UNIT - I

1 Web Essentials

Syllabus

Creating a Website - Working principle of a Website - Browser fundamentals - Authoring tools - Types of servers: Application Server - Web Server - Database Server.

Contents

2

1.1 Creating Web Site

- Web Site is a collection of Web pages.Hence for a web site design we need to design the web pages.
- Each Web page may contain texts, photos, videos, social media buttons and so on.
- Technically, a Web Page is a special type of document written in scripting language such as
  - HTML
  - CSS
  - JavaScript
  - PHP
  - And so on
- Web pages are written for web browsers. The web browsers are the programs like Internet Explorer, Google Chrome, and Safari. These browsers have a simple but crucially important job : they read the web page document and display the perfectly formatted result.

Definition of Web Site : Web Site is a collection of web pages that are grouped together to achieve certain task under single domain name.

Why do people visit web site ?

Generally, people look at websites for two primary reasons :

1. The most important reason is to **find the required information.** This could be anything from a student looking for images for a school project, to finding the latest stock quotes, for getting the address of the nearest restaurant and so on.
2. To **complete a task.** Visitors may want to buy the latest best-seller, download a software program, or participate in an online discussion about a favorite hobby.

1.1.1 Steps for Creating the Web Site

Following steps are followed for creating a web site -

Suppose you wish to publish some web page on the internet then following steps can be followed -

Step 1 : Web Site Creation

- **Create** a web pages using suitable **scripting language.**
- If any **image** is associated with this web page then convert this image into appropriate format (JPEG or GIFF format is preferable). Embed this image appropriately in this web page.

Step 2 : Choose the Web Hosting Service

- Web hosting company **hosts your web pages** on web server. Thus your web site will be available to any one who knows your URL.
- Most web hosting companies offer hosting services for both personal and business use.
- The web host provides you with **Internet access, email accounts,** and **space** for a personal or business web site.
- If you are building a web site for business use, your web host can register a personalized domain name for your web site.
- Small Web sites(around 15-20 pages of contents) do not need much more than 1 or 2 MB of server space that hold all the HTML pages and Graphics. Your web hosting package should provide at least MB of space so your web page has room to grow.

Step 3 : Registering Domain Name

- Domain name is an alias that points to actual location of your web site on Web server. Domain names are managed by the **I**nternet **C**orporation for **A**ssigned **N**ames and **N**umbers(ICANN). ICANN has agreements with a number of vendors to provide domain name registration services.

Step 4 : Planning your web site

When planning your website, you will need to make a number of important decisions :

- **Type :** The type of site you need. Is this a news or informational site, a site for a company or service, a non-profit or cause-driven site, an Ecommerce shop, etc. Each of these kinds of site has a slightly different focus that will influence its design.
- **Navigation design :** Navigation means indication that how users will move around your site affects

its information architecture as well as the overall usability of that site. Plan out the pages a site, create a sitemap, and develop a navigational structure from there.

- **Content :** The quality of your site's content will play an important role in it's success. Content is everything that your pages will contain, such as text, images, video and more. Before you start designing or building pages, you should have a clear strategy for the content that those pages will contain.

Step 5 : Uploading Files

- To publish a web site on the web, you must send the web pages created by you on the web server using a File Transfer Protocol(FTP). Using some software such as Microsoft Visual Studio or Adobe Dreamweaver one can upload the files on the web server.

1.1.2 Testing the Web Site

Testing must be performed throughout the development of web site. Even after creating the web site, it must be tested for as web pages are present live on the web. Various factors for testing the web site after publishing are -

1. **Multiple Browser :** It is necessary to display the web site on as many web browsers as possible to ensure that the contents of the web site are consistently displayed and the work done is portable.
2. **Multiple Operating Systems :** It is necessary to display the web site on different operating systems.
3. **Connection Speed :** Do not rely on the same connection speed when testing your web site, specially if you work in a corporate environment where the connection to the Internet usually is faster than the average user's.Also test the download time for different connection speed.
4. **Device Types :** Test the web site on the computers having different screen size. It is necessary to ensure that pages are displayed consistently on all screen size.
5. **Links :** Use a link validation tool to ensure that all of your links connect to a live page. Link validation tools are built into many HTML editors and are available as stand-alone tools. Many web sites also offer validation, including the WC's link validator at validator.w3.org/checklink.
6. **Security Testing :** This step is necessary to test the security vulnerabilities in application running on the web site. Security is an important part of any web development plan.

Review Question

1. *Explain the process of developing a web application and hosting on the web server.*

1.2 IP Addressing

- Each host on a TCP/IP network is assigned a unique 32-bit logical address that is divided into two main parts : the network number and the host number. This address is called IP address.
- The IP address is grouped four into 8-bits separated by dots. Each bit in the octet has binary weight.
- As we know the IP address is divided in two categories network number and host number.

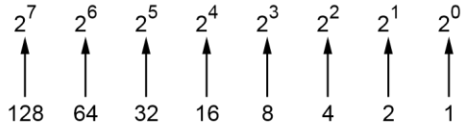


Fig. 1.2.1

Network number	Host number

- There are 5 classes based on two categories viz. A, B, C, D and E.

IP address class	Format	Range	Purpose
Class A	N.H.H.H	1 to 126	Very few large organizations use this class addressing.
Class B	N.N.H.H	127 to 191	Medium size organizations use this addressing.
Class C	N.N.N.H	192 to 223	Relatively small organizations use this class.

4

Class D	–	224 to 239	This class address is used for multicast groups.
Class E	–	240 to 254	This class addressing is reserved for experimental purpose.

- Here N stands for network number and H stands for host number. For instance in class C first three octets are reserved for network address and last 8-bits denote host address.
- IP address is assigned to the devices participating in computer network. The IP protocol makes use of this address for communication between two computers. Using IP address particular node can be identified in the network.

1.3 DNS

- It is very difficult to remember numerical information but it is simple to remember the textual information.
- Consider that we want to access Priyanka's PC, then accessing it using the IP address **www.192.168.0.101** is definitely not comfortable, rather if we have the address **www.priyanka@technical.com** then accessing and remembering Priyanka's PC address is very simple.
- The names which are used to identify computer within a network are called **domain names.** Thus domain name is the name given to a network for human reference.

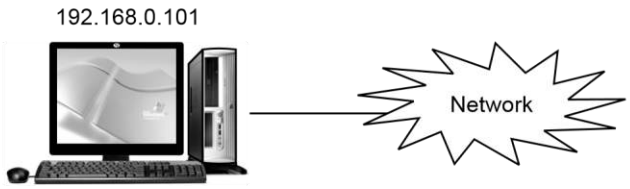


Fig. 1.3.1

- Hence in DNS, instead of using the IP address name of the Computer is used to access it. But two names can be the same. Hence to uniquely identify your computer the name must be referred using **DNS hierarchy.**
- Before understanding this hierarchy let us list out some commonly used domain names -

Domain names	Purpose
com	Commercial organization
gov	Government organizations
edu	Educational institutes/organizations
int	International organization
net	Network group
org	Non profit organization
mil	Military group/organizations
in	Sub domain name used to refer India
uk	Sub domain name used to refer United Kingdom
jp	Sub domain name used to refer Japan

- Refer Fig. 1.3.1 and Fig. 1.3.2.
- The **domain name space** is used to locate the computer uniquely. The internet logically arranges the domain names in an hierarchical form.
- There are some top level DNS such as com, org, edu, mil, net, uk, in and so on. Then each domain name is further divided into sub-domains then sub-sub-domains and so on.

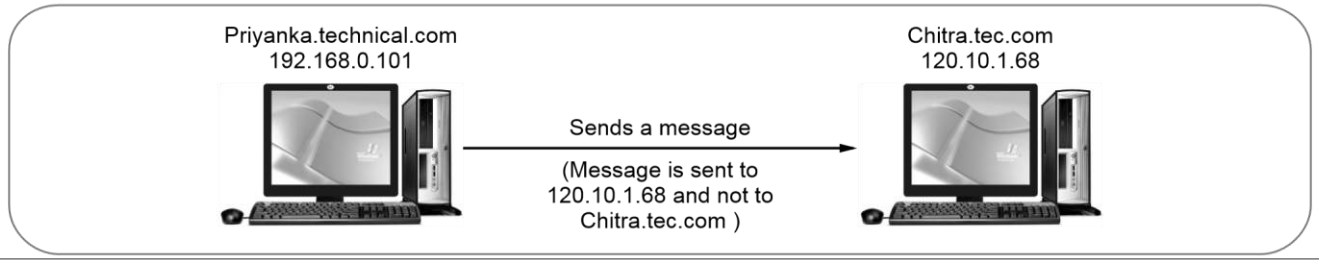


Fig. 1.3.2 Communication using domain names

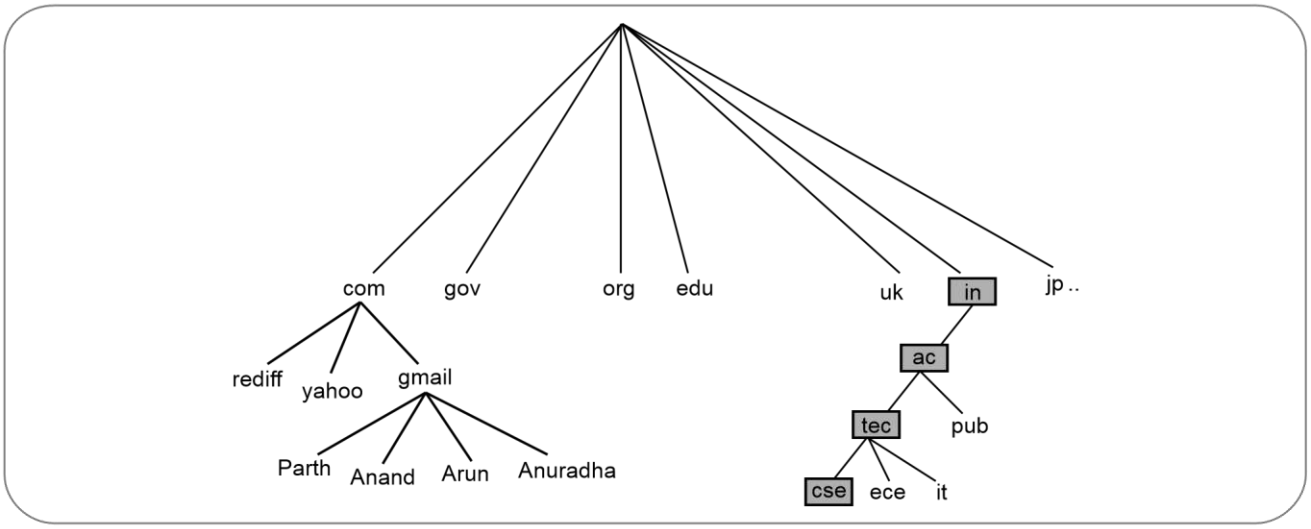


Fig. 1.3.3 Domain name space

- For example the complete path for **http://www.cse.tec.ac.in** can be uniquely traced out with the help of domain name space.
  (Refer Fig. 1.3.3)

Working of DNS

There are two tasks that can be carried out by DNS servers - (Refer Fig. 1.3.4)

1. Accepting and then requesting the programs to convert domain names to IP addresses.
2. Accepting and then requesting the other DNS servers to convert domain names to IP addresses.

- Suppose PC A is interested in knowing the IP address of **technicalpublications.org** then it contacts nearest DNS server. This DNS server maintains huge **database of domain names.** The entry domain name technicalpublications.org is searched within this database and if the IP address for corresponding name is found then the IP address is returned PC A. If the domain name requested by you is not there in the DNS server then name of another DNS server is suggested. If the request is made for some invalid domain name then the error message is returned.

1.4 URL

- The **U**niform **R**esource **L**ocator (URL) is unique address for the file that has to be accessed over the internet. When we want to access some website we enter it's URL in the address bar of the web browser. For example if we want to access **www.google.com** then we must specify its URL in the address bar as shown Fig. 1.4.1.
  (See Fig. 1.4.1 on next page.)

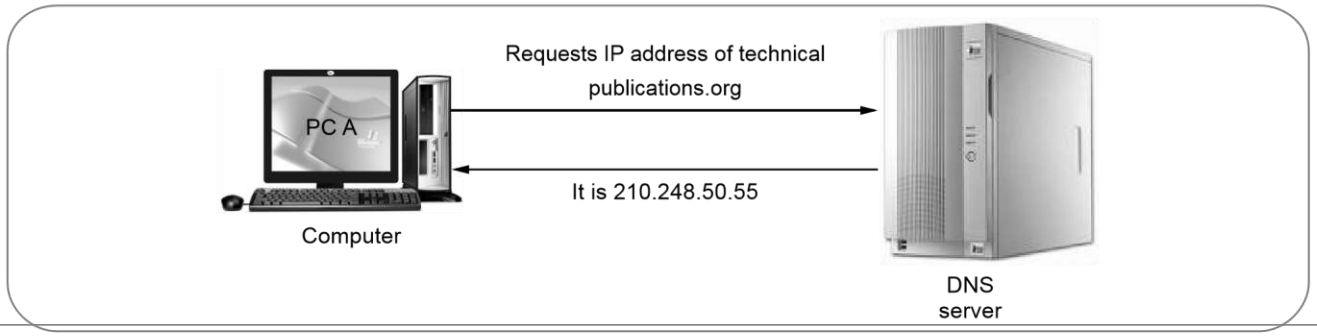


Fig. 1.3.4 Working of DNS

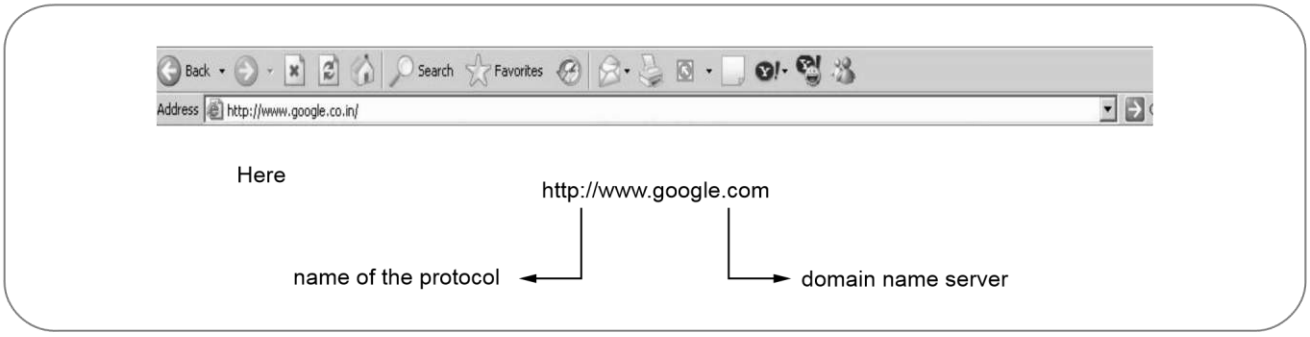


Fig. 1.4.1

- However any other file such as some text file or image file or some HTML file can also be specified. The URL contains name of the protocol such as **http://**

The URL may contain the name of the protocol as such as ftp. For example

ftp://ftp.funet.fi/pub/standards/RFC/rfc2166.txt

The protocol identifier and the resource name are separated by a colon and two forward slashes. The syntax of writing URL is as given below,

protocol://username@hostname/path/filename

Sometimes instead of domain name servers IP addresses can also be use

For example,

http://192.168.0.1

- But use of IP address as URL is not preferred because human can not remember numbers very easily but they can remember names easily.

Absolute and Relative URL

The absolute URL is a URL which directly point to a file. It exactly specifies exact location of a file or directory on the internet. Each absolute URL is unique. For example -

http://www.vtubooks.com/home.aspx

The relative URL points to the file or a directory in relation to the present directory. For example -

Consider the **absolute address** which refers an image mother.jpg

http://www.mywebsite.com/myphotos/mother.jpg

For the above given absolute address the **relative address** will be -

../myphotos/mother.jpg

- That means from the current URL the directory **myphotos** will be searched for the image **mother.jpg.**
- The two peroids .. instruct the server to move up one directory which is the *root* directory, then enter myphotos directory **(/myphotos)** and finally point at **mother.jpg.** Thus using relative URL writing of long path name can be avoided.

1.5 Working Principle of a Web Site

1.5.1 Features of Web Site Design

There are many features that need to be considered while designing the website for your business. Some of these features are mentioned below-

1. Quality Web Content :

- People desire information in fast and reliable fashion. For business websites, content should include important information. These type of web sites need to display high quality pictures of their products, and the highlight for client testimonials.

2. Clear, User-friendly Navigation :

- A user-friendly navigation scheme allows visitors to quickly find the information needed. Important links must be easy to find and given logical, simple, and include easy-to-understand labels. If there is a plethora of content, then a search box is suggested to make it faster to reach more specific pages within a website.

3. Simple and Professional Web Design :

- The web site design must be simple and professional. Google is an excellent example of such a site. To keep websites simple a balanced distribution of contents and graphics is required. The use of slightly contrasting colours and clear fonts is also necessary. Also, one should break up sizeable blocks of text with either spacing or images as appropriate.

4. Webpage Speed :

- People inherently lose patience quickly, when visiting a website. The website with heavy graphics, audio and video takes more time to load. A web design company must take care of all the controlling factors that will maintain the desirable speed of the web site.

5. Search Engine Optimization :

- A well-designed website generally will receive many visitors, and one method to attract visitors is search engine optimization. This allows the insertion of search keywords in website content, an appropriate link profile, social media signals.

6. Web Compatibility :

- A web site should easily render on various resolutions, screen sizes, and browsers; and with the increasing popularity of mobile devices, websites should function properly on these types of devices.

1.5.2 Web Site Design Issues

Jean Kaiser has suggested following design goals for the web design -

1. Simplicity

- It is a general tendency of web designers to provide lot of animations, huge amount of information, extreme visuals and so on. This makes the web design enormous and it should be avoided. The web application must be moderate and simple.

2. Identity

- Web design must be based on the nature of the web application. It is driven by the objective of the web application, category of user using it. A web engineer must work to establish an identity for the web application through the design.

3. Consistency

- The contents of the web application should be constructed consistently. For example: text formatting, font style should be the same over all the text document of the web application. Similarly, the graphics design, color scheme and style must be identical over all the web pages of the web application. Architectural design should produce the templates using which the consistent hypermedia structure can be formed. Interface design should define the consistent modes of interaction, navigation and content display. Navigation mechanism must be used consistently across web application elements.

4. Robustness

- The users always expects robust contents and functions of the web application. That means any required functionality should not be missing at all, If any function or content is missing or insufficient then that web application will fail.

5. Navigability

- The navigation should be simple and consistent. The design of navigations should intuitive and predictable in nature. That means any novice user should be in a position to make use of navigation links without any help.

6. Visual appeal

- The web applications are most visual and most dynamic and aesthetic in nature. There are various factors that contribute to visual appeal. These factors are -

  Look and feel of the content, interface layout, color co-ordination, the balance of text, graphics and other media, navigation mechanism and so on.

7. Compatibility

- The web application can be used in variety of environment and configurations such as different browsers, internet connection types, operating systems and various browsers.

1.6 Phases of Website Development

Web project can be designed in the four phases as given below -

Phase I : Strategy

- In this phase, a strategic planner or project manager along with the client determines the objective of the site. As an output of this phase **creative briefs** are prepared. The creative brief is a kind of document in which project objectives, requirements and key insights are clearly mentioned. Every team member makes use of creative brief as a guideline for the development.

Phase II : Design

- In this phase actual design of the website is done with the help of creative and technical team members. The front end is designed by the creative team in which user interface and interactions are designed. The back end is designed by the technical team which is responsible for designing the database architecture. As an outcome of this phase functional and technical specifications, site architecture are prepared.

Phase III : Production

- During this phase actual site is built using the source code. Functionalities and features of the website are closely examined. If the client demands for a change in any functionality then a change order is issued. At the end of this phase a production guide is prepared.

Phase IV : Testing

- At this phase all the functionalities and features of the website are tested, bugs are identified and resolved before launching the website. The QA manager develops the test plan. The test suit mentioned in it used to test the product thoroughly.

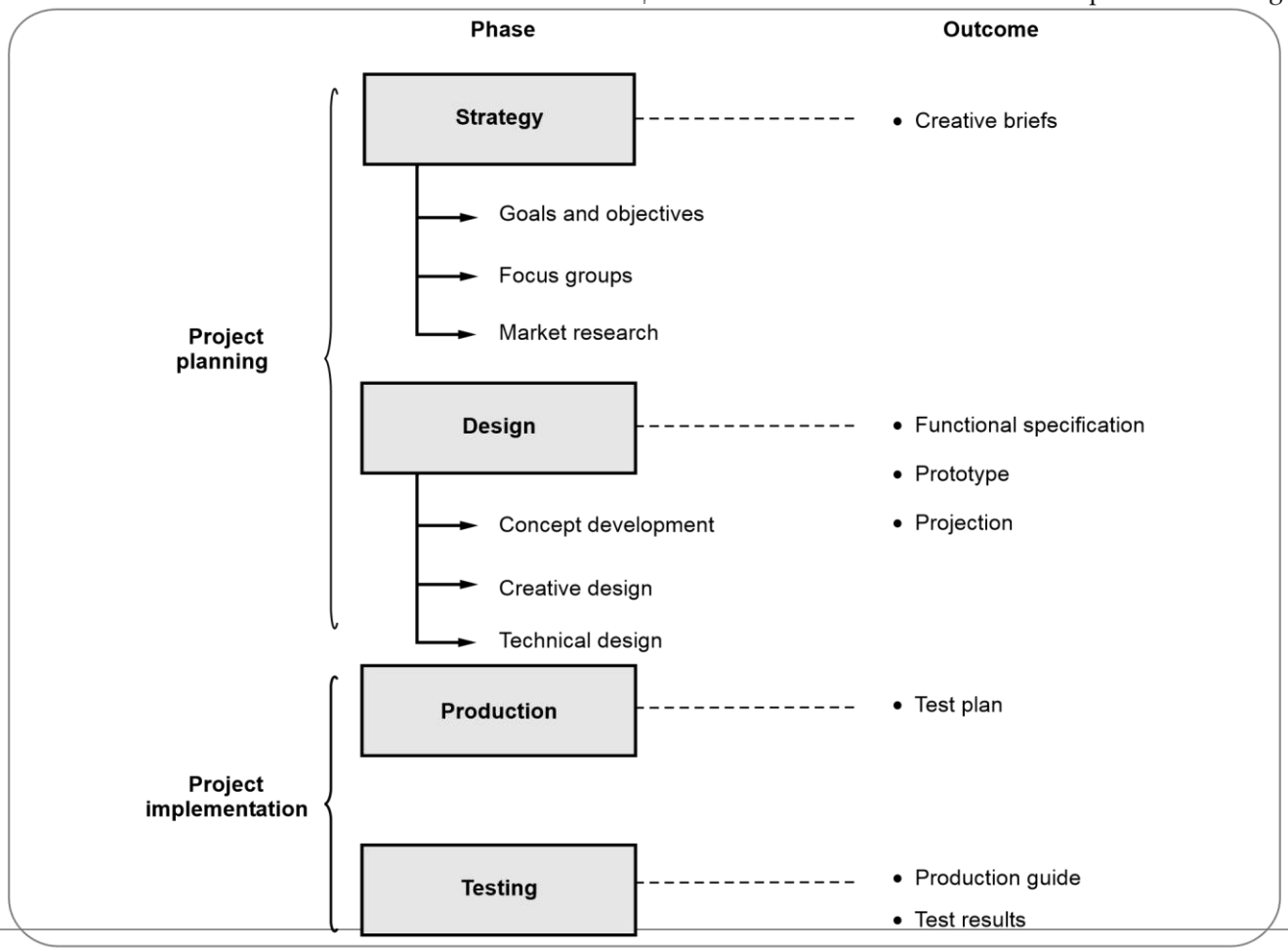


Fig. 1.6.1 Web project development phases

1.7 Enhancing Website

There are variety of ways by which one can enhance his website. The website can be enhanced using some key elements such as -

- **Content -** This is the most important element of website. It helps to spread the business message in an appropriate manner. The content should be easy to understand. Those should be to the point and relevant. The information available on the website must be useful to the user.
- **Graphics -** Adding too much graphics in the web page slows down its speed of loading. Hence Graphics is undesirable by any user. However the relevant and appealing graphics can be added to the website.
- **Colors and Text -** The colors and text that is appearing on the website must be pleasant to the eyes. As a rule of thumb, the entire site must use at the most five to six colors. The text should not be too small or too large. The selection of the family of font for displaying the text must be appropriate so that the text can be readable.
- **Flash -** Use of flash animation makes the site attractive but at the same time there are many drawbacks that are associated with this key element. The first drawback is the flash files take a large amount time to load the data on the web page. Secondly if the flash animation is placed on the web page then the link for downloading the flash player must also be provided so that the animation can be viewed by the user.
- **Frames -** Frames must be avoided while designing the website. Instead of using frames the web designers must prefer the tables. The reason why the use of frames must be avoided in the web page is that - the search engines find it difficult to search the contents from the site containing the frames.
- **Organizing Files -** The files required by the website must be categorized and must be stored in sorted manner. This makes it easier to manage the information.

Review Question

1. *Discuss on any four components that are needed to create a fully dynamic web page.*

1.8 Browser Fundamentals

Definition : Web browser is a kind of software which is basically used to use resources on the web.

- Over the networks, two computers communicate with each other.
- In this communication, when request is made by one computer then that computer is called **a client** and when the request gets served by another computer then that computer is called **server.** Thus exchange of information takes place via Client-Server communication.
- When user wants some web document then he makes the request for it using the web browser.
- The browsers are the programs that are running on the clients' machines. The request then gets served by the server and the requested page is then returned to the client. It is getting displayed to the client on the web browser.
- The web browsers can browse the information on the server and hence is the name.
- Various web browsers that are commonly used are

Browser	Vendor
Internet Explorer	Microsoft
Google Chrome	Google
Mozilla Firefox	Mozilla
Netscape Navigator	Netscape Communications Corp.
Opera	Opera Software
Safari	Apple

- Web browser supports variety of protocols but the most commonly used protocol on the web browser is **Hyper Text Transfer Protocol(HTTP).** This protocol is typically used when browser communicates with the server.

0

Fig. 1.8.1

1.8.1 Functions Defined by Web Browser

Various functions of web browser are -

1. Reformat the URL and send a valid HTTP request.
2. When user gives the address of particular web site it is in the form of domain name. The web browser coverts the DNS to corresponding IP address.
3. The web browser establishes a TCP connection with the Web browser while processing the user's request.
4. The web browsers send the HTTP request to the web server.
5. The web server processes the HTTP request sent by the web browser and returns the desired web page to the client machine. The web browser on the client's machine displays this web page in appropriate format.

1.8.2 Web Browser Architecture

The web browser architecture is represented by following figure -

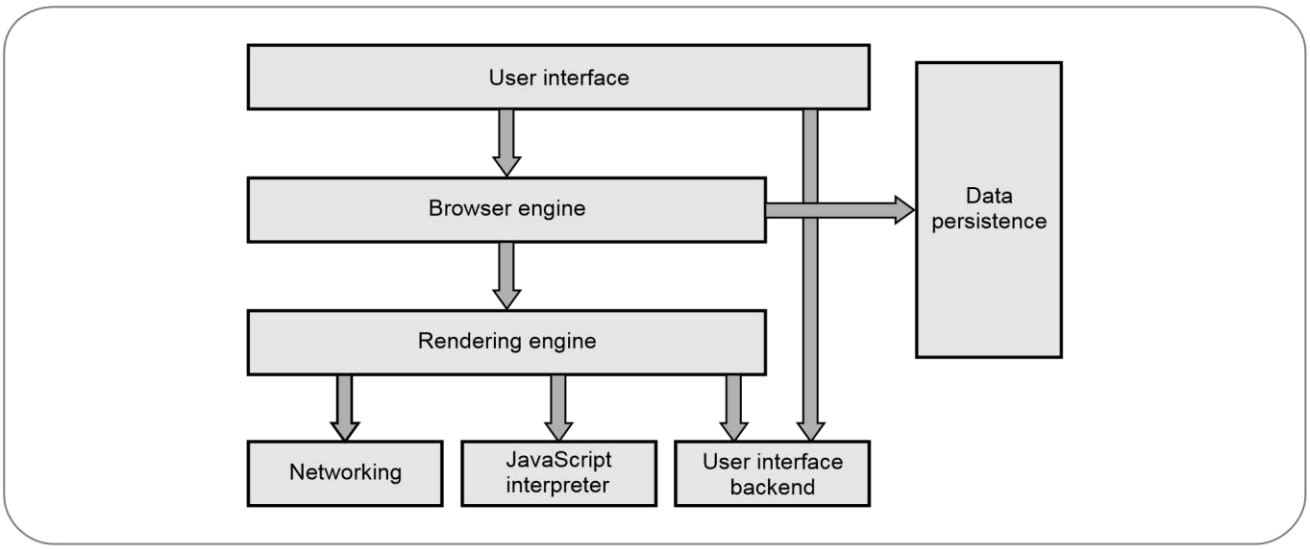


Fig. 1.8.2 Architecture of Web Browser

1

The main components of web browser architecture are as follows -

User Interface :

- Using the user interface user interacts with the browser engine.
- The user interface contains, Address bar, back/forward button, book mark menu and so on. The page requested by the user is displayed in this user interface.

Browser Engine :

- It contains the mechanism by which the input of user interface is communicated to the Rendering Engine.
- The browser engine is responsible for querying the rendering engine according to various user interfaces.

Rendering Engine

- It is responsible for displaying the requested contents on the screen.
- The rendering engine interprets the HTML, XML and JavaScript that comprises the given URL and generates the layout that is displayed in the user interface.
- The main component of rendering engine is HTML Parser. The job of the HTML parser is to parse the HTML markup into a parse tree.
- It is important to note that Chrome, unlike most browsers, holds multiple instances of the rendering engine - one for each tab, Each tab is a separate process.
- Different browsers use different rendering engines - Internet Explorer uses Trident, Firefox uses Gecko, Safari uses Webkit, Chrome and Opera uses WebKit.

Networking :

- The functionality of networking is to retrieve the URL using common internet protocols such as HTTP and FTP.
- The networking is responsible to handle the internet communication and security issues.
- The network component may use the cache for retrieved documents. This feature is useful for increasing the response time.

JavaScript Interpreter :

- The interpreter executes the JavaScript code which is embedded in a web page.

User Interface Backend :

- It is basically used to draw the widgets like combo boxes and windows.

Data Persistence :

- This is a small database created on local drives of the computer where the browser is installed.
- The data storage manages user data such as book marks, cookies, and preferences.

1.8.3 Working of Web Browser

We often browse the internet for several reasons. It is more interesting to know about how a web page demanded by us gets displayed on our browser. Following is the stepwise explanation of this process -

Step 1 :

- First user types the websites address for demanding the desired web page for example -

http://www.vtubooks.com/home.aspx

and then the home page of this website appears on the screen.

- The web address is divided into three parts:
  (i) the first part is the protocol. The **http** is a hypertext transfer protocol which tells the web browser that user wishes to communicate with web server on **port 80.** Port 80 is reserved for the communication between web server and web browser.
  (ii) The second part is the server address. This tells the web browser which server it needs to contact in order to retrieve the information you are looking for. The web browser communicates with a **Domain Name Server (DNS)** to find out the IP Address for the website. All communications on the Internet use IP Addresses for communications. Use of the numeric address for accessing the web

2

server is avoided because it is easier to remember textual information than that of numeric one. Hence normally the web server's addresses are textual.

(iii) The third part of this address denotes the resource user wants to see.

Step 2 :

- The web browser, on locating the IP Address which it requires (by communicating with the name server), sends a request directly to the web server, using port 80, asking for the file **home.aspx.**

Step 3 :

- The web server sends the html for this page back to user's web browser.
- If there are additional files needed in order to show the web page (like some images, for example) the web browser makes additional requests for each of these.

Refer Fig. 1.8.3.

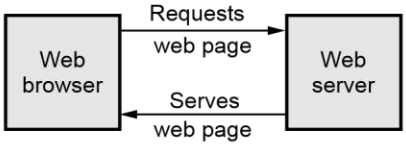


Fig. 1.8.3 Web browser server communication

Basic Features of Web browsers

1. Web browsers should be able to look at the web pages throughout Internet or connect to various sites to **access information.**
2. The Web browser must enable you to follow the hyperlinks on a Web and type in a URL for it to follow.
3. One of the main feature of a browser is to search the information on the current page as well as search the WWW itself.
4. Browser give you the facility to save a Web page in a file on your computer, print a Web page and send the contents of a Web page e-Mail to others on the Internet.
5. Web browser should be able to handle text, images of the World Wide Web, as well as the hyperlinks to digital video, or other types of information.
6. Web browsers interact not just with the Web, but also with your computer's operating system and with other programs, called plug-ins, that gives the browser enhanced features.
7. Another important feature to insist on in your browser is **caching.** A browser that caches keeps of the pages you visit so that it does not have to download them again if you want to return to them. Reloading a page from the cache is much quicker that downloading it again from the original source.
8. The most important feature of any browser is ease of use. While all Web browsers are fundamentally simple to use, it makes user comfortable.

Review Question

1. *Explain the process of client server communication for a web request.*

1.8.4 Comparison among Popular Browsers

Sr. No	Features	Firefox	Chrome	Internet Explorer
1	Fast JavaScript engine for better performance	Yes	Yes	Yes
2	Notification when add-ons slow browser performance	No	No	Yes
3	Simple browsing controls for a better browsing experience	Yes	Yes	Yes
4	Combined search and Address bar	No	Yes	Yes
5	Protection from malicious cross-site scripting attacks	Yes	Yes	Yes
6	Automatic recovery of crashed tabs	Yes	Yes	Yes
7	Compatibility mode to view websites designed for older browsers	No	No	Yes
8	Developer tools built-in to the browser	No	Yes	Yes
9	Reopen accidentally closed tabs	No	Yes	Yes
10	Best protection against phishing attacks	Yes	No	Yes
11	Different Operating System	Windows, Linux, Mac	Windows , Linux, Mac	Windows, Mac

3

1.8.5 HTTP Protocol

- **H**yper **T**ext **T**ransfer **P**rotocol (HTTP) takes part in web browser and web server communication. Hence it is called a **communication protocol.**
- The basic feature of HTTP protocol is that it follows the **request response model.** The client makes a request for desired web page by giving the URL in the address bar. This **request** is submitted to the web server and then web server gives the **response** to the web browser by returning the required web page.

1.8.5.1 HTTP Request Message Structure

The basic structure of request message is given by following general form -

<Start line>
<Header fields>
<Blank Line>
<Message Body>

Let us discuss this structure in detail

Start line

The **start line** consists of three parts which are separated by a single space. These parts are -

1) Request method 2) Request-URI
3) HTTP version

Let us discuss them in detail.

• Request method

The method defines the CONNECT method which is used during the web browser and server communication. It is always written in Upper Case letters. The primary method in HTTP is **GET.** The GET method is used when -

1. You type a URL in address bar.
2. When you click on some hyperlink which is present in the document.
3. When browser downloads images for display within a HTML document.

There is another commonly used method and i.e. POST. The POST method is typically used to send an information collected from a user form. Various methods used by HTTP are as given below -

HTTP method	Description
GET	This method means retrieve the information requested by the user.
	The GET method is used to retrieve information from a specified URI and is assumed to be a safe, repeatable operation by browsers.
POST	The POST method is used to request the server for desired web page and the request made is accepted as a new subordinate of the resource identified.
	The POST method is used for operations that have side effects and cannot be safely repeated. For example, transferring money from one bank account to another has side effects and should not be repeated without explicit approval by the user.
HEAD	The HEAD method is identical to GET. The only difference is that the server should not return a message-body in the response. The meta-information contained in the HTTP headers in response to a HEAD request should be similar to the information sent in response to a GET request.
OPTION	This method supports for the specified URL. It can be used to check the functionality of a web server by requesting '*' instead of a specific resource.
PUT	This method uploads a representation of the specified resource.
DELETE	This method is useful in deleting the specified resource.
TRACE	When request is made using TRACE method the server echoes back the received request so that a client can see what intermediate servers are adding or changing in the request.

• Request URI

The **U**niform **R**esource **I**dentifier (URI) is a string used to identify the names or resources on the Internet. The URI is a combination of URL and URN. The URL stands for **U**niform **R**esource **L**ocator and URN stands for **U**niform **R**esource **N**ame. The web address denotes the URL and specific name of the place or a person or item denotes the URN. For example

urn:ISBN 978-81-8431-123-2

specifies the address of some book.

Every URI consists of two parts, the part before the colon : denotes the scheme and the part after colon depends upon the **scheme.** The URIs are case insensitive but generally written in lower case. If the URI is written in the form of http: then it is both an URI and URL but there are some other URI which can also be used as URL. For example

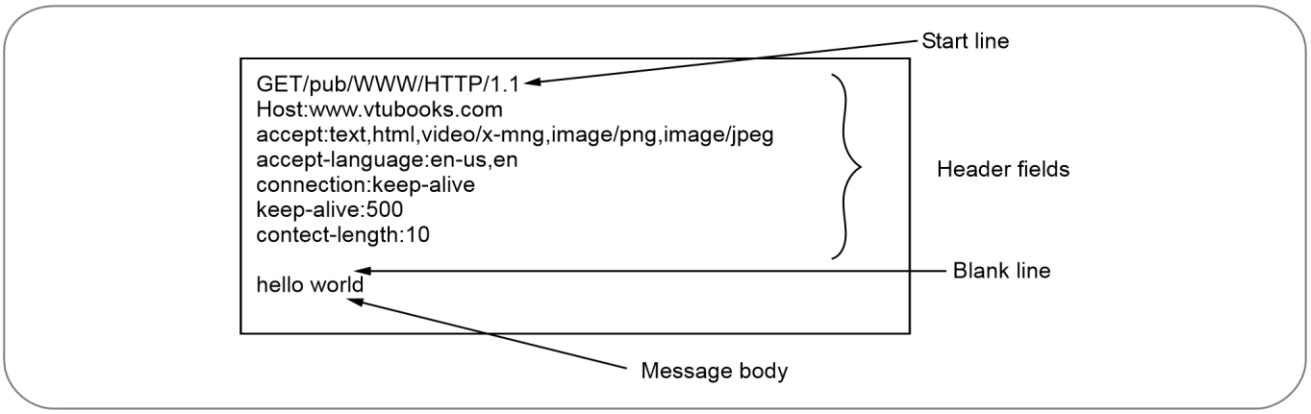


Fig. 1.8.4 HTTP request message structure

URL	Intended server
ftp://ftp.mywebsite.com/index.txt	File can be located on FTP server
telnet://mywebsite.org	Telnet server
mailto:myself@mywebsite.org	Mail box
http://www.mywebsite.com	Web server

- **HTTP version**

The first HTTP version was HTTP/0.9 but the official version of HTTP was HTTP/1.1

Header Fields and message body

The host header filed is associated with the http request. The header fields are in the form of **field name** and **field value.** Thus typical structure of http request is given be following example - Refer Fig. 1.8.4.

1.8.5.2 HTTP Response Message Structure

The structure of response message is similar to the request message structure. It is as follows

<Status line>
<Header fields>
<Blank Line>
<Message Body>

Let us discuss this structure in detail.

Status line

Status line is similar to the start line in the request message. It consists of three fields.

HTTP version	Status code	Reason phrase

The HTTP version denotes the HTTP version such as HTTP/1.1. The **status code** is a numeric code indicating the type of response. The **reason phrase** is in the text string form and presents the information about the status code.

For example -

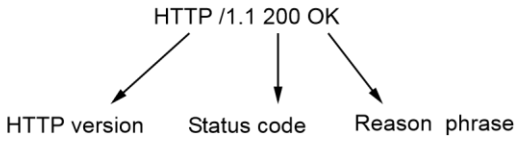


Fig. 1.8.5

Following table explains some commonly used status codes.

Status code	Reason phrase	Description
200	OK	This is a standard response for successful request.
201	Created	It shows that the request is fulfilled and a new resource is being created.
202	Accepted	When the request is accepted for processing but is not processed yet is denoted by this status code.
301	Moved permanently	The URI for requested resource is moved at some another location.
401	Unauthorized	The requested resource is protected by some password and the user has not provided any password.
403	Forbidden	The requested resource is present on the server but the server is not able to respond it.

404	Not Found	The requested resource is not present currently but may be available in future.
500	Internal Server Error	It is a generic error message that appears due to software internal failure.

The **header field** in response message is similar to that of request message. The **message body** consists of response message.

For example

```
HTTP/1.1 200 OK
Date: Fri, 1 Jan 2010 07:59:01 GMT
Server: Apache/2.0.50 (Unix) mod_perl/1.99_10 Perl/v5.8.4
mod_ssl/2.0.50 OpenSSL/0.9.7d DAV/2 PHP/4.3.8
Last-Modified: Mon, 23 Feb 2009 08:32:41 GMT
Accept-Ranges: bytes
Content-Length: 2010
Content-Type: text/html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>...</html>
```

The response header fields are enlisted in following table

Header field	Description
Date	It represents the date and time at which the response is generated.
Server	The name of the server which is responding
Last-Modified	The date and time at which the response is last modified.
Accept-ranges	It specifies the unit which is used by the client to accept the range request. For example if there is a large document and only a single web page is currently needed then this specifies the Accept-range.

1.8.5.3 Cache Control

- Many times the response header fields are used in conjunction with cache control. Cache is used as a repository. Use of cache improves the system performance. Many web browser stores web pages viewed by the client in the cache memory. This brings efficiency in browsing web pages. For instance, client reads a daily News paper on his PC then caching the corresponding web address or web pages will quickly display the web page.

1.8.5.4 HTTP Tunneling

- HTTP tunneling is a mechanism by which the communication performed by various network protocol is encapsulated(wrapped) by the HTTP protocol. HTTP tunneling can be used in the chat like applications for communication from network locations with restricted connectivity(for example if the communicating server is a proxy server then connectivity is restricted).
- The application that wishes to communicate with a remote host opens an HTTP connection to a mediator server. Using HTTP request the host communicates with the mediator server by encapsulating the actual communications within those requests. The mediator server then unwraps the data and sends to the remote destined host. The remote host when sends the response to the requesting host, wraps the response in the HTTP protocol and then the response is given. In this case the application becomes the tunneling client.

1.8.5.5 Features of HTTP Protocol

1. It is a **communication** protocol used between web browser and web server.
2. This protocol is based on **request-response** messaging. That means client makes the request of desired web page and then the server responds it by sending the requested resource.
3. It is a **stateless protocol.** That means HTTP protocol can not remember the previous user's information nor it remembers the number of times the user has visited particular website.
4. The request-response message consists of plain text in fairly **readable form.**
5. The HTTP protocol has a **cache control.** This is an advanced feature of HTTP. Most of the web browsers automatically store(Cache) the recently visited web pages. This is very useful feature because if the user requests the same web page that has been visited already then it can be displayed from the cache memory instead of requesting the web server and bringing it from there.

Review Question

1. Explain the process of client server communication for a web request.

6

1.9 Authoring Tools

- **Definition :** An e-learning content authoring tool is a software package which developers use to create and package e-learning content deliverable to end users. The multimedia authoring tools provide the capability for creating a complete multimedia presentation, including interactive user control.
- Some of the examples of authoring tools are -
  1. Macromedia Flash
  2. Macromedia Director
  3. Author Ware
  4. Quest
- Authoring software provides an integrated environment for combining the content and functions of a project.
- It enables the developer to create, edit, and import data.
- In multimedia authoring systems, multimedia elements and events are considered as **object.**
- Each object is assigned properties and modifiers. On receiving messages, the objects perform tasks depending upon properties and modifiers.

Features of Authoring Tools

1. Programming Features :

- Authoring tools offer the programming using high level languages or support for scripting environment.
- The tools that offer the programming features are Macromedia Flash, HyperCard, MetaCard and ToolBook.
- Some authoring tools offer direct importing of preformatted text, including facilities, complex text search mechanisms and hyper linkage tools.
- Visual authoring tools such as Authorware and IconAuthor are suitable for slide shows and presentations.

2. Interactivity Features

- The interactivity features allow the user to have control over flow of information.
- Using the interactivity features the contents are well organized by the user.
- The conditional branching support the complex programming logic, subroutines, event tracking and message passing among objects and elements.

3. Editing and Organizing Features

- The elements of multimedia - image, animation, text, digital audio and MIDI music and video clips - need to be created, edited and converted to standard file formats and the specialized applications provide these capabilities.
- Editing tools for these elements, particularly text and still images are often included in as authoring tools.
- Some authoring tools provide a visual flowcharting system or overview facility for illustrating your project's structure at a macro level. Storyboards or navigation diagrams too can help organize a project.

4. Delivery Features

- Delivering your project may require building a run-time version of the project using the multimedia authoring software.
- A run-time version allows your project to play back with out requiring the full authoring Multimedia Systems (MMS) software and all its tools and editors. Many times the run time version does not allow user to access or change the content, structure and programming of the project.

5. Cross Platform Features

- By this feature, it is possible to transfer the contents across the platform easily.
- The run time players are available for the providing the compatibility to the authoring tools to work on other platform.

Examples of Authoring Tools

1. **Macromedia Flash :** Adobe Flash Player (formerly Macromedia Flash Player) is a multimedia platform which has become the standard for implementing animation and interactivity into web pages to create ads, integrate video into websites.

2. **HyperCard :** It is a hypermedia program created for Macintosh computer. It combines database abilities with a graphical, flexible, user-modifiable interface. HyperCard also features HyperTalk, a programming language for manipulating data and the user interface.
3. **FrontPage :** It is a Web site administration tool from Microsoft for the Microsoft Windows. FrontPage consists of a Split View option to allow the user to code in Code View and preview in Design View without the hassle of switching from the Design and Code View tabs for each review. Interactive Buttons give users a new easy way to create Web graphics for navigation and links, eliminating the need for a complicated image-editing package
4. **Dreamweaver :** Dreamweaver is a web authoring tool rather than a multimedia authoring tool. It does however support a wide range of multimedia file types. These include graphics formats such as JPEG, GIF and PNG, as well as complied Flash files (SWF). Support exists for embedding other media such as audio and video within HTML or script. A range of interactive elements are pre-scripted as behaviours, including some that can be used for multimedia and interactivity. An extensive range of languages including HTML, XML, ASP, PHP, JSP, JavaScript and VBScript are supported by Dreamweaver.
5. **NetObjects Fusion :** NetObjects Fusionis a Web authoring tool that is the solution for small business Web sites, from planning, building, and managing your site to promoting and growing online business quickly and effectively. One can drag images, text, and other objects anywhere on the page and simply drop them in. The NetObjects Fusion is the first program to remove the tedious hand coding from creating pixel-precise page layouts in HTML.

1.10 Types of Servers

- A Server is a computer or device on a network that manages network resources. Servers are used for a multitude of reasons. For data collection and transmission, for hosting websites and other web client applications such as video games, and streaming.

1.10.1 Application Server

- An application server is a server program in a computer in a distributed network that provides the business logic for an application program.
- It basically provides middleware services for security and state maintenance, along with data access and persistence.
- Sometimes referred to as a type of **middleware.**
- The application server is frequently viewed as part of a three-tier application, consisting of a Graphical User Interface (GUI) server, an application (business logic) server, and a database and transaction server-
  1. A first-tier, front-end, Web browser-based graphical user interface, usually at a personal computer or workstation.
  2. A middle-tier business logic application or set of applications, possibly on a local area network or intranet server.
  3. A third-tier, back-end, database and transaction server, sometimes on a mainframe or large server.
- The Example of Application Servers are :
  - **JBoss :** Open-source server from JBoss community.
  - **Glassfish :** Provided by Sun Microsystem. Now acquired by Oracle.
  - **Weblogic :** Provided by Oracle. It more secured.
  - **Websphere :** Provided by IBM.

1.10.2 Web Server

- Web server is a program that uses HTTP to serve files that create web pages to users in response to their requests, which are forwarded by their computers HTTP connection.
- Always a web server is connected to the internet. Every Web server that connects to the Internet will be provided with a unique address which was arranged with a series of four numbers between 0 and 255 separated by periods.

- There are some popularly used web servers such as Apache and IIS from Microsoft.

Functions of web server

Various functions of web server are -

1. The web servers **accepts the requests** from the web browsers.
2. The user **request is processed** by the web server.
3. The web servers respond to the users by **providing the services** which they demand for over the web browsers.
4. The web servers **serve the web based applications.**
5. The DNS translate the **domain names into the IP addresses.**
6. The servers **verify** given address exists, **find necessary files,** run appropriate scripts exchange cookies if necessary and returns back to the browser.
7. Some servers actively participate in **session handling techniques.**

1.10.2.1 Apache

- It is an excellent server because of its two important features : **Reliability** and **Efficiency.**
- Secondly it is more popular because it is an open source software. That means it is freely available to anybody. Apache web server is best suitable for UNIX systems but it can also be used on Windows box. The apache web server can be configured as per the requirements using the file **httpd.conf.** This file is present in the Apache software package.

1.10.2.2 IIS

- The Internet Information Services or Internet Information Server is a kind of web server provided by Microsoft. This server is most popular on Windows platform.
- Following are some differences between Apache and IIS servers

Sr. No.	Apache web server	IIS web server
1.	Apache web server is useful on both Unix based systems and on Windows platform.	IIS web server is used on Windows platform.
2.	It is an open source product.	It is a vendor specific product and can be used on windows products only.
3.	The Apache web server can be controlled by editing the configuration file *httpd.conf*	For IIS server, the behaviour is controlled by modifying the window based management programs called *IIS snap in.* We can access IIS snap-in through the Control-Panel->Administrative Tools.

1.10.3 Database Server

- The term database server may refer to both hardware and software used to run a database, according to the context.
- As software, a database server is the back-end portion of a database application, following the traditional client-server model. This back-end portion is sometimes called the instance.
- It may also refer to the physical computer used to host the database. When mentioned in this context, the database server is typically a dedicated higher-end computer that hosts the database.
- When considering databases in the client-server model, the database server may be the back-end of the database application (the instance), or it may be the hardware computer that hosts the instance. Sometimes, it may even refer to the combination of both hardware and software.
- The database server holds the Database Management System (DBMS) and the databases. Upon requests from the client machines, it searches the database for selected records and passes them back over the network.
- A database server is useful for organizations that have a lot of data to deal with on a regular basis. If you have client-server architecture where the clients

9

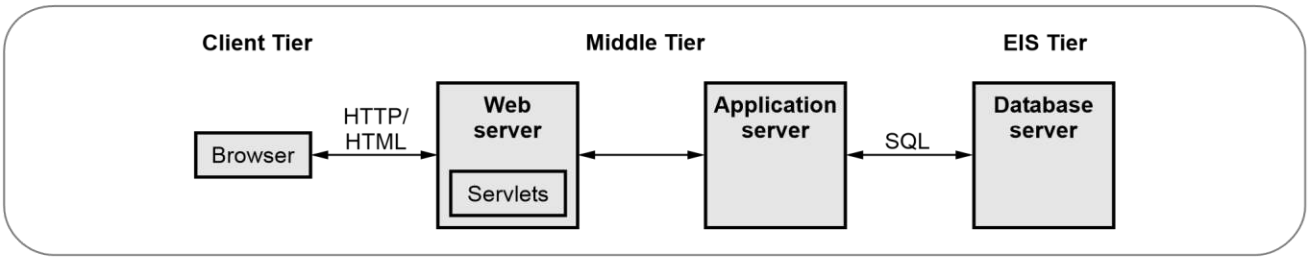


Fig. 1.10.1 Server architecture

need process data too frequently, it is better to work with a database server.

- All database functions are controlled by the database server. Any type of computer can be used as database server. It may be microcomputer, minicomputer or mainframe computer. In large organization networks, the mainframe computers are used as server.
- The Database server manages the recovery security services of the DBMS.
- It provides concurrent access control. It provides better security and server hides the DBMS from clients. It provides the multi-user environment. Several users can access the database simultaneously. All the data is stored on the data server therefore, the DBA can easily create the backup of the database.
- Examples of proprietary database servers include Oracle, DB2, Informix, and Microsoft SQL Server. Examples of free software database servers include PostgreSQL. (Refer Fig. 1.10.1.)

Comparison among various types of servers

Application Server	Web Server
A server that exposes business logic to client applications through various protocols including HTTP.	A server that handles HTTP protocol.
Application server is used to serve web based applications and enterprise based applications(i.e servlets, JSP and EJB etc). Application servers may contain a web server internally.	Web server is used to serve web based applications.(i.e servlets and JSP)
To deliver various applications to another device, it allows everyone in the network to run software off of the same machine.	Keeping HTML, PHP, ASP, etc files available for the web browsers to view when a user accesses the site on the web, handles HTTP requests from clients.
It makes use of distributed transaction and EJB's	It makes use of Servlets and JSP
Resource utilization is high	Resource utilization is low.

Web Server	Database Server
Web server makes use of the languages like PHP, ASP, JSP. It makes use of the protocols such as FTP and HTTP.	The database server has its own specific program language or query language.
Web server is used to save the static and dynamic content and pages of websites.	Database server deals with the storing and managing the data of a computer or computer programs
Web server only performs web based services.	Database server can manage the web based, enterprise based or business based services at the same time.
Apache HTTP Server, Microsoft Internet Information Services (IIS), Google Web Server (GWS) and Sun Java System Web Server are examples of web server.	Oracle, SAP, MySQL and DB2 are some common examples of database server.

Review Question

1. *Compare and contrast application server and database server.*

Two Marks Questions with Answers

Q.1 *Define the term Web site*

Ans. : Web Site is a collection of web pages that are grouped together to achieve certain task under single domain name.

Q.2 *Explain the reasons for which people tend to visit the web site.*

Ans. : People visit the web site because -

1. To get the required information

2. To perform buying and selling of products, to download software programs or to participate in online discussions.

Q.3 *Enlist the steps for creating the web sites*

Ans. :

1. Creation of web pages for web site.
2. Choosing web hosting services.
3. Registering Domain Name
4. Planning the web site
5. Uploading Files
6. Testing Web site

Q.4 *What are the issues to be considered while building a web site for common user ?*

Ans. : The issues to be considered while building web site for common user are -

1. Simplicity
2. Identity
3. Consistency
4. Robustness
5. Navigability
6. Visual Appeal
7. Compatibility

Q.5 *List desirable features that need to be considered while designing a web site*

Ans. : The features that need to be considered while designing a web site -

1. Quality Web Content
2. Clear, User-friendly Navigation
3. Simple and Professional Web Design
4. Webpage Speed
5. Search Engine Optimization
6. Web Compatibility

Q.6 *What are the factors that need to be considered while testing the web site?*

Ans. : The web site must be tested on -

1. Multiple browsers to ensure that the contents can be displayed consistently.
2. Multiple operating system
3. Different connection speed.

Q.7 *List any four authoring tools and highlight the importance of each tool ?*

Ans. : Some Authoring tools are -

1. Macromedia Flash
2. HyperCard
3. FrontPage
4. Dreamweaver
5. NetObjects Fusion

Q.8 *What is the use of IP address ?*

Ans. : IP address is assigned to the devices participating in computer network. The IP protocol makes use of this address for communication between two computers. Using IP address particular node can be identified in the network.

Q.9 *What is URL ?*

Ans. : The **U**niform **R**esource **L**ocator (URL) is unique address for the file that has to be accessed over the internet. When we want to access some website we enter it's URL in the address bar of the web browser. For example www.google.com

Q.10 *What is web browser ?*

Ans. : **Web browser** is a kind of software which is basically used to use resources on the web. For example - Mozilla Firefox, Internet Explorer, Chrome.

Q.11 *Enlist any two functions of web browser.*

Ans. : Various functions of web browser are -

1. Reformat the URL and send a valid HTTP request.
2. When user gives the address of particular web site it is in the form of domain name. The web browser coverts the DNS to corresponding IP address.
3. The web browser establishes a TCP connection with the Web browser while processing the user's request.

1

Q.12 *What is the function of rendering engine ?*

Ans. :

1) It is responsible for displaying the requested contents on the screen.

