Overview of Database Management System : Introduction, Data and Information, Database, Database Management System , Objectives of DBMS, Evolution of Database Management Systems , Classification of Database Management System, File-Based System , Drawbacks of File-Based System, Advantages of DBMS, Ansi /Spark Data, Data Models, Components and Interfaces of Database Management System, Database Architecture , Situations where DBMS is not Necessary, DBMS Vendors and their Products.

1.1 Introduction

Science, business, education, economy, law, culture, all areas of human development "work" with the constant aid of data. Databases play a crucial role within science research.

There are databases collecting all sorts of different data: nuclear structure, and genes sequences (the Human Genome Database), prisoners' DNA data ("DNA offender database"), names of people accused for drug offenses, telephone numbers, legal materials and many others.

1.2 Data and Information

Data are raw facts and images that constitute building block of information. Data are the heart of the DBMS. It is to be noted that **information** is obtained from processed data. In other words, data has to be interpreted in order to obtain information.

Data are a representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means.

The data in DBMS can be broadly classified into two types,

i. one is the collection of information needed by the organization and
ii. The other is "metadata" which is the information about the database.

1.3 Database

A database is a well-organized collection of data that are related in a meaningful way, which can be accessed in different logical orders. The simplified view of database system is shown in Fig. 1.1. From this figure, it is clear that several users can access the data in an organization still the integrity of the data should be maintained. A database is integrated when same information is not recorded in two places.

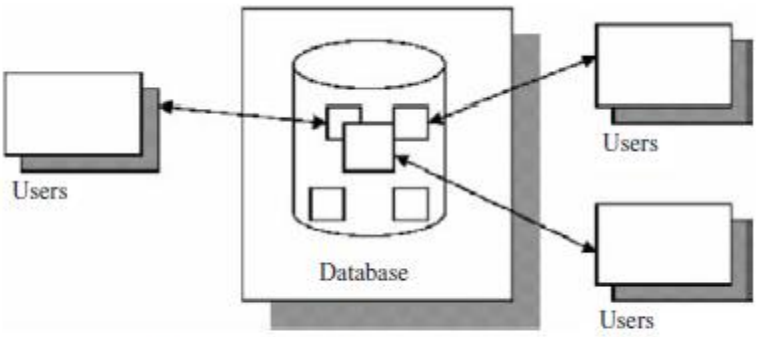


Fig. 1.1. Simplified database view

1.4 Database Management System

A database management system (DBMS) consists of collection of interrelated data and a set of programs to access that data. It is software that is helpful in maintaining and utilizing a database. A DBMS consists of:

– A collection of interrelated and persistent data. This part of DBMS is referred to as database (DB).

– A set of application programs used to access, update, and manage data. This part constitutes data management system (MS).

DBMS is a complex system that allows a user to do many things to data as shown in Fig. 1.2. From this figure, it is evident that DBMS allows user to input data, share the data, edit the data, manipulate the data, and display the data in the database. Because a DBMS allows more than one user to share the data; the complexity extends to its design and implementation.

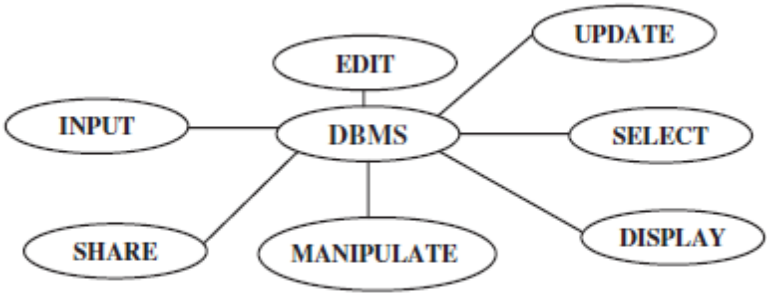


Fig. 1.2. Capabilities of database management system

1.4.1 Structure of DBMS

An overview of the structure of database management system is shown in Fig. 1.3. A DBMS is a software package, which translates data from its logical representation to its physical representation and back. The DBMS uses an application specific database description to define this translation. The database description is generated by a database designer from his or her conceptual view of the database, which is called the Conceptual Schema. The translation from the conceptual schema to the database description is performed using a data definition language (DDL) or a graphical or textual design interface.

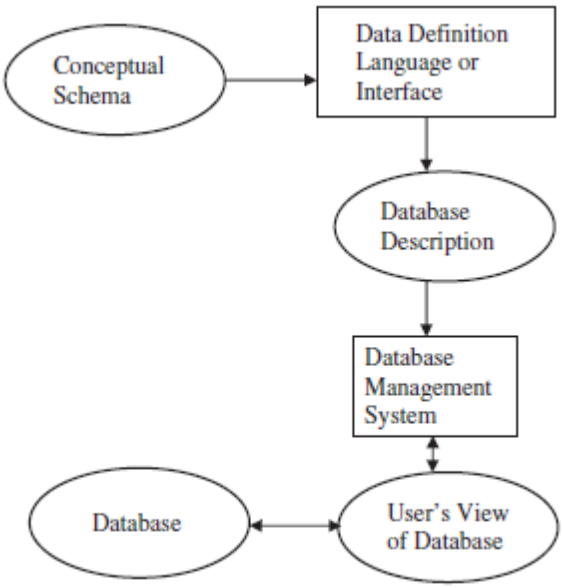


Fig. 1.3. Structure of database management system

1.5 Objectives of DBMS

- The main objectives of database management system are
  - data availability
  - data integrity
  - data security
  - Data independence.
- **Data Availability**
  Data availability refers to the fact that the data are made available to wide variety of users in a meaningful format at reasonable cost so that the users can easily access the data.
- **Data Integrity**
  Data integrity refers to the correctness of the data in the database. In other words, the data available in the database is a reliable data.
- **Data Security**
  Data security refers to the fact that only authorized users can access the data. Data security can be enforced by passwords. If two separate users are accessing a particular data at the same time, the DBMS must not allow them to make conflicting changes.
- **Data Independence**
  - Data Independence is defined as a property of DBMS that helps you to change the Database schema at one level of a database system without requiring to change the schema at the next higher level.
  - Data independence helps you to keep data separated from all programs that make use of it.

