K.D.K College Of Engineering,Nagpur

Unit 1: Introduction to DBMS

Unit 1

1.1 Introduction to DBMS:

Definition of Data: Data, we mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know.

Database Management System (DBMS) is a combination of two words that is database & management system. Combining the meaning of both gives the definition of DBMS.

A **database-management system** (DBMS) is a collection of interrelated data and a set of programs to access those data.

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is hence a *general-purpose software system* that facilitates the processes of *defining, constructing, manipulating,* and *sharing* databases among various users and applications. Defining a database involves specifying the data types, structures, and constraints for the data to be stored in the database.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 concurrently.

General properties of database:
A database has the following implicit properties:
• A database represents some aspect of the real world, sometimes called the miniworld or the universe of discourse (DoD). Changes to the miniworld are reflected in the database.
• A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
• A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.

DATABASE SYSTEM APPLICATION:
• 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.

WHAT IS FILE PROCESSING SYSTEM?
This typical file-processing system is supported by a conventional operating system. The system stores permanent records in various files, and it needs different application programs to extract records from, and add records to, the appropriate files. Before database management systems (DBMSs) came along, organizations usually
stored information in such systems.

DRAWBACKS OF FILE PROCESSING SYSTEM:

1)Data redundancy and inconsistency: Since different programmers create the files and application programs over a long period, the various files are likely to have different formats and the programs may be written in several programming languages. Moreover, the same information may be duplicated in several places (files). For example, the address and telephone number of a particular customer may appear in a file that consists of savings-account records and in a file that consists of checking-account records. This redundancy leads
to higher storage and access cost. In addition, it may lead to data inconsistency; that is, the various copies of the same data may no longer agree. For example, a changed customer address may be reflected in savings-account records but not elsewhere in the system.
2) Difficulty in accessing data: conventional file-processing environments do not allow
needed data to be retrieved in a convenient and efficient manner .Suppose that one of the bank officers needs to find out the names of all customers who live within a particular postal-code area. The officer asks the data-processing department to generate such a list. Because the designers of the original system did not anticipate this request, there is no application program on hand to meet it. There is, however, an application program to generate the list of all customers. The bank officer has now two choices: either obtain the list of all customers manually and extract the needed information manually or ask a system programmer to write the necessary application program. Both alternatives are obviously unsatisfactory.
3) Data isolation. Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.
4)Integrity problems. The data values stored in the database must satisfy certain
types of consistency constraints. For example, the balance of a bank account
may never fall below a prescribed amount (say, $25).
5) Atomicity problems. A computer system, like any other mechanical or electrical device, is subject to failure. In many applications, it is crucial that, if a failure occurs, the data be restored to the consistent state that existed prior to the failure. Consider a program to transfer $50 from account A to account B. If a system failure occurs during the execution of the program, it is possible that the $50 was removed from account A but was not credited to account B, resulting in an inconsistent database state. That is, the funds transfer must be atomic—it must happen in its entirety or not at all. It is difficult to ensure atomicity in a conventional file-processing system.
6)Concurrent-access anomalies. For the sake of overall performance of the system and faster response, many systems allow multiple users to update the data simultaneously. In such an

environment, interaction of concurrent updates may result in inconsistent data. To guard against this possibility, the system must maintain some form of supervision. But supervision is difficult to provide because data may be accessed by many different application programs.

7) Security problems. Not every user of the database system should be able to access all the data. For example, in a banking system, payroll personnel need to see only that part of the database that has information about the various bank employees. They do not need access to information about customer accounts.But, since application programs are added to the system in an ad hoc manner, enforcing such security constraints is difficult.

Advantages of DBMS:

The following advantages perform by the DMBS.

(1) Reduce Randomly (Duplication): Centralizes control of data by the DBA avoid a necessary duplication of data & effectively reduce the total amount of data storage.

(2) Shared data: The data base allows the sharing of data under it control by any number of application programs on users.

(3) Integrity: Centralizes control can also insure that they are incorporate in DBMS to provide data integrity that is data available on single system & access by many people.

