Essentials of Computer Networks, Internet and Database Technologies

V. S. Dixit • V. B. Singh

Alpha Science

Essentials of Computer Networks, Internet and Database Technologies

Essentials of Computer Networks, Internet and Database Technologies

V.S. Dixit
V.B. Singh

Alpha Science International Ltd.
Oxford, U.K.

Essentials of Computer Networks, Internet and Database Technologies

444 pgs. | 133 figs. | 02 tbls.

V.S. Dixit
Department of Computer Science
Atma Ram Sanatan Dharam College
Unversity of Delhi

V.B. Singh
Department of Computer Science
Delhi College of Arts and Commerce
Unversity of Delhi

ALPHA SCIENCE INTERNATIONAL LTD.
7200 The Quorum, Oxford Business Park North
Garsington Road, Oxford OX4 2JZ, U.K.

www.alphasci.com

ISBN 978-1-84265-751-5

E-ISBN 978-1-78332-001-1

Printed in India

Dedicated to

To the loving memory of my mother, Late (Smt.) Sushila Dixit,
whatever I am today that is just because of her.

V.S. Dixit

In memory of my father, Late Sh.Ramajeet Singh

V.B. Singh

Preface

In view of the widespread applications of computers in diverse areas, Computer Networks, Internet and Database Technologies have been introduced as papers in an assortment of streams of undergraduate education in all the universities of the country. The present book has been written keeping in view the syllabi of various universities. The book has been written in an easy to understand approach and students do not need any prerequisites knowledge about the subject. The book is organized into seven chapters covering different subjects namely Computer Networks, Internet, Database Technologies and Structured Query Language.

First Chapter deals with basic concept of Data Communication. In this chapter, we have discussed the topics like the concepts of Data Communication and their components, Transmission Mode and, Transmission Media (Guided and Unguided Media), Modulation Techniques, Multiplexing and Switching Techniques.

In the Second chapter detailed emphasis is given on Computer Networks and various topics namely Computer Network and their types, description of various networking devices and protocols, OSI and TCP/IP reference model, Network Topology and Types of Network have been discussed.

Third Chapter covers the Internet topic in an elaborative manner. In this chapter, we have discussed various aspects of internet technologies. Topics are presented in a simplified manner for the students.

Fourth Chapter covers the HTML topic in detail. The purpose of this chapter is to provide the knowledge of web designing using Fourth generation programming language HTML. In this chapter, there is a discussion about client-side scripting, server-side scripting, static documents, dynamic documents and active documents. There is an explanation of HTML concepts with the help of simple examples. At the end of chapter, we have demonstrated three major projects by incorporating important tags of the HTML.

In the Fifth Chapter, we have elaborated JAVASCRIPT in detail. JavaScript is used to create Dynamic web pages. Constructs of JavaScript are well described in this chapter. We have explained the basics of JavaScript, ASP with simple examples. Brief explanation about the DHTML has also been given.

In chapters Six and Seven, there is a detailed study on the concept of databases and SQL respectively. In chapter 6, evolution of database, Data models, Three Level Architecture of Database System, File Organizations and their access

methods, ER-Diagrams, Normalization and Relational Algebra has been discussed with suitable examples. Chapter 7 deals with the basics of SQL. In this chapter, we have discussed different DDL, DML, DCL and TCL statements with examples. We have also discussed different clauses used in SQL.

Several illustrative examples have been given in each chapter to clarify the important points. Each chapter also contains a set of questions for practice to enable the students to test the knowledge acquired in the chapter. We have also given four appendices which describe color schemes used in HTML, Open Database Connectivity, E-Commerce and the installation of MySQL.

Keeping in mind the audience of the book, we have given a large set of descriptive and objective sample questions which can help students in various competitive examinations.

It is hoped that the book will meet the needs of the students opting these papers and benefit them immensely.

Suggestions and comments pointed out by students and teachers are welcome and will be appreciatively accredited.

V.S. Dixit
V.B. Singh

Acknowledgements

Firstly, we wish to express our sincere thanks to students who gave their valuable ideas and suggestions from time to time during preparation of the book. We remain obliged to them for their valuable academic support.

Some people deserve special mention as Prof. S.L. Singh, Prof. P.K. Kapur, Prof. V.K. Sharma, Dr. B.L. Arora and Dr. M.S. Rawat as they have always been a source of inspiration.

We would also express our thanks to Prof. M.K. Gupta, Prof. Manoj Kumar Pandey, Dr. Sanjay Dwivedi, O.S. Srivastava, Dr. T. Dixit, Akhtar Hussain, Naveen Ahlawat, K.N. Misra, Prof. Rahul Mishra, Dr. O.P. Singh and Dr. A.K. Singh who were instrumental in instilling the curiosity for this book and shared their experience with us.

Sincere thanks to Mr. Amit Sharma and Mr. Basant Nainwal for their help during compilation of the book.

Special thanks to Mr. N.K. Mehra and his entire team of Narosa Publishing House for their whole hearted cooperation and support.

We are also indebted to our book reviewers and language editors, whose valuable comments helped in improving the quality of the book.

Last but not the least, we wish to thank our wives, children and families for their patience and encouragement during the many long weekends and evening that went into the writing of this book. And especially to God, who made everything possible.

V.S. Dixit
V.B. Singh

Contents

Data Communication

1
CHAPTER

Born	: 25, April 1874
Died	: 20, July 1937
Known For	: Radio, Wireless Communication
Notable Awards	: Nobel Prize for Physics (1909)


Guglielmo Marconi

1.1 WHAT IS DATA COMMUNICATION?

The word Communication has its inception from the beginning of human civilization. During early times, the strength of communication was limited only to face to face communication but later on communication took place between humans across any geographical location (Voice Communication). Over a period of time, communication technology got an edge and now a days in the digital world, the communication takes place between devices to transmit any kind of information like audio, video, images etc. Today any information in any form can be transmitted from one place to another irrespective of any geographical location. Computer technology advancements in the information technology has brought many benefits to the people all around the world.

Data Communication means transmission of the digital signals during a communication channel between the Source (Transmitter) and Host (Receiver) devices. These devices may be a computer or any other electronic equipment. The source transmits the data and the receiver receives it. The major issues with Data Communication System is (*i*) the transfer of data, the method of transfer and presentation of data during the transfer process (*ii*) to provide the highest transmission rate at the lowest possible power with least possible noise (*iii*) data communication system should also devise mechanism to reduce distortion and attenuation (attenuation means loss of energy in signals during transmission). Data Communication System also provides rules for Source and Host devices because different devices have different working environment. Different devices

use different Operating Systems, languages and media for communication. The rules that govern the computers/devices for communication is called **Protocol.** Data Communication Systems have different protocol standards.

1.1.1 Basic Components of a Data Communication System

Data Communication System consists of the following basic components:

1. Source/Sender/Transmitter
2. Host/Receiver
3. Medium
4. Message
5. Protocol.

The following diagram shows the logical diagram of Data Communication System:

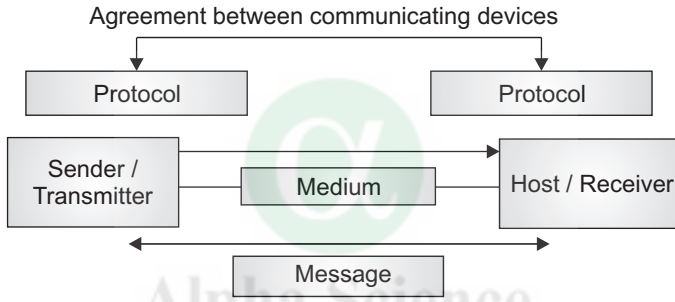


Figure 1.1

Now, we are going to describe the various components of Data Communication System.

(*i*) The device which sends the data/message is called **Source** or **Sender/ Transmitter**. Examples are:

- Terminal
- Computer
- Television broadcast
- Radio broadcast

(*ii*) The device which receives data/message is called **Receiver/Host**. Examples are:

- Radio
- Printer
- Terminal
- Computer
- Television

(*iii*) **Medium:** Through which data/message are transmitted between two devices. Examples are:

- Physical wire/Cabling
- Optical Fibers
- Radio Frequencies (RF)
- Infrared Wireless (IR)

A description of various mediums for Data Communication has been given in section 1.5.

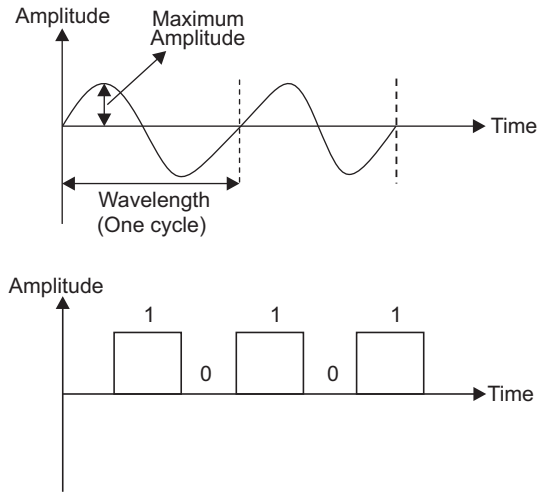
(*iv*) Actual content which needs to be transfered from one device to another device is called data / message.

(*v*) We need some set of rules called protocol for the transmission of data / message from one device to another.

1.2 TRANSMISSION MODE (FLOW OF DATA IN A COMMUNICATION CHANNEL)

Information in a network system is always transmitted through a pathway *i.e*. a Communication Channel. The pathway may be defined as a wired or wireless media (may be physical wire or a radio link). Information sent through a communication channel has a source from which the information originates, and a destination to which the information is delivered. Although information originates from a single source, there may be more than one destination, depending upon how many receiver stations are linked to the channel and how much energy is possessed by the transmitted signal.

Communication channel consists of medium (medium may be guided or unguided as discussed in section 1.6) through which the electrical signals can be transmitted. These signals may be analog or digital. As we know that a digital information consists of binary bits (0 and 1). A group of eight bits make a byte and collection of byte makes a frame of a message. In digital communication, the value of binary bits is represented by voltage pulses, where 1 represents 5 Volts and 0 represents 0 Volts. It means a Digital Communication Channel transmits binary bits represented by voltage pulse. The digital signal has only two states. An Analog communication channel transmits information in the form of sine waves. Unlike digital signal, an analog has many states as shown in the following figure,



There is always a direction associated with the Communication Channel. In a Communication Channel, the message source is the Transmitter, and the destination is the Receiver. Following are the different Transmission Modes:

1.2.1 Simplex Transmission Mode

A channel whose direction of transmission does not change it means that it works in only one direction which is called simplex transmission mode. Example of simplex transmission mode is a radio and TV station. They always transmit the signal to its listeners and never allow them to transmit back. In Simplex Communication, the data flows in only one direction on the data communication line (medium).

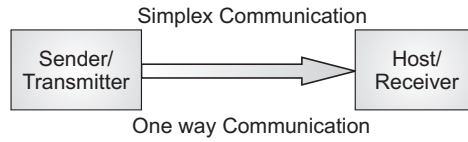


Figure 1.2

1.2.2 Half Duplex Transmission Mode

A **Half-Duplex** system provides communication in both the directions, but only in one direction at a time (not simultaneously). In Half Duplex communication, both transmitter and receiver works on the same frequency. So, both cannot communicate simultaneously.

An example of a Half-Duplex system is a "walkie-talkie" which is a two-way radio, wherein one must use "Over" or any other designated command to indicate the end of transmission from one party, and ensure that only one party transmits at a time.

A good analogy for a **Half-Duplex** system would be a one lane road with traffic police at each end. Traffic can flow in both the directions, but only one direction at a time.

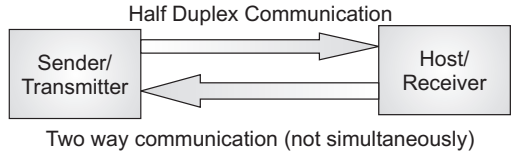


Figure 1.3

1.2.3 Full Duplex Transmission Mode

A **Full-Duplex**, system allows communication in both directions (simultaneously). Land-Line Telephone Networks are full-duplex since both parties can talk in one go. A good analogy for a Full-Duplex system would be a two-lane road with one lane for each direction.

Examples: Telephone, Mobile Phone etc.

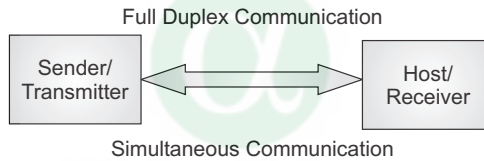


Figure 1.4

A Full Duplex system has flowing advantages over Half Duplex:

1. There are no collisions in Full Duplex Communication as it provides two separate lines.
2. Full bandwidth is available in both the directions.
3. Response time is good.

1.3 DATA COMMUNICATION MEASUREMENT

In this real world, we generally use different communication measurement on the basis of range of frequencies. Some of them are illustrated as follows:

1.3.1 ISDN (Integrated Services Digital Network)

Integrated Services Digital Network is a digital phone connection that sends voice, video, and data simultaneously over digital or ordinary phone lines, with a much faster speed and higher quality than an analog system. Due to ISDN, telephones display the caller's telephone number, name and address. The ISDN supports multiple channels interleaved by Time Division Multiplexing. ISDN has several channels like 4-kHz for analog telephone channel, 64-kbps digital

computer channel for voice or data, 8 or 16 kbps digital channel for Out-Of-Band signaling (Tanenbaum, 2003).

1.3.2 Narrowband

Narrowband replace the Analog Telephone System. Narrowband ISDN is suitable for both voice and non-voice traffic. It offers service at the speed of less than 1.544 Mbps.

1.3.3 Wideband

Wideband transmission channel is capable of transmitting more information than narrowband, but less than broadband. Its speed lies between 1.544 Mbps to 45 Mbps.

1.3.4 Broadband

Broadband ISDN is a Digital Virtual Circuit for moving fixed packet from source to destination. Its speed ranges from 1.54 Mbps to 1.244 Gbps. But, its speed mainly depends on the media we use for the transmission. There are different types of Broadband connection and some of them are Digital Cable Broadband (speed up to 1.5 Mbps), DSL (Digital Subscriber Line with speed up to 1.5Mbps), WI-FI (same speed as Digital Cable and DSL), T-Carrier Broadband with speed up to 45 Mbps and OC24 (Optical Carrier level 24) with speed up to 1.244 Gbps. Broadband ISDN is based on ATM (Asynchronous Transmission Mode) technology.

Note: In a Data Communication, Data is transmitted through some media from one device to another device in terms of binary bits. Media is one factor which determines how many bits will be transmitted in a single instance of time. There are two types of Communication namely Serial and Parallel. In Serial Communication only 1 bit of information can be transmitted over the data transmission medium at a time but in Parallel more than one bit of information is transmitted by using parallel lines. Parallel Communication is fast but is used for short distances whereas Serial Communication is slow and is used for long distances.

1.4 TRANSMISSION MEDIA

A **Transmission Medium** is defined as a material substance. A material substance may be solid, liquid or gas through which energy waves are propagated. For example, Sound waves are propagated through air; but solids and liquids may also be used as Transmission Media for sound.

Vacuum does not have air but through waves the energy is propagated and these waves are called electromagnetic waves. Examples of electromagnetic waves are radio and light.

The physical properties of medium is important and it determines the inside behavior of waves. Following is the classification of Transmission Medium:

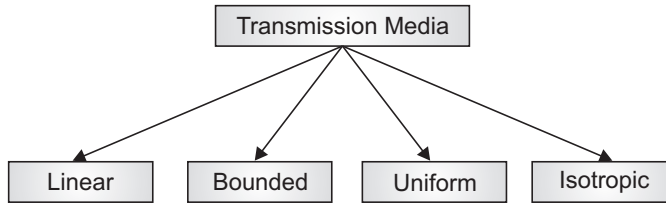


Figure 1.5

Linear Medium: If different waves at any particular point in the medium can be superposed, then it is categorized as a linear medium.

Bounded Medium if the finite is in extent, otherwise Unbounded Medium;

Uniform Medium is used only if its physical properties are unchanged at different points;

An Isotropic Medium is used only if physical property is same in different directions.

We can also classify the Transmission Media broadly in two categories as per their properties:

(*i*) Guided

(*ii*) Unguided

1.4.1 Guided Media (Wired Communication)

Guided Transmission Media is the media which has its physical existence and transmission taking place through that. In Guided Media, signals passes through a specific path. The data signals are bound by the cabling system. Guided Media is also known as bounded Media.

There are three basic types of Guided Media:

- Twisted Pair
- Coaxial Cable
- Fiber Optics.

1.4.1.1 Twisted Pair

Twisted Pair is the oldest and very common Transmission Medium. In Twisted Pair, the copper wires turn together in pair. The basic purpose of twisting is to reduce the electrical interferences from similar pair close by. Twisted Pair cable was invented by Alexander Graham Bell.

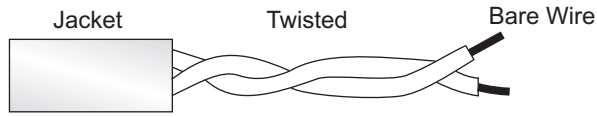


Figure 1.6

The most common application of Twisted Pair is the Telephone System. Twisted pair can be used for either analog or digital transmission. The bandwidth of Twisted Pair depends on the thickness of the wire. In a Twisted Pair, signals are not weakened frequently but it can run several kilometers without an amplification.

In a Twisted Pair, cable noise interference can be reduced by having more turns per foot. More the turns are, less is the interference. Further, noise interferences can be reduced by waving a shield around individual pairs as shown in Figure 1.6.

Twisted Pair Cable is of two types (*i*) Shielded Twisted (**STP)** as shown in Figure 1.7 and (*ii*) Cables without a shield *i.e*. Unshielded Twisted Pair (**UTP).**

- **STP** or Shielded Twisted Pair is used with the traditional Token Ring cabling, whereas **UTP** or Unshielded Twisted Pair cable is used on Ethernet and Token Ring.
- Both shielded twisted pair (STP) and unshielded twisted pair (UTP) have interference canceling capacities.
- STP cable can be used where more electrical interference occurs because shield prevents from external interferences. UTP cable is widely used in the Computer Networking.
- STP cable needs to be grounded due to the attraction of interferences to the shield. These cables are costly than the UTP. UTP cables can be easily installed in comparison to the STP. There are no external interferences to UTP cables.

UTP cable is categorized as per their data carrying capacities which are given below:

Category 1: Analog and Digital transmission (Telephone)

Category 2: ISDN (Integrated Services Digital Network) and data speed is up to 4 Mbps

Category 3: High Speed data up to 16 Mbps

Category 4: High Speed data up to 20 Mbps

Category 5: High Speed data up to 100 Mbps

Category 6: High Speed data up to 1000 Mbps

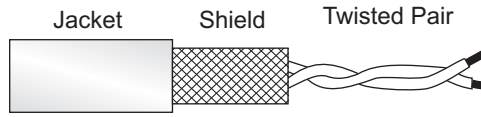


Figure 1.7

1.4.1.2 Coaxial Cable

Coaxial Cable was invented by the English engineer and mathematician Oliver Heaviside in the year 1880. Coaxial cable is mainly used for Cable Television and Local Area Network. Coaxial cable conducts electrical signal, using an inner conductor (copper wire) surrounded by an insulating layer and all enclosed by a shield (metallic braid) as shown in Figure 1.8. The cable is often protected by an outer insulating jacket.

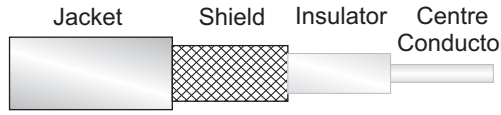


Figure 1.8

The outer shield protects the inner conductor from outside electrical signals. The distance between the outer conductor (shield) and inner conductor plus the type of material used for insulating the inner conductor determines the cable properties or impedence.

The **bandwidth** and **noise** immunity of a Coaxial cable depends upon their construction and shielding. The bandwidth also depends upon the length of the cable. There are two categories of Coaxial Cable (*i*) Baseband Coaxial Cable and (*ii*) Broadband Coaxial Cable. Broadband Systems are divided into multiple channels. Thats why television and data can be mixed in one cable. Broadband Systems can cover a large areas in comparison to the baseband with the help of Analog Amplifier which amplifies the signal periodically. For further stage, Broadband system has been developed as dual and single cable system. Typical impedences for Coaxial Cables are 75 ohms for Cable TV, 50 ohms for Ethernet, Thinnet and Thicknet. The excellent control of the impedence characteristics of the cable allow higher data rates to be transferred than a Twisted Pair Cable. Its bandwidth is up to 600 MHz.

1.4.1.3 Fiber Optics

The Transmission Medium in Optical Communication is an ultra thin fiber of glass. Glass fiber consists of a central core glass (50 mm) surrounded by a cladding, made up of glass which has slightly lower refractive index than the core's refractive index. The overall diameter of the fiber is about 125 to 200 mm. Optical Fiber works on the principle of **Refraction**. When light passes through a material; it may pass through the material or reflect depending upon the incidence angle. As shown in Figure 1.9**,** when an incident ray A passes from air to glass with incidence angle *a*, ray C is refracted with angle *c* and ray B is reflected with angle *b*.

Index of Refraction is defined as the ratio of light velocity in vacuum to light velocity in material.

An Optical Transmission System consists of three components: (*i*) light source (*ii*) transmission medium and (*iii*) detector. Conventionally, a pulse of light indicates 1 bit and the absence of light indicates a zero bit. The detector generates an electrical pulse when light falls on it. By attaching a light source to one end of an optical fiber and a detector to the other end, a unidirectional data transmission system that accepts an electrical signal, converts and transmits it by light pulses, and then reconverts the output to an electrical signal at the receiving end.

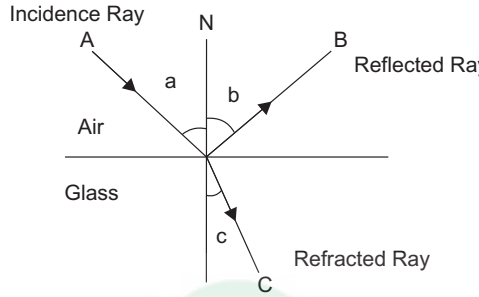


Figure 1.9

1.4.1.3.1 Types of Optical Fibers

- Based on the Refractive Index, we can categorise an optical fiber in two categories: (*i*) Step Index and (*ii*) Graded Index.
- Based on the number of modes propagating through the fiber, we can classify Optical Fiber in two categories (*i*) Single Mode and (*ii*) Multimode,

The mode describes the nature of propagation of electromagnetic waves.

Step Index Fiber: In the Step Index Fiber, the Refractive Index of the core is uniform throughout and undergoes a step change at the core cladding boundary. In the Step Index, light rays tend to bounce around, reflecting off the cladding, inside the core. It propagates in zig-zag manner. Due to this reason, some rays take a longer or shorted path through the core. Some take the direct path with minimum reflections while others bounce back and forth taking a longer path. And finally, the light rays arrive at the receiver end during different times as shown in Figure 1.10.

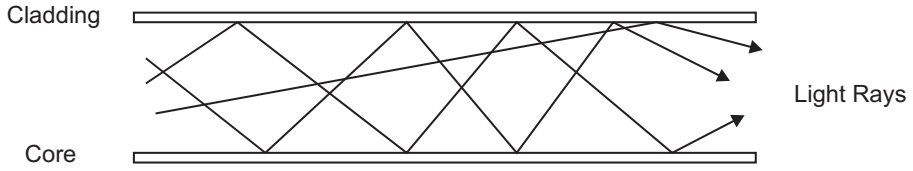


Figure 1.10

Graded Index Fiber: Graded Index has a gradual change in the Core's Refractive Index. This causes the light rays to bent back into the core path gradually. This is represented by a skewed curved reflective path in parabolic form as shown in the following Figure. It is better than the Step Index.

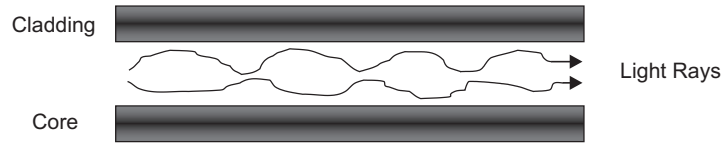


Figure 1.11

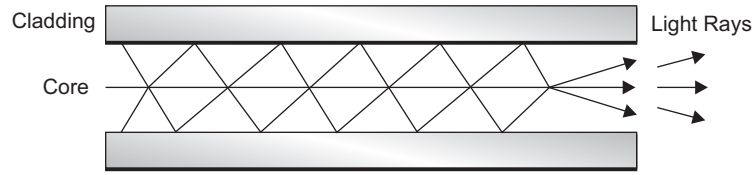
Single Mode Fiber: In a Single Mode Fiber, only one mode can propagate through the fiber. Single Mode has separate distinct Refractive Indexes for the cladding and core. In the Single Mode, light ray passes through the core with relatively few reflections off the cladding. It requires a laser and the core is very small: size of 9 microns (μ).

Figure 1.12

Multimode Fiber: Multimode Fibers allow a large number of modes for the light rays travelling through it, as shown in the following figure. In multimode fibers, the core is 50 microns (μ) in diameter.

Multimode Fiber is used for short distances, whereas Single mode is used for long distances.

Optical fiber bandwidth is up to 1 GHz.



Advantages:

(*i*) Immunity to Electromagnetic Interference.
(*ii*) Optical Fiber has low attenuation.
(*iii*) Smaller and lighter than copper wire.
(*iv*) Faster transmission rate.
(*v*) Higher bandwidth over long distance.
(*vi*) Easy installation.
(*vii*) More security.

Disadvantages:

(*i*) By bend it too much, it will be broken.

(*ii*) Difficult to fix together.

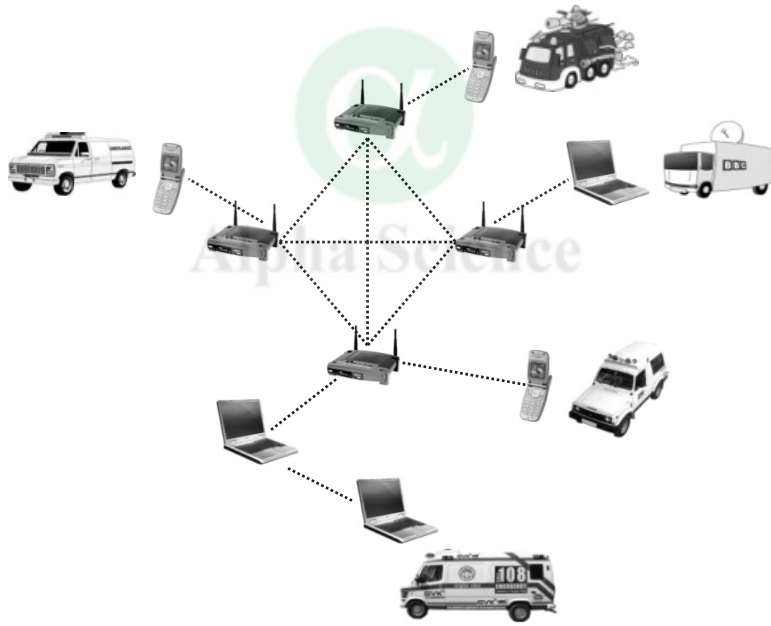
(*iii*) Installation is relatively costly.

(*iv*) Special test equipment is required.

(*v*) At higher transmission capacity, it is cheaper than copper. At lower transmission capacity, it is more expensive (trade off).

1.4.2 Unguided Media (Wireless Communication)

Wireless Communication is an indispensible demand today. Unguided Transmission Media is the data signals that flow through the air as shown in the following figure. They are not guided or bound to a channel to follow. They are classified by the type of wave propagation. Below diagram shows the structure of an unguided communication.



Following are the types of Unguided Media:

1.4.2.1 Radio Frequency (RF) Propagation

Radio Propagation is the behavior of radio waves when they are propagated from one point on the Earth to another. Like light waves, radio waves are also affected by the phenomena of reflection, refraction, diffraction, and absorption. Radio waves are highly useful in communication, both indoors and outdoors. These waves can travel long distances and has high penetration power. These

waves can travel in all the directions. These waves are frequency dependent. There are three types of RF (Radio Frequency) Propagation namely:

(*i*) Ground Wave,

(*ii*) Ionospheric and

(*iii*) Line of Sight (LOS) Propagation/Troposphere.

In atmosphere, basically there are three layers namely; Troposphere (0-10 km), Stratosphere (10-50 km) and Ionosphere (50-650 km).

Ground Wave Propagation follows the curvature of the Earth. In this propagation, waves travel along with curvature of the earth and has carrier frequencies up to 2 MHz. Ground Wave Radio Propagation is used to provide local radio communications coverage, especially by radio broadcast stations that is required to cover a particular locality. **FM** radio is an example of Ground Wave Propagation as shown in Figure 1.13.

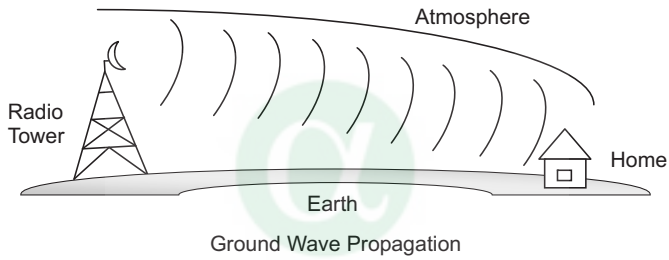


Figure 1.13

Ionospheric Propagation bounces off the Earth's Ionospheric Layer in the upper atmosphere. As early as 1902, Oliver Heaviside and Arthur E. Kennelly suggested the existence of a layer in the upper atmosphere that could account for the long-distance radio transmissions.

Ionosphere layer lies in the atmosphere between 50 to 650 km. It operates in the frequency range of 30–85 MHz. Since, it depends on the Earth's ionosphere; it can change with weather and time of day. The signal bounces off the ionosphere and back to the earth as shown in Figure 1.14.

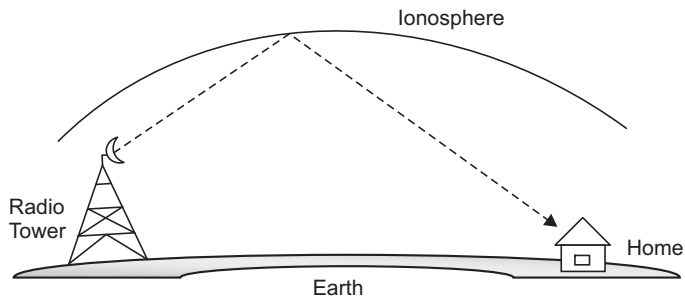


Figure 1.14

Line of Sight Propagation transmits exactly in the line of sight. The station which will receive signals must be in the view of the sender station. Line of sight propagation is also called Tropospheric Propagation. This propagation is dependent upon the curvature of the Earth for ground based stations (horizon to horizon as shown in Figure 1.15). Examples of Line of Sight Propagation are: FM Radio, Microwave and Satellite.

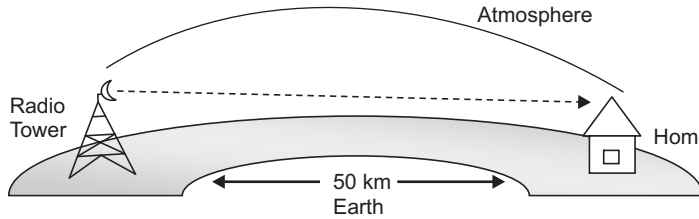


Figure 1.15

1.4.2.2 Radio Frequencies

Radio Waves of very long wavelength have the capacity to travel along the surface of the earth and into the water. Some of the popular usage of Radio Waves are Broadcast radio, short-wave radio, television, cellular telephones, walky-talkies, 2-way police radios, and satellite television. The range of Radio Frequencies is generally considered between 3 kHz to 300 GHz (1 Hz means one cycle per second). Each communication service uses a part of the spectrum which is suitable for its needs. Different range of the radio spectrum is used for different radio transmission technologies and applications. In all countries, Radio spectrum is government regulated and sold or licensed to operators of private radio transmission systems. Today the Radio spectrum is in high demand due its eased communication capabilities.

1.4.2.3 Microwave

Microwave transmission is an example of line of sight transmission. In this transmission sender station must be in visible contact with the receiving station. This sets a limit on the distance between stations depending on the local geography. Typically, the line of sight, is only 50 km to the horizon due to the Earth's curvature. Repeaters are required in the Microwave Communication to strengthen signals as shown in the following Figure 1.16.

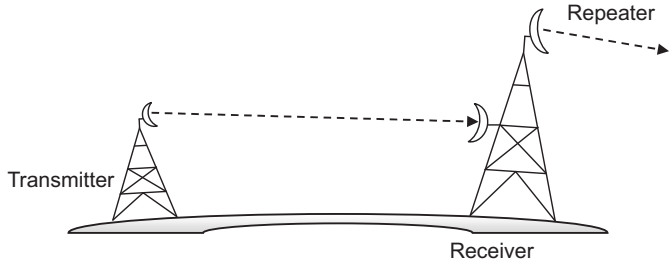


Figure 1.16

During the 1930's and 1940's various researchers conducting experiment discovered that higher frequencies could bring advantages to communication. Microwaves are radio waves with frequencies ranging between 300 MHz (0.3 GHz) and 300 GHz. This allows them to carry large quantities of data due to the large bandwidth.

Advantages:

(*i*) They can carry high quantities of information because of their frequencies.

(*ii*) Low cost in comparison with wired communication.

(*iii*) High frequency/short wavelength signals require small antenna.

Disadvantages:

(*i*) Due to the line of sight propagation, signals cannot pass through any object.

(*ii*) Attenuation by solid objects.

(*iii*) Electromagnetic interferences.

1.4.2.4 Satellite

A Satellite is an object that revolves around a planet in a circular or elliptical path. These satellites are typically 100 to 24, 000 miles away from the earth. Satellites are set in a geostationary orbit over the equator. Satellite consists of a transponder (a frequency band) which receives one frequency and retransmits another. The geostationary orbit is 36,000 km from the Earth's surface. Satellites are used in many applications which include data communications, scientific applications and weather analysis.

A diagrammatic presentation of satellite is shown in Figure 1.17.

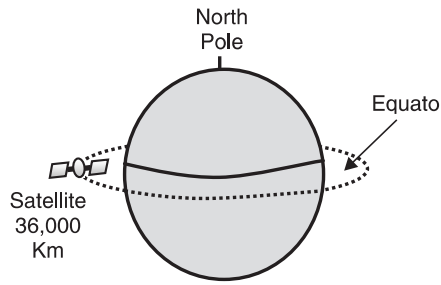


Figure 1.17

Communication through Satellite involves two processes:

- **Uplink:** It is the process of transmission of data from Sender to Satellite.
- **Downlink:** It is the process of receiving data from Satellite to Receiver.

Uplinks and Downlinks are also called **Earth Stations.** A Satellite is used to transmit information at any remote location on the earth. Mechanism of transmission through satellite has been shown in Figure 1.18.

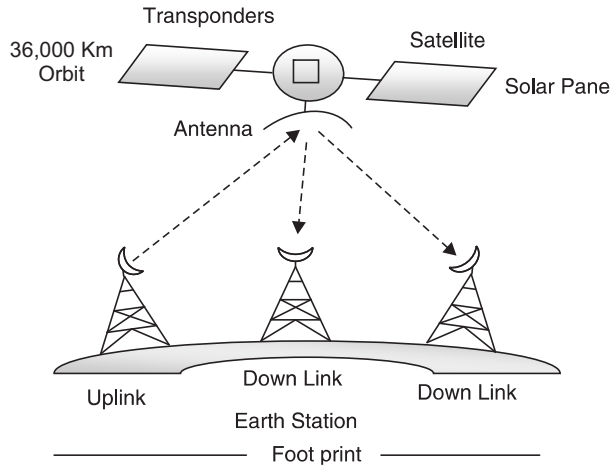


Figure 1.18

1.5 ASYNCHRONOUS AND SYNCHRONOUS TRANSMISSION (TIMING)

Data or Message consists of binary bits and is transmitted from one system to another system in a group of bits. Timing refers to how the receiving system knows that it has received the start of a group of bits and the end of a group of bits. There are two categories of timing schemes**:** Asynchronous and Synchronous Transmission**.**

Asynchronous Transmission: Asynchronous Transmission scheme transmits only 1 character at a time. In this transmission mode one and two bit is used as START and STOP in every character respectively. In this scheme, one bit is added at the most significant bit (MSB) position of the character known as start bit and one or two bits are added at least significant bit position (LSB) known as stop bit as shown in Figure 1.19.

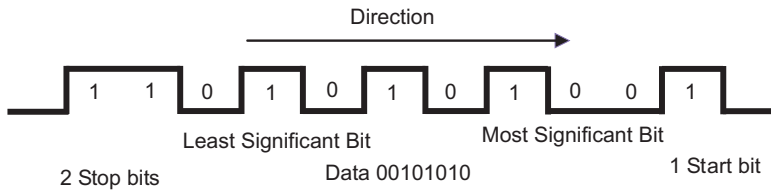


Figure 1.19

Asynchronous Transmission is mainly used to connect computer with a printer, fax machine and modem. Asynchronous Transmission suffers from high overload due to inclusion of start and stop bits for each character.

Example: If we want to transfer 20K Byte data message, then total 220 Kbits data will be transmitted *i.e.*

160 Kbits + 60 Kbits = total of 220 Kbits transmitted (one byte = 8 bits)

The efficiency of Asynchronous transmission will be calculated by dividing the data transmitted by total bits sent.

In this example the efficiency will be 72%.

For every byte of data, three bits are extra (add 1 Start Bit and 2 Stop Bits).

Synchronous Transmission: In a case of Synchronous Transmission, there is no need to add extra bit as a START and STOP bit with a character (byte). This transmission is based on synchronization mechanism where receiver 'clock is synchronized with a transmitter 'clock. Synchronous transmission sends packet of characters. Synchronous Transmission is fast in comparison to Asynchronous Transmission by having less number of bits during transmission. The application of this scheme is in all higher communication transfer rates like Ethernet, Token Ring etc

Note: Asynchronous Transmission is easy to implement but it suffers from high overhead. Synchronous Transmission is efficient but its implementation is difficult along with high cost.

1.6 SWITCHING TECHNIQUES

Transfer of data takes place between two devices which are directly connected through transmission medium. The transmission medium may be wired or wireless. But in a network system, where we have more devices cannot be connected directly to each other as it will need huge amount of cabling. For example, if we want to connect five nodes to each other, then we need 10 connections and in case of 10 nodes, we need 45 connections (formula is $n*(n-1)/2$, where n is the number of nodes). The problem of point to point connection can be solved by using switching technique. Switching is a path way through which data is transfered between two devices. There are various types of switching techniques depending upon their application and complexity.

1.6.1 Circuit Switching

Circuit Switching is the oldest switching technique and this has been used since the first communication network in the nineteenth century. This technique was invented by Almon B. Strowger. Circuit Switching directly connects sender and receiver by establishing a connection. Circuit Switching Communication System follows three steps (*i*) circuit establishment, (*ii*) data transfer and (*iii*) circuit disconnect. In this switching, once a circuit is established between the caller and the callee, it exists until the connection is terminated. During the connection establishment, resources are allocated between the caller and the callee. The set of resources allocated for a connection is called a circuit, as depicted in Figure 1.20. A path is a sequence of links located between nodes called switches. Example of a circuit-switched network is the telephone network,

where a call is made from one telephone to another; switches within the telephone exchanges, and create a continuous wire circuit between the two telephones, for as long as the call lasts.

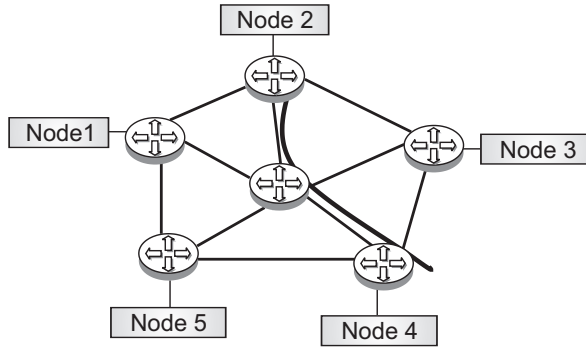


Figure 1.20

1.6.2 Message Switching

It is an alternative switching strategy which requires no physical copper path establishment in advance between the Sender and the Receiver. In such networks, the establishment in between nodes (switches) have the responsibility of conveying the received message from one node to another. Here, each intermediary node within the network stores all messages before retransmitting them. This characteristic is often referred as Store-and-Forward. A message-switching node is typically a general-purpose computer which needs sufficient storage capacity to store the incoming messages. In Message Switching, there is no limit on messages to be forwarded from one node to other.

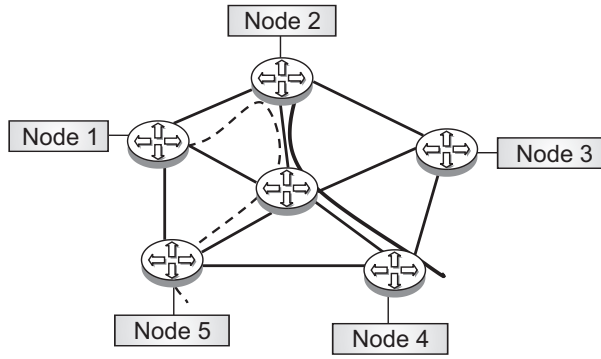


Figure 1.21

1.6.3 Packet Switching

To overcome the problem occurred in Circuit Switching (the need to allocate resources for a circuit, thus incurring link capacity is wasted when no data

flows on a circuit) and Message Switching (no limit on block size) packet switching was invented in the 1960's a by Paul Baran. In Packet Switching, a message is broken into smaller parts, called packets. In this switching, an appropriate source and destination address is attached with the packet. Each packet size in Packet Switching is fixed and typically ranges from 180 bits to 53 bytes. If no packet is sent during communication, then no resources are wasted as depicted in Figure 1.22. Due to fixed size of packet, it can be easily stored in the main memory.

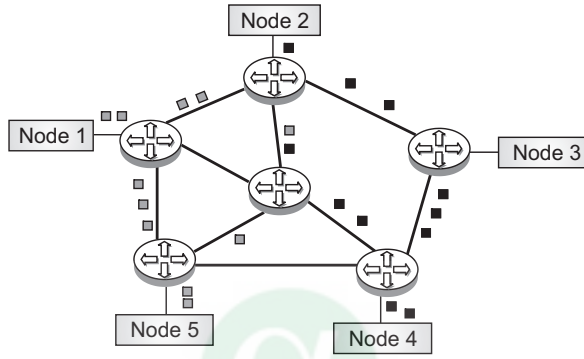


Figure 1.22

There is no fixed path in Packet Switching. Computer Networks are usually Packet Switched, occasionally Circuit Switched, but never Message Switched. Packet Switching is of two types:

1.6.3.1 Datagram

A **Datagram** is a Packet Switching Technique. It can take a variety of routes in the network and does not follow the same route, even if the destination address is same. A Datagram consists of Header and Data areas, where the Header contains source and destination addresses as well as a type field for routing and the data areas consist data.

1.6.3.2 Virtual Circuit

Virtual Circuit is also a Packet Switching Technique. In this technique, a fixed route is setup between the intermediate nodes for all packets. Virtual Circuit Packet Switching enjoys the advantages of both techniques Packet and Circuit Switching. In this technique, packets are transmitted on so-called logical circuits for which no physical resources like frequencies or time slots are allocated (see below Figure 1.23).

Virtual Circuits are more reliable due to the fact that it involves acknowledgements, flow control, and error control. Virtual Circuits take the advantages of Traffic Management Capability of Circuit Switching, and the resources Usage Efficiency of Datagram Packet Switching.

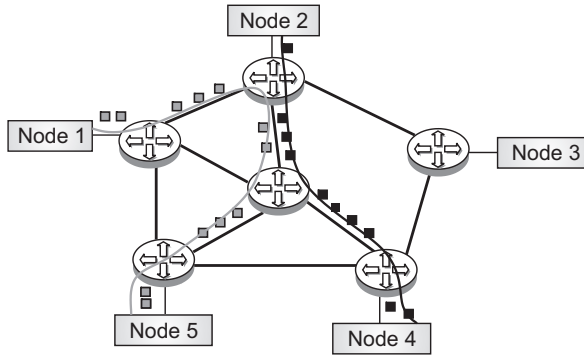


Figure 1.23

Comparison between Circuit Switching and Packet Switching:

Following is the summary of differences between **C**ircuit Switching and **P**acket Switching**:**

- Circuit Switching Technique needs a dedicated copper path, whereas it is not required in the case of Packet Switching Network.
- There is a wastage of bandwidth in Circuit Switching but not in Packet Switching.
- Packet-Switching uses store and forward transmission but in Circuit Switching there is a dedicated line.
- Each data transmission follows the same route in Circuit Switching but not in Packet Switching.

1.7 MODULATION TECHNIQUES

Modulation means converting digital signal into analog signal. After the modulation, analog signal can be transmitted over analog lines from one system to another. The reverse of this is called demodulation. For modulation and demodulation Modem device is used.

- Modem is a device which accepts a stream of bits as input and produces a modulated carrier wave as output (or vice-versa). The modem is placed between the Digital Computer and the Analog Telephone System.

As we know that there are three characteristics of any signal *i.e.* Amplitude, Frequency, and Phase. Hence, there are three basic modulation techniques**:**

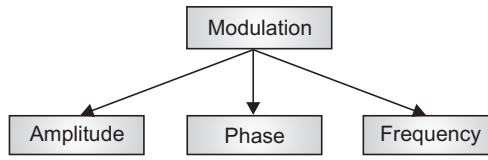


Figure 1.24

AM (Amplitude Modulation)

FM (Frequency Modulation)

PM (Phase Modulation)

A carrier is used to carry information/data. In a data communication, C**arrier Signal** is a single frequency that is used to carry data. The Modulation Technique that has been mentioned above, uses Carrier Signal. Any Digital Signal has only two values or state 0 and 1 unlike many states in Analog Signal.

1.7.1 AM—Amplitude Modulation

Amplitude Modulation (AM) is the oldest method of transmitting human voice electronically. Amplitude Modulation works by varying the amplitude of the transmitted signal. In Modulation Technique, the amplitude of the carrier is changed to represent 1's or 0's as shown in Figure 1.25.

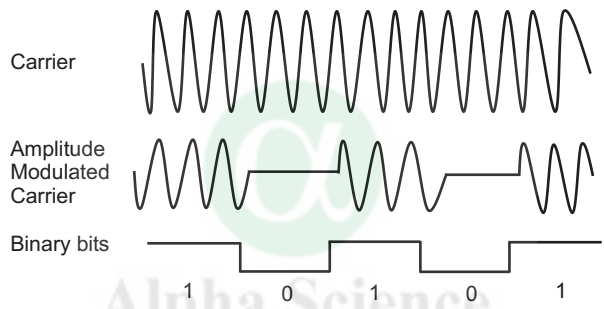


Figure 1.25

Amplitude Modulation is very simple to design but Noise spikes present on the Transmission Medium interfere with the Carrier Signal.

1.7.2 FM—Frequency Modulation

Frequency Modulation modifies the frequency of the carrier to represent the 1's or 0's. Here, 0 is represented by the original carrier frequency and 1 by a much higher frequency (the cycles are spaced closer together as shown in Figure 1.26).

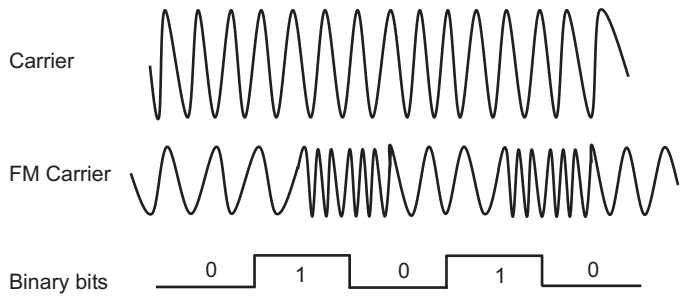


Figure 1.26

Frequency Modulation is immune to noise on transmission medium.

In this modulation technique, a signal is always present and the loss of signal is easily detected.

Disadvantages:

(*i*) This modulation technique needs two frequencies.

(*ii*) In case of loss of signal, detection circuit needs to recognize both frequencies.

1.7.3 PM—Phase Modulation

Phase Modulation modifies the phase of the carrier to represent 1 or 0 as shown in Figure 1.27.

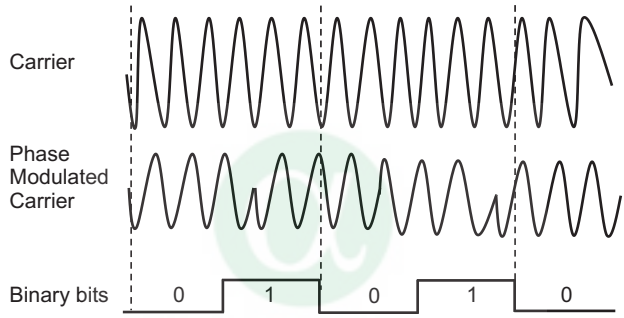


Figure 1.27

In **Phase Modulation** Technique, only one frequency is used and the loss of carriers are easily detected, but it is a complex technique as more complex circuit is required to generate and detect phase changes.

1.8 MULTIPLEXING

In a Data Communication System, if we want to transmit data from multiple source station/nodes to multiple destination stations/nodes, then we have to use many transmission lines. Since, the transmission line is very costly; therefore, the data communication becomes very expensive. To overcome this problem, Multiplexers are used. **Multiplexing** means multiple analog or digital signals combined into one signal and transmitted over a shared medium. A Multiplexer allows sharing of a common line to transmit many terminal communications. Its primary aim is to share a common resource. It is very cost effective.

After Multiplexing Multiple Signals, it is sent through a single high speed line and then it is Demultiplexed with the help of Demultiplexer and sent back to different nodes as shown in the below Figure 1.28.

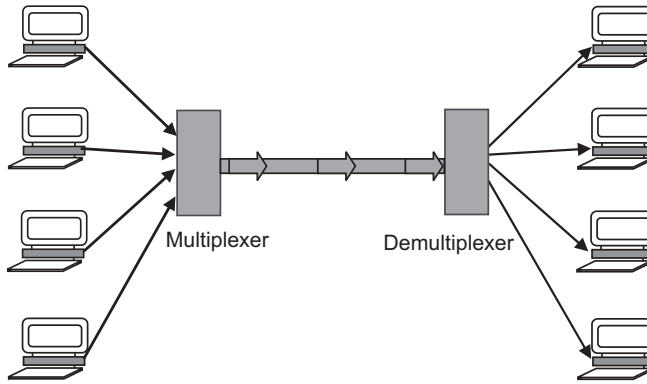


Figure 1.28

Various techniques of sharing a communication channel among a large number of users have been devised. These are known as multiplexing techniques. These signals are multiplexed using two basic techniques as shown in Figure 1.29 and 1.30.

Major multiplexing techniques can be categorized in two ways:

(*i*) Frequency Division Multiplexing (FDM)

(*ii*) Time Division Multiplexing (TDM)

1.8.1 FDM—Frequency Division Multiplexing

This technique of Multiplexing is very popular and commonly used. In this technique, the frequency range is divided to form logical channels or pathways. Each communication channel is assigned a carrier frequency. And it is obvious that channels will interfere with each other. To make sure that channels do not interfere with each other, a guard-band of defined and fixed range of frequency is used between the channels.

It is a communication technique that divides a communication channel into a number of equally spaced frequency bands. Television transmission uses FDM in which each channel is given a separate frequency and bandwidth and transmits the entire channel through a common cable. Let us understand FDM with an example, if we have 4 nodes/stations each requiring a bandwidth of 5 kHz and a 500 Hz guard-band, Node 1 will be assigned the lowest frequency channel 0 – 5 kHz, Node 2 will be assigned the next frequency channel 5.5 kHz – 10.5 kHz, Node 3 will be assigned the frequency channel 11 kHz – 16 kHz and finally Node 4 will be assigned the frequency channel 16.5 kHz – 21.5 kHz (as shown in following Figure 1.29).

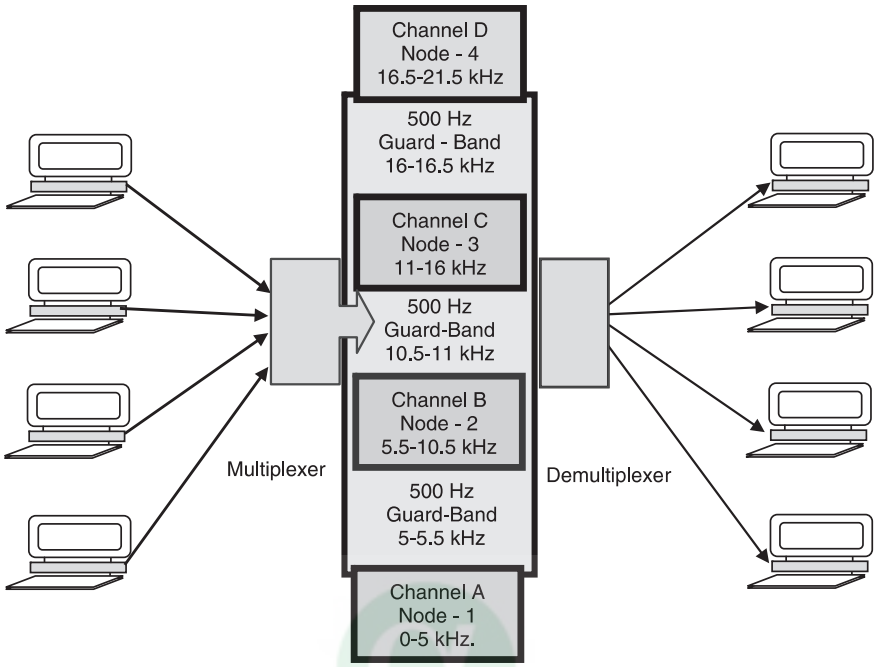


Figure 1.29

- Frequency Spectrum is divided into logical channels.
- Frequency Division Multiplexing is scalable which means more channels can be added easily.
- FDM is not sensitive to propagation delays.
- FDM is applicable for analog communication.
- Guard bands are required between the channels to keep the station separated.
- It is expensive to implement.
- Each user requires a precise carrier frequency.

The commercial application of FDM is FM radio.

1.8.2 TDM—Time Division Multiplexing

In Frequency Division Multiplexing, an analog circuit is required for wired or wireless communication. But, if we want to communicate through a computer, it will not work. In the Time Division Multiplexing, each user periodically gets the entire bandwidth for small time. In TDM, the signal is divided into time-slots; each time-slot carrying a separate signal.

In other way, TDM is a technique where a short time sample of each channel is inserted into the Multiplexed data stream. Each channel is sampled in turn and then the sequence is repeated.

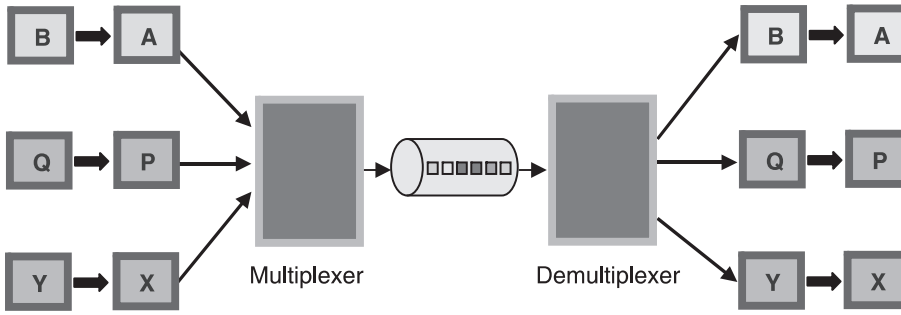


Figure 1.30

Each channel has a time slice assigned to it (see in the above Figure 1.30) whether the terminal is being used or not.

Let us take an example, during the transmission of any program on television, there is an advertisement as well. These two alternate among themselves on the same frequency. First the time allotted for a program, and then for an advertisement, then for program and so on.

- TDM uses relatively simple and less costly digital logic circuits.
- TDM is more efficient and easier to operate.
- It is used in digital communication.
- It requires all stations to synchronize in time, which is not practically possible.
- The ground stations to be capable of extremely high burst speeds.

Note: Code Division Multiplexing is another technique in which each channel transmits its bits as a coded channel-specific sequence of pulses. It is primarily used in Wireless Communication Technology.

1.9 HOW TO MEASURE THE QUALITY OF A DATA COMMUNICATION NETWORK

There are various factors which affect the quality of a Data Communication Network. Some important factors are given as follows:

1.9.1 Consistency

Consistency of a Data Communication Network is the main concern and it is defined as the expectedness of Response Time and Accuracy of data transmitted between the Sender and the Receiver. A proper consistency of a Communication Network increases the confidence level of its users. Users on the network always want consistent Response Time for getting any message or data.

1.9.2 Security

Security is the process of protecting Hardware, Software and Data from unauthorized access by an unauthorized user. To protect Hardware, Software and Data from unauthorized access, we can apply the following mechanism:

- Restricted Physical Access to computers,
- Running Anti-Virus monitoring programs,
- Password Protection,
- Limiting User Privileges, and
- Data Encryption.

1.9.3 Performance

The most important aspect of Data Communication Network is to transmit error free data from one device to another. The performance of network is defined as the rate of transferring error free data in a given unit of time and it is measured by the Response Time. On a network if more users are working, then Response Time will be poor. The Response Time of network also depends upon the transmission Speed, quality of Transmission Media, processor of the computer, Operating System of the network.

There are various parameters for measuring the performance:

(*i*) **Latency**: is the time required to transfer an empty message between computers *i.e*. sum total of delay introduced by Sender, Receiver, software, delay in accessing the network, and delay introduced by the network.

(*ii*) **Data Transfer Rate**: is defined as the speed at which the data can be transferred between the sender and the receiver in a network i.e. bit/sec.

Time consumed in transferring a message = latency + (size of message/data transfer rate).

(*iii*) **Bandwidth:** is the volume of data that can be transmitted in a given unit of time over a network.

To calculate the maximum number of data per second, the formula is given as follows:

$BW*\log_2 (1+S/N)$, where S/N is signal to noise ratio, where BW is the bandwidth of a channel.

1.9.4 Recovery

In case of any type of catastrophic failure in the network system, through recovery process, the network is able to return to a prescribed level of operation. The prescribed level can be defined, when the loss of data amount is at minimum.

1.9.5 Reliability

Reliability is defined as the probability of failure free operation of a given system in the given time under a given condition. A network failure always pertains to Hardware, Transmission Medium and Network Operating System etc.

1.9.6 Availability

The availability of a Communication Network can be calculated by computing the value of MTBF (Mean Time between Failures) and MTTR (Mean Time to Repair) and its formula is,

$$\frac{MTBF}{MTBF + MTTR}$$

SUMMARY

In this chapter, we have discussed the topics like the concepts of Data Communication and their components, Transmission Mode and Transmission Media (Guided and Unguided Media), Modulation Techniques, Multiplexing and Switching Techniques. Finally, we discuss how to measure the quality of a Data Communication System.

PROBLEMS

1. What are basic elements of a Data Communication System?
2. How to measure the quality of a Data Communication Network?
3. What are different Transmission Modes? Compare them.
4. Compare Asynchronous and Synchronous Transmission Mode.
5. Discuss different Switching Techniques.
6. Discuss various Modulation Techniques along with its advantages and disadvantages.
7. What do you mean by Data Communication and why is it needed?
8. Why more technological development in Data Communication is needed?
9. Differentiate the following:
   (*i*) TDM and FDM
   (*ii*) Asynchronous and Synchronous Transmission
   (*iii*) Guided and Unguided Media
10. What do you mean by Switching Technique?
11. Compare and contrast different Switching Techniques.
12. Write the major components of a Data Communication System.
13. Distinguish between Data Processing and Data Communications.
14. What are the various communication tasks that are being performed in a Data Communication System?

15. Define the following terms with suitable examples:
(*i*) Data, Information (*ii*) Analog data
(*iii*) Digital data (*iv*) Signal
(*v*) Data Transmission.

16. Compare the various types of transmission facilities for Data Communication.

17. Explain different Multiplexing techniques.

18. What are the characteristics that define effectiveness of a Data Communication System?

REFERENCE AND RELATED LINKS

[1] Tanenbaum S., *Computer Networks* (4th Ed.), PHI Learning Private Limited, India, 2001.

[2] Frozen Behrouz and Fegan S.C., *Data Communications and Networking*, McGraw Hill, 2006.

[3] Pahlavan K. and Krishnamurthy P., *Principles of Wireless Networks, A unified Approach*, Pearson Education Asia, 2002.

[4] Nicopolitidis P., M. S. Obaidat, G. I. Papadimitriou and A. S. Pomportis, *Wireless Networks*, John Wiley and Sons, Ltd., 2003.

[5] Tomasi W., *Introduction to Data Communications and Networking*, Pearson Education, 2007.

[6] Haykin S., *Digital Communications*, John Wiley & Sons, Inc., 2005.

[7] Gupta P.C., *Data Communications and Computer Networks*, Prentice-Hall of India, 2006.

[8] Peterson L. L. and Davie B. S., *Computer Networks: A Systems Approach* (3rd Ed.), Morgan Kaufmann, 2003.

[9] Stallings William, *Data and Computer Communications* (8th Ed.), Pearson Education, 2007.

[10] http://www.cs.virginia.edu/

[11] http://www.nickara.com/glossary_v0.htm

[12] Kurose James F. and, Ross Keith W. *Computer Networking: A Top-Down Approach* (5th Ed.), Pearson Education Inc., 2010.

[13] http://www.techbooksforfree.com.

Computer Networks

2
CHAPTER

Born : 07, April 1946
Known For : Co-invention of Ethernet


Robert Metcalfe

2.1 WHAT IS COMPUTER NETWORK?

Network means the connectivity between the autonomous entities of similar types. Here, entity means anything which has existence in the real world like man, machine, etc. There can be network between man, machine etc. Through networks, one can share the resources with each other within the defined network domain. Networks could be Social Network, Computer Network etc. In social networking, a person can communicate with each other to share his ideas, emotions, resources, etc. with each other see Figure 2.1(a), whereas a computer network is an interconnection of autonomous computers connected via communication channels to share and exchange information resources between them see Figure 2.1(b). These days social networking concepts are being implemented through Internet. Examples are; various social networking website like Facebook, Twitter, etc.

Figure 2.1 (a)

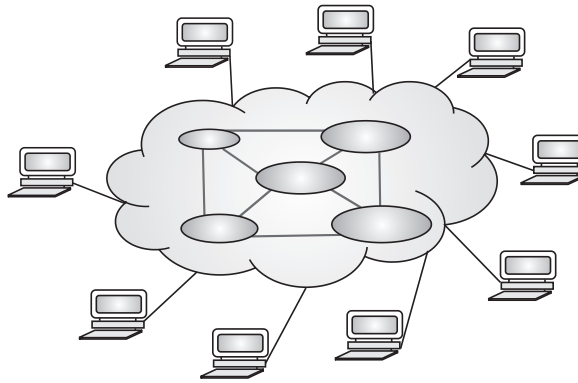


Figure 2.1 (b)

In the above paragraph, autonomous word is used which stands for "every entity in the defined network domain or outside are free from each other and can not be controlled or influenced by others"

Now the question arises why Computer Network is needed? Generally, the Computer Network is needed for the Communication and Resource Sharing purposes. The goal of Computer Network is to make all physical and logical resources (i.e. printer, scanner, program, and data) available to anyone on the network irrespective of geographical location. The following are the some examples:

1. To access remote I/O Devices.
2. To access remote Databases / Files.
3. Personal Communication through E-mail, chat, Video conferencing. Such type of communication is known as Person to Person Communication.
4. Entertainment (online movies and video games).
5. Online access of electronic resources like Books, Journals, News, etc.

The ability to merge information, communication and entertainment will surely increase the to computer networking field.

2.2 REFERENCE MODELS

2.2.1 Introduction to the ISO-OSI Model

International Standards Organization (ISO) has proposed a model in 1984 that is known as Open System Interconnection (OSI) model. The basic design work of OSI was actually done by a group headed by Mike Canepa and Charlie Bachman at Honeywell Information Systems, USA. This model has been in use for communication purposes between two devices.

OSI model proposed seven layer concepts for communication between two devices and it is widely known as OSI seven layers. Each layer has its own function. The organization of these layers are in stack form; one over each other. Each layer has its own function and the output of one layer which work as input for another. It means communication takes place between devices through these layers.

OSI layers are in proper sequence as shown in the following table, from top to bottom. In OSI reference model, the lowest layer is physical layer through which the data is transmitted physically and the top most layer is the application layer from where the application starts. All the layers have a set of protocols.

SEVEN LAYERS OF THE OSI MODEL

L-7	Application Layer (Top Layer)
L-6	Presentation Layer
L-5	Session Layer
L-4	Transport Layer
L-3	Network Layer
L-2	Data Link Layer
L-1	Physical Layer (Bottom Layer)

Below is a logical diagram of communication between two devices through different layers.

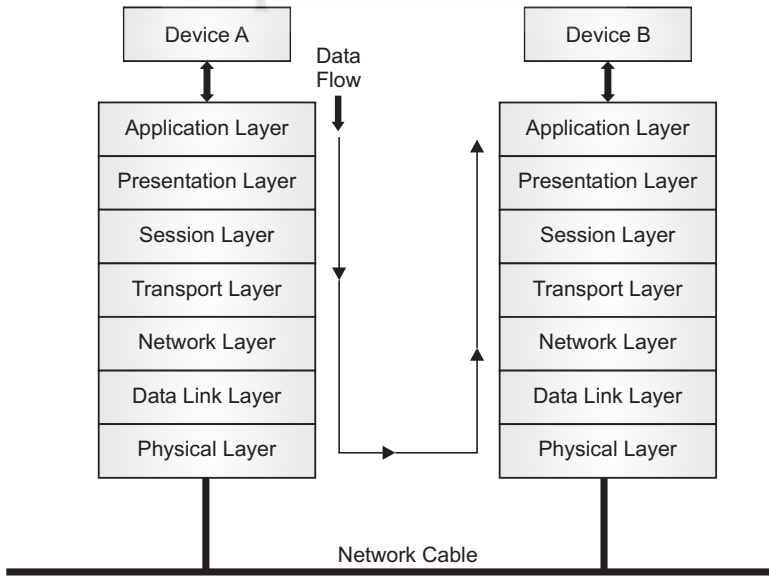


Figure 2.2

Now, we are describing the functions of each layer of OSI Reference Model.

Physical Layer

Physical Layer is the lowest layer. This is the layer through which data (in terms of bits) is transmitted from one device to another through wired or wireless media. Different media for data transmission has been described in the previous chapter. The physical layer describes the bit patterns to be used and how these bits will be encoded into media signals. This layer also describes the logical and physical organization of computers/nodes on network. This layer also deals on the flow of signals, whether it is Simplex, Half Duplex or Full Duplex (As described in the previous chapter). Some of the important technologies used in the physical layer are 10Base5- Thicknet, 10Base2-Thinnet, 10BaseT-twisted pair, ArcNet, etc.

Data Link Layer

Data Link Layer is the second layer of OSI model. Data Link Layer provides basic four functions which are as follows (Tanenbaum, 2001):

1. Data Link Layer provides services to the Network Layer. These services can be unacknowledged connectionless services, Acknowledged connectionless services, and Acknowledged connection–oriented services.
2. Data Link Layer determines how the bits of Physical Layer are grouped into frames, it breaks the bit stream up into discrete frames and computes the checksum for each frame. It is the checksum value of a packet which tells that information received at the destination is correct (different methods of framing like character count, starting and ending characters, with character stuffing, starting and ending flags, with bit stuffing and physical layer coding violations are used).
3. Data Link Layer deals with transmission errors and provides various mechanisms to check the errors so that original message can be received at the receiver end (various mechanisms are used like error-correcting codes, error-detecting codes).
4. This layer also regulates the flow of frames so that slow receivers can not be swamped by fast senders.

Some of the Data Link Layer protocols are Ethernet, FDDI, Generic Attribute Registration Protocol (GARP), Generic Multicast Registration Protocol (GMRP), Logical Link Control (LLC) protocol, SubNetwork Access Protocol (SNAP), Token Ring protocol, Address Resolution Protocol(ARP) and Reverse Address Resolution Protocol(RARP)

Network Layer

The primary function of Network Layer is to transmit packet from source nodes to destination. It deals with end to end transmission. The Network Layer provides

both connectionless and connection-oriented services. This layer is concerned about the path in the network through which the signals/data transmits. The main function of Network Layer is routing, switching and controlling the flow of information between hosts. The Network Layer converts the segments into smaller Datagrams. These Datagrams can be easily handled by the network. The Network Layer provides services to the Transport Layer.

Transport Layer

Transport Layer takes the data from the Session Layer and split into small data segments typically into size of 512 - 18K byte and transmit these to the Network Layer and ensure their correct transmission. When these data segments arrive to Transport Layer it also combines these data segments into a contiguous file. Transport Layer provides protocols for detecting errors and lost data during the transmission. It also recovers lost data and manages to retransmit the data again.

Session Layer

The Session Layer communicates with the Presentation Layer above and Transport Layer below. The Session Layer manages the communications between nodes in the network. It allows the users on different machines to establish sessions between them. A Session Layer also allows a user to log into a remote timesharing system or to transfer a file between two machines. *Sessions* are used to keep a track of individual connections to remote servers. Our web browser is an example of the use of *sessions*. It provides token management and synchronization. The Session Layer directs the information to the correct destination and identifies the source to the destination. This layer also manages Login Procedures and Password Recognition.

Presentation Layer

As we know that different machines have used different representation for the communication. In order to make it possible for computer, with different representations to communicate, a standard encoding is used on the wire. The presentation layer is concerned with the syntax and semantics of the information transmitted. The Presentation Layer presents the information by encoding in a standard which is agreed by both sender and a receiver. The Presentation Layer converts from the representation used inside the computer to the network standard representation and back. It also provides security at the file level by implementing file locking and user security.

Application Layer

Application Layer is the top most layer of the OSI Reference Model. It contains a variety of protocols. This layer is closest to the user. The various functions

involved with this layer are file transfer, e-mail etc. It involves with the various applications of the user.

2.2.2 TCP/IP Reference Model (Transmission Control Protocol/ Internet Protocol)

The TCP/IP model is a specification for the computer network protocols created in the 1970's by DARPA, an agency of the United States, Department of Defense. It laid the foundations for ARPANET, which were the world's first Wide Area Network and a predecessor of the Internet. The TCP/IP Model is sometimes called the Internet Reference Model, the DOD Model or the ARPANET Reference Model.

The TCP/IP Suite defines a set of rules to enable computers to communicate through over a network.

<table>
<tr><th>Kurose, Forouzan</th><th>Comer</th><th>Stallings</th><th>Tanenbaum</th><th>Cisco Academy</th><th>OSI Model</th></tr>
<tr><td rowspan="3">Application</td><td rowspan="3">Application</td><td rowspan="3">Application</td><td rowspan="3">Application</td><td rowspan="3">Application</td><td>Application</td></tr>
<tr><td>Presentation</td></tr>
<tr><td>Session</td></tr>
<tr><td>Transport</td><td>Transport</td><td>Host-to-host or transport</td><td>Transport</td><td>Transport</td><td>Transport</td></tr>
<tr><td>Network</td><td>Internet</td><td>Internet</td><td>Internet</td><td>Internet-work</td><td>Network</td></tr>
<tr><td>Data link</td><td>Data link (Network interface)</td><td>Network access</td><td>Data link</td><td rowspan="2">Network interface</td><td>Data link</td></tr>
<tr><td>Physical</td><td>(Hardware)</td><td>Physical</td><td>Physical</td><td>Physical</td></tr>
</table>

The Above table shows the layer names and the number of layers in the TCP/IP model as presented in the university course textbooks written by various authors.

Physical Layer

As Physical Layer is concerned there is no difference in the OSI model and TCP/IP model. Both model use same, kind of physical media.

Data Link Layer

The Data Link Layer transfer the data from the network layer of one machine to the network layer of another machine. It also converts the raw bit stream of Physical Layer into a group of bits which is called frame.

Internet Layer

The Internet Layer is the key player that holds the entire architecture all together. The internet layer defines the packet format and protocol called Internet Protocol (IP). The Internet Layer delivers IP packets to different destinations. The main concern of Internet Layer is packet routing. In this sense, we can say that internet layer in TCP/IP is similar to the functionality with Network Layer of OSI.

Transport Layer

The Transport Layer provides an end-to-end or host to host connection between source and destination. It takes data from different applications, combine them and send it to the lower layer. The Transport layer protocol makes sure that there is a reliable transmission between source and host nodes. The OSI Transport Layer Protocol and the internet Transport Control Protocol (TCP) have many similarities and differences. Transport Layer uses two protocols TCP (Transmission Control Protocol, and UDP (User Datagram Protocol).

Application Layer

Application Layer is the topmost layer in TCP/IP family. It contains all the higher level protocols. It uses various protocols like TELNET, FTP, SMTP, DNS, HTTP, and NNTP.

We are describing some of the important protocols as given below:

IP: Internet Protocol and its main job is to find the best route through the Internet to the destination. IP uses IP addresses to identify the host machine and the network. An IP address is of 32 bits. It is used in Internet Layer.

ARP: Address Resolution Protocol and it is used to map the IP addresses to MAC (Medium Access Layer) addresses. It is used in Data Link Layer.

TCP: Transmission Control Protocol and is used to guarantee host to host delivery of segments of data. TCP requests retransmission of lost data, rearranges out-of-order data and even helps in minimizing network congestion to reduce the occurrence of other problems.TCP is a connection-oriented service. This protocol belongs to the Transport Layer.

ICMP: Internet Control Message Protocol and is mainly used for troubleshooting TCP/IP network connections. Its use is to check internet connectivity by using ping command (Packet InterNet Gopher, the tool which is used to check the communication between the systems, routers, switches and also the Internet) and it also traces the route. It is used in Internet Layer.

UDP: Acronym for User Datagram Protocol and is a connectionless service. It is also a Transport Layer Protocol.

HTTP: Hypertext Transport Protocol which is used for transferring WebPages. It is an Application Layer Protocol.

SNMP: Simple Network Management Protocol which is used for managing network devices. It is an Application Layer Protocol.

FTP: File Transfer Protocol which is used for transferring files across the network and it is an Application Layer Protocol.

TFTP: Trivial File Transfer Protocol which is a low overhead fast transfer FTP protocol and comes under Application Layer.

SMTP: Simple Mail Transfer Protocol which is used for transferring email across the Internet. It is also an Application Layer Protocol.

TELNET: Terminal Emulation Link Network, an Application Layer Protocol used for remotely logging into a server across the network.

NNTP: Network News Transfer Protocol which is used for transferring news. It is an Application Layer Protocol.

IGP: Interior Gateway Protocol is used for exchanging routing information between gateways (hosts with routers) within an autonomous network. It comes under the Network Layer.

BGP: Border Gateway Protocol is a protocol for exchanging routing information between gateway hosts (each with its own router) in a network of autonomous systems. It also comes under the Network Layer.

2.3 COMPARISON BETWEEN TCP/IP AND OSI

We have discussed about OSI and TCP/IP model in the previous section. We found that both models have much in common and their functionality is almost similar.

In spite of the similarity, two said models also have many differences which are given below (Tanenbaum, 2001):

1. Initially, the TCP/IP model did not clearly distinguish between services, interface, and protocols.
2. The protocols in the OSI model are better hidden than in the TCP/IP model and can be replaced easily as the technology changes.
3. The OSI model was not biased towards one particular set of protocols because the model was devised before the protocols. In TCP/IP, the reverse is true.
4. OSI has seven layer whereas TCP/IP has four or five.
5. The OSI model supports both connectionless and connection oriented communication in the Network Layer, but only connection oriented communication in the Transport Layer. The TCP/IP model has only one in the Internet Layer (connectionless) but supports both models in the Transport Layer.

2.4 COLLISION DETECTION

During data transmission, when more than two frames are transmitted simultaneously, then it is possible that these frames can overlap in time and the resulting signal may jumble. This phenomenon is called a collision. At the time of occurrence of collision, both the packet/frame is destroyed. The network is monitored during the transmissions to detect collisions and the transmission is stopped immediately. If a node detects a collision, then immediately a signal is sent on the network to inform all the nodes about the collision. After that all the nodes disregard all the corrupted packets they may have been receiving, later a collided frame must be transmitted again. CSMA/CD protocol (Carrier Sense Multiple Access with Collision Detection) is widely used on the LANs in the MAC (Medium Access Sublayer) for this purpose.

MAC address is linked to the hardware of network adapter. MAC address is stored on the computer's network card and this address cannot be changed.It is a 48 or 64-bit address associated with a network adapter.

2.5 CONGESTION CONTROL

During the transmission of packets, there are situations when multiple input lines want same output line to send the packets and if router memory is not sufficient or processor is not much powerful, then some packets may be lost and performance degrades as shown in the following figure. This is called congestion and various algorithms have been devised to cope up with the congestion control. Some of them are traffic shaping, load shedding, jitter control etc.

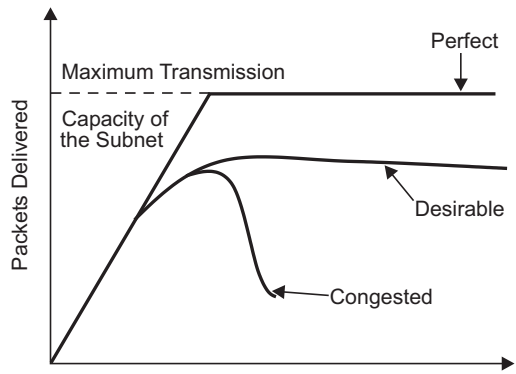


Figure 2.3

2.6 NETWORK SECURITY

Computer Network is used everywhere for different purposes and applications ranging from banking, ticketing, shopping to filing income tax return. Network Security can be categorized in four categories namely secrecy, authentication, no repudiation and integrity control. Various techniques and algorithms have been devised to make the network secure. Some of the techniques are cryptography, secret-key algorithms, public–key algorithms, authentication protocols, digital signatures etc.

2.7 COMPONENTS OF NETWORK INFRASTRUCTURE

Generally in Computer Network we talk about Server and Client but apart from that there are certain devices which are required for the interconnection of computers, their speedup and proper efficient functioning. The following network devices are required to setup network infrastructure:

- Hub
- Repeater
- Gateway
- Bridge
- Router
- Brouter
- Switch
- Terminal Server
- NIC (Network Interface Card)

Hub: is the basic physical hardware device used during the computer networking. Hubs are used in star topology and its basic purpose is to connect individual nodes/devices on a network so that they can make communication with each other. Hubs are also called Multi-port Repeaters or Connectors.

The diagram given below (Figure 2.8) shows how all the nodes are connected to a Hub. It is also clear that all the network cables come to the central hub. Due to this, it is especially easy to detect and fix cable problems.

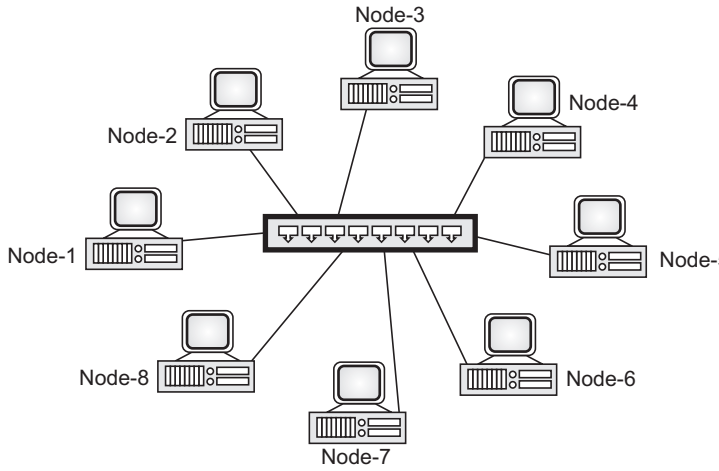


Figure 2.4

Repeater: When electrical signals are transmitted over a long distance, signals become weak and attenuated and their shapes are changed. Repeaters are physical hardware devices that have been used to strengthen/amplify or regenerate weak signals. It is an electronic device that receives a signal and retransmits it at a higher level or higher power (see Figure 2.5) Repeaters are needed to provide current to drive long cables. Repeaters are used at Physical Layer of Reference Model. Its primary function is to regenerate the electrical signal by:

- Reshaping the waveform
- Amplifying the waveform and
- Retiming the signal

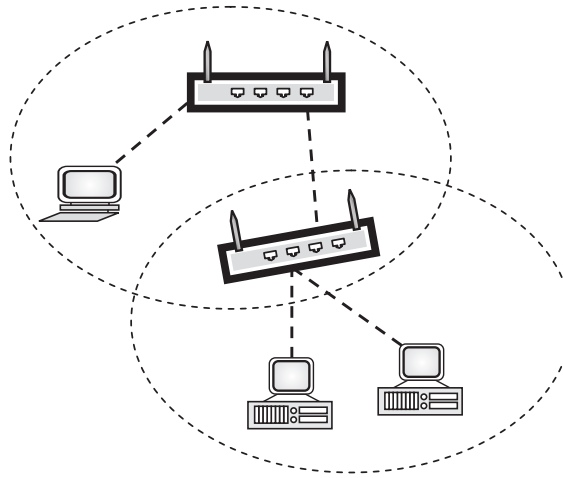


Figure 2.5

Gateway: Different networks use often different hardware and software platform. If we are in one network environment and want to communicate with another network, which has different hardware and software, then we need a machine, which can provide necessary transplantation in terms of both software and hardware. Thus, gateways are used to connect dissimilar type of networks (see Figure 2.6). Gateway works at Transport Layer. It is an internetworking device. It is used to interconnect LANs & WANs.

