Introduction to DBMS

DBMS stands for Database Management System. We can break it like this DBMS = Database + Management System. Database is a collection of data and Management System is a set of programs to store and retrieve those data. Based on this we can define DBMS like this: DBMS is a collection of inter-related data and set of programs to store & access those data in an easy and effective manner.

What is the need of DBMS?

Database systems are basically developed for large amount of data. When dealing with huge amount of data, there are two things that require optimization: Storage of data and retrieval of data.

Storage: According to the principles of database systems, the data is stored in such a way that it acquires lot less space as the redundant data (duplicate data) has been removed before storage. Let's take a layman example to understand this:

In a banking system, suppose a customer is having two accounts, one is saving account and another is salary account. Let's say bank stores saving account data at one place (these places are called tables we will learn them later) and salary account data at another place, in that case if the customer information such as customer name, address etc. are stored at both places then this is just a wastage of storage (redundancy/ duplication of data), to organize the data in a better way the information should be stored at one place and both the accounts should be linked to that information somehow. The same thing we achieve in DBMS.

Fast Retrieval of data: Along with storing the data in an optimized and systematic manner, it is also important that we retrieve the data quickly when needed. Database systems ensure that the data is retrieved as quickly as possible.

Purpose of Database Systems

The main purpose of database systems is to manage the data. Consider a university that keeps the data of students, teachers, courses, books etc. To manage this data we need to store this data somewhere where we can add new data, delete unused data, update outdated data, retrieve data, to perform these operations on data we need a Database management system that allows us to store the data in such a way so that all these operations can be performed on the data efficiently.

Characteristics of DBMS

Here are the characteristics and properties of Database Management System:

- **Provides security and removes redundancy**
- **Self-describing nature of a database system**
- **Insulation between programs and data abstraction**
- **Support of multiple views of the data**
- **Sharing of data and multiuser transaction processing**
- **Database Management Software allows entities and relations among them to form tables.**
- **It follows the ACID concept (Atomicity, Consistency, Isolation, and Durability).**
- **DBMS supports multi-user environment that allows users to access and manipulate data in parallel.**

Database systems are much better than traditional file processing systems.
DBMS vs File System.

DBMS vs. Flat File

DBMS	Flat File Management System
Multi-user access	**It does not support multi-user access**
Design to fulfill the need for small and large businesses	**It is only limited to smaller DBMS system.**
Remove redundancy and Integrity	**Redundancy and Integrity issues**
Expensive. But in the long term Total Cost of Ownership is cheap	**It's cheaper**
Easy to implement complicated transactions	**No support for complicated transactions**

Difference between File System and DBMS:

Basis	File System	DBMS
Structure	**The file system is software that manages and organizes the files in a storage medium within a computer.**	**DBMS is software for managing the database.**
Data Redundancy	**Redundant data can be present in a file system.**	**In DBMS there is no redundant data.**
Backup and Recovery	**It doesn't provide backup and recovery of data if it is lost.**	**It provides backup and recovery of data even if it is lost.**

Query processing	There is no efficient query processing in the file system.	Efficient query processing is there in DBMS.
Consistency	There is less data consistency in the file system.	There is more data consistency because of the process of normalization.
Complexity	It is less complex as compared to DBMS.	It has more complexity in handling as compared to the file system.
Security Constraints	File systems provide less security in comparison to DBMS.	DBMS has more security mechanisms as compared to file systems.
Cost	It is less expensive than DBMS.	It has a comparatively higher cost than a file system.
Data Independence	There is no data independence.	In DBMS data independence exists.
User Access	Only one user can access data at a time.	Multiple users can access data at a time.
Meaning	The user has to write procedures for managing databases	The user not required to write procedures.
Sharing	Data is distributed in many files. So, not easy to share	Due to centralized nature sharing is easy

	data	
Data Abstraction	It give details of storage and representation of data	It hides the internal details of Database
Integrity Constraints	Integrity Constraints are difficult to implement	Integrity constraints are easy to implement
Example	Cobol, C++	Oracle, SQL Server

