COURSE DETAILS

Course: Database Management System(DBMS)

Code : 18CS43

Sem : 4

Course learning objectives:

1. To discuss and realize the importance of Database Architecture Design notations, ER Modeling, Mapping and Schema design.
2. To gain the knowledge Relational algebra and learn the use of SQL and PL/SQL.
3. To introduce formal database design approach through normalization and discuss various normal forms.
4. To understand the importance of Concurrent Transactions and discuss issues and transaction control algorithms.

Course Outcome (Cos)

SNO	At the end of the course, the student will be able to,	Bloom's Level
1	**Apply** the database concepts and design database for given application scenerio	L3
2	**Apply** the concepts of Normalization and design database which eliminates all anomalies.	L3
3	**Create** database and develop database programming skills in SQL and PL/SQL.	L4
4	**Explain** the issue of concurrency control in transaction processing.	L2

Text Books:

1. Elmasri and Navathe: Fundamentals of Database Systems, Addison-Wesley, 3rd edition and onwards.

2. Raghu Ramakrishnan and Johannes Gehrke: Database Management Systems, McGraw-Hill, 2nd edition and onwards.

Reference Books:

1. Silberschatz, Korth and Sudharshan: Data base System Concepts, Mc-GrawHill, 3rd edition and onwards.

2. C.J. Date, A. Kannan, S. Swamynatham: A Introduction to Database Systems, Pearson education, 5th edition and onwards.

E Resources:

3. PL/SQL study material.

Scheme of Continuous Internal Evaluation (CIE):

Components	Addition of two IA tests	Average of two assignments	Quiz/Seminar/Course Project	Total Marks
Maximum marks |:50	15+15 = 30	10	10	50
Writing two IA tests is compulsory. Minimum marks required to qualify for SEE : 20 out of 50 marks				

Semester End Examination (SEE):

1. It will be conducted for 3 hours duration and 100 marks. It will be reduced to 50 marks for the calculation of SGPA and CGPA.
2. Minimum passing marks required to be scored in SEE: 40 out of 100 marks
3. Question paper will have 10 questions carrying 20 marks each. Students have to answer FIVE full questions selecting atleast one full question from each unit.

Introduction to Database Management System

Data: facts that can be recorded and stored

Database:

A database is a collection of related data.

A database can be of any size and complexity.

DBMS:

A database management system (DBMS) is a collection of programs that enables users to **create and maintain** a database.

The DBMS is a *general-purpose software system* that facilitates the processes of ***defining, constructing, manipulating, and sharing*** databases among various users and applications.

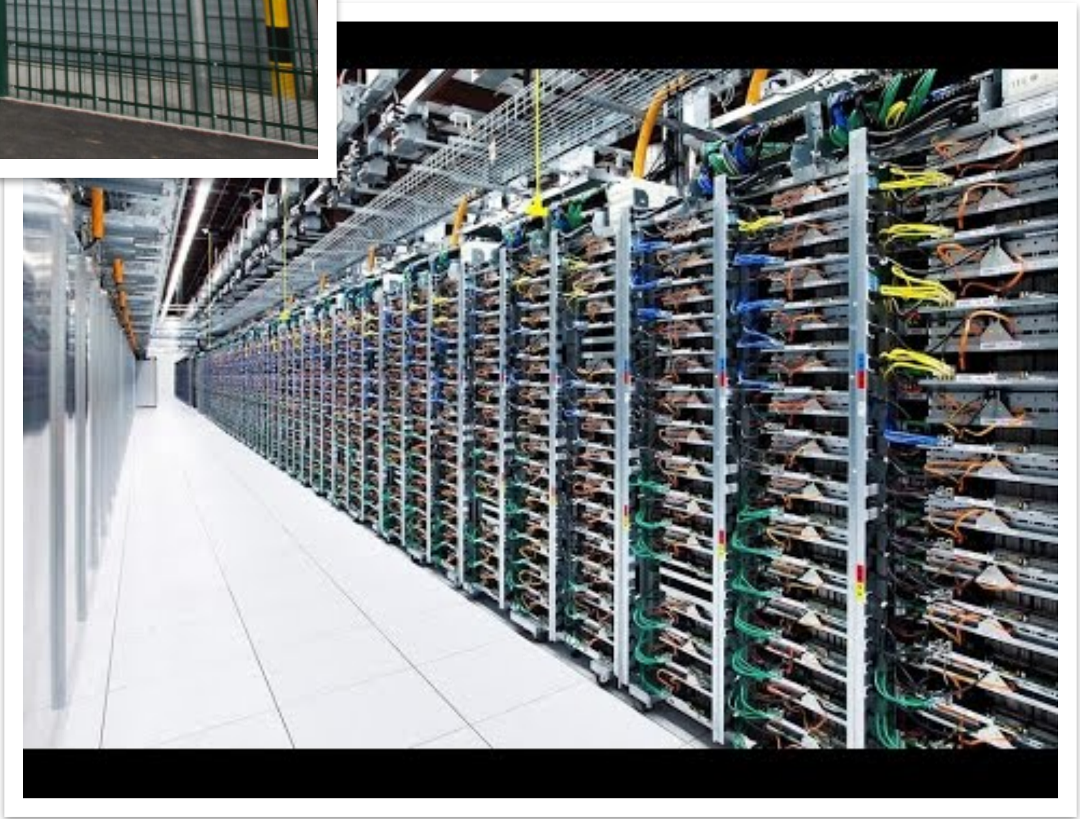
Database System:

The DBMS software together with the data itself is database system.

