Learning Resource
On
Database Management Systems

Chapter-2
Database System Concepts and Architecture

Prepared By:
Kunal Anand, Asst. Professor
SCE, KIIT, DU, Bhubaneswar-24

Chapter Outcomes

- After the completion of this chapter, the students will be able to:
  - Explain different types of data models.
  - Define Schema and its types.
  - Differentiate between database state and instance.
  - Describe the 3-schema or 3-layered architecture
  - Illustrate the different components of DBMS environment.
  - Identify the challenges in building a database

Organization of theChapter

- Introduction
- Data Models
- Schemas, Database states and Instances
- 3-Schema Architecture
- Components of a DBMS
- Challenges to build a DBMS

Introduction

- In its earlier times, the DBMS package was in the form of monolithic systems, where the whole DBMS software package was one tightly integrated system.
- Gradually with time, its evolution happened and today it is in the form of modular system with a client/server architecture.
- In a basic client/server architecture, the system functionality is distributed between two types of modules:
  - **Client module:** designed to run on a PC, workstation, or a mobile device.
  - **Server module:** handles data storage, access, search, and other functions.

Data Models

- One of the major characteristics of the database approach is to provide abstraction, so that different users can perceive the data at their preferred level of detail.
- This abstraction is achieved by **Data Model** in a DB environment.
- **Data Model** can be formally defined as a collection of concepts that can be used to describe the structure of a database. By "structure" we refer to the data types, relationships, and constraints that apply to the data.
- Most data models also include the basic operations for specifying retrieval and updation of the database.

Data Models (contd..)

- Additionally, the data models also include the concepts to describe the dynamic apsect or behavior of the system.

- It allows the database designers to specify a set of user defined operations that are possible on the database objects.
  - **For example:** a user defined operation **COMPUTE_GPA** can be applied to the **STUDENT** object whereas, generic operations like **INSERT, DELETE, MODIFY, or RETRIEVE** can be applied to **any database objects**.

- The concepts used to specify the dynamic aspect is mainly applicable to Object Oriented Models.

Categories of Data Models

- Based on concepts, data models can be classified into following three categories
  - **High level or Conceptual data model:** It provides concepts that are close to the way many users perceive the data.
  - **Low level or Physical data model:** It provides concepts that describe the details of how data is stored on the computer storage media.
  - **Representational or Implementation data model:** It provides the concepts that may be easily understood by the users but not too far from the way data is organized in computer storage.

Building blocks of a Data Model

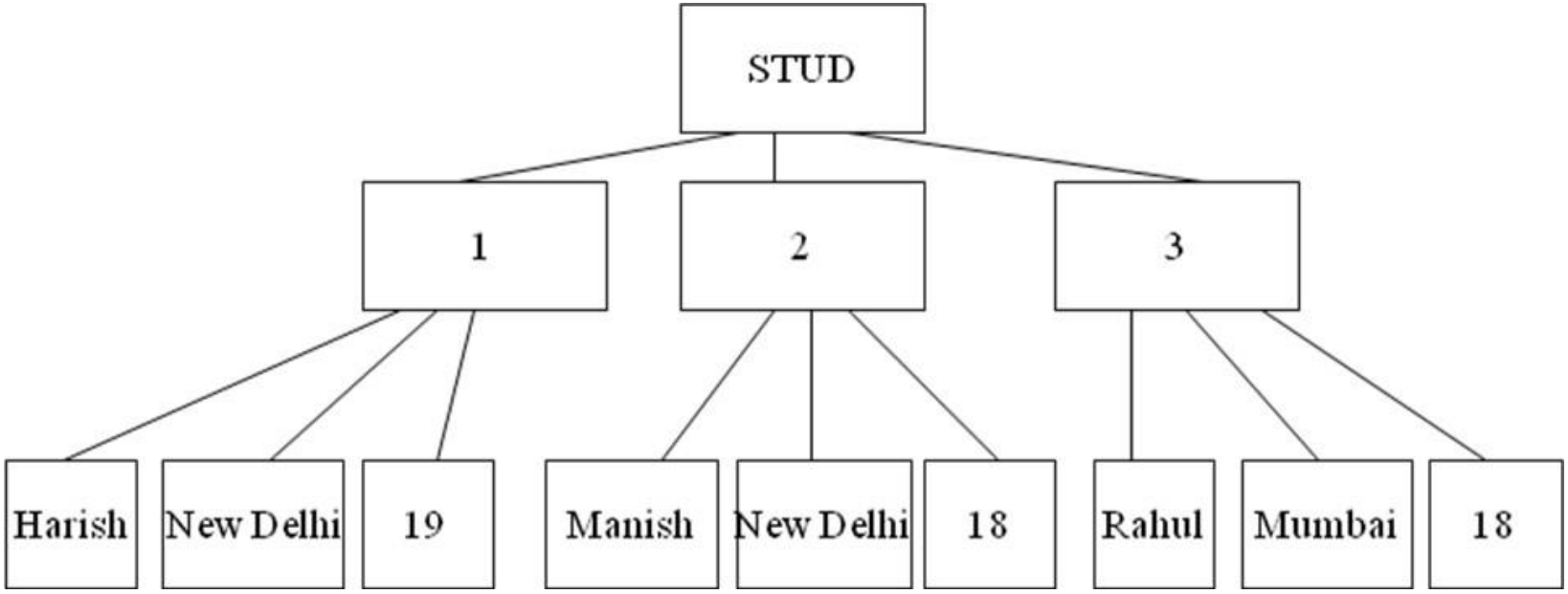
- Conceptual data models use the concepts like entity, attributes, relationship, and constraints. They are also known as the building blocks of a data model.
  - **Entity:** An entity represents a real-world concept that can be described in the database.
  - **Attribute:** property or feature or characteristic that may further describe an entity.
  - **Relationship:** When there are two or more than two entities in a database, they are associated to each other in one way or another. This association is known as relationship.
  - **Constraint:** A constraint is a restriction placed on a data. They are important as they help to achieve data integrity i.e., an assurance of the accuracy and consistency of, data over its entire life-cycle

