Chapter 2

Database and Database Management System (DBMS)

1. Introduction
2. **Database management system (DBMS)**
3. **Advantages of DBMS**
4. **Architecture of DBMS**
5. **Importance of DBMS**
6. **Components of DBS environment**
7. **Data independence**
8. **Instance , schema and mapping**
9. **ACID test**

2.1 Introduction

A **database** is a collection of information that is organized so that it can easily be accessed, managed, and updated. In one view, databases can be classified according to types of content: bibliographic, full-text, numeric, and images.

In computing, databases are sometimes classified according to their organizational approach. The most prevalent approach is the relational database, a tabular database in which data is defined so that it can be reorganized and accessed in a number of different ways. A distributed database is one that can be dispersed or replicated among different points in a network. An object-oriented programming database is one that is congruent with the data defined in object classes and subclasses.

Computer databases typically contain aggregations of data records or files, such as sales transactions, product catalogs and inventories, and customer profiles. Typically, a database manager provides users the capabilities of controlling read/write access, specifying report generation, and analyzing usage. Databases and database managers are prevalent in large mainframe systems, but are also present in smaller distributed workstation and mid-range systems such as the AS/400 and on personal computers. SQL(Structured Query Language) is a standard language for making interactive queries from and updating a database such as IBM's DB2, Microsoft's SQL Server, and database products from Oracle, Sybase, and Computer Associates.

2.1.1 Database System Applications

Databases are widely used. Here are some representative applications:

- **Banking:** For customer information, accounts, and loans, and banking transactions.
- **Airlines:** For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner—terminals situated around the world accessed the central database system through phone lines and other data networks.
- Universities: For student information, course registrations, and grades.

• **Credit card transactions:** For purchases on credit cards and generation of monthly statements.
• **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
• **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
• **Sales:** For customer, product, and purchase information.
• **Manufacturing:** For management of supply chain and for tracking production of items in factories, inventories of items in warehouses/stores, and orders for items.
• **Human resources:** For information about employees, salaries, payroll taxes and benefits, and for generation of paychecks.

2.1.2 Database Administrator

One of the main reasons for using DBMSs is to have central control of both the data and the programs that access those data. A person who has such central control over the system is called a database administrator (DBA). The functions of a DBA include:
• **Schema definition.** The DBA creates the original database schema by executing a set of data definition statements in the DDL.
• **Storage structure and access-method definition.**
• **Schema and physical-organization modification.** The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.
• **Granting of authorization for data access.** By granting different types of authorization, the database administrator can regulate which parts of the database various users can access. The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.
• **Routine maintenance**. Examples of the database administrator's routine maintenance activities are:

- ❖ Periodically backing up the database, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.

- ❖ Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
- ❖ Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

2.2 Database management system (DBMS)

A **database-management system (DBMS)** is a collection of interrelated data and a set of programs to access those data. The collection of data, usually referred to as the database, contains information relevant to an enterprise. The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.
Database systems are designed to manage large bodies of information. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information. In addition, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results.

2.2.1 Advantages of DBMS

1. Database Development: It allows organizations to place control of database development in the hands of database administrators (DBAs) and other specialists.
2. Data independence: Application programs should be as independent as possible from details of data representation and storage. The DBMS can provide an abstract view of the data to insulate application code from such details.
3. Efficient data access: A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. It allows different user application programs to easily access the same database. Instead of having to write computer programs to extract information, user can ask simple questions in a query language
4. Data integrity and security: If data is always accessed through the DBMS, the DBMS can enforce :

- Integrity constraints on the data. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded.
- Also, the DBMS can enforce access controls that govern what data is visible to different classes of users.

5. Crash recovery: the DBMS protects users from the effects of system failures.
6. Data administration and Concurrent access: When several users share the data(more than one user access the database at the same time), DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time.

2.2.2 The Three-Schema Architecture

The goal of the three-schema architecture, is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels:

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 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.

Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.

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.

The following Figure 2.1 is a graphical representation of three level of architecture.