Create, Store, Maintain,& Share Data

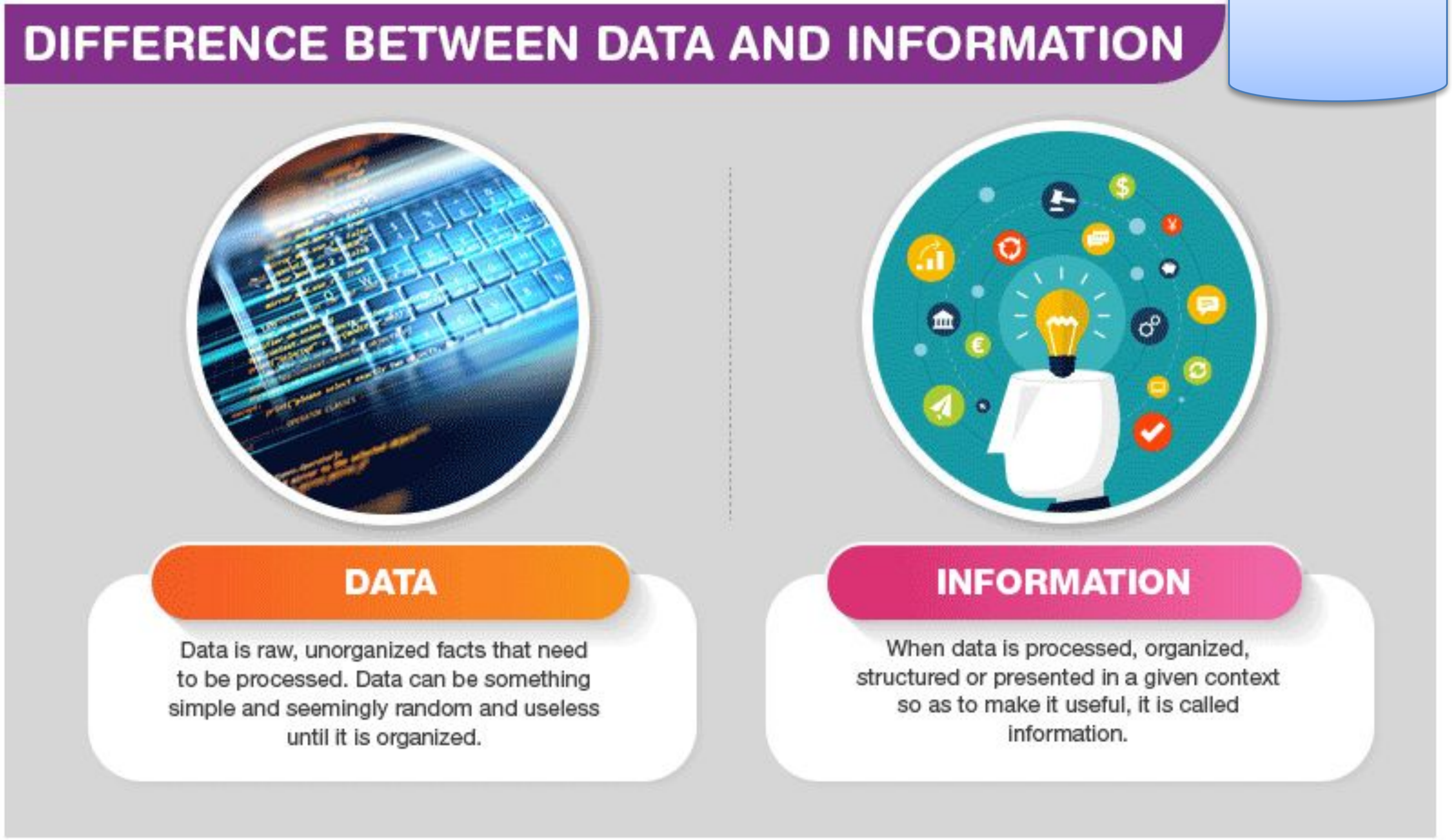
Then

Now



A database can be of any size and complexity.

Data vs. Information



DIFFERENCE BETWEEN DATA AND INFORMATION

DATA

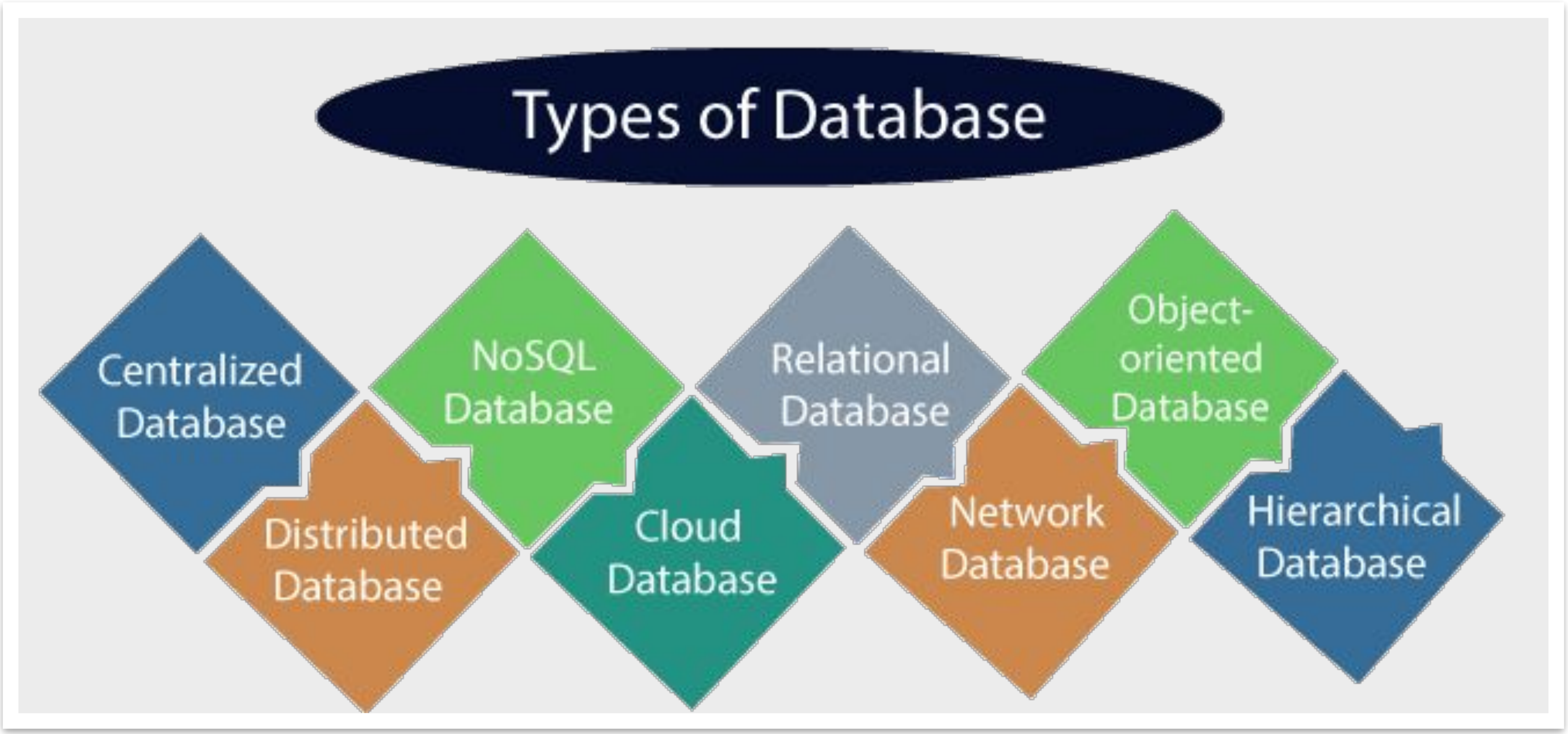
Data is raw, unorganized facts that need to be processed. Data can be something simple and seemingly random and useless until it is organized.

INFORMATION

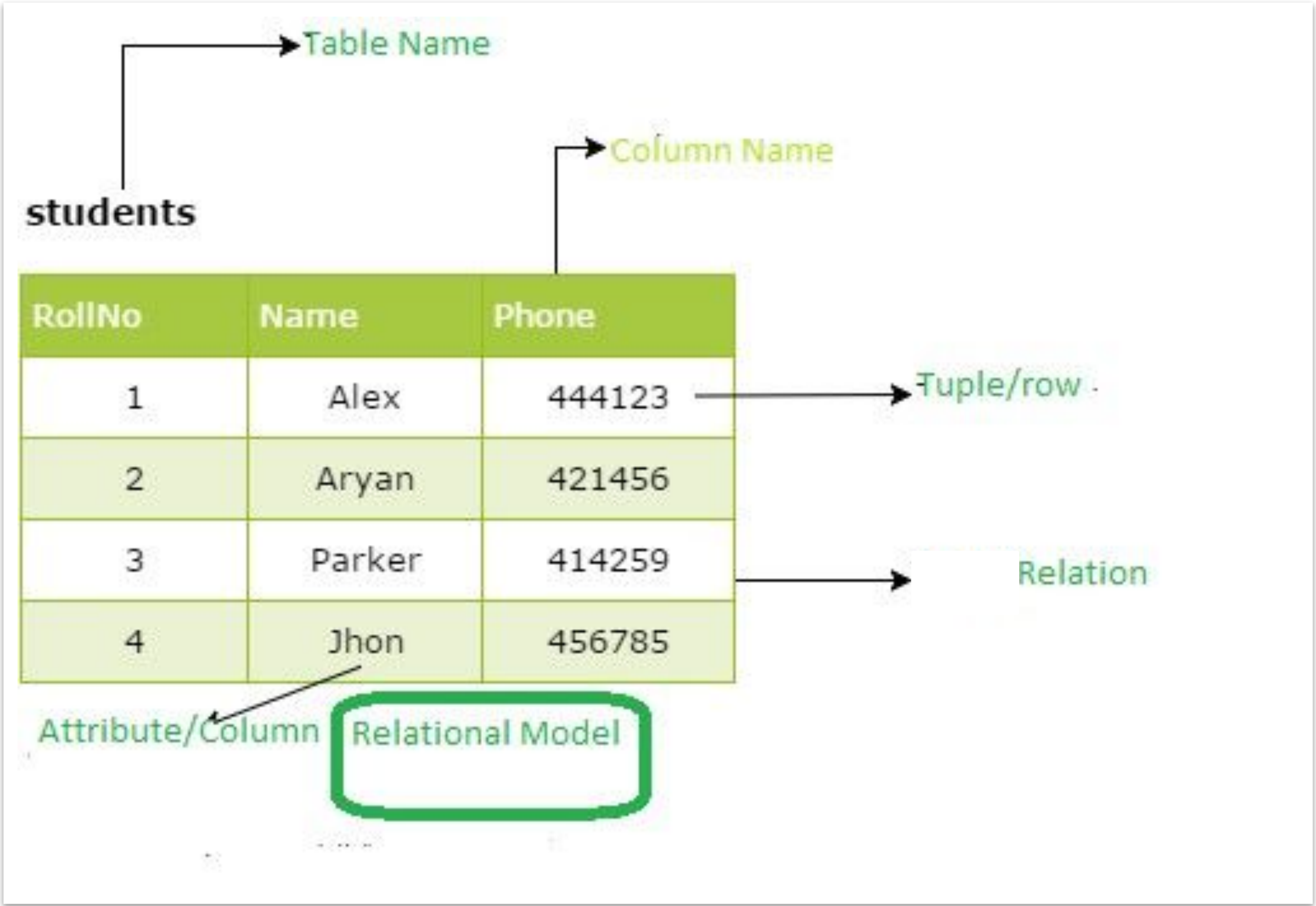
When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information.