Types of Data Models

- Broadly, data models are of following types:
  - Hierarchical Model
  - Network Model
  - Relational Model
  - Entity Relationship Model
  - Object-Oriented Model
  - Object-Relational Model

Hierarchical Model

- Hierarchical Model, one of the legacy data models, was developed in the 1960s to manage large amount of data for complex manufacturing projects.
- The basic logical structure is represented by an *upside-down tree*. The hierarchical structure contains levels of segments.

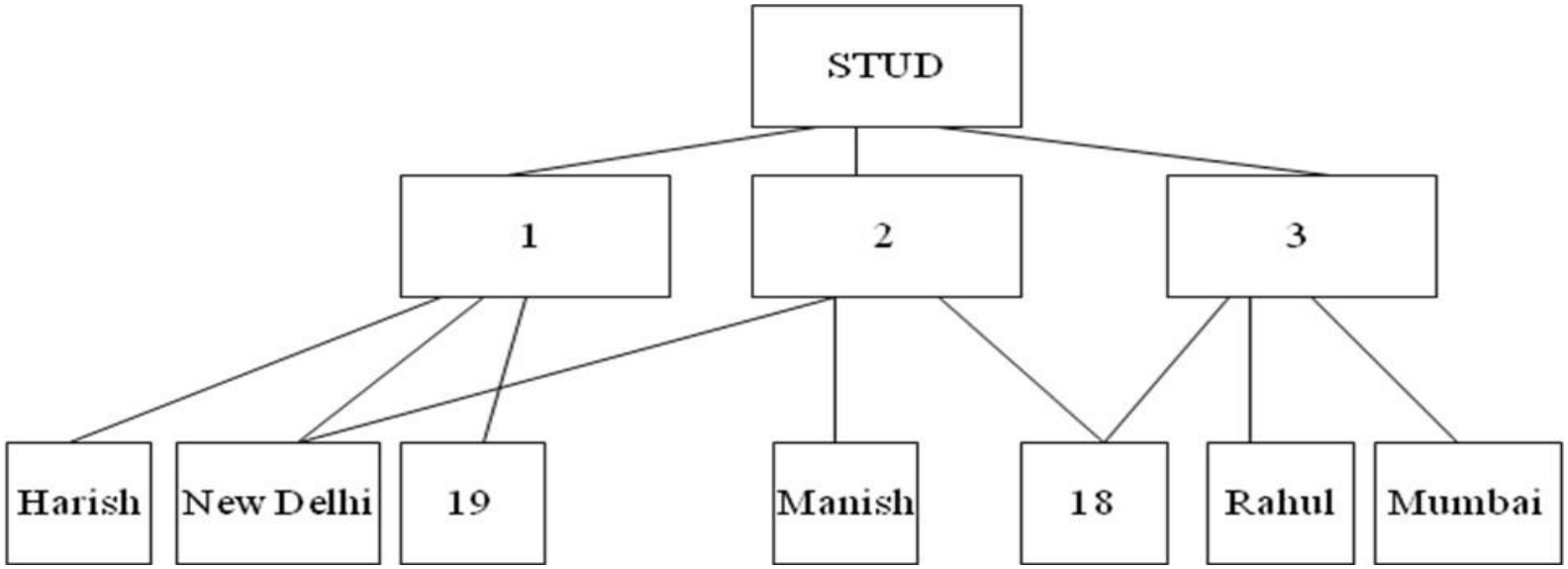


Advantages and Disadvantages

- **Advantages**
  - allows easy addition and deletion of new information.
  - relates very well to natural hierarchies such as assembly plants and employee organization in corporations.
  - relates well to anything that works through a one-to-many relationship.
- **Disadvantages**
  - The database can be very slow when searching for information on the lower entities.
  - Searching for data requires the DBMS to run through the entire model from top to bottom until the required information is found, making queries very slow.
  - Can only model one to many relationships, many to many relationships are not supported.

Network Model

- The network model, another legacy model, was created to represent complex data relationships more effectively than the hierarchical model, to improve database performance, and to impose a database standard.