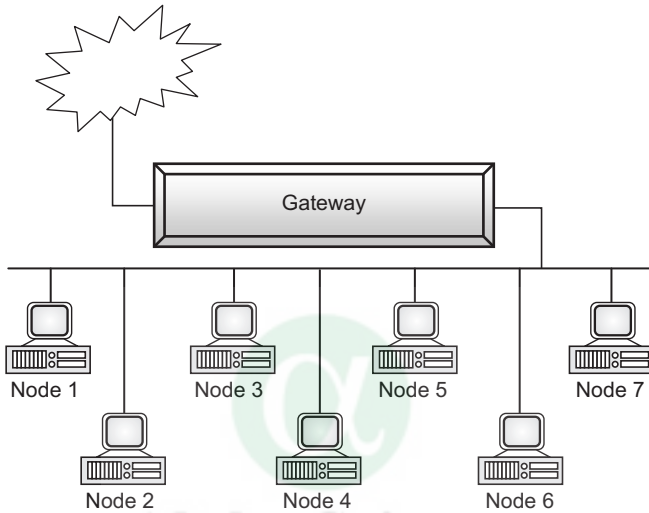


Figure 2.6

Bridge: Bridges (see Figure 2.7) are the hardware and software devices used to connect LANs because many organizations have multiple LANs. Bridges operate in the Data Link Layer. But the main question is why to have more LANs in any organization? There are several reasons (*i*) every department of the organization has its own autonomy and wants to connect its own departmental computers, but at the same time it wants to be connected with other departments also (using bridges). (*ii*) There are several geographically spread over buildings of the organization. It may be cheaper to have separate LAN in each building and connect them with bridges and infrared links (south campus colleges are spread over, so they are connected through infrared link to south campus) than to run a single coaxial cable over the entire site. (*iii*) It may be necessary to split a single LAN into multiple to accommodate the load. (*iv*) to provide more security and reliability.

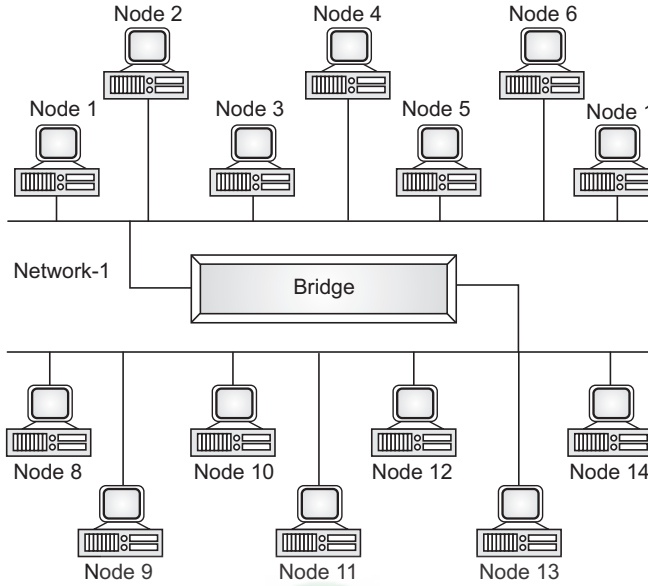


Figure 2.7

Router: Routers are devices which consist of both hardware and software. It is used to connect nodes across the internetwork. Routers operates in the Network Layer. Routers send the information through best possible route from the source to destination. Routers create and maintain a routing table which stores the best possible routes for network destination. Various routing algorithms are used for deciding optimal path for transmitting an incoming packet.

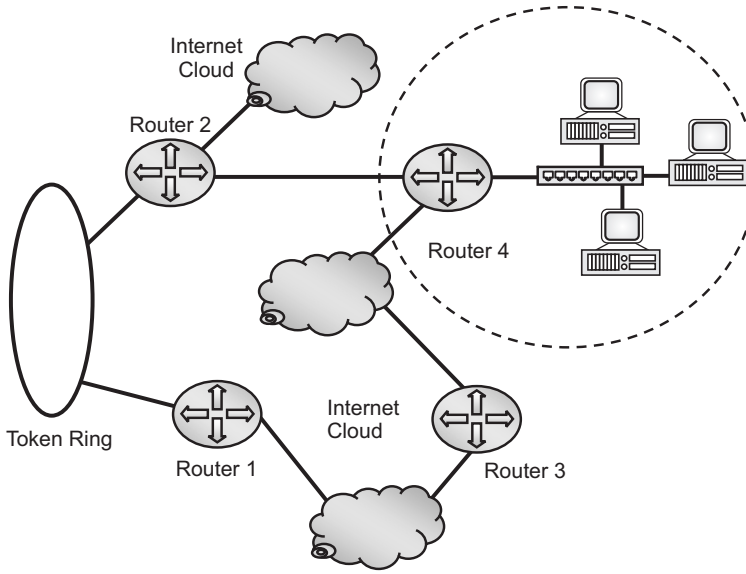


Figure 2.8

Brouter (Bridge/Router): Brouter is a network device which works like Bridge and Router, depending upon the situation. It works in Network Layer. Brouter handle both routable and non-routable features by acting as routers for routable protocols and bridges for non-routable protocols.

Token Ring: In Token Ring, a special packet (a special bit pattern) is passed around the ring called a Token. It is used in Physical and Data Link Layer. Nodes on the network can access any information through Token. If any node on network wants to access ring, then that the node has to grab Token first and check the destination address attached with the Token. The Token is passed all over the network. Physically, a Token Ring network is wired as a Star Topology by using a Hub.

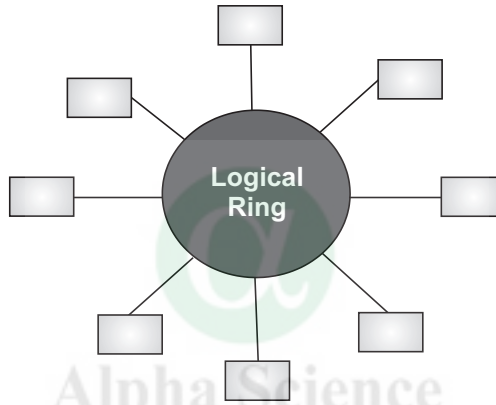


Figure 2.9

IEEE has included the Token Ring standard as IEEE 802.5.

Switch: A Switch is a network device which performs switching. It is a small hardware device that joins multiple computers together within one Local Area Network. Network switches operate at Data Link Layer. A switch has number of ports, where nodes of the network get connected. A Switch may be of 16 port switch or 32 ports switch or even more. Network Switches are capable of inspecting data packets as they are received, determining the source and destination address of each packet, and transmitting them to an appropriate node.

Network Interface Cards: A NIC is a hardware component designed to allow computers to communicate over a Computer Network**.** NIC provides physical access to a networking medium. It also provides a low-level addressing system through the use of MAC addresses. It allows users to connect to each other either by using wired or wireless media.

Following figure shows the structure of a Network by using different network devices.

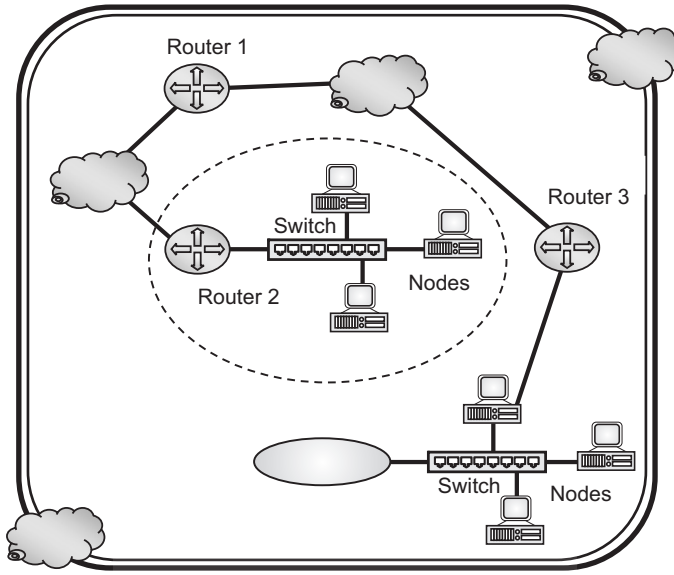


Figure 2.10

2.8 TYPES OF COMPUTER NETWORK (FUNCTION)

- Client-Server Network
- Peer to Peer Network

2.8.1 Client-Server Network

In a Client-Server Network model, users are called Clients (usually called computer workstations). Client devices are typically PCs with network software applications installed that request and receive information over the network. Server is a higher-powered central processor, with more memory, and larger disk drives than clients. A Server device typically stores files and databases including more complex applications like Web sites. For example file server, print server, data base server, web server, etc. Web Servers on the Internet are typically high-end dedicated server computers with very fast processors (or multiple processors) and huge hard disk arrays. The typical computing model for many applications is a Client/Server model. A Server computer has vast resources and responds to requests for resources and data from Client computers. Client computers initiate requests for resources or data from the Server computers. A good example of the Client/Server model of computing is Web Browsing. The Web Server stores all of the content associated with a Web Site (HTML files, graphics, audio and video files, etc.) and listens for incoming requests to view the information on a particular Web Page. When a page is requested, the Web Server sends the page and its associated files to the requesting client.

Main Server softwares are Apache, IIS, Windows 2000, Windows NT or Novell Netware.

The Client-Server model is particularly recommended for networks requiring a high degree of reliability. Its main advantages are:

- **Centralized resources:** In Client, the Server model provides all the resources to its clients. Client Server model is very much useful in the automation of any organization where only software is installed at Server site and serve the purpose of all clients/users. We can have centralized database to avoid problems caused by redundant and inconsistent data (redundancy and inconsistent terms are defined in database chapter).
- **Improved security:** This model is more secure as centralized security can be provided.
- **Accessibility:** Server can be accessed by any computer located nearby or at remote location.
- **Server level administration:** Clients can only made a request and get desired information and they do not play a major role in this model, as they require less administration.
- **Scalable network:** In this model, it is possible to remove or add clients without affecting the operation of the network and it does not need any major modification.

There are some disadvantages of the client-server model which are given below:

- **Dependability:** The whole operation of system is dependent upon server; if server fails, then the entire system will fail.
- **High cost:** In this model the cost will be increased due to high level configuration in terms of processor speed, memory capacity etc.

A Client/Server system operates as outlined in the following diagram:

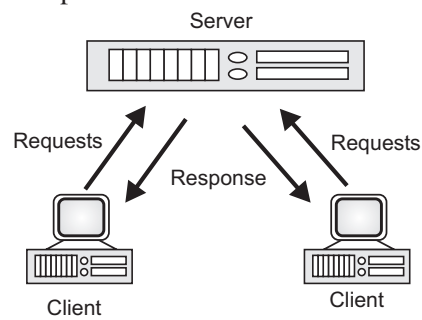


Figure 2.11

From above figure it is clear that:

- In this model, a Client sends a request to the Server using its IP address and the port.

- The Server receives the request and responds using the Client IP address and port.

2.8.2 Peer to Peer Network (P2P)

Which type of networking is to be done depends upon the requirements of the organization. In Peer to Peer networking, all the nodes are connected to each other via networking devices as shown in Figure 2.12. Peer to Peer networking is useful in a situation, when each node wants to share a common resource or get connected together for internet purpose:

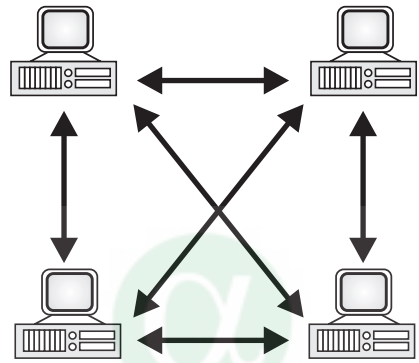


Figure 2.12

From the above figure it is clear that no dedicated server is required like Client Server model. All the nodes on network are equal and they manage their own security and administration locally.

Following are the advantages of using Peer to Peer Networking:

- **Low Cost:** The setup cost of Peer to Peer is very low as no dedicated server is required.
- **Easy Installation:** Installation of Peer to Peer network is very easy as no centralized software is needed to be installed.
- **Dependency:** As there is no dependency on any computer, all the computers are independent, and are connected only for a common purpose. There is no central dependency in P2P networking, hence it is more reliable.

But there are also some drawbacks of P2P networking.

- In P2P, security a is big issue. Since there is no centralized security management, we have to install local antivirus on all the nodes.
- Each computer should have their own local memory for storage of data and program.
- We can not use P2P networking for automation purposes in any organization, as automation process requires centralized software to be installed.

P2P networking is required for small network, where the all nodes want to share a common hardware resource or just want to communicate with each other, no centralized hardware or software resource is required.

2.9 TYPE OF COMPUTER NETWORKS

We also categorize the Computer Network on the basis of the distance between the computers in a network. Whether the computers are in a department or in a city or in a large geographical area like a country or a continent. There are three categories of network, which are described below:

- Local Area Networks (LANs)
- Metropolitan Area Networks (MANs)
- Wide Area Networks (WANs).

2.9.1 Local Area Networks

In Local Area Network, computers are spread within a building or campus of a few kilometers in size. LANs are generally used in offices or in a departmental lab to share resources such as printers or files. We are showing below a structure of a LAN:

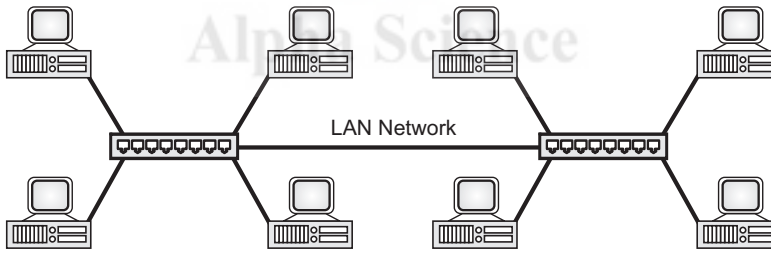


Figure 2.13

In order to set up a LAN, we need to have some network devices asdiscussed in the previous section along with wired or wireless media for data transmission. Different transmission media has been discussed in the previous chapter. Some of the LAN protocols are FDDI (Fiber Distributed Data Interface), Token Ring, and ATM (Asynchronous Transfer Mode) etc.

2.9.2 Metropolitan Area Networks

Its coverage area is larger than LAN. Metropolitan Area Network usually covers a city or a large campus. It connects a number of LAN, by using high speed connectivity and different network devices as shown in Figure 2.14. It is shown

how two campuses of the University of Delhi get connected through high speed connectivity media. The MAN is also applicable for the company or office which has different branches within a city. Some of the MAN protocols are X.25, ISDN (Asymmetric digital subscriber line), ADSL and ATM etc.

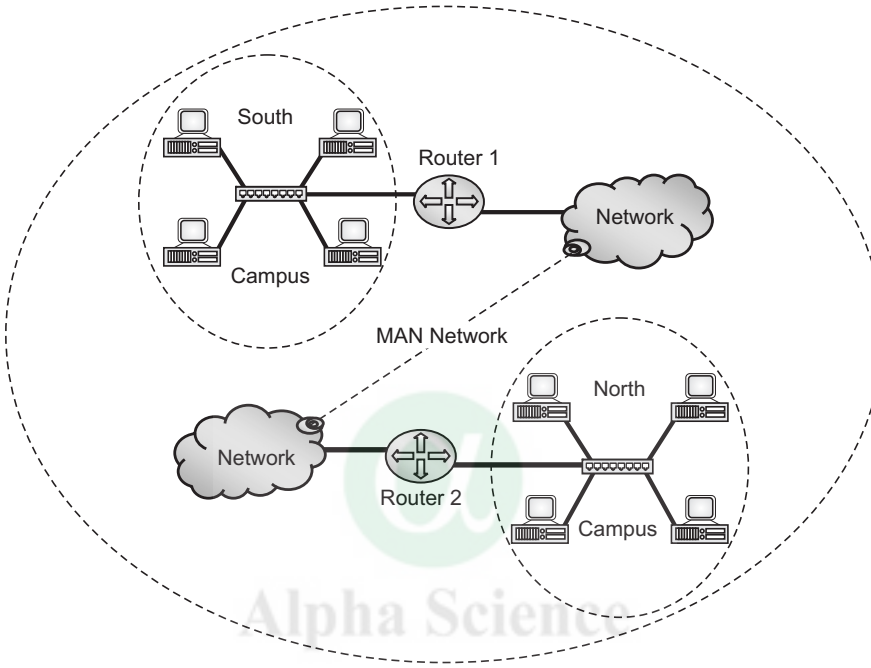


Figure 2.14

2.9.3 Wide Area Networks

In a Wide Area Network, nodes/computers/workstations are located at different geographical locations; a country or a continent. WAN connects Local Area Networks between cities. Today, WAN is widely used for different commercial applications like Banking Industry, ticket reservation, communication, etc. All the protocols of MAN are also used for WAN (see Figure 2.15).

There are other categories of network which are described below:

Personal Area Network: In Personal Area Network, we connect our personal computer with printer, fax machine, telephone or scanner by a wire or Bluetooth technology.

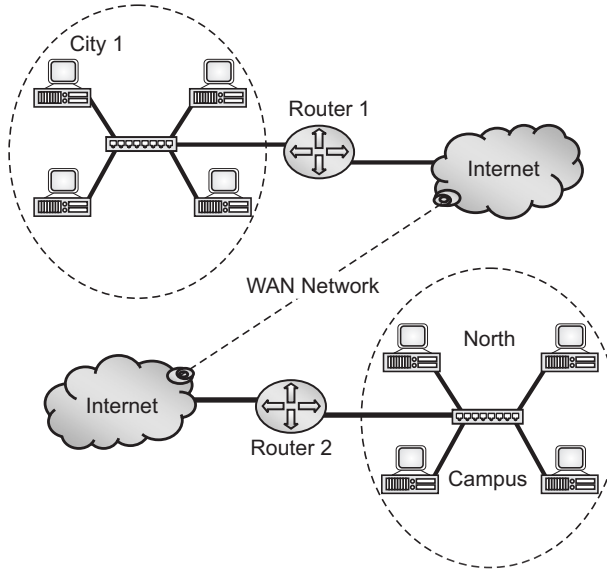


Figure 2.15

Internetwork: Internetwork means connectivity within networks. It may be an interconnection among or between private, public, commercial, or governmental networks. There are three categories of internetwork which are described below:

- Intranet
- Extranet
- Internet (Pl. see Internet chapter)

Intranet

Intranet is applicable within an organization, where computers are connected to each other. It is the internal network of an organization. An Intranet is a set of networks; which uses web browsers and file transfer applications that are under the control of a single administrator.

Extranet

In many cases, two organizations are connected to each other. It is the network between two organizations.

2.10 NETWORK TOPOLOGIES

Literally, Topology refers to the shape or structure of a system which consists of different elements. For any system, there are two things: one is physical topology, which means how the elements are physically connected and another one is logical topology which means what are the access mechanism i.e. how

the data is transferred between nodes/workstations. In a computer network, physical and logical topology methods are used to pass the information between nodes. In other words, Network topology means a kind of logical and physical arrangement of the nodes, link, etc. Basically, it gives a shape to the network. In simple terms, is the interconnection between nodes.

2.10.1 Bus Topology

In **Bus Network Topology,** a set of nodes are connected via a shared communications line, called a bus. In a bus network, when a message is transmitted, then it passes to all the nodes. If the address attached in the message is of a particular node, then it catches, otherwise it forwards it to the other node. In this network, each node has an equal priority in using the network to transmit data.

This topology has a mechanism of collision handling or collision avoidance for communication on the bus, when the message is transmitted from more than one node at the same time.

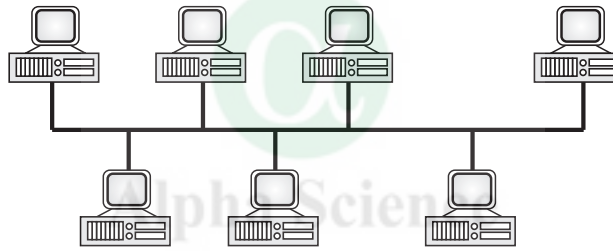


Figure 2.16

Advantages

- Bus Topology is easy to implement.
- It is scalable, which means more nodes can be added easily.
- It requires less cabling because all the nodes on network are connected through the same cable.
- This topology is cheaper.

Disadvantages

- The biggest problem with this topology is, if there is a problem with the cable, the entire network goes down.
- Maintenance costs may be higher in the long run.
- If the number of nodes is increased, then the performance degrades.
- It works best with limited number of nodes.
- It is slower than the other topologies.

2.10.2 Ring Topology

In a **Ring Network Topology** each node/workstation is exactly connected with two other nodes. In this topology, data travels from node to node, with each node handling every packet. This topology requires Repeater, because signals passes from one node to another.

Advantages

- We can create much larger network by using Token Ring.
- In a ring topology, every device/node has to access the token and opportunity to transmit information.
- It does not need any server.
- This is an efficient topology.

Disadvantages

- If we have a failure connection at any node, then the whole network will fail.
- Bandwidth is shared on all links between devices.
- Addition or changes in nodes affect the ring.
- Much slower than an Ethernet Network under normal load.

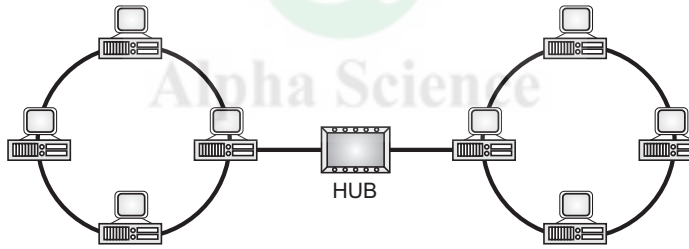


Figure 2.17

2.10.3 Mesh Topology

In a Mesh Network, all nodes/workstations/devices are interconnected to each other via communication media. Mesh Networks can still operate if any node breaks down or a connection goes bad. It is a very reliable network. This concept is applicable to Wireless Networks, Wired Networks. In fully a connected Mesh Topology, we need $n*(n-1)/2$ connections.

Advantages

- Theoretically, it is more reliable because all the nodes are connected to each other.
- It provides maximum speed and security.
- This topology is most suitable for Wireless Communication.

Disadvantages

- It is very expensive as it has redundant connections.
- It is not suitable for Computer Networking.

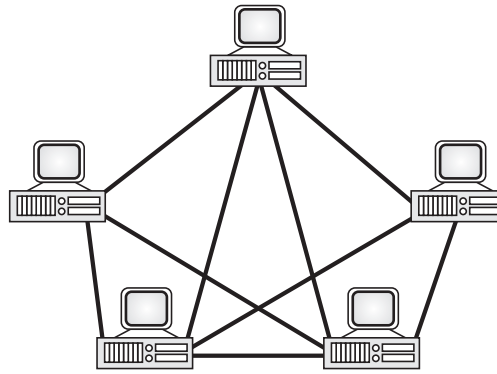


Figure 2.18

2.10.4 Tree Topology

Tree topology is a combination of both Bus and Star Topology. The branches of Tree Topology works as Bus topology and root follows star topology (See Figure 2.19). Tree topologies allow for the expansion of network. Nodes can be easily attached on branches and also more roots can be created. In a Tree topology, if any node is disrupted, then it will get isolated without affecting the network. But, if the central hub/root goes down, then all the nodes related to it also goes down.

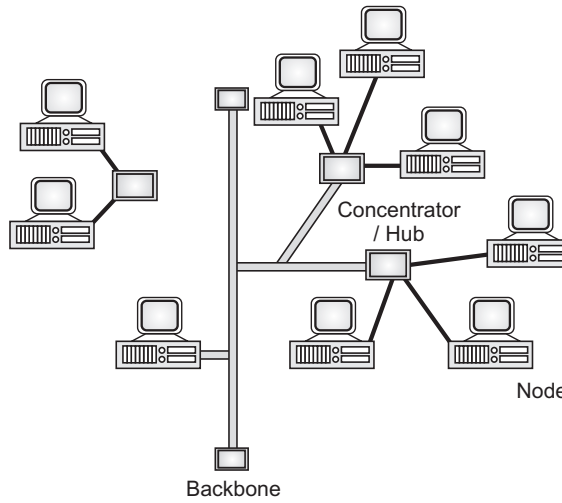


Figure 2.19

Advantages

- Tree Topology is suitable for large network.
- It can be managed easily because it is divided into various parts.

Disadvantages

- If the backbone line breaks, the entire segment goes down.
- More difficult to configure, in case we want to add very large number of nodes.

2.10.5 Star Topology

In a Star Topology, all the nodes/workstations/devices are connected to a Central Hub or Switch. In a Star Network Topology, any communication between nodes or among the nodes is through a hub or switch. The configuration in Star Topology can be done with Twisted Pair cable, Coaxial cable or Fiber Optic cable.

Advantages

- Easy to detect faults in Star Topology.
- Implementation of Star Topology is easy.
- We can detach/remove any node/device from Star Topology very easily.

Disadvantages

- The functionality of whole network depends upon the hub /switch, if it fails, then the whole network will fail.
- In a Star Topology, we require more cable in comparison to Linear Bus Topology.
- Star Topology is expensive.
- It requires more cabling.

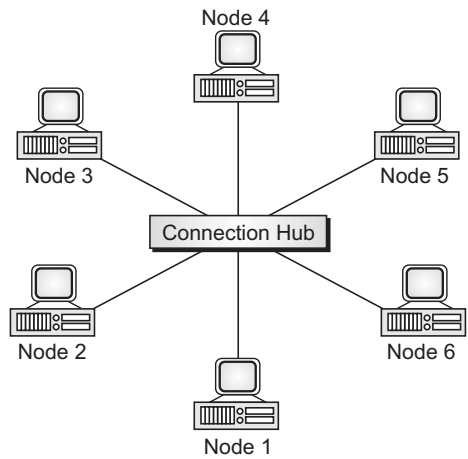


Figure 2.20

2.11 FACTORS AFFECT CHOOSING A TOPOLOGY

There are various factors which count in deciding the topology for various applications. Some of them are as follows:

- **Cable Category:** The most common cable is Unshielded Twisted Pair (UTP), which is most oftenly used with Star Topology.
- **Break Down:** Break Down of Network is also an important consideration, Mesh Topology has very less chances of break down as it provides Point to Point connection but is more expensive.
- **Size of cable needed:** The Linear Bus Network uses shorter length of cable.
- **Cost:** The cost of Bus Network is very less**.**
- **Application:** Selection of topology also depends on the application and for which purpose the networking is required.
- **Scalability:** Star Topology is more scalable.

2.12 SUMMARY CHART

In the following chart, we are showing which type of Physical Media and Protocols are required for a particular type of Topology.

Physical Topology	Common Cable	Common Protocol
Linear Bus	Twisted Pair Coaxial Fiber	Ethernet
Star	Twisted Pair Fiber	Ethernet
Mesh	Twisted Pair Coaxial Fiber	Point to Point Protocol
Ring	Twisted Pair	Token Ring
Tree	Twisted Pair Coaxial Fiber	Ethernet

BASIC TERMS OF DATA COMMUNICATION AND COMPUTER NETWORKS

- **Baud:** It is also known as Baud Rate. It is a unit of signaling speed equal to the number of discrete signal elements transmitted per second. It is measured in bits per second (bps). Its other higher level measurement, it is Kbps (Kilo byte per second), Mbps (Mega bit per second), Gbps (Giga bit per second) and Tbps (Tera bit per second).
- **Bit Stream:** The continuous series of transmitted bits through a transmission link.
- **Carrier Wave:** The wave upon which a signal is superimposed.
- **Check Bit:** It is a parity bit used as an error detecting code.
- **Collision:** Simultaneously used by two stations of a shared transmission medium in a LAN environment.

- **CSMA (Carrier Sense Multiple Access):** A contention-based LAN access method where terminal stations listen prior to transmitting, send a packet of data, then free the line for other stations to transmit . While stations do not transmit until the line is clear, transmission collisions still occur.
- **CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance):** A CSMA protocol using a slotted TDM to minimize a collision reoccurrence. For optimum results, CSMA/CA will work best, if the time slot is short as compared to the length of the packet.
- **CSMA/CD (Carrier Sense Multiple Access with Collision Detection):** A LAN access method where all stations attached to the network listen for transmissions before attempting to transmit. If two or more devices begin transmitting at the same time, each stops for a random period of time before attempting to retransmit. Also see IEEE 802.3 and Ethernet.
- **Hertz (Hz):** The same as cycles per second, used as a measurement of bandwidth or frequency.
- **Keying:** A method of encoding data by modulating the carrier either by phase or frequency
- **Nyquist Theory:** A communication theory recommending a two-sample per cycle process to characterize an analog signal limited by bandwidth. The rate of sampling must be twice the highest frequency component of the signal. For example, a 200 Hz analog signal would be sampled 400 times.
- **Parity Check:** Parity checking is an extensively used error-checking facility provided to insure correct recording of data.
- **Parity Bit:** An error-checking bit whose binary value (0 or 1) depends on whether the sum of bits with the value 1 in the unit of data being checked is odd or even.
- **Subnet Mask:** A Subnet Mask is a 32-bit number that masks an IP address and divides to IP address into network address and host address.
- **Fram Relay:** Fram Relay is a high-performance WAN protocol that operates at physical and data link layer of OSI. It is an example of packet-switched technology.

SUMMARY

After reading this chapter students have learned the following concepts:

- Computer Network and their types, description of various networking devices and protocols.
- OSI and TCP/IP reference model, Network Topology and Types of Network.

PROBLEMS

1. Explain the different networking devices.
2. Compare and contrast different network topologies.
3. How to setup a Computer Network?
4. Compare and contrast OSI and TCP/IP reference model.
5. What are the two types of transmission technology available?
6. Difference between the communication and transmission.
7. What are the possible ways of data exchange?
8. What is frame, in which layer it comes?
9. What is Brouter?
10. How Gateway is different from Router?
11. What is attenuation?
12. What is MAC address?
13. What is ICMP?
14. What are the data units at different layers of the TCP / IP protocol suite?
15. What are major type of networks and explain?
16. What are the important topologies for networks?
17. What is mesh network?
18. What is multicast routing?
19. What is Gateway?
20. What is IGP (Interior Gateway Protocol)?
21. What is autonomous system?
22. What is BGP (Border Gateway Protocol)?

REFERENCES AND RELATED LINKS

[1] James F. Kurose, Keith W. Ross, Computer Networking: A Top-Down Approach, 2008.

[2] Behrouz A. Forouzan, Data Communications and Networking, McGraw Hill, 2006.

[3] Douglas E. Comer, Internetworking with TCP/IP: Principles, Protocols and Architecture, Prentice Hall, 2005.

[4] Charles M. Kozierok, "The TCP/IP Guide", No Starch Press, 2005.

[5] William Stallings, Data and Computer Communications, Prentice Hall 2006, ISBN 0-13-243310-9.

[6] Andrew S. Tanenbaum, Computer Networks, Prentice Hall, 2001.

[7] Mark Dye, Mark A. Dye, Wendell, Network Fundamentals: CCNA Exploration Companion Guide, 2007.

[8] http://www.techbooksforfree.com

[9] http://fcit.usf.edu

Basics of Internet

3
CHAPTER

Born	: 23, June 1943 New Haven, Connecticut, USA	 Vint Cerf
Known For	: TCP/IP, Internet Society	
Notable Awards	: National Medal of Technology, Presidential Medal of Freedom, Turing Award.	

3.1 BASIC INTERNET TERMS

Browser: It is an application software through which user can download or upload the information in the World Wide Web.

Cyberspace: Cyberspace is nothing but World Wide Web from where the user can get any kind of relevant information of their interest.

DNS (Domain Name System): Since every computer has its own IP address. Computers can be connected through IP addresses. In the World Wide Web, it's very difficult to access the information using IP address. To simplify it, a name is assigned to IP address. This naming scheme (which consists of a hierarchical sequence of names, from the most specific to the most general (left to right), separated by dots) is implemented through DNS.

Downloading: Download is the process of getting information from a remote host computer to your local microcomputer.

Uploading: Upload is the process of sending information from local computer (Client) to remote host computer (Server).

Electronic Bulletin Board: A shared file where users can enter information for other users to read or download. Many bulletin boards are set up according to general topics and are accessible throughout a network.

E-Mail (Electronic Mail): An Internet tool that enables the exchange of information between network users or groups of network users.

Firewall: Firewall is used for security measures. It is designed to protect a networked system from unauthorized access.

FTP (File Transfer Protocol): FTP is used to send a copy of file from one computer to another.

Web page: It is an electronic page written by using a markup language(HTML).

Home Page: Main page (First) of a web site.

Web site: Collection of web pages.

HTTP (Hypertext Text Transfer Protocol): The web based client server protocol is used to download and upload the information dynamically.

IP (Internet Protocol): The Internet standard protocol that provides a common layer over dissimilar networks, used to move packets among the host computers and through gateways, if necessary.

IP Address: The numeric address of a computer connected to the Internet.

Ping (Packet Internet Gopher): A program designed to test the availability of a remote host.

Search Engine: This term refers to a program that helps the users in finding information in text-oriented databases.

URL (Uniform Resource Locator): The addressing scheme for information resources on the World Wide Web.

World Wide Web (WWW): A hypertext-based, distributed information system in which users may create, edit, and browse hypertext documents.

Archie: A program used to search files at FTP (File Transfer Protocol) sites.

Spam: Spamming is the process of sending the message to a large number of users, usually to advertise something.

Surfing: The process of looking casually on the internet.

ISP(Internet Service Provider): A company or organization which provides Internet Access.

3.2 INTRODUCTION

Vinton Gray "Vint" Cerf; (born, June 23, 1943) is an American computer scientist, who is recognized as one of "the fathers of the Internet", sharing this title with American computer scientist Bob Kahn.

Today, Internet is the necessity of every human being because Internet works as a hub where all the information resides and people can access the information of their own interest in an efficient manner at any time. Most probably, this is the biggest invention and fastest growing field of the current century. The concept of global village is now possible through Internet. In 1998, percentage of people using Internet in India was only 0.1% while it has grown up to 7% (of total

population) in the year 2010. India is among the ten fastest growing economic countries in the world and the third country in the world having most illiterate people. About 35% of **world's** illiterate population is Indian. Internet can be instrumental in overall growth of the country. Globally, there will be approximately 29% (of the total population) internet users by 2010. Internet is the interconnection between the computers cited at remote locations. It is a seamless network of fiber-optic cables, telephone lines or radio link (guided and unguided media) connecting hardware and software across the world using communication protocol to exchange information. You cannot estimate the size of the Internet. Internet is expanding at a phenomenal rate each year. Internet is being used by the people of various fields like academicians, researchers, journalists, educators, politicians, students, businessmen, and anyone, who has a computer connected to this network of networks through an Internet Service Provider (ISP). ISP is an organization which provides the Internet Access. There is no individual owner of the Internet and it is beyond the control of any individual.

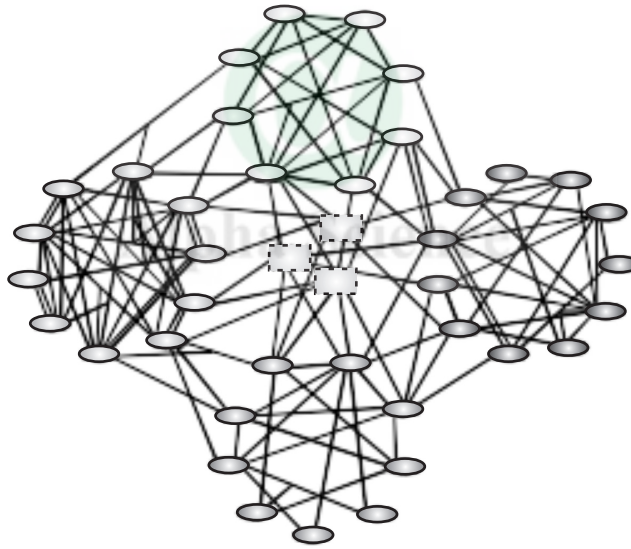


Figure 3.1

Every technology which we see today has its own history and has been grown over a period of time. Internet is no exception. The evolution of Internet is a step by step process in which many great scientists have given their contribution to bring Internet in its current shape. We are explaining the evolution of Internet in the chronological order as follows:

1. In 1962, J.C.R. Licklider, first introduced the concept of global network of computers. Further, Leonard Kleinrock of MIT and UCLA invented the theory of packet switching, which was the basis of Internet connections.

2. Lawrence Roberts of MIT connected a Massachusetts computer with a California computer in 1965 over dial-up telephone lines to show the feasibility of wide area networking, but also to find out that the telephone line's circuit switching was inadequate.
3. Roberts moved over to DARPA in 1966 and developed his plan for ARPANET.
4. The Internet has its root in the ARPANET system of the Advanced Research Project Agency of the US Department of defence. ARPANET was the first WAN and had only four sites.
5. Charley Kline at UCLA sent the first packets on ARPANET as he tried to connect to Stanford Research Institute on Oct 29, 1969.
6. E-mail was adapted for ARPANET by Ray Tomlinson of BBN in 1972. He singled out the '@' symbol from the available symbols on his teletype to link the username and address.
7. The telnet protocol, enabling the logging on to a remote computer, was published as a Request for Comments (RFC) in 1972.
8. The FTP protocol, enabling file transfers between Internet sites, was published as an RFC in 1973.
9. Ethernet, a protocol for many local networks, appeared in 1974, an outgrowth of Harvard student Bob Metcalfe's dissertation on "Packet Networks." Initially the dissertation was rejected by the University for not being analytical enough. It later got acceptance when he added some more equations to it.
10. The Unix to Unix Copy Protocol (UUCP) was invented in 1978 at Bell Labs.
11. Usenet was started in 1979 based on UUCP.
12. In 1986, the National Science Foundation funded NSFNet as a cross country 56 Kbps backbone for the Internet. The commands for e-mail, FTP, and telnet were standardized. It became a lot easier for non-technical people to learn to use the nets.
13. Other than library catalogs, the first effort to index the Internet was created in 1989, as Peter Deutsch and his crew at McGill University in Montreal, created an archive for FTP sites, which they named Archie. At the same time, Brewster Kahle, then at Thinking Machines, Corp. developed his Wide Area Information Server (WAIS).
14. The concept of web-based catalogs was introduced by Peter Scott of the University of Saskatchewan in 1990.
15. In 1991, Gopher and Spider was introduced to make user interface to the Internet at the University of Minnesota.

16. The creator of a web was a computer scientist named Tim Berners-lee in 1989 to establish communication between research scientists. He designed the web in such a manner that documents located on one computer on the internet could provide links to documents stored on the computers cited at remote locations. He created an application of the Standard Generalized Markup Language (SGML), a standard system to specify the document structure, and called it HTML. The concept of World Wide Web was introduced in 1991, which was based on hypertext.
17. Now the main issue was to create Graphical User Interface for the net users. In this process, the Browser named Mosaic was developed in 1993 by Marc Andreessen and his team at the National Center for Supercomputing Applications (NCSA). Later, Netscape Navigator was developed by Netscape Corp, which produced the most successful graphical type of browser and server until Microsoft developed its own Microsoft Internet Explorer.

A graphical presentation of internet evolution has been described in Figure 3.2.

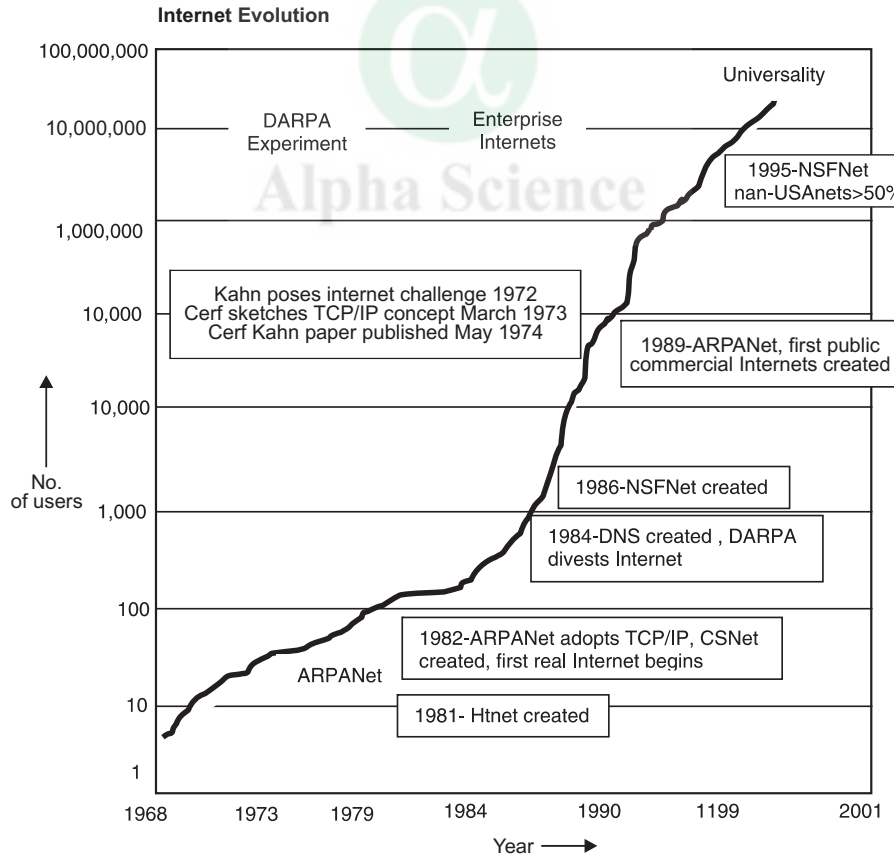


Figure 3.2

Whenever, we think about all the ways in which internet has influenced our lives, the ways in which it has made our lives better and more efficient, we find that it has greatly influenced our present lives; and there is no doubt about it. The maximum effect of the internet is actually on the daily life of the individuals because indeed there is no aspect of life which is not influenced by the internet and its uses.

Internet has surprised everyone by reducing the communication barrier between the worlds. We can imagine the time, around 20 years ago; when we used to write letters to our friends, and then wait for more than 14 days to get a reply. Today, sending emails is the fastest and most efficient way of communication. No matter how far your friend is living, it takes just seconds to send an email and the response is quite fast. The instant messenger has made it a lot better. Now you can even talk to your friends, see them through a web camera, when they are online. It is good way of making friends also. People are making a lot of friends all around the country on the net through Social Networking sites like Facebook, Twitter etc.

We can say, that Internet is a warehouse of information because of the availability of huge amount of information on the net. It is like an electronic library at your disposal. Just type in Google or any other Search Engine, what you are looking for and within seconds you will have the result. You can get a lot of information on a single topic with the help of a single click. Even virtual teaching and learning is also possible now.

Now, the Internet can also be used for banking, reservation purposes. You can transfer money from your account, view the balance report etc by just sitting at your home or by being present in any part of the world. Even Railway and Airline reservation can be done with the help of a click. These days there is no need to stand in queues and wait for your turn. Just open the site, click and book your ticket. Even shopping can be done by sitting at a desktop. Even though it may be unsafe to use credit cards on the net, but it is one of the most popular feature of the net. Although Internet is not very secure and these days Internet security is a very hot topic among the researchers.

Internet is also used for expanding your business by the help of a website. Now you can target a large variety of consumers from around the world for your business if you have a properly maintained website. Blogging is the trend of the new generation and with the help of your blog you can share your likes and dislikes with the world. Now internet can also be used as a TV, you can watch web cast of live matches or several other various events. Videos, music etc whatever you want, is all on the Internet. Now, you can even find your Life Partner on the net through various matrimonial sites.

As per our personal experiences, life has become drastically smooth and easy due to internet, In our time, while seeking for a job required several visits

to the offices and companies for interviews. These days all what is required is your detailed resume emailed to the company's HR division which assures saveral possibilities for getting the job instead of waiting and getting rejected at the end.

Internet can now be used to inquire about the flight timings and delays which make it easier to plan for the departure time.

Any piece of information regarding anything at all in our daily lives, may it be a cosmetic technique for the ladies or the men health problems, the cooking recipes for trying the new dishes or the home decoration tips, the information on the latest appliance or product you are going to buy or the search for the new house, it is all there ready for you on the internet.

Today, you can just log on the internet in your free time and talk to the people of similar interests, or watch movies play games, its all there on the internet.

We want to share another experience, while seeking for an admission of our kids, it requires several visits to the schools. Getting admissions in schools has become easier these days. Parents can easily fill up the online forms and submit them on the Internet since every school has its own website. Parents can see all the information (fee structure, Faculty members and their qualifications, sports activities, results, etc) about a particular school through the Internet. That's why people can take advantage of Internet, while admitting their children to the schools.

At the time of independence, the literacy rate was only 12% which has grown up to 68% (as per the survey) done in the year 2008. In general, the conditions of Government schools are not up to the mark in India. This is also the reality that most of the underprivileged and unemployed citizens send their children to the Government schools. Only in some of the states like Himachal Pradesh, Mizoram, Tamil Nadu and Kerala, the conditions of Govt schools are good. In such a scenario, Internet is instrumental for the improvement of educational standards in Govt. Schools.

Researchers can be benefited by downloading useful research information from the Internet, since almost all the reputed journals/books of all the streams are available on the net.

Government has made it mandatory that every organization must have its own website, where all the information must be uploaded online for the interested parties to ensure transparency and accountability in their day to day activities.

You can earn/lose a lot of money by doing transactions (on line Share trading) on share related websites (NSEindia.com, sharekhan.com, rediff.com, etc).

After so many advantages, as you know that nothing can be 100% in this real world, therefore the Internet has some bad effects also. Internet is a sort of

addiction for many people. Syndrome is indeed a matter of concern, with symptoms of "distraction". But it also depends on us, how we spend our time on the net.

To summarize, we can say that Internet gives us an extended social network, provides convenient recreational activities, it provides information on any topic you want to learn about. Some of the uses of Internet are given as follows:

- On-line communication
- Software Sharing
- Exchange of views on topic of common interest
- Posting of information of general interest
- Product promotion
  - Feed back about products
  - Customer support service
  - On-line journals and magazines
  - On-line shopping
  - World Wide Video Conferencing
  - Education and Training
  - Entertainment
  - Banking and Financial Transactions

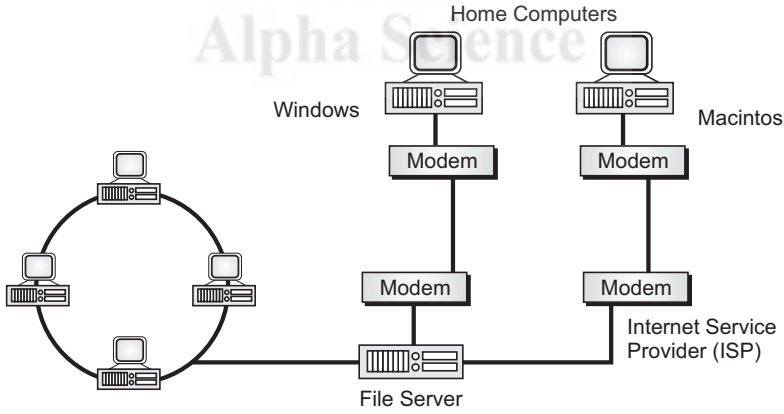


Figure 3.3

3.3 INTERNET PROTOCOL ADDRESSING

Every machine on Internet has its unique address assigned by the Network Information Center, NIC) called Internet Protocol address or IP address. There is a formal structure to the assignment of IP addresses. IP addresses are assigned by the Network Information Center (NIC) which is a central authority with the

responsibility of assigning network addresses. There are four parts of an IP address with total size of 32 bits. And each part size is 8 bit which means in decimal term ranging from 0 to 255 and each part is separated by a dot or period. The addressing scheme of machine is same as telephone numbers. Any telephone number consists of country code + exchange code+ number. Similarly, IP address consists of network address and a local number of machine. An IP address 122.134.212.20, where 20 is the machine address and 122.134.212 is the network address.

3.3.1 IP Address Classification

There are different IP addressing schemes used for different purposes. There are several classifications of IP addresses, depending upon the no. of hosts and the size of the network.

Class A (Large Network) Addresses

In this scheme, the first part is the network address and the last three is for the local host. Class A addresses always have bit 0, set to 0, and bits 1-7 are used as the network ID. Bits 8-31 are used as the host ID**;** 16,777,214 host per network. Due to this reason, class A networks are used by very large companies. The number of network available will be 126.

Class B (Medium Network) Addresses

In this scheme, the first two parts are the network address and the last two for the local host. Class B addresses always have bit 0 and 1 set to 10, bits 2-15 are used as the network ID. Bits 16-31 are used as the host ID. Means 65, 534, hosts are connected to one network. These networks are assigned to large companies and universities. The number of network available will be 16,384.

Class C (Small Network) Addresses

In this scheme, the first three parts are the network address and the last one is for the local host. Class C addresses always have bits 0-2 set to 110, bits 3-24 are used as the network ID. Bits 25-31 are used as the host ID. Class C network addresses are assigned to small companies and local Internet providers, which means 254 hosts are connected to one network. The number of network available will be 2,097,152.

In summary, a class A IP address is suitable when the internet consists of a small number of networks but each network consists of a large number of hosts. On the other extreme, class C addressing is suitable for internets with a very large number of networks, with a small number of hosts per network.

3.4 DOMAIN NAME SYSTEM

Domain Name System was invented by Paul Mockapetris in 1983 who wrote the first implementation. DNS is a textual addressing scheme.

Format: Host.subdomain.domain

Example: www.arsdcollege.net

In the above example, in www there is a computer name arsdcollege in .net domain.

On the Internet, every computer has an internal address which is a number that uniquely identifies each computer on the net. Since many computers are interconnected on the Internet, it is very difficult to remember the numeric address. Remembering a name is easier than numbers. To overcome this problem, the concept of DNS was introduced.

An Internet address has to be read right to left, and as you go along the domain it becomes progressively smaller. The rightmost part of the name is called its zone.

Name zones can be divided into the following categories:

- Three letter zones are set up by the type of an organization.
- Two letter zone names are organized geographically.

Three letter zone:

com	Commercial domains which are business.
edu	Educational institutes
net	This is for computers of network providers such as Internet
org	Standard organizations or non profit organizations
int	Organizations that have been established by international treaties.
gov	Governments
mil	Military

Two letter zone:

au	Australia	ca	Canada
in	India	tw	Taiwan
fi	Finland	us	United States
nz	New Zealand	uk	United Kingdom

3.5 UNIFORM RESOURCE LOCATOR (URL)

Uniform Resource Locator (URL) is used to address a document (or other data) on the World Wide Web. One of the most important aspect of the web is that Tim Berners-Lee devised a scalable way to locate all the resources of the net *i.e*. the URL. A full Web address like: http://ww.veersaindixit.wetpaint.com/myprofile.html follows these syntax rules:

scheme://host.domain:port/path/filename

The **scheme** defines the **type** of Internet service. The most common type is **http**.

The **domain** defines the Internet **domain name** like veersaindixit.wetpaint.com

The **host** defines the domain host. If omitted, the default host for http is **www**.

The port defines the **port number** at the host. The port number is normally omitted. The default port number for http is **80**.

The **path** defines a **path** (a sub directory) at the server. If the path is omitted, the resource (the document) must be located at the root directory of the web site.

The **filename** defines the name of a document. The default filename might be default. asp or index.html or something else depending on the settings of the Web server.

URL Schemes

Some examples of the most common schemes are as follows:

Schemes	Access
file	a file on your local PC
ftp	a file on a FTP server
http	a file on a World Wide Web Server
gopher	a file on a Gopher server
news	a Usenet newsgroup
telnet	a Telnet connection
WAIS	a file on a WAIS server

3.6 CONNECTIVITY TO THE INTERNET

Requirements to become the part of online Internet community are as follows:

- You require a computer with either a serial port for an external modem or an expansion slot for an internal modem.

- A modem is required. If the modem is external, an interface cable will be required to connect the modem with the computer.
- A dial-up telephone line is required.
- Communication software is required.
- An Internet Access account with an Internal Service Provider (ISP).

Now let us understand the difference between Internal modem and External modem:

Internal modem is plugged into an expansion slot inside the computer. Internal modem draws the power from the computer's power supply.

External modem sits on the table and is connected to the computer via cable. Light flash can be seen on the external modem as it communicates.

3.6.1 Connect a Single Computer to the Internet

To connect your computer to your Internet connection using a wired network

1. Contact an Internet service provider (ISP) and let them configure an Internet connection to your home.
2. Determine if your computer has a network adapter, in case the network adapter is not present, then add.
3. Shut down your computer by clicking Start, and then clicking Turn off Computer.

Figure 3.4

4. Connect the network cable supplied by your ISP to your network adapter.

Figure 3.5

5. Connect the other end of the network cable to your modem.

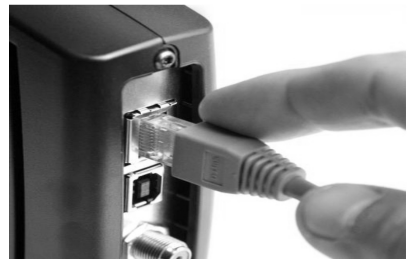


Figure 3.6

6. Start your computer and log on to Microsoft Windows. Windows will automatically detect that you are connected to the Internet.

3.6.2 Connect Multiple Computers to the Internet

To connect more than one computer to the Internet, you need a router. A router (which might also have a wireless networking built in) allows you to connect several computers to each other and to the Internet. The router sits between your computer and the modem provided by your ISP. You connect the router to the modem, and then connect all your computers to the router.

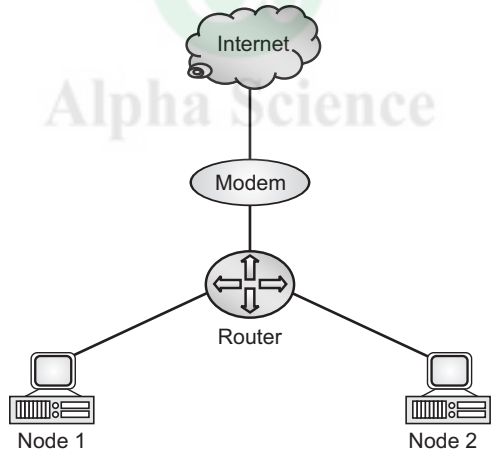


Figure 3.7

Steps to connect a router and multiple computers to the Internet

1. Contact an Internet service provider (ISP) and have them configure an Internet connection at your home.
2. Connect your router to your modem in one of the two following ways:
   - Currently, if you have a computer connected directly to your modem: Unplug the network cable from the back of your computer, and plug it into the port labeled Internet, WAN or WLAN on the back of your router.

- Currently, if you don't have your computer connected to the Internet: Plug one end of a network cable (included with your router) into your modem, and the other end of the network cable into the Internet, WAN or WLAN port on your router.

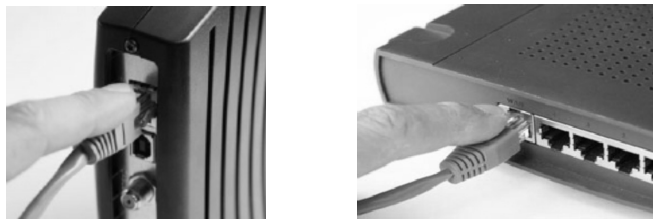


Figure 3.8

3. Plug in your router. After some time, the Internet, WAN or WLAN will light on; your router will light up, indicating that it has successfully connected to your modem.

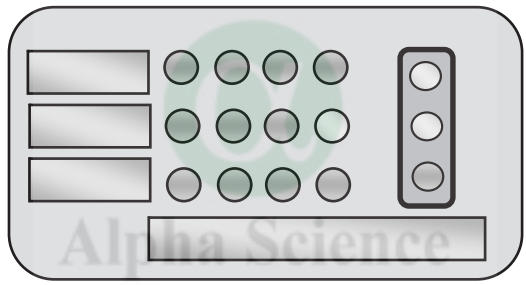


Figure 3.9

4. Determine whether your computer has a network adapter, add one, if necessary.
5. Shut down your computer by clicking Start and then clicking Turn off Computer.
6. Connect the network cable to your computer's network adapter.

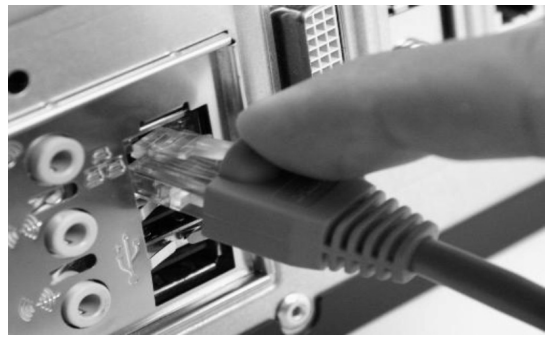


Figure 3.10

7. Connect the other end of the network cable to your network equipment.

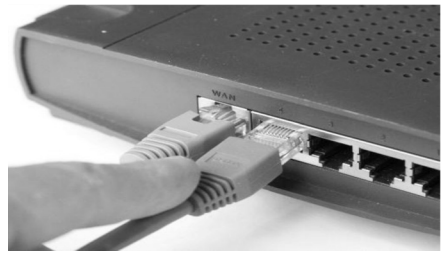


Figure 3.11

8. Start your computer and log on to Windows. Windows automatically detects that you are connected to the Internet.
9. Repeat steps 6, 7, and 8 for every computer.

Now you have your computers networked.

3.7 INTERNET APPLICATIONS

Now let's discuss some Internet applications which are given as follows:

Electronic Mail: Email is an Internet tool used to send messages to other Internet users.

Telnet: Offers remote log in to other computer systems on the Internet.

Finger Service: Allows you to ask information on system users.

Usenet: World Wide Bulletin Board System with thousands of discussion groups.

FTP (File Transfer Protocol): Allow users to copy files from one computer to another.

Talk: Allow two users to type real time messages to one another.

IRC (Internet Relay Chat): Allow multiple Internet users to type real time messages to one another.

Wais Servers: Provides access to database information available on the Internet.

World Wide Web: Hypertext based tool that allows retrieval of data, usually through graphical interfaces.

Electronic Magazines: Electronic subscriptions are available on Internet.

Mailing Lists: An organized system in which a group of people are sent message related to a particular topic.

Internet BBS's: A repository for messages and files usually related to a specific topic.

Games: Downloading off and online games are available via the Internet**.**

MUD (Multiple User Dimension): A service that allows Internet users to enter simulated environments. Through this service, we get feel of virtual reality.

3.8 WEB BROWSER

Browser is one of the most important elements of the web. Browser provides a platform to view web documents and access web-based services and applications. A browser fetches a page from a web server on a request. A browser is an application software program that allows users to see and navigate documents called web pages. There are two types of browsers**:**

Graphical: Text, images, audio, and video are retrievable through a graphical software program such as Netscape Navigator, Internet Explorer, Mozilla, Fire Fox or AOL. These browsers are available for both Windows-based and Macintosh computers. Navigation is accomplished by pointing and clicking with a mouse on highlighted words and graphics.

Text-only: *Lynx* is a browser that provides access to the Web showing only the text. To navigate, a person must use the arrow up and down keys to indicate links, and press the forward arrow (or Enter) key to follow the link. Pictures are translated by using any alternative text the creator has supplied. Unfortunately, many web page creators forget to include an alternative or Alt Tags for pictures. For more information, see the following web page, Albany Library's Guide to Using Lynx.

3.9 SEARCH ENGINES

A Search Engine is a program that can search the Web on a specific topic for the user. By typing a word or phrase (known as a keyword), the search engine will produce pages of links on that topic.

Example: The Google Search Engine

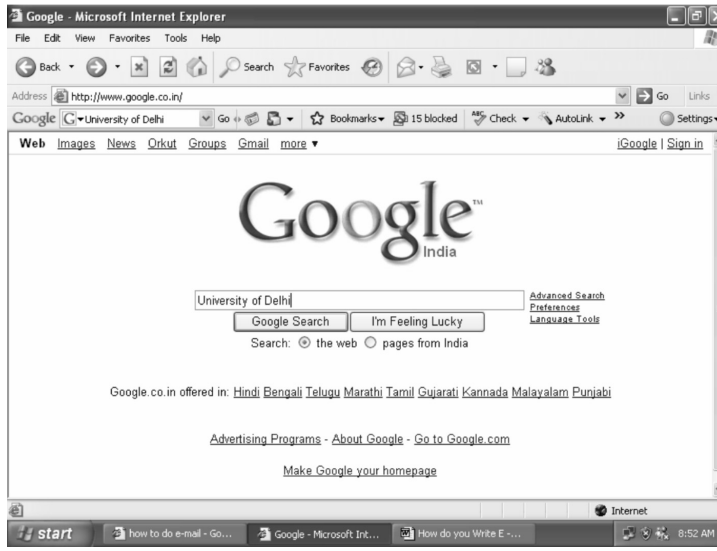


Figure 3.12

- An example of a search engine is the popular Google Search Engine. The homepage of this web site includes a small window, on which you can type a word or set of words on the topic you want the information about.
- Suppose you want some information about Delhi University on the Web. You can go to the Google site and enter your query in the entry field:

Figure 3.13

- Then, press on "Google Search" (other Search Engines will have slightly different forms). The search engine will respond by providing you the list of all the documents in its database that contain the word "University Of Delhi". Your browser will display a page with your keyword(s) and a list of documents in which the engine has found the desired word(s). For a Google search on "University Of Delhi", the results will look like:

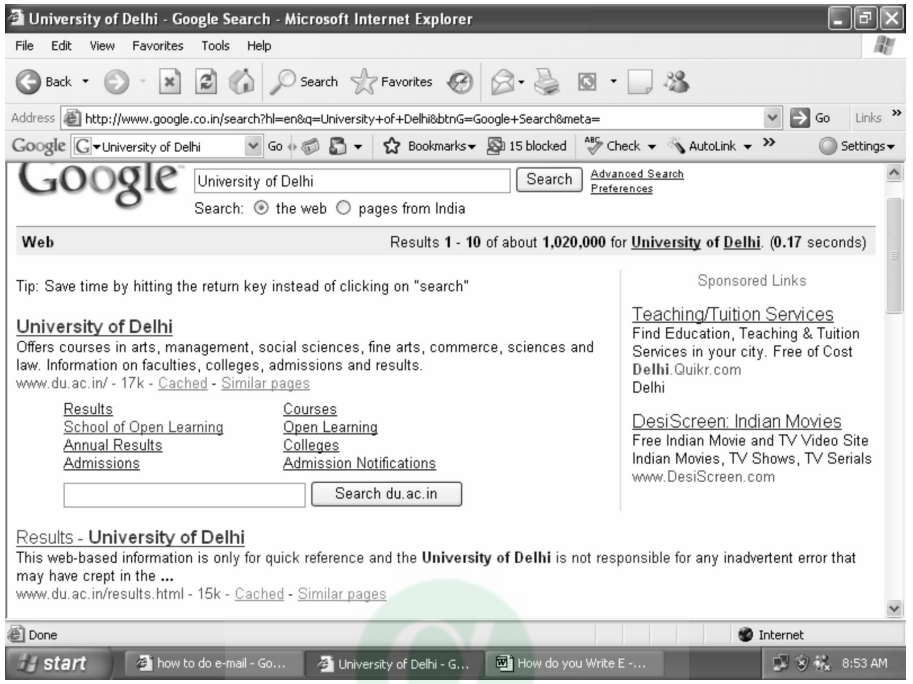


Figure 3.14

Search-refinement

Search engines can be used efficiently by **Search-refinement operators.** Search-refinement operators work with Google, and other search engines. Students should try the following eight operators, using search terms of their own choice, so that they have an understanding of how the operators work, both separately and in different combinations with each other.

Basic Search-refinement Operators

The search syntax is given below, inside the angle brackets (< >). This is to avoid the confusion with quotation marks, which are used to indicate "string" searches. The angle brackets should **not** be included when you use the search operators in Google (or other search engines).

1. "string" searching:

Example: <"Delhi University"> or <"Hyper exponential Distribution">

2. "wildcard" operator to search for strings with an 'unknown variable':

Example: <"Indian * universities"> would produce results for "Indian Agricultural University," "Indian Technical University," and so on. In turn, <"* exponential distribution">would produce "hyper exponential distribution," "hypo exponential distribution," and so on.

3. "plus" and "minus" Boolean Equivalents:

Example: <"MY" +HINDI "India" -NEPAL -PAKISTAN -BHUTAN> if one has searched just for the "MY" and "india" strings, who wants particularly to get references to MY HINDI, but does not wish to get results for pages which also discuss NEPAL, PAKISTAN and BHUTAN.

4. "site" Operator to Search Only Particular Sites or Domains:

Example: <"INDIAN" site:edu> or <"social security" site:gov>

Note: When using "site:" and similar operators, be sure that there is no space between the colon and the next word. If you accidentally put a space there, the search engines will think that "site:" is the word you desire.

5. "filetype" Operator to Search for Only Particular Types of Files [other than the default HTML], such as PDF, PPT, MP3, etc.

Example: <"web recommender system" filetype:ppt>.

For example, if searching for "web recommender system" in "edu" sites for "PPT" files, one will find many Powerpoint presentations on the web recommender system which university computer science teachers presents to their classes.

6. "define" Operator: This advanced search operator is useful to find the definitions of words, phrases, and acronyms.

Example: [define: blog] will show definitions for "Blog".

7. OR operator: This can be useful, if you are researching on a topic but are not sure which keywords will give, the information you need. It can be particularly handy in conjunction with the **site:** operator. Be sure to type 'OR' in capital letter.

Example: A Google search for <"a winter day" Shimla OR Dharamshala site: edu> will produce pages from educational-domain websites referring to both the "a winter day" string plus Shimla or Dharamshala.

8. "link" Operator: To check the number and type of other web pages which have linked to a particular page (or site). The query link:*URL* shows pages that point to that URL.

Example: To find pages that guides/enables to Google Guide's home page, enter:[link:www.googleguide.com]

Example: Find links to the Google home page and not on Google's own site.

[link:www.google.com-site:google.com]

Example: <link:www.uta.fi/FAST/GC/>or<link:www.uta.fi/~hh5678/hotfile.html>

Note: Works precisely only when the link from another page to the page you have identified has been programmed exactly the way you entered the search request.

General Search Engines

About.com: Commercial directory site contains a large collection of topics gathered by the company-certified subject specialists.

AllTheWeb: Returns results quickly from an extremely large database; which offers multimedia, Flash and FTP searches; also returns categorized topics to focus a on the search.

AltaVista: Searches Web sites and Usenet newsgroups with advanced Boolean and field search options. See also: Babel Fish, the AltaVista translation service.

AOL Search: Engine that defaults to AND logic and offers an Options template for easy search construction; has an option to view results by popularity.

Ask Jeeves: Submit questions in plain English and view the relevant sites.

Google: Ranks pages by the number of links from pages that has been ranked high by the service. Google offers a number of Services & Tools that are worth exploring.

LookSmart: Large collection of links to review the sites in thousands of categories.

Lycos: Emphasizes search results from the Open Directory and offers Web sites from the FAST Search index. The site includes its channel content with the results for searches on broad or popular terms.

Teoma: Return results in three sections: popularity-ranked Web pages based on the number of same-subject pages; suggested terms to refine a search; and link collections created by topic experts.

Yahoo!: One of the largest directories on the Internet but lacks reliable site evaluation, it is a subject directory site. This means it uses directories organized by subject which consist of links to Internet resources with annotations written below the web page creator.

Now let's discuss the directory and try to understand the difference between directory and the search engine:

A directory is built by someone looking at a web site to determine if it is worthy of being added to their database and if yes, then under what category it should fall. Regular search engines are not so discerning. As you might suspect, directories don't cover as much of the Web as a search engine, and there is a potential element of human bias in the categorization.

- Many directories have search tools to help you find specific items within each category like Looksmart and Yahoo.

3.10 METASEARCH ENGINES

Metasearch engine is a program which does your search in a number of search engines, and returns with some top websites from each search engine that responds to your search.

— You will get less search results than the same search performed using a keyword search engine because the time limit may prevent some individual engines from being included in any search you perform.

Note: So the search results may vary from search to search and from day to day, even though you may be typing the search exactly the same way each time.

The following metasearch engines are recommended:

Metacrawler, http://www.metacrawler.ca

Dogpile, http://www.dogpile.com/

3.11 INSTANT MESSAGING

An instant messaging service accepts instant messages from external sites and consists of the following points:

- It determines the user which the message should be delivered, and routes it accordingly.
- It accepts instant messages from internal hosts. It determines the destination system to which the message should be delivered, and routes it accordingly.
- Instant Messaging service can provide real in time conferencing, news and calendar alerts, and for offline users forwards the email message.
- The web server (or an application server with a web service embedded), downloads the Instant Messaging resources from a browser to the client.
- The resource files make up the client. Clients send messages to one another through a multiplexer, which forwards the messages to the Instant Messaging server.
- The directory server stores and retrieves local user and group delivery information such as preferences, location, and enables the multiplexer to route messages for this user.
- The Instant Messaging server receives a message and uses this information to determine where and how the message should be delivered. The directory server can also contain user information such as contact lists and subscriptions.
- Instant Messaging directly accesses Directory Server to verify user login name and passwords for mail clients that use Instant Messaging.

3.12 WAYS TO SPEEDUP DOWNLOAD TIME

We should always create web page keeping in mind the following guidelines which can speed up the download time:

- When there is an addition of graphic, audio and video files to a web site and if any user wants to access this website, then there will be a delay in downloading.
- Cut out unnecessary HTML elements, such as comments, font changes and table elements that are needed.
- Use Cascading Style Sheet to control appearance rather than changing fonts and appearance on every page.
- Always use correct width and height attributes in image tags.
- Resize and resample the graphics. Lower the quality settings of JPEG files.
- Use GIFs instead of JPGs when possible.
- Lower the resolution of a high resolution graphic.
- Display a low-resolution version of a graphic while downloading.
- Create thumbnail. Display alt text. Split content into more pages.
- Avoid unnecessary use of multimedia content like java applets, movies, images, animation and other elements.

3.13 BROWSING PATTERN

The user should not be a static watcher rather should be involved in clicking and entering his choices. In the cyber space, there are various categories of Internet users and the level of success is based on the level of user's satisfaction. The following might be the circumstances and expectations of the users:

1. Users do not enjoy getting lost, having system lockup or being tricked.
2. Users like to be able to click any where at any time and get what they want.
3. Users enjoy being in control and have easy navigation and multiple choices.
4. User demand fast loading pages and go elsewhere, if they do not get them.
5. User cannot see image maps or graphics due to technology limitations.
6. Users often dislike clumsy frames and bothersome drop down menus.
7. User likes hyper text links but hate entry tunnels like "Click here to enter".
8. Users often do not have 256 colors, therefore many graphics look bad in just 16 colours.

9. Users say they return to site that treat them well as a web customer.
10. Users prefer to click through multiple short pages than wait for long one.
11. Users love pictures and prefer being told how big they are, so they can decide whether to load them or not.
12. Users have little idea about how internet magic works but know instantly which sites they prefer and which they will bookmark for viewing later.
13. Users know the Internet is not Television, but still expect the web to grab their attention in engaging them the interactive way.
14. Cyber Users always enjoy being part of an online community, so the chat features which are included on to a site can triple the session lengths, double Kb per user and page impression downloads per session, bolster visit frequency by over 50%, and work a site's servers, a great deal harder.
15. Most web users are simply not Cookie Monsters and are turned off, if they are constantly being assailed with them.

3.14 NET ETIQUETTE

Net etiquette is a set of general guidelines for cyberspace behavior. Guidelines are given as follows:

- You don't have to be engaged in criminal activity to be careful. Any message you send could be saved or forwarded by its recipient because you have no control over where it goes.
- Always Netiquette mandates should do their best to act within the laws of society and cyberspace.
- First get a sense of people who are already doing their act. Then go ahead and participate.
- Generally, people have less time than ever today, precisely because they have so much information to absorb. Before you copy people on your messages, ask yourself whether they really need to know this particular information? If the answer is no, then don't waste their time. If the answer is "may be", think twice before you hit the send key.
- You must avoid offending anyone because everyone will know what you mean exactly.
- Share only the expert knowledge. Sharing your knowledge is fun. It's a long-time net tradition. And it makes the world a better place.
- Netiquette does prohibit the continuation of flame wars like series of angry letters, most of them from two or three people directed towards each other, which can dominate the tone and destroy the pleasant environment of a discussion group. It is unfair to the other members of the group. And while flame wars can initially be entertaining, it can hurt the sentiments of the people who aren't involved in them.

- Failing to respect other people's privacy is not just bad Netiquette, but it could also cost you your job.
- Knowing more than others or having more power than they do; does not give you the right to use or manipulate them. For example, sys-administrators should never read personal emails.
- If you decide to inform someone of his mistake, point it out courteously and if possible by personal email rather than in public. Always give people the benefit of doubt; assume that they just don't know any better. And never be highheaded or sanctimonious about it. Just as it's a law of nature that spelling flames always contain spelling errors, notes highlighting the Netiquette violations are often examples of poor Netiquette.

3.15 VIDEO CONFERENCING

People at different locations want to talk to each other at the same time, sometimes even sharing the computer applications for a group; Video conferencing offers possibilities for schools, colleges, and libraries to use these systems for a variety of purposes:

- formal instruction (courses, lessons, and tutoring)
- connection with guest speakers and experts
- multi-school project collaboration
- professional activities
- Community events.

Placing a video call is similar as to place a telephone call. Once you connect, you see the other person in color video on a TV screen and can transfer files or collaborate via options such as document sharing, etc.

3.15.1 Basic Video Conferencing Technology

- Compressed video systems allow a larger audience to experience the benefits of high-quality videoconferencing at a reasonable cost.
- A video conferencing system requires an audio visual equipment, which includes a monitor, camera, microphone, speaker, and the means of transmission.
- Internet-based connection, which are used by webcams, have to share bandwidth with other Internet data, does not produce smooth audio and video.
- The solution to this is a compressed video system on a dedicated bandwidth which provides smooth audio and video.

Note: The compressed video conferencing may be transmitted via an ISDN (Integrated Services Digital Network) line or over IP (Internet Protocol) lines. It is a cheaper solution for high-quality video conferencing.

3.15.2 Connecting

The most significant distinction among video conferencing systems is the method of transmission.

Reason: Transmission is important because two systems cannot connect if they are using different transmission methods.

Video conferences can be transmitted via two protocols:

- H.320 - ISDN (phone)
- H.323 - IP (Internet) lines.

In the past, most videoconferencing used ISDN lines; while these days, many people are using IP connections.

Reason: Cost savings.

If there is a connection between two units using different transmission methods, then a bridge must be used which will handle these mixed protocols.

ISDN call vs IP call:

- In an ISDN call, bandwidth is dedicated to only one video conference, while in an IP call; bandwidth may be used to transmit for multiple uses.
- ISDN calls can be very costly since you may be making a long distance call and in that case, long distance phone line charges apply, and ISDN lines take up 6 phone lines. Connections around the world has an average of 384 kbps.

Advantages:

- Educators from around the country say video conferencing technology impacts the student learning in a number of ways.
- Teaching professionals have observed the following:
  - Two-way videoconferencing increase students' motivation.
  - Improve communication and presentation skills.
  - A virtual field trip increases the depth of learning and provides a forum for a greater interaction with the outside world.
- A video conference can improve students' memory preservation by appealing to a variety of different learning styles by including the following:
  - Varied media such as video and audio clips.
  - Graphics, animation and computer applications.
- The visual connection and interaction among participants enhance the understanding and allow both the content providers and the students to feel connected to one another. That connection leaves a distinct impression on the students who have the opportunity to go on a virtual field trip.

3.16 WEB-BASED E-MAIL

"Web mail (or **Web-based e-mail**) can be defined as an e-mail service intended to be primarily accessed via a web browser, as opposed by an e-mail client, such as Microsoft Outlook, Mozilla's Thunderbird, or Apple Inc.'s Mail. The most popular web mail providers are Gmail, Yahoo! Mail, Hotmail and AOL."

Advantage of Web-based e-mail over Application-based e-mail:

A user has the ability to access their inbox from any Internet-connected computer around the world. However, the need for Internet access is also a shortcoming when one cannot access old messages if he/she is not connected to the Internet.

Hotmail was one of the most popular web-based e-mail which introduced its service in 1997. After the Hotmail's success, Google introduced Gmail in 2004 and sparked a period of rapid development in webmail.

WebMail was the first webmail software and was developed in Perl by Luca Manunza while working at CRS4, in Sardinia.

Some existing software packages allow the organizations to offer e-mail through the web for their associates. Some solutions are open source software like SquirrelMail, BlueMamba, RoundCube and IlohaMail, while others are closed source like the Outlook Web Access module for Microsoft Exchange.

3.17 WIRELESS TECHNOLOGIES

During recent years, the market for wireless communications has enjoyed tremendous growth. Wireless technology is capable of reaching virtually every location on the face of the earth. No longer bound by the harnesses of wired networks, people will be able to access and share information on a global scale nearly anywhere. An article written by Seema Singh in the month of September 2006 issue of spectrum entitled "India connects to Wireless Internet" mentions the same. In this article it was said that, WiMax (the IEEE standard for wide area wireless broadband connectivity) offers India the prospect of largely skipping connection to the Internet by DSL links and cable modems that are standardised in so many places. According to Arogya Swami Paulraj, chemical Engineer at Stanford University (in California), WiMax offers the best answer to last-mile broadband connectivity in a country like India. Videsh Sanchar Nigam Limited, has deployed WiMax infrastructure in 65 Indian cities, using the equipment made by Aperto Networks (A leading developer of WiMax base stations), in Milpitas, California. Various companies like Motorola, Intel, Aperto, Alcatel, Beceem Communications (key contributer to the IEEE 802.16e standard), NTT DoCoMo (developing IEEE 802.16e chip sets in Bangalore) etc. are trying to deliver low cost Wimax enabling devices to consumers in

India on the priority basis. India is at an edge as compared to the rest of the world in devising WiMax equipment and developing markets for it, but unavailability of appropriate equipment requiring bandwidth is the main problem. (For WiMax, 3.5 GHz is approved, while India's 3.5 GHz band is locked for Defense and space sectors and some bands in 2.5 GHz have been earmarked for third generation cell phones). Indian frequency regulators should synchronize properly with global trends.

Types of Wireless

- Wireless Local Area Networks
- 802.11b (Wireless Ethernet)
- 802.11a, HiperLAN ii, and homeRF
- Fixed Broadband Wireless Multiservice Wide Area Networks
- Mobile Wireless
- Wireless Personal Area Networks
- Based on Bluetooth technologies: Bluetooth in its most basic form is a cable replacement. Where cable now connects many devices, a wireless Bluetooth connection will provide low cost wireless communications and networking between PCs, mobile phones and other devices. Bluetooth is a RF (radio frequency) technology for short-range wireless networking.
- WiMax, the IEEE standard for wide area wireless broadband connectivity.

WAP

WAP is wireless application protocol which allows web-like content to be sent to mobile devices. WML is the language of WAP. WML stands for wireless markup language. WML produces static pages. WML is a fairly bare-bones programming language. In 1999, the first version of WML was introduced. WML has its own version of JavaScript, called WMLScript.

VRML (Virtual Reality Modeling Language)

A language used to create 3-D interactive Web Graphics, similar to those found in some video games, where a user can move around within a graphic image and interact with the objects.

ISP (Internet Service Provider)

Internet Service Provider are Private/Public/Government companies which provides internet connectivity to different companies or organizations. And these companies or organizations provide Internet access, usually for free. Generally we use modem to connect our ISP. Examples of Internet Service Providers are Dish net, MTNL, VSNL, etc.

SUMMARY

In this chapter we have discussed the various aspects of internet technologies. Topics are presented in simplified manner for the students.

PROBLEMS

1. What are the different services provided by the Internet?
2. Compare the FTP and TELNET utility.
3. Why do we use newsgroup services?
4. What is a spamming in Internet?
5. Name three search engines.
6. How will you sharpen the search?
7. What does a user expects to find on a web site.
8. What are the uses of Internet?
9. What are the different ways to increase the of speed download time?
10. What is Wireless technology. What are the different types of Wireless?
11. What are the requirements, to become a part of exciting online internet community?
12. What is URL?
13. What is DNS?
14. Explain the Internet address?

REFERENCES AND RELATED LINKS

[1] K. Pahlavan and P. Krishnamurthy, Principles of Wireless Networks, A unified Approach, Pearson Education Asia, 2002.
[2] P. Nicopolitidis, M. S. Obaidat, G. I. Papadimitriou and A. S. Pomportis, Wireless Networks, John Wiley and Sons, Ltd. 2003.
[3] S. Haykin, Digital Communications, John Wiley & Sons, Inc., 2005.
[4] P.C. Gupta, Data Communications and Computer Networks, Prentice-Hall of India, 2006.
[5] V. S. Dixit "A Handbook on Internet, Academic Publishers, 2006.
[6] http://www.geardownload.com/.