List of Popular Databases

MariaDB Foundation
PostgreSQL
A
MySQL
mongoDB
IBM DB2
Microsoft SQL Server
ORACLE


Types of Database
Centralized Database
Distributed Database
NoSQL Database
Cloud Database
Relational Database
Network Database
Object-oriented Database
Hierarchical Database

Relational Database(RDBMS)



RollNo	Name	Phone
1	Alex	444123
2	Aryan	421456
3	Parker	414259
4	Jhon	456785

Typical DBMS Functionality

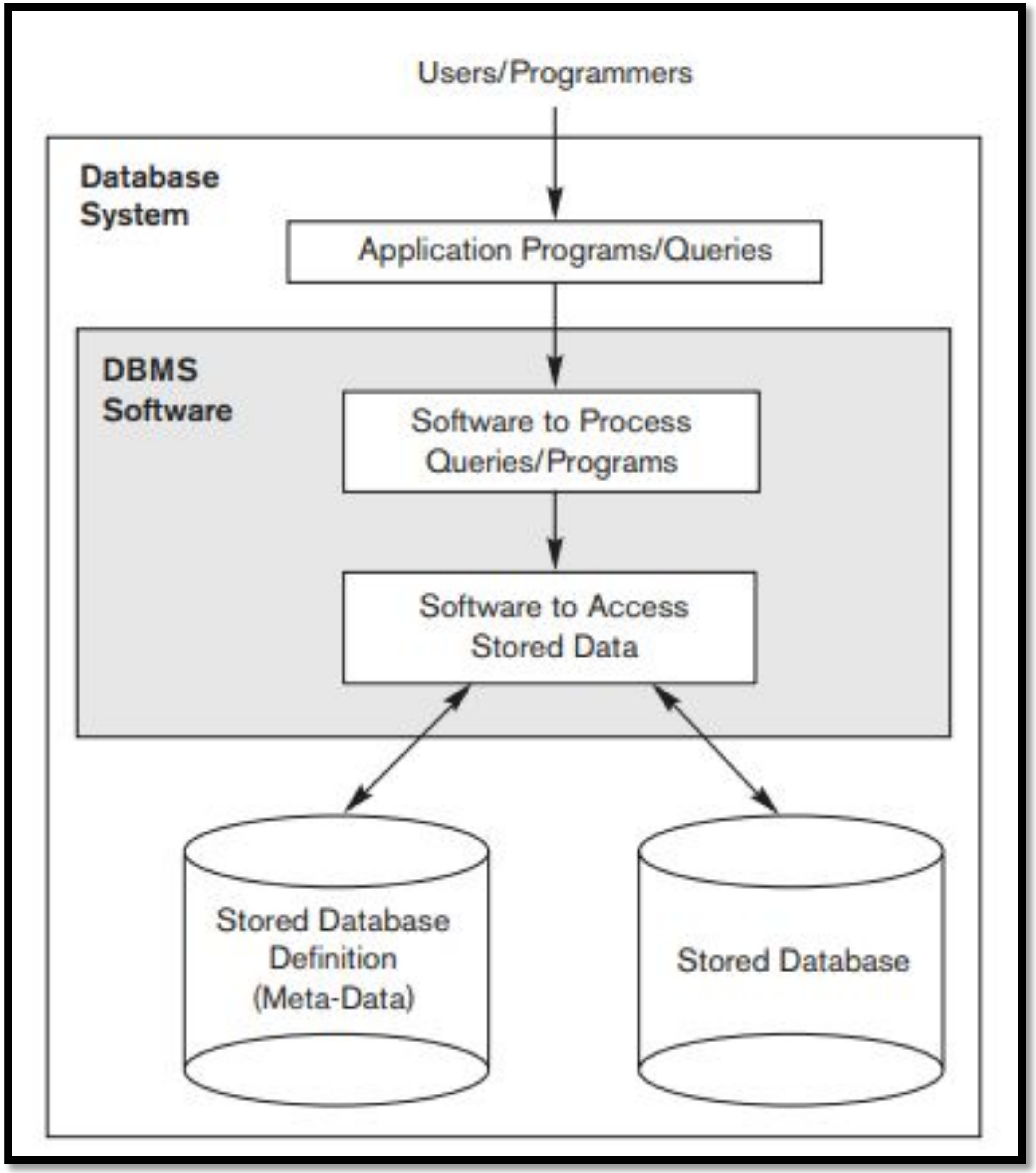
- **Define** a database : in terms of data types, structures and constraints
- **Construct** or Load the Database on a secondary storage medium
- **Manipulating** the database : querying, generating reports, insertions, deletions and modifications to its content
- **Concurrent Processing** and **Sharing** by a set of users and programs – yet, keeping all data valid and consistent

Typical DBMS Functionality

Other features:

- Protection or Security measures to prevent unauthorized access
- “Active” processing to take internal actions on data
- Presentation and Visualization of data to the end users

A Simplified Database System Environment.


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

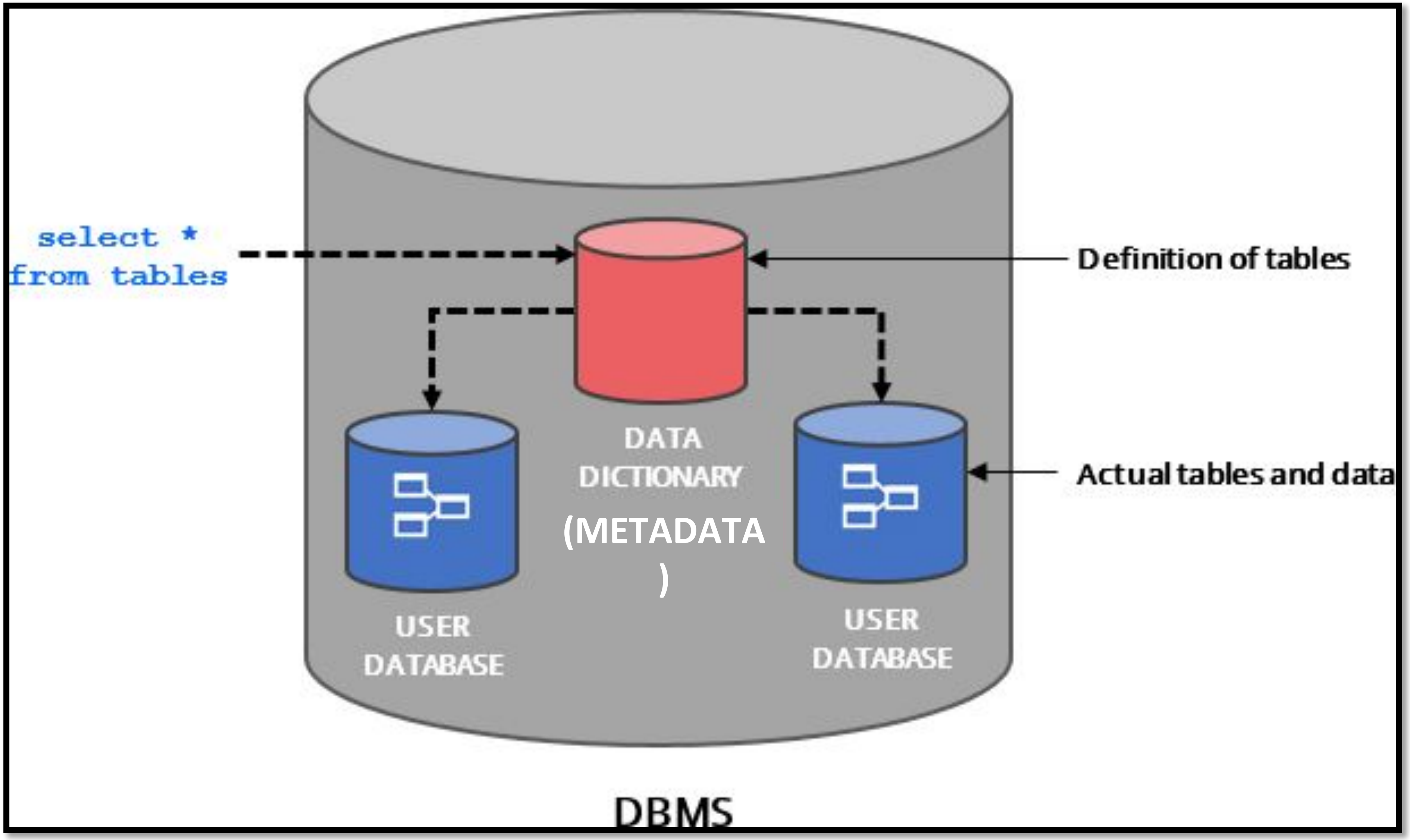
Characteristics of the Database Approach