2) The rendering engine interprets the HTML, XML and JavaScript that comprises the given URL and generates the layout that is displayed in the user interface.

Q.13 *What is HTTP ?*

Ans. : Hyper Text Transfer Protocol (HTTP) takes part in web browser and web server communication. Hence it is called a communication protocol.

Q.14 *What is GET and POST request ?*

Ans. : **GET -** This method means retrieve the information requested by the user.The GET method is used to retrieve information from a specified URI and is assumed to be a safe, repeatable operation by browsers.

POST - The POST method is used to request the server for desired web page and the request made is accepted as a new subordinate of the resource identified. The POST method is used for operations that have side effects and cannot be safely repeated. For example, transferring money from one bank account to another has side effects and should not be repeated without explicit approval by the user.

Q.15 *What is web browser ?*

Ans. : **Web browser** is a kind of software which is basically used to use resources on the web.

For example - Mozilla Firefox, Google Chrome are some popularly used web browsers.

Q.16 *Enlist some functions of web browser*

Ans. : Various functions of web browser are -

1. Reformat the URL and send a valid HTTP request.
2. When user gives the address of particular web site it is in the form of domain name. The web browser coverts the DNS to corresponding IP address.
3. The web browser establishes a TCP connection with the Web browser while processing the user's request.
4. The web browsers send the HTTP request to the web server.
5. The web server processes the HTTP request sent by the web browser and returns the desired web page to the client machine. The web browser on the client's machine displays this web page in appropriate format.

Q.17 *List out few ways you can reduce page load time ?*

Ans. : Following things can be done to reduce the page load time -

1. Reduce image size
2. Remove unnecessary widgets
3. Minimize redirects
4. Caching

Q.18 *What is HTTP protocol ?*

Ans. : • **H**yper **T**ext Transfer **P**rotocol (HTTP) takes part in web browser and web server communication. Hence it is called a **communication protocol.**

- The basic feature of HTTP protocol is that it follows the **request response model.** The client makes a request for desired web page by giving the URL in the address bar. This **request** is submitted to the web server and then web server gives the **response** to the web browser by returning the required web page.

Q.19 *What is HTTP tunnelling ?*

Ans. : HTTP tunneling is a mechanism by which the communication performed by various network protocol is encapsulated(wrapped) by the HTTP protocol. HTTP tunneling can be used in the chat like applications for communication from network locations with restricted connectivity(for example if the communicating server is a proxy server then connectivity is restricted).

Q.20 *What is Cache Control feature of HTTP ?*

Ans. : This is an advanced feature of HTTP. Most of the web browsers automatically store(Cache) the recently visited web pages. This is very useful feature because if the user requests the same web page that has been visited already then it can be displayed from the cache memory instead of requesting the web server and bringing it from there.

2

Q.21 *Why HTTP is called stateless protocol ?*

Ans. : HTTP protocol can not remember the previous user's information nor it remembers the number of times the user has visited particular website. Hence it is called stateless protocol.

Q.22 *What is authoring tool ?*

Ans. : Authoring tool is a software package which developers use to create and package e-learning content deliverable to end users. The multimedia authoring tools provide the capability for creating a complete multimedia presentation, including interactive user control.

Some of the examples of authoring tools are -

1. Macromedia Flash 2. Macromedia Director
3. Author Ware 4. Quest

Q.23 *Enlist some desirable features of authoring tool*

Ans. :

1. **Programming Features :** Authoring tools offer the programming using high level languages or support for scripting environment.
2. **Interactivity Features :** The interactivity features allow the user to have control over flow of information. Using the interactivity features the contents are well organized by the user.
3. **Editing and Organizing Features :** The elements of multimedia - image, animation, text, digital audio and MIDI music and video clips - need to be created, edited and converted to standard file formats and the specialized applications provide these capabilities.
4. **Delivery Features :** Delivering your project may require building a run-time version of the project using the multimedia authoring software.
5. **Cross Platform Features :** By this feature, it is possible to transfer the contents across the platform easily.

Q.24 *What is server ?*

Ans. : A Server is a computer or device on a network that manages network resources. Servers are used for a multitude of reasons. For data collection and transmission, for hosting websites and other web client applications such as video games, and streaming.

Q.25 *What is web server ?*

Ans. : Web server is a program that uses HTTP to serve files that create web pages to users in response to their requests, which are forwarded by their computers HTTP connection.

Q.26 *Mention what are some bad examples of web design ?*

Ans. :

1. Blinking, spinning or flashing images
2. Black background with white, light or pale text
3. Black backgrounds with dark text
4. Busy tiled background images with any color text
5. Everything Centered
6. Too many images or Huge images
7. List of links
8. Too many headlines or Blinking text

Q.27 *What is the importance of Authoring tools ?*

Ans. : **1) Time saving :** The use of authoring tool reduces substantial amount of development time.

2) Easy to use : Authoring tools are relatively easy to use when compared to traditional programming languages.

3) Accessible anytime, anywhere : The user does not need to rely on anything to access the authoring tools. Once installed on your desktop or through the Web, these tools can be used by anyone at any point to create interactive eLearning content.

4) Effective : The use of authoring tool is very effective in educational field to explain educational concepts to the students.

❑❑❑

2.1 Need for Scripting Languages

Definition Scripting Language : A scripting language is a programming language designed for integrating and communicating with other programming languages. Some of the most widely used scripting languages are HTML, JavaScript, VBScript, PHP, Perl, Python, Ruby, ASP and so on.

- In general, scripting languages are **easier to learn and faster** to code in than more structured and compiled languages such as C and C++.
- Scripting languages are useful tool for developing **interactive web pages** with minimum efforts.
- Scripting languages are often **interpreted** (rather than compiled).
- The scripting languages are useful for **producing dynamic web contents.** That means web page can be changed using user input.

Advantages of Scripting Languages

Following are advantages of scripting languages -

1. Scripting languages are **easy to learn.**
2. It requires **minimum** programming **knowledge** or **experience** to develop the web pages using scripting languages.
3. The scripting languages allow **simple creation** and editing in variety of text editors.
4. Using scripting languages we can develop **dynamic** and **interactive web pages.**
5. There are some scripting languages that **validate** the **information** entered by the user.

2.2 Types of Scripting Languages

There are two types of scripting languages - Client side scripting languages and server side scripting languages.

Client Side Scripting Languages -

The client side scripting is used to create the web pages as a request or response to server. These pages are displayed to the user on web browser.

For example - HTML, CSS, JavaScript, PHP.

Server Side Scripting Languages -

Server Side Scripting Languages -The server side scripting is used to create the web pages that provide some services.These scripts generally run on web servers.

For example - ASP, JSP, Servlet, PHP

Difference between Client Side Scripting and Server Side Scripting

Server Side Scripting	Client Side Scripting
The server side scripting is used to create the web pages that provide some services.	The client side scripting is used to create the web pages as a request or response to server. These pages are displayed to the user on web browser.
These scripts generally run on web servers.	These scripts generally run on web browser.
A user's request is fulfilled by running a script directly on the web server to generate dynamic HTML pages. This HTML is then sent to the client browser.	The processing of these scripts takes place on the end users computer. The source code is transferred from the web server to the users computer over the internet and run directly in the browser.
Uses : Processing of user request, accessing to databases.	**Uses :** Making interactive web pages, for interacting with temporary storages such as cookies or local storage, sending request to server and getting the response and displaying that response in web browser.
Example : PHP, ASP.NET, nearly all the programming languages including C++, Java and C#	**Example :** HTML, CSS, JavaScript(primarily)

2.2.1 Examples based on Scripting Languages

Ex. 2.2.1 : *Write a JavaScript program to count the number of unique alphabets present in the given string*

Sol. :

```
<!DOCTYPE html>
<html>
<head>
  <title>Finding number of alphabets in String</title>
</head>
<body>
<strong>
   <script type="text/javascript">
```

3

```
    var s1="Hello";
   document.write("Given string is : "+s1+"<br>");
    document.write("Number of alphabets in given string
                    are : "+s1.length+"<br>");
  </script>
 </strong>
</body>
</html>
```

Output

Ex. 2.2.2 : *Create a website for a online super market. Your web site should have a homepage which helps the users to navigate to various other pages. Every web page in the web site gives a detailed description of different items present in the super market. Make the website user friendly by adding relevant images and other formatting options. The last web page should present a feedback form for the customer.*

Sol. :

Step 1 : The home page can be created as follows -

main.html

```
<!DOCTYPE html>
<html>
  <frameset rows="40%,60%">
    <frame src="header.html">
     <frameset cols="40%,60%">
       <frame src="menu.html">
       <frame src="contents.html" name="description">
    </frameset>
  </frameset>
</html>
```

Step 2 : The header HTML document will be

header.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="magenta">
<center>
<table>
<tr><td><h3>ONLINE SUPER
MARKET</h3></center></td></tr>
<tr><td><img src="header1.jpg"/></td></tr>
</center>
</body>
</html>
```

Step 3 : The meu.html file contains the services available on the online super market. This document contains basically the hyperlinks for these services.

Menu.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="aqua">
<h3><a href="food.html" target="description">Food and
Vegetables<a/></h3>
<h3><a href="Health.html" target="description">Health
and Beauty<a/></h3>
<h3><a href="Grocery.html" target="description">
Grocery and Staples<a/></h3>
<h3><a href="offers.html" target="description">See All
Offers<a/></h3>
<h3><a href="feedback.html"
target="description">Feedback Form<a/></h3>
</body>
</html>
```

Step 4 : Now when we click on these links the corresponding web pages will be opened up. For example if we click on Food and Vegetable link, then the page containing list of all the food and vegetable items will be displayed along with its price,weight, and quantity options. Similarly other web links open up corresponding web pages. The sample HTML documents for **Food and Vegetables** and **Feedback Form** links are as given below

food.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="aqua">
<h3>Food and Vegetables<a/></h3>
<form name="form1" action="submitForm.php">
<table>
<tr><td><img src="tomato.jpg"/></td>
<td><img src="carrot.jpg"/></td>
<td><img src="cauliflower.jpg"/></td>
<tr><td><b>MRP: RS. 10
```

```
</b></td><td><b>MRP: RS. 20 </b></td><td><b>MRP: RS. 30 </b></td></tr>
<tr>
<td><select name="MyMenu1">
   <option value="250 gm">250 gm</option>
   <option value="500 gm">500 gm</option>
   <option value="1 kg">1 kg</option>
   </select>
</td>
<td><select name="MyMenu2">
   <option value="250 gm">250 gm</option>
   <option value="500 gm">500 gm</option>
   <option value="1 kg">1 kg</option>
   </select>
</td>
<td><select name="MyMenu3">
   <option value="250 gm">250 gm</option>
   <option value="500 gm">500 gm</option>
   <option value="1 kg">1 kg</option>
   </select>
</td>
</tr>
<tr>
<td><input type="text" name="qty1" value="" size="5"/>Qty</td>
<td><input type="text" name="qty2" value="" size="5"/>Qty</td>
<td><input type="text" name="qty3" value="" size="5"/>Qty</td>
</tr>
<tr>
<td><input type="button" name="button1" value="Add to Basket"/></td>
<td><input type="button" name="button2" value="Add to Basket"/></td>
<td><input type="button" name="button2" value="Add to Basket"/></td>
</tr>
<tr></tr><tr></tr><tr></tr><tr></tr><tr></tr><tr></tr>
<tr><td></td><td><input type="submit"name="submitFood" value="Submit"/> </td></tr>
</table>
</form>
</body>
</html>
```

feedback.html

```
<html>
<head><title>FeedBack Form</title></head>
<body bgcolor="aqua">
<h3><center>Feedback Form</center></h3>
<form name="form1" action="HandleFeedback.php">
<table>
<tr>
<td>Name: </td><td>
<input type="text" name="name" value="" s/></td>
</tr>
<tr>
```

```
<td>Email: </td><td>
<input type="text" name="email" value="" s/></td>
</tr>
<tr>
<td>Mobile Number: </td><td>
<input type="text" name="mobileno" value="" s/></td>
</tr>
<tr><td>Feedback:</td>
<td><textarea></textarea></td>
</tr>
<tr><td></td><td>
<input type="submit" name="submitFeedback" value="Submit"/></td></tr>
</table>
</form>
</body>
</html>
```

The sample output is as shown below -

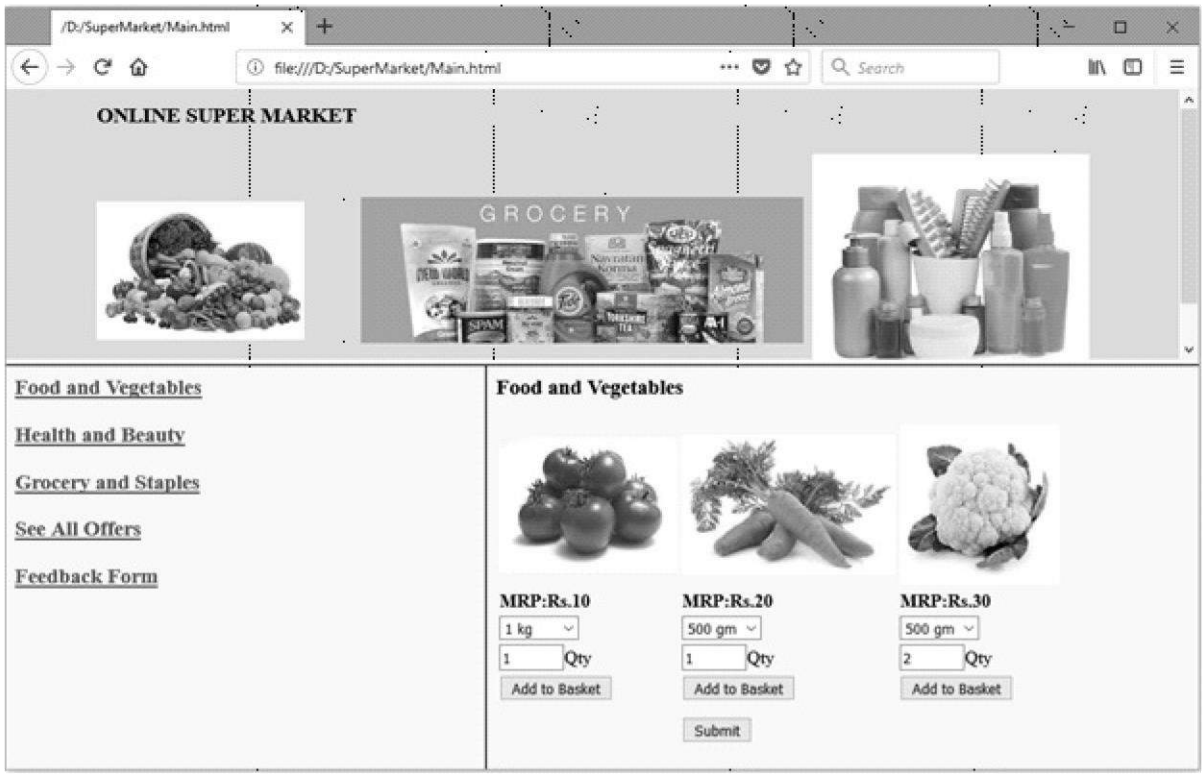


Fig. 2.2.1

Ex. 2.2.3 : *Using DIV element, write HTML code to design a web site layout for online movie booking. The top division provides the details about theatre. The next left division provides the links such as current movies and booking. The current movie link, when clicked should display the list of movies currently screened(in drop down list box) and when the movie name is selected from the list box, the corresponding show timings with availability status(checked and unchecked) must be displayed. The booking link, when clicked must display a form(movie name - dropdown list box, show timings - minimum two radio buttons, number of tickets - text box, email- textbox). Write JavaScript to validate email ID.*

Sol. :

Step 1 : we will create the main layout page as follows

Layout.html

```
<!DOCTYPE html>
<html>

  <head>
 <script src="http://code.jquery.com/jquery-1.4.min.js"></script>
 <script type="text/javascript">
     $(document).ready(function()
     {
     $("#display_movies").click(function(){   //on clicking the link current movie in left div block
        $('#right').load('movie.html');    //current movie list is displayed on right div block
     });

     $("#book_ticket").click(function(){  //on clicking the link bookings in left div block
      $('#right').load('bookings.html');   //the booking form will be displayed on right div block

     });
   });
</script>
  </head>

  <body>
  <div style = "background-color:LightPink; width:100%">
         <center><h1>CITY PRIDE MULTIPLEX</h1></center>
         <center><h4>Address: AAA, BBB, CCC</h4></center>
    <br/>
   </div>

    <div style="width:100%">
 <div id="left" style = "background-color:LightBlue;float:left; height:500px;width: 25%;">
     <h3><a id="display_movies" href="#"> Current Movie</a></h3>
     <h3><a id="book_ticket" href="#"> Bookings</a></h3>
      </div>
      <div id = "right" style = "background-color:LightYellow;float:left; width: 75%;">
 <center><img src="movie_Image.jpg"></center>
     </div>
</div>
</body>
</html>
```

The screenshot for the above code can be as follows -

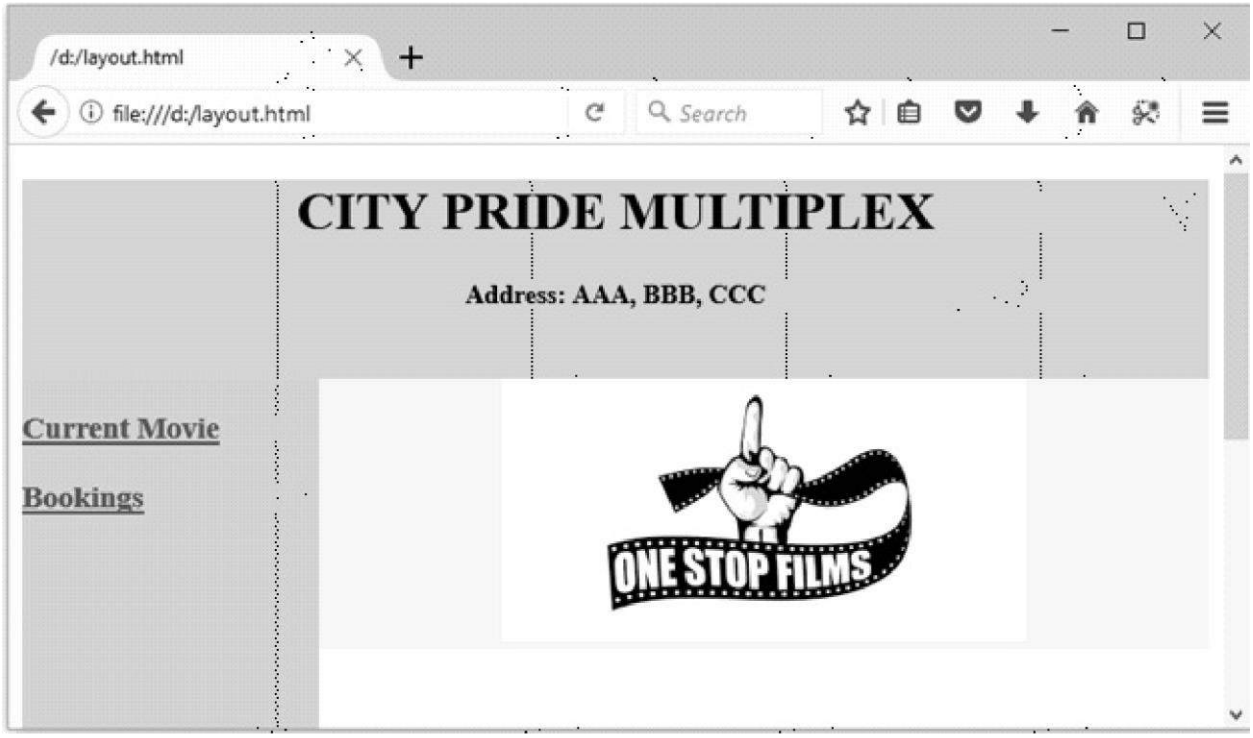


Fig. 2.2.2

Step 2 : If user clicks the **current Movie** link then the name of the current movies along with availability status and timings will be displayed. The code for this link will be as follows -

movie.html

```
<!DOCTYPE html>
<head>
<title>Current Movie</title>
<script type="text/javascript">
   function ShowMovies()
   {
   var current = document.getElementById("currentmovie");
      var status1 = document.getElementById("status1");
      var status2 = document.getElementById("status2");
    if(current.value=="1")
    {
        status1.style.display ="block";

    }
    else
    {
        status1.style.display ="none";
    }
    if(current.value=="2")
    {

        status2.style.display ="block";

    }
    else
    {
        status2.style.display ="none";
    }


   }
</script>
</head>
<body>
    <h3>Current Movie Status</h3>
    <select id = "currentmovie" onchange =
"ShowMovies()">
       <option value="N">choose Movie</option>
       <option value="1">XYZ</option>
       <option value="2">PQR</option>
    </select>
    <hr />
```

```
  <div id="status1" style="display: none"><h3> Availability(Checked box) Status for Movie XYZ is as follows
...</h3>
      <form >
  <input type="checkbox" name="timings" > 9 am to 12 pm<br>
      <input type="checkbox" name="timings" checked> 12 pm to 3 pm<br>
      <input type="checkbox" name="timings" checked> 3 pm to 6 pm<br>
      <input type="checkbox" name="timings" > 6 pm to 9 pm<br>
      <input type="checkbox" name="timings" checked> 9 pm to 12 am<br>
  </form>
  </div>
  <div id="status2" style="display: none"><h3> Availability(Checked box) Status for Movie PQR is as follows
...</h3>
      <form >
  <input type="checkbox" name="timings" checked>9 am to 12 pm<br>
  <input type="checkbox" name="timings" checked> 12 pm to 3 pm<br>
  <input type="checkbox" name="timings" checked> 3 pm to 6 pm<br>
  <input type="checkbox" name="timings" checked> 6 pm to 9 pm<br>
  <input type="checkbox" name="timings" checked> 9 pm to 12 am<br>
  </form>
  </div>
</body>
</html>
```

The screen shot for execution of above HTML code will be as follows -

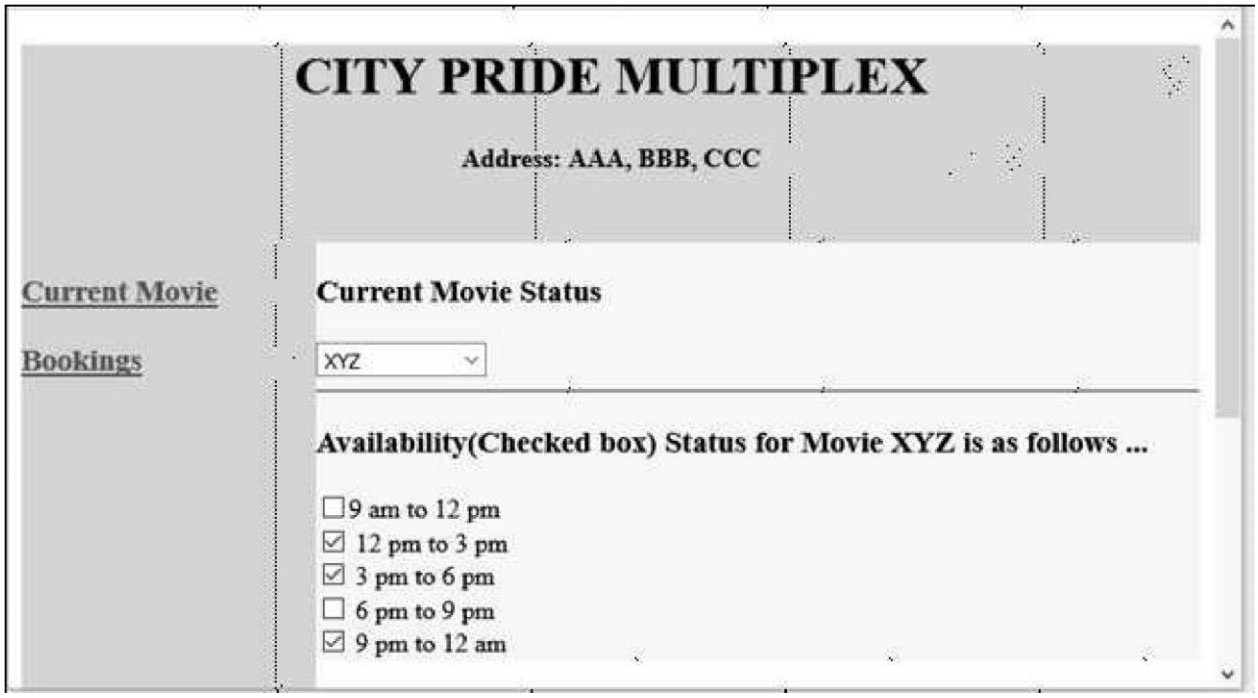


Fig. 2.2.3

Step 3 : If we click the **Bookings** link on the left pane, then we get the booking form. The email validation is done when user submits the form. The HTML document code is as follows -

bookings.html

```
<!DOCTYPE html>
<html>
<head><title> Booking Form </title>
    <script type=text/javascript>
    function ValidateEmail()
    {
    var str=document.myform.Email.value;
    var index_at=str.indexOf("@")
    var len=str.length;
    var index_dot=str.indexOf(".")
    var emailID=document.myform.Email;
    if ((emailID.value==null)||(emailID.value==""))
    {
        alert("Please Enter your Email ID")
        emailID.focus()
        return false
    }
    if (str.indexOf("@")==-1)
    {
       alert("Invalid E-mail ID")
       return false
    }
    if (str.indexOf(".")==-1 || str.indexOf(".")==0 ||
str.indexOf(".")==index_at)
    {
        alert("Invalid E-mail ID")
        return false
    }
    if (str.indexOf("@",(index_at+1))!=-1)
    {
        alert("Invalid E-mail ID")
        return false
    }
    if (str.indexOf(" ")!=-1)
    {
        alert("Invalid E-mail ID")
        return false
    }
    return true;
    }
    </script>
</head>
<body>
    <form name="myform" onsubmit="return
    ValidateEmail();">
    <h3><center>Book Your Tickets Here...
    </center></h3>
    <select id = "currentmovie" onchange =
    "ShowMovies()">
        <option value="N">choose Movie</option>
        <option value="1">XYZ</option>
        <option value="2">PQR</option>
    </select>
    <hr />
    <p>
        Show Timings:<br/>
        <input type="radio" name="time"> 9am to 12pm
    <br/>
        <input type="radio" name="time"> 12pm to 3pm
    <br/>
        <input type="radio" name="time"> 3am to 6pm
    <br/>
        <input type="radio" name="time"> 6pm to 9pm
    <br/>
        <input type="radio" name="time"> 9pm to 12am
    <br/>
        <br/>
    </p>
    <p>
        Number of Tickets:
        <input type="text" name="nooftickets" size ="5">
    </p>
    <p>
        Email Address
        <input type="text" name="Email" size ="30">
    </p>
    <br/>
    <p>
        <input type="submit" value="Submit">
    </p>
    </form>
</body>
</html>
```

The output of the above code will be -

0

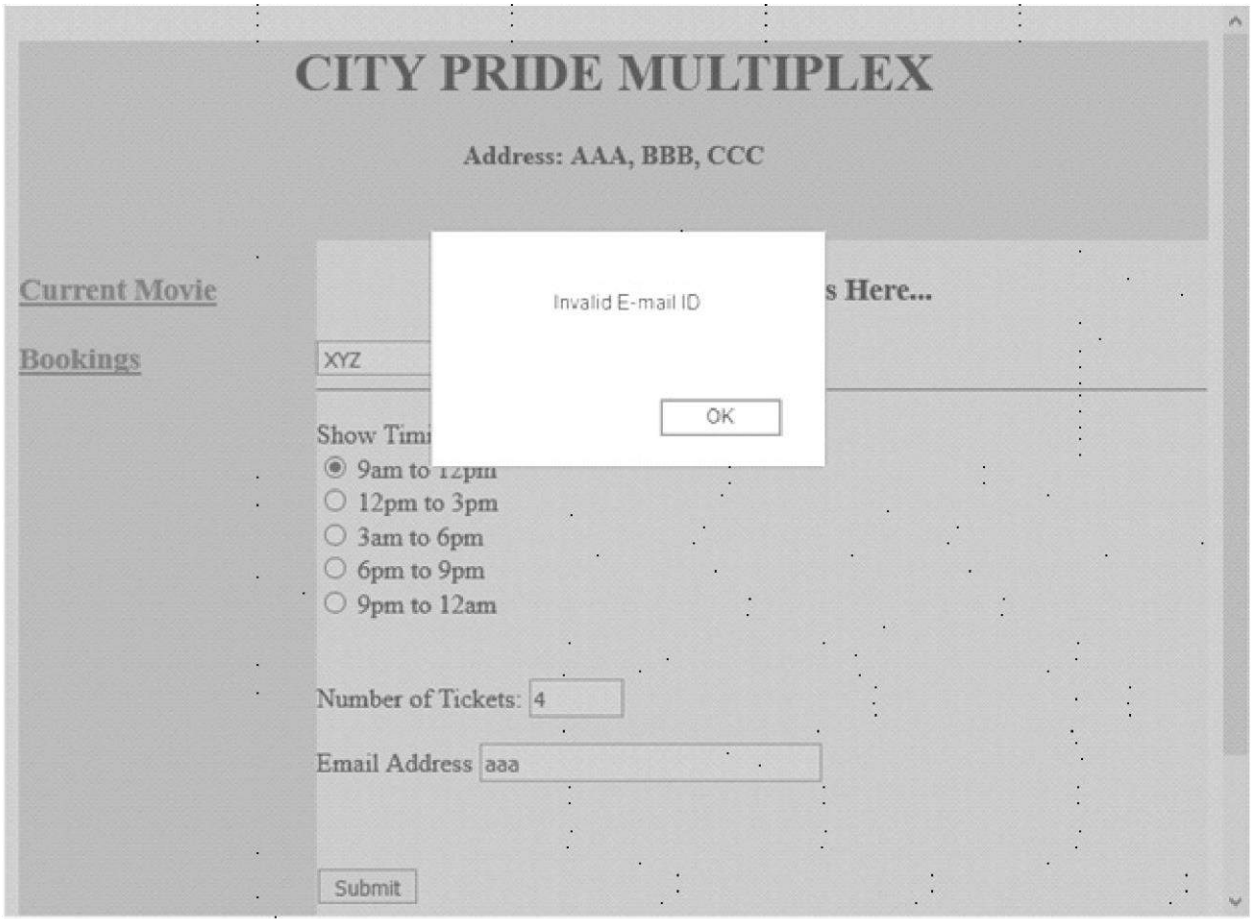


Fig. 2.2.4

Ex. 2.2.4 : *Using DIV element, write HTML code to design a web site for a online catering service.The web site should provide some highlighting facilities provided by them. The web site must provide the available menu and cost-of each menu item in a tabular form. The web site must also accept the order form which includes any number of items and quantity required. Write java script to validate the form and to compute the total amount to be paid. Draw the layout of your web site first and write HTML code for each page.*

Sol. : Step 1 :

ONLINE CATERING SERVICE		
Main Course	Features	Contact Us
---	---	
---	---	
---	---	
@copyright information		

The DIV element is used to create different blocks on the web page. These blocks can be distinguished by setting different background colors, width and height. Using <div style> tag we can set different blocks. The HTML code for main page of the web site will be

```
<!DOCTYPE html>
<html>

  <head>
    <title>Catering Service Demo</title>
  </head>

  <body>
    <div style = "width:100%">

   <div style = "background-color:#b5dcb3; width:100%">
    <center><h1>ONLINE CATERING
    SERVICES</h1></center>
      </div>
```

1

```
        <div style = "background-color:#aaa; height:300px; width:200px;float:left;">
          <div><h3>Main Course</h3></div>
          <a href="south.html">South Indian<br /></a>
          <a href="punjab.html">Punjabi<br /></a>
          <a href="maharashtra.html">Maharashtrian..</a>
        </div>

        <div style = "background-color:#eee; height:300px; width:400px;float:left;" >
  <h4>
    <li>India's first Online Service at your Doorstep.</li>
       <li>Restaurant like Experience for your Private Setting</li>
       <li>Variety of Cuisines</li>
    <li>Convenience a Click Away</li>
     </h4>
  </div>
   <div style = "background-color:#aaa; height:300px; width:100px;float:left;">
        <div><h3>Contact Us</h3></div>
    <h4>
        <a href="address.html">Address</a><br />
        <a href="phone.html">Phone</a><br />
        <a href="mail.html">E-mail...</a>
    </h4>
      </div>
   <div style = "background-color:#b5dcb3; clear:both">
        <center>
          Copyright ©2017 Tasteeaffair.com
        </center>
      </div>

    </div>
   </body>

</html>
```


Catering Service Demo
file:///D:/test.html
ONLINE CATERING SERVICES
Main Course
South Indian
Punjabi
Maharashtrian..
India's first Online Service at your Doorstep.
Restaurant like Experience for your Private Setting
Variety of Cuisines
Convenience a Click Away
Contact Us
Address
Phone
E-mail...
Copyright ©2017 Tasteeaffair.com

Fig. 2.2.5

Step 2 : Main Course block on the above web page allows the user to select, different courses. For instance - if we click on the **South Indian** link from **Main Course,** then the form will be displayed to the user on which user can enter his/her personal details and place the order.

Now we can create a form, using which the user can place the order of items by specifying number of items or quantity. The total bill will be calculated. All the form fields will be validated and Bill is computed using JavaScript

The code for this feature will be

south.html

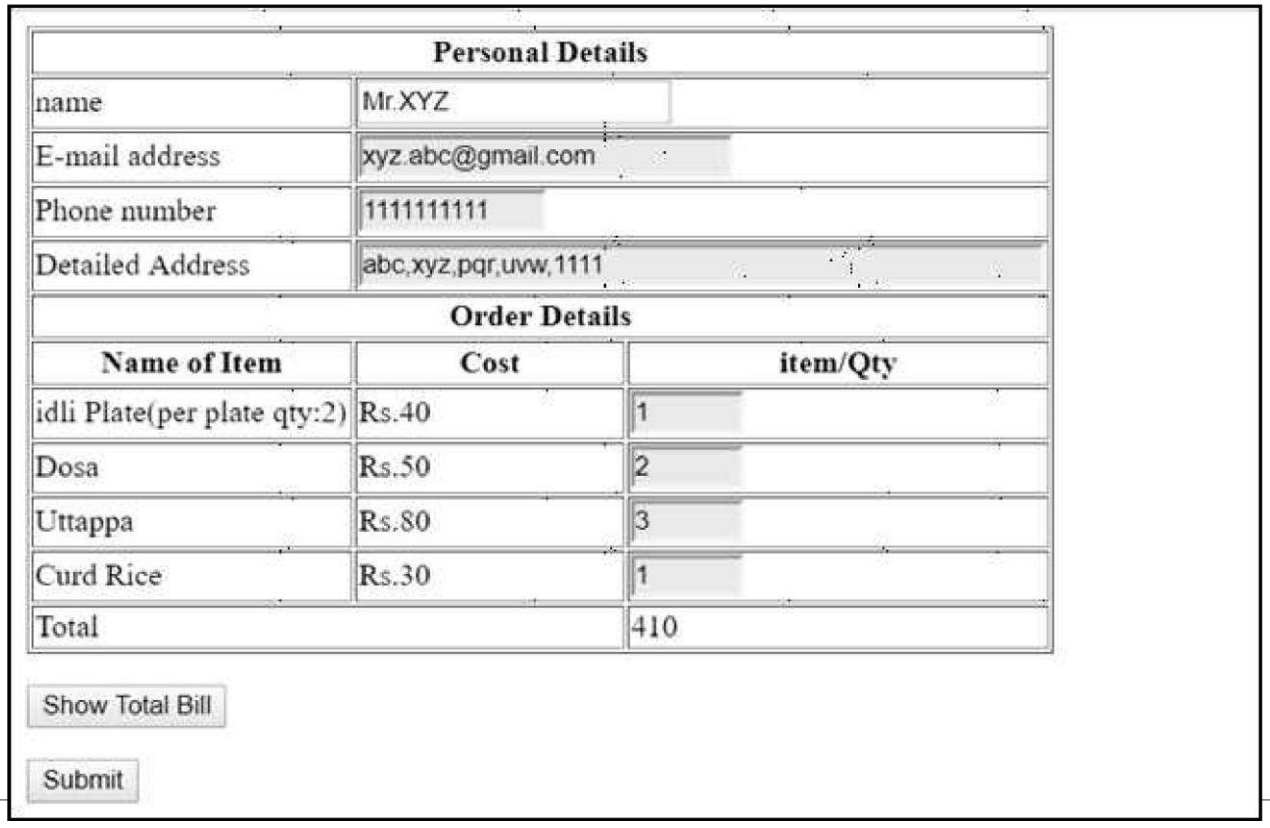
```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        function validateForm()
        {
            if( document.myForm.name.value == "" )
            {
                alert( "Please provide your name!" );
                document.myForm.name.focus() ;
                return false;
            }
            if( document.myForm.email.value == "" )
            {
                alert( "Please provide your Email!" );
                document.myForm.email.focus() ;
                return false;
            }
            if( document.myForm.phone.value == "" ||
                isNaN( document.myForm.phone.value ) ||
                document.myForm.phone.value.length != 10 )
            {
                alert("Please provide a valid phone number");
                document.myForm.phone.focus() ;
                return false;
            }
            return( true );
            }
        function calculateBill(val)
        {
            var item1,item2,item3,item4,bill;
            item1=item2=item3=item4=bill=0;
            if(document.myForm.item1.value !="")
            {
            item1=Number(document.myForm.item1.value)*40;
                }
                if(document.myForm.item2.value !="")
                {
            item2=Number(document.myForm.item2.value)*50;
                }
                if(document.myForm.item3.value !="")
                {
            item3=Number(document.myForm.item3.value)*80;
                }
                if(document.myForm.item4.value !="")
                {
            item4=Number(document.myForm.item4.value)*30;
                }
                bill=item1+item2+item3+item4;
        document.getElementById('myBill').innerHTML = bill;
            }
        </script>
</head>
<body>
    <form name="myForm" method="post"
onsubmit="return validateForm()">
    <table border="1">
    <tr>
        <th colspan="3">Personal Details</th>
    </tr>
    <tr>
        <td>name</td>
        <td colspan="2">
                <input type="text" name="name" size="20">
        </td>
    </tr>
    <tr>
        <td>E-mail address</td>
    <td colspan="2"><input type="text" name="email"
    size="25"></td>
    </tr>
    <tr>
        <td>Phone number</td>
        <td colspan="2"><input type="text" name="phone"
        size="10"></td>
    </tr>
    <tr>
        <td>Detailed Address</td>
        <td colspan="2"><input type="text"
name="address" size="50"></td>
    </tr>
    <tr>
        <th colspan="3">Order Details</th>
```

3

```
    </tr>
    <tr>
        <th>Name of
Item</th><th>Cost</th><th>item/Qty</th>
    </tr>
    <tr>
        <td>idli Plate(per plate qty:2)</td>
        <td>Rs.40</td>
        <td><input type="text" name="item1"
size="4"></td>
    </tr>
    <tr>
        <td>Dosa</td>
        <td>Rs.50</td>
        <td><input type="text" name="item2"
size="4"></td>
    </tr>
    <tr>
        <td>Uttappa</td>
        <td>Rs.80</td>
        <td><input type="text" name="item3"
size="4"></td>
    </tr>
    <tr>
        <td>Curd Rice</td>
        <td>Rs.30</td>
        <td><input type="text" name="item4"
size="4"></td>
    </tr>
    <tr>
        <td colspan=2>Total</td>
        <td><label id="myBill"></label></td>
    </tr>
    </table>
<p><input type="button" value="Show Total Bill"
    onclick="calculateBill()"></p>
<p><input type="submit" value="Submit"
name="B1"></p>
</form>
</body>
</html>
```

- We will fill up the personal details and order details. Then click the **Show Total Bill** button, and we will get the bill displayed on the form.

The sample form for the above code can be viewed as follows -



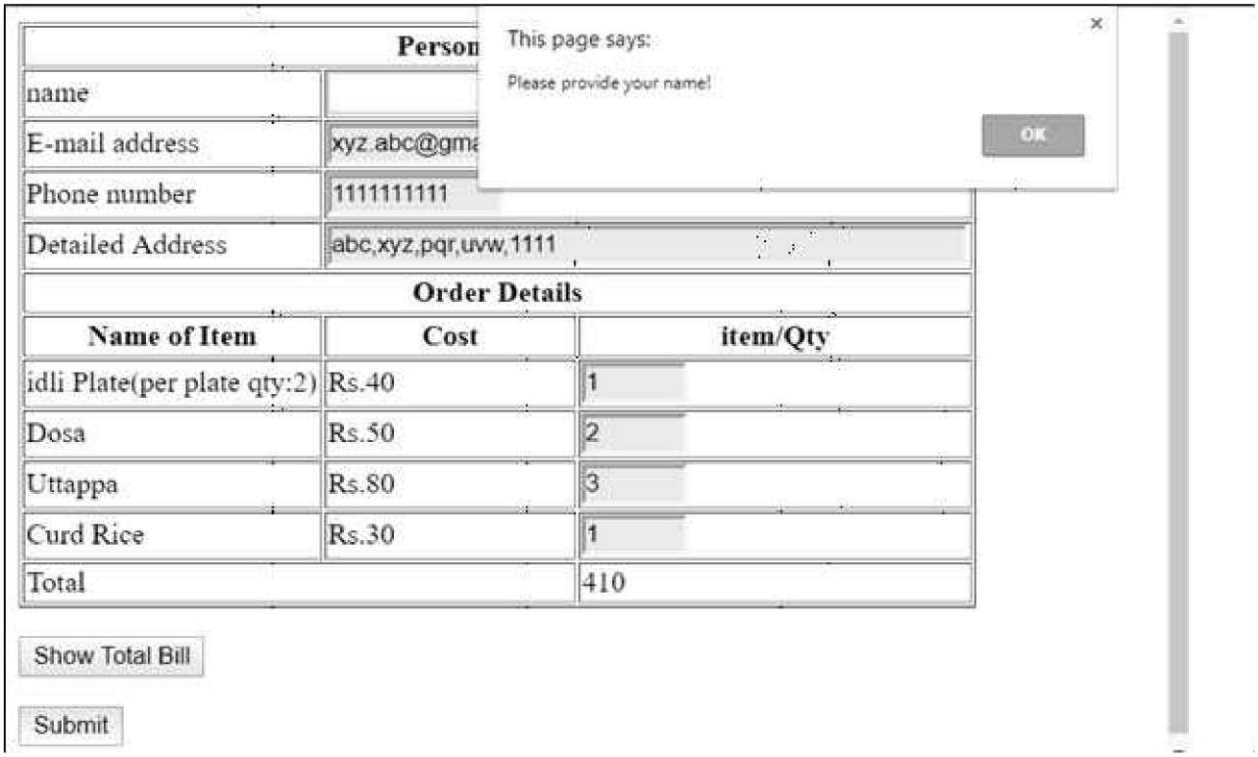
Personal Details		
name	Mr.XYZ	
E-mail address	xyz.abc@gmail.com	
Phone number	1111111111	
Detailed Address	abc,xyz,pqr,uvw,1111	
Order Details		
Name of Item	**Cost**	**item/Qty**
idli Plate(per plate qty:2)	Rs.40	1
Dosa	Rs.50	2
Uttappa	Rs.80	3
Curd Rice	Rs.30	1
Total		410

Show Total Bill

Submit

4

If we leave the **name** field blank and then **submit** the form by clicking the **submit** button, then we will get alter message as follows -



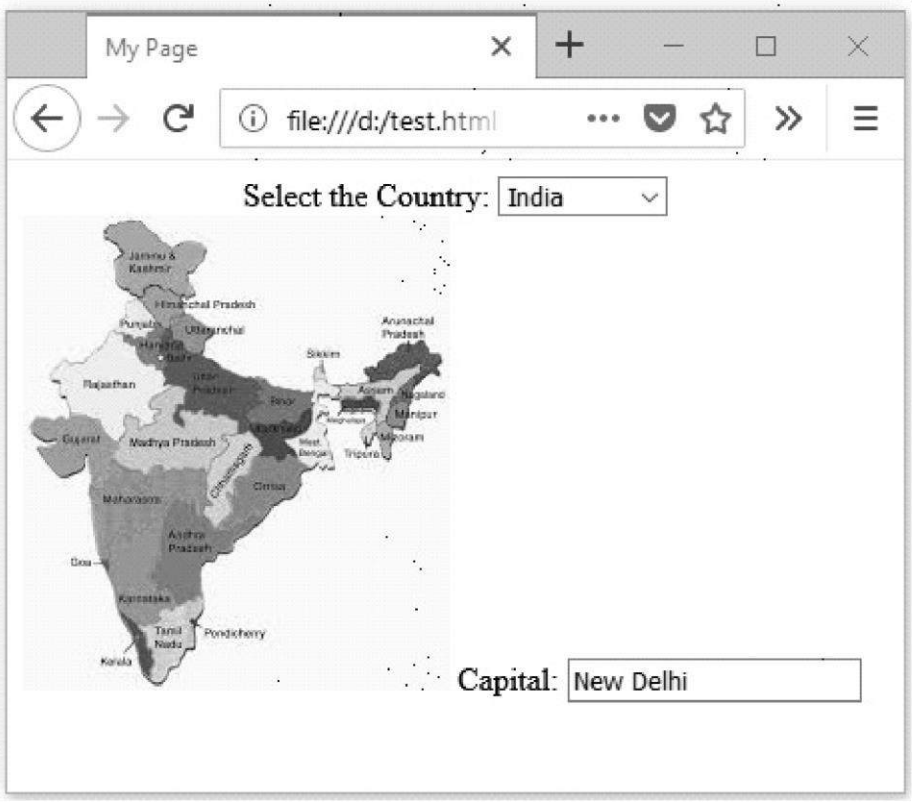
Ex. 2.2.5 *Design a form that offer choices to the user to select country name. Depending upon the name selected map image of country should be displayed on the same window and the capital of the country in the text format should be displayed near the image.*

Sol. :

```
<!DOCTYPE html>
<html>
  <head>
     <title> My Page </title>
  </head>
  <body>
  <div align="center">
  Select the Country:
     <select id="Country">
        <option value="India">India</option>
        <option value="Singapore">Singapore</option>
        <option value="Japan">Japan</option>
     </select>
  </div>
  <img id="map" src="Noimg.jpg"/>
  Capital: <input type="text" id="capital" value="None"/>
  <script type=text/javascript>
  function Display() {
     var img=document.getElementById("map");
     var country_name=this.value;
```

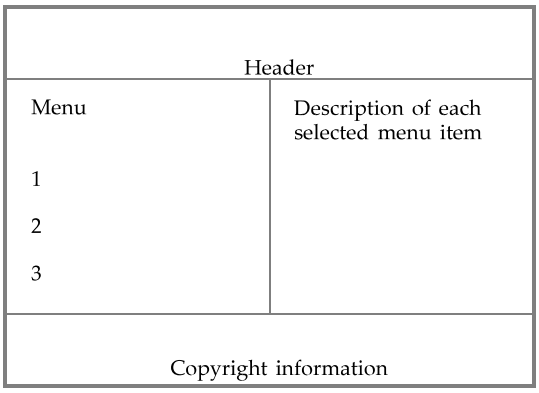
```
        img.src=country_name+".jpg";
        if(country_name=="India")
            document.getElementById("capital").value="New Delhi";
        if(country_name=="Singapore")
             document.getElementById("capital").value="Singapore City";
        if(country_name=="Japan")
             document.getElementById("capital").value="Tokyo";

        return false;
    }
     document.getElementById("Country").onchange=Display;
    </script>
    </body>
</html>
```

Output



Ex. 2.2.6 *Using a frame and form, design a website for an orphanage which should accept donation from the people for 1 day lunch and medical assistance expense. It should also inform the donor how the donation has been spent. The web pages should be properly designed and should be viewed in any screen (mobile, pc.). Amount transfer information should be done through validation. Draw the layout of your web site pages and write the code for each page.*

Sol. : The general layout of the website is


Header
Menu
Description of each selected menu item
1
2
3
Copyright information

Step 1 : The code for main HTML page is as follows

main.html

```
<!DOCTYPE html>
<html>
   <frameset rows="40%,55%,5%">
      <frame src="header.html">
       <frameset cols="20%,80%">
          <frame src="menu.html">
          <frame src="contents.html" name="description">
      </frameset>
   <frameset>
          <frame src="copyright.html">
   </frameset>
    </frameset>
</html>
```

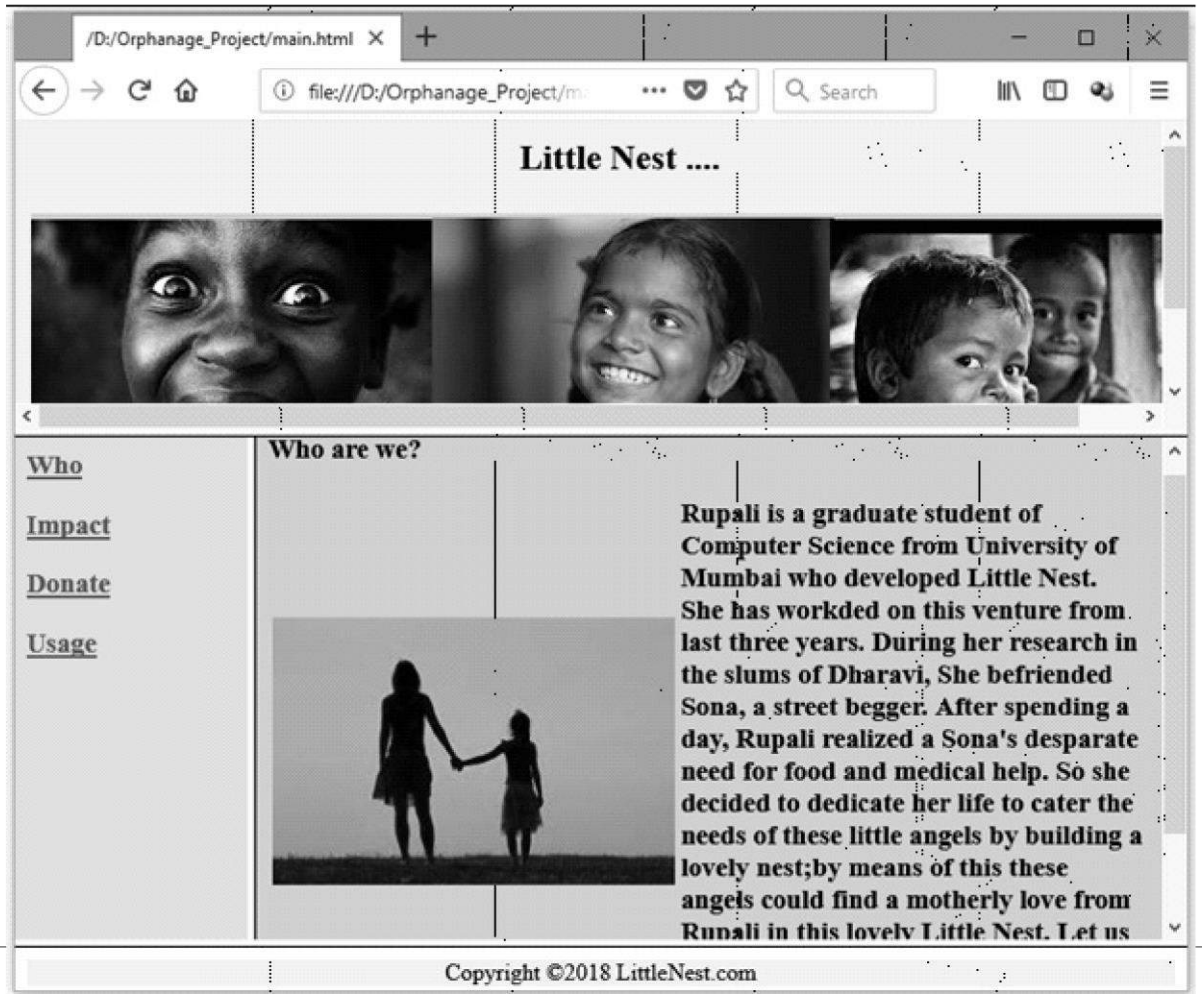
7

Output

Step 2 : To provide the information about the Orphanage and motivation behind it, the link Who can be selected.The code is

who.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="khaki">
<h3>Who are we?<a/></h3>
<form name="form1" action="submitForm.php">
<table>
<tr>
<td><img src="mother.jpg"/></td>
<td><h3>Rupali is a graduate student of Computer Science from University of Mumbai who developed Little Nest. She has worked on this venture from last three years. During her research in the slums of Dharavi, She befriended Sona, a street begger. After spending a day, Rupali realized a Sona's desparate need for food and medical help. So she decided to dedicate her life to cater the needs of these little angels by building a lovely nest;by means of this these angels could find a motherly love from Rupali in this lovely Little Nest. Let us support Rupali in her Godly Work!!!</h3></td>
</tr>
</table>
</form>
</body>
```



>

8

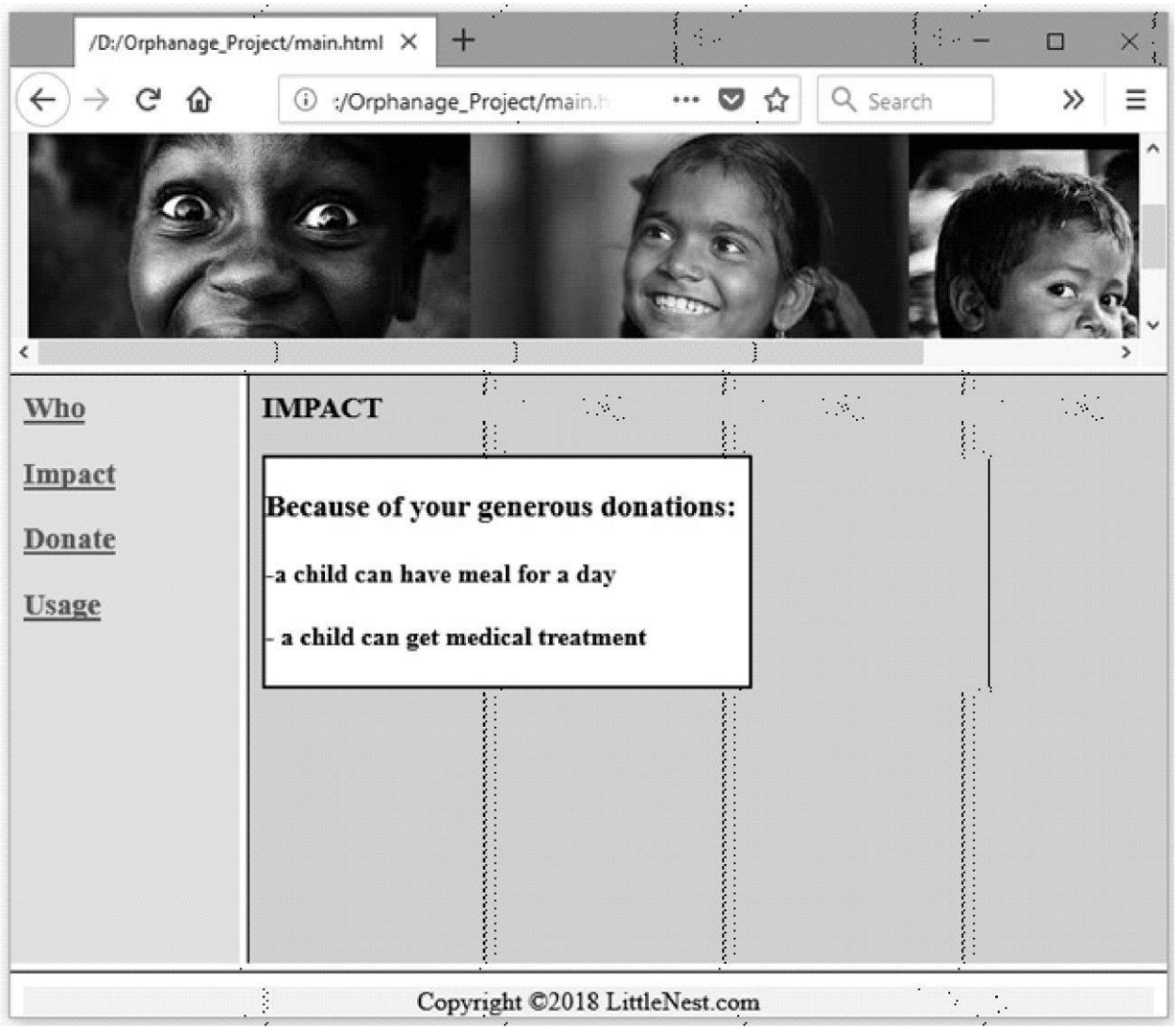
</html

Output

Step 3 : To know the impact of the donation made by the donor, click on the link Imapct,the code for this is as follows -

