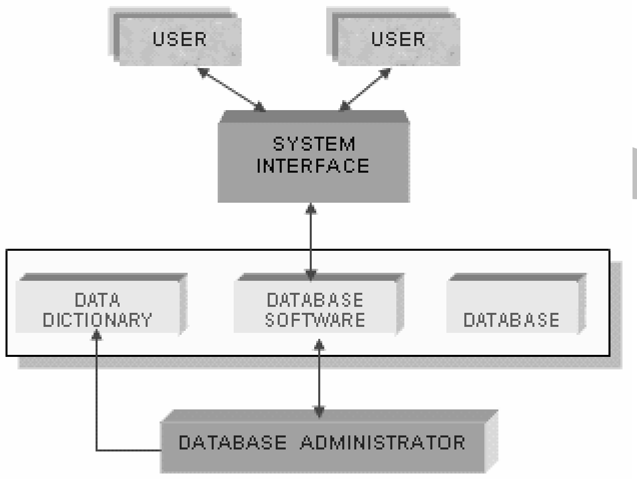
UNIT-III

DATABASE MANAGEMENT SYSTEM

Concepts of DBMS:

DBMS: It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications. Database management system consists of a collection of inter related data and a set of programs to access those data. The primary goal of DBMS is to provide an environment, i.e. both convenient and efficient to use in retrieving and storing database information.



Database: A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

Database system: The database and DBMS software together is called as Database system.

Data: Raw facts or the data having little meaning unless they have been organized in some logical manner.

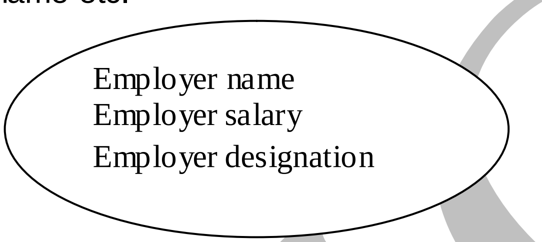
Field: a character or group of characters that has a specific meaning. A field is used to define and store data.

Record: A logically connected set of one or more fields that describe a person, place, or thing.

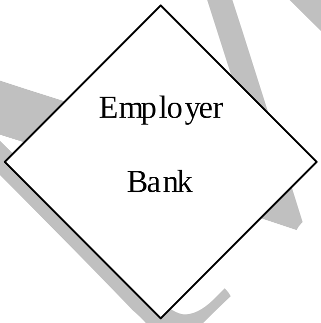
File: A collection of related records.

Entity: An entity is a thing or object in the real world. An entity set is a set of entities of same properties of attributes. An entity is represented in the shape of rectangle. The letters in the entity should be capital letters only. Ex-EMPLOYER

Attribute: They are descriptive properties possessed by each member of an entity set. An attribute is represented in the shape of ellipse. Ex- employer salary, employer designation, employer name etc.



Relationship: The association between two or more entities is called relationship. They are represented in the form of rhombus or diamond shape.



ENTITIES

- An entity is any object in the system that we want to model and store information about
- Individual objects are called entities
- Groups of the same type of objects are called entity types or entity sets
- Entities are represented by rectangles (either with round or square corners)

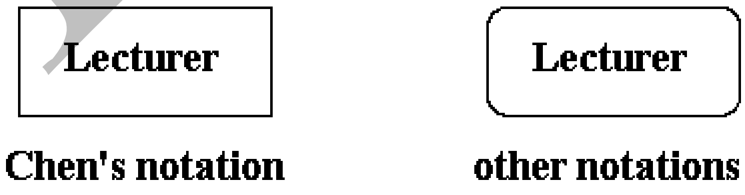
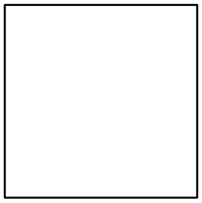


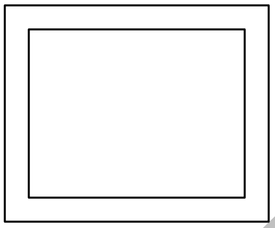
Figure: Entities

- There are two types of entities; weak and strong entity types.

