

# **DATABASE MANAGEMENT SYSTEM**

# Data Vs Information

The term “**Data**” means any representation of facts, figures, symbols, concepts or instructions which is communicated interpreted or processed by human or electronic machine. Data is represented with the help of characters like alphabets (A-Z, a-z), digits (0-9) or special characters (+, -, /, \*, <, >, = etc.).

**Information** is processed or structured data which has some meaningful values for the system or human. Information is the derived data on which decisions and actions are based. For the decision to be meaningful, the processed data must meet the following characteristics:

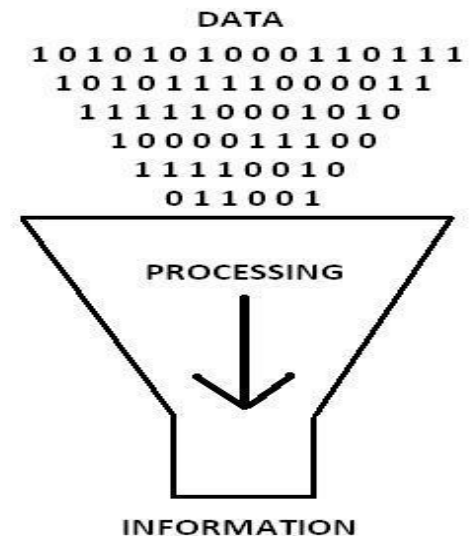
Timely - Information should be available when required.

Accuracy - Information should be accurate.

Completeness - Information should be complete.

## Examples of Data and Information

The area and population of different cities of a country is data. If this data is organized and analysed to find out the population densities of cities, then that is information.



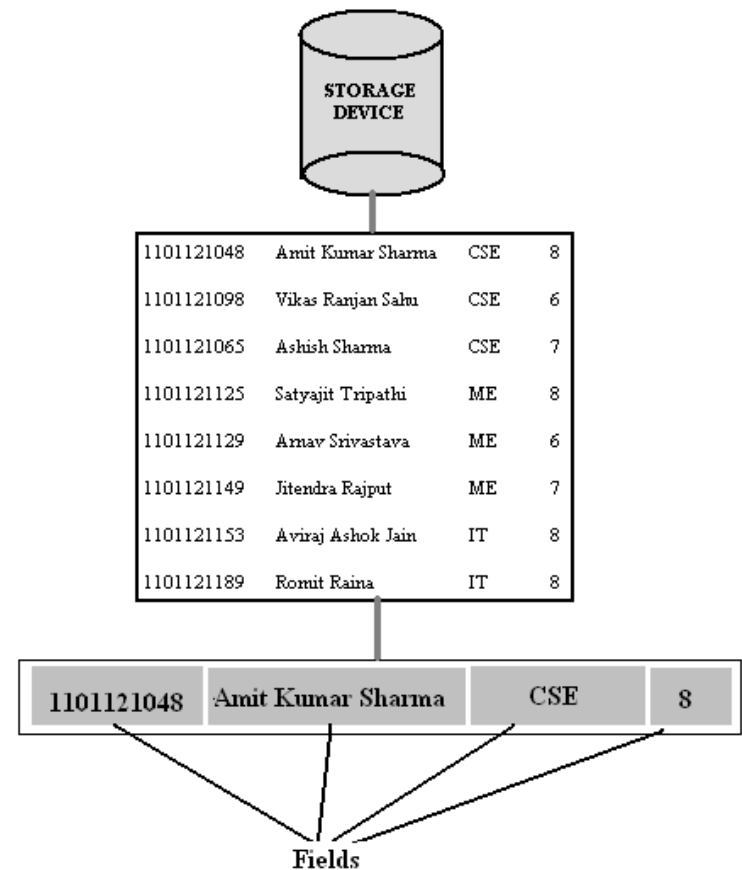
# What is Data

- The history of temperature readings all over the world for the past 50 years is data. If this data is organized and analysed to find that global temperature is rising, then that is information.

Data	Information
Data is raw, unprocessed facts and figures.	Information is processed, organized, structured or presented data.
Data is used as input to process for the computer system.	Information is the output of processed data.
Data does not depend on information.	Information depends on data.
Data does not have any meaning and useful until it is organized.	Information can be meaningful and useful when it is organized by data.
Data is always correct – it is a piece of truth, a thing that has happened.	Information is processed or derived from data that can be wrong.
Example of data: Each student's examination marks is one part of data.	Example of information: The average marks of a class or of the entire college is information that can be derived from the given data.

# What is Database?

- A database is a collection of information that is organized so that it can be easily accessed, managed and updated.
- A database can be as simple as an alphabetical arrangement of names in an address book or as complex as a database that provides information in a combination of formats.
- Traditional databases are organized by *fields*, *records*, and *files*. A field is a basic unit of data storage and contains information related to an attribute of the entity described by the database; a record is one complete set of fields; and a file is a collection of records. Generally, a database is stored as a file or a set of files on magnetic disk or tape, optical disk, or some other secondary storage device.
- For example, a telephone book is similar to a file. It contains a list of records, each of which consists of three fields: name, address, and telephone number.



# What is Database?

- A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.
- A database is a data structure that stores organized information. Most databases contain multiple tables, which may each include several different fields. For example, a company database may include tables for products, employees, and financial records. Each of these tables would have different fields that are relevant to the information stored in the table.

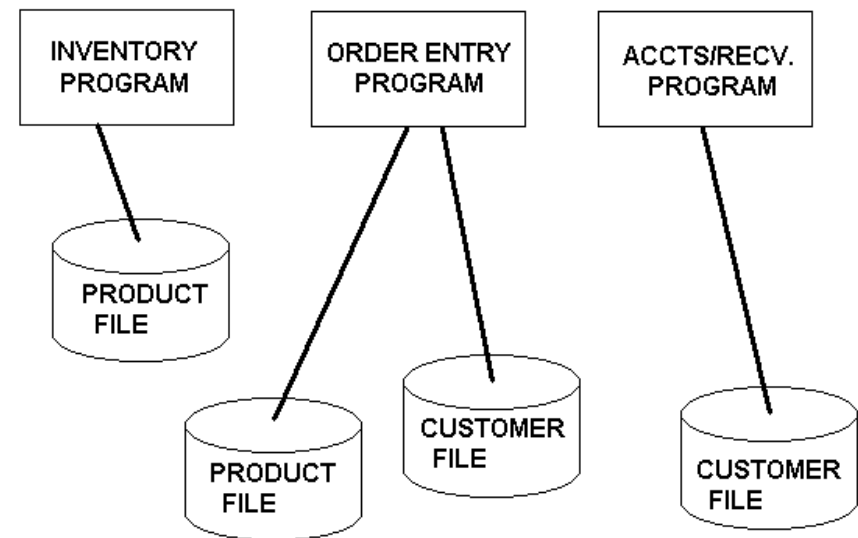
# Traditional File System

Traditional file processing system or simple file processing system refers to the first computer-based approach of handling the commercial or business applications. That is why it is also called a replacement of the manual file system.

Before the use computers, the data in the offices or business was maintained in the files manually. Obviously, it was very laborious, time-consuming, inefficient task especially in large organizations. Computers were initially designed for engineering & scientific applications. Since they helped in efficient management of data in files, file processing environment simply transformed manual file work to computers. So, processing becomes fast and efficient.

The concept of conventional file processing system is shown in figure, which consists of three applications namely Inventory, Order Entry, and Accounts/Received programs. These programs process the data stored in different files such as Customer File and Product File.

As file processing systems were used, their problems were also realized and some of them are very severe.



# Traditional File System

Most explicit and major disadvantages of file system when compared to database management systems are as follows:

- Data Redundancy
- Data Inconsistency
- Difficulty in Accessing Data
- Data Isolation
- Integrity Problems
- Security and access control
- Concurrency Problems

# Traditional File System Vs DBMS

- A database management system coordinates both the physical and the logical access to the data, whereas a file-processing system coordinates only the physical access.
- A database management system reduces the amount of data duplication by ensuring that a physical piece of data is available to all programs authorized to have access to it, whereas data written by one program in a file-processing system may not be readable by another program.
- A database management system is designed to allow flexible access to data (i.e., queries), whereas a file-processing system is designed to allow predetermined access to data (i.e., compiled programs).
- A database management system is usually designed to allow one or more programs to access different data files at the same time. In a file-processing system, a file can be accessed by two programs concurrently only if both programs have read-only access to the file.



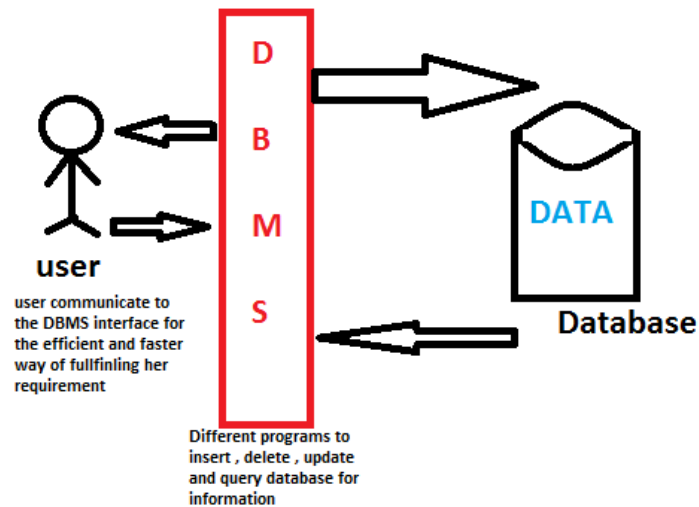
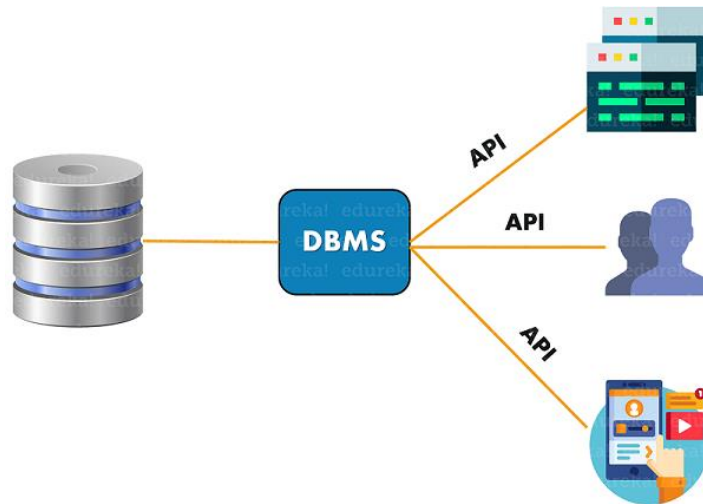
# Traditional File System Vs DBMS

- Redundancy is control in DBMS, but not in file system.
- Unauthorized access is restricted in DBMS but not in file system.
- DBMS provide backup and recovery. When data is lost in file system then it not recover.
- DBMS provide multiple user interfaces. Data is isolated in file system.