Project.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="khaki">
<h3>IMPACT<a/></h3>
<div style="background-color:white;width:300px;border:2px solid black">
<h3>Because of your generous donations:</h3>
<h4> -a child can have meal for a day</h4>
<h4>- a child can get medical treatment</h4>
</div>
```



</body>
</html>

Output

>

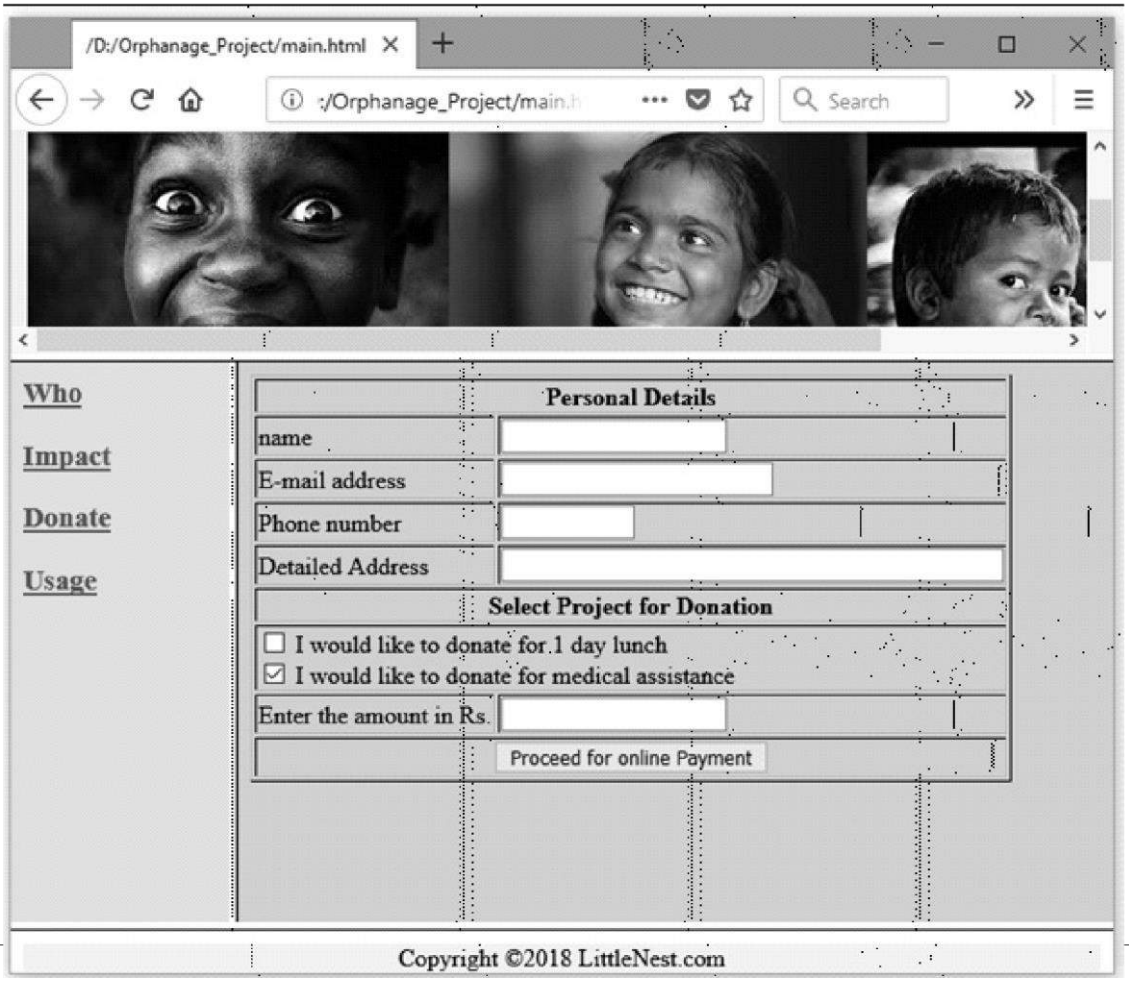
9

Step 4 : For filling up the form for entering personal details, selecting the type of donation and for proceeding for payment following code can be used.

Donation.html

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript">
     function validateForm()
     {
        if( document.myForm.name.value == "" )
        {
           alert( "Please provide your name!" );
           document.myForm.name.focus() ;
           return false;
        }
        if( document.myForm.email.value == "" )
        {
           alert( "Please provide your Email!" );
           document.myForm.email.focus() ;
           return false;
        }
        if( document.myForm.phone.value == "" ||
           isNaN( document.myForm.phone.value ) ||
           document.myForm.phone.value.length != 10 )
        {
           alert( "Please provide a valid phone number" );
           document.myForm.phone.focus() ;
           return false;
        }
        return( true );
        }
  </script>
</head>
<body bgcolor="khaki">
  <form name="myForm" method="post" onsubmit="return validateForm()">
  <table border="1">
  <tr>
        <th colspan="3">Personal Details</th>
  </tr>
  <tr>
     <td>name</td>
        <td colspan="2">
           <input type="text" name="name" size="20">
        </td>
  </tr>
  <tr>
        <td>E-mail address</td>
        <td colspan="2"><input type="text" name="email" size="25"></td>
  </tr>
  <tr>
        <td>Phone number</td>
        <td colspan="2"><input type="text" name="phone" size="10"></td>
```

```
        >
    </tr
    <tr>
            <td>Detailed Address</td>
            <td colspan="2"><input type="text" name="address" size="50"></td>
    </tr>
    <tr>
        <th colspan="3">Select Project for Donation</th>
    </tr>
    <tr>
        <td colspan="3"><input type="checkbox" name="lunch" value="lunch"> I would like to donate for 1 day
lunch<br>
        <input type="checkbox" name="medical" value="medical" checked> I would like to donate for medical
assistance<br>
        </td>
    </tr>
    <tr>
        <td>Enter the amount in Rs.</td><td><input type="textbox" name="amount" value=""/></td>
    </tr>
    <tr>
        <th colspan="3"> <input type="submit" value="Proceed for online Payment"></th>
    </tr>
</form>
</body>
```



>

1

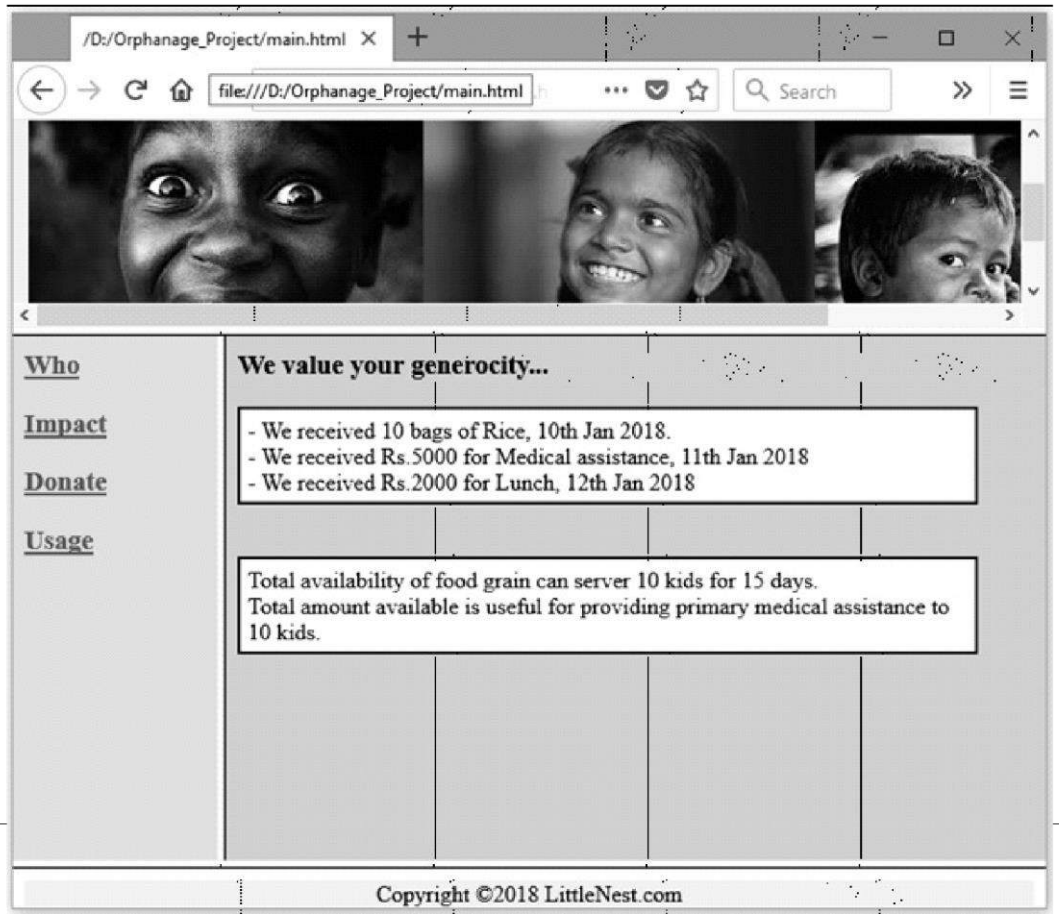
</html

Output

Step 5 : Users can know the usage of the donation made by orphanage by clicking the Usage link. The code is as follows -

Use.html

```
<html>
<head><title>LOGO</title></head>
<body bgcolor="khaki">
<h3>We value your generocity...<a/></h3>
<div style="background-color:white;width:500px;padding:5px;border:2px solid black;">
- We received 10 bags of Rice, 10th Jan 2018.
<br/>
- We received Rs.5000 for Medical assistance, 11th Jan 2018
<br/>
 - We received Rs.2000 for Lunch, 12th Jan 2018
</div>
<br/><br/>
<div  style="background-color:white;width:500px;padding:5px;border:2px solid black;">
 Total availability of food grain can server 10 kids for 15 days.
<br/>
 Total amount available is useful for providing primary medical assistance to 10 kids.
</div>
```



>

```
</body
</html>
```

Output

2.3 PHP

- PHP was developed in 1994 by Apache group.
- PHP stands for PHP: **H**ypertext **P**reprocessor.
- PHP is a server-side scripting language. It is mainly used for form handling and database access.
- It is free to download and use.

2.4 Working Principle of PHP

- PHP is a server side scripting language embedded in XHTML. It is an alternative to CGI, ASP,ASP.NET and JSP.
- The extension to the PHP files are .php,.php3 or .phtml.
- The php processor works in **two modes.** If the PHP processor finds XHTML tags in the PHP script then the code is simply copied to the output file. But when the PHP processor finds the PHP code in the script then that code is simply interpreted and the output is copied to the output file.
- If you click for view source on the web browser you can never see the PHP script because the output of PHP script is send directly to the browser but you can see the XHTML tags.
- PHP makes use of dynamic typing that means there is no need to declare variables in PHP. The type of variable gets set only when it is assigned with some value.
- PHP has large number of library functions which makes it flexible to develop the code in PHP.

2.4.1 Installation of PHP

For installing PHP either PHP installer is preferred or all in package like XAMPP/WAMPP is preferred.

Before installing PHP, install Apache web server on your PC. The php installer can be downloaded from www.php.net/download.

Ex. 2.4.1 : *Explain how can you create a web based application using XAMPP. Give all the steps required in detail*

Sol. : XAMPP is a free distribution package that makes it easy to install Apache Web Server, MySQL, PHP, PERL. Here in XAMPP(The X stands for any OS) or WAMPP(the W stands for Windows OS).

Step 1 : Go to the site
https://www.apachefriends.org/index.html

Step 2 : Click on Download XAMPP for Windows or Linux depending upon your operating system.

Step 3 : When prompted for the download, click "Save" and wait for your download to finish.

>

3

Step 4 : Install the program, and click on "Run." Accept default settings by clicking Next button. Finally you will get installation completion message.

Step 5 : On your drive, the XAMPP folder will be created. Click on **xampp_start** file, this will enable to start Apache, MySQL and Tomcat start.

Step 6 : Write a PHP script and save it in **C:\XAMPP\htdocs\php-examples** folder by giving the filename and extension as .php

Step 7 : Open the web browser and type **http://localhost/php-examples/yourfilename.php**

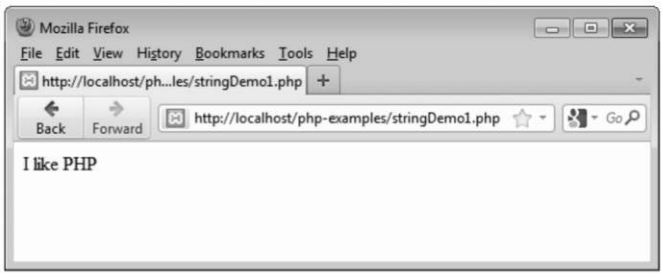
Step 8 : The web application will be executed within your web browser.

For example

PHP Script[stringDemo1.php]

```
<?php
$s="I like PHP";
echo $s;
?>
```

Output



2.5 General Syntactic Characteristics of PHP

- PHP code can be embedded in the XHTML document. The code must be enclosed within **<?php** and ?>
- If the PHP script is stored in some another file and if it needs to be referred then **include** construct is used. For instance :

```
Include("myfile.inc")
```

- The variable names in PHP begin with the $ sign.
- Following are some reserved keywords that are used in PHP.

and	default	false	if	or	this
break	do	For	include	require	true
case	else	foreach	list	return	var
class	elseif	function	new	static	virtual
continue	extends	gloabal	not	switch	while
					xor

- The **comments** in PHP can be #,//, /* …*/
- The PHP statements are terminated by semicolon.

How to write and execute PHP documents ?

Open some suitable text editor like Notepad and type the following code. Save your code by the extension **.php.**

It is expected that the PHP code must be stored in **htdocs** folder of Apache.

As I have installed **xampp,** I have got the directory c:\xampp\htdocs. I have created a folder named php-examples inside the htdocs and stored all my PHP documents in that folder.

Hence when I want to get the output of the PHP code I always give the URL

http://localhost/php-examples/programmName.php

The http://localhost refers to the path **c:\xampp\htdocs**

Following is the first example of PHP script

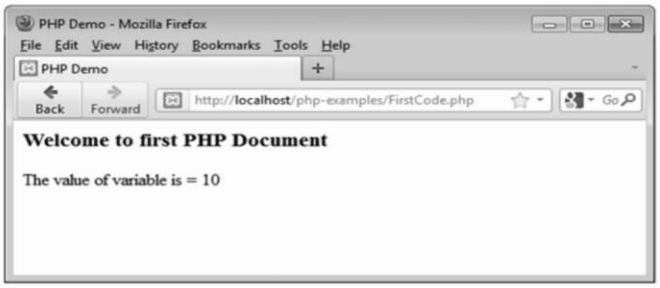
PHP Document[FirstCode.php]

```
<html>
 <head>
  <title> PHP Demo </title>
 </head>
 <body>
  <?php
   $i=10;
   echo "<h3>Welcome to first PHP Document</h3>";
   echo "The value of variable is = $i";
  ?>
```

4

```
</body>
</html>
```

Output



2.6 PHP Variable

- Variables are the entities that are used for storing the values.
- PHP is a dynamically typed language. That is PHP has no type declaration.
- The value can be assigned to the variable in following manner -

```
$variable_name=value
```

- If the value is not assigned to the variable then by default the value is NULL. The unsigned variables are called **unbound variable.**
- If the unbound variable is used in the expression then its NULL value is converted to the value 0.
- Following are some rules that must be followed while using the variables -

1. The variable must start with letter or underscore _, but it should not begin with a number.
2. It consists of alphanumeric characters or underscore.
3. There should not be space in the name of the variable.
4. While assigning the values to the variable the variable must start with the $. For example

```
$marks=100;
```

- Using the function **IsSet** the value of the variable can be tested. That means if IsSet($marks) function returns **TRUE** then that means some value is assigned to the variable marks.
- If the unbound variable gets referenced then the error reporting can be done with the help of a function **error_reporting(7)**. The default error reporting level is 7.

2.7 Data Types

There are four scalar types that are used in PHP and those are Integer, Boolean, Double and String. Let us discuss each one by one.

2.7.1 Integer Type

- For displaying the integer value the Integer type is used.
- It is similar to the long data type in C.
- The size is 32 bit.

2.7.2 Double Type

- For displaying the real values the double data type is used.
- It includes the numbers with decimal point, exponentiation or both. The exponent can be represented by E or e followed by integer literal.
- It is not compulsory to have digits before and after the decimal point. For instance .123 or 123. is allowed in PHP.

2.7.3 String Type

- There is no character data type in PHP. If the character has to be represented then it is represented using the string type itself;but in this case the string is considered to be of length 1.
- The string literals can be defined using either single or double quotes.
- In single quotes the escape sequence or the values of the literals can not be recognized by PHP but in double quotes the escape sequences can be recognized.For example

```
'The total marks are= $marks'
```

5

will be typed as it is but

"The total marks are= $marks"

will display the value of $marks variable.

2.7.4 Boolean Type

- There are only two types of values that can be defined by the Boolean type and those are TRUE and FALSE.
- If Boolean values are used in context of integer type variable then TRUE will be interpreted as 1 and FALSE will be interpreted as 0.
- If Boolean values are used in context of double type then the FALSE will be interpreted as 0.0.

2.8 Constants

- Constant is an identifier that contains some value.
- Once the constant value is assigned to this identifier it does not get changed.
- Constant is case sensitive by default.
- Generally the constant identifiers are specified in upper case.
- The valid constant name must start with letter or underscore. It may then followed by the digits.
- Using define function we can assign value to the constant. The first parameter in define function is the name of the constant and the second parameter is the value which is to be assigned.

For example :

ConstDemo.php

```
<?php
// Valid constant names
define("MYVALUE","10");
echo MYVALUE;
// Invalid constant names
define("1MYVALUE","something");
echo 1MYVALUE;
?>
```

These statements will cause error. If we remove these statements then we will get **10** as an output.

2.9 Operators

2.9.1 Arithmetic Operators and Operations

- PHP supports the collection of arithmetic operators such as +,-,/,*,%,++ and - with their usual meaning.
- While using the arithmetic operators if both the operands are integer then the result will be integer itself.
- If either of the two operands is double then the result will be double.
- PHP has large number of predefined functions. Some of these functions are enlisted in the following table -

Function	Purpose
floor	The largest integer less than or equal to the parameter is returned For example floor(4.9) will return 4,
ceil	The smallest integer less than or equal to the parameter is returned For example ceil(4.9) will return 5,
round	Nearest integer is returned.
abs	Returns the absolute value of the parameter.
min	It returns the smaller element .
max	It returns the larger element .

2.9.2 Relational Operators

- There are eight relational operators used in PHP.
- These are <,>,<=,>=,!=,== has their usual meaning. The are six traditional operators.
- The operator === is used in PHP. It returns true if both operands that are using === have same type and have same value.
- The operator !== is opposite of ===.
- If one of the operand in the six operators is not same then the coercion will occur automatically.

6

2.9.3 Boolean Operators

- The Boolean operators are

Operator	Meaning
and &&	The binary AND operation is performed.
or ||	The binary OR operation is performed.
xor	The XOR operation will be performed.

2.10 Displaying Messages on Web Page

- The PHP is a server side scripting language and is used to submit the web page to the client browser. Hence it is a standard method of embedding the PHP code within the XHTML document. That means we can use the XHTML tags in PHP while displaying the output.
- The **print** function is used to create simple unformatted output. For example: The string can be displayed as follows

```
print "I am proud of my <b>country</b>"
```

The numeric value can also be displayed using the print. For example -

```
print(100);
```

It will display the output as 100.

- PHP also makes use of the **printf** function used in C. For example

```
printf("The student %d has %f marks",$roll_no,$marks);
```

- Following is a simple PHP document which makes use of the statements for displaying the output.

PHP Document[OutputDemo.php]

```
<html>
<head>
   <title> Output Demo</title>
</head>
<body>
 <?php
   print "<h2>Welcome to my Website </h2>";
   print "<hr/>";
   $roll_no=1;
   $name="AAA";
   printf("<b>The roll number: %d</b>",$roll_no);
  print "<br/><b>";
   printf("The name: %s",$name);
   print "</b>";
?>
</body>
</html>
```

Output

2.11 Flow Control and Loop

2.11.1 if Statements

- The if statement, the **if ... else** statement or **if...elseif** statements are used as selection statements. This selection is based on some condition. Following is an example of it

PHP Document[selection.php]

```
<html>
<head>
   <title>Selection Demo</title>
</head>
<body>
 <?php
   print "<h2>Selection Statement </h2>";
   $a=10;
   $b=20;
   $c=30;
   if($a>$b)
     if($a>$c)
       print "<b> <I>a is the largest number </I></b>";
    else
       print "<b><I> c is the largest number</I> </b>";
   else
```

7

```
    if($b>$c)
      print "<b><I>b is the largest number</I>  </b>";
    else
      print "<b> <I>c is the largest number</I> </b>";
?>
</body>
</html>
```

Output

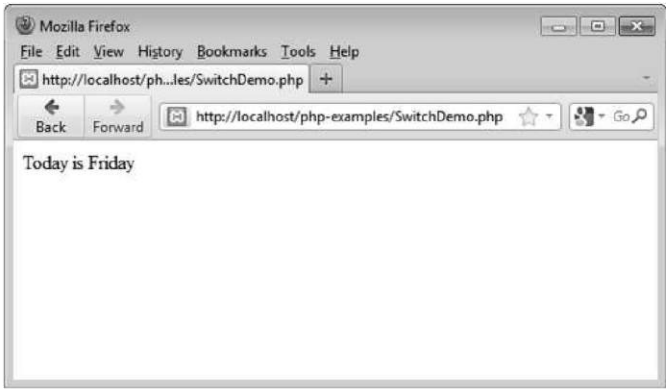
2.11.2 Switch Statement

Similar to **if** statement the **switch** statement can also be used for selection. Following is a simple PHP script for demonstrating switch statement

PHP Document[SwitchDemo.php]

```
<?php
$today = getdate();
switch($today['weekday'])
{
    case "Monday":print "Today is Monday";
                                    break;
    case "Tuesday":print "Today is Tuesday ";
                                    break;
    case "Wednesday":print "Today is Wednesday ";
                                    break;
    case "Thursday":print "Today is Thursday";
                                    break;
    case "Friday":print "Today is Friday";
                                    break;
    case "Saturday":print "Today is Saturday";
                                    break;
    case "Sunday":print "Today is Sunday ";
                                    break;
   default: print "Invalid input";
}
?>
```

Output



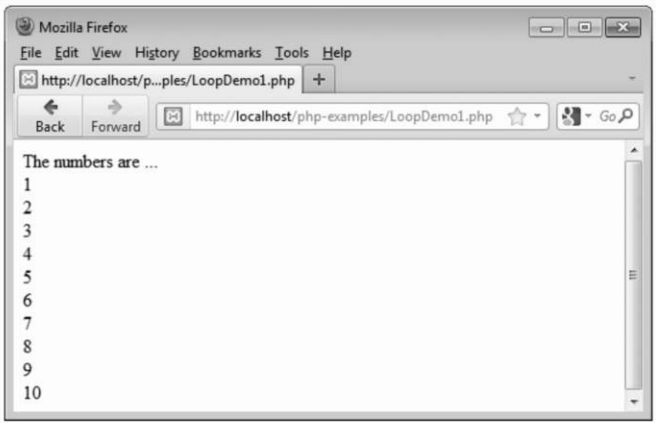
2.11.3 Loop Statements

- The while, for and do-while statements of PHP are similar to JavaScript.
- Following is a simple PHP script which displays the first 10 number

PHP Document[LoopDemo1.php]

```
<?php
$i=1;
print "The numbers are ...";
print "<br/>";
while($i<=10)
{
    print $i;
    print "<br/>";
    $i++;
}
?>
```

Output



Similarly we can modify the above script by using do-while and for loop as follows -

8

do …while

```
<?php
$i=1;
print "The numbers are ...";
print "<br/>";
do
{
    print $i;
    print "<br/>";
    $i++;
} while($i<=10);
?>
```

for

```
<?php
print "The numbers are ...";
print "<br/>";
for($i=1;$i<n)
{
    print $i;
    print "<br/>";
    $i++;
}
?>
```

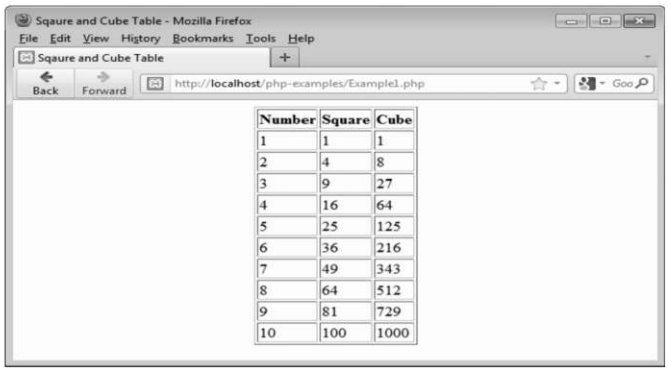
2.11.4 Examples based on Control Statement

Ex. 2.11.1 *Write a PHP script to display the squares and cubes of 1 to 10 numbers.*

Sol. :

```
<html>
<head>
<title> Sqaure and Cube Table </title>
</head>
<body>
<center>
<?php
print "<table border =1>";
print "<tr>";
print "<th>Number</th>";
print "<th>Square</th>";
print "<th>Cube</th>";
print "</tr>";
for($i=1;$i<=10;$i++)
{
print "<tr>";
print "<td>$i";
print "</td>";
print "<td>";
print $i*$i;
print "</td>";
print "<td>";
print pow($i,3);
print "</td>";
print "</tr>";
}
print "</table>";
?>
</center>
</body>
</html>
```

Output



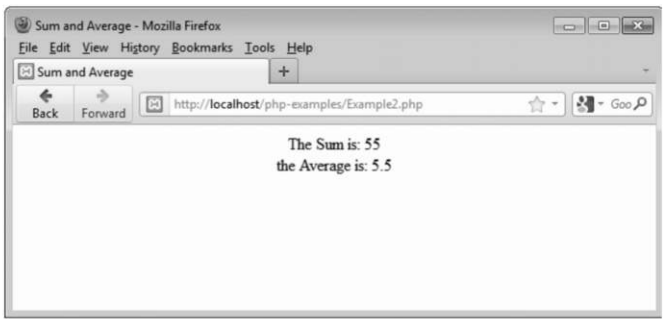
Number	Square	Cube
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Ex. 2.11.2 *Write a PHP script to compute the sum and average of N numbers .*

PHP Document[Example2.php]

```
<html>
<head>
<title> Sum and Average </title>
</head>
<body>
<center>
<?php
$sum=0;
for($i=1;$i<=10;$i++)
{
    $sum+=$i;
}
$avg=$sum/10;
print "The Sum is:  $sum";
print "<br/>";
print "the Average is: $avg";
?>
</center>
</body>
</html>
```

Output



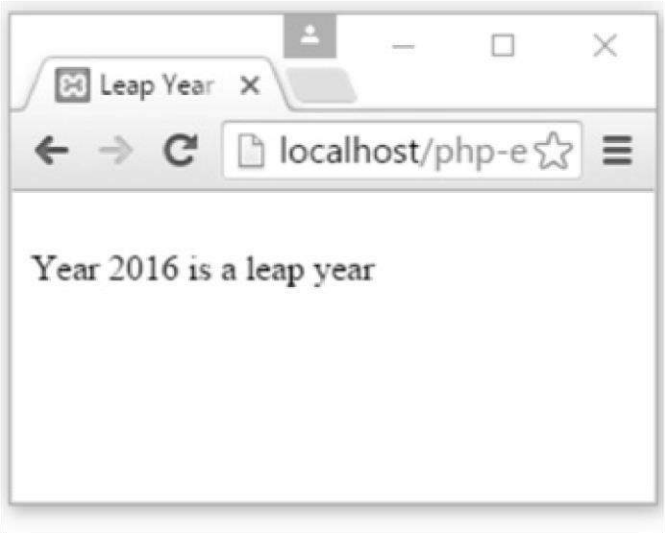
9

Ex. 2.11.3 *Write PHP programs to print whether current year is leap year or not.*

Sol. :

```
<html>
<head>
<title>Leap Year Demo</title>
</head>
<body>
<?php
 $year=2016;
 print "<br/>";
 if($year%4==1)
  {  printf("Year %d is not a leap year",$year); }
 else
  {  printf("Year %d is a leap year",$year); }
?>
</body>
</html>
```

Output

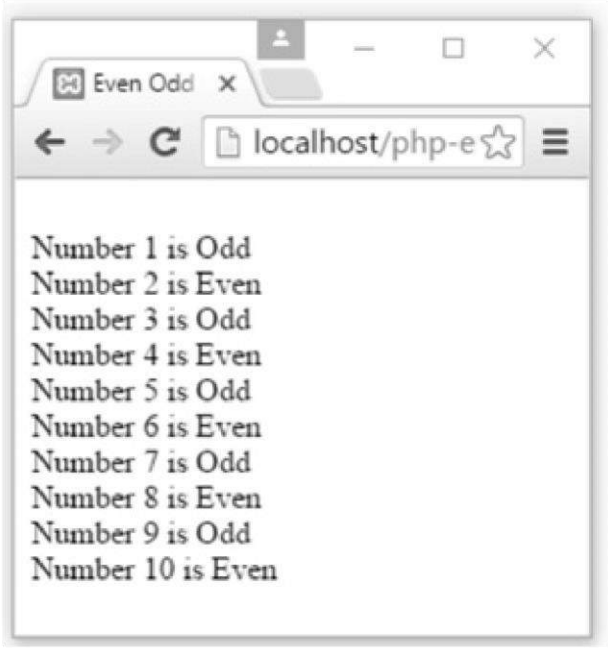


Ex. 2.11.4 *Write PHP programs to print whether given number is odd or even.*

Sol. :

```
<html>
<head>
<title>Even Odd Demo</title>
</head>
<body>
<?php
for($i=1;$i<=10;$i++)
{
 $num=$i;
 print "<br/>";
 if($num%2==1)
  {  printf("Number %d is Odd",$num); }
 else
  {  printf("Number %d is Even",$num); }
}
?>
</body>
</html>
```

Output

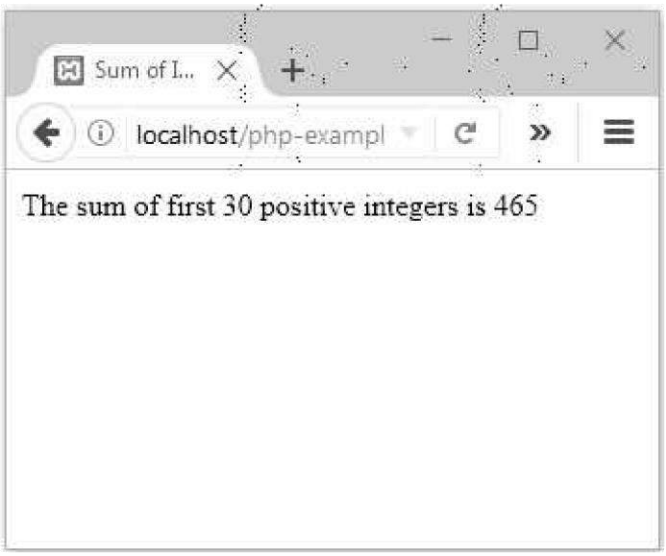


Ex. 2.11.5 *Write PHP script to compute the sum of positive integers upto 30 using do-while statement.*

Sol. :

```
<html>
<head>
<title>Sum of Integers</title>
</head>
<body>
<?php
$sum=0;
$i=1;
do
{
 $sum=$sum+$i;
 $i++;
}while($i<=30);
 printf("The sum of first 30 positive integers is %d ",$sum);
?>
</body>
</html>
```

Output

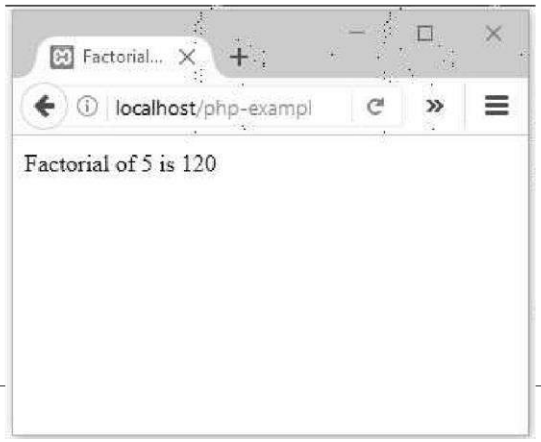


Ex. 2.11.6 *Write PHP script to compute factorial of 'n' using while or for loop construct.*

Sol. :

```
<html>
<head>
<title>Factorial Program</title>
</head>
<body>
<?php
$n = 5;
$factorial = 1;
for ($i=$n; $i>=1; $i--)
{
  $factorial = $factorial * $i;
}
echo "Factorial of $n is $factorial";
?>
</body>
</html>
```

Output

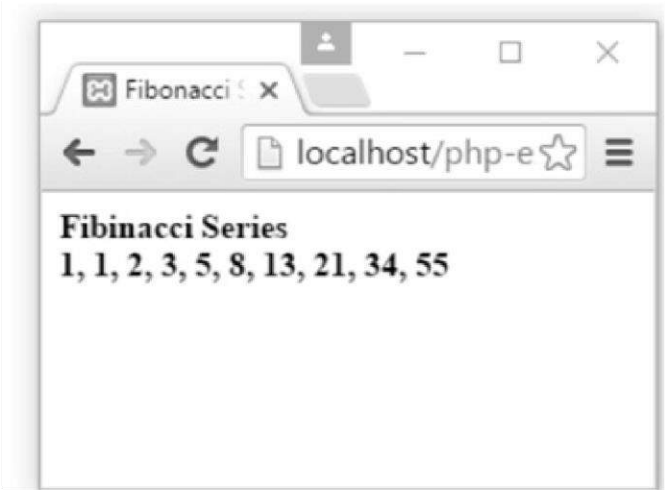


Ex. 2.11.7 *Write PHP script to display Fibonacci of length 10 .*

Sol. :

```
<html>
<head>
<title>Fibonacci Series</title>
</head>
<body>
<?php
  $i=1;
  $j=1;
  print "<b>Fibinacci Series<br/>";
  printf("%d, %d",$i,$j);
  for($count=1;$count<9;$count++)
  {
      $k=$i+$j;
      $i=$j;
      $j=$k;
      printf(", %d",$k);
  }
?>
</body>
</html>
```

Output



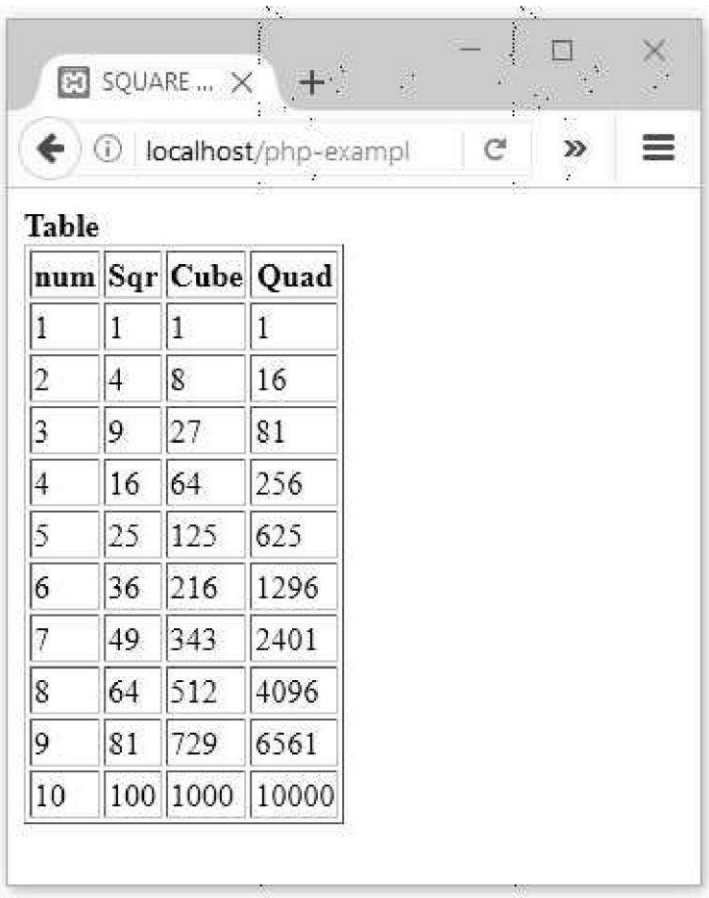
Ex. 2.11.8 *Construct a PHP script to compute the squareRoot, Square, Cube and Quad of 10 numbers.*

Sol. :

```
<html>
<head>
<title>SQUARE CUBE QUAD DEMO</title>
</head>
<body>
<?php
```

```
  print "<b>Table<br/>";
  print "<table border='1'>";
  print "<tr><th>num</th><th>Sqr</th><th>Cube</th><th>Quad</th></tr>";
  for($count=1;$count<=10;$count++)
  {
  $sq=$count*$count;
  $cube=$count*$count*$count;
  $quad=$count*$count*$count*$count;
       print "<tr><td>$count</td><td>$sq</td><td>$cube</td><td>$quad</td></tr>";
  }
  print "</table>";
?>
</body>
</html>
```

Output



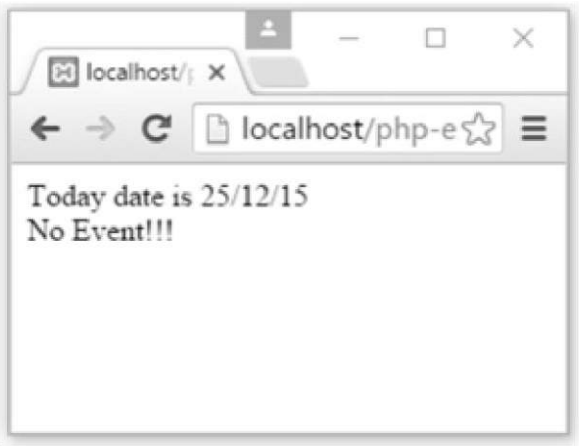
Ex. 2.11.9 *With the use of PHP, switch case and if structure perform the following and print appropriate message. i) Get today's date ii) If date is 3, it is dentist appointment. iii) If date is 10, go to conference. iv) If date is other than 3 and 10, no events are scheduled.*

Sol. :

```
<!DOCTYPE html>
<html>
<body>
<?php
  echo "Today date is " . date("d/m/y") . "<br>";
```

```
    if((date("d")<3)||(date("d")>10))
        echo "No Event!!!";
    else if((date("d")>3)&&(date("d")<10))
        echo "No Event!!!";
    else
    {
     switch(date("d"))
     {
       case 3 : echo "Dentist Appointment";
            break;
       case 10 : echo "Go to Conference";
            break;
     }
    }
?>
</body>
</html>
```

Output



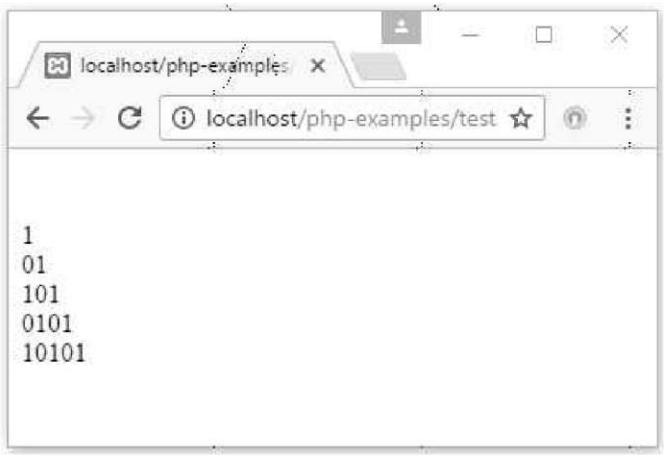
Ex. 2.11.10 *Write a PHP code to display the following pattern*

```
    1
  0 1
 1 0 1
 0 1 0 1
 1 0 1 0 1
```

Sol. :

```
<html>
<head>
</head>
<body>
<?php
    for($i=0;$i<7;$i++)
    {
        for($j=1;$j<$i;$j++)
        {
            if(($i+$j)%2==0)
            {
                printf("0");
            }
            else
            {
                printf("1");
            }
        }
        print "<br/>";
    }
?>
</body>
</html>
```

Output



2.12 Arrays

- Arrays is a collection of similar type of elements, but in PHP you can have the elements of mixed type together in single array.
- In each PHP, each element has two parts **key** and **value.**
- The key represents the index at which the value of the element can be stored.
- The **keys** are positive integers that are in ascending order.

2.12.1 Array Creation

In PHP there are two types of arrays -

1. **Indexed array :** Indexed array are the arrays with numeric index. The array values can be stored from index 0. For example -

3

```
<html>
<head>
    <title>PHP Indexed Arrays</title>
      </head>
     <body>
<?php
$names = array("AAA", "BBB", "CCC");
// Printing array structure
print_r($names);
?>
</body>
</html>
```

Output

Here values gets stored at corresponding index as follows -

```
$mylist[0]=10;
$mylist[1]=20;
$mylist[2]=30;
$mylist[3]=40;
$mylist[4]=50;
```

We can directly assign some value at specific index.

```
$mylist[5]=100;
```

2. **Associated array :** Associated arrays are the arrays with named keys. It is a kind of array with **name** and **value** pair. For example

```
<html>
<head>
   <title>PHP Associative Array</title>
</head>
<body>

<?php
$city["AAA"] = "Pune";
$city["BBB"] = "Mumbai";
$city["CCC"] = "Chennai";

// Printing array structure
print_r($city);
?>
</body>
</html>
```

Output

2.12.2 Accessing Array Elements

- Using an array subscript we can access the array element. The value of subscript is enclosed within the square brackets. For example -

```
$Citycode['Pune']=005;
$Name[0]="Chitra";
```

- Multiple values can be set to a single scalar variable using array. For example -

```
$people=array("Meena","Teena","Heena");
list($operator,$accountant,$manager)=$people;
```

- By this assignment Meena becomes operator, Teena becomes accountant and Heena becomes the manager.

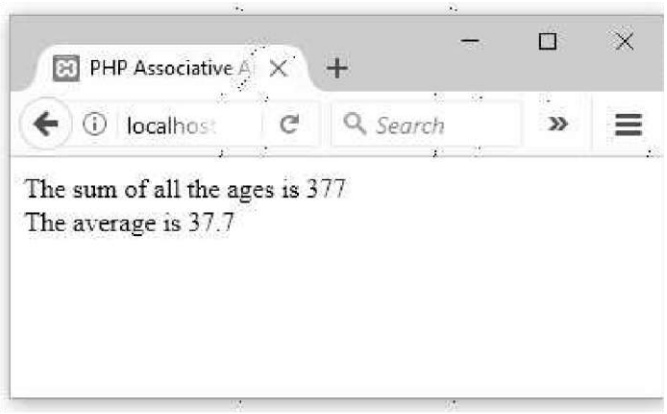
Ex. 2.12.1 *Consider an associative array called person_age with name and age of 10 persons. Write a PHP program to calculate the average age of this associative array*

Sol. :

```
<html>
<head>
   <title>PHP Associative Array</title>
</head>
<body>

<?php
$person_age=array(
        'AAA' =>10,'BBB' =>22,'CCC' => 45,'DDD'=> 31,
        'EEE' =>33,'FFF' =>25,'GGG' =>56,'HHH' =>73,
        'III' =>35,'JJJ' =>47
        );
// Printing array structure
$sum=array_sum($person_age);
print("The sum of all the ages is $sum");
$avg=$sum/10;
print("<br/>The average is $avg");
?>
</body>
</html>
```

4

Output



2.12.3 Functions for Dealing with Arrays

- The unset function is used to remove particular element from the array. For example consider following PHP document

PHP Document[ArrayFunDemo1.php]

```
<?php
$mylist=array(10,20,30,40,50);
unset($mylist[1]);
for($i=0;$i<=4;$i++)
{
    print $mylist[$i];
    print "  ";
}
?>
```

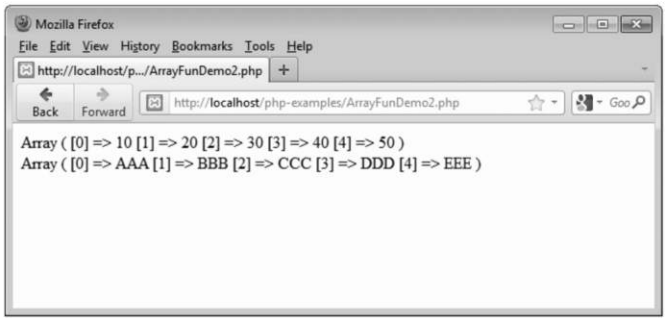
Output

- The functions **array_keys** and **array_values** are used to return the array keys and the values at corresponding key. For example consider the following PHP document -

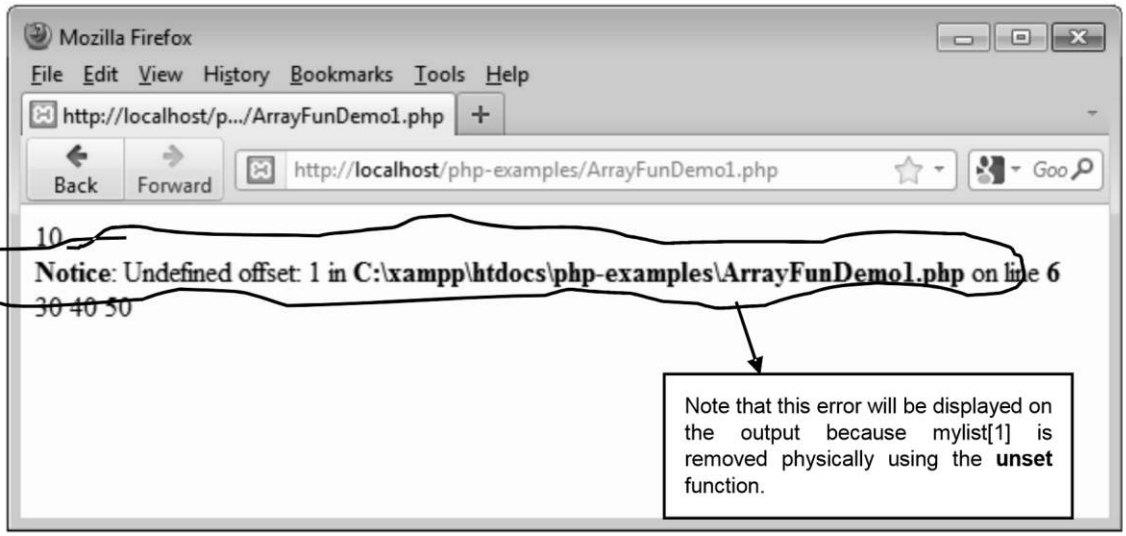
PHP Document[ArrayFunDemo2.php]

```
<?php
$mylist=array(10=>"AAA",20=>"BBB",30=>"CCC",40=>"D
DD",50=>"EEE");
$Roll=array_keys($mylist);
$Name=array_values($mylist);
print_r($Roll);
print "<br/>";
print_r($Name);
?>
```

Output



- The existence of particular key can be checked by using the **array_key_exists** function. This function returns the Boolean value.

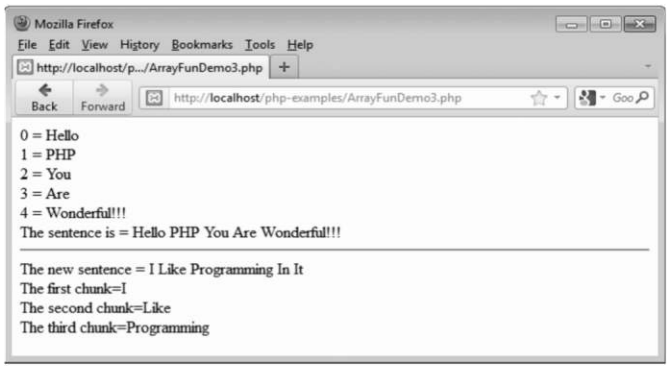


- The **is_array** returns the Boolean value. This function takes a variable as a parameter. If the function returns TRUE then that means the parameter passed to this function is of array type.
- The **implode** and **explode functions** are used to break the word into strings or vice versa. For example consider the following PHP code -

PHP Document[ArrayFunDemo3.php]

```
<?php
$mylist = array("Hello", "PHP","You","Are","Wonderful!!!");

$mySentence = implode(" ", $mylist);
for($i = 0; $i < count($mylist); $i++)
{
    echo "$i = $mylist[$i] <br />";
}
echo "The sentence is  = $mySentence <br />";
print "<hr/>";
$newSentence="I Like Programming In It";
$chunks=explode(" ",$newSentence);
echo "The new sentence = $newSentence";
print "<br/>";
echo "The first chunk=$chunks[0]";
print "<br/>";
echo "The second chunk=$chunks[1]";
print "<br/>";
echo "The third chunk=$chunks[2]";
print "<br/>";
?>
```

Output



2.12.4 Sequential Access to Array Elements

- The array element reference start at the first element and every array maintains an internal pointer using which the next element can be easily accessible.
- This helps to access the array elements in sequential manner.
- The pointer **current** is used to point to the current element in the array. Using the **next** function the next subsequent element can be accessed. Following PHP code illustrates this idea -

PHP Document[ArrayFunDemo4.php]

```
<?php
$mylist = array("Hello", "PHP","You","Are","Wonderful!!!");
$myval=current($mylist);
print("The current value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
?>
```

Output

- Using **each** function we can iterate through the array elements.

PHP Document[ArrayFunDemo5.php]

```
<?php
$mylist = array("Hello", "PHP","You","Are","Wonderful!!!");

while($myval=each($mylist))
{
    $val=$myval["value"];
    print("The current value of the array is <b>$val</b>");
```

6

```
    print "<br/>";
}
?>
```

Output

- The **foreach** function is used to iterate through all the elements of the loop. The syntax of foreach statement is as follows -

```
foreach($array as $value)
{
  statements to be executed
}
```

- The above code can be modified and written as follows -

PHP Document[ArrayFunDemo6.php]

```
<?php
$mylist = array("Hello", "PHP","You","Are","Wonderful!!!");

foreach($mylist as $value)
{
    print("The current value of the array is
<b>$value</b>");
    print "<br/>";
}
?>
```

The output will be the same as above.