(4) Security: Data is very important to and organization & may be confidential. Such confidential data must not be accessible by unauthorized user. The DBA who has the altimeters possibilities for the data in the DBMS can ensure that proper access procedure including authentication. DBMS check the permission before provide any access to other users.

(5) Conflict Regulations: Since the data base is under the control of DBA, Hence, various user can not access any data without the permission. < Logging name, Password >.

(6) Data independent: Data can be a physical or logical both data are independent so that change occur in hardware or software can not affect the access of data.

Disadvantages of DBMS:

1) The cost of purchasing & developing is more because it is more expansive than other applications

2)backup and recovery operation are complex..

3)more workspace is required for its execution and storage.

4)excessive data entries may currupt the total data.

Functions of DBMS:

1)addition of new data.

2)sorting of data.

3)searching particular data.

4)printing particular data.

5)editing or changing sorted data.

6)deleting data.

DIFFERENCE BETWEEN FILE SYSTEM & DBMS

FILE SYSTEM	**DBMS**
2)there is inconsistency of data.	2) inconsistency of data is reduced as redundancy is reduced.
3)no provision for data security.	3)provision of data security is made.
4)no standard representation of data.	4) standard representation of data is achieved using relational data model.
1)high rate of redundancy of data exists in a typical file processing system.	1)redundancy is reduced.
5)data integrity is not there.	5)data integrity is there.
6)data can not be accessed easily.	6)data can be accessed easily through rows and columns.
7)data cannot be shared.	7) data can be shared.
8)retrieval of data is time consuming.	8) retrieval of data is easy.
9)program-data independence is not there.	9) program-data independence is there.

Data Abstraction:

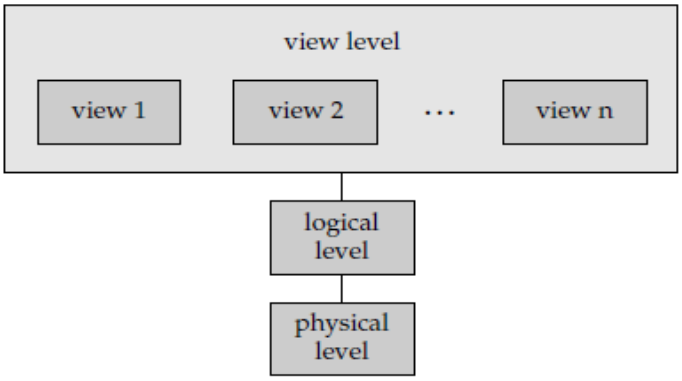


Figure 1.1 The three levels of data abstraction

For the system to be usable, it must retrieve data efficiently. The need for efficiency has led designers to use complex data structures to represent data in the database. Since many database-systems users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users' interactions with the system:

- Physical level- The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low-level data structures in detail.
- Logical level- The next-higher level of abstraction describes what data are stored in the database, and what relationships exist among those data. Database administrators, who must decide what information to keep in the database, use the logical level of abstraction.
- View level- The highest level of abstraction describes only part of the entire database. Even though the logical level uses simpler structures, complexity remains because of the variety of information stored in a large database. Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of

abstraction exists to simplify their interaction with the system. The system may provide many views for the same database.
Figure 1.1 shows the relationship among the three levels of abstraction.
Eg. An analogy to the concept of data types in programming languages may clarify the distinction among levels of abstraction. Most high-level programming languages support the notion of a record type. For example, in a Pascal-like language, we may declare a record as follows:

```
type customer = record
customer-id : string;
customer-name : string;
customer-street : string;
customer-city : string;
end;
```

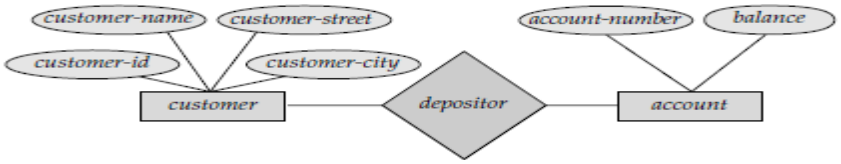
This code defines a new record type called customer with four fields. Each field has a name and a type associated with it. A banking enterprise may have several such record types, including
• account, with fields account-number and balance
• employee, with fields employee-name and salary