Evolution of Database Management Systems

- The chronological order of the development of DBMS is as follows:
  – Flat files – 1960s–1980s
  – Hierarchical – 1970s–1990s

– Network – 1970s–1990s
– Relational – 1980s–present
– Object-oriented – 1990s–present
– Object-relational – 1990s–present
– Data warehousing – 1980s–present
– Web-enabled – 1990s–present

1.7 Classification of Database Management System

The database management system can be broadly classified into

(1) Passive Database Management System and
(2) Active Database Management System

Passive Database Management System

- Passive Database Management Systems are program-driven.
- In passive database management system the users query the current state of database and retrieve the information currently available in the database.
- Applications send requests for operations to be performed by the DBMS and wait for the DBMS to confirm and return any possible answers.
- The operations can be definitions and updates of the schema, as well as queries and updates of the data.

Active Database Management System.

- Active Database Management Systems are data-driven or event-driven systems.
- In active database management system, the users specify to the DBMS the information they need.
- If the information of interest is currently available, the DBMS actively monitors the arrival of the desired information and provides it to the relevant users.
- An active DBMS reverses the control flow between applications and the DBMS instead of only applications calling the DBMS, the DBMS may also call applications in an active DBMS.

1.8 File-Based System

- Prior to DBMS, file system provided by OS was used to store information.
- File system is basically a way of arranging the files in a storage medium like hard disk.
- File system organizes the files and helps in retrieval of files when they are required.
- File systems consists of different files which are grouped into directories.
- The directories further contain other folders and files.
- File system performs basic operations like management, file naming, giving access rules etc.
- Consider University database, the University database contains details about student, faculty, lists of courses offered, and duration of course, etc.
- In File-based processing for each database there is separate application program which is shown in Fig. 1.4.
- One group of users may be interested in knowing the courses offered by the university.
- One group of users may be interested in knowing the faculty information.
- The information is stored in separate files and separate applications programs are written.

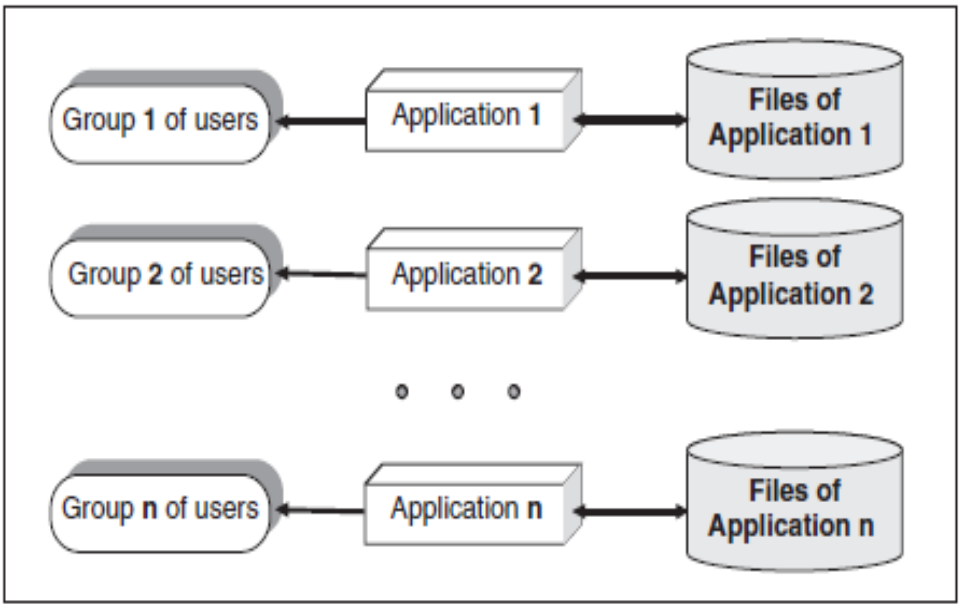


Fig. 1.4. File-based System

1.9 Drawbacks of File-Based System

- The limitations of file-based approach are
  - Duplication of data
  - Data dependence
  - Incompatible file formats
  - Separation and Isolation of data

1.9.1 Duplication of Data

- Duplication of data means same data being stored more than once.
- This can also be termed as data redundancy.
  - Data redundancy is a problem in file based approach due to the decentralized approach.
- The main drawbacks of duplication of data are:
  - Duplication of data leads to wastage of storage space. If the storage space is wasted it will have a direct impact on cost. The cost will increase.
  - Duplication of data can lead to loss of data integrity; the data are no longer consistent.

1.9.2 Data Dependence

- Data dependence means the application program depends on the data.
  - If some modifications have to be made in the data, then the application program has to be rewritten.
  - But in file-based system there is program-data dependence.

1.9.3 Incompatible File Formats

- As file-based system lacks program data independence, the structure of the file depends on the application programming language.
  - For example, the structure of the file generated by FORTRAN program may be different from the structure of a file generated by "C" program. The incompatibility of such files makes them difficult to process jointly.

1.9.4 Separation and Isolation of Data

- In file-based approach, data are isolated in separate files. Hence it is difficult to access data. The application programmer must synchronize the processing of two files to ensure that the correct data are extracted. This difficulty is more if data has to be retrieved from more than two files.
- The draw backs of conventional file-based approach are summarized later:

  1. We have to store the information in a secondary memory such as a disk. If the volume of information is large; it will occupy more memory space.
  2. We have to depend on the addressing facilities of the system. If the database is very large, then it is difficult to address the whole set of records.
  3. For each query, for example the address of the student and the list of electives that the student has chosen, we have to write separate programs.
  4. While writing several programs, lot of variables will be declared and it will occupy some space.
  5. It is difficult to ensure the integrity and consistency of the data when more than one program accesses some file and changes the data.
  6. In case of a system crash, it becomes hard to bring back the data to a consistent state.
  7. "Data redundancy" occurs when identical data are distributed over various files.
  8. Data distributed in various files may be in different formats hence it is difficult to share data among different application (Data Isolation).