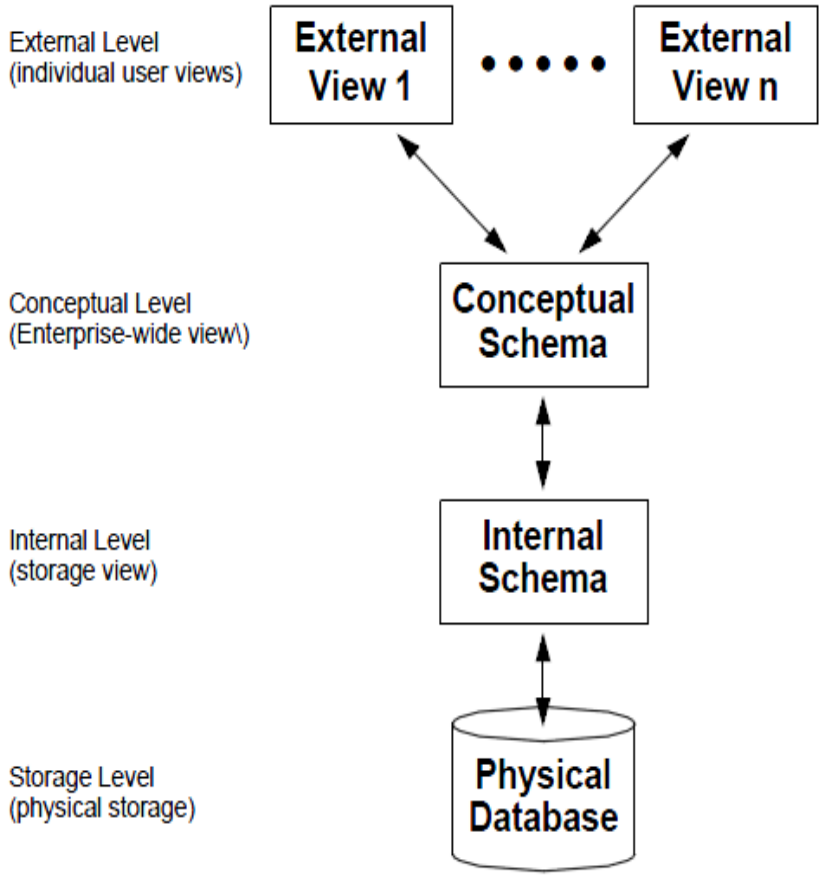


Figure 2.1 The three level of architecture

The processes of transforming requests and results between levels are called **mappings.**

2.3 Data Independence

The three-schema architecture can be used to explain the concept of data independence, which can be 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. We can define two types of data independence:

1. Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), or to reduce the database (by removing a record type or data item). In the latter case, external schemas that refer only to the remaining data should not be affected. Only the view definition and the mappings need be changed in a DBMS that supports logical data independence. Application programs that reference the external schema constructs must work as before, after the conceptual schema undergoes a logical reorganization. Changes to constraints can be applied also to the conceptual schema without affecting the external schemas or application programs.

2. Physical data independence is the capacity to change the internal schema without having to change the conceptual (or external) schemas. Changes to the internal schema may be needed because some physical files had to be reorganized—for example, by creating additional access structures—to improve the performance of retrieval or update. If the same data as before remains in the database, we should not have to change the conceptual schema.

2.4 Components of Database System Environment

The term **database system** refers to an organization of components that define and regulate the collection, storage, management, and use of data within a database environment. From a general management point of view, the database system is composed of the five major parts shown in the following Figure 2.2: hardware, software, people, procedures, and data.

