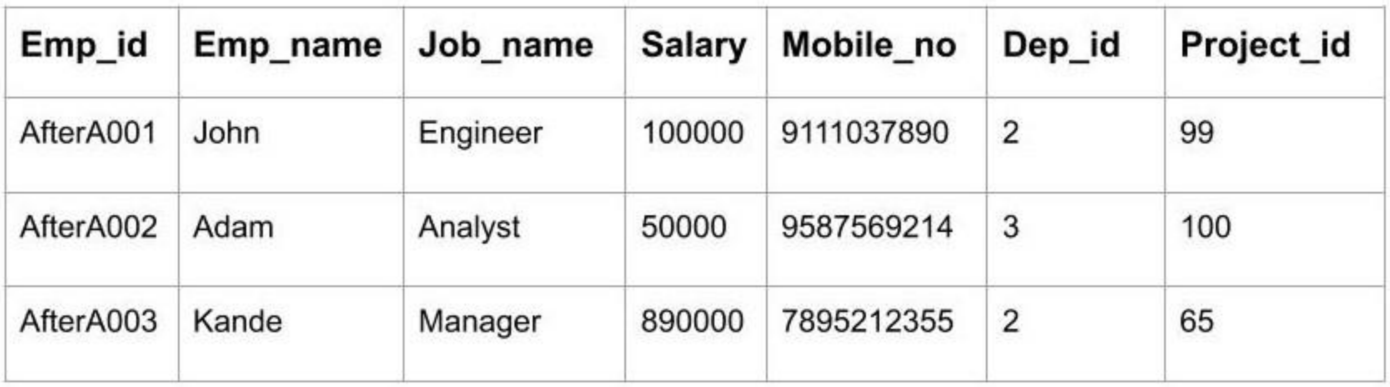
- ***Simple:*** Conceptually ER Model is very easy to build. If we know the relationship between the attributes and the entities we can easily build the ER Diagram for the model.
- ***Effective Communication Tool***: This model is used widely by the database designers for communicating their ideas.
- ***Easy Conversion to any Model***: This model maps well to the relational model and can be easily converted relational model by converting the ER model to the table. This model can also be converted to any other model like network model, hierarchical model etc.

Disadvantages of ER Model

- ***No industry standard for notation:*** There is no industry standard for developing an ER model. So one developer might use notations which are not understood by other developers.
- ***Hidden information:*** Some information might be lost or hidden in the ER model. As it is a high-level view so there are chances that some details of information might be hidden.

4. Relational Model

Relational Model is the most widely used model. In this model, the data is maintained in the form of a two-dimensional table. All the information is stored in the form of row and columns. The basic structure of a relational model is tables. So, the tables are also called *relations* in the relational model. ***Example:*** In this example, we have an Employee table.



Emp_id	Emp_name	Job_name	Salary	Mobile_no	Dep_id	Project_id
AfterA001	John	Engineer	100000	9111037890	2	99
AfterA002	Adam	Analyst	50000	9587569214	3	100
AfterA003	Kande	Manager	890000	7895212355	2	65

EMPLOYEE TABLE

Features of Relational Model

- ***Tuples / Rows / Records***: Each row in the table is called tuple. A row contains all the information about any instance of the object. In the above example, each row has all the information about any specific individual like the first row has information about John.
- ***Attribute / field /Properties / columns :*** Attributes are the property which defines the table or relation. The values of the attribute should be from the same domain. In the above example, we have different attributes of the *employee* like Salary, Mobile_no, etc.

Advantages of Relational Model

- ***Simple:*** This model is more simple as compared to the network and hierarchical model.
- ***Scalable:*** This model can be easily scaled as we can add as many rows and columns we want.
- ***Structural Independence:*** We can make changes in database structure without changing the way to access the data. When we can make changes to the database structure without affecting the capability to DBMS to access the data we can say that structural independence has been achieved.