1.10 DBMS Approach

- DBMS is software that provides a set of primitives for defining, accessing, and manipulating data.
- In DBMS approach, the same data are being shared by different application programs; as a result data redundancy is minimized. The DBMS approach of data access is shown in Fig. 1.5.

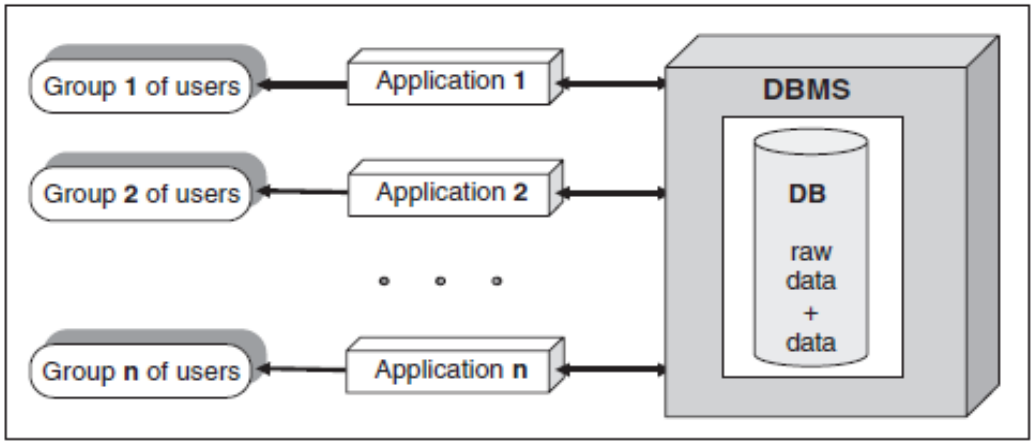


Fig. 1.5. Data access through DBMS

1.11 Advantages of DBMS

- There are many advantages of database management system. Some of the advantages are listed later:
  - 1. Centralized data management.
  - 2. Data Independence.
  - 3. Data Inconsistency.

1.11.1 Centralized Data Management

- In DBMS all files are integrated into one system thus reducing redundancies and making data management more efficient.

1.11.2 Data Independence

- Data independence means that programs are isolated from changes in the way the data are structured and stored.
- When changes are made to the data representation, the data maintained by the DBMS is changed but the DBMS continues to provide data to application programs in the previously used way.
- If major changes were to be made to the data, the application programs may need to be rewritten.
- Data independence can be physical data independence or logical data independence.
    - Physical data independence is the ability to modify physical schema without causing the conceptual schema or application programs to be rewritten.
    - Logical data independence is the ability to modify the conceptual schema without having to change the external schemas or application programs.

1.11.3 Data Inconsistency

- Data inconsistency means different copies of the same data will have different values.
- DBMS is designed to have data consistency. Some of the qualities achieved in DBMS are:
    - 1. Data redundancy −→ *Reduced in DBMS.*
    - 2. Data independence −→ *Activated in DBMS.*
    - 3. Data inconsistency −→ *Avoided in DBMS.*
    - 4. Centralizing the data −→ *Achieved in DBMS.*
    - 5. Data integrity −→ *Necessary for efficient Transaction.*
    - 6. Support for multiple views −→ *Necessary for security reasons.*

1.12 Ansi/Spark Data Model (American National Standard Institute / Standards Planning and Requirements Committee)

- The distinction between the logical and physical representation of data were recognized in 1978 when ANSI/SPARK committee proposed a generalized framework for database systems.
- This framework provided a three-level architecture, three levels of abstraction at which the database could be viewed.

1.12.1 Need for Abstraction

- The main objective of DBMS is to store and retrieve information efficiently; all the users should be able to access same data.
- The designers use complex data structure to represent the data, so that data can be efficiently stored and retrieved, but it is not necessary for the users to know physical database storage details. The developers hide the complexity from users through several levels of abstraction.

1.12.2 Data Independence

- Data independence means the internal structure of database should be unaffected by changes to physical aspects of storage. Because of data independence, the Database administrator can change the database storage structures without affecting the users view. The different levels of data abstraction are:
- 1. Physical level or internal level
- 2. Logical level or conceptual level
- 3. View level or external level

Physical Level

- It is concerned with the physical storage of the information. It provides the internal view of the actual physical storage of data. The physical level describes complex low-level data structures in detail.

Logical Level

- Logical level describes what data are stored in the database and what relationships exist among those data.

- Logical level describes the entire database in terms of a small number of simple structures. The implementation of simple structure of the logical level may involve complex physical level structures; the user of the logical level does not need to be aware of this complexity. Database administrator uses the logical level of abstraction.

View Level

- View level is the highest level of abstraction. It is the view that the individual user of the database has. There can be many view level abstractions of the same data. The different levels of data abstraction are shown in Fig. 1.6.

