UNIT I

Introduction: Database system, Characteristics (Database Vs File System), Database Users (Actors on Scene, Workers behind the scene), Advantages of Database systems, Database applications. Brief introduction of different Data Models; Concepts of Schema, Data independence- Physical & Logical Data Independence; Three Tier/ schema architecture for data independence; Database system structure.

1.1 Introduction

A **database** is a collection of related data. By **data,** we mean known facts that can be recorded and that have implicit meaning. Database plays a critical role in almost all areas where computers are used, including business, engineering, medicine, law, education, and library science.

Data : Collection of raw facts.

Information : Processed data.

Database : Collection of raw facts of one particular organization

DataBase Management Systems : The ordering and operations performed on a collection of data of one particular organization through some application programs is known as DataBase Management Systems.

Or

A **Database Management System (DBMS)** is a set of computer programs that controls the creation, maintenance, and the use of the database of an organization and its end users.

File processing system store data in separate computer files. File processing system is a system used to store and manage data that involves each department or area within an organization having its own set of files, often creating data redundancy and data isolation. File processing system store data in separate computer files.

Disadvantages of File Processing Systems include:

1. Program-Data Dependence: File descriptions are stored within each application program that accesses a given file.

2. Duplication of Data / Data Redundancy: Applications are developed independently in file processing systems leading to unplanned duplicate files. Duplication is wasteful as it requires additional storage space and changes in one file must be made manually in all files.

3. Limited data sharing: Each application has its own private files with little opportunity to share data outside their own applications. A requested report may require data from several incompatible files in separate systems.

4. Lengthy Development Times: There is little opportunity to leverage previous development efforts. Each new application requires the developer to start from scratch by designing new file formats and descriptions.

5. Expensive Program Maintenance: The preceding factors create a heavy program maintenance load.

6. Integrity problems: The data values stored in the database must satisfy certain types of consistency constraints. When new constraints are added, it is difficult to change the programs.

7. Security problems: Enforcing security constraints is difficult.

8. Data Inconsistency : when we are having the duplication of data then we will get inconsistency problem means modifications will be done in various places instead of single place or location.

9. Single User view : In files we can see the data in one system only means where you can compile the program there only data will be available.

DATABASE : In other words, a database has some source from which data are derived, some degree of interaction with events in the real world, and an audience that is actively interested in the contents of the database. A database can be of any size and of varying complexity.

A database may be generated and maintained manually or it may be computerized. The library card catalog is an example of a database that may be created and maintained manually. A computerized database may be created and maintained either by a group of application programs written specifically for that task or by a database management system.

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 the various users and applications.

- ✓ **Defining** a database involves specifying the data types, structures, and constraints for the data to be stored in the database. The database definition and descriptive information is also stored by DBMS in the form of database catalog or dictionary; it is called **meta data**.
- ✓ **Constructing** the database is the process of storing the data itself on some storage medium that is controlled by the DBMS.
- ✓ **Manipulating** a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
- ✓ **Sharing** a database allows multiple users and programs to access the database simultaneously.

The database and DBMS software together we will call a **database system.** An **application program** accesses the database by sending queries or requests for data to the DBMS. A **query** typically causes some data to be **retrieved**. A **transaction** may cause some data to be **read** and some data to be **written** into the database. Important functions provided by DBMS include **protecting** the database and **maintaining** it over a long period of time.

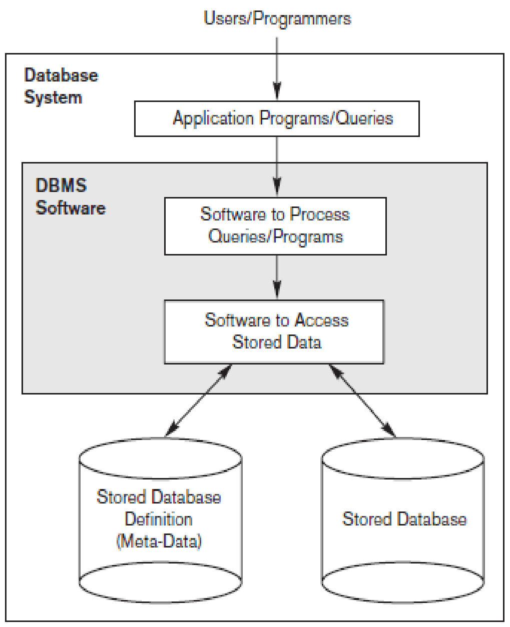

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Figure 1.1
A simplified database system environment.

1.2 An Example of Database

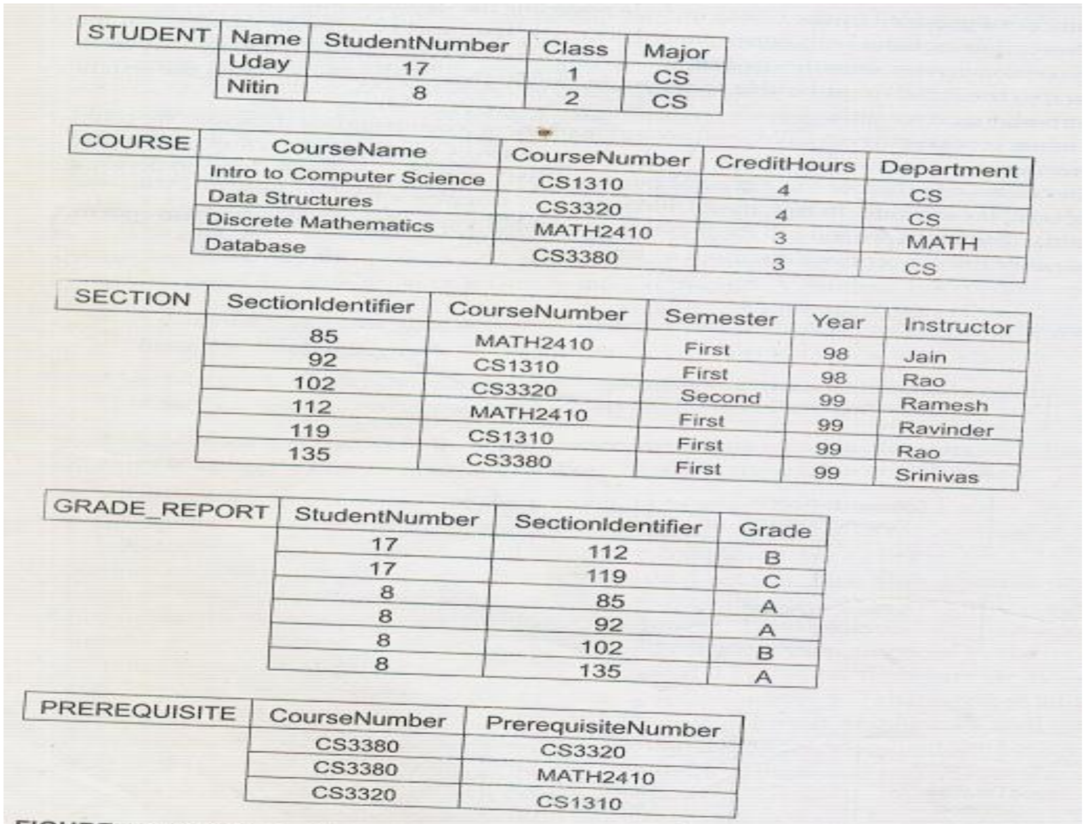
Let us consider an example: a UNIVERSITY database for maintaining information concerning students, courses, and grades in a university environment. Figure 1.2 shows the database structure and a few sample data for such a database. The database is organized as five files, each of which stores data records of the same type.