- A user perceives the network model as a collection of records in 1:M relationships



Advantages and Disadvantages

- **Advantages**
  - represents complex data relationships more effectively than the hierarchical model
  - handles more relationship types, such as M: N.
  - Data access is more flexible than hierarchical model
  - Improved database performance as it includes DDL and DML commands.
- **Disadvantages**
  - System complexity limits efficiency
  - Navigational system yields complex implementation and management
  - Structural changes require changes in all application programs
  - Networks can become chaotic unless planned carefully

Relational Model

- The relational model was introduced by E. F. Codd in 1970. This data model is implemented through RDBMS; which is easier to understand and implement.
- The most important advantage of the RDBMS is, its ability to hide the complexities of the relational model from the user.
- Another reason for the relational data model's rise to dominance is its powerful and flexible query language. Generally, SQL is used for this purpose

STUD

roll	name	city	age
1	Harish	New Delhi	19
2	Manish	New Delhi	18
3	Rahul	Mumbai	18

GRADE

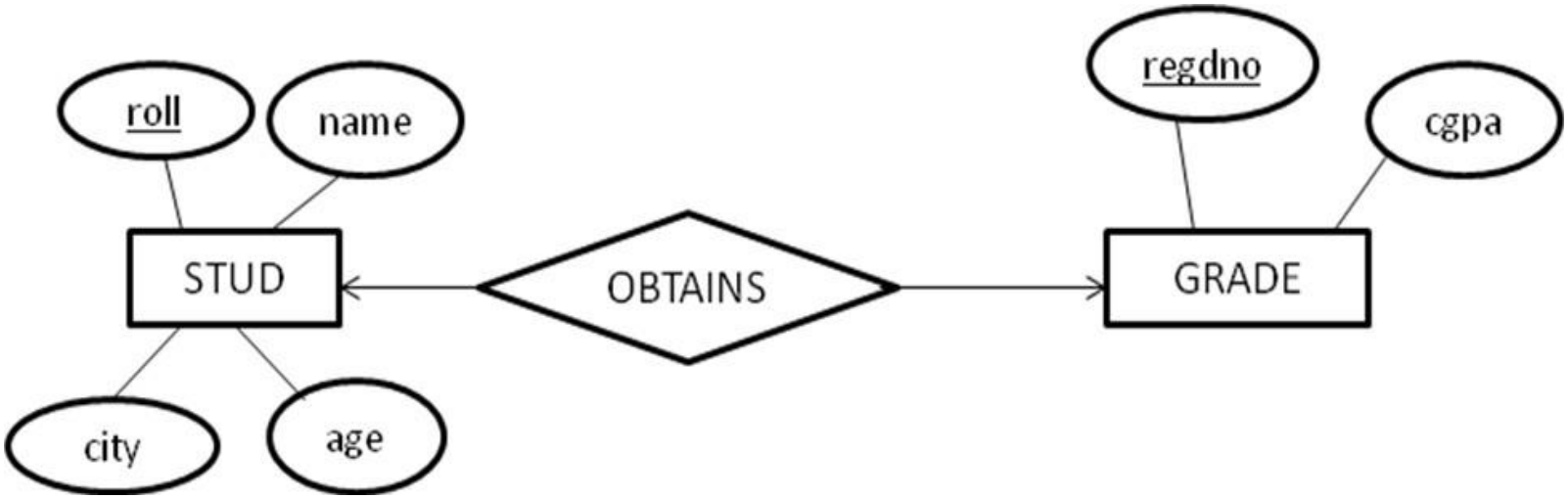
regdno	roll	cgpa
S001	1	7.9
S002	2	8.5
s003	3	9.4