# What is DBMS?

- A Database Management System (DBMS) is software designed to store, retrieve, define, and manage data in a database.
- Database Management System (DBMS) is a software for storing and retrieving users' data while considering appropriate security measures. It consists of a group of programs which manipulate the database.
- A database management system (DBMS) is system software for creating and managing databases. A DBMS makes it possible for end users to create, read, update and delete data in a database. The most prevalent type of data management platform, the DBMS essentially serves as an interface between databases and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

# What is DBMS?



# Functions of DBMS?

DBMS performs several important functions that guarantee the integrity and consistency of the data in the database. The most important functions of Database Management System are

(i) **Data Storage Management:** It provides a mechanism for management of permanent storage of the data. The internal schema defines how the data should be stored by the storage management mechanism and the storage manager interfaces with the operating system to access the physical storage.

A modern DBMS system provides storage not only for the data, but also for related data entry forms or screen definitions, report definitions, data validation rules, procedural code, structures to handle video and picture formats, and so on.

Data storage management is also important for database performance tuning. Performance tuning relates to the activities that make the database perform more efficiently in terms of storage and access speed. So, the data storage management is another important function of Database Management System.

# Functions of DBMS?

(ii) **Data Manipulation Management:** A DBMS furnishes users with the ability to retrieve, update and delete existing data in the database.

(iii) **Data Definition Services:** The DBMS accepts the data definitions such as external schema, the conceptual schema, the internal schema, and all the associated mappings in source form.

(iv) **Data Dictionary/System Catalog Management:** The DBMS provides a data dictionary or system catalog function in which descriptions of data items are stored and which is accessible to users. The data dictionary is full of “Metadata”, Database about a database. A data dictionary defines the structure of the database itself and is used in control and maintenance of large databases.

A data dictionary contains the following items-

- The definitions of all schema objects in the database (tables, views, indexes, clusters, synonyms, sequences, procedures, functions, packages, triggers, and so on).
- Space allotted and used by schema objects.
- Types of relationships between data elements.
- Access rights and frequency of access.
- All the users information.

# Functions of DBMS?

(v) **Database Communication Interfaces:** The end-user's requests for database access are transmitted to DBMS in the form of communication messages.

(vi) **Authorization / Security Management:** The DBMS protects the database against unauthorized access, either intentional or accidental. It furnishes mechanism to ensure that only authorized users can access the database.

Database security refers to the use of the DBMS features and other related measures to prevent data from unauthorized access, loss, corruption, or mishandling.

DBMS has some features to provide security to database. Those features are

- Data encryption,
- Authentication,
- Authorization and
- Data views.

# Functions of DBMS?

**Encryption** is a process to covert data into unreadable format. Unauthorized person cannot read and understand this encrypted data. Only authorized use will be able to read it.

**Authentication** is the process by which the system validates a user's logon information. A user's name and password are compared to an authorized list, and if the system detects a match, access is granted to the extent specified in the permission list for that user.

**Authorization** is any process by which someone is allowed to access the database by given user id and password, if password is not successfully entered, the user will be denied for accessing database.

In DBMS, sometimes it is required to provide partial information of database to some users and restrict them to access other part of database.

**Data Views** may be defined by DBA to allow user to access the database tables partially and hide other part of the tables or fields for them.

# Functions of DBMS?

(vii) **Backup and Recovery Management:** The DBMS provides mechanisms for backing up data periodically and recovering from different types of failures. This prevents the loss of data

The DBMS provides mechanisms for backing up data periodically and recovering from different types of failures. This ensures data safety, data integrity and prevents the loss of data.

A **Data Backup** is a copy of data. This copy can include important parts of the database, such as the control file and data files. A backup is a safeguard against unexpected data loss and application errors. If you lose the original data due to any reason, then you can reconstruct it by using a backup.

The reasons for data loss can be divided into five main groups:

- Program errors
- Administrator (human) errors
- Computer failures (system crash)
- Disk failures
- Catastrophes (fire, earthquake) or theft



# Functions of DBMS?

If database is damaged due to any reason, the DBMS restore a physical backup of a data file or control file to the correct state of the database and this process is called *Data Recovery*. It is very important to backup data periodically for proper recovery.

(viii) **Concurrency Control Service:** Since DBMSs support sharing of data among multiple users, they must provide a mechanism for managing concurrent access to the database. When more than one transactions are running simultaneously there are chances of a conflict to occur which can leave database to an inconsistent state. To handle these conflicts we need concurrency control in DBMS, which allows transactions to run simultaneously but handles them in such a way so that the integrity of data remains intact.

# Functions of DBMS?

## Problems of concurrency control

Several problems can occur when concurrent transactions are executed in an uncontrolled manner. Following are the three problems in concurrency control.

- Lost updates
- Dirty read
- Unrepeatable read

## Concurrency Control Protocols

Different concurrency control protocols offer different benefits between the amount of concurrency they allow and the amount of overhead that they impose.

- Lock-Based Protocols
- Two Phase
- Timestamp-Based Protocols
- Validation-Based Protocols

# Functions of DBMS?

(ix) **Transaction Management:** A transaction is a series of database operations, carried out by a single user or application program, which accesses or changes the contents of the database. Therefore, a DBMS must provide a mechanism to ensure either that all the updates corresponding to a given transaction are made or that none of them is made.

A transaction is a sequence of database operations, carried out by a single user or application program, which accesses or updates the contents of the database. DBMS provide a mechanism to ensure either that all the updates corresponding to a given transaction are made or that none of them is made.

In order to fully maintain data integrity and ensure good transactional behaviour, DBMS supports the ACID properties:

- 1) Atomicity:** If any part of a transaction fails, the database state is left unchanged.
- 2) Consistency:** Any transaction will leave the database in a consistent state.
- 3) Isolation:** During a transaction, modified data cannot be accessed by other operations.
- 4) Durability:** The DBMS can always recover the results of a committed transaction.

# Functions of DBMS?

(x) **Database Access and Application Programming Interfaces:** All DBMS provide interface to enable applications to use DBMS services. They provide data access via Structured Query Language (SQL). The DBMS query language contains two components: (a) a Data Definition Language (DDL) and (b) a Data Manipulation Language (DML).

## (xi) **Data Integrity**

- Data integrity refers to the overall completeness, accuracy and consistency of data and is an important feature of a database system.
- Data integrity means that the data contained in the database is accurate and reliable. Data integrity is a set of rules and standard procedures and enforced during the database design phase.
- Data integrity can be maintained through the use of various error checking methods and validation procedures.

DBMS supports basic types of database integrity constraints are:

- 1) **Entity integrity**, not allowing multiple rows to have the same identity within a table.
- 2) **Domain integrity**, restricting data to predefined data types, e.g.: dates.
- 3) **Referential integrity**, requiring the existence of a related row in another table, e.g. a Student for a given RegNo.

# Advantages & Disadvantages of DBMS

## Advantages

- 1) Reduction of Redundancies
- 2) Shared Data
- 3) Data Integrity
- 4) Security
- 5) Conflict Resolution or Concurrency Control
- 6) Transaction Management
- 7) Data Independence
- 8) Backup and Recovery