1. The **STUDENT** file stores data on each student.
2. The **COURSE** file stores data on each course.
3. The **SECTION** file stores data on each section of a course.
4. The **GRADE_REPORT** file stores the grades that students receive in the various sections they have completed.
5. The **PREREQUISITE** file stores the prerequisites of each course.

To define this database, we must specify the structure of the records of each file by specifying the different types of **data elements** to be stored in each record.

To construct the UNIVERSITY database, we store data to represent each student, course, section, grade report, and prerequisite as a record in the appropriate file.

Database manipulation involves querying and updating. Examples of queries are "retrieve the transcript—a list of all courses and grades of the particular student. Examples of updates are "change the class of Smith to new class name".



STUDENT

Name	StudentNumber	Class	Major
Uday	17	1	CS
Nitin	8	2	CS

COURSE

CourseName	CourseNumber	CreditHours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

SectionIdentifier	CourseNumber	Semester	Year	Instructor
85	MATH2410	First	98	Jain
92	CS1310	First	98	Rao
102	CS3320	Second	99	Ramesh
112	MATH2410	First	99	Ravinder
119	CS1310	First	99	Rao
135	CS3380	First	99	Srinivas

GRADE_REPORT

StudentNumber	SectionIdentifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

CourseNumber	PrerequisiteNumber
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

FIGURE 1.2 A database that stores student and course information.

1.3 Characteristics of the Database Approach

In traditional **file processing,** each user defines and implements the files needed for a specific application as part of programming the application. Both the users are interested in data about students, each user maintains separate files and programs to manipulate these files because each requires some data not available from the other user's files. This redundancy in defining and storing data results in wasted storage space and in redundant efforts to maintain common data up-to-date.

In the **database approach**, a single repository of data is maintained that is defined once and then is accessed by various users.

In **file system**, each application is free to name data elements independently. In contrast, in **database**, the names or labels of data are defined once, and used repeatedly by queries, transaction and applications.

The main characteristics of the database approach versus the file-processing approach are the following.

1.3.1 Self-Describing Nature of a Database System

Database system contains not only the database itself but also a complete definition or description of the database structure and constraints. This definition is stored in the system **catalog,** which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data. The information stored in the catalog is called **meta-data,** and it describes the structure of the primary database.

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, whose structure is declared in the application programs.

1.3.2 Insulation between Programs and Data, and Data Abstraction

In traditional file processing, the structure of data files is embedded in the access programs, so any changes to the structure of a file may require changing all programs that access this file. By contrast, DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property **program-data independence.** User can define operations on data as part of the database definition. An operation is specified in two parts.

- ✓ The **interface** (or signature) of an operation includes the operation name and the data types of its arguments.
- ✓ The **implementer** (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.

Data model is a type of data abstraction that is used to provide this conceptual representation. The data model uses logical concepts, such as objects, their properties, and their interrelationships, that may be easier for most users to understand than computer storage concepts.

1.3.3 Support of Multiple Views of the Data

A database typically has many 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.

1.3.4 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. When several reservation clerks try to assign a seat on an airline flight, the DBMS should ensure that each seat can be accessed by only one clerk at a time for assignment to a passenger. These types of applications are generally called **on-line transaction processing (OLTP)** applications. A fundamental role of multiuser DBMS software is to ensure that concurrent transactions operate correctly.

DATABSE USERS :

I Actors on the Scene:

A. Database Administrators

In a database environment, the primary resource is the database itself and the secondary resource is the DBMS and related software. Administering these resources is the responsibility of the **database administrator (DBA).**

A. Database Administrators(DBA)

A DBA has many responsibilities. A good performing database is in the hands of DBA.

1. Installing and upgrading the DBMS Servers: – DBA is responsible for installing a new DBMS server for the new projects. He is also responsible for upgrading these servers as there are new versions comes in the market or requirement. If there is any failure in upgradation of the existing servers, he should be able revert the new changes back to the older version, thus maintaining the DBMS working. He is also responsible for updating the service packs/ hot fixes/ patches to the DBMS servers.

2. Design and implementation: – Designing the database and implementing is also DBA's responsibility. He should be able to decide proper memory management, file organizations, error handling, log maintenance etc for the database.

3. Performance tuning: – Since database is huge and it will have lots of tables, data, constraints and indices, there will be variations in the performance from time to time. Also, because of some designing issues or data growth, the database will not work as expected. It is responsibility of the DBA to tune the database performance. He is responsible to make sure all the queries and programs works in fraction of seconds.

4. Migrate database servers: – Sometimes, users using oracle would like to shift to SQL server or Netezza. It is the responsibility of DBA to make sure that migration happens without any failure, and there is no data loss.

5. Backup and Recovery: – Proper backup and recovery programs needs to be developed by DBA and has to be maintained him. This is one of the main responsibilities of DBA. Data/objects should be backed up regularly so that if there is any crash, it should be recovered without much effort and data loss.

6. Security: – DBA is responsible for creating various database users and roles, and giving them different levels of access rights.

7. Documentation: – DBA should be properly documenting all his activities so that if he quits or any new DBA comes in, he should be able to understand the database without any effort. He should basically maintain all his installation, backup, recovery, security methods. He should keep various reports about database performance

B. Database Designers

Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data. These tasks are mostly undertaken before the database is actually implemented and populated with data. It is the responsibility of database designers to communicate with all prospective database users, in order to understand their requirements, and to come up with a design that meets these requirements.

Database designers typically interact with each potential group of users and develop a **view** of the database that meets the data and processing requirements of this group. These views are then analyzed and integrated with the views of other user groups. The final database design must be capable of supporting the requirements of all user groups.

C. End Users

End users are the people whose jobs require access to the database for querying, updating, and generating reports; the database primarily exists for their use.

There are several categories of end users:

1. **Casual end users** occasionally access the database, but they may need different information each time. They use a sophisticated database query language to specify their requests and are typically middle- or high-level managers or other occasional browsers.
2. **Naive** or **parametric end users** make up a sizable portion of database end users. Their main job function revolves around constantly querying and updating the database, using standard types of queries and updates. Bank tellers check account balances and post withdrawals and deposits. Reservation clerks for airlines, hotels, and car rental companies check availability for a given request and make reservations.
3. **Sophisticated end users** include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS so as to implement their applications to meet their complex requirements.