Advantages and Disadvantages

- **Advantages**
  - Changes in table's structure do not affect data access or application programs.
  - Tabular view eases the design, implementation, management, and use of the database.
  - Provides data consistency
  - RDBMS isolates the end users from the physical level details which enhances the implementation and simplicity.
- **Disadvantages**
  - Expensive due to the high set-up cost and maintenance cost
  - Advances in the complexity of data make the process of categorization of data more difficult.
  - Isolated databases create the problem of **"Island of information",** where information from one large system can not be shared easily to the other system.

Entity Relationship Model

- Peter Chen first introduced the ER data model in 1976.
- It was the graphical representation of entities and their relationships in a database structure that quickly became popular.
- Thus, the ER-model has become a widely accepted standard for data modeling.
- ER models are normally represented in an ER diagram



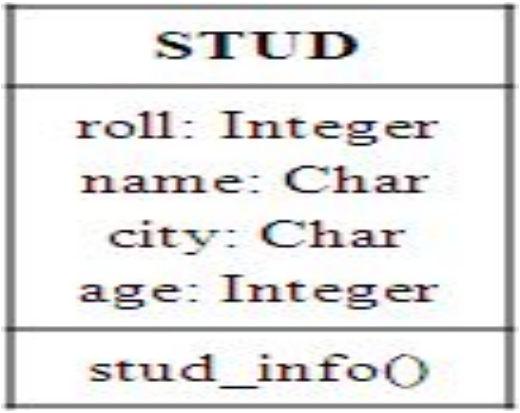
Advantages and Disadvantages

- **Advantages**
  - Visual modeling yields exceptional conceptual simplicity and makes it more communicative.
  - Basic building blocks of a database can be represented more effectively here.

- **Disadvantages**
  - There is limited relationship and constraint representation.
  - Loss of information content when attributes are removed in order to avoid crowded displays.

Object Oriented Model

- In object-oriented data model, both data and their relationships are contained in a single structure called an *object*.
- Like the relational model's entity, an object is described by its factual content. But quite unlike an entity, an object includes information about relationships between the facts within the object, as well as information about its relationships with other objects.
- Attributes describe the properties of an object. Objects that share similar characteristics are grouped in classes. Thus, a *class* is a collection of similar objects with shared structure (attributes) and methods.



Advantages of Object Oriented Model

- The object-oriented data model allows the 'real world' to be modeled more closely.
- OODBMSs allow new data types to be built from existing types.
- Unlike traditional databases (such as hierarchical, network or relational), the object-oriented database can store different types of data, for example, pictures, voice video, including text, numbers and so on.
- The tight coupling between data and applications in an OODBMS makes schema evolution more feasible.
- Applicability to advanced database applications like CAD, CASE, OIS, and Multimedia Systems.
- Improved performance

Disadvantages of Object-Oriented Model

- There is no universally agreed data model for an OODBMS, and most models lack a theoretical foundation.
- In comparison to RDBMSs the use of OODBMS is still relatively limited.
- Query optimization compromises encapsulations
- Perhaps one of the most significant issues that face OODBMS vendors is the competition posed by the RDBMS and the emerging ORDBMS products.
- Lack of support for views
- Lack of support for security

Object Relational Model

- The object-oriented data model is somewhat spherical in nature, allowing access to unique elements anywhere within a database structure, with extremely high performance. But it performs extremely poorly when retrieving more than a single data item.

- The relational data model is best suited for retrieval of groups of data but can also be used to access unique data items fairly and efficiently.

- Thus, by combining the features of relational data model and object-oriented data model, object-relational data model was created

Schema, Instances, and Database State

- In any data model, it is very important to distinguish between the description of the database and the database itself.

