Characteristics of the Database Approach

- In the database approach, a single repository maintains data.
- It should be accessed by various users repeatedly through queries, transactions, and application programs.
- The main characteristics of the database approach are:
  - Self-describing nature of a database system
  - Insulation between programs and data, and data abstraction
  - Support of multiple views of the data
  - Sharing of data and multiuser transaction processing

Self-Describing Nature of a Database System

- A fundamental characteristic of the database approach is
  - It should contain not only the database itself but also contains its complete definition.
- The definition contains
  - The structure of each file, the type and storage format of each data item, and various constraints on the data.
  - This information is called meta-data.

Example of a Database:

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Self-Describing Nature

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major_type is defined as an enumerared type with all known majors. XXXXNNNN is used to define a type with four alpha characters followed by four digits

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Insulation between Programs and Data

- In traditional file processing, the structure of data files is embedded in the application programs.
- So any changes to the structure of a file may require changing all programs that access that file.
- In DBMS, structure of data files is stored separately from the access programs.

Data Abstraction

- An operation (also called a function or method) is specified in two parts.
- The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters).
- The implementation (or method) of the operation is specified separately and can be changed without affecting the interface.
- User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented.
- This may be termed **program-operation independence**.
- The characteristic that allows **program-data independence** and **program-operation independence** is called **data abstraction**.
- A DBMS provides users with a conceptual representation of data that does not include many of the details of how the data is stored or how the operations are implemented.

Support of Multiple Views of the Data

- A database typically has many types of users, each of whom may require a different perspective or view of the database.
- A view may be a subset of the database or it may contain virtual data that is derived from the database files.
- But this view is not explicitly stored.

Sharing of Data and Multiuser Transaction Processing

- It must allow multiple users to access the database at the same time.
- It is essential if data for multiple applications is to be operated on a single database.
- For this, DBMS must include concurrency control software.

Advantages of DBMS over File System

- No Redundant Data: Redundancy removed by data normalization. No data duplication saves storage and improves access time.
- Data Consistency and Integrity: The root cause 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 authorised user is able to access it. Each user has a different set of access thus data is secured form 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 manage data in such a way so that the data is easily accessible with fast response times.
- Easy recovery: Since database systems keep the backup of data, its easier to do a full recovery of data in case of failure.
- Flexible: Database systems are more flexible than file processing systems.

Basis	File System	DBMS
1. Structure	File system is a software that manages and organizes the files in a storage medium within a computer.	DBMS is a software for managing the database.
2. Data Redundancy	Redundant data can be present in a file system.	In DBMS there is no redundant data.
3.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.
4. Query processing	There is no efficient query processing in file system.	Efficient query processing is there in DBMS.
5.Consistency	There is less data consistency in file system.	There is more data consistency because of the process of normalization.
6. Complexity	It is less complex as compared to DBMS.	It has more complexity in handling as compared to file system.
7.Security Constraints	File systems provide less security in comparison to DBMS.	DBMS has more security mechanisms as compared to file system.
8.Cost	It is less expensive than DBMS.	It has a comparatively higher cost than a file system.
9. Data Independence	There is no data independence.	In DBMS data independence exists

MCQ

- Data Isolation caused due to _____in traditional file system.
  1. Duplicate Data
  2. Scattering of Data
  3. Complex Data
  4. Atomic Data

MCQ

- Data Isolation caused due to _____in traditional file system.
  1. Duplicate Data
  2. Scattering of Data
  3. Complex Data
  4. Atomic Data

MCQ

- If person A want to transfer fund of Rs.500 to person B. If failure occurs after removing Rs.500 from Account A and before transferring to Account B then problem caused is _____.
  1. Data Isolation
  2. Data Atomicity
  3. None of these
  4. Data Redundancy

MCQ

- If person A want to transfer fund of Rs.500 to person B. If failure occurs after removing Rs.500 from Account A and before transferring to Account B then problem caused is _____.
  1. Data Isolation
  2. Data Atomicity
  3. None of these
  4. Data Redundancy