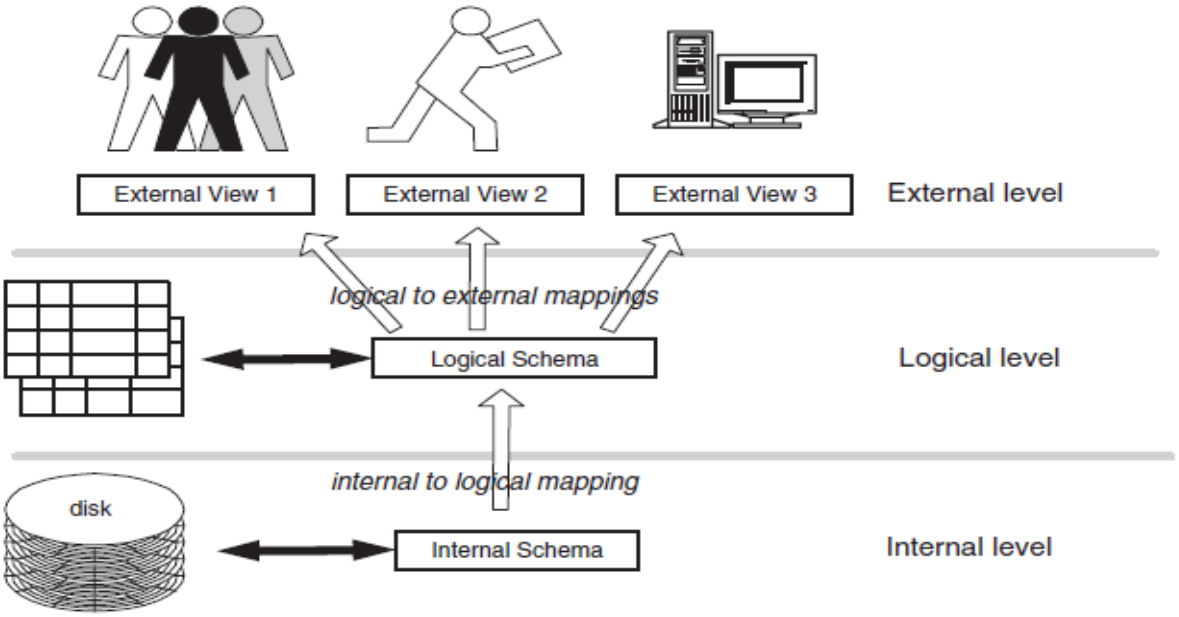
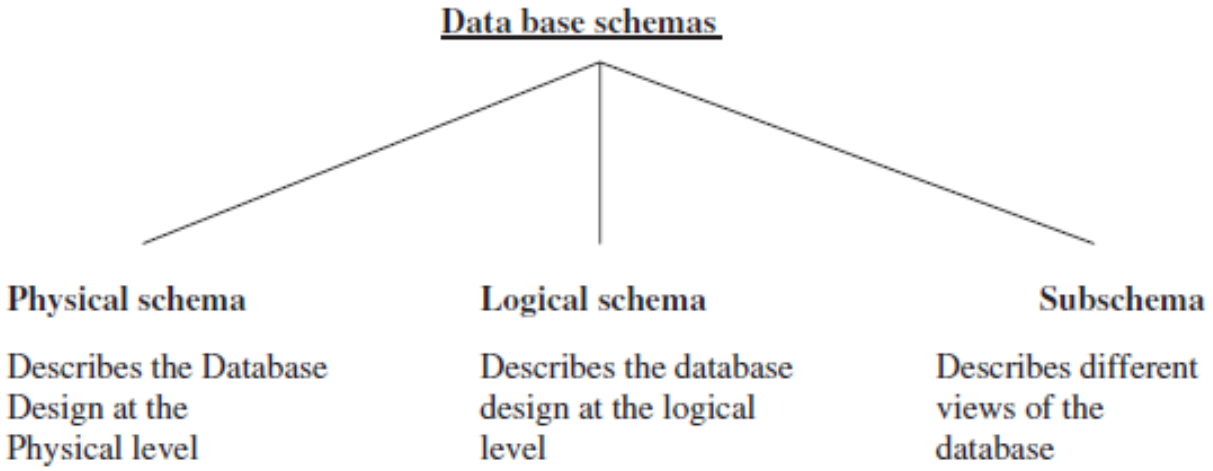


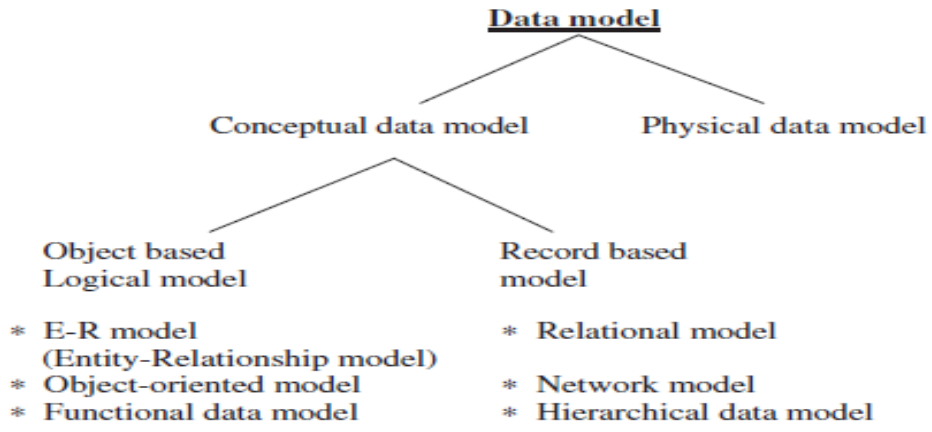
Fig. 1.6. ANSI/SPARK data model

- *Database Instances*
  - Database 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.
- *Database Schema*
  - The overall design of the database is called the database schema. A schema is a collection of named objects. Schemas provide a logical classification of objects in the database. A schema can contain tables, views, triggers, functions, packages, and other objects.



1.13 Data Models

- Data model is collection of conceptual tools for describing data, relationship between data, and consistency constraints.
- Data models help in describing the structure of data at the logical level.
- Data model describe the structure of the database.
- A data model is the set of conceptual constructs available for defining a schema.



1.13.1 Early Data Models

- Three historically important data models are the hierarchical, network, and relational models.
- Together they are often referred to as the "basic" data models.
- The hierarchical and network models, developed in the 1960s and 1970s, were based on organizing the primitive data structures in which the data were stored in the computer by adding connections or links between the structures.
    - As such they were useful in presenting the user with a well-defined structure, but they were still highly coupled to the underlying physical representation of the data.
    - Although they did much to assist in the efficient access of data, the principle of data independence was poorly supported.