Drawbacks of File system

- Data redundancy: Data redundancy refers to the duplication of data, lets say we are managing the data of a college where a student is enrolled for two courses, the same student details in such case will be stored twice, which will take more storage than needed. Data redundancy often leads to higher storage costs and poor access time.
- Data inconsistency: Data redundancy leads to data inconsistency, lets take the same example that we have taken above, a student is enrolled for two courses and we have student address stored twice, now lets say student requests to change his address, if the address is changed at one place and not on all the records then this can lead to data inconsistency.
- Data Isolation: Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.
- Dependency on application programs: Changing files would lead to change in application programs.
- Atomicity issues: Atomicity of a transaction refers to “All or nothing”, which means either all the operations in a transaction executes or none.
  For example: Lets say Steve transfers 100$ to Negan’s account. This transaction consists multiple operations such as debit 100$ from Steve’s account, credit 100$ to Negan’s account. Like any other device, a computer system can fail lets say it fails after first operation then in that case Steve’s account would have been debited by 100$ but the amount was not credited to Negan’s account, in such case the rollback of operation should occur to maintain the atomicity of transaction. It is difficult to achieve atomicity in file processing systems.
- Data Security: Data should be secured from unauthorised access, for example a student in a college should not be able to see the payroll details of the teachers, such kind of security constraints are difficult to apply in file processing systems.

Advantage of DBMS over file system

There are several advantages of Database management system over file system. Few of them are as follows:

- No redundant data: Redundancy removed by data normalization. No data duplication saves storage and improves access time.

- **Data Consistency and Integrity: As we discussed earlier the root cause of data inconsistency is data redundancy, since data normalization takes care of the data redundancy, data inconsistency also been taken care of as part of it**
- **Data Security: It is easier to apply access constraints in database systems so that only authorized user is able to access the data. Each user has a different set of access thus data is secured from the issues such as identity theft, data leaks and misuse of data.**
- **Privacy: Limited access means privacy of data.**
- **Easy access to data – Database systems manages data in such a way so that the data is easily accessible with fast response times.**
- **Easy recovery: Since database systems keeps the backup of data, it is easier to do a full recovery of data in case of a failure.**
- **Flexible: Database systems are more flexible than file processing systems.**

Disadvantages of DBMS:

- **DBMS implementation cost is high compared to the file system**
- **Complexity: Database systems are complex to understand**
- **Performance: Database systems are generic, making them suitable for various applications. However this feature affect their performance for some applications**

Users of DBMS

Following are the various category of users of DBMS

Component Name	Task
Application Programmers	**The Application programmers write programs in various programming languages to interact with databases.**
Database Administrators	**Database Admin is responsible for managing the entire DBMS system. He/She is called Database admin or DBA.**
End-Users	**The end users are the people who interact with the database management system. They conduct various operations on database like retrieving, updating, deleting, etc.**

Different types of Database Users:

Database users are categorized based up on their interaction with the data base.

These are seven types of data base users in DBMS.

1. **Database Administrator (DBA) :**
   Database Administrator (DBA) is a person/team who defines the schema and also controls the 3 levels of database.
   The DBA will then create a new account id and password for the user if he/she need to access the data base.
   DBA is also responsible for providing security to the data base and he allows only the authorized users to access/modify the data base.
   - DBA also monitors the recovery and back up and provide technical support.
   - The DBA has a DBA account in the DBMS which called a system or superuser account.
   - DBA repairs damage caused due to hardware and/or software failures.
2. **Naive / Parametric End Users :**
   Parametric End Users are the unsophisticated who don't have any DBMS knowledge but they frequently use the data base applications in their daily life to get the desired results.
   For examples, Railway's ticket booking users are naive users. Clerks in any bank is a naive user because they don't have any DBMS knowledge but they still use the database and perform their given task.
3. **System Analyst :**
   System Analyst is a user who analyzes the requirements of parametric end users. They check whether all the requirements of end users are satisfied.
4. **Sophisticated Users :**
   Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database. They can develop their own data base applications according to their requirement. They don't write the program code but they interact the data base by writing SQL queries directly through the query processor.
5. **Data Base Designers :**
   Data Base Designers are the users who design the structure of data base which includes tables, indexes, views, constraints, triggers, stored procedures. He/she controls what data must be stored and how the data items to be related.
6. **Application Program :**
   Application Program are the back end programmers who writes the code for the application programs.They are the computer professionals. These programs could be written in Programming languages such as Visual Basic, Developer, C, FORTRAN, COBOL etc.
7. **Casual Users / Temporary Users :**
   Casual Users are the users who occasionally use/access the data base but each time when they access the data base they require the new information, for example, Middle or higher level manager.