MCQ

- Duplication of data at several places is called as _____.
  1. Data Inconsistency
  2. Atomicity Problem
  3. Data Isolation
  4. Data Redundancy

MCQ

- Duplication of data at several places is called as _____.
  1. Data Inconsistency
  2. Atomicity Problem
  3. Data Isolation
  4. Data Redundancy

MCQ

- If in redundant file common fields are not matching then it results in .
  1. Data Integrity Problem
  2. Data Isolation
  3. Data Redundancy
  4. Data Inconsistency

MCQ

- If in redundant file common fields are not matching then it results in .
  1. Data Integrity Problem
  2. Data Isolation
  3. Data Redundancy
  4. Data Inconsistency

MCQ

- It is difficult to access conventional file system than Database System.
  1. True
  2. False

MCQ

- It is difficult to access conventional file system than Database System.
  1. True
  2. False

MCQ

- Suppose user have Saving Account and Checking Account in the Bank.

  Saving Account Stores following information -

  account-no
  name
  address
  mobile

  and Checking Account stores –

  account-no
  name
  address
  Mobile

- Which of the information is not redundant. Data Inconsistency
  1. address
  2. name
  3. mobile
  4. account-no

MCQ

- Suppose user have Saving Account and Checking Account in the Bank.

  Saving Account Stores following information -

  account-no
  name
  address
  mobile

  and Checking Account stores –

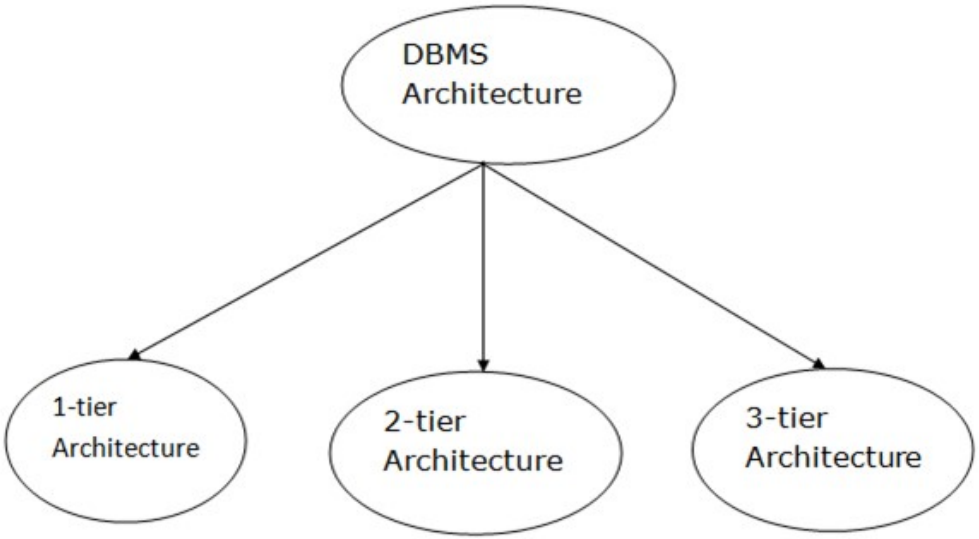
  account-no
  name
  address
  Mobile

- Which of the information is not redundant. Data Inconsistency
  1. address
  2. name
  3. mobile
  4. account-no

DBMS Architecture

- The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
- This architecture consists of many PCs and a workstation which are connected via the network.
- DBMS architecture depends upon how users are connected to the database to get their request done.

Types of DBMS Architecture:



Types of DBMS Architectures

1-Tier Architecture:

- The database is directly available to the user.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

