DATABASE MANAGEMENT SYSTEMS

UNIT I: Introduction

Syllabus:

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, Instance and data independence; Three tier schema architecture for data independence; Database system structure, environment, Centralized and Client Server architecture for the database.

Objectives:

After studying this unit, you will be able to:

- Define database management system
- Explain database system applications
- State the characteristics and the database approach
- Understand different data models
- Discuss the advantages and disadvantages of database Discuss the database architecture

Introduction

The information storage and retrieval has become very important in our day-to-day life. The old era of manual system is no longer used in most of the places. For example, to book your airline tickets or to deposit your money in the bank the database systems may be used. The database system makes most of the operations automated. A very good example for this is the billing system used for the items purchased in a super market. Obviously this is done with the help of a database application package. Inventory systems used in a drug store or in a manufacturing industry are some more examples of database. We can add similar kind of examples to this list.

Apart from these traditional database systems, more sophisticated database systems are used in the Internet where a large amount of information is stored and retrieved with efficient search engines. For instance, http://www.google.com is a famous web site that enables users to search for their favorite information on the net. In a database we can store starting from text data to very complex data like audio, video, etc.

1.1 Database Management Systems (DBMS)

A database is a collection of related data stored in a standard format, designed to be shared by multiple users. A database is defined as "A collection of interrelated data items that can be processed by one or more application programs".

A database can also be defined as "A collection of persistent data that is used by the application systems of some given enterprise". An enterprise can be a single individual (with a small personal database), or a complete corporation or similar large body (with a large shared database), or anything in between.

Example: A Bank, a Hospital, a University, a Manufacturing company

Data

Data is the raw material from which useful information is derived. The word data is the plural of Datum. Data is commonly used in both singular and plural forms. It is defined as raw facts or observations. It takes variety of forms, including numeric data, text and voice and images. Data is a collection of facts, which is unorganized but can be made organized into useful information. The term Data and Information come across in our daily life and are often interchanged.

Example: Weights, prices, costs, number of items sold etc.

Information

Data that have been processed in such a way as to increase the knowledge of the person who uses the data. The term data and information are closely related. Data are raw material resources that are processed into finished information products. The information as data that has been processed in such way that it can increase the knowledge of the person who uses it.

In practice, the database today may contain either data or information.

Data Processing

The process of converting the facts into meaningful information is known as data processing. Data processing is also known as information processing.

Metadata

Data that describe the properties or characteristics of other data.

Data is only become useful when placed in some context. The primary mechanism for providing context for data is Metadata. Metadata are data that describe the properties, or characteristics of other data. Some of these properties include data definition, data structures and rules or constraints. The Metadata describes the properties of data but do not include that data.

1.2 Database System Applications

Databases are widely used. Here are some representative applications:

1. Banking: For customer information, accounts, and loans, and banking transactions.
2. 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.
3. Universities: For student information, course registrations, and grades.
4. Credit card transactions: For purchases on credit cards and generation of monthly statements.
5. Telecommunication: For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
6. Finance: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
7. Sales: For customer, product, and purchase information.
8. Manufacturing: For management of supply chain and for tracking production of items in factories, inventories of items in warehouses / stores, and orders for items.
9. Human resources: For information about employees, salaries, payroll taxes and benefits, and for generation of paychecks.

1.3 File Systems Versus A DBMS (Characteristics)

In earlier days, the databases were created directly on top of file systems. File system has many disadvantages.

1. Not enough primary memory to process large data sets. If data is maintained in other storage devices like disks, tapes and bringing relevant data to main memory, it increases the cost of performance. Problem in accessing the large data due to addressing the data using 32 bit or 64 bit mode addressing mechanism.
2. Programs must be written to process the user request to process the data stored in files which are complex in nature because of large volume of data to be searched.
3. Inconsistent data and complexity in providing concurrent accesses.
4. Not sufficiently flexible to enforce security policies in which different users have permission to access different subsets of the data.