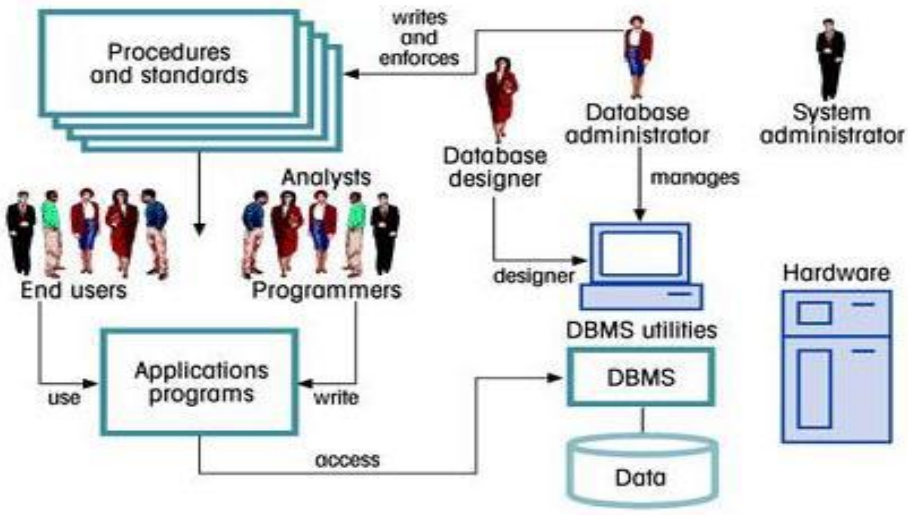


Figure 2.2 Components of database system environment

1. Hardware.

Hardware refers to all of the system's physical devices; for example, computers (PCs, workstations, servers, and supercomputers), storage devices, printers, network devices (hubs, switches, routers, fiber optics),and other devices (automated teller machines, ID readers, and so on).

2. Software.

Although the most readily identified software is the DBMS itself, to make the database system function fully, three types of software are needed: operating system software, DBMS software, and application programs and utilities.

manages all hardware components and makes it possible for all other software to run on the computers. Examples of operating system software include Microsoft Windows, Linux, MacOS, UNIX, and MVS.

b. DBMS software:

manages the database within the database system. Some examples of DBMS software include Microsoft's SQL Server, Oracle Corporation's Oracle, Sun's MySQL, and IBM's DB2.

c. Application programs and utility software:

3. People.

This component includes all users of the database system. On the basis of primary job functions, five types of users can be identified in a database system: system administrators, database administrators, database designers, system analysts and programmers, and end users. Each user type, described below, performs both unique and complementary functions.

1. **System administrators** oversee the database system's general operations.

2. **Database administrators**, also known as DBAs, manage the DBMS and ensure that the database is functioning properly.

3. **Database designers** design the database structure. They are, in effect, the database architects. If the database design is poor, even the best application programmers and the most dedicated DBAs cannot produce a useful database environment. Because organizations strive to optimize their data resources, the database

designer's job description has expanded to cover new dimensions and growing responsibilities.

4. **System analysts and programmers** design and implement the application programs. They design and create the data entry screens, reports, and procedures through which end users access and manipulate the database's data.
5. **End users** are the people who use the application programs to run the organization's daily operations. For example, salesclerks, supervisors, managers, and directors are all classified as end users. High-level end users employ the information obtained from the database to make tactical and strategic business decisions.

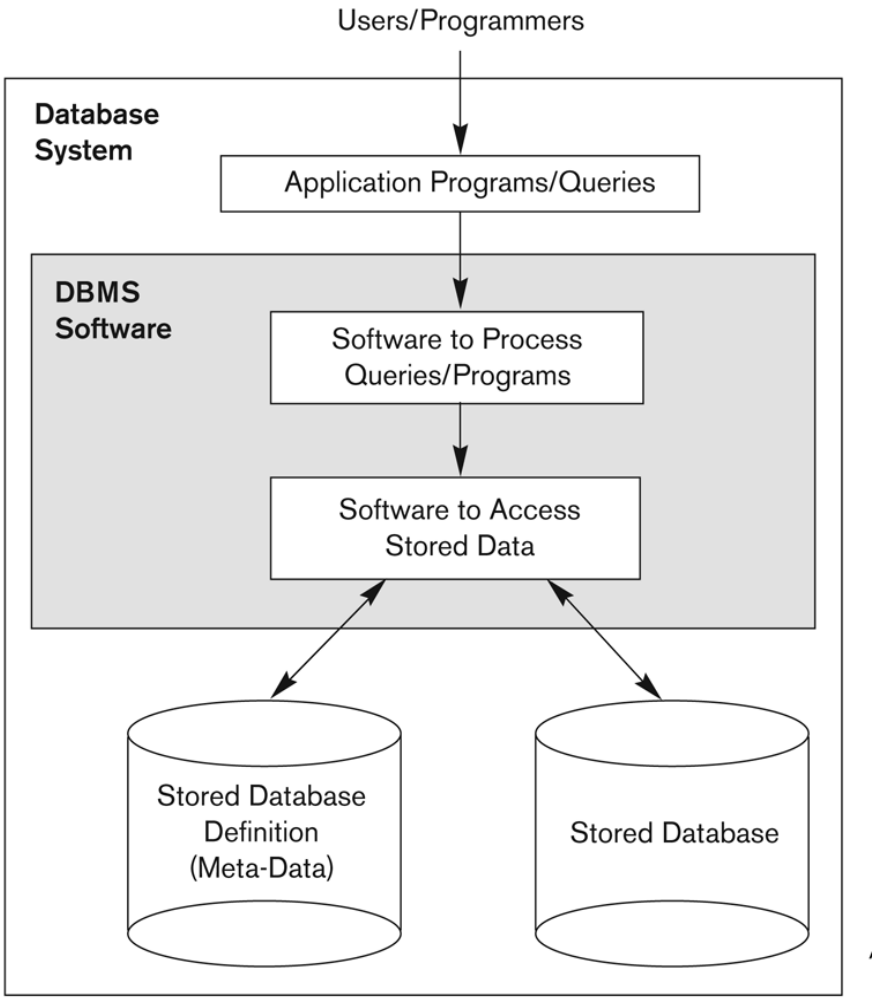
4. Procedures.