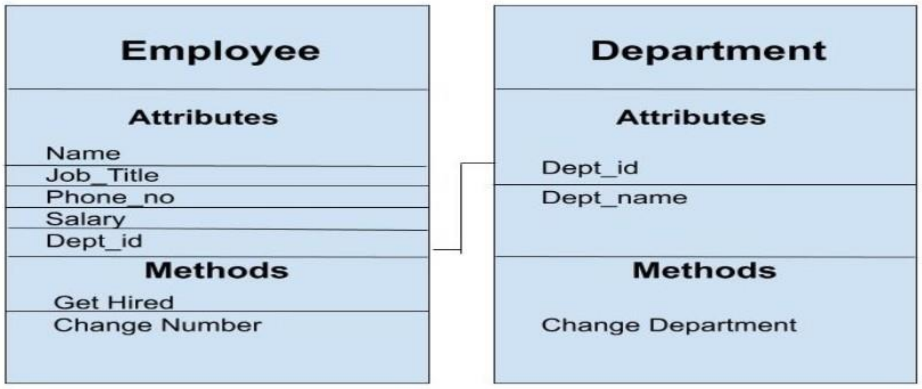
Disadvantages of Relational Model

- ***Hardware Overheads:*** For hiding the complexities and making things easier for the user this model requires more powerful hardware computers and data storage devices.
- ***Bad Design:*** As the relational model is very easy to design and use. So the users don't need to know how the data is stored in order to access it. This ease of design can lead to the development of a poor database which would slow down if the database grows.

But all these disadvantages are minor as compared to the advantages of the relational model. These problems can be avoided with the help of proper implementation and organization.

5. Object-Oriented Data Model

The real-world problems are more closely represented through the object-oriented data model. In this model, both the data and relationship are present in a single structure known as an object. We can store audio, video, images, etc in the database which was not possible in the relational model(although you can store audio and video in relational database, it is adviced not to store in the relational database). In this model, two are more objects are connected through links. We use this link to relate one object to other objects. This can be understood by the example given below.



Object_Oriented_Model

In the above example, we have two objects Employee and Department. All the data and relationships of each object are contained as a single unit. The attributes like Name, Job_title of the employee and the methods which will be performed by that object are stored as a single object. The two objects are connected through a common attribute i.e the Department_id and the communication between these two will be done with the help of this common id.

This is all about the various data model of DBMS. Hope you learned something new today.

Database Users :

Database users are the one who really use and take the benefits of database. There will be different types of users depending on their need and way of accessing the database.

Application Programmers – They are the developers who interact with the database by means of DML queries. These DML queries are written in the application programs like C, C++, JAVA, Pascal etc. These queries are converted into object code to communicate with the database. For example, writing a C program to generate the report of employees who are working in particular department will involve a query to fetch the data from database. It will include a embedded SQL query in the C Program.

Sophisticated End Users – They are database developers, who write SQL queries to select/insert/delete/update data. They do not use any application or programs to request the database. They **directly interact with the database** by means of query language like SQL. These users will be scientists, engineers, analysts who thoroughly study SQL and DBMS to apply the concepts in their requirement. In short, we can say this category includes designers and developers of DBMS and SQL.

Specialized End Users – These are also sophisticated users, but they write special database application programs. They are the developers who develop the complex programs to the requirement.

Stand-alone End Users – These users will have stand –alone database for their personal use. These kinds of database will have readymade database packages which will have menus and graphical interfaces.

Native/ Parametric End Users – these are the users who use the existing application to interact with the database. For example, online library system, ticket booking systems, ATMs etc which has existing application and users use them to interact with the database to fulfill their requests.

Database Administrators(DBA)

A DBA has many responsibilities. A good performing database is in the hands of DBA.

1. Installing and upgrading the DBMS Servers: – DBA is responsible for installing a new DBMS server for the new projects. He is also responsible for upgrading these servers as there are new versions comes in the market or requirement. If there is any failure in upgradation of the existing servers, he should be able revert the new changes back to the older version, thus maintaining the DBMS working. He is also responsible for updating the service packs/ hot fixes/ patches to the DBMS servers.

2. Design and implementation: – Designing the database and implementing is also DBA's responsibility. He should be able to decide proper memory management, file organizations, error handling, log maintenance etc for the database.