4. **Stand-alone users** maintain personal databases by using ready-made program packages that provide easy-to-use menu- or graphics-based interfaces. An example is the user of a tax package that stores a variety of personal financial data for tax purposes.

D. **System Analysts and Application Programmers (Software Engineers)**
System analysts determine the requirements of end users, especially naive and parametric end users, and develop specifications for canned transactions that meet these requirements.

E. **Application programmers** implement these specifications as programs; then they test, debug, document, and maintain these canned transactions. Such analysts and programmers (nowadays called **software engineers**) should be familiar with the full range of capabilities provided by the DBMS to accomplish their tasks.

II Workers behind the Scene

- **DBMS system designers** and implementers design and implement the DBMS modules and interfaces as a software package.
- **Tool developers** design and implement tools.
- Operators and maintenance personnel (**system administration personnel**) are responsible for the actual running and maintenance of the hardware and software environment for the database system.

PART - I : Advantage of Database Management System (DBMS):

Some of them are given as following below.

1. **Better Data Transferring:**

   Database management creates a place where users have an advantage of more and better managed data. Thus making it possible for end-users to have a quick look and to respond fast toany changes made in their environment.

2. **Better Data Security:**

   As number of users increases data transferring or data sharing rate also increases thus increasing the risk of data security. It is widely used in corporation world where companies invest money, time and effort in large amount to ensure data is secure and is used properly. A Database Management System (DBMS) provide a better platform for data privacy and security policies thus,helping companies to improve Data Security.

3. **Minimized Data Inconsistency:**

   Data inconsistency occurs between files when different versions of the same data appear indifferent places.

   For Example, data inconsistency occurs when a student name is saved as "John Wayne" on a main computer of school but on teacher registered system same student name is "William J. Wayne", or when the price of a product is $86.95 in local system of company and its National sales office system shows the same product price as $84.95.

   So if a database is properly designed then Data inconsistency can be greatly reduced henceminimizing data inconsistency.

4. **Faster data Access:**

   The Data base management system (DBMS) helps to produce quick answers to database queries thus making data accessing faster and more accurate. For example, to read or updatethe data. For example, end users, when dealing with large amounts of sale data, will have enhanced access to the data, enabling faster sales cycle.

   Some queries may be like:

   1. What is the increase of the sale in last three months?
   2. What is the bonus given to each of the salespeople in last five months?
   3. How many customers have credit score of 850 or more?

5. **Better decision making:**

   Due to DBMS now we have Better managed data and Improved data accessing because of which we can generate better quality information hence on this basis better decisions can bemade.

   Better Data quality improves accuracy, validity and time it takes to read data.

   DBMS does not guarantee data quality, it provides a framework to make it is easy to improvedata quality .

6. **Increased end-user productivity:**

   The data which is available with the help of combination of tools which transform data into useful information, helps end user to make quick, informative and better decisions that can make difference between success and failure in the global economy.

PART – II Advantages of Using a DBMS

1. Controlling Redundancy

This **redundancy** in storing the same data multiple times leads to several problems. First, there is the need to perform a single logical update—such as entering data on a new student multiple times. This leads to duplication of effort. Second, storage space is wasted when the same data is stored repeatedly, and this problem may be serious for large databases. Third, files that represent the same data may become **inconsistent**. This may happen because an update is applied to some of the files but not to others.

In the database approach, the views of different user groups are integrated during database design. For consistency, we should have a database design that stores each logical data item one place in the database. Controlling redundancy may be useful for improving the performance of queries.

2. Restricting Unauthorized Access

When multiple users share a database, it is likely that some users will not be authorized to access all information in the database. In addition, some users may be permitted only to retrieve data, whereas others are allowed both to retrieve and to update.

3. Providing Persistent Storage for Program Objects and Data Structures

Databases can be used to provide persistent storage for program objects and data structures. This is one of the main reasons for the emergence of the object-oriented database systems. Programming languages typically have complex data structures, such as record types in PASCAL or class definitions in C++. Object-oriented database systems are compatible with programming languages such as C++ and JAVA, and the DBMS software automatically performs any necessary conversions.

The persistent storage of program objects and data structures is an important function of database systems. Object-oriented database systems typically offer data structure compatibility with one or more object-oriented programming languages.

4. Permitting Inferencing and Actions Using Rules

Some database systems provide capabilities for defining deduction rules for inferencing new information from the stored database facts. In a traditional DBMS, an explicit procedural program code would have to be written to support such applications. But if the miniworld rules change, it is generally more convenient to change the declared deduction rules than to recode procedural programs. More powerful functionality is provided by active database systems, which provide active rules that can automatically initiate actions.

5. Providing Multiple User Interfaces

Because many types of users with varying levels of technical knowledge use a database, a DBMS should provide a variety of user interfaces. These include query languages for casual users; programming language interfaces for application programmers; forms and command codes for parametric users; and menu-driven interfaces and natural language interfaces for stand-alone users.

6. Representing Complex Relationships among Data

A database may include numerous varieties of data that are interrelated in many ways. A DBMS must have the capability to represent a variety of complex relationships among the data as well as to retrieve and update related data easily and efficiently.

7. Enforcing Integrity Constraints

Most database applications have certain integrity constraints that must hold for the data. A DBMS should provide capabilities for defining and enforcing these constraints. The simplest type of integrity constraint involves specifying a data type for each data item. Some constraints can be specified to the DBMS and automatically enforced. Other constraints may have to be checked by update programs or **at the time of data entry**.

8. 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. For example, if the computer system fails in the middle of a complex update program, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the program started executing.

9. **Potential for Enforcing Standards:** The database approach permits the DBA to define and enforce standards among database users in a large organization. The DBA can enforce standards in a centralized database environment more easily than in an environment where each user group has control of its own files and software.

10. **Reduced Application Development Time:** A prime selling feature of the database approach is that developing a new application such as the retrieval of certain data from the database for printing a new report takes very little time. Development time using a DBMS is estimated to be one-sixth to one-fourth of that for a traditional file system.

11. **Flexibility:** It may be necessary to change the structure of a database as requirements change. For example, a new user group may emerge that needs information not currently in the database. In response, it may be necessary to add a file to the database or to extend the data elements in an existing file. Modern DBMSs allow certain types of changes to the structure of the database without affecting the stored data and the existing application programs

12. **Availability of Up-to-Date Information:** A DBMS makes the database available to all users. As soon as one user's update is applied to the database, all other users can immediately seethis update. This availability of up-to-date information is essential for many transaction- processing applications, such as reservation systems or banking databases, and it is made possible by the concurrency control and recovery subsystems of a DBMS.