Strong Entity: A strong relationship also known, as identifying relationship, which is not dependent upon others.



Weak Entity: It is otherwise called as non-identifying relationship, which is dependent upon others.



ATTRIBUTE

- All the data relating to an entity is held in its attributes.
- An attribute is a property of an entity.
- Each attribute can have any value from its domain.
- Each entity within an entity type:
  - May have any number of attributes.
  - Can have different attribute values than that in any other entity.
  - Have the same number of attributes.
- Attributes can be simple or composite
- single-valued or multi-valued
- Attributes can be shown on ER models
- They appear inside ovals and are attached to their entity.
- Note that entity types can have a large number of attributes... If all are shown then the diagrams would be confusing. Only show an attribute if it adds information to the ER diagram, or clarifies a point.

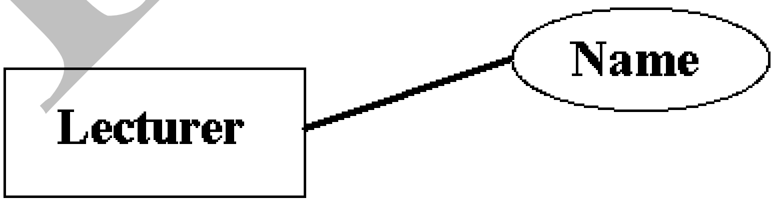


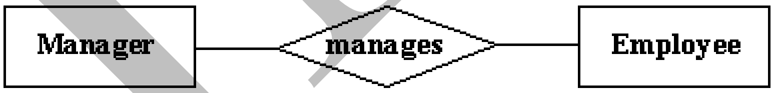
Figure: Attributes

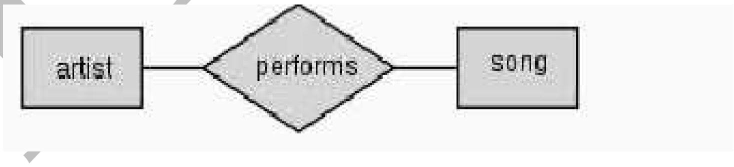
TYPES OF ATTRIBUTES

Simple attributes: The attributes are simple i.e. they are not divided. For example, age, sex, marital status would be classified as simple attributes.
Composite attributes: They can be divided into sub-parts. For example, address can be sub-divided into street, city, state, etc.
Single valued attribute: The attribute, which has a single value for a particular entity. For example, a company located at Hyderabad.
Multi valued attribute: Any attributes that have one or more number of dependents. For example, a person may have several college degrees and a household may have several phones with different numbers.
Null attribute: A null value is used when an entity doesn't have a value for null hypothesis. For example a student who fails in Ist year and II year and III year of B.com.
Derived attribute: It is one whose value is calculated from other attributes i.e. which is dependent on other, for example using student marks to find out total, average, and division.

RELATIONSHIPS

- A *relationship type* is a meaningful association between entity types
- A *relationship* is an association of entities where the association includes one entity from each participating entity type.
- Relationship types are represented on the ER diagram by a series of lines.
- As always, there are many notations in use today...
- In the original Chen notation, the relationship is placed inside a diamond, e.g. managers manage employees:



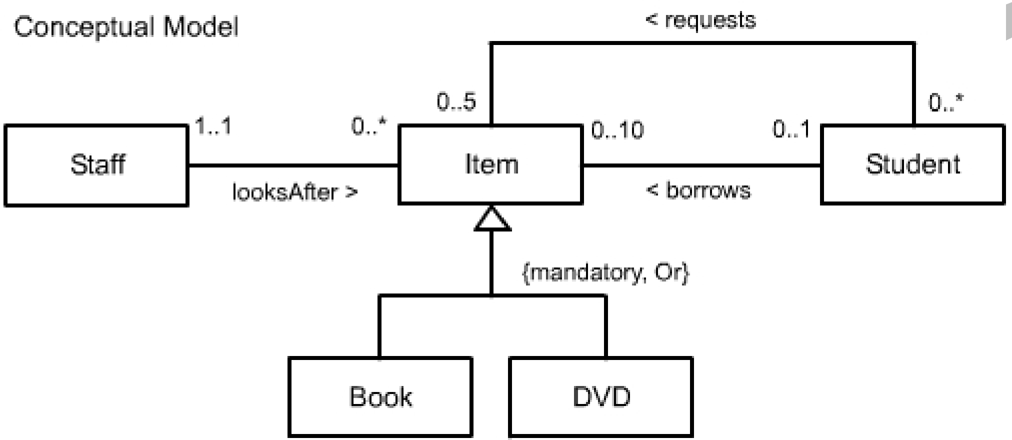


From the example we understand that artist is an **entity** whose **attribute** is to perform a song where perform is a **relation.**

DATABASE DESIGN

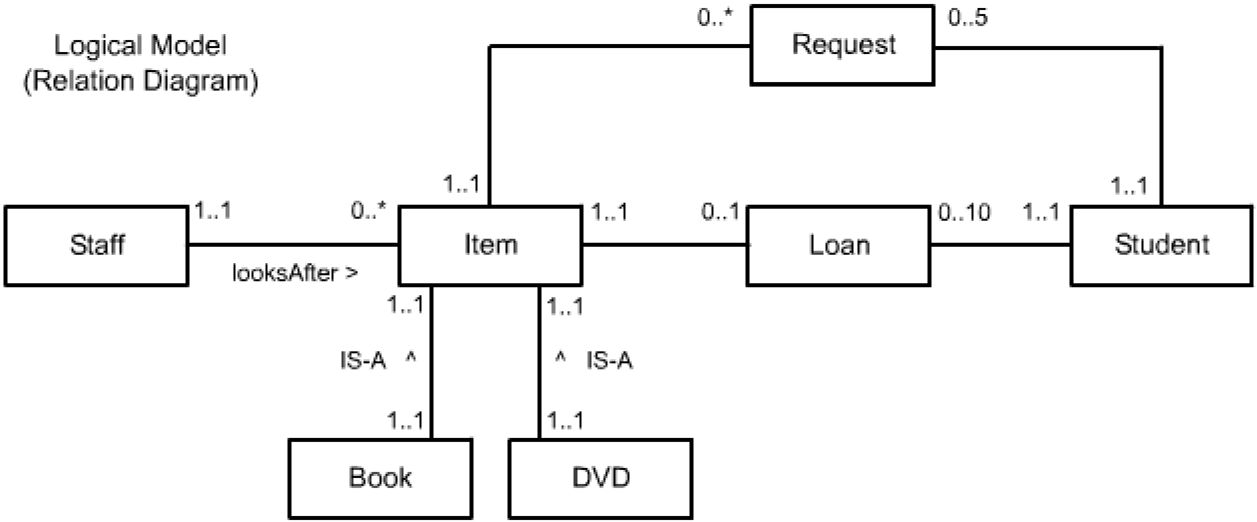
1. Conceptual Data Model

The conceptual model purely documents the data and information within the business and how it is used. The logical model is different from the conceptual model in that it takes into consideration the relational or object-oriented theory, which will be used to store the data. In some cases, the conceptual data model may be the same as the logical data model.
An example conceptual model is shown below. It is based on the same system as modeled in the logical data model above. Note that extra entities / relations have been added to enable the information and relationships to be stored in a relational database.