Procedures are the instructions and rules that govern the design and use of the database system. Procedures are a critical, although occasionally forgotten, component of the system. Procedures play an important role in a company because they enforce the standards by which business is conducted within the organization and with customers. Procedures are also used to ensure that there is an organized way to monitor and audit both the data that enter the database and the information that is generated through the use of those data.

5. Data.

The word data covers the collection of facts stored in the database. Because data are the raw material from which information is generated, the determination of what data are to be entered into the database and how those data are to be organized is a vital part of the database designer's job.

The following Figure 2.3 is a graphical representation of Database system.



(Database (DB) + DBMS → Database System (DBS))

Figure 2.3 Database system

2.5 Instance , schema and mapping

Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an **instance** of the database. The overall design of the database is called the database **schema**. Schemas are changed infrequently, if at all.

The concept of database schemas and instances can be understood by analogy to a program written in a programming language. A database schema corresponds to the variable declarations (along with associated type definitions) in a program. Each variable has a particular value at a

given instant. The values of the variables in a program at a point in time correspond to an *instance* of a database schema.
Database systems have several schemas, partitioned according to the levelsof abstraction:

- Physical schema describes the database design at the physical level.
- Logical schema describes the database design at the logical level.
- A database may also have several schemas at the view level, sometimes called subschemas, which describe different views of the database.

Employee-schema=(emp-id, emp-name, salary,sex,department).

Of these, the logical schema is by far the most important, in terms of its effect on application programs, since programmers construct applications by using the logical schema. The physical schema is hidden beneath the logical schema, and can usually be changed easily without affecting application programs. Application programs are said to exhibit **physical data independence** if they do not depend on the physical schema, and thus need not be rewritten if the physical schema changes.

The processes of transforming requests and results between levels are called **mappings.**

2.6 ACID Model

ACID ***(Atomicity, Consistency, Isolation, Durability)*** is a set of properties that guarantee database transactions are processed reliably. A transaction a single logical operation on the data is called a transaction. For example, a transfer of funds from one bank account to another, even though that might involve multiple changes (such as debiting one account and crediting another), is a single transaction.

The ACID model is one of the oldest and most important concepts of database theory. There are four goals that every

database management system should be achieved: atomicity, consistency, isolation and durability. No database can be considered **reliable** if the database fails to meet any of these four goals.Jim Gray defined these properties of a **reliable transaction system** in the late 1970s and developed technologies to automatically achieve them.

2.6.1 Characteristics of ACID Test

1- Atomicity

Atomicity requires that database modifications must follow an "all or nothing" rule. Each transaction is said to be atomic. If one part of the transaction fails, the entire transaction fails and the database state is unchanged. It is important that the database management system maintain the atomic nature of transactions in spite of any DBMS, operating system or hardware failure.