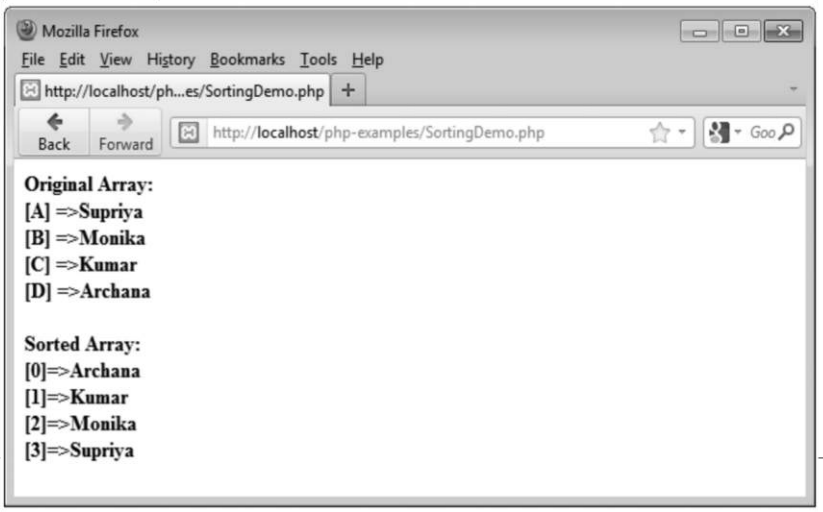
2.12.5 Sorting Arrays

- Sorting is a process in which the elements of arrays in some specific order. There are two types ordering which are followed in sorting - ascending order and descending order.
- Basically PHP uses **sort** function for sorting the array elements. There are some other functions that are also available for sorting the arrays in desired manner.
- The **sort** function sorts the array based on the values. After applying the **sort** function this function assigns new keys to the values of the array.
- Following PHP document illustrates these functions -

PHP Document[SortingDemo.php]

```
<?php
$A = array("A" => "Supriya", "B" => "Monika", "C" =>
          "Kumar","D"=>"Archana");
print "<b>Original Array:<b><br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br />");
sort($A);
print "<br/>";
print "<b>Sorted Array:<b><br/>";
foreach($A as $key=>$value)
 print("[$key]=>$value <br />");
?>
```

Output

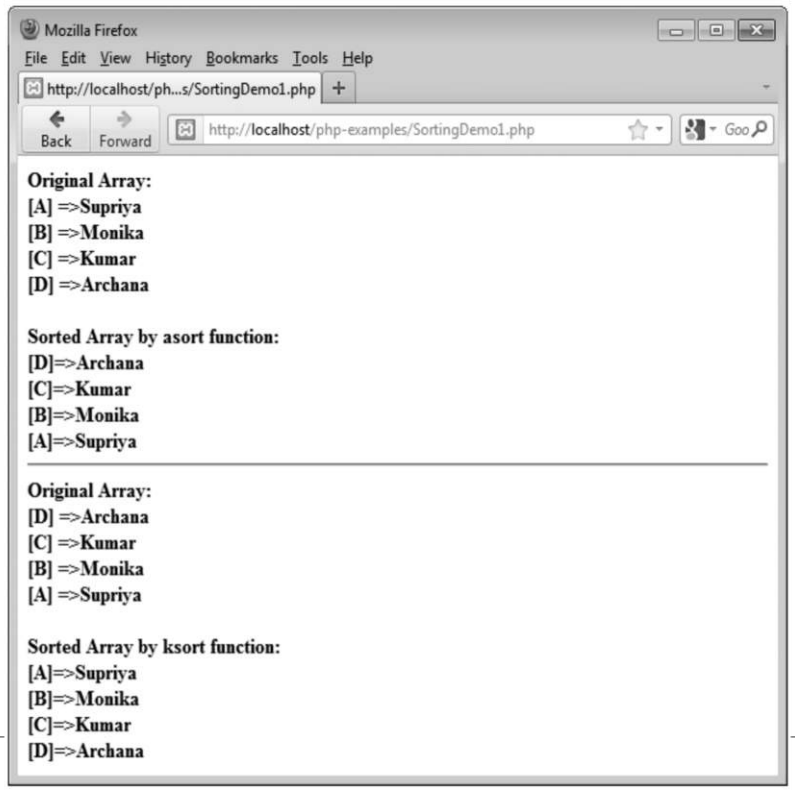


- The **asort()** function sorts an array by the values. But the original keys are kept.
- The **ksort** function sorts the array by keys but each value's original key is retained.
- Following PHP illustrates both of these functions -

PHP Document[SortingDemo1.php]

```
<?php
$A = array("A" => "Supriya", "B" => "Monika", "C" => "Kumar","D"=>"Archana");
print "<b>Original Array:<b><br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br />");
asort($A);
print "<br/>";
print "<b>Sorted Array by asort function:<b><br/>";
foreach($A as $key=>$value)
 print("[$key]=>$value <br />");
print "<hr/>";
$B = array(30 => "CCC", 20 => "BBB", 40 => "DDD",10=>"AAA");
print "<b>Original Array:<b><br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br />");
ksort($A);
print "<br/>";
print "<b>Sorted Array by ksort function:<b><br/>";
foreach($A as $key=>$value)
 print("[$key]=>$value <br />");
?>
```

Output

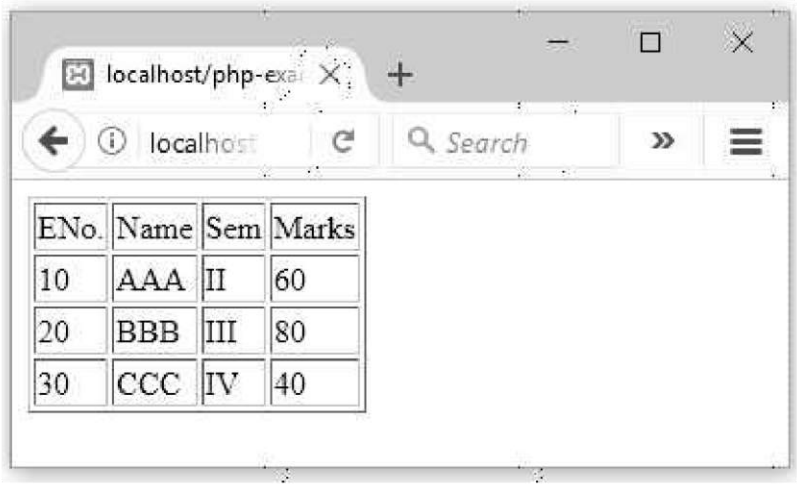


Ex. 2.12.2 *Use an array to store student information such as enrollment no, name, semester and percentage. Output the data to a web page using PHP.*

Sol. :

```
<html>
<head></head>
<body>
<?php
$a=array(array(10,"AAA","II",60),array(20,"BBB","III",80),array (30,"CCC","IV",40));
echo "<table border='1'>";
echo "<tr>";
echo"<td>ENo.</td><td>Name</td><td>Sem</td><td>Marks</td>";
echo "</tr>";

for($i=0;$i<3;$i++)
{
 echo "<tr>";
 for($j=0;$j<4;$j++)
 {
  echo "<td>";
  echo $a[$i][$j];
  echo "</td>";
 }
 echo "</tr>";
}
 echo "</table>";
?>
</body>
</html>
```

Output

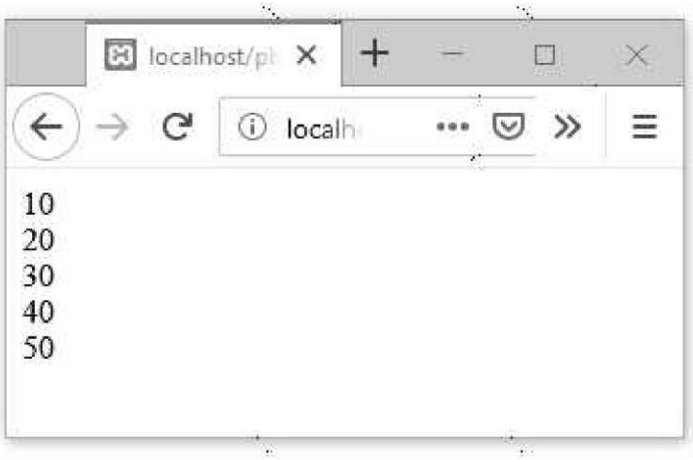


Ex. 2.12.3 *Write a PHP program to declare an array of integers, to sort them in ascending order and to display the sorted array using for-each.*

Sol. :

```
<!DOCTYPE html>
<html>
<body>
<?php
    $numbers = array(30, 10, 50, 20, 40);
    sort($numbers);
    foreach( $numbers as $value )
    {
        echo "$value<br>";
    }
?>
</body>
</html>
```

Output



Review Question

1. *Write a PHP program to declare an array of integers, to sort them in ascending order, and to display the sorted array using for-each.*

2.13 Strings

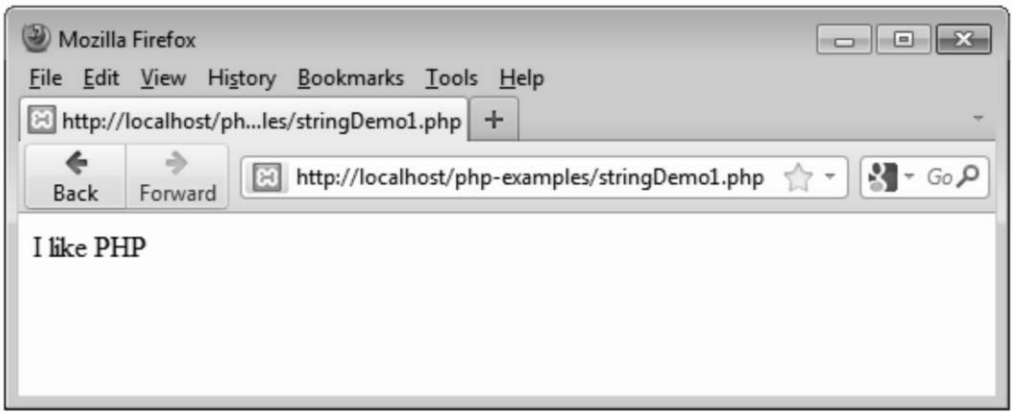
- String is a collection of characters.
- In PHP the string is denoted within a double quote.
- Concatenation is the only one operator used in string.It is denoted by dot.
- Strings are treated as the array of characters. The first position of the character is indexed as 0.
- The sample PHP script that stores the string in a variable is as given below -

PHP Script[stringDemo1.php]

```
<?php
$s="I like PHP";
echo $s;
?>
```

0

Output



Note that the variable **s** is assigned with the string. The string is given in a double quotes. Then using the echo whatever string is stored in the variable **s** is displayed on the console.

- Various functions used for string handling are -

Function	Purpose	Sample PHP Code	Output
strlen(string1)	It finds the total number of characters in the string	<?php $s="Friend"; echo $s <?php	6
strcmp (string1,string2)	It compares the two strings. It is case sensitive. If this function returns 0 then two strings are equal If this function returns >0 then string1 is greater than string2 If this function returns <0 then string1 is less than string2	<?php echo strcmp("PHP","PHP"); ?>	0
strtolower (string1)	This function converts the characters in string1 to lower case.	<?php echo strtolower("PHP."); ?>	php
strtoupper (string1)	This function converts the characters in string1 to upper case.	<?php echo strtolower("php"); ?>	PHP

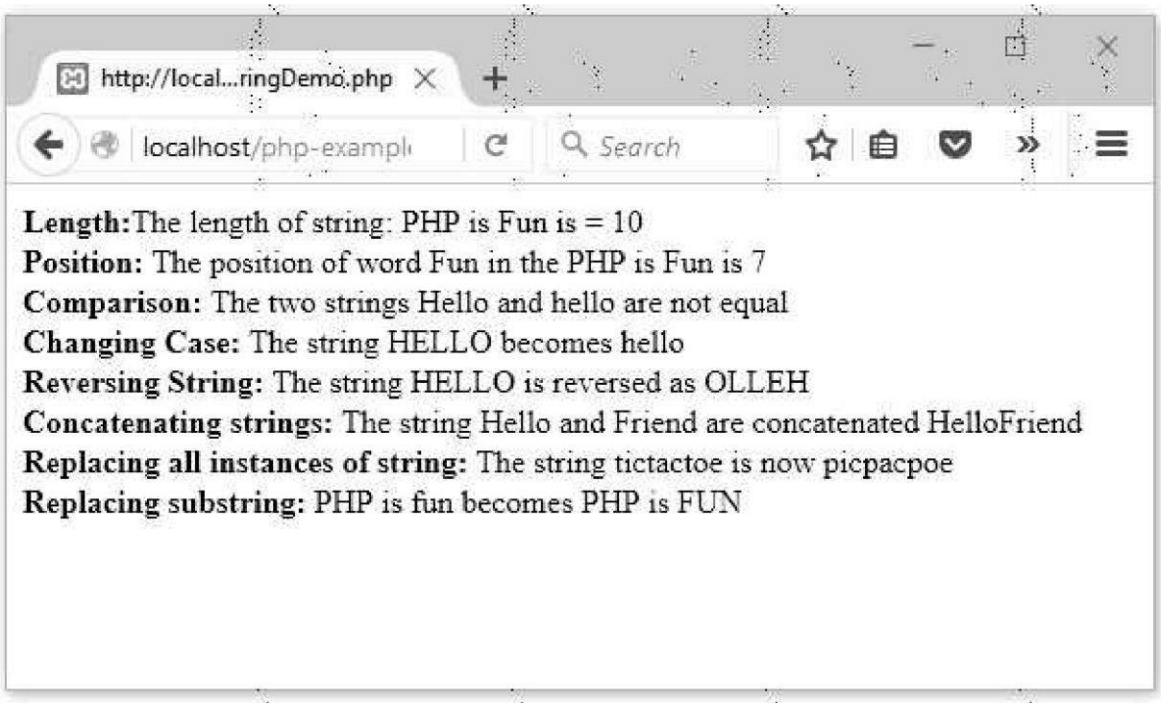
trim(string1)	This function eliminates the white space from both the ends of the string	<?php $str = " PHP "; echo " : $str "; echo " :".trim($str); echo " "; ?>	: PHP : PHP

Ex. 2.13.1 *Write a PHP program to do string manipulations.*

Sol. : For this program we will apply various built in string manipulating functions to the string. The PHP code is as follows -

```
<?php
  $Str1="PHP is Fun";
  $length = strlen($Str1);
  echo "<b> Length:</b>The length of string: $Str1 is = $length";
  echo "<br/><b>Position: </b>The position of word Fun in the $Str1 is ".strpos($Str1,'Fun');
  $Str1="Hello";
  $Str2="hello";
  if(strcmp($Str1,$Str2))
      echo "<br/><b>Comparison: </b> The two strings $Str1 and $Str2 are not equal";
  else
      echo "<br/><b>Comparision: </b> The two strings $Str1 and $Str2 are equal";
  $Str1="HELLO";
  echo "<br/><b>Changing Case: </b> The string $Str1 becomes ".strtolower($Str1);
  echo "<br/><b>Reversing String: </b> The string $Str1 is reversed as ".strrev($Str1);
  $Str1="Hello";
  $Str2="Friend";
  echo "<br/><b>Concatenating strings: </b> The string $Str1 and $Str2 are concatenated ".$Str1.$Str2;
  echo "<br/><b>Replacing all instances of string: </b> The string tictactoe is now ";
  echo str_replace("t","p","tictactoe");
  $Str1="PHP is fun";
  $newstring=substr_replace($Str1,"FUN",7,9);
  echo "<br><b>Replacing substring: </b> $Str1 becomes $newstring";
?>
```

Output



2.14 Functions

The functions in PHP are very much similar to the functions in C. Let us discuss it in detail -

2.14.1 General Characteristics of Functions

- The syntax of the function definition is as follows -

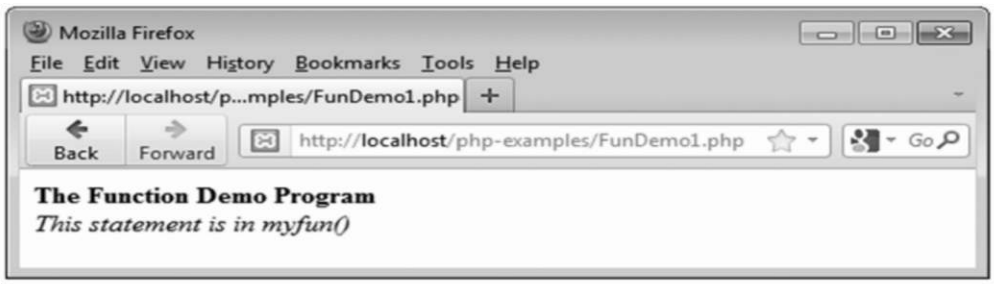
```
function name_of_function(parameter list)
{
  statements to be executed in function
 …
 …
 …}
```

- The function gets executed only after the call to that function. The call to the function can be from anywhere in the PHP code.For example -

PHP Document[FunDemo1.php]

```
<?php
function myfun()
{
 print "<i>This statement is in myfun()</i>";
}
print "<b>The Function Demo Program</b>";
print "<br/>";
myfun();
?>
```

Output



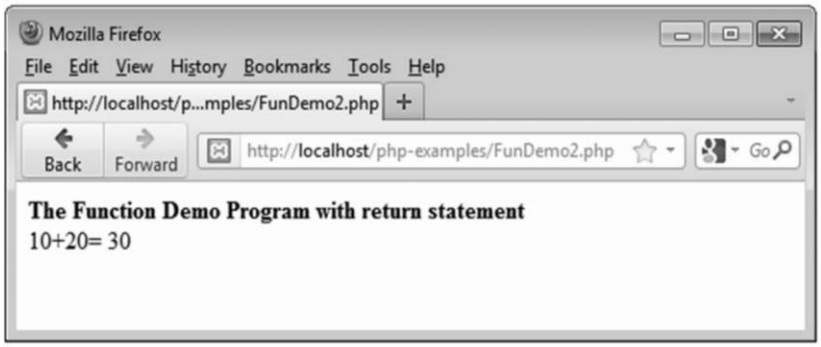
- The **return** statement is used for returning some value from the function body. Following PHP scripts shows this idea.

PHP Document[FunDemo2.php]

```
<?php
function Addition()
{
 $a=10;
 $b=20;
 $c=$a+$b;
 return $c;
}
print "<b>The Function Demo Program with return statement</b>";
print "<br/>";
print "10+20= ".Addition();
?>
```

3

Output



2.14.2 Parameters

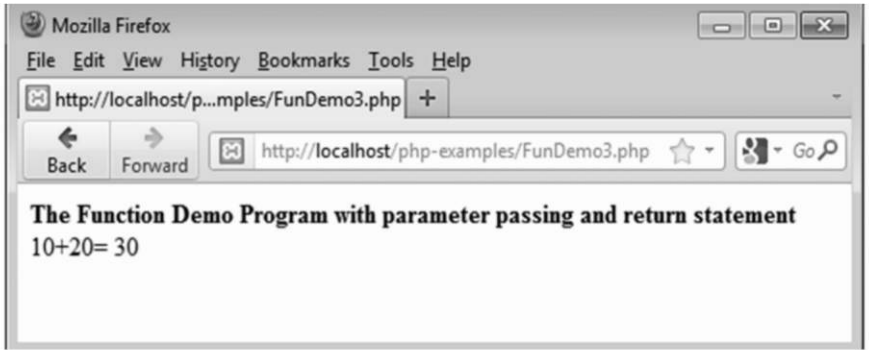
- The parameters that we pass to the function during the call is called the **actual parameter.** These parameters are generally the expressions.
- The parameters that we pass to the function while defining it is called the **formal parameters.** These are generally the variables. It is not necessary that the number of actual parameters should match with the number of formal parameters.
- If there are few actual parameter and more formal parameters then the value of formal parameter is will be some unbounded one.
- If there are many actual parameters and few formal parameters then the excess of actual parameters will be ignored.
- The default parameter passing technique in PHP is **pass by value.** The parameter passing by value means the values of actual parameters will be copied in the formal parameters. But the values of formal parameters will not be copied to the actual parameters.
- Following PHP script illustrates the functions with parameters

PHP Document[FunDemo3.php]

```
<?php
function Addition($a,$b)
{
 $c=$a+$b;
 return $c;
}
print "<b>The Function Demo Program with parameter passing and return statement</b>";
print "<br/>";
$x=10;
$y=20;
print "10+20= ".Addition($x,$y);
?>
```

4

Output



There are two ways to pass parameters by reference.

1. **Add & at the beginning of the name of the formal parameter.** For example -

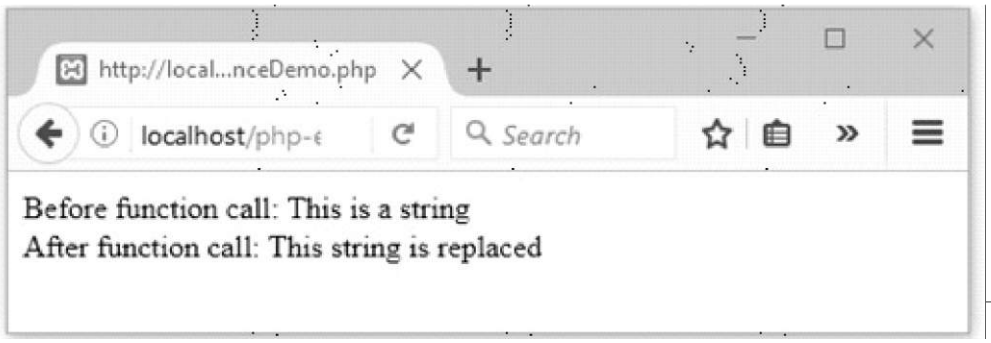
```
<?php
function add_some_extra($string)
{
   $string= 'This string is replaced';
}

$str = 'This is a string ';
$str1=&$str;  //adding & at the beginning of the name of formal parameter.
print "Before function call: $str<br/>";
add_some_extra($str1);
print "After function call: $str<br/>";
?>
```

2. **Add & to actual parameter in the function call.** For example -

```
<?php
function add_some_extra(&$string)   //adding & to actual parameter in function call
{
   $string= 'This string is replaced';
}

$str = 'This is a string ';
print "Before function call: $str<br/>";
add_some_extra($str);
print "After function call: $str<br/>";
?>
```

The output of the above code is one and the same it will be as follows -

Output



5

2.14.3 The Scope of Variables

- In PHP the scope of the variable is **local.** That means we can define a variable within a function and by the same name another variable can be defined outside the function. These two variables are considered to be different entities.
- Example :

```
<?php
function myfun()
{
    $a=10;
    print "The value of a=$a";
}
myfun();
print "<br/>";
$a=20;
print "The value of a=$a";
?>
```

Output

```
The value of a=10
The value of a=20
```

- We can define the **global variable** also. Following is the PHP script -

PHP Document

```
<?php
$a=10;
function myfun()
{
    global $a;
    $a+=10;
    print "The value of a=$a";
}
myfun();
print "<br/>";
$a=$a-5;
print "The value of a=$a";
?>
```

Output

```
The value of a=20
The value of a=15
```

2.14.4 The Life Time of Variables

- The lifetime of a variable can be defined as the time from which it is used first to the end of function execution.
- In PHP the **static variable** is used to remember the previous values.
- The lifetime of static variable is the time when the variable if first used and it ends when the script terminates its execution.
- For the declaration of the static variable the keyword **static** is used. For example -

```
function myfun()
{
  static $count=0;
  count++;  print "The number of times you have visited
this page is $count <br/>";
}
```

2.15 File Handling

PHP is known as a server-side scripting language. Hence file handling functions such as create, read, write , append are some file related operations that are supported by PHP.

2.15.1 Opening and Closing Files

- The first step in file handling is opening of the file.
- It takes two parameters - The first parameter of this function contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened.

Modes	Description
r	Read only. Starts reading from the beginning of the file.
r+	Read/Write. Starts reading from the beginning of the file.
w	Write only. Opens and clears the contents of file; or creates a new file if it is not created.
w+	Read/Write. Opens and clears the contents of file; or creates a new file if it is not created.
a	Append. Opens and writes to the end of the file or creates a new file if it is not created.
a+	Read/Append. Preserves file content by writing to the end of the file.

For example :

```
$my_file = 'file.txt';
$file_handle = fopen($my_file, 'a')
                or die('Cannot open file:  '.$my_file);
```

The **fopen** function returns TRUE if the required file is opened.

2.15.2 Reading from File

- The **fread** is the function which is used to read the file.
- It takes two parameters. The first parameter is the handle to the file and the second parameter is the number of bytes to be read. The **filesize** is the function which takes the filename as the parameter. For example -

```
$mystring = fread($file_handle, filesize("file.txt"));
```

- There is another function named **file_get_contents** using which the contents of the file can be obtained.
- The **fgets()** function is used to read a single line from the file. For example, following code displays the contents of the file line by line. -

```
while(!feof($file_handle))
 {
     echo fgets($file_handle). "<br />";
 }
```

Ex. 2.15.1 *Write a PHP program to read a text file line by line and to display it on screen.*

Sol. :

Step 1 : Create a input file Myfile.txt as follows

```
Hello
everybody
how are
you ?
```

Step 2 : Create a PHP script for reading the input file line by line as follows -

Filedemo.php

```
<?php
$file = fopen("Myfile.txt","r");
while(! feof($file))
{
  echo fgets($file). "<br />";
}
fclose($file);
?>
```

Output

2.15.3 Writing to a File

- The **fwrite** is the function which is used to write the contents to the file.
- It takes two parameters - The first parameter is the handle to the file and the second parameter is the number of bytes to be written. For example -

```
$Written_string = fread($file_handle, $my_data);
```

2.15.4 Locking Files

Multiple PHP scripts can access the same file at a time. But this causes a conflict problems. That means - there can be the situation in which one script is reading the file and at the same time other script is writing to that file. Sometimes there can be a situation in which two scripts are trying to write different data to the same file. These are totally undesirable in file handling technique.

The solution to this problem is to lock the file when one script is accessing it. Due to locking of the file simultaneous access to the same file by two different scripts can be avoided.

The php uses flock function for locking the files

Syntax of flock is

flock(file,lock,block)

where

file is the name of the file which need to be accessed.

lock is a kind of lock being used. Possible values are

LOCK_SH - Shared lock (reader). Allow other processes to access the file

LOCK_EX - Exclusive lock (writer). Prevent other processes from accessing the file

LOCK_UN - Release a shared or exclusive lock

LOCK_NB - Avoids blocking other processes while locking

block is optional parameter.

Example

```
<?php
$file = fopen("myfile.txt","w+");
// exclusive lock
    flock($file,LOCK_EX))
    fwrite($file,"I am Writing this line to file");
    // release lock
    flock($file,LOCK_UN);
    fclose($file);
?>
```

2.16 PHP and HTML

The PHP code can be embedded within the HTML document. The sample script is as given below -

```
<html>
 <head>
    <title> PHP Demo </title>
 </head>
 <body>
    <?php
    echo "<h3>Welcome to first PHP Document</h3>";
    ?>
 </body>
</html>
```

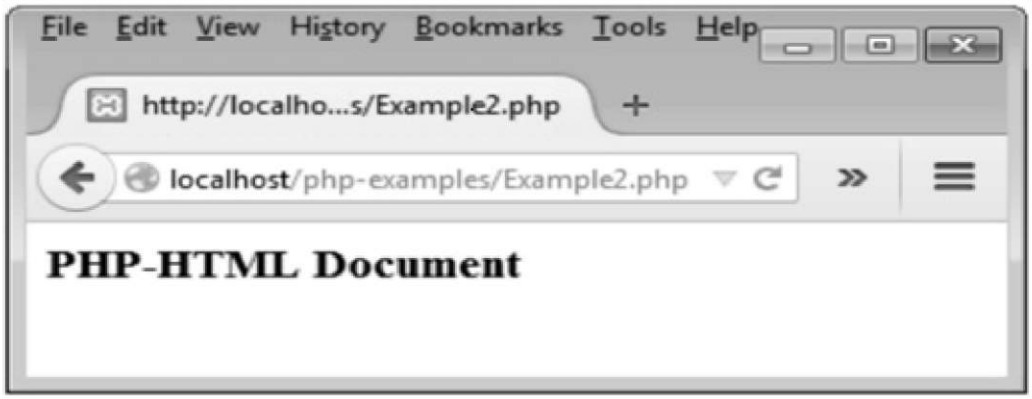
Output

We can add the HTML tags within the PHP code to enhance the look of the web page on the web browser. Following sample script illustrates this -

```
<?php
echo "<h3> PHP-HTML Document</h3>";
  ?>
```

8

Output



2.16.1 Form Handling

- PHP is used for form handling. For that purpose the simple form can be designed in XHTML and the values of the fields defined on the form can be transmitted to the PHP script using GET and POST methods.
- For forms that are submitted via "GET" method, we can obtain the form via the **$_GET** array variable.
- For forms that are submitted via "POST" method, we can obtain the form via the **$_POST** array variable.

Ex. 2.16.1 *Create HTML form with one textbox to get user's name. Also write PHP code to show length of entered name when, the HTML form is submitted.*

Sol. :

Step 1 : The HTML form can be created as follows

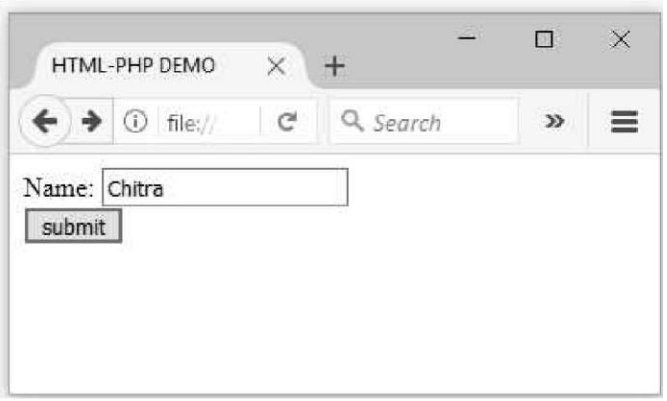
```
<!DOCTYPE html>
<html>
<head><title> HTML-PHP DEMO</title>
</head>
<body>
   <form method="post"
action="http://localhost/getdata.php">
    Name: <input type="text" name="myname" size="20"/>
    <br/>
    <input type="submit" name="submit"
value="submit"/>
   </form>
</body>
</html>
```

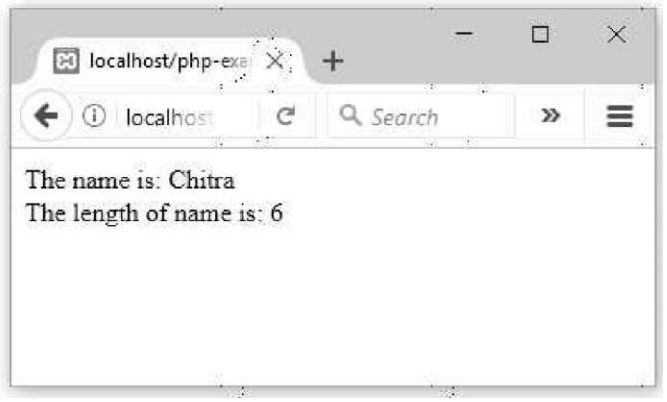
Step 2 :

The PHP script to display the length of submitted name is as written below

```
<?php
print "The name is: ";
print $_POST["myname"];
$len= strlen($_POST["myname"]);
print "<br/> The length of name is: ";
print $len;
?>
```

Output





9

Ex. 2.16.2 *Create HTML form to enter one number. Write PHP code to display the message about number is odd or even.*

Sol. :

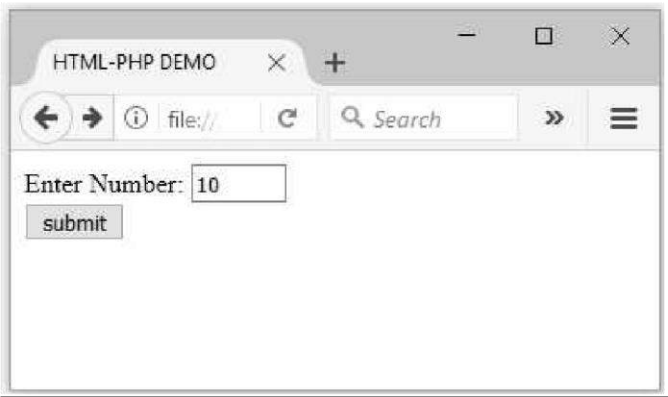
Step 1 : The HTML form for accepting number is created as below -

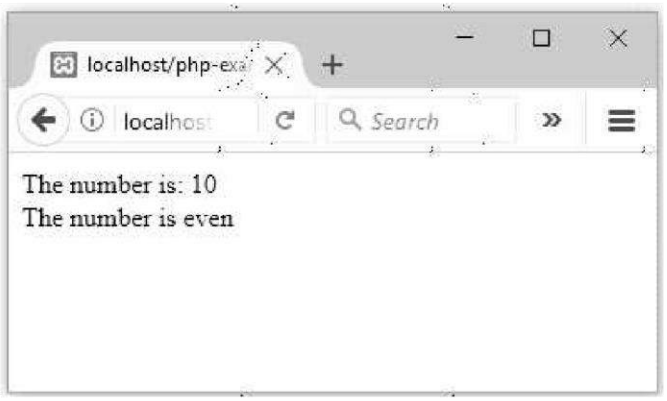
```
<!DOCTYPE html>
<html >
 <head><title> HTML-PHP DEMO</title>
</head>
<body>
   <form method="post"
action="http://localhost/getdata.php">
    Enter Number: <input type="text" name="mynum"
                  size="5"/>
    <br/>
   <input type="submit" name="submit" value="submit"/>
   </form>

</body>
</html>
```

Step 2 : The PHP script deciding whether the number is even or odd is as given below -

```
<?php
print "The number is: ";
print $_POST["mynum"];
$a= $_POST["mynum"];
if($a%2==1)
 print "<br/> The number is odd ";
else
 print "<br/> The number is even ";
?>
```

Output





Ex. 2.16.3 *Create a form containing information sl.no, title of the book, publishers, quantity,price read the data from the form and display it using PHP script.*

Sol. :

Step 1 : Create an HTML page for inputting the data. Following is the code for HTML script .

HTML Document[input.html]

```
<!doctype html>
<html>
<head>
<title> Book Order Form </title>
</head>
<body>
<h3> Enter the Book Data </h3>
<form method="post"
action="http://localhost/php-examples/formdemo.php">
<table>
<tr>
<td>Sr.No.</td>
<td><input type="text" name="SLNo"></td>
</tr>
<tr>
 <td>Book name</td>
 <td><input type="text" name="BName"></td>
</tr>
<tr>
 <td>Publisher</td>
 <td><input type="text" name="PUBName"></td>
</tr>
<tr>
 <td>Price</td>
 <td><input type="text" name="Price"></td>
</tr>
```

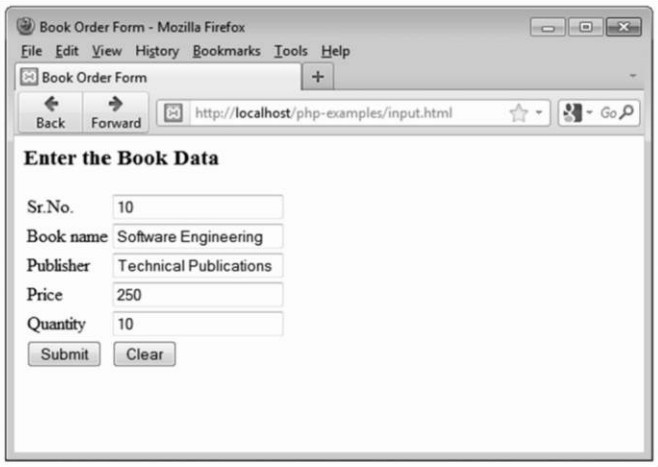
```
 <td>Quantity</td>
 <td><input type="text" name="Qty"></td>
</tr>
<tr>
 <td><input type="submit" value="Submit"></td>
 <td><input type="submit" value="Clear"></td>
</tr>
</table>
</form>
</body>
</html>
```

Step 2 : Create a PHP script which will read out the data entered by the user using HTML form. The code is as follows -

PHP Document[formdemo.php]

```
<html>
<head>
 <title>Book Information</title>
</head>
<body>
<?php
$BName=$_POST["BName"];
$PUBName=$_POST["PUBName"];
$Price=$_POST["Price"];
$Qty=$_POST["Qty"];
?>
<center>
<h3> Book Data </h3>
<table border=1>
<tr>
 <th>Book name</th>
 <th>Publisher</th>
 <th>Price</th>
 <th>Quantity</th>
</tr>
<tr>
 <td><?php print ("$BName"); ?></td>
 <td><?php print ("$PUBName"); ?></td>
 <td><?php printf("%3.2f",$Price); ?></td>
 <td><?php printf("%d",$Qty); ?></td>
</tr>
</table>
</center>
</body>
</html>
```

Step 3 : Open some suitable web browser and enter the address for the HTML file which you have created in step 1.



Now click on the Submit button and the PHP file will be invoked. The output will then be as follows -

Ex. 2.16.4 *Write a PHP program to accept a positive integer 'N' through a HTML form and to display the sum of all the numbers from 1 to N*

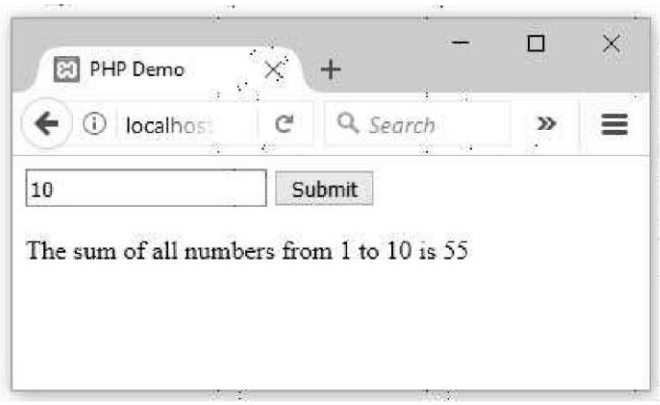
Sol. :

```
<html>
   <head>
      <title> PHP Demo </title>
   </head>
   <body>
      <form method="post">
         <input type="text" name="num"/>
         <input type="submit" value="Submit"/>
      </form>
   </body>
</html>
```

1

```
<?php
    //getting values from HTML form
    $n = intval($_POST['num']);
    $sum=0;
    for($i=1;$i<=$n;$i++)
        $sum=$sum+$i;
    echo "The sum of all numbers from 1 to ".$n." is ".$sum;
?>
```

Output



2.17 PHP and MySQL

2.17.1 Benefits of using PHP and MySQL

- PHP is a server side scripting language and it has an **ability** to create **dynamic pages** with customized features. Using PHP-MySQL user friendly and **interactive web sites** can be created.
- Both PHP and MySQL are **open-source technologies** that work hand-in-hand to create rich internet applications. The purchased code provides you the encrypted source code to prevent replication or modification, whereas open-source programs encourage users to utilize, scrutinize and customize the code.
- Due to availability of these technologies as free of cost, the **cost effective web solutions** can be created.
- PHP-MySQL are **stable technologies** and have **cross platform compatibility.** Hence the web application developed using these technologies become **portable.**
- Since **HTML can be embedded** within the PHP, there is no need to write separate code for web-scripting.
- Open-source coding has been checked and doubled checked by thousands or even millions of people around the world. Hence one can built the **reliable web application** using these technologies.
- The PHP has got the **support** from several content management programs such as WordPress, Joomla, Drupal and so on.
- It has got a strong support for developing **e-commerce applications** using the technologies such as Ecommerce, Drupal and so on.
- The most **popular web sites** being developed using PHP and MySQL technologies are -
  1. Facebook
  2. WordPress
  3. Wikipedia
  4. Yahoo

2.17.2 Structured Query Language(SQL)

MYSQL is a open source database product and can be downloaded from the web site **http://dev.mysql.com/downloads/mysql.**

MYSQL is a kind of database in which the records are stored in an entity called **tables.** In the tables the data is arranged in the rows and columns. We can query a database to retrieve particular information. **Query** is a request or a question for the database. There is a common practice of making use of Structured Query Language(SQL).

2.17.3 Connecting PHP to MYSQL

- The PHP function **mysql_connect** connects to the MYSQL server. There are three parameters that can be passed to this function. For example -

```
mysql_connect("localhost", "root", "mypassword") or die(mysql_error());
```

Local host on which the MYSQL is running — Root — Password

//

2

- The database can be selected by using the command **mysql_select_db.**

For example

mysql_select_db("test")

will select the database named **test.**

2.17.4 Requesting MySQL Operations

1. Creating Database

- We can create a database using the function **mysql_query.** The **mysql_error()** function is used to obtain the error messages if any command gets failed.
- **mysql_query** function in php is used to pass a sql query to mysql database.

Syntax

mysql_query (string query [, resource link_identifier])

This function returns the query handle for SELECT queries, TRUE/FALSE for other queries, or FALSE on failure.

Example

mysql_query("CREATE DATABASE mydb",$con)

- The **mysql_connect()** function Open a connection to a MySQL Server.

Syntax

mysql_connect (string server , string username , string password)

Returns a MySQL link identifier on success, or FALSE on failure.

Example

```
$conn=mysql_connect("localhost ","root","mypassword");
```

- The **mysql_close()** function is used to close the database connection.

Syntax

mysql_close (Connection)

PHP Example For Creating Database

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
  Create a database
if(mysql_query("CREATE DATABASE mydb",$conn))
{
  print "Database created";
}
else
{
  print "Error creating database : " . mysql_error();
}
mysql_close($conn); //closing the database
?>
```

2. Selecting Database

- The database can be selected using the function **mysql_select_db().**

Syntax

mysql_select_db (string database_name [, resource link_identifier])

Where

mysql_select_db() attempts to select existing database on the server associated with the specified link identifier. It returns TRUE on success, or FALSE on failure.

For example -

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost :3306/mydb
","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Select a database
mysql_select_db("mydb", $conn);
mysql_close($conn); //closing the database
?>
```

3. Counting Number of Rows

The number of rows present in the database table can be obtained using **mysql_num_rows** function.

Syntax

int mysql_num_rows (resource $result)

This returns number of rows in result on success, or NULL on error.

//

nple

ıp

ıke a MySQL Connection

n=mysql_connect("localhost :3306/mydb

Make a MySQL Connection

$conn=mysql_connect("localhost ","root","mypassword")

if(!$conn)

{

```
//
4
mysql_query($query,$conn);//Execution of Query

mysql_close($conn); //closing the database
?>
```

Sometimes values that can be inserted in the table can be obtained from some another script and these values might be present in the variables. Insertion of such data can be done using $_POST variables. It is as shown below -

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$conn); //select the database
$query=" INSERT INTO my_table (id,name)
VALUES('$_POST[MyID]','$_POST[MyName]')";
mysql_query($query,$conn);//Execution of Query

mysql_close($conn); //closing the database
?>
```

7. Displaying or Retrieving Records

For displaying the records present in the database table, we use SELECT query. For example

```
//Execution of Query for displaying the data
$result=mysql_query("SELECT * FROM my_table");
```

The above execution returns a result handle. Then The **mysql_fetch_array()** is used to retrieve a row of data as an array from a MySQL result handle.

Purpose of mysql_fetch_array() :

The mysql_fetch_array() is used to retrieve a row of data as an array from a MySQL result handle.

Syntax :

```
mysql_fetch_array(result, result_type)
```

PHP Script for Displaying records

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$conn); //select the database
  Execution of Query for displaying the data
$result=mysql_query("SELECT * FROM my_table");
 while($row = mysql_fetch_array($result))
 {
 echo $row['id'] . " " . $row['name'];        //Each record will
                                                be displayed
                                                  //line by line
  echo "<br />";
 }
mysql_close($conn); //closing the database
?>
```

8. Finding number of affected rows

The mysql_affected_rows query is used to get number of affected rows in previous MySQL operation such as INSERT, DELETE, UPDATE queries.

Syntax

```
mysql_affected_rows(connection)
```

Example

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$con); //select the database
$query=" INSERT INTO my_table (id,name)
VALUES(1,'SHILPA')";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO my_table (id,name)
VALUES(2,'MONIKA')";
mysql_query($query,$con);//Execution of Query
echo "Number of rows affected are
:".mysql_affected_rows();
mysql_close($conn); //closing the database
?>
```

Ex. 2.17.1 *Write a PHP script to create a new database table with 4 fields of your choice and perform following database operations. i) insert ii) update iii) Delete*

Sol. : We will create a table in the database **test.** The name of the table is mytable. Then we will insert the record into the table using the INSERT command,update particular field of the record using the command UPDATE and Delete the record using the command DELETE.

The PHP script is as follows -

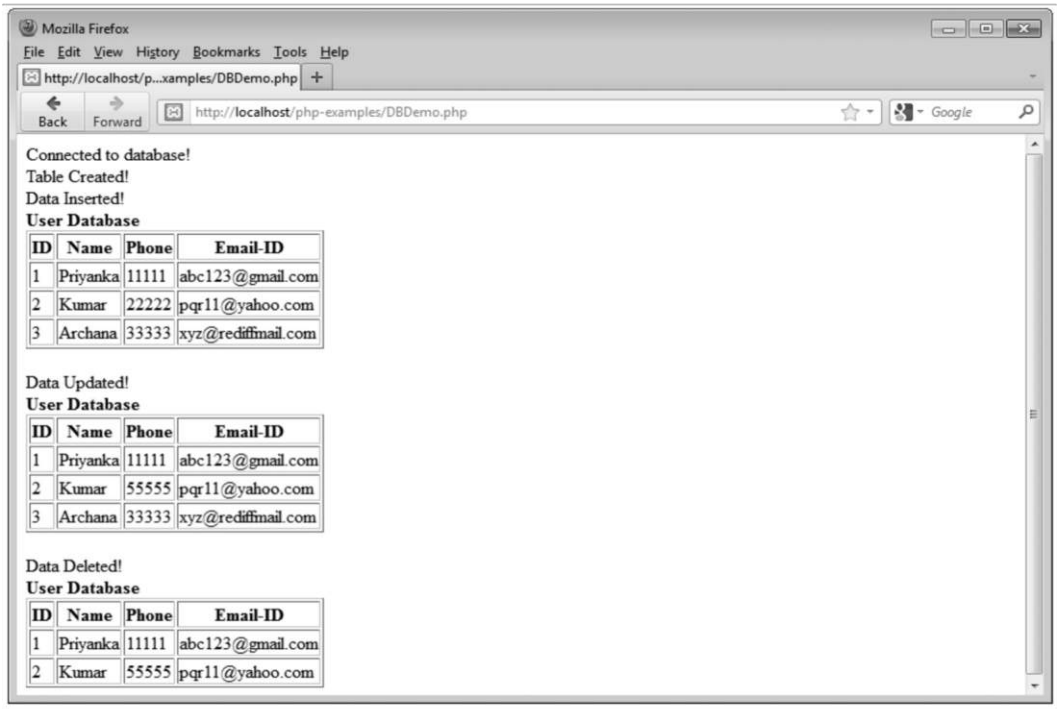
PHP Document[DBDemo.php]

```
<?php
// Make a MySQL Connection
mysql_connect("localhost", "root", "mypassword") or die(mysql_error());
mysql_select_db("test") or die(mysql_error());
echo "Connected to database!";
mysql_query("CREATE TABLE mytable(id INT NOT NULL AUTO_INCREMENT, PRIMARY KEY(id), name
VARCHAR(30),
 phone INT,emailId VARCHAR(30))")
 or die(mysql_error());
print "<br/>";
echo "Table Created!";

// Insert a row of information into the table "example"
mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Priyanka', '11111','abc123@gmail.com' ) ")
or die(mysql_error());

mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Kumar', '22222','pqr11@yahoo.com' ) ")
or die(mysql_error());

mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Archana', '33333','xyz@rediffmail.com' ) ")
or die(mysql_error());
print "<br/>";
echo "Data Inserted!";
$result =mysql_query("SELECT * FROM mytable")
or die(mysql_error());
print "<br/>";
print "<b>User Database</b>";
echo "<table border='1'>";
echo "<tr><th>ID</th> <th>Name</th> <th>Phone</th><th>Email-ID</th> </tr>";
while($row = mysql_fetch_array( $result ))
  {
  // Print out the contents of each row into a table
  echo "<tr><td>";
  echo $row['id'];
  echo "</td><td>";
  echo $row['name'];
  echo "</td><td>";
  echo $row['phone'];
  echo "</td><td>";
  echo $row['emailId'];
  echo "</td></tr>";
}
echo "</table>";
```

```
$result = mysql_query("UPDATE mytable SET phone='55555' WHERE phone='22222'")
or die(mysql_error());

print "<br/>";
```

7

Output



Ex. 2.17.2 *Create a HTML form "result.html" with a text box and a submit button to accept registration number of the student. Write a "result.php" code to check the status of the result from the table to display whether the student has "PASS" or "FAIL" status. Assume that the MYSQL database "my_db" has the table "result_table" with two columns REG_NO and STATUS*

Sol. :

Step 1 : Create a database named my_db. Create a table result_table for this database and insert the values in this table. The table is created as follows -

Sr. No.	REG_NO	STATUS
1.	101	PASS
2.	102	FAIL
3.	103	PASS
4.	104	FAIL
5.	105	PASS

Table 2.17.1

Step 2 : Create an HTML form to accept the registration number, the HTML document is as follows -

result.html

```
<!DOCTYPE html>
<html>
    <head>
        <title> STUDENT RESULT </title>
    </head>
    <body>
        <form name="myform" method="post"
action="http://localhost/php-examples/result.php">
            <input type="text" name="reg_no"/>
            <input type="submit" value="Submit"/>
        </form>
    </body>
</html>
```

Step 3 : Create a PHP script to accept the registration number. This php script will connect to MYSQL database and the status(PASS or FAIL) of the corresponding registration number will be displayed.

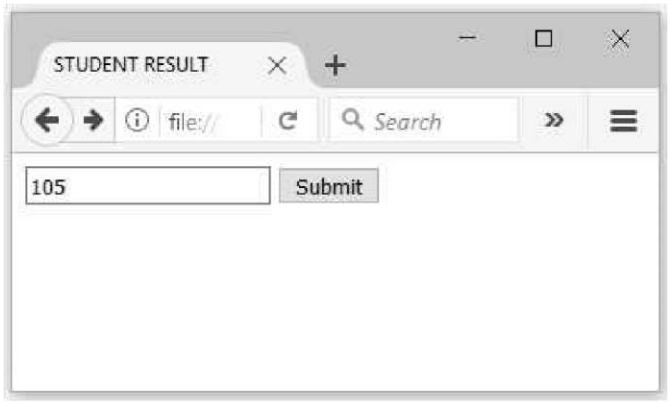
result.php

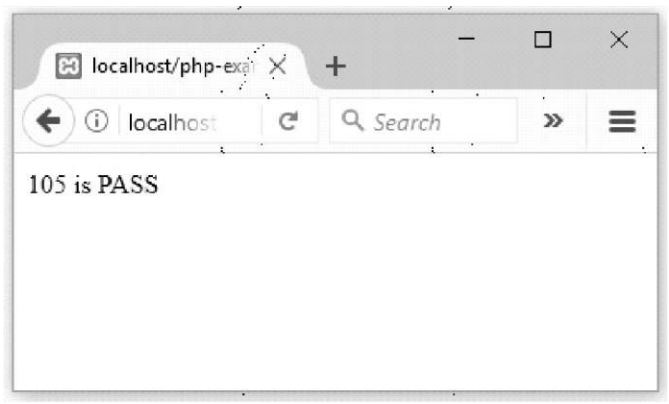
```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","");
if(!$conn)
```

8

```
{
die('error in connection'.mysql_error());
}
mysql_select_db("my_db",$conn); //select the database
//Execution of Query for displaying the data
$reg_no = intval($_POST['reg_no']);

$result=mysql_query("SELECT REG_NO,STATUS FROM
result_table where REG_NO=$reg_no ");
 while($row = mysql_fetch_array($result))
  {
   echo $row['REG_NO'] . " is " . $row['STATUS'];
   echo "<br />";
  }
mysql_close($conn); //closing the database
?>
```

Step 4 : Load the HTML form created in Step 2 and click the submit button by entering some registration number.





Ex. 2.17.3 *Write a PHP program to delete a record from result_table(Refer example 2.17.2)*

Sol. :

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("my_db",$conn); //select the database
//Execution of Query for displaying the data
$reg_no = intval($_POST['reg_no']);

$result=mysql_query("DELETE FROM result_table where
REG_NO=$reg_no");
 while($row = mysql_fetch_array($result))
  {
   echo $row['REG_NO'] . " is " . $row['STATUS'];
                          //Each record will be displayed
                             //line by line
   echo "<br />";
  }

mysql_close($conn); //closing the database
?>
```

Ex. 2.17.4 *Write a user defined function 'CalculateInterest' using PHP to find the simple interest to be paid for a loan amount. Read the loan amount, the number of years and rate of interest from a database table called LOANDETAILS having three fields AMT, YEARS, and RATE, and Calculate the interest using the user defined function.*

Sol. :

Step 1 : Create a database table named LOANDETAILS having three fields AMT, YEARS and RATE. Insert the values in it. The sample table will be as follows -

AMT	YEARS	RATE
10000	5	6.9

Table 2.17.2

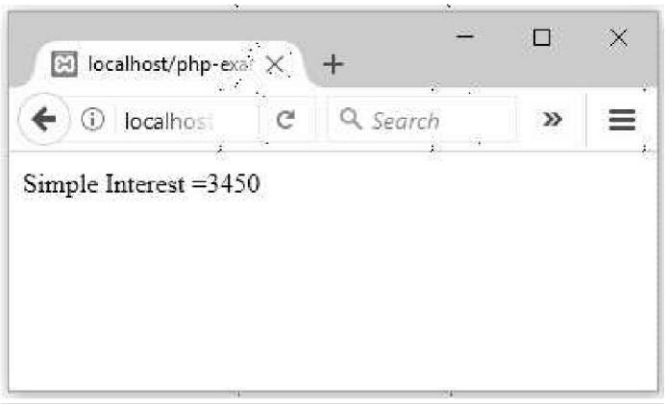
Step 2 : The PHP code for calculating the simple interest the above values in a function will be as follows -

Interest.php

```
<?php
function CalculateInterest()
```

```
{
    // Make a MySQL Connection
    $conn=mysql_connect("localhost","root","");
    if(!$conn)
    {
        die('error in connection'.mysql_error());
    }
    mysql_select_db("my_db",$conn); //select the database
    //Execution of Query for displaying the data
    $result=mysql_query("SELECT * FROM
LOANDETAILS");
    $row = mysql_fetch_array($result);
    $amount= $row['AMT'];
    $rate= $row['RATE'];
    $years= $row['YEARS'];
      $interest=($amount * $rate *$years)/100;
    mysql_close($conn); //closing the database
    return $interest;
}
print "Simple Interest =".CalculateInterest();
?>
```

Output



Ex. 2.17.5 *Consider a table PRODUCT_DETAILS with PID, PNAME and UNIT_PRICE. Write HTML form to accept PID and QUANTITY_REQUIRED from the user and display the total amount to be paid in another textbox. Get the unit price for given PID from a database table.*

Sol. :

Bill.php

```
<html>
    <head>
        <title> PHP Demo </title>
    </head>
    <body>
        <form method="post">
            <input type="text" name="PRODUCT_ID"/>
            <input type="text" name="QUANTITY"/>
            <input type="submit" value="Submit"/>

    <?php
    //getting values from HTML form
    $id = intval($_POST['PRODUCT_ID']);
    $qty=intval($_POST['QUANTITY']);

    // Make a MySQL Connection
    $conn=mysql_connect("localhost","root","");
    if(!$conn)
    {
        die('error in connection'.mysql_error());
    }
    mysql_select_db("my_db",$conn); //select the database
    //Execution of Query for displaying the data
    $result=mysql_query("SELECT UNIT_PRICE FROM
product_table where PID=$id ");
      $row = mysql_fetch_array($result);
      $price= $row['UNIT_PRICE'];
    $total=$price* $qty;
      mysql_close($conn); //closing the database
    ?>
    <input type="text" name="amount" value="<?php echo
@$total;?>" />
    </form>
    </body>
</html>
```

2.18 Cookies

In this section we will discuss how to create and read cookies.

2.18.1 Introduction to Cookies

- Cookie is a small file that server embeds in the user's machine.
- Cookies are used to identify the users.
- A cookie consists of a **name** and a **textual value.** A cookie is created by some software system on the server.
- In every HTTP communication between browser and server a header is included. The header stores the information about the message.
- The header part of http contains the cookies.
- There can be one or more cookies in browser and server communication.

2.18.2 PHP Support for Cookies

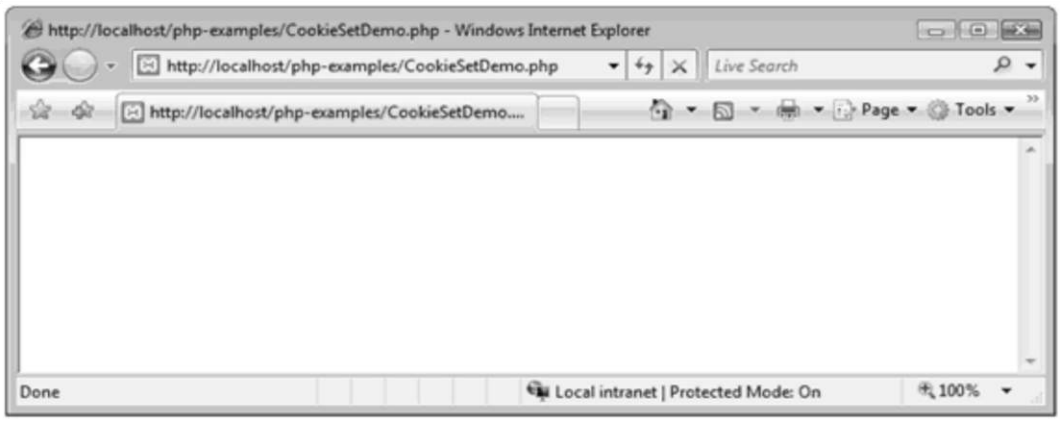
- PHP can be used to create and retrieve the cookies.
- The cookie can be set in PHP using the function called **setcookie()**
- The syntax for the cookie is -

  setcookie(name,value,expire_period,path,domain)
- Following is a simple PHP document which illustrates how to set the cookies -

PHP Document[CookieSetDemo.php]

```
<?php
$Cookie_period=time()+60*60*24*30;
setcookie("Myname", "Monika", $Cookie_period);
?>
```

Output



Note that you have got the blank screen it indicates that the cookie is set. In above PHP document we have set the PHP script for one month. Just observer the third parameter of the setcookie function.

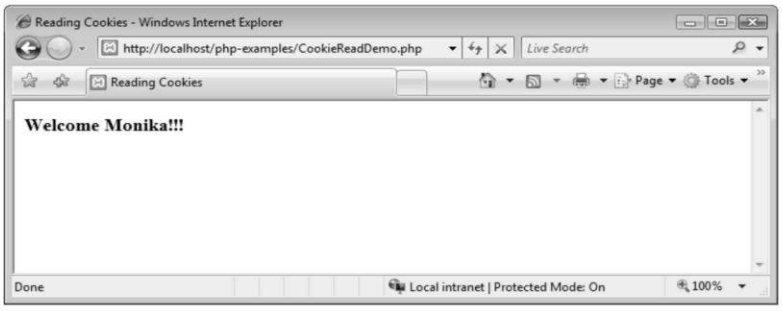
Now you can retrieve the cookie and read the value to ensure whether or the cookie is set.

PHP Document[CookieReadDemo.php]

```
<html>
 <head><title>Reading Cookies</title>
<body>
 <?php
 if (isset($_COOKIE["Myname"]))
   echo "<h3>Welcome " . $_COOKIE["Myname"]."!!!</h3>";
 else
   echo "<h3>Welcome guest!</h3>";
 ?>
</body>
</html>
```

1

Output



Program Explanation

The **isset** function is used for checking whether or not the cookie is set. Then using the $_COOKIE the value of the cookie can be retrieved.

Review Question

1. *What are cookies and how are they used ?*

Two Marks Questions with Answers

Q.1 *Why JavaScript is called client side scripting ?*

Ans. : JavaScript code is written into an HTML page. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it's up to the browser to do something with it. Hence JavaScript is called client side scripting language.

Q.2 *How you use Form's Action Attribute and Submit Button in HTML ?*

Ans. : The action attribute specifies where to send the form-data when a form is submitted.