Popular DBMS Software

Here, is the list of some popular DBMS system:

- **MySQL**
- **Microsoft Access**
- **Oracle**
- **PostgreSQL**
- **dBASE**
- **FoxPro**
- **SQLite**
- **IBM DB2**

- **LibreOffice Base**
- **MariaDB**
- **Microsoft SQL Server etc.**

Application of DBMS

Below are the popular database system applications:

Sector	Use of DBMS
Banking	For customer information, account activities, payments, deposits, loans, etc.
Airlines	For reservations and schedule information.
Universities	For student information, course registrations, colleges and grades.
Telecommunication	It helps to keep call records, monthly bills, maintaining balances, etc.
Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.
Sales	Use for storing customer, product & sales information.
Manufacturing	It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.
HR Management	For information about employees, salaries, payroll, deduction, generation of paychecks, etc.

Disadvantage of DBMS

DBMS may offer plenty of advantages but, it has certain flaws-

- **Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.**
- **Most database management systems are often complex systems, so the training for users to use the DBMS is required.**
- **In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media**
- **Use of the same program at a time by many users sometimes lead to the loss of some data.**
- **DBMS can't perform sophisticated calculations**

When not to use a DBMS system?

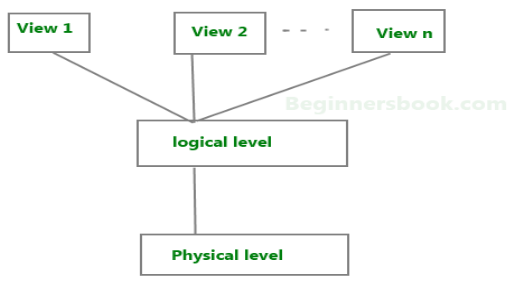
Although, DBMS system is useful. It is still not suited for specific task mentioned below:
Not recommended when you do not have the budget or the expertise to operate a DBMS. In such cases, Excel/CSV/Flat Files could do just fine.

Summary

- **DBMS definition: A database is a collection of related data which represents some aspect of the real world**
- **The full form of DBMS is Database Management System. DBMS stands for Database Management System is a software for storing and retrieving users' data by considering appropriate security measures.**
- **DBMS Provides security and removes redundancy**
- **DBMS has many advantages over tradition Flat File management system**
- **Some Characteristics of DBMS are Security, Self-describing nature, Insulation between programs and data abstraction, Support of multiple views of the data, etc.**
- **End-Users, Application Programmers, and Database Administrators are they type of users who access a DBMS**
- **DBMS is widely used in Banking, Airlines, Telecommunication, Finance and other industries**
- **The main Four DBMS types are 1) Hierarchical 2) Network 3) Relational 4) Object-Oriented DBMS**
- **DBMS serves as an efficient handler to balance the needs of multiple applications using the same data**
- **Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization**

Data Abstraction in DBMS

Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.



Three Levels of data abstraction

We have three levels of abstraction:

Physical level: This is the lowest level of data abstraction. It describes how data is actually stored in database. You can get the complex data structure details at this level.

Logical level: This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

View level: Highest level of data abstraction. This level describes the user interaction with database system.

Example: Let's say we are storing customer information in a customer table. At physical level these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are often hidden from the programmers.

At the logical level these records can be described as fields and attributes along with their data types, their relationship among each other can be logically implemented. The programmers generally work at this level because they are aware of such things about database systems.