3. Performance tuning: – Since database is huge and it will have lots of tables, data, constraints and indices, there will be variations in the performance from time to time. Also, because of some designing issues or data growth, the database will not work as expected. It is responsibility of the DBA to tune the database performance. He is responsible to make sure all the queries and programs works in fraction of seconds.

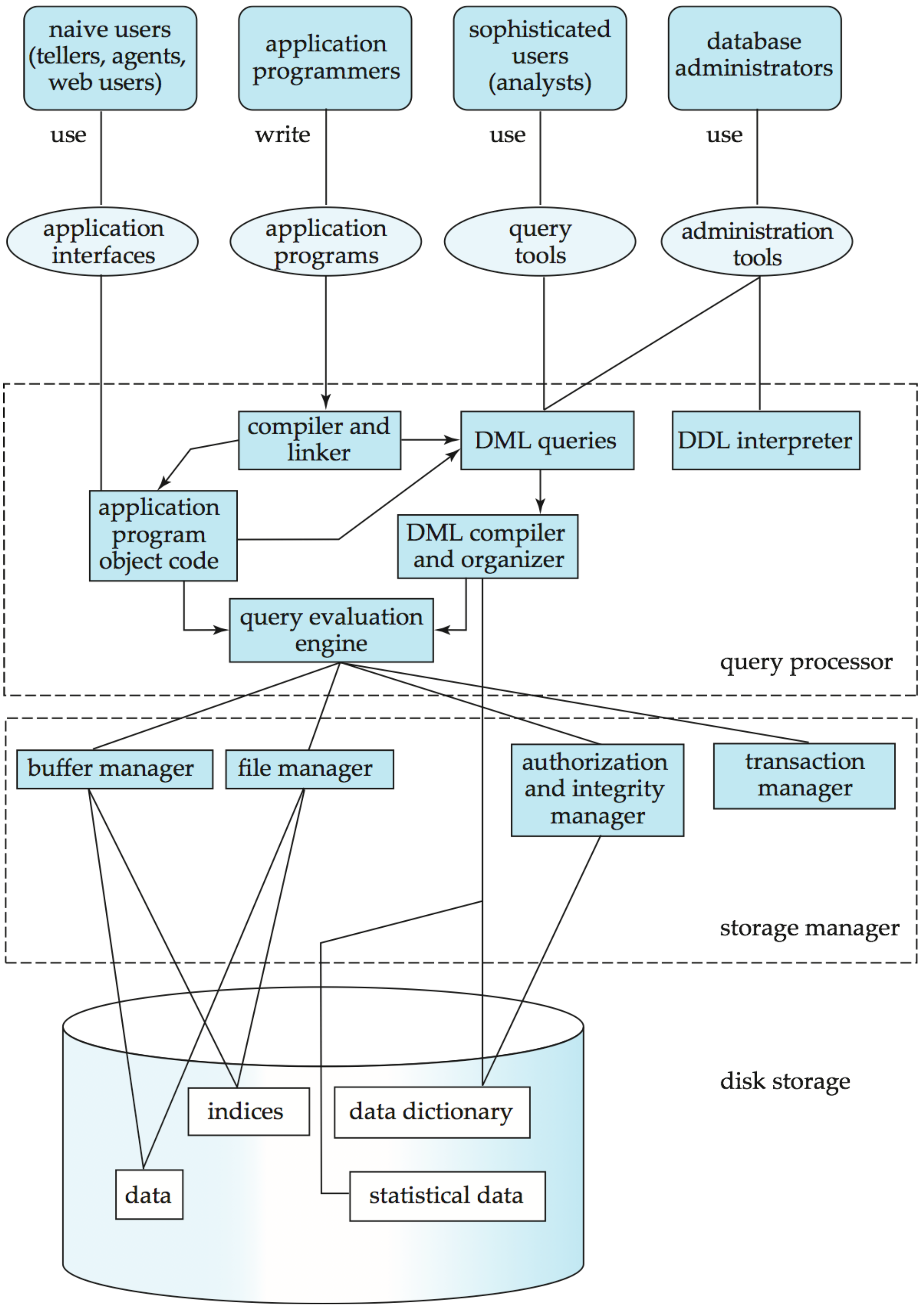
4. Migrate database servers: – Sometimes, users using oracle would like to shift to SQL server or Netezza. It is the responsibility of DBA to make sure that migration happens without any failure, and there is no data loss.