For example - On clicking the submit button, the **mypage.php** page will be invoked for further processing.

```
<form action="mypage.php" method="get">
 First name: <input type="text" name="name"><br>
 <input type="submit" value="Submit">
</form>
```

Q.3 *Write a piece of JavaScript code to check if the value of two fields - 'password' and 'confirmPassword' are the same, and display a message if they are not.*

Ans. :

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function Validate()
  {
var password = document.getElementById("Pwd").value;
var confirmPassword =
        document.getElementById("ConfirmPwd").value;
if (password != confirmPassword)
{
alert("Passwords do not match.");
return false;
}
return true;
}
</script>
</head>
<body>
<form>
Password:
<input type="password" id=":Pwd" />
Confirm Password:
<input type="password" id="ConfirmPwd" />
<input type="submit" value="Submit" onclick="return
        Validate()" />
</form>
</body>
</html>
```

Q.4 *What is the difference between GET and POST methods ?*

Ans. :

Sr.No.	GET	POST
1.	This method is used to request data from a specified resource.	This method is used to submit the data to be processed to a specific resource.
2.	The GET request remains in the browser history.	The POST request never remains in the browser history.

r

2

.	GET request is never used while dealing with sensitive information.	POST request is used when there is carrying of sensitive information.
.	GET requests have length restrictions	POST requests do not have length restriction

4. It is possible to perform system functions su as creating, opening, reading from, writing and closing the files on the system, exec the system commands, create directories, a

r

.3 *Write the output of the following code and explain*

```
html>
head>
```

false. print can be used as part of a more comp
expression whereas echo cannot. echo
marginally faster since it doesn't set a return val

Q.17 *What is the difference between "Inse*

r

:2 *What is the purpose of $Global variable in PHP ?*

s. : The $Global variable is a kind of variable always accessible, regardless of scope. We can

```
<?php
    echo "Welcome user ".$_POST["name"]
?>
```

Q.28 *What is the difference between fgets and fr*

CHAPTER 13

Cookies, Sessions, and Authentication

As your web projects grow larger and more complicated, you will find an increasing need to keep track of your users. Even if you aren't offering logins and passwords, you will still often need to store details about a user's current session and possibly also recognize people when they return to your site.

Several technologies support this kind of interaction, ranging from simple browser cookies to session handling and HTTP authentication. Between them, they offer the opportunity for you to configure your site to your users' preferences and ensure a smooth and enjoyable transition through it.

Using Cookies in PHP

A *cookie* is an item of data that a web server saves to your computer's hard disk via a web browser. It can contain almost any alphanumeric information (as long as it's under 4 KB) and can be retrieved from your computer and returned to the server. Common uses include session tracking, maintaining data across multiple visits, holding shopping cart contents, storing login details, and more.

Because of their privacy implications, cookies can be read only from the issuing domain. In other words, if a cookie is issued by, for example, *oreilly.com*, it can be retrieved only by a web server using that domain. This prevents other websites from gaining access to details for which they are not authorized.

Due to the way the Internet works, multiple elements on a web page can be embedded from multiple domains, each of which can issue its own cookies. When this happens,

they are referred to as *third-party cookies*. Most commonly, these are created by advertising companies in order to track users across multiple websites.

Because of this, most browsers allow users to turn cookies off either for the current server's domain, third-party servers, or both. Fortunately, most people who disable cookies do so only for third-party websites.

Cookies are exchanged during the transfer of headers, before the actual HTML of a web page is sent, and it is impossible to send a cookie once any HTML has been transferred. Therefore, careful planning of cookie usage is important. Figure 13-1 illustrates a typical request and response dialog between a web browser and web server passing cookies.

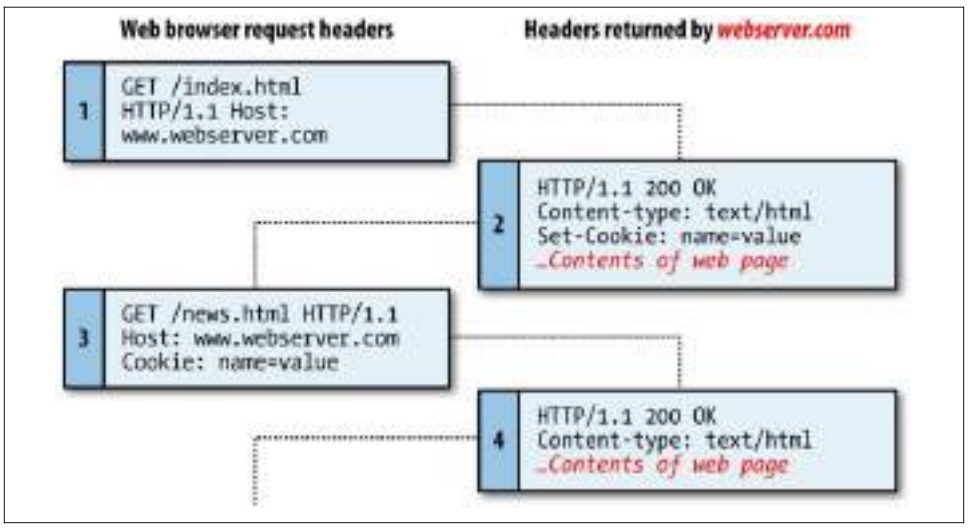


Figure 13-1. A browser/server request/response dialog with cookies

This exchange shows a browser receiving two pages:

1. The browser issues a request to retrieve the main page, *index.html*, at the website *http://www.webserver.com*. The first header specifies the file, and the second header specifies the server.
2. When the web server at *webserver.com* receives this pair of headers, it returns some of its own. The second header defines the type of content to be sent (*text/html*), and the third one sends a cookie of the name *name* and with the value *value*. Only then are the contents of the web page transferred.
3. Once the browser has received the cookie, it will then return it with every future request made to the issuing server until the cookie expires or is deleted. So, when the browser requests the new page */news.html*, it also returns the cookie *name* with the value *value*.

4. Because the cookie has already been set, when the server receives the request to send */news.html*, it does not have to resend the cookie, but just returns the requested page.

Setting a Cookie

Setting a cookie in PHP is a simple matter. As long as no HTML has yet been transferred, you can call the `setcookie` function, which has the following syntax (see Table 13-1):

```
setcookie(name, value, expire, path, domain, secure,
httponly);
```

Table 13-1. The setcookie parameters

Parameter	Description	Example
`name`	The name of the cookie. This is the name that your server will use to access the cookie on subsequent browser requests.	`username`
`value`	The value of the cookie, or the cookie's contents. This can contain up to 4 KB of alphanumeric text.	`Hannah`
`expire`	(*Optional*.) Unix timestamp of the expiration date. Generally, you will probably use `time()` plus a number of seconds. If not set, the cookie expires when the browser closes.	`time() + 2592000`
`path`	(*Optional*.) The path of the cookie on the server. If this is a `/` (forward slash), the cookie is available over the entire domain, such as *www.webserver.com*. If it is a subdirectory, the cookie is available only within that subdirectory. The default is the current directory that the cookie is being set in, and this is the setting you will normally use.	`/`
`domain`	(*Optional*.) The Internet domain of the cookie. If this is *.webserver.com*, the cookie is available to all of *webserver.com* and its subdomains, such as *www.webserver.com* and *images.webserver.com*. If it is *images.webserver.com*, the cookie is available only to *images.webserver.com* and its subdomains such as *sub.images.webserver.com*, but not, say, to *www.webserver.com*.	`.webserver.com`
`secure`	(*Optional*.) Whether the cookie must use a secure connection (*https://*). If this value is TRUE, the cookie can be transferred only across a secure connection. The default is FALSE.	FALSE
`httponly`	(*Optional*; implemented since PHP version 5.2.0.) Whether the cookie must use the HTTP protocol. If this value is TRUE, scripting languages such as JavaScript cannot access the cookie. (Not supported in all browsers.) The default is FALSE.	FALSE

So, to create a cookie with the name `username` and the value `Hannah` that is accessible across the entire web server on the current domain, and will be removed from the browser's cache in seven days, use the following:

```
setcookie('username', 'Hannah', time() + 60 * 60 * 24 *
7, '/');
```


Accessing a Cookie

Reading the value of a cookie is as simple as accessing the `$_COOKIE` system array. For example, if you wish to see whether the current browser has the cookie called `user name` already stored and, if so, to read its value, use the following:

```
if (isset($_COOKIE['username'])) $username =
$_COOKIE['username'];
```

Note that you can read a cookie back only after it has been sent to a web browser. This means that when you issue a cookie, you cannot read it in again until the browser reloads the page (or another with access to the cookie) from your website and passes the cookie back to the server in the process.

Destroying a Cookie

To delete a cookie, you must issue it again and set a date in the past. It is important for all parameters in your new `setcookie` call except the timestamp to be identical to the parameters when the cookie was first issued; otherwise, the deletion will fail. Therefore, to delete the cookie created earlier, you would use the following:

```
setcookie('username', 'Hannah', time() - 2592000, '/');
```

As long as the time given is in the past, the cookie should be deleted. However, I have used a time of 2592000 seconds (one month) in the past in case the client computer's date and time are not correctly set.

You can try PHP cookies for yourself using the file *php-cookies.php* in this chapter's matching folder of the accompanying examples archive (available for free at *http://lpmj.net*). The folder also contains the file *javascript-cookies.htm*, which does the same thing using JavaScript.

HTTP Authentication

HTTP authentication uses the web server to manage users and passwords for the application. It's adequate for most applications that ask users to log in, although some

applications have specialized needs or more stringent security requirements that call for other techniques.

To use HTTP authentication, PHP sends a header request asking to start an authentication dialog with the browser. The server must have this feature turned on in order for it to work, but because it's so common, your server is very likely to offer the feature.

Although it is usually installed with Apache, HTTP authentication may not necessarily be installed on the server you use. So attempting to run these examples may generate an error telling you that the feature is not enabled, in which case you must install the module, change the configuration file to load the module, or ask your system administrator to do these fixes.

After entering your URL into the browser or visiting via a link, the user will see an "Authentication Required" prompt pop up requesting two fields: User Name and Password (Figure 13-2 shows how this looks in Firefox).

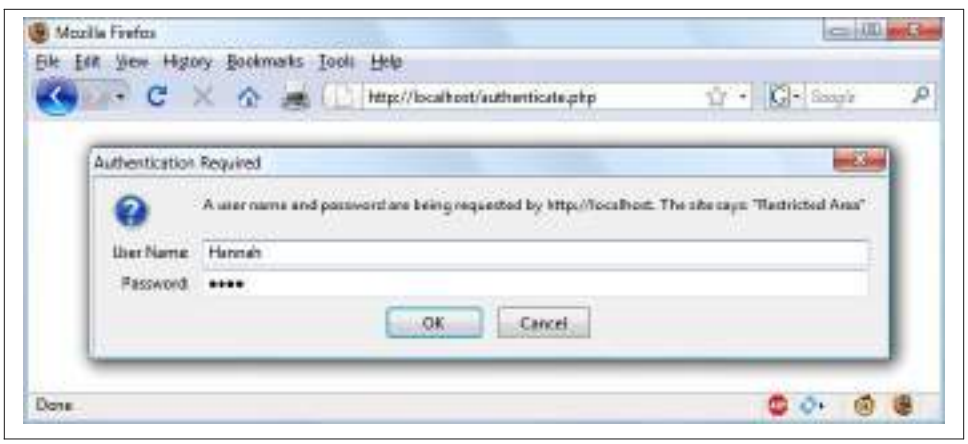


Figure 13-2. An HTTP authentication login prompt

Example 13-1 shows the code to make this happen.

Example 13-1. PHP authentication

```
<?php
  if (isset($_SERVER['PHP_AUTH_USER']) &&
isset($_SERVER['PHP_AUTH_PW']))   {     echo
"Welcome User: " . $_SERVER['PHP_AUTH_USER']
.
        " Password: "    . $_SERVER['PHP_AUTH_PW'];
  }
else
  {
```

```
    header('WWW-Authenticate: Basic realm="Restricted
Section"');
    header('HTTP/1.0 401 Unauthorized');
    die("Please enter your username and password");
  }
?>
```

> By default, the type of interface Zend Server uses is *cgi-fcgi*, which is incompatible with basic authentication. However, configuring Zend is beyond the scope of this book, so if you are using it for Examples 13-1 through 13-5, you may prefer to test them on a different server. To determine the interface of a server, you can call the `php_sapi_name` function, which will return a string such as `'cgifcgi'`, `'cli'`, and so on. Basic authentication is not recommended anyway on a production website, as it is very insecure, but you need to know how it works for maintaining legacy code. For further details, refer to *http://php.net/php_sapi_name*.

The first thing the program does is look for two particular values: `$_SERVER['PHP_AUTH_USER']` and `$_SERVER['PHP_AUTH_PW']`. If they both exist, they represent the username and password entered by a user into an authentication prompt. If either of the values does not exist, the user has not yet been authenticated and you display the prompt in Figure 13-2 by issuing the following header, where `Basic realm` is the name of the section that is protected and appears as part of the pop-up prompt:

```
WWW-Authenticate: Basic realm="Restricted Area"
```

If the user fills out the fields, the PHP program runs again from the top. But if the user clicks the Cancel button, the program proceeds to the following two lines, which send the following header and an error message:

```
HTTP/1.0 401 Unauthorized
```

The `die` statement causes the text “Please enter your username and password” to be displayed (see Figure 13-3).

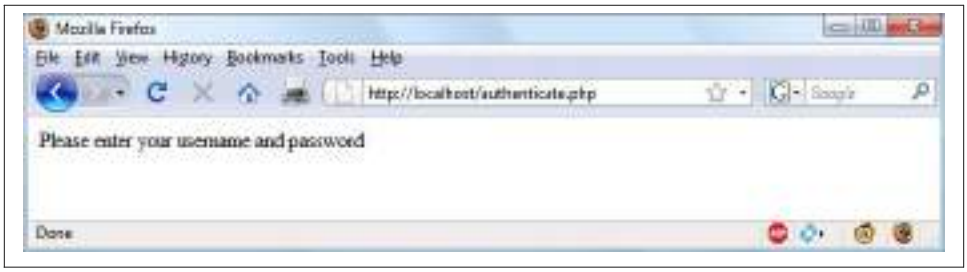


Figure 13-3. The result of clicking the Cancel button

> Once a user has been authenticated, you will not be able to get the authentication dialog to pop up again unless the user closes and reopens all browser windows, as the web browser will keep returning the same username and password to PHP. You may need to close and reopen your browser a few times as you work through this section and try different things out.

Now let's check for a valid username and password. The code in Example 13-1 doesn't require you to change much to add this check, other than modifying the previous welcome message code to test for a correct username and password, and then issuing a welcome message. A failed authentication causes an error message to be sent (see Example 13-2).

Example 13-2. PHP authentication with input checking

```
<?php
  $username = 'admin';
  $password = 'letmein';

  if (isset($_SERVER['PHP_AUTH_USER']) &&
isset($_SERVER['PHP_AUTH_PW']))
  {
    if ($_SERVER['PHP_AUTH_USER'] ==
$username &&
$_SERVER['PHP_AUTH_PW']   == $password)
echo "You are now logged in";
    else die("Invalid username / password combination");
  }
else
  {
```

```
    header('WWW-Authenticate: Basic realm="Restricted
Section"');
    header('HTTP/1.0 401 Unauthorized');
    die ("Please enter your username and password");
  }
?>
```

Incidentally, take a look at the wording of the error message: `Invalid username / password combination`. It doesn't say whether the username or the password or both were wrong—the less information you can give to a potential hacker, the better.

A mechanism is now in place to authenticate users, but only for a single username and password. Also, the password appears in clear text within the PHP file, and if someone managed to hack into your server, he would instantly know it. So let's look at a better way to handle usernames and passwords.

Storing Usernames and Passwords

Obviously MySQL is the natural way to store usernames and passwords. But again, we don't want to store the passwords as clear text, because our website could be compro-

mised if the database were accessed by a hacker. Instead, we'll use a neat trick called a *one-way function*.

This type of function is easy to use and converts a string of text into a seemingly random string. Due to their one-way nature, such functions are virtually impossible to reverse, so their output can be safely stored in a database—and anyone who steals it will be none the wiser as to the passwords used.

In previous editions of this book I recommended using the *md5* hashing algorithm for your data security. Time marches on, however, and now md5 is considered easily hackable and therefore unsafe, while even its previously recommended replacement of *sha1* can apparently be hacked (plus sha1 and sha2 were designed by the NSA and therefore considerable caution is recommended for their use in highly secure implementations). So now I have moved on to using the PHP `hash` function, passing it a version of the *ripemd* algorithm, which was designed by the open academic community and which (like md5) returns a 32-character hexadecimal number—so it can easily replace md5 in most databases. Use it like this:

```
$token = hash('ripemd128',
'mypassword');
```

That example happens to give `$token` the value:

```
7b694600c8a2a2b0897c719958713619
```

By using the `hash` function, you can keep up with future developments in security and simply pass the hashing algorithm to it that you wish to implement, resulting in less code maintenance (although you will probably have to accommodate larger hash lengths than 32 characters in your databases).

Salting

Unfortunately, `hash` on its own is not enough to protect a database of passwords, because it could still be susceptible to a brute force attack that uses another database of known 32-character hexadecimal tokens. Such databases do exist, as a quick Google search will verify, although probably only for md5 and sha1 or sha2 at the moment.

Thankfully, though, we can put a spanner in the works of any such attempts by *salting* all the passwords before they are sent to `hash`. Salting is simply a matter of adding some text that only we know about to each parameter to be encrypted, like this (with the salt highlighted in bold):

```
$token = hash('ripemd128', 'saltstringmypassword');
```

In this example, the text *saltstring* has been prepended to the password. Of course, the more obscure you can make the salt, the better. I like to use salts such as this:

```
$token = hash('ripemd128', 'hqb%$tmypasswordcg*l');
```

Here some random characters have been placed both before and after the password. Given just the database, and without access to your PHP code, it should now be next to impossible to work out the stored passwords.

All you have to do when verifying someone's login password is to add these same random strings back in before and after it, and then check the resulting token from a `hash` call against the one stored in the database for that user.

Let's create a MySQL table to hold some user details and add a couple of accounts. So type and save the program in Example 13-3 as *setupusers.php*, then open it in your browser.

Example 13-3. Creating a users table and adding two accounts

```
<?php // setupusers.php
require_once 'login.php';
  $connection =
    new mysqli($db_hostname, $db_username, $db_password,

$db_database);   if ($connection->connect_error)

die($connection->connect_error);

  $query = "CREATE TABLE users (
forename VARCHAR(32) NOT NULL,
surname  VARCHAR(32) NOT NULL,
username VARCHAR(32) NOT NULL
UNIQUE,     password VARCHAR(32)
NOT NULL
  )";
  $result = $connection-
>query($query);   if (!$result)
die($connection->error);

  $salt1    = "qm&h*";
  $salt2    = "pg!@";

  $forename = 'Bill';
  $surname  = 'Smith';
  $username = 'bsmith';
  $password = 'mysecret';
  $token    = hash('ripemd128', "$salt1$password$salt2");

add_user($connection, $forename, $surname, $username,

$token);

  $forename = 'Pauline';
  $surname  = 'Jones';
  $username = 'pjones';
```

```
  $password = 'acrobat';
  $token    = hash('ripemd128', "$salt1$password$salt2");

add_user($connection, $forename, $surname, $username,

$token);

  function add_user($connection, $fn, $sn, $un, $pw)
  {
    $query  = "INSERT INTO users VALUES('$fn', '$sn',
'$un', '$pw')";
    $result = $connection-
>query($query);     if (!$result)
die($connection->error);
  }
?>
```

This program will create the table *users* within your *publications* database (or whichever database you set up for the *login.php* file in Chapter 10). In this table, it will create two users: Bill Smith and Pauline Jones. They have the usernames and passwords of *bsmith/ mysecret* and *pjones/acrobat*, respectively.

Using the data in this table, we can now modify Example 13-2 to properly authenticate users, and Example 13-4 shows the code needed to do this. Type it, save it as *authenti cate.php*, and call it up in your browser.

Example 13-4. PHP authentication using MySQL

```
<?php // authenticate.php
require_once 'login.php';
  $connection =
    new mysqli($db_hostname, $db_username, $db_password,

$db_database);   if ($connection->connect_error)

die($connection->connect_error);

  if (isset($_SERVER['PHP_AUTH_USER']) &&
isset($_SERVER['PHP_AUTH_PW']))
  {
    $un_temp = mysql_entities_fix_string($connection,
$_SERVER['PHP_AUTH_USER']);     $pw_temp =
mysql_entities_fix_string($connection, $_SERVER['PHP_AUTH_PW']);

    $query  = "SELECT * FROM users WHERE username='$un_temp'";
    $result = $connection-
>query($query);     if (!$result)
die($connection->error);
    elseif ($result->num_rows)
    {
```

```
        $row = $result->fetch_array(MYSQLI_NUM);

          $result->close();

        $salt1 = "qm&h*";
        $salt2 = "pg!@";
        $token = hash('ripemd128', "$salt1$pw_temp$salt2");

        if ($token == $row[3]) echo "$row[0] $row[1] :
          Hi $row[0], you are now logged in as
'$row[2]'";         else die("Invalid
username/password combination");
    }
    else die("Invalid username/password combination");
  }
else
  {
    header('WWW-Authenticate: Basic realm="Restricted
Section"');
    header('HTTP/1.0 401 Unauthorized');
    die ("Please enter your username and
password");   }   $connection->close();

  function mysql_entities_fix_string($connection, $string)
  {
    return htmlentities(mysql_fix_string($connection,
$string));   }

  function mysql_fix_string($connection, $string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return $connection->real_escape_string($string);
  }
?>
```

As you might expect at this point in the book, some of the examples such as this one are starting to get quite a bit longer. But don't be put off. The final 10 lines are simply Example 10-22 from Chapter 10. They are there to sanitize the user input—very important.

The only lines to really concern yourself with at this point start with the assigning of two variables, `$un_temp` and `$pw_temp`, using the submitted username and password, highlighted in bold text. Next, a query is issued to MySQL to look up the user `$un_temp` and, if a result is returned, to assign the first row to `$row`. (Because usernames are unique, there will be only one row.) Then the two salts are created in `$salt1` and `$salt2`, which are then added before and after the submitted password `$pw_temp`. This string is then passed to the `hash` function, which returns a 32-

character hexadecimal value in `$token`. Now all that's necessary is to check `$token` against the value stored in the database, which happens to be in the fourth column—which is column 3 when starting from 0. So `$row[3]` contains the previous token calculated for the salted password. If the two match, a friendly welcome string is output, calling the user by his or her first name (see Figure 13-4). Otherwise, an error message is displayed. As mentioned before, the error message is the same regardless of whether such a username exists, as this provides minimal information to potential hackers or password guessers.

You can try this out for yourself by calling up the program in your browser and entering a username of `bsmith` and password of `mysecret` (or `pjones` and `acrobat`), the values that were saved in the database by Example 13-3.

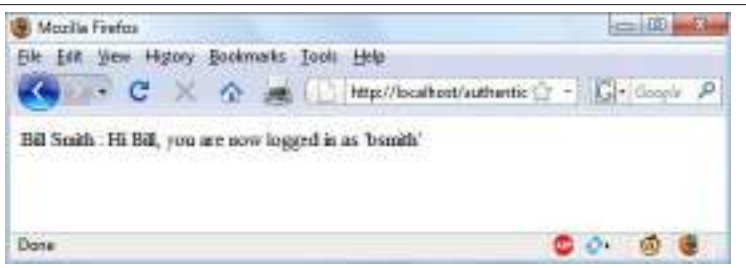


Figure 13-4. Bill Smith has now been authenticated

By sanitizing input immediately after it is encountered, you will block any malicious HTML, JavaScript, or MySQL attacks before they can get any further, and will not have to sanitize this data again. Remember, however, that if a user has characters such as < or & in a password (for example), these will be expanded to `<` or `&` by the `htmlemtities` function. But as long as your code allows for strings that may end up larger than the provided input width, and as long as you always run passwords through this sanitization, you'll be just fine.

Using Sessions

Because your program can't tell what variables were set in other programs—or even what values the same program set the previous time it ran—you'll sometimes want to track what your users are doing from one web page to another. You can do this by setting hidden fields in a form, as seen in Chapter 10, and checking the value of the fields after the form is submitted, but PHP provides a much more powerful and simpler solution in the form of *sessions*. These are groups of variables that are stored on the server but relate only to the current user. To ensure that the right variables are applied

to the right users, PHP saves a cookie in the users' web browsers to uniquely identify them.

This cookie has meaning only to the web server and cannot be used to ascertain any information about a user. You might ask about those users who have their cookies turned off. Well, that's not a problem as of PHP 4.2.0, because it will identify when this is the case and place a cookie token in the `GET` portion of each URL request instead. Either way, sessions provide a solid way of keeping track of your users.

Starting a Session

Starting a session requires calling the PHP function `session_start` before any HTML has been output, similarly to how cookies are sent during header exchanges. Then, to begin saving session variables, you just assign them as part of the `$_SESSION` array, like this:

```
$_SESSION['variable'] = $value;
```

They can then be read back just as easily in later program runs, like this:

```
$variable = $_SESSION['variable'];
```

Now assume that you have an application that always needs access to the username, password, first name, and last name of each user, as stored in the table *users*, which you should have created a little earlier. So let's further modify *authenticate.php* from Example 13-4 to set up a session once a user has been authenticated.

Example 13-5 shows the changes needed. The only difference is the content of the `if ($token == $row[3])` section, which we now start by opening a session and saving these four variables into it. Enter this program (or modify Example 13-4) and save it as *authenticate2.php*. But don't run it in your browser yet, as you will also need to create a second program in a moment.

Example 13-5. Setting a session after successful authentication

```
<?php //authenticate2.php
require_once 'login.php';
  $connection =
    new mysqli($db_hostname, $db_username, $db_password,

$db_database);   if ($connection->connect_error)

die($connection->connect_error);

  if (isset($_SERVER['PHP_AUTH_USER']) &&
isset($_SERVER['PHP_AUTH_PW']))
  {
    $un_temp = mysql_entities_fix_string($connection,
$_SERVER['PHP_AUTH_USER']);
    $pw_temp = mysql_entities_fix_string($connection,
$_SERVER['PHP_AUTH_PW']);

    $query = "SELECT * FROM users WHERE
username='$un_temp'";
    $result = $connection->query($query);

    if (!$result) die($connection->error);
    elseif ($result->num_rows)
    {
          $row = $result->fetch_array(MYSQLI_NUM);

          $result->close();

          $salt1 = "qm&h*";
          $salt2 = "pg!@";
        $token = hash('ripemd128', "$salt1$pw_temp$salt2");
```

```
          if ($token == $row[3])
          {
               session_start();
               $_SESSION['username'] = $un_temp;
               $_SESSION['password'] = $pw_temp;
               $_SESSION['forename'] = $row[0];
               $_SESSION['surname']  = $row[1];
echo    "$row[0]    $row[1]    :    Hi    $row[0],
you are now logged in as '$row[2]'";
               die ("<p><a href=continue.php>Click here to
continue</a></p>");
          }
          else die("Invalid username/password
combination");
    }
    else die("Invalid username/password combination");
  }
else
  {
    header('WWW-Authenticate: Basic realm="Restricted
Section"');
    header('HTTP/1.0 401 Unauthorized');
    die ("Please enter your username and
password");   }   $connection->close();

  function mysql_entities_fix_string($connection, $string)
  {
    return htmlentities(mysql_fix_string($connection,
$string));   }

  function mysql_fix_string($connection, $string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return $connection->real_escape_string($string);
  }
?>
```

One other addition to the program is the "Click here to continue" link with a destination URL of *continue.php*. This will be used to illustrate how the session will transfer to another program or PHP web page. So create *continue.php* by entering the program in Example 13-6 and saving it.

Example 13-6. Retrieving session variables

```
<?php // continue.php
session_start();

  if (isset($_SESSION['username']))
```

```
  {
    $username = $_SESSION['username'];
    $password = $_SESSION['password'];
    $forename = $_SESSION['forename'];
    $surname  = $_SESSION['surname'];

    echo "Welcome back $forename.<br>
          Your full name is $forename $surname.<br>
          Your username is
'$username'           and your
password is '$password'.";
  }
  else echo "Please <a href='authenticate2.php'>click here</a>
to log in."; ?>
```

Now you are ready to call up *authenticate2.php* into your browser. Enter a username of `bsmith` and password of `mysecret` (or `pjones` and `acrobat`) when prompted, and click the link to load in *continue.php*. When your browser calls it up, the result should be something like Figure 13-5.

Figure 13-5. Maintaining user data with sessions

Sessions neatly confine to a single program the extensive code required to authenticate and log in a user. Once a user has been authenticated and you have created a session, your program code becomes very simple indeed. You need only to call up `session_start` and look up any variables to which you need access from `$_SESSION`.

In Example 13-6, a quick test of whether `$_SESSION['username']` has a value is enough to let you know that the current user is authenticated, because session variables are stored on the server (unlike cookies, which are stored on the web browser) and can therefore be trusted.

If `$_SESSION['username']` has not been assigned a value, no session is active, so the last line of code in Example 13-6 directs users to the login page at *authenticate2.php*.

The *continue.php* program prints back the value of the user's password to show you how session variables work. In practice, you already know that the user is logged in, so you shouldn't need to keep track of (or display) any passwords, and in fact doing so would be a security risk.

Ending a Session

When the time comes to end a session, usually when a user requests to log out from your site, you can use the `session_destroy` function in association, as in Example 13-7. That example provides a useful function for totally destroying a session, logging a user out, and unsetting all session variables.

Example 13-7. A handy function to destroy a session and its data

```
<?php
  function destroy_session_and_data()
  {
    session_start();
    $_SESSION = array();
    setcookie(session_name(), '', time() - 2592000, '/');
    session_destroy();
  }
?>
```

To see this in action, you could modify *continue.php* as in Example 13-8.

Example 13-8. Retrieving session variables and then destroying the session

```
<?php
session_start();

  if (isset($_SESSION['username']))
  {
    $username = $_SESSION['username'];
    $password = $_SESSION['password'];
    $forename = $_SESSION['forename'];
    $surname  = $_SESSION['surname'];

destroy_session_and_data();

    echo "Welcome back $forename.<br>
          Your full name is $forename $surname.<br>
          Your username is
'$username'           and your
password is '$password'.";

  }
```

```
  else echo "Please <a href='authenticate2.php'>click here</a> to log
in.";

  function destroy_session_and_data()
  {
    $_SESSION = array();
    setcookie(session_name(), '', time() - 2592000, '/');
    session_destroy();
  }
?>
```

The first time you navigate from *authenticate2.php* to *continue.php*, it will display all the session variables. But, because of the call to `destroy_session_and_data`, if you then click on your browser's Reload button, the session will have been destroyed and you'll be prompted to return to the login page.

Setting a Timeout

There are other times when you might wish to close a user's session yourself, such as when the user has forgotten or neglected to log out, and you want the program to do so for his for her own security. You do this by setting the timeout after which a logout will automatically occur if there has been no activity.

To do this, use the `ini_set` function as follows. This example sets the timeout to exactly one day:

```
ini_set('session.gc_maxlifetime', 60 * 60 * 24);
```

If you wish to know what the current timeout period is, you can display it using the following:

```
echo ini_get('session.gc_maxlifetime');
```

Session Security

Although I mentioned that once you had authenticated a user and set up a session you could safely assume that the session variables were trustworthy, this isn't exactly the case. The reason is that it's possible to use *packet sniffing* (sampling of data) to discover session IDs passing across a network. Additionally, if the session ID is passed in the `GET` part of a URL, it might appear in external site server logs. The only truly secure way of preventing these from being discovered is to implement a *Secure Socket Layer (SSL)* and run HTTPS instead of HTTP web pages. That's beyond the scope of this book, although you may like to take a look at *http://apache-ssl.org* for details on setting up a secure web server.

Preventing session hijacking

When SSL is not a possibility, you can further authenticate users by storing their IP address along with their other details by adding a line such as the following when you store their session:

```
$_SESSION['ip'] = $_SERVER['REMOTE_ADDR'];
```

Then, as an extra check, whenever any page loads and a session is available, perform the following check. It calls the function `different_user` if the stored IP address doesn't match the current one:

```
if ($_SESSION['ip'] != $_SERVER['REMOTE_ADDR'])
different_user();
```

What code you place in your `different_user` function is up to you. I recommend that you simply delete the current session and ask the user to log in again due to a technical error. Don't say any more than that, or you're giving away potentially useful information.

Of course, you need to be aware that users on the same proxy server, or sharing the same IP address on a home or business network, will have the same IP address. Again, if this is a problem for you, use SSL. You can also store a copy of the browser *user agent string* (a string that developers put in their browsers to identify them by type and version), which might also distinguish users due to the wide variety of browser types, versions, and computer platforms. Use the following to store the user agent:

```
$_SESSION['ua'] = $_SERVER['HTTP_USER_AGENT'];
```

And use this to compare the current agent string with the saved one:

```
if ($_SESSION['ua'] != $_SERVER['HTTP_USER_AGENT'])
different_user();
```

Or, better still, combine the two checks like this and save the combination as a `hash` hexadecimal string:

```
$_SESSION['check'] = hash('ripemd128',
$_SERVER['REMOTE_ADDR'] .
  $_SERVER['HTTP_USER_AGENT']);
```

And use this to compare the current and stored strings:

```
if ($_SESSION['check'] != hash('ripemd128', $_SERVER['REMOTE_ADDR']
.   $_SERVER['HTTP_USER_AGENT'])) different_user();
```

Preventing session fixation

Session fixation happens when a malicious user tries to present a session ID to the server rather than letting the server create one. It can happen when a user takes advantage of the ability to pass a session ID in the `GET` part of a URL, like this:

```
http://yourserver.com/authenticate.php?PHPSESSID=123456
789
```

In this example, the made-up session ID of 123456789 is being passed to the server. Now, consider Example 13-9, which is susceptible to session fixation. To see how, save it as *sessiontest.php*.

Example 13-9. A session susceptible to session fixation

```
<?php // sessiontest.php
session_start();

  if (!isset($_SESSION['count']))
$_SESSION['count'] = 0;   else
++$_SESSION['count'];

  echo
$_SESSION['count']; ?>
```

Once it's saved, call it up in your browser using the following URL (prefacing it with the correct pathname, such as *http://localhost/*):

```
sessiontest.php?PHPSESSID=1234
```

Press Reload a few times, and you'll see the counter increase. Now try browsing to:

```
sessiontest.php?PHPSESSID=5678
```

Press Reload a few times here, and you should see it count up again from 0. Leave the counter on a different number than the first URL and then go back to the first URL and see how the number changes back. You have created two different sessions of your own choosing here, and you could easily create as many as you needed.

The reason this approach is so dangerous is that a malicious attacker could try to distribute these types of URLs to unsuspecting users, and if any of them followed these links, the attacker would be able to come back and take over any sessions that had not been deleted or expired!

In order to prevent this, add a simple check to change the session ID using `session_re generate_id`. This function keeps all current session variable values, but replaces the session ID with a new one that an attacker cannot know.

To do this, check for a special session variable that you arbitrarily invent. If it doesn't exist, you know that this is a new session, so you simply change the session ID and set the special session variable to note the change.

Example 13-10 shows how the code to do this might look, using the session variable `initiated`.

Example 13-10. Session regeneration

```
<?php
session_start();

  if (!isset($_SESSION['initiated']))
  {
    session_regenerate_id();
    $_SESSION['initiated'] = 1;
  }

  if (!isset($_SESSION['count']))
$_SESSION['count'] = 0;   else
++$_SESSION['count'];

  echo
$_SESSION['count']; ?>
```

This way, an attacker can come back to your site using any of the session IDs that he or she generated, but none of them will call up another user's session, as they will all have been replaced with regenerated IDs. If you want to be ultra-paranoid, you can even regenerate the session ID on each request.

Forcing cookie-only sessions

If you are prepared to require your users to enable cookies on your website, you can use the `ini_set` function like this:

```
ini_set('session.use_only_cookies', 1);
```

With that setting, the `?PHPSESSID=` trick will be completely ignored. If you use this security measure, I also recommend you inform your users that your site requires cookies, so they know what's wrong if they don't get the results they want.

Using a shared server

On a server shared with other accounts, you will not want to have all your session data saved into the same directory as theirs. Instead, you should choose a directory to which only your account has access (and that is not web-visible) to store your sessions, by placing an `ini_set` call near the start of a program, like this:

```
ini_set('session.save_path',
'/home/user/myaccount/sessions');
```

The configuration option will keep this new value only during the program's execution, and the original configuration will be restored at the program's ending.

This sessions folder can fill up quickly; you may wish to periodically clear out older sessions according to how busy your server gets. The more it's used, the less time you will want to keep a session stored.

Remember that your websites can and will be subject to hacking attempts. There are automated bots running riot around the Internet trying to find sites vulnerable to exploits. So whatever you do, whenever you are handling data that is not 100% generated within your own program, you should always treat it with the utmost caution.

At this point, you should now have a very good grasp of both PHP and MySQL, so in the next chapter it's time to introduce the third major technology covered by this book, JavaScript.

Questions

1. Why must a cookie be transferred at the start of a program?
2. Which PHP function stores a cookie on a web browser?
3. How can you destroy a cookie?
4. Where are the username and password stored in a PHP program when you are using HTTP authentication?
5. Why is the `hash` function a powerful security measure?
6. What is meant by "salting" a string?
7. What is a PHP session?
8. How do you initiate a PHP session?

9. What is session hijacking?
10. What is session fixation?

See "Chapter 13 Answers" on page 648 in Appendix A for the answers to these questions.

CHAPTER 14

Exploring JavaScript

JavaScript brings a dynamic functionality to your websites. Every time you see something pop up when you mouse over an item in the browser, or see new text, colors, or images appear on the page in front of your eyes, or grab an object on the page and drag it to a new location—all those things are done through JavaScript. It offers effects that are not otherwise possible, because it runs inside the browser and has direct access to all the elements in a web document.

JavaScript first appeared in the Netscape Navigator browser in 1995, coinciding with the addition of support for Java technology in the browser. Because of the initial incorrect impression that JavaScript was a spin-off of Java, there has been some long-term confusion over their relationship. However, the naming was just a marketing ploy to help the new scripting language benefit from the popularity of the Java programming language.

JavaScript gained new power when the HTML elements of the web page got a more formal, structured definition in what is called the *Document Object Model*, or *DOM*. The DOM makes it relatively easy to add a new paragraph or focus on a piece of text and change it.

Because both JavaScript and PHP support much of the structured programming syntax used by the C programming language, they look very similar to each other. They are both fairly high-level languages, too; for instance, they are weakly typed, so it's easy to change a variable to a new type just by using it in a new context.

Now that you have learned PHP, you should find JavaScript even easier. And you'll be glad you did, because it's at the heart of the Web 2.0 Ajax technology that provides the fluid web frontends that (along with HTML5 features) savvy web users expect these days.

JavaScript and HTML Text

JavaScript is a client-side scripting language that runs entirely inside the web browser. To call it up, you place it between opening `<script>` and closing `</script>`

HTML tags. A typical HTML 4.01 “Hello World” document using JavaScript might look like Example 14-1.

Example 14-1. “Hello World” displayed using JavaScript

```
<html>
  <head><title>Hello World</title></head>
  <body>
    <script type="text/javascript">
document.write("Hello World")
    </script>
    <noscript>
      Your browser doesn't support or has disabled
JavaScript
    </noscript>
  </body>
</html>
```

You may have seen web pages that use the HTML tag `<script language="javascript">`, but that usage has now been deprecated. This example uses the more recent and preferred `<script type="text/ javascript">`, or you can just use `<script>` on its own if you like.

Within the `<script>` tags is a single line of JavaScript code that uses its equivalent of the PHP `echo` or `print` commands, `document.write`. As you’d expect, it simply outputs the supplied string to the current document, where it is displayed.

You may also have noticed that, unlike with PHP, there is no trailing semicolon (`;`). This is because a newline serves the same purpose as a semicolon in JavaScript. However, if you wish to have more than one statement on a single line, you do need to place a semicolon after each command except the last one. Of course, if you wish, you can add a semicolon to the end of every statement and your JavaScript will work fine. The other thing to note in this example is the `<noscript>` and `</noscript>` pair of tags. These are used when you wish to offer alternative HTML to users whose browser does not support JavaScript or who have it disabled. Using these tags is up to you, as they are not required, but you really ought to use them because it’s usually not that difficult to provide static HTML alternatives to the operations you provide using JavaScript. However, the remaining examples in this book will omit `<noscript>` tags, because we’re focusing on what you can do with JavaScript, not what you can do without it.

When Example 14-1 is loaded, a web browser with JavaScript enabled will output the following (see Figure 14-1):

```
Hello World
```

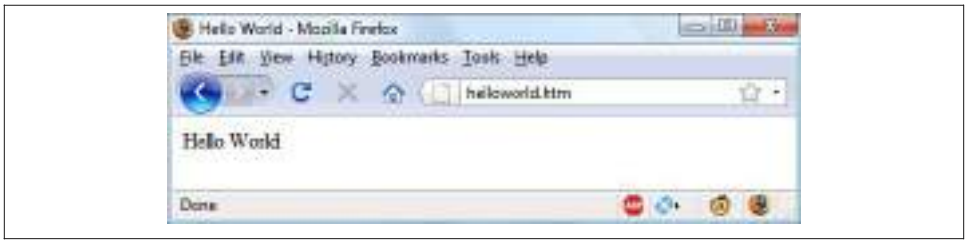


Figure 14-1. JavaScript, enabled and working

A browser with JavaScript disabled will display the message in Figure 14-2.

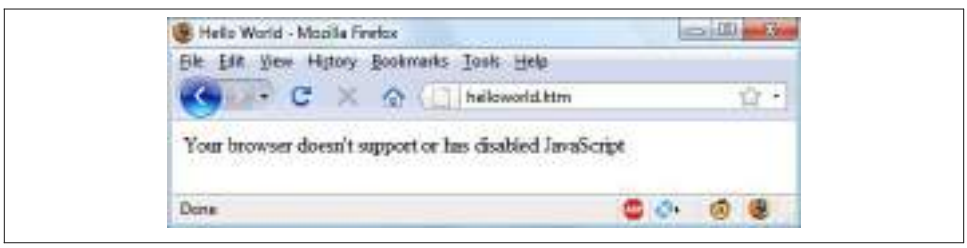


Figure 14-2. JavaScript has been disabled

Using Scripts Within a Document Head

In addition to placing a script within the body of a document, you can put it in the `<head>` section, which is the ideal place if you wish to execute a script when a page loads. If you place critical code and functions there, you can also ensure that they are ready to use immediately by any other script sections in the document that rely on them. Another reason for placing a script in the document head is to enable JavaScript to write things such as meta tags into the `<head>` section, because the location of your script is the part of the document it writes to by default.

Older and Nonstandard Browsers

If you need to support browsers that do not offer scripting, you will need to use the HTML comment tags (`<!--` and `-->`) to prevent them from encountering script code that they should not see. Example 14-2 shows how you add them to your script code.

Example 14-2. The "Hello World" example modified for non-JavaScript browsers

```
<html>
```

```
  <head><title>Hello World</title></head>
  <body>
    <script type="text/javascript"><!-
document.write("Hello World")
    // --></script>
  </body>
</html>
```

Here an opening HTML comment tag (`<!--`) has been added directly after the opening `<script>` statement and a closing comment tag (`// -->`) directly before the script is closed with `</script>`.

The double forward slash (`//`) is used by JavaScript to indicate that the rest of the line is a comment. It is there so that browsers that *do* support JavaScript will ignore the following `-->`, but non-JavaScript browsers will ignore the preceding `//`, and act on the `-->` by closing the HTML comment.

Although the solution is a little convoluted, all you really need to remember is to use the two following lines to enclose your JavaScript when you wish to support very old or non-standard browsers:

```
<script type="text/javascript"><!-
  (Your JavaScript goes here...)
// --></script>
```

However, the use of these comments is unnecessary for any browser released over the past several years.

There are a couple of other scripting languages you should know about. These include Microsoft's VBScript, which is based on the Visual Basic programming language, and Tcl, a rapid prototyping language. They are called up in a similar way to JavaScript, except they use types of `text/vbscript` and `text/tcl`, respectively. VBScript works only in Internet Explorer; use of it in other browsers requires a plugin. Tcl always needs a plug-in. So both should be considered nonstandard, and neither is covered in this book.

Including JavaScript Files

In addition to writing JavaScript code directly in HTML documents, you can include files of JavaScript code either from your website or from anywhere on the Internet. The syntax for this is:

```
<script type="text/javascript"
src="script.js"></script>
```

Or, to pull a file in from the Internet, use:

```
<script type="text/javascript"
src="http://someserver.com/script.js">
```

```
</script>
```

As for the script files themselves, they must *not* include any `<script>` or `</script>` tags, because they are unnecessary: the browser already knows that a JavaScript file is being loaded. Putting them in the JavaScript files will cause an error.

Including script files is the preferred way for you to use third-party JavaScript files on your website.

It is possible to leave out the `type="text/javascript"` parameters; all modern browsers default to assuming that the script contains JavaScript.

Debugging JavaScript Errors

When you're learning JavaScript, it's important to be able to track typing or other coding errors. Unlike PHP, which displays error messages in the browser, JavaScript handles error messages in a way that changes according to the browser used. Table 14-1 lists how to access JavaScript error messages in each of the five most commonly used browsers.

Table 14-1. Accessing JavaScript error messages in different browsers

Browser	How to access JavaScript error messages
Apple Safari	Safari does not have an Error Console enabled by default, but you can turn it on by selecting Safari→Preferences→Advanced→"Show Develop menu in menu bar." However, you may prefer to use the Firebug Lite JavaScript module, which many people find easier to use.
Google Chrome	Click the menu icon that looks like a page with a corner turned, then select Developer→JavaScript Console. You can also use the shortcut Ctrl-Shift-J on a PC, or Command-Shift-J on a Mac.
Microsoft Internet Explorer	Select Tools→Internet Options→Advanced, then uncheck the Disable Script Debugging box and check the "Display a Notification about Every Script Error" box.
Mozilla Firefox	Select Tools→Error Console or use the shortcut Ctrl-Shift-J on a PC, or Command-Shift-J on a Mac.
Opera	Select Tools→Advanced→Error Console.

OS X users: although I have shown you how to create an Error Console for JavaScript, you may prefer to use Google Chrome (for Intel OS X 10.5 or higher).

To try out whichever Error Console you are using, let's create a script with a minor error. Example 14-3 is much the same as Example 14-1, but the final double quotation mark has been left off the end of the string `"Hello World"`—a common syntax error.

Example 14-3. A JavaScript "Hello World" script with an error

```
<html>
  <head><title>Hello World</title></head>
  <body>
    <script type="text/javascript">
document.write("Hello World)
    </script>
  </body>
</html>
```

Enter the example and save it as *test.html*, then call it up in your browser. It should succeed only in displaying the title, not anything in the main browser window. Now call up the Error Console in your browser, and you should see a message such as the one in Example 14-4. To the right there will be a link to the source, which, when clicked, shows the error line highlighted (but does not indicate the position at which the error was encountered).

Example 14-4. A Mozilla Firefox Error Console message

```
SyntaxError: unterminated string literal
```

In Microsoft Internet Explorer, the error message will look like Example 14-5, and there's no helpful arrow, but you are given the line and position.

Example 14-5. A Microsoft Internet Explorer Error Console message

```
Unterminated string constant
```

Google Chrome and Opera will give the message in Example 14-6. Again, you'll be given the line error number but not the exact location.

Example 14-6. A Google Chrome/Opera Error Console message

```
Uncaught SyntaxError: Unexpected token ILLEGAL
```

And Apple Safari provides the message in Example 14-7, with a link to the source on the right stating the line number of the error. You can click the link to highlight the line, but it will not show where on the line the error occurred.

Example 14-7. An Opera Error Console message

```
SyntaxError: Unexpected EOF
```

If you find this support a little underwhelming, the Firebug plug-in for Firefox (and now Chrome too) is very popular among JavaScript developers for debugging code, and is definitely worth a look.

If you will be entering the following code snippets to try them out, don't forget to surround them with `<script>` and `</script>` tags.

Using Comments

Due to their shared inheritance from the C programming language, PHP and JavaScript have many similarities, one of which is commenting. First, there's the single-line comment, like this:

```
// This is a comment
```

This style uses a pair of forward slash characters (`//`) to inform JavaScript that everything following is to be ignored. And then you also have multiline comments, like this:

```
/* This is a
section    of
multiline comments
that will not be
interpreted */
```

Here you start a multiline comment with the sequence `/*` and end it with `*/`. Just remember that you cannot nest multiline comments, so make sure that you don't comment out large sections of code that already contain multiline comments.

Semicolons

Unlike PHP, JavaScript generally does not require semicolons if you have only one statement on a line. Therefore, the following is valid:

```
x += 10
```

However, when you wish to place more than one statement on a line, you must separate them with semicolons, like this:

```
x += 10; y -= 5; z = 0
```

You can normally leave the final semicolon off, because the newline terminates the final statement.

There are exceptions to the semicolon rule. If the first character of a line is either a left parenthesis or a left bracket, then JavaScript will assume it follows on from the previous line. If this is not what you intend, then insert a semicolon between the two lines to separate them. Semicolons are also required by JavaScript bookmarklets because all the code must be on a single line. So, when in doubt, use a semicolon.

Variables

No particular character identifies a variable in JavaScript as the dollar sign does in PHP. Instead, variables use the following naming rules:

- A variable may include only the letters `a–z`, `A–Z`, `0–9`, the `$` symbol, and the underscore (_).
- No other characters, such as spaces or punctuation, are allowed in a variable name.
- The first character of a variable name can be only `a–z`, `A–Z`, `$`, or _ (no numbers).
- Names are case-sensitive. `Count`, `count`, and `COUNT` are all different variables.
- There is no set limit on variable name lengths.

And yes, you're right, that is a `$` there in that list. It *is* allowed by JavaScript and *may* be the first character of a variable or function name. Although I don't recommend keeping the `$` symbols, it means that you can port a lot of PHP code more quickly to JavaScript that way.

String Variables

JavaScript string variables should be enclosed in either single or double quotation marks, like this:

```
greeting = "Hello there"
warning  = 'Be careful'
```

You may include a single quote within a double-quoted string or a double quote within a single-quoted string. But you must escape a quote of the same type using the backslash character, like this:

```
greeting = "\"Hello there\" is a
greeting" warning  = '\'Be careful\' is
a warning'
```

To read from a string variable, you can assign it to another one, like this:

```
newstring = oldstring
```

or you can use it in a function, like this:

```
   status  =  "All  systems  are
working"  document.write(status)
```

Numeric Variables

Creating a numeric variable is a simple as assigning a value, like these examples:

```
count       = 42
temperature = 98.4
```

Like strings, numeric variables can be read from and used in expressions and functions.

Arrays

JavaScript arrays are also very similar to those in PHP, in that an array can contain string or numeric data, as well as other arrays. To assign values to an array, use the following syntax (which in this case creates an array of strings):

```
toys = ['bat', 'ball', 'whistle', 'puzzle', 'doll']
```

To create a multidimensional array, nest smaller arrays within a larger one. So, to create a two-dimensional array containing the colors of a single face of a scrambled Rubik's Cube (where the colors red, green, orange, yellow, blue, and white are represented by their capitalized initial letters), you could use the following code:

```
face =
[
  ['R', 'G', 'Y'],
  ['W', 'R', 'O'],
  ['Y', 'W', 'G']
]
```

The previous example has been formatted to make it obvious what is going on, but it could also be written like this:

```
face = [['R', 'G', 'Y'], ['W', 'R', 'O'], ['Y', 'W',
'G']]
```

or even like this:

```
top = ['R', 'G',
'Y'] mid = ['W',
'R', 'O'] bot =
['Y', 'W', 'G']

face = [top, mid, bot]
```

To access the element two down and three along in this matrix, you would use the following (because array elements start at position 0):

```
document.write(face[1][2])
```

This statement will output the letter `O` for orange.

JavaScript arrays are powerful storage structures, so Chapter 16 discusses them in much greater depth.

Operators

Operators in JavaScript, as in PHP, can involve mathematics, changes to strings, and comparison and logical operations (`and`, `or`, etc.). JavaScript mathematical operators look a lot like plain arithmetic; for instance, the following statement outputs `15`:

```
document.write(13 + 2)
```

The following sections teach you about the various operators.

Arithmetic Operators

Arithmetic operators are used to perform mathematics. You can use them for the main four operations (addition, subtraction, multiplication, and division) as well as to find the modulus (the remainder after a division) and to increment or decrement a value (see Table 14-2).

Table 14-2. Arithmetic operators

Operator	Description	Example
+	Addition	`j + 12`
-	Subtraction	`j - 22`
*	Multiplication	`j * 7`
/	Division	`j / 3.13`
%	Modulus (division remainder)	`j % 6`
++	Increment	`++j`
--	Decrement	`--j`

Assignment Operators

The assignment operators are used to assign values to variables. They start with the very simple =, and move on to +=, −=, and so on. The operator += adds the value on the right side to the variable on the left, instead of totally replacing the value on the left. Thus, if `count` starts with the value 6, the statement:

```
count += 1
```

sets `count` to 7, just like the more familiar assignment statement:

```
count = count + 1
```

Table 14-3 lists the various assignment operators available.

Table 14-3. Assignment operators

Operator	Example	Equivalent to
=	`j = 99`	`j = 99`
+=	`j += 2`	`j = j + 2`
+=	`j += 'string'`	`j = j + 'string'`
−=	`j -= 12`	`j = j - 12`
*=	`j *= 2`	`j = j * 2`
/=	`j /= 6`	`j = j / 6`
%=	`j %= 7`	`j = j % 7`

Comparison Operators

Comparison operators are generally used inside a construct such as an `if` statement where you need to compare two items. For example, you may wish to know whether a variable you have been incrementing has reached a specific value, or whether another variable is less than a set value, and so on (see Table 14-4). *Table 14-4. Comparison operators*

Operator	Description	Example
==	Is **equal** to	`j == 42`
!=	Is **not equal** to	`j != 17`
>	Is **greater than**	`j > 0`
<	Is **less than**	`j < 100`
>=	Is **greater than or equal** to	`j >= 23`
<=	Is **less than or equal** to	`j <= 13`
===	Is **equal** to (and of the same type)	`j ===56`
!==	Is **not equal** to (and of the same type)	`j !== '1'`

Logical Operators

Unlike PHP, JavaScript's logical operators do not include `and` and `or` equivalents to `&&` and `||`, and there is no `xor` operator (see Table 14-5).

Table 14-5. Logical operators

Operator	Description	Example
`&&`	And	`j == 1 && k == 2`
`||`	Or	`j < 100 || j > 0`
`!`	Not	`!(j == k)`

Variable Incrementing and Decrementing

The following forms of post- and pre-incrementing and decrementing you learned to use in PHP are also supported by JavaScript:

```
++x
--y x
+= 22
y -=
3
```

String Concatenation

JavaScript handles string concatenation slightly differently from PHP. Instead of the . (period) operator, it uses the plus sign (+), like this:

```
document.write("You have " + messages + " messages.")
```

Assuming that the variable `messages` is set to the value 3, the output from this line of code will be:

```
You have 3 messages.
```

Just as you can add a value to a numeric variable with the += operator, you can also append one string to another the same way:

```
name =  "James"
name += " Dean"
```

Escaping Characters

Escape characters, which you've seen used to insert quotation marks in strings, can also insert various special characters such as tabs, newlines, and carriage returns. Here is an example using tabs to lay out a heading; it is included here merely to illustrate escapes, because in web pages, there are better ways to do layout:

```
heading = "Name\tAge\tLocation"
```

Table 14-6 details the escape characters available.

Table 14-6. JavaScript's escape characters

Character	Meaning
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Tab
\'	Single quote (or apostrophe)
\"	Double quote

Character	Meaning
\\	Backslash
XXX	An octal number between 000 and 377 that represents the Latin-1 character equivalent (such as \251 for the © symbol)
\x*XX*	A hexadecimal number between 00 and FF that represents the Latin-1 character equivalent (such as \xA9 for the © symbol)
\u*XXXX*	A hexadecimal number between 0000 and FFFF that represents the Unicode character equivalent (such as \u00A9 for the © symbol)

Variable Typing

Like PHP, JavaScript is a very loosely typed language; the *type* of a variable is determined only when a value is assigned and can change as the variable appears in different contexts. Usually, you don't have to worry about the type; JavaScript figures out what you want and just does it.

Take a look at Example 14-8, in which:

1. The variable n is assigned the string value `838102050`, the next line prints out its value, and the `typeof` operator is used to look up the type.
2. n is given the value returned when the numbers 12345 and 67890 are multiplied together. This value is also `838102050`, but it is a number, not a string. The type of variable is then looked up and displayed.
3. Some text is appended to the number n and the result is displayed.

Example 14-8. Setting a variable's type by assignment