2- **Consistency**

Consistency states that only valid data will be written to the database. If a transaction is executed that violates the database's consistency rules, the entire transaction will be rolled back and the database will be restored to the state consistent with those rules. if a transaction successfully executes, it will take the database from one state that is consistent with the rules to another state that is also consistent with the rules.

3- Isolation

Isolation requires that multiple transactions occurring at the same time not impact each other's execution. For example, if salaam issues a transaction against a database at the same time that layla issues a different transaction; both transactions should operate on the database in an isolated manner. The database should either perform salaam's entire transaction before executing layla's or vice- versa. That leads to prevents salaam's transaction from reading intermediate data produced as a side effect of part of

layla's transaction that will not eventually be committed to the database. The isolation property does not ensure which transaction will execute first, but it will not interfere with each other. Isolation is helped to decrease the speed of this type of concurrency management. To respect the isolation property is better to use a **serial model** where no two transactions can occur on the same data at the same time and where the result is predictable.

2.7 Database Design

Database design is the process of producing a detailed data model of a database , this data model which can then be used to create a database.
The term database design can be used to describe many different parts of the design , it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and view

The Design Process

The design process consists of the following steps:

1. Determine the purpose of your database - This helps prepare you for the remaining steps.

2. Find and organize the information required - Gather all of the types of information you might want to record in the database, such as product name and order number.

3. Divide the information into tables - Divide your information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.

4. Turn information items into columns - Decide what information you want to store in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.

5. Specify primary keys - Choose each table's primary key. The primary key is a column that is used to uniquely identify each row. An example might be Product ID or Order ID.

6. Set up the table relationships - Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.
7. Refine your design - Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed.
8. Apply the normalization rules - Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed.

2.8 Database Model

A **database model** refers to the logical structure, representation or layout of a database and how the data will be stored, managed and processed within it. It helps in designing a database and serves as blueprint for application developers and database administrators in creating a database.

A database model is primarily a type of data model. Depending on the model in use, a database model can include entities, their relationships, data flow, tables and more. For example, within a hierarchal database mode, the data model organizes data in the form of a tree-like structure having parent and child segments.

Some of the popular database models include relational models, hierarchical models, flat file models, object oriented models, entity relationship models and network models.

In history of database design, three models have been in use.

- Hierarchical Model
- Network Model
- Relational Model

2.8.1 Hierarchical Model

In this model each entity has only one parent but can have several children . At the top of hierarchy there is only one entity which is called Root.

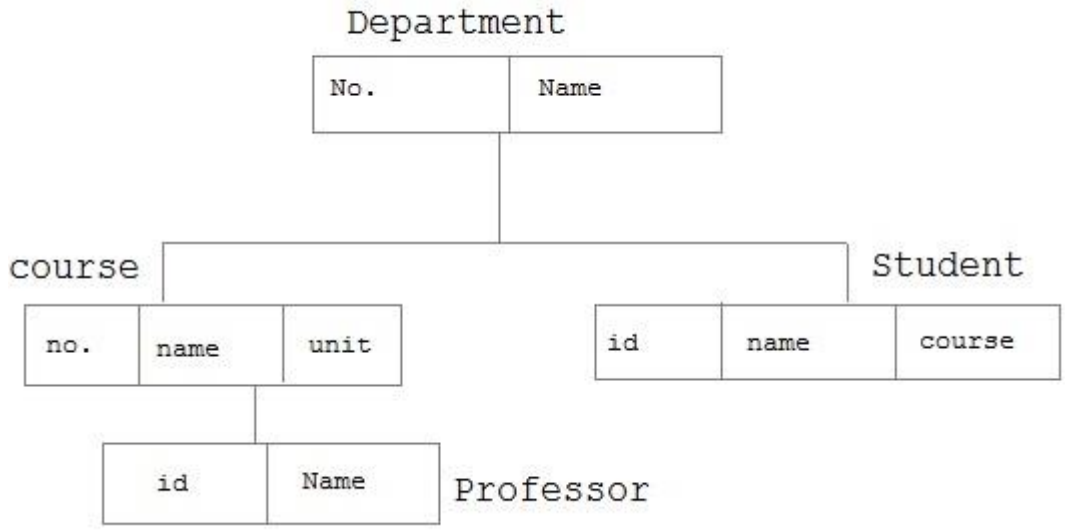


Figure 2.4 Hierarchical Model

2.8.2 Network Model

In the network model, entities are organized in a graph, in which some entities can be accessed through several path