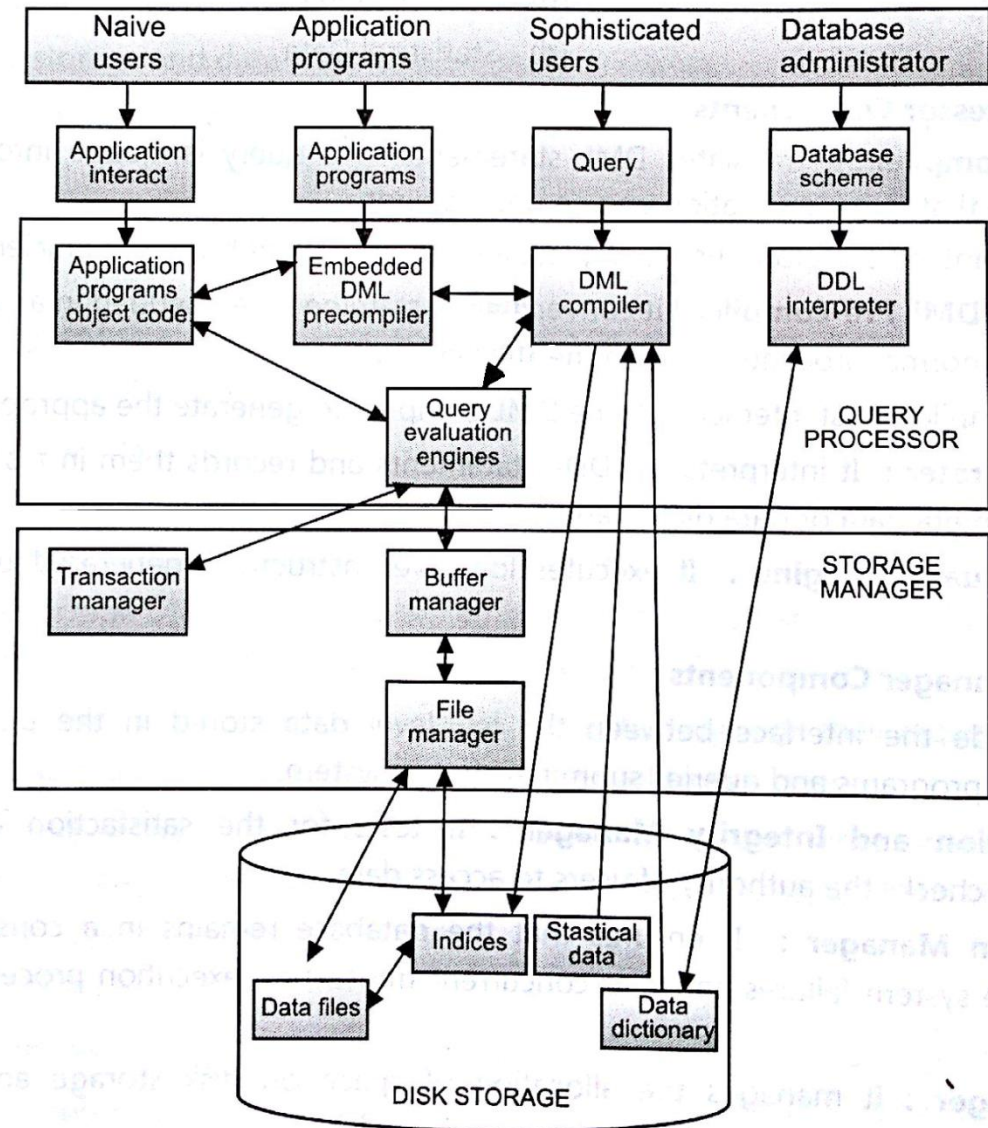
## Disadvantages

- 1) Costly
- 2) Degradation of System Performance due to processing overhead
- 3) Backup and Recovery operations are complex

# Structure of DBMS

The **database system** is divided into three components:

- Query Processor,
- Storage Manager, and
- Disk Storage



System structure

# Structure of DBMS

## **1. Data Definition Language Interpreter**

It execute the low-level statements and records them in a set of tables that having metadata.

## **2. Data Manipulation Language Compiler**

It converts DML statements in source form of query language into necessary object form (i.e. low-level instructions) that query evaluation engine understands.

## **3.Embedded DML Precompiler**

It converts DML statements embedded in application program to normal procedure calls in host language. To generate appropriate code, it must interacts with DML Compiler.

## **4. Query Evaluation Engine**

It executes the low-level instruction generated by DML compiler.

## **5. Transaction Manager**

It ensure that the database remains in consistent state despite the system failure, and concurrent transaction executions proceed without conflicting.

# Structure of DBMS

## 6. Buffer Manager

It is responsible for data fetching from disk storage into main memory and deciding what data to cache in memory.

## 7. File Manager

It manages allocation of space on disk storage and data structure used to represent information that stored on disk.

Some of the Data Structure are needed as the part of physical system for implementation:

- i. **Indices:** These provide fast access to the data items that hold particular values.
- ii. **Statistical Data:** This store statistical information about data in database. To execute a query, this information is used by query processor.
- iii. **Data Files:** These are actual files that store the data in database, i.e. these are database files.
- iv. **Data Dictionary:** This stores metadata about each and every entity of the database along with the security and entity constraints.



# Structure of DBMS

## Information Stored in Data Dictionary (DBMS)

### 1. Database Schema Information

Table and column details (names, data types, sizes).

Indexes, views, and foreign key relationships.

### 2. Constraints and Rules

Primary keys, foreign keys, unique constraints.

NOT NULL, CHECK constraints, referential integrity.

### 3. Storage and Performance Statistics

Number of rows in tables, table size.

Index usage, fragmentation details.

### 4. User and Security Information

Username, roles, and permissions.

Access privileges (READ, WRITE, EXECUTE).

Audit logs (who accessed/modified data).

### 5. Procedural and Programmatic Information

Stored procedures, functions, triggers.

Views, materialized views, default values.

### 6. Query Optimization and Execution Plans

Execution statistics of queries.

Query optimization hints, indexing suggestions.

### 7. Backup and Recovery Information

Last backup time, transaction logs.

Recovery and rollback details.

## Statistical Data Stored in DBMS

### 1. Table and Index Statistics

Number of rows, pages, and distinct values.

Index selectivity, clustering factor.

### 2. Column-Level Statistics

Min, max, mean, median, standard deviation.

Null values count, histogram distribution.

### 3. Query Execution Statistics

Query execution time, disk I/O operations.

Cache hit/miss ratio, join complexity.

### 4. Transaction and Concurrency Statistics

Number of transactions, commit/rollback rates.

Lock contention, deadlock occurrences.

### 5. System Performance Statistics

CPU, memory, and disk usage.

Network traffic in distributed databases.

### 6. User and Access Statistics

Number of active users, login attempts.

Access logs and security audit trails.

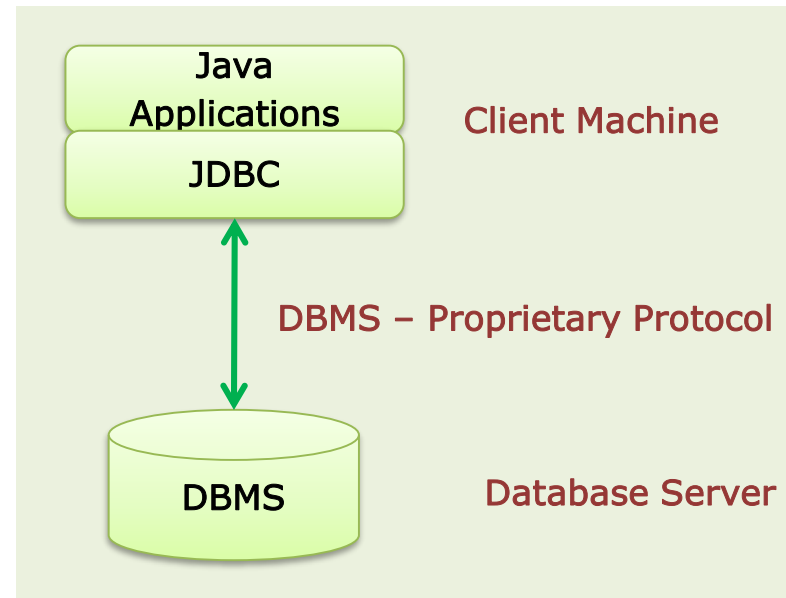
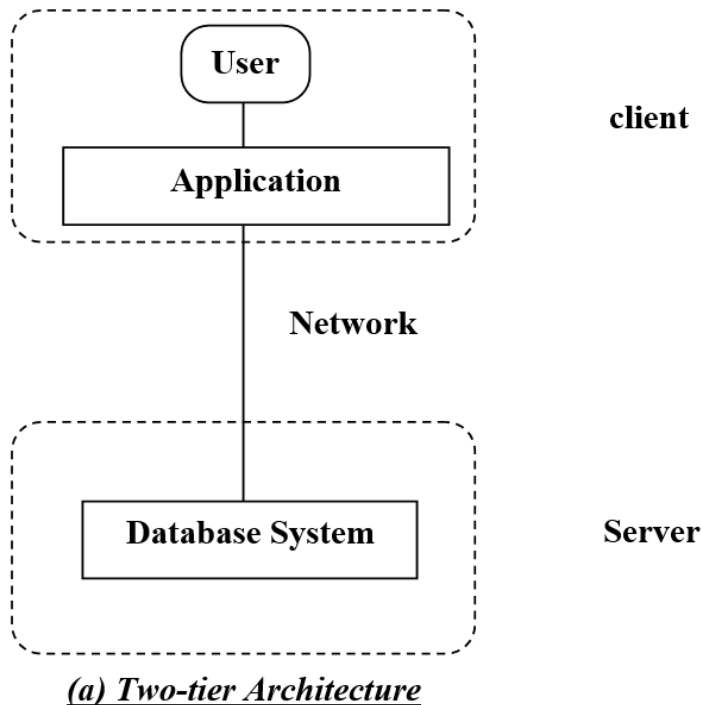
### 7. Backup and Recovery Statistics

Backup frequency, recovery time, transaction logs.

# Database Applications Architectures

Database Applications are partitioned into 2-tier and 3-tier parts as shown in figure.

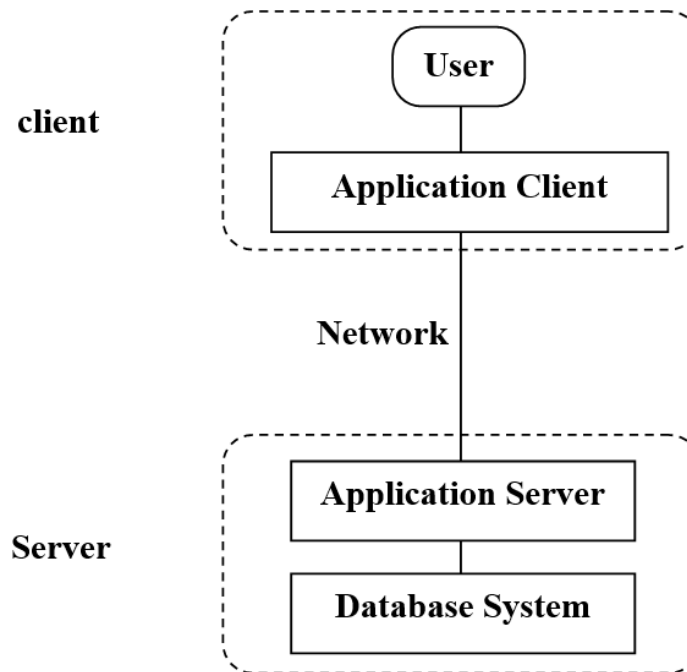
- **Two-tier Architecture:** In the two-tier architecture, the application is partitioned into a component that resides at the client machine and calls the database system functionality at the server machine using query language statements. Application program interface standards like ODBC and JDBC are used for interaction between the client and the server. Two-tiered application examples include desktop applications, games, and music players.