- The physical description of a database is known as **schema**. This is specified during the database design, and it is not expected to change frequently.

- A displayed schema is known as **schema diagram**. It displays the structure of each record type but not the actual data of the records.

- Each object in the schema diagram is known as **schema construct.**

Example of Schema Diagram

Schema Diagram for University Database

No. of Schema construct :- 4

STUDENT

Name	Stud_No.	Class	major

COURSE

course_name	course Number	credit	Dept.

SECTION

Sec_ID	course Number	Sem	Year	Inst.

GRADE

Stud_No.	Sec_ID	Grade

contd..

- The data in the database at a particular point of time is called a **"Database State"**. Any entry in the database state at any point of time is called **"Instance"** i.e., Database state is a collection of instances at any point of time.
- The DBMS stores the description of the schema and constraints in the form of a catalogue, known as **meta-data** so that DBMS software can refer to the schema whenever needed.
- Database schema, once defined, does not change frequently whereas, the database state keeps on changing continuously. Schema is known as **"Intension"** whereas the database state is known as **"extension"** of the schema.
- Though, the database schema does not change very frequently. It may undergo some changes when there is any change in the application requirement. This is known as **"Schema Evolution".**

3-Tier or 3-Schema Architecture

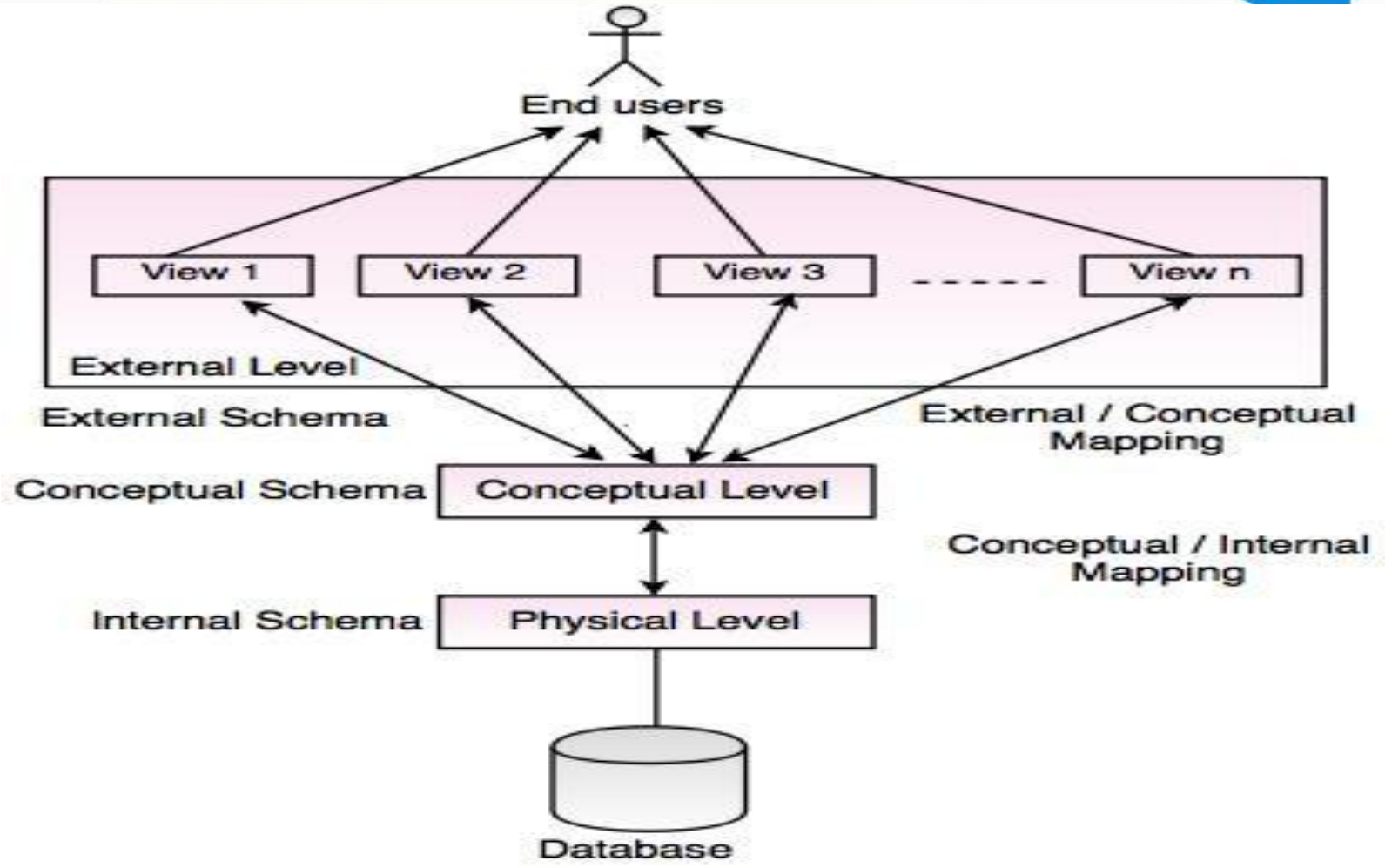


Fig. Three Level Architechture of DBMS

contd..

- **External Level**
  - includes several external schemas or user views.
  - Each external schema describes the part of the database that a particular user is interested in and hides the rest of the database from the user.
- **Conceptual Level**
  - This level has a conceptual schema that describes the structure of the database for a community of users.
  - It hides the details of physical storage structure and concentrates on describing entities, datatypes, relationship, user operations, and constraints.
