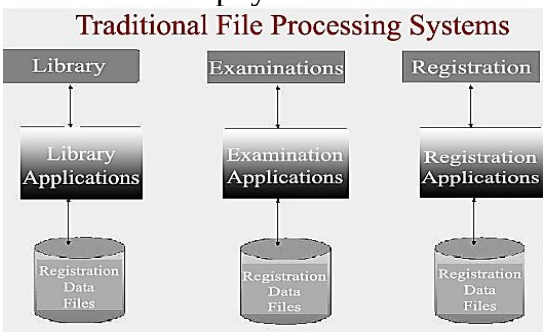
- Definition
- Limitations of traditional file system
- Advantages of DBMS
- Components of DBMS
- Database Users
- Database Architecture - 2 tier and 3 tier architecture
- Schemas and instances
- Database languages
- Database Structure
- Types of data models- Relational, Network, Hierarchical, Distributed, Hybrid

****

In earlier days, data was stored manually, using pen and paper but after computer was discovered, the same task could be done by using files. A Manual file system was used to maintain the records and files. A data was stored in files and easy to find any information.

In traditional file system, each file is independent of other file and data in the different file can be integrated by writing an individual program for each application. Data is stored in Different Files in forms of Records. A groups of records stored in separate files called **file processing system.** A **file system** is a method of storing and organizing a set of computer files and data. Files are easily accessed by using File system. It maintains the physical location of the files on storage device.



Limitations of traditional file processing system

◆ Data redundancy

Data Redundancy means repetition of data i.e. multiple copies of the same data. Data Redundancy occurs when the same data are stored unnecessarily at different files. The duplication of data creates many problems such as requires additional storage space, time, money and requires additional efforts to keep all files up to date.

Example: A student is having record in college library and admission department. Name, roll number and class will be same in both departments. Also these departments are not dependent on each other. So it create lots of duplication of student data. Change his name or class in both departments otherwise it will create inconsistency problem.

◆ Data inconsistency

Data inconsistency occurs due to the same data is stored in multiple files. The changes made by user in one application doesn't update the changes in the other application. Data inconsistency mean that different files may contain different information of a particular object.

Example: If you change student name in library then his name should be changed in admission departments. If data isn't changed then it may create a risk of data inconsistent.

◆ Difficulty in accessing data

It is difficult to retrieve specific type of data from the files, then programmer have to create separate applications for that data. But to access a various type of data then we required to create multiple programs. So, convenient and efficient data accessing is almost impossible.

Example: A college admin wants student information like his Roll no., Name, Marks and Class. Program is written already, but if admin wants those students' records who secured more than 80 percent. Now, he wants to create a different program for extract data from file. It is difficult to access data.

◆ **Data isolation**

In file processing system data are scattered in various files. These files may be in different formats. If you want to extract data from two file, it is very difficult to write new application programs to retrieve the appropriate data.

Example: In Bank Suppose the

- Saving Account file has address fields: Add line1, City, State, Pin code.
- Current account file has address fields: House No, Street No, City, State, Pin code.

Both files have different or same address field. If Bank Manager wants list of customers who living in a particular area. We retrieve list of all the customer's data from both files. But data in different format.

◆ **Data Atomicity**

Atomicity is required to save the transaction or operation either completes all (100%) or none (0%) canceled at all. If system failed at any time, the data should be restored to the consistent state that existed before to the failure.

Example: Suppose a user wants to transfer money form **account A** to **account B**. This process consist of two steps:

- Deduct the money form **account A.**
- Add the money to **account B.**

If a system failure occurs during the execution of fund transfer operation, it may possible that, amount is debited from **account A**, but is not credited to **account B**. This situation can make data inconsistent. It is difficult to maintain atomicity in file processing system due to data redundancy, data isolation etc.

◆ **Poor Data Security**

There is no security provided in file processing system to secure the data from unauthorized user access. There is very less security in File Processing System as anyone can easily modify and change the data stored in the files. All the users must have some restriction of accessing data up to a level.

Example: If a student can access his data in the college library then he can easily change books issued date. Also he can change his fine to zero.

◆ **Poor Data Integrity**

Integrity means reliability and accuracy of data. The data values stored in the database must satisfy certain types of consistency constraints. Integrity problem arises when the database fails to satisfy certain integrity conditions. In some cases constraints are applied on the data before inserting into database. There is no direct checking constraints automatically in the file system. It is very difficult to apply constraints on files.

Example: the phone number cannot be greater than 10 digits, bank balance should not go below 1000 etc. The actual problem arises when we would like to include new such conditions with the existing database. It is hard to make those changes.

◆ **Data Sharing**

Data are scattered in various files with different formats. Due to data isolation, it is difficult to share data among different applications. A file-based processing systems do not provide data sharing among multiple users on the network.

◆ **Concurrent access anomalies (Allow multiple users to access database)**

To increase performance of the system and faster response to applications, if multiple users are updating the same data simultaneously. Result of concurrent execution is data in inconsistent state. It is very difficult to handle simultaneous update of data in file processing. In file processing system handled this situation through application program.

Example: Bank **account A** containing Rs.500. If two customers withdraw funds Rs.50 and Rs.100 respectively from account A at the same time, the result of the concurrent executions may leave the account in an incorrect (inconsistent) state. To guard this, the system must supervise concurrent execution. But supervision is difficult to provide because data may be accessed by many different application programs that have not been coordinated. It is difficult to handle in file processing system

due to the fact of data isolation, redundancy etc.