A DBMS is a piece of software that is designed to make the preceding tasks easier. By storing data in a DBMS, rather than as a collection of operating system Files, we can use the DBMS's features to manage the data in a robust and efficient manner.

1.4 Advantages of DBMS

One of the main advantages of using a database management system is that the organization can exert via the DBA, centralized management and control over the data. The database administrator is the focus of the centralized control.

The following are the major advantages of using a Database Management System (DBMS):
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.

Efficient data access: A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is stored on external storage devices.

Data integrity and security: The DBMS can enforce integrity constraints on the data. The DBMS can enforce access controls that govern what data is visible to different classes of users.

Data administration: When several users share the data, centralizing the administration of data can offer significant improvements. It can be used for organizing the data representation to minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient.

Concurrent access and crash recovery: A 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. Further, the DBMS protects users from the effects of system failures. .

Reduced application development time: Clearly, the DBMS supports many important functions that are common to many applications accessing data stored in the DBMS.

1.5 Disadvantages of DBMS

The disadvantage of the DBMS system is **overhead cost**. The processing overhead introduced by the DBMS to implement security, integrity, and sharing of the data causes a degradation of the response and throughput times. An additional cost is that of migration from a traditionally separate application environment to an integrated one.

Even though centralization reduces duplication, the lack of duplication requires that the database be adequately backup so that in the case of failure the data can be recovered.

Backup and recovery operations are complex in a DBMS environment, and this is an increment in a concurrent multi-user database system. A database system requires a certain amount of controlled redundancies and duplication to enable access to related data items.

1.6 Data Models

A **data model** is a collection of high-level data description constructs that hide many low-level storage details. A DBMS allows a user to define the data to be stored in terms of a data model. Most database management systems today are based on the **relational data model.**

A schema is a description of a particular collection of data, using the given data model. The relational model of data is the most widely used model today.

Main concept: relation, basically a table with rows and columns.
Every relation has a schema, which describes the columns, or fields.

Data Model is a collection of high-level data description constructs that hide many low-level storage details. A DBMS allows a user to define the data to be stored in terms of a data model. Most database management systems today are based on the Relational data model. Relational models include – IBM's DB2, Informix, Oracle, Sybase, Microsoft's Access, Foxbase, Paradox, Tandem and Teradata.

1.7 Categories of data models

- **Conceptual (high-level, semantic) data models:** Provide concepts that are close to the way many users perceive data (Also called entity-based or object-based data models).
- **Physical (low-level, internal) data models:** Provide concepts that describe details of how data is stored in the computer.
- **Implementation (representational) data models:** Provide concepts that fall between the above two.

1. Hierarchical models:

Advantages:

- Hierarchical model is simple to construct and operate on.
- Corresponds to a number of natural hierarchical organized domains – e.g., assemblies in manufacturing, personal organization in companies.
- Language is simple; uses constructs like GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT etc.,

Disadvantages:

- Navigational and procedural nature of processing.
- Database is visualized as a linear arrangement of records.
- Little scope for "query optimization".
- One-to-many relationships.

2. Network model:

Advantages:

- Network model is able to model complex relationships and represents semantics of add/delete on the relationships.
- Can handle most situations for modeling using record types and relationship types.
- Language is navigational; uses constructs like FIND, FIND member, FIND owner, FIND NEXT within set, GET etc. Programmers can do optimal navigation through the database.

Disadvantages:

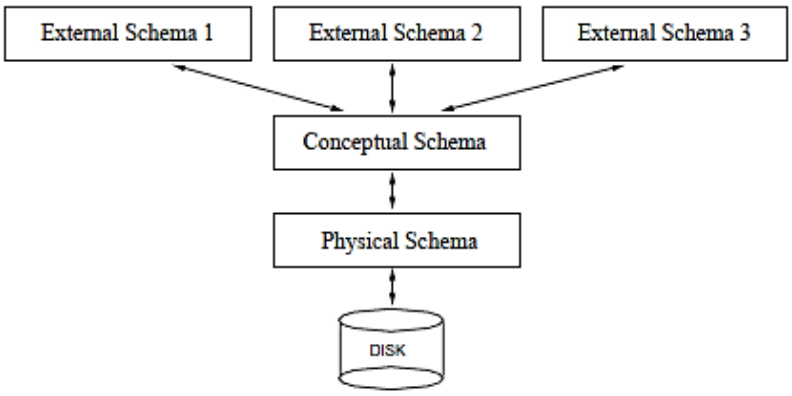
- Navigational and procedural nature of processing.
- Database contains a complex array of pointers that are expensive and difficult to update when inserting and deleting.
- Little scope for automated "query optimization".

3. Relational model:

- A relation, basically a table with rows and columns.
- Every relation has a schema, which describes the columns, or fields.
- Student information in a university database may be stored in a relation with the following schema
- Students (sid: string, name: string, login: string, age: integer, gpa: real)

1.8 Levels of Abstraction in a DBMS (Three tier schema architecture)

The data in a DBMS is described at three levels of abstraction.
The database description consists of a schema at each of these three levels of abstraction.
External, Conceptual and **Physical**
Views describe how users see the data.
Conceptual schema defines logical structure.
Physical schema describes the files and indexes used.



Conceptual schema:

- The conceptual schema(also called as logical schema) describes the stored data in terms of the data model of the DBMS.
- In a relational DBMS, the conceptual schema describes all relations that are stored in the database.
- In our sample university database, these relations contain information about entities, such as students and faculty, and about relationships, such as students' enrollment in courses.

```
Students(sid: string, name: string, login: string, age: integer, gpa:
    real)

Faculty(fid: string, fname: string, salary : real)

Courses(cid: string, cname: string,  credits: integer)

Rooms(nw: integer, address: string, capacity: integer)

Enrolled (sid: string, cid: string, grade: string)

Teaches (fid: string, cid: string)
```

The choice of relations, and the choice of fields for each relation, is not always obvious, and the process of arriving at a good conceptual schema is called conceptual database design.

Physical Schema:

- The physical schema specifies storage details.
- It summarizes how the relations described in the conceptual schema are actually stored on secondary storage devices such as disks and tapes.
- Decides what file organizations to use to store the relations and create auxiliary data structures, called indexes, to speed up data retrieval operations.
- A sample physical schema for the university database is to store all relations as unsorted files of records.
    - Create indexes on the first column of the students, faculty and courses relations, the salary column of faculty, and the capacity of column of rooms.

External Schema:

- This schema allows data access to be customized at the level of individual users or groups of users.
- A database has exactly one conceptual schema and one physical schema, but it may have several external schemas.
- An external schema is a collection of one or more views and relations from the conceptual schema.
- A view is conceptually a relation, but the records in a view are not stored in the DBMS.

1.9 Data Independence

Application programs are insulated from changes in the way the data is structured and stored.

Data independence is achieved through use of the three levels of data abstraction.

Logical data independence: users can be shielded from changes in the logical structure of the data, or changes in the choice of relations to be stored. This is the independence to change the conceptual schema without having to change the external schemas and their application programs.

Physical data independence: the conceptual schema insulated users from changes in physical storage details. This is the independence to change the internal schema without having to change the conceptual schema.

1.10 Architecture of a DBMS

The functional components of a database system can be broadly divided into query processor components and storage manager components. The query processor includes:

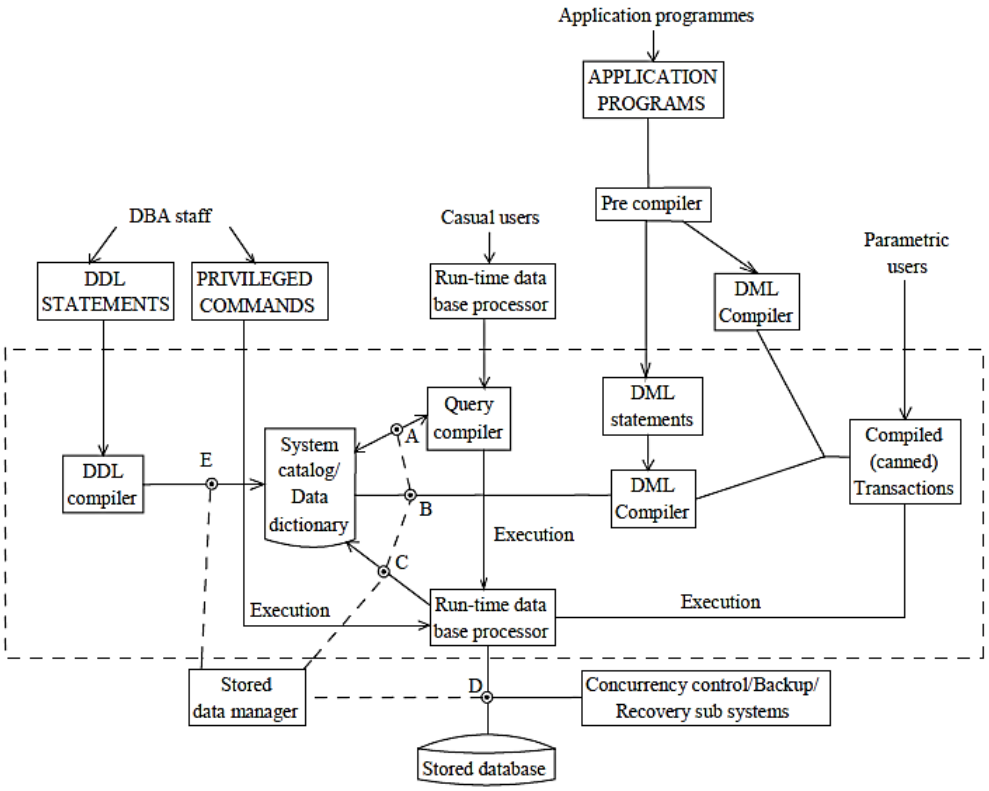
1. ***DML Compiler:*** It translates DML statements in a query language into low-level instructions that the query evaluation engine understands.
2. ***Embedded DML Pre-compiler:*** It converts DML statements embedded in an application program to normal procedure calls in the host language. The pre-compiler must interact with the DML compiler to generate the appropriate code.
3. ***DDL Interpreter:*** It interprets DDL Stateline its and records them in a set of tables containing

metadata.

4. ***Transaction Manager:*** Ensures that the database remains in a consistent (correct) state despite system failures, and that concurrent transaction executions proceed without conflicting.
5. ***File Manager:*** Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
6. ***Buffer Manager:*** Is responsible for fetching data from disk storage into main memory and deciding what data to cache in memory.

Also some data structures are required as part of the physical system implementation:

1. ***Data Files:*** The data files store the database by itself.
2. ***Data Dictionary:*** It stores metadata about the structure of the database, as it is used heavily.
3. ***Indices:*** It provides fast access to data items that hold particular values.
4. ***Statistical Data:*** It stores statistical information about the data in the database. This information used by the query processor to select efficient ways to execute a query.



1.11 People Who Deal With Databases

Quite a variety of people are associated with the creation and use of databases. Obviously, there are **database implementors**, who build DBMS software, and **end users** who wish to store and use data in a DBMS.

Database implementors work for vendors such as IBM or Oracle. End users come from a diverse and increasing number of fields.

In addition to end users and implementors, two other classes of people are associated with a DBMS: **application programmers** and **database administrators (DBAs).**

Database application programmers develop packages that facilitate data access for end users, who are usually not computer professionals, using the host or data languages and software tools that DBMS vendors provide.

The task of designing and maintaining the database is entrusted to a professional called the **database administrator**.