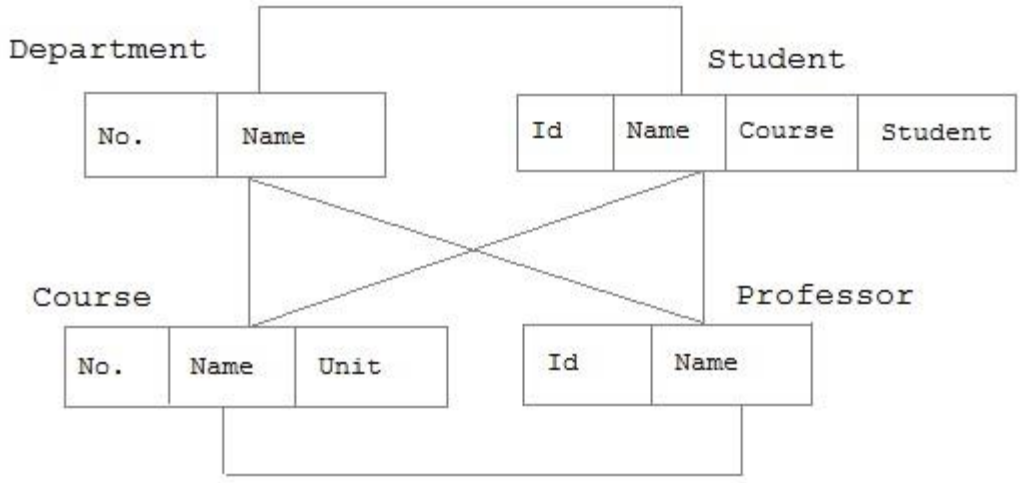


Figure 2.5 Network Model

2.8.3 Relational Model

In this model, data is organized in two-dimensional tables called relations. The tables or relation are related to each other.

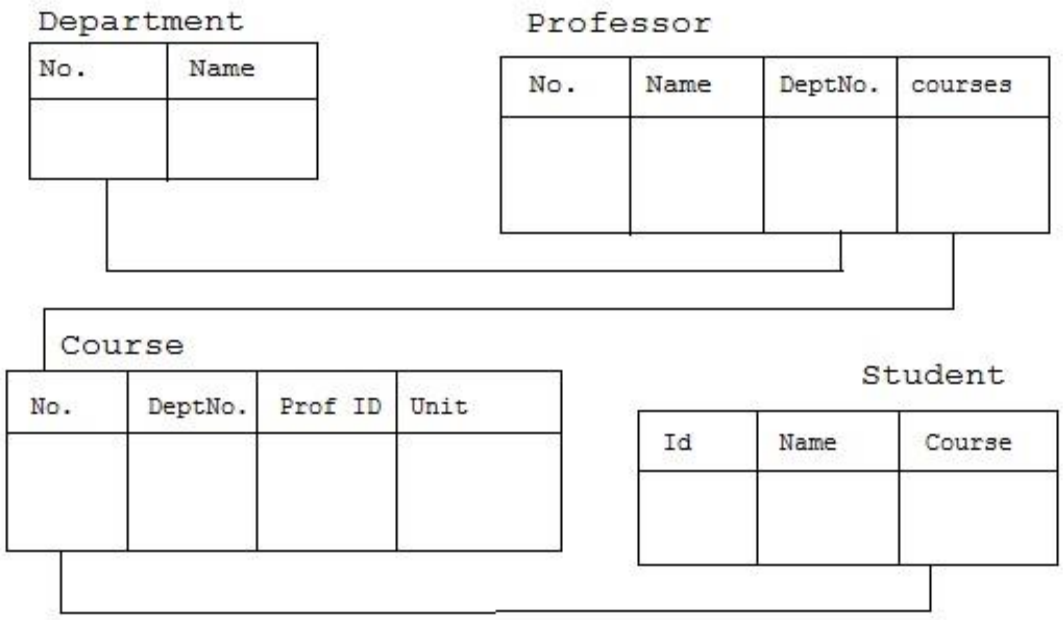


Figure 2.6 Relational Model