◆ **Data Dependence**

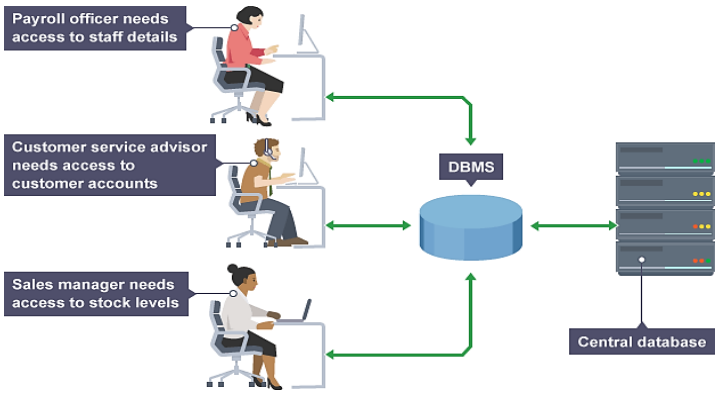
- In file-based processing systems, structure of data files is attached with application program. If format of file is changed, then application program needs to be changed.
- The physical structure of data files or records are defined in the application program code. It is difficult to change the structure of data files or records.
- If you want to change the structure of data file (format of file), then you have to modify the application program.

Database

A **DataBase Management System** (DBMS) is a collection of interrelated data and programs to access, update and manage stored data.

The main feature of data in a database are:

1. Saves Storage Space and Cost
2. High Level of Security
3. Data abstraction
4. Multi-user Access
5. Backup and recovery of data
6. Large Database Maintenance



Example: A company have different departments, as well as customers, who each need to see different kinds of data. Each employee in the company will have different levels of access to the database with their own customized front-end application.

The primary goal of a DBMS provides a way to store and retrieve a data conveniently. Database is shared, persistence, valid, secure, consistent, non-redundant and independent.

Advantages of DBMS

◆ **Avoid Data Redundancy**

- Data Redundancy occurs due to same data are stored unnecessarily at different places.
- Data Redundancy is eliminated in DBMS because all files are stored at a single database (centralized database) and there is no chance of data duplication.
- **Example:** A library or Admission dept. contains duplicate values student record, but when they are converted into a single database, all the duplicate values are removed.

◆ **Data Consistency**

- Data consistency means if you want to update data in any files then all the files should not be updated again.
- In DBMS, data is stored in a single database (centralized database) so data becomes more consistent. Also updated data are available to all the users immediately.

◆ **Sharing of Data**

- DBMS system allows users and applications to share data with multiple applications and users. The database administrator manage and gives rights to users to access the data.
- Many users can be authorized to access the same piece of information simultaneously.
- The remote users can also share same data. Similarly, the data of same database can be shared between different application programs.

◆ **Data integration**

- Data is stored in different tables but different tables that are linked to each other.
- Many users feed entries in these tables so it is important to maintain data items and association between data items. This makes easy to retrieve and update data.
- DBMS allows data integrity that makes it easy to decrease data duplication. Data integration reduces redundancy as well as data inconsistency.

- **Data Security**
  - To protect a data from unauthorized access. Data in DBMS should be kept secure and safe from unauthorized modifications.
  - DBMS ensures that only authorized users have access to data and there is a mechanism to define access privileges.
  - DBMS always keep database secure and theft free.
- **Backup and Recovery**
  - In traditional file processing system, a user needs to backup data after a regular interval of time that wastes lots of time and resources. If the volume of data is large then this process may take a very long time.
  - DBMS have a backup-and-recovery feature built within them that automatically backs-up all important data, and restores it when needed.
  - **Example**: if a system fails in the middle of any process then DBMS stores the values of that state in which database were before query execution.
- **Data Integrity**
  - Integrity constraints rules can be applied on database. Before inserting data into database, DBMS can check data is correct. i.e. accepts only correct data.
  - The constraints may be applied on single or multiple column.
  - Database integrity provides the validity and consistency of stored data.
  - DBMS maintain the integrity of data, provide capabilities for defining and enforcing the constraints.
  - **Example:** If we want to enter salary of an employee within the range 10000 to 40000, using CHECK (salary>10000 AND salary <40000) constraint. When salary is entered, it will automatically check for the range specified.
- **Data Atomicity**
  - DBMS makes sure either the transaction is fully complete or rolled back to the previous committed state. It doesn't allow the system to be in a partially committed state.
  - **Example:** In railway reservation system, if user has completed the process of ticket reservation then his record will be stored and amount of money will be deducted from his account otherwise no amount will be deducted and if deducted it will be given back.
- **Concurrency Control**
  - DBMS provide access to multiple users to access the database at the same time. It may create concurrency.
  - DBMS has the power to control concurrency, so that no transactions are lost.
  - It has its own mechanism to have concurrency accesses and avoid any of information no loss of integrity
- **Data Independence**
  - The separation of data structure from the application program used to access it is known as data independence.
  - In a DBMS, database and application program are maintained separately from each other. DBMS acting as a mediator between them. You can easily change the database structure without affecting the application program.

Disadvantages of DBMS

- **Cost:** DBMS requires high initial investment for hardware, software and trained staff.
- **Complexity:** A DBMS fulfill lots of requirement and it solves many problems related to database. But all these functionality has made DBMS extremely complex software.
- **Technical staff requirement:** A team of technical staff is required who understand DBMS
- **Database Failure:** all files are stored in single database, so chances of database failure. Any accidental failure of component may cause loss of valuable data.
- **Cost of Data Conversion:** Data conversion may require at any time and organization.