1.14 Components and Interfaces of Database Management System

- A database management system involves five major components: data, hardware, software, procedure, and users. These components and the interface between the components are shown in Fig. 1.7.

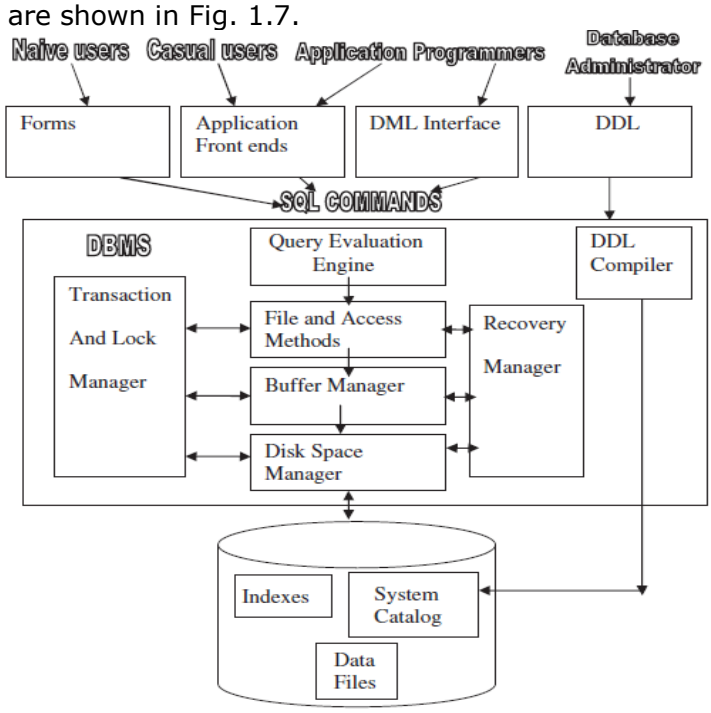


Fig. 1.7. Database management system components and interfaces

1.14.1 Hardware

- When we say Hardware, we mean computer, hard disks, I/O channels for data, and any other physical component involved before any data is successfully stored into the memory.
- When we run Oracle or MySQL on our personal computer, then our computer's Hard Disk, our Keyboard using which we type in all the commands, our computer's RAM, ROM all become a part of the DBMS hardware.

1.14.2 Software

- The software includes the DBMS software, application programs together with the operating systems including the network software if the DBMS is being used over a network.
- The DBMS software is capable of understanding the Database Access Language and interpret it into actual database commands to execute them on the DB.

1.14.3 Data

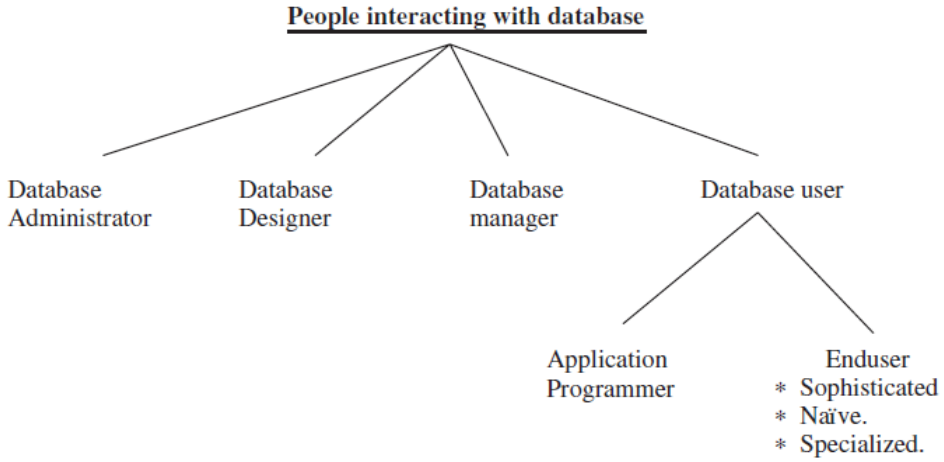
- The data in the database is persistent, integrated, structured, and shared.
- *Integrated Data*
  - A data can be considered to be a unification of several distinct data files and when any redundancy among those files is eliminated, the data are said to be integrated data.
- *Shared Data*
  - A database contains data that can be shared by different users for different application simultaneously.
- *Persistent Data*
  - Persistent data are one, which cannot be removed from the database as a side effect of some other process.

1.14.4 Procedure

- Procedures refer to general instructions to use a database management system.
- This includes procedures to setup and install a DBMS, To login and logout of DBMS software, to manage databases, to take backups, generating reports etc.

1.14.5 People Interacting with Database

- The people who manages the database, database administrator, people who design the application program, database designer and the people who interacts with the database, database users.



- **Database Administrator**
- Database Administrator or DBA is the one who manages the complete database management system.
- DBA takes care of the security of the DBMS, it's availability, managing the license keys, managing user accounts and access etc.