1. Self-describing nature of a database system
2. Insulation between programs and data, and data abstraction
3. Support of multiple views of the data
4. Sharing of data and multiuser transaction processing

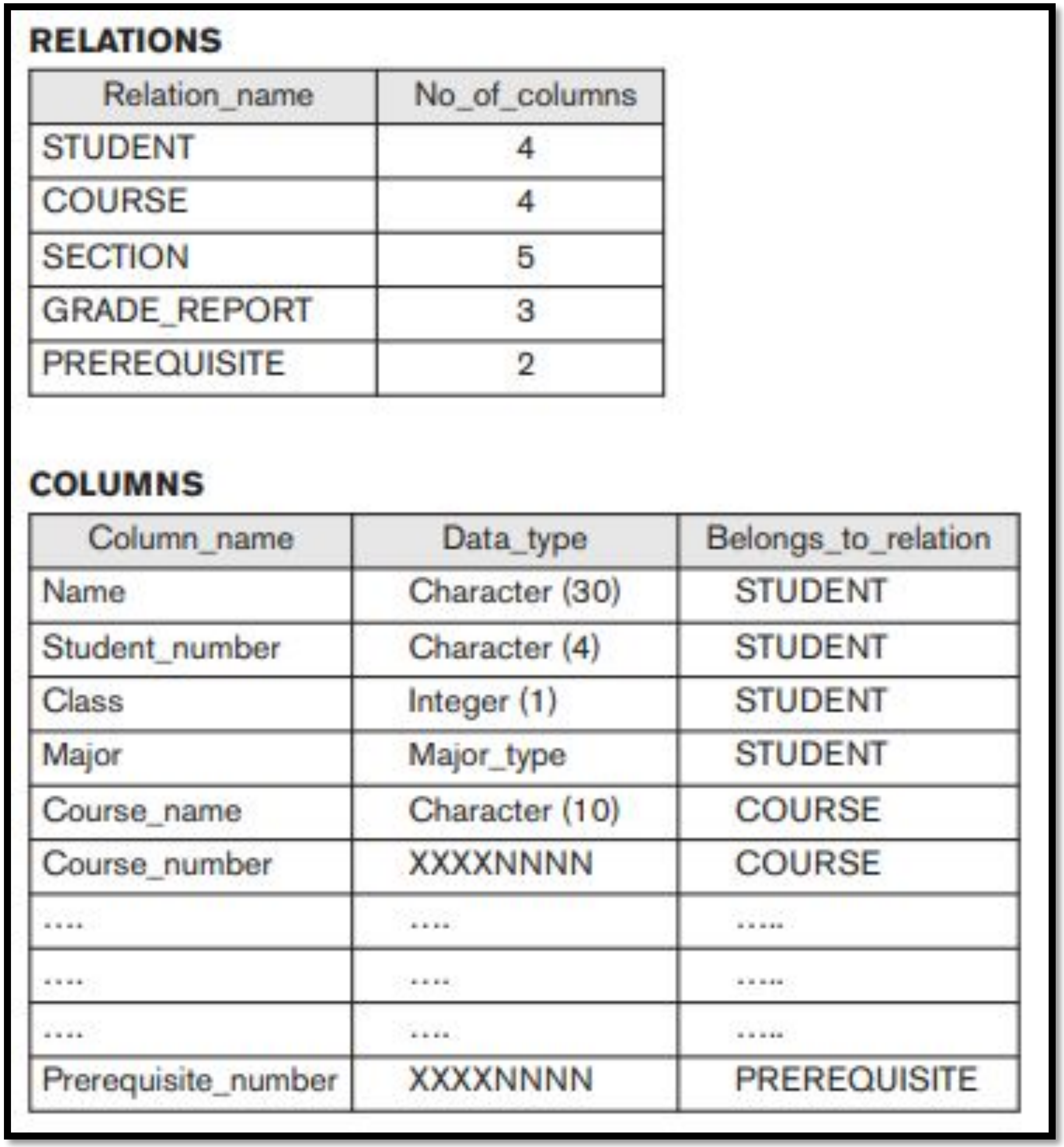
1. Self-describing nature of a database system

- The database system contains not only the database itself but also a complete definition or description of the database structure and constraints, which is known as **metadata**.
- This metadata is stored in **DBMS catalog** and is used by the DBMS software and also by database users who need information about the database structure.
- The DBMS software must work equally well with any number of database as long as the database definition is stored in the catalog.
- In traditional file processing, data definition is typically part of the application programs themselves. Hence, these programs are constrained to work with only one specific database.

1. Self-describing nature of a database system



An example of a database catalog for the database



RELATIONS

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

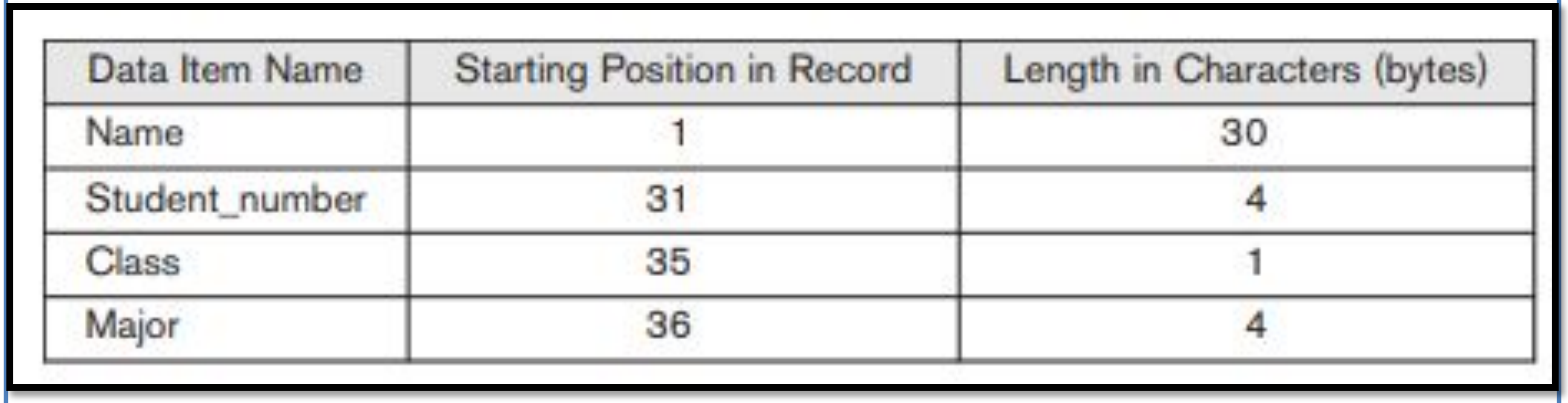
COLUMNS

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

2. Insulation between programs and data, and data abstraction

- 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.
- DBMS provides **program-data independence** property i.e. The structure of data files is stored in the DBMS catalog separately from the access programs.
- Some object oriented databases provides **program-operation independence,** an operation as two parts one is interface(signature) and the other is implementation(or method). Interface can be changed without changing the implementation
- The characteristic that allows program-data independence and program-operation independence is called **data abstraction.**