*At the physical level, a customer, account, or employee record can be described as a block of consecutive storage locations (for example, words or bytes). The language compiler hides this level of detail from programmers.
*At the logical level, each such record is described by a type definition, as in the previous code segment, and the interrelationship of these record types is defined as well. Programmers using a programming language work at this level of abstraction. Similarly, database administrators usually work at this level of abstraction.
*Finally, at the view level, computer users see a set of application programs that hide details of the data types. Similarly, at the view level, several views of the database are defined, and database users see these views. In addition to hiding details of the logical level of the database, the views also provide a security mechanism to prevent users from accessing certain parts of the database. For example, tellers in a bank see only that part of the database that has information on customer accounts; they cannot access information about salaries of employees.

DATA MODELS:
Underlying the structure of a database is the data model: a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.
A data model provides a way to describe the design of database at physical, logical and view level.
1) Object based logical model
2) Record based logical model
3) Physical data model

Object base logical model :
Object based logical model are used in describing data at the conceptual & view levels. There are different model use as object base logical model.
a) The entity relationship model
b) The oriented model

c) Semantic data model
d) Functional data model.
a)The entity relationship model:
The entity relationship model The entity-relationship (E-R) data model is based on a perception of a real world that consists of a collection of basic objects, called entities, and of relationships among these objects.A relationship is an association among several entities. For example, a depositor relationship associates a customer with each account that she has. The set of all entities of the same type and the set of all relationships of the same type are termed an entity set and relationship set, respectively.The overall logical structure (schema) of a database can be expressed graphically by an E-R diagram, which is built up from the following components:
• Rectangles - which represent entity sets
• Ellipses - which represent attributes
• Diamonds - which represent relationships among entity sets
• Lines - which link attributes to entity sets and entity sets to relationships.



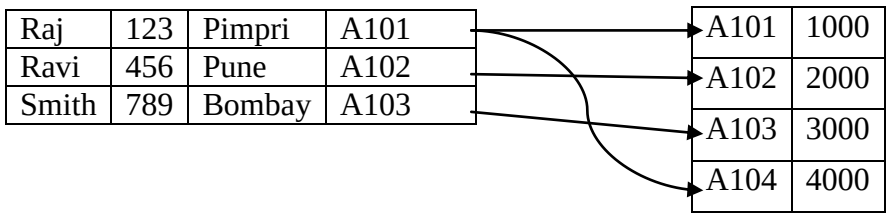
b)Object oriented model : This model is based on a collection of object. An object contains values store in a variable with the object. The object oriented model accept the values from variables & functions.
2)Record based logical model :
Record based logical model are used at the conpetual & view levels. It use all the object oriented base logical data. They are classified in 3 groups.
i)Relational model
ii)Network model .
iii)Hierarchical model
i)Relational model : Relational model represent data & relationship among data by a collection of table. Each table contains number of rows & columns & they are logically group up into the single unit.
E.g.:
Customer Table

Name	SSN	Street	Account No.
Raj	123	Pimpri	A101
Ravi	456	Pune	A!02
Smith	789	Bombay	A103
Raj	123	North	A104

Account Table

Account No.	Balance
A101	1000
A102	2000
A103	3000
A104	4000

ii) Network model : Data in the network model are represented by collection of records & relationship as parent-child. It records or organize in a arbitrary graphs. Hence, each node logically connect to each other.



iii)Hierarchical Model : Data in the hierarchical model represented as a tree & manage all data in parent – child relationship. All data internally connected to each other but their relations as a tree structure.
(Refer fig. from labmanual)

3) Physical data model

Physical data model are used to described data at the lowest level. This data model concered with physical data available in disk & stored the structure with its column name & data type. These physical data model are consist of following models.

1) unifying model.
2) frame memory model.

Some Important Defination:

Database Schema : The description of a database is called the database schema, which is specified during database design and is not expected to change frcquentlv.

Schemas Diagrams : Most data models have certain conventions for displaying schemas as diagrams. A displayed schema is called a schema diagram.

Instances / Database State :The data in the database at a particular moment in time is called a database state or snapshot. It is also called the current set of occurrences or instances in the database.