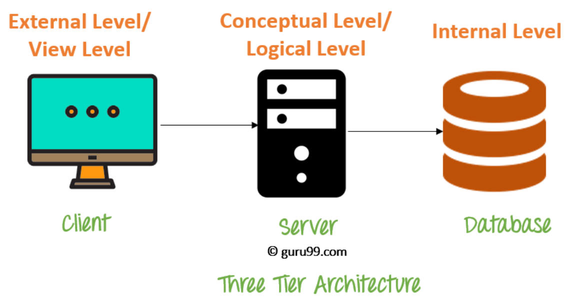
At view level, user just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

DATA ABSTRACTION:

DBMS Schemas: Internal, Conceptual, External

Database systems comprise of complex data structures. Thus, to make the system efficient for retrieval of data and reduce the complexity of the users, developers use the method of Data Abstraction.
There are mainly three levels of data abstraction:

1. Internal Level: Actual PHYSICAL storage structure and access paths.
2. Conceptual or Logical Level: Structure and constraints for the entire database
3. External or View level: Describes various user views



Internal Level/Schema

The internal schema defines the physical storage structure of the database. The internal schema is a very low-level representation of the entire database. It contains multiple occurrences of multiple types of internal record. In the ANSI term, it is also called "stored record'.
Facts about Internal schema:

- The internal schema is the lowest level of data abstraction
- It helps you to keeps information about the actual representation of the entire database. Like the actual storage of the data on the disk in the form of records
- The internal view tells us what data is stored in the database and how
- It never deals with the physical devices. Instead, internal schema views a physical device as a collection of physical pages

Conceptual Schema/Level

The conceptual schema describes the Database structure of the whole database for the community of users. This schema hides information about the physical storage structures and focuses on describing data types, entities, relationships, etc.

This logical level comes between the user level and physical storage view. However, there is only single conceptual view of a single database.
Facts about Conceptual schema:

- Defines all database entities, their attributes, and their relationships
- Security and integrity information
- In the conceptual level, the data available to a user must be contained in or derivable from the physical level

External Schema/Level

An external schema describes the part of the database which specific user is interested in. It hides the unrelated details of the database from the user. There may be "n" number of external views for each database.
Each external view is defined using an external schema, which consists of definitions of various types of external record of that specific view.
An external view is just the content of the database as it is seen by some specific particular user. For example, a user from the sales department will see only sales related data.
Facts about external schema:

- An external level is only related to the data which is viewed by specific end users.
- This level includes some external schemas.
- External schema level is nearest to the user
- The external schema describes the segment of the database which is needed for a certain user group and hides the remaining details from the database from the specific user group

Goal of 3 level/schema of Database

Here, are some Objectives of using Three schema Architecture:

- Every user should be able to access the same data but able to see a customized view of the data.
- The user need not to deal directly with physical database storage detail.
- The DBA should be able to change the database storage structure without disturbing the user's views
- The internal structure of the database should remain unaffected when changes made to the physical aspects of storage.

Advantages Database Schema

- You can manage data independent of the physical storage
- Faster Migration to new graphical environments
- DBMS Architecture allows you to make changes on the presentation level without affecting the other two layers
- As each tier is separate, it is possible to use different sets of developers
- It is more secure as the client doesn't have direct access to the database business logic
- In case of the failure of the one-tier no data loss as you are always secure by accessing the other tier

Disadvantages Database Schema

- Complete DB Schema is a complex structure which is difficult to understand for every one
- Difficult to set up and maintain
- The physical separation of the tiers can affect the performance of the Database

Summary

- There are mainly three levels of data abstraction: Internal Level, Conceptual or Logical Level or External or View level
- The internal schema defines the physical storage structure of the database
- The conceptual schema describes the Database structure of the whole database for the community of users
- An external schema describe the part of the database which specific user is interested in
- DBMS Architecture allows you to make changes on the presentation level without affecting the other two layers

Data Independence in DBMS: Physical & Logical with Examples

What is Data Independence of DBMS?

Data Independence is defined as a property of DBMS that helps you to change the Database schema at one level of a database system without requiring to change the schema at the next higher level. Data independence helps you to keep data separated from all programs that make use of it.
You can use this stored data for computing and presentation. In many systems, data independence is an essential function for components of the system.