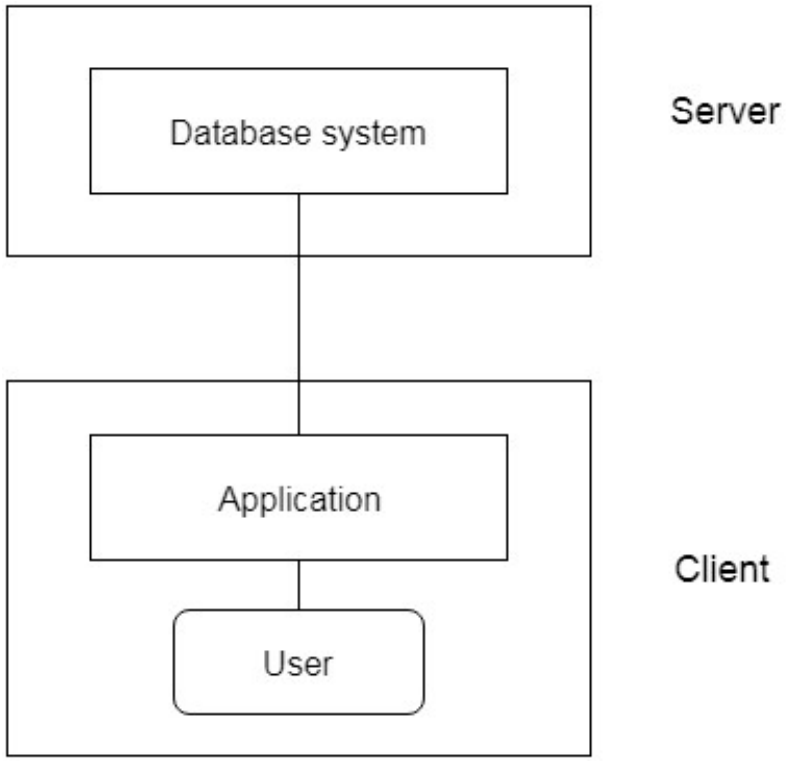
2-Tier Architecture:

- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
- 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 the DBMS, client-side application establishes a connection with the server side.

Types of DBMS Architectures

2-Tier Architecture:

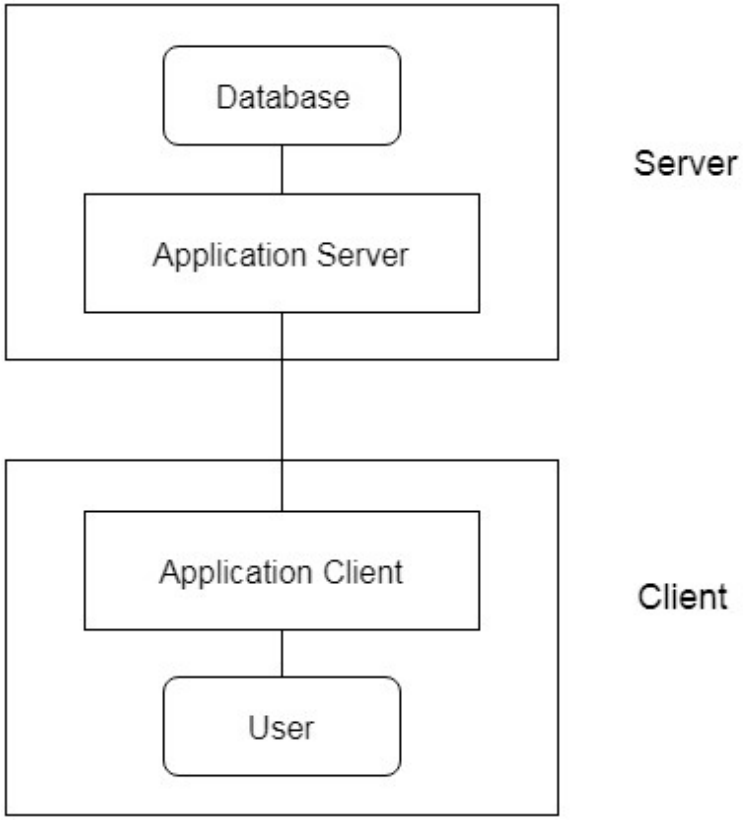
- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
- The user interfaces and application programs are run on the client-side. The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with the DBMS, client-side application establishes a connection with the server side.



Types of DBMS Architectures

3-Tier Architecture:

- The 3-Tier architecture contains another layer between the client and server. Here, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.