2. Logical Data Model

Logical database design has the aim of creating a data model that is completely independent from any particular DBMS or software/hardware platform. A conceptual model is typically needed before the logical model is constructed. If the system is a particularly large one, it is often the case that individual logical models are constructed for each user view or area within the business. These separate models are then merged into a global logical data model.
An example logical data model of a simple library system is shown below:



3. Physical Data model

The primary goal of physical database design is *data processing efficiency* .We will concentrate on choices often available to optimize performance of database services. Physical Database Design requires information gathered during earlier stages of the design process.

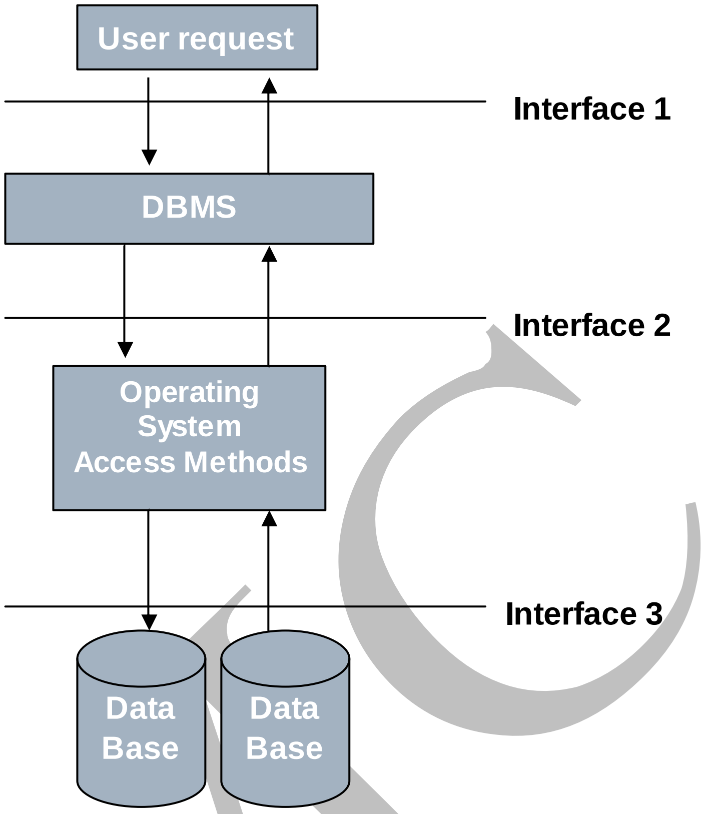
Information needed for physical file and database design includes:

- Normalized relations plus size estimates for them.
- Definitions of each attribute.
- Descriptions of where and when data are used, entered, retrieved, deleted, updated, and how often used.
- Expectations and requirements for response time, and data security, backup, recovery, retention and integrity.
- Descriptions of the technologies used to implement the database.

There are several critical decisions that will affect the integrity and performance of the system:

- Storage Format
- Physical record composition
- Data arrangement
- Indexes
- Query optimization and performance tuning

Physical model



Interface 1: User request to the DBMS. The user presents a query, the DBMS determines which physical DBs are needed to resolve the query.
Interface 2: The DBMS uses an internal model access method to access the data stored in a logical database.
Interface 3: The internal model access methods and OS access methods access the physical records of the database.

Tables: A table is a collection of different types of records. The data is entered into the table with help of different field names.

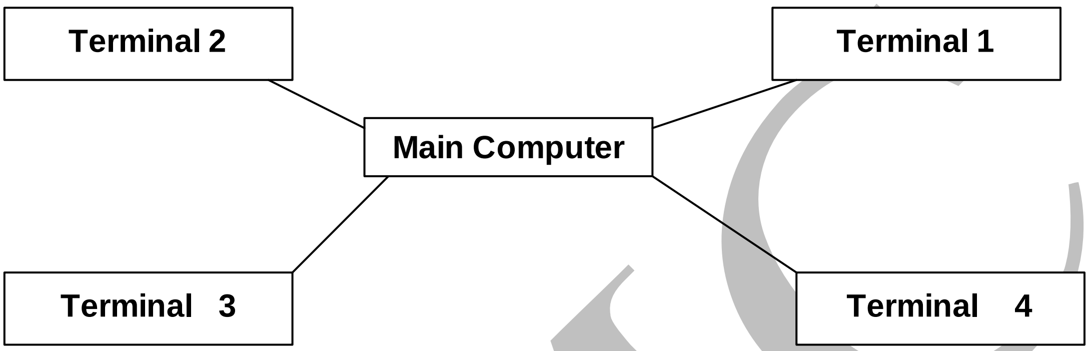
Queries: In database management system, a method retrieving and displaying specific data from the database. A query is a question, which gives answer. Query allows us to create a new table that contains only those fields and records in which we are interested.