The DBA is responsible for many critical tasks:

- **Design of the conceptual and physical schemas:** The DBA is responsible for interacting with the users of the system to understand what data is to be stored in the DBMS and how it is likely to be used. Based on this knowledge, the DBA must design the conceptual schema (decide what relations to store) and the physical schema (decide how to store them).
- **Security and authorization**: The DBA is responsible for ensuring that unauthorized data access is not permitted. In general, not everyone should be able to access all the data. In a relational DBMS, users can be granted permission to access only certain views and relations.
- **Data availability and recovery from failures:** The DBA must take steps to ensure that if the system fails, users can continue to access as much of the uncorrupted data as possible.
- **Database tuning:** The needs of users are likely to evolve with time. The DBA is responsible for modifying the database, in particular the conceptual and physical schemas, to ensure adequate performance as user requirements change.

1.12 Database Environment

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.

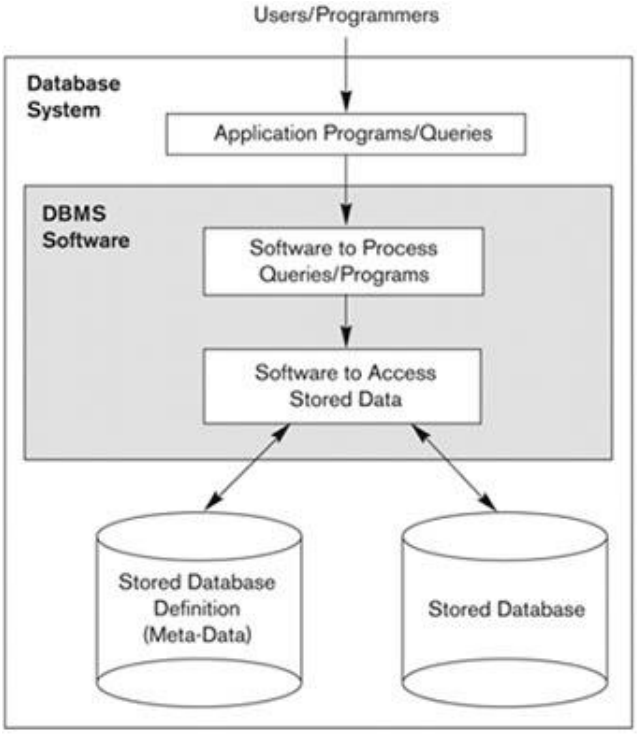
Constructing the database is the process of storing the data itself on some storage medium that is controlled by the DBMS.

Manipulating a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the mini world, and generating reports from the data.

Sharing a database allows multiple users and programs to access the database concurrently.

Other important functions provided by the DBMS include *protecting* the database and *maintaining* it over a long period of time.

Protection includes both *system protection* against hardware or software malfunction (or crashes), and *security protection* against unauthorized or malicious access. A typical large database may have a life cycle of many years, so the DBMS must be able to maintain the database system by allowing the system to evolve as requirements change over time. We can call the database and DBMS software together a database system.



1.13 Database Architecture

Database architecture uses programming languages to design a particular type of software for businesses or organizations. Database architecture focuses on the design, development, implementation and maintenance of computer programs that store and organize information for businesses, agencies and institutions.

The architecture of a DBMS can be seen as either single tier or multi-tier. The tiers are classified as follows:

1-tier architecture

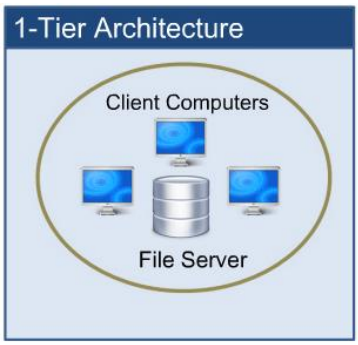
2-tier architecture

3-tier architecture

n-tier architecture

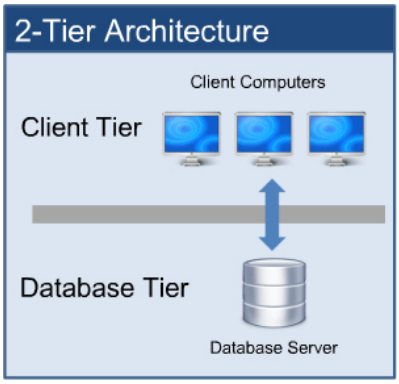
1-tier architecture:

One-tier architecture involves putting all of the required components for a software application or technology on a single server or platform.