```
<script>
  n = '838102050'        // Set 'n' to a string
  document.write('n = ' + n + ', and is a ' + typeof n +
'<br>')

  n = 12345 * 67890;     // Set 'n' to a number
  document.write('n = ' + n + ', and is a ' + typeof n +
'<br>')

  n += ' plus some text' // Change 'n' from a number to a
string   document.write('n = ' + n + ', and is a ' + typeof
n + '<br>') </script>
```

The output from this script looks like:

```
n = 838102050, and is a string n =
838102050, and is a number n =
```

```
838102050 plus some text, and is a
string
```

Variable Typing

If there is ever any doubt about the type of a variable, or you need to ensure that a variable has a particular type, you can force it to that type using statements such as the following (which respectively turn a string into a number and a number into a string):

```
n = "123"
n *= 1    // Convert 'n' into a number

n = 123
n += ""   // Convert 'n' into a string
```

Or, of course, you can always look up a variable's type using the `typeof` operator.

Functions

As with PHP, JavaScript functions are used to separate out sections of code that perform a particular task. To create a function, declare it in the manner shown in Example 14-9. *Example 14-9. A simple function declaration*

```
<script>
  function product(a, b)
  {
    return a * b
  }
</script>
```

This function takes the two parameters passed, multiplies them together, and returns the product.

Global Variables

Global variables are ones defined outside of any functions (or within functions, but defined without the `var` keyword). They can be defined in the following ways:

```
    a = 123              // Global
scope var b = 456              //
Global scope if (a == 123) var c =
789 // Global scope
```

Regardless of whether you are using the `var` keyword, as long as a variable is defined outside of a function, it is global in scope. This means that every part of a script can have access to it.

Local Variables

Parameters passed to a function automatically have local scope; that is, they can be referenced only from within that function. However, there is one exception. Arrays are passed to a function by reference, so if you modify any elements in an array parameter, the elements of the original array will be modified.

To define a local variable that has scope only within the current function, and has not been passed as a parameter, use the `var` keyword. Example 14-10 shows a function that creates one variable with global scope and two with local scope.

Example 14-10. A function creating variables with global and local scope

```
<script>
function
test()
  {
        a = 123                //
Global scope     var b = 456
// Local scope     if (a == 123) var c
= 789 // Local scope
  }
</script>
```

To test whether scope setting has worked in PHP, we can use the `isset` function. But in JavaScript there isn't one, so Example 14-11 makes use of the `typeof` operator, which returns the string `undefined` when a variable is not defined.

Example 14-11. Checking the scope of the variables defined in function test

```
<script>
test()

  if (typeof a != 'undefined') document.write('a = "' + a
+ '"<br>')   if (typeof b != 'undefined')
document.write('b = "' + b + '"<br>')   if (typeof c !=
'undefined') document.write('c = "' + c + '"<br>')

  function test()
  {     a
= 123
var b = 456

    if (a == 123) var c = 789
  }
</script>
```

The output from this script is the following single line:

```
a = "123"
```

This shows that only the variable `a` was given global scope, which is exactly what we would expect, because the variables `b` and `c` were given local scope by being prefaced with the `var` keyword.

If your browser issues a warning about `b` being undefined, the warning is correct but can be ignored.

The Document Object Model

The designers of JavaScript were very smart. Rather than just creating yet another scripting language (which would have still been a pretty good improvement at the time), they had the vision to build it around the *Document Object Model*, or *DOM*. This breaks down the parts of an HTML document into discrete *objects*, each with its own *properties* and *methods* and each subject to JavaScript's control.

JavaScript separates objects, properties, and methods using a period (one good reason why + is the string concatenation operator in JavaScript, rather than the period). For example, let's consider a business card as an object we'll call `card`. This object contains properties such as a name, address, phone number, and so on. In the syntax of JavaScript, these properties would look like this:

```
card.name
card.phone
card.address
```

Its methods are functions that retrieve, change, and otherwise act on the properties. For instance, to invoke a method that displays the properties of object `card`, you might use syntax such as:

```
card.display()
```

Have a look at some of the earlier examples in this chapter and look at where the statement `document.write` is used. Now that you understand how JavaScript is based around objects, you will see that `write` is actually a method of the `document` object.

Within JavaScript, there is a hierarchy of parent and child objects, which is what is known as the Document Object Model (see Figure 14-3).

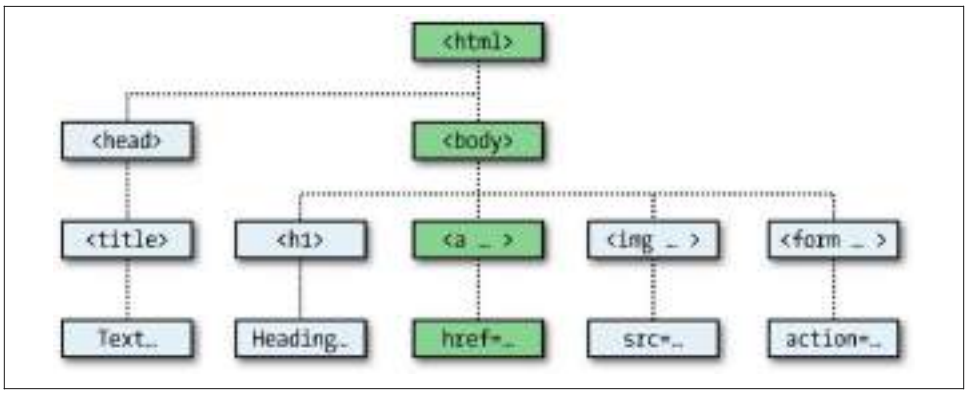


Figure 14-3. Example of DOM object hierarchy

The figure uses HTML tags that you are already familiar with to illustrate the parent/child relationship between the various objects in a document. For example, a URL within a link is part of the body of an HTML document. In JavaScript, it is referenced like this:

```
url = document.links.linkname.href
```

Notice how this follows the central column down. The first part, `document`, refers to the `<html>` and `<body>` tags; `links.linkname` to the `<a>` tag; and `href` to the `href` attribute.

Let's turn this into some HTML and a script to read a link's properties. Save Example 14-12 as *linktest.html*, then call it up in your browser.

If you are using Microsoft Internet Explorer as your main development browser, skim through this section first (without trying the example), then read the section entitled "But It's Not That Simple" on page 340, and finally come back here and try the example with the `getElementById` modification discussed there. Without it, this example will not work for you.

Example 14-12. Reading a link URL with JavaScript

```
<html>
  <head>
    <title>Link Test</title>
  </head>
  <body>
    <a id="mylink" href="http://mysite.com">Click
me</a><br>
    <script>
```

```
      url = document.links.mylink.href
document.write('The URL is ' + url)
    </script>
  </body>
</html>
```

Note the short form of the `<script>` tags where I have omitted the parameter `type="text/JavaScript"` to save you some typing. If you wish, just for the purposes of testing this (and other examples), you could also omit everything outside of the `<script>` and `</script>` tags. The output from this example is:

```
Click me
The URL is http://mysite.com
```

The second line of output comes from the `document.write` method. Notice how the code follows the document tree down from `document` to `links` to `mylink` (the `id` given to the link) to `href` (the URL destination value).

There is also a short form that works equally well, which starts with the value in the `id` attribute: `mylink.href`. So you can replace this:


```
url = document.links.mylink.href
```

with the following:

```
url = mylink.href
```

But It's Not That Simple

If you tried Example 14-12 in Safari, Firefox, Opera, or Chrome, it will have worked just great. But in Internet Explorer it will fail, because Microsoft's implementation of JavaScript, called JScript, has many subtle differences from the recognized standards. Welcome to the world of advanced web development!

So what can we do about this? Well, in this case, instead of using the `links` child object of the parent `document` object, which Internet Explorer balks at, you have to replace it with a method to fetch the element by its `id`. Therefore, the following line:

```
url = document.links.mylink.href
```

can be replaced with this one:

```
url = document.getElementById('mylink').href
```

And now the script will work in all major browsers. Incidentally, when you don't have to look up the element by `id`, the short form that follows will still work in Internet Explorer, as well as the other browsers:

```
url = mylink.href
```

Another Use for the $ Symbol

As mentioned earlier, the `$` symbol is allowed in JavaScript variable and function names. Because of this, you may sometimes encounter strange-looking code like this:

```
url = $('mylink').href
```

Some enterprising programmers have decided that the `getElementById` function is so prevalent in JavaScript that they have written a function to replace it called `$`, shown in Example 14-13.

Example 14-13. A replacement function for the getElementById method

```
<script>
function
$(id)
  {
    return document.getElementById(id)
  }
</script>
```

Therefore, as long as you have included the `$` function in your code, syntax such as:

```
$('mylink').href
```

can replace code such as:

```
document.getElementById('mylink').href
```

Using the DOM

The `links` object is actually an array of URLs, so the `mylink` URL in Example 14-12 can also be safely referred to on all browsers in the following way (because it's the first, and only, link):

```
url = document.links[0].href
```

If you want to know how many links there are in an entire document, you can query the `length` property of the `links` object like this:

```
numlinks = document.links.length
```

You can therefore extract and display all links in a document like this:

```
for (j=0 ; j < document.links.length ; ++j)
document.write(document.links[j].href + '<br>')
```

The `length` of something is a property of every array, and many objects as well. For example, the number of items in your browser's web history can be queried like this:

```
document.write(history.length)
```

However, to stop websites from snooping on your browsing history, the `history` object stores only the number of sites in the array: you cannot read from or write to these values. But you can replace the current page with one from the history, if you know what position it has within the history. This can be very useful in cases in which you know that certain pages in the history came from your site, or you simply wish to send the browser back one or more pages, which you do with the `go` method of the `history` object. For example, to send the browser back three pages, issue the following command:

```
history.go(-3)
```

You can also use the following methods to move back or forward a page at a time:

```
history.back()
history.forward()
```

In a similar manner, you can replace the currently loaded URL with one of your choosing, like this:

```
document.location.href = 'http://google.com'
```

Of course, there's a whole lot more to the DOM than reading and modifying links. As you progress through the following chapters on JavaScript, you'll become quite familiar with the DOM and how to access it.

CHAPTER 8

Introduction to MySQL

With well over 10 million installations, MySQL is probably the most popular database management system for web servers. Developed in the mid-1990s, it's now a mature technology that powers many of today's most-visited Internet destinations.

One reason for its success must be the fact that, like PHP, it's free to use. But it's also extremely powerful and exceptionally fast—it can run on even the most basic of hardware, and it hardly puts a dent in system resources.

MySQL is also highly scalable, which means that it can grow with your website (for the latest benchmarks, see *http://mysql.com/why-mysql/benchmarks*).

MySQL Basics

A *database* is a structured collection of records or data stored in a computer system and organized in such a way that it can be quickly searched and information can be rapidly retrieved.

The *SQL* in MySQL stands for *Structured Query Language*. This language is loosely based on English and also used in other databases such as Oracle and Microsoft SQL Server. It is designed to allow simple requests from a database via commands such as:

```
SELECT title FROM publications WHERE author = 'Charles
Dickens';
```

A MySQL database contains one or more *tables*, each of which contains *records* or *rows*. Within these rows are various *columns* or *fields* that contain the data itself. Table

8-1 shows the contents of an example database of five publications detailing the author, title, type, and year of publication.

Table 8-1. Example of a simple database

Author	Title	Type	Year
Mark Twain	The Adventures of Tom Sawyer	Fiction	1876
Jane Austen	Pride and Prejudice	Fiction	1811
Charles Darwin	The Origin of Species	Non-Fiction	1856
Charles Dickens	The Old Curiosity Shop	Fiction	1841
William Shakespeare	Romeo and Juliet	Play	1594

Each row in the table is the same as a row in a MySQL table, and each element within a row is the same as a MySQL field.

To uniquely identify this database, I'll refer to it as the *publications* database in the examples that follow. And, as you will have observed, all these publications are considered to be classics of literature, so I'll call the table within the database that holds the details *classics*.

Summary of Database Terms

The main terms you need to acquaint yourself with for now are:

Database
: The overall container for a collection of MySQL data

Table
: A subcontainer within a database that stores the actual data

Row
: A single record within a table, which may contain several fields

Column
: The name of a field within a row

I should note that I'm not trying to reproduce the precise terminology used in academic literature about relational databases, but just to provide simple, everyday terms to help you quickly grasp basic concepts and get started with a database.

Accessing MySQL via the Command Line

There are three main ways in which you can interact with MySQL: using a command line, via a web interface such as phpMyAdmin, and through a programming language like PHP. We'll start doing the third of these in Chapter 10, but for now, let's look at the first two.

Starting the Command-Line Interface

The following sections describe relevant instructions for Windows, OS X, and Linux.

Windows users

If you installed the Zend Server Free Edition WAMP (as explained in Chapter 2), you will be able to access the MySQL executable from one of the following directories (the first on 32-bit computers, and the second on 64-bit machines):

```
C:\Program Files\Zend\MySQL55\bin
C:\Program Files (x86)\Zend\MySQL55\bin
```

If you installed Zend Server in a place other than *\Program Files* (or *\Program Files (x86)*), you will need to use that directory instead.

By default, the initial MySQL user will be *root* and will not have had a password set. Seeing as this is a development server that only you should be able to access, we won't worry about creating one yet.

So, to enter MySQL's command-line interface, select Start→Run, enter `CMD` into the Run box, and press Return. This will call up a Windows Command Prompt. From there, enter one of the following (making any appropriate changes as just discussed):

```
"C:\Program Files\Zend\MySQL55\bin\mysql" -u root
"C:\Program Files (x86)\Zend\MySQL55\bin\mysql" -u root
```

Note the quotation marks surrounding the path and filename. These are present because the name contains spaces, which the Command Prompt doesn't correctly interpret, and the quotation marks group the parts of the filename into a single string for the command program to understand.

This command tells MySQL to log you in as user *root*, without a password. You will now be logged into MySQL and can start entering commands. So, to be sure everything is working as it should be, enter the following (the results should look similar to the output shown in Figure 8-1):

```
SHOW databases;
```

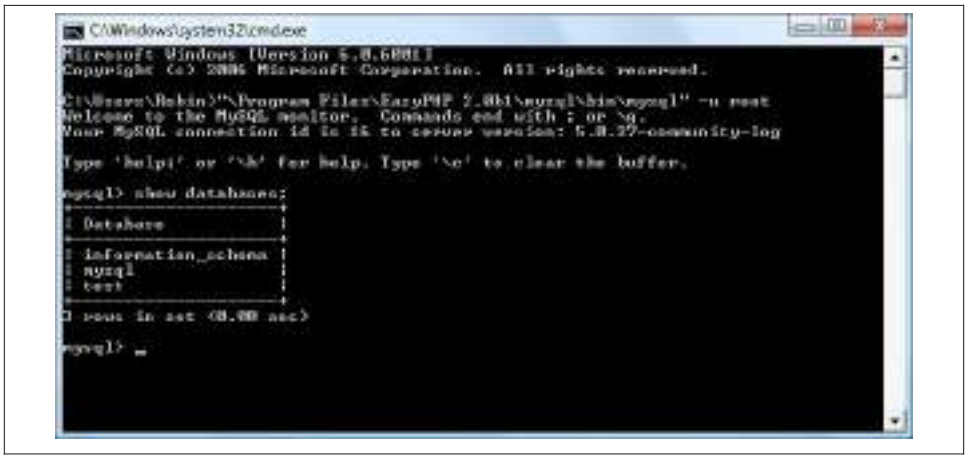


Figure 8-1. Accessing MySQL from a Windows Command Prompt

If this has not worked and you get an error, make sure that you have correctly installed MySQL along with Zend Server (as described in Chapter 2). Otherwise, you are ready to move on to the next section, "Using the Command-Line Interface" on page 177.

OS X users

To proceed with this chapter, you should have installed Zend Server as detailed in Chapter 2. You should also have the web server already running and the MySQL server started.

To enter the MySQL command-line interface, start the Terminal program (which should be available in Finder→Utilities). Then call up the MySQL program, which will have been installed in the directory */usr/local/zend/mysql/bin*.

By default, the initial MySQL user is *root*, and it will have a password of *root* too. So, to start the program, type the following:

```
/usr/local/zend/mysql/bin/mysql -u root
```

This command tells MySQL to log you in as user *root* and not to request your password. To verify that all is well, type the following (the result should look like the output shown in Figure 8-2):

```
SHOW databases;
```

Figure 8-2. Accessing MySQL from the OS X Terminal program

If you receive an error such as `Can't connect to local MySQL server through socket`, you haven't started up the MySQL server, so make sure you followed the advice in Chapter 2 about configuring MySQL to start when OS X starts.

You should now be ready to move on to the next section, "Using the Command-Line Interface" on page 177.

Linux users

On a system running a Unix-like operating system such as Linux, you will almost certainly already have PHP and MySQL installed and running, and you will be able to enter the examples in the next section. But first you should type the following to log into your MySQL system:

```
mysql -u root -p
```

This tells MySQL to log you in as the user *root* and to request your password. If you have a password, enter it; otherwise, just press Return.

Once you are logged in, type the following to test the program (you should see something like Figure 8-3 in response):

```
SHOW databases;
```

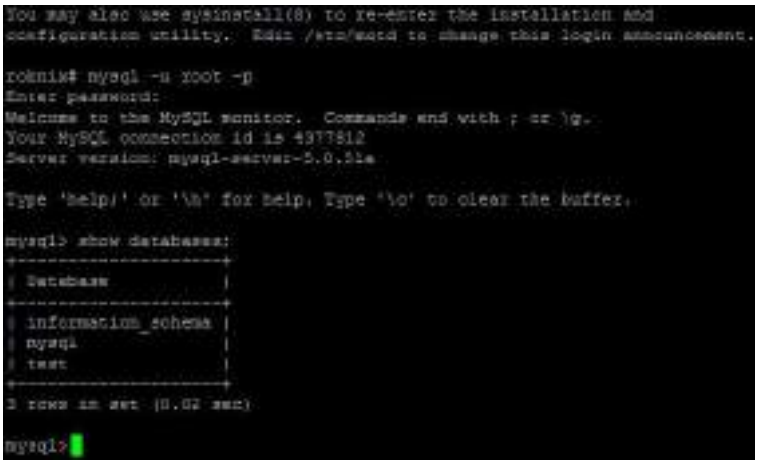


Figure 8-3. Accessing MySQL using Linux

If this procedure fails at any point, refer to the section "Installing a LAMP on Linux" on page 38 in Chapter 2 to ensure that you have MySQL properly installed. Otherwise, you should now be ready to move on to the next section, "Using the Command-Line Interface" on page 177.

MySQL on a remote server

If you are accessing MySQL on a remote server, you should Telnet (or preferably, for security, use SSH) into the remote machine, which will probably be a Linux/FreeBSD/Unix type of box. Once in there, you might find that things are a little different, depending on how the system administrator has set the server up, especially if it's a shared hosting server. Therefore, you need to ensure that you have been given access to MySQL and that you have your username and password. Armed with these, you can then type the following, where *username* is the name supplied:

```
mysql -u username -p
```

Enter your password when prompted. You can then try the following command, which should result in something like Figure 8-3:

```
SHOW databases;
```

There may be other databases already created, and the *test* database may not be there.

Bear in mind also that system administrators have ultimate control over everything and that you can encounter some unexpected setups. For example, you may find that you

are required to preface all database names that you create with a unique identifying string to ensure that you do not conflict with databases created by other users.

Therefore, if you have any problems, talk with your system administrator, who will get you sorted out. Just let the sysadmin know that you need a username and password. You should also ask for the ability to create new databases or, at a minimum, to have at least one database created for you ready to use. You can then create all the tables you require within that database.

Using the Command-Line Interface

From here on out, it makes no difference whether you are using Windows, Mac OS X, or Linux to access MySQL directly, as all the commands used (and errors you may receive) are identical.

The semicolon

Let's start with the basics. Did you notice the semicolon (`;`) at the end of the `SHOW databases;` command that you typed? The semicolon is used by MySQL to separate or end commands. If you forget to enter it, MySQL will issue a prompt and wait for you to do so. The required semicolon was made part of the syntax to let you enter multipleline commands, which can be convenient because some commands get quite long. It also allows you to issue more than one command at a time by placing a semicolon after each one. The interpreter gets them all in a batch when you press the Enter (or Return) key and executes them in order.

It's very common to receive a MySQL prompt instead of the results of your command; it means that you forgot the final semicolon. Just enter the semicolon and press the Enter key, to get what you want.

There are six different prompts that MySQL may present you with (see Table 8-2), so you will always know where you are during a multiline input.

Table 8-2. MySQL's six command prompts

MySQL prompt	Meaning
`mysql>`	Ready and waiting for a command
`->`	Waiting for the next line of a command
`'>`	Waiting for the next line of a string started with a single quote
`">`	Waiting for the next line of a string started with a double quote
`` `> ``	Waiting for the next line of a string started with a backtick

`/*>`	Waiting for the next line of a comment started with `/*`

Canceling a command

If you are partway through entering a command and decide you don't wish to execute it after all, whatever you do *don't press Control-C*! That will close the program. Instead, you can enter `\c` and press Return. Example 8-1 shows how to use it. *Example 8-1. Canceling a line of input* `meaningless gibberish to mysql \c`

When you enter that line, MySQL will ignore everything you typed and issue a new prompt. Without the `\c`, it would have displayed an error message. Be careful, though: if you have opened a string or comment, close it first before using the `\c` or MySQL will think the `\c` is just part of the string. Example 8-2 shows the right way to do this. *Example 8-2. Canceling input from inside a string* `this is "meaningless gibberish to mysql" \c`

Also note that using `\c` after a semicolon will not work, as it is then a new statement.

MySQL Commands

You've already seen the `SHOW` command, which lists tables, databases, and many other items. The commands you'll probably use most often are listed in Table 8-3.

Table 8-3. A selection of common MySQL commands

Command	Action
`ALTER`	Alter a database or table
`BACKUP`	Backup a table
`\c`	Cancel input
`CREATE`	Create a database
`DELETE`	Delete a row from a table
`DESCRIBE`	Describe a table's columns
`DROP`	Delete a database or table
`EXIT (CTRL-C)`	Exit
`GRANT`	Change user privileges
`HELP (\h, \?)`	Display help

Command	Action
INSERT	Insert data
LOCK	Lock table(s)
QUIT (\q)	Same as EXIT
RENAME	Rename a table
SHOW	List details about an object
SOURCE	Execute a file
STATUS (\s)	Display the current status
TRUNCATE	Empty a table
UNLOCK	Unlock table(s)
UPDATE	Update an existing record
USE	Use a database

I'll cover most of these as we proceed, but first, you need to remember a couple of points about MySQL commands:

- SQL commands and keywords are case-insensitive. `CREATE`, `create`, and `CrEaTe` all mean the same thing. However, for the sake of clarity, the recommended style is to use uppercase.
- Table names are case-sensitive on Linux and OS X, but case-insensitive on Windows. So for portability purposes, you should always choose a case and stick to it. The recommended style is to use lowercase for tables.

Creating a database

If you are working on a remote server and have only a single user account and access to a single database that was created for you, move on to the section "Creating a table" on page 181. Otherwise, get the ball rolling by issuing the following command to create a new database called *publications*:

```
CREATE DATABASE publications;
```

A successful command will return a message that doesn't mean much yet—`Query OK, 1 row affected (0.00 sec)`—but will make sense soon. Now that you've created the database, you want to work with it, so issue:

```
USE publications;
```

You should now see the message `Database changed` and will then be set to proceed with the following examples.

Creating users

Now that you've seen how easy it is to use MySQL, and created your first database, it's time to look at how you create users, as you probably won't want to grant your PHP scripts root access to MySQL; it could cause a real headache should you get hacked.

To create a user, issue the `GRANT` command, which takes the following form (don't type this in; it's not an actual working command):

```
GRANT PRIVILEGES ON database.object TO
'username'@'hostname'
  IDENTIFIED BY 'password';
```

All this should be pretty straightforward, with the possible exception of the `database.object` part, which refers to the database itself and the objects it contains, such as tables (see Table 8-4).

Table 8-4. Example parameters for the GRANT command

Arguments	Meaning
`*.*`	All databases and all their objects
`database``.*`	Only the database called *`database`* and all its objects
`database.object`	Only the database called *`database`* and its object called *`object`*

So let's create a user who can access just the new *publications* database and all its objects, by entering the following (replacing the username *jim* and the password *mypasswd* with ones of your choosing):

```
GRANT ALL ON publications.* TO 'jim'@'localhost' IDENTIFIED BY
'mypasswd';
```

What this does is allow the user *jim@localhost* full access to the *publications* database using the password *mypasswd*. You can test whether this step has worked by entering `quit` to exit and then rerunning MySQL the way you did before, but instead of entering **`-u root -p`**, type **`-u jim -p`**, or whatever username you created. See Table 8-5 for the correct command for your operating system. Modify it as necessary if the *mysql* client program is installed in a different directory on your system. *Table 8-5. Starting MySQL and logging in as jim@localhost*

OS	Example command
Windows	`"C:\Program Files\Zend\MySQL55\bin\mysql" -u jim -p`

Mac OS X	`/Applications/MAMP/Library/bin/mysql -u jim -p`
Linux	`mysql -u jim –p`

All you have to do now is enter your password when prompted and you will be logged in. By the way, if you prefer, you can place your password immediately following the `-p` (without any spaces) to avoid having to enter it when prompted. But this is considered a poor practice, because if other people are logged into your system, there may be ways for them to look at the command you entered and find out your password.

You can grant only privileges that *you* already have, and you must also have the privilege to issue `GRANT` commands. There is a whole range of privileges you can choose to grant if you are not granting all privileges. For further details, visit *http://tinyurl.com/mysqlgrant*, which also covers the `REVOKE` command, which can remove privileges once granted.

Also be aware that if you create a new user but do not specify an `IDENTIFIED BY` clause, the user will have no password, a situation that is very insecure and should be avoided.

Creating a table

At this point, you should now be logged into MySQL with `ALL` privileges granted for the database *publications* (or a database that was created for you), so you're ready to create your first table. Make sure the correct database is in use by typing the following (replacing *publications* with the name of your database if it is different):

```
USE publications;
```

Now enter the commands in Example 8-3 one line at a time.

Example 8-3. Creating a table called classics

```
CREATE TABLE classics (
author VARCHAR(128),
title VARCHAR(128),
type VARCHAR(16),  year
CHAR(4)) ENGINE MyISAM;
```

You could also issue this command on a single line like this:

```
CREATE TABLE classics (author VARCHAR(128), title
VARCHAR(128), type VARCHAR(16), year CHAR(4)) ENGINE
MyISAM;
```

but MySQL commands can be long and complicated, so I recommend one line per instruction until you are comfortable with longer lines.

MySQL should then issue the response `Query OK, 0 rows affected`, along with how long it took to execute the command. If you see an error message instead, check your syntax carefully. Every parenthesis and comma counts, and typing errors are easy to make. In case you are wondering, the `ENGINE MyISAM` tells MySQL the type of database engine to use for this table.

To check whether your new table has been created, type:

```
DESCRIBE classics;
```

All being well, you will see the sequence of commands and responses shown in Example 8-4, where you should particularly note the table format displayed. *Example 8-4. A MySQL session: creating and checking a new table*

```
mysql> USE publications;
Database changed mysql>
CREATE TABLE classics (
->  author VARCHAR(128),
    ->  title VARCHAR(128),
    ->  type VARCHAR(16),
    ->  year CHAR(4)) ENGINE
MyISAM; Query OK, 0 rows affected
(0.03 sec)

mysql> DESCRIBE classics;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra | +--------+---
-----------+------+-----+---------+-------+
| author | varchar(128) | YES  |     | NULL    |       |
| title  | varchar(128) | YES  |     | NULL    |       |
| type   | varchar(16)  | YES  |     | NULL    |       |
| year   | char(4)      | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

The `DESCRIBE` command is an invaluable debugging aid when you need to ensure that you have correctly created a MySQL table. You can also use it to remind yourself about a table's field or column names and the types of data in each one. Let's look at each of the headings in detail:

`Field`
: The name of each field or column within a table.

`Type`
: The type of data being stored in the field.

`Null`
: Whether a field is allowed to contain a value of `NULL`.

`Key`

MySQL supports *keys* or *indexes*, which are quick ways to look up and search for data. The `Key` heading shows what type of key (if any) has been applied.

`Default`
: The default value that will be assigned to the field if no value is specified when a new row is created.

`Extra`
: Additional information, such as whether a field is set to auto-increment.

Data Types

In Example 8-3, you may have noticed that three of the table's fields were given the data type of `VARCHAR`, and one was given the type `CHAR`. The term `VARCHAR` stands for *VARiable length CHARacter string*, and the command takes a numeric value that tells MySQL the maximum length allowed for a string stored in this field.

This data type is very useful, as MySQL can then plan the size of databases and perform lookups and searches more easily. The downside is that if you ever attempt to assign a string value longer than the length allowed, it will be truncated to the maximum length declared in the table definition.

The `year` field, however, has more predictable values, so instead of `VARCHAR` we use the more efficient `CHAR(4)` data type. The parameter of `4` allows for four bytes of data, supporting all years from −999 to 9999; a byte comprises 8 bits and can have the values 00000000 through 11111111, which are 0 to 255 in decimal.

You could, of course, just store two-digit values for the year, but if your data is going to still be needed in the following century, or may otherwise wrap around, it will have to be sanitized first—much like the "millennium bug" that would have caused dates beginning on January 1, 2000, to be treated as 1900 on many of the world's biggest computer installations.

The reason I didn't use the `YEAR` data type in the *classics* table is because it supports only the year 0000, and years 1901 through 2155. This is because MySQL stores the year in a single byte for reasons of efficiency, but it also means that only 256 years are available, and the publication years of the titles in the *classics* table are well before this.

Both `CHAR` and `VARCHAR` accept text strings and impose a limit on the size of the field. The difference is that every string in a `CHAR` field has the specified size. If you put in a smaller string, it is padded with spaces. A `VARCHAR` field does not pad the text; it lets the size of the field vary to fit the text that is inserted. But `VARCHAR` requires a

small amount of overhead to keep track of the size of each value. So CHAR is slightly more efficient if the sizes are similar in all records, whereas VARCHAR is more efficient if sizes can vary a lot and get large. In addition, the overhead causes access to VARCHAR data to be slightly slower than to CHAR data.

The CHAR data type

Table 8-6 lists the CHAR data types. All these types offer a parameter that sets the maximum (or exact) length of the string allowed in the field. As the table shows, each type has a built-in maximum number of bytes it can occupy.

Table 8-6. MySQL's CHAR data types

Data type	Bytes used	Examples
CHAR(*n*)	exactly *n* (< 256)	CHAR(5) *"Hello" uses 5 bytes* CHAR(57) *"Goodbye" uses 57 bytes*
VARCHAR(*n*)	up to *n* (< 65,536)	VARCHAR(7) *"Morning" uses 7 bytes* VARCHAR(100) *"Night" uses 5 bytes*

The BINARY data type

The BINARY data type is used for storing strings of full bytes that do not have an associated character set. For example, you might use the BINARY data type to store a GIF image (see Table 8-7).

Table 8-7. MySQL's BINARY data types

Data type	Bytes used	Examples
BINARY(*n*) or BYTE(*n*)	exactly *n* (< 256)	As CHAR but contains binary data
VARBINARY(*n*)	up to *n* (< 65,536)	As VARCHAR but for binary data

The TEXT and VARCHAR data types

The differences between TEXT and VARCHAR are small:

- Prior to version 5.0.3, MySQL would remove leading and trailing spaces from VARCHAR fields.
- TEXT fields cannot have default values.
- MySQL indexes only the first *n* characters of a TEXT column (you specify *n* when you create the index).

What this means is that VARCHAR is the better and faster data type to use if you need to search the entire contents of a field. If you will never search more than a certain number of leading characters in a field, you should probably use a TEXT data type (see Table 8-8).

Table 8-8. MySQL's TEXT data types

Data type	Bytes used	Attributes
TINYTEXT(*n*)	up to *n* (< 256)	Treated as a string with a character set
TEXT(*n*)	up to *n* (< 65,536)	Treated as a string with a character set
MEDIUMTEXT(*n*)	up to *n* (< 1.67e+7)	Treated as a string with a character set
LONGTEXT(*n*)	up to *n* (< 4.29e+9)	Treated as a string with a character set

The BLOB data type

The term BLOB stands for *Binary Large OBject* and therefore, as you would think, the BLOB data type is most useful for binary data in excess of 65,536 bytes in size. The main other difference between the BLOB and BINARY data types is that BLOBs cannot have default values (see Table 8-9).

Table 8-9. MySQL's BLOB data types

Data type	Bytes used	Attributes
TINYBLOB(*n*)	up to *n* (< 256)	Treated as binary data—no character set
BLOB(*n*)	up to *n* (<= 65,536)	Treated as binary data—no character set
MEDIUMBLOB(*n*)	up to *n* (< 1.67e+7)	Treated as binary data—no character set
LONGBLOB(*n*)	up to *n* (< 4.29e+9)	Treated as binary data—no character set

Numeric data types

MySQL supports various numeric data types from a single byte up to double-precision floating-point numbers. Although the most memory that a numeric field can use up is 8 bytes, you are well advised to choose the smallest data type that will adequately handle the largest value you expect. Your databases will be small and quickly accessible.

Table 8-10 lists the numeric data types supported by MySQL and the ranges of values they can contain. In case you are not acquainted with the terms, a signed number is one with a possible range from a minus value, through 0, to a positive one, and an unsigned one has a value ranging from 0 to a positive one. They can both hold the same number

of values; just picture a signed number as being shifted halfway to the left so that half its values are negative and half are positive. Note that floating-point values (of any precision) may only be signed. *Table 8-10. MySQL's numeric data types*

Data type	Bytes used	Minimu value		Maximum value	
		Signed	Unsigned	Signed	Unsigned
TINYINT	1	−128	0	127	255
SMALLINT	2	−32,768	0	32,767	65,535
MEDIUMINT	3	−8.38e+6	0	8.38e+6	1.67e+7
INT or INTEGER	4	−2.15e+9	0	2.15e+9	4.29e+9
BIGINT	8	−9.22e+18	0	9.22e+18	1.84e+19
FLOAT	4	−3.40e+38	*n/a*	3.40e+38	*n/a*
DOUBLE or REAL	8	−1.80e+308	*n/a*	1.80e+308	*n/a*

To specify whether a data type is signed or unsigned, use the `UNSIGNED` qualifier. The following example creates a table called *tablename* with a field in it called *fieldname* of the data type `UNSIGNED INTEGER`:

```
CREATE TABLE tablename (fieldname INT UNSIGNED);
```

When creating a numeric field, you can also pass an optional number as a parameter, like this:

```
CREATE TABLE tablename (fieldname INT(4));
```

But you must remember that, unlike `BINARY` and `CHAR` data types, this parameter does not indicate the number of bytes of storage to use. It may seem counterintuitive, but what the number actually represents is the display width of the data in the field when it is retrieved. It is commonly used with the `ZEROFILL` qualifier like this:

```
CREATE TABLE tablename (fieldname INT(4) ZEROFILL);
```

What this does is cause any numbers with a width of less than four characters to be padded with one or more zeros, sufficient to make the display width of the field four characters long. When a field is already of the specified width or greater, no padding takes place.

DATE and TIME

The main remaining data types supported by MySQL relate to the date and time and can be seen in Table 8-11.

Table 8-11. MySQL's DATE and TIME data types

Data type	Time/date format
DATETIME	'0000-00-00 00:00:00'
DATE	'0000-00-00'
TIMESTAMP	'0000-00-00 00:00:00'
TIME	'00:00:00'
YEAR	0000 (Only years 0000 and 1901–2155)

The DATETIME and TIMESTAMP data types display the same way. The main difference is that TIMESTAMP has a very narrow range (from the years 1970 through 2037), whereas DATETIME will hold just about any date you're likely to specify, unless you're interested in ancient history or science fiction.

TIMESTAMP is useful, however, because you can let MySQL set the value for you. If you don't specify the value when adding a row, the current time is automatically inserted.
You can also have MySQL update a TIMESTAMP column each time you change a row.

The AUTO_INCREMENT data type

Sometimes you need to ensure that every row in your database is guaranteed to be unique. You could do this in your program by carefully checking the data you enter and making sure that there is at least one value that differs in any two rows, but this approach is error-prone and works only in certain circumstances. In the *classics* table, for instance, an author may appear multiple times. Likewise, the year of publication will also be frequently duplicated, and so on. It would be hard to guarantee that you have no duplicate rows.

The general solution is to use an extra column just for this purpose. In a while, we'll look at using a publication's ISBN (International Standard Book Number), but first I'd like to introduce the AUTO_INCREMENT data type.

As its name implies, a column given this data type will set the value of its contents to that of the column entry in the previously inserted row, plus 1. Example 8-5 shows how to add a new column called *id* to the table *classics* with auto-incrementing.

Example 8-5. Adding the auto-incrementing column id

```
ALTER TABLE classics ADD id INT UNSIGNED NOT NULL
AUTO_INCREMENT KEY;
```

This is your introduction to the ALTER command, which is very similar to the CREATE command. ALTER operates on an existing table, and can add, change, or delete columns. Our example adds a column named *id* with the following characteristics:

```
INT UNSIGNED
```

Makes the column take an integer large enough for you to store more than 4 billion records in the table.

`NOT NULL`

Ensures that every column has a value. Many programmers use `NULL` in a field to indicate that the field doesn't have any value. But that would allow duplicates, which would violate the whole reason for this column's existence. So we disallow `NULL` values. `AUTO_INCREMENT`

Causes MySQL to set a unique value for this column in every row, as described earlier. We don't really have control over the value that this column will take in each row, but we don't care: all we care about is that we are guaranteed a unique value.

`KEY`

An auto-increment column is useful as a key, because you will tend to search for rows based on this column, as explained in the section "Indexes" on page 192.

Each entry in the column *id* will now have a unique number, with the first starting at 1 and the others counting upward from there. And whenever a new row is inserted, its *id* column will automatically be given the next number in sequence.

Rather than applying the column retroactively, you could have included it by issuing the `CREATE` command in slightly different format. In that case, the command in Example 8-3 would be replaced with Example 8-6. Check the final line in particular.

Example 8-6. Adding the auto-incrementing id column at table creation

```
CREATE TABLE classics (  author VARCHAR(128),
title VARCHAR(128),  type VARCHAR(16),  year
CHAR(4),  id INT UNSIGNED NOT NULL AUTO_INCREMENT
KEY) ENGINE MyISAM;
```

If you wish to check whether the column has been added, use the following command to view the table's columns and data types:

```
DESCRIBE classics;
```

Now that we've finished with it, the *id* column is no longer needed, so if you created it using Example 8-5, you should now remove the column using the command in Example 8-7.

Example 8-7. Removing the id column

```
ALTER TABLE classics DROP id;
```

Adding data to a table

To add data to a table, use the `INSERT` command. Let's see this in action by populating the table *classics* with the data from Table 8-1, using one form of the `INSERT` command repeatedly (Example 8-8).

Example 8-8. Populating the classics table

```
INSERT INTO classics(author, title, type, year)
VALUES('Mark Twain','The Adventures of Tom
Sawyer','Fiction','1876');
INSERT INTO classics(author, title, type, year)
 VALUES('Jane Austen','Pride and
Prejudice','Fiction','1811');
INSERT INTO classics(author, title, type, year)
 VALUES('Charles Darwin','The Origin of Species','Non-
Fiction','1856');
INSERT INTO classics(author, title, type, year)
 VALUES('Charles Dickens','The Old Curiosity
Shop','Fiction','1841');
INSERT INTO classics(author, title, type, year)
 VALUES('William Shakespeare','Romeo and
Juliet','Play','1594');
```

After every second line, you should see a `Query OK` message. Once all lines have been entered, type the following command, which will display the table's contents (the result should look like Figure 8-4):

```
SELECT * FROM classics;
```

Don't worry about the `SELECT` command for now—we'll come to it in the section "Querying a MySQL Database" on page 198. Suffice it to say that, as typed, it will display all the data you just entered.

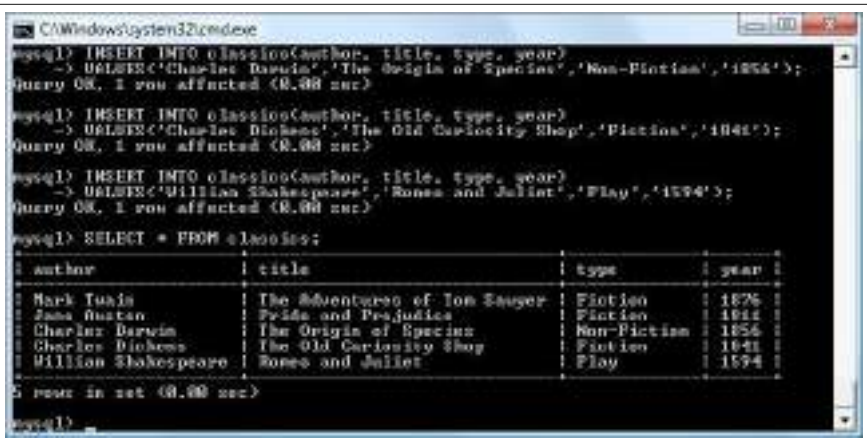


Figure 8-4. Populating the classics table and viewing its contents

Let's go back and look at how we used the `INSERT` command. The first part, `INSERT INTO classics`, tells MySQL where to insert the following data. Then, within parentheses, the four column names are listed—*author*, *title*, *type*, and *year*—all separated by commas. This tells MySQL that these are the fields into which the data is to be inserted.

The second line of each `INSERT` command contains the keyword `VALUES` followed by four strings within parentheses, and separated by commas. This supplies MySQL with the four values to be inserted into the four columns previously specified. (As always, my choice of where to break the lines was arbitrary.)

Each item of data will be inserted into the corresponding column, in a one-to-one correspondence. If you accidentally listed the columns in a different order from the data, the data would go into the wrong columns. And the number of columns must match the number of data items.

Renaming a table

Renaming a table, like any other change to the structure or meta information about a table, is achieved via the `ALTER` command. So, for example, to change the name of table *classics* to *pre1900*, use the following command:

```
ALTER TABLE classics RENAME pre1900;
```

If you tried that command, you should revert the table name by entering the following, so that later examples in this chapter will work as printed:

```
ALTER TABLE pre1900 RENAME classics;
```

Changing the data type of a column

Changing a column's data type also makes use of the `ALTER` command, this time in conjunction with the `MODIFY` keyword. So to change the data type of column *year* from `CHAR(4)` to `SMALLINT` (which requires only two bytes of storage and so will save disk space), enter the following:

```
ALTER TABLE classics MODIFY year SMALLINT;
```

When you do this, if the conversion of data type makes sense to MySQL, it will automatically change the data while keeping the meaning. In this case, it will change each string to a comparable integer, and so on, as the string is recognizable as referring to an integer.

Adding a new column

Let's suppose that you have created a table and populated it with plenty of data, only to discover you need an additional column. Not to worry. Here's how to add the new column *pages*, which will be used to store the number of pages in a publication:

```
ALTER TABLE classics ADD pages SMALLINT UNSIGNED;
```

This adds the new column with the name *pages* using the `UNSIGNED SMALLINT` data type, sufficient to hold a value of up to 65,535—hopefully that's more than enough for any book ever published!

And, if you ask MySQL to describe the updated table using the `DESCRIBE` command, as follows, you will see the change has been made (see Figure 8-5):

```
DESCRIBE classics;
```

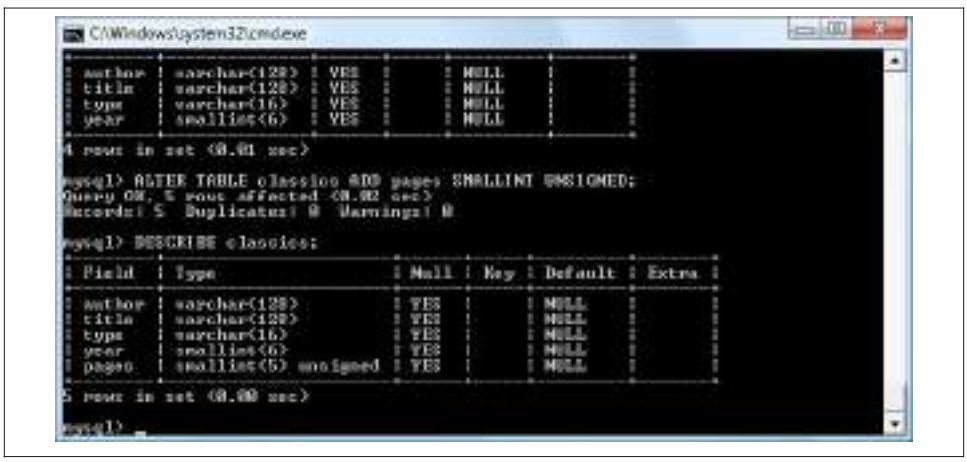


Figure 8-5. Adding the new pages column and viewing the table

Renaming a column

Looking again at Figure 8-5, you may decide that having a column named *type* can be confusing, because that is the name used by MySQL to identify data types. Again, no problem—let's change its name to *category*, like this:

```
ALTER TABLE classics CHANGE type category VARCHAR(16);
```

Note the addition of `VARCHAR(16)` on the end of this command. That's because the `CHANGE` keyword requires the data type to be specified, even if you don't intend to change it, and `VARCHAR(16)` was the data type specified when that column was initially created as *type*.

Removing a column

Actually, upon reflection, you might decide that the page count column *pages* isn't actually all that useful for this particular database, so here's how to remove that column using the DROP keyword:

```
ALTER TABLE classics DROP pages;
```

Remember that DROP is irreversible and you should always use it with caution, because you could inadvertently delete entire tables (and even databases) with it if you are not careful!

Deleting a table

Deleting a table is very easy indeed. But, because I don't want you to have to reenter all the data for the *classics* table, let's quickly create a new table, verify its existence, and then delete it by typing the commands in Example 8-9. The result of these four commands should look like Figure 8-6.

Example 8-9. Creating, viewing, and deleting a table

```
CREATE TABLE disposable(trash INT);
DESCRIBE disposable;
DROP TABLE disposable;
SHOW tables;
```

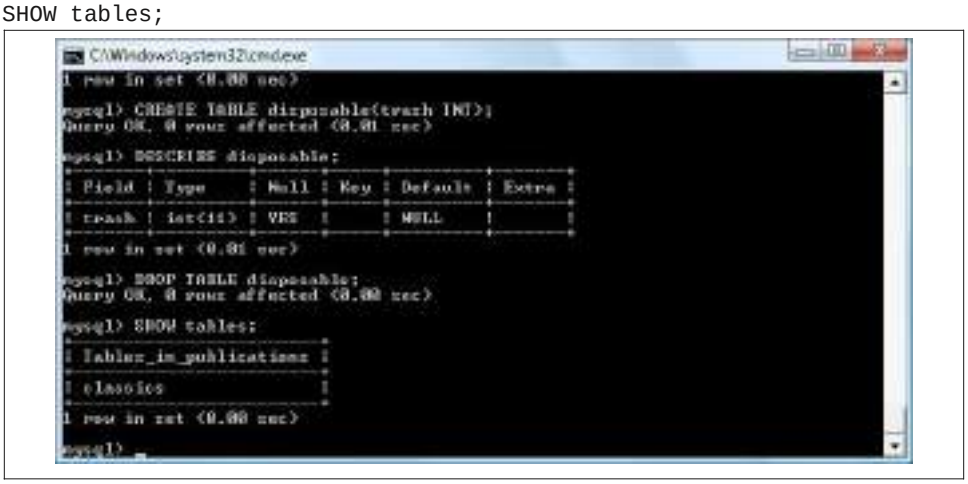


Figure 8-6. Creating, viewing, and deleting a table

Indexes

As things stand, the table *classics* works and can be searched without problem by MySQL—until it grows to more than a couple of hundred rows, that is. At that point, database accesses will get slower and slower with every new row added, because MySQL has to search through every row whenever a query is issued. This is like searching through every book in a library whenever you need to look something up.

Of course, you don't have to search libraries that way, because they have either a card index system or, most likely, a database of their own. And the same goes for MySQL, because at the expense of a slight overhead in memory and disk space, you can create a "card index" for a table that MySQL will use to conduct lightning-fast searches.

Creating an Index

The way to achieve fast searches is to add an *index*, either when creating a table or at any time afterward. But the decision is not so simple. For example, there are different index types such as a regular `INDEX`, `PRIMARY KEY`, and `FULLTEXT`. Also, you must decide which columns require an index, a judgment that requires you to predict whether you will be searching any of the data in that column. Indexes can also get complicated, because you can combine multiple columns in one index. And even when you've decided that, you still have the option of reducing index size by limiting the amount of each column to be indexed.

If we imagine the searches that may be made on the *classics* table, it becomes apparent that all of the columns may need to be searched. However, if the *pages* column created in the section "Adding a new column" on page 190 had not been deleted, it would probably not have needed an index, as most people would be unlikely to search for books by the number of pages they have. Anyway, go ahead and add an index to each of the columns, using the commands in Example 8-10.

Example 8-10. Adding indexes to the classics table

```
ALTER TABLE classics ADD INDEX(author(20));
ALTER TABLE classics ADD INDEX(title(20));
ALTER TABLE classics ADD
INDEX(category(4)); ALTER TABLE classics
ADD INDEX(year);
DESCRIBE classics;
```

The first two commands create indexes on both the *author* and *title* columns, limiting each index to only the first 20 characters. For instance, when MySQL indexes the following title:

```
The Adventures of Tom Sawyer
```

It will actually store in the index only the first 20 characters:

```
The Adventures of To
```

This is done to minimize the size of the index, and to optimize database access speed. I chose 20 because it's likely to be sufficient to ensure uniqueness for most strings in these columns. If MySQL finds two indexes with the same contents, it will have to waste time going to the table itself and checking the column that was indexed to find out which rows really matched.

With the *category* column, currently only the first character is required to identify a string as unique (F for Fiction, N for Non-Fiction, and P for Play), but I chose an index of four characters to allow for future category types that may be unique only after four characters. You can also re-index this column later, when you have a more complete set of categories. And finally, I set no limit to the *year* column's index, because it's an integer, not a string.

The results of issuing these commands (and a `DESCRIBE` command to confirm that they worked) can be seen in Figure 8-7, which shows the key `MUL` for each column. This key means that multiple occurrences of a value may occur within that column, which is exactly what we want, as authors may appear many times, the same book title could be used by multiple authors, and so on.

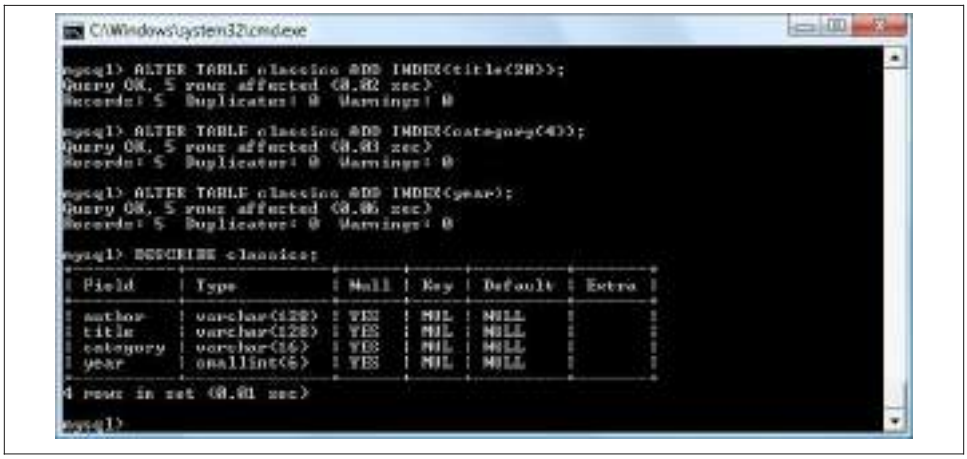


Figure 8-7. Adding indexes to the classics table

Using CREATE INDEX

An alternative to using `ALTER TABLE` to add an index is to use the `CREATE INDEX` command. They are equivalent, except that `CREATE INDEX` cannot be used to create a `PRIMARY KEY` (see the section "Primary keys" on page 195). The format of this command is shown in the second line of Example 8-11.

Example 8-11. These two commands are equivalent

```
ALTER TABLE classics ADD INDEX(author(20));
CREATE INDEX author ON classics (author(20));
```

Adding indexes when creating tables

You don't have to wait until after creating a table to add indexes. In fact, doing so can be time consuming, as adding an index to a large table can take a very long time. Therefore, let's look at a command that creates the table *classics* with indexes already in place. Example 8-12 is a reworking of Example 8-3 in which the indexes are created at the same time as the table. Note that to incorporate the modifications made in this chapter, this version uses the new column name *category* instead of *type* and sets the data type of *year* to `SMALLINT` instead of `CHAR(4)`. If you want to try it out without first deleting your current *classics* table, change the word *classics* in line 1 to something else like *classics1*, then drop *classics1* after you have finished with it.

Example 8-12. Creating the table classics with indexes

```
CREATE TABLE classics
(  author
VARCHAR(128),  title
VARCHAR(128),
```

```
category VARCHAR(16),
year SMALLINT,
INDEX(author(20)),
 INDEX(title(20)),
 INDEX(category(4)),
 INDEX(year)) ENGINE MyISAM;
```

Primary keys

So far, you've created the table *classics* and ensured that MySQL can search it quickly by adding indexes, but there's still something missing. All the publications in the table can be searched, but there is no single unique key for each publication to enable instant accessing of a row. The importance of having a key with a unique value for each row will come up when we start to combine data from different tables.

The section "The AUTO_INCREMENT data type" on page 186 briefly introduced the idea of a primary key when creating the auto-incrementing column *id*, which could have been used as a primary key for this table. However, I wanted to reserve that task for a more appropriate column: the internationally recognized ISBN number. So let's go ahead and create a new column for this key. Now, bearing in mind that ISBNs are 13 characters long, you might think that the following command would do the job:

```
ALTER TABLE classics ADD isbn CHAR(13) PRIMARY KEY;
```

But it doesn't. If you try it, you'll get the error `Duplicate entry` for key 1. The reason is that the table is already populated with some data and this command is trying to add a column with the value `NULL` to each row, which is not allowed, as all values must be unique in any column having a primary key index. However, if there were no data already in the table, this command would work just fine, as would adding the primary key index upon table creation.

In our current situation, we have to be a bit sneaky and create the new column without an index, populate it with data, and then add the index retrospectively using the commands in Example 8-13. Luckily, each of the years is unique in the current set of data, so we can use the *year* column to identify each row for updating. Note that this example uses the `UPDATE` and `WHERE` keywords, which are explained in more detail in the section "Querying a MySQL Database" on page 198.

Example 8-13. Populating the isbn column with data and using a primary key

```
ALTER TABLE classics ADD isbn CHAR(13);
UPDATE classics SET isbn='9781598184891' WHERE year='1876';
UPDATE classics SET isbn='9780582506206' WHERE year='1811';
UPDATE classics SET isbn='9780517123201' WHERE year='1856';
UPDATE classics SET isbn='9780099533474' WHERE year='1841';
UPDATE classics SET isbn='9780192814968' WHERE year='1594';
ALTER TABLE classics ADD PRIMARY KEY(isbn);
```

```
DESCRIBE classics;
```

Once you have typed these commands, the results should look like Figure 8-8. Note that the keywords `PRIMARY KEY` replace the keyword `INDEX` in the `ALTER TABLE` syntax (compare Examples 8-10 and 8-13).