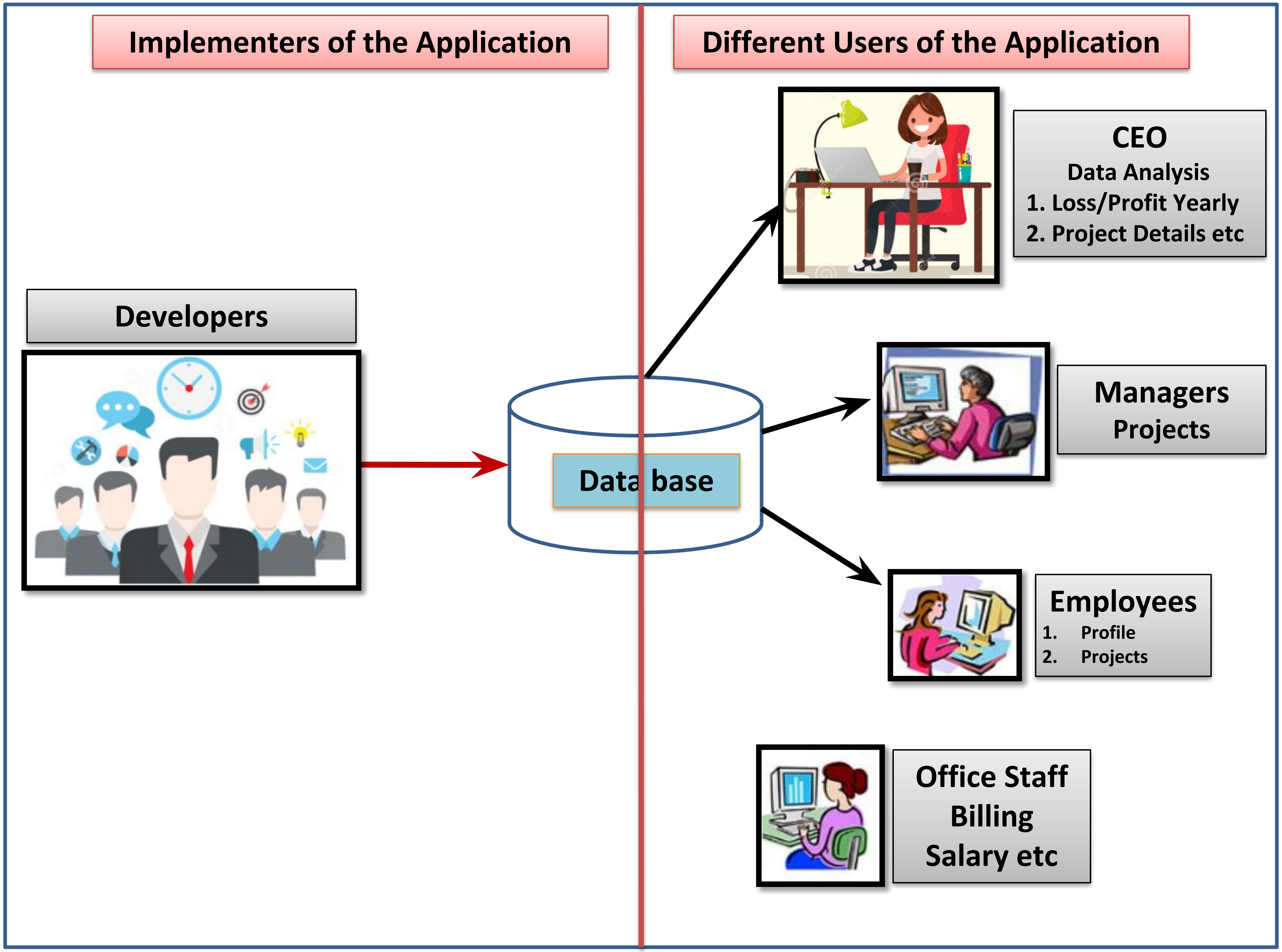
- A DBMS provides users with a **conceptual representation of data** that does not include many of the details of how the data is stored or how the operations are implemented
- Internal storage format for a STUDENT record, based on the database catalog



Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
Student_number	31	4
Class	35	1
Major	36	4

3. 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 is not explicitly stored.


Implementers of the Application
Different Users of the Application
Developers
Data base
CEO
Data Analysis
1. Loss/Profit Yearly
2. Project Details etc
Managers
Projects
Employees
1. Profile
2. Projects
Office Staff
Billing
Salary etc

4. Sharing of Data and Multiuser Transaction Processing

- A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time.
- This is essential if the data for multiple applications is to be integrated and maintained in a single database.
- The DBMS must include concurrency control software to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct.

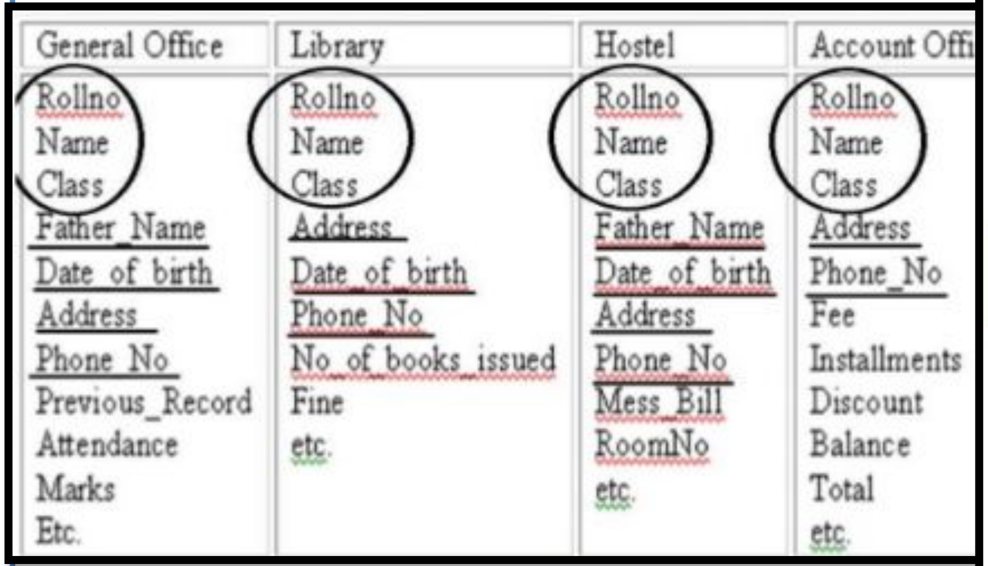
Advantages of Using the DBMS Approach

1. Controlling **redundancy**
2. **Sharing** of data among multiple users.
3. Restricting **unauthorized access** to data.
4. Providing **persistent** storage for program
5. Providing Storage Structures and Search Techniques for **Efficient Query Processing**
6. Providing **backup** and **recovery** services.
7. Providing **multiple interfaces** to different users.
8. Representing **complex relationships** among data.
9. Enforcing **integrity constraints** on the database.
10. Drawing Inferences and Actions using rules