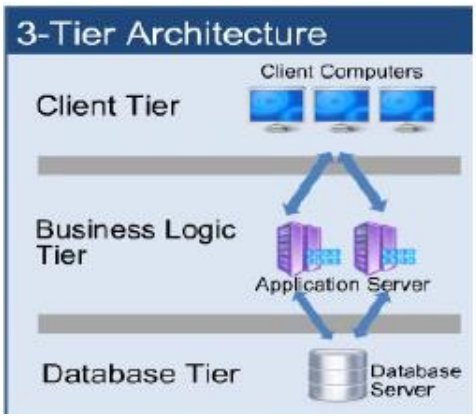
2-tier architecture:

The two-tier is based on Client Server architecture. The two-tier architecture is like client server application. The direct communication takes place between client and server. There is no intermediate between client and server.



3-tier architecture:

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.

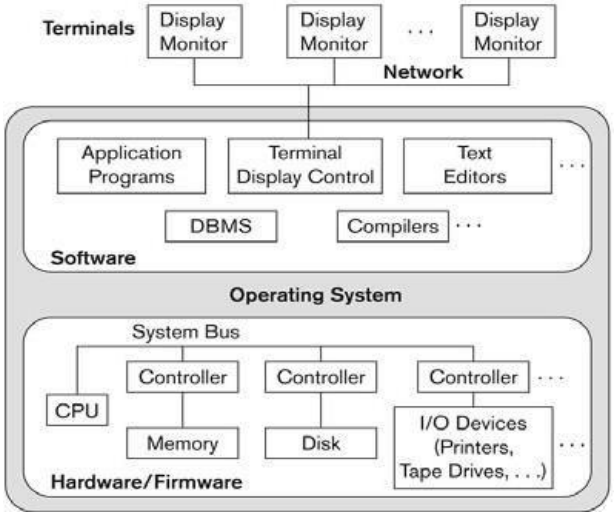


1.14 Centralized DBMS Architecture

Architectures for DBMSs have followed trends similar to those for general computer system architectures. Earlier architectures used mainframe computers to provide the main processing for all functions of the system, including user application programs and user interface programs, as well as all the DBMS functionality.

As prices of hardware declined, most users replaced their terminals with personal computers (PCs) and workstations. At first, database systems used these computers in the same way as they had used display terminals, so that the DBMS itself was still a centralized DBMS in which all the DBMS functionality, application program execution, and user interface processing were carried out on one machine.

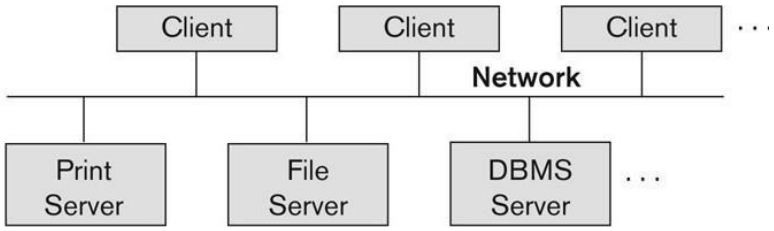
Gradually, DBMS systems started to exploit the available processing power at the user side, which led to client/server DBMS architectures.



1.15 Client/Server Architecture:

The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers, and other equipment are connected via a network. The idea is to define specialized servers with specific functionalities.

The resources provided by specialized servers can be accessed by many client machines. The client machines provide the user with the appropriate interfaces to utilize these servers, as well as with local processing power to run local applications. This concept can be carried over to software, with specialized software-such as a DBMS or a CAD (computer-aided design) package being stored on specific server machines and being made accessible to multiple clients.