5. Backup and Recovery: – Proper backup and recovery programs needs to be developed by DBA and has to be maintained him. This is one of the main responsibilities of DBA. Data/objects should be backed up regularly so that if there is any crash, it should be recovered without much effort and data loss.

6. Security: – DBA is responsible for creating various database users and roles, and giving them different levels of access rights.

7. Documentation: – DBA should be properly documenting all his activities so that if he quits or any new DBA comes in, he should be able to understand the database without any effort. He should basically maintain all his installation, backup, recovery, security methods. He should keep various reports about database performance

System architecture:



Storage Manager:

- It is a program module which provides interface between the low level data stored in the database and the application programs and queries submitted to the system.
- It is responsible for interaction with the *File Manager.*
- It translates DML commands into low level file system commands.
- It is responsible for *storing, retrieving* and *updating* data in database.
- It includes the following components:
    a) **Authorization and Integrity Manager:** Tests the integrity constraints and authorization of users.
    b) **Transaction Manager:** Ensures the consistency of database despite of system failures by non-conflicting schemes.
    c) **File Manager:** Manages the allocation of space on disk storage and data structures used to represent data on disk.
    d) **Buffer Manager:** It is responsible for fetching data from disk storage to main memory.

The Storage Manager implements several data structures at physical level of database. They are:

- **Data Files:** Stores database.
- **Data Dictionary**: Stores metadata of stored database.
- **Indices:** Provides fast access to database. *Hashing is alternative to indexing.*

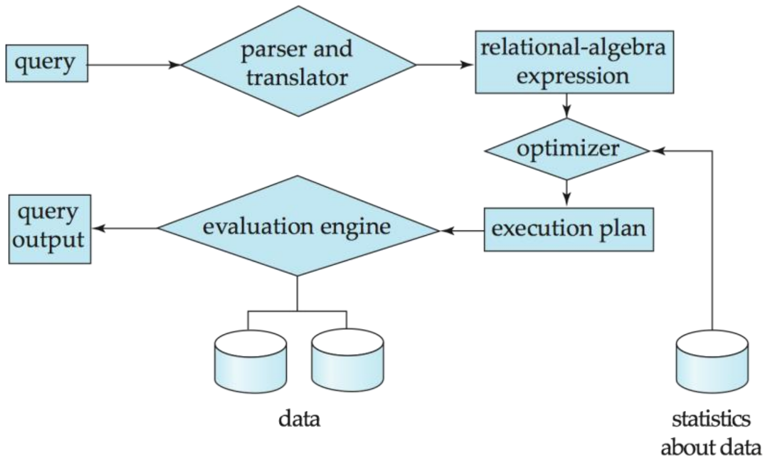
The Query Processor:

The Query Processor contains the following components:

i. **DDL Interpreter:** Interpreters DDL statements and records the definition into data dictionary.
ii. **DML Compiler:** Translates DML statements into an evaluation plan consisting of low level instructions that the query evaluation engine understands. It translates the query into a number of evaluation plans. DML Compiler performs query optimization i.e. selects the optimal evaluation plan for execution of query.
iii. **Query Evaluation Plan:** Executes the low level instructions generated by DML compiler.