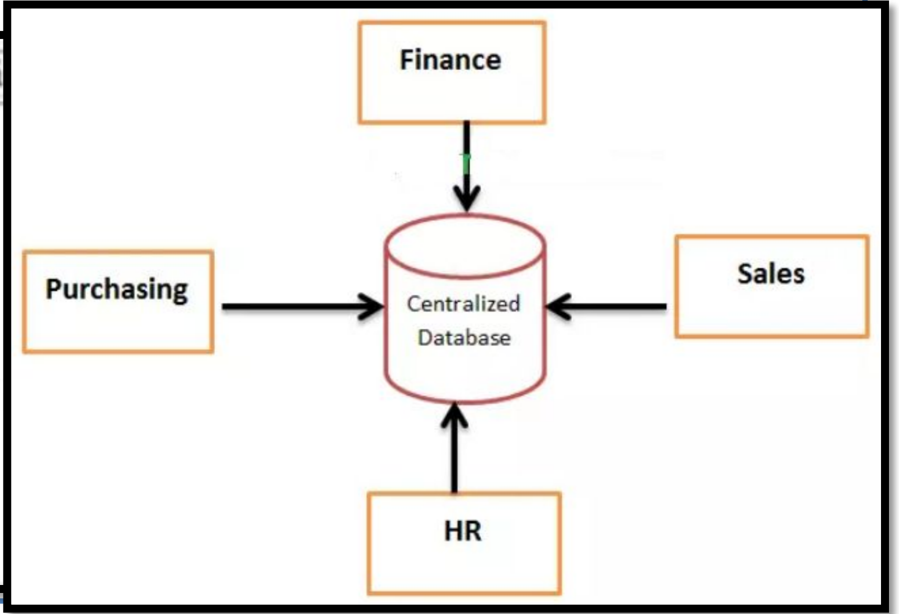
1. Controlling redundancy

Redundancy is storing the same data multiple times at several places, this leads to several problems,

- First, It leads to *duplication of efforts.*
- *Second, Storage space is wasted.*
- Third, files that represent the same data may become be *inconsistent.*
- However, in practice, it is sometimes necessary to use **controlled redundancy to** improve the performance of queries.



General Office	Library	Hostel	Account Offi
Rollno	Rollno	Rollno	Rollno
Name	Name	Name	Name
Class	Class	Class	Class
Father_Name	Address	Father_Name	Address
Date_of_birth	Date_of_birth	Date_of_birth	Phone_No
Address	Phone_No	Address	Fee
Phone_No	No_of_books_issued	Phone_No	Installments
Previous_Record	Fine	Mess_Bill	Discount
Attendance	etc.	RoomNo	Balance
Marks		etc.	Total
Etc.			etc.

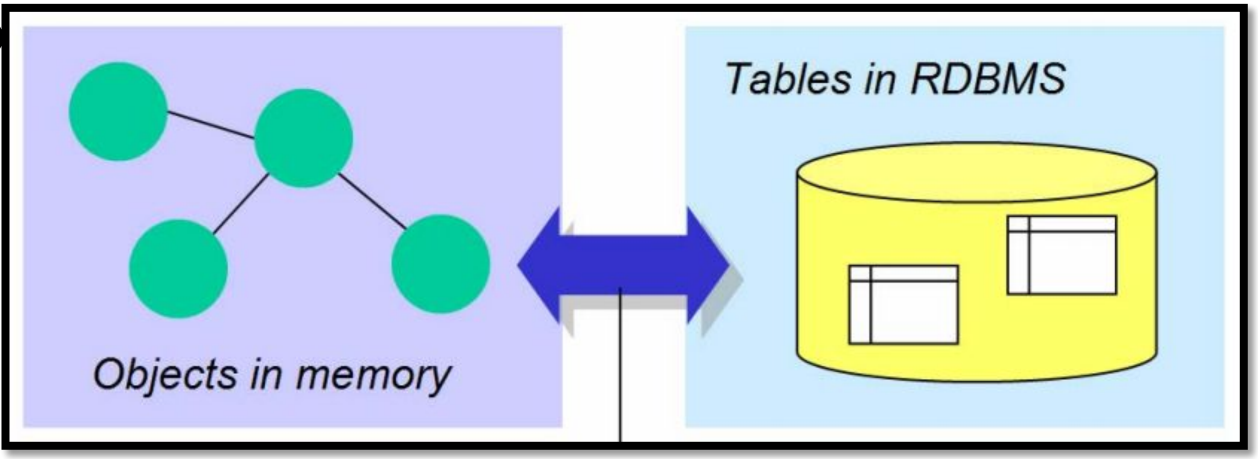


2. Restricting unauthorized access to data

- When multiple users share a large database, it is likely that most users will not be authorized to access all information in the database.
- A DBMS should provide a **security and authorization subsystem, which the DBA uses to create** accounts and to specify account restrictions and privileges.

3. Providing persistent storage for program objects

- Object Oriented Databases can be used to provide **persistent storage for program objects and data** structures.
- Programming languages typically have complex data structures, such as structs or class definitions in C++ or Java.
- Impedance mismatch is nothing but mismatch of data structures wrt DBMS and Programming languages, which requires co



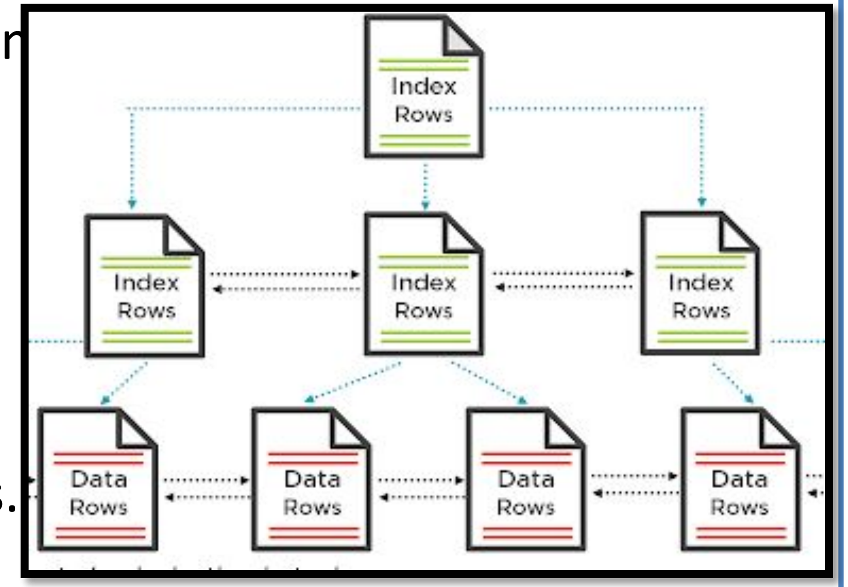
4. Providing Storage Structures and Search Techniques for Efficient Query Processing

- Auxiliary files called **indexes are often used for this purpose.** Indexes are typically based on tree data structures or hash data structures that are suitably modified for disk search.

- Auxiliary files called **indexes are often used for this purpose.** Indexes are typically based on tree data structures or hash data structures that are suitably modified for disk search.

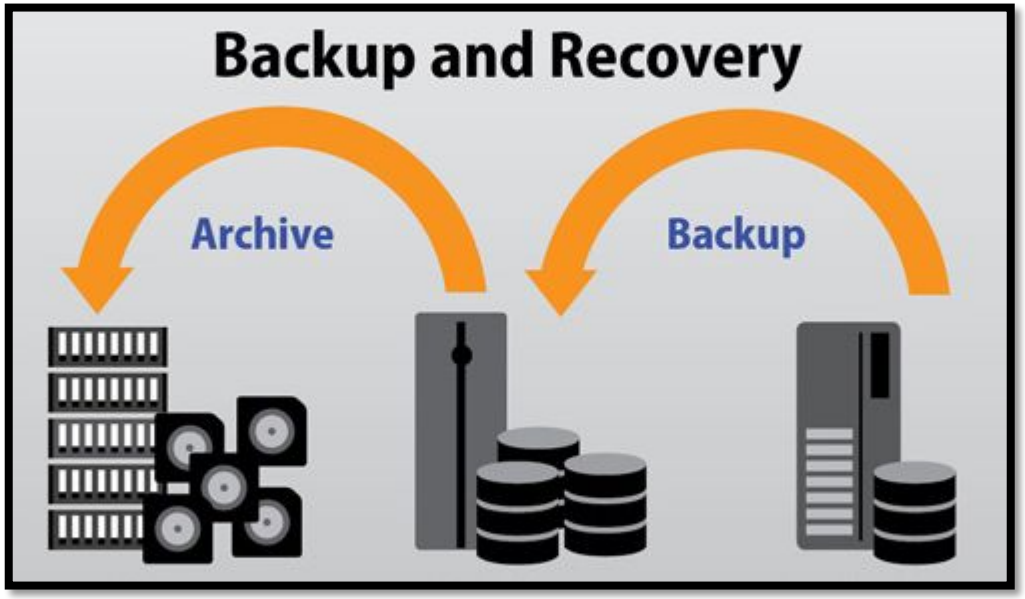
- DBMS has **Buffering Module** to keep so

- The **query processing and optimization module of the DBMS is responsible for** choosing an efficient query execution plan for each query based on the existing storage structures.