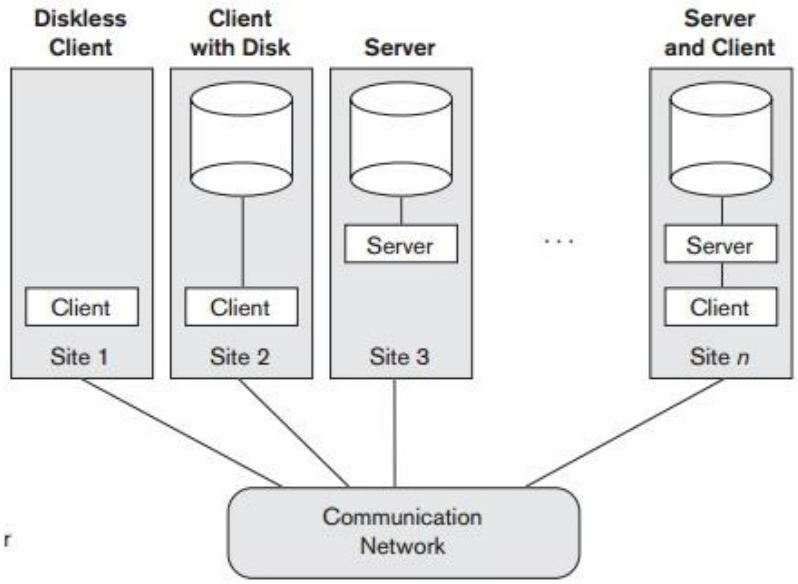


The concept of client/server architecture assumes an underlying framework that consists of many PCs and workstations as well as a smaller number of mainframe machines, connected via local area networks and other types of computer networks. A client in this framework is typically a user machine that provides user interface capabilities and local processing. When a client

requires access to additional functionality-such as database access-that does not exist at that machine, it connects to a server that provides the needed functionality.

A server is a machine that can provide services to the client machines, such as file access, printing, archiving, or database access. In the general case, some machines install only client software, others only server software, and still others may include both client and server software. However, it is more common that client and server software usually run on separate machines.

In client/server architecture, the user interface programs and application programs can run on the client side. When DBMS access is required, the program establishes a connection to the DBMS (which is on the server side); once the connection is created, the client program can communicate with the DBMS. A standard called Open Database Connectivity (ODBC) provides an application programming interface (API), which allows client-side programs to call the DBMS, as long as both client and server machines have the necessary software installed. Most DBMS vendors provide ODBC drivers for their systems.



Review Questions

1. Distinguish between database systems and file systems.
2. Discuss about the client server architecture of the database.
3. Define DBMS. Explain database users in detail.
4. What are advantages of DBMS? Explain.
5. With a neat diagram, explain the structure of Database Management System.
6. What is data independence and how does a DBMS support it? Explain.
7. What is a Database model? List out various database models and explain any two of them.
8. Explain the difference between external, logical and physical level schemas. How are these different schema layers related to the concepts of logical and physical data independence?
9. Compare and contrast various Data Models.
10. Demonstrate data abstraction implementation in DBMS.
11. List and explain various data models used for database design.
12. Explain about Buffer management in DBMS.

13. Who are the different database users? Explain their interfaces to database management system.
14. Describe the client server architecture for the database with necessary diagram.
15. Define Schema. Explain three level architecture in DBMS.
16. Explain Data Independence and its types in detail.
17. How does DBMS provide data abstraction? Explain the concept of data independence.
18. With a neat diagram describe the overall system structure of DBMS.

References:

- Raghurama Krishnan, Johannes Gehrke, *Database Management Systems*, 3rd Edition, Tata McGraw Hill.
- C.J. Date, *Introduction to Database Systems*, Pearson Education.
- Elmasri Navrate, *Fundamentals of Database Systems*, Pearson Education.