Hyper Text Markup Language

4
CHAPTER

Born	:	08, June 1955 London, UK.
Known For	:	Inventing the World Wide Web
Title	:	Professor Knight


Tim Berners-Lee

Introduction

The World Wide Web or the web is one of the most popular services provided by the internet. The web is a collection of all browsers, servers, files and browser-accessible services available through the Internet. The creator of a web was a computer scientist named Tim Berners-lee. In the year 1989, he established a communication between research scientists. He designed the web in a manner that documents located on one computer on the internet could provide links to documents stored on computers cited at remote locations. He created an application for Standard Generalized Markup Language (SGML), a standard system to specify the document structure, and called it HTML. HTML reduces the complexity of SGML in a great manner to enhance transmission on the Internet. The first working draft of HTML included various elements such as titles, headings, paragraphs, and lists. HTML was intended for simple document structure. The great idea given by Berners-Lee for the use of hypertext to link information on related topics on the Internet. Hypertext is a non-linear way (You can jump from one related topic to another, quickly find the information of your interest) of organizing information. The early internet consisted of only text and binary files, and when new types of media were introduced, a new term came into the picture called hypermedia, which enabled the linking of different types of media on the Internet. A markup language is a structured language which helps us in the identification of common sections of a document

such as headings, paragraphs, lists etc. For example, HTML markup elements indicate how the document sections display in a browser.

Browser is one of the most important elements of the web. Browser provides the capability to view web documents and access web-based services and applications. To increase the efficiency of document retrieval, browser uses a caching technique. A copy of each document or image that a user sees on the local hard disk is maintained by the browser. Whenever a document is required, the browser checks the cache before requesting the document from a server on the network. If the cache contains the document, the browser obtains the copy from the cache without using the network. Performance of the document accessing will improve by keeping documents in a cache. Users can control the length of time by keeping the documents in the cache.

Definition:

"Web browser is an application software through which the web pages are executed".

Microsoft's internet explorer, Netscape's Navigator, Mosaic, Mozilla, FireFox, etc. are the example of browsers".

Today web browsers extend Graphical User interface features with multimedia capabilities and with browser programming languages. Browser programming languages can be characterized as following:

4.1 CLIENT-SIDE SCRIPTING

Client-side scripting are the programs on the web that are executed at client-side by the web browser. Web authors write client-side scripts in languages such as JavaScript, VB Script, Dynamic HTML etc. Client side scripts are often embedded within a HTML document, but may also be contained in a separate file, which is in reference by the document.

In a client-side scripting, the necessary files are sent to the user's computer by the web server on request in which they reside. The user's web browser executes the script and, then displays the document. If you see the file that contains the script, users may be able to see its source code. Client-side scripts have a greater access to the information available on the client's browser. Due to security restrictions, client-side scripts cannot be allowed to access the client's computer beyond the browser application and this can be done by using explicit techniques like Active X controls.

User can see the script's source code. See the following information about the client-side scripting:

Language	:	Hyper Text Markup Language
Scripting Language	:	Dynamic HTML, JavaScript VB script.
Data Handling Language	:	Extensible Markup Language (XML)
Applications	:	Active X controls, Applets, Plug-ins

4.2 SERVER–SIDE SCRIPTING

It is a web server technology, in which user's request is fulfilled by running a script directly on the web server. It generates dynamic HTML pages. Server-side scripting is used to create interactive web sites that interface to databases. Through server-side scripting, we can highly customize the response based on the client's requirements, access rights or queries into databases. Server-side scripts are written in languages like PHP, Perl and VB-scripts, which are executed by web servers namely (Apache, IIS (Internet Information Server), Netscape Enterprise, etc.) when the client requests a document. Server-side scripts produce an output in a format which is understandable by the web browsers, which is then sent to the client's computer. The user cannot see the script's source code. Server side scripts have greater access to the information and function available on the servers.

There are number of server side technologies which execute various programs on the web server. They are:

- Common Gateway Interface (CGI)
- Server Side Java Scripts (SSJS)
- Personal Home Page (PHP)
- Servlet
- Active Server Pages (ASP)
- Java Server Pages (JSP)
- ASP.net

Java Server Pages (JSP) technology provides a simplified and quick method for creating web pages that display dynamically generated content. JSP technology has been designed to build web based applications that run on variety of web servers, application servers, browsers and development tools.

The comparison of various popular server side technologies with JSP are given below:

CGI Vs JSP

JSP maintains the state on the server between different requests where CGI's each request is treated as a new one. JSP does not have to be loaded each time, once it has been initiated. JSP runs in a ready loaded JVM as an extension to the web server.

ASP Vs JSP

JSP and ASP are fairly similar in the functionality that they provide. JSP may have slightly higher learning curve. Both allow embedded code in a HTML page, session variables and database access and manipulation. ASP is mostly found on Microsoft platforms i.e. Windows NT. JSP can operate on any platform that conforms to J2EE specification. JSP allows component reuse by using JavaBeans and EJBs. ASP provides the use of COM (Component Object Model) / ActiveX controls

ASP.NET Vs JSP

ASP.NET is based on the Microsoft .NET framework. The .NET framework allows applications to be developed using different programming languages such as Visual Basic, C# and JavaScript. JSP and Java still has an advantage, that it is supported on many different platforms. The Java community has many years of experience in designing and developing Enterprise quality scalable applications. It does not mean that ASP.NET is bad, actually it is quite an improvement over the old ASP code.

Servlet Vs JSP

Servlet and JSP both are written in Java. All that is possible through JSP and can also be done by using Servlet. However, JSP provides a more distinct division between presentation and logic and is also easy to write. Furthermore, Servlet is lengthy as it needs a lot of println statements.

We will emphasize on the topics JAVASCRIPT and ASP in Chapter 5.

Note:

- "Web browser is an Application software through which the web pages are being executed".
- "Active-X controls are user defined controls which can be executed dynamically".
- Applet is a program which can be executed by a browser. Applet programs are developed using Java programming language. You can create dynamic web pages using applets.

Web documents can be categorized as follows:

- **Static:** This type of web document resides in a file that is associated with a web server. The contents of static web document do not change; each request for a static document gets in exactly the same response. Static document exists in a predefined form because the author of this type of document determines the contents at the time of document creation.
- **Dynamic:** This type of document is created by a Web Server at the moment browser's request for the document. On the arrival of a request, the Web Server runs an application program that creates the dynamic document. In response to the browser that requested the document, the server returns the output of the program. The contents of such type of document vary, as per the request made and for each request a fresh document is created.
- **Active:** When browser requests an active document, the server returns a copy of the program that the browser must run on the client side. The active document program can interact with the user and change the display continuously during the run time. The modification in Active document will be continued as long as running of the program will be allowed by the user [6].

In the remaining part of this chapter, we will emphasize on HTML (Hyper Text Markup Language).

4.3 ABOUT HTML

Tim Berners-Lee is the creator of HTML. He developed this language at a European Laboratory for particle Physics CERN in 1989. Since then, it has evolved through several major revisions. Each revision adds new tags that increases the expressive power and applicability of the language. HTML is the primary format used on the World Wide Web. HTML can display web pages with a wide range of colors, shapes and objects.HTML has following properties:

- HTML is an easy language to learn.
- HTML is used to create a web page.
- HTML is a scripting non procedural (4th generation) language.
- HTML is a non case sensitive language.
- HTML is a way of writing layout information within the documents.
- HTML document is a plain text file that contains only text.
- HTML file will be opened by a web browser like Microsoft Internet Explorer, Netscape Navigator, Fire Fox, Mozilla, etc.

- HTML code will be executed by web browser and perform the operations (like changing the layout, inserting the images or creating links to other pages) accordingly and produce the result.
- HTML code can be written in the simple text editors such as Front page, Dream weaver and note pad.
- HTML contains tags with formatted text using different fonts, size, color and so on. Some tags point to the images to include in the display as well as to define links to other web pages.
- All HTML tags are enclosed in an angular bracket like <>.When you will write any tag, then please ensure that it is enclosed by writing the tag followed by forward slash within the angular bracket like</ >.
- You can write any HTML code using any editor.
- All HTML documents have the same structure.
- In any basic HTML document, there are eight tags (<HTML>, <HEAD>, <TITLE>, </TITLE>, </HEAD>, <BODY>, </BODY>, </HTML>) and are always same.
- <! —— Text ——> is used for the comment.

Note:

Tags: HTML contains tags that define how text is to be formatted on the display.

CERN: Conseil European pours la Recherché Nucleaire, The European laboratory for particle Physics located in Geneva, Switzerland.

4.4 STRUCTURE OF HTML DOCUMENTS

```
<HTML>
<HEAD>
   <TITLE>
```

This section is used for title and technical information of the web page.

```
   </TITLE>
</HEAD>
<BODY>
```

This section will contain all the information that you want to show on the web page.

```
</BODY>
</HTML>
```

Every web page consists of a head and a body. This is true for all normal web pages. The head is used for text and tags that do not show directly on the page. The body is used for text and tags that are shown directly on the page. All

web pages have a <HTML>tag at the beginning and at the end, telling the browser where the document starts and where it ends. The <TITLE> and </TITLE> encapsulate the title of your page. The title is always shown on the top of the browser window when the page is being loaded.

For creating the web page, first open the editor, then type the HTML code, and save the document in editor and put an .html or .htm extension to the file.

Example: A piece of text as it appears on the screen:

Hello Welcome to HTML

The HTML code for the above example is as follows:

```
<b> Hello Welcome To HTML</b>
```

In the above example, the start tag indicates that text should be written in bold. The corresponding end tag indicates that the browser should stop writing text in bold.

4.5 SIX LEVEL HEADINGS

HTML provides six level headings. The font size of first level header will be the highest and the size of sixth level header will be the lowest. Let us see with an example: Firstly, select a text editor like note pad or word pad and open it. After that, write the following code:


```
<HTML>
<HEAD>
<TITLE> Demonstration of six level headings</TITLE>
</HEAD>
<BODY>
<H1> ARSD COLLEGE</H1>
<H2> ARSD COLLEGE</H2>
<H3> ARSD COLLEGE</H3>
<H4> ARSD COLLEGE</H4>
<H5> ARSD COLLEGE</H5>
<H6> ARSD COLLEGE</H6>
</BODY>
</HTML>
```

Now, click the file menu and save the page with extension ".html".

Naming convention is used when we are giving a name to the file: Please follow the following naming conventions:

1. No space between the characters of the name.
2. File name should be of eight characters or less (This is not an essential requirement for some operating systems).

3. Generally, keep the file names in the lower case.
4. All the file names must end with the extension of .htm or .html.

You should keep all four naming conventions into your mind at the time of naming the file. If you want to change the file type, then the following procedure will be followed:

Pull down the File menu and click on the Save as file sub menu. A dialog box will be opened. Select only the, text or text document. Review your saving choices before checking on the Save button. After saving the document, you will see the name of the document at the top of the editor.

The output is given below:

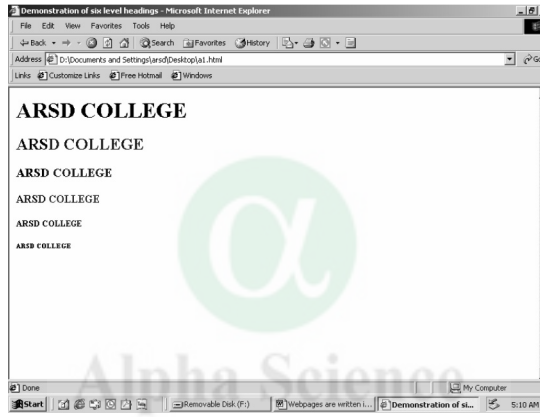


Figure 4.1

4.6 ENTER TEXT ON YOUR PAGES

To enter a text on your pages, simply enter the text. If you do not specify any attributes for text, it will use the default size, font etc. of the browser. Browsers can only show fonts available on the PC (Personal Computer). To specify the overall font for your page, add the <basefont> tag at the beginning of the <body> section.

See the following code:

```
<html>
<head>
<title>my first page</title>
</head>

<body>
```

```
<basefont face="arial, verdana, courier" size="6" color="red">
        This is my first page.<br>
        texts looks good<br>
        hello arsd.<br>
</body>
</html>
```

The color attribute selects the desired color for your text. The face attribute selects the desired font.
 tag is used to break the line.

The size attribute specifies the desired size between 1 (smallest) and 7 (biggest).The <font> tags will change the font.

Below is the output of the above html code:

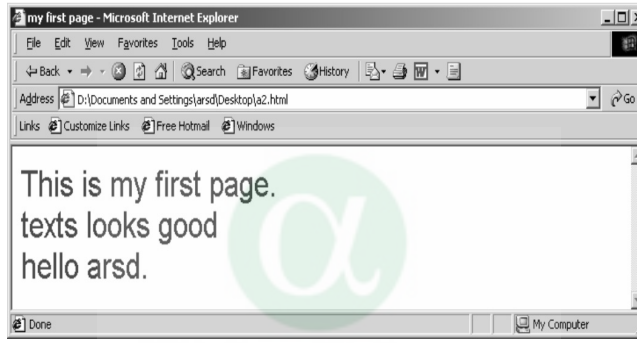


Figure 4.2

4.7 CREATION OF LINKS

You will use an anchor tag to create links. The Anchor tag is divided into three parts. The first part is the beginning of the anchor tag; second part is the text or images that the user would click on and third part is the end of anchor tag.

Syntax:

```
<a href="name and location of document"> Text that appears as a link </a>
```

Let's see the following html code:

```
<HTML>
<HEAD>
<BODY>
<HR>
<H3> ARSD COLLEGE</H3>
<B>THE DETAILS</B><BR> ABOUT THE COLLEGE<BR>
<HR> <A HREF=HTTP://WWW.ARSDCOLLEGE.NET> GO TO COLLEGE WEB SITE</A>
</BODY>
</HTML>
```

<hr> tag is used to insert a horizontal line in the web page.
The Output of the above html code is as follows:

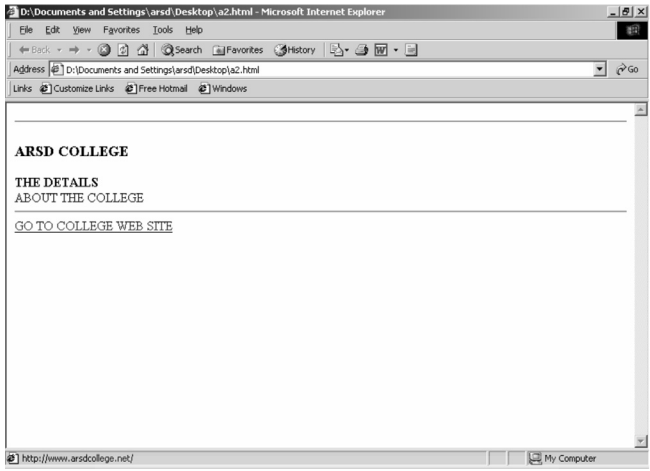


Figure 4.3

When you click on the link colored in blue, the following site will be open:

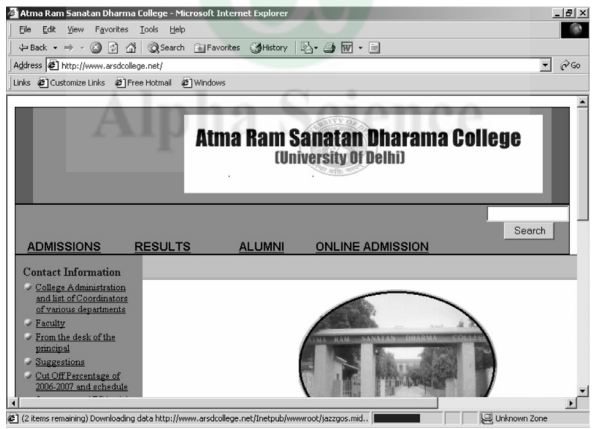


Figure 4.4

4.8 SOME IMPORTANT TEXT FORMATTING TAGS

`<b>text</b>`	writes text as bold.
`<i>text</i>`	writes text in italics.
`<u>text</u>`	writes underlined text.
`<sub>text</sub>`	used to create subscript.
`<sup>text</sup>`	used to create super scripts.
`<strike>text</strike>`	strikes a line through the text.

```
<tt>text</tt>                    writes text as on classic
                                 typewriter.
<pre>text</pre>                    writes text as it is including
                                 spaces.
<em> text</em>                   display text in italics.
<strong>text</strong>            display text in bold.
<big>text</big>                  increase the size by one.
<small>text</small>              decrease the size by one.
<font size="1">                  text</font> display text in
                                 smallest font size. (8 pt).
<font size="7">                  display text in biggest font
                                 size. (36 pt).
<BLINK>text text text</BLINK> used to blink a given text.
```

Note:

The <blink> tag is only supported by Netscape browsers.

The <small> and <big> tags are special in the sense that they can be repeated.

If you want to increase the font size with a factor of two, then you can do it with the help of the following example:

```
<HTML>
<HEAD>
<BODY>
dixit dixit dixit <big><big>whatever</big></big> dixit dixit
dixit
</BODY>
</HTML>
```


The output is given below:

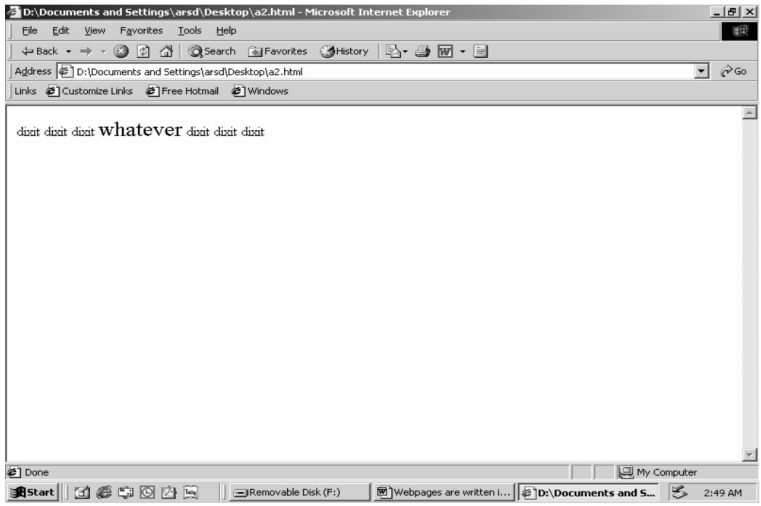


Figure 4.5

4.9 TAGS FOR CONTROLLING THE LAYOUT

text	Adds A paragraph break after the text (2 line breaks).
text	Left justify text in paragraph.
text	Center text in paragraph.
text	Right justify text in paragraph.
text 	Adds a single line break at the place of a tag.
<nobr>text</nobr>	Turns off automatic line breaks.
text<wbr>	Allows the browser to insert a line break exactly at this point and it is called a word break.
text	Center text.
text	Center text.
text	Left justify text.

4.10 BULLETED AND NUMBERED LIST

To create a bulleted list you are required to add a <ul> and a tag (unordered list) at the beginning and end of the list.

Numbered lists have <ol> tags (ordered list). To separate single list items use <li> and tags.

Let's see the uses of above tags in the following html source code:

Syntax:

```
<ul>
<li>text</li>
<li>text</li>
<li>text</li>
</ul>
```

The code is as follows:

```
<HTML>
<HEAD>
<BODY>
<ul type="man">
<li>This is one man</li>
<li>This is another man</li>
<li>And this is the last man</li>
</ul>
```

```
<ul type="woman">
<li>This is one woman</li>
<li>This is another woman</li>
<li>And this is the last woman</li>
</ul>
<ul type="child">
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ul>
</BODY>
</HTML>
```

The output is given below:

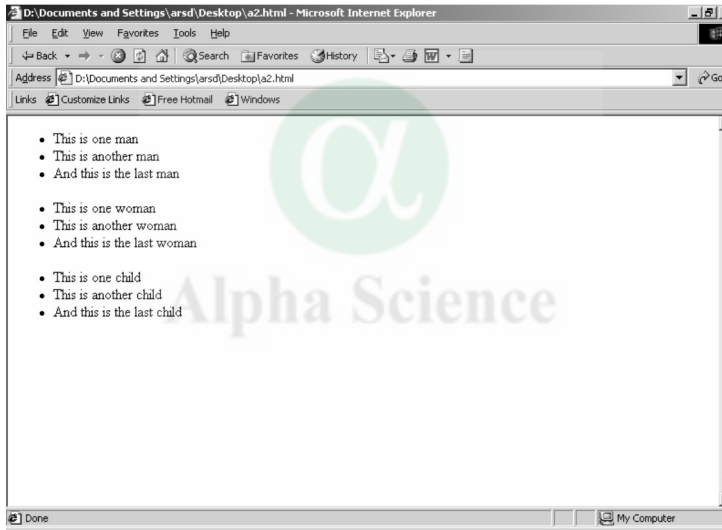


Figure 4.6

Now let us see another example to demonstrate the number options:

Syntax:

```
<ol>
<li>text</li>
<li>text</li>
<li>text</li>
</ol>
```

The html source code is as follows:

```
<HTML>
```

```
<HEAD>
<BODY>
<ol type=”A”>
<li>This is one man</li>
<li>This is another man</li>
<li>And this is the last man</li>
</ol>
<ol type=”a”>
<li>This is one woman</li>
<li>This is another woman</li>
<li>And this is the last woman</li>
</ol>
<ol start=”child”>
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ol>
<ol type=”I”>
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ol>
<ol type=”i”>
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ol>
<ol type=”i” start= “7”>
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ol>
<ol type=”1">
<li>This is one child</li>
<li>This is another child</li>
<li>And this is the last child</li>
</ol>

</BODY>
</HTML>
```

The output is given below:

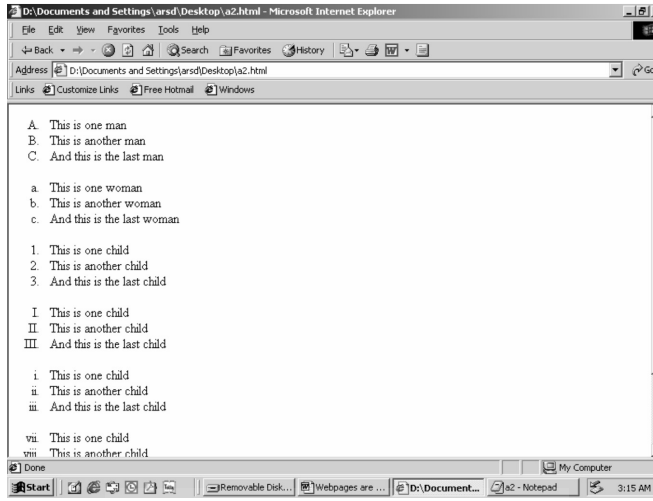


Figure 4.7

4.11 INSERTION OF IMAGES IN WEBPAGES

Now let us demonstrate how to insert images in web pages. You can include an image into your web document using IMG tag.

Now see the following html code:

```
<html>
<head>
</head>
<body>
<img src="c:\windows/cloud.gif" alt="cloud picture">
<p>
<img src="c:\windows/content.gif" alt="content picture">
<p>
<a href="http://www.arsdcollege.net" alt = "College web site"><img src="c:\windows/cloud.gif"></a>
</body>
</html>
```

The output of the above html code is as follows:

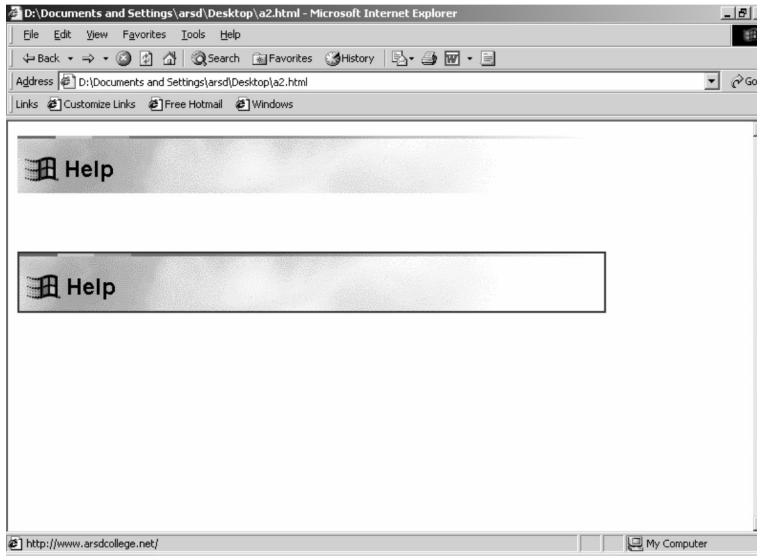


Figure 4.8

ALT tag is used to display the text related to the image. Another advantage of ALT tag is that, if the image is not displayed, then the name related with that image can be displayed. <p> tag in the above example is used for paragraph.

You should always add alternative texts to your images, so that the users can get an idea about the image, before it is loaded. This becomes particularly important if the image is a link.

Image Map

The <map> tag is used to define a client-side image-map. An image-map is an image with clickable areas.

The name attribute is required in the map element. This attribute is associated with the <img>'s usemap attribute which creates a relationship between the image and the map.

The map element contains a number of area elements that defines the clickable areas in the image map. It is made up of the actual image that is embedded within the <img>-Tag in HTML code. The image gets the attribute usemap simultaneously.

The imagemap comprises of <map>-Tag and in the detail of the <area>-tags which defines the fields with the hyperlinks. These are similar to the <a>-Tag defining which the URL should be opened.

Syntax:

```
<img src="image name" alt="alternative text" usemap="#mapname"
/>
<mapname="mapname">
 <area shape="shape name" coords="" href="file name" />
</map>
```

Let's see the following code which defines a rectangular area (0, 0, 67, 398) with a link to a webpage:

Example:

```
<html>
<body>
   <img src="C:\Documents and Settings\user\My Documents\My

  Pictures\Picture\Picture  023.jpg"  width=300  height=400
alt="Hello"
 usemap="#x" >
 <map name="x">
       <area shape="rectangle" coords="0, 0, 67, 398"
 href="www.veersaindixit.wetpaint.com" >
 </map>
</body>
</html>
```

OUTPUT:

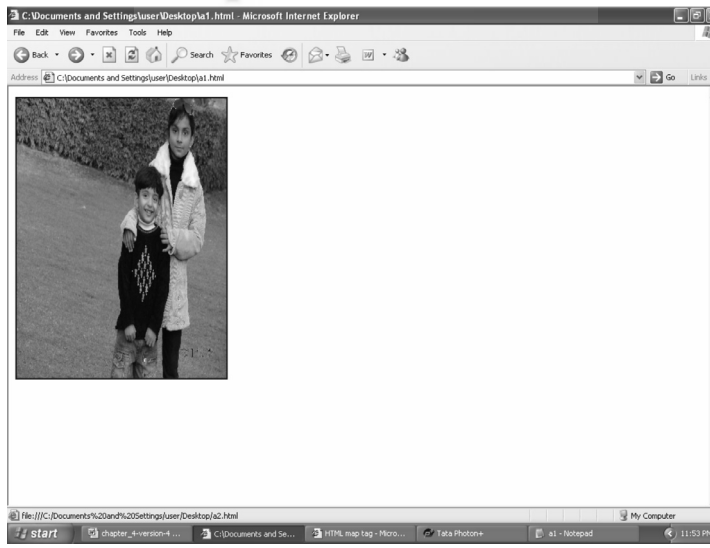


Figure 4.9

After clicking on the rectangular area, the following link will open:

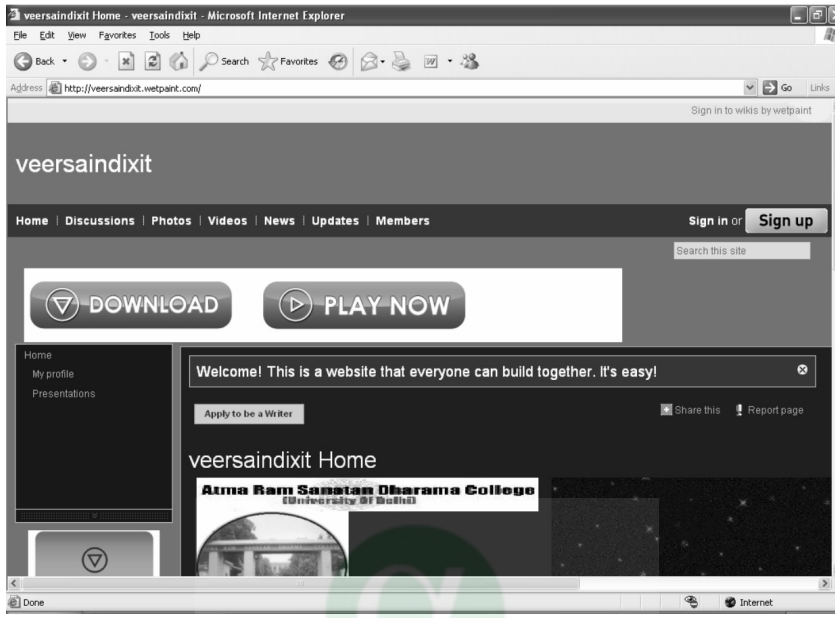


Figure 4.10

4.12 ADDITION OF BACKGROUND COLORS

The following html code demonstrates the addition of background colors:

```
<html>
<head>
</head>
<body bgcolor =”#FFFFFF”>Display the background<p>
</body>
</html>
```

Note: See the Appendix for color codes.

4.13 INSERTION OF BACKGROUND IMAGE

HTML code to insert the background images:

```
<html>
<head>
</head>
<body background =”c:\windows/cloud.gif”>Display the background<p>
</body>
</html>
```

The output of the above html code is given below:

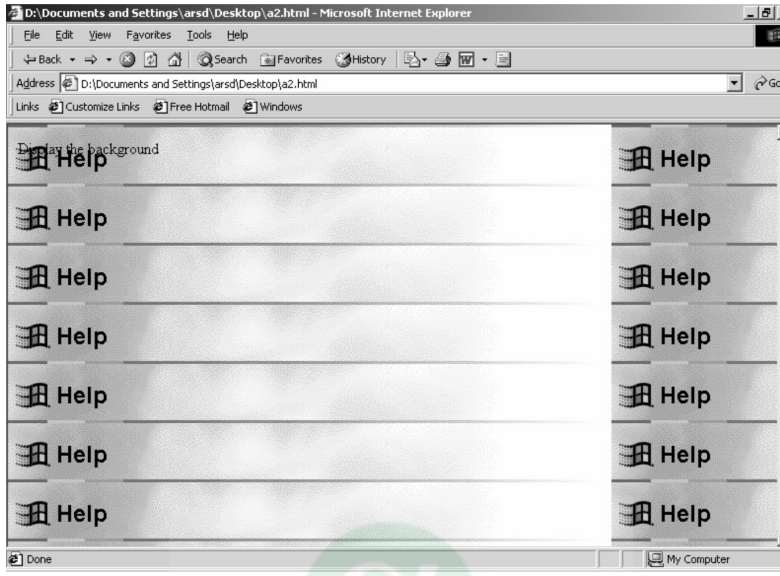


Figure 4.11

Note:

- Banners, buttons, dividers, clipart and other simple images usually work best as GIF's (Graphics Interchange Format). Photographs and other complex images usually work best as JPEG's (Joint Picture Expert Group).
- If an image is stored in the same folder as the HTML page, you can leave out the domain reference (http://www.arsdcollege.net/) and simply insert the image with this code:

```
<img src="c:\windows/cloud.gif">
```

You can change the size of an image by changing the width and height attributes.

```
<img src="c:\windows/cloud.gif" width="70" height="80">
```

If you are not assigning the values of width and height, then the browser will use the default size of the image.

You can add a border to the image by using the border attribute as shown in the example given below:

```
<img src="c:\windows/cloud.gif" border="6">
```

Note: Netscape browsers will only show the border if the image is a link.

You can easily add space over and under your images with the Vspace attribute.

In the similar way, you can add space to the left and right of the image by using the Hspace attribute.

The following HTML code demonstrates the above said thing:

```
<img src="cloud.gif" Hspace="60" Vspace="10">
```

4.14 DEFINE COLORS FOR ALL LINKS ON THE PAGE

The general color of text links is specified in the <body> tag, as given in the following example:

```
<body link="#0000FF" vlink="#808080" alink="#FF0000">
```

link - standard link - to a page the visitor hasn't been visited yet. (Standard color is blue - #0000FF).

vlink - visited link - to a page the visitor has visited before. (Standard color is purple - #808080).

alink - active link - the color of the link when the mouse is on it. (Standard color is red - #FF0000).

4.15 CREATION OF EMAIL LINKS

Email links can be created, using the <a href> tag.

An email link will require the following code:

```
<a href="mailto:youremailaddress">Email Me</a>
```

This will result in the visitor's email program, opening a new email with your address already in the To: field.

If you wish to have a specific subject in the email, you can add it to the html code using the subject:

```
<a href="mailto:veersaindixit@rediffmail.com?subject=Personal">
Send Email</a>
```

Suppose you want an email link for your visitors containing specific text in the body of their message, simply add &body=. Let's see the following code:

```
<a href="mailto:veersaindixit@rediffmail.com &body=Please send
me the proposal!">Send Email</a>
```

or

Combine all the options and allow your visitor to send email with the address, subject and text already entered.

```
<a href="mailto:veersaindixit@rediffmail.com?subject=marriage
proposal&body=Please send me a proposal!">Email Me</a>
```

When you choose to use a background image for the page it is always a good idea to specify a background color.

```
<body background="c:\windows/cloud.gif" bgcolor="#444444">
```

By adding the bgproperties="fixed", you force the browser to let the background be fixed, even if the user is scrolling down the page.

Point to be noted that fixed, backgrounds are only supported by MSIE (Microsoft Internet Explorer) and do not work in Netscape browsers.

4.16 CREATION OF TABLE

Table can be created using <table> tag. Table is represented in terms of rows and columns. <tr> tag will represent a row. <td> tag will represent a column/ data. You can span the rows and columns by using the attribute ROWSPAN and COLUMNSPAN. Every table will also have the header. Header will be represented by <th> tag. Table can also have the border. For this purpose, we use BORDER attribute.

Let's see the following html code:

```
<HTML>
<HEAD>
<TITLE> Demonstration of Table creation</TITLE>
</HEAD>
<BODY>
<TABLE BORDER>
<TR>
 <TD>
    <TH ROWSPAN=3></TH>
    <TH COLSPAN=3> HELLO</TH>
 </TD>
</TR>
<TR>
 <TD>
    <TH ROWSPAN=3>FRIENDS</TH>
    <TH COLSPAN=3>HOW ARE YOU</TH>
 </TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

The output of the above html code is:

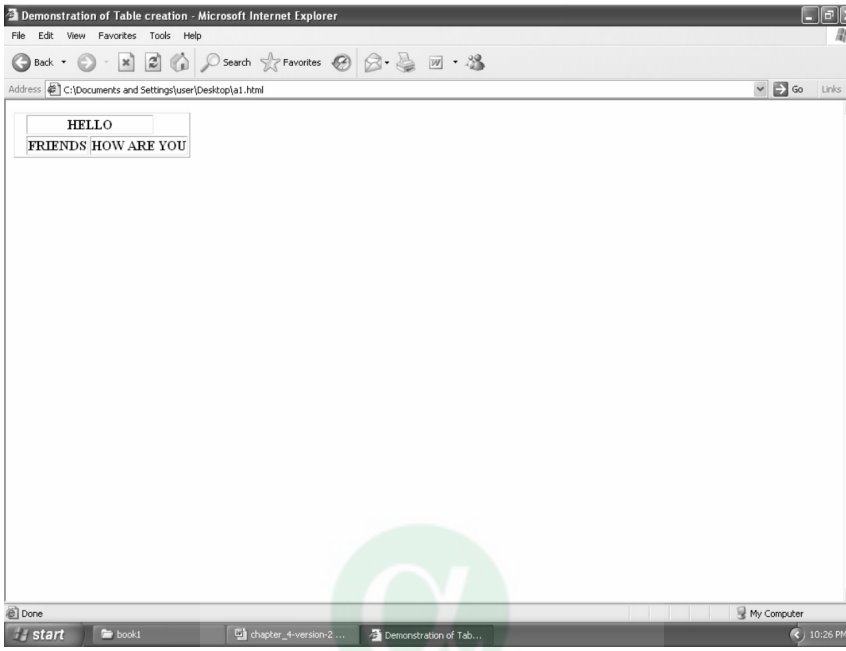


Figure 4.12

Tables are used on the websites for arranging information in a tabular form.

4.17 CREATION OF FRAME

Frames divide the web page into separate windows.

Figure 4.13

Each of these windows can contain a HTML document. A file that specifies how the screen is divided into frames is called a frameset. If you want to make a homepage that uses frames, you should make the normal HTML documents that should be loaded into each of these frames. When a frameset page is loaded, then the browser automatically loads all the pages associated with the frames.

Disadvantages:

- The web developer must keep a track of more HTML documents.
- It is difficult to print the entire page.

Caution: Whenever you are creating web pages, the use of frames should be minimum because more frames will degrade the performance.

A frameset is simply a HTML document that tells the browser how to divide the screen into split windows.

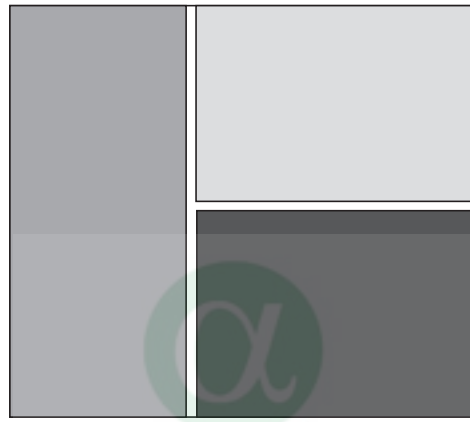


Figure 4.14

Let's write the following html code:

```
<html>
<head>
<title>My Frames Page</title>
</head>

<frameset cols="100,*">
<frame src="c:\windows/cloud.gif" name="menu">
<frameset rows="*,60">
<frame src="c:\windows/content.gif" name="main">
<frame src="c:\windows/cloud.gif" name="bottom">
</frameset>
</frameset>

</html>
```

The output is given below:

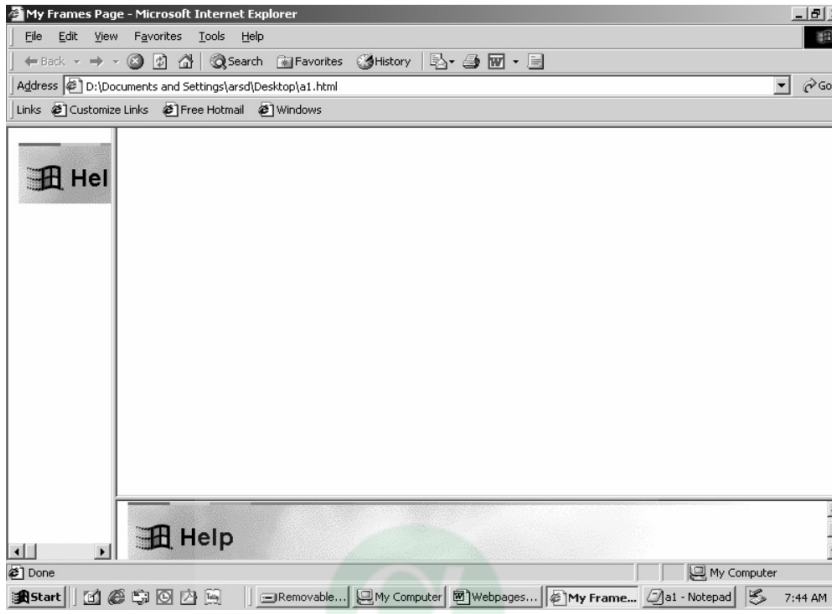


Figure 4.15

In the above HTML code, the screen is divided into two columns. The left being 60 pixels and the right using the rest of the screen (indicated by the *).

To make the frame borders invisible, you simply need to add the parameters "cols-line" to the frameset as given in the following html code:

```
<frameset cols="60,*" frameborder="0" border="0">
<framespacing="0">
<frame src="a1.htm" name="abcd" >
<frame src="a2.htm" name="hello" >
</frameset>
```

The entire frameset would now look like this:

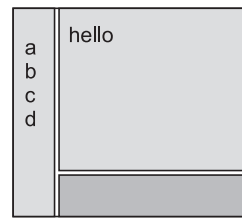


Figure 4.16

If you don't want the frame windows to be resizable, you should add the parameter "noresize" to the frame src lines as given in the following html code:

```
<frameset cols="60,*" frameborder="0" border="0">
<framespacing="0">
<frame src="a1.htm" name="abcd" noresize>
<frame src="a2.htm" name="hello" noresize>
</frameset>
```

The main window should have a scrollbar if needed (if the HTML document doesn't fit in the window), but if not needed - there should be no scrollbars.

The following html code demonstrates the above concepts:

```
<frameset cols="100,*" frameborder="0" border="0">
<framespacing="0">
<frame src="a1.htm" name="abcd" noresize scrolling=no>
<frame src="a2.htm" name="hello" noresize scrolling=auto>
</frameset>
```

If you leave the setting for scrolling, a scrollbar will appear by default if needed-otherwise not.

4.18 CREATION OF FORM

Form is used for the data entry purpose. A form is simply an area that can contain form fields. Form fields are the objects that allow the user to enter information, for example, text boxes, drop-down menus or radio buttons. When the user clicks a submit button, the content of the form is usually sent to a program that runs on the server. When a form is submitted, then all fields on the form are being sent. The <form> tag tells the browser where the form starts and ends. You can add all kinds of HTML tags between the <form> and </form> tags.

Example: Below is a simple form that will send the data using the GET method. Copy, paste this code and save it as "XYZ.html".

```
<form method="GET" action="XYZ.asp">
Name <input type="text" name="Salary"/>
Age <input type="text" name="Age"/>
<input type="submit" value="Submit Query" />
</form>
```

Output:

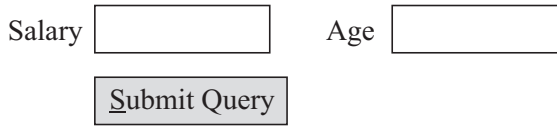


Figure 4.17

Example:

```
<FORM action="http://www.big-llc.com/formmailer/submit"
                                                    method=post>
   <input type="hidden" name="fm-to"
                          value="principal@arsdcollege.net ">
   <input type="hidden" name="fm-title" value="MySite.com
                                                     Feedback">
   <input type="hidden" name="fm-redirect" value=
                                    "http://arsdcollege.net">
 </FORM>
```

In the above code, the post method is used to inform the server that the information appended to the request is to be sent to the specified URL (Uniform Resource Locator). The post method is typically used to send the form data and other information to CGI (Common Gateway Interface) programs.

The following fields can also be added in the forms:

- Text field
- Password field
- Hidden field
- Text area
- Check box
- Radio button
- Drop-down menu
- Submit button
- Reset button
- Image button

4.18.1 Text Field

Text fields are one line areas that allow the user to input text. If you want several lines, then you should use a text area in place of text field.

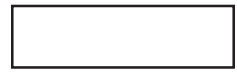


Figure 1.18

Below is a listing of attributes for text fields:

text =	One line text field
size =	Characters shown(width of the field).
maxlength =	Max characters allowed(maximum length of the field).
name =	Name of the field(internal name of the field so that the program can identify the shift field).

value = Initial value in the field(what will appear in the box as default value).

align = Alignment of the field(defines how the field is aligned(top, middle, bottom, right, left, texttop, baseline, absmiddle, absbottom).

tabindex = Tab order of the field (in which order the different fields should be activated when the users click the tab key).

Example:

```
<input type="text" name=" email" value=" " maxlength="100">
```

4.18.2 Password Field

Passwords are used for the security purposes. Password fields are similar to text fields. The only difference is that what ever is entered into a password field is shown as dots on the screen (to prevent others from reading the password on the screen).

Figure 4.19

Below is a list of valid attributes for password fields:

password One line password field

size= Characters shown(width of the field).

maxlength= Max characters allowed(maximum length of the field).

name= Name of the field(internal name of the field so that the program can identify the field).

value= Initial value in the field(what will appear in the box as default value).

align= Alignment of the field(defines how the field is aligned (top, middle, bottom, right, left, texttop, baseline, absmiddle, absbottom).

tabindex= Tab order of the field (in which order the different fields should be activated when the users clicks the tab key).

Example:

```
<input type="password" name=" email" value=" " maxlength="100">
```

4.18.3 Hidden Fields

Hidden fields are similar to the text fields; with one difference (hidden field does not show on the page). Therefore, the user can't type anything into a hidden field which shows the purpose of the field to submit information that is not entered by the user.

Below is a list of valid settings for hidden fields:

hidden field

name= Name of the field(internal name of the field so the program that handles the form can identify the fields).

value= Value of the field(what will be sent once the form is submitted. The hidden field does not show, but still, when the form is submitted, the hidden field is sent with it).

Example:

```
<input type="hidden" name="fm-to"
                          value="principal@arsdcollege.net ">
<input type="hidden" name="fm-title" value="MySite.com
                                                    Feedback">
<input type="hidden" name="fm-redirect" value=
                                      "http://arsdcollege.net">
```

4.18.4 Text Area Field

Text areas are text fields that can span several lines. Text areas are not defined with an <input> tag. Instead, you enter a <textarea> tag where you want the text area to start and a closing </textarea> tag, where you want the area to end. Everything written between these tags will be presented in the text area box.

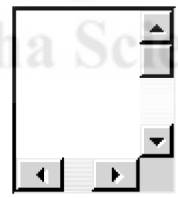


Figure 4.20

Below is a list of valid attributes for text areas:

textarea	Text area - several lines
rows =	Rows in the field.
cols =	Columns in the field.
name =	Name of the field.
tabindex =	Tab order of the field.
wrap off =	Turns off line breaking
virtual =	Shows line breaking but sends text as entered.
physical =	Inserts line breaks when needed and even sends it.

Example:

```
<TEXTAREA TYPE=TEXT NAME="message" WRAP COLS=60 ROWS=15>
```

```
</TEXTAREA>
```

4.18.5 Check Box Field

Check boxes are used when you want the user to select one or more options from a set of alternatives.

Figure 4.21

Below is a list of valid attributes for check boxes:

checkbox	Choose one or more options
name =	Name of the field.
value =	Value which is submitted, if checked
align =	Alignment of the field
tabindex =	Tab order of the field
checked =	Default check this field.

Example:

```
<INPUT type="checkbox" id=checkbox1 name=eng>
                                        B.A. (H) English</td>
```

4.18.6 Radio Button Field

Radio buttons are used when the user is allowed to select only one option from a set of alternatives at a time. If more options are to be allowed at the same time, then you should use check boxes.

Example:

```
<input type=radio name= value = >
```

4.18.7 Drop-down Menu Field

Depending on your properties, drop-down menus can serve the same purpose as radio buttons (one selection only) or check boxes (multiple selections allowed).

Advantage:

Advantage of a drop-down menu, as compared to radio buttons or check boxes is that it consumes less space.

Disadvantage:

User can't see all the options in the menu instantly.

Below is a list of valid attributes for drop-down menus:

select	Drop-down menu
name =	Name of the field.
size =	Visible items in the list.
multiple =	Allows multiple choices, if the answer is yes.
option =	Individual items in the menu.
selected =	Default selects the item.
value =	Value to send, if selected.

Example:

```
<SELECT name=date><OPTION value=""selected><OPTION value=1>1
                    <OPTION value=2>2
                    <OPTION value=3>3
                    <OPTION value=4>4
                    <OPTION value=5>5
                    <OPTION value=6>6
                    <OPTION value=7>7
                    <OPTION value=8>8
                    <OPTION value=9>9
                    <OPTION value=10>10
                    <OPTION value=11>11
                    <OPTION value=12>12
                    <OPTION value=13>13
                    <OPTION value=14>14
                    <OPTION value=15>15
                    <OPTION value=16>16
                    <OPTION value=17>17
                    <OPTION value=18>18
                    <OPTION value=19>19
                    <OPTION value=20>20
                    <OPTION value=21>21
                    <OPTION value=22>22
                    <OPTION value=23>23
                    <OPTION value=24>24
                    <OPTION value=25>25
                    <OPTION value=26>26
                    <OPTION value=27>27
                    <OPTION value=28>28
                    <OPTION value=29>29
                    <OPTION value=30>30
                    <OPTION value=31>31
                    </OPTION></SELECT>
```

4.18.8 Submit Button

When a user clicks a submit button, the form is sent to the address specified in the action properties of the <form> tag.

Below is a list of valid properties for submit button:

submit	Submit button
name=	Name of the button.
value=	Text written on the button
align=	Alignment of the button.
tabindex=	Tab order of the button.

Example:

```
<INPUT type="submit" value="Submit" id=submit1 name=submit1>
```

4.18.9 Reset Button

When a user clicks a reset button, the entries are reset to the default values. Below is a list of valid properties for reset buttons:

Reset =	reset button
name =	Name of the button
value =	Text written on the button
align =	Alignment of the button
tabindex =	Tab order of the button.

Example:

```
<INPUT type="reset" value="Reset" id=reset1 name=reset1>
```

4.18.10 Image Button

Image buttons have the same effect as submit buttons. When a visitor clicks an image button, then the form is sent to the address specified in the action properties of the <form> tag.

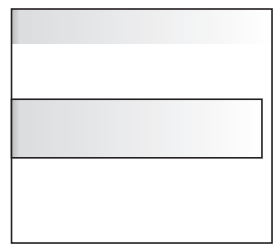


Figure 4.22

Below is a list of valid settings for image buttons:

image	Submit button
name=	name of the image
src =	URL of the image
align =	Alignment of the image
border =	border width around the image
width =	width of the image
Height =	height of the image
vspace =	spacing over and under the image
hspace =	spacing left and right of the image
tabindex =	tab order of image

In section 4.21, we have shown the uses of above tags by using various examples.

4.19 USE OF STYLES

Style sheets is a specification that gives the webpage author, visual control over HTML elements such as location, size, color, background, images, etc.

When a browser reads a style sheet, the browser will format the document according to Style sheet.

You can insert style sheet in three ways:

External Style Sheet

External style sheet is recommended when the style is applied to many pages.. You can change the look of an entire Web site by changing one file, by using an external style sheet. Each page must link to the style sheet using the <link> tag. The <link> tag will be inside the head section.

```
<head>
<link rel="stylesheet" type="text/css" href="style1.css">
</head>
```

Internal Style Sheet

When a single document has a unique style, then an Internal Style Sheet is recommended. You can define internal styles in the head section with the <style> tag.

```
<head>
<style type="text/css">
body {background-color: yellow}
p {margin-left: 30px}
</style>
</head>
```

Inline Styles

When a unique style is to be applied to a single occurrence of an element, then Inline Style Sheet is recommended.

To use inline styles, you can use the style attribute in the relevant tag. The style attribute can contain any CSS (Cascade Style Sheet) property.

The following example shows how to change the color and the left margin of a paragraph:

```
<p style="color: yellow; margin-left: 30px">
```

This is a paragraph</p>

Style Tags:

Tag	Description
<style>	Defines a style definition
<link>	Defines a resource reference
	Defines a section in a document
	Defines a section in a document

Example: Design of menubar using Style Sheet:

```
<style type="text/css" media=screen>
<!–
#barameniu {
   margin: 0 auto;
   padding: 0;
   position: relative;
   width: 980px;
   height: 45px;
      background: #CCCCFF url(bg36.png) 0 0 repeat-x;
   font: bold 12px tahoma, Arial, Helvetica, sans-serif;
```

```
    }
#barameniu ul {
    margin: 0;
    padding: 0;
    float: none;
}

#barameniu li {
    padding: 0 0 0 0;
    margin: 0;
    float: left;
        border-right:double #ccccff ; list-style:none;
}

#barameniu ul li a:link, #barameniu ul li a:active,
                                #barameniu ul li a:visited {
    display: block;
    float: left;
    padding: 0 20px;
    height: 28px;
    line-height: 28px;
    color: #cc99ff ;
    text-decoration: none;
}

#barameniu ul li a:hover {
background: url(bg36.png) 0 100% repeat-x;
}
—>
</style>
```

4.20 THE META ELEMENT

The head element contains the general information (meta-information) about a document. HTML also includes a Meta element that goes inside the head element. The purpose of the Meta element is to provide meta-information about the document.

Mostly, often the Meta element is used to provide the information that is relevant to browsers or search engines like describing the content of your document.

Note: According to W3C (World Wide Web Consortium), "Some user agents support the use of META to refresh the current page after a specified number of seconds with the option of replacing it by a different URL. Authors should not use this technique to forward users to different pages as this makes the page inaccessible to some users. Instead, automatic page forwarding should be done using server-side redirects".

Keywords for Search Engines

Some search engines on the WWW use the name and content attributes of the Meta tag to index the pages.

This Meta element defines a description of your page:

```
<meta name="description" content="Free Web tutorials on HTML,
                                        CSS, XML, and XHTML">
```

This Meta element defines keywords for your page:

```
<meta name="keywords" content="HTML, DHTML, CSS, XML, XHTML,
                                    JavaScript, VBScript">
```

The intention of the name and content attributes it to describe the content of a page.

4.21 SOLVED PROBLEMS

Three projects are coded below. Read the html code of these projects very carefully and try to understand them.

Project 1: Creation of a suggestion box: In this project, we are creating a suggestion box in which the user will submit his suggestion to the college.

```
<HTML>
<HEAD>
<TITLE>Current Messages</TITLE>
</HEAD>

<body bgcolor=#ffdda4>
<H1>The college Email is <i>principal@arsdcollege.net</i>
                                                    </H1></CENTER>

<CENTER><H1>Suggestion Box</H1></CENTER>

<HR>

</BODY>
</HTML>
<form action="http://www.big-llc.com/formmailer/submit"
                                                      method=post>
```

```
<input type="hidden" name="fm-to"
                        value="principal@arsdcollege.net ">
<input type="hidden" name="fm-title" value="MySite.com
                                                    Feedback">
<input type="hidden" name="fm-redirect" value=
"http://arsdcollege.net">
<P>
<table border="0">
<tbody>
   <tr>
         <td class="small" width=500 >First Name</td>
         <td class="small" width=500 >Last Name</td>
   </tr>
   <tr>
      <td class="small" width=500><input type="text"
                                              name="firstname"
      value="" maxlength="20"></font><font size="2"
                                                  face="Arial"
      color="#CC3333"> *</font></td>

      <td class="small" width=500><input type="text"
                                               name="lastname"
      value="" maxlength="250"></font><font size="2"
           face="Arial" color="#CC3333"> *</font></td>
   </tr>

   <tr>
      <td class="small" width=70%>Your E-Mail Address </td>
   </tr>
   <tr>
      <td ><input type="text" name=" email" value=""
            maxlength="100"><font size="2" face="Arial"
            color="#CC3333"> *</font>
      </td>
   </tr>
</tbody></table></td></tr>
<h1>Now Type your suggestion :< /h1><BR>

   <TEXTAREA TYPE=TEXT NAME="message" WRAP COLS=60 ROWS=15>
   </TEXTAREA>

   <INPUT TYPE=SUBMIT NAME="submit" VALUE="Submit ">
   <INPUT TYPE=RESET NAME="reset" VALUE="Clear ">
</FORM>
```

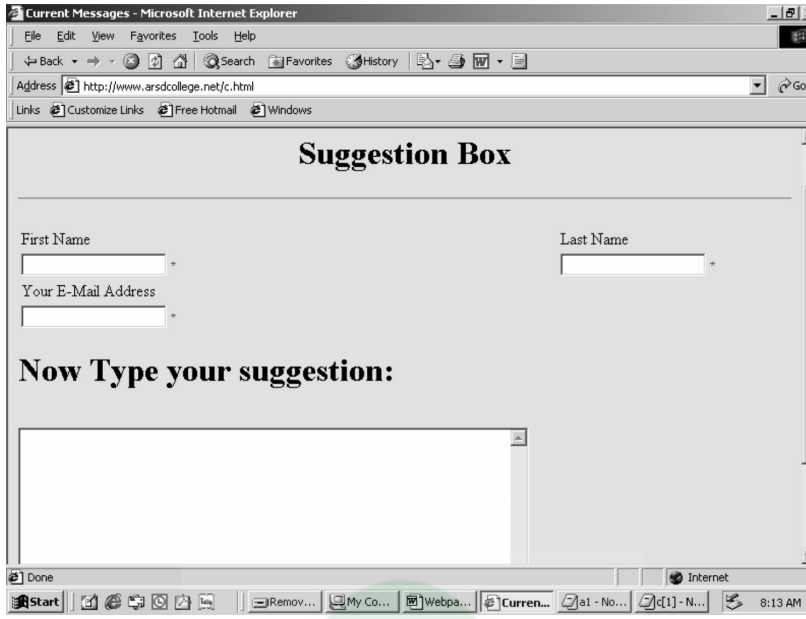


Figure 4.23

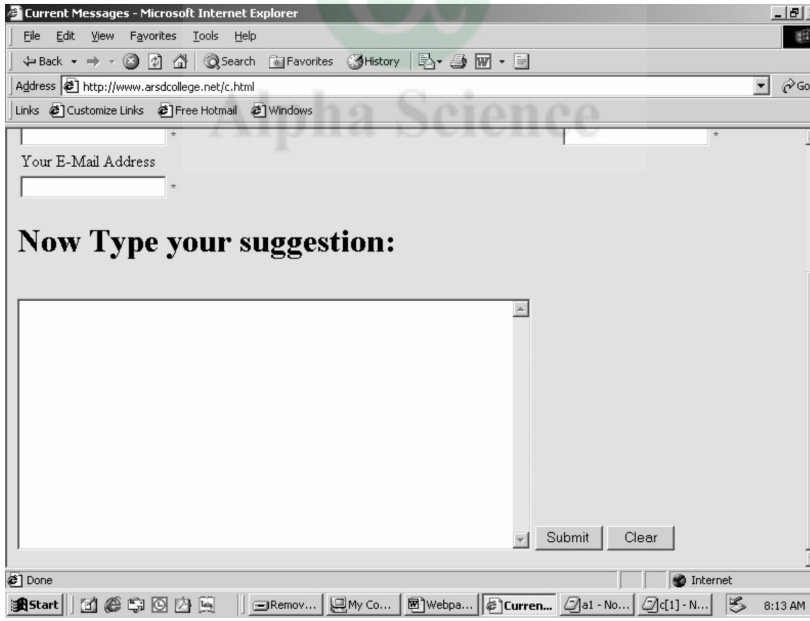


Figure 4.24

In the above code, the post method is used to inform the server that the information appended to the request is to be sent to the specified URL (Uniform Resource Locator). The post method is used to send the form data and other information to CGI (Common Gateway Interface) programs.

Project 2: Creation of College alumni form: Through this alumni form, the former students of the institution can register themselves.

```
<html>
<head>
<title>ARSD Alumni </title>
</head>

<body    bgcolor="#FFFFFF"    leftmargin="0"    topmargin="0"
marginwidth="0" marginheight="0" background=/images/bakgrnd.gif
link="#000066" vlink="#000066" alink="#000066">
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr bgcolor="#FFAB1B"><td> </td></tr>
  <tr bgcolor="#0A0D6F">
  <td>
    <table width="100%" border="0" cellspacing="0"
                                              cellpadding="0">
    <tr>
        <td valign="bottom" align="left" width="75%"><img
        src=/images/Clear.gif><img src=/images/
                                               lumdb.gif></td>

      <td align="center" valign="middle" width="25%"><img
      src=/images/arsdlogoblue.gif></td>
    </tr>
    </table>
    </td>
    </tr>
</table>
    <br>
    <br>
    <table width="100%" border="0" cellspacing="0"
                                              cellpadding="0">
    <tr>
    <td width="16%" align="left" valign="top">
      <br>
    <table width="155" border="0" cellspacing="5"
                                              cellpadding="0">
    <tr>
      <td colspan="2" align="left" valign="top"><font
      color="#CC3333">A.R.S.D. - ians</font></td>
    </tr>
    <tr>
      <td width="9 %">< /td>
```

```
    <td width="91%" bgcolor="#FFFFFF"><img src=/images/
                                                    lear.gif
    width="1" height="1"></td>
  </tr>
  <tr>
    <td width="9%" align="left" valign="top" ><img
   src=/images/DbleArrow.gif></td>
  <td width="91%" class="small"><a
 href=http://arsdcollege.net>Home</a></td>
</tr>
<tr>
  <td width="9%"></td>
  <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
 width="1" height="1"></td>
</tr>
<tr>
  <td width="9%" align="left" valign="top" ><img
 src=/images/DbleArrow.gif></td>
  <td width="91%" class="small">
      <a href="Collegepride.html">College pride</a>
  </td>
</tr>
<tr>
  <td width="9 %">< /td>
  <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
 width="1" height="1"></td>
</tr>

<tr>
  <td width="9%" align="left" valign="top" ><img
 src=/images/DbleArrow.gif></td>
  <td width="91%" class="small">
    <a href="INFOFRIENDS.html">  Infofriends! </a>
  </td>
</tr>

<tr>
  <td width="9%"></td>
  <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
 width="1" height="1"></td>
</tr>
<tr>
  <td width="9%" align="left" valign="top" ><img
```

```
 src=/images/DbleArrow.gif></td>
  <td width="91%" class="small">
    <a href="alumni.asp">UpdateInfo</a>
  </td>
</tr>

<tr>
  <td width="9%"></td>
  <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
width="1" height="1"></td>
</tr>

</table>
<br>
</td>

<td width="84%" align="left" valign="top">


<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tbody>
<! – BEGIN: Title –>
<tr>
<td align="right" valign="top" width="7 %"><img src=/images/
YellCircBig.gif></td>
<td align="left" valign="top" width="93 %">
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tbody>
  <tr align="left" valign="top">
    <td><font size="+1" color="#000066"><b>Information
database</b></font></td>
  </tr>
  <tr align="left" valign="top" bgcolor="#000066">
    <td><img src=/images/clear.gif width="1"
height="1"></td>
  </tr>
</tbody>
</table>
</td>
</tr>
<! – END: Title –>

<tr align="left" valign="top">
<td></td>
<td>
```

```
<table width=”80%” border=”0" cellspacing=”0" cellpadding=”0">
<tbody>
<tr><td>
<p><br>

<B><font color=#000066>Note!!</B></font>
<br> All EX-ARSDians are requested to help us in gathering
information by  filling up the form
 </p>

<form      action=”http://www.big-llc.com/formmailer/submit”
method=post>
<input               type=”hidden”               name=”fm-to”
value=”principal@arsdcollege.net”>
  <input  type=”hidden”  name=”fm-title”  value=”MySite.com
Feedback”>
  <input  type=”hidden”  name=”fm-redirect”  value=”http://
arsdcollege.net”>
<table cellspacing=”0" border=”0">
<tbody>

<TR><TD><font size=”2" face=”Arial” color=”#CC3333">* = required
field</font></TD></TR>
<tr><th         align=”left”         bgcolor=”#0A0D6F”><font
color=”white”>Form</font></th></tr>
<tr><TD     bgColor=#ffffcc><B>Secured     Information<font
size=1>(Which you can use to update your Information)</font></
B> </TD></TR>
<tr bgcolor=#E8E8E8><td>
  <table border=”0">
    <tbody>

</tbody></table></td></tr>
<tr><TD bgColor=#ffffcc><B>Personal Information</B> </TD></
TR>

<tr bgcolor=#E8E8E8><td>
  <table border=”0">
    <tbody>
 <tr>
      <td class=”small” width=500 >First Name</td>
      <td class=”small” width=500 >Last Name</td>
   </tr>
    <tr>
```

```
      <td class="small" width=500><input type="text"
name="firstname" value=""            maxlength="20"></
font><font size="2" face="Arial"
color="#CC3333"> *</font></td>
      <td class="small" width=500><input type="text"
name="lastname" value=""          maxlength="250"></
font><font size="2" face="Arial"
color="#CC3333"> *</font></td>
    </tr>
    <tr>
      <td class="small" width=500 >Course </td>
      <td class="small" width=500 >Batch </td>
    </tr>
<tr>
<TD width=500>
<SELECT   name=course><OPTION   value=""   selected><OPTION
value=0>Bachelor of Information Sciences
    <OPTION value=1>B.Sc(H) Electronics<OPTION value=2>B.Sc(H)

                Physics
    <OPTION value=3>B.Sc(H) Maths
    <OPTION value=4>B.Sc(H) Chemistry
    <OPTION value=5>B.Sc(Gen) Non-Electronics
    <OPTION value=6>B.Sc(Gen) Electronics
    <OPTION value=7>B.Sc(Gen) Industrial Chemistry
    <OPTION value=8>B.Com.(H)
    <OPTION value=9>B.Com(pass)
    <OPTION value=10>B.A. (pass)
    <OPTION value=11>B.A. (H) English
    <OPTION value=12>B.A. (H) Maths
    <OPTION value=13>B.A. (H) Economics
    <OPTION value=14>B.A. (H) History
    <OPTION value=15>B.A. (H) Hindi
    <OPTION value=16>B.A. (H) Political Science
    <OPTION value=17>M.Com
    <OPTION value=18>M.A. English
    <OPTION value=19>M.A. Hindi
    <OPTION value=20>M.A. Political Science
    </OPTION>
</SELECT><font size="2" face="Arial" color="#CC3333"> *</
font>
</td>

<TD width=500>
<SELECT name=batch>
```

```
      <OPTION value="" selected>
      <OPTION value=1980>1980
      <OPTION value=1981>1981
      <OPTION value=1982>1982
      <OPTION value=1983>1983
      <OPTION value=1984>1984
      <OPTION value=1985>1985
      <OPTION value=1986>1986
      <OPTION value=1987>1987
      <OPTION value=1988>1988
      <OPTION value=1989>1989
      <OPTION value=1990>1990
      <OPTION value=1991>1991
      <OPTION value=1992>1992
      <OPTION value=1993>1993
      <OPTION value=1994>1994
      <OPTION value=1995>1995
      <OPTION value=1996>1996
      <OPTION value=1997>1997
      <OPTION value=1998>1998
      <OPTION value=1999>1999
      <OPTION value=2000>2000
      <OPTION value=2001>2001
      <OPTION value=2002>2002
      </OPTION>
      </SELECT>
<font size="2" face="Arial" color="#CC3333"> *</font>
</td>

</tr>
<tr>
<td class="small" width=70%>Roll number </td>
</tr>
<tr>
<td    ><input    type="text"    name="rollno"    value=""
maxlength="100"><font         size="2"         face="Arial"
color="#CC3333"> *</font>
</td>
</tr>
<tr>
<td class="small" width=70%>E-Mail </td>
</tr>
<tr>
```

```
<td ><input type="text" name="email" value="" maxlength="100">
<font size="2" face="Arial" color="#CC3333"> *</font></
td>
</tr>
</tbody></table>
</td>
</tr>

<tr><td bgcolor="#FFFFCC" align=left><b>Residential Address</
b></td>
</TR>

<tr bgcolor=#E8E8E8><td>
  <table border="0">
    <tbody>
      <tr>

        <td class='small' width=150>Address</td>
        <td ><input type="text" name="address"
value="" maxlength="200"><font size="2"        face="Arial"
 color="#CC3333"> *</font></td>

        </tr>
        <tr height=30></tr>
      <tr>

      <td class='small' width=150>Postal Code</td>
      <td ><input type="text" name="postalcode" value=""
 maxlength="100" size=10><font size="2"
 face="Arial"
 color="#CC3333"> *</font></td>
      </tr>
      <tr height=30></tr>
      <tr>

      <td class='small' width=150>Telephone:</td>
      </tr>
      <tr height=0></tr>
      <tr>
        <td class='small' width=150>STD/ISD
 Code</td>
        <td width=10><input type="text"
 name="SIcode" value="" maxlength="7"
 size=5><font size="1" face="Arial"
 color="#CC3333"> *</font>
 </td>
        <td class='small' width=100>Number</td>
         <td ><input type="text" name="telnumber"
```

```
 value="" maxlength="100" size=20><font
 size="2" face="Arial"
 color="#CC3333"> *</font>
 </td>
      </tr>
       </tbody>

       </table>
       </td></tr>

<tr>
<td bgcolor="#FFFFCC" align=left><b>Professional Information</
b></Td>
</TR>
<tr bgcolor=#E8E8E8>
<td>
  <table>
      <tbody>
      <tr><td>Job Description</td>
      <td><input type="text" name="jobdes" value=""
 maxlength="200" size=50%></td>
      </tr>
      </tbody>
      </table>
  <table border="0">
    <tbody>


      <tr height=15></tr>
         <tr>

<td class='small' width=150>Office Address</td>
<td    ><input    type="text"    name="offaddr"    value=""
maxlength="200"></td>

                   </tr>
              <tr height=15></tr>
              <tr>
<td class='small' width=150>Postal Code</td>
<td    ><input    type="text"    name="offposcode"    value=""
maxlength="100" size=10></td>
                                              </tr>
              <tr height=15></tr>
              <tr>

<td class='small' width=150>Telephone:</td>
```

```
                </tr>
                <tr height=0></tr>
                <tr>

<td class='small' width=150>STD/ISD Code</td>
<td><input type="text" name="offSIcode" value="" maxlength="7"
size=5></td>
                 <td class='small' width=100>Number</td>
                 <td><input type="text" name="offtelnum" value=""
maxlength="100" size=20></td>
<td class='small' width=100 align=right>Ext</td>
<td ><input type="text" name="offext" value="" maxlength="100"
size=5></td>
                 </tr>
            </tbody>

       </table>

       </td></tr>

<tr height=25><td align="left" bgcolor="#0A0D6F"></td></tr>

<tr height=30></tr>
<tr><td><INPUT type="reset" value="Reset" id=reset1 name=reset1>
<INPUT type="submit" value="Submit" id=submit1 name=submit1>
</td></tr>
</tbody>
</table>
</form>

</td></tr></tbody></table>
</td>
<!—BEGIN Right-Hand Column—>
<TD>      </td>
<!—END Right-Hand Column—>
</tr></tbody></table>
<!— BEGIN: Footer —>
</td></tr></table>
<p> </p>
</body>
</html>
```

The output of the above html code is given below:

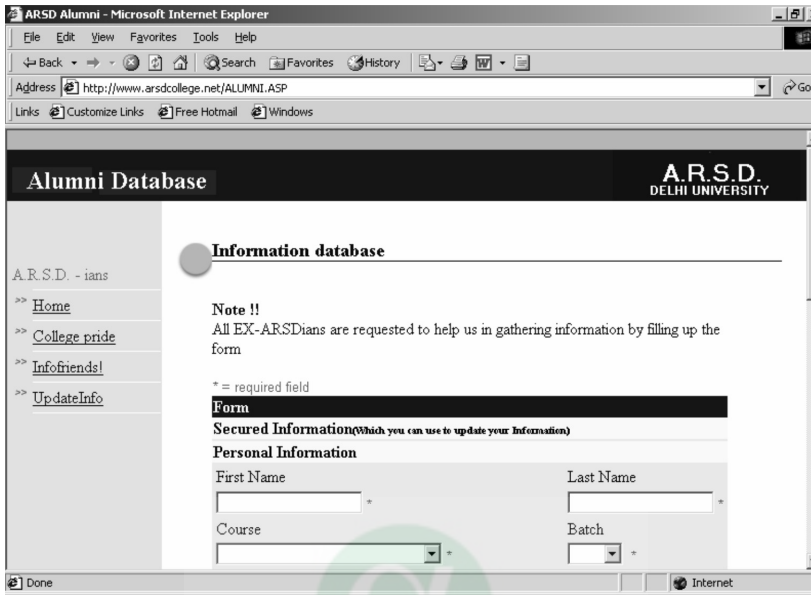


Figure 4.25

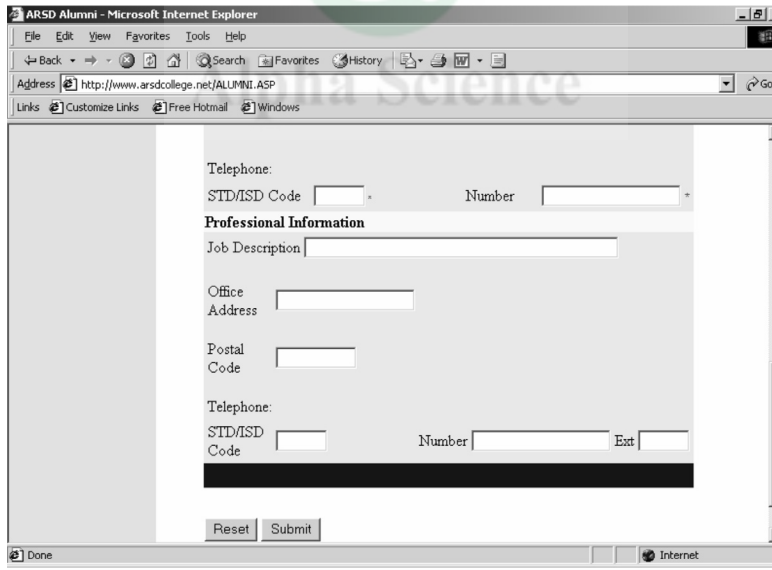


Figure 4.26

Project 3: **Creation of online admission form**: Now, we are creating an online admission form through which the students can register themselves for the admission in the institution:

```
<html>
<head>
<title>ARSD ONLINE ADMISSIONS </title>
</head>

<body bgcolor="#FFFFFF" leftmargin="0" topmargin="0"
marginwidth="0" marginheight="0" background=/images/bakgrnd.gif
link="#000066" vlink="#000066" alink="#000066">
<table width="100%" border="0" cellspacing="0" cellpadding="0">
 <tr bgcolor="#FFAB1B"><td> </td></tr>
 <tr bgcolor="#0A0D6F">
 <td>
 <table width="100%" border="0" cellspacing="0" cellpadding="0">
 <tr>
 <td valign="bottom" align="left" width="75%"><img
 src=/images/clear.gif width="15" height="8"><img
 src=/images/admTitleONLINE.gif width="228"
 height="37"></td>
 <td align="center" valign="middle" width="25%"><img
 src=/images/arsdlogoblue.gif width="150" height="47"></td>
 </tr>
 </table>
 </td>
 </tr>
</table>
<br>
<br>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
 <tr>
 <td width="16%" align="left" valign="top">
    <br>
<table width="155" border="0" cellspacing="5" cellpadding="0">
<tr>
<td     colspan="2"     align="left"     valign="top"><font
color="#CC3333">Apply To A.R.S.D.</font></td>
</tr>
<tr>
       <td width="9%"></td>
<td width="91%" bgcolor="#FFFFFF"><img src=clear.gif width="1"
height="1"></td>
    </tr>
    <tr>
       <td width="9%" align="left" valign="top" ><img
 src=/images/DbleArrow.gif width="15" height="10"></td>
```

```
        <td width="91%" class="small"><a
    href=http://arsdcollege.net>Home</a></td>
    </tr>
    <tr>
      <td width="9%"></td>
      <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
    width="1" height="1"></td>
    </tr>
    <tr>
      <td width="9%" align="left" valign="top" ><img
    src=/images/DbleArrow.gif width="15" height="10"></td>
      <td width="91%" class="small">
      <a href="cutoff2004.html">CUT-OFF Lists</a>
      </td>
      </tr>

      <tr>
      <td width="9%"></td>
      <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
    width="1" height="1"></td>
      </tr>
      <tr>
      <td width="9%" align="left" valign="top" ><img
    src=/images/DbleArrow.gif width="15" height="10"></td>
      <td width="91%" class="small">    <a href=admin.html>
         Rules &amp;amp; Regulations</td>
      </tr>
      <tr>
      <td width="9 %">< /td>
      <td width="91%" bgcolor="#FFFFFF"><img src=/images/clear.gif
width="1" height="1"></td>
      </tr>
   </table>
   <br>
   </td>
    <td width="84%" align="left" valign="top">
   <table width="100%" border="0" cellspacing="0" cellpadding="0">
   <tbody>
   <!— BEGIN: Title —>
   <tr>
   <td align="right" valign="top" width="7%"><img src=/images/
YellCircBig.gif width="34" height="39"></td>
   <td align="left" valign="top" width="93 %">
```

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
<tbody>
   <tr align="left" valign="top"><td><font size="+1"
 color="#000066"><b>ADMISSIONS</b></font></td></tr>
   <tr align="left" valign="top" bgcolor="#000066"><td><img
 src=clear.gif width="1" height="1"></td></tr>
</tbody>
</table>
</td>
</tr>
<!— END: Title —>

<tr align="left" valign="top">
<td></td>
<td>

<table width="80%" border="0" cellspacing="0" cellpadding="0">
<tbody>
<tr><td>
<p><br>

<B><font color=#000066>Welcome!!</B></font><br> Refer to the
list of <a href=courses.html>courses</a> before filling the form
 </p>

<form      action="http://www.big-llc.com/formmailer/submit"
method=post>
<input               type="hidden"               name="fm-to"
value="principal@arsdcollege.net ">
  <input  type="hidden"  name="fm-title"  value="MySite.com
Feedback">
  <input  type="hidden"  name="fm-redirect"  value="http://
arsdcollege.net">



<table cellspacing="0" border="0">
<tbody>
<tr><td>
</TD></TR>
<TR><TD><font size="2" face="Arial" color="#CC3333">* = required
field</font></TD></TR>
<tr><th         align="left"         bgcolor="#0A0D6F"><font
color="white">Form</font></th></tr>
<tr>
```

```
   <TD bgColor=#ffffcc><B> (To be filled by the candidate) </B>
</TD></TR>

   <tr bgcolor="#E8E8E8"><td>
      <table border="0">
         <tbody>

         <tr>
            <td class="small" colspan=0>1. Name </td>
            <td class="small"><input type="text" name="name"
    value="" maxlength="250"><font size="2" face="Arial"
    color="#CC3333"> *</font></td>
         </tr>
         <tr>
            <td class="small" width=70%>2. Date of Birth (as on
    1.10.2003) </td>
            <td>
   <SELECT name=date><OPTION value=""selected><OPTION value=1>1
            <OPTION value=2>2
            <OPTION value=3>3
            <OPTION value=4>4
            <OPTION value=5>5
            <OPTION value=6>6
            <OPTION value=7>7
            <OPTION value=8>8
            <OPTION value=9>9
            <OPTION value=10>10
            <OPTION value=11>11
            <OPTION value=12>12
            <OPTION value=13>13
            <OPTION value=14>14
            <OPTION value=15>15
            <OPTION value=16>16
            <OPTION value=17>17
            <OPTION value=18>18
            <OPTION value=19>19
            <OPTION value=20>20
            <OPTION value=21>21
            <OPTION value=22>22
            <OPTION value=23>23
            <OPTION value=24>24
            <OPTION value=25>25
            <OPTION value=26>26
```

```
            <OPTION value=27>27
            <OPTION value=28>28
            <OPTION value=29>29
            <OPTION value=30>30
            <OPTION value=31>31
            </OPTION></SELECT>
<SELECT name=month><OPTION value="" selected>

<OPTION value=1>Jan
<OPTION value=2>Feb
<OPTION value=3>Mar
            <OPTION value=4>Apr
            <OPTION value=5>May
            <OPTION value=6>Jun
            <OPTION value=7>Jul
            <OPTION value=8>Aug
            <OPTION value=9>Sep
            <OPTION value=10>Oct
            <OPTION value=11>Nov
            <OPTION value=12>Dec
            </OPTION></SELECT>

<SELECT name=year><OPTION value=""
selected><OPTION value=1973>1973
            <OPTION value=1974>1974
            <OPTION value=1975>1975
            <OPTION value=1976>1976
            <OPTION value=1977>1977
            <OPTION value=1978>1978
            <OPTION value=1979>1979
            <OPTION value=1980>1980
            <OPTION value=1981>1981
            <OPTION value=1982>1982
            <OPTION value=1983>1983
            <OPTION value=1984>1984
            <OPTION value=1985>1985
            <OPTION value=1986>1986
            <OPTION value=1987>1987
            <OPTION value=1988>1988
            </OPTION></SELECT><font size="2" face="Arial"
            color="#CC3333"> *</font></td>
      </tr>
      <tr>
```

```
            <td colspan=0>3. Father's/Guardian Name</td>
            <td class="small" ><input type="text" name="fname"
         value="" maxlength="100"><font size="2"
face="Arial" color="#CC3333"> *</font></td>
         </tr>
         <tr>
            <td colspan=0>   (a)
Occupation</td>
            <td class="small" ><input type="text" name="focc"
         value="" maxlength="100"><font size="2"
face="Arial" color="#CC3333"> *</font></td>
         </tr>
         <tr>
            <td colspan=0>   (b) Monthly
      Income</td>
            <td class="small" ><input type="text"
      name="fincome" value="nil" maxlength="100"></td>
         </tr>
         <tr>
            <td colspan=0>   (c) Local
      Address</td>
            <td class="small" ><input type="text" name="fadd"
         value="" maxlength="100"><font size="2"
face="Arial" color="#CC3333"> *</font></td>
         </tr>
         <tr>
            <td>4. Nationality</td>

         </tr>
         <tr>
            <td colspan=0>   (a) Home
      Town</td>
            <td class="small" ><input type="text"
      name="hometown" value="" maxlength="100"></td>
         </tr>
         <tr>
            <td colspan=0>   (b) State</td>
            <td class="small" ><input type="text" name="state"
         value="" maxlength="100"></td>
         </tr>
         <tr>
            <td colspan=0>   (c) Country</td>
            <td class="small" ><input type="text" name="country"
         value="" maxlength="100"></td>
         </tr>
```

```
          <tr>
             <td colspan=0>5. Permanent Address(native
       place)</td>
             <td class="small" ><input type="text"
name="address"          value="" maxlength="100"><font size="2"
             face="Arial" color="#CC3333"> *</font></td>
          </tr>
          <tr>
          <td>6. Category</td>
           <td valign='top'><input type=radio name=category
          value=Gen>Gen  <input type=radio
name=category value = SCST checked>SC/ST
             </td>
    </tr>
          <tr>
             <td colspan=0>   Registration no.
          SC/St/Foreign</td>
             <td class="small" ><input type="text"
       name="scstregno" value="nil" maxlength="100"><font
       size="2" face="Arial"
color="#CC3333"> *</font></td>
          </tr>
          </TBODY>
          </TABLE>
          </td></tr>
   <tr><TD bgColor=#ffffcc width=30% align=left><B>Academic
Information</B> </TD></tr>
   <tr bgcolor="#E8E8E8"><td>
   <table width="100%" border="0" cellspacing="0"
cellpadding="0" style="border-collapse: collapse"
bordercolor="#336699">
   <tr width="100 %">
   <td width="12%" align="center"><font size="1"
             face="Verdana"><b>Name of Degree</b></font></td>
   <td width="15%" align="center"><font size="1"
               face="Verdana"><b>Board / Univ</b></font></td>
   <td width="15%" align="center"><font size="1"
                face="Verdana"><b>Roll Number</b></font></td>
   <td width="8%" align="center"><font size="1"
        face="Verdana"><b>Year<br>of<br>passing</b></font></td>
   <td width="15%" align="center"><font size="1"
                   face="Verdana"><b>Subjects</b></font></td>
   <td width="15%" align="center"><font size="1"
                  face="Verdana"><b>Max Marks</b></font></td>
```

```
<td width="2%" align="center"><font size="1"
face="Verdana"><b>Marks Obtd.</b></font></td>
<td width="12%" align="center"><font size="1"
                face="Verdana"><b>%Percentage<br>(Best of 4
          sub.)<br>including<br>one language</b></font></td>
</tr>

<tr><td><hr></td><td><hr></td><td><hr></td><td><hr>
               </td><td><hr></td><td><hr></td><td><hr></td>
                                          <td><hr></td></tr>
<tr>
<td width="12%" align="center"><input type="text"
                       name="degree" size="8" value=""></td>
<td width="15%" align="center"><input type="text"
                       name="board" size="10" value=""></td>
<td width="15%" align="center"><input type="text"
                      name="rollno" size="10" value=""></td>
<td width="8%" align=left><input type="text" name=
                          "passyear" size="4" value=""></td>
<td width="15%" align="center">1.<input type="text"
                           name="s1n" size="7" value=""></td>
<td width="8%" align="center"><input type="text"
                        name="s1max" size="5" value=""></td>
<td width="12%" align=left><input type="text" name="s1m"
                                    size="7" value=""></td>
<td width="12%" align=left><input type="text" name="bf4"
                                  size="7" value=""> %< /td>
</tr>
<tr>
<td width="12%" align="center"></td>
<td width="15%" align="center"></td>
<td width="15%" align="center"></td>
<td width="8%" align="center"></td>
<td width="15%" align="center">2. <input type="text"
                           name="s2n" size="7" value=""></td>
<td width="8%" align="center"><input type="text"
                        name="s2max" size="5" value=""></td>
<td width="12%" align=left><input type="text" name="s2m"
                                    size="7" value=""></td>
</tr>
<tr>
<td width="12%" align="center"></td>
<td width="15%" align="center"></td>
<td width="15%" align="center"></td>
```

```
<td width="8%" align="center"></td>
<td width="15%" align="center">3.<input type="text"
                          name="s3n" size="7" value=""></td>
<td width="8%" align="center"><input type="text"
                        name="s3max" size="5" value=""></td>
<td width="12%" align=left><input type="text" name="s3m"
                                    size="7" value=""></td>
</tr>
<tr>
<td width="12%" align="center"></td>
<td width="15%" align="center"></td>
<td width="15%" align="center"></td>
<td width="8%" align="center"></td>
<td width="15%" align="center">4.<input type="text"
                          name="s4n" size="7" value=""></td>
<td width="8%" align="center"><input type="text"
                        name="s4max" size="5" value=""></td>
<td width="12%" align=left><input type="text" name="s4m"
                                    size="7" value=""></td>
<td><font size="0">(Science Group)</font></td>
</tr>
<tr>
<td width="12%" align="center"></td>
<td width="15%" align="center"></td>
<td width="15%" align="center"></td>
<td width="8%" align="center"></td>
<td width="15%" align="center">5.<input type="text"
                          name="s5n" size="7" value=""></td>
<td width="8%" align="center"><input type="text"
                        name="s5max" size="5" value=""></td>
<td width="12%" align=left><input type="text" name="s5m"
                                    size="7" value=""></td>
<td width="12%" align=left>PCM<input type="text" name="pcm"
                                   size="3" value="">%</td>
</tr>
<tr>
<td width="12%" align="center"></td>
<td width="15%" align="center"></td>
<td width="15%" align="center"></td>
<td width="8%" align="center"></td>
<td width="15%" align="center">Total</td>
<td width="8%" align="center"><input type="text"
                    name="total_max" size="5" value=""></td>
```

```
<td width="12%" align=left><input type="text"
                    name="total_ob" size="7" value=""></td>
<td width="12%" align=left>PM   <input type=
                  "text" name="pm" size="3" value="">%</td>
</tr>
</table>
</tr></td>
<tr><TD bgColor=#ffffcc width=30% align=left><B>Courses
                                        Offered</B> </TD></tr>
<tr bgcolor="#E8E8E8"><td><font size=1><b>NOTE :
            </B></FONT><font size=1>Please tick the courses
                                            which you wish to
apply.</font><br><br></td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=bis>Bachelor of
                                Information Sciences</tr></td>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=elec>B.Sc(H)
                                      Electronics</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=phy>B.Sc(H)
                                         Physics</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=math>B.Sc(H)
                                           Maths</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=chem>B.Sc(H)
                                       Chemistry</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1
       name=gnelec>B.Sc(Programme) Non-Electronics</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1
            name=gelec>B.Sc(Programme) Electronics</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1
     name=gic>B.Sc(Programme) Industrial Chemistry</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=com>
                                        B.Com.(H)</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1
                              name=pcom>B.Com(pass)</td></tr>
```

```
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=pba>B.A.
                                        (Programme)</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=eng>B.A. (H)
                                           English</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=bamath>B.A. (H)
                                             Maths</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=eco>B.A. (H)
                                          Economics</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=his>B.A. (H)
                                           History</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=hin>B.A. (H)
                                             Hindi</td></tr>
<tr bgcolor="#E8E8E8">
<td><INPUT type="checkbox" id=checkbox1 name=polsc>B.A. (H)
                                 Political Science</td></tr>
</td>
</tr>
<tr bgcolor="#E8E8E8"><td><br><br><br></td></tr>
<tr><th align="left" bgcolor="#0A0D6F" height=23></th></tr>

</table>
<br><br>
<INPUT type="reset" value="Reset" id=reset1 name=reset1>
<INPUT type="submit" value="Submit" id=submit1 name=submit1>
</form>
</td></tr></table>
</td>
<!—BEGIN Right-Hand Column—>
<TD>      </td>
<!—END Right-Hand Column—>
</tr></table>
<!— BEGIN: Footer —>
</td></tr></table>
<p> </p>
</body>
</html>
```

The output of the above html code is given below:

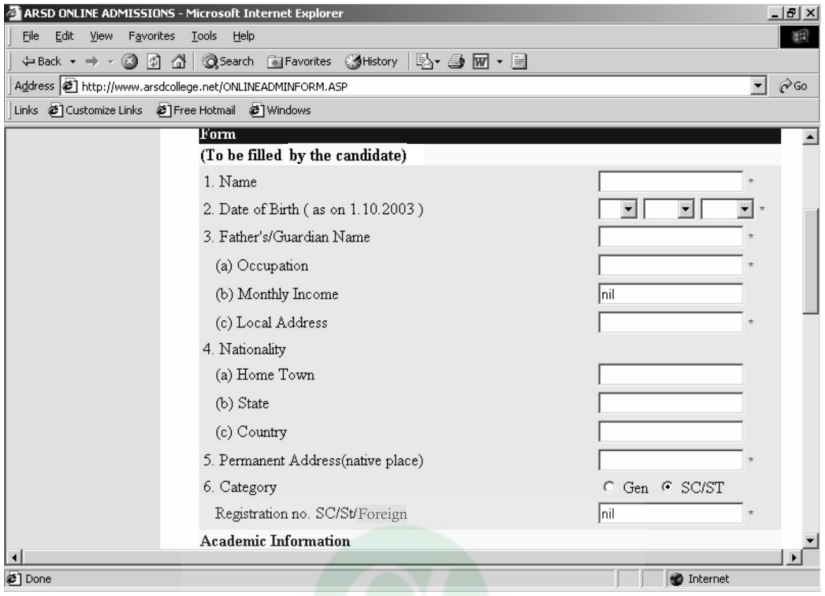


Figure 4.27

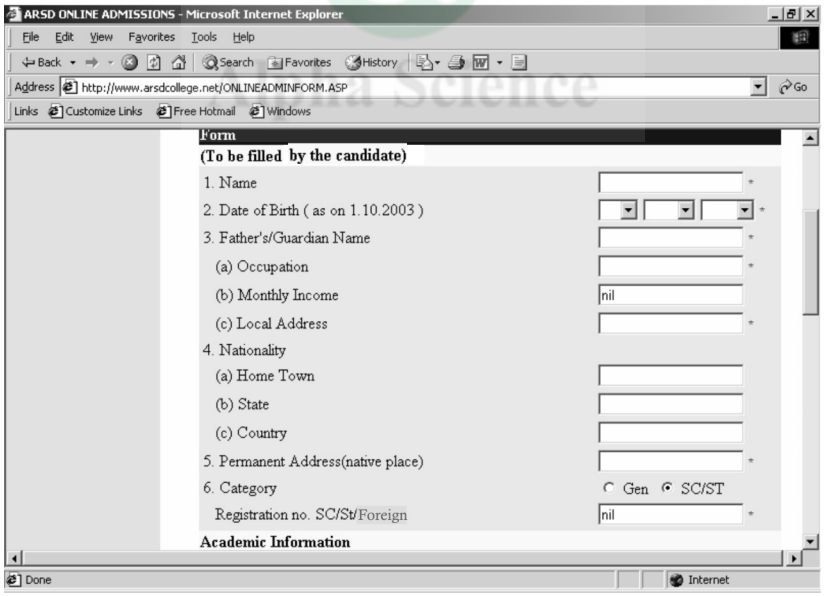


Figure 4.28

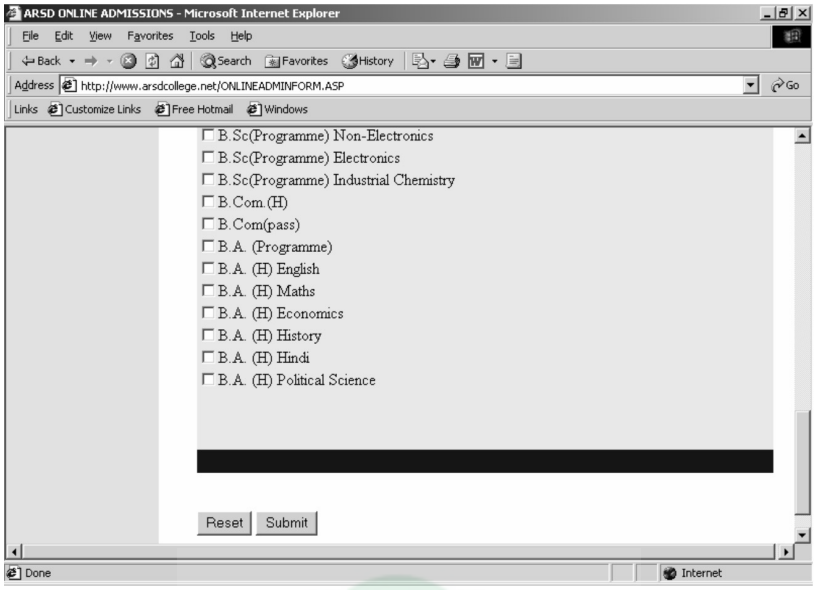


Figure 4.29

4.22 SUMMARY OF TAGS

Tags	Purpose
<!DOCTYPE>	Defines the document type
<html>	Defines a html document
<body>	Defines the body element
<h1> to <h6>	Defines header 1 to header 6
<p>	Defines a paragraph
 	Inserts a single line break
<hr>	Defines a horizontal rule
<!—...—>	Defines a comment
<b>	Defines bold text
<font>	Defines text font, size and color
<i>	Defines italic text
	Defines emphasized text
<big>	Defines big text
	Defines strong text
<small>	Defines small text
<sup>	Defines superscripted text
<sub>	Defines subscripted text
<bdo>	Defines the direction of the text display
<u>	Defines underlined text

Contd....