- **Currency Maintenance:** DBMS requires to updates itself daily.
- **Size:** DBMS becomes big software due to its functionalities so it requires lots of space and memory to run its application efficiently.

Database System Applications

1. Banking – For customer information, accounts, loans and banking transactions.
2. Airlines – For reservation and schedule information.
3. Universities – For student information, course registrations, and grades.
4. Credit Card Transactions – For purchases and generation of monthly reports
5. Telecommunication – For keeping records of calls, generating monthly bills, maintaining balances on prepaid calling cards and storing information about communication networks.
6. Finance – For storing information about holdings, sales, and purchases of stocks and bonds.
7. Sales – For customer, product, and purchase information.
8. Manufacturing – For management of supply chain and for tracking production of items in factories, inventories of items in warehouses/stores, and orders for items.
9. Human Resources – For information about employees, salaries, payroll taxes and benefits, and generation of paychecks.

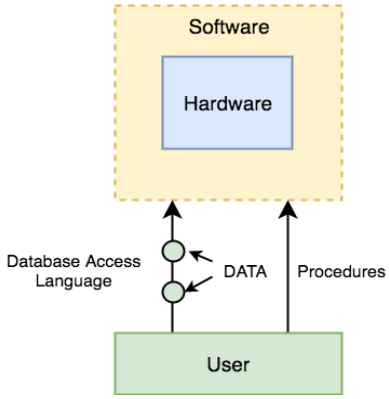
Parameters	DBMS	File System
Data Redundancy	Repetition of data can be reduced.	Repetition of data cannot be reduced.
Data Inconsistency	Inconsistency of data can be avoided.	Inconsistency of data cannot be avoided.
Sharing of Data	Data can be shared easily.	Data cannot be shared easily.
Concurrency Control	Data can be accessed by multiple users at same point of time.	Problem in concurrency control.
Security	It restricts unauthorized access.	Lack of security.
Data Recovery	It provides backup and restores database to its original state after database failure.	It does not provide back up and data recovery. Data loss if operation is failed.

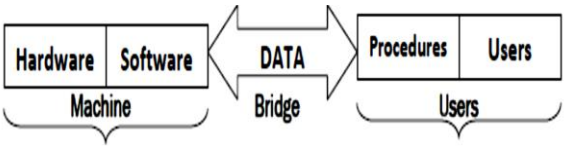
Components of DBMS

This DBMS component is used for keeping and storing the data in the database. The major components of database management system are:

➢ Hardware

The hardware component mainly contains processor, main memory and secondary storage device. The processor and main memory are used for execution of DBMS. Secondary storage is used for store data. i.e. database physical exist in the hard disk. Generally database systems require a minimum amount of disk. When DBMS system allows multi user that time large amount of memory and network devices is needed. It is impossible to implement the DBMS without the hardware devices. **Example**-database server, high data processing systems, router.





➢ Software

Software is an interface between users and physical database. It is a set of programs used to handle, control and manage the overall database. The DBMS software is capable of understanding the Database Access Language and interpret it into actual database commands to execute them on the Database. Performing these operations such as insertion, deletion and updation, we can use the Query Languages like SQL or application software's such as .NET, JAVA, Python.

➢ **Data**

The data is a bridge between the machine (hardware and software) and the users which directly access it or through application programs. The database contains

- **User Data** - It consists of a table of data called Relation where Columns and rows
- **Metadata (**data about data) - A description of the structure of the database is known as Metadata. System Tables store the Metadata which includes no. of Tables with names, no. of fields with names and Key Fields.

➢ **User**

Users are those persons who need the information from the database. There are many different types of user's access to the database totally or partially. User may be end user, online user, application programmer, database administrator and database designer.

➢ **Procedure**

Procedures refer to the instructions and rules that help to design the database and to use the DBMS. The user operate and manage the DBMS through procedure. This includes procedures to setup and install a DBMS, to login and logout of DBMS software, to manage databases, to take backups, generating reports etc.

Database Users

There will be different types of users depending on their need and way of accessing the database.

1) End user:

The clients of the database are generally treated as end users. They are also called **Naive or agile user.** Naive Users are Un-sophisticated users, which has no knowledge of working, database design, the access mechanism etc. These users are interact with database with the help of previously developed application programs.

Example- Cashier in any bank is a naïve user because they don't have any DBMS knowledge but they still use the database and perform their given task.

2) Online users:

Online users are communicating with database directly using online terminal or in directly using user interface. The online users are aware of the presence of the database system. They are permitted for limited interaction with database.

3) Application programmer:

They are expert who write application programs in programming languages (Java, .NET) to interact with databases through DML queries. These DML queries are written in the application programs like JAVA, Python etc. These queries are converted into object code to communicate with the database.

Example- writing a JAVA program to generate report of employees who are working in particular department. To use a query to fetch the data from database. It will include an embedded SQL query in the JAVA Program.

4) Sophisticated users:

They are database developers, who write SQL queries to select/insert/delete/update data. They directly interact with the database by query language like SQL. Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database.

5) Specialized users:

These are also sophisticated users, but they write special database application programs for retrieve complex data for some specific needs. e.g. graphics data and audio data.

6) Database administrator:

The database administrator is person or group who is responsible for managing everything that is related to database is called database administrator or DBA is super-user of the system. He makes the policies, strategies and provides technical supports. They perform the complete administration of database systems with full permission.