Simplified database system environment

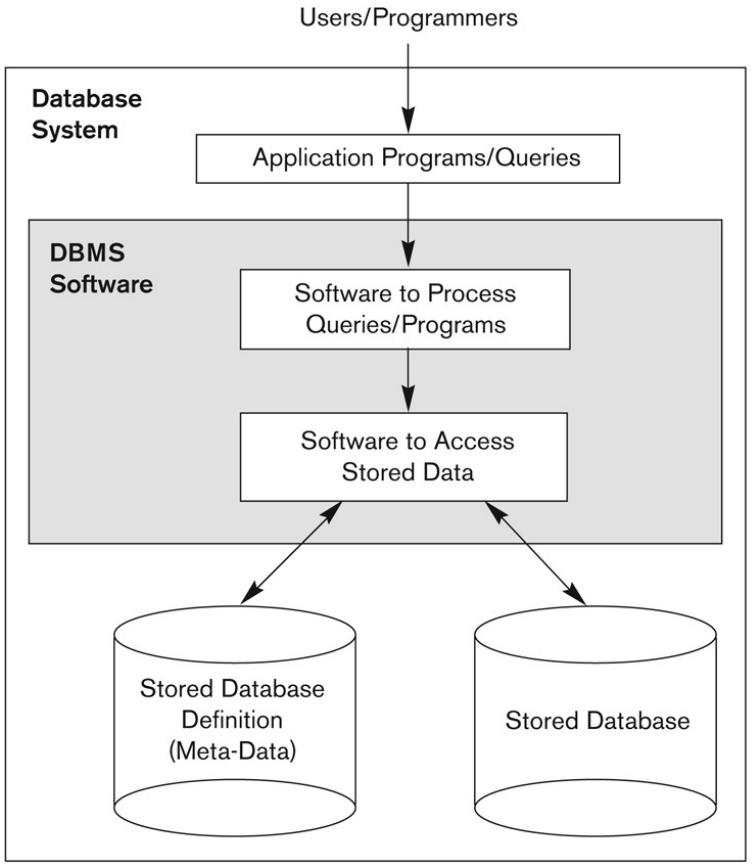


Figure 1.1
A simplified database system environment.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Data Base Schemas, Schema Architecture, Data Models

Database Schema

- It is important to distinguish between the *description* of the database and the *database itself*.
- The description of a database is called the database schema.
- Includes descriptions of the database structure, data types, and the constraints on the database.
- It is specified during database design and is not expected to change frequently.
- Each object in the schema—such as STUDENT or COURSE— called a schema construct.

Example of a simple database

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Example of a Database Schema

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Figure 2.1
Schema diagram for the database in Figure 1.2.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Database State or Instance

- The data in the database at a particular moment in time is called a **database state** or **snapshot.**
- It changes every time we add new data in database.
- It is also called the *current* set of **occurrences** or **instances** in the database.
  - The term *instance* is also applied to individual database components, e.g. *record instance, table instance, entity instance.*
- A state that satisfies the structure and constraints of the database is called valid state.