<pre>	Defines preformatted text
	Defines centered text
<q>	Defines a short quotation
<cite>	Defines a citation
<inrs>	Defines an inserted text
<del>	Defines deleted text
<s>	Defines strikethrough text
<strike>	Defines strikethrough text
<a>	Defines an anchor
<link>	Defines a resource reference
<frame>	Defines a sub window (a frame)
<frameset>	Defines a set of frames
<noframes>	Defines a noframe section
<iframe>	Defines an inline sub window (frame)
<form>	Defines a form
<input>	Defines an input field
<textarea>	Defines a text area
<button>	Defines a push button
<select>	Defines a selectable list
<optgroup>	Defines an option group
<option>	Defines an item in the list box
<label>	Defines a label for a form control
<fieldset>	Defines a fieldset
<legend>	Defines a title in a fieldset
<isindex>	Defines a single-line input field
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dir>	Defines a directory list
<dl>	Defines a definition list
<dt>	Defines a definition term
<dd>	Defines a definition description
<menu>	Defines a menu list
<img>	Defines an image
<map>	Defines an image map
<area>	Defines an area inside an image map
<table>	Defines a table
<caption>	Defines a table caption
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<thead>	Defines a table header
<Marquee>	Rotates the object horizontally/vertically

<tbody>	Defines a table body
<tfoot>	Defines a table footer
<col>	Defines attributes for table columns
<colgroup>	Defines groups of table columns
<style>	Defines a style definition
	Defines a section in a document
	Defines a section in a document
<head>	Defines information about the document
<title>	Defines the document title
<meta>	Defines meta information
<base>	Defines a base URL for all the links in a page
<basefont>	Defines a base font
<script>	Defines a script
<noscript>	Defines a noscript section
<applet>	Defines an applet
<object>	Defines an embedded object
<param>	Defines a parameter for an object

SUMMARY

The purpose of this chapter is to provide the knowledge of web designing, using Fourth generation programming language HTML. In this chapter, there is a discussion about client-side scripting, server-side scripting, static documents, dynamic documents and active documents. There is an explanation of HTML concepts with the help of simple examples. At the end of the chapter, we have demonstrated three major projects by incorporating important tags of HTML.

PROBLEMS

1. Create a web page on the topic "Global Warming and consequences" using HTML.
2. Create an On line admission form using HTML.
3. Create a table using HTML, including border.
4. Create a web page on the topic "My Resume" by including image, table, hyperlinks, option buttons and list box.
5. What is the difference between submit and reset button?
6. What is the difference between radio buttons and check box?
7. What are the Meta tags?
8. What do you mean by Hyperlink? Give the steps for creating Hyperlink.
9. Define 6 level headers.

10. Give the purpose of the following tags:

1. <b>
2. <i>
3. <hr>
4.
5.
6. <img>
7. <a>
8.
9.
10.
11. <form>
12. <table>

11. HTML is a subset of which Markup language?

12. List three characteristics of HTML that makes it ideal for the WWW.

13. What are the benefits of viewing source code on the web?

REFERENCE AND RELATED LINKS

[1] Joel Sklar "Principles of Web Design", course technology, fourth edition, cengage learning 2009.

[2] http://www.w3.org.

[3] http://www.htmlcodetutorial.com/

[4] http://www.echoecho.com

[5] http://www.tizag.com

[6] Douglas E Comer "Everything You Need to Know About Computer Networking and How the Internet Works" 3rd edition[Paperback]

JavaScript and Active Server Pages

5
CHAPTER

Born : 1961
Pittsburg, Pennsylvania, USA.

Known For : JavaScript


Brendan Eich

As discussed in the previous chapter, there are two types of scripting languages. One is Client Side and another one is Server Side. In the following sections, we have chosen JavaScript as Client Side and ASP as server side scripting language for the discussion. Since ASP uses VBScript as default language for writing code, therefore in the ASP section we have used the syntax of VBScript at most of the places. In the first and second section of the chapter, we have discussed JavaScript and ASP. In the third section, we have discussed the concept of DHTML.

5.1 INTRODUCTION TO JAVASCRIPT

Although, JavaScript bears the name of Java but it is a very different language that used for client-side scripting, as well as for server-side scripting. Initially, Netscape developed a language called Live Script to add a basic scripting capability to both Netscape's Navigator and Web Server line of products. When it adds the support for Java Applets in the release of Navigator 2; Netscape replaces Live Script with JavaScript. JavaScript is used in both client side and server side scripting. Client-side scripts are used to create dynamic web pages that are interactive, responsive, tightly integrated with plug-ins, Active X components, and Java Applets. JavaScript scripts are embedded in HTML documents via the HTML <SCRIPT> tag. JavaScript is a case sensitive scripting language. JavaScript supports the object-based programming concept. In a JavaScript, we use predefined objects. The predefined objects enable your scripts

to control the way information is displayed by your browser and also how your browser responds to the user events. JavaScript is supported by Netscape Navigator, Microsoft Internet Explorer and Opera software's Opera browser. JavaScript is an interpreted language with a C like syntax. JavaScript supports many advanced concepts such as recursion, lambda and closures. JavaScript is also a camel-cased language. That is if you are trying to express more than one word, you will eliminate the spaces. Leave the first letter in lower case of the first word and capitalize the first letter of each word after the first word. For example,"getelementbyid" becomes "getElementById". JavaScript is an event driven language which means your scripts react to events that you set up.

JavaScript is not a strongly typed language which means you have to concern yourself with the type of data, a variable stores. In a JavaScript, the variable can store anything, even the functions.

To do a programming in a JavaScript, a text-editor and a browser is needed. Let's create a blank HTML page; open the notepad editor and create a file. Write the following code in a file:

```
<HTML>
    <HEAD>
        <TITLE> JavaScript </TITLE>
    </HEAD>
    <BODY>
            <P> Hello University of Delhi
    </BODY>
</HTML>
```


Save the file with extension .html or .htm and then open the browser and type the file name you just created to see the results.

5.1.1 How to Define a JavaScript Block in Web Page?

You can place the following script block any where on the page, as per your requirement:

```
<script type= "text/JavaScript">

    // write code

</script>
```

Generally what happens if we create our web page using HTML, always the web page will always be static and not interactive. But, if you want to generate a dynamic web page, then you can embed the above JavaScript block any where in your HTML code.

In a JavaScript, every statement will be terminated by a semicolon (;).

5.1.2 Comments

If there is any comment in your script, this will always be ignored by the language interpreter. It is used to explain your code. Through comments you can enhance the readability of the script. JavaScript supports two kind of comments:

Firstly, slashes (//) to tell JavaScript to ignore everything till the end of the line. This is known as single line comment.

Secondly, comment block with a slash-asterisk (/*) and JavaScript will ignore every thing from the start of the comment block until it encounters an asterisk-slash (*/). This is known as a multiline comment.

Example:

```
var x=7; // the value of variable is 7.
{/ * This is
       multiline comment * / }
```

5.1.3 Variable

A variable is a storage location that contains the data value. The value of a variable may vary during the execution of a program. Through variable, we can store the value in computer's memory. Variable is a named unit of storage. JavaScript is not a strongly typed language. Declaration of variables is not required. Whatever value we assign to the variables, the data type of variable will be same as data type of value. JavaScript supports dynamic typing.

In a JavaScript, variables can store anything, even the functions. Variable names can begin with an upper case letter (A – Z), lower case letter (a – z), an underscore character(_), or dollar sign character ($). The remaining part of the variable can consist of any combination of letters, the underscore characters, the dollar sign characters, or digits (0 through 9).Variable names are case sensitive. This means that a variable name add refers to a different value than one named Add, aDd.

Note: A variable cannot not be a JavaScript reserved word or begin with a number or any symbol other than $ or _. In an Internet Explorer, you should also avoid variable with the same name as html elements you have named.

The following are invalid variable names:

```
76sum: cannot start with a number
*sum: cannot start with *.
```

The functions can also be defined and act like variables. Once a function has been defined, it can be passed to other functions as an argument. **This process is known as lambda.**

5.1.3.1 Declaration of variable

Let's first start with the following example:

```
<HEAD>
<SCRIPT language="JavaScript">
<!—
var name=value;
//—>
</SCRIPT>
</HEAD>
```

The explanation of the above code is given below:

1. **var**
   This indicates that you are declaring a variable.
2. **name**
   It is the name you give to the variable. You can any name you like (other than a JavaScript reserve words such as "while" or "onMouseClick".).
3. **value**
   This is the initial value that you are assign to a variable. It can be a number, string, true, false, or null.

5.1.4 Data Types

JavaScript supports the following data types namely number, string, Boolean, null, and undefined.

5.1.4.1 Number

Number data type can be an integer, floating-point numbers and the special NaN (not a number value). Numbers use a 64-bit IEEE 754 format. Integers can be represented in JavaScript in decimal, hexadecimal, or octal forms. You can assign a number value to a variable by placing the desired number after the assignment operator:

```
var number1=6;
```

You can also use a number with a decimal value. JavaScript isn't too choosy about, whether the value is an integer or a decimal.

```
var abc =2.25;
```

5.1.4.2 String

A variable of string data type consists of string values that are enclosed in single or double quotes. To insert a quote character in a string, you must precede it by backslash (\) escape character.

A string is a group of characters, such as a sentence or a word. To define a string, you need to place single or double quoted marks around the value. Following are the examples of string data type.

```
var myName="Ananya";
var hello=" hello arsd"
```

Also, if you place numbers inside the quotes, they are treated as a string rather than a numerical value.

5.1.4.3 Boolean

A variable of Boolean data type consists of the logical value which is either true or false.

JavaScript automatically converts the Boolean values of true and false into 1 and 0 respectively, when they are used in numerical expressions.

Examples:

document.write(true*8+false*9) prints the value 8**.**

Assign the variable a value of true or false.

```
var hello=true;
```

5.1.4.4 Null

A variable of Null data type consists of a single value, which identifies a null, empty or a non existant reference. The null value is automatically converted into default values of other types when used in an expression because the null value is common to all JavaScript types. Null value is used to set a variable to an initial value which is different from other valid values.

If you declare something as null, it means nothing, not even zero:

```
var yes=null;
```

5.1.4.5 Undefined

When a variable consists of a single value, undefined, it only indicates that a variable has been created but not assigned a value. The undefined value is also common to all JavaScript types and is automatically converted to default values of these types. The undefined value is converted to Nan for numeric types, false for Boolean, and undefined for string data types.

JavaScript automatically converts values from one type to another when they are used in an expression.

Example:

```
"myageis " +20
gives myageis20.
```

In the above example, the expression "myageis"+20 will convert the numeric value 20 into string "20" and append it to the string "myageis", producing "myageis20".

JavaScript's automatic type conversion also allows you to assign a value of one type to a variable and then assign a value of a different to the same variable. When JavaScript assigns a value to a variable, it changes the type associated with the variable to the type of the value that is assigned because Java Script supports dynamic typing.

5.1.4.6 Variable Scope

Variable in JavaScript has a function scope. All variables are global unless they are explicitly defined inside a function and even then child-functions have access to their parent's variables. If a function defines a new variable without using the var keyword, then that variable will be global in scope.

5.1.5 Keywords

Keywords are different from variable names. Keywords are used by the language to define its rules and structure. Keywords cannot be used as variable names.

List of Keywords in JavaScript

abstract	final
Boolean	finally
break	float
byte	for
case	function
catch	go to
char	if
class	implements
comment	import
const	in
continue	instance Of
debugger	
default	throw
delete	throws
do	transient
double	true
else	try
enum	typeof
export	var
extends	void
false	while

with	null
static	label
super	long
switch	native
synchronized	new
this	package
return	private
short	protected
public	

JavaScript has a few pre-defined variables or keywords with special meaning:

1. **Not a Number (NAN):** When an arithmatic operation returns an invalid result, then NaN is generated.
2. **Infinity**: Is a keyword which is returned when an arithmatic operation overflows JavaScript's precision. This is in the order of 300 digits. You can find the exact minimum and maximum range for your JavaScript implementation by using Number MAX_VALUE and MIN_VALUE.
3. **Null** is a reserved word. When null is used in a Boolean operation, it evaluates the false.
4. JavaScript **supports true and false** as Boolean values.
5. **Undefined**: If a variable hasn't been assigned yet, then that variable will be assigned by an undefined value. In Boolean operations undefined evaluates them as false.

5.1.6 Expressions

"An expression is a combination of constants, variables, function calls, and operators when evaluated, returns a value".

or

"Expression is the combination of operands and operators".

Operands contain the values which an operator uses and operators are special symbols which represent simple computations like addition and multiplication. JavaScript uses the following operators:

5.1.6.1 Arithmetic Operators

(*i*) Addition +

(*ii*) Substraction -

(*iii*) Multiplication *

(*iv*) Division /

(*v*) Modulus or remainder of division %

(*vi*) Pre or post increment ++

(*vii*) Pre or post decrement - -

Now let's discuss the prefix and postfix operators. The position of ++ and – – operators around the variable, on which they are operating are important. If the operator appears before the variable, then they will be incremented or decremented immediately. If they appear after the variable, then they will be incremented or decremented after the value is assigned.

Example 1:

var t= 10; // z=10,t=11

var z =t ++ ;

Example 2:

var t = 10;

var z = t...; // z=10 ,t= 9

Example 3:

var t= 10;

var z= ++ t; // z=11, t=11

Example 4:

var t =10;

var z= – –t; // z=9, t=9

In the first two examples, (1 & 2) z is assigned the value of t and then t is incremented or decremented by one. In the last two examples (3 & 4), t is incremented or decremented by one and then z is assigned the value of t.

5.1.6.2 Logical Operators

Logical operators are used to perform Boolean operations on Boolean operands. There are three logical operators in JavaScript namely logical and (&&), logical or(||) and logical not(!).

Example A: x>0 && x<10 is true, only if x is greater than zero is true, and less than 10 is true.

Example B: x%2 ==0 || x%3==0 is true, if either of the condition is true, that is if the number is divisible by 2 or 3.

Example C :!(x>y) is true, if (x>y) is false.

In **example A,** '&&' is a logical operator. If both the conditions are true, then only the value of whole expression will be true other wise it will be false.

Truth table of logical operator '&&'

x	y	x && y
1	1	1
1	0	0
0	1	0
0	0	0

In **example** B, 'ǁ' operator returns the value of expression as true, if any one of the conditions is true. Or the operator will return the value false, only if both the conditions are false.

Truth table of logical operator 'ǁ'

x	y	x ǁ y
0	0	0
0	1	1
1	0	1
1	1	1

In **example C, !Operator** has the effect of negating or inverting a Boolean expression.

```
!0=1 and!1=0
```

Logical operators provide a way to simplify nested conditional statement.

Example D:

```
if 0<x {
    if x<10{
        document.write("x is a positive number");
            }
        }
```

can be rewritten **:**

```
if 0<x and x<10 {
    document.write( "x is a positive number");
                }
```

5.1.6.3 Relational Operators

Conditional statements allow you to check certain conditions and change the behavior of the program accordingly. The following are the relational operators used in JavaScript:

>, <, = =, = = =, !=, != =, <=, >=

x > y // x is greater than y

x < y // x is less than y

x= =y // x equals y

x= = = y // x strictly equals to y

x!= y // not equal to y

x!= = y // not strictly equal to y

x >= y // x is greater than or equal to y

x <= y //x is less or equal to y

Note: The = = and ! = operators perform type conversions before testing for equality. If ("7" = =7), evaluates to true. The = = = and ! = = do not perform type conversions before testing for equality. If ("7"= = =7), evaluates false. If (7 ! = =7), returns to true.

5.1.6.4 Assignment Operation

Assignment operator "=" is used to assign value to a variable. The following are the different form of assignment operators:

A=A+B ← A+=B

A=A-B ← A-=B

A=A*B ← A*=B

A=A/B ← A/=B

A=A%B ← A%=B

A and B are two variables and if we write the expression:

A=B, we are assigning the value of B to A.

5.1.6.5 Bitwise Operator

JavaScript defines several bitwise operators which can be applied only to the integer. The bitwise operators act upon the individual bits of their operands:

& Bitwise AND

| Bitwise OR

^ Bitwise Exclusive or

<< Left shift

>> Right shift

>>> Zero-fill right shift

Example '^' operator returns the value of expression as true, if one input is true and another is false. If both the inputs are same (either both are true or both are false), then the expression will return a false value.

Truth table of operator '^'

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

Example:

var a=2 has the binary form 010

var b=4 has the binary form 100

var c=a & b

If we print the value of c, the outcome will be 0. When we perform Bitwise operators in any expression, first we convert operands into binary form, then apply Bitwise operators. JavaScript does it automatically.

Example:

var a=2 (Binary form 010)

var b=4 (Binary form 100)

var c=a|b

If we print the value of c, the outcome will be 6.

Example:

var a=2 (Binary form 010)

var b=4 (Binary form 100)

var c=a^b

If we print the value of c, the outcome will be 6.

When we perform Bitwise operations in any expression, first we convert operands into binary form, then apply bitwise operators.

Left Shift: The left shift operator << shifts all the bits in a value to the left by a specified number of times.

Value<<num

where, num specifies the number of positions to the left shift .

a=35

a<<1

document.write(a)

70

Since, each left shift has the effect of doubling the original value, programmers frequently use this fact as an efficient alternative to multiply by 2 for each position shifts towards left.

Looking at the same operation in binary, it shows more clearly how this happens.

00100011 35

<<1

01000110 70

Right Shift: The right shift operator >> shifts all the bits in a value to the right by a specified number of times.

Value>>num

Where, num specifies the number of positions to the right shift.

a=35

a>>2

document.write(a)

8

Since, each right shift has the effect of doing half the original value, programmers frequently use this fact as an efficient alternative to divide by 2 for each position shift towards right.

Looking at the same operation in binary, it shows more clearly how this happens.

00100011 35

>>2

1000 8

>>> (Shift Right Zero Fill)

The shift right zero fill operator looks at the integer to the left of the operator as a 32-bit binary number. All the bits in this number are shifted to the right by the number of positions specified by an integer to the right of the operator. As the bits are shifted to the right, zeros are filled on the left, regardless of the sign of the original integer. The extra bits on the right are lost. The 32-bit binary result of this shifting operation is converted into an integer value and returned from the shift right zero fill operation.

```
<html>
    <script  language="JavaScript">
    <!—
    x = -6;
    x >>>= 2;
    document.write("x = ",x);
    —>
    </script>
</html>
```

In the above code, we have given input as -6 which is equivalent to binary number 11111111111111111111111111111010

After applying >>> to input, the output is:

1073741822

This is equivalent to binary number
00111111111111111111111111111110

5.1.7 Summary of Operators

Operator	Description
+	Adds two numbers or objects. e.g. 5 + 6=11and 'c'+ 'd' = 'cd' Here + is being used in two ways; one for the addition and other for concatenation of two strings.
–	Substract one number from the other
*	Multiply two numbers
/	Divide x by y
%	Returns remainder of the division. e.g. 7%5 = 2
<<	Shift bit of the number to the left by the no. of bits given. So, it is called a left shift.
>>	Shift bit of the number to the right by the no. of bits given. So, it is called a right shift.
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
~	Bitwise complement
<	Less than
>	Greater than
<=	Less than equal to
>=	Greater than equal to
==	Equal to
!=	Not equal to
!	Boolean NOT
&&	Boolean AND
||	Boolean OR
++	Pre/post increment
– –	Pre/post decrement
=	Assignment

5.1.8 Rules of Precedence and Associativity

When more than one operator appears in an expression, then the order of evaluation depends on the rules of precedence:

Highest

()	[]	.	function call array subscript
~	!	-	++ – – type of new void delete
*	/	%	
+	–		
>>		<<	
>	>=	<	<=
==	!=		
&			
^			
|			
&&			
||			
?:			
=	op=		
Comma (,) operator			

Lowest

When the precedence of two operators are same, the associativity will be from left to right. The evaluation of assignment operator is always from right to left. Parenthesis raises the precedence of the operations that are inside. This is often necessary to obtain the result as we desire. For example,

```
x>>y+8
x>>(y+8)
```

This expression first adds 8 to y, then shifts x right by that result. However, if we want to first shift x right by y positions and then add 8 to that result.

```
(x>>y) +8.
```

By adding parenthesis, we can reduce the ambiguity.

5.1.9 String Operator

String operators are used to perform operations on the strings. In a JavaScript, when operator + is operated on strings, it works as string concatenation operator. It is used to join two strings together.

"AR"+ "SD" produces "ARSD".

5.1.10 Ternary Operator

Ternary operator is the conditional operator (? :).This operator is called ternary because it takes three operands (a condition to be evaluated and two alternative values to be returned, based on the truth or falsity of the condition).

Syntax:

```
condition ? value 1 : value 2
```

If the condition is true, value 1 is the result of the expression otherwise, value 2 is the result.

Example:

```
(a>b) ? 10: 11
```

If the value of variable a is greater than the value of variable b, then 10 is the result of the expression. If the value of variable a is less than the value of variable b, then 11 will be the result of the expression.

5.1.11 Some Special Operators

Comma (,) operator: When two expressions are given and there is a comma operator between them, then this operator evaluates two expressions and returns the value of the second expression.

Example:

If X = (a + b), (c + d), then both the expressions will be evaluated and the value of second expression will be assigned to variable X.

typeOf operator: This operator returns a string value that identifies the type of an operand.

void operator: This operator returns no value.

new operator: It is used to allocate memory at the time of creating an object.

delete operator: It is used to de-allocate the allocated memory. It is also used to delete an element in an array index.

5.1.12 Input and Output Statements

5.1.12.1 User Input

Sometimes you need to ask for input from the user and process it. Following syntax is used to input from the user:

```
<input id = 'uid' size=70>
<button onClick= 'userSubmit( )'> submit </button><BR>
<P> <div id= 'xyz'<>/div>
```

In the above example, we have created an input field for the user input of size 70 and given it a value UID. We have created a HTML button with an onClick event that will call the function userSubmit( ).When the user will click the submit button , the onClick event will call the userSubmit( )function.

Since, JavaScript is an event driven language. So, whatever events we set up, our scripts will react accordingly. **In the above source code, onClick is an event**.

5.1.12.2 Output Statement

writeln( ) method:

Syntax:

```
document.writeln(string)
```

Above method is used to display the contents on computer screen. This output method can also be used while the page is being constructed. After the page has finished loading a new document, writeln (string) command will delete the page in the browser, so use this only while the page is loading. If you use this method after the page has loaded, the browser will destroy the page and start constructing a new one. Dynamic content during page load is better served by the server-side scripting languages.

Example:

```
<html>
    <head>
        </head>
        <body>
            <script  type='text/javascript'>
            document.writeln ('Hello-University of Delhi').
            </script>
        </body>
        </html>
```


alert( ): This is used for debugging purposes. Alert boxes will stop your scripts from running until the user clicks the OK button.

Example: Let's see the following code:

```
<html>
    <head>
        </head>
    <body>
            <script  type='text/javascript'>
                      alert(Hello University of Delhi);
            </script>
    </body>
</html>
```

getElementById( ): This is the most powerful output statement: Since, we know that everything on a web page resides in a box. For example—a paragraph (<p>) is a box. Another example of this is when you mark something as bold, you create a little box around that text which will contain the bold text. You can give each and every box in HTML a unique identifier, and javaScript can find boxes you have labeled and let you manipulate them. The only purpose of giving unique identification to each box is to check the code.

Example: Let's see the following code:

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<div id= 'xyz'></div>
<script type= 'text/javascript'>
document.getElementByID('xyz').InnerHTML = 'Hello ARSD';
</script>
</BODY>
</HTML>
```

In the above example, we have defined the box div and given id = 'xyz'. We are passing id no. to the method as an argument. For example getElementByID('xyz'). The purpose of this is that we can use the JavaScript to find that block and modify it. The left part of the statement says that on the web page (document), find a block we have named 'xyz'. We can modify the context of 'xyz' at any time, even after the page has finished loading. Inner HTML will process your string and basically redraw the web page within the context. This is supported by every major browser.

List of common events which can be attached to any HTML tag or form element as following:

Event	**Explanation**
onAbort ( )	Will be triggered when an image fails to load
onUnload( )	Triggered when the user is navigating away from a page
onSubmit( )	When a form's submit button has been pressed.
onSelect( )	When the text has been selected.
onResize( )	When the window or frame was resized.
onMouseUp( )	When the mouse button was released.
onMouseOver( )	When the mouse moved over this element.
onMouseOut( )	When mouse moved off this element.
onMouseMove( )	When the mouse moved.

onMouseDown( )	When button is pressed.
onLoad( )	When object (image, script, frame) finish loading.
onKeyUp( )	When a key is released.
onKeyPress( )	When a key is pressed or released.
onKeyDown( )	When a key is pressed.
onFocus( )	When the user just moved into this form element.
onError( )	When an error occurrs while loading an image.
onDoubleClick( )	When the user double clicks on the item.
onChange( )	When the contents of a field has changed.
onBlur( )	When user moves to another field.

5.1.13 Control Structures

Normally the statements in a program are executed sequentially i.e. one after another. This is known as normal flow of control. One of the keys which designs intelligent program is able to make the decisions. So, there is a possibility of changing the sequence of statements based on the decisions. In a JavaScript, curly braces forms the block of codes. Block of codes is the group of statements which executes simultaneously.

Control structures are used to control the flow of data during the execution of a program.

5.1.13.1 Iterative Control Statements

Iterative statements create loops. A loop executes the same set of instructions repeatedly until a termination condition is met.

For statement:

Syntax:

for (initialization; an evaluation; increment)

Example:

```
A= new array(3, 4, 5, 6, 8, 9);
S=0;
for (i=0; i<A.length;++i)
   S+=A[i]
```

In the above example, we have created an array of 6 elements. In the second statement, variable S is initialized to 0. In statement, firstly i is initialized to 0 and tests the length of A, to see if it is greater than i. Since, i is 0 and A.length is 6, the statement enclosed within the braces is executed and the value of s is incremented by A[i], which is 3. Then, the value of i will be incremented by 1.The condition is again tested and since i is less than 6, the statement enclosed within braces is then re-executed. This time s is incremented by 4 and its value becomes 7.

The steps are repeated until i<A.length is no longer true. The final value of s will be 35.

Example: Write a program to evaluate the function sin(6), using the series for the function sin(x):

$$\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

The following program will evaluate the value of sin(6):

```
<html>
<head><title>SUM OF THE SINE SERIES</title>
<script>

i=1;
den=1;
sn=-1;
x=6;
sum=x;
num=x;
for  (i=1;i<5;i++)
    {
    num=num*x*x;
    den=den*(2*i)*(2*i+1);
    sum=sum+(num/den)*sn;
    sn=-sn;
}
    document.write(  sum);

</script>
</head>
<body> </body>
</html>
```

Output:

6

and final sum is:

6

In the above example, first of all we have defined the variables used in the program. Since, we have to evaluate the sum of the sine series, we are taking a counter i and using that counter in for loop. The given "for loop" will iterate four times. Firstly, i will be initialized with 1, then it will check the condition. If the condition is satisfied, then the statements will be executed. Now, i will be

incremented by 1. Again i will be initialized with two. It is less than 5, so the statement will be executed. Now, the value of i will be incremented to 3. Again, i is less than 5; The statement will be executed. After execution, i will be incremented *i.e*. 4. Again i will be initialized with 4. Now, the condition will be checked. Since, this is less than 5. Again the statements will be executed. Now i will be incremented. Since, i is not less than 5 *i.e*. the condition is false. Now you will come out of the loop. After the fourth iteration, the final summed value will be printed *i.e*. 6.

Example: Calculate the factorial of numbers up to 9.

```
<html>
<head>  <title>Factorial</title>
<script>
 i=1;
f=1;
for  (i=1;i<10;i++)
 {
 f=f*i;
 document.write( i+'\t'+f);
 document.write('\n');
 }
</script>
</head>
<body> </body>
</html>
```


The output is:

1	1
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880

In the above example, we have initialized the variables used in the program. i is working as a counter in "for loop". As in the earlier example, first the value of counter will be initialized with i=1, then the condition will be checked *i.e*.1<10. If this is true, then the statement will be executed. Initial value of f is 1. The value of f will be substituted in the expression. The final value of f will be 1. After the execution of the statement, the counter will be incremented by one

i.e. 2. Now again the condition will be checked. Here, the condition is true. The final value of f is 2 now. This will continue until the condition is false. When the condition is false, you will come out from the loop. In the above program, we have used two statements:

```
document.write(i+'\t'+f);
document.write('\n');
```

where '\t' and '\n' are the escape sequences. Escape sequences are used to print the non printable characters. '\t' and '\n' has been used for eight character space between two values and new line respectively.

List of escape sequences:

Character	Meaning
\b	Backspace
\f	form feed
\n	new line
\r	carriage return
\t	Tab
\\	backslash character

Example: Write a program to compute the factors of a number.

Let us assume number n=20.

```
<html>
<head><title> Factors</title>
<script>
      i=0;
n=20;
x=0;
      for  (i=1;i<n+1;i++)
{
          if  ( n%i==0)
{
           x=i;
           document.write(x);
}
}
</script>
</head>
<body> </body>
</html>
```

Output

1 2 4 5 10 20

In the above example, firstly, we have defined the variables which will be used in the program. As in the earlier example, first the value of counter will be initialized with i=1, then the condition will be checked. If this is true, then the statement will be executed. The initial value of variable n is 20. This value will be substituted in the expression of if statement. If we divide 20 by 1, then the remainder is 0. The condition is true. So, the statement will be executed. Now, the value of x will be substituted in the expression, *i.e*. 1. The value of x will be printed as 1. This will continue until the condition is false. When the condition is false, you will come out from the loop. % operator is used for remainder. document.write() method is used to print the value in the same line.

Example: Write a program to find the sum of the series,

1+1/2+1/3+1/4+1/5+1/6+1/7+1/8+1/9

```
<html>
<head><title> Sum of the series</title>
<script>
    j=0;
    sum1=0.0;
    for( j=1;j<10;j++)
    {
        sum1=sum1+1.0/j;

    }
        document.write( sum1);
    </script>
    </head>
    <body> </body>
    </html>
```

Output:

2.82896825397

In the above example, firstly, we have defined the variables which will be used in the program. As in the earlier example, first the value of counter will be initialized with j=1, then the condition will be checked. If this is true, then the statement will be executed. The initial value of variable sum1 is 0.0. This value will be substituted in the expression. The final value of sum1 will be 1. After the execution of the statement, the counter will be incremented by one. Now, the value of j is 2. Since, 2<10, the condition is true. So, the statement will be executed. Now, the value of sum1 will be substituted in the expression. The final value will be assigned to sum1. This will continue until the condition is false. When the condition is false, you will come out from the loop and the final

value of sium1 will be printed *i.e.* 2.82896825397. % operator is used for remainder. document.write() method is used to print the value.

Example: Program to check which members from 2 to 10 are prime numbers.

```
<html>
<head><title> </title>
<script>
for(t=2;t<10;++t)
    {
    for(y=2;y<t;++y)
        {
            if (t%y ==0)
            document.write( t+ "is not a prime number");
            document.write('\n');
                break;
            else
                document.write( t+ "is a prime number");
                document.write('\n');
        }
    }
</script>
<head>
<body> </body>
<html>
```


Output

3 is a prime number
4 is not a prime number
5 is a prime number
6 is not a prime number
7 is a prime number
8 is not a prime number
9 is not a prime number
10 is not a prime number

In the above example, firstly, we have defined the variables which will be used in the program. As in the earlier example, first the value of counter will be initialized, then the condition will be checked. If this is true, then the statement will be executed. After the execution of the statement, the counter will be incremented by one. This will be continued unless the condition is false. In this example, we are using two for loops. We are nesting one for loop within another. For each iteration of first for loop, all the iterations of the second for loop will

be executed. When the condition is false, you will come out from the loop. % operator is used for remainder. document and write() method is used to print the value. Escape sequence '\n' is used for new line character. In the first for loop, when the value of t is 2 and 2<10 the condition is true. Control will go to second for loop. The value of y will be initialized with 2. Now, the condition of second for loop will be executed. 2<2 *i.e*. condition is false. So, second for loop will not be executed. Now, the value of t in first for loop will be incremented to 3. Since, 3<10, the condition is true. For the value of t in the first for loop, the value of y, in the second for loop will be 2. Since, 2<3, the statements within the second for loop will be executed. Now, if condition will be checked. Now, if statement will be checked. When 2 is divided by 2, the remainder is 0. It means the value is true. If 3 is divided by 2, then the remainder is 1. Means, the condition is false. So, else part will be executed and the message that 3 is a prime number will be printed. Now, the value of y in second for loop will be incremented. Condition will be checked. The condition is false. So, again the control will go to first for loop. Now, the value of t will be incremented to 4. Again the condition will be checked, *i.e*. true. Control will now go to second for loop. The execution will be continued in the similar fashion. And the output will be produced.

Example:

Program to print sum of squares of odd numbers:

```
<html>
<head><title> </title>
<script>
s=0;
for (num=1; num<8;num++)
   {
      if (num%2!=0)
      s=s+num*num;

   document.write(s);
}

</script>
</head>
<body> </body>
</html>
```

Output is:

84

In the above example, first of all we have defined the variables which will be used in the program. As in the earlier example, first the value of counter will be initialized, then the condition will be checked. If this is true, then the statement will be executed. After the execution of the statement, the counter will be incremented by one. This will be continued unless the condition is false. When the condition is false you will come out from the loop. % operator is used for remainder. document.write() method is used to print the value. If the value of num is 1, then the statement will check whether the remainder is zero or not. In this case the remainder is 1. So, the condition is true. The 0 value of s will be substituted into the expression. The final value of s will be 1. Now, the value of num will be incremented. Here, 2 is less than 8. The condition is true. If we divide 2 by 2, then the remainder is 0; the condition is false. So, the statement within if will not be executed. Now, again num will be incremented by 1 and the value of num will be 3. Again, the condition will be checked. 3 < 8 *i.e.* true. Again, if condition will be checked; since the remainder is 1 here; the condition is true. The value of s will be 10. The execution will be continued in the same fashion until and unless the value of num is less than 8. The final value of sum will be 84.

5.1.13.2 for / IN

JavaScript has a changed form of for loop while dealing with JavaScript objects.

Syntax:

```
for (variable name in object name )
{
   set of statements;

}
```

Consider the following example:

```
var x = { ' B ' : ' Banarasi saree'
    'O': 'Online American'
    'T' : 'Tiruvantpuram'}
for ( var f in x )
     { document . writeln (f + 'means' +x [ f ] + '< br >');
     }
```

Output:

B means Banarasi Saree

O means Online American

T means Tiruvant puram

5.1.13.3 while

while loop will continue to execute until its test condition evaluates to false or the loop encounters a break statement.

Syntax:

```
While (condition)
{set of statements ;}
```

Example: Program to generate log table of integer numbers.

```
<html>
   <head> <title> </title>
   <script>
          x=1.0;
          while (x<5.0)
      {
             document.writeln( x+ '\t'+Math.log(x));
             x=x+1.0;
      }
</script>
</head>
<body> </body>
</html>
```

We have initialized the variable x with value 1.0. The control will go to the while loop. In while loop, the condition will be checked. If it returns true, then the statements will be executed. In the above example, 1 < 5 *i.e.* true. Writeln method prints the value of x and the log value of x. There will be eight character space between x and math.log(x). Where '\t' is an escape sequence, which is a nonprintable character. \t represents a tab character; you can include escape sequence any where in a string. Now, the value of x will be incremented by one that is 2.0. Again, the value of x will be retested and returns true. Again, the values will be printed. The value of x will be increased by one and so on. This process will be continued until the condition in the while loop becomes false.

The output of the above program will be as follows:

1.0	0.0
2.0	0.69314718056
3.0	1.09861228867
4.0	1.38629436112
5.0	1.60943791243

Applications: Probability, statistical and mathematical computations.

Example:

program to generate Fibonacci series up to 10.

```
<html>
<head><title> </title>
<script>
x=0;
y =1;
while (y<10)
    {
        document.write( y);
        document.write('\n');
        x=y;
        y=x+y;
    }
</script>
</head>
<body> </body>
</html>
```

Output:

1
1
2
3
5
8

We have initialized the variable x and y with value 0 and 1 respectively. x and y variables are working as previous and next terms respectively. In Fibonacci series, the sum of previous term and next term will be the value of next to next term. The control will go to the while loop. In while loop, the condition will be checked. If it returns true, then the statements will be executed. In the above example 1<10 i.e. true. Write method prints the value of y. In the line number 9, '\n' is an escape sequence, which is a nonprintable character. \t represents a new line character. You can include an escape sequence any where in a string. Now, the value of y will be assigned to x and the value of x+y will be assigned to y. The statements between the curly braces will be executed again and again until the condition in the while loop becomes false.

Applications: The Fibonacci series has practical application in botany, electrical network theory, sorting and searching.

5.1.13.4 do while

Syntax:

```
do { set of statements}
while ( condition) ;
```

There is one difference between while and do while. That is, in the while loop the condition will be checked, first then the statements will be executed. In the case of do, at least one or more than one time the statement will be executed, then the condition will be checked.

Example: Program to reverse a given number.

```
<html>
<head><title> </title>
<script>
n=345678;
do
   {
      document.write( n%10);
      document.write('\n');
      n=n/10;
   }while(n!=0);
</script>
</head>
<body> </body>
</html>
```


We have initialized the variable n with value 345678. Now, the control will go to the 7th line. The value 8 will be printed. Write method, prints the value of n%10. In the line number 8, '\n' is an escape sequence, which is a nonprintable character.'\n' refers to a new line character. In the line number 9, the expression n=n/10 gives the quotient value *i.e*. 34567.The control will go to the while statement. In while statement, the condition will be checked. If it returns true, then the control will go to the do statement. In the above example 34567!=0 *i.e*. true. The statements between the curly braces will be executed again and again until the condition in the while loop returns false.

Applications: Hashing and information retrieval, database applications.

The output is:

8
7
6
5
4
3

Example: Write a program that finds the sum of the digit of a given number:

```
<html>
<head><title> </title>
<script>
a=0;
sum=0;
num=234567;
do
       {
       a=num%10;
       num=num/10;
       sum=sum+a;
   }while(num!=0);
document.write("Sum of digits"+sum);
</script>
</head>
<body> </body>
</html>
```

We have initialized the variable num with value 234567. We have initialized variables, a sum with value 0s. Now the control will go to the 9th line. The value 7 will be printed. Write method prints the value of n%10. In the line number 10, the expression num=num/10 gives the quotient value *i.e*. 23456. Initial value of sum is 0. After assigning the value of sum+a, the final value of the sum will be 7. Now the control will go to the a while statement. In a while statement, the condition will be checked. If it returns true, then the control will go to the do statement. In the above example 23456!=0 *i.e*. true. The statements between the curly braces will be executed again and again until the condition in the while loop returns false.

Output:

Sum of digits 27

Example: Write a program to convert Decimal number into Binary:

```
<html>
<head><title> </title>
<script>
a=1;
bin=0;
num=45;
do
```

```
{
    rem=num%2;
    bin=bin+rem*a;
    num/=2;
    a*=10;
    document.write( bin);
}while(num>0);
</script>
</head>
<body> </body>
</html>
```

Output is:

101101

We have initialized the variable num with value 45. We have initialized variables, a bin with 1 and 0 respectively. Now the control will go to the 9th line. The value 1 will be stored into rem variable. In the line number 10, the expression bin=bin+rem*a. Initial value of bin is 0. After assigning the value of bin+rem*a, the final value of rem will be 1. In the line number 11, the quotient value will be stored into the num variable. In the line number 11, the value of a will be multiplied by 10. In the line number 12, the value of a bin will be printed. Now the control will go to the while statement. In a while statement, the condition will be checked. If it returns true, then the control will go to the do statement. In the above example 22>0 *i.e.* true. The statements between the curly braces will be executed again and again until the condition in the while loop returns false.

Applications: Interpretation of stored computer data and instructions.

5.1.13.5 Conditional statements

We apply conditions in our program through IF, ELSE IF, ELSE statements. "if statement" is JavaScript's conditional branch statement. It can be used to route program execution through two different paths. If the statement allows us to control the flow of execution based upon conditions known only during run time. The "IF" statement enables the programmer to selectively execute statements based on a given condition. The code block governed by the "IF'' is executed, if the condition is true.

Syntax:

```
if ( expression )
{
Statement 1;

}   7
```

```
else if ( expression )
{
Statement 2;
}
else
statement;
```

There can be zero or more else, if parts and the else part is optional.

Example:

```
<html>
<head><title> </title>
<script>
          X=12;
          Y=13;
          if (x<y)
          {
             document.write( x+ "is less than "+ y);
          }
          else if (x>y)
          {
             document.write(x+ "is greater than"+ y);
          }
          else
             document.write( x+ "and"+ y +"are equal");
</script>
</head>
<body> </body>
</html>
```

In the above example, first of all we have defined the variables x and y with value 12 and 13 respectively. Now the control will go to "if statement". If the value of expression is true, then the statement will be executed, otherwise else if part will be executed. If expression in the else if part will return true,then the statement will be executed, otherwise else part will be executed. In the above example, if the value of x is less than the value of y, then 8th line will be executed; if the expression returns false, "else if" will be executed; if this is again false "else part" will be executed. So, the very first expression returns the true value. The following will be the output:

Output: 12 is less than 13

Lets retest, if statement with the help of following two examples:

```
<html>
<head><title> </title>
<script>
          X=13;
          Y=12;
          if (x<y)
          {
             document.write( x+ "is less than "+ y);
          }
          else if (x>y)
          {
             document.write(x+ "is greater than"+ y);
          }
          else
             document.write( x+ "and"+ y +"are equal");
</script>
</head>
<body> </body>
</html>
```

Output: 13 is greater than 12.


```
<html>
<head><title> </title>
<script>
          X=12;
          Y=12;
          if (x<y)
          {
             document.write( x+ "is less than "+ y);
          }
          else if (x>y)
          {
             document.write(x+ "is greater than"+ y);
          }
          else
             document.write( x+ "and"+ y +"are equal");
</script>
</head>
<body> </body>
</html>
```

Output: 12 and 12 are equal

Example: To check whether a given number is even or odd.

```
<html>
<head><title> </title>
<script>
       x=24;
              if (x%2 ==0)
              {
                     document.write( x+ " is even");
              }
              else
                     document.write(x+ "is odd");
</script>
</head>
<body> </body>
</html>
```

Output: 24 is even

Example: To check whether a given number is even or odd.

```
<html>
<head><title> </title>
<script>
 x=25;
          if (x%2 ==0)
   {
              document.write( x+ " is even");
   }
          else
              document.write(x+ "is odd");
</script>
</head>
<body> </body>
</html>
```


Output: 25 is odd

In the above two examples, if the number x is divisible by 2, then 7th line will be executed otherwise else part will be executed. If we divide any number by 2 and remainder is 0, then the number will be even, otherwise odd.

Example: Write a program to calculate gross salary of employees in an organization. If the basic salary of the employee is less than 15000, then HRA is 10% of the basic and DA will be 90% of the basic. Otherwise HRA=500 and

DA will be 98% of the basic. Calculate the gross salary of an employee using the formula, gs=bs+hra+da. In the program given below bs, hra, da, gs stands for basic salary, house rent allowance dearness allowance and gross salary respectively:

```
<html>
<head><title> </title>
<script>
bs=8000;  //Basic salary of an employee
if (bs<1500)
    {
    hra=bs*10/100; //House rent Allowances
    da=bs*90/100; //Dearness Allowance
    }
else
    {
    hra=500;
    da=bs*98/100;
    }

gs=bs+hra+da; # Gross salary
document.write( gs); #Print gross salary
</script>
</head>
<body> </body>
</html>
```


Output is:

16340

One if statement can be nested within another if statement.

See the following example:

```
<html>
<head> <title> </title>
<script>
 x=12;
 y=13;

          if (x= =y)
              {
              document.write(x+"and"+ y+ "are equal");
              }
          else
```

```
                {

                if (x<y)
                    {
                    document.write(x+ "is less than"+ y);
                    }
            else
                    {
                document.write(x+ "is greater than"+ y);
                    }
                }
</script>
</head>
<body> </body>
</html>
```

Output: 12 is less than 13.

5.1.13.5.1 *Conditional Switch*

If you have to do a large number of tests, it makes more sense to use a switch statement instead of nested ifs:

Syntax:

```
switch (x)
       {
          case 1:{ }
          case 2: { }
          .

          default: { }
       }
```

The switch statement evaluates the expression and determines if any of the values match the expression's value. If one of them matches, then the statements of that particular case are executed, and the statement execution continues after the switch statement. If there is no matching value, then the statements for the default case are executed.

The break statements may be omitted. If they are omitted, execution continues with next case.

Example: Creation of menu:

```
document.write  ("***MENU:***<br>");
document.write  ("0:QUIT<br>");
```

```
document.write  ("1:add<br>");
document.write  ("2:multiply<br>");
var ch;
c=window.prompt ("enter your choice(-1 to quit)", "0");
ch=parseInt(c);

while (ch!=0)
{
switch (ch)
{
case 1:add();
break;
case 2:multiply ( );
break;
default: document.write ("br>wrong choice!<br>");
}
c=window.prompt ("enter your text choice (0 to quit)", "0");
 ch=parseInt(c);
}
```

Note: parseInt ( ) function converts character data type into Number data type.

5.1.14 The Label Statement

Any statement may be labeled, by placing label name before the statement.

Syntax:

```
Labelname:
Statement
```

Example:

```
X:
y= 2*x
```

The above code causes to be labeled with X.

5.1.15 with Statement

It is used to eliminate retyping the name of an object, that is to be referenced in a series of property references and method invocations.

Syntax:

with (variable name)

{ statements }

The variable name identifies the default object to be used with the statements enclosed in braces.

Example:

```
with ( x )
{write ( " < H1> Hello < /H1>" )
write ( " <P>" )
}
```

The need to prefix each write ( ) method invocation with the x object is eliminated because x is identified in with statement. Without with statement, the above code needs to be written as:

```
x. write( " <H1> Hello </H1>" )
x.write ("<P>")
```

5.1.16 Jump Statements

Jump statements transfer control from one part of the program to another part. JavaScript supports three jump statements i.e. break, continue and return.

5.1.16.1 Break Statement

By using the Break statement, you can terminate a loop by immediately passing the conditional expression.

Example: In the code given below, for loop will be executed 100 times but at line 6 we are using the break statement which means, when the value of *i* is equal to 10, we will come out of the loop. The values from 0 to 9 will be printed.

```
i=0;
for (i=1;i<100;++i)
{
       if (i==10)
              {
              break;
              }

       document.write( "I is "+i);
       i=i+1;
}
```


The output is:

I is 0
I is 1
I is 2
I is 3
I is 4
I is 5
I is 6
I is 7
I is 8
I is 9

5.1.16.2 Continue Statement

Sometimes it is useful to force an early iteration of a loop. Which means you might want to continue running the loop, but stop processing the remainder of the code in its body for that particular iteration

Example:

```
for (i=0;i<10;++i)
      {
      document.write(i);
      if (i%2==0)
         continue;
      document.write( '\n');
      }
```

Output

0 1
2 3
4 5
6 7
8 9

5.1.16.3 return

The return statement is used to return value from a function explicitly.

5.1.17 Arrays

Arrays are objects that are capable of storing a sequence of values. These values are stored in indexed locations within an array. Suppose in a class, there are 4 students and you want to display the name of the students on a webpage, you can keep a track of their names in an array variable named student.

An array must be declared before it is used. An Array may be declared using either of the following two statement forms:

(*i*) array name = new Array (array length)

(*ii*) array name = new Array( )

student = new array(4)

and store the name of the students in the array using the following statements:

student [0]= “Aakash”
student [1]= “Rahul”
student [2]= “Mehraj”
student [3]= “Rekha”

You can access the names of the individual students by referring to the individual elements of the array.

Example: You can also display the names of your students using statements such as the following:

```
document.write ( student[ 0 ])
document. write ( student[ 1 ])
document. write ( student[ 2 ])
document. write ( student[ 3 ])
```

In the second array declaration, the length of an array is not specified and the results in declaration of an array of length 0.

```
y= new array ( )
```

declares an array of length 0.

JavaScript automatically extends the length of an array when new array elements are initialized.

```
y= new array ( )
y[56]= "hello"
```

When JavaScript encounters the reference to y[56] in the above example, it extends the length of the array to 57.

5.1.18 Function

A function is a special type of relation. A function plays an important role in computer science. A function states the relation between two distinct objects or an object with itself. A function is also known as mapping or transformation**.** A function is a named sequence of statements that performs a desired operation which is specified in the function definition. Mathematically, in any expression operands would be treated as parameters and operators as functions. For example, if the expression is a + b, then operands a and b will be treated as parameters and operator + is a function.

Syntax:

```
<HEAD>
<SCRIPT language="JavaScript">
<!—
function name (List of formal parameters)
{
 JavaScript Statements and declarations
}

//—>
</SCRIPT>
</HEAD>
```

1. **function**
   Indicates that you are going to create a function.
2. **name**
   This is the name you give to the function.
3.
   The parameters are variables that are sent to the function when it is called. You can have, one , two, three parameters, and so on.
4. **{**
   This symbol lets you add JavaScript statements and declarations.
5. **}**
   This indicates the end of the statements, and the end of the function.

There can be zero or more parameters which are separated by commas. The list of parameters is optional. Parameters provide the information to be used inside the function. The list of parameters is called formal parameters. We can call the function.

Note: The variables written after the function name, in a statement are often called the formal parameters of the function. The values supplied when you call the function, are called the actual parameters.

To make use of the function, you have to make a call to the function. You call a function by using the name, any parameters you want to send, and a semicolon, like this:

```
function xyz (sum, value);
```

where, sum and value both are the actual parameters.

The same function can be repeatedly called. Also, one function can call another function.

The advantage of using a function is that it makes the program smaller by eliminating the repetitive code.

5.1.18.1 Variable Scope within a Function

While defining a function, it is often necessary to define variables that will be used to store values calculated by the function. Such variables are known as local variables. Local variables are accessible only within the function in which they are declared. Local variables are declared in the same manner as global variables, except that the local variable declarations are preceded by the keyword var. Local variables may have the same name as global variables. If the local variable and global variable have the same name, all references to the variable name within the function, that defines the local variable refer to the local variable of that function and not to the global variable.

5.1.18.2 Mathematical Functions

- *abs()* - Returns the absolute value of a number
- *acos()* - Returns the arccosine (in radians) of a number
- *asin()* - Returns the arcsine (in radians) of a number
- *atan()* - Returns the arctangent (in radians) of a number
- *atan2()* - Returns the arctangent of the quotient of its arguments
- *ceil()* - Returns the smallest integer greater than or equal to the number
- *cos()* - Returns the cosine of a number
- *exp()* - Returns the E^{number}, where number is the argument, and E is Euler's constant, the base of the natural logarithms.
- *floor()* - Returns the largest integer less than or equal to a number
- *log()* - Returns the natural logarithm (base E) of a number.
- *max()* - Returns the greater of two numbers
- *min()* - Returns the lesser of two numbers
- *pow()* - Returns base of the exponent power, that is $base^{exponent}$
- *random()* - Returns a pseudo-random number between 0 and 1
- *round()* - Returns the value of a number rounded to the nearest integer
- *sin()* - Returns the sine of a number
- *sqrt()* - Returns the square root of a number
- *tan()* - Returns the tangent of a number

Example:

```
//Returns the sine value of a given number
function getSine(x) {
 return Math.sin(x);
}
```

Example:

```
// Returns logarithmic value of a given number

function getLog(x){
 return Math.log(x);
 }
```

Example: Write a program to generate random numbers and remove duplicates

```
<html>
<head>
<script>
sz=window.prompt ("enter the array size:");
n=parseInt (sz);
```

```
var a=new Array(n);
for (i=0; i<n;i++)
{
a[i]=Math.round(1+Math.radom( )*6);
}
document.write ("The array is:");
document.write ("<br>");
for (i=0; i<n; i++)
document.write (a[i], " ");

document.write ("<br>");

for (i=0; i<; i++)
for (j=i+1; j<n; j++)
{
if (a[i] = = a[j])
{
a[i]=" ";
}
}
document.write("The array after removing duplicates is:");
document.write ("<br>");
for (i=0;i<n;i++)
document.write (a[i]," ");
</script>
</head>
<body>
</body>
</html>
```

Output

The array is:

4 2 6 1 4

The array after removing duplicate is:

2 6 1 4

5.1.18.3 Some Other Important Functions

Some of the most useful methods provided by the window object are alert(), confirm() and prompt(). These methods support dialog with the user.

alert (): Displays a dialog box containing a message and an OK button. This method is used to provide the user the critical information that must be acknowledged, by means of OKbutton.

confirm(): It produces a dialog box with a message , an OK button , and a cancel button. It returns true, if the user clicks the OK button and false, if the user clicks the cancel button. Confirm() method is used to inform the user and ask him to confirm whether he wants to perform a particular action.

prompt(): It displays a message to the user and prompts the user to type the information into a text field. It provides the capability to display default text in the text field. Prompt() method is used to obtain text input from the user such as a name or a URL. The value entered by the user will be the return value of the prompt method.

Let's see the following example:

```
<HTML>
<HEAD>
<TITLE>To see the working of methods</TITLE>
<script>
a=window.prompt ("enter the number of terms to be generated");
b=parseInt(a);
document.write(b);
window.confirm("enter the number of terms to be generated");
window.alert ("enter the number of terms to be generated");
</script>
</HEAD>
<BODY> </BODY>
</HTML>
```


Prompt:

Figure 5.1

CONFIRM

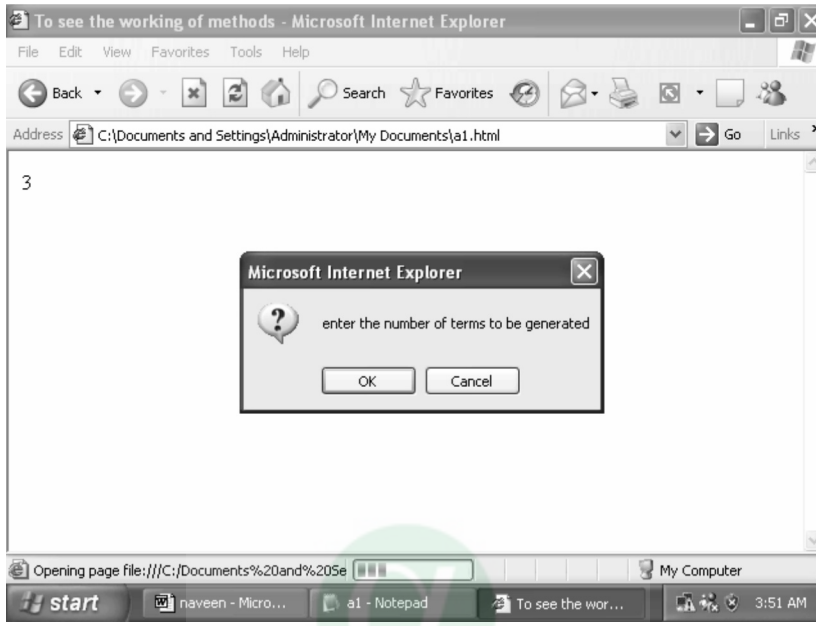


Figure 5.2

ALERT

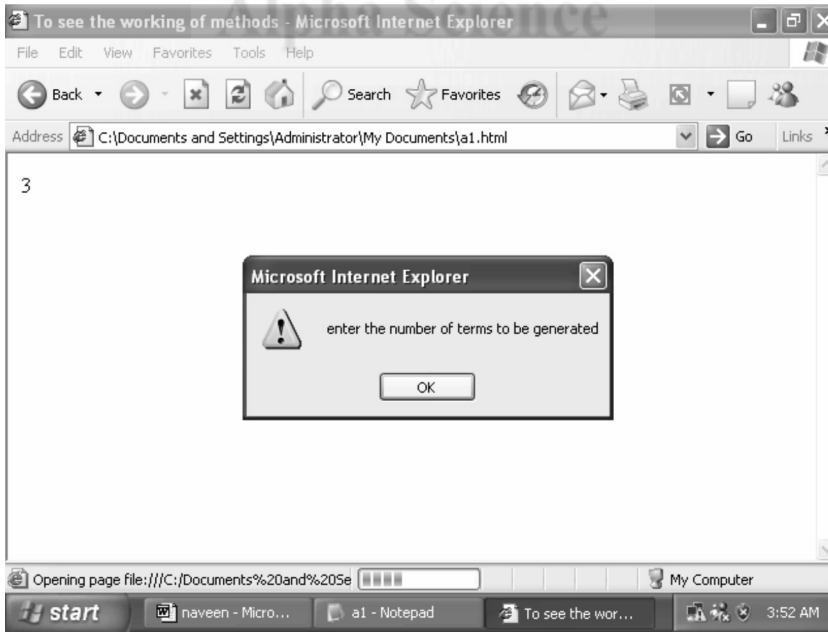


Figure 5.3

5.1.18.4 Conversion Functions

Functions are collection of JavaScript code that perform a particular task, and often return a value. A function may take zero or more parameters. These parameters are used to specify the data to be processed by the function. JavaScript provides three functions which are used to perform explicit type conversion.

eval( ): can be used to convert a string expression to a numeric value .

e.g.: x = eval (" 412+1")

eval () function returns the numeric value 413 assigned to x variable.

parseInt( ): It is used to convert a string value into an integer. parseInt ( ) returns the first integer contained in the string, or 0 if the string does not begin with an integer

Examples: parseInt ("abc") returns 0.

parseInt ("456abc") returns 456.

The parseInt () also parses hexadecimal and decimal integers.

parseFloat( ): It returns the first floating point number contained in a string, or 0 if the string does not begin with a valid floating point number.

Examples: parseFloat("2.1e4xyz") returns 21000 and parseFloat ("xyz") returns 0.

5.1.18.5 JavaScript Objects

Since, Java is an object based scripting language. In JavaScript, every thing is in terms of objects. Every object contains pre-defined methods to perform the capabilities. In the following figure, we have shown the hierarchy of objects that reflects the structure of the HTML page itself. The pre-defined objects that are most commonly used are the window and document objects. The window object contains methods through which you can create new windows with the open() and close() methods. You can also create message boxes using alert(), confirm(), and prompt() through window object, as we have discussed earlier.

The document object refers the current web page. The document object contains arrays which are the container of all the components constituting the contents of your web page, such as images, links, and forms. You can access and call methods on these elements of your web page through the arrays. The objects in this pre-defined hierarchy can be accessed and modified. To refer to specific properties, you must specify the property name and all its ancestors, spelling out the complete hierarchy. A period, '.', is used in between each object and the name of its property. Generally, a property / object gets its name from the NAME attribute of the HTML tag. Consider the following statement:

document.vsdform.mytext.value

Where document is the object which refers to the current document, vsdform is the name of the form, mytext is the name of the text field and value is the property of the text field.

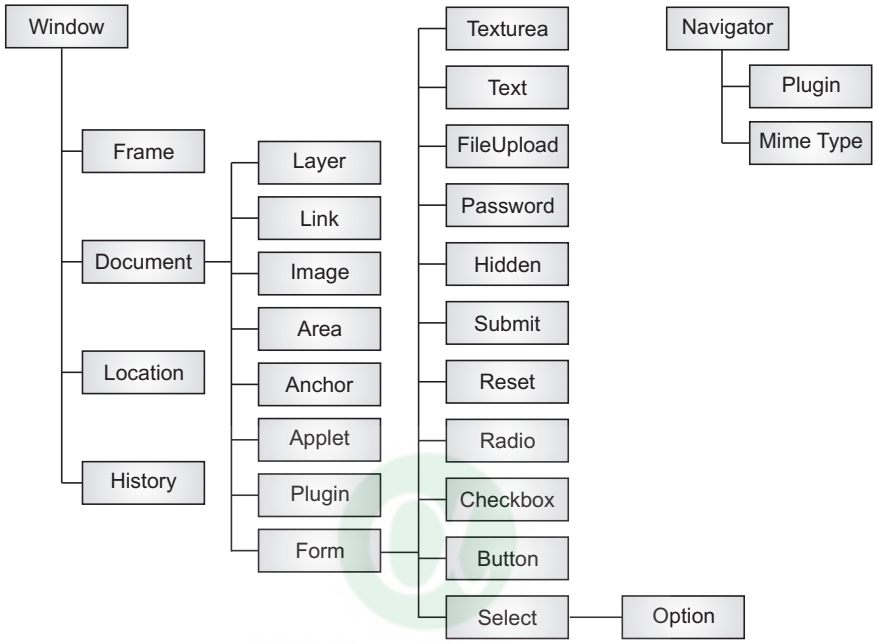


Figure 5.4

5.1.18.6 Date and Time Functions

There is a **DATE class** in JavaScript which is predefined. All the functions related with date and time is defined within the DATE class. The Date class works like the Array class. You will have to create a new object of the Date class, then use various built in methods to extract the information you want. For example, to return the current month, you use the **getMonth()** method; to return the four-digit year, you use the **getFullYear()** method; to return the current hour, you use the **getHours()** method, and so on. The implementation of the DATE class functions is given below:

```
<html>
<head><title> </title>

<script >

var t = new Date(); // t stands for time
var y = t.getYear(); // y stands for year
//Month is getMonth() + 1 because computer scientists
//start counting at 0. January is traditionally 1, so
```

```
//add 1 to correct this problem.
var m = t.getMonth() + 1;
var d = t.getDate(); //d stands for day
var h = t.getHours(); // h stands for hour
var min = t.getMinutes(); // min stands for minutes
var D = m + '/' + d + '/' + y; //D stands for date
var t = h + ':' + min;
document.write("The current date is " + D + "<br>");
document.write("The current time is " + t + "<br>");
</script>
</head>
<body> </body>
</html>
```

Output:

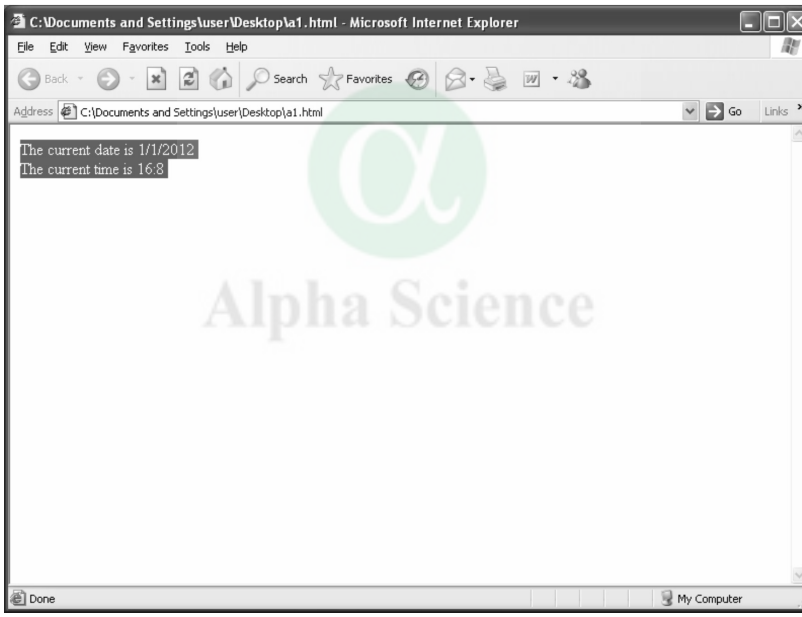


Figure 5.5

It is also possible to retrieve the day-of-week with **"getDay()"**. Let's see the following code:

```
<html>
<head><title> </title>
<script >

   var t = new Date(); // t stands for time
   var y = t.getYear(); // y stands for year
   var m = t.getMonth(); //m stands for month
```

```
    var d = t.getDate(); //d stands for day
    var wd = t.getDay();//wd stands for day of the week
    var h = t.getHours(); //h stands for hours
    var min =t.getMinutes();//m stands for minutes
    var ampm = "am";
    var Months = new Array();
    var WeekDay = new Array();

     Months[0] = "January";
     Months[1] = "February";
     Months[2] = "March";
     Months[3] = "April";
     Months[4] = "May";
     Months[5] = "June";
     Months[6] = "July";
     Months[7] = "August";
     Months[8] = "September";
     Months[9] = "October";
     Months[10] = "November";
     Months[11] = "December";
     WeekDay[0] = "Sunday";
     WeekDay[1] = "Monday";
     WeekDay[2] = "Tuesday";
     WeekDay[3] = "Wednesday";
     WeekDay[4] = "Thursday";
     WeekDay[5] = "Friday";
     WeekDay[6] = "Saturday";

 if (h> 12)
    {
    ampm = "pm";
    }

    if (min < 10)
    {
    min = "0" + min;
    }

document.write("vsd Date " + WeekDay[wd] + ", "+
       Months[m] + " " + d + ", "
       + y + "<br>");
document.write("vsd Time " + h + ":" + min + " " +
       ampm + "<br>");
```

```
</script>
    </head>
    <body>
    </body>
    </html>
```

Output:

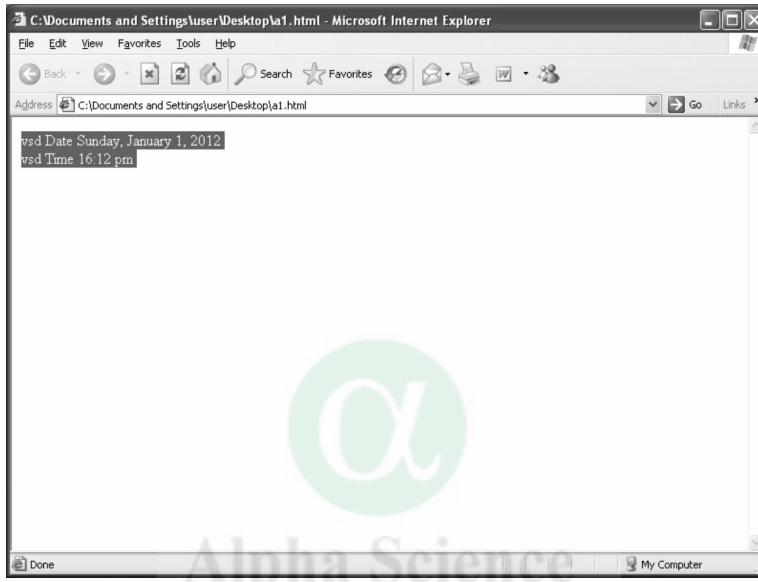


Figure 5.6

5.1.18.7 String Functions

There is a STRING class in JavaScript which is predefined. All the functions related with string operations are defined within the STRING class.

Now you will see the implementation of various 'String' Class methods:

toLowerCase () ——————Changes the string into lower case
toUpperCase ()——————Changes the string into upper case.
lastIndexOf ()——————Gives the last index of a string.
indexOf ()——————Gives the first index of a string.
split ()——————splits the string.
slice (First , second) —— Slices the string from first position to second position.
chartAt (position)——————Gives the character at a given position.
charCodeAt (position)——————Gives the character code at given position.
concat (string)——————concatenating the string

Now we are going to use the above string functions in the following program:

```
<html>
<head>
<script language="javascript">
var str="javascript is user friendly ";
function button_pressed()
{
searchForm.first.value =
str.indexOf(searchForm.inputVal.value);

searchForm.sec.value =
str.lastIndexOf(searchForm.inputVal.value);
searchform.third.value =
str.toLowerCase(searchform.inputval.value);
searchForm.fourth.value =
str.toUpperCase(searchForm.inputVal.value);
searchForm.fifth.value = str.indexOf
(searchForm.inputVal.value,3);
searchForm.sixth.value =
str.lastIndexOf(searchForm.inputVal.value,3);
searchForm.seventh.value =
str.split(searchForm.inputVal.value);
searchForm.eight.value = str.slice(5,7);
searchForm.ninth.value = str.chartAt(4);
searchForm.tenth.value = str.charCodeAt(10);
searchForm.ele.value = str.concat("language.");
searchForm.twe.value = substr.slice(5,10);
}
</script>
</head>
<body>
<hl>javascript is user friendly</h1>
<form name="searchForm">
<p>enter the substring to search for
<input name="inputVal" type="text" size=10>
<input name="search" type="button" value="search"
onClick="button_pressed()">
</p>
<p>first index located at:
<input name="first" type="text" size ='25'><br>
<p>Last index located at:
<input name="sec" type="text" size='25'><br>
<p>to lower case:
<input name="third" type="text" size='25'><br>
```

```
<p>to upper case:
<input name="fourth" type="text" size='25'><br>
<p>first index starting from index3:
<input name="fifth" type="text" size='25'><br>
<p>last index starting from index3:
<input name="sixth" type="text" size='25'><br>
<p>after splitting the string at substring:
<input name="seventh" type="text" size='25'><br>
<p>after slicing the string from 5 to seven:
<input name="eight" type="text" size='25'><br>
<p>after charAt(4):
<input name="ninth" type="text" size='25'><br>
<p>after charCodeAt(10)
<input name="tenth" type="text" size=' 25'><br>
<p>after concatenating "language" to string:
<input name="ele" type="text" size='25'><br>
<p>after substr(5, 10)
<input name="twe" type="text" size='25'><br>

<br>
</form>
</body>
</html>
```


Output

The string to be searched is:

Javascript is user friendly

Enter the substring to search for script — search

First index located at: 4

Last index located at: 4

To lower case: javascript is user friendly

To upper case: JAVASCRIPT IS USER FRIENDLY

First index starting from index3: 4

Last index starting from index: -1

After splitting the string at substring: java,t is user friendly

After slicing the string from 5 to seven: cr

After charAt(4): s

After charAt(10): 32

After concatenating "language" to string: javascript is user friendly language

Afte substr (5, 10): cript is u

Let's see another example of how to change String to lowercase: toLowerCase() method changes string into lower case.