2-tier DBMS Application Architecture

# Database Applications Architectures

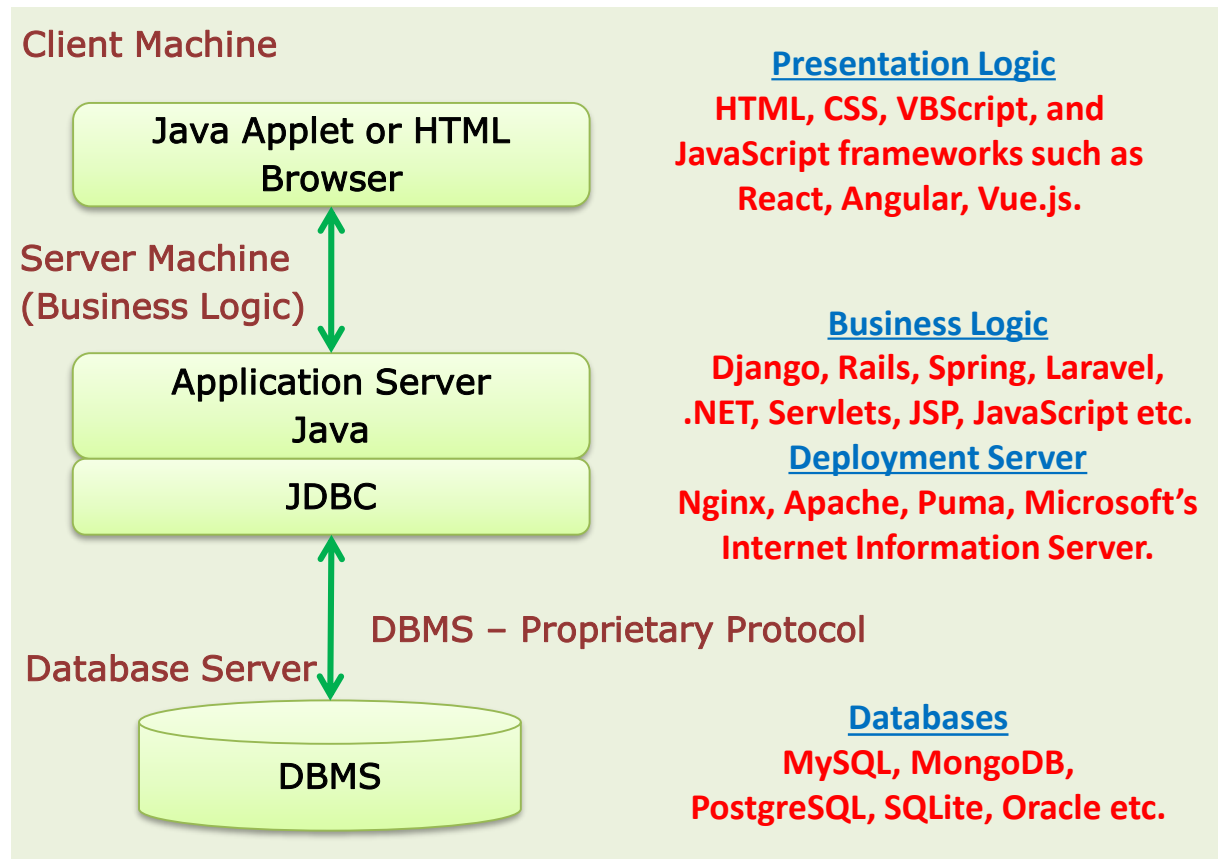
- **Three-tier Architecture:** In three-tier architecture, the client machine contains a form interface and does not contain any direct database calls. It communicates with an application server which in turn communicates with the database system. Thus, the required functionalities are not distributed among the multiple clients. This architecture is appropriate for large applications that run on the World Wide Web. A good example is modern web applications.



(a) Three-tier Architecture

# Database Applications Architectures

The **Business Logic** of the application, which says what actions to carry out under what conditions, is embedded in the application server, instead on being distributed across multiple clients. 3-tier architecture is more appropriate for large organizations, and for applications that run on the World Wide Web.

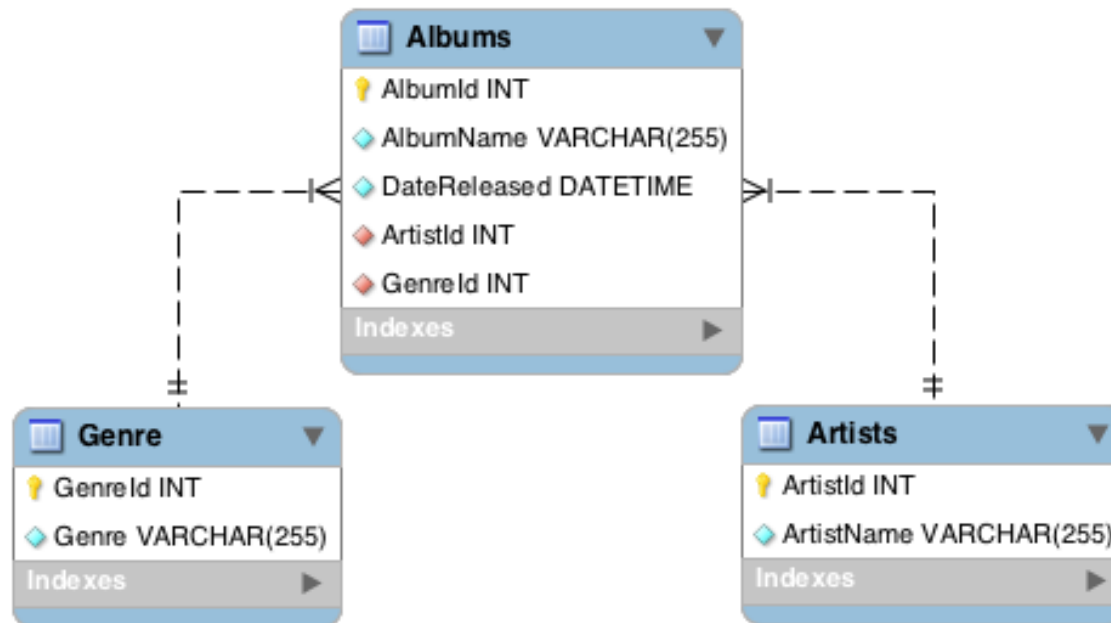


3-tier DBMS Application Architecture

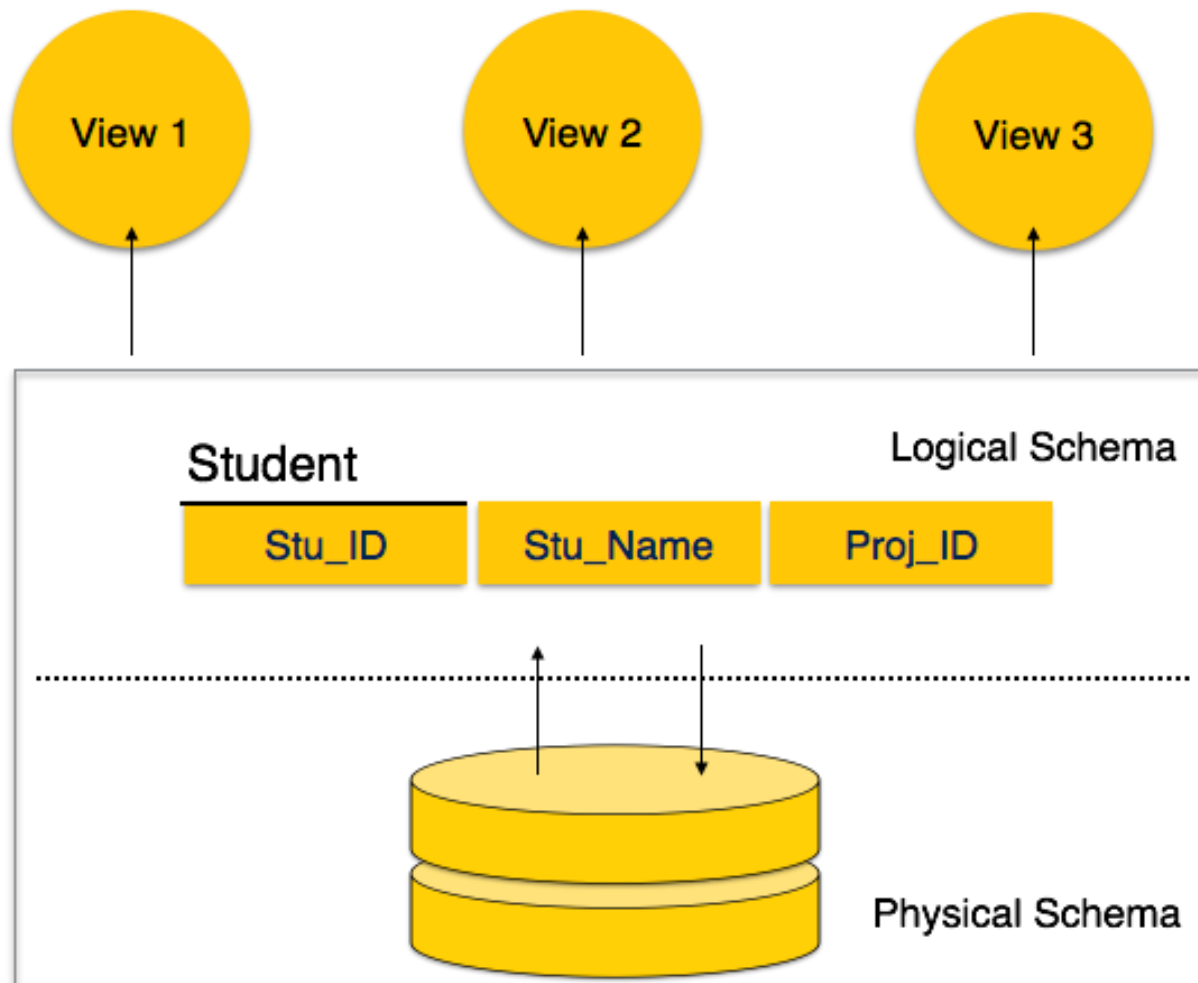
# Schema, Subschema and Instance

## Schema

- Schema is a the logical description of the database.
- The overall design of the database is called Database Schema.
- A schema contains schema objects, which could be tables, columns, data types, views, stored procedures, relationships, primary keys, foreign keys, etc.
- A database schema can be represented in a visual diagram, which shows the database objects and their relationship with each other.