Example of a database state

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

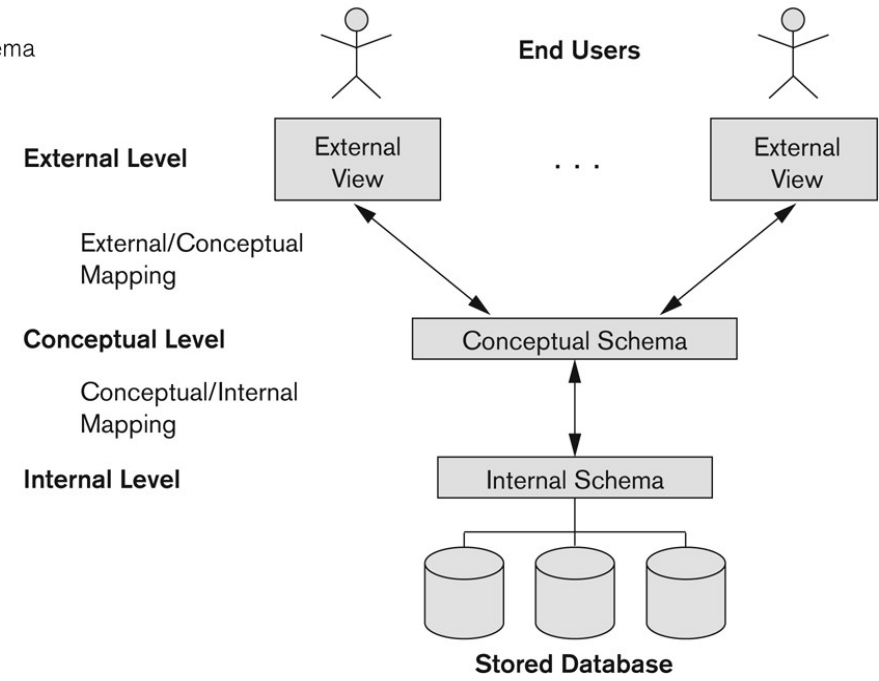
Figure 1.2
A database that stores student and course information.

Ramez Elmasri and Shamkant B. Navathe. 1989. Fundamentals of database systems. Benjamin-Cummings Publishing Co., Inc., USA.

Three-Schema Architecture

- **Internal schema** describes the physical storage structure and access paths.
  - Typically uses a **physical** data model.
- **Conceptual schema** describes the structure and constraints for the whole database for different users.
  - The conceptual schema hides the details of physical storage structures and focuses on describing entities, data types, relationships, constraints, etc.
  - Uses a **conceptual** or an **implementation** data model.
- **External schemas** (view schemas) describes the part of the database that a particular user is interested in and hides the remaining database from that user group.
  - The view schema describes the end user interaction with database systems.
  - Usually uses the same data model as the conceptual schema.

Figure 2.2
The three-schema architecture.



Three-Schema Architecture

- Mappings among schema levels are needed to transform requests and data.
- Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution.
- Data extracted from the internal schema is reformatted to match the user's external view. E.g. formatting the results of an SQL query for display in a Web page.

Data Independence

The ability to modify the schema at one level of the database system without altering the schema at the next higher level.

- **Logical Data Independence:**
  - The capacity to change the conceptual schema without having to change the external schemas and their associated application programs.
- **Physical Data Independence:**
  - The capacity to change the internal schema without having to change the conceptual schema.
    - For example, the internal schema may be changed when certain file structures are reorganized or new indexes are created to improve database performance.
    - Then only the **mappings** between this schema and higher-level schemas need to be changed in a DBMS.
    - The application programs need not to be changed since they refer to the external schemas.

Data Abstraction

- An operation (also called a function or method) is specified in two parts.
- The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters).
- The implementation (or method) of the operation is specified separately and can be changed without affecting the interface.
- User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented.
- This may be termed **program-operation independence**.
- The characteristic that allows **program-data independence** and **program-operation independence** is called **data abstraction**.
- A DBMS provides users with a conceptual representation of data that does not include many of the details of how the data is stored or how the operations are implemented.

View of data 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 the users and provides abstract view of data to users. This process of hiding irrelevant details from user is called Data Abstraction.
- Abstraction is one of the main features of database systems.

Levels of Abstraction

- **Physical Level (Internal 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 (Conceptual Level):**
  - This is the middle level of 3-level data abstraction architecture.
  - It describes what data is stored in database and what relationships exist among those data.

- **View Level (External Level):**
  - Highest level of data abstraction.
  - This level describes only part of the entire database, i.e., it describes the user interaction with database system via application programs that hide details of data types.

Three levels of Abstraction

View Level:

- Highest level of abstraction
- User just interact with system using GUI. They are not aware of how the data is stored.

Logical Level:

- Deals with what data is stored, along with their data types, their relationship among each other.
- The programmers generally work at this level because they are aware of such things about database systems.
- Physical Data Independence.

Physical Level:

- Lowest Level of Abstraction
- Deals with how the data is stored in the memory.
- These details are often hidden from the programmers.
- Complex data structures