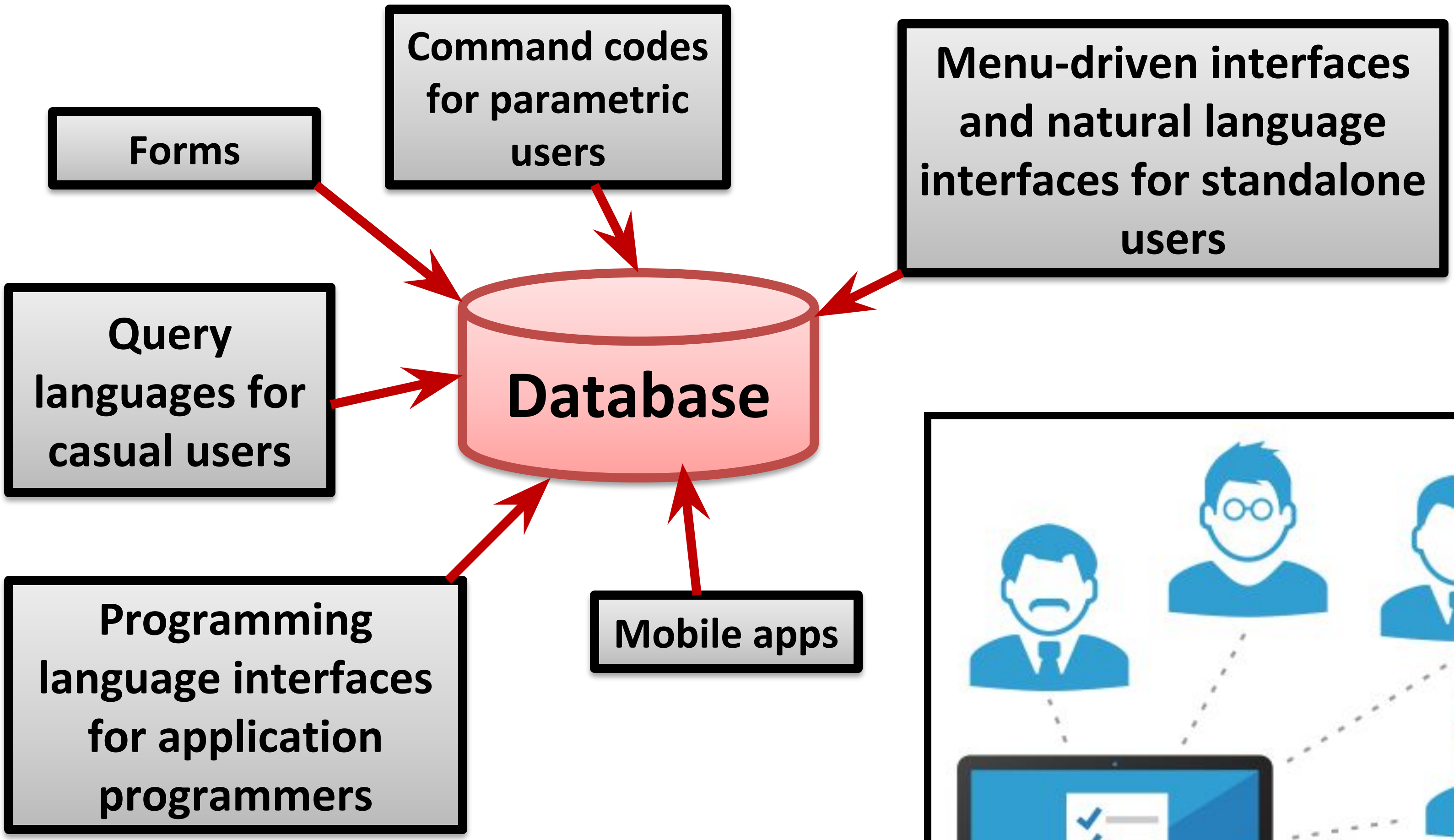


5. Providing Backup and Recovery

- A DBMS must provide facilities for recovering from hardware or software failures.
- The **backup and recovery subsystem of the DBMS is responsible for recovery.**
- **Disk Back up** is important in case of catastrophic disk failures.



6. Providing Multiple User Interfaces


Forms
Command codes for parametric users
Menu-driven interfaces and natural language interfaces for standalone users
Query languages for casual users
Database
Programming language interfaces for application programmers
Mobile apps

7. Representing Complex Relationships among Data

- A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to retrieve and update related data easily and efficiently.

8. Enforcing Integrity Constraints

- The simplest type of integrity constraint involves specifying a data type for each data item.
- Similarly there are **referential integrity key or uniqueness constraint.**
- A data item may be entered erroneously and still satisfy the specified integrity constraints, DBMS can’t handle such things.

9.Permitting Inferencing and Actions Using Rules and Triggers

- Some database systems provide capabilities for defining *deduction rules for inference* new information from the stored database facts.
- A **trigger** is a form of a rule activated by updates to the table, which results in performing some additional operations to some other tables, sending messages, and so on. More involved procedures to enforce rules are popularly called **stored procedures.**

10.Additional Implications of Using the Database Approach

- Potential for Enforcing Standards
- Reduced Application Development Time
- Flexibility
- Availability of Up-to-Date Information

Data Models, Schemas, and Instances

- **Data abstraction** generally refers to the **suppression of details of Data organization and storage**, and the highlighting of the essential features for an improved understanding of Data.
- **Data model**—a collection of concepts that can be used to describe the **structure of a Database**—provides the necessary means to achieve this abstraction.
- The Data in the Database at a particular moment in time is called a **Database state or snapshot.** It is also called the ***current set of occurrences or instances*** *in* the Database.
- The description of a Database is called the **Database schema**, which is specified during Database design and is not expected to change frequently.

Categories of Data Models

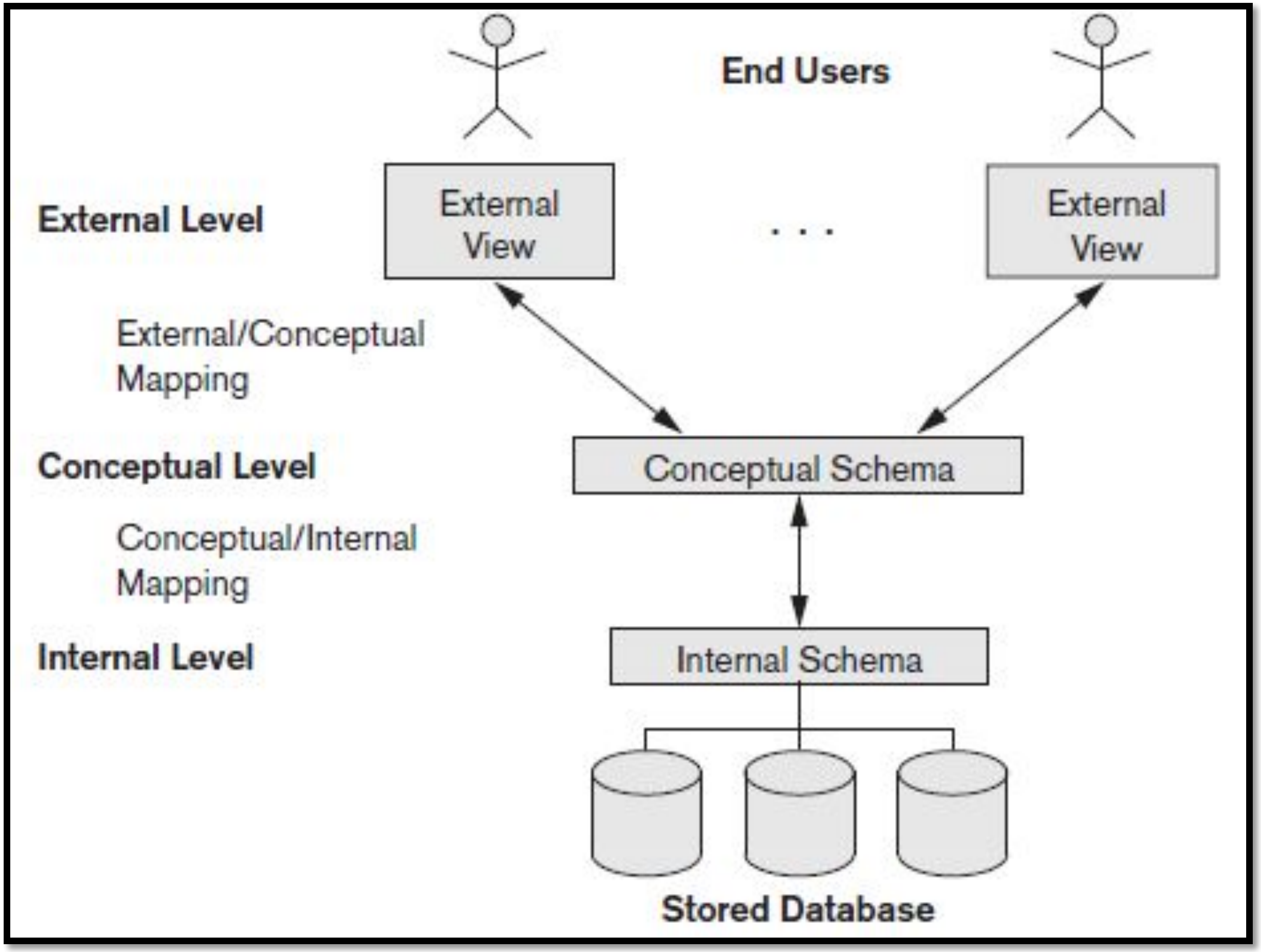
1. **High-level or conceptual Data models:** Provide concepts that are close to the way many users perceive Data. Conceptual Data models use concepts such as entities, attributes, and relationships.