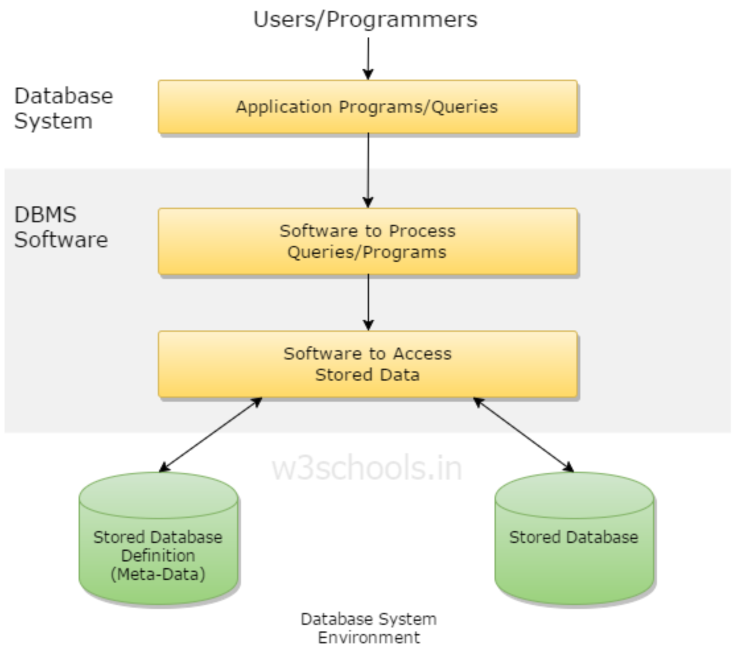
Query Processing:

1. Parsing and translation
2. Optimization
3. Evaluation



DATABASE ENVIRONMENT

One of the primary aims of a database is to supply users with an abstract view of data, hiding a certain element of how data is stored and manipulated. Therefore, the starting point for the design of a database should be an abstract and general description of the information needs of the organization that is to be represented in the database. And hence you will require an environment to store data and make it work as a database. In this chapter, you will learn about the database environment and its architecture.

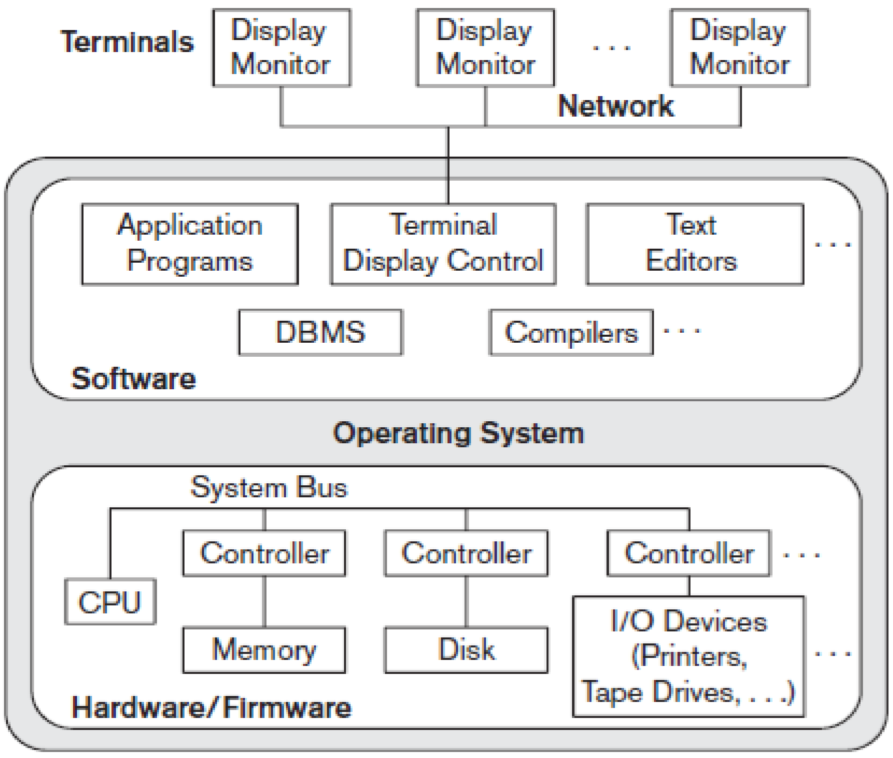

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
w3schools.in
Stored Database Definition (Meta-Data)
Stored Database
Database System Environment

A database environment is a collective system of components that comprise and regulates the group of data, management, and use of data, which consist of software, hardware, people, techniques of handling database, and the data also.
Here, the hardware in a database environment means the computers and computer peripherals that are being used to manage a database, and the software means the whole thing right from the operating system (OS) to the application programs that include database management software like M.S. Access or SQL Server. Again the people in a database environment include those people who administrate and use the system. The techniques are the rules, concepts, and instructions given to both the people and the software along with the data with the group of facts and information positioned within the database environment.