➢ Role or Functions of DBA

A database administrator's responsibilities can include the following task

- Backup and recovery of databases
- Installing and upgrading the database server and application tools
- Allocating storage for the database system and planning future requirements
- Modifying the database structure and designing and ensuring key constraints
- Granting of User Access and maintaining system security
- Controlling and monitoring user access to the database
- Monitoring performance of database
- Ensuring fulfillment with database vendor license agreement
- Contacting database vendor for technical support

Database Architecture

The main purpose of Database Architecture is to store or manage data in a proper way. That is easily access and maintain. The design of a DBMS depends on architecture. DBMS architecture helps to design, development, implementation, and maintenance of a database. The design of DBMS may be in three types: **centralized** (all the data stored at one location), **decentralized** (multiple copies of database at different locations) or **hierarchical** (Tree structure). The architecture of a DBMS can be seen as either single tier or multi-tier. Database Architecture are:

1. 1-tier DBMS architecture
2. 2-tier DBMS architecture
3. 3-tier DBMS architecture

➢ 1-tier architecture

The database is directly available to the user for accessing and storing data. Any changes will be directly done on the DBMS itself. Generally such setup is used for local application development, where programmers directly communicate with the database for quick response.

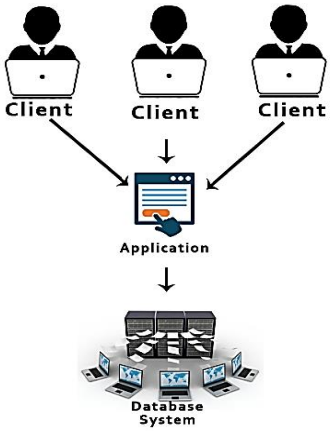
The simplest of Database Architecture are **1 tier** where the Client, Server and Database all reside on the same machine. Anytime you can access database. Database designers and programmers normally use a single-tier architecture. It is very simple and easy to use. But such architecture is rarely used in production.

Example: Suppose you want to access the records of employee from the database and the database is available on your computer system, so the request will be done by your computer and the records will be accessed from the database by your computer. This type of system is local database system.

➢ 2-tier DBMS Architecture

The 2-Tier architecture is same as basic client-server architecture. Application at the client end directly communicates with the database at the server side.

- The user interfaces and application programs are run on the client-side.



- The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with DBMS, client-side application establish a connection with server side.
- Application Program Interface (API) standards like ODBC (Open Database Connectivity) and

JDBC(Java Database Connectivity) are used for interaction between the client and the server.

- This architecture provides Direct and faster communication.
- The Client-Application sends the request to the server and it processes the request and sends back with data.
- Client Application handles both the Presentation layer (application interface) and the Application layer (logical operations). The client-Application layer can be build using C, C++, Java, Python, PHP, .NET languages.

Example: Consider a railway ticket reservation system. A two persons are reserving the ticket from Delhi to Goa on particular day at the same time. Now requirement of two tickets for different persons. It takes the request from both person and queues the requests entered by each person. Here the request entered to application layer and request is sent to database layer. Once the request is processed in database, the result is sent back to application layer for the user.

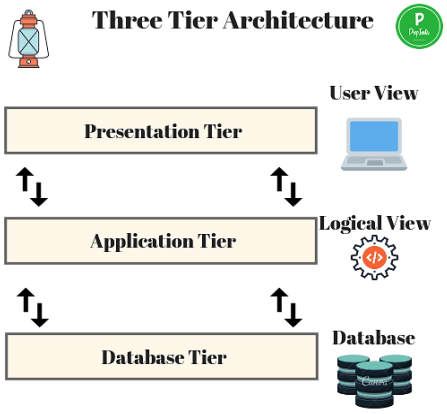
➢ 3-tier DBMS architecture

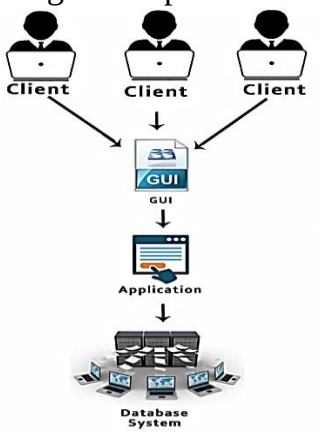
It is an extension of the 2-tier architecture. In 3-tier architecture, an additional Presentation or GUI Layer is added, which provides a graphical user interface for the End user to interact with the DBMS. This intermediate layer exchange of partially processed data between server and client.

- 3-Tier architecture is the most complex among all three, but solves almost all the issues that occurs in 2-Tier and 1-Tier architecture.
- Client can't directly communicate with the server.
- End user has no idea about the existence of the database beyond the application server vice-versa. 3-Tier architecture is used in case of large web application.

3-Tier DBMS architecture are:

- **Database Server (Tier-1):** it deals with all the data and information. Also, it guarantees that all the data is stored in a secured manner and there might not occur a situation of data inconsistency or data redundancy.
- **Application Layer (Tier-2):** The application layer acts as an intermediate between the User/Client and the actual database. But it ensures to present abstracted view of the database and provides a way to respond to the queries requested by the user and fetching the response from the database in tier-1.





- **User/Client Layer (Tier-3):** This is the top most layer where users/clients can request data. Data is fetched from the database and passed to the user/client through the application layer. It also provides the graphical user interface(GUI) to the users

Advantages of Three Tier DBMS Architecture:

- Scalability - application layer implements load balancing, there can be handle many clients.
- Data Integrity - Data corruption and bad requests can be avoided because of the checks performed in the application layer on each client request.
- Security - The removal of the direct connection between the client and server systems via abstraction reduces unauthorized access to the database.