Forms: They are applications used for entering the data through the input device into database. To make database more users friendly, forms are created and displayed on the screen. People then type data into these various forms and it is automatically entered into the database. Once the data has been entered the forms can also be used to view, edit, or delete it.

Reports: In the database management system a report can contain printed output with page numbers, headings, input and calculated information and reports can be created either as needed or at regularly scheduled times.

DISTRIBUTED DATABASE

Distributed database - A collection of multiple, logically interrelated databases distributed over a computer network. It simply means that databases of a single system are spread over some different locations. Sometimes it is not possible to have all the databases at a single location due to problem of space, geographical set up of building.



(Networking of different computers with a main computer)

Distributed Database Management System (DDBMS) - a software system that permits the management of a distributed database and makes the distribution transparent to the users. If heterogeneous, it may allow transparent simultaneous access to data on multiple dissimilar systems.

Advantages

1. Improves performance, e.g. it saves communication costs and reduces query delays by providing data at the sites where it is most frequently accessed.
2. Improves the reliability and availability of a system by providing alternate sites from where the information can be accessed.
3. Increases the capacity of a system by increasing the number of sites where the data can be located.
4. Allows users to exercise control over their own data while allowing others to share some of the data from other sites.
5. Helps solve more complex database problems.

Disadvantages

1. Increases the complexity of the system and introduces several technical as Well as management challenges especially when geographical and organizational boundaries are crossed.
2. Makes central control more difficult and raises several security issues because a data item stored at users at the remote site can always access remote site.
3. Makes performance evaluation difficult because a process running at one node may impact the entire network.

Why to distribute data?

1. **Load:** The load on a few systems may be more, separate systems are required to be used to reduce burden on the computers. Terminals at various locations are installed and are connected with the main system.

2. **Security:** It is advisable to keep the databases at different locations. Databases of important nature are kept at two or more locations. If the data at one location is damaged due to mishandling, natural calamities, it will be available on the computer at other locations. Password systems can also be used in LAN system for safety measures.

3. **Reduced cost of software:** The main software can be shared by the terminals at different locations. There is no need to purchase costly software for each of the terminals. The software can be shared and sharing is always cost effective.

4. **Reduced cost of hardware:** The cost of hardware is always high. Networking can play an important role in saving high costs. For example there is no need to buy printers for each terminal. The printer at different location can be shared by the user at other location.

5. **Terminal versatility:** The user at different location provides the options. Two different soft wares cannot be run at single computer at the same time. Suppose the user is working in SQL at one location and the data is being processed. Mean while he can go to other terminal and run FoxPro and process data therein.

<u>SQL client/server architecture</u>

The term Client/Server (CS) architecture involve multiple computers connected in a network is a concept of CS systems is that one or more of these computers may function as a provider of services to the remaining computers, which function as Clients that process applications. Client and Server have more formally been defined as follows:

Client: A computer or workstation attached to a network that is used to access network resources.