# Schema, Subschema and Instance



# Schema, Subschema and Instance

**Schema and Database are the same things or different?**

Part of the reason for the confusion is that database systems tend to approach schemas in their own way.

- **MySQL Documentation:** A schema is synonymous with a database. Therefore, a schema and a database are the **same thing**.
- **Oracle Database Documentation:** Certain objects can be stored inside a database but not inside a schema. Therefore, a schema and a database are **two different things**.
- **SQL Server Technical Article:** A schema is a separate entity inside the database. So, they are **two different things**.

# Schema, Subschema and Instance

## 1. Internal Schema or Physical Schema

It describes that how data are actually stored in the blocks of storage devices such as hard disk.

## 2. Logical Schema or Conceptual Schema

It describes the structure of the database to the database designer. Programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures

## 3. External Schema or Subschema

It is a subset of main schema having the same properties as schema. It describes the different parts, sets, records and data names of the database to the end user. It allows users to view only the parts of the main database.



# Schema, Subschema and Instance

## Subschema

- It is a subset of the schema having the same properties that a schema has.
- It allows the user to view only that part of the database that is of interest to him.

## How does a subschema work?

- A subschema lets users see only the parts of the database that are relevant to them.
- It can restrict access to the database.
- It can identify which records, areas, and elements are accessible.

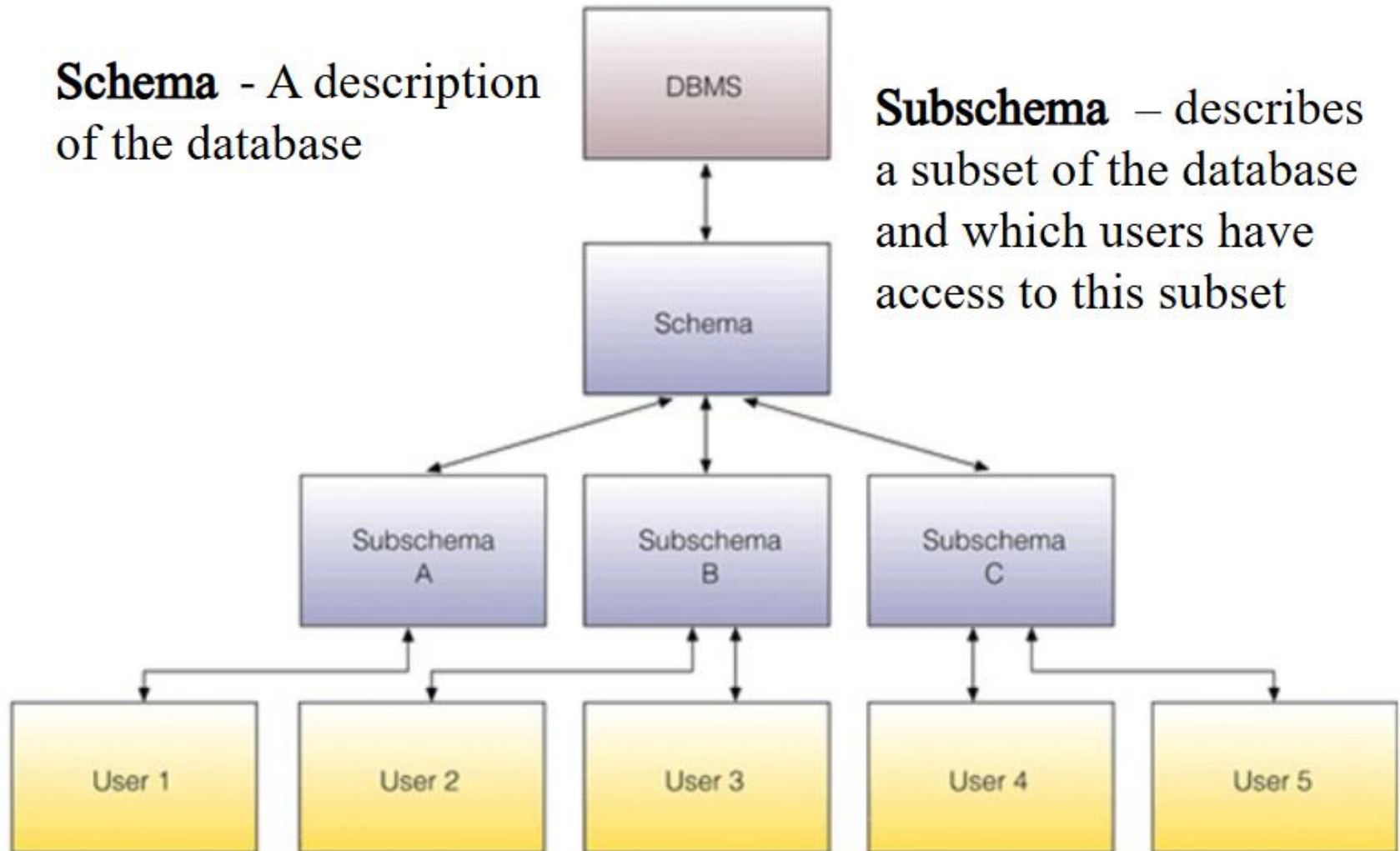
## Why use subschemas?

- Subschemas help promote security by limiting access to data
- They help manage permissions
- They help control access to data at different levels
- They allow different application programs to have different views of the data

# Schema, Subschema and Instance

**Schema** - A description of the database

**Subschema** – describes a subset of the database and which users have access to this subset



# Schema, Subschema and Instance

## Database 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.

A database schema is variable declarations in a program. Variable has particular value at a given instant. Then, the value of variable at particular instant is called database instance.

Database schema (names of columns + the types associated with them)

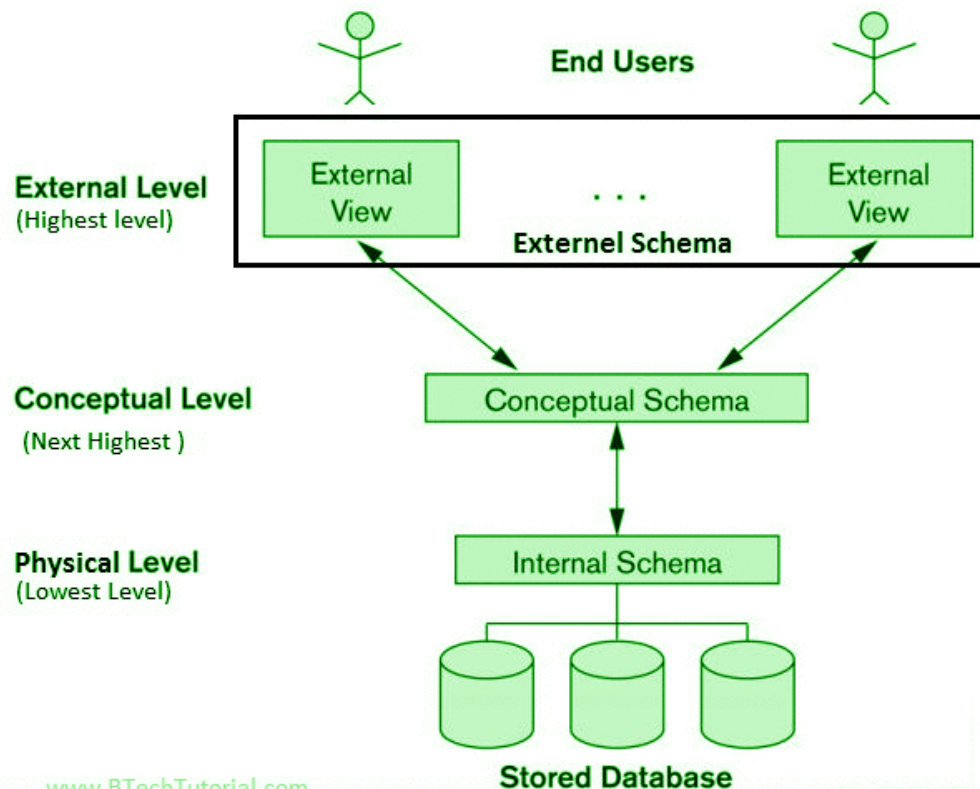
<b>Name</b>	<b>DOB</b>	<b>Address</b>	<b>Job</b>	<b>Scale</b>
<i>String</i>	<i>Date</i>	<i>String</i>	<i>String</i>	<i>Int</i>