```
<html>
<title>string into lower case</title
<head>
<script>
t=window.prompt ("enter the string :");
document.write ("the string is :");
document.write ("<br>");
document.write ( t );
document.write ("br>");
document.write ("After changing into lower case :");
document.write ("<br>");
document.write (t.toLowerCase ());
</script>
</head>
<body>
</body>
</html>
```


Output

The string is:
ARSDCOLLEGE
After changing into lower case:
arsdcollege

5.2 ACTIVE SERVER PAGES

Key Points

- ASP is the acronym for Active Server Pages.
- ASP is a specification for a dynamically created Web page.
- By creating dynamic web pages, network traffic can be reduced.
- ASP file contains text, HTML, XML, and scripts also.
- ASP file must have an .asp file extension.
- ASP file can be created using a simple text editor.
- ASP is a program that will run inside IIS or PWS.

- IIS stands for Internet Information Server.
- PWS stands for Personal Web Server.
- ASP is a Microsoft Technology.
- ASP is a program which runs inside IIS.
- Scripts in an ASP file are executed on the server only.
- ASP is different than HTML as per the following:
  - When HTML file is requested by Browser, the Server returns the file.
  - When ASP file is requested by Browser, Internet Information Server passes the request to the ASP engine. ASP file is read by the ASP engine on the line by line basis and executes the scripts in the file. Ultimately, Browser receives ASP file from ASP engine as plain HTML.

A Web Server Technology that allows the creation of dynamic, interactive sessions with the user is known as Active Server Pages. We can create a Web Page using ASP by containing HTML code and an embedded programming code written in VBScript or JScript. It was introduced with Version 3.0 of Microsoft's Internet Information Server. When Internet Information Server encounters an ASP page requested by the browser, it executes the embedded program. Active Server Pages are Microsoft's alternative to CGI (Common Gateway Interface) scripts and Java Server Pages (JSPs) which allow Web Pages to interact with databases and other programs.

Microsoft IIS and PWS are free Web Server Components. Your PC can act as Web Server, if you install IIS or PWS.

5.2.1 Creating an ASP Document

Creating an ASP document is easy. To begin coding in ASP, you require a simple-text editor like Notepad or Editplus.

- ASP can be connected to databases.
- ASP dynamically formats the page.
- ASP executes statements.

You may use VBScript, JavaScript, PerlScript, and PythonScript with Active Server Pages. VBScript is always set as the default scripting language for ASP.

After the installation of IIS or PWS on your PC, you should follow these steps:

1. Open up "My Computer" or "Windows Explorer" so that you can view your system's files and directories.

2. Select and Open the C drive.
3. Double click the Inetpub folder.
4. Double click the wwwroot folder.

Note: The full path to this location is "C:\Inetpub\wwwroot".

5. Within the wwwroot directory, write ASP code and save it into directory C:\Inetpub\wwwroot\.

5.2.1.1 First Program

Firstaspprogram.asp code

1. Lets save your ASP file in a folder that exists inside C:\Inetpub\wwwroot.
2. Save file to C:\Inetpub\wwwroot\myfolder\Firstaspprogram.asp.
3. Run your ASP file by using Internet Explorer and typing "http://localhost/Firstaspprogram.asp"

Normally, ASP file contains HTML tags, just like HTML file. However, an ASP file can also contain server scripts, surrounded by the delimiters <% and %>.The response.write command is used to write output to a browser. The following example sends the text "My First asp Program" to the browser:

```
<html>
<body>
<%
    response.write("My First asp Program")
%>
</body>
</html>
```

Output:

My First asp Program

How to Code ASP:

ASP need tags to create dynamic web sites. Below is the ASP script we used in the previous example:

```
<html>
<body>
<%
response.write("My First asp Program")
%>
</body>
<html>
```

ASP tags <% and %> will encapsulate your ASP code. ASP code can occur any where, even within HTML tag like the following:

```
<a href="<% response.write("a1.asp") %>">EXIT</a>
```

5.2.2 Server Side ASP Code using Vbscript

```
<html>
<body>
<%
Dim aString
aString = Date()
response.write("The date is: " &aString)
%>
</body>
</html>
```

Output:

The date is: 01/02/2012

Note: Dim is used to give "dimension" to variables before its use, as we have done with a String in the above example.

5.2.3 JavaScript in Client Side Scripting

If we want to use ASP code along with some client side, Javascript code can be used. In the below given example, we use the JavaScript's write function and embed our ASP Code "to show the Date" within JavaScript's write function.

```
<script language=JavaScript>
document.write("The date is:<%
Dim aString
aString = Date()
response.write(aString)
%>")
</script>
```

Output:

The date is: 03/26/2011

5.2.4 Variable

Variable is a named unit of storage. Through variables, values can be stored the into the memory.

Declaration of a Variable in ASP

Always declare all the variables before using them.

- Declare a variable with the use of the Dim keyword in ASP, which is an acronym for Dimension.
- In computational literature, dimension refers to space in computer memory.

All the variables can be declared at once. Refer the example described below:

ASP Code:

```
<%
'Single Variable Declarations
Dim aVar1
Dim aVar2
'Multiple Variable Declarations
Dim aVar9, aVar10, aVar11
%>
```

5.2.4.1 Variable Naming Conventions in ASP

Default language for ASP is VBScript therefore, it uses VBScripts variable naming.

Naming conventions are given below:

1. Variable name must start with an alphabetic character (A - Z or a - z).
2. Variable name must be without a period.
3. Length of the variable name cannot be longer than 255 characters.
4. Variables must be unique in the scope, in which it is declared.

5.2.4.2 Defining ASP Variables

You can assign values to ASP variables using "=" operator. Variable definition is described in the following code:

Example:

ASP Code:

```
<%
Dim aS, aN, aG
aN = 27
aS = "Hi"
aG = 100
aG = "value of changed variable"
Response.Write("aN = " & aN & "<br />")
Response.Write("aS = " & aS & "<br />")
Response.Write("aG = " & aG & "<br />")
%>
```

Output:

aN = 27

aS = Hi

aG = value of a changed variable

Example:

ASP Code:

```
<html>
<body>
<%
dim x
x="Good"
response.write("You are: " &x)
%>
</body>
</html>
```

Output:

You are: Good

5.2.5 Array Concepts

Array is a collection of similar type of data. Memory allocation is contiguous in Array. Element of an array can be accessed using indexes. Always, an array index starts with 0 and ends with MAX-1where Max is the size of an Array.

In the example demonstrated below, we have taken an array "Aarray(6)" which can store seven elements. After storing values into array, we have used for loop to display all the elements.

ASP Code:

```
<html>
<body>
<%
Dim Aarray(6),j
Aarray(0) = "Manmohan Singh"
Aarray(1) = "Sania Mirza"
Aarray(2) = "Sachin Tendulkar"
Aarray(3) = "Vishwanathan Anand"
Aarray(4) = "Vijender Singh"
Aarray(5) = "Rahul Gandhi"
Aarray(6) = "Yashpal"
for j = 0 to 6
   response.write(Aarray(j) & "<br />")
next
%>
</body>
</html>
```

Output:

Manmohan Singh
Sania Mirza
Sachin Tendulkar
Vishwanathan Anand
Vijender Singh
Rahul Gandhi
Yashpal

5.2.5.1 Fixed Size Array

```
<%
Dim aFixedArray(3) 'Fixed size array
%>
```

Assigning Value to a Fixed Size Array

Let's fill up fixed size array with values. You must consider three things to assign a value to Fixed Size Array:

1. The name of the array that you have declared.
2. The value you want to store in to the array.
3. The position in the array where you want to store the value.

You can access group of variables inside an array by specifying the position inside the array.

In the example demonstrated below, an array named aFixedArray has five positions: 0, 1, 2, 3 and 4. It assigns some values to array.

```
<%
Dim aFixedArray(4) 'Fixed size array
aFixedArray(0) = "Mahesh Bhupati"
aFixedArray(1) = "Liander Paes"
aFixedArray(2) = "Vijay Amritraj"
aFixedArray(3) = "Anand Amritraj"
aFixedArray(4) = "Ramesh Krishnan"
%>
```

5.2.5.2 Dynamic Sized Arrays

```
<%
Dim aDynamicArray() 'Dynamic size array
%>
```

When you declare an array whose size can be changed at any time; simply do not put a number within the parenthesis; use the ReDim keyword when you know what size you want the array to be. Use ReDim as many times as you wish.

A ReDim command loses any current data in the array. This might be fine if you're just getting started, but if someone is halfway through transaction it will be bad to lose their data. In that case, you would instead use the Preserve keyword, if you want to keep your data that already exists in the array.

Below an example is demonstrated which uses ReDim Keywords:

```
<%
Dim aArray(),i 'Dynamic size array
ReDim aArray(1)
aArray(0) = "abc"
aArray(1) = "def"
ReDim Preserve aArray(3)
aArray(2) = "ghi"
aArray(3) = "jkl"

for each i in aArray
   response.write(i & "<br />")
next
%>
```

Output:

abc def ghi jkl

5.2.6 Operators

5.2.6.1 Arithmetic Operators

The following are the arithmetic operators:

Operator	Description
+	Addition
-	Substraction
*	Multiplication
/	Division
^	Exponential
Mod	Modulus
-	Negation
\	Integer Division

However, some operators are not included in ASP like ++, – –, +=.

5.2.6.2 Relational Operators

When in a given expression relational operator is used, expression always returns the Boolean value in terms of either TRUE or FALSE.

Operator	Description	Example
=	Equal To	4 = 3
<	Less Than	4 < 3
>	Greater Than	4 > 3
<=	Less Than Or Equal To	4 <= 3
>=	Greater Than Or Equal To	4 >= 3
<>	Not Equal To	4 <>3

5.2.6.3 Logical Operators

When the decision is based on true values of logical variables, a logical operator is used.

Operator	Description
and	Both Must be TRUE
or	One Must be TRUE
not	Flips True Value

Example:

Example	Result
True and False	False
True or False	True
Not True	False

5.2.6.4 String Operators

Operator	Description
&	String Concatenation

String Concatenation Operator "&" that takes two strings and concatenate them together to form a new string.

Example: If string1 = "abc" and string2 = "def".

The following code would combine these two strings into one:

string3 = string1 & string2

Example	Result
String3=”abc”&”def”	String3= “abcdef”

5.2.7 Conditional Statement

We apply conditions in our program through IF, ELSE IF, ELSE statements. If the statement is ASP’s conditional branch statement, then it can be used to route program execution through two different paths. The statement allows us to control the flow of execution based upon conditions known only during run time. The “IF” statement enables the programmer to selectively execute statements based on a given condition. The code block governed by the “IF’’ is executed, if the condition is true.

Syntax:

```
if expression
   then
            Statement 1;

   elseif expression

            Statement 2;

   else

            statement;
end if
```


There can be zero or more else, if parts and the else part is optional.

Example 1

```
<%
Dim aNum
aNum = 8
If aNum = 8 Then
Response.Write(“Variable aNum = 8”)
End If
%>
```

In the above example, if expression returns true value, statement of then part will be executed.

Output:

Variable aNum = 8

Example 2:

Below is an example in which Else portion of the If Statement is always executed as the returned value of expression if, will always be false.

```
<%
Dim aNum
aNum = 25
If aNum = 7 Then
response.write("Variable aNum = 7")
Else
response.write("Variable aNum = " & aNum)
End If
%>
```

Output:

Variable aNum = 25

Else If conditional statement is used, when multiple conditions are to be checked. Below is an example in which second if statement (elseif) is always true.

Example 3:

```
<%
Dim abc
abc = "hello"

If abc = "raj" Then
Response.write("JUMP")

ElseIf abc = "hello"
Then
Response.write("SING")

Else
Response.write("PLAY")
End If
%>
```

Output:

SING

5.2.7.1 ASP Select Case

Syntax:

```
select case variable
    case 1
            Set of statements

    case 2
             Set of statements
            .
            .
    case else
            set of statements
    end select
```

- The list of case statements is checked against the variable that appears immediately after the Select Case.
- Within the Select Case block of code, these case statements are contained.

Below is an ASP Select Case example which checks for integer values:

Example 1:

```
<%
Dim anum
aNum = 2
Select Case anum
Case 2
   Response.Write("The Number is Two")
Case 3
   Response.Write("The Number is Three")
Case 5
   Response.Write("The Number is Five")
Case Else
   Response.Write("The Number is " & aNum)
End Select
%>
```

Output:

The Number is Two

Select Case with String Variables

- You can also use a string as the variable to be used in the statement.

Below we select against a string:

Example 2:

```
<%
Dim aveg
aveg = "to"
Select Case aveg
Case "po"
   Response.Write("I  PLAY")
Case "to"
   Response.Write("I  EAT")
Case Else
   Response.Write("I  SING")
End Select
%>
```

Output:

I EAT

5.2.8 Iterative Statements

Iterative Statements Create loops. A loop repeatedly executes the same set of instructions until a termination condition is met.

5.2.8.1 Do While...Loop

Syntax:

```
Do While expression

   Set of statements
Loop
```

Example:

```
<%
x=0
Do While x<4

   response.write("Hi<HR>")
   x=x+1
Loop
%>
```

In this example the condition is "x<4" and the instructions define a response text and an increment of the variable "x". In the above example, x will be increased until it gets a value of 4, then the loop will be terminated. Several instructions can be used within the loop.

5.8.2 Do Until....Loop

Syntax:

```
Do Until Expression

   Set of statements
Loop
```

Example:

```
<%
x=0
Do Until x=4

   response.write("Hi<HR>")
   x=x+1

Loop
%>
```

In this example, the condition is "x=5", so x will increase until it is equal to 4, and then the loop will be terminated.

5.2.9 Procedures

```
<html>
<head>
<%
sub sum(num1,num2)
  response.write(num1+num2)
end sub
%>
</head>
<body>
<p>You can call a procedure like this:</p>
<p>
Result: <%call sum(8,9)%></p>
<p>or, like this:</p>
<p>Result: <%sum 8,9%></p>
</body>
</html>
```

Output:

17

Note:

- While calling a JavaScript or a VBScript procedure from an ASP file, you can use the "call" keyword followed by the procedure name.
- While using the "call" keyword, if a procedure requires parameters, the parameter list must be enclosed in parentheses.
- If you omit the "call" keyword, the parameter list must not be enclosed in parentheses. The parentheses are optional, if the procedure has no parameters.

5.2.9.1 ASP Procedure Using both VbScript and JavaScript

```
<html>
<head>
<%
sub sum1(num1,num2)
  response.write(num1+num2)
end sub
%>
<script language="JavaScript" runat="server">
function sum2(num1,num2)
{
response.write(num1+num2)
}
</script>
</head>
<body>
<p>TOTAL: <%call sum1(9,7)%></p> <p>TOTAL: <%call
                                sum2(9,7)%></p> </body> </html>
```


Output:

TOTAL:16

TOTAL: 16

5.2.10 Create an HTML Form

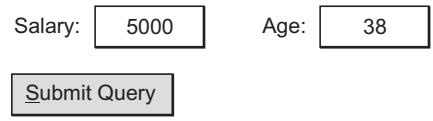
- You need to create HTML form that will send information to your ASP page, before processing the information.
- For sending data to an ASP form, there are two methods named GET and POST.

Example 1:

Below is a simple form that will send the data using the method GET. Save the following code as "code.html":

```
<form method="GET" action="submit.asp">
Name <input type="text" name="Salary"/>
Age <input type="text" name="Age"/>
<input type="submit" value="Submit Query" />
</form>
```

Output:

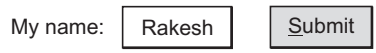


Example 2:

Form Example in ASP Using Method="GET"

```
<html>
<body>
<form action="A2.asp" method="GET">
My name: <input type="text" name="fname" size="20" />
<input type="submit" value="Submit" />
</form>
<%
dim myname
myname=Request.QueryString("myname")
If myname<>"" Then
Response.Write("Hello " & myname & "!<br />")
Response.Write("I am fine")
End If
%>
</body>
</html>
```

Output:



Results:

Hello Rakesh!

I am fine

Connectivity of ASP to the databases is discussed in the Appendix section**.**

5.3 DHTML CONCEPTS

DHTML stands for Dynamic Hyper Text Markup Language. It consist of several technologies and describes how these technologies interact. DHTML can be

described as animated HTML, because DHTML allows a Web page to change, after it is loaded into the browser and does not require any communication with the server for an update. Through DHTML, you can create active pages.

Various authors have defined DHTML as follows:

Gardner (1998) stated about the DHTML as, "DHTML is the combination of several built-in browser features in fourth-generation browsers that enable a Web page to be more dynamic."

According to Hyman, 1999, p. 8, "Dynamic HTML is a set of commands mixed with text that describe how a Web page should appear".

According to Teague, 1999, p. 3, "Dynamic HTML" is a marketing term coined by both Netscape and Microsoft to describe a series of technologies introduced in the 4.0 versions of their browsers, to enhance the "dynamic" capabilities of those browsers."

Finally, DHTML can be defined as:

Dynamic HTML = CSS + DOM + Scripting

Figure 5.7

Where CSS stands for **Cascading Style Sheets and DOM stands for Document Object Model.**

5.3.1 Advantages of DHTML

- DHTML makes documents active.
- DHTML allows the designer to control how the HTML displays Web pages content.

  DHTML react and change with the actions of the visitor.

  DHTML can exactly position any element in the window, and change that position after the document has been loaded.
- According to Teague, 1999, p. 3, DHTML can hide and show content as and when needed.
- According to Schurman and Pardi, 1999, p. 5, DHTML allows any HTML element (any object on the screen that can be controlled independently using JavaScript) in Internet Explorer to be manipulated at any time, turning plain HTML into dynamic HTML. With DHTML, the changes occur entirely on the client-side (on the user's browser). (Weiss, 1998).

- According to Richmond, 1969, Using DHTML gives the author more control over how the page is formatted and how the content is positioned on the page.

5.3.2 Components of DHTML

Dynamic HTML includes the following components:

- **HTML**
- **Scripts** – JavaScript as Small programs designed to manipulate the pages.
- **Document Object Model (DOM)** – DOM describes a path that leads from the HTML document down to the various elements on the page. Teague, 1998, p. 138, gave the example as DOM for an image called button1 would be: document.images.button1. The objective of creation of Object Document model is given as follows:
  1. All the components of a Web document will be represented in a hierarchical manner.
  2. You can modify the Document Object model by adding and removing the contents.
  3. You can monitor and manipulate the characteristics of content on the page in different ways using a Document Object Model.
  4. According to Schurman, E. and Pardi, W., 1999, p. 74, DOM provides information about how the items on a page interrelate with the user and to each other.

The web page author can use scripts to control everything on the page and to modify anything at any time. The following are the examples of effects the designer can create,

5.3.2.1 Using the DOM with JavaScript

1. The color of hyperlink changes when the user puts the mouse over it.
2. The user can move text or images around on a page.
3. Some of the items on a page move around without the user interaction and end up positioned exactly where the web page author wants them.
4. When the user positions the mouse over an image, the text appears.
5. The Web page author can change the formatting of text, graphics, and tables on the fly.
6. Parts of the page are dynamically created or destroyed as they appear on the screen.

The above examples are given by Schurman, E. & Pardi, W., 1999, pp. 75.

As per Teague, 1998, p. 139, The DOM works by tracing a path from the script down to the individual elements to be acted upon with the HTML document. The steps in this hierarchy are:

window
(parent)
(frame)
document
ID
image
link
anchor
form

5.3.2.2 Cascading Style Sheets (CSS)

– CSS allows you to define how HTML tags should display their content.

According to Hyman, 1999, p.35, **Cascading Style Sheets** is a specification that gives the Web page author powerful visual control over HTML elements such as location, size, color, background, images, etc.

As per Heinle, CSS is a separate language, one of the several you need to know to master DHTML.

As per Teague, 1998, p.15, Cascading Style Sheets is supported in HTML 4.0, released in December 1997, by the World Wide Web Consortium (http://www.w3.org/Style/CSS).

Teague, 1998, p.16 gave CSS rules which define what the HTML should look like and how it should behave in the browser window.

Rules consist of three parts:

1. Selectors – Alpha-numeric characteristics that identify the rule
2. Properties – What is being defined?
3. Values – Values assigned to a property by defining its nature. A value can be a keyword, such as "yes" or "no," a number, or a percentage.

5.3.2.2.1 Cascading Style Sheets-Positioning (CSS-P)

You can position HTML elements anywhere in the window, as well as control the visibility of those elements.

SUMMARY

In the previous chapter we have focused on HTML for creating static web pages. In this chapter, the main focus is on the JavaScript. JavaScript is used to create Dynamic web pages. Constructs of JavaScript are well described in this chapter.

We have explained the basics of JavaScript, ASP with simple examples. Brief explanation about the DHTML has also been given.

PROBLEMS

1. Write a program to print gcd of two given numbers.
2. Write a program to find the sum of the following series:
   1–2+3–4+5
3. Write a program to generate prime number less than 100.
4. Write a program to remove duplicate from an ordered list.
5. Write a program to print the table of 2.
6. Write a program to find the largest number among three given numbers.
7. Write a program that converts a Fahrenheit temperature into Celsius temperature.
8. Write a program to interchange the values of two integer numbers x and y without using a third variable.
9. Write a program to find out if a year is a leap year or not.
10. Write a program to produce the following output:

1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

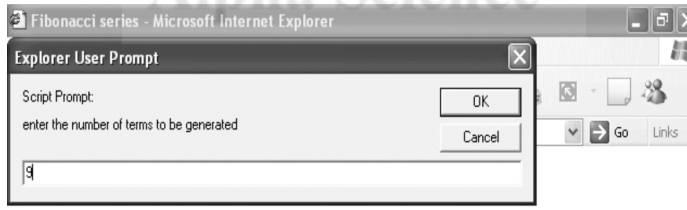
11. Write a program to convert binary numbers into decimal numbers.
12. Write a program to print all prime numbers between 25 and 50.
13. Write a program to find the value of one number raised to the power of another.
14. Write a program to check out whether a given number is an Armstrong number or not.
15. Write a program to find out whether a given number is a palindrome or not.
16. What do you mean by nested if-else? Explain it with suitable program?
17. Explain all three types of loops with suitable program.
18. Write a program to compute the sum of exponential series.
19. Write a program to compute the sum of cos series.
20. Explain the concept of break statement with example.
21. Explain the concept of continue statement with example.
22. Explain the concept of return statement with example.
23. Write a program to generate Fibonacci series,

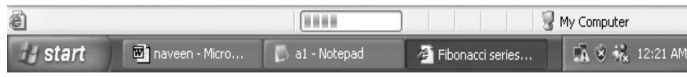
    Hint:

```
<html>
<title>Fibonacci  series</title>
<head>
<script>
```

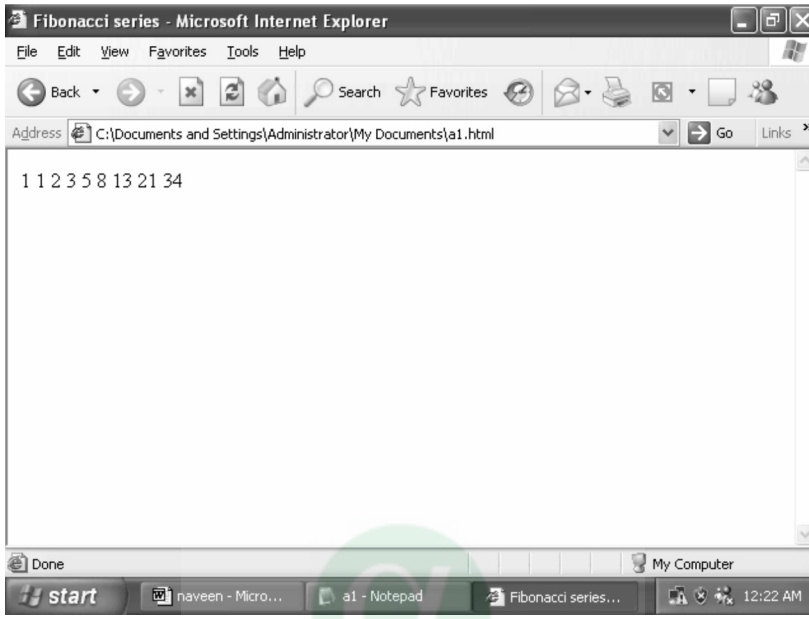
```
a=window.prompt ('enter the number of terms to be generated');
b=parseInt(a);
for (i=1; i<=b; i++)
{
      document.write (fib (i) );
      document.write ('\n');
}

function fib (n)
{
if (n==0||n==1)
return n;

else
      return (fib(n-1) + fib(n-2));
}
</script>
</head>
<body>
</body>
</html>
```

After executing the above code, a dialog box will appear as given below. We have entered the term value 9. Press OK button.





The output is as follows:



24. Write a program to generate Fibonacci series in reverse order:

Hint:

```
<html>
<head>
<script>
function f ()
{
x=window.prompt ('enter the number of terms to be generated',0);
document.write ('the Fibonacci series is :');
document.write ( '<br>');
for(i=x;i>=1;i—)
{
document.write (fib (i) +' ');
}
function fib (n)
{
if (n==1)
return 0;
if (n==2)
return 1;
else return fib(n-1) + fib(n-2);
}
```

```
</script>
</head>
<body onLoad='f( )'>
</body>
</html>
```

Output:

The Fibonacci series is:
21 13 8 5 3 2 1 1 0

25. Write a program to implement Linear Search.

Hint:

```
/*LINEAR SEARCH*/
<HTML>
<HEAD>
<SCRIPT>

s=window.prompt ("enter the size of array: ");
s1=parseInt (s);
var a=new Array (s1);
for (i=0; i<s1;i++)

va=window.prompt ("enter the "+(i+1) +" 'th value : ");
b=parseInt (va);
a[i]=b;
}
x=parseInt (window.prompt ("Enter the number to be searched",
0) );

for (i=0;i<s1;i++)
{
if (a[i] = = x)
{
flag=1;

break;
else
{
flag =0;
}
}
if (flag = = 1)
{
document.write ("The number is found at " + (i+1)
}
```

```
else
{
document.write ("Unsuccessful search");
}
</SCRIPT>
</HEAD>
<BODY></BODY>
</HTML>
```

26. Write a program to implement binary search.

Hint:

```
<!—binary search>
<html>
<head>
<script>
//binary search

var pos=-1;

s=window.prompt ("enter the size of array: ");
s1=parseInt (s);

var a=new Array (s1);

for (i=0;i<s1;i++)
{
va= window.prompt ("enter the "+(i+1) +" ith value of a
sorted array: ");
b=parseInt (va);
a[i]=b;
}
document.write ("The Array is: ");
for (i=0;i<s1;i++)
document.write (a[i]+"<BR>);
mid=0;

x1=window.prompt ("enter the number to be searched: ");
x=parseInt (x1);
first=0
last=(a.length)-1;
while ( (pos= =-1) &&(first<=last))
{
middle=first+last)/2;
mid=Math.round(middle);
```

```
if (a[mid]= =x)
{ pos=mid;
break;
}
else
if (a[mid]<x)
first=mid+1;
else
if (a[mid]>x)
last=mid-1;

}

if (pos>0)
document.write("element found at location "+ ++pos);
else
document.write ("unsuccessful search");

</script>
</head>
</html>
```

27. Write a program to compute factorial of a number.

Hint:

```
<HTML>
<head>
<script>
function f ( )
{
x= window.prompt ("enter the number : ","0");
y=fact (x);
document.write ("the factorial of "+ x +" : "+y);
}
function fact (n)
{
if (n= =0)
return 1;
if (n= =1)
return 1;
else return n*fact (n-1);
}
</script>
</head>
```

```
<body onload="f ( )">
</body>
</html>
```

Output:

the factorial of 5 : 120

28. Write a program to implement matrix operations.

Hint:

```
<html>
<head>
<script>
var r1, c1, r2, c2;

q=window.prompt ("enter the no of rows in first matrix: ");
r1=parseInt (q);
w=window.prompt ("enter the no of cols in first matrix: ");
c1=parseInt (w);

q1= window.prompt ("enter the no of rows in second matrix:
");
r2=parseInt (q1);
w1= window.prompt ("enter the no of cols in second matrix:
");
c2=parseInt (w1);

var a=new Array (r1);
for (i=0; i<r1;i++)
a[i]=new Array (c1);

var b=new Array (r2);
for (i=0; i<r2;i++)
b[i]=new Array (c2);

//INPUT THE FIRST MATRIX
window.alert ("enter the first matrix: ");
for (i=0; i<r1;i++)
for (j=0; j<c1; j++)

{
x=window.prompt ("enter the ["+i+"] ["+j+"]th value");
x1=parseInt (x);
a[i] [j]=x1;
}
```

```
//PRINTS THE ORIGINAL FIRST MATRIX
document.write (The original first array is:");
document.write (<br>);
for (k=0<r1;k++)
{
for (l=0; l<c1; l++)
{
document.write (a[k] [l]+" ");
}

document.write ("<br>");
}
//INPUT THE SECOND MATRIX
window.alert ("Enter the second matrix :");
for (i=0;i<r2;i++)
for (j=0; j<c2; j++)
{
xj=window.prompt ("enter the ["+i+"] ["+j+"]th value")
x2=parseInt (xj);
b[i] [j] = x2;
}

//PRINTS THE ORIGINAL SECOND MATRIX
document.write ("The original second matrix is:");
document.write ("<br>");
for (k=0; k<r2;k++)
{
for (l=0; l<c2; l++)
{
document.write (b[k] [l]+" ");
}
document.write ("<br>");
}

document.write ("***MENU:***<br>");
document.write ("0:QUIT<br>");
document.write ("1:add<br>");
document.write ("2:multiply<br>");
var ch;
c=window.prompt ("enter your choice(-1 to quit)", "0");
ch=parseInt( c );

while (ch!=0)
```

```
{
switch (ch)
{
case 1:add();
break;
case 2:multiply ( );
break;
default: document.write ("br>wrong choice!<br>");
}
c=window.prompt ("enter your text choice (0 to quit)", "0");


ch=parseInt (c );
}
function add ( )
{
//ADD
if ( (r1=r2) & (c1= = c2) )
{
document.write("<br>addition possible!");

//declare third matrix that stores the sum
var c=new Array(r1);
for (i=0;i<r1;i++)
c[i]=new Array(c1);

//Add the matrices
for (i=0;i<r1;i++)
for (j=0;j<c1;j++)
{
c[i] [j]=a [i] [j] +b[i] [j];
//print the matrix after addition.
document.write("<br> the resultant matrix is:<br>");
for (k=0; k<r1;k++)
{
for (l=0; l<c1;l++)
{
document.write(c[k][l]+:"  ");
}
document.write ("<br>");
}
}
else
```

```
document.write ("addition not possible");
}
function multiply ( )
{
//MULTIPLY
if (c1= = r2)
{
var c=new Array (r1);
for (i=0; i<c2; i++)
c[i]=new Array (c1);
document.write("<br>Multiplication possible! <br>");
for (i=0; i<r1;i++)
{

for (j=0;j<c2;j++)
{
c[i] [j]=0;
for (s=0;s<c1;s++)
{
c[i] [j]+=a[i] [s]*b [s] [j];
}
}
}
document.write ("<br>the array after multiplication:<br>");
for (k=0; k<r1; k++)
{
for (l=0; l<c1; l++)
{
document.write (c[k] [l]+" ");
}
document.write ("<br>);
}
}
else
document.write ("<br>matrices can't be multiplied<br>");
}
</script>
</head>
</html>
```

Output:

The original first array is:

5 6

4 9

The original second array is:
4 9
2 3
MENU:
0:QUIT
1:add
2:multiply

wrong choice!

Multiplication possible!

The array after multiplication:
32 63
34 63

addition possible!
The resultant matrix is:
9 15
6 12

29. Write a program to find the smallest and largest element of an array.
Hint:

```
<html>
<head>
<script>
p=window.prompt ("enter the array size");
q=parseInt (p);
var a=new Array (q);
for (i=0;i<q;i++)
{
v= window.prompt ("Enter the number");
k=parseInt (v);
a[i]=k;

}
document.write ("The input array is:");
document.write ("<br>");
for (i=0;i<q; i++)

document.write(a[i]);
```

```
document.write(" ");
}
document.write ("<br>");
bubsort (a);
document.write ("The smallest element is:");
document.write("<br>");
document.write (a[0]);
document.write ("<br>");
document.write ("The largest element is:");
document.write ("<br>");
document.write (a[q-1]);
function bubsort (b)
{
for (i=0;i<b.length; i++)
{
for (j=i+1;j<b.length; J++)
{
if (b[i]>b[j])
{
var t=b [i];
b[i]=b[j];
b[j]=t;

}
}
}
}

</script>
</head>
<body>
</body>
</html>
```


Output:

The input array is:
6 5 2 3 4
The smallest element is:
2
The largest element is:
6

30. Write a program to implement Bubble sort.

Hint:

```
<html>
<script>
<head>

// BUBBLE SORT

q=window.prompt ("enter the size of array:");
s=parseInt (q);

var a=new Array (s);

for (i=0; i<s; i++)
{
x=window.prompt ("enter the "+i+" ,the value");
x1=parseInt(x);
a[i]=x1;
}
document.write ("<br>");
document.write ("The original array is: <br>");
for (i in a)
{
document.write (a[i]);
document.write ("<br>");
}
for (i=0; i<s; i++)
for (j=i+1; j<s; j++)
{
if (a[i}>a[j])
{
temp=a[i];
a[i]=a[j];
a[j]=temp;
}
}
document.write("The sorted array is :<br>");
for (i in a )
{
document.write (a[i});
document.write ("<br>");
</script>
</head>
</html>
```

Output:

The original array is:
4
8
2
1
3
4
The sorted array is:
1
2
3
4
4
8

31. Write a program to make a calculator.

Hint:

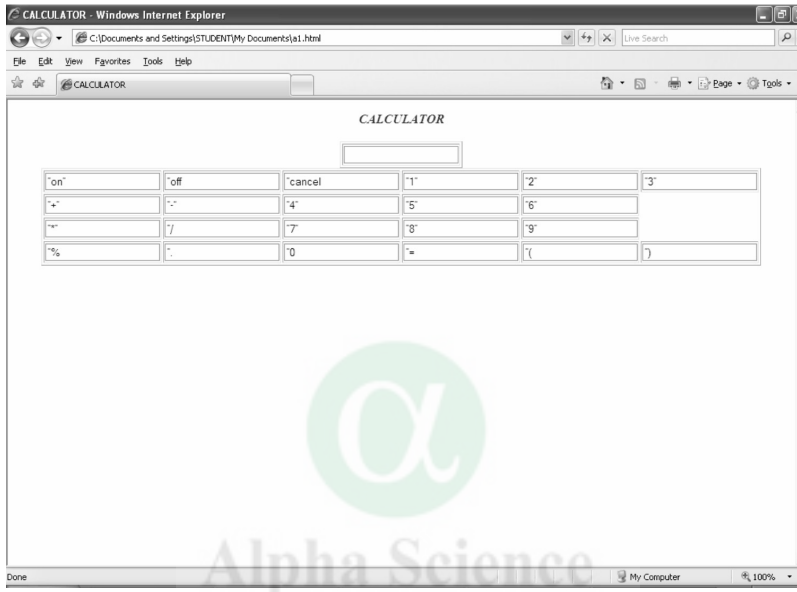
```
<html>
<title>CALCULATOR</title>
<head>
<script language="javascript">
var inp=" ";
function display(str)
{
inp+=str;
calculator.inpVal.value=inp;
}
function calon( )
{
Window.alert("calculator is on");
calculator.inpVal.value="0";
}

Function caloff()
{
 Window.alert("calculator is off")
 Calculator.inp.val=" ";
}
function evaluate (inp)
{
calculator.inpval.value=eval(inp);
```

```
}
</script>
</head>
<body>
<center>
<font color="brown">
<b><i>CALCULATOR<i><b>
</font>
<form name="calculator">
<table border="1">
<tr>
<td><input name="inpval" type="text" size="42"></td>
</tr>
</table>

<table border="1">
<tr>
<td><input name="on" type="button" value="on"
onClick="calon ( )" ></td>
<td><input name="off" type="button" value="off "
onClick="caloff ( )" ></td>

<td><input name="refresh" type="button" value="cancel "
onClick="inpval.value=' '; inp=' '" ></td>

<td><input name="0" type="button" value="1"
onClick="display ('1')" ></td>


<td><input name="+" type="button" value="2"
onClick="display (2)" ></td>


<td><input name="=" type="button" value="3"
onClick="display (3)" ></td>
</tr>
<tr>
<td><input name="1" type="button" value="+"
onClick="display ('+')" ></td>

<td><input name="2" type="button" value="-"
onClick="display ('-')" ></td>
```

```
<td><input name="3" type="button" value="4"
onClick="display ('4')" ></td>

<td><input name="-" type="button" value="5"
onClick="display ('5')" ></td>

<td><input name="%" type="button" value="6"
onClick="display ('6')" ></td>

</tr>
<tr>
<td><input name="4" type="button" value="*"
onClick="display ('*')" ></td>

<td><input name="5 " type="button" value="/ "
onClick="display ('/')" ></td>

<td><input name="6 " type="button" value="7"
onClick="display ('7')" ></td>

<td><input name="*" type="button" value="8"
onClick="display ('8')" ></td>

<td><input name=" ." type="button" value="9"
onClick="display ('9')" ></td>

</tr>
<tr>
<td><input name="7" type="button" value="% "
onClick="display ('%')" ></td>

<td><input name=" 8" type="button" value=". "
onClick="display ('.' )" ></td>
<td><input name=" 9" type="button" value="0 "
onClick="display ('0' )" ></td>
<td><input name="/" type="button" value="= "
onClick="evaluate (inp )" ></td>

<td><input name=" (" type="button" value="( "
onClick="display ('(' )" ></td>
<td><input name=")" type="button" value=") "
onClick="display (')' )" ></td>
</tr>
```

```
<center>
</table>
</form>
</body>
</html>
```

Output:



31. State whether the following variable names are valid or not:

(1) 86abc
(2) tfz$
(3) for
(4) t_fz
(5) 76mul

32. What do you mean by a keyword?

33. Evaluate the following expressions:

(1) a+b*c**d-e/f
(2) t=d/e*f-g/x+4

if the value of a=4,b=5,c=6,d=4,e=9,f=8,g=8 and x=7

34. Can you embed a comment within another comment in a Javascript?

35. If i and j are integer type variables, what will be the result of the expression:

i%j :

Where, i=–16 and j=–3?

(1) –1
(2) 1
(3) 5
(4) None of the above

36. Determine the hierarchy of operators and evaluate the following expression:
(1) t=3/2*4+3/8+3
(2) i=2*3/4+4/4+8-2+5/8

37. What is the difference between = and == operator?

38. Define comments. Why are comments needed?

39. Write down the Boolean value of each of the following expression if x=10,b=12,c=7:
(*i*) x==9 &&y!=11
(*ii*) x=6&&y=12
(*iii*) x==10||y=12
(*iv*) a==10&&b==12&&c=7
(*v*) a=10&&b==12||c<=7
(*vi*) b=10||c<=7

40. Can we also apply Bitwise operators with floating point values?

41. If you want to multiply a number by 2, which bitwise operator will you use?

42. If you want to divide a number by 2, which bitwise operator will you use?

43. What is the associativity of an assignment operator?

44. What is the naming convention with variables in JavaScript and VbScript programming language?

45. Data typing is dynamic in Javascript and VbScript. Explain it with example.

REFERENCE AND RELATED LINKS

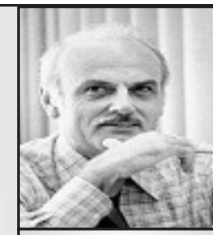
[1] Joel Sklar "Principles of web design, fourth edition, cengage learning, 2009.
[2] James Jaworski(1999)"Mastering Java script and Jscript". BPB Publication.
[3] www.edb.utexas.edu
[4] Tiley, E. ClubIE and clubwin, guide to dhtml. [Online]. Available:
[5] Weiss. A. (1998), Introduction to dynamic html. [Online]. Available:

Database Concepts

6

CHAPTER

Born	: 23, August 1923 Isle of Portland, UK
Died	: 18, April 2003 Williams Island, Florida
Known For	: Relational Model, OLAP Edgar Frank "Ted" Codd
Notable Awards	: Turing Award



Edgar Frank "Ted" Codd

Introduction

Database and Database technology has grown rapidly during the last three decades. Now, it is an integral part of every organization. To understand any subject completely, the knowledge of the past is a must and database is no exception. Therefore, to understand the concept of databases in a better way, you must know the history of database evolution. Now, we are describing the evolution of database from historical perspective.

In 1960's (Era of Database managing through File System) database were managed through a file system. Before the evolution of DBMS (Database Management System), most of the organizations used to store information in the file systems (see Figure 6.2) which were supported by a conventional operating system. The system stores permanent records in various files. The application programs were needed to extract, add and delete records. Different programmers were working on a single project, therefore various files were created by different programmers at some interval of time. These files were created in different formats and different programs were written using different programming languages. If we manage database through a file system; a lot of difficulty occurs for managing the data. Consequently, a lot of redundancies (Data will be redundant, if it is stored at more than one place in a file) will be in the database. If the same data is present at more than one place in the database,

then the main disadvantage is the wastage of storage space and access cost. If duplicity in the database persists, the database could be inconsistent (which means that if we change some record at one place, the change will not be reflected at all the places) and we would receive wrong results while manipulating the data. File processing system do not allow data to be accessed in a convenient manner. As the data is stored in various files and these files may be stored in different formats, writing an application program to retrieve the data is difficult. **This is known as data isolation**. It was very difficult to apply the Integrity Constraints (detailed about integrity constraint is discussed in the subsequent section) on the database through a file system. Multiple users can not update the data using the file system simultaneously. It can also leave the data in an inconsistent state because through file system, concurrent access was not possible. Data was not secured in the file system.

Note: Data is used for both singular and plural. For singular data we use term datum.

A Database System is usually organized according to a data model. Earlier two main data models were developed: Network (CODASYL) and Hierarchical (IMS) model. Access to database was through low-level pointer operations linking to records. Storage details were dependent on the type of data to be stored. Thus, adding an extra field to your database requires rewriting the underlying access/modification scheme. Emphasis was on records to be processed, not on overall structure of the system. A user would need to know the physical structure of the database, in order to satisfy the query for information. One major commercial success was SABRE system from IBM and American Airlines.

Between **1970-72 (Era of Centralized Database),** the concept of Centralized database came into existence (see figure 6.3); the following are the definitions of database:

- "Database is the collection of interrelated data".
- "Database is a collection of data with harmful redundancies wholly or partially reduced and shared among the multiple users".
- "A collection of actual data which constitutes the information regarding an organization stored into a database". For example, there are 2000 employees in an organization and we have to store their personal details like employee's name, hire date etc. These details will be stored in a database.
- Database is a collection of records, a record is a collection of various fields and a field is a collection of group of bytes and a byte consists of eight bits. For student's database, complete information about a student is a record which consist of student's name, roll. no., address, course, etc. For example: (name, roll.no…. are field name)

Most importantly, **your database should be non redundant, consistent and shared among the multiple users**. Data is the raw form of data stored in the database and information is the processed data. **E. F. Codd** proposed a Relational Model for databases in the research paper on how to think about the databases. He disconnects the schema (logical organization) of a database from the physical storage methods by giving the concepts of Data Independence (The concept of data independence will be discussed in the subsequent section). This system has been standardized ever since. **Between 1974 and 77 (Era of Relational Database Management System)** two main prototypes for relational systems were developed named as Ingres and System R. Ingres was developed at UCB. This ultimately led to Ingres Corp., Sybase, MS SQL Server, Britton-Lee, and Wang's PACE. This system used QUEL as query language. System R was Developed at IBM San Jose and led to IBM's SQL/DS & DB2, Oracle, HP's Allbase, Tandem's Non-Stop SQL. This system used SEQUEL as query language. The term Relational Database Management System (RDBMS) came into existence during this period. In 1976, **P. Chen** proposed the **Entity-Relationship (E-R**) model for database design giving yet another important insight into **Conceptual Data Models**. Such higher level modeling allows the designer to concentrate on the use of data instead of Logical Table structure. During **Mid-1980's SQL** (Structured Query Language) became standardized. DB2 became IBM's popular product. Network and Hierarchical models went into the background, though there is no development of these systems today but legacy systems are still in use. According to the estimates, more than 70% of the data of large corporates are still on mainframe computers. Development of the IBM PC gives rise to many DB companies and products such as RIM, RBASE 5000, PARADOX, OS/2 Database Manager, Dbase III, IV (later FoxBASE, even later Visual FoxPro), and Watcom SQL. In **early 1990's**, the development centers on the client tools for application development such as PowerBuilder (Sybase), Oracle Developer, VB (Microsoft), etc. Client-Server model for computing becomes the norm for future business decisions. Development of personal productivity tools such as Excel/Access (MS) and ODBC, led to the beginning of Object Database Management Systems (ODBMS) prototypes. A brief summary of the evolution of database has been described in the Figure 6.1. [17]

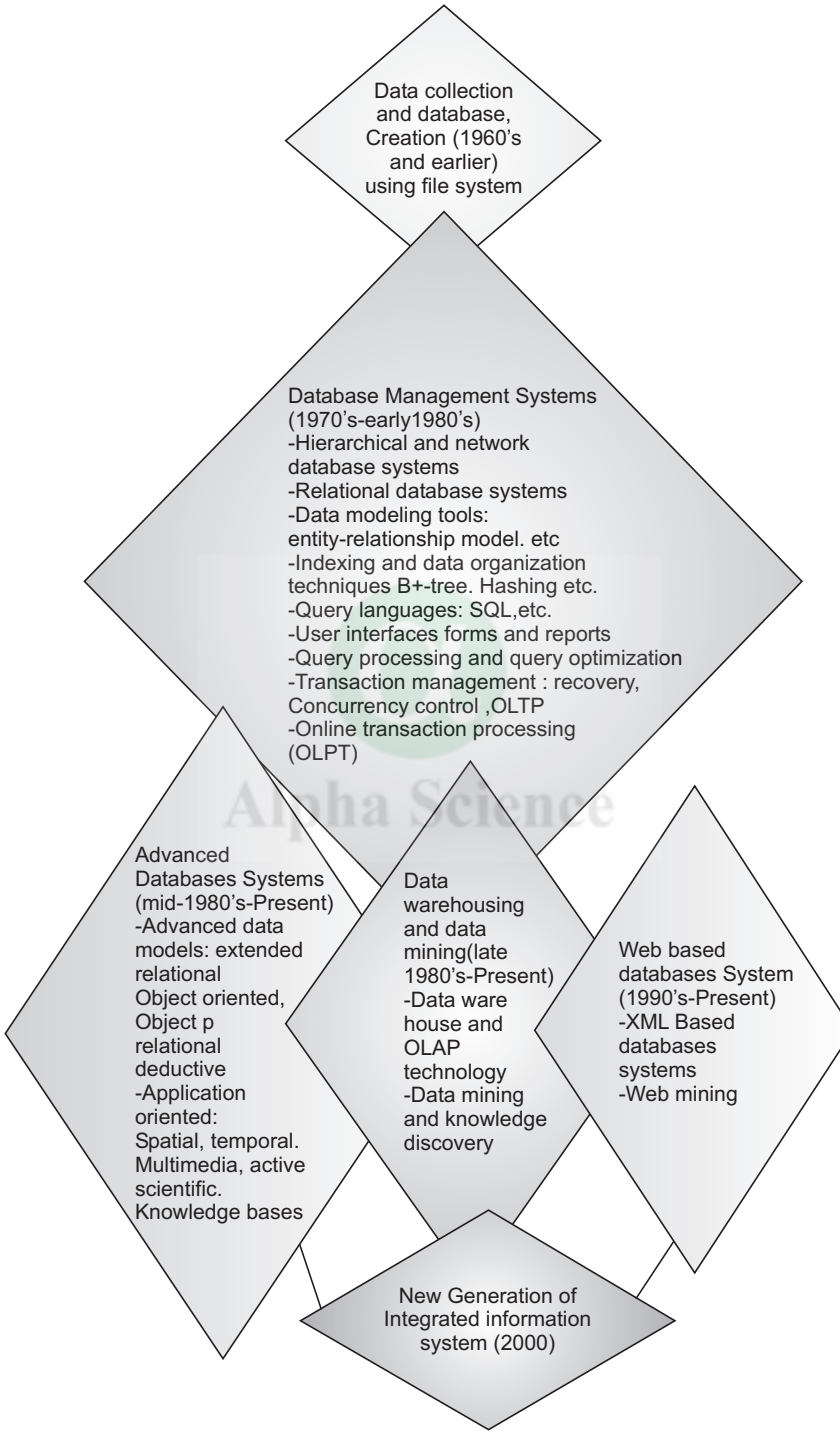

Data collection and database, Creation (1960's and earlier) using file system
Database Management Systems (1970's-early1980's)
-Hierarchical and network database systems
-Relational database systems
-Data modeling tools: entity-relationship model. etc
-Indexing and data organization techniques B+-tree. Hashing etc.
-Query languages: SQL,etc.
-User interfaces forms and reports
-Query processing and query optimization
-Transaction management : recovery, Concurrency control ,OLTP
-Online transaction processing (OLPT)
Advanced Databases Systems (mid-1980's-Present)
-Advanced data models: extended relational
Object oriented,
Object p relational
deductive
-Application oriented:
Spatial, temporal.
Multimedia, active
scientific.
Knowledge bases
Data warehousing and data mining(late 1980's-Present)
-Data ware house and OLAP technology
-Data mining and knowledge discovery
Web based databases System (1990's-Present)
-XML Based databases systems
-Web mining
New Generation of Integrated information system (2000)

Figure 6.1

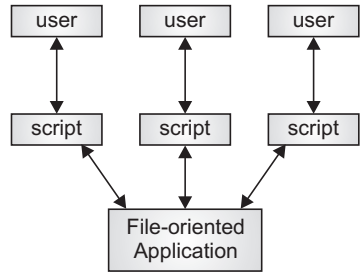


Figure 6.2: File-oriented system

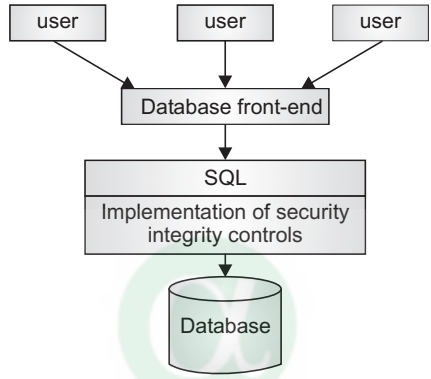


Figure 6.3: DBMS-oriented system

Please note that Relational, Hierarchical and Network data models were record based but the difference was in organizing the records. All three data models were designed to process large amount of data in simple and fixed format. All the DBMS were based on Relational, Hierarchical and Network models. Today's applications have moved from centralized mainframe computer to distributed environment, for example, Computer Aided Design (CAD), multimedia system, design of complex projects using software engineering and knowledge database. These applications require complex operations and data structure representation. Now, we are on the verge of another generation of database system which is called object oriented DBMS based on object oriented programming paradigm. OODBMS manages more complex kind of data, for example, multimedia objects. The other kind of next generation DBMS is knowledge based management system (KDBMS). The main use of KDBMS is to support the management of the shared knowledge. KDBMS supports a large number of complex rules for automatic data **inferencing** (retrieval) and maintenance of data integrity. RDBMS were never designed to allow the nested structure. These types of applications are extensively found in CAD/CAE (Computer Aided Design/Computer Aided Manufacturing), Aerospace, etc. OODBMS can easily support these applications. It is much easier and natural to navigate through these complex structures in the form of objects that model

the real world in OODBMS rather than table, tuples and records in RDBMS. In OODBMS, the data and procedures are together and the object responds to messages.

Knowledge Base Management System (KBMS) manages the knowledge in a given domain of interest and exhibits reasoning power to the level of a human expert in this domain. Operations in knowledge base are more complex than those in traditional databases. When a rule is added to the system, system must check for contradiction and redundancy. Knowledge can be defined as a body of facts and principles accumulated by human kind or the act, fact or state of knowing. KBMS is used to manage and manipulate shared knowledge. A Knowledge Base Management System is a computer system that manages the knowledge in a given domain or field of interest and exhibits reasoning power to the level of human expert in this field. In KBMS, facts are stored with the rules (inference rules). Whenever we query the data from the KBMS, always the rules will be checked as per the query and the rules and facts will be extracted.

During mid-**1990's (Era of Internet),** WWW was introduced. In **Late-1990's,** Active Server Pages, Front Page, Java Servlets, JDBC, Enterprise Java Beans, ColdFusion, Dream Weaver, Oracle Developer 2000 were developed. Open source solution came online with widespread use of gcc, cgi, Apache, MySQL, etc. Open source solutions are freely available on the Internet. Source code is available to the users. Users can download the open source software free of cost. Online Transaction processing (OLTP) and online analytical processing (OLAP) came of age with many merchants using point-of-sale (POS) technology on a daily basis. In the **early 21st century (Era of Terabyte Systems),** huge systems known as terabyte are appearing; and will require novel means of handling and analyzing data. Data Mining, Data Warehousing, Data Marts are the used technologies today.

"Data Warehousing is the process of accumulating operational and legacy data from heterogeneous data sources (It is an integrated, time-variant and non-volatile collection of data), transforming, cleaning, normalizing and filtering it for creation and incremental uploads into the data warehouse".[14]

"A datamart is a subset of a Data Warehouse designed to meet specific requirements of user groups". [14]

"Data Mining is the process of extracting unknown and potentially useful patterns, trends, anomalies and rules from stored historical data for business promotions, decision making or classification" [14].

Successors to SQL (and perhaps RDBMS) will emerge in the future. Yet, attempts to standardize SQL successors have not been successful. Most probably this will be overtaken by XML (Extensible Mark up language) and other emerging technologies. Through XML, you can create user defined tags. You can create databases on the server side as well as client side. Mobile database use is a product which is entering the market in different ways. Distributed

transaction processing is becoming the standard for business planning in many arenas. Linux with Apache supporting mySQL and Oracle on relatively cheaper hardware is a major threat to high cost legacy systems of Oracle and DB2.

SOME IMPORTANT TERMS

Database

A collection of interrelated data with harmful redundancies wholly or partially reduced which is shared among the multiple users and organized in the form of tables.

Tuple: A row of a table.

Table: Data arranged in rows and columns with certain properties.

Record: A complete unit of information; named collection of data items.

Field: Information about data item.

File: Collection of records.

View: Logical container of data derived from one or more underlying tables.

Primary key: Unique identification property of a row.

File organization*:* Physical ordering of records in the file.

Domain: A set of values in a given column.

Degree: Number of attributes in a table.

Cardinality: Number of records in a table.

Alternate key: A candidate key which is not a primary key.

Candidate keys: Field combinations in a table that can serve as a primary key.

Integrity: Is required to protect the data from invalid users. It ensures that data is of high quality, correct and consistent.

Data Independence: Immunity of application to change the storage structure and its access strategies.

Logical Data Independence: If we commit any changes to conceptual level, there should not be any effect on the external level.

Physical Data Independence: If we commit any changes to Physical level, there should not be any effect on the conceptual level.

6.1 COMPARISION BETWEEN FILE SYSTEM APPROACH AND CENTRALIZED APPROACH

- If we manage database through a file system, a lot of difficulty occurs in managing the data. Consequently, lot of redundancies will be occur in the database.

- If the same data is present at more than one place in the database, then the main disadvantage is the wastage of storage space and access cost.
- If duplicity in the database persists, the database could be inconsistent (which means that if we change some record at one place, the change will not be reflected in all the places) and we would receive wrong results while manipulating the data.
- File processing system does not allow data to be accessed in a convenient manner. As the data is stored in various files and these files may be stored in different formats, writing an application program to retrieve the data is difficult. **This is known as data isolation**.
- It was very difficult to apply the Integrity constraints on the database through a file system.
- Multiple users could not update the data simultaneously using the file system. It can also lead the data to an inconsistent state since through a file system, concurrent access was not possible.
- Data was not secure in the file system.

Centralized Database

- Data management is easier.
- Data redundancies can be reduced to some extent.
- Reduced access cost and saving of storage space.
- No inconsistency in the database.
- Easier data access from the centralized database.
- Easy to apply Integrity constraints on the database.
- Data is secure.
- Concurrency is possible.
- Automatic backup and recovery.

Most importantly, **your database should be non redundant, consistent and shared among the multiple users**.

6.2 DATABASE MANAGEMENT SYSTEM (DBMS)

- "DBMS is an interface between the user's application program and the data physically stored into the database". DBMS software is used to retrieve the data from the database, to manipulate the data, store them into the database and to design the structure of the database.
- "A collection of programs that enables to store, modify and extract information from a database is known as DBMS".

The primary goal of a DBMS is to provide a way to store & retrieve database information in a convenient and efficient manner.

DBMS is designed to manage a large collection of information. Management of data involves both defining structures for storage of information and providing way for manipulation of data. Note that the DBMS must ensure the safety of data. There are many different types of DBMS ranging from small systems that run on personal computers to huge systems that run on mainframes. Good data management is an essential prerequisite to corporate success.

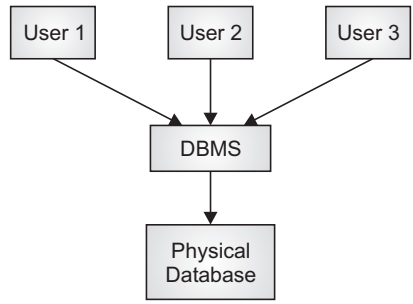


Figure 6.4

6.3 APPLICATIONS OF DBMS

Databases are applied in a very wide range of applications. The following are some examples of database applications:

- In banks, the database is used for customer information, accounts, loans & other banking transactions.
- In airlines, the database is used for reservation & schedule information.
- In universities and colleges, the database is used for student information, course registration, grades etc.
- In Credit card transaction, the database is used for purchase of credit cards & generation of monthly statements.
- In telecommunication, the database is used for keeping the records of calls made, generating monthly bill, etc.
- In finance, the database is used for information about holdings, sales & purchase of financial statements.
- There are numerous other applications like Sales (customer, product & purchase information), Manufacturing (management of supply chain) and human resource (storing information about employees, salaries, tax, benefits) (Genome sequence database (Biological database)), etc, in which the database is used.

To summarize, whenever there is a need to have automation in the organization, we need to have a Database System.

6.4 DATA ABSTRACTION

If we maintain our database using File System and commit any changes in the file, then we will have to change the whole application program. This shows that user application program is not independent of the database. This thing can be rectified using the centralized database. If we maintain the centralized database, then there is no need to change the application program with every change in the database structure. This property is called Data Independence. Data Independence is the immunity of the application software to change the structure of the database and its access strategies. A centralized database contains a number of files and certain programs to access and modify these files. But the actual data is not shown to the user; the system hides actual details of how the data is stored and maintained. This is known as data abstraction property. As all the details are not used by the user, so we hide the actual (complex) details from users through Centralized Database.

Various levels of abstraction to data is provided which are given below,

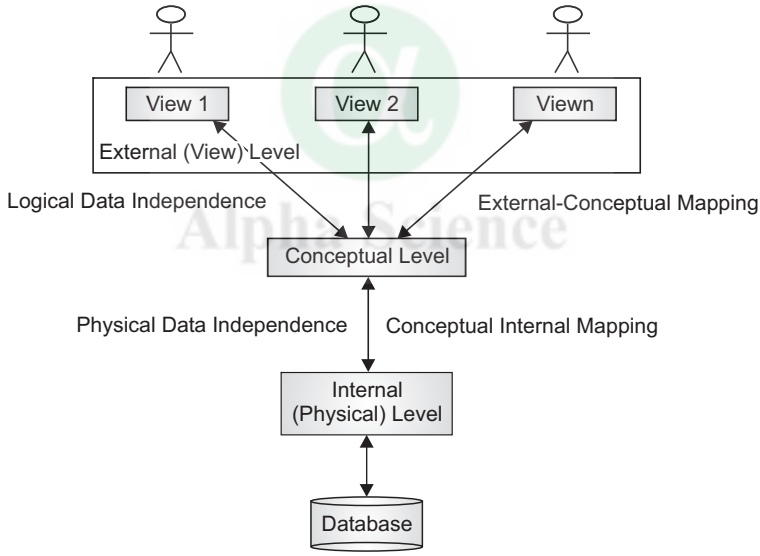


Figure 6.5

6.4.1 Physical Level/Internal Level

The data stored in the database in raw machine form is Physical level. It is the lowest level of abstraction and specifies how the data is actually stored. It describes the complex data structure in details. File organization is a way the data is stored, so that it can be retrieved when needed, including the physical order and layout of records on storage devices'. File organization is the technique used to represent and store the records on a file.

Different types of storage media used for storing, taking backup and data processing:

6.4.1.1 Magnetic Tape

Magnetic Tapes have following properties:

- Is a non volatile storage media.
- In the Magnetic Tape, a magnetically coated strip of plastic is used to store data in encoded form.
- Tapes for computers are similar to the tapes which are used to store music.
- Storing data on tapes is considerably cheaper than storing data on disks.
- Tapes have large storage capacities, ranging from a few hundred kilobytes to several gigabytes.
- However, accessing data on tapes is much slower than accessing data on disks.
- Tapes are sequential-access media, which means that to get to a particular point on the tape, the tape must go through all the preceeding points. In contrast, disks are **random-access** media because a disk drive can access any point at random without passing through intervening points.
- Since tapes are so slow, they are generally used only for long-term storage and backup. In case the data is required to be used regularly, it must always be kept on a disk.
- Tapes are also used for transporting large amounts of data.
- Tapes come in a variety of sizes and formats.
- Tapes are sometimes called streamers or streaming tapes.
- An additional cost benefit is that the tapes can be erased and reused several times.
- Magnetic tapes are used for storing archival of data and for backups.
- Magnetic tapes are most widely used for transferring data from one organization to another.

Following are the disadvantages of using tapes:

- Only Sequential recording and access is possible.
- Magnetic tape is inconvenient for storing Large Number of small records.
- Non standardization of Drives make compatibility difficult.

Magnetic tapes are plastic types coated with magnetic material (Ferrous Oxide). The tape is divided as per the following:

- Into number of tracks and data.

Storage pattern of data is somewhat similar to the punched paper tape. If there is an increase in number of tracks, the data storage capacity also increases. The records are stored in block forms and are separated by an identification gap called Inter Record Gap (IRG) or Inter Block Gap (IBG). The IRG is a gap that exists between two consecutive records stored in a Magnetic tape. IRG allows the tape drive to detect the end of the record and also to increase the speed required for reading and to slow down after reading. (See Figure 6.7) [15]

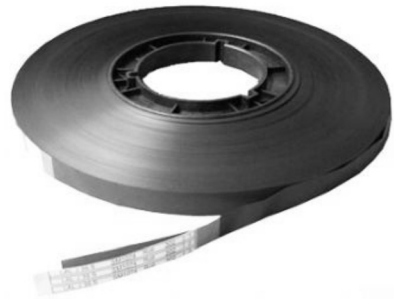


Figure 6.6

6.4.1.2 Magnetic Disks

Since, lot of time is wasted in reading and writing the data on the storage devices like Magnetic Tape, there is a need for devices which requires less access time. By keeping this important factor in view, Magnetic Disks are developed.

Disks have the following properties:

- Are direct access storage devices.
- Are the prime storage medium used with computers for fast access to programs and data.
- Data is stored as a series of magnetized or non magnetized spots on the flat surfaces of one or more disks, there is a common spindle that rotates spinning the disk at the speed of 60 or more revolutions per second. The flat sides of the disk platter are the recording surfaces, the tone arm is the access arm and the needle works as the read/write head.(See Figure 6.7)
- Large storage capacity.
- Disks are reusable.
- Ability to store large number of files in one unit.
- Direct addressing leading to random access, addition and deletion of records.
- Costlier than tapes.
- Fixed disk not portable.

- Is non volatile.
- A disk is always coated with magnetic material.
- A Read/Write head moves across the surface, while the disk is regularly rotating at high speed.
- The disk is divided into number of tracks and sectors.
- There is no effect on the data kept on a disk due to a system failure.
- Read/Write heads must be positioned over the surface of disk for reading or writing data.
- The disk is used to revolve under the read/write heads.

Problem:

Problem such as head crashes occur because of mechanical movement involved in using disks.

Block:

The data is stored and transfered in fixed size units called blocks. Blocks can hold more than one record. Block is a contiguous set of bytes on a single track of a single disk surface. Block size is always a multiple of 1024 bytes.

Physical Record:

A block is also known as page or a physical record.

Logical record:

Data records are known as logical records.

Physical record vs Logical Record:

One physical record contains several logical records.

Blocking Factor:

The number of logical records in a block is called blocking factor. The purpose of blocking is to save space.

Header:

Some overhead information is needed for each physical record on a track. At the beginning of each physical record, there is header. A Header contains the address of physical record.

Purpose of using header:

The purpose of using a header is given as follows:

- To know whether the track is defective.
- The length of the record key.
- The length of the data in the record.
- To know where the record begins.

To break the block into separate logical records in the buffer, to pass the buffer location of the desired logical record to the requesting process. (A record is a collection of fields and the total bytes required to store the entire record is known as Record length). Three types of record formats are generally used.

Fixed Length record:

If all the logical records in the database files have the same length, such types of records are known as fixed length records. Therefore, all records occupy the same space. Processing such records is much easier but there is always a wastage of some storage space.

Spanning record:

If there is some space left over the end of the blocks, it would be possible to place a part of a record at the end of one block and the remainder of the record at the beginning of the next block. This technique is known as spanning records. There is one disadvantage with spanning record which is retrieving a record sometimes requires two disk accesses.

Variable length record:

If database files have records with different lengths, then it is known as variable length records or records which reserve varied number of bytes for its records. For each type of record, record length is calculated separately. As a result, the storage space is saved but processing becomes cumbersome. [15]

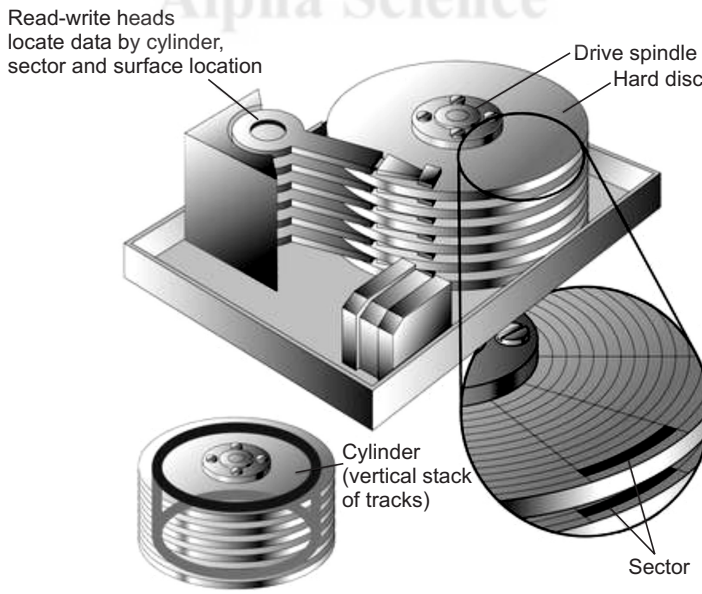


Figure 6.7

6.4.1.3 Main Memory

Properties

- Main memory is a volatile memory. The part of main memory is known as buffer. Buffer holds records for processing. Whenever a record is requested, always the system checks if the record is already in the buffer. If the record is already in the buffer, it passes the address in the buffer to the requesting process. If the record is not in the buffer, then the system finds the location on the disk where the record is stored and directs the reading record into buffer.
- Main memory is directly accessible to the CPU.
- Main memory reads more than one record at a time.
- These days, primary memory is typically implemented using Dynamic Random Access Memory (DRAM) and Integrated Circuits (IC's.).

Main memory stores data and instructions at the time of program execution. Von-Neumann gave the concept of stored program. According to him, program and data are stored in the same memory unit. Prior to this idea, the computers used to store programs and data on separate memories. Entering and modifying these programs were very difficult, as they were entered manually by setting switches, plugging, and unplugging. The memory unit stores the information in a group of memory cells, also known as memory locations in the form of binary digits. Each memory location has a unique address and can be addressed independently. By referring to the address of the memory location, we can provide the contents of the desired memory locations to the CPU. The amount of information which can be transfered between CPU and memory depends on the size of BUS connecting the two. Instructions will be executed in Von Neumann machine in a sequential manner unless the program changes it. There is only a single path between the main memory and control unit in Von Neumann machine.

The DBMS runs in main memory and the processor can only access the data which is currently in the main memory. The handling of the differences between disk and main memory effectively is at the heart of a good quality DBMS.

6.4.1.4 Cache Memory

To use cache memory in an efficient manner, the system will find which data will be needed next time and try to check that the data is in the cache. When data is needed, the system searches first in cache; then in Dynamic RAM. If the data resides in cache at the time of request, then the data accessing will be faster. Cache memory resides between main memory and CPU.

6.4.1.5 Access Methods

Access methods are techniques used to locate and extract stored records. Database processing uses the following access methods:

6.4.1.5.1 Serial Organization

- Records are retrieved in the same order as they were stored in a file, without arranging in a particular order. Serial files are stored in chronological order which means the way each record is received it is stored in the next available storage location. This technique is known as Serial File Organization.
- In a Serial file, there is no importance of key. A serial file is also known as non-keyed sequential file.
- Magnetic tape is the only medium to store serial organization of files.
- This type of file organization means that the records are in no particular logical order, serial file is equivalent to an unordered sequence. To retrieve a single record, the whole file needs to be read from the beginning to end.
- Serial organization is the method used for creating Transaction files (unsorted), Work and Dump files. [5][8]

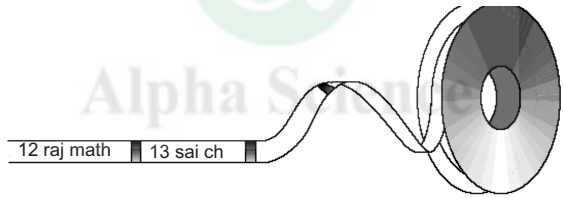


Figure 6.8

Sequential Organization

Sequential files

- Are serial files whose records are sorted and stored in an ascending or descending order on a particular primary key field.
- If the access to the file is strictly sequential, a sequential file is suitable.
- The physical order of the records on the disk is not necessarily sequential, as most manufacturers support an organization, where certain records (inserted after the file has been set up) are held in a logical sequence but are physically placed into an overflow area.
- Overflow area provides additional space for record addition without the creation of a new file. They are no longer physically contiguous with the preceding and following logical records, but they can be retrieved in a sequence.

- A sequential file can be stored on a sequential storage device such as a Magnetic Tape.
- To search a particular record in sequential file, it requires an average access to half of the records in the file.

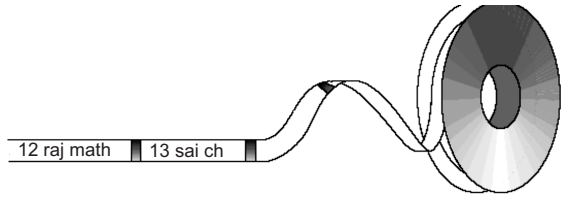


Figure 6.9

Advantages:

- Easy access to the next record.
- Simplicity of organization.
- The absence of auxiliary data structures.
- Well suited for applications that use batch processing of set of records.
- Sequential organization is largely used by DBMS's for producing backups, copies of database kept on tape, or in case the database must be reconstructed after a disk failure.

Disadvantages:

- Simple queries are time consuming for large sequential files because to retrieve less number of records, the entire file is to be processed.
- Whenever you are applying for different sort order in the application, every time a new file is created by storing data in that particular order.
- A single file is an expensive affair because updates require the creation of a new file. Thus, more space is required to store the data. To reduce the cost per update, all requests for updating are batched and sorted in the order of the sequential file and then used to update the sequential file in a single step. File containing the updates to be made to a sequential file is known as a transaction file.
- It is not possible to insert a new record in the middle of the file.
- Record insertion, deletion, and update are done by rewriting the entire file.
- Such file organization does not suit the application, where only a few records are retrieved and processed from a file.

6.4.1.5.2 Indexed Sequential Organization

To improve the query response time of a sequential file, the concept of Indexed Sequential File came into the picture.

- A sequential file which is sorted on primary key and is indexed is called an Index Sequential File.
- Index provides random access to records, while sequential nature of the file provides an easy access to the subsequent records as well as sequential processing. This type of organization is logically same as sequential organization, but an index is built indicating the block containing the record with a given value for the Key field. This method combines the advantages of a sequential file with the possibility of direct access using the Primary Key (the primary Key is the field that uniquely identifies the records).
- These days manufacturers provide Indexed Sequential Software which allow the building of indexes using fields other than the primary Key.
- These additional fields on which indexes are built are called Secondary Keys.
- There is no impact of indexes on the physical organization of records which means different indexes use the same file without rearranging the records of file.
- Insertion, updation and deletion are easier with Indexed Sequential File organization.
- Records can be updated in place by simply writing over the old record within the same track which means immediate updating is facilitated by Indexed Sequential File organization.
- For deletion, we locate the track and put a deletion flag at the beginning of the record to indicate that it should not be read.
- Insertion is also easy because we can leave some room on each track for additional records. By this, there is no need to rewrite the entire file whenever we need to insert a record. Indexed Sequential Files can only be stored on Direct Access Storage Devices.
- More space will be consumed to store indexes. Since, a file occupies many tracks, the track index will be very long, and searching the index might be very time consuming. In this case, we can set up another level of index.
- Retrieval time is more as compared to retrieval time in direct access organization.

Example:

Take an application which requires to list all the teachers of department of Computer Science from Teachers' file which is sorted department wise. In this scenario, Computer Science will be searched for an index for its address. Once the location is found, the records in that file will be searched in a sequential manner.

Index

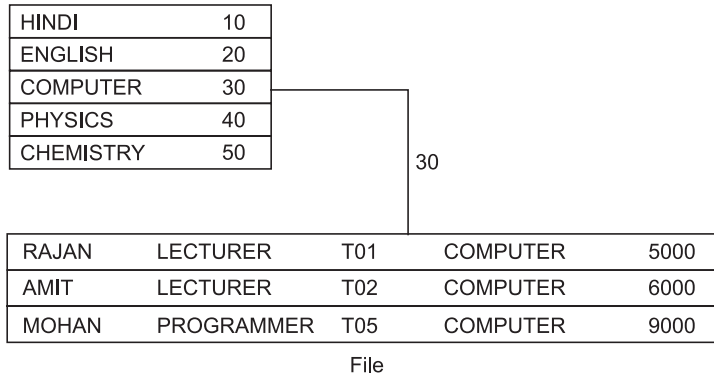


Figure 6.10

Finding Record having Department Computer using Index

6.4.1.5.3 Random (or Direct) File

In the indexed sequential file organization, the mapping from the search key value to the storage location is via index entries, while the key value is mapped directly to the storage location in direct file organization.

- This direct mapping is done by performing some arithmetic manipulation of the key value.
- A randomly organized file contains records arranged physically without regard to the Sequence of the primary key.
- Records are loaded on the disk by establishing a direct relationship between the Key of the record and its address on the file; normally by the use of a formula (or algorithm) we convert the primary Key to a physical disk address. This relationship is also used for retrieval. The use of a formula (or algorithm) is known as **Key Transformation**. This process is known as hashing.

  Key value…….. >>……Hash function……>>…….Address

- A Hash table is a simple index structure which a database can implement. The component of a hash index is the **hash function** and the **buckets**. It is the responsibility of DBMS to construct an index for every table you create that has a PRIMARY KEY attribute also.

Example:

```
CREATE TABLE employee (
 id Number PRIMARY KEY
 , Employeename varchar(50)
);
```

In the table employee, Lets store four rows as follows:

insert into employee values (1,'rekha');insert into employee values (2,'amitabh');

insert into employee values (3,'ajitabh');insert into employee values (4,'isha');

- The algorithm breaks the places in which the rows are to be stored into areas, known as buckets.
- If a row's primary key matches the requirements to be stored in that bucket, then the primary key will be stored in bucket.
- The algorithm which decides the bucket to be used is known as the hash function. For example, we assume to have a hash function where the bucket number equals the primary key. When the index is created, we also have to decide that how many buckets are there. In this example, we have four buckets. (See Figure 6.11)

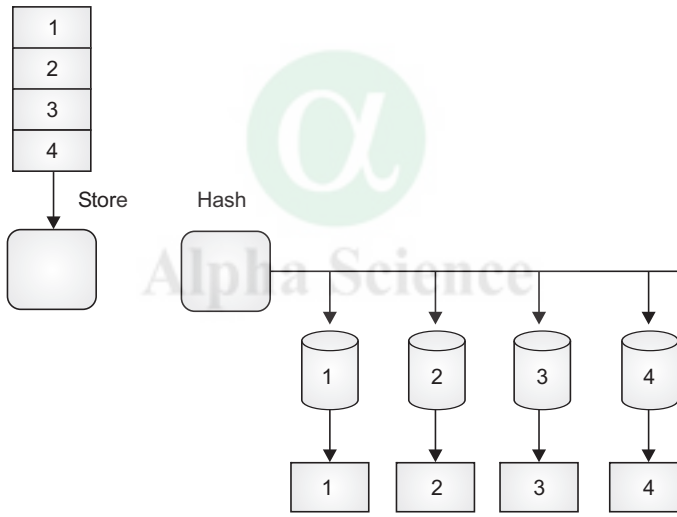


Figure 6.11: Hash Table with no collisions

Now, we can find id 3 quickly and easily by visiting bucket three and looking into it. But now the buckets are full. To add more values, we will have to reuse the buckets.

- If we have to put more than one row in some of the buckets, then a hash collision occurs. The more collisions we have, the longer is the collision chain and slower will be the system.
- In Database Management Systems, we can usually ask for a hash index for a table, the number of buckets required. This approach is better when we know how many rows are likely to be. Most systems will handle the hash table for you. Hashing remains a popular indexing technique.

- Instant retrieval to desired record is possible in Random File Organization. In such type of file organization, Records are stored on faster storage media like disks. More processing time is consumed in record address calculation. [15]

6.4.1.6 Data Structure

6.4.1.6.1 Tree

A tree is defined as data structure that consists of a hierarchy of nodes. Nodes contain data and are connected by lines. Root node at the highest level of tree. The root node can have any number of children. A node that has no children is known as leaf node. If the nodes are children of the same parent, then they are called siblings. The root of the tree is assigned to level 0. Its children are on level 1. Their children are on level two, and so on. The height or depth of a tree has the maximum number of nodes on the longest hierarchical path from the root to a leaf. A tree is said to be balanced, if every path from the root node to a leaf has the same length. The degree or order of a tree is the maximum no. of children any node has.

Binary Tree

DBMS use a data structure to hold indexes. Binary tree is always of order two in which each node has no more than two children.

Note: A binary tree has much smarter approach than hash tables and attempts to solve the problem of not knowing how many buckets you might need and some collision chains which might be much longer than others. It attempts to create indexes wherein all rows can be found in a similar number of steps through the storage blocks.

B+ Tree

- B+ Tree can be used to store an efficient and flexible hierarchical index that provides both sequential and direct access to records.
- The index contains two parts, one is index set and another is sequence set.
- With B+ tree, the order of the original data is maintained in its creation order. This allows multiple B+ tree indices to be kept for the same set of data records.
- A B+ tree is a data structure in which records associated with the search keys are at the leaves of the tree.
- This provides efficient retrieval, insertion and removal of records. Keys are triplicate to the non-leaf nodes to provide a path to the searched record.
- In a B+ tree all the keys which are in non leaf node, will point to next leaf, so we can traverse it sequentially.

- B+ Tree is an advanced technique of the B tree, which is used for sorting data.
- In the B+ tree, all keys are maintained in leaves, and the keys are replicated in non leaf nodes to define path for locating individual records. The leaves are linked together to provide a sequential path for traversing the keys in the tree.
- A B+ tree is a type of tree which represents sorted data in a way that allows efficient insertion, retrieval and removal of records, each of which is identified by a key.
- It is a dynamic, multilevel index, with maximum and minimum bounds on the number of keys in each index segment (usually called a 'block' or 'node').
- In a B+ tree, in contrast to a B-tree, all the records are stored at the lowest level of the tree. Only keys are stored in interior blocks.
- The primary value of a B+ tree is in storing data for efficient retrieval in a block-oriented storage context. Given a storage system with a block size of b, a B+ tree which stores a number of keys equal to a multiple of b will be very efficient when compared to a binary search tree (the corresponding data structure for non-block-oriented storage contexts).
- NT File System, JFS2 file system and Relational database often use this data structure for indices.

B+ Tree Example

B^+ trees are slightly different than binary trees in which the key and data are separated. B^+-trees are commonly used in disks.

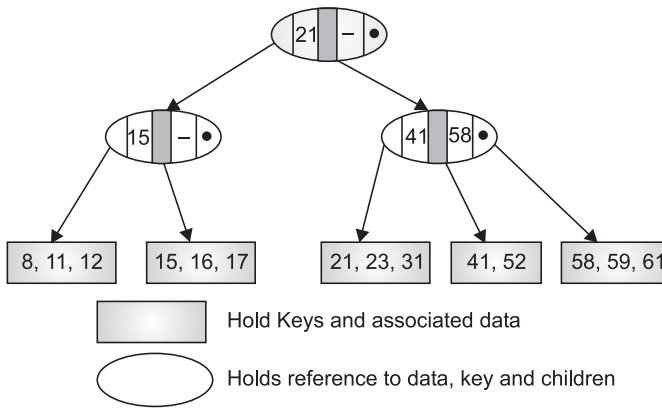


Figure 6.11(a)

Insertion

By adding 1,

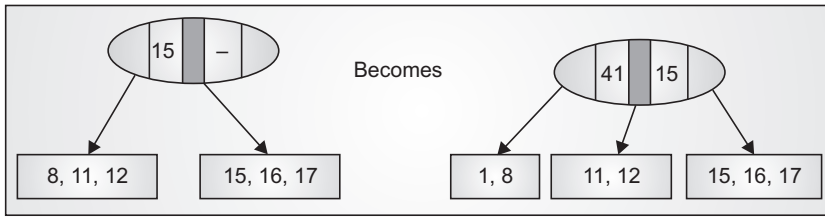


Figure 6.12

Nodes can also propagate up if there isn't a spare child available.

6.4.2 Logical Level/Conceptual Level

The logical information resides at the conceptual level. This logical information will be visible to the users in the high level form. It is the next level of abstraction and describes which data is stored in the database and what relationship exists between various data. It is less complex than physical level and specifies simple structures. Though the detail of physical level is required at logical level but the users of logical level need not know these details.

6.4.3 View Level/External Level

All the user application programs used by the user are of external level. This level contains the actual data which is shown to the users. This is the highest level of abstraction and the user of this level need not know the actual details of data storage.

If you commit any changes in the Physical level *i.e*. changing the access and storage strategies, then there will not be any effect on the conceptual level. This is known as **Physical Data Independence**. On the other hand, if there is any change in the conceptual level *i.e*. if you are changing the logical structure of the database System (removing or adding the attributes), then there will not be any effect on the External level. This is called **Logical Data Independence**.

6.5 DATABASE LANGUAGES

For retrieving the data from the database, manipulating the data, defining and storing the data into the database, some computer programming language is required. Generally database languages are fourth generation (Non procedural) programming languages. The example of a database language is SQL (Structured Query Language).The following are the two basic database languages:

6.5.1 Data Definition Language (DDL)

DDL statements are compiled, resulting in a set of tables stored in a special file called data dictionary. This is also known as Meta Data or data about data. DDL is used to define the storage structure of the database and to define the access methods. The Basic idea is to hide implementation details of the database schemes from the users.

6.5.2 Data Manipulation Language (DML)

DML is used to retrieve the information from the database, insertion of new information into the database, deletion of information in the database and updation of information in the database. A DML is a language which enables users to access and manipulate data. The goal is to provide efficient human interaction with the system.

Types of DML

- **Procedural:** the user specifies what data is needed and how to get it (PL/SQL).
- **Nonprocedural:** the user only specifies what data is needed (SQL).

DML is easier for the user. DML is unable to generate code by as efficiently as that produced by procedural languages.

A **query language** is a portion of a DML involving information retrieval only.

6.6 DATABASE SCHEMA

- It is the overall structure of a database.
- Database schema is usually the graphical presentation of the whole database which is also known as schema diagram.
- The description of the database is called the database schema, which is specified during database design and is not expected to change frequently.
- Tables are connected with external keys and key columns.
- While accessing data from several tables, database schema will be required in order to find the joining data elements and in complex cases to find proper intermediate tables.
- Some database products use the schema to join the tables automatically.
- Database system has several schemas according to the level of abstraction. The Physical schema describes the database design at physical level. The Logical schema describes the database design at logical level. A database can also have sub-schemas (view level) that describe different views of a database.

6.7 DATABASE INSTANCE

Databases change over time. The information in a database at a particular point of time is called an instance of the database.

6.8 META-DATA

"Meta-data contains definition or description of data."

or

"Meta Data is definitional data."

Meta Data provides information about other data managed within an application or environment.

Example: Meta-data will document data about:

- Data elements or attributes (name, size, data type, etc.)
- Data about records or data structures (length, fields, columns, etc.)
- Data about data (where it is located, how it is associated, ownership, etc.)
- Meta-data may include descriptive information about the context, quality and condition, or characteristics of the data.

6.9 DATABASE MANAGER

The 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.

1. Databases require a lot of storage space (gigabytes). This must be stored on disks. Data is moved between disk and main memory as and when needed.
2. The goal of the database system is to simplify and facilitate access to data. Performance is very important. Views provide simplification.

6.9.1 Responsibilities of Database Manager

1. **Interaction with the File Manager:** Storing raw data on disk; using the file system usually provided by a conventional operating system. The database manager must translate DML statements into low-level file system commands (for storing, retrieving and updating data in the database).

2. Integrity enforcement: Checking that updates in the database does not violate consistency constraints.

3. Security enforcement: Ensuring those users who have access to the information and they are authorized to use.

4. Backup and recovery: Detecting failures due to power failure, disk crash, software errors, etc., and restoring the database to its state before the failure.

5. Concurrency control: Preserving data consistency when there are concurrent users. When multiple users are accessing the same data at the same time is known as concurrency.

Note: Some small database systems may miss some of these features, resulting in simpler database managers (For example, no concurrency is required on a PC running MS-DOS.). These features are necessary on larger systems.

6.10 DATABASE ADMINISTRATOR

The database administrator is a person having central control over data and programs accessing that data.

6.10.1 Responsibilities of Database Administrator

1. Defining the schema: It is the creation of the original database schema. This involves writing a set of definitions in a DDL (data storage and definition language), compiled by the DDL compiler into a set of tables stored in the data dictionary.

2. Defining the storage structure and access method: Writing a set of definitions translated by the data storage and definition language compiler

3. Modification of schema and physical organization: Writing a set of definitions used by the DDL compiler to generate modifications to appropriate internal system tables (e.g. data dictionary).

4. Controlling data: Granting different types of authorization for data access to various users. Revoking different types of authorizations for data access from various users.

5. Specifying integrity constraints: DBA is responsible for generating integrity constraints. These are consulted by the database manager module whenever updates occur.

6.11 DATABASE USERS

The following are the categories of database users:

Application programmers are computer professionals interacting with the system through DML calls embedded in a program, written in a host language (*e.g.* C, PL/1, and Pascal). The DML pre-compiler converts DML calls (prefaced by a special character like $, #, etc.) to normal procedure calls in a host language.

Then the host language compiler generates the object code.

Sophisticated users interact with the system without writing programs. They make requests by writing queries in a database query language.

These are submitted to a query processor that breaks a DML statement down into instructions for the database manager module.

Specialized users are sophisticated users writing special database application programs. These may be CAD systems, knowledge-based and expert systems, complex data systems (audio/video), etc.

Naive users are unsophisticated users who interact with the system by using permanent application programs.

6.12 COMPONENTS OF DATABASE SYSTEM

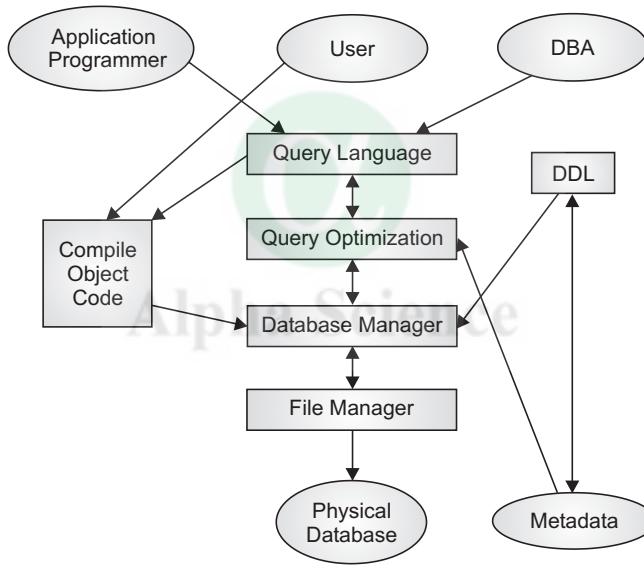


Figure 6.13

1. File Manager: Manages allocation of disk space and data structures used to represent information on disk.

2. Database Manager: The interface between the low-level data and application programs or queries.

3. Query processor: It translates statements in a query language into low-level instructions which the database manager can understand.

4. DML precompiler: It converts DML statements embedded in an application program to normal procedure calls in a host language. The pre-compiler interacts with the query processor.

5. DDL compiler: It converts DDL statements to the set of tables containing metadata stored in a data dictionary.

In addition, several data structures are required for physical system implementation:

1. Data files: Store the database itself.

2. Data dictionary: Stores information about the structure of the database. It is used heavily. Great emphasis should be laid on developing a good design and efficient implementation of the dictionary.

3. Indexes: Indexes provide a fast access to data items holding particular values.

6.13 DATA MODELS

Data models can be defined as "a collection of conceptual tools for describing data, data relationships, data semantics and data constraints".

A data model is an elaboration of both a container for data and methodology for storing and retrieving data from that container. Data models are abstractions, oftenly mathematical algorithms and concepts. Data models are basically the arrangement of records in various ways. Data can be arranged in various ways using different algorithms or data structures. We can categorize the data models in the following categories:

6.13.1 Hierarchical Data Model

Data is stored in terms of records in Hierarchical Data Model; and the records are organized in the form of collection of trees. A hierarchical database consists of the following:

- It contains nodes connected by branches.
- The top node is called the root.
- If the multiple nodes appear at the top level, the nodes are called root segments.
- Each node has exactly one parent except the root node.
- One parent may have many children.

The Hierarchical Database Model defines hierarchically-arranged data. A single table acts as the parent of the database, from which other child tables branch out. A child node may only have one parent node but a parent can have multiple children. Parents and children are linked together by pointers. A parent will have a list of pointers to each of their children. In a hierarchical data model, the problem in insertion, updation and deletion exists (See Figure 6.14). We can solve these problems by using some indirect methods. One cannot add a record to a child table until it has already been incorporated into the parent table. This might be troublesome. The hierarchical database model still creates repetition of data within the database. In this case, there could be redundancy. If there is redundancy in the database, then inconsistency may arise. Hierarchical databases only handle one-to-many relationships but do not handle many-to-many relationships. This is because a child may only have one parent.

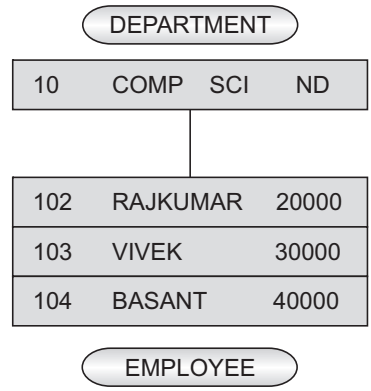


Figure 6.14

In the above figure, a Department has many employees namely RAJKUMAR, VIVEK and BASANT with employees number and salary.

Though this problem can be solved using some indirect methods, the solution is very complex and awkward. IMS (Information Management System) released by IBM in 1966 is one of the earliest DBMS's, and it is ranked as the dominant system in the commercial market for support of large-scale accounting and inventory systems.

6.13.2 Network Data Model

The evolution of Network Data Model was introduced to solve serious problems which existed in the Hierarchical model. Data is stored in terms of records in Network Data Model; and the records are organized in the form of collection of Linked List nodes. The Network Data Model uses a Linked List data structure, in which a record can have many parents as well as many children. It can be represented as follows (See Figure 6.15):

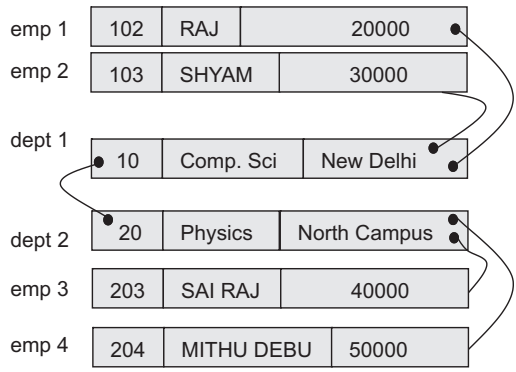


Figure 6.15

The Network Data Model supports many-to-many relationship amongst records. In Network model, the child tables were allowed to have more than one parent. This allows the network model, to support many-to-many relationship. Records are connected to each other through pointers. Unlike hierarchical model, the problem of insertion, updation and deletion anomalies will not arise in the network data model. The structure of network data model will be complex, if we are dealing with very large data. The model was difficult to implement and maintain. Most implementations of the network model were used by computer programmers rather than real users. In comparison to hierarchical data model, the network data model is more versatile. The Network model solves the problem of data redundancy by representing relationships in terms of pointers. The model had its origins in the Conference on Data Systems Languages (CODASYL) which had created the Data Base Task Group to explore and design a method to replace the hierarchical model. This was used in Honeywell's Integrated Data Store, IDS.

To summarize, Data is stored along with pointers, which specifies the relationship between entities. Network model is complex. It is difficult to understand both the way data is stored and the way data is manipulated (as clear in Figure 6.15). It is capable of supporting many-to-many relationship between entities, which a hierarchical model doesn't.

6.13.3 Relational Model

Dr. E. F. Codd introduced the relational model in 1970 and it has evolved since then, through a series of writings and updation. In a relational database, the data and relations between them are organized in the form of two dimensional tables. A table is a collection of records and each record in a table contains the same field. A table is composed of the Rows and columns. Rows are known as records/tuples and columns are the attributes/fields of the table. Total no. of rows in a table is called cardinality of table. Cardinality of a table always changes with time. This depends on the condition applied by the user. Suppose in a table the number of records are three, then the cardinality of that table will be 3 on that particular instance. The total number of attributes in a table is known as the degree of that table. Suppose any table is having four attributes, then the degree of that table will be 4. The organization of data into relational tables is known as the logical view of the database *i.e*. the form in which a relational database presents data to the user and the programmer. The way in which the database software physically stores the data on a computer disk system is called the internal view. The internal view differs from application to application.

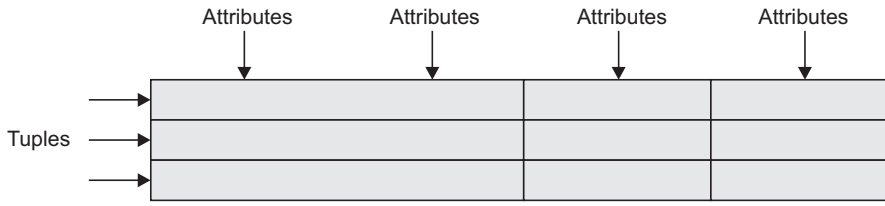


Figure 6.16: Relations in the Relational Data Model

A relational database allows the definition of data structures, storage and retrieval operations and integrity constraints. Some properties of relational data model are given below:

- Values are atomic
- Each row is unique
- Column values are of the same kind
- The Sequence of Columns are insignificant
- The Sequence of rows are insignificant
- Each Column has a unique name; certain fields may be designated as keys, which mean that search for specific values of that field will use indexing to speed up. In case, when fields in two different tables take values from the same set, a joint operation can be performed to select related records in the two tables by matching values in those fields. Often, the fields will have the same name in both the tables. This can be extended to join multiple tables on multiple fields. The **Relational Database** model is based on the **Relational Algebra**.

In 1985, Codd published two articles. In these articles, he suggested some rules and principles for a Database Management System. Any DBMS that follows all the rules given by Codd will be complete Relational Database Management System. He gave the following rules which are popularly known as Codd's twelve rules:

Zero Rule: The management of stored data can be done only by using RDBMSs relational capabilities.

First Rule: All information must be represented as values in tables and at logical level.

Second Rule: RDBMS should access any data item in the database by giving its table name, Column name and primary key value.

Third Rule: RDBMS must represent Null values in a systematic way, regardless of the data type of the item.

Fourth Rule: The RDBMS must maintain the System to have the logical description of the database and must be represented in the same way as ordinary data.

Fifth Rule: RDBMS must support the data sublanguage that allows statements expressed as character strings to support data definition, definition of views, data manipulation, integrity rules, user authorization and a method of identifying units of recovery.

Sixth Rule: RDBMS must support the view.

Seventh Rule: RDBMS must support the insertion, deletion and update operations without any anomaly.

Eighth Rule: RDBMS must support the physical data independence and logical data independence concepts.

Ninth Rule: RDBMS must support the integrity rules.

Tenth Rule: Concurrency must be controlled by the RDBMS.

Eleventh Rule: RDBMS must support Distribution independence in case database is distributed.

Twelfth Rule: There should be no way to modify the structure other than through multiple row database language like SQL which is called **non subversion rule**.

RDBMS must support automatic backup and recovery.

A basic understanding of the relational model is necessary to use relational database software such as **Oracle**, **Microsoft SQL Server**, or even personal database systems such as **Access** or **FoxPro** effectively which are based on the relational model.

6.14 ASSOCIATIONS

There could be more than one table related to each other. The tables could be associated with each other in the following manner:

1. One to One Association: An entity in entity set X is associated with maximum one entity in entity set Y, and entity in Y is associated with maximum one entity in entity set X.

Let's understand this with the following example:

If there is one employee in each department and each department is having only one employee (head of the department). Such type of association is known as 1:1 association. The cardinality of the association is one to one in this case.

Figure 6.17

2. One to Many Association: An entity in entity set X is associated with any number of entities in entity set Y (any number of entities means that it can be zero, one or many), and an entity in entity set Y is associated with at most one entity in entity set X. Lets understand this with an example as follows:

If more than one employee is working in one department but one employee cannot work in more than one departments. This is the example of one to many association. This is represented by 1 : *m*. The cardinality of such type of association will be 1 : *m*.

Figure 6.18

3. Many to Many association: An entity in entity set X can be associated with any number of entities in entity set Y and an entity in entity set Y can be associated with any number of entities in entity set X. Lets understand this with an example as follows:

If more than one employee is working on one project and more than one project is being handled by one employee. Such type of association is many to many (m:n) association. The cardinality of the association in this case is m:n.

Figure 6.19

4. Many to One association: An entity in X is associated with maximum one entity in entity set Y, and one entity in entity set Y can be associated with any number of entities in entity set X. Lets understand this with an example:

If more than one employee is working on one project, such type of association is m : 1 association. The cardinality of the association in this case will be m : 1.

Figure 6.20

Cardinality of association in a table refers that how many records of one table is being associated with another table. Association between the tables is

required because if we want to extract the data from more than one table, then there should be some associations between the tables. Association between the tables can be established in different ways as described above. We can define the degree of association also. If in the association two tables are involved, then the degree of that association is two. If there is an association of three tables, then the degree of that association will be three. At the time of associating the tables, we consider two types of participation constraints in relationship sets, one is Total participation and another one is Partial participation. Following is the explanation of the Total participation and Partial participation.

If each entity in entity set X is participating at least in one relationship with entity set Y, this is known as **Total Participation**. In ER diagram, total participation is represented by double lines between entity set X and relationship set. If some of the entities of entity set X is not participating in relationship with entity set Y it is known as **Partial Participation**. In ER diagram, partial participation is represented by single line between entity set X and relationship set.

Note: Data Objects are representation of any composite information that is processed by software. A data object can be an external entity, thing, occurrence, event, role, an organizational unit, place or a structure. The description of the data object includes data object and its attributes. A data object contains only data.

Data Attributes name a data object, describe its characteristics and sometimes make reference to another object. One or more attributes must be identified as a key which acts as an identifier.

Relationships indicate the manner in which data objects are connected to one another.

Cardinality of a relationship is the actual number of related occurrences for each of the two entities. It defines the maximum number of objects participating in a relationship. It does not indicate whether a data object should participate in relationship or not.

Modality of a relationship can be 0 or 1. It is 1, if an occurrence of relationship is must. It is 0, if an occurrence of relationship is optional.

6.15 INTEGRITY

Integrity is required to protect the data and structure of the table. To protect the data and its structure, we define the integrity constraints. Integrity constraints are used to ensure accuracy and consistency of data in a relational database.

Following are the two integrity rules:

- **Entity integrity:** The value of primary key cannot be null which means in any table, if one attribute is set as primary key, then you will have to

enter the data for that attribute. You cannot leave the value of that attribute blank.

- **Referential Integrity:** Every Foreign key refers its primary key attribute. Foreign key is required if we want to establish the association between the tables. If we set the value of a foreign key null, then the tables will be de-associated. Lets see the example given below:

If there is an association between table of type 1 : m, then there should be one attribute which is common in both the tables (For example department number attribute in both employee and department table). Relationship between the department and employee table is 1 to many because, one department is having more than one employee; one employee cannot be in more than one department. So, the department table is uniquely identifying each employee See figure 6.21(b). In this case, department is the parent table and employee is the child table. The common attribute of the association in the parent table will be a primary key and in the child table the common attribute will be a foreign key. A table will be known as a strong table if it contains the primary key and weak table, if it does not contain the primary key. There could be more than one foreign key in a table to associate with different other tables.

We have described the concept of foreign key with examples in SQL chapter.

Codd, initially defined two sets of constraints, but in his second version of the relational model, he came up with five integrity constraints:

- Entity integrity
- Referential integrity
- Domain integrity
- Column integrity
- User Defined integrity

Domain integrity

If the value of a column is derived from the domain, then Data is said to contain domain integrity. For example, column employee number must be a valid number. All valid employee numbers form one domain. If the value of employee number is an invalid number, then it is said to violate domain integrity.

Column integrity

Column integrity refers to the requirement that data stored in a column must adhere to the same format and definition. This include data type, data length, default value of data, range of possible values, whether duplicate values are allowed, or whether null values are allowed.

User-defined integrity

We need to implement our own code logic to make sure data is saved accurately and consistently across all business domains. Code logic can be implemented by using database triggers, stored procedures or functions etc.

6.16 CONCURRENCY

When multiple users access the same resource at the same time, concurrent problem arises. In the case of File Server, Concurrency is not possible whereas in the Client Server Network Concurrency is possible. The concurrency is controlled through locking techniques. In Oracle RDBMS, three types of locking techniques are possible (Table level locking, Row level locking and Page level locking).

Table level locking: In the table level locking, all the users can read the data but the table will be locked for one user and only he will be able to write. If there is no **where** clause in the query, a table level lock is used.

Row level locking: In the table, every row will be locked for one user only. That user can write the data into that particular row. Please remind that every user can read the data from all the columns of that row. If the **where** clause in a query evaluates to only one row in the table, a row level lock is used.

Page level locking: If **where** clause in a query evaluates a set of data, a page level lock is used.

Note: Oracle supports two types of locking: Explicit locking (user locking) and implicit locking. Implicit locking is done by the RDBMS itself. Explicit locking, always overrides Oracle's default locking strategies.

Note: Implicit locking is implemented internally by the System, whereas Explicit Locking is implemented through the user.

The Concurrency Control procedures can be implemented only through Database Administrator.

6.17 DATABASE BACKUP AND RECOVERY

A DBMS is said to be an ideal DBMS, if there is an automatic backup and recovery facility. A DBMS should maintain a log of all transactions. Transaction is the execution of a set of DML statements within a given session. By maintaining a log of all transactions, DBMS can recover the data in case of any catastrophic failures like occurrences of power failures, disk crash etc. Automatic backup facility should be provided by the DBMS.

6.18 KEYS

Attributes are the properties of a table. Key is a unique identification property of a row. By defining a key attribute on a table, we can stop the redundancies. If one attribute in a table is defined as a key attribute, then that key attribute will have unique value in each row. The following are the different categories of key:

1. **Alternate key:** An alternate key is any candidate key which is not selected to be the primary key.

2. **Candidate key:** A candidate key is a field or combination of fields that can act as a primary key field for that table to uniquely identify each record in that table.

3. **Composite key:** is a key that consists of two or more attributes.

4. **Primary key:** A primary key is a value that can be used to identify a unique row in a table. Attributes are associated with it. Examples of primary keys are Social Security numbers (associated to a specific person) or ISBNs (associated to a specific book).

In the relational data model, a primary key is chosen as the main method of uniquely identifying a tuple in a relation.

A **primary key** is a candidate key (there may be more than one) chosen by the Database designer to identify entities in an entity set.

5. **Super key:**

- A **super key** is a set of one or more attributes; which collectively, allow us to identify uniquely an entity in the entity set. Super key differs from a candidate key in the sense, that it does not require the non redundancy property.
- A super key may contain extraneous attributes, and we are often interested in the smallest super key. A super key for which there is no subset as a super key is called a **candidate key**.

6. **Foreign key:** a foreign key (FK) is a field or group of fields in a database record that points to a key field or group of fields forming a key of another database record in other table. Usually a foreign key in one table refers to the primary key (PK) of another table. This way references can be made to link information together and it is an essential part of database normalization.

An entity set that does not possess sufficient attributes to form a primary key is called a **Weak Entity Set (as in figure 6.21(b) dependent entity set)**. One that does have a primary key is called a **Strong Entity Set (as in figure 6.21(b) Employee entity set)** as we have discussed earlier.

For a weak entity set to be meaningful, it must be a part of one-to-many relationship set. Member of a strong entity set is a dominant entity. Member of a weak entity set is a subordinate entity. A weak entity set does not have a primary key, but we need a means of distinguishing among the entities. The discriminator of a weak entity set is a set of attributes that allows this distinction to be made (as in Figure 6.21(b) name attribute is working as a discriminator). The primary key of a weak entity set is formed by taking the primary key of the strong entity set on which its existence depends plus its discriminator.

6.19 E-R MODEL

During the requirement analysis phase of the software development life cycle (SDLC), we gather all the information related to the organization and product. Then, we gather the information which is important to the organization and product. After this, we do conceptual modeling using E-R Model. (E-R Model stands for Entity Relationship Model). An ENTITY is a 'thing' which can be distinctly identified and has its existence in the real world. For example a person, a car, a subroutine, a wire, an event. An entity is also an object or concept about which you want to store information. We can also establish relationship among entities through ER model. RELATIONSHIP is an association among entities, *e.g*. department OWNS employees; is an association between a department and employee.

6.19.1 Attribute, Value, Domain and Entity Set

The information about one entity is expressed by a set of (attribute, value) pairs, *e.g*. a Department could be:

Name = Computer Science
DNO = 20
LOC = Rishikesh

Values of attributes belong to different value-sets or domains.

Entities defined by the same set of attributes can be grouped into an ENTITY SET.

A given set of attributes may be referred as an entity type. All entities in a given Entity Set are of the same type, but sometimes there can be more than one set of the same type. A set of all the persons who are customers at a given bank can be defined as an entity set customer. The individual entity that constitutes a set are said to be an extension of an entity set. So, all the individual bank customers are the extension of an entity set customer.

Each entity has a value for each of its attributes. For each attribute, there is a set of permitted values called domain or value set.

6.19.2 Attributes and their Classification

A **Simple Attribute** cannot be divided into sub-parts and has got one value for its attribute whereas a **Composite Attribute** is one which can be divided into sub-parts. For example, an attribute name can be divided into first name, middle name & last name. An attribute which has got only one value is known as **Single Valued Attribute**. For example, the employee number attribute will have only one employee number. There may be cases when an attribute has a set of values for a specific entity. For example an attribute phone_no. may have a value zero, one or several phone numbers. This is known as **Multivalued attribute**. The value of derived attribute is not stored into the database. **Derived attribute's** value is derived from the value of other related attributes. For

example, an attribute annual salary can be calculated from another attribute salary from employee entity set or age derived from the date of birth(Where salary and date of birth are the stored attributes). **Stored attributes** are those attributes which can be stored in the database.

You must also note the use of calculated or derived fields. For example, a table contains SALARY, COMMISSION and ANNUALSALARY where ANNUALSALARY is calculated as 12 * SALARY added by COMMISSION. As one of these values can be calculated from the other two, therefore it need not be held in the database table.

The overall logical structure of a database can be expressed in pictorial form by an E-R diagram. Peter Chen developed ERD's in 1976. Later on Charles Bachman and James Martin added some improvements to the basic ERD principles.

6.19.3 Components of E-R Diagram

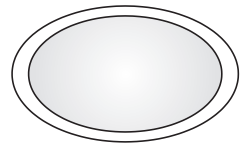
1. Rectangles represent entity set or table.

2. Ellipse represents an attribute.

3. Diamonds represent a relationship set

4. Lines link attributes to entity sets and entity sets to relationship sets.

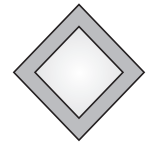
5. Double ellipses represent multi-valued attributes.



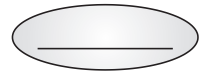
6. Dashed ellipse represents derived attributes.

7. Double Rectangle represents weak entity set.

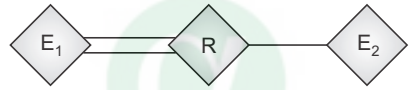
8. Double Diamond represents a relationship set with participation of some weak entity sets.



9. An underlined attribute within ellipse represents a primary key.



10. Total participation of E_1 into R.



11. Partial participation of E_1 into R.

Case Study-I

MOBILE CARD SHOP MANAGEMENT SYSTEM

Lets take a case study of Mobile card shop. The aim of this case study is the computerization of the shop, which includes stock management, purchase & sales of cards and expenditure etc. Due to the competition in the retailing of mobile cards, their operations are getting too large and complicated, therefore, the owner has decided to computerize his regular manual system. The owner can keep an account of sales and expenses of his 2 shops at different locations and to know his actual profits and expenses and if there is any flaws in the system what measures he should take? Basically, he needs a computerized system which can help him in decision making and generate more profits.

The package helps the users and owner by providing him timely and appropriate reports.

Now lets define the structure of various tables:

We will use the following inputs for the case study:

Structure table: tblogin (Login Table)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	Username	varchar2	20	Login Username
2	Passwd	varchar2	15	Login Password

Primary Key: Username

Structure Table: tbmstate (State Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	StateCode	Number	4	State Code
2	StateName	Varchar2	20	State Name

Primary key: StateCode

Structure table: tbmcity (City master)

Table description: CITY DETAIL

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CityCode	Number	4	Unique id to generate the City Code
2	StateCode	Number	4	Code of The State
3	City Name	Varchar2	25	City Name
4	Std	Varchar2	7	Std Code

Primary key: CityCode

Foreign Key: StateCode

Structure Table: tbmmobco (Mobile Company Master)

Table Description: Store the details of mobile company

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CoCode	Number	4	Company Code
2	CoName	Varchar2	20	Company name

Primary key: CoCode

Structure table: tbmprerechtype (Prepaid Recharge type)

Table description: Store the details of prepaid recharge type

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CoCode	Number	4	Company Code
2	PreRechType	Number	4	Prepaid RechargeType
3	TalkTime	Number	4	Talk Time
4	Validity	Varchar2	8	Validity
5	Owcomm	Number	5,1	Owner's Commission

Primary key: CoCode+PreRechtype

Foreign key: CoCode

Structure table: tbmexp(Expenditure Master)

Table Description: Store the expenditure details

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	ExpCode	Number	4	Expenditure Code
2	ExpName	Varchar2	20	Expenditure Name

Primary key: ExpCode

Structure table: tbmiproof (Identity Proof Documents Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	IpCode	Number	4	Identity Proof Document Code
2	IpName	Varchar2	30	Identity Proof Documents Code

Primary key: IPCode

Structure Table: tbmshop (Shop Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	ShCode	Number	4	Shop Number
2	ShName	Varchar2	25	Name of Shop
3	Add1	Varchar2	30	Address 1
4	Add2	Varchar2	30	Address2
5	CityCode	Number	4	CityCode
6	Pin	Number	6	Pin Code
7	Phone	Varchar2	8	Phone No
8	Mobile	Varchar2	10	Mobile Number
9	Email	Varchar2	40	Email.ID

Primary key: ShCode Foreign key: CityCode

Structure Table: tbmemp (Employee Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	EmpNo	Number	4	Employee Number
2	EmpName	Varchar2	25	Name of Employee
3	Add1	Varchar2	30	Address 1
4	Add2	Varchar2	30	Address2
5	CityCode	Number	4	CityCode
6	Pin	Number	6	Pin Code
7	Phone	Varchar2	8	Phone No
8	Mobile	Varchar2	10	Mobile Number
9	Email	Varchar2	40	Email.ID

Primary key: EmpNo Foreign key: CityCode

Structure table: tbmcust (Customer Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CustCode	Number	4	Shop Number
2	CustName	Varchar2	30	Name of Customer
3	Add1	Varchar2	30	Address 1 of Customer
4	Add2	Varchar2	30	Address2 of Customer
5	CityCode	Number	4	CityCode
6	Pin	Number	6	Pin Code
7	Phone	Varchar2	8	Phone No of Customer
8	Mobile	Varchar2	10	Mobile No of Customer
9	Email	Varchar2	40	Email.Id of Customer
10	IpCode	Number	4	Identity proof Document Code

Primary key: Cust_Code Foreign key: CityCode, IpCode

Structure Table: tbcardsaleby (prepaid/postpaid card sale)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	Frmno	Number	8	Form Number
2	Cust_Code	Number	4	Customer Code
3	Empno	Varchar2	4	Employee Number
4	Owcomm	Number	5,2	Owner's Commission
5	Empcomm	Number	5,2	Employee Commission

Primary key: Frmno

Foreign key: Cust_Code, Empno

Structure Table: tbpostcardinfo (Postpaid Card Information)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	Frmno	Number	8	Form Number
2	Date	Varchar2	10	Date in which card sold
3	CustCode	Number	4	Customer Code
4	Empno	Varchar2	4	Employee Number
5	Std	Varchar2	5	STD facility available or not
6	Isd	Varchar2	5	ISD facility available or not
7	Amt. charges	Number	7,2	Amount Charged

Primary Key: Frmno

Foreign key: Cust_code, Empno

Structure Table: tbexptra (Expenditure Transcription)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	ExpDate	Varchar2	10	Expenditure Date
2	ExpTime	Varchar2	12	Specifies the time of an expense
3	ExpCode	Number	4	Expenditure code
4	ShCode	Number	4	Shop Code
5	EmpNo	Number	4	Employee Number
6	Amt	Number	10,2	Amount of expense

Primary key: ExpDate+ExpTime+ExpCode

Foreign Key: ShCode, EmpNo

Structure Table: tbnewprecardstock (New Prepaid card stock)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CoCode	Number	4	Company Code
2	Totcards	Number	4	Total No. of New Cards

Foreign key: CoCode

Structure Table: tbprerechstockpurchase(New Prepaid Stock)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	CoCode	Number	4	Company Code
2	PreRech Type	Number	4	Prepaid Recharge Type
3	TotCards	Number	4	Total No. of New Cards

Primary key: CoCode+PreRechType

Structure Table: tbmdiststock (Distribution Stock Master)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	DistCode	Number	4	Distribution Code
2	ShCode	Number	4	Shop Code
3	CoCode	Number	4	Company Code
4	PreRechType	Number	4	Prepaid Recharge Type
5	TotCards	Number	3	Total Number of Cards to be distributed

Primary Key: DistCode

Foreign key: ShCode+CoCode

Structure Table: tbtDiststock (Distribution Stock transaction)

S.No.	FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
1	DistCode	Number	4	Distribution Code
2	DistDate	Varchar2	10	Distribution Date
3	DistTime	Varchar2	15	Distribution Time
4	Empno	Varchar2	4	Employee Number
5	TotCards	Number	3	Total Number of Cards
6	Frmno	Number	8	Form Number

Primary key: Frmno

Foreign key: Distcode, Empno

The following is the ER diagram of the above case study:

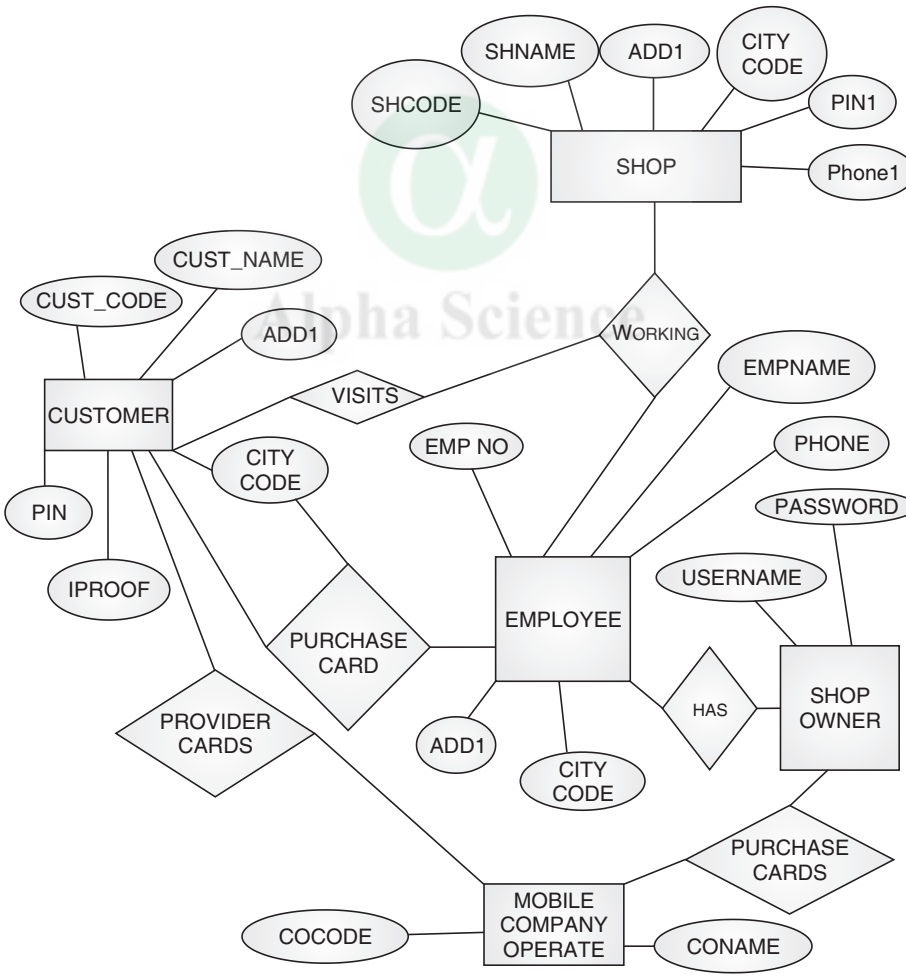


Figure 6.21 (a)

Case Study-II

ARSD College has several departments. Each department has a coordinator and at least one non teaching employee. Non teaching employees must be assigned to at least one, but possibly more departments. At least one non teaching employee is assigned to a project, but a non teaching employee may be on a vacation and not assigned to any project. The important data fields are the names of the departments, projects, coordinators and non teaching employees, as well as the coordinator and non teaching employee number and a unique project number.

1. Identify Entities: The entities in this system are Department, non teaching employee, coordinator and Project. One is tempted to make ARSD College an entity, but it is a false entity because it has only one instance in this problem. True entities must have more than one instance.

2. Find Relationships: We construct the following entity relationship matrix:

entity set / entity set	Department	Non teaching employee	Coordinator	Project
Department		is assigned	Run by	
Non Teaching Employee	belongs to			works on
coordinator	Runs			
Project		uses		

3. Draw ERD: We connect the entities whenever a relationship exists in the Entity Relationship Matrix.

Guidelines for Effective ER Diagrams

1. Make sure that each entity appears only once per diagram.
2. Name every entity, relationship, and attribute on your diagram.
3. Examine relationships between the entities closely. Are they necessary? Are there any relationships missing? Eliminate any redundant relationships. Don't connect relationships to each other.

E-R diagram with an attribute attached to a relationship set.

If a relationship set has also some attributes associated with it, then we link these attributes to that relationship set.

6.19.4 Lets take another simple ER Diagram of Company Database and the steps for converting ER Diagram into relational form

Rule 1: For each non-weak entity, create a relation that includes all the simple attributes of that entity. Do not include Multi Valued attributes or Derived

Attributes at this time. If you have a composite attribute, include only the component attributes. (For example, if you had a composite attribute ADDRESS made up of the component Attributes STREET, CITY, STATE and ZIP, you will include the 4 components and not the composite as fields.) Choose one of the candidate keys as the primary key of the table. If the candidate key you choose to be the primary key is a composite attribute, then all of its component attributes together will become the primary key.

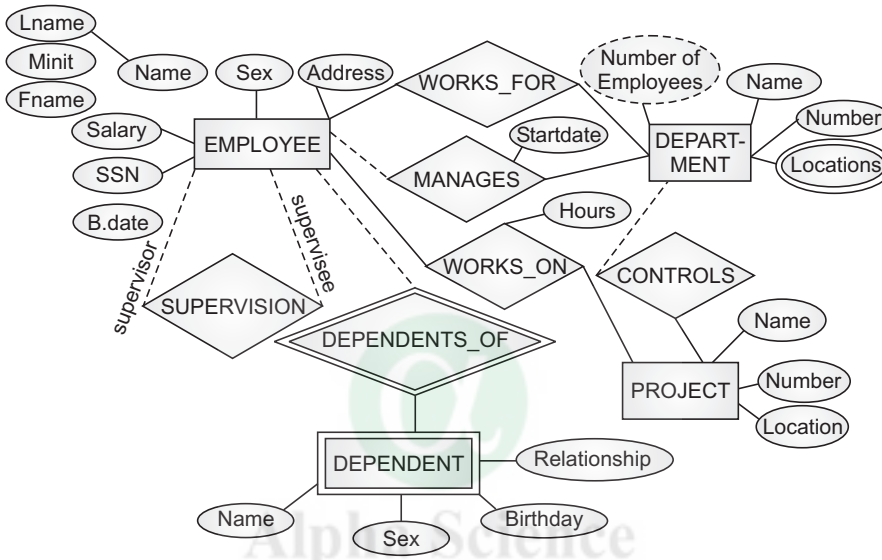


Figure 6.21 (b)

EXAMPLE: From the Company E-R diagram Figure 6.21(b), you will get the following table definitions by following step 1:

EMPLOYEE

Fname	Minit	Lname	SSN	Bdate	Address	Sex	Salary

DEPARTMENT

Name	Number

PROJECT

Name	Number	Location

Rule 2: For each weak entity, create a relation that includes all simple attributes (or simple components of composite attributes) of the weak entity. In addition, include a foreign key attribute as the primary key of the owning entity. The primary key of this relation will be the combination of the primary key of the owning entity and the partial key of the weak entity.

EXAMPLE: The relation for the weak entity DEPENDENT would look like this after rule 2:

DEPENDENT

				FK
Name	Sex	Birthdate	Relationship	Emp_SSN

Rule 3: For each binary 1:1 relationship, identify the two entities that participate in that relationship. Choose one of the entities — preferably the one with total participation in the relationship — and think of it as E1. The other entity is E2. Take the primary key from E2 and include it as a foreign key in E1. If the relationship has simple attributes, include them in the relation for E1.

EXAMPLE: Manages a 1:1 relationship between EMPLOYEE and DEPARTMENT. We choose DEPARTMENT as E1, because it participates in the relationship totally. So after rule 3, the EMPLOYEE and DEPARTMENT relations would look like this:

DEPARTMENT (E1)

	FK		
Name	Number	Mgr_SSN	Mgr_Startdate

EMPLOYEE (E2)

Note: The EMPLOYEE relation is unchanged from rule 1

Fname	Minit	Lname	SSN	Bdate	Address	Sex	Salary

Rule 4: For each non-weak binary 1:N relationship, identify the entity E1 that is at the N-side (the "many" side) of the relationship. The other entity in the relationship is E2. Include it as a foreign key in E1 the primary key of E2. Include simple attributes (or simple components of composite attributes) of the relationship as attributes of E1.

EXAMPLE: We have three 1:N relationships: *Works_for*, *Controls*, and *Supervision*.

Works_for: EMPLOYEE is on the N-side of the relationship, so after applying rule 4 for *Works_for*, it will look like as the following:

EMPLOYEE

								FK
Fname	Minit	Lname	SSN	Bdate	Address	Sex	Salary	Dept_Num

Controls: PROJECT is on the N-side of the relationship, so after applying rule 4 for *Controls*, it will look like the following:

PROJECT

			FK
Name	Number	Location	Dept_Num

Supervision: EMPLOYEE is on the N-side of the relationship in the superviser role, so after applying rule 4 for *Supervision*, it will look like the following:

EMPLOYEE

								FK	FK
Fname	Minit	Lname	SSN	Bdate	Address	Sex	Salary	Dept_Num	Super_SSN

Note that in this case, the Foreign Key Super_SSN has actually come from the SSN in the EMPLOYEE relation, since the EMPLOYEE entity is acting in two different roles in the *Supervision* relationship.

Rule 5: For each binary M:N relationship, create a new relation to represent the relationship. Include in this relation as foreign keys, the primary keys of each of the entities that participate in the relationship. The combination of these foreign keys will make up the primary key for the relation. Also, include simple attributes (or simple components of composite attributes) of the relationship.

EXAMPLE: We have one M:N relationship, *Works_On*. After rule 5, we will have a relation for WORKS_ON that looks like this:

WORKS_ON

FK	FK	
Emp_SSN	Proj_Num	Hours

Rule 6: For each multivalued attribute, create a new relation that includes the attribute plus the primary key of the entity to which attribute belongs as a foreign key. The primary key of this new relation will be the combination of the foreign key and the attribute itself. If the multivalued attribute is also composite, we include only its simple components.

EXAMPLE: We have only one multivalued attribute, the Locations attribute of DEPARTMENT. We create a relation called DEPT_LOCATIONS that will look like this:

DEPT_LOCATIONS

	FK
Location	Dept_Num

Rule 7: For each relationship with 3 or more participating entities, create a new relation to represent the relationship. Include the primary keys of each of

the participating entities in the new relation as foreign keys. Also, include simple attributes (or simple components of composite attributes) of the relationship. The primary key will usually be a combination of all the foreign keys that represent the entities which participate in the relationship. However, if any of the participating entities are on the 1-side of the relationship, then the primary key of the relation should not include the foreign key from that entity.

Total Participation: If all the instances of one relation is participating with other relations in a relationship, it is known as total participation. In ER Diagram, total participation is represented by two parallel lines.

Partial Participation: If all the instances of one relation are not participating with other relations in a relationship, it is known as partial participation. In ER Diagram, partial participation is represented by a single line.

6.20 NORMALIZATION

Lets remember some important points while designing a Relational Schemas (Logical Designing):

- Relational Schema should reflect clear meaning semantically.
- Redundant values in tuples should be avoided.
- Minimum Null values in tuples. If NULL values are unavoidable, make sure that they apply in exceptional cases only.
- No generation of spurious tuples after joining the relational schemas.

To achieve the above stated points, we do normalize relations. Normalization can be defined as,

"Normalization is the process of breaking the Relation into sub relations so that those broken relations should be redundancy less and modification anomalies should not be there".

"The process of breaking the relation into sub relations to efficiently organize the data in a database is called normalization". The purpose of normalization is to remove redundant data (storing the same data in more than one table) to save the wasted storage space and ensuring data dependencies (only storing related data in a table), as we have already defined that database is a collection of related data.

Both these factors are the most important aspects because they reduce the amount of space a database consumes and ensures that data is logically stored. There are five normal forms, which are the guidelines for ensuring that databases are normalized. The lowest form of normalization, refered as the first normal form or 1NF and highest form of normalization, refered as fifth normal form or 5NF. Data normalization is a set of rules and techniques concerned with identifying relationships among attributes, combining attributes to form relations and combining relations to form a database.

The main objective of breaking the relation is to avoid modification anomalies. Modification anomalies come in three forms:

Lets see the following student database:

Course no	Student id	Student name	Faculty-id	Schedule	Room	Grade
X100	206	MOHIT	T1	X	12	A
X100	207	SAHIL	T1	X	12	
X100	208	ANANYA	T1	X	12	B
Y101	209	OM	T2	Y	37	A
Y101	208	UJAAWAL	T2	Y	37	C
Z102	206	OJAS	T3	Z	38	

1. An **insertion anomaly** is a failure to place information of a new database entry into all the places in the database, where the information about that new entry needs to be stored. In a properly normalized database, information about a new entry needs to be inserted only one place in the database. In an inadequately normalized database, in the information about a new entry may need to be inserted into more than one place, and human fallibility with some of the needed additional insertions may be missed. For example, in the above student database, if we fire a query to insert a course no, the student has registered. All remaining attribute of that corresponding tuple will be Null.
2. **Deletion anomaly:** In a properly normalized database, information about an old, to-be-gotten-rid-of entry needs to be deleted from only one place in the database. In an inadequately normalized database, information about that old entry may be deleted from more than one place, and human fallibility being what it is, some of the needed additional deletions may be missed. For example, if student 206 dropped out of Z102, we would lose all the information about that student.
3. **An update anomaly:** An update of a database involves modifications that may be additions, deletions, or both. Thus, 'update anomalies' can be of either kinds of anomalies as discussed above. For example, if we change the schedule of course X100 to W, you can easily understand, what kind of anomalies are raised.

All three kinds of anomalies are highly undesirable, since their occurrence constitutes inconsistencies in the database. Properly normalized databases are much less inconsistent than un-normalized database. In attempt to pinpoint the causes of update, insertion, deletion anomalies, researchers have identified three kinds of dependencies:

- Functional dependency
- Multivalued dependency
- Join dependency

Lets first discuss Functional dependency.

6.20.1 Functional Dependencies (FD)

Determinant and Dependent

The terms determinant and dependent can be described as follows:

1. The expression X → Y means, 'if the value of X is known to us, then we can obtain the value of Y' (in a table).
2. In the expression X → Y, X is the **determinant** and Y is the **dependent** attribute.

We can say that the value X **determines** the value of Y and the value Y **depends on** the value of X.

An attribute is functionally dependent, if its value is determined by another attribute *i.e.*, if we know the value of one or many data items, then we can find the value of another.

Functional dependencies can be expressed as:

$$X \rightarrow Y,$$

Where X is the determinant and Y is the functionally dependent attribute. There will be only one value of Y for one value of X.

Lets see various cases with simple examples:

If C → (D, E), then C → D and C → E,

If (C, D) → E, then it is not necessarily true that C → E and D → E,

If C → D and D → C, then C and D are in a 1-1 relationship,

and,

If C → D, then for C, there would be only one value for D.

16.20.1.1 Trivial Functional Dependency

In trivial functional dependencies, the dependent is a subset of the determinant. If the entire attribute in the set on the RHS are included in the set on LHS of the dependency, or if two sides are same, the FD is Trivial. Lets see some examples:

A, B → A

A, B → B

A, B → A, B

6.20.1.2 Armstrong Axioms (Inference Rules)

Let A, B, C, and D be the subsets of attributes of relation R. Then, the following axioms hold**:**

Reflexivity: If B is a proper subset of A, then A → B. This also implies that A → A always holds.

Augmentation: If A → B then, AZ → BZ holds**.**

Transitivity: If A → B and B → C, then A → C holds.

Additivity: If A → B and A → C, then A → BC hold.

Projectivity: If A → BC, then A → B and A → C holds**.**

Pseudotransitivity: If A → B and CB → D, then AC → D holds**.**

6.20.1.3 Closure of a Set of Functional Dependencies

If F is a set of functional dependencies for a relation R, then the set of all functional dependencies that can be derived from it is called the closure of F. This is also denoted by F^+.

Example: Find the FD's in following relations. Identify the problem and give the solution.

Table 1.

STUDNAME	COURSE	PHONE NUMBER	DEPARTMENT	GRADE
Ram	353	237–4539	HINDI	**A**
Syam	329	427–7390	ENGLISH	**A**
Ram	328	237–4539	HINDI	**B**
Mohan	456	388–5183	MATH	**C**
Sohan	293	371–6259	POLSC.	**B**
Tripti	491	823–7293	HISTORY	**C**
Tripty	353	823–7293	HISTORY	**B**
Ram	491	237–4539	HINDI	**C**
Jyoti	353	842–1729	HINDI	**A+**
Basant	**379**	**839–0827**	**BIO**	**B**

In the above relation, we have observed the following FDs:

STUDNAME→PHONENUMBER

STUDNAME→DEPARTMENT

STUDNAME, COURSE→GRADE

In the above relation, unless the student takes at least one course, we cannot enter the data for the student. If we change the phonenumber or department, it leads to inconsistencies in the database. To rectify these two problems, we have decomposed the above relation as follows:

Table 1–A

STUDNAME	PHONENUMBER	DEPARTMENT
Ram	237–4539	**HINDI**
Syam	427–7390	**ENGLISH**
Mohan	388–5183	**MATH**
Sohan	371–6259	**POLSC.**
Tripti	823–7293	**HISTORY**
Jyoti	842–1729	**HINDI**
Basant	**839–0827**	**BIO**

Table 1–B

STUDNAME	COURSE	GRADE
Ram	353	**A**
Syam	329	**A**
Ram	328	**B**
Mohan	456	**C**
Sohan	293	**B**
Tripti	491	**C**
Tripti	353	**B**
Ram	491	**C**
Jyoti	353	**A+**
Basant	**379**	**B**

Example: Two sets of FD's for a relation R (T, U, V, X, Y). Are they equivalent?

1. $T \rightarrow U$, $TU \rightarrow V$, $X \rightarrow TV$, $X \rightarrow Y$

2. $T \rightarrow UV$, $X \rightarrow TY$

Solution:

1. If $T \rightarrow U$, it means U can be determined by the given value of T, which implies that in FD $TU \rightarrow V$, ONLY by knowing T, we can determine U and also V also.

If $T \rightarrow U$ exists, so $T \rightarrow V$ also exists.

$X \rightarrow TV$ means $X \rightarrow T$ and $X \rightarrow V$. Since $T \rightarrow V$ already exists, so

By applying transitivity rule,

$X \rightarrow T$ and $T \rightarrow V$ implies $X \rightarrow V$, can be dropped, leaving the FD

$X \rightarrow T$.

Now the irreducible set is as follows:

$T \rightarrow U$

$T \rightarrow V$

$X \rightarrow T$

$X \rightarrow Y$

2.

$T \rightarrow UV$ can be reduced by using projectivity rule to $T \rightarrow U$ and $T \rightarrow V$

$X \rightarrow TY$ can be reduced to $X \rightarrow T$ and $X \rightarrow Y$

These set of FD's are equivalent by using projectivity rule to the above derived irreducible set.

Hence, the two given sets are equivalent.

6.20.1.4 Advantages of Normalization

- Removal of redundant data storage.
- Close modeling of real world entities, processes and their relationships.
- Structuring of data to make the logical modeling flexible.

Normalization process can be categorized into the following normal forms based on the level of Normalization:

6.20.2 First Normal Form (1NF)

A table is in first normal form, if the entire attribute contains no repeating groups or for every attribute of table, the value in each cell is atomic.

Assume the ORDER table as the as following example:

ORDER				
Orderno	**Customerno**	**Item1**	**Item2**	**Item3**
444	222	Xyz	pqr	tuv
555	333	Rst		

This structure creates the following problems:

- Order 444 has no room for more than 3 items.
- Order 555 has wasted space for item2 and item3.

In order to create a table which is in first normal form, we must extract the repeating groups and place them in a separate table named ORDER1.

ORDER1	
Orderno	**Customerno**
444	222
555	333

Now item1, item2 and item3 has been removed, so the problem of repeating group has vanished.

ORDER2	
Orderno	**Item**
444	Xyz
444	Pqr
444	Tuv
555	Rst

Each row contains one item for one order, so an order contains any number of items.

In tables, order1 and order 2, the atomicity property is being fulfilled. Now, tables order 1 and order 2 are in the first normal form.

6.20.3 Second Normal Form (2NF)

A table, will be in second normal form (2NF), only if it will be in 1NF and there is full functional dependency, between non key attributes and key attribute. Key attribute will be the whole of the primary key (*i.e*. there will not be any partial dependencies).

As per the definition, anomalies can occur when attributes are functionally dependent only on a, part of a composite key. A table will be in second normal form, when all the non-key attributes are dependent on the whole key attributes. That is, no attribute is dependent only on the part of the key.

Full Functional Dependincy: If the Non key attributes of a relation are functionally dependent on the whole of the key attribute is Full Functional Dependency.

Partial Functional Dependency: If the Non key attributes of a relation are not functionally dependent on the whole of the key attribute is Partial Functional Dependency.

Any table having a key with a single attribute, will be in second normal form.

Lets take a table named order as an example:

order (orderno, custno, address, contact, date of order, ordertotal)

with the following functional dependencies:

Custno → Address, contact

Orderno → Date of order, ordertotal

In the above table, **address** and **contact** are functionally dependent on **custno** but not on **orderno**, therefore they are not dependent on the whole key. The above table will be in 2NF, if custno, address, contact attributes are removed from the table and is placed in another table as given below:

Customer

Custno	Address	contact

Order

Orderno	Date of order	Ordertotal

6.20.4 Third Normal Form (3NF)

A table will be in third normal form (3NF) only if it is in 2NF and every non key attribute is non transitively dependent on the primary key.

Anomalies will occur when a table contains one or more transitive dependencies. A table will be in 3NF, when it will be in 2NF and will not have transitive dependencies.

Transitive Dependencies (TD)

An attribute is transitively dependent if its value is determined by another non key attribute.

A transitive dependency exists when $X \rightarrow Y \rightarrow Z$.

Lets take a table named order as the following example:

order (orderno, custno, address, contact, date of order, ordertotal)

With the following functional dependencies:

Orderno $\rightarrow$ custno, Date of order; custno $\rightarrow$ address, contact

In the above table, the attributes **address** and **contact** are functionally dependent on **custno** which is not a key. The above table will be in 3NF if custno, address, contact attributes are removed from the table and placed in another table as given below:

Customer

Custno	Address	contact

Order

Orderno	Custno	Date of order	Order total

6.20.5 Boyce-Codd Normal Form

A table will be in Boyce-Codd normal form (BCNF) only if it is be in 3NF and every determinant is a candidate key.

Note: Anomalies will occur in tables in 3NF, if there is a composite key in part of a key that has a determinant which is not itself a candidate key.

1. This can be expressed as R($\underline{X}$, $\underline{Y}$, Z), Z $\rightarrow$ X where:
   - ⇨ The relation contains attributes X, Y and Z.
   - ⇨ X and Y form a candidate key.
   - ⇨ Z is the determinant for X
   - ⇨ Z is not a part of any key.

2. Anomalies will occur when a table contains multiple candidate keys where:
   - ⇨ The keys contain more than one attribute (they are composite keys).
   - ⇨ An attribute is common to more than one key.

Lets take a table named timetable as following example:

timetable(campusname, coursename, classno, classtime, building)

Take the following sample data:

Campusname	coursename	Classno	classtime	building
NORTH	BCA	1	12:00-2:00	viveka
NORTH	BCA	2	10:00-11:00	shraddha
SOUTH	BCA	3	12:00-2:00	astha

Note, that no two buildings have the same name, thus

building → compusname

As the determinant is not a candidate key, this table is NOT in Boyce-Codd normal form.

This table should be broken into the following tables:

R1(coursename, classno, building, classtime)

R2(building, campusname)

6.20.6 Multivalued Dependencies

Let R be a relation having attributes or a set of attributes A, B and C. There is a multivalued dependency of attribute B on Attribute A only if the set of B values associated with a given A value is independent of C values.

If A ↠ B in R (A, B, C), then A ↠ C holds as well.

Lets understand in other following ways:

The **multivalued dependency**, A ↠B, exists when two tuples exist having the same A values:

T1 (a, b1, c1) and T2 (a, b2, c2),

implies the two tuples,

T4(a, b2, c1) and T3(a, b1, c2) also exist

The MVD A ↠ B is also written as A ↠ B | C

4NF

A relation is in 4 NF only if, it is in BCNF and there are no multivalued dependencies.

Example:

Lets consider a table of departments, their jobs and the parts they stock. The MVD's in the table will be:

Department ↠ Job

Department ↠ Part

Assume that department d1 works on jobs j1, and j2 with parts p1 and p2; that department d2 works on jobs j3, j4, and j5 with parts p2 and p4; and that department d3 works on job j2 only with parts p5 and p6. The table would look like this:

Department	Job	Part
d1	j1	p1
d1	j1	p2
d1	j2	p1
d1	j2	p2
d2	j3	p2
d2	j3	p4
d2	j4	p2
d2	j4	p4
d2	j5	p2
d2	j5	p4
d3	j2	p5
d3	j2	p6

If you want to add a part to a department, you must create more than one new row. Likewise, to remove a part or a job from a row, it can destroy the information. Updating a part or job name will also require multiple rows to be changed.

Solution:

Split this table into two tables, one with **(department, job)** and another with **(department, parts)**.

The definition of 4NF is that we have no more than one MVD in a table. If a table is in 4NF, it is also in BCNF.

6.20.7 Lossless Joins

A set of relations satisfy the lossless join property, if the instances can be joined without creating invalid data. A join that is not lossless will contain extra, invalid rows. A join that is lossless will not contain extra, invalid rows. We have relation R and we break it into two relations *i.e*. R1 and R2. After taking Natural Join of R1 and R2, it will give all the rows same as original (in R). This process is known as lossless join.

If we get an extra row (spurious tuple), then the process is not lossless. This is known as **Join Dependency**.

5NF

"A relation is in fifth normal form (5NF), if every join dependency is implied by the candidate keys."

Design relational schema so that they can be joined with equality conditions on attributes that are either Primary keys or Foreign keys, in a way it guarantees that no spurious tuples are generated.

Do not have relations that contain matching attributes other than foreign-primary key combination.

Example: The relation employee is given below:

Ename	Role	Projectname
Om	Designer	Tehri
Om	Programmer	Pos
Om	Designer	Pos
Ananya	Designer	Pos

Now break employee relation into two sub relations R1 and R2 as follows:

R1

Ename	role
Om	designer
Om	programmer
Ananya	designer

R2

Role	projname
Designer	tehri
Programmer	pos
Designer	pos

Now take natural join of R1 and R2

Ename	role	Projname
Om	designer	Tehri
Om	designer	Pos
Om	programmer	Pos
Ananya	designer	Tehri
Ananya	designer	Pos

Now we have five rows, while originally we had only 4 rows. So, we are getting one extra row (row number 4) and that is spurious tuple.

6.21 STRUCTURE OF ORACLE

The structure of ORACLE RDBMS is organized in three layers.

KERNEL
Data Dictionary
SQL

6.21.1 Kernel

The kernel provides communication with the database and connection with the other kernel in a distributed database environment. Kernel also verifies the integrity constraints, control of concurrent process, and optimization of SQL queries, maintaining indexes, clusters and the management of physical data storage in the form of OS (Operating System) files.

6.21.2 Data Dictionary

The Data Dictionary is collection of data element definitions. More advanced data dictionary contains database schema with reference keys, more advanced Data Dictionary contains entity-relationship model of the data elements or objects. It describes the database structure, constraints, applications, authorizations and so on. These are often used as an integral tool for information resource management.

Elements of Data Dictionary

Description of the schemas of the database system is as follows:

Detailed information on physical database design, such as storage structures, access paths, file and record sizes.

1. Database user's description, their responsibilities and their access rights.
2. High-level description of the database transactions, applications and relationships of the users to transactions.
3. Information about the different type of associations between the database transactions and the data items referenced by them. Data dictionary is useful in determining which transactions are affected whenever certain data definitions are changed.
4. Usage statistics such as frequencies of queries, transitions and access counts to different portions of the database.

The above given Meta data in the data dictionary is available to DBA's, designers and authorized users as on-line system documentation. In case of Oracle database, Data Dictionary provides the details on the database objects

(such as tables, columns, views, etc.), the oracle users and the privileges and rights of the users over different objects. Each operation that affects the structure of the database automatically causes the dictionary to be updated. The dictionary is made up of tables and views. They are created at the time of Oracle installation and are manipulated through SQL. These tables are not accessible by the users. The view derived from the tables may be consulted by the oracle users.

Select * from dict;

The above query will show all the contents of a dictionary.

6.21.3 SQL

SQL will be discussed in detail in the subsequent chapters.

6.22 RELATIONAL ALGEBRA

Let's first define Relation:

Formally, given domains D_1, D_2, …. Dn a **relation** r is a subset of

$$D_1 \times D_2 \times \ldots \times D_n$$

Thus, a relation is a set of n-tuples $(a_1, a_2, \ldots, a_n)$ where each $a_i \in D_i$

Schema of a relation consists of,

attribute definitions

- name
- type/domain

integrity constraints

In addition to define the database structure and constraints, a data model must include a set of operations to manipulate the data. A basic set of relational model operations constitute the **Relational Algebra.** These operations enable the user to specify basic retrieval requests. *Relational algebra* is a procedural query language, which consists a set of operations that take one or two relations as input and produce a new relation as their result.

Actually Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus are Pure languages. Pure languages form underlying basis of query languages (*e.g.* SQL) that people use.

Now lets compare Relational Algebra with SQL:

Relational Algebra:

- is closed (the result of every expression is a relation)
- has a rigorous foundation

- has simple semantics
- is used for reasoning, query optimization, etc.

SQL:

- is a superset of relational algebra
- has convenient formatting features, etc.
- has complicated semantics
- is an end-user language.

The following are the set of operators in Relational Algebra:

Basic Operators

select: σ

project: Π

union: $\cup$

set difference: $-$

Cartesian product: x

Additional Operators

Set intersection

Natural join

Aggregation

Outer Join

Division

Aggregate Operators

Rename Operators ρ

Fundamental Operators: σ, Π, $\times$, $\cup$, -, ρ

A set of fundamental operators is known as the complete set. Any operator which can be derived from the complete set will be a derivable operator.

Derivable operators: $\bowtie$, $\cup$

For Example**:**

Let $A \bowtie_c B \models \pi_{a_1,a_2,...a_N} (\sigma_c(A \times B))$

where c is the join condition (*e.g.* $A.a_1 = B.a_1$), and $a_1, a_2,...a_N$ are all the attributes of A and B without repetition.

Lets define a Relation named "account" as follows:

branch-name	account-number	balance
D	100	5000
M	200	7000
P	101	4000
R	302	3500
B	202	9000
Re	224	70000
Br	218	75000

***select* operator:** Is a unary operation, which means it operates on one relation. Its function is to select tuples that satisfy a given predicate. To denote selection, the lowercase Greek letter sigma (σ) is used. The predicate appears as a subscript to σ.

Notation: $\sigma\, p(r)$

p is called the **selection predicate**

Defined as:

$$\sigma\, p(\mathbf{r}) = \{t \mid t \in r \textbf{ and } p(t)\}$$

where p is a formula in propositional calculus consisting of **terms** connected by:

$\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)

Each **term** is one of:

<attribute> *op* <attribute> or <constant>

where *op* is one of the relational operators: =, ≠, >, ≥. <. ≤

Example of selection:

$$\sigma_{\text{branch–name = "P"}} \text{ (account)}$$

The argument relation is given in the parentheses following the σ.

Above example, select the tuples of the account relation where the branch is "P".

The results of the query are as following:

branch-name	account-number	balance
P	101	4000

Comparisons like (=, ≠, <, >, ≥, ≤) can also be used in the selection predicate. For example, query using a comparison is to find all tuples in which the balance amount is more than ₹ 70,000 would be written as:

$$\sigma_{\text{balance > 70000}} \text{ (account)}$$

Rename Operator

The rename operator returns an existing relation under a new name. $\rho_A(B)$ is the relation B with its name changed to A. For example,

$$\rho_{emp2}(\text{employee})$$

Project operator: The *project* operation is a unary operation that returns its argument relation with certain attributes left. Since a relation is a set, any duplicate rows are eliminated. Projection is denoted by the Greek letter pi (π). The attributes that wish to appear in the result are listed as a subscript to π. The argument relation follows in parentheses.

Notation:

$$\Pi_{A_1, A_2, ..., A_k}(r)$$

where A_1, A_2 ... are the attribute names and r is a relation name.

The result is defined as the relation of k columns obtained by erasing the columns that are not listed.

Duplicate rows are removed from the result, since relations are sets.

Example: To select the *branch_name* attribute of *account*

$$\Pi_{\text{brench\_name}}(account)$$

Now lets take another set of Relations:

EMPLOYEE

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SYAM	40000	SALES	EXECUTIVE
SITA	30000	CHEMISTRY	VICE PRINCIPAL

EMP

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SYAM	40000	SALES	EXECUTIVE
RAVI	30000	LIBRARY	PROGRAMMER
GIDU	20000	PURCHASE	OFFICE ATTENDENT
MOHAN	200000	PHYSICS	PRINCIPAL

The *union* operator yields the results that appear in either or both of the two relations. It is a binary operation denoted by the symbol.

Notation: $r \cup s$

Defined as:

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

For $r \cup s$ to be valid.

r, s must have the *same* **arity** (same number of attributes).

The attribute domains must be **compatible** (example: 2nd column of r deals with the same type of values, as does the 2nd column of s)

Example: to find all employees with either an employee or a emp.

$\Pi_{ename, sal, dname, job}$ (employee) $\cup$ $\Pi_{ename, sal, dname, job}$ (emp)

EMPLOYEE UNION EMP:

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SYAM	40000	SALES	EXECUTIVE
SITA	30000	CHEMISTRY	VICE PRINCIPAL
RAVI	30000	LIBRARY	PROGRAMMER
GIDU	20000	PURCHASE	OFFICE ATTENDENT
MOHAN	200000	PHYSICS	PRINCIPAL

The ***set-difference*** **operator**: denoted by the –, results in finding tuples that are in one relation but not in another. The expression $r - s$ results in a relation containing those tuples in r but not in s.

Notation: $r - s$

Defined as:

$$r - s = \{t \mid t \in r \text{ and } t \notin s\}$$

Set differences must be taken between **compatible** relations.

r and s must have the same arity

attribute domains of r and s must be compatible

For example, the query to find all the employees who are in employee relation but not in relation emp, is written as:

$\Pi_{ename, sal, dname, job}$ (employee) – $\Pi_{ename, sal, dname, job}$ (emp)

EMPLOYEE MINUS EMP:

ENAME	SAL	DNAME	JOB
SITA	30000	CHEMISTRY	VICE PRINCIPAL

For a set difference to be valid, it must be taken between compatible relations just as in the union operation.

Set Intersection operator: It is denoted by the symbol $\cap$. It is not a fundamental operation, however it is a more convenient way to write $r - (r - s)$.

Notation: $(r \cap s)$

Defined as:

$$r \cap s = \{t \mid t \in r \text{ and } t \in s\}$$

For *r s* to be valid:

r, s must have the *same* **arity** (same number of attributes).

The attribute r and s must be **compatible.**

For example, a query of the operation to find all employees who have common entities can be written as:

$\Pi_{ename\ ,sal\ ,dname\ ,job}$ (employee) $\cap$ $\Pi_{ename\ ,sal\ ,dname\ ,job}$ (emp)

EMPLOYEE INTERSECTION EMP:

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SYAM	40000	SALES	EXECUTIVE

Union, intersection and difference operations always take two input relations, which must be **union-compatible**:

- Same number of fields.
- 'Corresponding' fields have the same type.

Cartesian product operator:

Notation: $r \times s$

Defined as:

$$r \times s = \{t\, q \mid t \in r \text{ and } q \in s\}$$

Assume that attributes of *r*(R) and *s*(S) are disjoint. (That is, $R \cap S = \varnothing$).

If attributes of *r* and *s* are not disjoint, then the renaming must be used.

A	*B*
α	1
β	2

r

C	*D*	*E*
α	10	a
β	10	a
β	20	b
γ	10	b

s

A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

(Result of $r \times s$)

Natural Join Operator

Let *r* and *s* be relations on schemas *R* and *S* respectively.

Then, *r* Join *s* is a relation on schema $R \cup S$ obtained as follows:

Consider each pair of tuples *tr* from *r* and *ts* from *s*.

If *tr* and *ts* have the same value on each of the attributes in $R \cap S$, add a tuple t to the result, where,

- *t* has the same value as *tr* on *r*
- *t* has the same value as *ts* on *s*

Example:

$$R = (A, B, C, D)$$

$$S = (E, B, D)$$

Result schema = (*A, B, C, D, E*)

r join *s* is defined as:

$$\Pi_{r.A,\ r.B,\ r.C,\ r.D,\ s.E}\ (\sigma_{r.B = s.B \wedge r.D = s.D}\ (r \times s))$$

Relations r, s:

A	*B*	*C*	*D*
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	*C*	*D*
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ε

s

A	*B*	*C*	*D*	*E*
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

$r \bowtie s$

Aggregate Operators:

Aggregation function takes a collection of values and returns a single value as a result.

avg: average value

min: minimum value

max: maximum value

sum: sum of values

count: number of values

Aggregate Operation in relational algebra,

$$G_1, G_2, ..., G_n \; \mathcal{J} F_1(A_1), F_2(A_2), ..., F_n(A_n) \; (E)$$

E is any relational-algebra expression

$G_1, G_2, \ldots, G_n$ is a list of attributes on which it is to be grouped (can be empty)

Each F_i is an aggregate function

Each A_i is an attribute name

Examples:

Retrieve the total salary of employees:

$$\mathcal{F}_{\text{sum(salary)}}(\text{employee})$$

Retrieve no. of employees who are getting salary:

$$\mathcal{F}_{\text{count(salary)}}(\text{employee})$$

Find total salary of employees department wise:

$$_{\text{deptnum}}\mathcal{F}_{\text{sum(salary)}}(\text{employee})$$

Outer Join Operators:

Outer Join is an extension of the join operation that avoids loss of information. You can retrieve the data from more than one relation and one attribute is common in both the relation. Now if you perform join on both the relations and suppose one relation is deficient, in such a case some of the records will not be printed. If there are two relations employee (ename, eno, sal, deptno) and Department(deptno, dname, loc) and one common attribute deptno. Suppose in a department relation, there is department number 50 and in employee relation there is no such department, now if we perform join on both the relations, then the row of department number 50 will not printed. To print that row, we use the concept of Outer Join. In this case, the deficient relation is employee which is the child relation. In this case, we use left outer join.

Computes the join and then adds tuples form one relation that does not match tuples in the other relation to the result of the join.

Lets take an example of two relations (Loan and Borrower) as follows:

Loan:

loan_number	branch_name	amount
L-17	Deo	30000
L-23	Rao	40000
L-26	Pia	17000

Borrower:

customer_name	loan_number
Jai	L-17
Sai	L-23
Harry	L-156

Right Outer Join:

loan ⟖ *borrower*

loan_number	customer_name	amount	branch_name
L-17	Deo	30000	Jai
L-23	Rao	40000	Sai
L-156	*null*	*null*	Harry

If the Right hand side (R.H.S.) (master relation) is deficient, then we apply Right Outer Join.

Full Outer Join:

Loan ⟗ *borrower*

loan_number	branch_name	amount	customer_name
L-17	Deo	30000	Jai
L-23	Rao	40000	Sai
L-26	Pia	17000	*null*
L-155	*null*	*null*	Harry

In the same manner, If Left hand side relation is deficient, then we can apply.

Left Outer Join ⟕

Uses of null values:

- *null* signifies that the value is unknown or does not exist.
- All comparisons involving *null* are false by definition.
- It is possible for tuples to have a null value, denoted by *null*, for some of their attributes.
- *null* signifies an unknown value or a value that does not exist.
- The result of any arithmatic expression involving *null* is *null.*
- Aggregate functions simply ignore null values (as in SQL).
- For duplicate elimination and grouping, null is treated like any other value, and two nulls are assumed to be the same (as in SQL).

Recursive Closure: A table can be joined itself on column that contains the same type of information. A self join joins rows from a table to another, or the same rows in that table**.**

If the employee table is joined itself using the columns employeenumber and mgrnumber, then both the Employeenumber and mgrnumber columns contain the same type of information. For each employee, (for each row in the table) mgrnumber is the employee number of the employee's manager and the Employeenumber is the employee number of the employee named in that row.

EMP $\left(\bowtie\right)$ MANAGER

MGRNO=EMPNO

Note: If you omit the join clause, a Cartesian join is performed.

Division Operator:

Precondition: In A/B, the attributes in B must be included in the schema for A. Also, the result has attributes A-B.

Relation A

Snum	Pnum
S1	P1
S1	P2
S1	P3
S1	P4
S2	P1
S2	P2
S3	P2
S4	P2
S4	P4

Relation B

Pnum
P2

Relation A/B

Sno
S1
S2
S3
S4

Solved Examples:

Lets consider the following relations:

Stud(ssn, name, address, major)

Cour(code, title)

Reg(ssn, code)

From the above given relations semantics, it is clear that the students are registering themselves in some course.

Example 1: List the codes of courses in which at least one student is registered (registered courses):

$$\Pi_{code}(\text{Reg})$$

Example 2: List the titles of registered courses (of those in 1.)

$$\Pi_{code}(\text{Cour} \bowtie \text{Reg})$$

Example 3: List the codes of courses for which no student is registered.

$$\Pi_{code}(\text{Cour}) - \Pi_{code}(\text{Reg})$$

Example 4: The titles of courses for which no student is registered.

Note: In the previous query, we found the codes; natural join with course to find the titles.

$$\Pi_{name}((\Pi_{code}(\text{Cour}) - \Pi_{code}(\text{Reg})) \bowtie \text{Cour})$$

Example 5: Names of students and the titles of the courses they registered to.

$$\Pi_{name,title}(\text{Stud} \bowtie \text{Reg} \bowtie \text{Cour})$$

Example 6: SSNs of students who are registered for 'Database Systems' or 'Algorithms'.

$$\Pi_{ssn}(Stud \bowtie Reg \bowtie (\sigma_{title='Database\ Systems'}, Cour)) \cup$$
$$\Pi_{ssn}(Stud \bowtie Reg \bowtie (\sigma_{title='Algorithms'}, Cour))$$

Example 7: SSNs of students who are registered for both 'Database Systems' and 'Algorithms'.

$$\Pi_{ssn}(\ Stud \bowtie Reg\ (\sigma_{title='Database\ Systems'}, Cour))\ )" \cap$$
$$\Pi_{ssn}(\ Stud \bowtie Reg\ (\sigma_{title='Algorithms'}, Cour))$$

The name of those students:

$$A = \Pi_{ssn}(Stud \bowtie Reg \bowtie (\sigma_{title='Database\ Systems'}, Cour))\ )" \cap$$
$$\Pi_{ssn}(\ Stud \bowtie Reg \bowtie (\sigma_{title='\ Algorithms'}, Cour))$$
$$\Pi_{name}(\ A \bowtie Stud)$$

Note: used A = instead of ρ( ) function.

Example 8: List of courses in which all the students are registered.

$$\Pi_{code,\ ssn}(\ Reg\ ) / \Pi_{ssn}(Stud)$$

Example 9: List of courses in which all 'FCMPS' major students are registered.

$$\Pi_{code,\ ssn}(\ Reg) / \Pi_{ssn}(\sigma_{major='FCMPS'}, Stud)$$

6.23 RELATIONAL CALCULUS

As we have already stated that Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus are pure languages based on Mathematical concepts. TRC and DRC use the concept of "LOGIC" while Relational Algebra uses the concept of "Set Theory". Pure languages form underlying basis of query languages (*e.g.* SQL) that people use. Relational Calculus provides two flavors as stated below:

6.23.1 Tuple Relational Calculus

A general expression of the Tuple Relational Calculus is denoted by,

$$\{x \mid EXP(x)\ \}$$

where, x is a tuple variable which ranges over the whole tuple.

A general expression of TRC is denoted as follows:

$$\{x_1.a_j,\ x_2.a_k \ldots . \mid EXP(x_1,\ x_2\ \ldots)$$

where, EXP is a formula. A formula is made up of one or more atoms connected via logical operators AND, OR and NOT and is defined recursively by rules (*a*) and (*b*) as follows:

(*a*) Every atom is a formula.

(*b*) If X_1 and X_2 are formulas, then so are

X_1 AND X_2, X_1 OR X_2 and NOT(X_1).

In LOGIC, AND is known as conjunction, OR is known as Disjunction and NOT is known as complement. Also denoted by $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**).

Example 1: Select the employees whose salary is greater than 10000.

{x | employee(x) and salary>10000}

Example 2: Select First name, last name of employees having salary greater than 6000.

{x.fname, x.lname | employee(x) and x.salary>6000}

Example 3: Select birthdate and address of employee R.P. Bhardwaj.

{x.bdate, x.address | employee(x) and x.fname='R' and
x.minit='P.' and x.lname="Bhardwaj"}

6.23.2 Domain Relational Calculus

DRC is denoted as:

$$\{x_1,x_2,\ldots x_n \mid EXP(x_1,x_2,\ldots.)\}$$

where, x_1, x_2… are Domain variables; which individually range over their corresponding attributes.

Example: Retrieve the name, Joiningdate of employees whose salary is 15000.

$$\{q\ s \mid \in p \in q \in r \in s\ (employee(pqrs) \text{ and } r=15000)\}$$

where, $\in$ is Existential quantifier.

Safe Expression:

A safe Expression in a Relational Calculus is one that guarantees to yield a finite no. of tuples as its result; Otherwise unsafe.

$$\{x \mid NOT(employee(x))\}$$

The above expression will return all the rows except employee relation which is infinite. So it is unsafe.

Relationally Complete Language

Any relational language which is as powerful as Relational Algebra and Relational Calculus is called Relationally Complete. A relationally complete language can perform all basic, meaningful operations on relations. Since, SQL is a superset of Relational Algebra and Relational Calculus, it is also Relationally Complete.

6.24 DISTRIBUTED DATABASES

"A collection of multiple logically interrelated databases, physically distributed over a Computer Network". [13]

Where logically interrelated databases means applications view data as an integrated database, independently of data physical placement and there is no logical distinction between a distributed and centralized database system. Physically distributed means data is kept in different nodes of a network (LAN or WAN). (See Figure 6.22)

In a distributed database, data is stored on computers cited on the remote locations. Indexes will be maintained by a number of centralized computers to search the data stored on different computers cited on the remote locations. Using the address of data, computers can communicate with each other and find the required information. Distributed databases do not concern only the addresses on which the data is stored but also on the audit trail to know who has done the updating or retrieval. The purpose of auditing the data is to pinpoint errors in the system and understand where there is a security breach (confidentiality of the system breaks down). When a computer requests data from another computer, an audit trail is created by storing who sent the data, where and when? When this computer passes the data to another computer, the information needs to be updated in the original computer. Since, the number of computers receiving the data increases, the task of auditing becomes cumbersome (complex) and more difficult. A very basic fact about Distributed Database is that a DDBS should appear as non-DDBS.

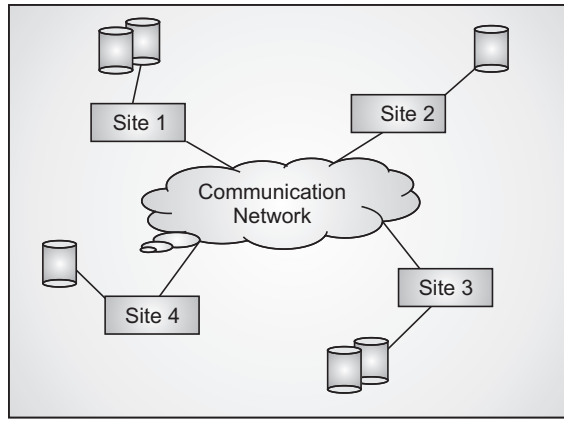


Figure 6.22

Lets see the example of a distributed database:

Example of a distributed database is the World Wide Web pages. The data of these pages are stored on a different computer cited on the remote locations, and such computers are known as Web servers. The address of each file is the location address you enter, when you want to see the page. This location address is an index to the Web pages. Centralized computers keep the domains. Subsequent detailed addresses are kept in the Web servers. There are two steps for searching the data from a distributed database. First, a crawler program must index the content of the databases, then another search engine program would search the index for your request. Whenever a match is found, then the index is used to find the address of the information. This address is provided to the search engine that assembles a list for you. When you click on the items identified by the search engine, you use the address to retrieve the data items.

Limitations of Distributed Databases:

- Distributed databases exchange files. In case of corrupted files or viruses, it may affect the entire system. Security of these databases is difficult to maintain. There may be a global database administrator responsible for the whole system as well as a local database administrator of each location.
- In distributed databases the type of data to be exchanged, the process of addressing the data and the protocol for updating the data must be agreed upon ahead of time and plans must be in place for updating the process.
- When different groups and systems are involved in maintaining the data, there is more opportunity for differences in quality of data to emerge. A distributed database needs to have procedures for determining the quality (accuracy, reliability, etc.) of the data.
- Disadvantage of data replication is the increased overhead on update.

Advantages of Distributed Database:

- In centralized databases, lack of backup or inadequate backup may result in complete loss of data, while in a distributed data systems the data loss is limited to the nodes affected.
- Distributed databases are more flexible and allow different units to update and maintain their own data.
- Each data item is replicated at multiple locations thats why failure of some locations will not result in total system failure. Two advantages, one is higher availability of the system and other is increased parallelism.
- Person at one location may be able to access data stored at other location.
- Provides replication independence.
- Provides hardware, operating system, network, DBMS and location independence.
  - ⇨ Continuous operation.
  - ⇨ No reliance on central site.

Distributed Database Management System (DDBMS)

"A software system that permits the management of DDBS and makes the distribution transparent to the users". [13]

SUMMARY

This chapter deals with the basics of database concepts. In this chapter, there is an explanation about evolution of database, Data models, three level architecture of DBMS, File Organizations and access methods, ER-Diagrams, Normalization and Relational Algebra which is the basis of relational database, etc. Topics are represented in simpler manner with the appropriate examples.

PROBLEMS

1. Define the term Database. Why do we need centralized database?
2. Define DBMS. Give examples of some DBMSs.
3. What do you mean by Integrity? State two Integrity Rules.
4. What is Data Independence? State the difference between Logical data Independence and Physical Data Independence.
5. What do you mean by data models? What are the differences between Hierarchical Data Model, Network Data Model and Relational Data Model?
6. State the three level architecture of the DBMS.
7. What do you mean by associations? What are the different types of associations between the relations?
8. Describe the concept of database languages.

9. Define the terms cardinality and degree of table and associations.

10. Define First Normal Form, Second Normal Form, Third Normal Form and BCNF with examples.

11. The company M-series has decided to store information about musicians who perform on their albums in a database. The company has wisely chosen to hire you as a database designer:

(*i*) Each instrument used in songs recorded at M-Series has a name (*e.g*. guitar, synthesizer, and flute) and a musical key (*e.g*. C, B-flat, and E-flat).

(*ii*) Each musician that records at M-Series has a SSN, a name, an address, and a phone number. Poorly paid musicians often share the same address, and no address has more than one phone.

(*iii*) Each album recorded at M-Series has a title, a copyright date, a format and an album identifier.

(*iv*) Each song recorded at M-Series has a title and an author.

(*v*) Each musician may play several instruments and a given instrument may be played by several musicians.

(*vi*) Each album has a number of songs on it, but no song may appear in more than one album.

(*vii*) Each song is performed by one or more musicians and a musician may perform a number of songs.

(*viii*) Each album has exactly one musician who acts as the producer. A musician may produce several albums.

Design an ER diagram for M-Series

12. The aim of the Mail Monitoring System is to register and monitor efficiently and effectively, all the incoming mails from various units at Air Force record office. One of the main objectives of Air Force Record Office, is to maintain information about the personnel of the IAF, which is essential to plan the promotions, postings and various courses for its personnel, and to assist the planners to take timely and right decisions. The accurate decisions often depend on various types of communications such as service letters, service notes, signals, fax, e-mails etc., from various units as well as higher organizations. Now in AFRO, all incoming and outgoing mails are dealt manually, which causes a lot of difficulty in tracing out a particular mail, which may be of high importance. The very purpose of Mail Monitoring System is to keep a track of all such incoming mails to this office. It further monitors their movements to various wings and subsections, and helps locate the exact position of a particular mail. This system records the details of the person handing/taking over the mail along with the date and time as well as the period of stay in particular section. This system also maintains the database of the personnel who are authorized to collect the mail.

Total Information Required for Monitoring Mail Movement

1. Full details of the mail.

2. Full particulars of personnel who are authorized to collect the mail.

3. Details of Wings/Sections, which deal with the mail.
4. Date of movement of a mail from one section to another.

Sources of Input

1. Mail.
2. Authorization letter from sections.

Design an ER diagram for Mail monitoring System

13. How are fixed-length records different from variable-length records?

14. What do you mean by a file organization? Give examples of some file organizations.

15. (*a*) What is data independence? Discuss logical and physical data independence.
(*b*) What are the advantages and disadvantages of using a distributed database?
(*c*) Explain one-to-many relationship giving suitable example.
(*d*) Consider the following relation for published books:
Book (Book_title, Authorname, Book_type, Listprice, Author_affil, Publisher)
Author_affil refers to the affiliation of author.
Suppose the following dependencies exist:
Book_title → Publisher, Book_type
Book_type → Listprice
Authorname → Author_affil
(*i*) The relation is in which normal form. Explain.
(*ii*) Normalize the relation upto 3NF. Explain each step.
(*e*) Consider the following tables:
Emp (Emp_Id, Dept_Id, Emp_Name, Salary)
Dept (Dept_Id, Dept_Name, Supervisor)
perform the following queries using relational algebra:
(*i*) Get names of all the employees working in the "Accounts" Deptt.
(*ii*) Increase the salary of all the employees by 10% whose Deptt. Is "Library".
(*iii*) Delete all the employees whose salary is less than 2000.
(*iv*) Find the sum of salaries for each Deptt.

16. Attempt the following:
With respect to an educational institution, the following observation has been made:
1. It is engaged in imparting education to students in several courses.
2. It has several teaching departments according to various subjects.
3. It has core facilities belonging to one of its teaching departments.
4. Each course consists of several papers. Each paper belongs to a particular teaching department.
5. Each paper is characterized by its name, maximum marks, number of lecturers per week and the department to which it belongs.

You are required to analyze and conceptualize above organizational reality, using ER Model and depict the same as ER diagram.

17. Attempt the following pairs:

(*a*) What are the main types of database end users? Discuss the main activities of each.

(*b*) What are the limitations of File Oriented approach?

18. (*a*) List and explain three characteristics of a DBMS.

(*b*) Describe hierarchical model with a suitable example.

(*c*) Who is a DBA? What are his responsibilities?

(*d*) State referential integrity rule with a suitable example.

(*e*) Define the terms:

(*i*) Domain,

(*ii*) Candidate key.

22. What do you understand by Normalization? Normalize the following relation into first and second normal forms:

Bills (Bill No., Date, Item No., Item Description, Unit Price, Qty Bought, Total Cost)

Assume that each bill contains a list of multiple items. Each item has a unique Item No., and is identified by Item-description and a unit price.

23. Consider the following relational database:

EMPLOYEE (emp_name, street, city)

WORKS (emp_name, comp_name, salary)

COMPANY (comp_name, city)

MANAGES (emp_name, manager_name)

For each of the following queries, give the Relational Algebra expression:

(*i*) Find the names of all the employees who work for XYZ Bank Corporation.

(*ii*) Find the names of all the employees who live in the same city, where the company for which they work is located.

(*iii*) Find the names of those employees who earn more than every employee of the XYZ Bank Corporation.

(*iv*) Find the name, street and city of those employees who work for the XYZ Bank Corporation and earn more than Rs. 2,50,000 per annum.

(*v*) Find the names of managers who work in a bank located in Delhi.

REFERENCE AND RELATED LINKS

[1] Martin, J. and McClure, C., "Diagramming Techniques for Analysts and Programmers" (publ: Prentice Hall, 1985)

[2] Howe, D. R., "Data Analysis for Database Design" (publ: Edward Arnold 1983)

[3] Yao, S. B. (editor) "Principles of Database Design Volume 1" (publ: Prentice Hall, 1985).

[4] Chen P., "The Entity-Relationship Model-Towards a Unified view of Data" ACM Trans on Database systems vol. 1, no. 1 Mar 1976.

[5] Date, C. J., "An Introduction to Database Systems Vol. 1 Fourth edition" (publ: Addison Wesley 1986).

[6] Kent, W., "A Simple Guide to Five Normal Forms in Relational Database Theory" Communications of ACM Vol. 26, No. 2, Feb. 1983.

[7] Smith, H., "Database Design: Composing fully normalized tables from a rigorous dependency diagram," Communications of the ACM, Vol. 28, No. 8, Aug. 1985.

[8] Codd, E. F. Is Your Database Really Relational? "Computerworld," (Oct. 14, 1985).

[9] Date, C. J., A Practical Approach to Database Design,Addison-Wesley, 1986.

[10] Davis, C. (editor), "Entity-Relationship Approach to Software Engineering" (publ: North Holland, 1983)

[11] Kent, W., "Data and Reality", (publ: North Holland, 1978).

[12] Ross, R. G., "Entity Modeling: Techniques and Application," Database Research Group, 1987.

[13] Ozsu, M. T., Valdaries, P., "Principles of Distributed System", 2nd edition, Prentice Hall, Eaglewood diffs, N. J., 1991.

[14] Rajan Chattamvelli, "Data Mining Methods", Narosa Publishing House.

[15] Ramez Elmasri, S. B. Navathe, "Fundamentals of Database Systems", Pearson.

[16] http://www.tonymarston.net/

[17] Jiawei Han, Micheline Kamber and Jain Pei 'Data Mining Concepts and Techniques, Morgan Kaufmann Publishers.

Structured Query Language

7

CHAPTER

SQL was originally invented by IBM researchers in the 1970's.

Date of Birth: 1944, California, USA

Don Chamberlin, invented SQL together with Ray Boyce and Jim Melton

Don Chamberlin

Introduction

For the database programming, we use SQL (STRUCTURED QUERY LANGUAGE). Its original version was called SEQUEL and a predecessor of sequel was named SQUARE. SQUARE was the data definition and data manipulation language for system R**.** According to ANSI (American National Standards Institute), it is the standard language for Relational Database Management Systems. Dr. E. F. Codd published the paper, "A Relational Model of Data for Large Shared Data Banks", in June 1970 in the Association of Computing Machinery (ACM) journal, *Communications of the ACM*. Codd's model is now accepted as the definitive model for Relational Database Management Systems (RDBMS). The language, Structured English Query Language (SEQUEL) was developed by IBM Corporation, Inc., to use Codd's model. SEQUEL later became SQL. In 1979, Relational Software, Inc (now Oracle) introduced the first commercially available implementation of SQL. Today, SQL is accepted as the standard RDBMS language. SQL statements are used to perform tasks such as to the update data on a database or retrieve the data from a database. Some common Relational Database Management Systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, My SQL, etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually used on their system.

- SQL stands for Structured Query Language
- SQL allows to access a database
- SQL is an ANSI standard computer language
- SQL can execute queries
- SQL can retrieve data from a database
- SQL can insert new records in a database
- SQL can delete records from a database
- SQL can update records in a database
- SQL is easy to learn as fourth generation programming language

Note: Most of the SQL database programs also have their own proprietary extensions, in addition to the SQL standard.

A Relational Database Management System handles the requests generated from the SQL interface, producing or modifying data in response to these requests. This involves a multilevel processing system.

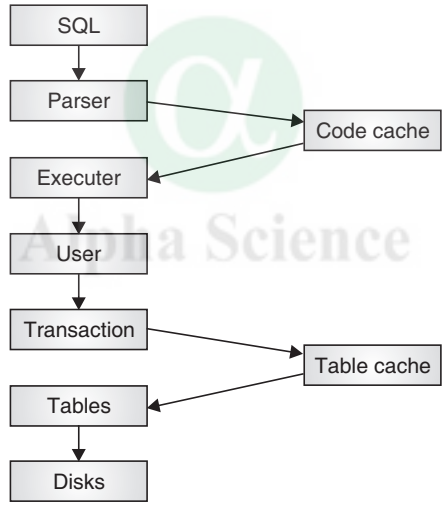


Figure 7.1

Whenever the user submits the SQL query, the processing will be done at the following levels (as shown in Figure 7.1):

- **Parser:** The SQL query is parsed and tokenized by the Parser. Syntax errors are reported back to the user. Since Parsing can be time consuming, therefore, good quality DBMS implementations maintain cache for query after they have been parsed, so that if the same query is submitted again the cached copy can be reused.
- **Executer:** SQL tokens are received by the Executor which translate them into relational algebra. Each relational algebra fragment is optimized, and passed down to the next level for processing.

- **User:** The concept of the user is required at this stage. This gives the query context, and also allows security to be implemented on a per-user basis.
- **Transactions:** The queries are executed in the transaction phase. The same query from the same user can be executed multiple times in different transactions. Each transaction is quite separate.
- **Tables:** The table structure is controlled at a low level. The schema itself is stored in tables.
- **Table cache:** Disks are slow; in spite of the fact that a disk is the best way of storing long-term data. Memory is much faster, so it makes sense to keep as much table information as possible in the memory. The disk remains synchronized to memory as a part of the Transaction Control System.
- **Disks:** Almost all Database Systems have the disk storage system. This provides storage for the DBMS system tables, user information, schema definitions, and the user data itself. It also provides the means for transaction logging.

7.1 SQL LANGUAGE

Data-Definition Language (DDL): DDL provides commands for defining, dropping, and modifying relation schemas. CREATE, DROP and ALTER are the DDL commands.

Data-Manipulation Language (DML): DML includes a query language based on both the Relational Algebra and the Tuple Relational Calculus. It also includes commands to insert tuples into, delete tuples from, and modify tuples into the database. INSERT, DELETE and UPDATE are the DML commands.

Transaction Control Language: SQL includes commands for specifying the beginning and ending of transactions. Transaction is the execution of a set of DMLS in one session. COMMIT, SAVE POINT and ROLLBACK are the TCL commands.

Data Control Language: DCL is used to give authorization to the user and revoke authorization from the user to control the data. GRANT and REVOKE are the DCL commands. These authorization commands are generally used by DBA (Database Administrator).

Embedded SQL: Through Embedded SQL, SQL statements can be embedded within general-purpose programming languages, such as C, C++, Java, COBOL, Pascal, and FORTRAN. For this purpose, we use pre-compilers. PRO*C, PRO*FORTRAN, etc. are the examples of pre-compilers.

A Relational Database system contains one or more tables. The information for the database is stored in these tables. Tables are uniquely identified by their

names and are made of columns and rows. Columns contain the column name, data type and other attributes for the column. Rows contain the records or data for the columns. Here is a sample table named as "Employee".

Employeenumber, employeename, employeejob, salary, departmentnumber, employeecity, joiningdate, mgrnumber are the columns or field names of Employee table. The rows/tuples contain the data for this table:

Employee							
Employee number	Employee city	Depart-ment no.	Joining date	Mgr number	Employee Name	Employee Job	Salary
O11	New Delhi	1	10-jul-1956	023	Shikha	Manager	25000
012	Gaziabad	2	10-aug-1975	011	Ananya	programmer	10000
013	Noida	3	10-aug-1990	012	Neelam	Salesman	5000
014	New Delhi	1	05-sept-1981	012	Arya	Executive	40000
015	Gaziabad	2	06-aug-1945	013	Mohan	Salesman	6000
016	Noida	3	07-nov-1984	012	Raj	Executive	8000

Here is another sample table named as "department".

Department		
Department number	Department city	Department name
1	New Delhi	Human Resource
2	Ghaziabad	Finance
3	Noida	Sales

7.2 INVOKING SQL PLUS

SQL PLUS is an environment through which you can execute your SQL commands. SQL PLUS provides environment for SQL. SQL PLUS can be invoked from a system prompt in two ways:

(*i*) SQLPLUS

(*ii*) SQLPLUS SCOTT/TIGER

OR

>SQL

Enter user name

SQL>

Enter password

SQL>

Commands can be entered at the SQL, prompt either in upper or lower case. By using "/" command, you can execute SQL statements. By using ^D (Control +D) or typing exit at SQL prompt, you go to the operating system prompt. You can also write SQL statements using the notepad editor (or any other editor supported by SQL). Save the file with .sql extension, then on the SQL prompt use the command @filename.sql.

7.3 DATA-DEFINITION LANGUAGE

Create command

The create table statement is used to create a new table.

Syntax:

```
create table "tablename"
("column1" "data type"
[constraint],
"column2" "data type"
[constraint],
"column3" "data type"
[constraint]);
```

[ ] indicates that constraints are optional and you may have as many columns as you like.

Example:

```
create table employee
   (employeenumber number(3),
   employeename varchar(15),
   job varchar(15),
   salary number(16),
   employeecity varchar(10),
   joiningdate date,
   departmentno. number (3), mgrnumber number(3));
```

Example:

```
create table department
(departmentnumber number(3),
   departmentname varchar(10),
   departmentcity varchar(10));
```

Note that all SQL statements should be terminated by semi colon (;).

There is one to many relationship between employee and department table. Department is the parent table and employee is the child table. Departmentnumber column in department table is a primary key and departmentno. Column in employee table is a foreign key.

Note Constraints are discussed in the forthcoming sections.

7.4 NAMING CONVENTION

The table and column names should start with a letter and can be followed by letters, numbers or underscores. The length should not exceed a total of 30 characters. Please do not use any SQL reserved keywords as name for the tables or columns (such as "select", "create", "insert", etc).

Data types specify what the type of data can be used for that particular column. If a column called "Last_Name" is used to hold names, then that particular column should have a "varchar" (variable-length character) data type.

7.5 DATA TYPES

The following is the list of in-built data types:

VARCHAR2 (*size* [BYTE | CHAR])

Variable-length character string has maximum length *size in bytes* or characters. Maximum *size* is 4000 bytes or characters and minimum is 1 byte or 1 character. You must specify *size* for VARCHAR2.

BYTE indicates that the column will have byte length semantics;

CHAR indicates that the column will have character semantics.

NVARCHAR2(*size*)

Variable-length Unicode character string has maximum length *size* characters. The number of bytes can be up to two times the *size* for AL16UTF16 encoding and three times the *size* for UTF8 encoding. Maximum *size* is determined by the national character set definition with an upper limit of 4000 bytes. You must specify *size* for NVARCHAR2.

NUMBER[(*precision* [, *scale*]])

Number having precision *p* and scale *s*. The precision *p* can range from 1 to 38. The scale *s* can range from -84 to 127.

LONG

Character data of variable length up to 2 gigabytes.

DATE

Valid date range from January 1, 4712 BC to December 31, 9999 AD. The default format is determined explicitly by the NLS_DATE_FORMAT parameter or implicitly by the NLS_TERRITORY parameter. The size is fixed at 7 bytes. This data type contains the date and time fields. Its format consists of YEAR, MONTH, DAY, HOUR, MINUTE, and SECOND. It does not have fractional seconds or a time zone.

BINARY_FLOAT

This data type has 32-bit floating point number. It requires 5 bytes including the length byte.

BINARY_DOUBLE

It is a 64-bit floating point number. This data type requires 9 bytes, including the length byte.

RAW(*size*)

Raw binary data of length *size* bytes. Maximum *size* is 2000 bytes. You must specify *size* for a RAW value.

LONG RAW

Raw binary data of variable length up to 2 gigabytes.

ROWID

Base 64 string representing the unique address of a row in its table. This data type is primarily for the values returned by the ROWID pseudo column.

UROWID [(*size*)]

Base 64 string representing the logical address of a row of an index-organized table. The optional *size* is the size of a column of type UROWID. The maximum size and default is 4000 bytes.

CHAR [(*size* [BYTE | CHAR])]

Fixed-length character data of length *size* bytes. Maximum *size* is 2000 bytes or characters. Default and minimum *size* is 1 byte. BYTE and CHAR have the same semantics as for VARCHAR2.

NCHAR[(*size*)]

Fixed-length character data of length *size* characters. Maximum *size* is determined by the national character set definition, with an upper limit of 2000 bytes. Default and minimum *size* is 1 character.

CLOB

A character large object containing single-byte or multi byte characters. Both fixed-width and variable-width character sets are supported, both using the database character set. Maximum size is (4 gigabytes - 1) * (database block size).

NCLOB

A character large object containing Unicode characters. Both fixed-width and variable-width character sets are supported, using the database national character set. Maximum size is (4 gigabytes - 1) * (database block size). It stores national character set data.

BLOB

A binary large object. Maximum size is (4 gigabytes - 1) *(database block size).

BFILE

It contains a locator to a large binary file stored outside the database. It enables byte stream I/O access to external LOBs residing on the database server. Maximum size is 4 gigabytes.

INTERVAL YEAR [(*year_precision*)] TO MONTH

Stores a period of time in years and months, where *year_precision* is the number of digits in the YEAR datetime field.

Accepted values are 0 to 9. The default is 2. The size is fixed at 5 bytes.

INTERVAL DAY [(*day_precision*)] TO SECOND [(*fractional_seconds*)]

Stores a period of time in days, hours, minutes, and seconds,

Where,

- *day_precision* is the maximum number of digits in the DAY datetime field. Accepted values are 0 to 9. The default is 2.
- *fractional_seconds_precision* is the number of digits in the fractional part of the SECOND field. Accepted values are 0 to 9. The default is 6.

The size is fixed at 11 bytes.

TIMESTAMP [(*fractional_seconds*)]

Year, month and day values of date, as well as hour, minute, and second values of time, where *fractional_seconds_precision* is the number of digits in the fractional part of the SECOND datetime field. Accepted values of *fractional_seconds_precision* are 0 to 9. The default is 6. The default format is determined explicitly by the NLS_DATE_FORMAT parameter or implicitly by the NLS_TERRITORY parameter. The size varies from 7 to 11 bytes, depending upon the precision. This data type contains the datetime fields YEAR, MONTH, DAY, HOUR, MINUTE and SECOND. It contains fractional seconds but does not have a time zone.

TIMESTAMP [(*fractional_seconds*)] WITH TIME ZONE

All values of TIMESTAMP as well as Time Zone Displacement value, where *fractional_seconds_precision* is the number of digits in the fractional part of the SECOND datetime field. Accepted values are 0 to 9. The default is 6. The default format is determined explicitly by the NLS_DATE_FORMAT parameter or implicitly by the NLS_TERRITORY parameter. The size is fixed at 13 bytes. This data type contains the datetime fields YEAR, MONTH, DAY, HOUR, MINUTE, SECOND, TIMEZONE_HOUR, and TIMEZONE_MINUTE. It has fractional seconds and an explicit time zone.

TIMESTAMP [(*fractional_seconds*)] WITH LOCAL TIME ZONE

All values of TIMESTAMP WITH TIME ZONE, with the following exceptions:

- Data is normalized to the database time zone, when it is stored in the database.
- when the data is retrieved, users see the data in the session time zone.

The default format is determined explicitly by the NLS_DATE_FORMAT parameter or implicitly by the NLS_TERRITORY parameter. The size varies from 7 to 11 bytes depending on the precision.

To summarize, some of the important in-built data types are:

char(size)	Fixed-length character string. Size is specified in parenthesis.
varchar(size)	Variable-length character string. Max size is specified in parenthesis.
number(size)	Number value with a max number of column digits specified in parenthesis.
Date	For each date value, A date column stores the century, year, month, day, and hour, minute and second. The 24 hours format is used for time.
RAW(size)	Used to store byte oriented data character strings and graphic character sequence. ORACLE displays RAW data as hexadecimal character values; The size parameter is optional. If omitted, it defaults to 1.
Long	Stores variable length string. A table may have one long column. Long columns cannot be indexed. They cannot be reproduced by the function. They cannot be used in expression, nested query, distributed query, in a query involving set operators (such as UNION, INTERSECT or MINUS), where, group by, order by, connect by or distinct.
Long Raw	To store byte oriented data, character strings and graphics character sequence. ORACLE displays RAW data as hexadecimal character values.

Note: If you insert a date with no time, default is 00:00:00 midnight. If you insert a time with no date, default is system date. If you don't specify a date format, default is DD-MON-YY (07-FEB-96). Addition and Substraction can be performed between two dates.

7.6 INTEGRITY CONSTRAINTS

Integrity is required to protect the data. The DDL includes commands for specifying integrity constraints that the data stored in the database must satisfy. Updates that violate integrity constraints are not allowed. Through Integrity constraints, we can stop un-authorized users to enter the invalid data.

In SQL, we can categorize data constraints in two parts:

Column Level Constraints

If the constraints are defined along with the column definition, it is called as a column level constraint. Column level constraint can be applied to any column at a time *i.e*. they are local to a specific column. If the constraint spans across multiple columns, the user will have to use table level constraints.

Table Level Constraints

If the data constraint attached to a specific cell in a table references the contents of another cell in the table, then the user will have to use table level constraints. Table level constraints are stored as a part of the global table definition.

Different constraints that can be applied on the table are as follows:

NULL Value Concepts

If a row lacks a data value for a particular column, that value is said to be null. Columns of any data types may contain null values. Setting a null value is appropriate, when the actual value is unknown. A null value is not same as value of zero. A null value will evaluate to null in any expression. When a column name is defined, as not null, then the user is forced to enter the data into that column.

NOT NULL as a column constraint

```
CREATE TABLE Employee
(employeenumber varchar2(7) NOT NULL,
employeename varchar2(8) NOT NULL,
);
```


7.7 KEY CONCEPTS

Primary key uniquely identifies a row and if one column is used, it is as good as multiple columns. Multiple columns (composite keys) are used only when the system designed requires a primary key that cannot be contained in a single column.

Example:

PRIMARY KEY as a column constraint:

```
CREATE TABLE employee
employeenumber varchar2(7) PRIMARY KEY,
employeename varchar2(10), job varchar2(20), joiningdate date,
employeecity varchar2(10), departmentnumber number(3), salary
number(6), mgrnumber number(3));
```

PRIMARY KEY as a table constraint:

```
CREATE TABLE employee
( employeenumber varchar2(7) primary key,
employeename varchar2(10), job varchar2(20), joiningdate date,
employeecity varchar2(10), departmentnumber number(3), salary
number(6), mgrnumber number(3), constraint pk_sk primary
key(employeenumber));
```

Here, pk_sk is a constraint name.

7.8 UNIQUE KEY

A unique key is similar to a primary key, except that the purpose of a unique key is to ensure that information in the column for each record is unique, with the department or employee numbers. A table may have many unique keys**.**

Example:

UNIQUE as a column constraint:

```
CREATE TABLE client
(client_no varchar2( 7) CONSTRAINT cn_ukey UNIQUE,
name varchar2(10), address 1 varchar2(10), address2
varchar2(10),
city varchar2(10), state varchar2(l0), pincode number(6));
```

UNIQUE as a table constraint:

```
CREATE TABLE client
(client_no varchar2( 7) ,
name varchar2(10), address 1 varchar2(10), address2
varchar2(10),
city varchar2(10), state varchar2(l0), pincode number(6),
CONSTRAINT cn_ukey UNIQUE (client_no));
here, cn_ukey is a constraint name.
```

7.9 DEFAULT VALUE

A ‘default value’ can be assigned to a cell. When the user is loading a ‘record’ with values and leaves this cell empty, the cell will automatically be loaded with the default value specified (The data type of the default value should match the data type of the column). You can use the default clause to specify any default value as per your requirements.

```
create table employee
(employeenumber number(3),
employeename varchar(15),
job varchar(15),
salary number(16),
employeecity varchar(10),
```

```
joiningdate date,
departmentno. number(3), mgrnumber number(3), employee_type
default 'm');
```

7.10 FOREIGN KEY

Relationships between tables can be established using a Foreign Key. A Foreign Key is a Column (or a group of columns) whose values are derived from the primary key of the same or some other table. As Referential Integrity says that every foreign key refers its primary key. A foreign key must have a corresponding primary key value in the primary key table.

Example: The departmentnumber column is the primary key of the table Department. In table, Employee departmentno., is a foreign key that refer to the departmentnumber values in table department.

The Foreign Key References constraint: Note the following points at the time of imposing the foreign key constraints:

- Foreign key must refer only the table, not a view or cluster;
- Please check that the FOREIGN KEY column(s) and the CONSTRAINT column(s) have matching data types; this is the essential requirement.
- The foreign key constraint can refer the same table named in the CREATE TABLE statement;
- The foreign key constraint does not refer the same column more than once (in a single constraint).
- Foreign key rejects an INSERT or UPDATE of a value, if a corresponding value does not exist in the primary key table currently.
- Foreign key rejects a DELETE, if it invalidates a REFERENCES constraint

FOREIGN KEY as a column constraint:

```
CREATE TABLE employee
( departmentno. number(3) REFERENCES
department.departmentnumber);
```

FOREIGN KEY as a table constraint:

```
CREATE TABLE employee
( departmentno. varchar2( 6),
FOREIGN KEY (departmentno.) REFERENCES
department.departmentnumber);
```

7.11 CHECK INTEGRITY CONSTRAINTS

CHECK Integrity constraints are used when you need to enforce integrity rules that can be evaluated based on a logical expression. Please do not use CHECK constraints, if the constraint can be defined using not null, a primary key or foreign key constraint.

Example:

```
CREATE TABLE employee
(employeenumber varchar2(7) CONSTRAINT ck_empno
CHECK (salary >5000));
```

Here, constraint name is ck_empno.

Limitations of check constraint condition:

- The condition does not contain sub queries or sequences.
- The condition does not include the SYSDATE, UID, USER or USERENV SQL functions

7.12 ALTER TABLE

The ALTER TABLE statement is used to add or drop columns in an existing table.

ALTER TABLE table_name

ADD column_name datatype

ALTER TABLE table_name

DROP COLUMN column_name

Note: Some database systems don't allow dropping of a column, in a database table (DROP COLUMN column_name).

Defining Integrity constraints in the ALTER TABLE command:

You can also define integrity constraints using the constraint clause(add, modify and drop) in the ALTER TABLE command.

Example:

1. Add PRIMARY KEY constraint on column employeenumber in table employee;

ALTER TABLE employee

ADD PRIMARY KEY (employeenumber);

2. Add FOREIGN KEY constraint on column employeenumber in table employee referencing table department, modify column departmentcity include NOT NULL constraint.

ALTER TABLE employee
ADD CONSTRAINT a2
FOREIGN KEY (departmentno.) REFERENCES department
MODIFY (departmentcity varchar(8) NOT NULL);
here, a_2 is a constraint name.

Dropping Integrity constraints in the ALTER TABLE command:

You can drop an integrity constraint, if the rule that it enforces, is no longer true or if the constraint is no longer needed.

Example:

Drop the PRIMARY KEY constraint from employee;

```
ALTER TABLE employee
    DROP PRIMARY KEY;
```

Note: Dropping UNIQUE and PRIMARY KEY constraints drops the associated indexes.

Drop command

The drop table command is used to delete a table and all the rows in the table. Dropping the table removes the table definition as well as that of its rows.

Syntax:

```
drop table "tablename";
drop table employee;
```


7.13 SELECT COMMAND (DATA MANIPULATION COMMAND)

The select statement is used to query the database and retrieve the selected data which match the criteria that you specify.

Syntax:

```
select "column1"
[,"column2",etc]
from "tablename"
[where "condition"];
[] means optional
   or
SELECT [ALL | DISTINCT] column1[,column2]
FROM table1[,table2]
[WHERE "conditions"]
[GROUP BY "column-list"]
[HAVING "conditions]
[ORDER BY "column-list" [ASC | DESC] ]
```

You can select as many column names as you like, or you can use a "*" to select all the columns. This is known as global query.

Example: Select * from employee;

Where clause (optional) specifies which data values or rows will be returned or displayed, based on the criteria described after the keyword 'where'.

Concatenation operator:

We can concatenate columns using || operator.

Example: Display employee name concatenated by employee job separated by comma.

Select employeename || ','|| employeejob from employee;

Example: Display all the data from employee table; separate each column by comma and give the single column name to result as 'output'.

Select employeenumber||','||employeename||','||employeejob||','||mgrnumber||','||salary||','||employeecity||','||departmentno. As "output";

Is null:

Example: Display employeename, employeejob of all the employees who do not have a manager.

In the table, employee the query defined below will show the employees whose mgrnumber value is null.

Select employeename, employeejob from employee where mgrnumber is null;

Dual table: Dual table is a system table that contains one row and one column.

Example: Display current date.

```
select sysdate from dual;
sysdate
.............
15-sep-08
```

Example: Display employee name and calculate number of months between today and the date when the employee joined the organization.

Select employeename, months_between(sysdate, joiningdate) from employee;

Where, months_between is a function which contains two argument sysdate and joining date.

A **Pseudocolumn** behaves as a table column. Pseudocolumn is not stored in the table.

You can retrieve the data from pseudocolumns, but you cannot insert, update, or delete their values.

ROWID Pseudocolumn:

For each row in the table, the ROWID pseudocolumn returns the address of the row.

ROWID values have the following important uses:

- They are the fastest way to access a single row.
- They can show you, how the rows in a table are stored.
- They are unique identifiers for rows in a table.

Note: Don't use ROWID as the primary key of a table.

Example:

The following statement retrieves the address of all rows that contain data for employee in department 30:

```
SELECT ROWID, ename
FROM employee
WHERE deptno = 30;
```

Now let see the removal of duplicate records from a table:

There are many ways to do it, but we are going to discuss the two most common ways:

First approach which is Simpler and Faster where, we simply create another table say NTABLE having only the distinct rows of the original table say ORIGINTABLE, drop the ORIGINTABLE , and finally rename the NTABLE to the ORIGINTABLE.

Note: Dropping a table will drop all the indexes and privileges as well. So, you'll need to create them again.

```
CREATE TABLE NTABLE AS SELECT DISTINCT * FROM ORIGINTABLE;
DROP TABLE ORIGINTABLE;
RENAME NTABLE TO ORIGINTABLE;
...create indexes/privileges on ORIGINTABLE now...
```

Second approach which is also known as ROWID approach; We simply compare the ROWID of the records having the same key values (duplicate records) and select only one of the duplicate rows (the one having either the min or the max ROWID).

Note: These ROWID's are system generated and will never be duplicates, You won't be having more than one min (or max).

```
DELETE FROM ORIGINTABLE A1 WHERE ROWID > (SELECT MIN (ROWID)
FROM ORIGINTABLE A2 WHERE A1.KEY = A2.KEY);
OR
DELETE FROM ORIGINTABLE A1 WHERE ROWID < (SELECT MAX (ROWID)
FROM ORIGINTABLE A2 WHERE A1.KEY = A2.KEY);
```

Above, KEY represents the set of columns based on which the duplicates will be decided.

ROWNUM Pseudocolumn:

For each row returned by a query, the ROWNUM returns a number, representing the order in which SQL selects the row from a table or set of joined rows.

The first row selected has a ROWNUM 1, the second has 2, and so on.

Note: You can use ROWNUM to limit the number of rows returned by a query.

Example:

```
SELECT * FROM employee WHERE ROWNUM < 20;
```

If an ORDER BY clause follows ROWNUM in the same query, then the rows will be reordered with ORDER BY clause.

If you push in the ORDER BY clause in a subquery and place the ROWNUM condition in the top-level query, then you can force the ROWNUM condition to be applied after ordering of the rows.

Example: the following query returns the employees with 20 smallest employee numbers.

This is also known as **top-N reporting**.

Example:

```
SELECT * FROM
(SELECT * FROM employee ORDER BY empno)
WHERE ROWNUM < 21;
```


The ROWNUM values are of top-level SELECT statement, so they are generated after the rows have already been ordered by empno in the subquery.

Example:

Retrieve information of top 5 employees salary wise:

```
SELECT ROWNUM, empno, ename, salary
FROM (SELECT empno, ename, salary
From employee
ORDER BY salary DESC)
WHERE ROWNUM<=5;
```

You can also use ROWNUM to assign unique values to each row of a table.

Example:

```
UPDATE X
SET column1 = ROWNUM;
```

Where, X is the table name.

7.14 ARITHMETIC OPERATORS

SQL supports the following five basic arithmetic operators:

+	addition
-	substraction
*	multiplication
/	division
%	modulus

Example: Display employee name, salary for all employees whose salary has increased by 5%.

Select employeename, salary, from employee where sal=sal+sal*0.05;

The modulus operator determines the integer remainder of the division. This operator is not ANSI SQL supported, though, most databases support it.

7.15 BOOLEAN OPERATORS

Two Boolean operators (AND and OR) can be used to combine the conditions:

The AND operator can be used to join two or more conditions in the WHERE clause. Both sides of AND condition must be true, for the conditions to be met and for rows to be displayed.

Syntax:

```
SELECT column1,
SUM(column2) FROM "list-of-tables" WHERE "condition1" AND
"condition2";
```

The OR operator can be used to join two or more conditions in the WHERE clause. However, either side of the OR operator can be true and the condition will be met. Hence the rows will be displayed.

Example:

```
SELECT employeenumber, employeename, salary FROM employee WHERE
salary >= 70000 AND employeejob = 'manager';
```

Both the above conditions must be true for the rows to be returned in the query. If either is false, then it will not be displayed.

You can use parenthesis around your conditional expressions to make it easier to read:

```
SELECT employeenumber, employeename, salary FROM employee WHERE
(salary >= 70000) AND (employeejob = 'manager');
```

Example:

```
SELECT employeenumber, employeename, salary FROM employee WHERE
(employeejob='programmer') or (employeejob = 'manager');
```

The statement will select the employeenumber, employeename, employeejob and salary from the employee table where the employeejob is either equal to 'programmer' OR the employeejob is equal to 'manager'.

7.16 IN AND BETWEEN CONDITIONAL OPERATORS

Syntax:

```
SELECT col1, SUM (col2) FROM "list-of-tables" WHERE col3 IN
(list-of-values);
SELECT col1, SUM (col2) FROM "list-of-tables" WHERE col3 BETWEEN
value1 AND value2;
```

IN conditional operator (set membership test operator) is used to test whether or not a value is "in" the list of values provided after the keyword IN.

Example:

```
SELECT employeenumber, employeename, salary
FROM employee
WHERE employeename IN ('shikha', 'aranya', 'neelam', 'arya');
```

This statement will select the employeenumber, employeename, salary from the employee table, where the employeename is equal to either shikha, aranya, neelam or arya. It will return the rows, if it is any of these values.

You can also use NOT IN to exclude the rows in your list.

Example: Display name, job, salary of all the employees whose job is manager or programmer and their salary is not equal to 2000, 5000, 15000.

```
Select employeename, employeejob, salary from employee where
job='manager' or job='programmer' and sal not in (1000, 3000,
15000);
```

The BETWEEN conditional operator is used to test whether or not a value is "between" the two values. The BETWEEN ... AND operator selects a range of data between two values. These values can be numbers, text or dates.

```
SELECT column_name FROM table_name
WHERE column_name
BETWEEN value1 AND value2
```

Example:

```
SELECT employeenumber, salary, employeename FROM employee WHERE
salary BETWEEN 6000 AND 7000;
```

This statement will select the employeenumber, salary, employeename from the employee table, where the salary is between 6000 and 7000 (including 6000 and 7000).

You can also use NOT BETWEEN to exclude the values between your range.

Example: Display employeename and salary for all the employees whose salary is not in the range of 3000 and 5000.

Select employeename, salary from employee where salary is not between 3000 and 5000.

7.17 STRING OPERATIONS

The most commonly used operation on strings is pattern matching, using the operator, like. We describe patterns by using two special characters:

Percent (%): The % character matches any sub string.

Underscore (-): The – character matches any character.

Patterns are case sensitive, i.e. is, uppercase characters do not match lowercase characters, or vice versa.

Example:

'Raj%' matches any string beginning with "Raj".

'%comp%' matches any string containing "comp" as a sub string.

'- - -' matches any string of exactly three characters.

'- - %' matches any string of at least two characters.

SQL expresses patterns by using the like comparison operator.

Example:

Consider the query "Find the names of all the employees whose name includes the substring 'kha' .This query can be written as:

```
select employeename
from employee
where employeename like '%kha%';
```

Example: Display name of all the employees where third letter of their name is b.

Select employeename from employee where ename like '– –b%';

7.18 SET OPERATIONS

There are three set operators namely UNION, INTERSECT and MINUS. Calculus includes a version of the basic set operations of union, intersection, and set difference. To make binary operations possible, the two relations on

which they are performed must be union compatible. Union compatibility means that it is possible to do a union operation since they have the same basic structure (they must have the same degree and attributes in the corresponding position in both the tables).

THE UNION OPERATION

```
(select ename
from employee)
union

(select ename
from emp)
```

The union operation eliminates duplicates automatically, unlike the select clause. If we want to retain all the duplicates, we must write union all in place of union:

```
( select ename
from employee)
union all (select ename
from emp)
```

The number of duplicate tuples in the result is equal to the total number of duplicates that appear in both the tables.

THE INTERSECT OPERATION

```
(select distinct ename
from employee)
intersect (select distinct ename
from emp)
```

The intersect operation eliminates duplicates automatically. If we want to retain all the duplicates, we must write intersect all in place of intersect:

```
(select ename
from employee)
intersect all (ename
from emp)
```

The number of duplicate tuples that appear in the result is equal to the minimum number of duplicates in both the tables.

THE MINUS OPERATION

```
(select ename
from employee)

 minus

(select ename
from emp)
```

EMPLOYEE

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SAGAR	40000	SALES	EXECUTIVE
ESHA	30000	CHEMISTRY	VICE PRINCIPAL

EMP

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SAGAR	40000	SALES	EXECUTIVE
RAVI	30000	LIBRARY	PROGRAMMER
UJJAWAL	20000	PURCHASE	OFFICE ATTENDENT
MOHAN	200000	PHYSICS	PRINCIPAL

EMPLOYEE UNION EMP:

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SAGAR	40000	SALES	EXECUTIVE
ESHA	30000	CHEMISTRY	VICE PRINCIPAL
RAVI	30000	LIBRARY	PROGRAMMER
UJJAWAL	20000	PURCHASE	OFFICE ATTENDENT
MOHAN	200000	PHYSICS	PRINCIPAL

EMPLOYEE INTERSECTION EMP:

ENAME	SAL	DNAME	JOB
RAJ	50000	COMPUTER	LECTURER
SAGAR	40000	SALES	EXECUTIVE

EMPLOYEE MINUS EMP:

ENAME	SAL	DNAME	JOB
ESHA	30000	CHEMISTRY	VICE PRINCIPAL

All the columns in the first table must have the same domain as the the columns in the second table, in one to one correspondence. The result of each set operations is a new table with the same structure as two original tables.

7.19 MATHEMATICAL FUNCTIONS

ABS(x)	returns the absolute value of x
SIGN(x)	returns the sign of input x as -1, 0, or 1 (negative, zero, or positive respectively)
MOD(x,y)	returns the integer remainder of x divided by y (same as x%y)
FLOOR(x)	returns the largest integer value that is less than or equal to x
CEILING(x) or CEIL(x)	returns the smallest integer value that is greater than or equal to x
POWER(x,y)	returns the value of x raised to the power of y
ROUND(x)	returns the value of x rounded to the nearest whole integer
ROUND(x,d)	returns the value of x rounded to the number of decimal places specified by the value d
SQRT(x)	returns the square-root value of x

Examples:

SELECT round(salary), employeename FROM employee;

The above statement will select the salary rounded to the nearest whole value and the employee name from the employee table.

Example: select abs(-34) from dual;

In the above statement, we have used function abs(n) which returns the absolute value of n *i.e*. 34.

Example: select power(4,2) from dual;

In the above statement, we have used function power(m,n) returns m raised to nth power *i.e*. 16; where n must be an integer value.

Example: select sqrt(36) from dual;

In the above example, sqrt(n) returns square root of n *i.e*. 6. If the value of n is less than 0, then sqrt(n) will return a null value.

7.20 CONDITIONAL OPERATORS

=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<>	Not equal to
LIKE	*See note below

*The LIKE pattern matching operator can also be used in the conditional selection of the where clause. Like is a very powerful operator that allows you to select only rows that are "like" which has been specified. The percent sign "%" can be used as a wild card to match any possible character which might appear before or after the characters specified.

Examples:

```
select employeename, employeecity
from employee
where employeename LIKE 'sh%';
```

This SQL statement will match any name that starts with 'sh'. Strings must be in single quotes.

Or

You can specify,

```
select employeename
from employee
where employeename LIKE '%a';
```

This statement will match any name that ends with 'a'.

```
select * from employee
where employeename = 'Rajan';
```

This will only select rows, where the name equals 'Rajan' exactly.

Example: Display name and salary of employees earning more than 4000.

Select employeename, salary from employee, where salary>4000;

Example: Display employee name and department number for employee number 111.

```
Select employeename, departmentno. From employee where
employeenumber=111;
```

7.21 ALIASES

With SQL, aliases can be used for column names and table names.

Column Name Alias

The syntax is:

```
SELECT column AS column_alias FROM table
Select employeename as ename from employee;
```

Table Name Alias

The syntax is:

```
SELECT column FROM table AS table_alias
Select employeename from employee as my_employee;
```

7.22 THE RENAME OPERATION

SQL provides a mechanism for renaming both relations and attributes.

Syntax:

old-name as new-name

Example:

```
select employeename as name, employeecity as my_city from
employee;
```

ALL and DISTINCT keywords are used to select either ALL (default) or the "distinct" or unique records in your query results. If you would like to retrieve just the unique records in the specified columns, you can use the "DISTINCT" keyword. DISTINCT will discard the duplicate records for the columns you specified, after the "SELECT" statement:

For example:

```
SELECT DISTINCT employeename FROM employee;
```

This statement will return all the unique names in the employee table.

ALL will display "all" of the specified columns, including all the duplicates. ALL keyword is default, if nothing is specified.

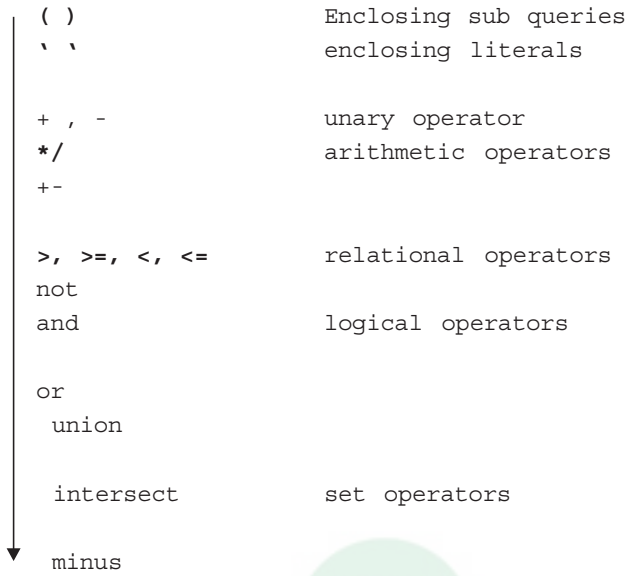
Example: Display unique employee jobs from employee table.

```
Select distinct employeejob from employee;
```

7.23 RULES OF PRECEDENCE AND ASSOCIATIVITY

When more than one operator appears in an expression, the order of evaluation depends on the rules of precedence:

Highest



Lowest

When the precedence of two operators is same, the associativity will be from left to right. The evaluation of assignment operator is always from right to left. Parenthesis raises the precedence of the operations that are inside them. Often this becomes necessary to obtain the result we desire. For example,

```
x*y+8

x*(y+8)
```

In this expression, first multiply x to y, then add 8. However, if you want to first add 8 to y, then multiply it to x.

By adding parenthesis we can reduce the ambiguity.

7.24 GROUP (AGGREGATE) FUNCTIONS

MIN	returns the smallest value in a given column
MAX	returns the largest value in a given column
SUM	returns the sum of numeric values in a given column
AVG	returns the average value of a given column
COUNT	returns the total number of values in a given column
COUNT(*)	returns the number of rows in a table

Aggregate functions are used to compute against a "returned column of numeric data" from your SELECT statement. They basically summarize the results of a particular column of selected data. These functions cannot be used with "GROUP BY" clause.

For example:

```
SELECT AVG(salary) FROM employee;
```

This statement will return a single result which contains the average value of everything returned in the salary column from the employee table.

Example:

```
SELECT AVG(salary) FROM employee WHERE employname = 'shikha';
```

This statement will return the average salary for all the employees whose title is equal to 'shikha'

Example:

```
SELECT Count(*) FROM employee;
```

This particular statement is slightly different from the other aggregate functions since the column is not supplied to the count function. This statement will return the number of rows in the employee table.

Example: Display highest, lowest, sum and average salary of all the employees.

```
Select max(salary), min(salary), sum(salary), avg(salary)
From employee;
```


Example: Display average salary of employees.

Select avg(salary) from employee;

7.25 SQL CLAUSES

In the following sections, we are describing some of the clauses used in SQL.

7.25.1 Group by Clause

GROUP BY... was added to SQL because aggregate functions (like SUM) return the aggregate of all column values every time they are called, and without the GROUP BY function it was impossible to find the sum for each individual group of column values.

The GROUP BY clause will gather all the rows together that contain data in the specified column(s) and will allow aggregate functions to be performed on one or more columns.

Syntax:

```
SELECT column1, group function(column2)
FROM "list-of-tables" GROUP BY "column-list";
```

If you would like to retrieve a list of the lowest paid salaries in each dept:

```
SELECT min(salary), departmentno. FROM employee GROUP BY
departmentno.;
```

This statement will select the minimum salary for the people in each unique department. The salary for the person who gets the maximum in each department, will be displayed. Their salary and their department will be returned.

```
SELECT max(salary), departmentno. FROM employee GROUP BY
departmentno.;
```

GROUP BY **- Multiple Grouping Columns**

Example:

If we write like this,

```
SELECT employeename, max(salary), departmentno. FROM employee
GROUP BY departmentno., employeename;
```

This is known as "multiple grouping columns".

Example: Display minimum, maximum, sum and average salary for each jobtype.

```
Select max(salary), min(salary), sum(salary), avg(salary) from
employee group by job;
```

Example: Display Number of people with same job.

```
Select employeejob, count(employeejob) from employee group by
employeejob;
```

Example: Display the difference between highest and lowest salaries:

```
Select  max(salary)-min(salary)
From employee;
```

7.25.2 Having Clause

The HAVING clause allows you to specify conditions on the rows for each group. The HAVING clause should follow the GROUP BY clause, if you are going to use it.

Syntax:

```
SELECT column1, group function(column2) FROM "list-of-tables"
GROUP BY "column-list" HAVING "condition";
```

For example, You have an employee table containing the employee's name, departmentno., salary and employeejob. If you would like to select the average salary for each employee in each department, you could enter:

```
SELECT departmentno., avg(salary) FROM employee GROUP BY
departmentno.;
```

But, if you want to only calculate & display the average, if their salary is more than 50000:

```
SELECT departmentno., avg(salary) FROM employee GROUP BY
departno. HAVING avg(salary) > 50000;
```

7.25.3 Order by Clause

ORDER BY clause will allow you to display the results of your query in a sorted order (either ascending order or descending order) based on the columns that you specify to order. This is an optional clause.

Syntax:

```
SELECT column1, group function(column2)
FROM "list-of-tables"
ORDER BY
"column-list" [ASC | DESC];
[ ] means as optional.
ASC = Ascending Order
DESC = Descending Order
```

Example:

```
SELECT employeenumber, departmentno, employeename, salary
FROM employee
WHERE salary=6000
ORDER BY employeename;
```


This statement will select the employeenumber, departmentnumber, employeename and salary from the employee table, where the salary equals 6000 and will list the results in ascending (default) order based on their employee name.

If you would like to order based on multiple columns, you must separate the columns with commas.

Example:

```
SELECT employeenumber, departmentno., employeename, salary
FROM employee WHERE salary = '15000'

ORDER BY employeenumber, employeename DESC;
```

Example: Display the employee name, job, joining date of employees joined between Feb 20, 1981 and May1, 1995. Order the query in ascending order of the start date.

```
Select employeename, employeejob, joiningdate from employee
where joiningdate between '20-feb-1981' and '01-may-1995' order
by joiningdate asc;
```

Example: Display employeename and departmentnumber of all the employees in department 01 and 03 in alphabetical order by name.

```
Select employeename, departmentnumber from employee where
departmentno. =01 or departmentno. =03 order by ename asc;
```

If we will not mention any specific order by clause, then the results will be displayed in ascending order.

7.26 JOIN OPERATORS

Joins allow you to link data from two or more tables together into a single query result or it help in retrieving data from more than one table. If you are retrieving the data from more than one table without giving any condition, then the query will print all the ordered pairs of rows of both the tables. This is Cartesian product.

Syntax:

```
SELECT "list-of-columns" FROM table1, table2 WHERE "search-
condition(s)"
```

Note that each of the table has a common column. And the common column is used to join the two tables. Here is an example of a join statement: If there are two tables employee (ename,eno,sal, deptno) and Department(deptno, dname, loc) and one common attribute deptno.

```
SELECT emp.ename, emp.sal, dept.dname, dept.loc
FROM employee emp, department dept WHERE emp.deptno =
dept.deptno;
(In this example, we are using aliases as emp and dept).
```

Or

```
SELECT employee.ename, employee.sal, department.dname,
department.loc
FROM employee, Department WHERE employee.deptno =
department.deptno;
```

This particular "Join" is known as an "Inner Join" or "Equijoin". Notice that each of the column is always preceded with the table name and a period.

Outer Join: You are retrieving the data from more than one table and one attribute is common in both the tables. Now, if you perform join on both the tables and suppose one table is deficient, in such case some of the records will not be printed. For example, there are two tables, employee with attributes (ename, eno., sal, deptno) and Department with attributes (deptno, dname, loc) and one common attribute deptno. Suppose in a department table, there is department number 50 and in employee table there is no such department, Now if we perform join on both the tables then the row of department number 50 will

not printed. To print that row, we use the concept of outer join. In this, case deficient table is employee which is the child table. And, here we will use left outer join.

Example:

```
SELECT emp.ename, emp.sal, dept.dname,dept.loc
FROM employee emp, Department dept WHERE emp.deptno(+) =
dept.deptno;
```

After executions of this query, the details of employee having department number 50 will also be printed. If the right hand side table(parent table) is deficient, then we use right outer join.

```
SELECT emp.ename, emp.sal, dept.dname,dept.loc
FROM employee emp, Department dept WHERE emp.deptno =
dept.deptno(+)
```

Self Join: A table can be joined with itself on column that contains the same type of information. A self join joins rows from a table to other or same rows in that table**.**

If the employee table is joined to itself using the columns employeenumber and mgrnumber, then both the Employeenumber and mgrnumber columns contain the same type of information of employee numbers. For each employee, (for each row in the table) mgrnumber is the employee number of the employee's manager, and the Employeenumber is the employee number of the employee named in that row.

Example: List the name and Job title of each employee who has a manager:

```
Select worker.employeename, worker.employeejob,
worker.mgrnumber,
manager.employeenumber,manager.mgrno,manager.employeename
mname, manager.employeejob mjob
From employee worker, employee manager
Where worker.mgrnumber=manager.empno;
```

Note: If you omit the join clause, a Cartesian join is performed.

7.27 NESTED QUERIES

The result of one query can be dynamically substituted into the where clause of another query. SQL queries are formed by the multiple queries called nested queries. The sub query enclosed in parenthesis is evaluated first and the retrieved value is substituted directly into the outer query. Queries can be nested up to 16 levels. Up to 255 sub queries can appear on a level.

Types of nested queries:

Nested queries returning a single row: It contains an inner query which returns a single result. The inner query is evaluated first. Its result is then used as a constant in the outer query.

Example:

```
Select employeename, salary from employee where
employeejob=(select employeejob from employee where
employeename='shikha');
```

Nested queries returning multiple rows: Sub queries may return a list of values. In such queries, the operators IN or NOT IN should be used to link the outer query to the list of values retrieved by the sub query.

Nested queries returning multiple columns: Queries can be linked by more than one column. In such queries, the order of columns named in the where clause must be placed in parenthesis.

Example: Find out the highest paid employee in each department:

Select employeename, departmentno., employeejob, salary from the employee where (departmentno., salary) in(select departmentno., max(salary) from emp group by departmentno.).

Example: Display the employee name and joining date for all the employees who are in same department as 'shikha'.

```
Select employeename, joiningdate from employee where
departmentnumber=(select departmentno. From employee where
employeename='shikha');
```

Example: Display employee number and employee name of all employees who earn more than average salary.

```
Select employeenumber, employeename from employee where
salary>(select avg(salary) from employee);
```

7.28 SOME OTHER IMPORTANT FUNCTIONS

lower(char): Returns char with all the letters in lowercase.

Example: select lower('rajat') from dual;

upper(char): Returns char with all the letters in uppercase.

Example: select upper('shikha') from dual;

length(char): returns the length of char.

Example: select length('shikha') from dual;

Length is the only character function which returns number. In the given example, the length of char is 6.

lpad(char1, n, char2): Returns char1, left padded to length n with the sequence of characters in char2.

rpad(char1, n, char2): Returns char1, right padded to length n with the sequence of characters in char2.

ltrim(char, set): Returns char after removing characters to its left, up to the first character not in set.

rtrim(char, set): Returns char after removing characters to its right, after the last character not in set.

substr(char,x,y): Returns a part of character, from beginning at character x, up to y characters long; if y is not given, then up to the end of char.

to_char(number, 'format'): This function converts a number data type value into char data type, using the optional format string. If format is omitted, then the number is converted to a character value.

to_char(date, 'format'): This function converts a date data type value into char data type using the optional format string. If format is omitted, then the date is converted to a character value having the default date format, i.e. (dd-mon-yy).

to_number(char): This function converts character data type value containing a number to a number data type value.

to_date(char, format): This function converts character data type value to a date data type value.

Initcap(char): Converts the characters into characters with first letter in capital.

NVL (column, value1): NULL values are not used while evaluating expression or functions. If a null value is in arithmetic calculation, the result will be always a NULL value. NVL function converts a null value into not null value. If the value of column is null, then it will be replaced by value1.

Example: Display name and salary for all employees. Format the salary to be 20 character long, left padded with a @ sign.

```
Select employeename, lpad(concat('@', salary),20) from employee;
```

Example: Display employees name with first letter capitalized and all other in lower case and length of their name for all employees whose name starts with ktp.

```
Select initcap(lower(employeename)), length(employeename) from
employee where employeename like 'ktp%';
```

Example: Display name, hiredate and day of the week on which employee started.

```
Select employeename, joiningdate, to_char(joiningdate, 'day')
From employee;
```

Decode function: Working of decode function in SQL is same as switch statement in javascript.

Example: Display employee name and salary amount. If the employee does not earn salary put 'no salary'.

```
Select ename, decode (salary, null, 'no salary',salary) from
employee;
```

In the above example, if an employee is not getting any salary, i.e. salary is null; Then "no salary" will be printed in the salary column otherwise salary amount will be printed.

Some other Functions:

Add_month(date, number)

Months_between(date1, date2)—find how many months between two dates.

Last_day(date)————————————Last day of the month

Next_day(date, dayname)————Specific day of the week.

Vsize(str)—————Find the no. of characters required to store the strings.

Soundex(str)————————Produces a code for a character string.

7.29 INSERT DATA INTO A TABLE (DML COMMAND)

The INSERT statement is used to insert or add a row of data into the table.

Syntax:

```
insert into "tablename"
(first_column,...last_column)
values  (first_value,...last_value);
```

The values that you enter will be held in the rows and they will match with the column names that you specify. Strings and dates should be enclosed in single quotes.

Example:

```
insert into employee
(employeename, employeejob, salary, joiningdate, employeecity)
values ('Rajan', 'manager', 32000, '10-sept-1982',
'Gurgaon');
```

Note that all strings (char and date datatype) should be enclosed between single quotes: 'Gurgaon'.

Suppose we have to enter 1000 records into the table, every time we will have to write the INSERT command again and again. This is a very cumbersome process. For this, we use substitution parameter(&). By using substitution parameter, we can provide the value at run time.

```
insert into employee
(employeename, employeejob, salary, joiningdate, employeecity)
values ('&ename', '&ejob', &sal, '&joindate', '&empcity');
```

When we run insert command using / at SQL prompt, the values will be asked at run time like the following:

Enter the value of ename: raj

Enter the value of ejob: manager

Enter the value of sal: 40000

Enter the joining date: 12-oct-1991

Enter the value of employee city: New Delhi

These values will be stored into the table. For entering another record, use / and enter at the SQL prompt.

This process will be repeated until all the records are entered.

7.30 MODIFICATION OF THE DATABASE (DML COMMAND)

UPDATE

The update statement is used to update or change the records that matches the specified criteria.

Syntax:

```
update "tablename"
set "columnname" =
"newvalue"
[,"nextcolumn" =
"newvalue2"...]
where "columnname"
OPERATOR "value"
[and|or "column"
OPERATOR "value"];
[] means optional
```

Example:

```
update employee
set salary = 5000
where employeenumber=113;
```

7.31 DELETING RECORDS (DML COMMAND)

The delete statement is used to delete records or rows from the table.

Syntax:

```
delete from "tablename"
where "columnname"
OPERATOR "value"
```

```
[and|or "column"
OPERATOR "value"];
[ ] = optional
```

Example: delete from employee;

Note, that if you leave off the where clause, all the records will be deleted!

```
delete from employee
where employeename = 'shikha';
delete from employee
where employeename = 'rajan' or employeename = 'dixit';
```

7.32 INDEXES

By creating indexes you can make the searching of records faster.

The following statement will create an index on the primary key departmentno. of table employee.

```
create index endx
on employee (departmentno.)
```

You can also create unique index on the primary key departmentno. of the table employee.

```
create unique index endx
on employee(departmentno.)
```


You can drop indexes using drop command.

```
drop index endx;
```

Above, endx is the index name.

7.33 INDEX STRUCTURE AND ACCESS

- The top level of an index is usually held in memory. It is read once from disk at the start of queries.
- Each index entry points to either another level of the index, a data record, or a block of data records.
- The top level of the index is searched to find the range within which the desired record lies.
- The appropriate part of the next level is read into memory from disk and searched.
- This continues until the required data is found.

Costing Index and File Access

- The major cost of accessing an index is associated with reading in each of the intermediate levels of the index from a disk (milliseconds).

- Searching the index, once it is in the memory is comparatively inexpensive (microseconds).
- The major cost of accessing the data records involves waiting for the media to recover the required blocks (milliseconds).
- Some indexes mix the index blocks with the data blocks, which means the disk accesses can be saved because the final level of the index is read into memory with the associated data records.

7.34 USE OF INDEXES

- A DBMS may use different file organizations for its own purposes.
- A DBMS user is generally given little choice of file type.
- A B+ Tree is likely to be used wherever an index is needed.
- Indexes are generated:
  - (Probably) for fields specified with 'PRIMARY KEY' or 'UNIQUE' constraints in a CREATE TABLE statement.
  - For fields specified in SQL statements such as CREATE [UNIQUE] INDEX indexname ON tablename (col [,col]...);
- Primary Indexes have unique keys.
- Secondary Indexes may have duplicates.
- An index on a column which is used in an SQL 'WHERE' predicate is likely to speed up a query.
- Execution of Query speed up when '=' is involved (equijoin)
- No improvement will occur with 'IS [NOT] NULL' statements.
- An index is best used on a column with widely varying data.
- Indexing a column of Y/N values might slow down enquiries.
- An index on telephone numbers might be very good but an index on area code might be a poor performer.
- Multicolumn index can be used, and the column which has the biggest range of values or is the most frequently accessed should be listed first.
- Avoid indexing for small relations, frequently updated columns, or those with long strings.
- There may be several indexes on each table. Note that partial indexing normally supports only one index per table.
- Reading or updating a particular record should be fast.
- Inserting records should be reasonably fast. However, each index has to be updated too, so increasing the indexes makes the process slower.
- Deletion may be slow, particularly when the indexes have to be updated.
- Deletion may be fast, if the records are simply flagged as 'deleted'.

7.35 VIEWS

Views are the logical container of data. In SQL, a VIEW is a virtual table based on the result-set of a SELECT statement.

A view contains rows and columns, just like a table. The fields in a view are fields from one or more tables in the database. You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data is coming from a single table.

Note: The database design and structure will not be affected by the functions, where, or join statements in a view.

A view could be used from inside a query, a stored procedure, or from inside of another view. By adding functions, joins, etc., to a view, it allows you as a DBA to present exactly the data you want for the user.

Syntax:

```
create view view_name as
select columnname, columnname
from tablename
 where columnname=expression list
```

Example:

Create a view consisting of the names, numbers and jobs of employees in department 40.


```
create view dept40 as
select employeenumber, employeename, employeejob
from employee
where deparmentno.=40;
```

The create view command causes SQL to store the query in the data dictionary. A view cannot be used to delete row from a table. A view can be used for updating a table except where the view contains virtual column or join tables. If you will make any changes to the original table, the changed data in the original table will be automatically reflected in the view. You can also select a data set from the view.

7.36 DROPPING A VIEW

Syntax: drop view viewname:

Example:

drop view dept40;

Advantages:

- Simplification of queries
- In a concurrent environment, where several people are querying a database, different users can have varying views on the same underlying table.
- View provides a powerful and flexible means of controlling access to data.
- View is a powerful security mechanism in SQL. Rather than allowing access to an entire table, you can create a view on the rows and columns which are necessary for a particular application.
- View can be queried as a table itself.
- Avoids data redundancy.

7.37 CLUSTERS

Performance of join queries can be improved by using cluster object.

Syntax:

```
Create cluster cluster-name
   column-name1 type1 [, column name2 type 2]…)

Alter cluster cluster-name
Add table table name
   Where [cluster-name] cluster key column1=[table name]
```

Always clusters should be created and tables added to the clusters before data is loaded.

For example:

```
   Create cluster xyz
   ( deptno Number);
```

Cluster created.

```
Alter cluster xyz
Add table department
 Where xyz.deptno=dept.deptno
```

You should not select the data from the cluster.

Always select the data from the table within the cluster.

7.38 CREATE SEQUENCE

CREATE SEQUENCE statement is used to create a **sequence**; which is a database object from which the multiple users may generate unique integers. You can use sequences to generate primary key values automatically.

Whenever a sequence number is generated, the sequence is incremented which is independent of the transaction commit or rolling back operation. Sequence numbers are generated independently of the tables, so the same sequence can be used for one or multiple tables. Once a sequence is created, you can access its values in SQL statements with the **CURRVAL** pseudocolumn (which returns the current value of the sequence) or the NEXTVAL pseudocolumn (which increments the sequence and returns the new value).

Syntax:

```
CREATE SEQUENCE
schema -sequence
INCREMENT BY
START WITH integer
MAXVALUE integer
NOMAXVALUE
MINVALUE integer
NOMINVALUE
CYCLE
NOCYCLE
CACHE integer
NOCACHE
ORDER
NOORDER
;
```


To create a sequence that increment **without bound**, for ascending sequences, omit the MAXVALUE parameter or specify NOMAXVALUE. For descending sequences, omit the MINVALUE parameter or specify the NOMINVALUE.

To create a sequence that **stops at a predefined limit**, for an ascending sequence, specify a value for the MAXVALUE parameter. For a descending sequence, specify a value for the MINVALUE parameter. Also, specify the NOCYCLE. Any attempt to generate a sequence number, once the sequence has reached its limit results in an error.

To create a sequence that **restarts after reaching a predefined limit**, specify values for both the MAXVALUE and MINVALUE parameters. Also, specify the CYCLE. If you do not specify MINVALUE, then it defaults to NOMINVALUE (that is the value 1).

7.38.1 Description of Sequence Parameters

INCREMENT BY Specify the interval between sequence numbers. This integer value can be any positive or negative integer, but it cannot be 0. This value can have 28 or fewer digits. If you omit this clause, the interval defaults to 1.

START WITH Specify the first sequence number to be generated. Use this clause to start an ascending sequence at a value greater than its minimum or to start a descending sequence at a value less than its maximum. This integer value can have 28 or fewer digits.

MAXVALUE Specify the maximum value that a sequence can generate. This integer value can have 28 or fewer digits.

NOMAXVALUE Specify NOMAXVALUE to indicate a maximum value of 1027 for an ascending sequence or -1 for a descending sequence. This is the default.

Note: This value is not necessarily the value to which an ascending cycling sequence cycles after reaching its maximum or minimum value.

MINVALUE Specify the minimum value of the sequence. This integer value can have 28 or fewer digits.

NOMINVALUE indicates a minimum value of 1 for an ascending sequence or -1026 for a descending sequence. This is the default.

CYCLE indicates that the sequence continues to generate values after reaching either its maximum or minimum value.

NOCYCL indicates that the sequence cannot generate more values after reaching its maximum or minimum value. This is the default.

CACHE Specify how many values of the sequence SQL pre-allocates and keeps in memory for faster access. This integer value can have 28 or fewer digits. The minimum value for this parameter is 2.

NOCACHE indicates that the values of the sequence are not Pre-allocated. If you omit both CACHE and NOCACHE, Oracle caches 20 sequence numbers by default.

ORDER guarantees that sequence numbers are generated in order of request.

NOORDER Specify NOORDER, if you do not want to guarantee sequence numbers in order of the request. This is the default.

Example:

The following statement creates the sequence my_seq in the schema x. This sequence can be used to provide employee ID numbers, when the rows are added to the employee table.

```
CREATE SEQUENCE my_seq
START WITH 2000
INCREMENT BY 1
NOCACHE
NOCYCLE;
```

The first reference to my_seq.nextval returns 2000. The second returns 2001. Each subsequent reference will return a value 1 greater than the previous reference.

7.39 TRANSACTION CONTROL LANGUAGE

There are three TCL Commands in SQL which are given as follows:

7.39.1 COMMIT

Use of COMMIT statement is to end your current transaction and make permanent changes performed in the transaction. A **transaction** is a sequence of SQL statements that SQL treats as a single unit. This statement also erases all savepoints in the transaction and releases the transaction's locks.

Example 1: The following statement inserts a row into the employee table and commits this change:

```
INSERT INTO employee VALUES (5, 'Raj');
COMMIT;
```

Example 2: The following statement commits the current transaction and associates a comment with it:

```
COMMIT
COMMENT 'In-doubt transaction Code 36, Call (415) 555-2637';
```


7.39.2 ROLLBACK

The use of ROLLBACK statement is to undo the work done in the current transaction.

To roll back your current transaction, no privileges are necessary.

Example 1: The following statement rolls back your entire current transaction:

```
ROLLBACK;
```

Example 2: The following statement rolls back your current transaction to savepoint x:

```
ROLLBACK TO SAVEPOINT x;
```

Note: It is recommended that you must explicitly end transactions in application programs using either a COMMIT or ROLLBACK statement. If you do not explicitly commit the transaction and the program terminates abnormally, then the Oracle rolls back the last uncommitted transaction.

7.39.3 SAVEPOINT Clause

Use of the SAVEPOINT statement is to identify a point in a transaction which you can roll back later. Specify the name of the savepoint to be created. Savepoint

names must be distinct within a given transaction. If you create a second savepoint with the same identifier as the earlier savepoint, then the earlier savepoint will be erased. After a savepoint has been created, you can continue processing, commit your work, roll back the entire transaction, or roll back to the savepoint.

Example:

To update x's and y's salary in the table employees, check that the total department salary does not exceed 314,000, then re-enter y's salary, and enter:

```
SAVEPOINT savepoint ;
UPDATE employees
SET salary = 6000
WHERE last_name = 'x';
SAVEPOINT t;
UPDATE employees
SET salary = 11000
WHERE last_name = 'y';
SAVEPOINT s;
SELECT SUM(salary) FROM employees;
ROLLBACK TO SAVEPOINT t;
UPDATE employees
SET salary = 12000
WHERE last_name = 'y';
COMMIT;
```


7.40 DATA CONTROL LANGUAGE

There are two DCL Commands in SQL which are given as follows:

7.40.1 GRANT Command

Use the GRANT statement to grant:

- System privileges to users and roles
- Roles to users and roles. Both privileges and roles are local, global, or external.
- Object privileges for a particular object to users, roles, and PUBLIC.

Syntax: GRANT privileges

```
ON object
TO User1, User2….Usern;
```

Example 1:

To grant all privileges on the table employee, issue the following statement:

```
GRANT ALL ON employee TO x
WITH GRANT OPTION;
```

Example 2: To grant SELECT and UPDATE privileges on the view emp_view to all the users, issue the following statement:

```
GRANT SELECT, UPDATE
ON emp_view TO PUBLIC;
```

All the users can subsequently query and update the view of employee details.

7.40.2 REVOKE Command

Use of the REVOKE statement to:

Revoke system privileges from users and roles.

Revoke roles from users and roles.

Revoke object privileges for a particular object from users and roles.

Syntax: REVOKE privileges

```
ON object
FROM User1, User2….Usern;
```

Example 1: The following statement revokes the DROP ANY TABLE system privilege from the users x and y:

```
REVOKE DROP ANY TABLE
FROM x,y;
```

Example 2: The following statement revokes the CREATE TABLESPACE system privilege from the R role:

```
REVOKE CREATE TABLESPACE
FROM R;
```

Installation of My SQL has been given in Appendix 4.

7.41 SOLVED PROBLEMS

Consider the following relations:

```
employee (ename, city, homeaddress)
worksfor (ename, institutename, salary, job)
institute (institutename , city)
supervises (ename, hodname)
```

Where, employee resides in some city and works for some institution. Institution is situated in some city and employees are supervised by the Principal. Perform the following queries:

1. Get names of employees working for institute ARSD:

```
SELECT ename
FROM worksfor
WHERE institutename = "ARSD";
```

2. Get the names and the cities of residence of employees working for DCAC:

```
SELECT e.ename, e.city
FROM worksfor w, employee e
WHERE w.ename = e.ename AND w.institutename = "DCAC";
```

3. Get the names and the cities of residence of employees working for institute HINDU and earning more than Rs. 400000:

```
SELECT e.ename, e.city
FROM worksfor w, employee e
WHERE w.ename = e.ename AND w.institutename
              ="HINDU" and w.salary = 400000;
```

4. Get the names of employees working in the same city where they live.

```
SELECT e.ename
FROM worksfor w, employee e, institute c
WHERE w.ename = e.ename AND w.institutename
= c.institutename AND e.city = c.city;
```

5. Get the names of employees, who are not working for ARSD:

```
SELECT ename
FROM employee
MINUS
SELECT ename
FROM worksfor
WHERE institutename = "ARSD";
```

6. Get the total and average salary paid by each institute.

```
SELECT institutename, sum(salary), avg(salary)
FROM worksfor
Group by institutename;
```

7. Get the names of employees, who earn more than every employee in ARSD.

```
SELECT ename
FROM worksfor
WHERE salary > (SELECT max(salary)
                FROM worksfor
                WHERE institutename ="ARSD");
```

8. Get the names of employees, who live in the same place and city as their Principals.

```
SELECT ename
 FROM employee e, supervises m, employee p
   WHERE e.ename = m.ename AND m.hodname = p.ename AND
 e.homeaddress = p.homeaddress AND e.city = p.city;
```

9. Get the institute with maximum no. of employees.

```
SELECT institutename
 FROM worksfor
 GROUP BY institutename
 HAVING count (ename) = (SELECT max(count(ename))
                  FROM worksfor
 GROUP BY institutename);
```

10. Get the institute with smallest total salary paid to the employee.

```
SELECT institutename
FROM worksfor
GROUP BY institutename
HAVING sum (salary) = (SELECT min (sum(salary))
                  FROM worksfor
GROUP BY institutename);
```

11. Get the institute whose employees earn more salary on an average than the average salary of institute ARSD.

```
SELECT institutename
FROM worksfor
GROUP BY institutename
HAVING avg(salary) > (SELECT avg(salary)
                  FROM worksfor
WHERE institutename="ARSD");
```

12. Get name, job, salary of all the employees whose job is of a manager or programmer and their salary is not equal to 50,000, 60,000, and 23,000.