The goal of Three-tier architecture is:

- The separation of an user applications and physical database
- One or more user access same data but different customize views of data. It can changes the views the data but change should not affect other users.

- A user's interaction with the database should be independent.
- DBA changes database storage and conceptual structure without affecting the user's views.

➢ **N-tier architecture** divides the whole system into related but independent n modules, which can be independently modified, altered, changed, or replaced.

➢ **Three levels or Schema Database architecture**

The three schema architecture is also called ANSI/SPARC architecture or three-level architecture. The three level architecture describes how data is viewed or represented by user in database.

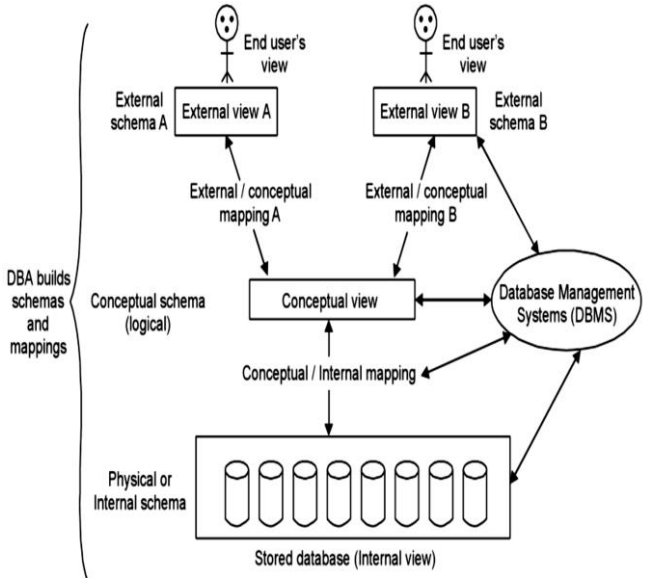
1) External or View level
2) Conceptual or Logical level
3) Internal or Physical level.

1. External / View Level

- External level is related to the data which is viewed by individual users.
- This level is closest to the user with high level data abstraction.
- A user can access only that part of database which will be authorized by DBA. i.e. Remaining part of database (data type, attribute & other information) is hidden from the user for security purposes. End user cannot access entire database.
- External level provides different view of the same data to different users. This view hides all complexity. Create only user interaction with system.
- External level consists of user specific view of the logical records and their relationships.

2. Conceptual /Logical Level

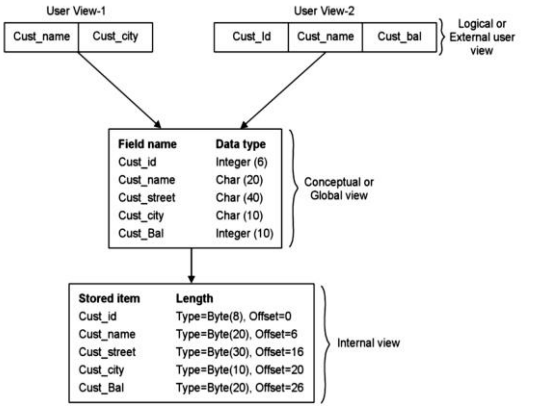
- This is the next higher level of data abstraction.
- It describes what data is stored in the database and what the relationships between among the data.
- This level describes the overall logical structure of whole database for a DBA & group of users.
- It describes entities, data types, relationships and constraints.
- Conceptual schema is a representation of the entire structure of database and hides internal details of physical storage structure



Consider the following customer's database :

Cust_id	Cust_name	Cust_street	Cust_city	Cust_Bal

The Fig. 1.8 below shows the specific information actually available at each level regarding a particular user.



3. Internal / Physical Level

- This is the lowest level of data abstraction close to physical storage of data.
- Physical level describes the physical storage structure of data in database. It describes how data are actually stored in the database or device.
- Data is stored in the form of bits with physical addresses but users are viewed in the form of files.
- Internal level achieve optimal runtime performance and storage space utilization.
- It provides internal view of physical storage of data.
- It completely describe the complex low-level data structures. It also deals with Data Compression and Encryption techniques.

Example: Consider College Database system. If User1 is a Library clerk, the external view would contain only the student and book information. If User2 is an account office clerk then he may be interested in students detail and fee detail. Shows specific information actually available at each level regarding a particular user.

➢ **Different Mappings in Three Level Architecture of DBMS**

Process of transforming request and results between three levels it's called **mapping.**

1. Conceptual/Internal Mapping:

- The conceptual/internal mapping defines the communication between the conceptual view and stored database on physical device.
- If structure of stored database is changed, then conceptual/internal mapping must be changed accordingly. Conceptual schema can remain unchanged.

2. External/Conceptual Mapping:

- The external/conceptual mapping defines the communication between a particular external view and conceptual view.
- If structure of the conceptual schema is changed, then external/conceptual mapping must be changed accordingly, External schema can remain unchanged.
- Many number of external views can exist at the same time.

➢ **Data Independence**

Data independence is a capacity to change the schema at one level of database system without having to change the schema at the next higher level.

1. Logical Data Independence: If a change at the conceptual level without affecting the user level or external level, it is known as *logical data independence*. E.g. the name field in conceptual view is stored as first name, middle name and last name but in external view as a single name field.