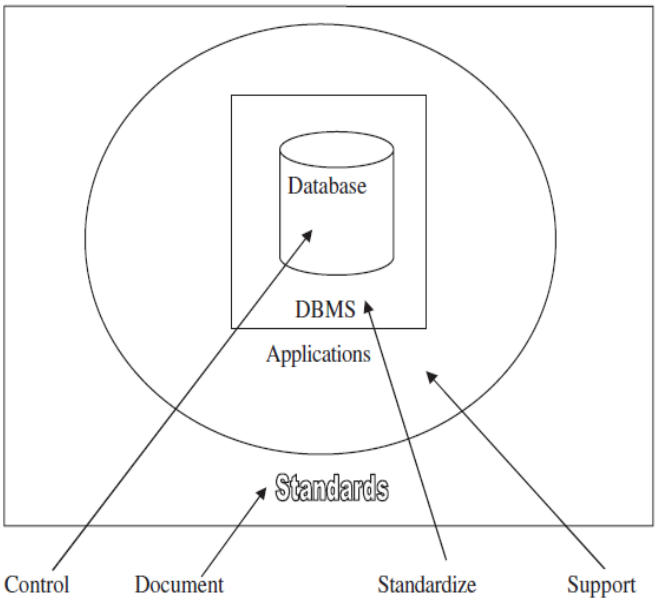
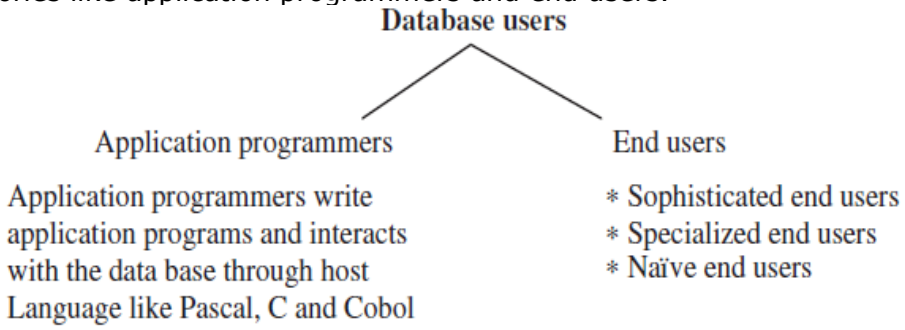


Fig. 1.8. Objectives of database administration

- **Responsibilities of Database Administrator (DBA)**
- The responsibilities of the database administrator are summarized as follows:
  - 1. Authorizing access to the database.
  - 2. Coordinating and monitoring its use.
  - 3. Acquiring hardware and software resources as needed.

- 4. Backup and recovery. DBA has to ensure regular backup of database, incase of damage, suitable recovery procedure are used to bring the database up with little downtime as possible.

- **Database Designer**
- Database manager is a program module which provides the interface between the low level data stored in the database and the application programs and queries submitted to the system:
- – The database manager would translate DML statement into low level file system commands for storing, retrieving, and updating data in the database.
- – *Integrity enforcement. Database manager enforces integrity by checking* consistency constraints like the bank balance of customer must be maintained to a minimum of Rs. 300, etc.
- – *Security enforcement. Unauthorized users are prohibited to view the information* stored in the data base.
- – *Backup and recovery. Backup and recovery of database is necessary to ensure* that the database must remain consistent despite the fact of failures.
- **Database Manager**
- Database manager is a program module which provides the interface between the low level data stored in the database and the application programs and queries submitted to the system:
- – The database manager would translate DML statement into low level file system commands for storing, retrieving, and updating data in the database.
- – *Integrity enforcement. Database manager enforces integrity by checking* consistency constraints like the bank balance of customer must be maintained to a minimum of Rs. 300, etc.
- – *Security enforcement. Unauthorized users are prohibited to view the information* stored in the data base.
- – *Backup and recovery. Backup and recovery of database is necessary to ensure* that the database must remain consistent despite the fact of failures.
- **Database Users**
- Database users are the people who need information from the database to carry out their business responsibility. The database users can be broadly classified into two categories like application programmers and end users.



- **Application Programmer or Software Developer:**
- This user group is involved in developing and designing the parts of DBMS.
- **End User:**
- End users are the one who store, retrieve, update and delete data. End users can be
- *Specialized End Users*
    - Specialized end users write specialized database application that does not fit into data-processing frame work. Application involves knowledge base and expert system, environment modeling system, etc.
- *Naive End Users*

- Naive end user interact with the system by using permanent application program Example: Query made by the student, namely number of books borrowed in library database.

- *System Analysts*
  - System analysts determine the requirements of end user, and develop specification for canned transaction that meets this requirement.
- *Canned Transaction*
  - Ready made programs through which naive end users interact with the database is called canned transaction.

1.14.6 Data Dictionary

- A data dictionary, also known as a "system catalog," is a centralized store of information about the database.
- It contains information about the tables, the fields of the tables, data types, primary keys, indexes, the joins which have been established between those tables, referential integrity, cascades update, cascade delete, etc.
- This information stored in the data dictionary is called the "Metadata."
  - Thus a data dictionary can be considered as a file that stores Metadata.
- The data dictionary can be integrated within the DBMS or separate.
- One of the major functions of a true data dictionary is to enforce the constraints placed upon the database by the designer, such as referential integrity and cascade delete.

1.14.7 Functional Components of Database System Structure

- The functional components of database system structure are:
  - 1. Storage manager.
  - 2. Query processor.

Storage Manager

- Storage manager is responsible for storing, retrieving, and updating data in the database. Storage manager components are:
  - 1. Authorization and integrity manager.
  - 2. Transaction manager.
  - 3. File manager.
  - 4. Buffer manager.

Transaction Management

A transaction is a collection of operations that performs a single logical function in a database application. Transaction-management component ensures that the database remains in a consistent state despite system failures and transaction failure. Concurrency control manager controls the interaction among the concurrent transactions, to ensure the consistency of the database.

*Authorization and Integrity Manager:*Checks the integrity constraints and authority of users to access data.

Transaction Manager: It ensures that the database remains in a consistent state despite system failures. The transaction manager manages the execution of database manipulation requests. The transaction manager function is to ensure that concurrent access to data does not result in conflict.

File Manager

- File manager manages the allocation of space on disk storage. Files are used to store collections of similar data. A file management system manages independent files, helping to enter and retrieve information records.
- File manager establishes and maintains the list of structure and indexes defined in the internal schema. The file manager can:
  - – Create a file
  - – Delete a file
  - – Update the record in the file

– Retrieve a record from a file

Buffer

- The area into which a block from the file is read is termed a buffer.
- The management of buffers has the objective of maximizing the performance or the utilization of the secondary storage systems, while at the same time keeping the demand on CPU resources tolerably low.
- The use of two or more buffers for a file allows the transfer of data to be overlapped with the processing of data.