Applications of Database :

1. Enterprise Information

- *Sales*: For Customer, Product, And Purchase Information.?
- *Accounting*: For Payments, Receipts, Account Balances, Assets And OtherAccounting Information.
- *Human Resources*: For Information About Employees, Salaries, Payroll Taxes, And Benefits, And For Generation Of Paychecks.
- *Manufacturing*: For Management Of The Supply Chain And For Tracking Production Of Items In Factories, Inventories Of Items Inwarehouses And Stores, And Orders For Items.
- *Online Retailers*: For Sales Data Noted Above Plus Online Order Tracking, Generation Of Recommendation Lists, And Maintenance Of Online Product Evaluations.

2. Banking And Finance

- ***Banking***: For Customer Information, Accounts, Loans, And Banking Transactions.
- ***Credit Card Transactions***: For Purchases On Credit Cards And Generation OfMonthly Statements.
- ***Finance***: For Storing Information About Holdings, Sales, And Purchases Of Financial Instruments Such As Stocks And Bonds; Also For Storing Real-Time

  Market Data To Enable Online Trading By Customers And Automated Trading by the firm.

3. UNIVERSITIES:

- For student information, course registrations, and grades (in addition to standard enterprise information such as human resources and accounting).

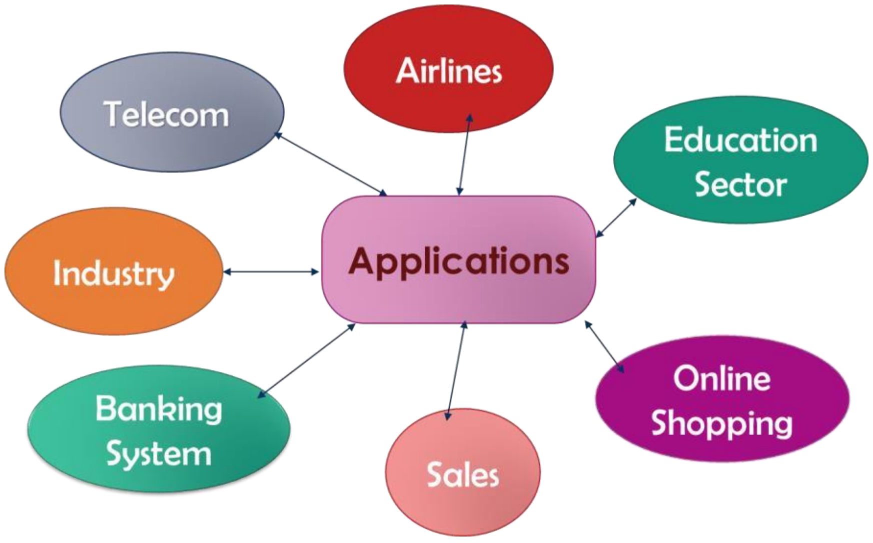
4. AIRLINES:

- For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.

5. TELECOMMUNICATION:

- For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

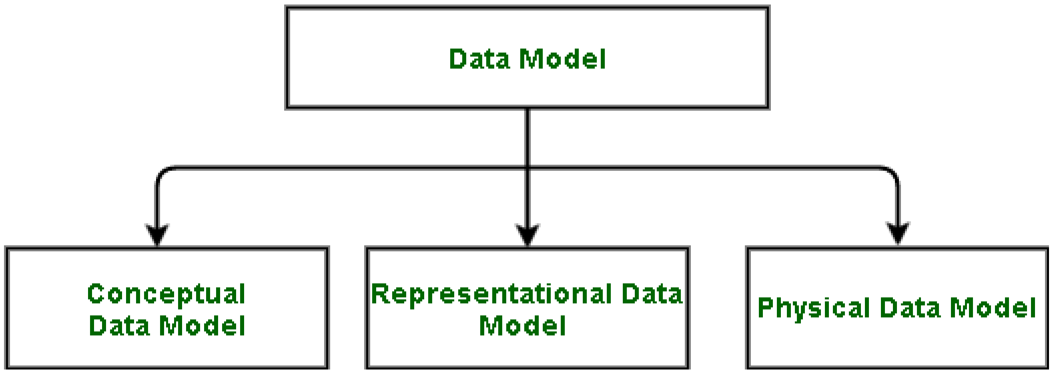
- Amazon,CNN,eBay,Facebook,Fandango,Filemaker ,(MacOS)Microsoft Access,Oracle, SAP (**S**ystems, **A**pplications & **P**roducts in Data Processing), Ticketmaster, Wikipedia Yelp, YouTube,Google ,My SQL



Data Models

A **Data Model** in Database Management System (DBMS), is the concept of tools that are developed to summarize the description of the database.

It is classified into 3 types:



1.Conceptual Model or High Level

2. Representation Model

3.Physical Model or Low Level

1. **Conceptual Data Model:**

Conceptual data model, **describes the database at a very high level** and is useful to understand the needs or requirements of the database. It is this model, that is used in the requirement gathering process i.e., before the Database Designers start making a particular database. In this model we identify **what are the Entities and attributes and their relationships between the entities** by ER Model.

2. **Representational Data Model:**

This type of data model is used to represent only the logical part of the database and does not represent the physical structure of the databases. In this we different types of data models like bellow.

1. Hierarchical Model
2. Network Model
3. Entity-Relationship Model
4. Relational Model
5. Object-Oriented Data Model

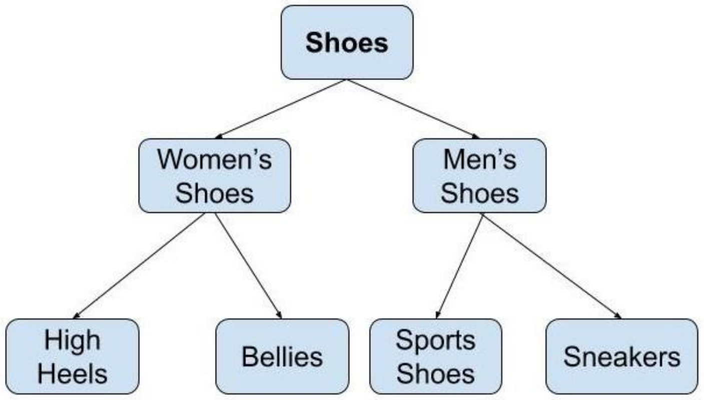
3. Physical Data Model:

Ultimately, all data in a database is stored physically on a secondary storage device such as discs and tapes. This is stored in the form of files, records and certain other data structures.

2.Representation Models:

1. Hierarchical Model

Hierarchical Model was the first DBMS model. This model organizes the data in the hierarchical tree structure. The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node. This model easily represents some of the real-world relationships like food recipes, sitemap of a website etc. ***Example:*** We can represent the relationship between the shoes present on a shopping website in the following way:



Hierarchical Model

Features of a Hierarchical Model :

1. ***One-to-many relationship:*** The data here is organized in a tree-like structure where the one-to-many relationship is between the data types. Also, there can be only one path from parent to any node. ***Example:*** In the above example, if we want to go to the node *sneakers* we only have one path to reach there i.e through men's shoes node.