A database instance

<b>Name</b>	<b>DOB</b>	<b>Address</b>	<b>Job</b>	<b>Scale</b>
A. Johnson	2/04/1960	London	Programmer	12
B. Holiday	3/10/1947	Leeds	Analyst	14
C. Clark	12/08/1971	York	Programmer	10

# 3-Level schema Architecture

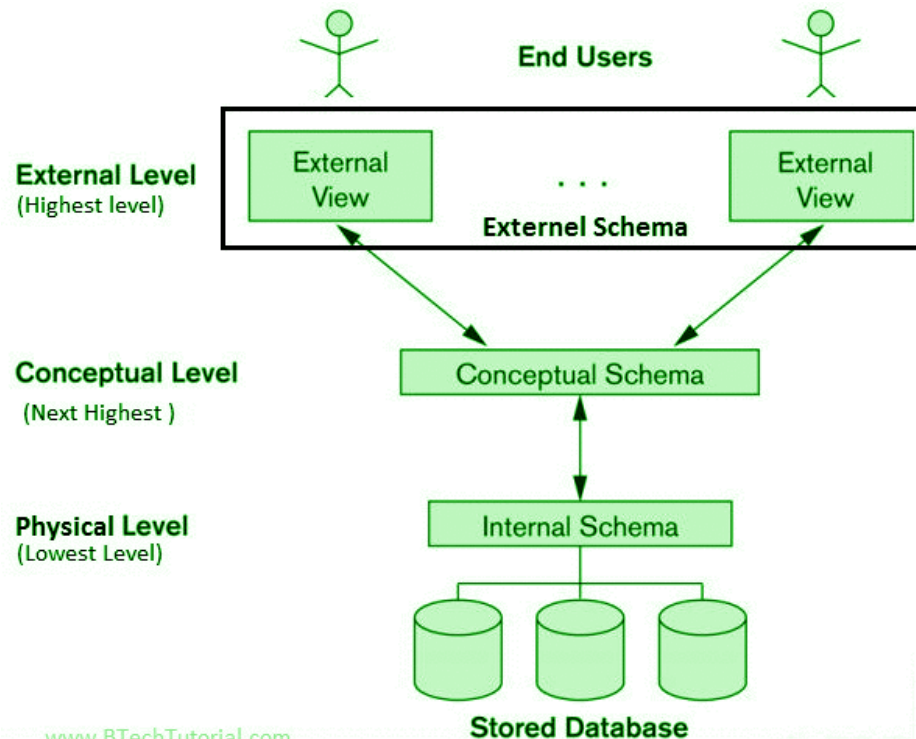
- The three schema architecture is also called ANSI/SPARC architecture or three-level architecture.
- This framework is used to describe the structure of a specific database system.
- The three schema architecture is also used to separate the user applications and physical database. The three schema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



# 3-Level schema Architecture

Figure 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.



# 3-Level schema Architecture

## ***Internal level:***

- This is the lowest level of data abstraction.
- It describes How the data are actually stored on storage devices.
- It is also known as physical level.
- It provides internal view of physical storage of data.
- It deals with complex low level data structures, file structures and access methods in detail.
- It also deals with Data Compression and Encryption techniques, if used.

## ***Conceptual level:***

- This is the next higher level than internal level of data abstraction.
- It describes What data are stored in the database and What relationships exist among those data.
- It is also known as Logical level.
- It hides low level complexities of physical storage.
- Database administrator and designers work at this level to determine What data to keep in database.
- Application developers also work on this level.

# 3-Level schema Architecture

## ***External Level:***

- This is the highest level of data abstraction.
- It describes only part of the entire database that a end user concern.
- It is also known as an view level.
- End users need to access only part of the database rather than entire database.
- Different user need different views of database. And so, there can be many view level abstractions of the same database.

## **Advantages of 3-level schema Architecture:**

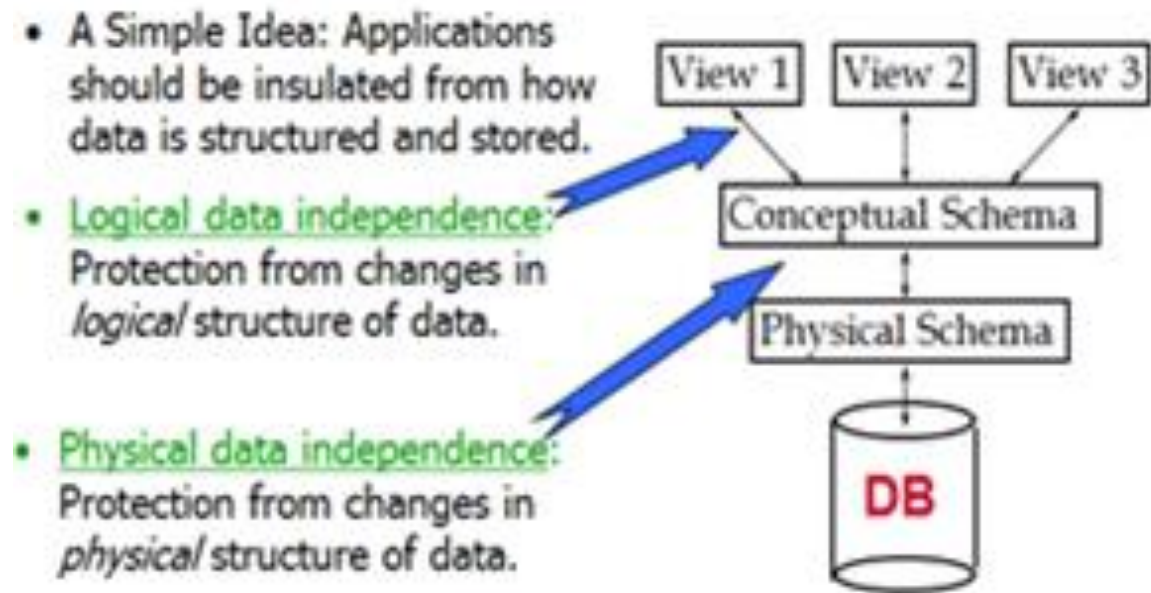
- The main objective of it is to provide data abstraction.
- Same data can be accessed by different users with different customized views.
- The user is not concerned about the physical data storage details.
- Physical storage structure can be changed without requiring changes in internal structure of the database as well as users view.
- Conceptual structure of the database can be changed without affecting end users.

# Data Independence

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

Data independence can be explained using the three-schema architecture.

Data independence refers characteristic of being able to modify the schema at one level of the database system without altering the schema at the next higher level.





# Data Independence

## Logical Data Independence

- ☐ Logical data is data about database, that is, it stores information about how data is managed inside.
- ☐ Logical data independence is used to separate the external level from the conceptual view.
- ☐ If we do any changes in the conceptual view of the data, then the user view of the data would not be affected.
- ☐ Logical data independence occurs at the user interface level.
- ☐ For example, a table (relation) stored in the database and all its constraints, applied on that relation.
- ☐ Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

# Data Independence

## Physical Data Independence

- ❑ All the schemas are logical, and the actual data is stored in bit format on the disk. Physical data independence is the power to change the physical data without impacting the schema or logical data.
- ❑ If we do any changes in the storage size of the database system server, then the Conceptual structure of the database will not be affected.
- ❑ Physical data independence is used to separate conceptual levels from the internal levels.
- ❑ Physical data independence occurs at the logical interface level.
- ❑ For example, in case we want to change or upgrade the storage system itself – suppose we want to replace hard-disks with SSD – it should not have any impact on the logical data or schemas.

# Data Abstraction

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..***

**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.

# Data Abstraction

**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.

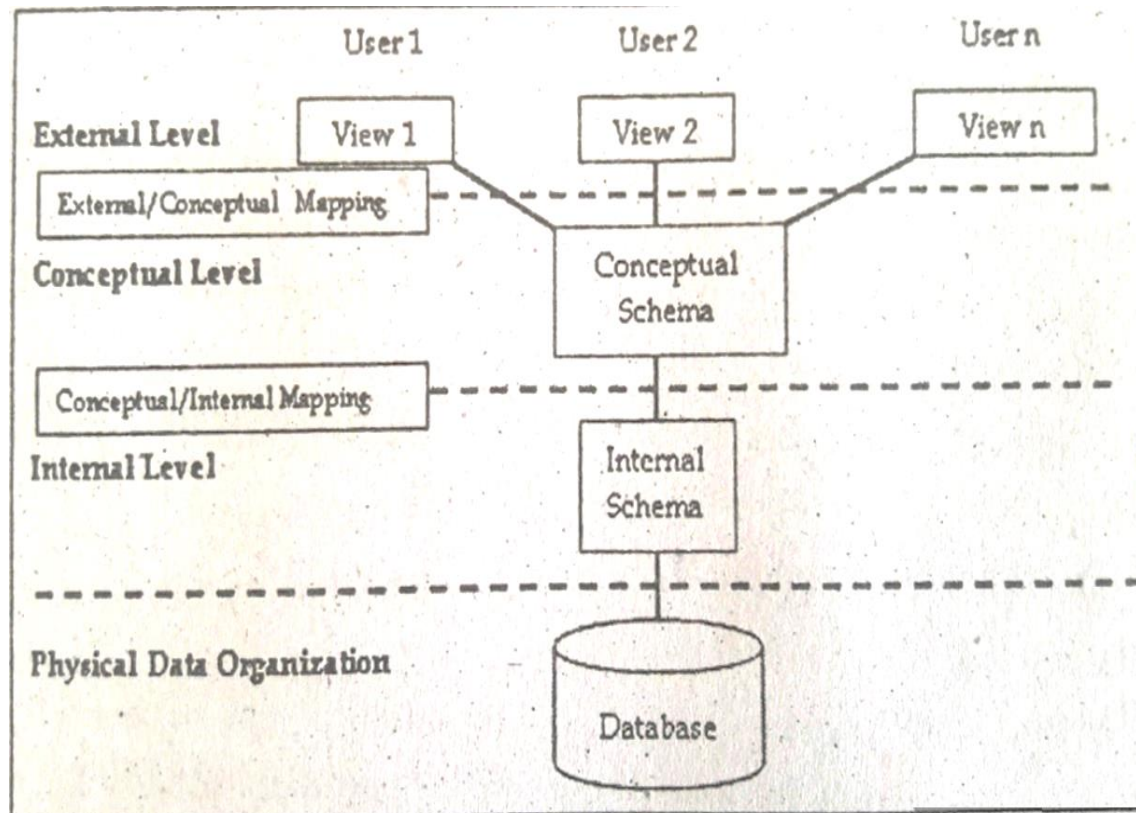
At **view level**, user just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

# Mapping

Mapping is a process of converting one level to another level. In this process, the data in one level is related to the data at another level.