- **Internal Level**
  - This level has an internal schema that describes the physical storage structure of the database.

Points to be noted

- The **3-schema architecture** is used to visualize different schemas in a database. Though most DBMS supports the architecture, they don't separate the three levels completely and explicitly.

- A RDBMS like Oracle represents both conceptual and external schema in the same data model.

- Some DBMS use different models to represent different level schemas at conceptual and external level.
  - Ex: Universal Database represents conceptual schema using relational model and external schema using object- oriented model.

contd..

- The three schemas are only the description of the data, the stored data actually exists at the physical level in the form a database.

- In a 3-schema based DBMS, each user group refers only to its own external schema. So, the DBMS must transform a request specified on external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the database.

- The data extracted from the stored database is transformed back into the respective external's view. This process of transforming requests and results between levels are known as **“Mapping”**.

Data Independence

- Data independence is the ability to change the schema at one level without changing the schema at the next higher level.
- It is of following two types:
  - **Logical data independence**
    - The ability to change the conceptual schema without having to change the external schema.
    - Conceptual schema may have to be changed to expand the database, to change constraints, or to reduce the database. However, the change in conceptual schema must not impact the working of application program that refers the external schema.
  - **Physical data independence**
    - The ability to change the internal schema without changing the conceptual schema as well as the external schema.

Note:

- Genrally, physical data independence exists in most of the DBMS whereas the logical data independence is hard to achieve.