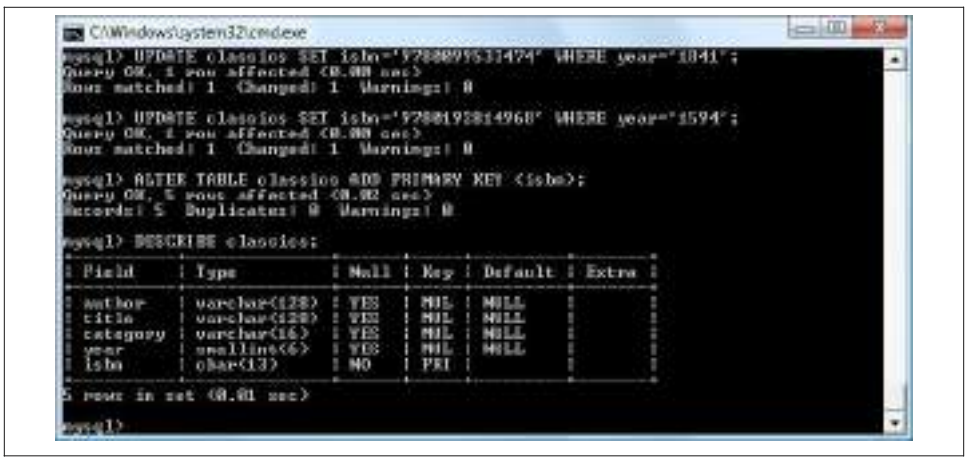


Figure 8-8. Retrospectively adding a primary key to the classics table

To have created a primary key when the table *classics* was created, you could have used the commands in Example 8-14. Again, rename *classics* in line 1 to something else if you wish to try this example for yourself, and then delete the test table afterward.

Example 8-14. Creating the table classics with a primary key

```
CREATE TABLE
classics (  author
VARCHAR(128),
title VARCHAR(128),
category
VARCHAR(16),  year
SMALLINT,  isbn
CHAR(13),
 INDEX(author(20)),
 INDEX(title(20)),
 INDEX(category(4)),
 INDEX(year),
 PRIMARY KEY (isbn)) ENGINE MyISAM;
```

Creating a FULLTEXT index

Unlike a regular index, MySQL's `FULLTEXT` allows super-fast searches of entire columns of text. It stores every word in every data string in a special index that you can search using "natural language," in a similar manner to using a search engine.

Actually, it's not strictly true that MySQL stores *all* the words in a FULLTEXT index, because it has a built-in list of more than 500 words that it chooses to ignore because they are so common that they aren't very helpful for searching anyway. This list, called *stopwords*, includes *the*, *as*, *is*, *of*, and so on. The list helps MySQL run much more quickly when performing a FULLTEXT search and keeps database sizes down. Appendix C contains the full list of stopwords.

Here are some things that you should know about FULLTEXT indexes:

- FULLTEXT indexes can be used only with MyISAM tables, the type used by MySQL's default storage engine (MySQL supports at least 10 different storage engines). If you need to convert a table to MyISAM, you can usually use the MySQL command `ALTER TABLE tablename ENGINE = MyISAM;`.
- FULLTEXT indexes can be created for CHAR, VARCHAR, and TEXT columns only.
- A FULLTEXT index definition can be given in the CREATE TABLE statement when a table is created, or added later using ALTER TABLE (or CREATE INDEX).
- For large data sets, it is *much* faster to load your data into a table that has no FULLTEXT index and then create the index than to load data into a table that has an existing FULLTEXT index.

To create a FULLTEXT index, apply it to one or more records as in Example 8-15, which adds a FULLTEXT index to the pair of columns *author* and *title* in the table *classics* (this index is in addition to the ones already created and does not affect them). *Example 8-15. Adding a FULLTEXT index to the table classics*

```
ALTER TABLE classics ADD FULLTEXT(author,title);
```

You can now perform FULLTEXT searches across this pair of columns. This feature could really come into its own if you could now add the entire text of these publications to the database (particularly as they're out of copyright protection) and they would be fully searchable. See the section "MATCH ... AGAINST" on page 202 for a description of searches using FULLTEXT.

If you find that MySQL is running slower than you think it should be when accessing your database, the problem is usually related to your indexes. Either you don't have an index where you need one, or the indexes are not optimally designed. Tweaking a table's indexes will often solve such a problem. Performance is beyond the scope of this book, but in Chapter 9 I give you a few tips so you know what to look for.

Querying a MySQL Database

So far, we've created a MySQL database and tables, populated them with data, and added indexes to make them fast to search. Now it's time to look at how these searches are performed, and the various commands and qualifiers available.

SELECT

As you saw in Figure 8-4, the `SELECT` command is used to extract data from a table. In that section, I used its simplest form to select all data and display it—something you will never want to do on anything but the smallest tables, because all the data will scroll by at an unreadable pace. Let's now examine `SELECT` in more detail. The basic syntax is:

```
SELECT something FROM tablename;
```

The *`something`* can be an `*` (asterisk) as you saw before, which means "every column," or you can choose to select only certain columns. For instance, Example 8-16 shows how to select just the *author* and *title* and just the *title* and *isbn*. The result of typing these commands can be seen in Figure 8-9.

Example 8-16. Two different SELECT statements

```
SELECT author,title FROM classics;
SELECT title,isbn FROM classics;
```

Figure 8-9. The output from two different SELECT statements

SELECT COUNT

Another replacement for the *something* parameter is `COUNT`, which can be used in many ways. In Example 8-17, it displays the number of rows in the table by passing `*` as a parameter, which means "all rows." As you'd expect, the result returned is `5`, as there are five publications in the table.

Example 8-17. Counting rows

```
SELECT COUNT(*) FROM classics;
```

SELECT DISTINCT

This qualifier (and its synonym `DISTINCTROW`) allows you to weed out multiple entries when they contain the same data. For instance, suppose that you want a list of all authors in the table. If you select just the *author* column from a table containing multiple books by the same author, you'll normally see a long list with same author names over and over. But by adding the `DISTINCT` keyword, you can show each author just once. So let's test that out by adding another row that repeats one of our existing authors (Example 8-18).

Example 8-18. Duplicating data

```
INSERT INTO classics(author, title, category, year, isbn)
 VALUES('Charles Dickens','Little Dorrit','Fiction','1857',
'9780141439969');
```

Now that Charles Dickens appears twice in the table, we can compare the results of using `SELECT` with and without the `DISTINCT` qualifier. Example 8-19 and Figure 8-10 show that the simple `SELECT` lists Dickens twice, and the command with the `DISTINCT` qualifier shows him only once.

Example 8-19. With and without the DISTINCT qualifier

```
SELECT author FROM classics;
```

```
SELECT DISTINCT author FROM classics;
```

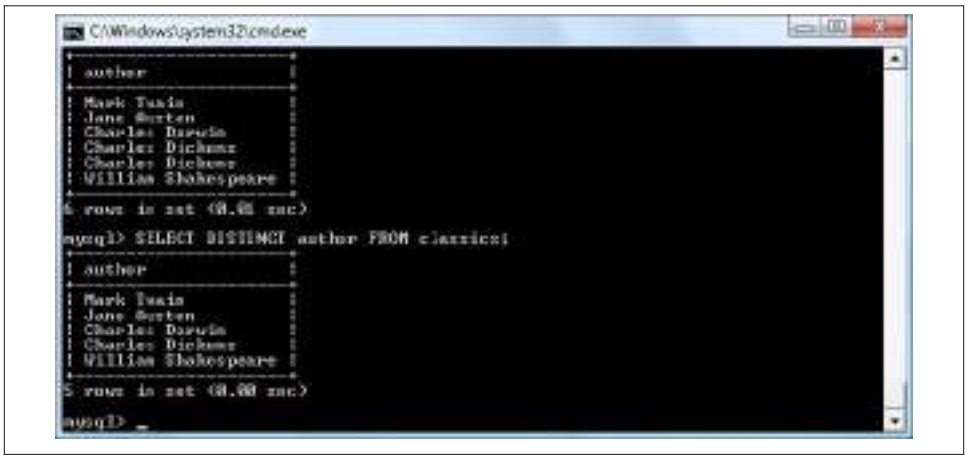


Figure 8-10. Selecting data with and without DISTINCT

DELETE

When you need to remove a row from a table, use the `DELETE` command. Its syntax is similar to the `SELECT` command and allows you to narrow down the exact row or rows to delete using qualifiers such as `WHERE` and `LIMIT`.

Now that you've seen the effects of the `DISTINCT` qualifier, if you entered Example 8-18, you should remove *Little Dorrit* by entering the commands in Example 8-20.

Example 8-20. Removing the new entry

```
DELETE FROM classics WHERE title='Little Dorrit';
```

This example issues a `DELETE` command for all rows whose *title* column contains the string `Little Dorrit`.

The `WHERE` keyword is very powerful, and important to enter correctly; an error could lead a command to the wrong rows (or have no effect in cases where nothing matches the `WHERE` clause). So now we'll spend some time on that clause, which is the heart and soul of SQL.

WHERE

The `WHERE` keyword enables you to narrow down queries by returning only those *where* a certain expression is true. Example 8-20 returns only the rows where the column exactly matches the string `Little Dorrit`, using the equality operator `=`. Example 8-21 shows a couple more examples of using `WHERE` with `=`.

Example 8-21. Using the WHERE keyword

```
SELECT author,title FROM classics WHERE author="Mark
Twain";
SELECT author,title FROM classics WHERE isbn="9781598184891
";
```

Given our current table, the two commands in Example 8-21 display the same results. But we could easily add more books by Mark Twain, in which case the first line would display all titles he wrote and the second line would continue (because we know the ISBN is unique) to display `The Adventures of Tom Sawyer`. In other words, searches using a unique key are more predictable, and you'll see further evidence later of the value of unique and primary keys.

You can also do pattern matching for your searches using the `LIKE` qualifier, which allows searches on parts of strings. This qualifier should be used with a `%` character before or after some text. When placed before a keyword, `%` means "anything before" and after a keyword it means "anything after." Example 8-22 performs three different queries, one for the start of a string, one for the end, and one for anywhere in a string. You can see the results of these commands in Figure 8-11.

Example 8-22. Using the LIKE qualifier

```
SELECT author,title FROM classics WHERE author LIKE
"Charles%";
SELECT author,title FROM classics WHERE title LIKE
"%Species";
SELECT author,title FROM classics WHERE title LIKE "%and%";
```

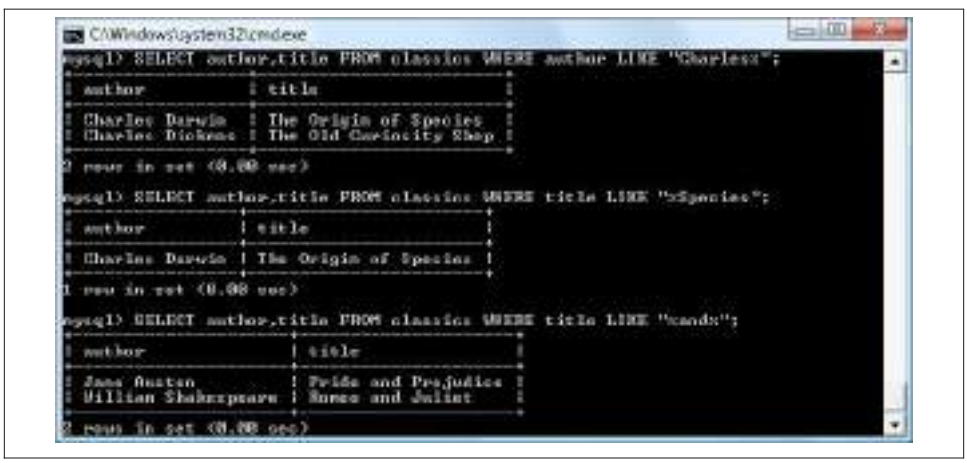


Figure 8-11. Using WHERE with the LIKE qualifier

The first command outputs the publications by both Charles Darwin and Charles Dickens because the `LIKE` qualifier was set to return anything matching the string `Charles` followed by any other text. Then just `The Origin of Species` is returned, because it's the only row whose column ends with the string `Species`. Last, both `Pride and Prejudice` and `Romeo and Juliet` are returned, because they both matched the string `and` anywhere in the column.

The `%` will also match if there is nothing in the position it occupies; in other words, it can match an empty string.

LIMIT

The `LIMIT` qualifier enables you to choose how many rows to return in a query, and where in the table to start returning them. When passed a single parameter, it tells MySQL to start at the beginning of the results and just return the number of rows given in that parameter. If you pass it two parameters, the first indicates the offset from the start of the results where MySQL should start the display, and the second indicates how many to return. You can think of the first parameter as saying, "Skip this number of results at the start."

Example 8-23 includes three commands. The first returns the first three rows from the table. The second returns two rows starting at position 1 (skipping the first row). The last command returns a single row starting at position 3 (skipping the first three rows). Figure 8-12 shows the results of issuing these three commands.

Example 8-23. Limiting the number of results returned

```
SELECT author,title FROM classics LIMIT 3;
SELECT author,title FROM classics LIMIT 1,2;
SELECT author,title FROM classics LIMIT 3,1;
```

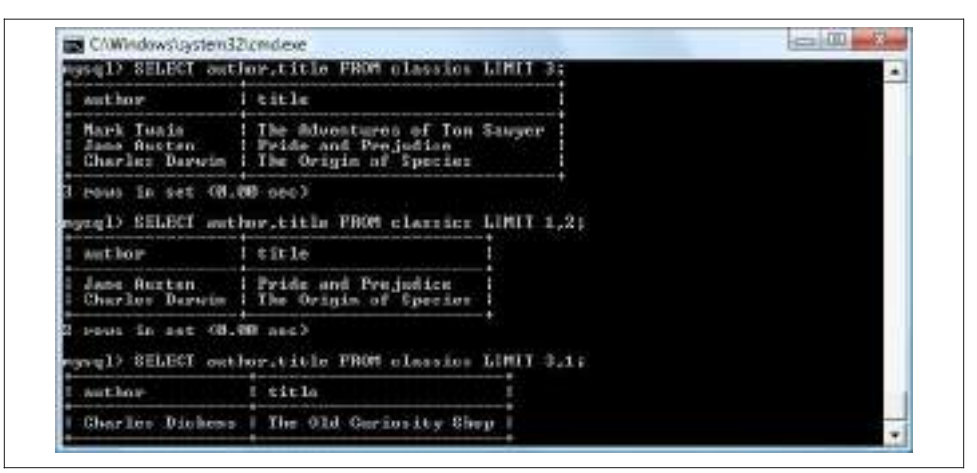


Figure 8-12. Restricting the rows returned with LIMIT

Be careful with the `LIMIT` keyword, because offsets start at 0, but the number of rows to return starts at 1. So `LIMIT 1,3` means return *three* rows starting from the *second* row.

MATCH ... AGAINST

The `MATCH ... AGAINST` construct can be used on columns that have been given a `FULLTEXT` index (see the section "Creating a FULLTEXT index" on page 196). With it, you can make natural-language searches as you would in an Internet search engine. Unlike the use of `WHERE ... =` or `WHERE ... LIKE`, `MATCH ... AGAINST` lets you enter multiple words in a search query and checks them against all words in the `FULLTEXT` columns. `FULLTEXT` indexes are case-insensitive, so it makes no difference what case is used in your queries.

Assuming that you have added a `FULLTEXT` index to the *author* and *title* columns, enter the three queries shown in Example 8-24. The first asks for any of these columns that contain the word *and* to be returned. Because *and* is a stopword, MySQL will ignore it and the query will always produce an empty set—no matter what is stored in the columns. The second query asks for any rows that contain both of the words *old* and *shop* anywhere in them, in any order, to be returned. And the last query applies the same kind of search for the words *tom* and *sawyer*. Figure 8-13 shows the results of these queries.

Example 8-24. Using MATCH ... AGAINST on FULLTEXT indexes

```
SELECT author,title FROM classics
 WHERE MATCH(author,title) AGAINST('and');
SELECT author,title FROM classics
 WHERE MATCH(author,title) AGAINST('old shop');
SELECT author,title FROM classics
 WHERE MATCH(author,title) AGAINST('tom sawyer');
```

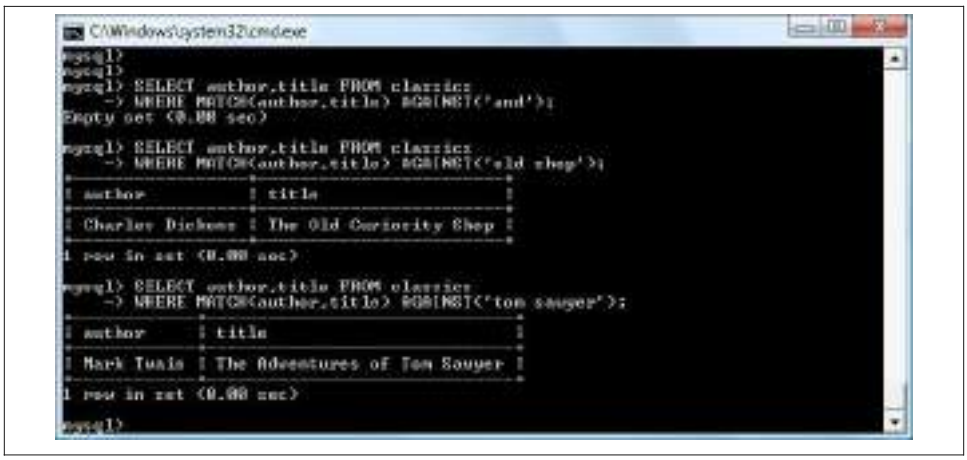


Figure 8-13. Using MATCH ... AGAINST on a FULLTEXT index

MATCH ... AGAINST ... IN BOOLEAN MODE

If you wish to give your `MATCH ... AGAINST` queries even more power, use Boolean mode. This changes the effect of the standard `FULLTEXT` query so that it searches for any combination of search words, instead of requiring all search words to be in the text. The presence of a single word in a column causes the search to return the row.

Boolean mode also allows you to preface search words with a + or − sign to indicate whether they must be included or excluded. If normal Boolean mode says, "Any of these words will do," a plus sign means "This word must be present; otherwise, don't return the row." A minus sign means "This word must not be present; its presence disqualifies the row from being returned."

Example 8-25 illustrates Boolean mode through two queries. The first asks for all rows containing the word *charles* and not the word *species* to be returned. The second uses double quotes to request that all rows containing the exact phrase *origin of* be returned. Figure 8-14 shows the results of these queries.

Example 8-25. Using MATCH ... AGAINST ... IN BOOLEAN MODE

```
SELECT author,title FROM classics
 WHERE MATCH(author,title)
 AGAINST('+charles -species' IN BOOLEAN MODE);
SELECT author,title FROM classics
 WHERE MATCH(author,title)
```

```
AGAINST('"origin of"' IN BOOLEAN MODE);
```

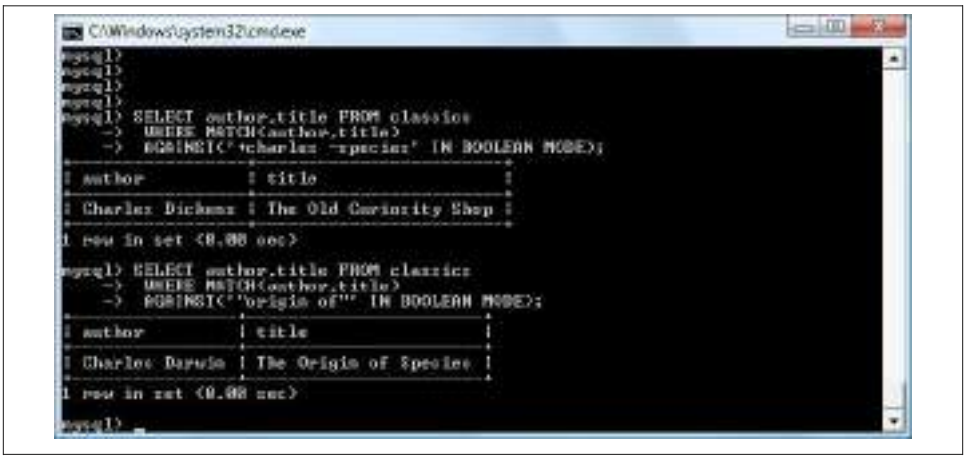


Figure 8-14. Using MATCH ... AGAINST ... IN BOOLEAN MODE

As you would expect, the first request returns only `The Old Curiosity Shop` by Charles Dickens, because any rows containing the word *species* have been excluded, so Charles Darwin's publication is ignored.

There is something of interest to note in the second query: the stopword *of* is part of the search string, but is still used by the search because the double quotation marks override stopwords.

UPDATE ... SET

This construct allows you to update the contents of a field. If you wish to change the contents of one or more fields, you need to first narrow in on just the field or fields to be changed, in much the same way you use the `SELECT` command. Example 8-26 shows the use of `UPDATE ... SET` in two different ways. You can see the results in Figure 8-15.

Example 8-26. Using UPDATE ... SET

```
UPDATE classics SET author='Mark Twain (Samuel Langhorne
Clemens)'
 WHERE author='Mark Twain';
UPDATE classics SET category='Classic Fiction'
 WHERE category='Fiction';
```

Figure 8-15. Updating columns in the classics table

In the first query, Mark Twain's real name of Samuel Langhorne Clemens was appended to his pen name in brackets, which affected only one row. The second query, however, affected three rows, because it changed all occurrences of the word *Fiction* in the *category* column to the term *Classic Fiction*.

When performing an update, you can also make use of the qualifiers you have already seen, such as `LIMIT`, and the following `ORDER BY` and `GROUP BY` keywords.

ORDER BY

`ORDER BY` sorts returned results by one or more columns in ascending or descending order. Example 8-27 shows two such queries, the results of which can be seen in Figure 8-16.

Example 8-27. Using ORDER BY

```
SELECT author,title FROM classics ORDER BY author;
SELECT author,title FROM classics ORDER BY title DESC;
```

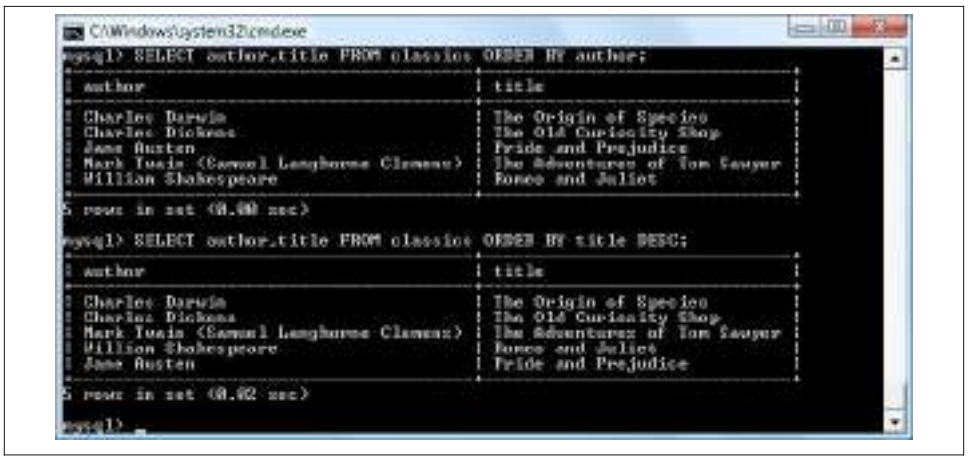


Figure 8-16. Sorting the results of requests

As you can see, the first query returns the publications by *author* in ascending alphabetical order (the default), and the second returns them by *title* in descending order.

If you wanted to sort all the rows by *author* and then by descending *year* of publication (to view the most recent first), you would issue the following query:

```
SELECT author,title,year FROM classics ORDER BY
author,year DESC;
```

This shows that each ascending and descending qualifier applies to a single column. The `DESC` keyword applies only to the preceding column, *year*. Because you allow *author* to use the default sort order, it is sorted in ascending order. You could also have explicitly specified ascending order for that column, with the same results:

```
SELECT author,title,year FROM classics ORDER BY author ASC,year
DESC;
```

GROUP BY

In a similar fashion to `ORDER BY`, you can group results returned from queries using `GROUP BY`, which is good for retrieving information about a group of data. For example, if you want to know how many publications there are of each category in the *classics* table, you can issue the following query:

```
SELECT category,COUNT(author) FROM classics GROUP BY
category;
```

which returns the following output:

```
+----------------+--------------+
| category       | COUNT(author) |
```

```
+----------------+--------------+
| Classic Fiction |            3 |
| Non-Fiction     |            1 |
| Play            |            1 |
+----------------+--------------+
3 rows in set (0.00 sec)
```

Joining Tables Together

It is quite normal to maintain multiple tables within a database, each holding a different type of information. For example, consider the case of a *customers* table that needs to be able to be cross-referenced with publications purchased from the *classics* table. Enter the commands in Example 8-28 to create this new table and populate it with three customers and their purchases. Figure 8-17 shows the result. *Example 8-28. Creating and populating the customers table*

```
CREATE TABLE
customers (  name
VARCHAR(128),  isbn
VARCHAR(13),
 PRIMARY KEY (isbn)) ENGINE MyISAM;
INSERT INTO customers(name,isbn)
 VALUES('Joe Bloggs','9780099533474');
INSERT INTO customers(name,isbn)
 VALUES('Mary Smith','9780582506206');
INSERT INTO customers(name,isbn)
 VALUES('Jack Wilson','9780517123201');
SELECT * FROM customers;
```

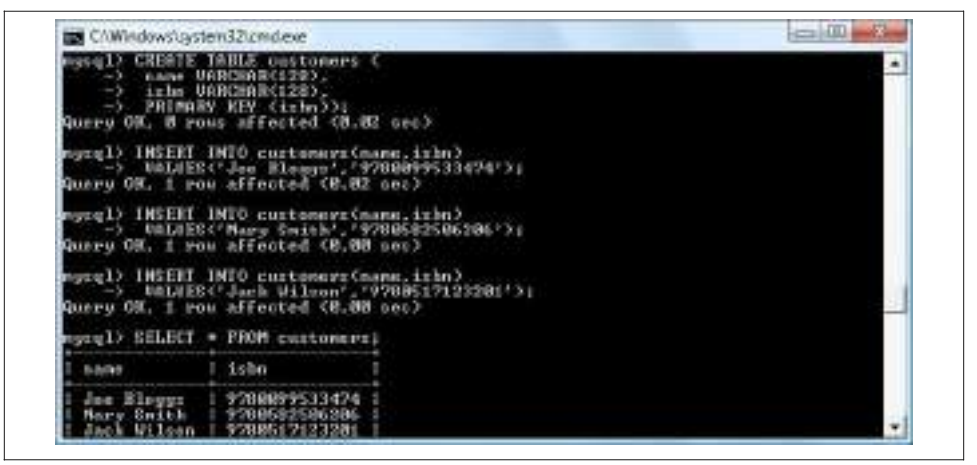


Figure 8-17. Creating the customers table

There's also a shortcut for inserting multiple rows of data, as in Example 8-28, in which you can replace the three separate `INSERT INTO` queries with a single one listing the data to be inserted, separated by commas, like this:

```
INSERT INTO customers(name,isbn) VALUES
('Joe Bloggs','9780099533474'),
('Mary Smith','9780582506206'),
('Jack Wilson','9780517123201');
```

Of course, in a proper table containing customers' details there would also be addresses, phone numbers, email addresses, and so on, but they aren't necessary for this explanation. While creating the new table, you should have noticed that it has something in common with the *classics* table: a column called *isbn*. Because it has the same meaning in both tables (an ISBN refers to a book, and always the same book), we can use this column to tie the two tables together into a single query, as in Example 8-29.

Example 8-29. Joining two tables into a single SELECT

```
SELECT name,author,title from customers,classics
 WHERE customers.isbn=classics.isbn;
```

The result of this operation is the following:

```
+-------------+-----------------+-----------------------+
| name        | author          | title                 |
+-------------+-----------------+-----------------------+
| Joe Bloggs  | Charles Dickens | The Old Curiosity Shop |
| Mary Smith  | Jane Austen     | Pride and Prejudice    |
| Jack Wilson | Charles Darwin  | The Origin of Species  |
+-------------+-----------------+-----------------------+
3 rows in set (0.00 sec)
```

See how this query has neatly tied both tables together to show the publications purchased from the *classics* table by the people in the *customers* table?

NATURAL JOIN

Using `NATURAL JOIN`, you can save yourself some typing and make queries a little clearer. This kind of join takes two tables and automatically joins columns that have the same name. So, to achieve the same results as from Example 8-29, you would enter:

```
SELECT name,author,title FROM customers NATURAL JOIN
classics;
```

JOIN...ON

If you wish to specify the column on which to join two tables, use the `JOIN ... ON` construct, as follows, to achieve results identical to those of Example 8-29:

```
SELECT name,author,title FROM customers
 JOIN classics ON customers.isbn=classics.isbn;
```

Using AS

You can also save yourself some typing and improve query readability by creating aliases using the AS keyword. Follow a table name with AS and the alias to use. The following code, therefore, is also identical in action to Example 8-29. Aliases can be particularly useful when you have long queries that reference the same table names many times.

```
SELECT name,author,title from
customers AS cust, classics AS
class   WHERE
cust.isbn=class.isbn;
```

The result of this operation is the following:

```
+-------------+-----------------+------------------------+
| name        | author          | title                  |
+-------------+-----------------+------------------------+
| Joe Bloggs  | Charles Dickens | The Old Curiosity Shop |
| Mary Smith  | Jane Austen     | Pride and Prejudice    |
| Jack Wilson | Charles Darwin  | The Origin of Species  |
+-------------+-----------------+------------------------+
3 rows in set (0.00 sec)
```

Using Logical Operators

You can also use the logical operators AND, OR, and NOT in your MySQL WHERE queries to further narrow down your selections. Example 8-30 shows one instance of each, but you can mix and match them in any way you need.

Example 8-30. Using logical operators

```
SELECT author,title FROM classics WHERE
 author LIKE "Charles%" AND author LIKE "%Darwin";
SELECT author,title FROM classics WHERE
 author LIKE "%Mark Twain%" OR author LIKE "%Samuel Langhorne
Clemens%";
SELECT author,title FROM classics WHERE
author LIKE "Charles%" AND author NOT LIKE
"%Darwin";
```

I've chosen the first query, because Charles Darwin might be listed in some rows by his full name, *Charles Robert Darwin*. Thus, the query returns publications as long as the *author* column starts with *Charles* and ends with *Darwin*. The second query searches for publications written using either *Mark Twain*'s pen name or his real name, *Samuel Langhorne Clemens*. The third query returns publications written by authors with the first name *Charles* but not the surname *Darwin*.

MySQL Functions

You might wonder why anyone would want to use MySQL functions when PHP comes with a whole bunch of powerful functions of its own. The answer is very simple: the MySQL functions work on the data right there in the database. If you were to use PHP,

you would first have to extract raw data from MySQL, manipulate it, and then perform the database query you first wanted.

Having functions built into MySQL substantially reduces the time needed for performing complex queries, as well as their complexity. If you wish to learn more about the available string and date/time functions, you can visit the following URLs:

- *http://tinyurl.com/mysqlstrings*
- *http://tinyurl.com/mysqldates*

However, to get you started, Appendix D describes a subset containing the most useful of these functions.

Accessing MySQL via phpMyAdmin

Although to use MySQL you have to learn these main commands and how they work, once you understand them, it can be much quicker and simpler to use a program such as *phpMyAdmin* to manage your databases and tables.

However, you will need to install phpMyAdmin before you can use it. To do this, call up the Zend UI by entering the following into your browser, and log in (as shown in Figure 8-18):

```
http://localhost:10081/ZendServer/
```

Figure 8-18. The Zend Dashboard

Now click on the left and right arrows to the right of the DEPLOY SAMPLE APPS section until you see the phpMyAdmin logo and click it to initiate the download; then click Next when you're finished. Click Next again, after you have viewed the README information, to call up the Application Details screen (see Figure 8-19).

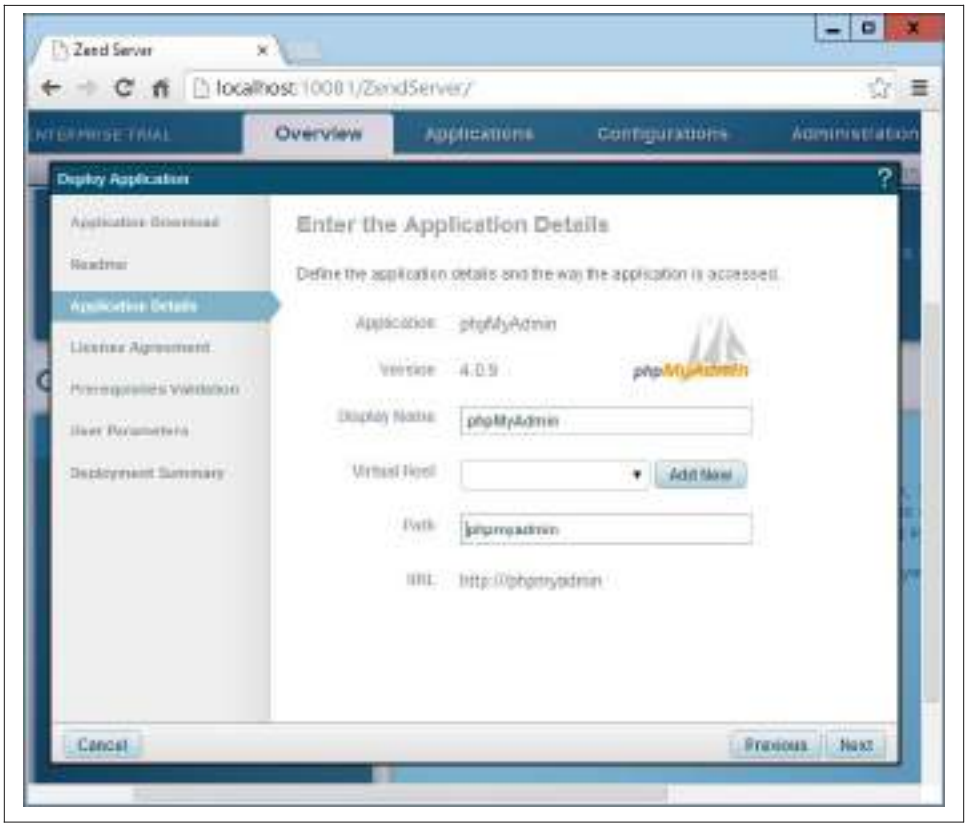


Figure 8-19. Configuring phpMyAdmin for Zend

Here you should probably accept the defaults for Display Name and Virtual Host, but will need to specify a directory name for phpMyAdmin in order to keep it away from your document root files. I have entered the name ***phpmyadmin*** (all in lowercase so that I won't have to enter any capital letters whenever I type the URL to call it up). Continue clicking Next and accepting any license agreements until you get to the screen in Figure 8-20. Here you should select the "Use HTTP (Apache) Basic Authentication?" checkbox and supply a login and password. The default login is ***DBadmin***, but I have chosen to use simply ***admin***; your login and password are up to you. Unless you have configured them differently, you can probably leave the IP, Port, Database User, and Password as displayed.

Accessing MySQL via phpMyAdmin

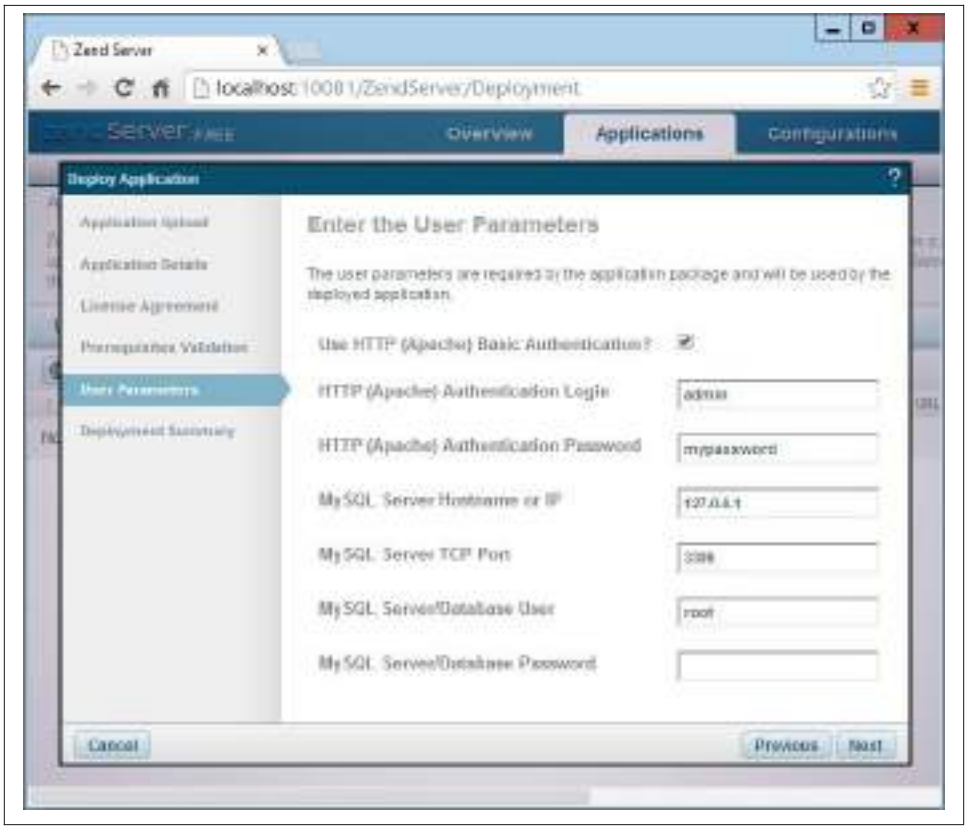


Figure 8-20. Entering phpMyAdmin user parameters

Now click Next, review the summary displayed, and when ready, click the Deploy button. After a few seconds, you should see that the application was successfully deployed, at which point you'll be ready to access phpMyAdmin by entering the following into your browser:

```
http://localhost/phpmyadmin
```

This will bring up the dialog shown in Figure 8-21, where you should enter your username and password before clicking the Log In button.

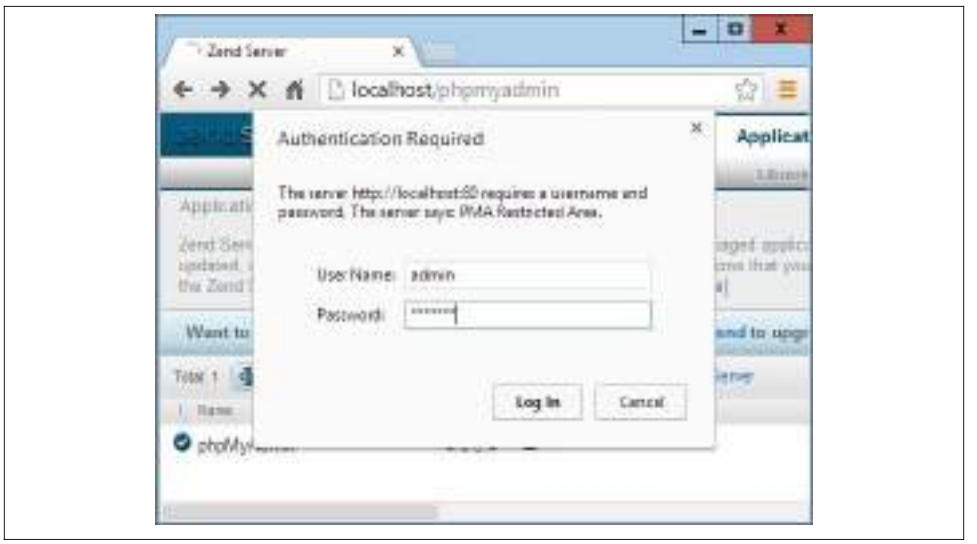


Figure 8-21. Logging into phpMyAdmin

Your browser should now look like Figure 8-22, and you're ready to use phpMyAdmin in place of the MySQL command line.

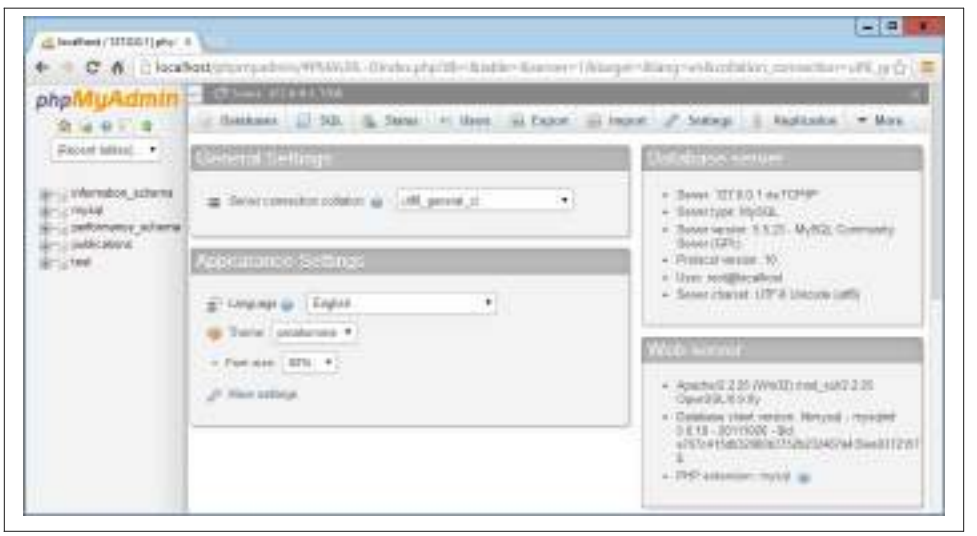


Figure 8-22. The phpMyAdmin main screen

Full details on this installation process are on the Zend website at the following (shortened) URL: *http://tinyurl.com/installpma*.

Using phpMyAdmin

In the lefthand pane of the main phpMyAdmin screen, you can click on the drop-down menu that says "(Databases)" to select any database you wish to work with. This will open the database and display its tables.

From here you can perform all the main operations, such as creating new databases, adding tables, creating indexes, and much more. To read the supporting documentation for phpMyAdmin, visit *https://docs.phpmyadmin.net*.

If you worked with me through the examples in this chapter, congratulations—it's been quite a long journey. You've come all the way from learning how to create a MySQL database through issuing complex queries that combine multiple tables, to using Boolean operators and leveraging MySQL's various qualifiers.

In the next chapter, we'll start looking at how to approach efficient database design, advanced SQL techniques, and MySQL functions and transactions.

Questions

1. What is the purpose of the semicolon in MySQL queries?
2. Which command would you use to view the available databases or tables?
3. How would you create a new MySQL user on the local host called *newuser* with a password of *newpass* and with access to everything in the database *newdatabase*?
4. How can you view the structure of a table?
5. What is the purpose of a MySQL index?
6. What benefit does a `FULLTEXT` index provide?
7. What is a stopword?
8. Both `SELECT DISTINCT` and `GROUP BY` cause the display to show only one output row for each value in a column, even if multiple rows contain that value. What are the main differences between `SELECT DISTINCT` and `GROUP BY`?

CHAPTER

10 Accessing MySQL Using PHP

If you worked through the previous chapters, you're proficient in using both MySQL and PHP. In this chapter, you will learn how to integrate the two by using PHP's builtin functions to access MySQL.

Querying a MySQL Database with PHP

The reason for using PHP as an interface to MySQL is to format the results of SQL queries in a form visible in a web page. As long as you can log into your MySQL installation using your username and password, you can also do so from PHP. However, instead of using MySQL's command line to enter instructions and view output, you will create query strings that are passed to MySQL. When MySQL returns its response, it will come as a data structure that PHP can recognize instead of the formatted output you see when you work on the command line. Further PHP commands can retrieve the data and format it for the web page.

To get you started, in this chapter I use the standard, procedural `mysql` function calls, so that you'll be up and running quickly, and able to maintain older PHP code. However, the new object-oriented `mysqli` functions (the `i` stands for improved) are becoming the recommended way to interface with MySQL from PHP, so in the following chapter I'll show you how to use these too (or instead, because the old functions have become deprecated and could be removed from PHP at some point).

The Process

The process of using MySQL with PHP is:

1. Connect to MySQL.
2. Select the database to use.
3. Build a query string.
4. Perform the query.
5. Retrieve the results and output them to a web page.
6. Repeat Steps 3 to 5 until all desired data has been retrieved.
7. Disconnect from MySQL.

We'll work through these sections in turn, but first it's important to set up your login details in a secure manner so people snooping around on your system have trouble getting access to your database.

Creating a Login File

Most websites developed with PHP contain multiple program files that will require access to MySQL and will thus need the login and password details. Therefore, it's sensible to create a single file to store these and then include that file wherever it's needed. Example 10-1 shows such a file, which I've called *login.php*. Type the example, replacing placeholder values (such as *username*) with the actual values you use for your MySQL database, and save it to the web development directory you set up in Chapter 2. We'll be making use of the file shortly. The hostname `localhost` should work as long as you're using a MySQL database on your local system, and the database *publications* should work if you're typing the examples I've used so far.

Example 10-1. The login.php file

```
<?php // login.php
  $db_hostname = 'localhost';
  $db_database = 'publications';
  $db_username = 'username';
  $db_password = 'password';
?>
```

The enclosing `<?php` and `?>` tags are especially important for the *login.php* file in Example 10-1, because they mean that the lines between can be interpreted *only* as PHP code. If you were to leave them out and someone were to call up the file directly from your website, it would display as text and reveal your secrets. But, with the tags in place, all that person will see is a blank page. The file will correctly include in your other PHP files.

The `$db_hostname` variable will tell PHP which computer to use when connecting to a database. This is required, because you can access MySQL databases on any computer connected to your PHP installation, and that potentially includes any host anywhere on the Web. However, the examples in this chapter will be working on the

local server. So, in place of specifying a domain such as *mysql.myserver.com*, you can just use the word `localhost` (or the IP address 127.0.0.1).

The database we'll be using, `$db_database`, is the one called *publications*, which you probably created in Chapter 8, or the one you were provided with by your server administrator (in which case you have to modify *login.php* accordingly).

The variables `$db_username` and `$db_password` should be set to the username and password that you have been using with MySQL.

Another benefit of keeping these login details in a single place is that you can change your password as frequently as you like and there will be only one file to update when you do, no matter how many PHP files access MySQL.

Connecting to MySQL

Now that you have the *login.php* file saved, you can include it in any PHP files that will need to access the database by using the `require_once` statement. This is preferable to an `include` statement, as it will generate a fatal error if the file is not found. And believe me, not finding the file containing the login details to your database *is* a fatal error.

Also, using `require_once` instead of `require` means that the file will be read in only when it has not previously been included, which prevents wasteful duplicate disk accesses. Example 10-2 shows the code to use.

Example 10-2. Connecting to a MySQL server

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);

  if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error()); ?>
```

This example runs PHP's `mysql_connect` function, which requires three parameters: the *hostname*, *username*, and *password* of a MySQL server. Upon success it returns an *identifier* to the server; otherwise, `FALSE` is returned. Notice that the second line uses an `if` statement with the `die` function, which does what it sounds like and quits from PHP with an error message if `$db_server` is not `TRUE`.

The `die` message explains that it was not possible to connect to the MySQL database, and—to help identify why this happened—includes a call to the `mysql_error` function. This function outputs the error text from the last called MySQL function.

The database server pointer `$db_server` will be used in some of the following examples to identify the MySQL server to be queried. By using identifiers this way, we can connect to and access multiple MySQL servers from a single PHP program.

The `die` function is great for when you are developing PHP code, but of course you will want more user-friendly error messages on a production server. In this case you won't abort your PHP program, but format a message that will be displayed when the program exits normally, such as:

```
  function mysql_fatal_error($msg)
  {
      $msg2 =
mysql_error();       echo
<<< _END
   We are sorry, but it was not possible to
  complete   the requested task. The error
             message we got was:

      <p>$msg: $msg2</p>

  Please click the back button on your
browser   and try again. If you are still
having problems,   please <a
href="mailto:admin@server.com">email   our
administrator</a>. Thank you.
  _END;
}
```

Selecting a database

Having successfully connected to MySQL, you are now ready to select the database that you will be using. Example 10-3 shows how to do this.

Example 10-3. Selecting a database

```
<?php
  mysql_select_db($db_database)
    or die("Unable to select database: " .
mysql_error()); ?>
```

The command to select the database is `mysql_select_db`. Pass it the name of the database you want and the server to which you connected. As with the previous example, a `die` statement has been included to provide an error message and explanation, should the selection fail—the only difference being that there is no need to retain the return value from the `mysql_select_db` function, as it simply returns either `TRUE` or `FALSE`. Therefore the PHP `or` statement was used, which means "if the previous command failed, do the following." Note that for the `or` to work, there must be no semicolon at the end of the first line of code.

Building and executing a query

Sending a query to MySQL from PHP is as simple as issuing it using the `mysql_query` function. Example 10-4 shows you how to use it. *Example 10-4.*

Querying a database

```
<?php
  $query  = "SELECT * FROM classics";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

First, the variable `$query` is set to the query to be made. In this case, it is asking to see all rows in the table *classics*. Note that, unlike with MySQL's command line, no semicolon is required at the tail of the query, because the `mysql_query` function is used to issue a complete query; it cannot be used for queries sent in multiple parts, one at a time. Therefore, MySQL knows the query is complete and doesn't look for a semicolon.

This function returns a result that we place in the variable `$result`. Having used MySQL at the command line, you might think that the contents of `$result` will be the same as the result returned from a command-line query, with horizontal and vertical lines, and so on. However, this is not the case with the result returned to PHP. Instead, upon success, `$result` will contain a *resource* that can be used to extract the results of the query. You'll see how to extract the data in the next section. Upon failure, `$result` contains `FALSE`. So the example finishes by checking `$result`. If it's `FALSE`, it means that there was an error, and the `die` command is executed.

Fetching a result

Once you have a resource returned from a `mysql_query` function, you can use it to retrieve the data you want. The simplest way to do this is to fetch the cells you want, one at a time, using the `mysql_result` function. Example 10-5 combines and extends the previous examples into a program that you can type and run yourself to retrieve the returned results. I suggest that you save it in the same folder as *login.php* and give it the name *query.php*.

Example 10-5. Fetching results one cell at a time

```
<?php // query.php   require_once 'login.php';   $db_server
= mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());
```

```
  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());
  $query  = "SELECT * FROM classics";
  $result = mysql_query($query);   if (!$result) die

("Database access failed: " . mysql_error());   $rows

= mysql_num_rows($result);

  for ($j = 0 ; $j < $rows ; ++$j)
  {
    echo 'Author: '   . mysql_result($result,$j,'author')
. '<br>';     echo 'Title: '    .
mysql_result($result,$j,'title')    . '<br>';     echo
'Category: ' . mysql_result($result,$j,'category') . '<br>';
echo 'Year: '     . mysql_result($result,$j,'year')     .
'<br>';     echo 'ISBN: '     .
mysql_result($result,$j,'isbn')     . '<br><br>';
  }
?>
```

The final 10 lines of code are the new ones, so let's look at them. They start by setting the variable `$rows` to the value returned by a call to `mysql_num_rows`. This function reports the number of rows returned by a query.

Armed with the row count, we enter a `for` loop that extracts each cell of data from each row using the `mysql_result` function. The parameters supplied to this function are the resource `$result`, which was returned by `mysql_query`, the row number `$j`, and the name of the column from which to extract the data.

The results from each call to `mysql_result` are then incorporated within `echo` statements to display one field per line, with an additional line feed between rows. Figure 10-1 shows the result of running this program.

As you may recall, we populated the *classics* table with five rows in Chapter 8, and indeed, five rows of data are returned by *query.php*. But, as it stands, this code is actually extremely inefficient and slow, because a total of 25 calls are made to the function `mysql_result` in order to retrieve all the data, a single cell at a time. Luckily, there is a much better way of retrieving the data, which is getting a single row at a time using the `mysql_fetch_row` function.

In Chapter 9, I talked about First, Second, and Third Normal Form, so you may have now noticed that the *classics* table doesn't satisfy these, because both author and book details are included within the same table. That's because we created this table before encountering normalization. However, for the purposes of illustrating access to

MySQL from PHP, reusing this table avoids the hassle of typing in a new set of test data, so we'll stick with it for the time being.

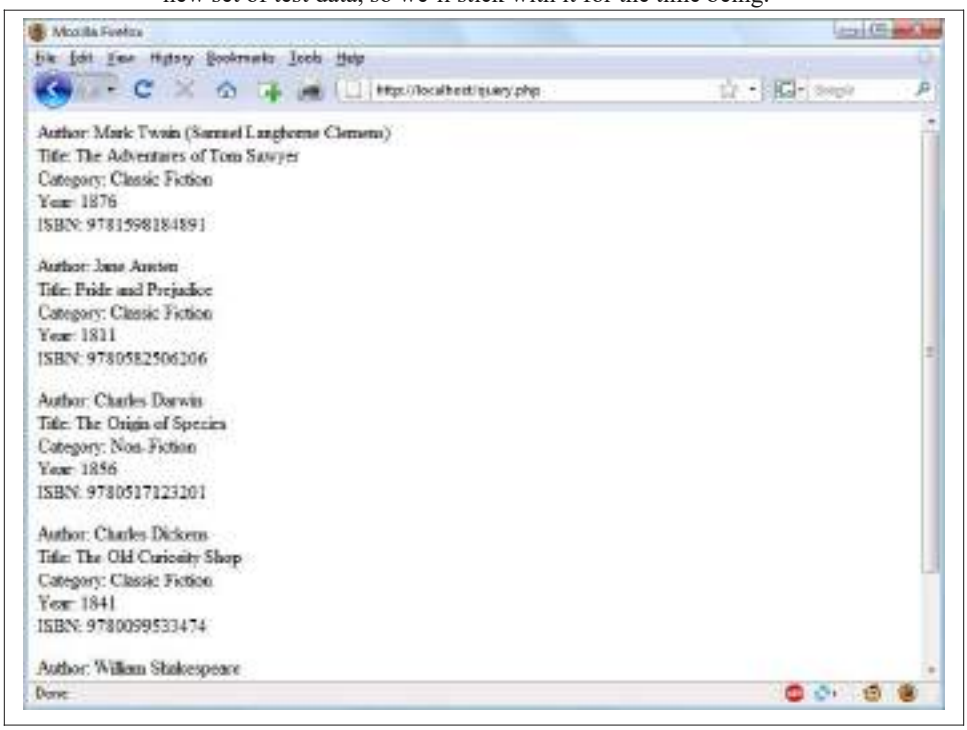


Figure 10-1. The output from the query.php program in Example 10-5

Fetching a row

It was important to show how you can fetch a single cell of data from MySQL, but now let's look at a much more efficient method. Replace the `for` loop of *query.php* (in Example 10-5) with the new loop in Example 10-6, and you will find that you get exactly the same result that was displayed in Figure 10-1.

Example 10-6. Replacement for loop for fetching results one row at a time

```
<?php
  for ($j = 0 ; $j < $rows ; ++$j)
  {
    $row = mysql_fetch_row($result);

    echo 'Author: ' .     $row[0] .
'<br>';     echo 'Title: ' .
$row[1] . '<br>';     echo 'Category: '
.   $row[2] . '<br>';     echo 'Year: '
.       $row[3] . '<br>';     echo
'ISBN: ' .       $row[4] . '<br><br>';
```

```
  }
?>
```

In this modified code, only one-fifth of the calls are made to a MySQL-calling function (a full 80% less), because each row is fetched in its entirety via the `mysql_fetch_row` function. This returns a single row of data in an array, which is then assigned to the variable `$row`.

All that's necessary, then, is to reference each element of the array `$row` in turn (starting at an offset of 0). Therefore `$row[0]` contains the *Author* data, `$row[1]` the *Title*, and so on, because each column is placed in the array in the order in which it appears in the MySQL table. Also, by using `mysql_fetch_row` instead of `mysql_result`, you use substantially less PHP code and achieve much faster execution time, due to simply referencing each item of data by offset rather than by name.

Closing a connection

When you have finished using a database, you should close the connection. You do so by issuing the command in Example 10-7.

Example 10-7. Closing a MySQL server connection

```
<?php

mysql_close($db_server)
; ?>
```

We have to pass the identifier returned by `mysql_connect` back in Example 10-2, which we stored in the variable `$db_server`.

All database connections are automatically closed when PHP exits, so it doesn't matter that the connection wasn't closed in Example 10-5. But in longer programs, where you may continually open and close database connections, you are strongly advised to close each one as soon as you're finished accessing it.