**There are two level of Mapping:**

- ❑ From the Conceptual Level to Internal Level
- ❑ From the External Level to Conceptual Level



# Mapping

## Conceptual to Internal Mapping

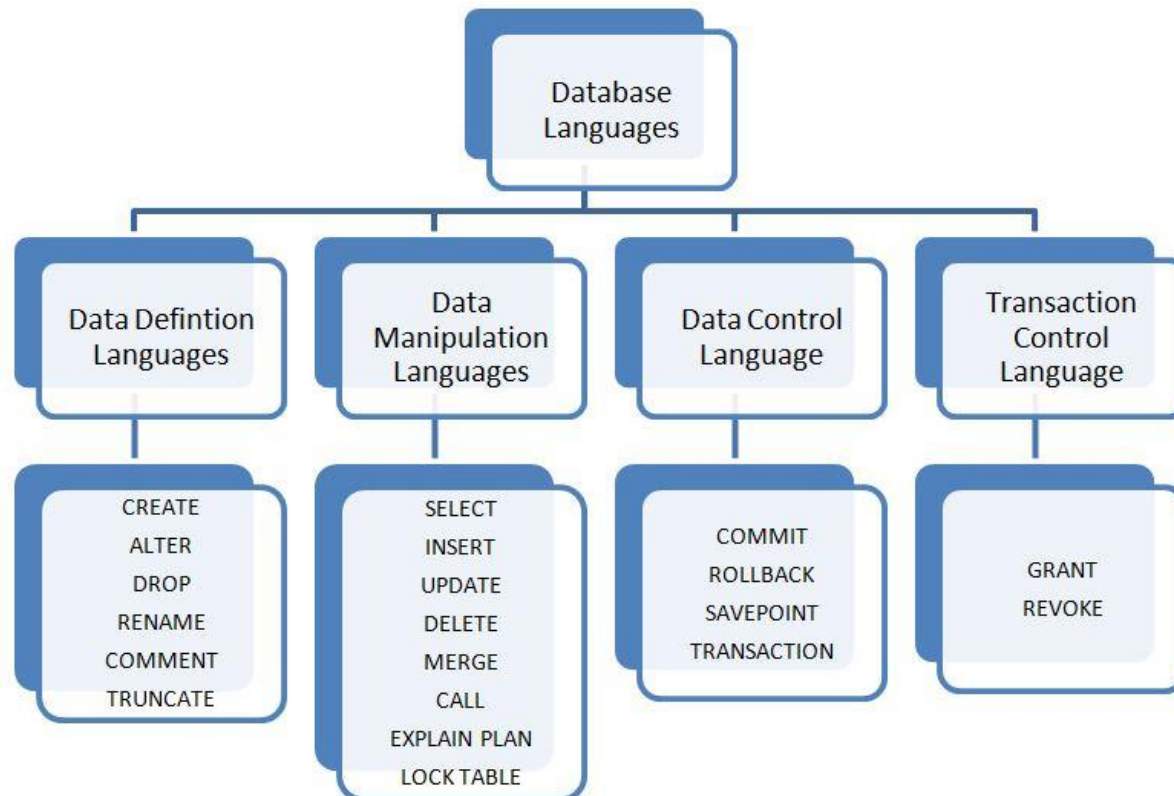
- It defines the correspondence between the conceptual view and stored data.
- It specifies that how conceptual records and fields are represented at internal level.
- If the structure of stored data is changed, then conceptual-internal mapping must be changed.
- It is the responsibility of DBA to manage such changes.

## External to Conceptual Mapping

- It defines the correspondence between a particular external views and conceptual view.
- The difference between these two levels is similar to the difference between Conceptual Level and Internal Level.

# Database Languages

A DBMS must provide appropriate languages and interfaces for each category of users to express database queries and updates. Database languages are used to read, update and store data in a database. There are large numbers of database languages like Oracle, MySQL, MS Access, dBase, FoxPro etc. SQL (Structured Query Language) is commonly used in Oracle and MS Access can be categorized as data definition language (DDL), data control language (DCL) and data manipulation language (DML).



# Database Languages

## Data Definition Language (DDL)

DDL is used for specifying the database schema. It is used for creating tables, schema, indexes, constraints etc. in database. Lets see the operations that we can perform on database using DDL:

- To create the database instance – **CREATE**
- To alter the structure of database – **ALTER**
- To drop database instances – **DROP**
- To delete tables in a database instance – **TRUNCATE**
- To rename database instances – **RENAME**
- To drop objects from database such as tables – **DROP**
- To Comment – **Comment**

All of these commands either defines or update the database schema that's why they come under Data Definition language.



# Database Languages

## Data Manipulation Language (DML)

DML is used for accessing and manipulating data in a database. The following operations on database comes under DML:

- To read records from table(s) – **SELECT**
- To insert record(s) into the table(s) – **INSERT**
- Update the data in table(s) – **UPDATE**
- Delete all the records from the table – **DELETE**

## Data Control language (DCL)

DCL is used for granting and revoking user access on a database –

- To grant access to user – **GRANT**
- To revoke access from user – **REVOKE**

# Database Languages

## Transaction Control Language (TCL)

The changes in the database that we made using DML commands are either performed or roll backed using TCL.

- To persist the changes made by DML commands in database – **COMMIT**
- To rollback the changes made to the database – **ROLLBACK**

In practical data definition language, data manipulation language and data control languages are not separate language, rather they are the parts of a single database language such as SQL.

# Different types of Database Users

Any person who uses the database and takes the benefits from the Database is considered as Database User. They can be programmer, scientist, engineers, businessman or can be any employee.

## Database Administrator

- It is a person or team, who is responsible for overall management of the Database Management System design.
- It's a leader of the database; it is also like the superuser of the system.
- It is responsible for the administration of all 3 levels of database.
- He has all the privileges on the database; he can also assign or remove the privileges from the other database users.
- Each database requires at least one database administrator (DBA). A Database system can be large and can have many users. Therefore, database administration is sometimes not a one-person job, but a job for a group of DBAs who share responsibility.

# Different types of Database Users

- A database administrator's responsibilities can include the following tasks:
  - ✓ Installing and upgrading the Oracle Database server and application tools
  - ✓ Allocating system storage and planning future storage requirements for the database system
  - ✓ Creating primary database storage structures (tablespaces) after application developers have designed an application
  - ✓ Creating primary objects (tables, views, indexes) once application developers have designed an application
  - ✓ Modifying the database structure, as necessary, from information given by application developers
  - ✓ Enrolling users and maintaining system security
  - ✓ Ensuring compliance with Oracle license agreements
  - ✓ Controlling and monitoring user access to the database
  - ✓ Monitoring and optimizing the performance of the database
  - ✓ Planning for backup and recovery of database information
  - ✓ Maintaining archived data on tape
  - ✓ Backing up and restoring the database
  - ✓ Contacting Oracle for technical support

# Different types of Database Users

## 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 an embedded SQL query in the C Program.

## Sophisticated Users

- They interact with the system without writing programs.
- They form requests by writing queries in a database query language. These are submitted to a **query processor** that breaks a DML statement down into instructions for the database manager module.
- 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.

# Different types of Database Users

## Specialized 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. These may be CADD systems, knowledge-based and expert systems, complex data systems (audio/video), etc.

## Native 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.

# Different types of Database Users

## System Analyst

- He/she analyses the requirements of the end users, especially naïve users and parametric end users. They are responsible for the design, structure and properties of the database.
- The main concerns of the system analyst is on feasibility, economic aspects and technical aspects.
- Analysts are one among the sophisticated users. They use the tools to perform their task such as:
  1. **Online analytical processing (OLAP)** - It helps the analysts to view them the summaries of the data in different ways.
  2. **Data Mining Tools** – It helps the analysts find a certain kind of pattern in the given data.

# File Organizations

The File Organization is the physical organization of the records of a file for the convenience of storage and retrieval of data records. System designer basically choose to organize, access and process the records of the various files in different ways, depending upon the type of the application and needs of the uses are:

1. Ease of Retrieval
2. Convenience of Updates
3. Economy of Storage
4. Reliability
5. Security
6. Integrity

There are some commonly used file organizations are-

- A. Serial File
- B. Sequential File
- C. Index Sequential File
- D. Direct File
- E. Multilist File
- F. Tree based File



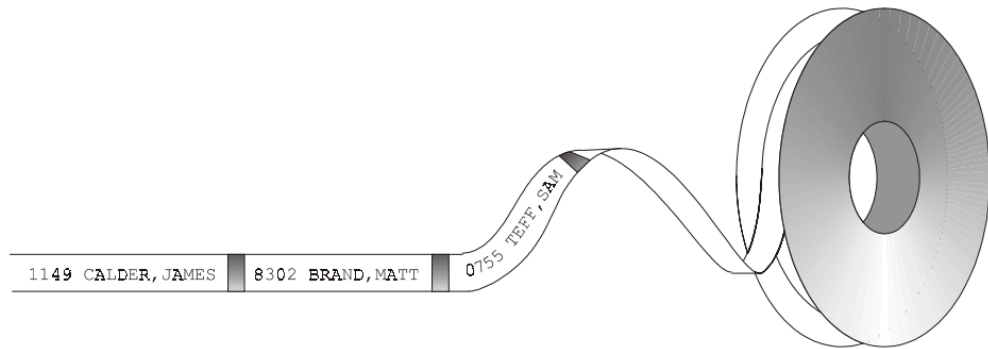
# File Organizations

## Serial File

- Records in a file are stored and accessed one after another.
- The records are not stored in any way on the storage medium this type of organization is mainly used on **magnetic tapes**.
- The records on a serial file are not in any particular sequence, and so this type of organisation would not be used for a master file as there would be no way to find a particular record except by reading through the whole file, starting at the beginning, until the right record was located. Serial files are used as temporary files to store transaction data.