```
SELECT ename, job, salary
FROM employee
WHERE job ="manager" OR job="programmer" AND salary NOT
                                IN (50000, 60000, 23000);
```

SUMMARY

This chapter deals with the basics of SQL. In this chapter, we have discussed different DDL, DML, DCL and TCL statements. We have also discussed different clauses used in SQL. Topics are represented in simpler manner with the appropriate examples.

PROBLEMS

1. Consider the following relations:

Hotel(hotelno, hotelname, city)

Room(roomno, hotelno, type, price)

Booking(hotelno, guestno, datefrom, dateto, roomno)

Guest(guestno, gname, gaddress)

Write appropriate queries in SQL for the following:

(*a*) Find the average price of a room.

(*b*) List the names and addresses of all the guests with bookings for a hotel in Varanasi, alphabetically ordered by name.

(*c*) Find the total income from all the rooms of the Hotels in Delhi.

(*d*) List the name(s) of guest(s) at the Taj Hotel who are paying highest price for a room.

2. Consider the following relations:

```
ITEM (I#, INAME)
ORDER (O#, ONAME, CNAME)
ORDERITEM (I#, O#)
```

Write the queries in SQL for the following:

(*i*) Get the details of items ordered by the customer "Sohan".

(*ii*) Get the Item Nos. of items which are included in all the orders.

3. The following relations are used to store data about students, courses and enrolment of students in courses and teachers of the courses.

(Primary key in each relation is marked by *)

```
Students (rollno*, sname, saddr)
Enrol (rollno, cno*, grade)
Teach (tno*, tname, cno*)
```

Write Create Table statements for defining the above tables.

Further write SQL statements to get rollno and names of the students who have got an 'A' grade in a course taught by "Rohan".

4. Consider the following relations:

```
Emp (EmpNo, Ename, DNo, Sal)
Dept (DeptNo, Dname, DMgrNo)
Proj (ProjNo, ProjName, DNum)
Works_on (EmpNo, PNo, Hrs)
```

Write appropriate queries in SQL for the following:

(*i*) Retrieve the names of all the employees in a department who work for more than 10 hrs per week on 'Proj X' project.

(*ii*) Find the names and addresses of all the employees who work for Research Department.

(*iii*) Find all the employees who do not work on any project.

(*iv*) Find all the employees who work on all the projects of Department No. 5.

(*v*) For every project, list Project No, Dept no and the Department Manager's name, number and salary.

5. Consider the relations:

```
GRADE (stud_id, subject_id, grade)
SUBJECT (subject_id, s_name, teacher)
```

(*i*) Write the SQL query for the following :

- List the student number, subject names and grades of the student whose id is 100.
- List the grades of all the students in the subjects taught by teacher "XYZ".

6. Consider the following relations in the context of a BPO:

```
Customer (C_code, C_name, C_phone)
Product (P_code, P_desc, P_price)
Call (Ca_id, C_code, P_code, Ca_result, Ca_date)
```

Write the appropriate SQL commands for the following:

(*i*) Find the name of customers to whom a call is not made.

(*ii*) Find the total number of calls made to the customer whose name is 'Raj'.

(*iii*) List C_name and P_desc of the calls made today.

(*iv*) Find the product details of those products whose price is above average.

7. A relational database consists of the following relations about students in an institute,

```
STUDENT (ROLLNO#, STUDENT_NAME, ADDRESS, AGE)
DEPARTMENT (DEPT_NAME, COURSE#, TEACHER)
TIMETABLE (COURSE#, SUBJECT_NAME, SEMESTER, TIME, ROOM#)
PERFORMANCE (ROLLNO#, COURSE#, GRADE)
```

Write a query in SQL to get the names of all the students who have secured 'A' grade in a course offered by the Computer Science Department in the First Semester.

8. Consider the following relational database:

```
EMPLOYEE (emp_name, street, city)
WORKS (emp_name, comp_name, salary)
COMPANY (comp_name, city)
MANAGES (emp_name, manager_name)
```

For each of the following queries, give the SQL expression:

(*i*) Find the names of all the employees who work for XYZ Bank Corporation.

(*ii*) Find the names of all the employees who live in the same city, where the company for which they work is located.

(*iii*) Find the names of those employees who earn more than every employee of the XYZ Bank Corporation.

(*iv*) Find the name, street, and city of those employees who work for the XYZ Bank Corporation and earn more than Rs. 2,50,000 per annum.

(*v*) Find the names of managers who work in a bank located in Delhi.

9. Define SQL. State the components of SQL.

10. What is the difference between JOIN and Cartesian product?

11. State the different Types of JOINs with example.

12. Define the term Integrity constraints. What are the different types of Integrity Constraints in SQL? Explain each of them with example.

13. What do you mean by embedded SQL? Give the name of pre-compilers.
14. What is the difference between DROP and DELETE commands in SQL?

REFERENCE AND RELATED LINKS

[1] Bipin C. Desai. West Group; (1990) "Introduction to Database Management System", 11th edition, West Group.
[2] Evan Bayross (2005) "Database concepts & systems", Shroff Publishers and Distributors Pvt. Ltd.
[3] Catherine M. Recardo (2004) "Database illuminated", Publisher, Jones & Bartlett Publishers.

HTML Colour Names

A list of the color names that are supported by all the major browsers is given below in a tabular form.

Note: If your pages are validated with an HTML or a CSS validator, W3C has listed 16 color names that you can use: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow.

The 16 Basic Colors:

Black	Gray	Silver	White
Yellow	Lime	Aqua	Fuchsia
Red	Green	Blue	Purple
Maroon	Olive	Navy	Teal

If you want to use other colors, you must specify their RGB or HEX value as follows in [4].

Color Name	Color HEX	Color
AliceBlue	#F0F8FF	
AntiqueWhite	#FAEBD7	
Aqua	#00FFFF	
Aquamarine	#7FFFD4	
Azure	#F0FFFF	
Beige	#F5F5DC	
Bisque	#FFE4C4	
Black	#000000	
BlanchedAlmond	#FFEBCD	
Blue	#0000FF	
BlueViolet	#8A2BE2	
Brown	#A52A2A	
BurlyWood	#DEB887	
CadetBlue	#5F9EA0	
Chartreuse	#7FFF00	
Chocolate	#D2691E	

Coral	#FF7F50	
CornflowerBlue	#6495ED	
Cornsilk	#FFF8DC	
Crimson	#DC143C	
Cyan	#00FFFF	
DarkBlue	#00008B	
DarkCyan	#008B8B	
DarkGoldenRed	#B8860B	
DarkGray	#A9A9A9	
DarkGrey	#A9A9A9	
DarkGreen	#006400	
DarkKhaki	#BDB76B	
DarkMagenta	#8B008B	
DarkOliveGreen	#556B2F	
Darkorange	#FF8C00	
Dark Orchid	#9932CC	
DarkRed	#8B0000	
DarkSalmon	#E9967A	
DarkSeaGreen	#8FBC8F	
DarkSlateBlue	#483D8B	
DarkSlateGray	#2F4F4F	
DarkSlateGrey	#2F4F4F	
DarkTurquoise	#00CED1	
DarkViolet	#9400D3	
DeepPink	#FF1493	
DeepSkyBlue	#00BFFF	
DimGray	#696969	
DimGrey	#696969	
DodgerBlue	#1E90FF	
FireBrick	#B22222	
FloralWhite	#FFFAF0	
ForestGreen	#228B22	
Fuchsia	#FF00FF	
Gainsboro	#DCDCDC	
GhostWhite	#F8F8FF	
Gold	#FFD700	
GoldenRed	#DAA520	
Gray	#808080	
Grey	#808080	
Green	#008000	
GreenYellow	#ADFF2F	
HoneyDew	#F0FFF0	
HotPink	#FF69B4	

IndianRed	#CD5C5C	
Indigo	#4B0082	
Ivory	#FFFFF0	
Khaki	#F0E68C	
Lavender	#E6E6FA	
LavenderBlush	#FFF0F5	
LawnGreen	#7CFC00	
LemonChiffon	#FFFACD	
LightBlue	#ADD8E6	
LightCoral	#F08080	
LightCyan	#E0FFFF	
LightGoldenRedYellow	#FAFAD2	
LightGray	#D3D3D3	
LightGrey	#D3D3D3	
LightGreen	#90EE90	
LightPink	#FFB6C1	
LightSalmon	#FFA07A	
LightSeaGreen	#20B2AA	
LightSkyBlue	#87CEFA	
LightSlateGray	#778899	
LightSlateGrey	#778899	
LightSteelBlue	#B0C4DE	
LightYellow	#FFFFE0	
Lime	#00FF00	
LimeGreen	#32CD32	
Linen	#FAF0E6	
Magenta	#FF00FF	
Maroon	#800000	
MediumAquaMarine	#66CDAA	
MediumBlue	#0000CD	
MediumOrchid	#BA55D3	
MediumPurple	#9370D8	
MediumSeaGreen	#3CB371	
MediumSlateBlue	#7B68EE	
MediumSpringGreen	#00FA9A	
MediumTurquoise	#48D1CC	
MediumVioletRed	#C71585	
MidnightBlue	#191970	
MintCream	#F5FFFA	
MistyRose	#FFE4E1	
Moccasin	#FFE4B5	
NavajoWhite	#FFDEAD	
Navy	#000080	
OldLace	#FDF5E6	

Olive	#808000	
OliveDrab	#6B8E23	
Orange	#FFA500	
OrangeRed	#FF4500	
Orchid	#DA70D6	
PaleGoldenRed	#EEE8AA	
PaleGreen	#98FB98	
PaleTurquoise	#AFEEEE	
PaleVioletRed	#D87093	
PapayaWhip	#FFEFD5	
PeachPuff	#FFDAB9	
Peru	#CD853F	
Pink	#FFC0CB	
Plum	#DDA0DD	
PowderBlue	#B0E0E6	
Purple	#800080	
Red	#FF0000	
RosyBrown	#BC8F8F	
RoyalBlue	#4169E1	
SaddleBrown	#8B4513	
Salmon	#FA8072	
SandyBrown	#F4A460	
SeaGreen	#2E8B57	
SeaShell	#FFF5EE	
Sienna	#A0522D	
Silver	#C0C0C0	
SkyBlue	#87CEEB	
SlateBlue	#6A5ACD	
SlateGray	#708090	
SlateGrey	#708090	
Snow	#FFFAFA	
SpringGreen	#00FF7F	
SteelBlue	#4682B4	
Tan	#D2B48C	
Teal	#008080	
Thistle	#D8BFD8	
Tomato	#FF6347	
Turquoise	#40E0D0	
Violet	#EE82EE	
Wheat	#F5DEB3	
White	#FFFFFF	
WhiteSmoke	#F5F5F5	
Yellow	#FFFF00	
YellowGreen	#9ACD32	

HTML Entity Code of ASCII Characters

There are 128 printable characters which can be represented by different 7-BIT ASCII codes. HTML uses standard 7-BIT ASCII, while transmitting data over the internet.

The following table lists all the 7-BIT ASCII codes and their equivalent HTML entity codes.

7-BIT ASCII Printable Characters

ASCII Character	HTML Entity Code	Description
		Space
!	!	exclamation mark
“	"	quotation mark
#	#	number sign
$	$	dollar sign
%	%	percent sign
&	&	ampersand
‘	'	apostrophe
(	(	left parenthesis
)	)	right parenthesis
*	*	asterisk
+	+	plus sign
,	,	comma
-	-	hyphen
.	.	period
/	/	slash
0	0	digit 0
1	1	digit 1
2	2	digit 2
3	3	digit 3
4	4	digit 4
5	5	digit 5
6	6	digit 6
7	7	digit 7
8	8	digit 8
9	9	digit 9
:	:	colon
;	;	semicolon
<	<	less-than
=	=	equals-to

>	>	greater-than
?	?	question mark
@	@	at sign
A	A	uppercase A
B	B	uppercase B
C	C	uppercase C
D	D	uppercase D
E	E	uppercase E
F	F	uppercase F
G	G	uppercase G
H	H	uppercase H
I	I	uppercase I
J	J	uppercase J
K	K	uppercase K
L	L	uppercase L
M	M	uppercase M
N	N	uppercase N
O	O	uppercase O
P	P	uppercase P
Q	Q	uppercase Q
R	R	uppercase R
S	S	uppercase S
T	T	uppercase T
U	U	uppercase U
V	V	uppercase V
W	W	uppercase W
X	X	uppercase X
Y	Y	uppercase Y
Z	Z	uppercase Z
[	[	left square bracket
\	\	backslash
]	]	right square bracket
^	^	caret
_	_	underscore
‘	`	grave accent
A	a	lowercase a
B	b	lowercase b
C	c	lowercase c
D	d	lowercase d

E	e	lowercase e
F	f	lowercase f
G	g	lowercase g
H	h	lowercase h
I	i	lowercase i
J	j	lowercase j
K	k	lowercase k
L	l	lowercase l
M	m	lowercase m
N	n	lowercase n
O	o	lowercase o
P	p	lowercase p
Q	q	lowercase q
R	r	lowercase r
S	s	lowercase s
T	t	lowercase t
U	u	lowercase u
V	v	lowercase v
W	w	lowercase w
X	x	lowercase x
Y	y	lowercase y
Z	z	lowercase z
{	{	left curly brace
|			vertical bar
}	}	right curly brace
~	~	tilde

HTML Symbol Entities

This entity reference includes Mathematical Symbols, Greek characters, various arrows, technical symbols and shapes.

Caution: Entity names are case sensitive.

Mathematical Symbols Supported by HTML

Character	Entity Number	Entity Name	Description
∀	∀	∀	for all
∂	∂	∂	Part
∃	∃	&exists;	Exists
Φ	∅	∅	Empty
∇	∇	∇	Nabla

∈	∈	∈	Isin
∉	∉	∉	Notin
∋	∋	∋	Ni
∏	∏	∏	Prod
∑	∑	∑	Sum
−	−	−	Minus
*	∗	∗	Lowest
√	√	√	square root
∝	∝	∝	proportional to
∞	∞	∞	Infinity
∠	∠	∠	Angle
∧	∧	∧	And
∨	∨	∨	Or
∩	∩	∩	Cap
∪	∪	∪	Cup
∫	∫	∫	Integral
∴	∴	∴	Therefore
~	∼	∼	similar to
≅	≅	≅	approximately equal
≈	≈	≈	almost equal
≠	≠	≠	not equal
≡	≡	≡	Equivalent
≤	≤	≤	less or equal
≥	≥	≥	greater or equal
⊂	⊂	⊂	subset of
⊃	⊃	⊃	superset of
⊄	⊄	⊄	not subset of
⊆	⊆	⊆	subset or equal
⊇	⊇	⊇	superset or equal
⊕	⊕	⊕	circled plus
⊗	⊗	⊗	circled times
⊥	⊥	⊥	Perpendicular
.	⋅	⋅	dot operator

Other Entities Supported by HTML

Character	Entity Number	Entity Name	Description
Œ	Œ	Œ	capital ligature OE
œ	œ	œ	small ligature oe
Š	Š	Š	capital S with caron
š	š	š	small S with caron
Ÿ	Ÿ	Ÿ	capital Y with diaries
f	ƒ	ƒ	f with hook

ˆ	ˆ	ˆ	modifier letter circumflex accent
˜	˜	˜	small tilde
–	–	–	En dash
—	—	—	Em dash
‘	‘	‘	left single quotation mark
’	’	’	right single quotation mark
‚	‚	‚	single low-9 quotation mark
“	“	“	left double quotation mark
“	”	”	right double quotation mark
„	„	„	double low-9 quotation mark
†	†	†	Dagger
‡	‡	‡	double dagger
•	•	•	Bullet
…	…	…	horizontal ellipsis
‰	‰	‰	per mille
'	′	′	Minutes
"	″	″	Seconds
‹	‹	‹	single left angle quotation
›	›	›	single right angle quotation
‾	‾	‾	Overline
€	€	€	Euro
™	™	™	Trademark
←	←	←	left arrow
↑	↑	↑	Up arrow
→	→	→	right arrow
↓	↓	↓	down arrow
↔	↔	↔	left right arrow
↵	↵	↵	carriage return arrow
⌈	⌈	⌈	left ceiling
⌉	⌉	⌉	right ceiling
⌊	⌊	⌊	left floor
⌋	⌋	⌋	right floor
≃	◊	◊	Lozenge
♠	♠	♠	Spade
♣	♣	♣	Club
♥	♥	♥	Heart
♦	♦	♦	Diamond

The Most Common Character Entities

Result	Description	Entity Name	Entity Number
	non-breaking space		
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
“	quotation mark	"	"
‘	apostrophe	' (does not work in IE)	'

Some Other Commonly Used Character Entities:

Result	Description	Entity Name	Entity Number
¢	Cent	¢	¢
£	Pound	£	£
¥	Yen	¥	¥
	Euro	€	€
§	Section	§	§
©	copyright	©	©
®	registered trademark	®	®
×	multiplication	×	×
÷	division	÷	÷

REFERENCES

[1] http://www.w3.org/
[2] http://www.htmlcodetutorial.com/
[3] http://www.echoecho.com/html.htm
[4] http://www.w3schools.com/

Open Database Connectivity

In order to use any GUI for retrieving database like MS-ACCESS to connect to a database server over a network, you need something called middleware. We are going to use Open Database Connectivity (ODBC) middleware to create a link to data on a different computer, using a different operating system and running a different DBMS.

MICROSOFT ACCESS

One very important thing about MS-ACCESS is that it provides powerful and easy-to-use tools for organizing data, creating forms, producing reports, and automating tasks. Demerit of The MS-ACCESS is that the database engine itself is not designed to support high transaction volumes or a large number of simultaneous users. For example, one simply can not run Railway reservation system on top of an ACCESS database.

One solution seems that, ACCESS can play the role of the "client", while connecting to an industrial strength client/server database. Its significance is that you can continue to work within ACCESS without storing store multiple copies of your organization's data in multiple independent desktop database systems. Instead, many desktop systems can link to a single source of data that is stored and administered centrally.

Microsoft Access is a Relational Database Management System. Access provides a database engine and a Graphical User Interface (GUI) for the data definition and manipulation. For the PC (Personal Computer) platform, Access is very popular implementation of the relational data model. We can implement various applications through Access from personal applications (maintaining an inventory of your personal audio and video collection) to small business applications, such as maintaining business specific customer information. Users can develop forms and reports for input/output operation against the database through the use of wizards. Wizards are interactive programs that guide the

user through a series of questions in a dialog mode. Using the Object Linking and Embedding (OLE) technology, a user can include documents created in another component on a report or form within Access. In Access, you can create professional reports and save your data in HTML format for viewing in a browser.

To understand ACCESS, Let us create a database having two tables with the specified fields, to computerize a library system of an institution:

Table 1: For each book in the library:

Accession number of the book
Book Title
Name of the author
Name of the department
Purchasing date of the book
Price of the book.

Table 2: For each book issued:

Accession number of book
Borrower

And generate a report listing the books belonging to "Mathematics" Department for books which are issued.

Solution is given as follows:

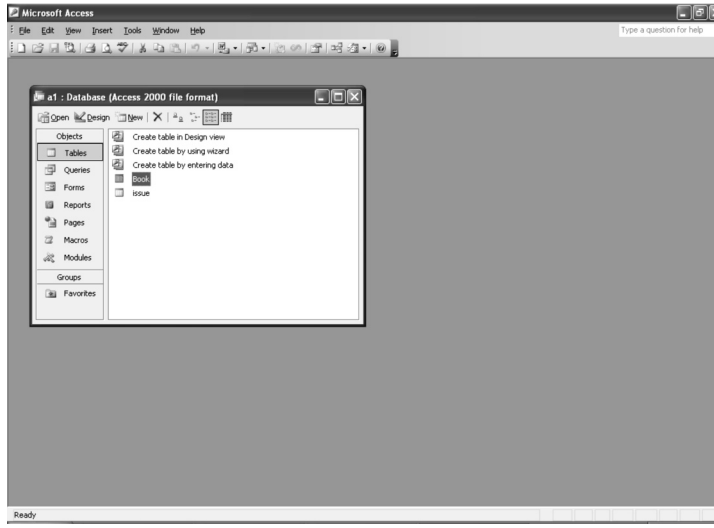
Step 1: Create two tables named as Book and issue.

The steps are as follows:

In the **Database** window, click the **Table** object (or choose tables from the **View** menu).

Select the New button. The **New Table Dialog box** will be displayed.

Select the **New Table** button. A blank Table Window in **Design View** will be opened by MS-Access.



The primary key of a table consists of one or more Attributes that uniquely identify each record stored in the table. The primary key value is always different for each record.

Before saving the table, we need to assign a primary key. In a table, we can not define more than one primary key, but we can have a multiple-Attribute primary key (one primary key defined on several Attributes).

Steps to set primary key as follows:

1. Click on the name of the attribute that will be the primary key. To create a multiple attribute primary key, hold down the **control key** and click the attribute selector to the left of each attribute that you want to include

2. Click on the **Primary key button** on the toolbar.

3. On the left of the Attribute name will appear a key indicating that this Attribute is the primary key of the table.

Note: Remember that the Attribute or group of Attributes forming the primary key of a table cannot contain null values and neither have two rows in the table with the same value in the primary key Attributes.

Step 2: Enter the values in both the tables.

You should start entering your data into the table by typing in the cells. To delete typing mistakes, use the **BACKSPACE** key. To delete changes in the current Attribute, press the **ESC** key.

Note: Only the latest changes will be deleted.

To move between the cells or records, use **arrow keys** or the **Tab key**.

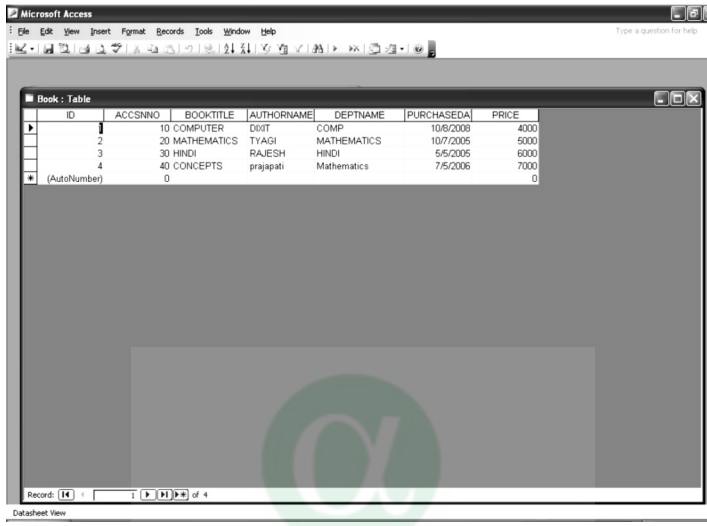
To add a new record, click **New Record** in the toolbar.

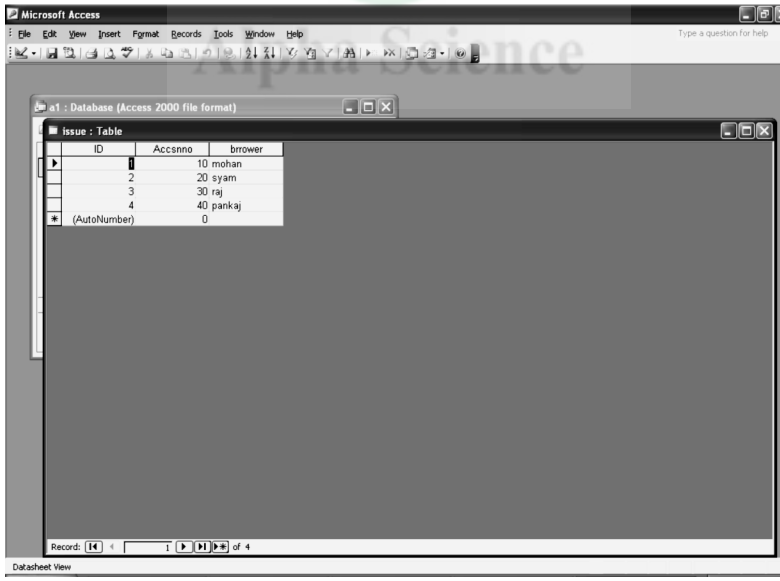
Note: You cannot add records, if your cursor is on the blank record.

To edit data in an Attribute, click in that Attribute and type in the new data.

To replace the entire value in the Attribute, move the pointer to the left corner of the Attribute until it changes to the plus sign, and click. Type in the new data; **Microsoft Access** saves your data when you move to another record.

We have entered data into two tables named book and issue as follows:





Step 3: Establish the relationship between both the tables named Book and issue. Common attribute is Accession number.

The following are the steps for setting of the relationship:

Accssnno attribute of Book table should be linked to accsnno attribute of issue table in the following ways:

1. Adding of tables:

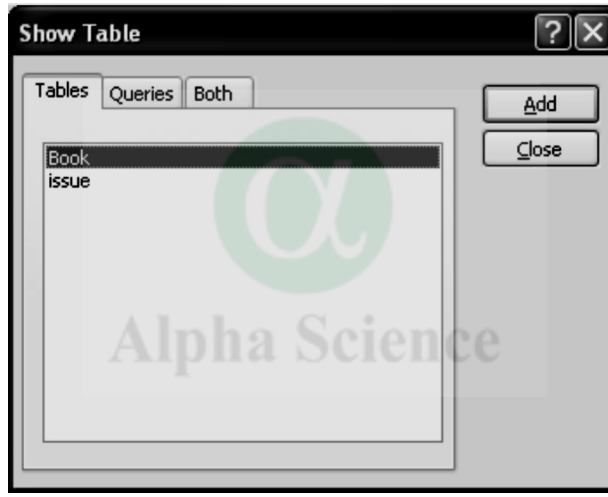
In a **Database** window, drop down **Tools** menu, and select **Relationships...** option.

The **Show table** dialogue box will appear:

Click on one of the tables required in the relationship and click on the **Add** button; or double-click the name of the table.

Repeat the previous step to add the second table, and so on.

Finally click on the **Close** button to finish adding tables.



2. Now the **Relationships** window will appear with the tables added before.

The **Edit relationships** dialogue box appears next:

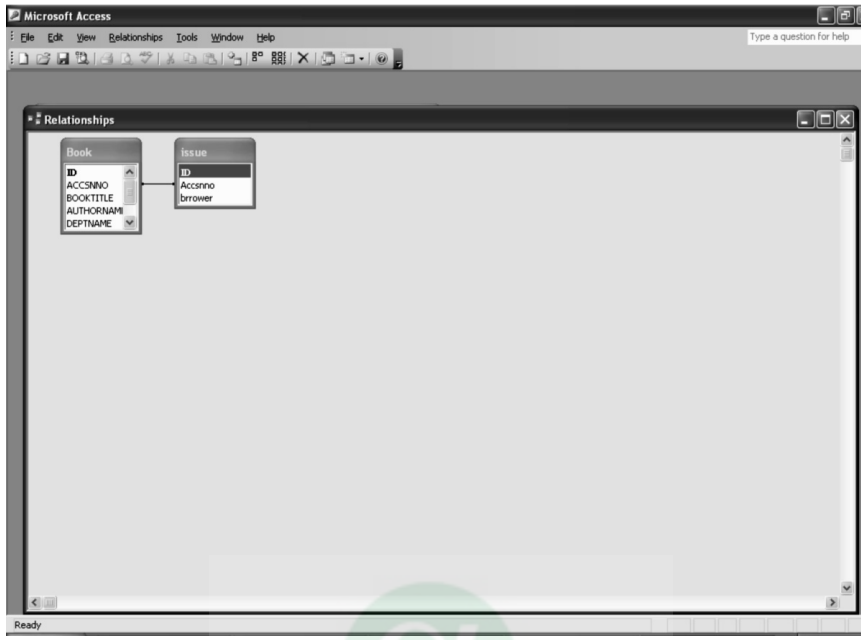
At the top should be the names of the related tables (book and issues) and below this, the names of the related Attributes.

Observe at the bottom, the **Relationship type** assigned depends on the characteristics of the related Attributes (in our case **One-to-Many**)

Activate the **Enforce Referential integrity with Cascade, Delete and Update related records** by clicking on it.

To terminate, click on the **Create** button.

The relationship is created and will appear in the **Relationships** window.



Step 4: Create the query named query1. Give the criteria deptname=”mathematics”.

Creating a query

To create a query without a query wizard, following steps are taken:

In the **Database window**, click the **Query** object.

Select the **New** Button.

The **New Query dialog box** will be displayed.

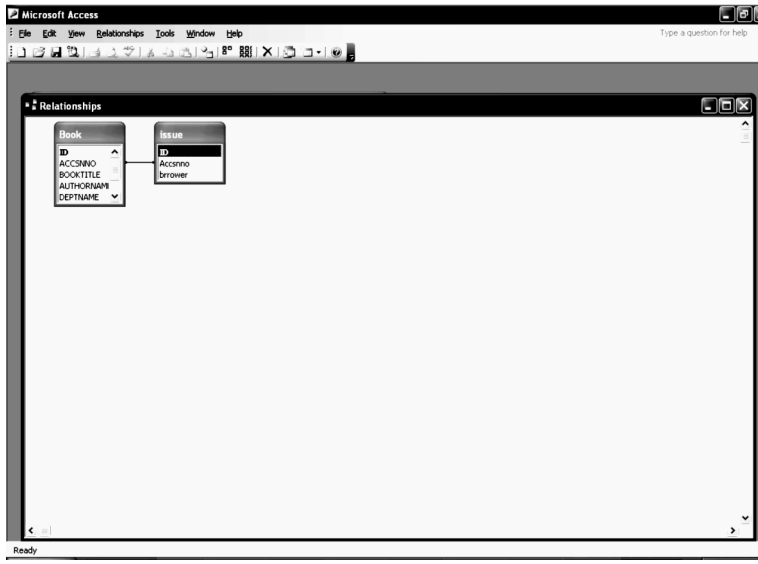
Select the **New Query** button.

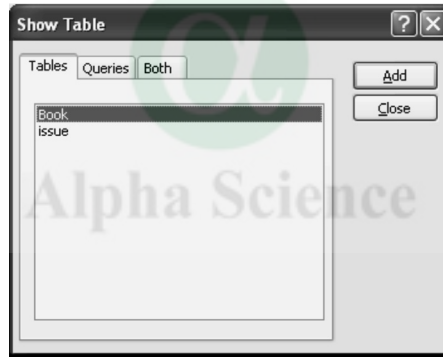
Select **Query window** will be opened and the **add table dialog box** will be displayed. The Add Table dialog box displays the tables and queries in your database.

Select the table you want to add to your query.

An attribute list for each table you select, will be displayed in the **Query window**.

Select the **Close** button.





If we look at the screen below, we have a **table's area**. In this area, we put the tables that contain the data we want to see in the result of the query, and in the part below named the **QBE grid, where** we define the query.

Every column in the **QBE grid** corresponds to an Attribute.

Every row has a function:

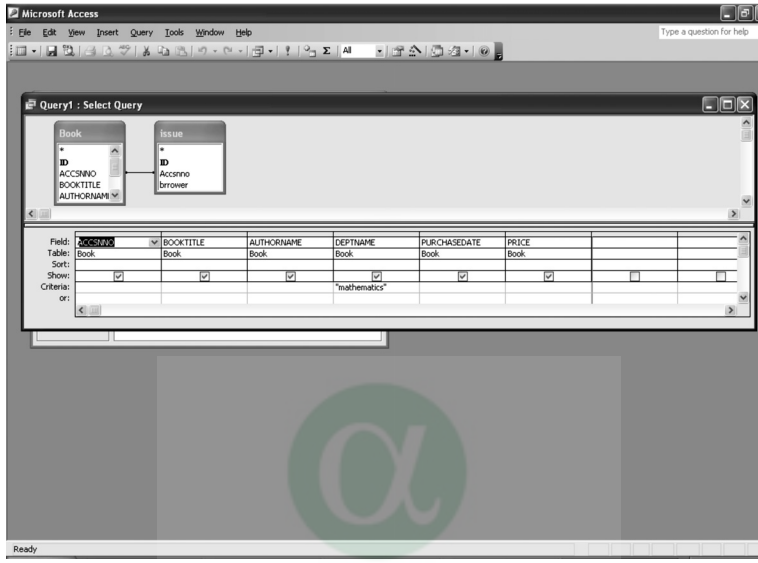
Attribute: We place the Attribute which will be the Attribute to visualize. It could be an Attribute from the table or a calculated Attribute.

Table: Show name of the table we want to extract the Attribute from. This will be useful, when we are defining queries based on various tables.

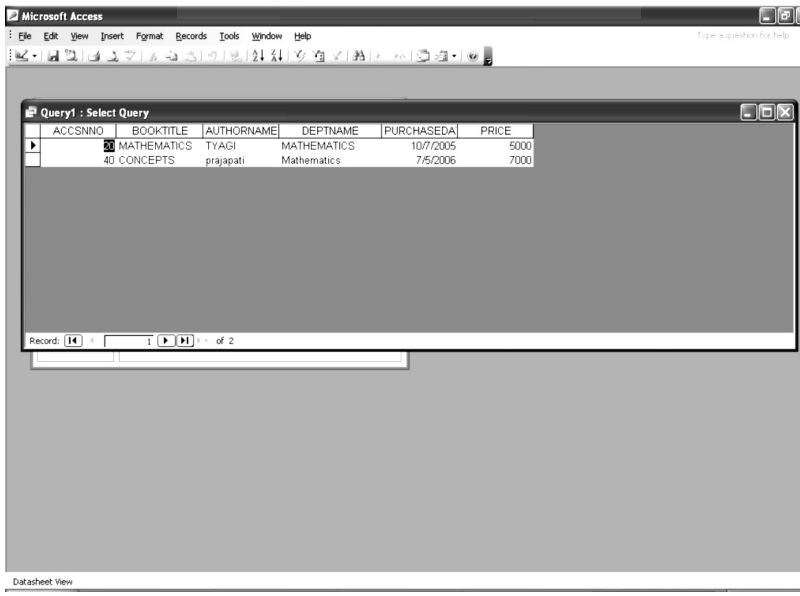
Sort: To arrange the resulting rows in a determinate order.

Show: if not marked, the column will not appear in the result. It is usually unmarked, when we want to use the Attribute to define the query but don't want the Attribute to appear in the result.

Criteria: You have to specify the **lookup criteria while applying the condition, for example BookTitle=”mathematics”.** A lookup **criteria** is a condition that records need to comply with, to appear in the result of the query. It can be made up from one or various conditions, joined by **AND** and **OR** operators.



Step 5: The query will return the following result:



Step 6: Create the form, through which you can enter the data in both the tables.

In the **Database window**, click the **Form** object.

Select the **New** button.

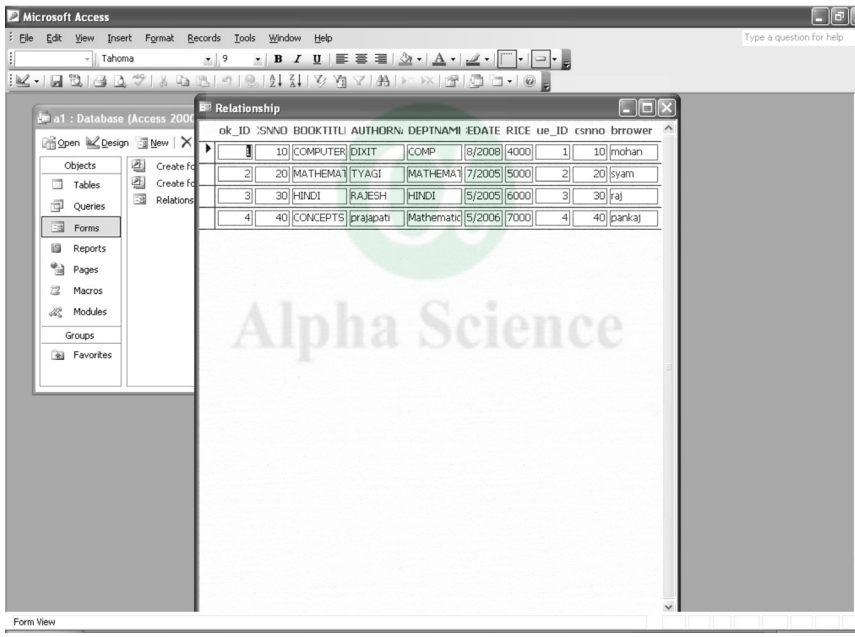
The **New Form dialog box** will be displayed.

From the Select a **Table/Query** list box, select the table or query that contains the data you want to display on the form (If the form does not contain the data, please don't make selection in this list.).

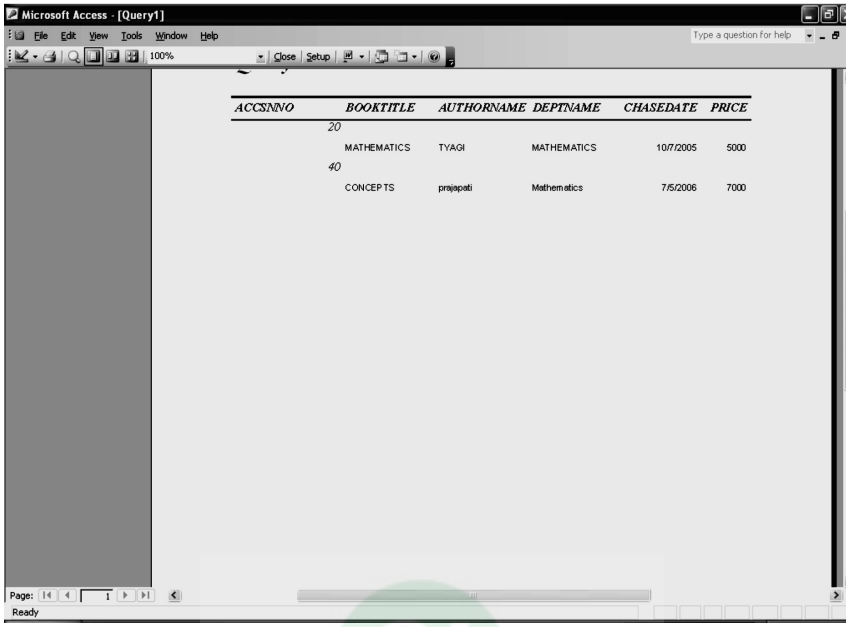
Select the **Blank Form** button.

The form in **Design view** will be displayed.

The output is as follows:



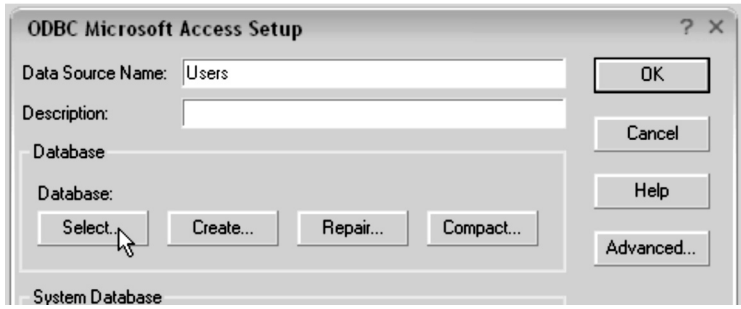
Step 7: Using Query1, create a report using wizard, listing the books belonging to "Mathematics" Department for books which are issued.



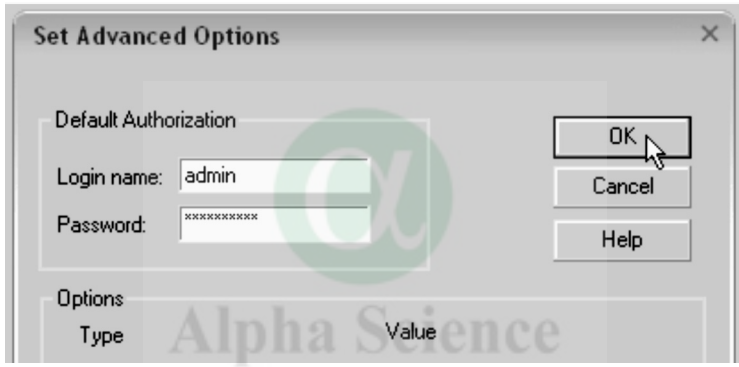
How to set up an ODBC DSN in Windows?

Start by opening the **Windows Control Panel**, click **Administrative Tools** and then click **Data Sources (ODBC).** This should open up the **ODBC Data Sources Administrator:**

Click the **System** tab at the top, then click **Add** to create a **System DSN**. Select "Microsoft Access Driver (*.mdb)" from the list, then click **Finish** to open up the **ODBC Microsoft Access Setup** screen:



Enter "Users" as the **Data Source Name,** then click **Select** and locate your database. If you have password to protect your Access database, click the **advanced** button, then enter your **Login Name** and **Password**.



Now your System DSN is ready to access in your ASP program. Click **OK** until you return to the Windows Control Panel.

How to access database through ASP?

Now we are showing, how one can access database through ASP in the subsequent sections step by step from the start.

Once you know how to connect to a database, then you would be able to know how to add, update and delete records.

How we can connect to our database?

We have already set up the ODBC DSN in the previous section.

```
<%
 Set oConn = Server.CreateObject("ADODB.Connection")
 oConn.Open "DSN=Users"
%>
```

However, the Access ODBC driver is known to cause problems when multiple users are connected.

If you know the physical path to your database on the server, it's better to bypass ODBC completely and connect using the **Jet OLEDB** Provider:

```
<%
 sPathToDB = "C:\Inetpub\a1.mdb"
 Set oConn = Server.CreateObject("ADODB.Recordset")
  oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" &
sPathToDB & ";"
%>
```

Now, our database connection is open and we can go about modifying or viewing its contents. Once completed, we should ALWAYS close and destroy the database connection object:

```
<%
 oConn.Close
 Set oConn = Nothing
%>
```

How to add records to databases?

Now we'll go one step further and add a record to our database using ASP.

For this example, we include **adovbs.inc** so that we can use the **adLockOptimistic** constant.We start by opening a connection to the database called **oConn**, using the DSN "Users":

```
<!—#include  virtual="/adovbs.inc"—>
<%
 Set oConn = Server.CreateObject("ADODB.Connection")
 oConn.Open "DSN=Users"
%>
```

Now, we need an SQL query to open our **recordset** with. Using an SQL Select query "where 0=1", we deliberately create an empty recordset, since we aren't interested in any existing records:

```
<%
 sSQL = "select * from Users where 0=1"
%>
```

The next step is to create our Recordset (rs). We tell it to use the connection we opened (oConn) and our SQL query (sSQL). The LockType property

"adLockOptimistic" locks the table while executing the update statement. If you don't use adovbs.inc, adLockOptimistic can be replaced with the number 3.

```
<%
 Set rs = Server.CreateObject("ADODB.Recordset")
 rs.ActiveConnection = oConn
 rs.LockType = adLockOptimistic
 rs.Source = sSQL
 rs.Open
%>
```

Now our recordset is open, we use the AddNew method to indicate that we want to add to the database. Then, we enter the information to pass into each field in the table. Finally, we use the Update method to add the record.

```
<%
 rs.AddNew
 rs("Name of the author") = Request.Form("Name of the author")
 rs("Book Title") = Request.Form("Book Title")
 rs.Update
%>
```


Now its completed, free up the server resources by closing and destroying the objects:

```
<%
 rs.Close
 Set rs = Nothing
 oConn.Close
 Set oConn = Nothing
%>
```

How to update database records?

Once again, we'll start by including adovbs.inc and use the Users DSN to open a connection to our database:

```
<!—#include  virtual="/adovbs.inc"—>
<%
 Set oConn = Server.CreateObject("ADODB.Connection")
 oConn.Open "DSN=Users"
%>
```

Lets assume that an User ID has been passed through the Querystring from another page - we'll use this to select the correct record from the database:

```
<%
 iUserID = Request.QueryString("UserID")
 sSQL = "select * from Users where ID=" & iUserID
%>
```

Now, we open our Recordset using our SQL query (sSQL) and use the LockType "adLockPessimistic" to lock the table while calling the update method:

```
<%
 Set rs = Server.CreateObject("ADODB.Recordset")
 rs.ActiveConnection = oConn
 rs.LockType = adLockOptimistic
 rs.Source = sSQL
 rs.Open
%>
```

Updating a database is almost identical to adding-the only difference is we want to use an existing record, so there's no need to use rs.AddNew.

```
<%
 rs("Name of the author")
 rs("Book Title")
 rs.Update
%>
```

All that's left to do now is clean up our objects. We might also want to redirect to another page after updating the database. While redirecting to another page, always make sure this is done AFTER destroying the Recordset and Connection.

```
<%
 rs.Close
 Set rs = Nothing
 oConn.Close
 Set oConn = Nothing
 response.redirect "user.asp?UserID=" & iUserID
%>
```

How to use ASP to delete records from the database?

As usual, start by opening a connection to the database using the Users DSN.

```
<%
 Set oConn = Server.CreateObject("ADODB.Connection")
 oConn.Open "DSN=Users"
%>
```

The SQL query to delete a record is very simple and once again, we'll assume that the User ID has been passed from the previous ASP page:

```
<%
 iUserID = Request.QueryString("UserID")
 sSQL = "delete * from Users where ID=" & iUserID
%>
```

Now, all we need to do is to execute the query to delete the record. Notice that we don't have to close the Recordset this time - this is done automatically after the record is deleted.

```
<%
 Set rs = Server.CreateObject("ADODB.Recordset")
 rs.ActiveConnection = oConn
 rs.Source = sSQL
 rs.Open
%>
```

We still need to destroy the objects we have used:

```
<%
 Set rs = Nothing
 oConn.Close
 Set oConn = Nothing
%>
```

Electronic–Commerce (E-Commerce)

3

APPENDIX

E-Commerce is an acronym for Electronic Commerce; which means to do business on Internet.

Commerce is defined as the exchange or buying and selling of commodities involving transportation from place to place, for money. Thus, commerce may involve buying something at a grocery store, producing a product in a factory and selling it to a wholeseller, etc.

Generally, in Commercial activities the following things are involved:

Producers – Are the people who create the products and services that sellers offer to buyers. The producer sells the products produced to the wholesellers, retailers or directly to the consumer.

- Wholesellers or distributors who sell to retailers and other businesses.
- Buyers, Sellers, retailers (who sell directly to consumers)

Commerce is a very simple concept like a person making and selling toys on a street corner or as complex as company delivering electronic goods e.g., engine to different companies. Commerce at its simplest level involves buyers, sellers and producers.

Basic elements of a commercial activity: The basic elements of a commercial activity are the following:

You must have a **product or service** to offer.

You must have a **place** from where you sell your products.

You must do proper marketing to sell your products successfully.

You should have a system of billing and **collections.**

You may require customer service and **technical support** departments to give satisfactory service to the customers. You can not do a successful business, unless you have satisfied the customers.

In the same manner, E-Commerce (or Electronic Commerce) is similar to commerce over electronic systems such as the Internet and other computer networks. It consists of distributing, buying, selling, marketing and servicing of products. It involves the following things:

- Electronic Funds Transfer
- Supply Chain Management
- E-Marketing
- Online Marketing
- Online Transaction Processing
- Electronic Data Interchange (EDI)
- Automated Inventory Management Systems
- Automated Data Collection Systems

E-commerce uses electronic communications technology of the World Wide Web, at some point in the transaction's lifecycle, although it frequently depends on the following:

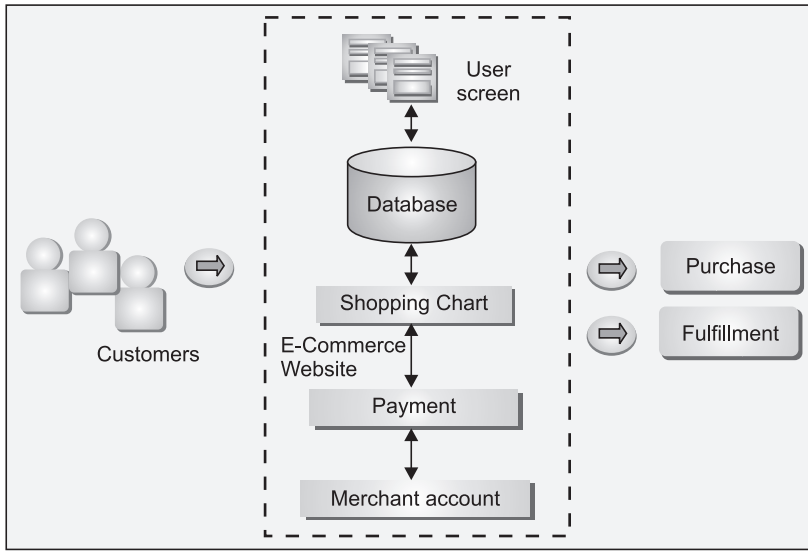
Computer technologies: Databases, Computer Networks, Data Communication, Internet (E-mail), etc.

Non-computer technologies: Transportation for physical goods sold via e-commerce.

In fact, E-Commerce encompasses a very wide range of business activities and processes, from e-banking to offshore manufacturing to e-logistics. An endless variety of products and services can be sold over the Internet.

Elements to conduct E-commerce

- A product
- A place to sell the product- in e-commerce, a Web site displays the products invite and acts as the place
- A way to invite people to come to your Web site
- A way to accept orders - normally an on-line form.
- A way to accept money - normally a merchant account handling **Credit Card** payments. This piece requires a secure ordering page and a connection to a bank.
- You may also use more traditional billing techniques, either online or through the mail.
- A fulfillment facility to ship the products to customers (often outsourceable). In the case of software and information, fulfillment can occur over the Web, through a file download mechanism.
- A way to accept returns.
- A way to handle warranty claims, if necessary.
- A way to provide customer service (often through email, on-line forms, on-line knowledge bases and FAQs, etc.)



Types of E-commerce

Different types of E-Commerce activities can be categorized as:

- B2B - Business to Business
- B2C - Business to Consumer
- C2B - Consumer to Business
- B2E - Business to Employee
- C2C - Consumer to Consumer

B2B (Business to Business) Commerce

The purchasing, selling, and exchanging of the business takes place between two business entities by matching their sale and purchase requirements online. It involves electronic data interchange (EDI) between two business houses (where as today the most popular connection between the two is via internet). The two businesses pass information electronically to each other. B2B e-commerce currently makes up about 94% of all e-commerce transactions. In the B2B environment, E-Commerce can be used in the process of the following:

- Procurement
- Order fulfillment
- Managing trading-partner relationships.

B2B E-commerce is being used as following:

- To Attract, develop, retain, and cultivate relationships with customers,
- Streamline the supply chain, manufacturing, and procurement processes,

- Automate corporate processes to deliver the right products and services to customers quickly and cost-effectively.

B2C - Business to Consumer

In Business to Consumer e-commerce, the consumer accesses the system of the supplier through the Internet. The supplier may be an existing retail outlet in a high street store. This type of business has been successful in using e-commerce to deliver services to customers.

For example, a student wishes to purchase a good book on a subject. The user accesses the Internet site,

http://www.booksonline.com and follows the links to read a report on the recommended books on the subject. After reading the reports, the user follows the links to place an order along with the delivery and payment details directly into the merchant's inventory system. Then, the book is dispatched from the suppliers warehouse.

With the help of sophisticated shopping cart software, business sells to general public through online catalogs. Yahoo e-commerce is a perfect example of Business-to-Consumer.

C2B - Consumer to Business

In Consumer to Business, the consumer requests a specific service from the business. For example, you may be interested in going to Mumbai from Delhi in the last week of June and willing to pay only Rs. 4000 for a return flight. You place a request with in a web based C2B facility. The facility www.makeMytrip.com receives your request, and being a lean period, it offers you a return fare for Rs. 4000 on the available airlines.

B2E - Business to Employee

Business to Employee e-commerce is a web site developed to provide imformation to the employees of an organization. This form of e-commerce is more commonly known as an 'Intranet' and usually accessed through the organizations network.

Consumer to Consumer E-commerce

Web sites are usually a form of an auction site. One consumer lists some items for sale with a commercial auction site, while other consumers access the site and place bids on these items. Then, the site provides a connection between the seller and buyer to complete the transaction. The site provider usually charges a transaction fee for the transaction. There are many sites offering free classifieds,

auctions, and forums where individuals can buy and sell with the help of online payment systems where people can send and receive money online with ease.

The Advantages of E-commerce

The following are the main advantages of e-commerce:

- Economic efficiency resulting from the reduction in communication costs.
- Low-cost technological infrastructure.
- Speedier and more economic electronic transactions with suppliers.
- Lower global information sharing and advertising costs.
- Cheaper customer service alternatives.
- Lower transaction costs.
- Larger purchases per transaction - Amazon offers a feature that no normal store offers. When you read the description of a book, you also can see "what other people who ordered this book also purchased." That is, you can see the related books that people are actually buying. Because of the features like these, it is common for people to buy more books that they might buy at a normal bookstore.
- Integration into the business cycle - A Web site which is well-integrated into the business cycle can offer customers more information than previously available. For example, if Dell tracks each computer through the manufacturing and shipping Integration into the business cycle - A Web site that is well-integrated into the business cycle can offer customers more information than previously available. For example, if Dell tracks each computer through the manufacturing and shipping process, customers can see exactly the position location of their order. This is what FedEx did when they introduced on-line package tracking - FedEx made more information available to the customer.

REFERENCES

http://www.ecommercesolutionspoint.com/

Installation Procedure MySQL

XAMPP for Windows

The XAMPP stands for Apache, MySQL, PHP and PhpMyAdmin. The X stands for any operating system. The new version of XAMPP 1.7.7 (http://www.apachefriends.org/en/xampp-windows.html), includes following packages:

- Apache 2.2.21
- MySQL 5.5.16
- PHP 5.3.8
- phpMyAdmin 3.4.5
- FileZilla FTP Server 0.9.39
- Tomcat 7.0.21 (with mod_proxy_ajp as connector)

XAMPP for Windows are available in three types of installations:

Installer Probably the most comfortable way to install XAMPP.

ZIP For purists: XAMPP as ordinary ZIP archive.

7zip For purists with low bandwidth: XAMPP as 7zip archive.

There are two types of installation methods, Kindly download the appropriate file from the http://www.apachefriends.org/en/xampp-windows.html and follow the steps accordingly. Below are the formal steps to install the XAMPP

Method A: Installation with the Installer

This is one of the simple way to install the XAMPP environment, just click on the installer file and follow the steps. **Microsoft Windows Vista users may face some problem during the installation** because of missing or insufficient write permissions in "C:\Program Files". Hence, they should use alternate folder for XAMPP ("C:\xampp" or "C:\myfolder\xampp"). Once the installation is complete, XAMPP will appear at Start | Programs | Apache Friends | XAMPP. One can also use the XAMPP Control Panel to start/stop the servers and also install/uninstall services related to XAMPP.

Method B: "Installation" without the Installer (ZIP or 7zip)

Unzip the respective ZIP or 7zip into any folder of your choice. XAMPP is extracted to the subdirectory "xampp" under the selected target directory. Now

start the file "setup_xampp.bat", to adjust the XAMPP configuration to your system. One should avoid the use of using a root directory "C:\" as target. Other instruction will be same as the installer version, and use the "XAMPP Control Panel" for other required tasks.

The control panel for the XAMPP looks like as shown in Fig. 1. This control panel can be obtained from "Start/Programs/ApacheFriends/XAMPP/XAMPP Control Panel".

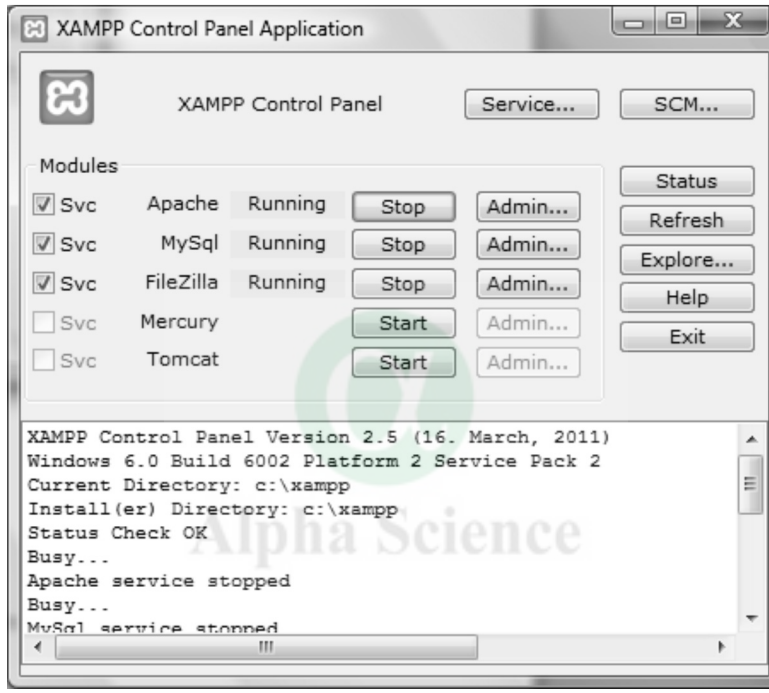


Figure 1. XAMPP Control Panel

After starting from the XAMPP control panel, the environment can be tested as starting of Apache (and MySQL), and then open the URL http://localhost/ or http://127.0.0.1/ in any web browser and test the XAMPP tools as shown in Figure 2.

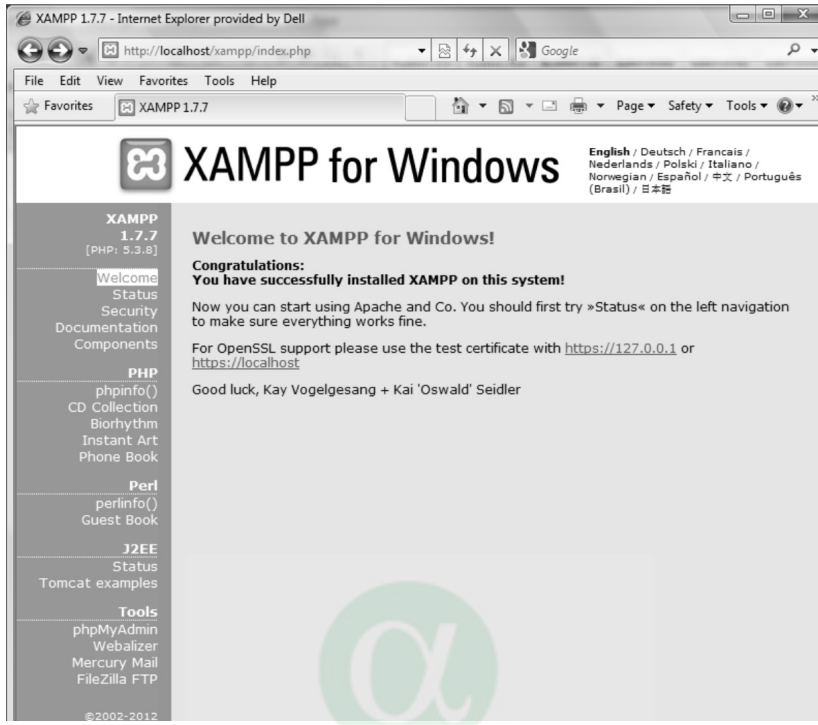


Figure 2. Home Screen on Web Browser

Now user can click on any of the link provided in the browser to test particular environment. For example, to test the MySQL environment, click on phpMyAdmin (GUI environment to access MySQL) following screen will appear as Figure 3.

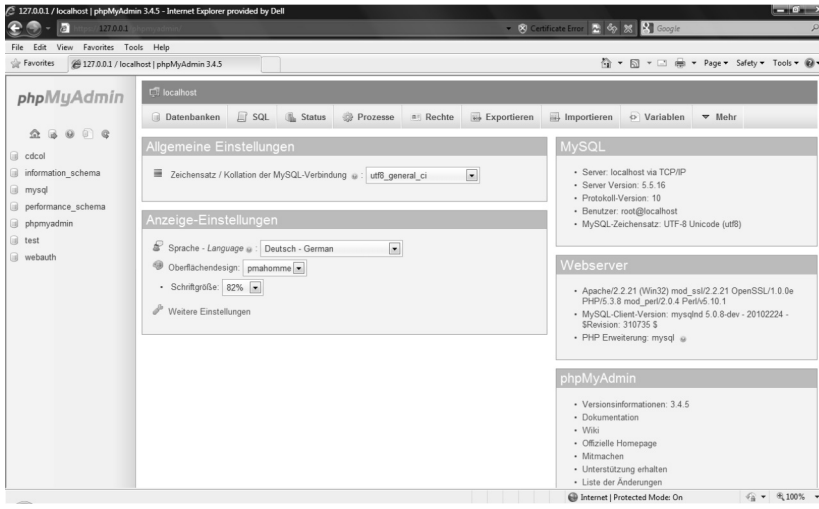


Figure 3. phpMyAdmin environment for MySQL

Now select the language as English and Theme/Style as Original and screen will appear as Figure 4.

Figure 4. Environment set in English Language with Original Theme

Now click on the test in the left frame of the screen and test database is ready for use. The user can create table (for example Student with two columns) into test database as shown in Figure 5 and then click on GO as in Figure 6. After filling the column names, click on save and browser will be displayed as shown in Figure 7.

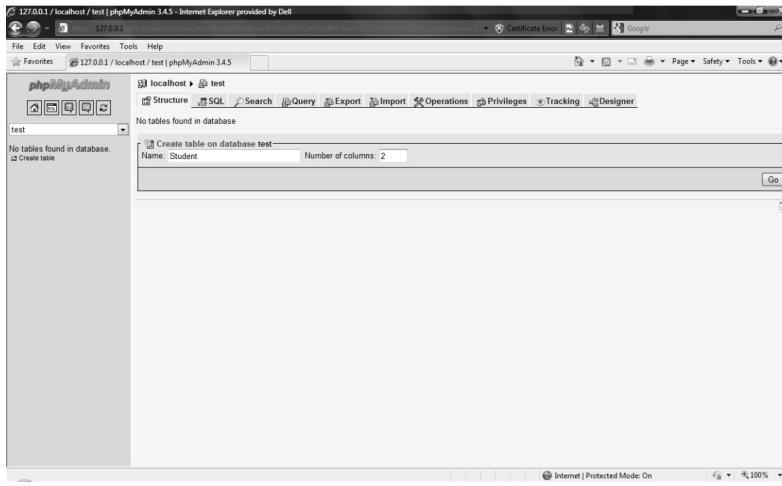


Figure 5. Creation of a Table in test Database

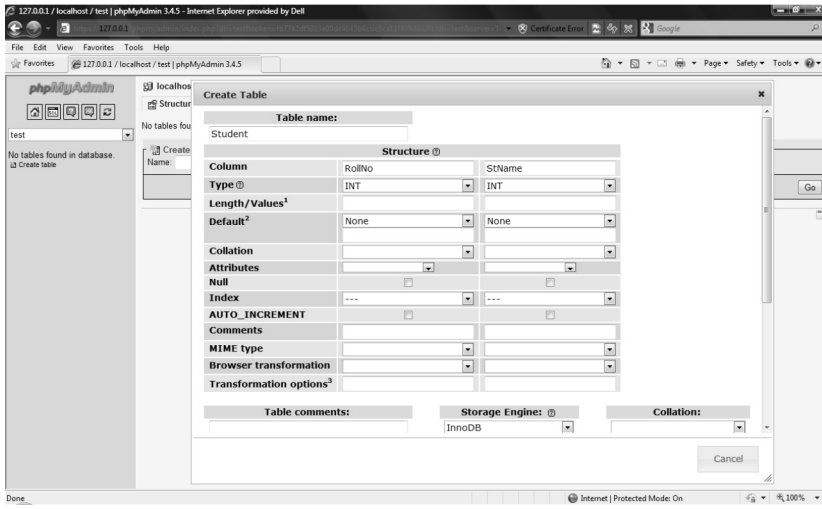


Figure 6. Define the attributes for the table

Now the table is ready to accept SQL queries to access and manipulate the records of the table.

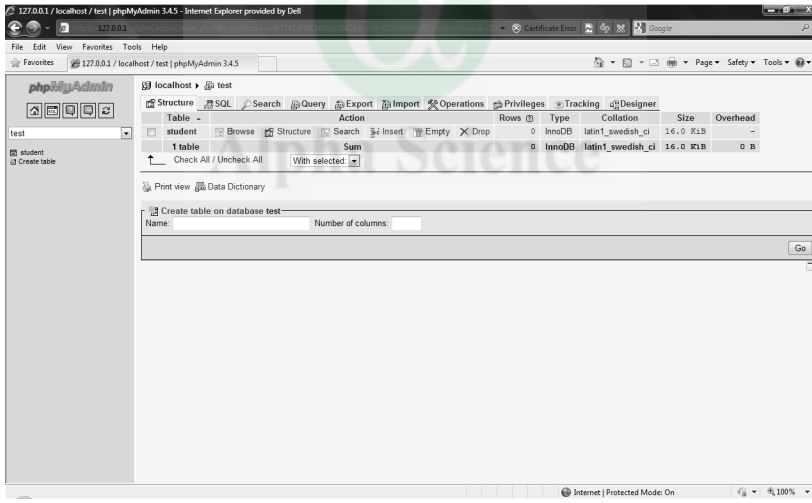


Figure 7. Table is created in the test database

After completing the work, the services may be stopped by clicking on the exit button of the XAMPP control panel and browser can also be closed to stop working with XAMPP platform.

If any problem persists, visit the site http://www.apachefriends.org/ and follow the discussion forums/FAQs for installation and usage.

Sample Questions

SAMPLE-A

1. (*a*) List and explain three characteristics of a DBMS.

(*b*) Describe hierarchical model with a suitable example.

2. (*a*) Who is a DBA? What are his responsibilities?

(*b*) State referential integrity rule with a suitable example.

(*c*) Define the terms:

(*i*) Domain;

(*ii*) Candidate key.

3. (*a*) What are the advantages of using SQL as a query language?

(*b*) What do you understand by normalization? Normalize the following relation into first and second normal forms.

Bills (**Bill No.**, Date, Item No., Item Description, Unit Price, Qty Bought, Total Cost)

Assume that each bill contains a list of multiple items. Each item has a unique Item No., and is identified by Item-description and a unit price.

4. (*a*) Name the tag used to define a hyperlink. Also, describe the minimum components contained in this tag, with a suitable example.

(*b*) Define an imagemap.

(*c*) Describe the use of the following tags/attributes in HTML:

(*i*) <PRE>

(*ii*) (BASEFONT)

(*iii*) MARGINHEIGHT

(*iv*) CELLPADDING

5. (*a*) What is the output of the following HTML code fragments?

(*i*) <HR NOSHADE ALIGN=”CENTER” WIDTH=”200">

(*ii*) <A HREF=”docl.html” TARGET=”_blank”>

New document </A>

(*iii*)
```
<TABLE>
    <TR>
        <TD COLSPAN="2"> Main subjects
        <TD> Subsidiary
        <TR>
            <TD> English
            <TD> Hindi
            <TD> French
        <TR>
            <TD> Algebra
            <TD> Calculus
</TABLE>
```

(*b*) Name and describe two block level elements used for functional and logical divisions.

(*c*) State the limitations of using frames.

6. (*a*) State any two instances in which a browser uses the alternate text (ALT attribute) in an image tag.

(*b*) Compare GIF and JPEG image file formats.

(*c*) How will you create a select list in a form with the following items:

Apple
Banana
Mango

With 'Mango' as the default selected item.

7. (*a*) Write a Javascript to display the following message:

She said, "Welcome to the world of Javascript."

(*b*) Declare and initialize an array to store marks in 4 subjects.

(*c*) Three Javascript variables are declared as follows:

s = "start"
I = 123
T = true

Write down the value of:

(*a*) s + I

(*b*) I + T

(*c*) What are the advantages of ASP over HTML?

8. (*a*) Name the major components of a computer network system.

(*b*) Differentiate between:

(*i*) Client server and peer to peer architecture.

(*ii*) Serial and parallel communication.

(*c*) What do you understand by Network topology? Describe the bus topology for Networks and give its disadvantages.

9. (*a*) Differentiate between e-Commerce and e-Governance.

(*b*) Name the protocol used to transfer the files between machines over the Internet.

(*c*) What is a URL? What are the main parts of a URL?

SAMPLE-B

1. Attempt the following:

With respect to a commercial organization, the following observations have been made:

- Each employee works under one of the several departments.
- Each department is headed by a Manager.
- The organization is engaged in project work for its clients.
- The employees working for a project may belong to different departments.
- The clients are billed for project work on the basis of man-hours.
- The rate of man-hours varies from department to department.

You are required to analyze and conceptualize the above organizational reality using ER Model and depict the same as ER diagram.

2. Attempt any one of the following pairs:

(*a*) Discuss the problems that are caused by data redundancy. To what extent data redundancy can be eliminated by adopting the database approach and how?

(*b*) Discuss the variety of services that are available using Internet Technology.

3. Attempt any one of the following pairs:

(*a*) Differentiate between logical data independence and physical data independence.

(*b*) What do you understand by World Wide Web (WWW)? How does it differ from Internet?

(*c*) What do you understand by the term Web Server and Website. Discuss the basics of language that is used for creating web pages.

(*d*) Explain the term Key. definite Primary Key, Foreign Key and Candidate Key.

(*e*) What do you understand by Web Server and Web Site? Explain Web Pages designing language.

SAMPLE-C

1. (*a*) What are ordered and unordered list in HTML?

(*b*) Write code to create the following lists:

List 1:

- ☐ HINDI
- ☐ ENGLISH
- ☐ MATHS
- ☐ HISTORY

List 2:

(*a*) BIRDS

(*b*) ANIMALS

(*c*) SNAKES

(*d*) FISHES

(*c*) Describe the use of the following tags:

(*i*) ROWSPAN

(*ii*) COLSPAN

(*iii*) TARGET

(*iv*) STYLE

(*d*) Create the following HTML page:

1. Chapter 1
2. Chapter 2
3. Chapter 3
4. Frame1 Frame2

This page contains two frames, frame 1 and frame 2.

Frame 1 contains 3 links. When you click on a particular link, its contents are displayed in frame 2.

2. (*a*) Write a JavaScript to create the following form:

Name:

Age:

A message should appear if a user enters less than 5 characters in a name.

(*b*) Given var text= "javascript"

Pos = text.indexof ("a");

What is the value of pos in the above statement?

Write a java script to the following:

Declare a variable 'day' and print the day of the week depending on value of the day. For example, if day = 0 print "Sunday"; if day = 1, print Monday; etc.

(*c*) Write a java script to calculate the sum of given series using the following form:

(*d*) $1^2 + 2^2 + 3^2 + \ldots\ldots + n^2$

Enter the value of n and produce the output. Make a calculate button and Use 'On Click ()' event on calculate button.

3. (*a*) What is star topology? Discuss its advantages and disadvantages.

(*b*) Differentiate between:

(*i*) Internet and World Wide Web;

(*ii*) Packet switching and Circuit switching.

(*c*) Write short notes on:

(*i*) Internet telephony

(*ii*) Video conferencing.

(*d*) Differentiate with the help of an example between Guided and Unguided media.

4. (*a*) What is data independence? Discuss logical and physical data independence.

(*b*) What are the advantages and disadvantages of using distributed database?

(*c*) Explain one-to-many relationship by giving suitable example.

(*d*) Consider the following relation for published books:

Book (Book_title, Authorname, Book_type, Listprice, Author_affil, Publisher)

Author_affil refers to the affiliation of the author.

Suppose the following dependencies exist:

Book_title → Publisher, Book_type

Book_type → Listprice

Authorname → Author_affil

(*i*) What normal form is the relation in? Explain.

(*ii*) Normalize the relation upto 3NF. Explain each step.

(*e*) Consider the following tables:

Emp (Emp_Id, Dept_Id, Emp_Name, Salary)

Dept (Dept_Id, Dept_Name, Supervisor)

Give the SQL for the following:

(i) Get names of all the employees working in the "Accounts" Deptt.

(ii) Increase the salary of all the employees by 10% whose Deptt. is "Library".

(*iii*) Delete all the employees whose salary is less than 2000.

(*iv*) Find the sum of salaries for each Deptt.

SAMPLE-D

1. Attempt the following:

With respect to an educational institution, the following observations have been made:

- It is engaged in imparting education to students in several courses.
- It has several teaching departments according to various subjects.
- It has core facilities belonging to one of its teaching departments.
- Each course consists of several papers. Each paper belongs to a particular teaching department.
- Each paper is characterized by its name, maximum marks, number of lectures per week and the department to which it belongs.

You are required to analyze and conceptualize above organizational reality, using ER Model and depict the same as ER diagram.

2. Attempt any one of the following pairs:

(*a*) What are the main types of database end users? Discuss the main activities of each.

(*b*) What are the limitations of file oriented approach?

(*c*) What are the advantages of using Internet for a commercial organization?

(*d*) What is meant by Telnet? How is this useful to a commercial organization that has branches across the country?

3. Attempt any one of the following pairs:

(*a*) What do you understand by the term Key? Explain Foreign Key and Candidate Key.

(*b*) What is meant by complex attribute? How does it differ from multi-valued attribute?

(*c*) Discuss the type of browsers that are used for Net-Surfing.

(*d*) What do you understand by Web Server? How does it differ from Web Browsers?

MULTIPLE CHOICE OBJECTIVE QUESTIONS

1. The ————————is the physical path over which a message travels.

(*a*) protocol (*b*) medium

(*c*) message (*d*) signal

2. In most bus networks, ____ cabling is used as the physical medium.

(*a*) coaxial (*b*) fiber-optic

(*c*) twisted-pair (*d*) Either A or B

3. Coaxial cable is an example of,

(*a*) Guided media (*b*) Unguided media

(*c*) Both (*a*) and (*b*) (*d*) Circuit switching

4. Which of the following is not an example of unguided media?

(*a*) Radio waves (*b*) Satellite

(*c*) Optical fiber (*d*) Micro wave

5. Sharing of medium and its path is known as?

(*a*) multiplexing (*b*) encoding

(*c*) modulation (*d*) line discipline

6. If the bandwidth of a signal is 10 khz and the lowest frequency is 60 khz, what is the highest frequency?

(*a*) 70 khz (*b*) 6 khz

(*c*) 50 khz (*d*) 75 khz

7. What is the bandwidth of a signal that ranges from 50 khz to 5 MHz?

(*a*) 4.95 MHz (*b*) 55 MHz

(*c*) 5.50 MHz (*d*) 4.5 MHz

8. The process of converting analog signals into digital signals is referred as:

(*a*) Modulation (*b*) Demodulation

(*c*) Synchronization (*d*) Amplification

9. Which of the following is not a broadband communication channel?
(*a*) Satellite (*b*) Fiber optic
(*c*) Cable TV (*d*) Ethernet

10. Which of the following communication modes support two-way, but in only one direction at one time?
(*a*) Simplex (*b*) Half-duplex
(*c*) Full duplex (*d*) Both (*a*) and (*b*)

11. ISDN is an example of _____ network
(*a*) Circuit switched (*b*) Packet switched
(*c*) Message switching (*d*) both (*a*) and (*b*)

12. What topology is best used to connect remote sites (WANs), using a router to determine the best path?
(*a*) star (*b*) bus
(*c*) mesh (*d*) Token ring

13. Ports are:
(*a*) Input devices (*b*) Output devices
(*c*) Used to allow I/O devices to communicate with the CPU
(*d*) Able to process incoming data

14. Which transmission mode allows both communicating devices to transmit and receive data simultaneously?
(*a*) Full-duplex (*b*) Simplex
(*c*) Half-duplex (*d*) Synchronous

15. Which of the following is wireless standard?
(*a*) 802.11 (*b*) RJ45
(*c*) 802.2 (*d*) RJ11

16. OSI Stands for?
(*a*) Open System Interface (*b*) Open System Interconnection
(*c*) Open standard Interface (*d*) Office system Interface

17. Topology refers to
(*a*) The physical arrangement of a network
(*b*) The logical arrangement of a network
(*c*) Option (*a*) and (*b*)
(*d*) Collection of computers

18. In which network topology, every node has exactly two neighbors?
(*a*) Ring (*b*) Star
(*c*) Mesh (*d*) Bus

19. A full domain name is a sequence of labels separated by _____.
(*a*) semicolons (*b*) dots
(*c*) colons (*d*) commas

20. What is the maximum data capacity for optical fiber cable?
(*a*) 10 mbps (*b*) 100 mbps
(*c*) 1000 mbps (*d*) 10000 mbps

21. In which of the following network architecture, each workstation has equal capability and responsibility?
(*a*) Peer to peer (*b*) Client server
(*c*) LAN (*d*) MAN

22. Each IP address is a unique ____bit number.
(*a*) 32 (*b*) 8
(*c*) 16 (*d*) 24

23. When a host on network A sends a message to a host on network B, which address does the router look at?
(*a*) port (*b*) logical
(*c*) Physical (*d*) All of the above

24. In which network topology, bi-directional link exist between each possible node?
(*a*) bus network (*b*) ring network
(*c*) star network (*d*) Mesh network

25. In a ____ topology, each node is connected to the two nearest nodes, and data is transmitted clockwise.
(*a*) bus (*b*) star
(*c*) ring (*d*) mesh

26. Which network device uses a MAC address table to segment the network?
(*a*) hub (*b*) Gateway
(*c*) Router (*d*) switch

27. In a Mesh network, to connect n nodes how many links are required?
(*a*) n+3 (*b*) n*n
(*c*) n*n*n (*d*) n(n-1)/2

28. The failure of a computer in the network does not affect network functioning.
(*a*) bus (*b*) star
(*c*) ring (*d*) Both (*a*) and (*b*)

29. Which LAN has the highest data rate?
(*a*) 10base2 (*b*) 10baseT
(*c*) token ring (*d*) FDDI

30. The default lease time of an IP address is _____.
(*a*) 30 minutes (*b*) 1 hour
(*c*) 2 hours (*d*) 3 hours

31. Which port is used by TELNET?
(*a*) 23 (*b*) 21
(*c*) 25 (*d*) 27

32. What is the minimum length of an Ethernet frame?
(*a*) 64 bytes (*b*) 72 bytes
(*c*) 10000 bytes (*d*) 80 bytes

33. Frame Relay operates in the _____ .
(*a*) physical layer (*b*) data link layer
(*c*) network layer (*d*) transport layer

34. Bluetooth is a type of radio wave information transmission system that is good for about?
(*a*) 30 feet (*b*) 30 yards
(*c*) 30 miles (*d*) 300 miles

35. How many OSI layers are covered in the X.25 standard?
(*a*) Two (*b*) Three
(*c*) Seven (*d*) Six

36. The data unit in the TCP/IP Data link layer is called——
(*a*) Frame (*b*) Packet
(*c*) Segment (*d*) message

37. The overall logical structure of a database can be expressed graphically by _____
(*a*) ER diagram (*b*) Records
(*c*) Relations (*d*) Hierarchy

38. A table can have maximum _____ no. of Long Raw datatype columns
(*a*) One (*b*) No limit
(*c*) 255 (*d*) depends on tablespace

39. A user is setting up a join operation between Emp and Dept tables. The query should return all the employees, who are assigned dept, as well as the employees who are not assigned any dept yet.
(*a*) Select e.empid, d.head from emp e, dept d;
(*b*) Select e.empid, d.head from emp e, dept d where e.deptno = d.deptno;
(*c*) Select e.empid, d.head from emp e, dept d where e.deptno = d.deptno (+);
(*d*) Select e.empid, d.head from emp e, dept d where e.deptno (+)= d.deptno;

40. A Cartesian product is
(*a*) A group function
(*b*) Produced as a result of a join select statement with no where clause
(*c*) The result of fuzzy logic
(*d*) A special feature of Oracle server

41. The maximum length of varchar2, when used in SQL is,
(*a*) 16k (*b*) 32k
(*c*) 4k (*d*) Either (*b*) or (*c*)

42. The max length of varchar2, when used in a table is,
(*a*) 16k (*b*) 32k
(*c*) 4k (*d*) Either (*b*) or (*c*)

43. GRANT and REVOKE are
(*a*) DDL statements (*b*) DML statements
(*c*) DCL statements (*d*) None of these

44. An Oracle 8i database can store
(*a*) Text (*b*) Audio/Video
(*c*) Images (*d*) All the above

45. Oracle 8i can be best described as
(*a*) Object-based DBMS (*b*) Object-oriented DBMS
(*c*) Object-relational DBMS (*d*) Relational DBMS

46. INITCAP ('oracle') returns
(*a*) oracle (*b*) ORACLE
(*c*) ORACLe (*d*) None of these

47. The value of INSTR('CALIFORNIA','A',1,2) is
(*a*) 2 (*b*) 1
(*c*) 10 (*d*) 9

48. The SQL query to retrieve current time in Oracle is
(*a*) Select SysTime from dual
(*b*) Select SysDate from dual
(*c*) Select TO_CHAR (SysDate,'HH:MI:SS') from dual
(*d*) Select TO_CHAR (SysDate,'HH:MM:SS') from dual
(*e*) Both (*c*) and (*d*)

49. The correct syntax of DECODE statement is,
(*a*) DECODE(if1,then1,if2,then2,……,else)
(*b*) DECODE(if1,then1,else1,if2,then2,else2……,)
(*c*) DECODE(value,if1,then1,else1,if2,then2,else2……,)
(*d*) None of these

50. Consider the following where clause,
WHERE A.no=B.no(+)
The above outer join lists
(*a*) All matching & non-matching rows of table B
(*b*) All matching & non-matching rows of table A
(*c*) All non-matching rows of table B
(*d*) All non-matching rows of table A

51. TRUNCATE TABLE statement
(*a*) Drops a table temporarily
(*b*) Removes all the rows of a table
(*c*) Removes specified number of rows from a table
(*d*) Removes all constraints from a table

52. Oracle 8i runs on?
(*a*) Windows NT (*b*) UNIX
(*c*) Solaris (*d*) both (*a*) and (*b*)
(*e*) all (*a*), (*b*) and (*c*)

53. The owner of all data dictionary tables in Oracle is,
(*a*) SYSTEM (*b*) SYS
(*c*) INTERNAL (*d*) Users

54. Which of the following is not a RDBMS?
(*a*) MS Access (*b*) Oracle 8i
(*c*) MS SQL Server (*d*) Informix

55. The character used for concatenating strings in Oracle is,
(*a*) && (*b*) +
(*c*) | (*d*) None of these

56. A pre-compiler converts
(*a*) PL/SQL code into executable machine code
(*b*) Embedded SQL statements into native library calls
(*c*) A program written in a 3GL into a PL/SQL code
(*d*) A PL/SQL code into a 3GL code

57. In a query with a GROUP BY clause, the columns in the SELECT clause
(*a*) must be aggregate functions only
(*b*) must also appear in the GROUP BY clause
(*c*) must also appear in the GROUP BY and HAVING clauses
(*d*) Both (*a*) and (*b*)
(*e*) Can be selected arbitrarily

58. The name of the only column in the DUAL table is,
(*a*) X (*b*) C
(*c*) DUMMY (*d*) None of these

59. The CHAR and NCHAR data types
(*a*) have no difference
(*b*) belong to two different character sets
(*c*) have different storage requirements
(*d*) have different storage requirements and retrieval time

60. The wild card character % used in strings means,
(*a*) any number of characters (*b*) one or more characters
(*c*) at least one character (*d*) at least one alphabet

61. Which of the following objects are dropped automatically, when a table is dropped?
(*a*) Procedure (*b*) Constraints
(*c*) Views (*d*) Synonyms

62. In order to perform an inner join, which criteria must be true?
(*a*) The common columns in the join do not require shared values.
(*b*) The tables in the join need to have common columns.
(*c*) The common columns in the join may or may not have shared values.
(*d*) The common columns in the join must have shared values.

63. A JavaScript is...
(*a*) the same as Java (*b*) similar like Java
(*c*) different than Java (*d*) other written part of Java

64. A JavaScript is...
(*a*) Subjective (*b*) Objective
(*c*) Evil (*d*) Object based

65. To comment out a line in JavaScript...
(*a*) Precede it with two forward slashes, *i.e.* '//'
(*b*) Precede it with an asterisk and a forward slash, *i.e.* '*/'
(*c*) Precede it with an asterisk, *i.e.* '*'
(*d*) Precede it with a forward slash and an asterisk, *i.e.* '/*'

66. JavaScript can only run on Windows.
(*a*) True (*b*) False

67. Semicolons are optional at the end of a JavaScript statement.
(*a*) True (*b*) False

68. What tag is used to add columns to tables?
(*a*) <colspan> (*b*) <td>
(*c*) <tr>

69. To change the size of an image in HTML uses?

(*a*) Big and small (*b*) Height and Width

(*c*) Bigger and smaller

70. Choose the correct HTML tag to make a text italic

(*a*) <italic> (*b*) <i>

71. How can you make a list that lists the items with numbers?

(*a*) <dl> (*b*)

(*c*) (*d*) <list>

72. What does HTML stands for?

(*a*) Hyper Text Markup Language

(*b*) Hyperlinks and Text Markup Language

(*c*) Home Tool Markup Language.

Index

D

E

F

G

H

I

R

S

T

U

V

W

X

About the Authors

V.S. Dixit, working as an Assistant Professor & Coordinator, Department of Computer Science, ARSD College, University of Delhi. He has around fourteen years of teaching experience in Computer Science subjects. He has authored Computer Science Books on the topics like Python Programming language, Internet, Computer Fundamentals, etc and Blocks on TOC, and Java Multithreading. He has keen interest in research and published research papers in international journals and conferences.

V.B. Singh is currently working as Associate Professor in the Department of Computer Science at Delhi College of Arts and Commerce, University of Delhi, Delhi (India). He received his M.C.A. degree from M. M. M. Engineering College, Gorakhpur, U.P. (India) and Ph.D. degree from University of Delhi. He has published more than thirty research papers. His research interests include mining software repositories, empirical software engineering and software reliability engineering. He is member of ACM.