Buffer Manager

- Buffer manager is responsible for fetching data from disk storage into main memory.
- Programs call on the buffer manager when they need a block from disk.
- The requesting program is given the address of the block in main memory, if it is already present in the buffer.
- If the block is not in the buffer, the buffer manager allocates space in the buffer for the block, replacing some other block, if required, to make space for new block.
- Once space is allocated in the buffer, the buffer manager reads in the block from the disk to the buffer, and passes the address of the block in main memory to the requester.

Indices

- Indices provide fast access to data items that hold particular values.
- An index is a list of numerical values which gives the order of the records when they are sorted on a particular field or column of the table.

1.15 Database Architecture

- Database architecture essentially describes the location of all the pieces of information that make up the database application.
- The database architecture can be broadly classified into two-, three-, and multitier architecture.

1.15.1 Two-Tier Architecture

- The two-tier architecture is a client–server architecture in which the client contains the presentation code and the SQL statements for data access.
- The database server processes the SQL statements and sends query results back to the client.
- The two-tier architecture is shown in Fig. 1.9. Two-tier client/server provides a basic separation of tasks.
- The client, or first tier, is primarily responsible for the *presentation of data to the user and the "server," or second* tier, is primarily responsible for supplying *data services to the client.*

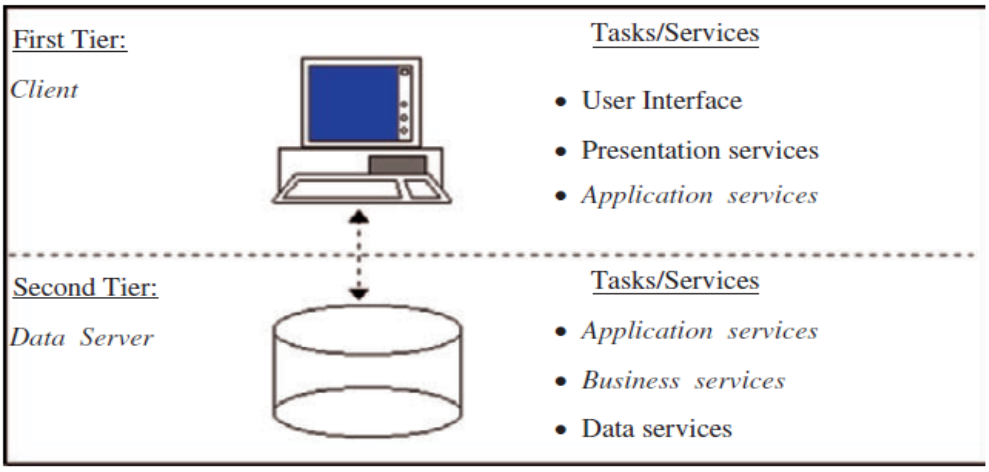


Fig. 1.9. Two-tier client–server architecture

Presentation Services

- "Presentation services" refers to the portion of the application which presents data to the user.

Business Services/objects

- "Business services" are a category of application services.
- These rules are derived from the steps necessary to carry out day-to-day business in an organization.
- These rules can be validation rules, used to be sure that the incoming information is of a valid type and format, or they can be process rules, which ensure that the proper business process is followed in order to complete an operation.

Application Services

- "Application services" provide other functions necessary for the application.

Data Services

- "Data services" provide access to data independent of their location.
- The data can come from legacy mainframe, SQL RDBMS, or proprietary data access systems.
- Once again, the data services provide a standard interface for accessing data.

Advantages of Two-tier Architecture

- The two-tier architecture is a good approach for systems with stable requirements and a moderate number of clients.
- The two-tier architecture is the simplest to implement, due to the number of good commercial development environments.

Drawbacks of Two-tier Architecture

- Software maintenance can be difficult because PC clients contain a mixture of presentation, validation, and business logic code.
- To make a significant change in the business logic, code must be modified on many PC clients.
- Moreover the performance of two-tier architecture can be poor when a large number of clients submit requests because the database server may be overwhelmed with managing messages.
- With a large number of simultaneous clients, three-tier architecture may be necessary.

1.15.2 Three-tier Architecture

- Three-tier architecture offers a technology neutral method of building client/server applications with vendors who employ standard interfaces which provide services for each logical "tier."

- The three-tier architecture is shown in Fig. 1.10. From this figure, it is clear that in order to improve the performance a second-tier is included between the client and the server.