Centralized and Client/Server Architecture for DBMS

In Centralized Architecture, the mainframe computers are used for processing all system functions including User application Programs and User Interface Programs as well as DBMS functionalities. This is because in earlier days, most users accessed such systems via Computer Terminals, which can’t Process and they have only display capability. Therefore the Processing used to takeplace in these Computer Systems and the display information is sent to display terminals and these terminals are connected to mainframe computers via various kinds of Networks.

As the days pass by, we are now having PersonalComputers(PC’s) in the market. But still in the beginning Centralized Architecture for DBMS was used. Gradually the DBMS systems started to make use of Processing Power in the used side i.e Computers have come with Processing Power and in turn led to the use of Client/Server Architecture.



Client/Server Architecture:

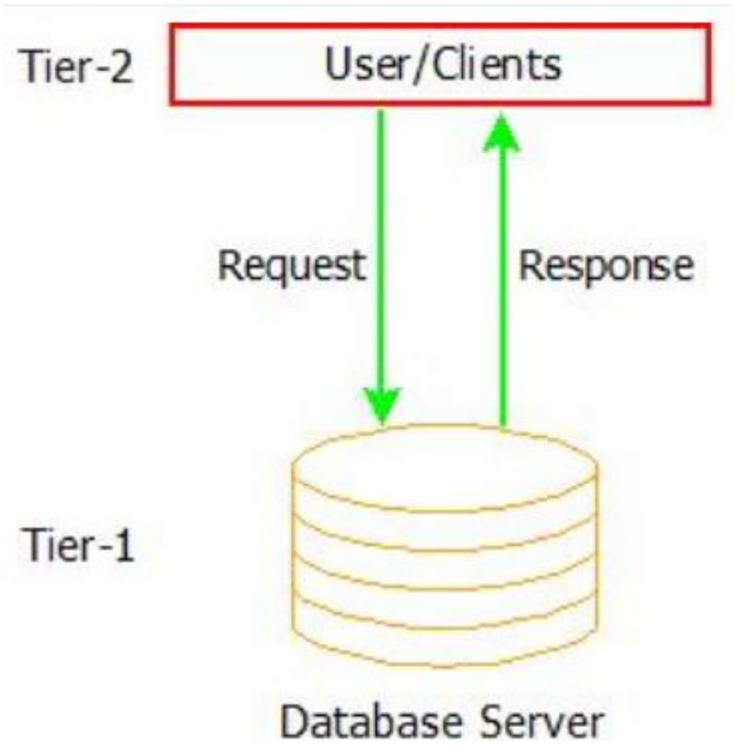
The concept of client/server architecture assumes an underlying framework that consists of many PCs as well as a smaller number of mainframe machines, connected via LANs and other types of computer networks.A client in this framework is typically a user machine that provides user interface capabilities and local processing. When a client requires access to additional functionality, such as database access which does not exist at that machine, it connects to a server that provides the needed functionality.

A Server is a system which contains both Hardware and Software which provides services to client Machines like file access, printing and database access.

1)Two Tier Client/Server Architecture for DBMS:

Here Two-tier means that our Architecture has two layers, which are client layer and Datalayer. In Client layer we have several Client machines which can have the access to the database server. The API present on the client machine will establish the connection between the machine and the Database server through JDBC somthing else.This is because Clients and Database Server may be at different different locations.Once this connection gets established,the Interface present on the client machine contains an Application Program on the back-side which contains a query. This query will be processed by the Database server and in turn the queried information will be sent to the client machine.

For example if we query the database to retrieve some information, the query will be Processed by Database server and that information will be sent to the client by Database server itself!!!



2) Three-Tier client/server Architecture for DBMS:

Here there is an additional layer which acts as an intermediate between Client layer and Datalayer called Business logic layer. Business logic layer is the layer where the Application Programs are processed. Here the Application Programs are processed in the Application server itself, which makes it different from Two-tier Architecture where queries are processed in the database server.Simply the Client machines will contact Application Server which in turn processes our Application Programs and fetches the Required Data from Database and then sends this Information back to the client machine in the suitable format only.