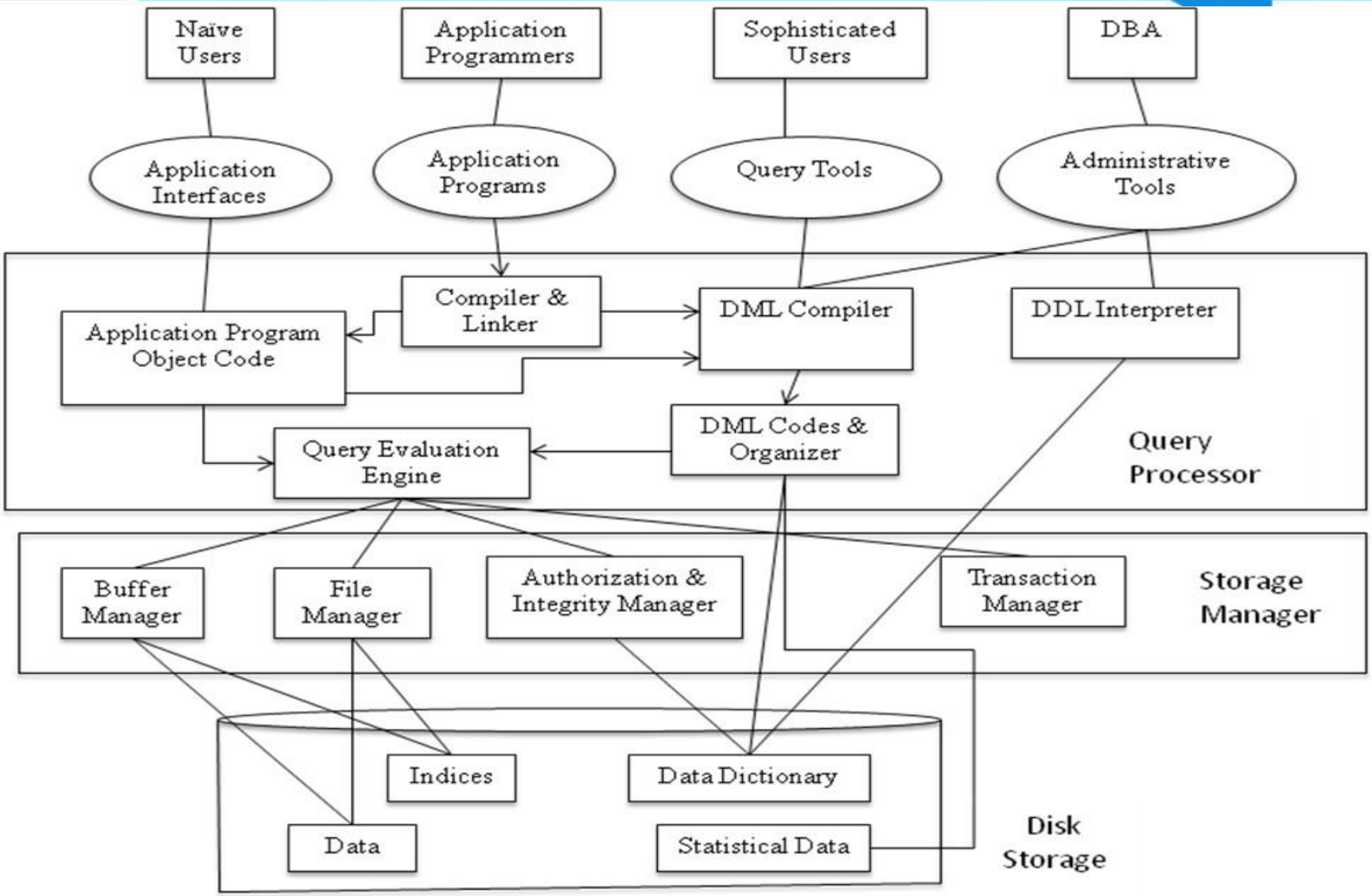
- This is because changing the conceptual schema without having any impact on the application programs is a much strict requirement.

- Achieveing true data independence creates some overhead on the DBMS due to this multiple mapping. It affects the efficiency of the DBMS.

Components of DBMS

- A typical DBMS environment consists of following components:
  - Database Users
  - Application programs, interfaces, query tools, and administrative tools.
  - Query processor
  - Storage manager
  - Disk storage

Components of DBMS (contd..)



Components of DBMS (contd..)

- **Storage Manager:**
  - A storage manager is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
  - The storage manager is responsible for the interaction with the file manager. Storage manager is also responsible for storing, retrieving and updating data in the database.
- The storage manager component includes:
  - **Authorization and Integrity Manager:** This module ensures the integrity and authorized access of the data.
  - **Transaction Manager:** They ensure that the database remains in a consistent state despite of any failure and the concurrent transactions can proceed without any conflicts.

Components of DBMS (contd..)

- **File Manager:** This module manages the allocation of space on disk storage and data structures used to represent information stored on the disk.

- **Buffer Manager:** Buffer manager is responsible for fetching data from the disk storage into main memory. The buffer manager is a critical part of the database system.

Note: Additionally, the storage manager implements several data structures like data files, data dictionary, and indices as part of the physical system implementations.

Components of DBMS (contd..)

- **Query Processor:** The work of query processor is to execute the query successfully. The major components of query processor include:
  - **DDL Interpreter** : This interpreter is used to interpret DDL statements and records the definitions in the data dictionary.
  - **DML Compiler** :
    - DML compiler translates the DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.
    - When a user wants to perform a DML operation, the data dictionary must be checked for the validation purpose
  - **Query Evaluation Engine**: This module executes the low-level instructions generated by the DML compiler.

Challenges in building a DBMS

- The challenges in building an efficient DBMS are as below:
  - Increased data volume
  - Growing complexities.
  - Data security
  - Decentralized data management
  - Limits on scalability
  - The Management of Cloud-Based Databases
  - Data Integration from Various Sources