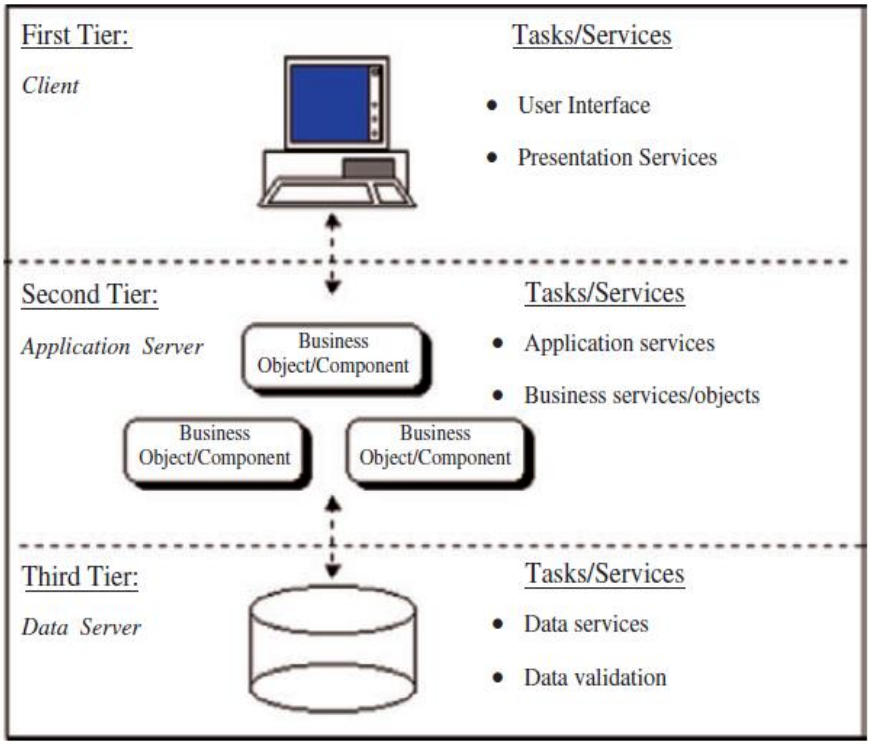


Fig. 1.10. Three-tier client–server architecture

- Through standard tiered interfaces, services are made available to the application.
- A single application can employ many different services which may reside on dissimilar platforms or are developed and maintained with different tools.
- This approach allows a developer to leverage investments in existing systems while creating new application which can utilize existing resources.
- Although the three-tier architecture addresses performance degradations of the two-tier architecture, it does not address division-of-processing concerns.
- The PC clients and the database server still contain the same division of code although the tasks of the database server are reduced. Multiple-tier architectures provide more flexibility on division of processing.

1.15.3 Multitier Architecture

- A multi-tier, three-tier, or N-tier implementation employs a three-tier logical architecture superimposed on a distributed physical model.
- Application Servers can access other application servers in order to supply services to the client application as well as to other Application Servers.
- The multiple-tier architecture is the most general client–server architecture.
- It can be most difficult to implement because of its generality.
- However, a good design and implementation of multiple-tier architecture can provide the most benefits in terms of scalability, interoperability, and flexibility.

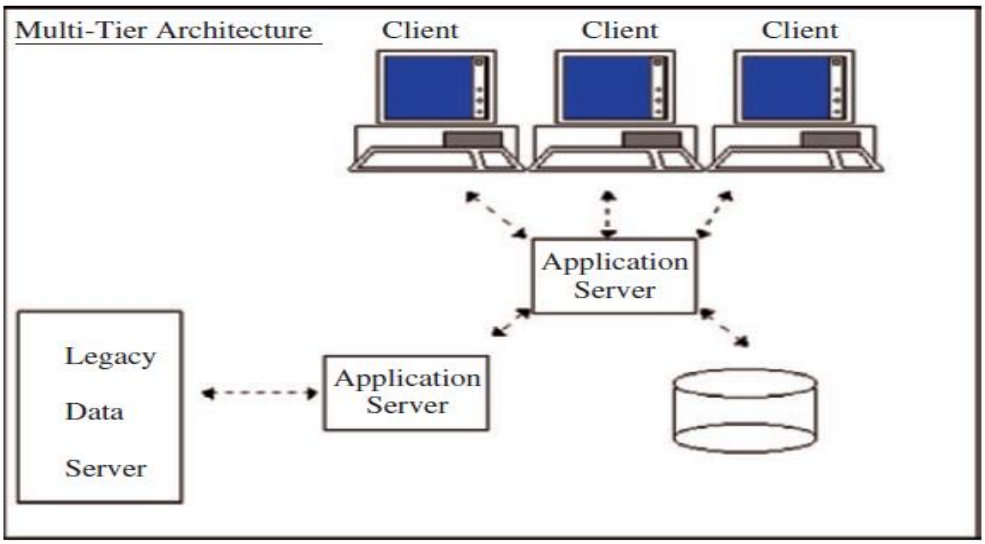


Fig. 1.11. Multiple-tier architecture

- For example, in the diagram shown in Fig. 1.11,
- the client application looks to *Application Server #1 to supply data from a mainframe-based application.*
- *Application Server #1 has no direct access to the mainframe application,* but it does know, through the development of application services, that *Application Server #2 provides a service to access the data from the mainframe* application which satisfies the client request.
- Application Server #1then invokes the appropriate service on Application Server #2 and receives the requested data which is then passed on to the client.
- Application Servers can take many forms. An Application Server may be anything from custom application services, Transaction Processing Monitors, Database Middleware, Message Queue to a CORBA/COM based solution.

1.16 Situations where DBMS is not Necessary

- DBMS is undesirable under following situations:
  - If traditional file processing system is working well, and if it takes more money and time to design a database, it is better not to go for the DBMS.
  - Moreover if only one person maintains the data and that person is not skilled in designing a database as well as not comfortable in using the DBMS then it is not advisable to go for DBMS.
  - DBMS is undesirable if the application is simple, well-defined, and not expected to change.
  - Runtime overheads are not feasible because of real-time requirements.
  - Multiple accesses to data are not required.
- Compared with file systems, databases have some disadvantages:
  - 1. High cost of DBMS which includes:
    - – Higher hardware costs
    - – Higher programming costs
    - – High conversion costs
  - 2. Slower processing of some applications
  - 3. Increased vulnerability
  - 4. More difficult recovery

1.17 DBMS Vendors and their Products

Some of the popular DBMS vendors and their corresponding products are given Table 1.1.

Table 1.1. DBMS vendors and their products

vendor	product
IBM	–DB2/MVS
	–DB2/UDB
	–DB2/400
	–Informix Dynamic Server (IDS)
Microsoft	–Access
	–SQL Server
	–DesktopEdition(MSDE)
Open Source	–MySQL
	–PostgreSQL
Oracle	–Oracle DBMS
	–RDB
Sybase	–Adaptive Server Enterprise (ASE)
	–Adaptive Server Anywhere (ASA)
	–Watcom

Questions:

1. What are the drawbacks of file – based Processing system?
2. Define data, Information and Data base? Explain the difference between the data and information?
3. Explain the advantages and disadvantages of DBMS?
4. Explain the Architecture of Data base?
5. Explain the various situations where DBMS is not necessary?
6. Explain about components and interfaces of DBMS.