2. ***Parent-Child Relationship:*** Each child node has a parent node but a parent node can have more than one child node. Multiple parents are not allowed.

3. ***Deletion Problem:*** If a parent node is deleted then the child node is automatically deleted.

4. ***Pointers:*** Pointers are used to link the parent node with the child node and are used to navigate between the stored data. *Example:* In the above example the '*shoes*' node points to the two other nodes '*women shoes*' node and '*men's shoes*' node.

Advantages of Hierarchical Model:

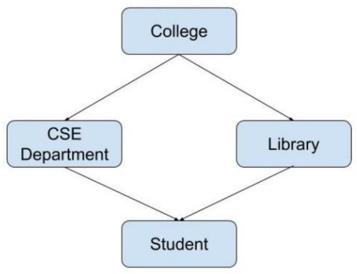
- It is very simple and fast to traverse through a tree-like structure.
- Any change in the parent node is automatically reflected in the child node so, the integrity of data is maintained.

Disadvantages of Hierarchical Model

- Complex relationships are not supported.
- As it does **not support more than one parent** of the child node so if we have some complex relationship where a child node needs to have two parent node then that can't be represented using this model.
- If **a parent node is deleted** then the child node is automatically deleted.

2. Network Model

This model is an extension of the hierarchical model. It was the most popular model before the relational model. This model is the same as the hierarchical model, the only difference is that a record can have more than one parent. It replaces the hierarchical tree with a graph. ***Example:*** In the example below, we can see that node **student has two parents** i.e. CSE Department and Library. This was earlier not possible in the hierarchical model.



Network Model

Features of a Network Model

1. ***Ability to Merge more Relationships:*** In this model, as there are more relationships so data is more related. This model has the ability to manage **one-to-one relationships** as well as **many-to-many relationships**.
2. ***Many paths:*** As there are more relationships so there can be more than one path to the same record. This makes data access fast and simple.
3. ***Circular Linked List:*** The operations on the network model are done with the help of the circular linked list. The current position is maintained with the help of a program and this position navigates through the records according to the relationship.

Advantages of Network Model

- The data can be accessed faster as compared to the hierarchical model. This is because the data is more related in the network model and there can be more than one path to reach a particular node. So the data can be accessed in many ways.
- As there is a parent-child relationship so data integrity is present. Any change in parent record is reflected in the child record.

Disadvantages of Network Model

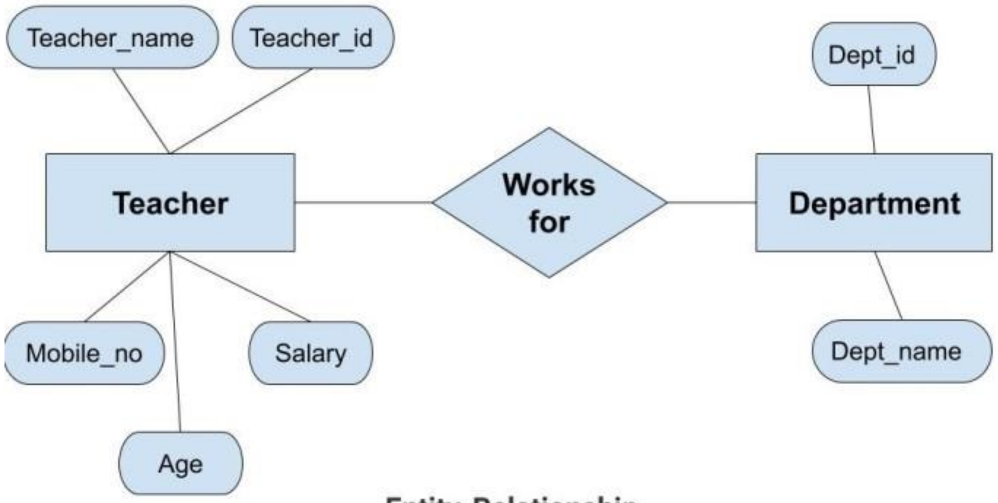
- As more and more relationships need to be handled the system might get complex. So, a user must be having detailed knowledge of the model to work with the model.
- Any change like updation, deletion, insertion is very complex.

3. Entity-Relationship Model

Entity-Relationship Model or simply ER Model is a high-level data model diagram. In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand. It is also very easy for the developers to understand the system by just looking at the ER diagram. We use the ER diagram as a visual tool to represent an ER Model. ER diagram has the following three components:

- ***Entities:*** Entity is a real-world thing. It can be a person, place, or even a concept. *Example:* Teachers, Students, Course, Building, Department, etc are some of the entities of a School Management System.
- ***Attributes:*** An entity contains a real-world property called attribute. This is the characteristics of that attribute. *Example:* The entity teacher has the property like teacher id, salary, age, etc.
- ***Relationship:*** Relationship tells how two attributes are related. *Example:* Teacher works for a department.

Example:



Entity-Relationship Model

In the above diagram, the entities are Teacher and Department. The attributes of ***Teacher*** entity are Teacher_Name, Teacher_id, Age, Salary, Mobile_Number. The attributes of entity ***Department*** entity are Dept_id, Dept_name. The two entities are connected using the relationship. Here, each teacher works for a department.

Features of ER Model

- ***Graphical Representation for Better Understanding:*** It is very easy and simple to understand so it can be used by the developers to communicate with the stakeholders.
- ***ER Diagram:*** ER diagram is used as a visual tool for representing the model.
- ***Database Design:*** This model helps the database designers to build the database and is widely used in database design.

Advantages of ER Model

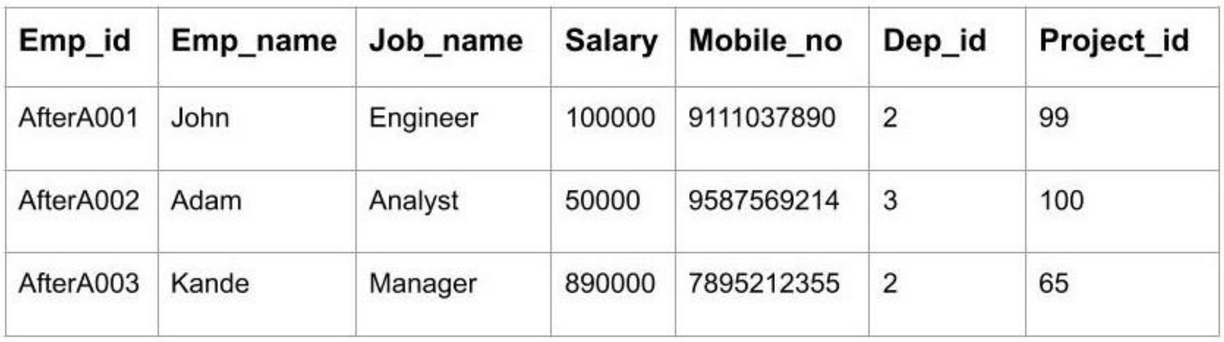
- ***Simple:*** Conceptually ER Model is very easy to build. If we know the relationship between the attributes and the entities we can easily build the ER Diagram for the model.
- ***Effective Communication Tool***: This model is used widely by the database designers for communicating their ideas.
- ***Easy Conversion to any Model***: This model maps well to the relational model and can be easily converted relational model by converting the ER model to the table. This model can also be converted to any other model like network model, hierarchical model etc.