## Advantages

- It is simple
- It is cheap



## Disadvantages

- It is cumbersome to access because you have to access all proceeding records before retrieving the one being searched.
- Wastage of space on medium in form of inter-record gap.
- It cannot support modern high speed requirements for quick record access.

# File Organizations

## Sequential File

- Storing and sorting in contiguous block within files on tape or disk is called as **sequential access file organization**.
- In sequential access file organization, all records are stored in a sequential order. The records are arranged in the ascending or descending order of a key field.
- Sequential file search starts from the beginning of the file and the records can be added at the end of the file.
- In sequential file, it is not possible to add a record in the middle of the file without rewriting the file.
- Searching of a record requires, on average, access to half the records in the file.

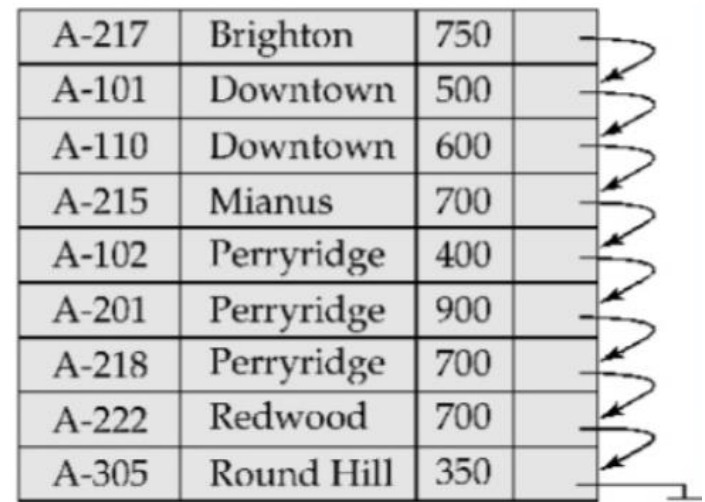
## Advantages

- It is simple to program and easy to design.
- Sequential file is best use if storage space.

## Disadvantages

- Sequential file is time consuming process.
- It has high data redundancy.
- Random searching is not possible.

A-217	Brighton	750	
A-101	Downtown	500	
A-110	Downtown	600	
A-215	Mianus	700	
A-102	Perryridge	400	
A-201	Perryridge	900	
A-218	Perryridge	700	
A-222	Redwood	700	
A-305	Round Hill	350	



# File Organizations

## Direct File or Random Access File

- Direct access file is also known as random access or relative file organization.
- In direct access file, all records are stored in direct access storage device (DASD), such as hard disk. The records are randomly placed throughout the file.
- The records does not need to be in sequence because they are updated directly and rewritten back in the same location.
- This file organization is useful for immediate access to large amount of information. It is used in accessing large databases.
- It is also called as hashing.

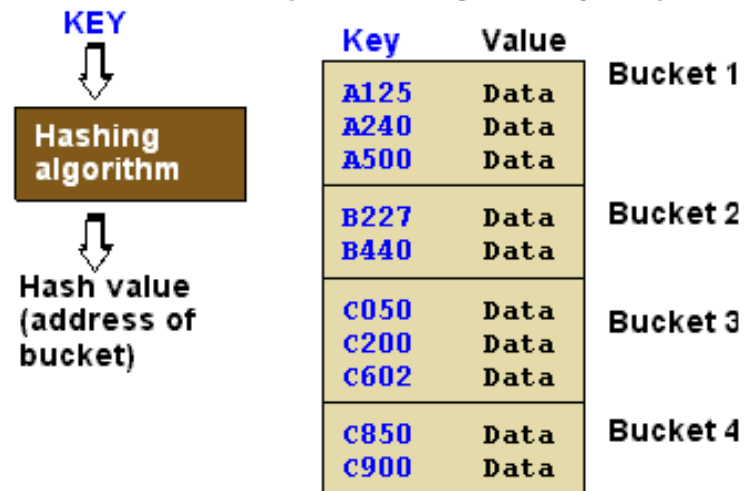
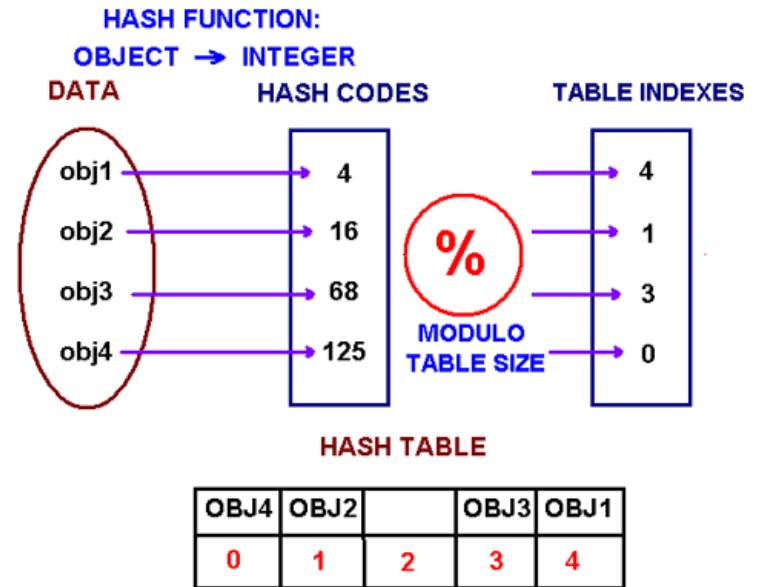
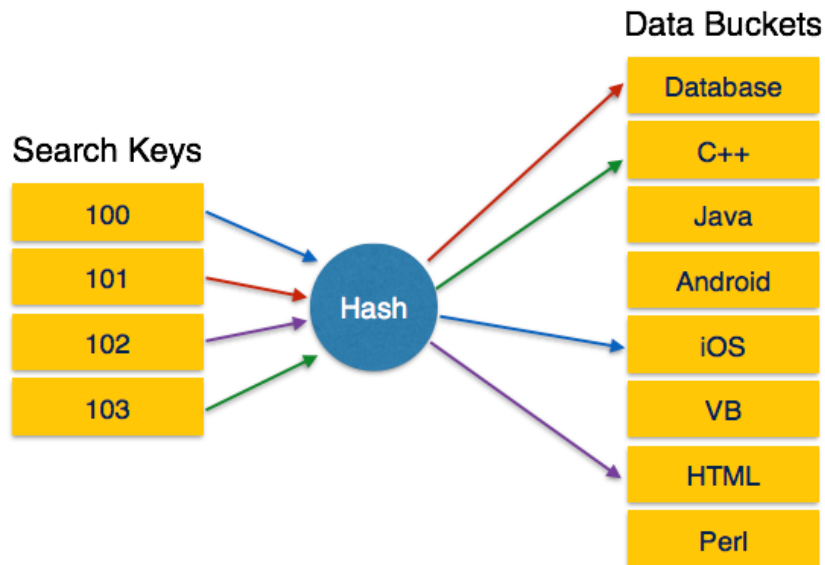
## Advantages

- In direct access file, sorting of the records are not required.
- It accesses the desired records immediately.
- It updates several files quickly.
- It has better control over record allocation.

## Disadvantages

- Direct access file does not provide back up facility.
- It is expensive.
- It has less storage space as compared to sequential file.

# File Organizations



# File Organizations

## Indexed Sequential File

An indexed sequential file consists of records that can be accessed sequentially. Direct access is also possible. It consists of two parts –

- **Data File** contains records in sequential scheme.
- **Index File** contains the primary key and its address in the data file.

Following are the key attributes of sequential file organization –

- Records can be read in sequential order just like in sequential file organization.
- Records can be accessed randomly if the primary key is known. Index file is used to get the address of a record and then the record is fetched from the data file.
- Sorted index is maintained in this file system which relates the key value to the position of the record in the file.
- Alternate index can also be created to fetch the records.

# File Organizations

## Indexed Sequential File

Partial index

bingham	5
callendar	10
	15
..	..

Main file

#	name	tutor	sex
0	ashok	Ebo	m
1	aldham	Ebo	m
2	amdhal	Okl	f
3	azerty	Ebo	m
4			
5	bingham	Okl	f
6	bjalko	Okl	f
7	blantyre	Jhl	m
8	brambell	Ftr	f
9	byzantium	Jhl	m
10	callendar	Ebo	m
..	..	..	..

Block 1

Block 2

Block 3

Data Records

R1	AA6DK
R2	BS8KA
R5	SA7VD
R7	DS46G
R8	XS5GF

Data Blocks in memory

DS46G
XS5GF
BS8KA
DH4FD
AA6DK

R9	DH4FD
----	-------

SA7VD
-------

# File Organizations

## Advantages

- Sequential file and random file access is possible.
- It accesses the records very fast if the index table is properly organized.
- It provides quick access for sequential and direct processing.
- It reduces the degree of the sequential search.

## Disadvantages

- Indexed sequential access file requires unique keys and periodic reorganization.
- It requires more storage space.
- It is expensive because it requires special software.
- It is less efficient in the use of storage space as compared to other file organizations.