Types of Data Independence

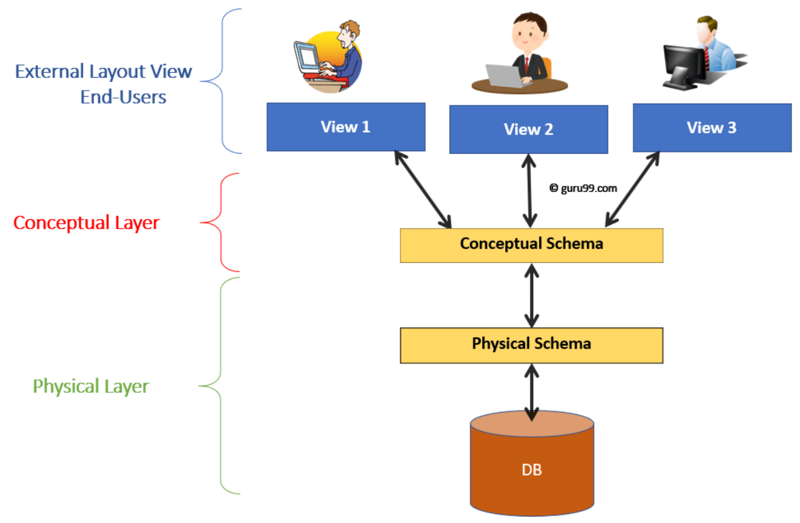
In DBMS there are two types of data independence

1. Physical data independence
2. Logical data independence.

Levels of Database

Before we learn Data Independence, a refresher on Database Levels is important. The database has 3 levels as shown in the diagram below

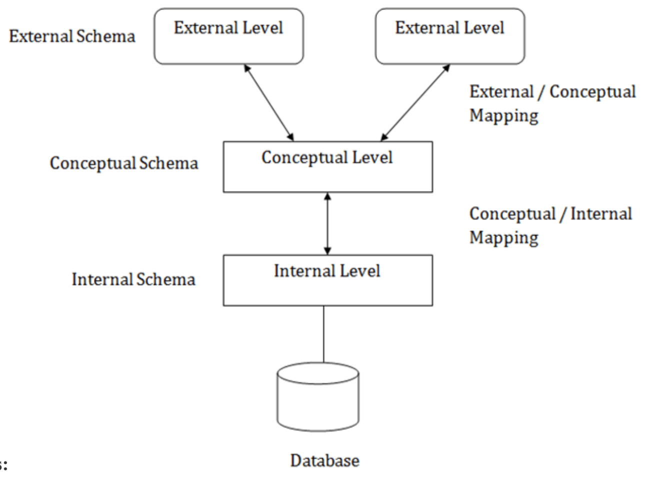
1. Physical/Internal
2. Conceptual
3. External



https://www.geeksforgeeks.org/indexing-in-databases-set-1/

Three schema Architecture

- **The three schema architecture is also called ANSI/SPARC architecture or three-level architecture.**
- **This framework is used to describe the structure of a specific database system.**
- **The three schema architecture is also used to separate the user applications and physical database.**
- **The three schema architecture contains three-levels. It breaks the database down into three different categories.**



The three-schema architecture is as follows:

In the above diagram:

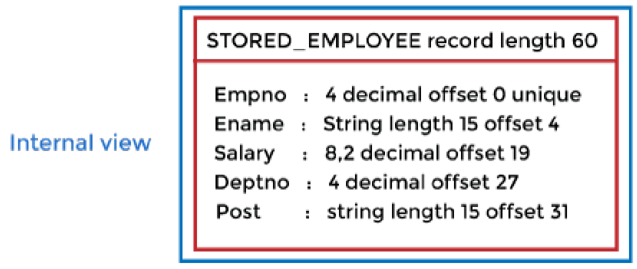
- It shows the DBMS architecture.
- Mapping is used to transform the request and response between various database levels of architecture.
- Mapping is not good for small DBMS because it takes more time.
- In External / Conceptual mapping, it is necessary to transform the request from external level to conceptual schema.
- In Conceptual / Internal mapping, DBMS transform the request from the conceptual to internal level.