The Three-Schema Architecture:

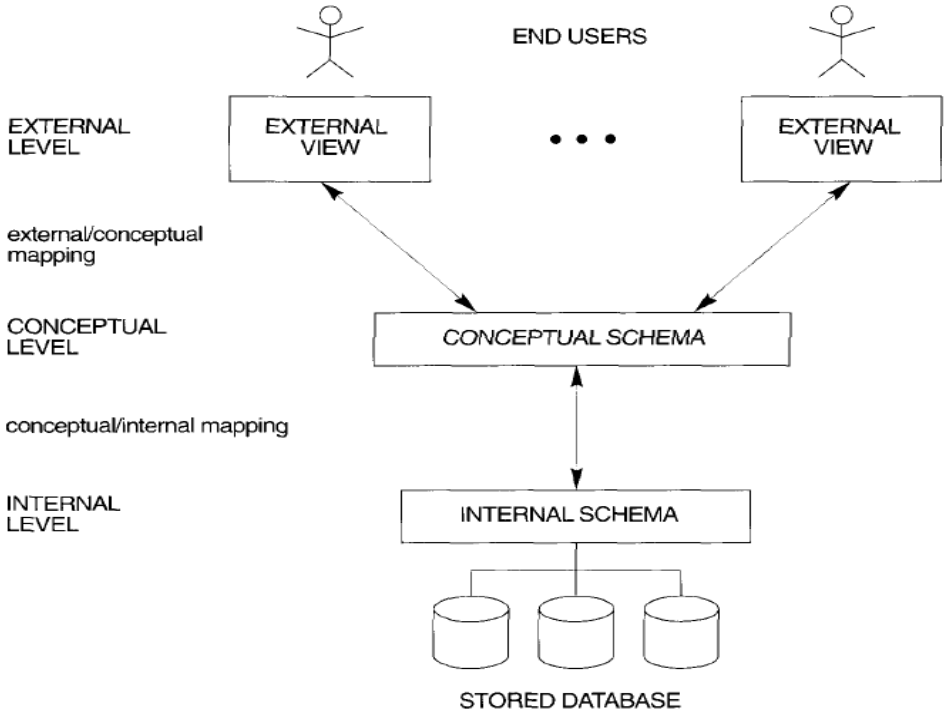


FIGURE 2.2 The three-schema architecture

The goal of the three-schema architecture, illustrated in Figure 2.2, is to separate the user applications and 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.
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.

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

Data Independence:

The three-schema architecture can be used to further explain the concept of data independence.

Data Independence:Is 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 schernas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).

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

Database Languages:

A database system provides a data definition language to specify the database schema and a data manipulation language to express database queries and updates.

Data-Definition Language

-We specify a database schema by a set of definitions expressed by a special language called a data-definition language (DDL).

For instance, the following statement in the SQL language defines the account table:

```
create table account
        (account-number char(10),
        balance integer)
```

Execution of the above DDL statement creates the account table. In addition, it updates a special set of tables called the data dictionary or data directory.

Data-Manipulation Language

Data manipulation is

- The retrieval of information stored in the database
- The insertion of new information into the database
- The deletion of information from the database
- The modification of information stored in the database

-A data-manipulation language (DML) is a language that enables users to access or manipulate data as organized by the appropriate data model.

There are basically two types:

- Procedural DMLs require a user to specify what data are needed and how to get those data.
- Declarative DMLs (also referred to as nonprocedural DMLs) require a user to specify what data are needed without specifying how to get those data.

*A **query** is a statement requesting the retrieval of information. The portion of a DML that involves information retrieval is called a **query language**.

Database Users:

There are four different types of database-system users, differentiated by the way they expect to interact with the system.

1) Naive users: are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. Forexample, a bank teller who needs to transfer $50 from account A to account B invokes a program called transfer. This program asks the teller for the amount of money to be transferred, the account from which the money is to be transferred, and the account to which the money is to be transferred.

2) Application programmers :are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces. Rapid application development (RAD) tools are tools that enable an application programmer to construct forms and reports without writing a program.