2. **Low-level or physical Data models:** Provide concepts that describe the details of how Data is stored on the computer storage media, typically magnetic disks.

3. **Representational**(or **implementation) Data models:** Provide concepts that may be easily understood by end users but that are not too far removed from the way Data is organized in computer storage.

Three-Schema Architecture

The goal of three schema architecture is to separate the user applications from the physical database



Three-Schema Architecture and Data Independence

1. **The internal level has an internal schema, which describes the physical** storage structure of the Database. The internal schema uses a physical Data model and describes the complete details of Data storage and access paths for the Database.
2. **The conceptual schema** hides the details of physical storage structures and concentrates on describing entities, Data types, relationships, user operations, and constraints. Usually, a representational Data model is used to describe the conceptual schema.
3. **The external or view level** includes a number of external schemas or user views. Each external schema describes the part of the Database that a particular user group is interested in and hides the rest of the Database from that user group.

Data Independence

Data independence: It is defined as the capacity to change the schema at one level of a Database system without having to change the schema at the next higher level

1. **Logical Data independence** is the capacity to **change** the **conceptual schema** without having to change external schemas or application programs.

2. **Physical Data independence** is the capacity to **change** the **internal schema** without having to change the conceptual schema. Hence, the external schemas need not be changed as well.

DBMS
Architecture

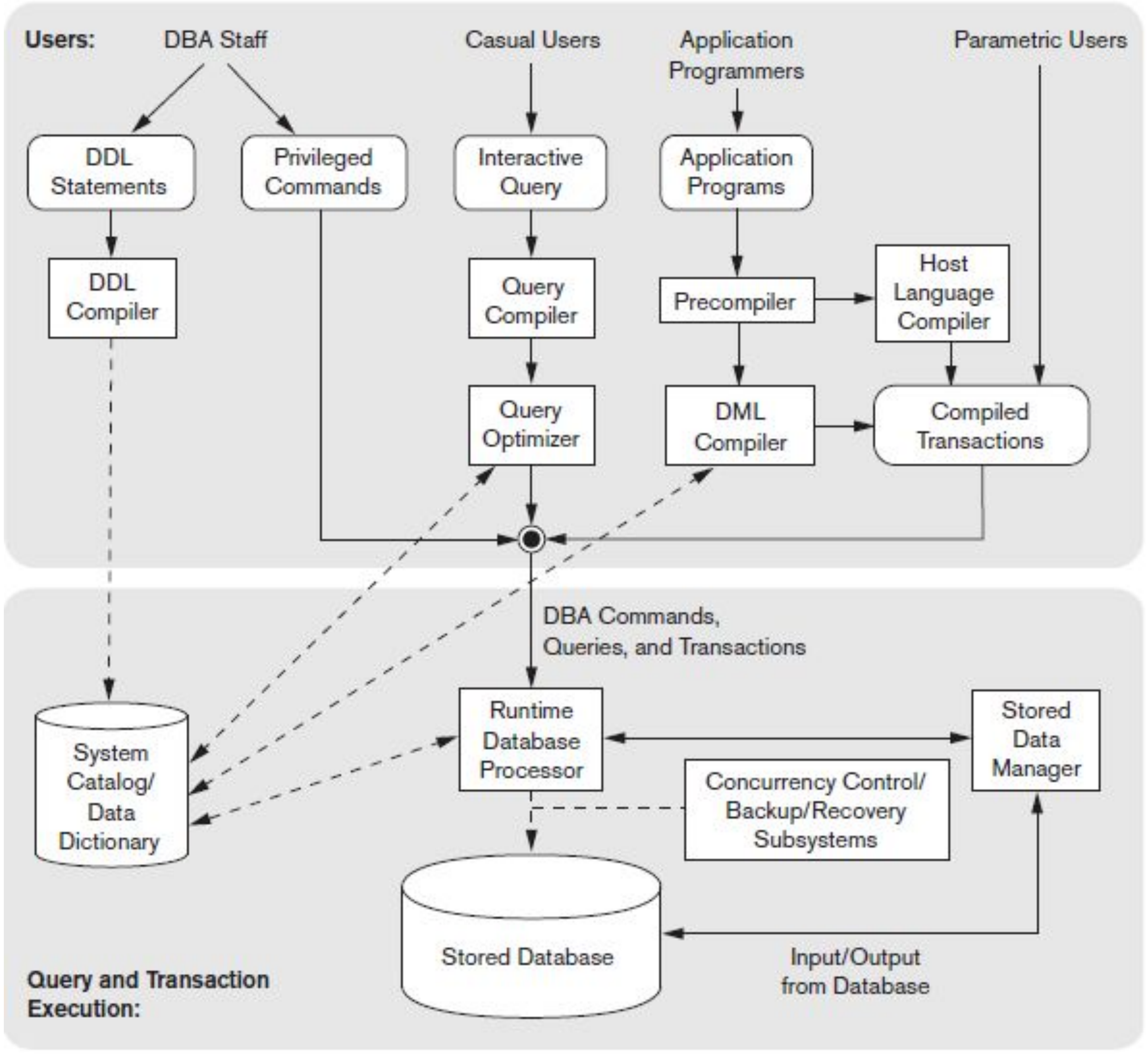


Figure: Component modules of a DBMS and their interactions.

Figure illustrates, in a simplified form, the typical DBMS components. The figure is divided into two parts. The top part of the figure refers to the various users of the database environment and their interfaces. The lower part shows the internal modules of the DBMS responsible for storage of data and processing of transactions.

The top part of Figure shows interfaces for the DBA staff, casual users who work with interactive interfaces to formulate queries, application programmers who create programs using some host programming languages, and parametric users who do data entry work by supplying parameters to predefined transactions. The DBA staff works on defining the database and tuning it by making changes to its definition using the DDL and other privileged commands.

The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog.

In the lower part of Figure 2.3, the **runtime database processor executes (1) the** privileged commands, (2) the executable query plans, and (3) the canned transactions with runtime parameters. It works with the **system catalog and may update it** with statistics. It also works with the **stored data manager, which in turn uses basic** operating system services for carrying out low-level input/output (read/write) operations between the disk and main memory. The runtime database processor handles other aspects of data transfer, such as management of buffers in the main memory.