Objectives of Three schema Architecture

The main objective of three level architecture is to enable multiple users to access the same data with a personalized view while storing the underlying data only once. Thus it separates the user's view from the physical structure of the database. This separation is desirable for the following reasons:

- Different users need different views of the same data.
- The approach in which a particular user needs to see the data may change over time.
- The users of the database should not worry about the physical implementation and internal workings of the database such as data compression and encryption techniques, hashing, optimization of the internal structures etc.
- All users should be able to access the same data according to their requirements.
- DBA should be able to change the conceptual structure of the database without affecting the user's
- Internal structure of the database should be unaffected by changes to physical aspects of the storage.

1. Internal Level


Internal view

```
STORED_EMPLOYEE record length 60

Empno   :  4 decimal offset 0 unique
Ename   :  String length 15 offset 4
Salary  :  8,2 decimal offset 19
Deptno  :  4 decimal offset 27
Post    :  string length 15 offset 31
```

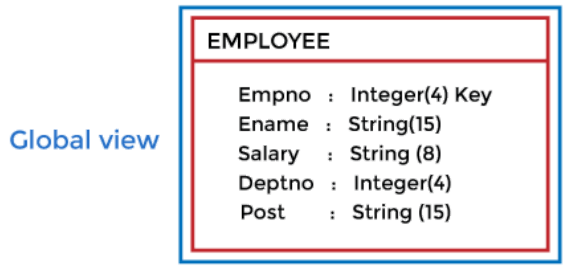
- The internal level has an internal schema which describes the physical storage structure of the database.
- The internal schema is also known as a physical schema.

- It uses the physical data model. It is used to define that how the data will be stored in a block.
- The physical level is used to describe complex low-level data structures in detail.

The internal level is generally is concerned with the following activities:

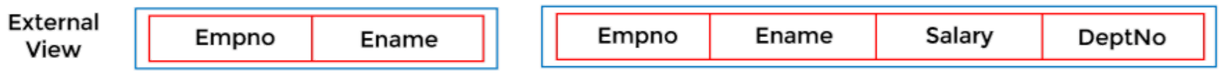
- Storage space allocations.
  For Example: B-Trees, Hashing etc.
- Access paths.
  For Example: Specification of primary and secondary keys, indexes, pointers and sequencing.
- Data compression and encryption techniques.
- Optimization of internal structures.
- Representation of stored fields.

2. Conceptual Level



- The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.
- The conceptual schema describes the structure of the whole database.
- The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.
- In the conceptual level, internal details such as an implementation of the data structure are hidden.
- Programmers and database administrators work at this level.

3. External Level



- At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database.
- An external schema is also known as view schema.
- Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.
- The view schema describes the end user interaction with database systems.

Physical Data Independence

Physical data independence helps you to separate conceptual levels from the internal/physical levels. It allows you to provide a logical description of the database without the need to specify physical structures. Compared to Logical Independence, it is easy to achieve physical data independence.
With Physical independence, you can easily change the physical storage structures or devices with an effect on the conceptual schema. Any change done would be absorbed by the mapping between the conceptual and internal levels. Physical data

independence is achieved by the presence of the internal level of the database and then the transformation from the conceptual level of the database to the internal level.

Examples of changes under Physical Data Independence

Due to Physical independence, any of the below change will not affect the conceptual layer.

- **Using a new storage device like Hard Drive or Magnetic Tapes**
- **Modifying the file organization technique in the Database**
- **Switching to different data structures.**
- **Changing the access method.**
- **Modifying indexes.**
- **Changes to compression techniques or hashing algorithms.**
- **Change of Location of Database from say C drive to D Drive**

Logical Data Independence

Logical Data Independence is the ability to change the conceptual scheme without changing

1. **External views**
2. **External API or programs**

Any change made will be absorbed by the mapping between external and conceptual levels.
When compared to Physical Data independence, it is challenging to achieve logical data independence.

Examples of changes under Logical Data Independence

Due to Logical independence, any of the below change will not affect the external layer.