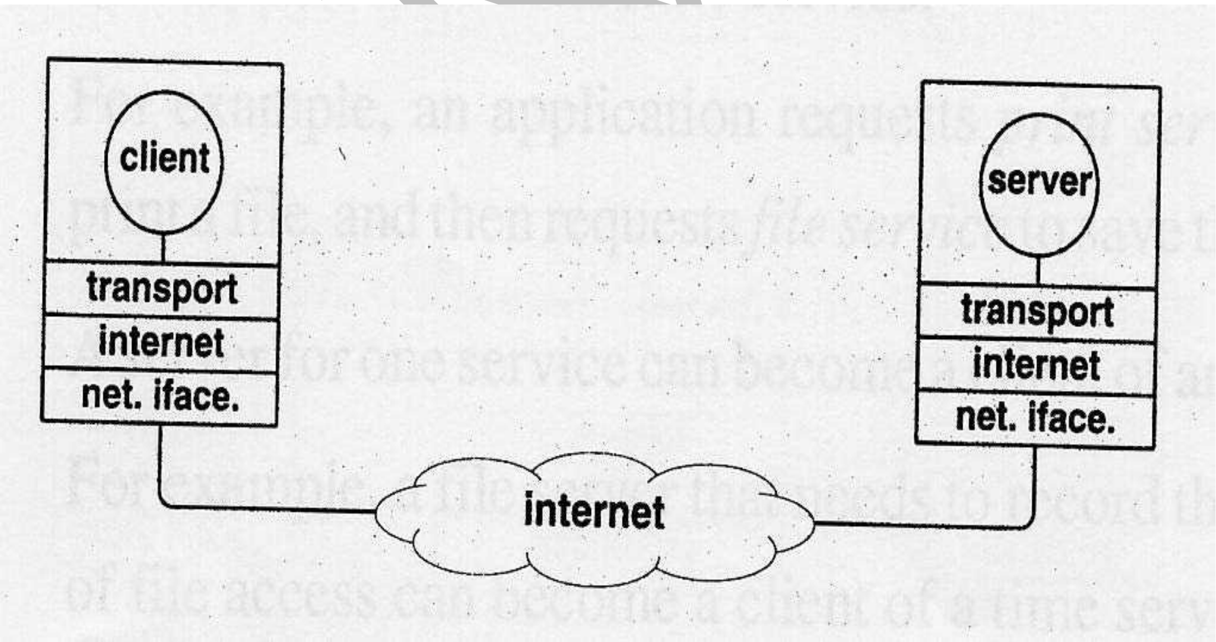
Server: A computer that furnishes clients with services such as database, connection to a network, or large disk drives. Server can be mainframes, minicomputers, large workstations or LAN devices. More than one server can be involved providing services to clients.

Server Program

- It is a program designed for providing a particular service.
- When the server computer boots up, the server program is usually invoked automatically, i.e., the service is always available.
- It waits passively for clients' requests. Upon receiving one request, it performs the necessary computation and returns the result to the Client.
- In many applications, a server is designed such that it can serve multiple clients simultaneously (e.g., web servers).
- It usually requires powerful hardware. It requires an operating system that supports multi tasking.

Client Program

- It is an application program executed by an end user on a local computer.
- When it needs a service, it sends a request to the server program, and then waits for the response from the server program. It can request multiple services when it is necessary.
- **Client Server Interaction**–The client and the server use a transport protocol to send and receive data. For example, they can use the TCP/IP (Transmission Control Protocol and Internet Protocol) suite to communicate through the Internet.



The **client and the server can interact in many possible** manners, e.g.

- An application (a client) interacts with one server only.
- An application becomes a client of one service, and later becomes a client of another service.
- For example, an application requests print service to print a file, and then requests file service to save the file.
- A server for one service can become a client of another.
- For example, a file server that needs to record the time of file access can become a client of a timeserver.

Multiple Services–A powerful computer can run multiple server programs to provide multiple services at the same time. Example: a computer runs an ftp (file transfer protocol) server and a web server.

To provide multiple services:

- The computer must have sufficient resources (e.g., fast processor(s) and large memory);
- The OS supports multitasking (e.g., UNIX, windows)