2. Physical Data Independence: If the changes are made at the internal level without affecting the above layers, it is known as *physical data independence*. E.g. The location of the database, if changed from C drive to D drive will not affect the conceptual view or external view as the commands are independent of the location of the database.

Physical Data Independence	Logical Data Independence
How the data is stored into the system.	The structure or the changing data definition.
It is easy to retrieve.	It is difficult to retrieve because the data is dependent on the logical structure of data.
Any change at the physical level, does not require to change at the application level.	The change in the logical level requires a change at the application level.
The modifications made at the internal level may or may not be needed to improve the performance of the structure.	The modifications made at the logical level is significant whenever the logical structure of the database is to be changed.
It is concerned with the internal schema.	It is concerned with the conceptual schema.
Example: Change in compression techniques, and storage devices etc.	Example: Add/Modify or Delete a new attribute.

Schemas and Instances

➢ **Schema**: The overall design of a database is called the schema. It contains a descriptive detail of database which includes data structures, table design and relation of data from table to table based on queries etc. three types: Physical schema, logical schema and view schema.

1) Design of a database at physical level is called **physical schema**, how the data stored in blocks of storage is described at this level.

2) Design of database at logical level is called **logical schema**, programmers and DBA work at this level, data can be described, what data is stored in the database, what are the relationship that exists among all of the data.

3) Design of database at view level is called **view schema**. It describes end user interaction with database systems.

Subschema: describe different views of the database

Example: student_ table

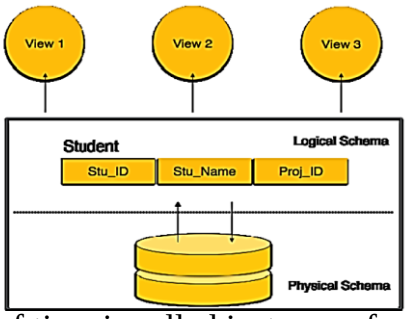
Stu_ID	Stu_Name	Stu_Address	Mobile_No	Class

stu_project

Stu_ID	proj_ID	Proj_Name

Example: Instances of Student_table

Stu_ID	Stu_Name	Stu_Address	Mobile_No	Class
3001	Ram	Sangola	9876543210	Mca
4041	Laxman	Pune	7896543218	Msc



- **Instance:** The data stored in database at a particular moment of time is called instance of database.
  Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

Database languages

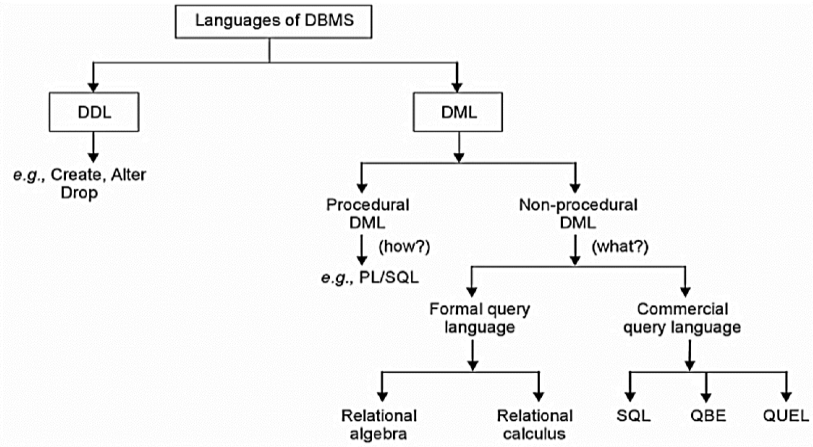
Database languages are used for read, update and store data in a database. There are several languages can be used for this purpose; one of them is SQL (Structured Query Language).

- **Data Definition Language**
  - **DDL** stands for **D**ata **D**efinition **L**anguage. It is used to define database structure.
  - It is used to create schema, tables, indexes, constraints, etc. in the database.
  - Using the DDL statements, you can create the skeleton of the database.
  - Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Commands under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.
- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.
- **Comment:** It is used to comment on the data dictionary.

These commands are used to update the database structure.



- **Data Manipulation Language (DML)**: DML is used for accessing and manipulating data in a database. It allows users to insert, update, delete and select (retrieve) data from the database. DML divide into procedural DML and Non procedural DML.
  - **Select:** It is used to retrieve data from a database.
  - **Insert:** It is used to insert data into a table.
  - **Update:** It is used to update existing data within a table.
  - **Delete:** It is used to delete all records from a table.
  - **Merge:** It performs UPSERT operation, i.e., insert or update operations.
  - **Call:** It is used to call a structured query language or a Java subprogram.
  - **Explain Plan:** It has the parameter of explaining data.

- **Lock Table:** It controls concurrency.

Procedural DML	Non-procedural DML
This language requires user to specify what data is required and how to get those data.	This language requires user to specify what data is required without specifying how to get those data.
It is embedded in a high-level language.	It uses directly the queries to DBMS to find certain data.
Network and Hierarchical DMLs are normally procedural.	Relational DBMSs use these languages for data retrievals, (like SQL, QBE, QUEL etc.)
They are difficult to use and learn. PL/SQL is a procedural DML.	They are normally easier to learn and use Because less work is done by the user.