3) Sophisticated users: interact with the system without writing programs. Instead, they form their requests in a database query language. They submiteach such query to a query processor, whose function is to break down DML statements into instructions that the storage manager understands. Analysts who submit queries to explore data in the database fall in this category.

4) Specialized users :are sophisticated users who write specialized databaseapplications that do not fit into the traditional data-processing framework. Among these applications are computer-aided design systems, knowledge base and expert systems, systems that store data with complex data types (for example, graphics data and audio data), and environment-modeling systems.

Database Administrator & Responsibilities of DBA:

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.

Overall architecture of DBMS:

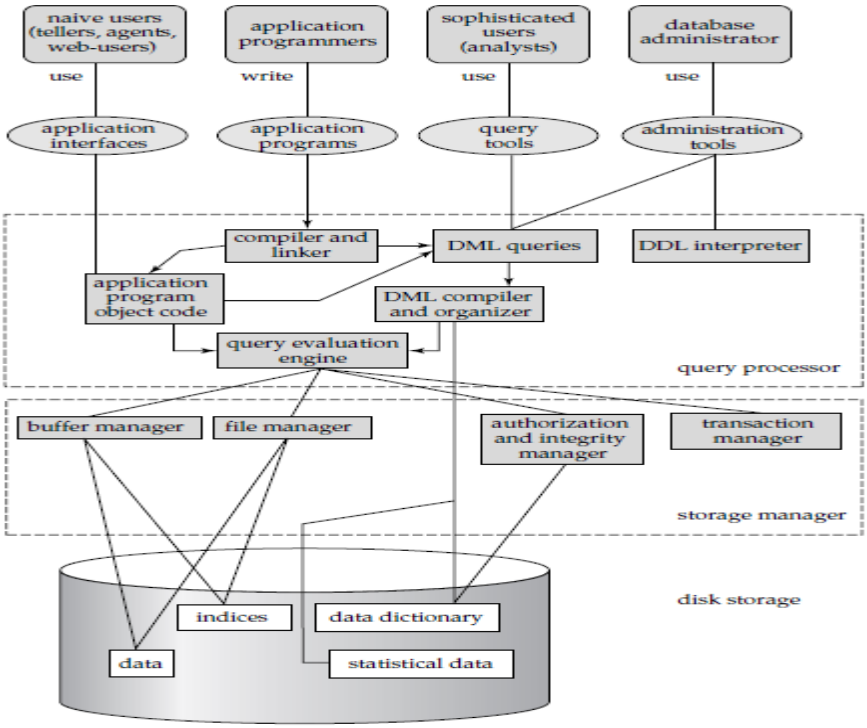


Fig: System Structure

A database system is partitioned into modules that deal with each of the responsibilities of the overall system. The functional components of a database system can be broadly divided into :

i)storage manager

ii)query processor components.

i) 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. The raw data are stored on the disk using the file system, which is usually provided by a conventional operating system. The storage manager translates the various DML statements into low-level file-system commands. Thus, the storage manager is responsible for storing, retrieving, and updating data in the database.

The storage manager components include • Authorization and integrity manager-which tests for the satisfaction of integrity constraints and checks the authority of users to access data.

• Transaction manager- which ensures that the database remains in a consistent (correct) state despite system failures, and that concurrent transaction executions proceed without conflicting.

• File manager- which manages the allocation of space on disk storage and the data structures used to represent information stored on disk.

• Buffer manager- which is responsible for fetching data from disk storage into main memory, and deciding what data to cache in main memory. The buffer manager is a critical part of the database system, since it enables the database

to handle data sizes that are much larger than the size of main memory.
The storage manager implements several data structures as part of the physical system implementation:
• Data files- which store the database itself.
• Data dictionary- which stores metadata about the structure of the database, in Particular the schema of the database.
• Indices- which provide fast access to data items that hold particular values.

ii)The Query Processor :
The query processor components include,
• DDL interpreter - which interprets DDL statements and records the definitions in the data dictionary.
• DML compiler - which translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.
A query can usually be translated into any of a number of alternative evaluation plans that all give the same result. The DML compiler also performs query optimization, that is, it picks the lowest cost evaluation plan from among the alternatives.
• Query evaluation engine - which executes low-level instructions generated by the DML compiler.