Disadvantages of ER Model

- ***No industry standard for notation:*** There is no industry standard for developing an ER model. So one developer might use notations which are not understood by other developers.
- ***Hidden information:*** Some information might be lost or hidden in the ER model. As it is a high-level view so there are chances that some details of information might be hidden.

4. Relational Model

Relational Model is the most widely used model. In this model, the data is maintained in the form of a two-dimensional table. All the information is stored in the form of row and columns. The basic structure of a relational model is tables. So, the tables are also called *relations* in the relational model. ***Example:*** In this example, we have an Employee table.



Emp_id	Emp_name	Job_name	Salary	Mobile_no	Dep_id	Project_id
AfterA001	John	Engineer	100000	9111037890	2	99
AfterA002	Adam	Analyst	50000	9587569214	3	100
AfterA003	Kande	Manager	890000	7895212355	2	65

EMPLOYEE TABLE

Features of Relational Model

- ***Tuples / Rows / Records***: Each row in the table is called tuple. A row contains all the information about any instance of the object. In the above example, each row has all the information about any specific individual like the first row has information about John.
- ***Attribute / field /Properties / columns :*** Attributes are the property which defines the table or relation. The values of the attribute should be from the same domain. In the above example, we have different attributes of the *employee* like Salary, Mobile_no, etc.

Advantages of Relational Model

- ***Simple:*** This model is more simple as compared to the network and hierarchical model.
- ***Scalable:*** This model can be easily scaled as we can add as many rows and columns we want.
- ***Structural Independence:*** We can make changes in database structure without changing the way to access the data. When we can make changes to the database structure without affecting the capability to DBMS to access the data we can say that structural independence has been achieved.

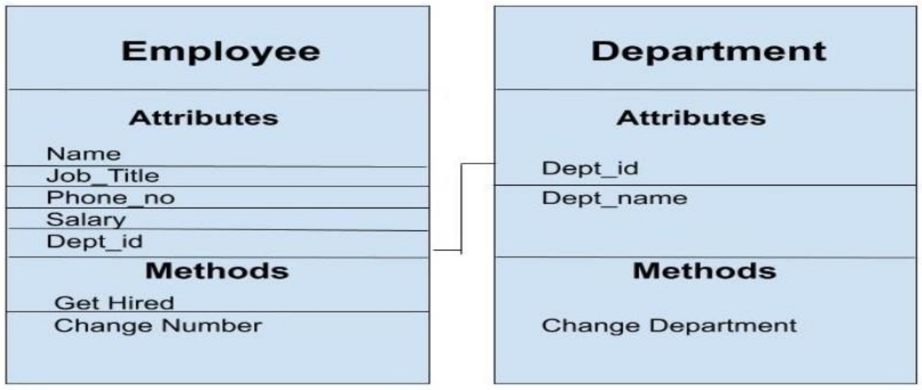
Disadvantages of Relational Model

- ***Hardware Overheads:*** For hiding the complexities and making things easier for the user this model requires more powerful hardware computers and data storage devices.
- ***Bad Design:*** As the relational model is very easy to design and use. So the users don't need to know how the data is stored in order to access it. This ease of design can lead to the development of a poor database which would slow down if the database grows.

But all these disadvantages are minor as compared to the advantages of the relational model. These problems can be avoided with the help of proper implementation and organization.

5. Object-Oriented Data Model

The real-world problems are more closely represented through the object-oriented data model. In this model, both the data and relationship are present in a single structure known as an object. We can store audio, video, images, etc in the database which was not possible in the relational model (although you can store audio and video in relational database, it is adviced not to store in the relational database). In this model, two are more objects are connected through links. We use this link to relate one object to other objects. This can be understood by the example given below.



Object_Oriented_Model

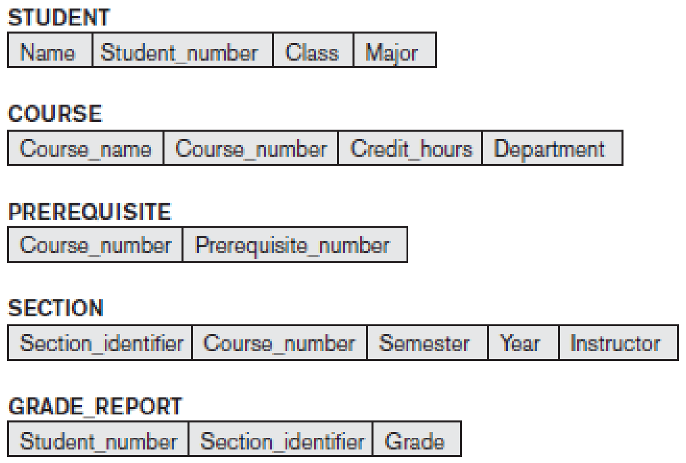
In the above example, we have two objects Employee and Department. All the data and relationships of each object are contained as a single unit. The attributes like Name, Job_title of the employee and the methods which will be performed by that object are stored as a single object. The two objects are connected through a common attribute i.e the Department_id and the communication between these two will be done with the help of this common id.

This is all about the various data model of DBMS. Hope you learned something new today.

Schemas, Instances, and Database State

The description of a database is called the **database schema,** which is specified during database design and is not expected to change frequently. A displayed schema is called a **schema diagram.** Figure 2.1 shows a schema diagram for the UNIVERSITY database.

Figure 2.1
Schema diagram for the database in Figure 1.2.


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

A schema diagram displays only some aspects of a schema, such as the names of record types and data items, and some types of constraints. Other aspects are not specified in the schema diagram. The actual data in a database may change quite frequently; for example, the database shown in Figure 1.2 changes every time we add a student or enter a new grade for a student. 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.