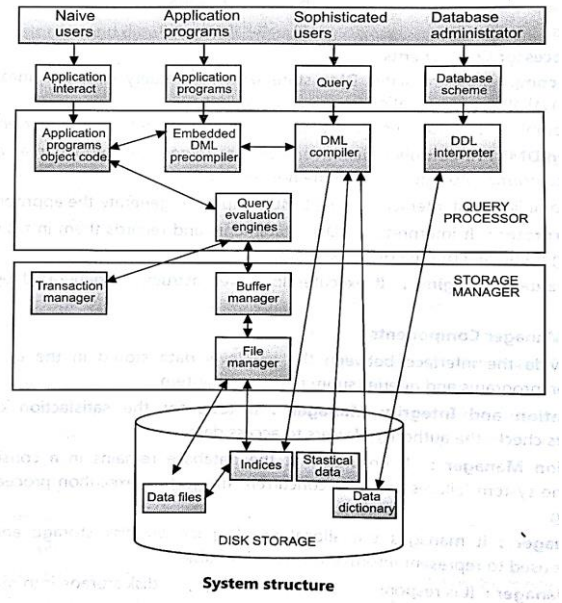
➢ **Data Control language (DCL)**: It is used to give the permission to user for use the database or object. It can also take back the permission. It allows central administration of user privileges and access. Data control language provides command to grant and take back authority.
- **Grant:** It is used to give user access privileges to a database.
- **Revoke:** It is used to take back permissions from the user.

➢ **TCL: Transaction Control Language:** TCL statements are used to manage the changes made by DML statements. It allows statements to be grouped together into logical transactions. These commands can cancel changes made by other commands i.e. rolling back to original state. It can also make changes permanent.
- **Commit:** It is used to save the transaction on the database.
- **Rollback:** It is used to restore the database to original since the last Commit.
- **Save Point:** to save temporarily

Database Structure

A database system is divided into several components and each component is assigned a specific operation to perform. Database is running on:

i. Centralized
ii. Client-server
iii. Parallel (multi-processor)
iv. Distributed



System structure

➢ **Query Processor:**

It provides interface between database and application programs. The query processor will accept query from user and solves it by accessing the database. It contains following components:

- **DDL interpreter**

It interprets DDL statements and fetch the definitions in the data dictionary.

- **DML compiler**

a. This will translates DML statements in a query language into low level instructions that the query evaluation engine understands.
b. A query can be translated into any of a number of alternative evaluations plans for same query result DML compiler will select best plan for query optimization.

- **Query evaluation engine**

This engine will execute low-level instructions generated by the DML compiler on DBMS. They mainly deal with solving all problems related to queries and query processing. It helps database system simplify and facilitate access to data.

- **Embedded DML precompiler –**

Which converts DML Statements embedded in an application program to normal procedure calls in the host language. The precompiler must interact with the DML compiler to generate the appropriate code.

➢ **Storage Manager/Storage Management:**

- A storage manager acts as an interface between the stored data in a database and application programs and queries submitted to the system.
- He is responsible for storing, retrieving and updating data in the database.
- **The storage manager components include:**
  - **Authorization and integrity manager:** Checks for integrity constraints and authority of users to access data.
  - **Transaction manager:** Ensures that the database remains in a consistent state although there are system failures.
  - **File manager:** Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
  - **Buffer manager:** responsible for retrieving data from disk storage into main memory.
  - **Data structures implemented by storage manager.**
  - **Data files:** Stored in the database itself.
  - **Data dictionary:** Stores metadata about the structure of the database.
  - **Indices:** Provide fast access to data items.

Types of Data Models

A Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in the database system. Individual database models are designed based on the rules and concepts.

Types of data model such as

1. Object based logical model
   a. ER-model
   b. Functional model
   c. Object oriented model
   d. Semantic model
2. Record based logical model
   a. Hierarchical database model
   b. Network model
   c. Relational model
3. Physical model

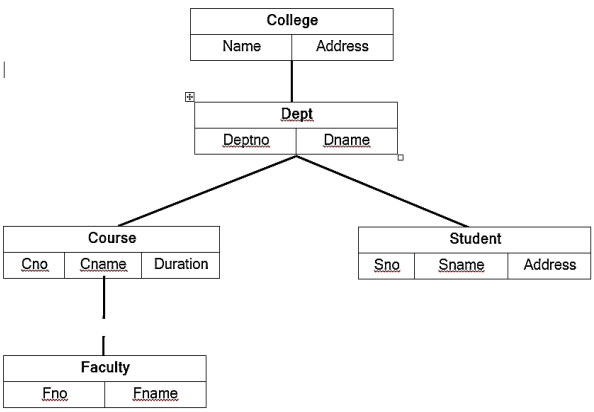
A data model contains three components:

- A structural part, according a set of rules databases can be constructed.
- A manipulative part, operations that are used for updating or retrieving data from the database and for changing the structure of the database.
- A set of integrity rules, which ensures that the data is accurate.

Record Based Logical Model:-

Hierarchical Model

It is one of the oldest database model developed by IBM for information Management System. This data model organizes the data in the tree structure with a single root, to which all the other data is linked. The hierarchy starts from the **root** data and then it expands in the form of a tree adding child node to the parent node. i.e each parent can have many children but each child only has one parent. If a parent node is deleted then the child node is automatically deleted.



This is the oldest form of database. This model use **one-to-many relationship.** Example- One department can have many courses, many professors and of-course many students.

♦ **Advantages:**

- 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.