1. **Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs**
2. **Merging two records into one**
3. **Breaking an existing record into two or more records**

Difference between Physical and Logical Data Independence

Logica Data Independence	Physical Data Independence
Logical Data Independence is mainly concerned with the structure or changing the data definition.	Mainly concerned with the storage of the data.
It is difficult as the retrieving of data is mainly dependent on the logical structure of data.	It is easy to retrieve.
Compared to Logic Physical independence it is difficult to achieve logical data independence.	Compared to Logical Independence it is easy to achieve physical data independence.
You need to make changes in the Application program if new fields are added or deleted from the database.	A change in the physical level usually does not need change at the Application program level.

Modification at the logical levels is significant whenever the logical structures of the database are changed.	Modifications made at the internal levels may or may not be needed to improve the performance of the structure.
Concerned with conceptual schema	Concerned with internal schema
Example: Add/Modify/Delete a new attribute	Example: change in compression techniques, hashing algorithms, storage devices, etc

Importance of Data Independence

- Helps you to improve the quality of the data
- Database system maintenance becomes affordable
- Enforcement of standards and improvement in database security
- You don't need to alter data structure in application programs
- Permit developers to focus on the general structure of the Database rather than worrying about the internal implementation
- It allows you to improve state which is undamaged or undivided
- Database incongruity is vastly reduced.
- Easily make modifications in the physical level is needed to improve the performance of the system.

Summary

- Data Independence is the property of DBMS that helps you to change the Database schema at one level of a database system without requiring to change the schema at the next higher level.
- Two levels of data independence are 1) Physical and 2) Logical
- Physical data independence helps you to separate conceptual levels from the internal/physical levels
- Logical Data Independence is the ability to change the conceptual scheme without changing
- When compared to Physical Data independence, it is challenging to achieve logical data independence
- Data Independence Helps you to improve the quality of the data

ROLE OF DBA:

A database administrator (DBA) is a person or group in charge of implementing DBMS in an organization. The DBA job requires a high degree of technical expertise. DBA consists of a team of people rather than just one person.

The primary role of Database administrator is as follows −

- Database design
- Performance issues
- Database accessibility

- **Capacity issues**
- **Data replication**
- **Table Maintenance**

Responsibilities of DBA

The responsibilities of DBA are as follows −

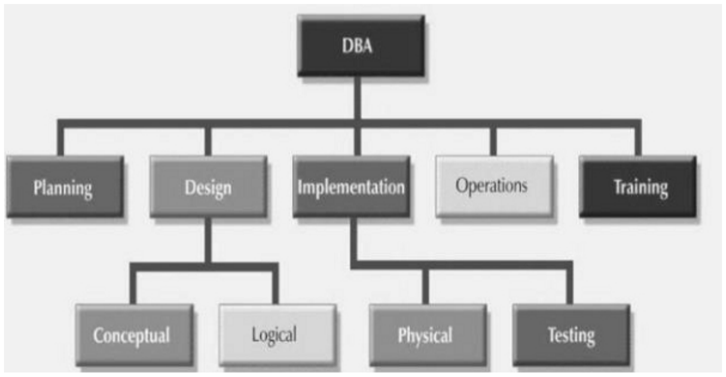
- **Makes the decision concerning the content of the database.**
- **Plans the storage structure and access strategy.**
- **Provides the support to the users.**
- **Defines the security and integrity checks.**
- **Interpreter backup and recovery strategies.**
- **Monitoring the performance and responding to the changes in the requirements.**

Skills required for DBA

The skills required to be a successful DBA are as follows −

- **Database designing.**
- **Knowledge of Structured Query Language (SQL).**
- **Know about distributed architecture.**
- **Knowledge on different operating servers.**
- **Idea on Relational Database Management System (RDBMS).**
- **Ready to face challenges and solve the problems quickly.**

The role of DBA is as shown below −



Describe overall architecture of DBMS with diagram.