PART I : DBMS Architecture and Data Independence

The Three-Schema Architecture

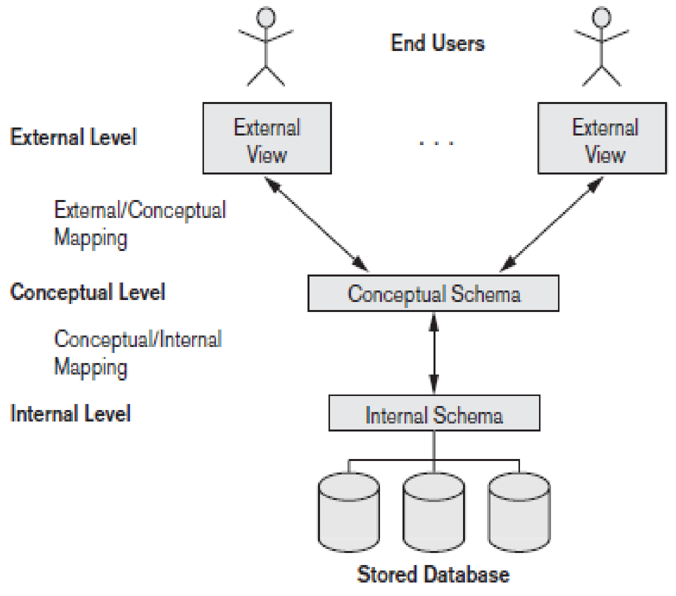
In this section we specify architecture for database systems, called the **three-schema architecture** which was proposed to achieve and visualize these characteristics. The goal of the three-schema architecture, is to separate the user applications and the physical database. In this architecture, schemas can be defined at the following three levels:

- ✓ 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.

- ✓ The **conceptual level** has a **conceptual schema,** which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. A high-level data model or an implementation data model can be used at this level.

- ✓ 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. A high-level data model or an implementation data model can be used at this level.

Figure 2.2
The three-schema architecture.



The three-schema architecture is a convenient tool for the user to visualize the schema levels in a database system. The process of transforming requests and results between levels are called **mappings.**

PART II : Three schema Architecture

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

View of Data / Data Abstraction

Physical Level:

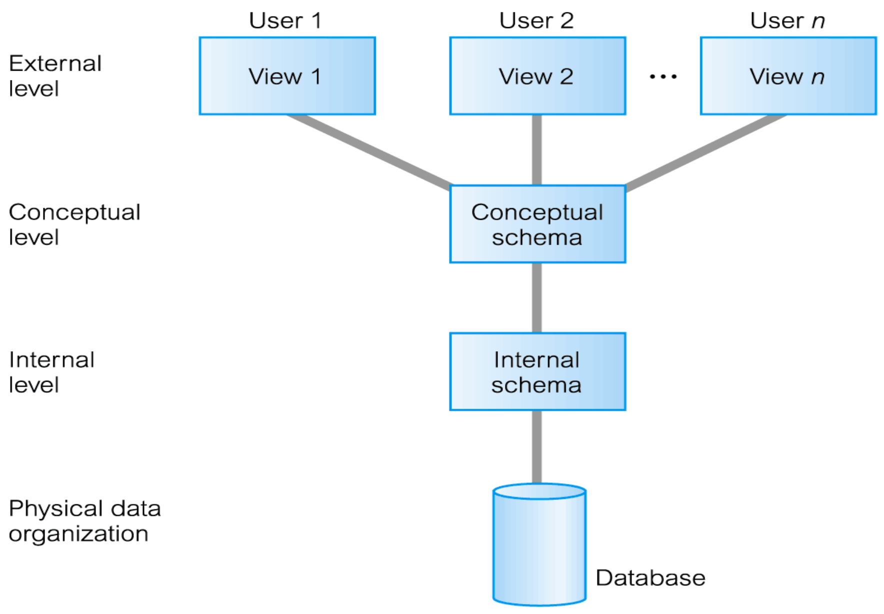
- Describes HOW data are actually stored
- It deals with the data structures used in physical
- It is otherwise called as Internal level

Logical Level:

- Describes WHAT data is stored in database and their relationships among data
- This level is usually designed by using ER diagrams
- It is otherwise called as Conceptual Level, Schema level

External Level:

- Describes only part of the database i.e. only part of the database can be viewed at atime but not the entire database
- Database has many views
- It is otherwise called as Subschema level, View level



Data Independence

- 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 thedatabase system without altering the schema at the next higher level.

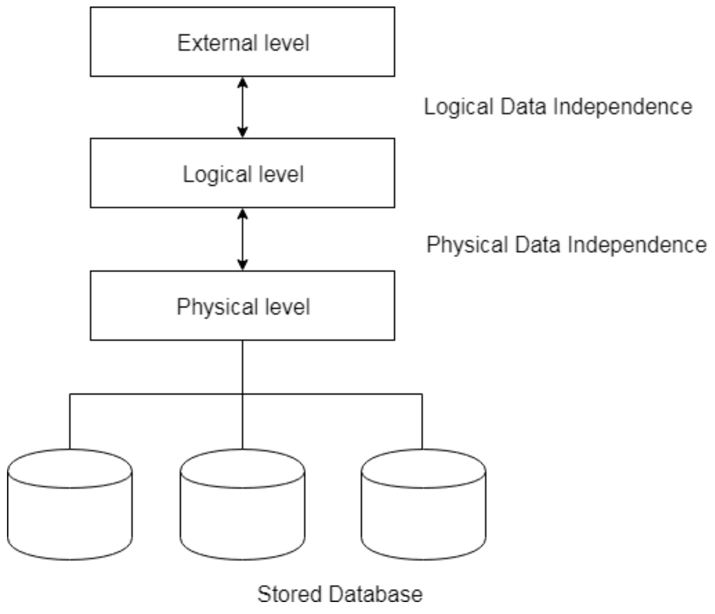
There are two types of data independence:

2. Logical Data Independence

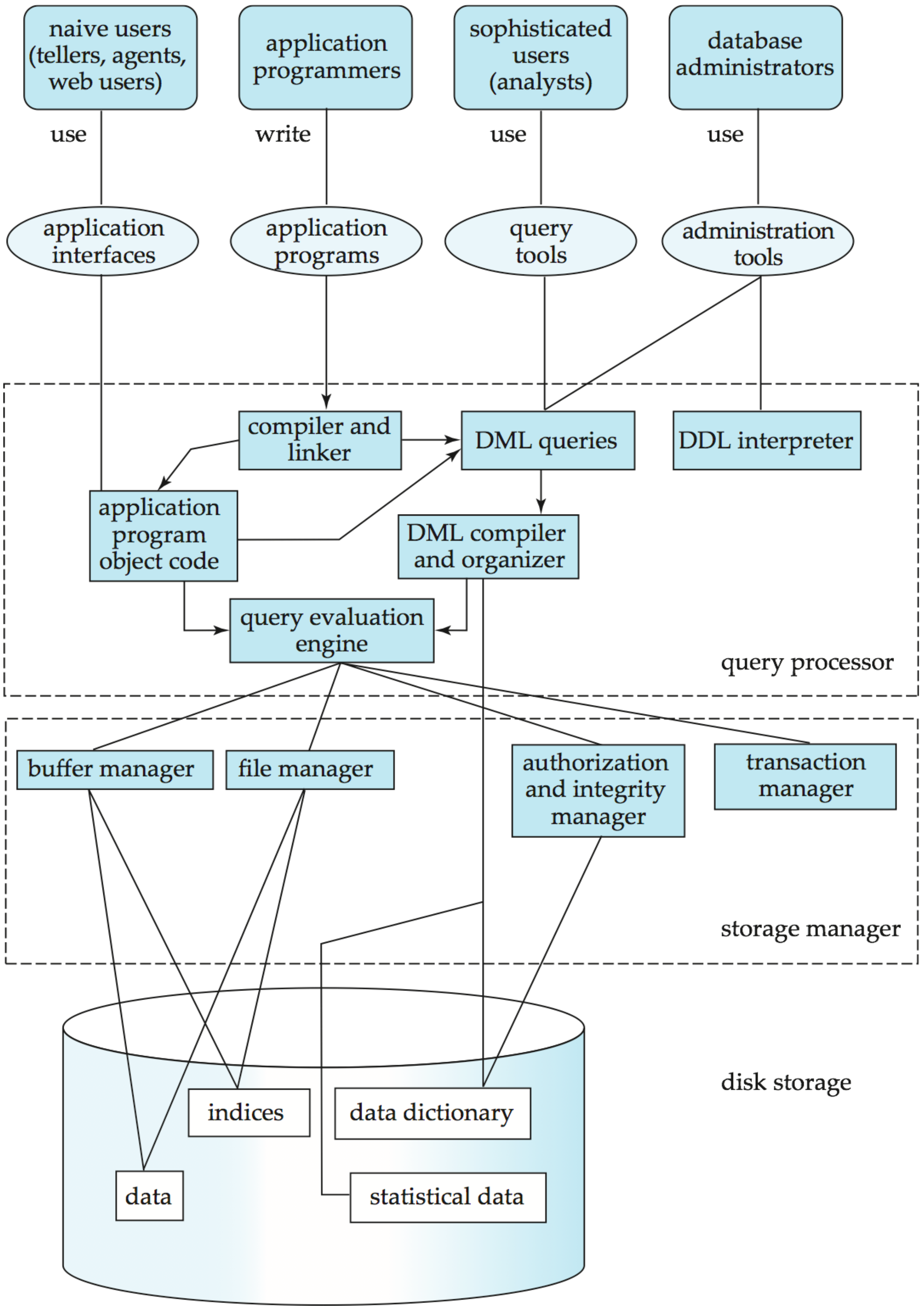
- Logical data independence refers characteristic of being able to change the conceptual schemawithout having to change the external schema.
- Logical data independence is used to separate the external level from the conceptual view.
- If we do any changes in the conceptual view of the data, then the user view of the data would notbe affected.
- Logical data independence occurs at the user interface level.

3. Physical Data Independence

- Physical data independence can be defined as the capacity to change the internal schemawithout having to change the conceptual schema.
- If we do any changes in the storage size of the database system server, then the Conceptualstructure 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.



Database System architecture:



Types of Users:

1. **Naïve Users:** These are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. The typical user interface for Naïve Users is a *form interface*, where the user can simply fill the appropriatefields of the form.
2. **Application Programmers:** Application programmers are computer professionals who write application programs. They choose many tools among available for developing user interfaces.
3. **Sophisticated Users:** They interact with the system without writing programs. They formtheir requests either using a database query language or by using tools like data analysis software. Analyst who submit queries to analyze data in database fall in this category.
4. **Specialized Users:** These users write specialized database applications that do not fit into traditional data processing framework. These applications can be knowledge base, expert systems, systems that have complex data, etc.
5. **Database Administrator:**
   Person who has central control over data and programs that access those data in a database system is called Database Administrator (DBA). The major functions of DBA are:
   a. ***Schema Definition:*** The DBA creates the original database schema by executing a set of data definition statements in the DDL.
   b. ***Storage Structure and Access Method Definition:*** The storage structure and accessing methods of data at physical storage medium is modified by DBA only.
   c. ***Schema and Physical Organization Modification:*** The DBA carries out changes to the schema and physical organization in-order to meet the requirement of organization when needed and to increase the performance of system.
   d. ***Granting of Authorization for Data Access:*** DBA regulates the access of various parts of database. Users can access only limited parts of database. A user must take authorization from DBA in-order to access the parts which theuser cannot access.
   e. ***Routine Maintenance:*** The routine activities of DBA are:
      - Periodical backups of database
      - Upgrading disk space in-order to provide space for normal operations
      - Monitoring tasks running on the database and ensuring theperformance of database without degrading

Storage Manager:

- It is a program module which provides interface between the low level data storedin the database and the application programs and queries submitted to the system.
- It is responsible for interaction with the *File Manager*.
- It translates DML commands into low level file system commands.
- It is responsible for *storing*, *retrieving* and *updating* data in database.
- It includes the following components:
  a) **Authorization and Integrity Manager:** Tests the integrity constraints andauthorization of users.
  b) **Transaction Manager:** Ensures the consistency of database despite of systemfailures by non-conflicting schemes.
  c) **File Manager:** Manages the allocation of space on disk storage and datastructures used to represent data on disk.
  d) **Buffer Manager:** It is responsible for fetching data from disk storage to mainmemory.

The Storage Manager implements several data structures at physical level of database.They are:

- **Data Files:** Stores database.
- **Data Dictionary**: Stores metadata of stored database.

Indices: Provides fast access to database. *Hashing is alternative to indexing.*

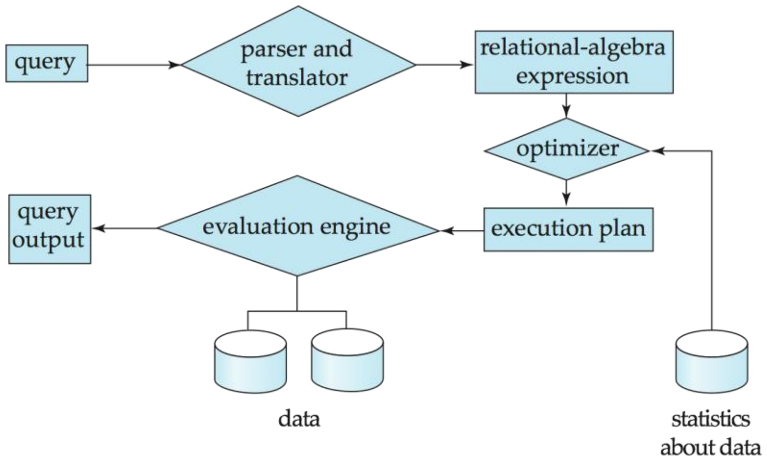
The Query Processor:

The Query Processor contains the following components:

i. **DDL Interpreter:** Interpreters DDL statements and records the definition into data dictionary.
ii. **DML Compiler:** Translates DML statements into an evaluation plan consisting of low level instructions that the query evaluation engine understands. It translates the query into a number of evaluation plans. DML Compiler performs query optimization
i.e. selects the optimal evaluation plan for execution of query.
iii. **Query Evaluation Plan:** Executes the low level instructions generated by DML compiler.

Query Processing:

1. Parsing and translation
2. Optimization
3. Evaluation


query
parser and translator
relational-algebra expression
optimizer
execution plan
evaluation engine
query output
data
statistics about data