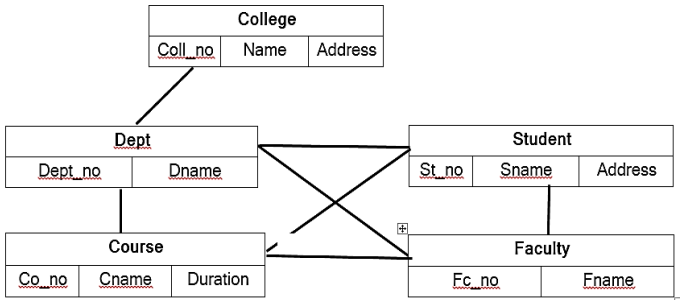
♦ **Disadvantages**

- Complex relationships are not supported and navigation is complex.
- It does not support multi-parent child relationship i.e. a child node needs to have two parent node that can't be represented.
- If a parent node is deleted then the child node is automatically deleted.

➢ **Network Model**

Network Model is an extension of the Hierarchical model. In this model data is organized like a graph and there exist a parent child relationship. A child can have more than one parent. Entities are represented as a connected network with each other.

More relationships are established, data accessing is also easier and faster than hierarchical model. This model allows **one-to-one, one-to-many and many-to-many relationships** i.e. it allowed to child establish relationship with multiple parent.



Example- Each student can be enrolled for several courses and each course may have a number of students enrolled for it.

♦ **Advantages**

- Conceptually simple and easy to design.
- Multi-parent support i.e. Handle (1:M) and (M:M) relationships
- Representation of relationship between entities is helped to database integrity.
- Data manipulation can be done easily.
- Any change in parent record is reflected in the child record.

♦ **Disadvantages**

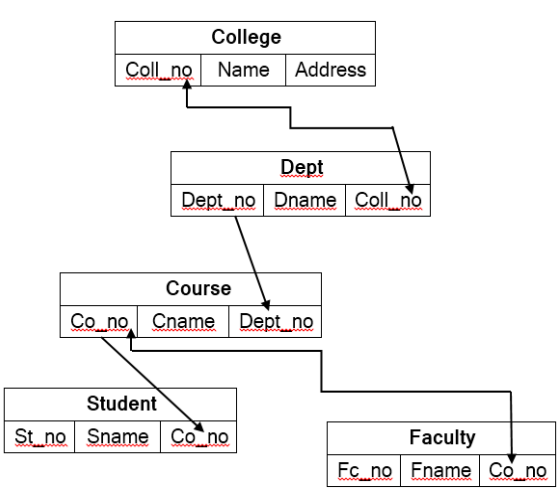
- Detailed structural knowledge is required.
- lack of structural independence.
- Updation, deletion, insertion is very complex.
- Data relationships must be predefined.
- Much more complex than the hierarchical date model.

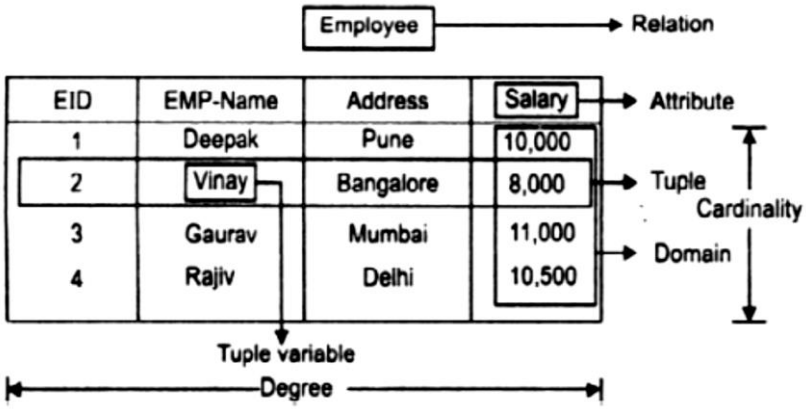
➢ **Relational Model**

The relational database was developed by E. F. Codd at IBM in 1970.The relational model represents data is organized in rows and column structure i.e., two-dimensional tables and the relationship is maintained by storing a common field. 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. Each relation have a group of column with unique names and rows, where column represents attribute of an entity and rows represents records.

- **Attribute /field:** An attribute is a descriptive property or characteristics of an entity i.e. *Columns in a table are called attributes.* Example- EID Emp_name, Address etc.
- **Tuple:** Each row in the table is called tuple. A row contains all the information about any instance of the object.

- **Degree:** The number of column/attributes in a relation is called as degree.
  Example: In the employee Relation Degree is 4





- **Cardinality:** The number of rows (tuples) in a table is called cardinality.
  Example: In the employee Relation cardinality is 4
- **Domain:** Domain is the set of all permitted values or information for one or more attribute.
  Example: In relation Employee, domain attribute salary is 10000,8000,11000.

Properties of Relation:

- Table names in the database should be unique.
- Attribute names should be unique.
- Attributes and Tuples are unordered
- Each record in the table is unique i.e. no duplicate rows.
- There is only one (atomic) value for each attribute of a tuple.

♦ Advantages

- Use of SQL language to access data.
- Easier database design implementation and management
- Improves conceptual simplicity.
- Structural Independence-Changes in database structure without changing the way to access the data
- Data independence.
- Multi-level relationships between data.

♦ Disadvantages

- Data anomalies.
- People need training if they want to use the system effectively and efficiently.
- Slower processing than hierarchical and network models.

****

Q1. What is database? Explain advantages of DBMS?
Q2. Define data independence Distinguish between physical data independence and logical data independence.
Q3. Define Schema and Instance.
Q4.Why 3-tier architecture useful? Explain Relational model.
Q5. What is Join? Explain types of join with example.
Q6. Explain different types of database users.