The architecture of a database system is greatly influenced by the underlying computer system on which the database is running:

i. Centralized.

ii. Client-server.

iii. Parallel (multi-processor).

iv. Distributed

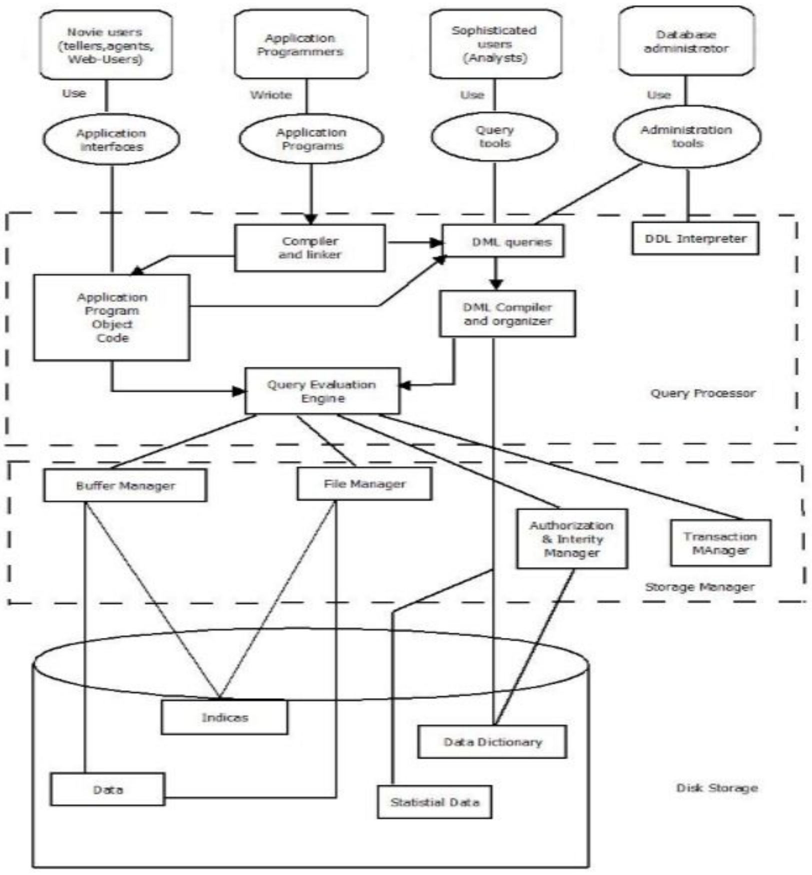


Fig1. Database System Architecture.

Database Users:

Users are differentiated by the way they expect to interact with the system:

- **Application programmers:**
  - **Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces.**
  - **Rapid application development (RAD) tools are tools that enable an application programmer to construct forms and reports without writing a program.**
- **Sophisticated users:**
  - **Sophisticated users interact with the system without writing programs. Instead, they form their requests in a database query language.**
  - **They submit each such query to a query processor, whose function is to break down DML statements into instructions that the storage manager understands.**
- **Specialized users :**
  - **Specialized users are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework.**
  - **Among these applications are computer-aided design systems, knowledge base and expert systems, systems that store data with complex data types (for example, graphics data and audio data), and environment-modeling systems.**
- **Naïve users :**
  - **Naive users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously.**
  - **For example, a bank teller who needs to transfer $50 from account A to account B invokes a program called transfer. This program asks the teller for the amount of money to be transferred, the account from which the money is to be transferred, and the account to which the money is to be transferred.**

Database Administrator:

- **Coordinates all the activities of the database system. The database administrator has a good understanding of the enterprise's information resources and needs.**
- **Database administrator's duties include:**
    - **Schema definition: The DBA creates the original database schema by executing a set of data definition statements in the DDL.**
    - **Storage structure and access method definition.**
    - **Schema and physical organization modification: The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.**
    - **Granting user authority to access the database: By granting different types of authorization, the database administrator can regulate which parts of the database various users can access.**
    - **Specifying integrity constraints.**
    - **Monitoring performance and responding to changes in requirements.**

Query Processor:

The query processor will accept query from user and solves it by accessing the database.

Parts of Query processor:

- **DDL interpreter**
  This will 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 usually be translated into any of a number of alternative evaluation 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.

Storage Manager/Storage Management:

- **A storage manager is a program module which acts like interface between the data stored in a database and the application programs and queries submitted to the system.**
- **Thus, the storage manager 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: It is responsible for retrieving data from disk storage into main memory. It enables the database to handle data sizes that are much larger than the size of 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.**