A Practical Example

It's time to write our first example of inserting data in and deleting it from a MySQL table using PHP. I recommend that you type Example 10-8 and save it to your web development directory using the filename *sqltest.php*. You can see an example of the program's output in Figure 10-2.

Example 10-8 creates a standard HTML form. Chapter 12 explains forms in detail, but in this chapter I take form handling for granted and just deal with database interaction.

Example 10-8. Inserting and deleting using sqltest.php

```
<?php // sqltest.php   require_once 'login.php';
$db_server = mysql_connect($db_hostname, $db_username,
$db_password);   if (!$db_server) die("Unable to connect to
MySQL: " . mysql_error());

  mysql_select_db($db_database, $db_server)
or die("Unable to select database: " .
mysql_error());

  if (isset($_POST['delete']) && isset($_POST['isbn']))
  {
    $isbn  = get_post('isbn');
    $query = "DELETE FROM classics WHERE isbn='$isbn'";

    if (!mysql_query($query,
$db_server))       echo "DELETE
failed: $query<br>" .
      mysql_error() .
"<br><br>";   }

  if (isset($_POST['author']) &&
isset($_POST['title']) &&
isset($_POST['category']) &&
isset($_POST['year']) &&
isset($_POST['isbn']))
  {
    $author   = get_post('author');
    $title    = get_post('title');
    $category = get_post('category');
    $year     = get_post('year');
    $isbn     = get_post('isbn');

    $query = "INSERT INTO classics VALUES" .
      "('$author', '$title', '$category', '$year',
'$isbn')";

    if (!mysql_query($query,
$db_server))       echo "INSERT
failed: $query<br>" .
      mysql_error() .
"<br><br>";   }

  echo <<<_END
  <form action="sqltest.php" method="post"><pre>
    Author <input type="text" name="author">
     Title <input type="text" name="title">
  Category <input type="text" name="category">
      Year <input type="text" name="year">
```

```
      ISBN <input type="text" name="isbn">
           <input type="submit" value="ADD RECORD">
  </pre></form>
_END;

  $query  = "SELECT * FROM classics";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error());   $rows = mysql_num_rows($result);

  for ($j = 0 ; $j < $rows ; ++$j)
  {
    $row = mysql_fetch_row($result);
    echo <<<_END
  <pre>
    Author $row[0]
     Title $row[1]
  Category $row[2]
      Year $row[3]
      ISBN $row[4]
  </pre>
  <form action="sqltest.php" method="post">
  <input type="hidden" name="delete" value="yes">
  <input type="hidden" name="isbn"
value="$row[4]">   <input type="submit"
value="DELETE RECORD"></form>
_END;
  }

mysql_close($db_server

);

  function get_post($var)
  {
    return mysql_real_escape_string($_POST[$var]);
  }
?>
```

At over 80 lines of code, this program may appear daunting, but don't worry—you've already covered many of them in Example 10-5, and what the code does is actually quite simple.

It first checks for any inputs that may have been made and then either inserts new data into the table *classics* of the *publications* database or deletes a row from it, according to the input supplied. Regardless of whether there was input, the program then outputs all rows in the table to the browser. So let's see how it works.

The first section of new code starts by using the `isset` function to check whether values for all the fields have been posted to the program. Upon confirmation, each of the lines within the `if` statement calls the function `get_post`, which appears at the end of the program. This function has one small but critical job: fetching the input from the browser.

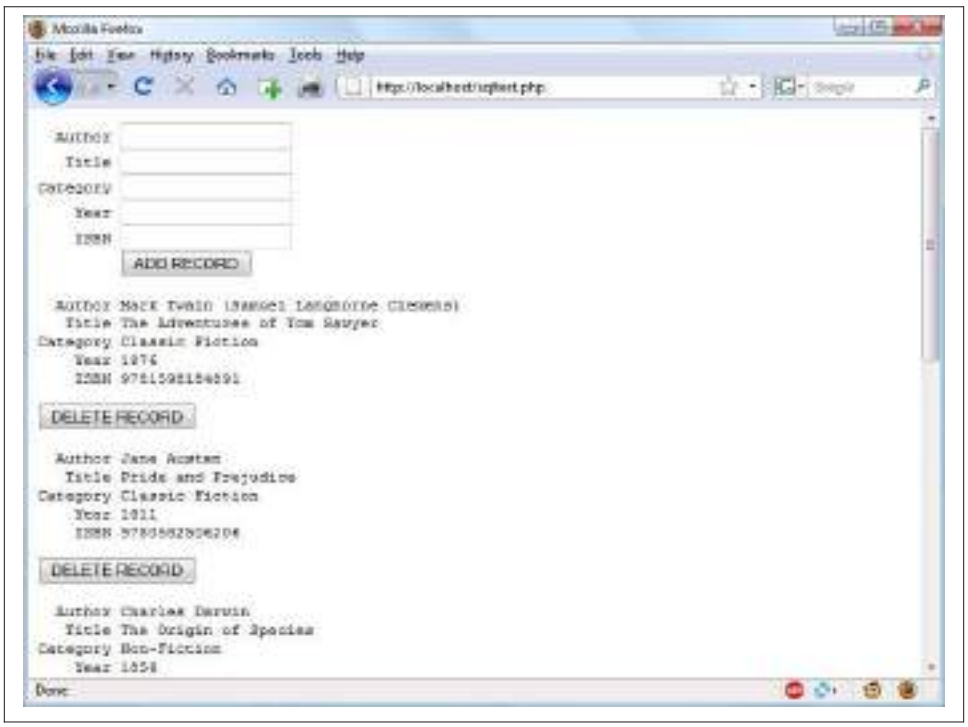


Figure 10-2. The output from Example 10-8, sqltest.php

The $_POST Array

I mentioned in an earlier chapter that a browser sends user input through either a `GET` request or a `POST` request. The `POST` request is usually preferred, and we use it here. The web server bundles up all of the user input (even if the form was filled out with a hundred fields) and puts in into an array named `$_POST`.

`$_POST` is an associative array, which you encountered in Chapter 6. Depending on whether a form has been set to use the `POST` or the `GET` method, either the `$_POST` or the `$_GET` associative array will be populated with the form data. They can both be read in exactly the same way.

Each field has an element in the array named after that field. So, if a form contained a field named `isbn`, the `$_POST` array contains an element keyed by the word `isbn`. The PHP program can read that field by referring to either `$_POST['isbn']` or `$_POST["isbn"]` (single and double quotes have the same effect in this case). If the `$_POST` syntax still seems complex to you, rest assured that you can just use the convention I've shown in Example 10-8, copy the user's input to other variables, and forget about `$_POST` after that. This is normal in PHP programs: they retrieve all the fields from `$_POST` at the beginning of the program and then ignore it.

There is no reason to write to an element in the `$_POST` array. Its only purpose is to communicate information from the browser to the program, and you're better off copying data to your own variables before altering it.

So, back to the `get_post` function: it passes each item it retrieves through the `mysql_real_escape_string` function to strip out any characters that a hacker may have inserted in order to break into or alter your database.

Deleting a Record

Prior to checking whether new data has been posted, the program checks whether the variable `$_POST['delete']` has a value. If so, the user has clicked on the DELETE RECORD button to erase a record. In this case, the value of `$isbn` will also have been posted.

As you'll recall, the ISBN uniquely identifies each record. The HTML form appends the ISBN to the `DELETE FROM` query string created in the variable `$query`, which is then passed to the `mysql_query` function to issue it to MySQL. `mysql_query` returns either `TRUE` or `FALSE`, and `FALSE` causes an error message to be displayed explaining what went wrong.

If `$_POST['delete'])` is not set (and there is therefore no record to be deleted), `$_POST['author'])` and other posted values are checked. If they have all been given values, then `$query` is set to an `INSERT INTO` command, followed by the five values to be inserted. The variable is then passed to `mysql_query`, which upon completion returns either `TRUE` or `FALSE`. If `FALSE` is returned, an error message is displayed.

Displaying the Form

Next we get to the part of code that displays the little form at the top of Figure 10-2. You should recall the `echo <<<_END` structure from previous chapters, which outputs everything between the `_END` tags.

Instead of the `echo` command, the program could also drop out of PHP using `?>`, issue the HTML, and then reenter PHP processing with `<?php`. Whichever style used is a matter of programmer preference, but I always recommend staying within PHP code for these reasons:

- It makes it very clear when debugging (and also for other users) that everything within a *.php* file is PHP code. Therefore, there is no need to go hunting for dropouts to HTML.
- When you wish to include a PHP variable directly within HTML, you can just type it. If you had dropped back to HTML, you would have had to temporarily reenter PHP processing, output the variable, and then drop back out again.

The HTML form section simply sets the form's action to *sqltest.php*. This means that when the form is submitted, the contents of the form fields will be sent to the file *sqltest.php*, which is the program itself. The form is also set up to send the fields as a `POST` rather than a `GET` request. This is because `GET` requests are appended to the URL being submitted to and can look messy in your browser. They also allow users to easily modify submissions and try to hack your server. Therefore, whenever possible, you should use `POST` submissions, which also have the benefit of hiding the posted data from view.

Having output the form fields, the HTML displays a Submit button with the name ADD RECORD and closes the form. Note the use of the `<pre>` and `</pre>` tags here, which have been used to force a monospaced font and allow all the inputs to line up neatly. The carriage returns at the end of each line are also output when inside `<pre>` tags.

Querying the Database

Next, the code returns to the familiar territory of Example 10-5 where, in the following four lines of code, a query is sent to MySQL asking to see all the records in the *classics* table. After that, `$rows` is set to a value representing the number of rows in the table and a `for` loop is entered to display the contents of each row.

I have altered the next bit of code to simplify things. Instead of using the `<br>` tags for line feeds in Example 10-5, I have chosen to use a `<pre>` tag to line up the display of each record in a pleasing manner.

After the display of each record, there is a second form that also posts to *sqltest.php* (the program itself) but this time contains two hidden fields: `delete` and `isbn`. The `delete` field is set to "yes" and `isbn` to the value held in `$row[4]`, which contains the ISBN for the record. Then a Submit button with the name DELETE RECORD is displayed and the form is closed. A curly brace then completes the `for` loop, which will continue until all records have been displayed.

Finally, you see the definition for the function `get_post`, which we've already looked at. And that's it—our first PHP program to manipulate a MySQL database. So, let's check out what it can do.

Once you have typed the program (and corrected any typing errors), try entering the following data into the various input fields to add a new record for the book *Moby Dick* to the database:

```
Herman Melville
Moby Dick
Fiction
1851
9780199535729
```

Running the Program

When you have submitted this data using the ADD RECORD button, scroll down to the bottom of the web page to see the new addition. It should look like Figure 10-3.

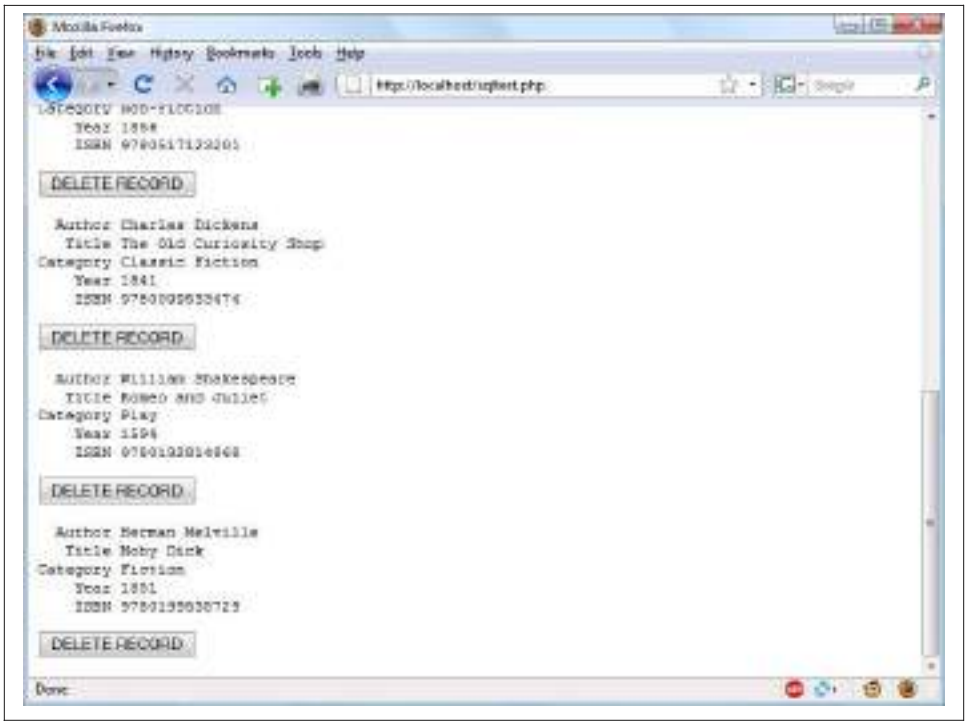


Figure 10-3. The result of adding Moby Dick to the database

Now let's look at how deleting a record works by creating a dummy record. So try entering just the number 1 in each of the five fields and click on the ADD RECORD button. If you now scroll down, you'll see a new record consisting just of 1s. Obviously this record isn't useful in this table, so now click on the DELETE RECORD button and scroll down again to confirm that the record has been deleted.

Assuming that everything worked, you are now able to add and delete records at will. Try doing this a few times, but leave the main records in place (including the new one for *Moby Dick*), as we'll be using them later. You could also try adding the record with all 1s again a couple of times and note the error message that you receive the second time, indicating that there is already an ISBN with the number 1.

Practical MySQL

You are now ready to look at some practical techniques that you can use in PHP to access the MySQL database, including tasks such as creating and dropping tables; inserting, updating, and deleting data; and protecting your database and website from malicious users. Note that the following examples assume that you've created the *login.php* program discussed earlier in this chapter.

Creating a Table

Let's assume you are working for a wildlife park and need to create a database to hold details about all the types of cats it houses. You are told that there are nine *families* of cats—Lion, Tiger, Jaguar, Leopard, Cougar, Cheetah, Lynx, Caracal, and Domestic—so you'll need a column for that. Then each cat has been given a *name*, so that's another column, and you also want to keep track of their *ages*, which is another. Of course, you will probably need more columns later, perhaps to hold dietary requirements, inoculations, and other details, but for now that's enough to get going. A unique identifier is also needed for each animal, so you also decide to create a column for that called *id*.

Example 10-9 shows the code you might use to create a MySQL table to hold this data, with the main query assignment in bold text.

Example 10-9. Creating a table called cats

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());
```

```
  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());
  $query = "CREATE TABLE cats (
id SMALLINT NOT NULL
AUTO_INCREMENT,     family
VARCHAR(32) NOT NULL,     name
VARCHAR(32) NOT NULL,     age
TINYINT NOT NULL,
    PRIMARY KEY (id)
  )";   $result =

mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

As you can see, the MySQL query looks pretty similar to how you would type it directly in the command line, except that there is no trailing semicolon, as none is needed when you are accessing MySQL from PHP.

Describing a Table

When you aren't logged into the MySQL command line, here's a handy piece of code that you can use to verify that a table has been correctly created from inside a browser. It simply issues the query `DESCRIBE cats` and then outputs an HTML table with four headings—*Column*, *Type*, *Null*, and *Key*—underneath which all columns within the table are shown. To use it with other tables, simply replace the name `cats` in the query with that of the new table (see Example 10-10).

Example 10-10. Describing the table cats

```
<?php
  require_once 'login.php';   $db_server =

mysql_connect($db_hostname, $db_username, $db_password);

if (!$db_server) die("Unable to connect to MySQL: " .

mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "DESCRIBE cats";
  $result = mysql_query($query);   if (!$result) die ("Database

access failed: " . mysql_error());   $rows =

mysql_num_rows($result);   echo

"<table><tr><th>Column</th><th>Type</th><th>Null</th><th>Key</th>

</tr>";
```

```
  for ($j = 0 ; $j < $rows ; ++$j)
  {
    $row = mysql_fetch_row($result);
    echo "<tr>";
    for ($k = 0 ; $k < 4 ;
++$k)       echo
"<td>$row[$k]</td>";

    echo
"</tr>";   }

  echo
"</table>"; ?>
```

The output from the program should look like this:

```
   Column Type
Null Key id
smallint(6) NO   PRI
family varchar(32) NO
name   varchar(32) NO age
tinyint(4)  NO
```

Dropping a Table

Dropping a table is very easy to do and is therefore very dangerous, so be careful. Example 10-11 shows the code that you need. However, I don't recommend that you try it until you have been through the other examples, as it will drop the table *cats* and you'll have to re-create it using Example 10-9.

Example 10-11. Dropping the table cats

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "DROP TABLE cats";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

Adding Data

Let's add some data to the table using the code in Example 10-12.

Example 10-12. Adding data to table cats

```
<?php
  require_once 'login.php';
  $db_server = mysql_connect($db_hostname, $db_username,
$db_password);
  if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "INSERT INTO cats VALUES(NULL, 'Lion', 'Leo',
4)";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

You may wish to add a couple more items of data by modifying `$query` as follows and calling up the program in your browser again:

```
$query = "INSERT INTO cats VALUES(NULL, 'Cougar',
'Growler', 2)";
$query = "INSERT INTO cats VALUES(NULL, 'Cheetah',
'Charly', 3)";
```

By the way, notice the `NULL` value passed as the first parameter? This is because the *id* column is of type `AUTO_INCREMENT`, and MySQL will decide what value to assign according to the next available number in sequence, so we simply pass a `NULL` value, which will be ignored.

Of course, the most efficient way to populate MySQL with data is to create an array and insert the data with a single query.

Retrieving Data

Now that some data has been entered into the *cats* table, Example 10-13 shows how you can check that it was correctly inserted.

Example 10-13. Retrieving rows from the cats table

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
```

```
    or die("Unable to select database: " . mysql_error());

  $query  = "SELECT * FROM cats";
  $result = mysql_query($query);   if (!$result) die ("Database

access failed: " . mysql_error());   $rows =

mysql_num_rows($result);   echo "<table><tr> <th>Id</th>

<th>Family</th><th>Name</th><th>Age</th></tr>";

  for ($j = 0 ; $j < $rows ; ++$j)
  {
    $row =
mysql_fetch_row($result);
echo "<tr>";

    for ($k = 0 ; $k < 4 ;
++$k)       echo
"<td>$row[$k]</td>";

    echo
"</tr>";   }

  echo
"</table>"; ?>
```

This code simply issues the MySQL query `SELECT * FROM cats` and then displays all the rows returned. Its output is as follows:

```
Id Family  Name    Age
1  Lion    Leo     4
2  Cougar  Growler 2
3  Cheetah Charly  3
```

Here you can see that the *id* column has correctly auto-incremented.

Updating Data

Changing data that you have already inserted is also quite simple. Did you notice the spelling of Charly for the cheetah's name? Let's correct that to Charlie, as in Example 10-14.

Example 10-14. Renaming Charly the cheetah to Charlie

```
<?php
  require_once 'login.php';   $db_server =

mysql_connect($db_hostname, $db_username, $db_password);
```

```
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "UPDATE cats SET name='Charlie' WHERE
name='Charly'";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

If you run Example 10-13 again, you'll see that it now outputs the following:

```
Id Family  Name    Age
1  Lion    Leo     4
2  Cougar  Growler 2
3  Cheetah Charlie 3
```

Deleting Data

Growler the cougar has been transferred to another zoo, so it's time to remove him from the database; see Example 10-15.

Example 10-15. Removing Growler the cougar from the cats table

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "DELETE FROM cats WHERE name='Growler'";
  $result = mysql_query($query);

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

This uses a standard `DELETE FROM` query, and when you run Example 10-13, you can see that the row has been removed in the following output:

```
Id Family  Name    Age
1  Lion    Leo     4
3  Cheetah Charlie 3
```

Using AUTO_INCREMENT

When using `AUTO_INCREMENT`, you cannot know what value has been given to a column before a row is inserted. Instead, if you need to know it, you must ask MySQL afterward using the `mysql_insert_id` function. This need is common: for instance, when you process a purchase, you might insert a new customer into a *Customers* table and then refer to the newly created *CustId* when inserting a purchase into the purchase table.

Example 10-12 can be rewritten as Example 10-16 to display this value after each insert.

Example 10-16. Adding data to table cats and reporting the insertion id

```
<?php
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "INSERT INTO cats VALUES(NULL, 'Lynx',
'Stumpy', 5)";
  $result = mysql_query($query);
  echo "The Insert Id was: " . mysql_insert_id();

  if (!$result) die ("Database access failed: " .
mysql_error()); ?>
```

The contents of the table should now look like the following (note how the previous *id* value of 2 is *not* reused, as this could cause complications in some instances):

```
Id Family  Name    Age
1  Lion    Leo     4
3  Cheetah Charlie 3
4  Lynx    Stumpy  5
```

Using insert IDs

It's very common to insert data in multiple tables: a book followed by its author, or a customer followed by his purchase, and so on. When doing this with an auto-increment column, you will need to retain the insert ID returned for storing in the related table.

For example, let's assume that these cats can be "adopted" by the public as a means of raising funds, and that when a new cat is stored in the *cats* table, we also want to create a key to tie it to the animal's adoptive owner. The code to do this is similar to that in

Example 10-16, except that the returned insert ID is stored in the variable `$insertID`, and is then used as part of the subsequent query:

```
$query    = "INSERT INTO cats VALUES(NULL, 'Lynx',
'Stumpy', 5)";
$result   = mysql_query($query);
$insertID = mysql_insert_id();

$query    = "INSERT INTO owners VALUES($insertID,
'Ann', 'Smith')";
$result   = mysql_query($query);
```

Now the cat is connected to its "owner" through the cat's unique ID, which was created automatically by `AUTO_INCREMENT`.

Using locks

A completely safe procedure for linking tables through the insert ID is to use locks (or transactions, as described in Chapter 9). It can slow down response time a bit when there are many people submitting data to the same table, but it can also be worth it. The sequence is:

1. Lock the first table (e.g., *cats*).
2. Insert data into the first table.
3. Retrieve the unique ID from the first table through `mysql_insert_id`.
4. Unlock the first table.
5. Insert data into the second table.

You can safely release the lock before inserting data into the second table, because the insert ID has been retrieved and is stored in a program variable. You could also use a transaction instead of locking, but that slows down the MySQL server even more.

Performing Additional Queries

OK, that's enough feline fun. To explore some slightly more complex queries, we need to revert to using the *customers* and *classics* tables that you created in Chapter 8. There will be two customers in the *customers* table; the *classics* table holds the details of a few books. They also share a common column of ISBNs, called *isbn*, that we can use to perform additional queries.

For example, to display all of the customers along with the titles and authors of the books they have bought, you can use the code in Example 10-17. *Example 10-17. Performing a secondary query*

```
<?php
```

```
  require_once 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query  = "SELECT * FROM customers";
  $result = mysql_query($query);   if (!$result) die

("Database access failed: " . mysql_error());   $rows

= mysql_num_rows($result);

  for ($j = 0 ; $j < $rows ; ++$j)
  {
    $row = mysql_fetch_row($result);
echo "$row[0] purchased ISBN
$row[1]:<br>";

    $subquery  = "SELECT * FROM classics WHERE
isbn='$row[1]'";
    $subresult = mysql_query($subquery);     if

(!$subresult) die ("Database access failed: " .

mysql_error());     $subrow = mysql_fetch_row($subresult);

    echo "  '$subrow[1]' by $subrow[0]<br>";
  }
?>
```

This program uses an initial query to the *customers* table to look up all the customers and then, given the ISBN of the book each customer purchased, makes a new query to the *classics* table to find out the title and author for each. The output from this code should be as follows:

```
Mary Smith purchased ISBN 9780582506206:
  'Pride and Prejudice' by Jane
Austen Jack Wilson purchased ISBN
9780517123201:
  'The Origin of Species' by Charles Darwin
```

Of course, although it wouldn't illustrate performing additional queries, in this particular case you could also return the same information using a `NATURAL JOIN` query (see Chapter 8), like this:

```
SELECT name,isbn,title,author FROM customers
  NATURAL JOIN classics;
```

Preventing SQL Injection

It may be hard to understand just how dangerous it is to pass user input unchecked to MySQL. For example, suppose you have a simple piece of code to verify a user, and it looks like this:

```
$user  = $_POST['user'];
$pass  = $_POST['pass'];
$query = "SELECT * FROM users WHERE user='$user' AND
pass='$pass'";
```

At first glance, you might think this code is perfectly fine. If the user enters values of `fredsmith` and `mypass` for `$user` and `$pass`, respectively, then the query string, as passed to MySQL, will be as follows:

```
SELECT * FROM users WHERE user='fredsmith' AND
pass='mypass'
```

This is all well and good, but what if someone enters the following for `$user` (and doesn't even enter anything for `$pass`)?

```
admin' #
```

Let's look at the string that would be sent to MySQL:

```
SELECT * FROM users WHERE user='admin' #' AND pass=''
```

Do you see the problem there? In MySQL, the # symbol represents the start of a comment. Therefore, the user will be logged in as *admin* (assuming there is a user *admin*), without having to enter a password. In the following, the part of the query that will be executed is shown in bold; the rest will be ignored.

```
SELECT * FROM users WHERE user='admin' #' AND pass=''
```

But you should count yourself very lucky if that's all a malicious user does to you. At least you might still be able to go into your application and undo any changes the user makes as *admin*. But what about the case in which your application code removes a user from the database? The code might look something like this:

```
$user  = $_POST['user'];
$pass  = $_POST['pass'];
$query = "DELETE FROM users WHERE user='$user' AND
pass='$pass'";
```

Again, this looks quite normal at first glance, but what if someone entered the following for `$user`?

```
anything' OR 1=1 #
```

This would be interpreted by MySQL as:

```
DELETE FROM users WHERE user='anything' OR 1=1 #' AND pass=''
```

Ouch—that SQL query will always be TRUE and therefore you've lost your whole *users* database! So what can you do about this kind of attack?

Well, the first thing is not to rely on PHP's built-in *magic quotes*, which automatically escape any characters such as single and double quotes by prefacing them with a backslash (\). Why? Because this feature can be turned off; many programmers do so in order to put their own security code in place. So there is no guarantee that this hasn't happened on the server you are working on. In fact, the feature was deprecated as of PHP 5.3.0 and has been removed in PHP 6.0.0.

Instead, you should always use the function `mysql_real_escape_string` for all calls to MySQL. Example 10-18 is a function you can use that will remove any magic quotes added to a user-inputted string and then properly sanitize it for you.

Example 10-18. How to properly sanitize user input for MySQL

```
<?php
  function mysql_fix_string($string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return mysql_real_escape_string($string);
  }
?>
```

The `get_magic_quotes_gpc` function returns TRUE if magic quotes are active. In that case, any slashes that have been added to a string have to be removed, or the function `mysql_real_escape_string` could end up double-escaping some characters, creating corrupted strings. Example 10-19 illustrates how you would incorporate `mysql_fix_string` within your own code.

Example 10-19. How to safely access MySQL with user input

```
<?php
  $user  = mysql_fix_string($_POST['user']);
  $pass  = mysql_fix_string($_POST['pass']);
  $query = "SELECT * FROM users WHERE user='$user' AND
pass='$pass'";
  function mysql_fix_string($string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return mysql_real_escape_string($string);
  }
?>
```

Remember that you can use `mysql_real_escape_string` only when a MySQL database is actively open; otherwise, an error will occur.

Using Placeholders

Another way—this one virtually bulletproof—to prevent SQL injections is to use a feature called *placeholders*. The idea is to predefine a query using `?` characters where the data will appear. Then, instead of calling a MySQL query directly, you call the predefined one, passing the data to it. This has the effect of ensuring that every item of data entered is inserted directly into the database and cannot be interpreted as SQL queries. In other words, SQL injections become impossible.

The sequence of queries to execute when using MySQL's command line would be like that in Example 10-20.

Example 10-20. Using placeholders

```
PREPARE statement FROM "INSERT INTO classics
VALUES(?,?,?,?,?)";

SET @author   = "Emily Brontë",
    @title    = "Wuthering Heights",
    @category = "Classic Fiction",
    @year     = "1847",
    @isbn     = "9780553212587"; EXECUTE statement

USING @author,@title,@category,@year,@isbn;

DEALLOCATE PREPARE statement;
```

The first command prepares a statement called `statement` for inserting data into the *classics* table. As you can see, in place of values or variables for the data to insert, the statement contains a series of `?` characters. These are the placeholders.

The next five lines assign values to MySQL variables according to the data to be inserted. Then the predefined statement is executed, passing these variables as parameters. Finally, the statement is removed, in order to return the resources it was using.

In PHP, the code for this procedure looks like Example 10-21 (assuming that you have created *login.php* with the correct details to access the database).

Example 10-21. Using placeholders with PHP

```
<?php
```

```
  require 'login.php';   $db_server =
mysql_connect($db_hostname, $db_username, $db_password);
if (!$db_server) die("Unable to connect to MySQL: " .
mysql_error());

  mysql_select_db($db_database)
    or die("Unable to select database: " . mysql_error());

  $query = 'PREPARE statement FROM "INSERT INTO classics
VALUES(?,?,?,?,?)"';   mysql_query($query);

  $query = 'SET @author = "Emily Brontë",' .
           '@title = "Wuthering Heights",' .
           '@category = "Classic Fiction",' .
           '@year = "1847",' .
           '@isbn = "9780553212587"';
mysql_query($query);

  $query = 'EXECUTE statement USING
@author,@title,@category,@year,@isbn';   mysql_query($query);

  $query = 'DEALLOCATE PREPARE statement';

mysql_query($query);
?>
```

Once you have prepared a statement, until you deallocate it you can use it as often as you wish. Such statements are commonly used within a loop to quickly insert data into a database by assigning values to the MySQL variables and then executing the statement. This approach is more efficient than creating the entire statement from scratch on each pass through the loop.

Preventing HTML Injection

There's another type of injection you need to concern yourself about—not for the safety of your own websites, but for your users' privacy and protection. That's *cross-site scripting*, also referred to as *XSS*.

This occurs when you allow HTML, or more often JavaScript code, to be input by a user and then displayed back by your website. One place this is common is in a comment form. What happens most often is that a malicious user will try to write code that steals cookies from your site's users, allowing him or her to discover username and password pairs or other information. Even worse, the malicious user might launch an attack to download a Trojan onto a user's computer.

But preventing this is as simple as calling the `htmlentities` function, which strips out all HTML markup codes and replaces them with a form that displays the characters, but does not allow a browser to act on them. For example, consider the following HTML:

```
<script src='http://x.com/hack.js'>
</script><script>hack();</script>
```

This code loads in a JavaScript program and then executes malicious functions. But if it is first passed through `htmlentities`, it will be turned into the following, totally harmless string:

```
&lt;script src='http://x.com/hack.js'&gt;
&lt;/script&gt;&lt;script&gt;hack();&lt;/script&gt;
```

Therefore, if you are ever going to display anything that your users enter, either immediately or after storing it in a database, you need to first sanitize it with `htmlentities`. To do this, I recommend that you create a new function, like the first one in Example 10-22, which can sanitize for both SQL and XSS injections.

Example 10-22. Functions for preventing both SQL and XSS injection attacks

```
<?php
  function mysql_entities_fix_string($string)
  {
    return
htmlentities(mysql_fix_string($string));
}

  function mysql_fix_string($string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return mysql_real_escape_string($string);
  }
?>
```

The `mysql_entities_fix_string` function first calls `mysql_fix_string` and then passes the result through `htmlentities` before returning the fully sanitized string. Example 10-23 shows your new "ultimate protection" version of Example 10-19.

Example 10-23. How to safely access MySQL and prevent XSS attacks

```
<?php
  $user  = mysql_entities_fix_string($_POST['user']);
  $pass  = mysql_entities_fix_string($_POST['pass']);
  $query = "SELECT * FROM users WHERE user='$user' AND
pass='$pass'";
```

```
  function mysql_entities_fix_string($string)
  {
    return
htmlentities(mysql_fix_string($string));
}

  function mysql_fix_string($string)
  {
    if (get_magic_quotes_gpc()) $string =
stripslashes($string);
    return mysql_real_escape_string($string);
  }
?>
```

Now that you have learned how to integrate PHP with MySQL and avoid malicious user input, the next chapter will explain how to use the improved `MySQLi` extension for your MySQL queries.

Questions

1. What is the standard PHP function for connecting to a MySQL database?
2. When is the `mysql_result` function not optimal?
3. Give one reason why using the `POST` form method is usually better than `GET`.
4. How can you determine the most recently entered value of an `AUTO_INCREMENT` column?
5. Which PHP function escapes a string, making it suitable for use with MySQL?
6. Which function can be used to prevent XSS injection attacks?

See "Chapter 10 Answers" on page 646 in Appendix A for the answers to these questions.

UNIT - III

3 Networking Essentials

Syllabus

Fundamental computer network concepts - Types of computer networks - - Network layers - TCP/IP model - Wireless Local Area Network - Ethernet - WiFi - Network Routing - Switching - Network components.

Contents

2

3.1 Fundamental Computer Network Concepts

- Communication means to convey a message, a picture, speech or an idea that is received and understood clearly and correctly by the person for whom it is conveyed. Data communication containing messages, pictures and voice has taken the importance.
- Network is a set of devices connected by media links. The links connecting the devices are often called communication channels. The type of network is determined by size, its ownership, physical architecture and the distance it covers.
- Computer networking consists of two or more computers that are linked in order to share resources, exchange files, or allow electronic communications. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams.
- Data communication consists of five elements. They are sender, receiver, message, transmission medium and protocol. Fig 3.1.1 shows data communication system.
  1. **Sender :** Sender machine creates data and send it to the receiver machine.
  2. **Receiver :** Receive data and information from sender.
  3. **Message :** The message is the information or data that is to be communicated. It may consist of text, numbers, pictures, sounds, videos or any combination of these.
  4. **Protocol :** A set of rules that defines how data is formatted and processed on a network.
  5. **Transmission medium :** It is path between sender and receiver. Message is transmitted through this medium.
- In data communication, the point to point is commonly used to establish a direct connection between two networking devices. The above Fig. 3.1.1 is point to point communication. Point-to-point networks provide a dedicated link between any two stations. The data packets are sent from source station to the destination station.
- Multi-point communication means one to many i.e. one source machine communicate with multiple receiver machine.

3.1.1 Topology

- The topology of a network defines how the nodes of a network are connected. The way the nodes are connected to form a network is known as its topology.
- A network is defined by a physical topology and a logical topology
- Physical topology defines how the nodes of the network are physically connected. Logical topology dedicated connections between certain selected source-destination pairs using the underlying physical topology.
- Routing and flow control build on properties of topology. In order to have an efficient system, the logical topology should be chosen such that either the average hop distance or the packet delay or the maximum flow on any link must be minimal.
- Physical topology : Actual layout of the media.

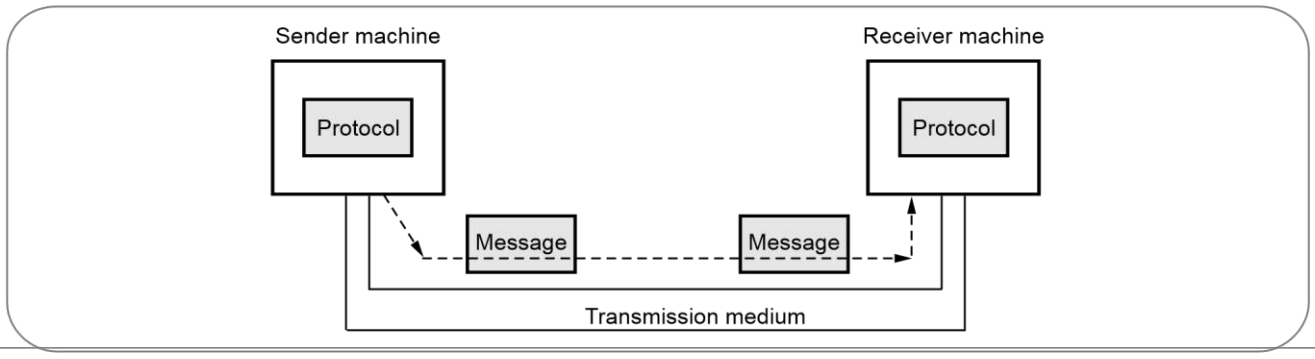


Fig. 3.1.1 Data communication system

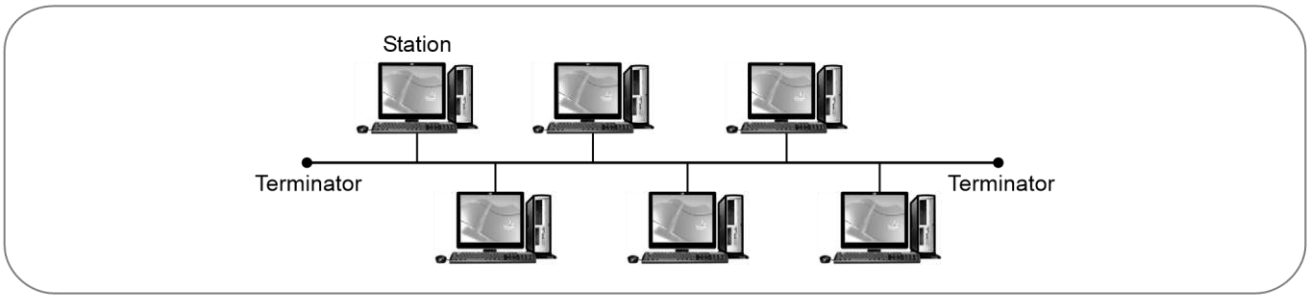


Fig. 3.1.2 bus topology

- Physical topologies fall into three main categories : bus, star, and ring topology.
- Logical topology : How the hosts access the media.
- Commonly used topologies in the computer networks are bus, star, ring, mesh and tree topology.

1. Bus Topology

- In bus network, all stations are attached to a single cable. Fig 3.1.2 shows bus topology. A bus topology is linear local area network architecture.
- When a station sends a message, it is **broadcast** down on the cable in both directions. Terminators at the end of the cable prevent the signal from reflecting back to the sender.
- All stations on the cable constantly monitor for messages meant to them. When a station detects a message meant for it, it reads the message from the cable and the other stations will ignore it.
- Since all stations are sharing the same cable, some form of control is needed to make sure which station will transmit when; otherwise there will be a collision.
- The most widely used LAN implementations, Standard Ethernet/IEEE 802.3 networks implement a bus topology in which all devices are connected to a central cable, called the bus or backbone.
- The bus topology uses the baseband signaling method.

Advantages of Bus Topology

- Easy to implement
- Good for temporary networks that must be set up in a hurry
- Failure of one station does not affect others

Disadvantages of Bus Topology

- Difficult to troubleshoot
- Limited cable length and number of stations
- No redundancy
- The bus cable can be a bottleneck when network traffic gets heavy

2. Star Topology

- In star network, each station is connected via a point-to-point link to a central point. This central point is called hub, or concentrator.
- All signals, instructions, and data going to and from each node must pass through the hub to which the node is connected.
- Fig 3.1.3 shows star topology.

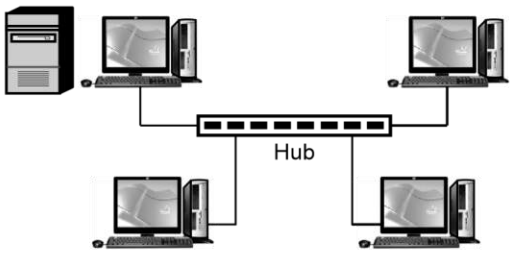


Fig. 3.1.3 Star topology

- The star topology can use coaxial, twisted pair, or fiber optic cable. This topology uses the baseband signaling method.
- Ethernet with a hub-based star topology is a broadcast LAN, whenever a hub receives a bit from one of its interfaces; it sends a copy out on all of its other interfaces.
- Each networked device in star topology can access the media independently

4

Advantages of Star Topology

- Easy to connect new nodes or devices
- Centralized management. It helps in monitoring the network
- Failure of one node or link doesn't affect the rest of network

Disadvantages of Star Topology

- If the central hub fails, the whole network fails to operate.
- Each device requires its own cable segment

3. Ring Topology

- Each node is connected to the two nearest nodes so the entire network forms a circle. One method for passing data on ring networks is token passing. Fig 3.1.4 shows ring topology.

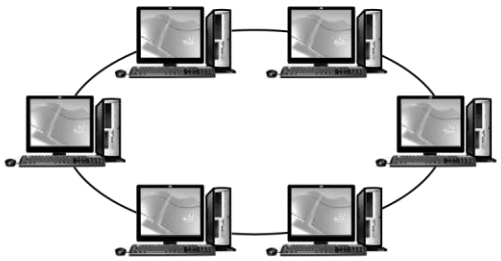


Fig. 3.1.4 Ring topology

- Ring topology is a LAN architecture in which all devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it.
- The signals travel on the cable in only one direction. Since each computer retransmits what it receives.
- A ring topology is both a logical and a physical topology. As a logical topology, a ring is distinguished by the fact that message packets are transmitted sequentially from node to node, in a predefined order, and as such it is an example of a point-to-point system.
- As a physical topology, a ring describes a network in which each node is connected to exactly two other nodes.
- All nodes on the network receive each message, to a point. A node may be either active or inactive. Active node means the node is able to send or receive messages. Inactive node is not able to send or receive messages.
- Ring speeds are 4 Mbps, 16 Mbps and 100 Mbps and uses twisted pair and fiber optic cable.
- Ring technique is based on the use of a small frame called a token that circulates when all stations are idle. Whenever a station wishes to send a frame it waits until it receives the token.
- Since ring topologies use token passing to control access to the network, the token is returned to sender with the acknowledgement

Advantages of Ring Topology

- Easier to manage;
- Easier to locate a defective node or cable problem
- Well-suited for transmitting signals over long distances on a LAN
- Handles high-volume network traffic
- Enables reliable communication

Disadvantages of Ring Topology

- Adding or removing nodes disrupts the network.
- Failure of one node on the ring can affect the whole network.
- Cost of cable is more in ring network.

4. Mesh Topology

- Every networked node is directly connected to every other networked node. Fig. 3.1.5 shows mesh topology.

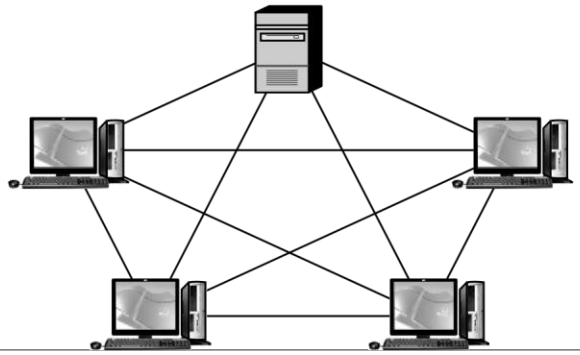


Fig. 3.1.5 Mesh topology

- Mesh networks operate in two ways : by either routing the data or flooding the data.
- A simpler mesh network takes a flood approach, where the data flows continuously throughout the network. If a station sees data with its address, it simply grabs it.
- Much of the bandwidth available in mesh configuration is wasted.
- Full mesh is generally utilized as a backbone where there are few nodes but a great need for fault tolerance, such as the backbone of a telecommunications company or ISP.

Advantages of Mesh Topology

- Provides redundant paths between device.
- Troubleshooting is easy.

Disadvantages of Mesh Topology

- Difficulty of installation.
- Costly because of maintaining redundant links.
- Difficulty of reconfiguration.

Review Question

1. *Explain the various networking topologies in detail.*

3.2 Types of Computer Networks

- Computer networks can be categorized depending on their physical size. Major categories of computer networks are as follows :
  1. Local Area Networks (LAN)
  2. Metropolitan Area Networks (MAN)
  3. Wide Area Networks (WAN)

3.2.1 Local Area Networks

- A Local Area Network (LAN) is a network that is confined to a relatively small area. It is generally limited to a geographic area such as a college, lab or building. It is typically a privately owned data communication system in which the user shares resources. LAN is used for communication in a small community in which resources, such as printers, software and servers are shared. Fig 3.2.1 shows LAN.

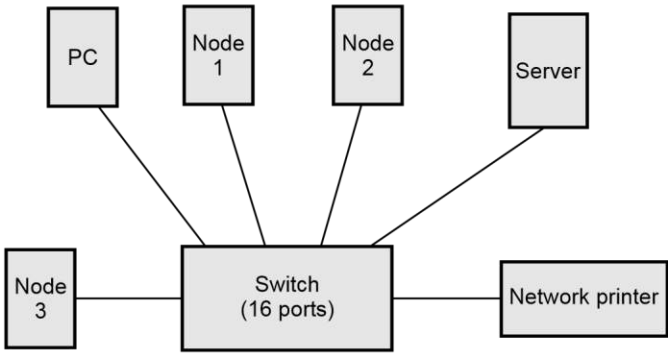


Fig. 3.2.1 LAN

- Characteristics of LANs
  1. Easily resource sharing.
  2. Data transfer rate are high.
  3. Small area covered by LAN
  4. Cost of setting up the network is usually low.
  5. Flexibility, low error rates and reliability of operation and simple maintenance.
- LAN generally uses only one type of transmission media. LANs are capable of transmitting data at very fast rates, much faster than data can be transmitted over a telephone line; but the distances are limited, and there is also a limit on the number of computers that can be attached to a single LAN.
- The following characteristics differentiate one LAN from another :
  1. **Topology :** The geometric arrangement of devices on the network.
  2. **Protocols :** The rules and encoding specifications for sending data. The protocols also determine whether the network uses a peer-to-peer or client/server architecture.
  3. **Media :** Devices can be connected by twisted-pair wire, coaxial cables, or fiber optic cables.
- LAN provides two way communications between large varieties of data communication terminals within a limited geographical area.

Advantages of Local Area Networks

1. Ability to share hardware and software resources.

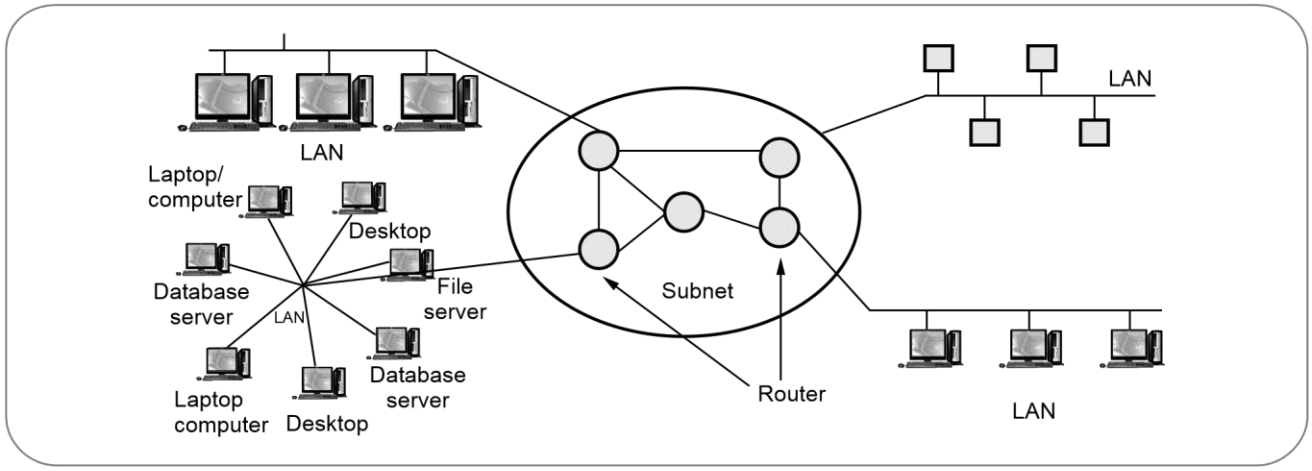


Fig. 3.2.2 WAN

2. Support for heterogeneous forms of hardware and software.
3. Access to other LANs and WANs.
4. Private ownership.
5. Secure transfers at high speeds with low error rates.

3.2.2 Wide Area Network

- WAN provides long distance transmission of data and voice. Computers connected to a wide-area network are often connected through public networks, such as the telephone system. They can also be connected through leased lines or satellites. The largest WAN in existence is the Internet. Fig. 3.2.2 shows WAN.
- WAN consists of host and collection of machines. User program is installed on the host machine and nodes. All the hosts are connected by each other through communication subnet. Subnet carries messages from host to host.
- Subnet consists of transmission lines and switching elements. The transmission line is used for data transfer between two machines. Switching elements are used for connecting two transmission lines. Switching elements are specialized computers. It selects the proper outgoing line for incoming data and forwards the data on that line.
- The switching elements are basically computers and they are called packet switching nodes, intermediate systems and data switching exchanges. These switching elements are also called routers. Host are owned by users and subnet is owned by the telephone company or an Internet service provider.
- Each node in a WAN is a router that accepts an input packet, examines the destination address, and forwards the packet on to a particular telecommunications line. A router must select the one transmission line that will best provide a path to the destination and in an optimal manner.
- In the WAN, when the packet is sent from one router to another via one or more intermediate routers, the packet is received at each intermediate router in its entirety. This packet is stored in that router until the required output line is free. The subnet which uses this principle is called point to point, store and forward, or packet switched subnet. Almost all the WANs use store and forward subnets. If the packets are small and of same size, they are also called cells.
- WAN uses hierarchical addressing because they facilitate routing. Addressing is required to identify which network input is to be connected to which network output.

3.2.3 Metropolitan Area Networks (MAN)

- The Metropolitan Area Network standards are sponsored by the IEEE, ANSI and the Regional Bell operating companies. MAN standard is organized around a topology and technique called Distributed Queue Dual Bus (DQDB).

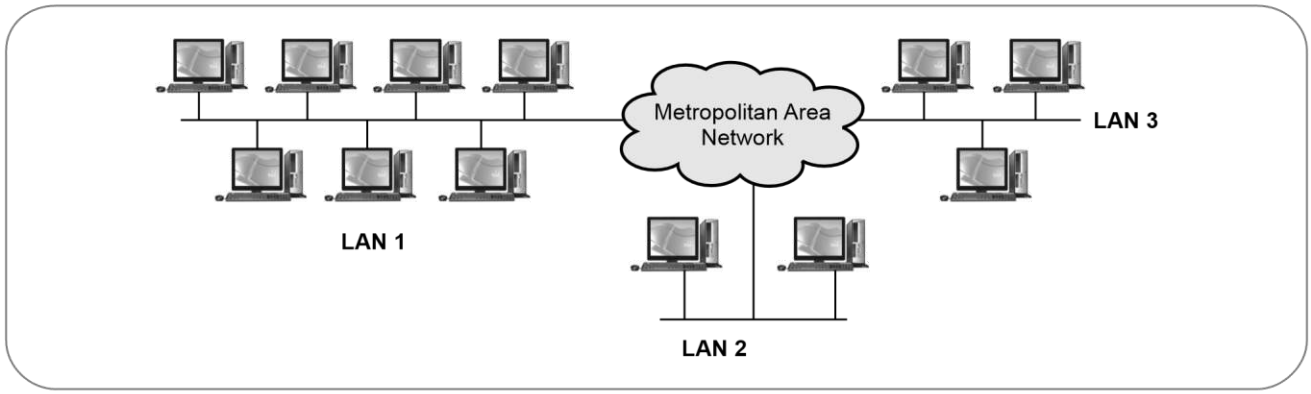


Fig. 3.2.3 MAN

- The network size falls intermediate between LANs and WANs. A MAN typically covers an area of between 5 and 50 km diameter. Many MANs cover an area the size of a city, although in some cases MANs may be as small as a group of buildings.
- A MAN often acts as a high speed network to allow sharing of regional resources (similar to a large LAN). It is also frequently used to provide a shared connection to other networks using a link to a WAN. MAN provides the transfer rates from 34 to 150 Mbps.
- Fig. 3.2.3 shows MAN. Backbone of MAN is high-capacity and high-speed fiber optics. MAN is works in between Local Area Network and Wide Area Network. MAN provides uplink for LANs to WANs or Internet.
- MAN as a special category is that a standard has been adopted for them and this standard is now being implemented. It is called IEEE 802.6. In MAN, bit error rate and delay higher than LAN.

3.2.4 Difference between LAN, WAN and MAN

LAN	WAN	MAN
A network that connects a relatively small number of machines in a relatively close geographical area	A network that connects two or more local-area networks over a potentially large geographic distance	The communication infrastructures that have been developed in and around large cities
Covers small area. i.e. within the building	Covers large geographical area	Covers larger than LAN and smaller than WAN
Low bit error rate and delay	High bit error rate and delay.	Bit error rate and delay higher than LAN
Support higher data transfer rate	Support lower data transfer rate	Support moderate data transfer rate
LAN uses inexpensive equipment	WAN uses most expensive equipment	MAN uses moderately expensive equipment

3.3 Network Layer

- The main objective of the network layer is to allow end systems, connected to different networks, to exchange information through intermediate systems called router. The unit of information in the network layer is called a packet.
- **Network Layer** is responsible for addressing messages and data so they are sent to the correct destination, and for translating logical addresses and names into physical addresses.
- This layer is also responsible for finding a path through the network to the destination computer. Lowest layer that deals with host-to-host communication, call this end-to-end communication.

Functions of network layer

a. **Logical addressing :** Data link layer implements physical addressing. When a packet passes network boundary, an addressing system is needed to distinguish source and destination, network layer performs these function.

b. **Routing :** Network layer route or switch the packets to its final destination in an internetwork.

c. Frame fragmentation : If it determines that a downstream router's Maximum Transmission Unit (MTU) size is less than the frame size, a router can fragment a frame for transmission and re-assembly at the destination station.

- Fig. 3.3.1 shows network layer.

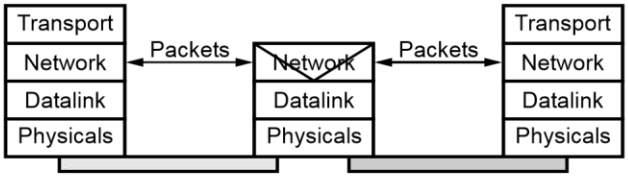


Fig. 3.3.1 Network layer

- Principles of network layer :
  1. Each network layer entity is identified by a network layer address. This address is independent of the data link layer addresses that it may use.
  2. The service provided by the network layer does not depend on the service or the internal organization of the underlying data link layers.
  3. The network layer is conceptually divided into two planes : **Data plane and control plane.** The data plane contains the protocols and mechanisms that allow hosts and routers to exchange packets carrying user data. The control plane contains the protocols and mechanisms that enable routers to efficiently learn how to forward packets towards their final destination.
- Network layer provides two types of services :
  1. An unreliable connectionless service
  2. Connection-oriented, reliable or unreliable, service
- Internal organizations of the network layer uses datagram and virtual circuits. Datagram is used to provide a connectionless service while a virtual circuit is used in networks that provide a connection-oriented service.
- Datagram is a type of packet that happens to be sent in a connectionless manner over a network. Every datagram carries enough information to let the network forward the packet to its correct destination.
- The network layer limits the maximum packet size. Thus, the information must have been divided in packets by the transport layer before being passed to the network layer.

3.3.1 IPv4

- The Internet Protocol (IP) is the network layer protocol of the TCP/IP protocol suite.
- IP is a connectionless, unreliable, best-effort delivery protocol. All the nodes are identified using an IP address. Packets are delivered from the source to the destination using IP address
- IPv4 addresses are encoded as a 32 bits field. IPv4 addresses are often represented in dotted-decimal format as a sequence of four integers separated by a dot.
- An IPv4 address is used to identify an interface on a router or a host. IPv4 addresses are unique. Two devices on the internet can never have the same address at the same time.
- The address structure was originally defined to have a two level hierarchy : **Network ID and host ID.**
- The network ID identifies the network the host is connected to. The host ID identifies the network connection to the host rather than the actual host.

Packet Format

- The IP datagram consists of a header followed by a number of bytes of data. Fig. 3.3.2 shows header format of IPv4. All IPv4 packets use the 20 bytes header.
- **Version field** indicates the version of IP used to build the header. Current version is 4.
- **Header Length** indicates the length of the IP header in 32 bits words. This field allows IPv4 to use options if required, but as it is encoded as a 4 bits field, the IPv4 header cannot be longer than 64 bytes.
- **Type of service :** The service type is an indication of the quality of service requested for this IP datagram.

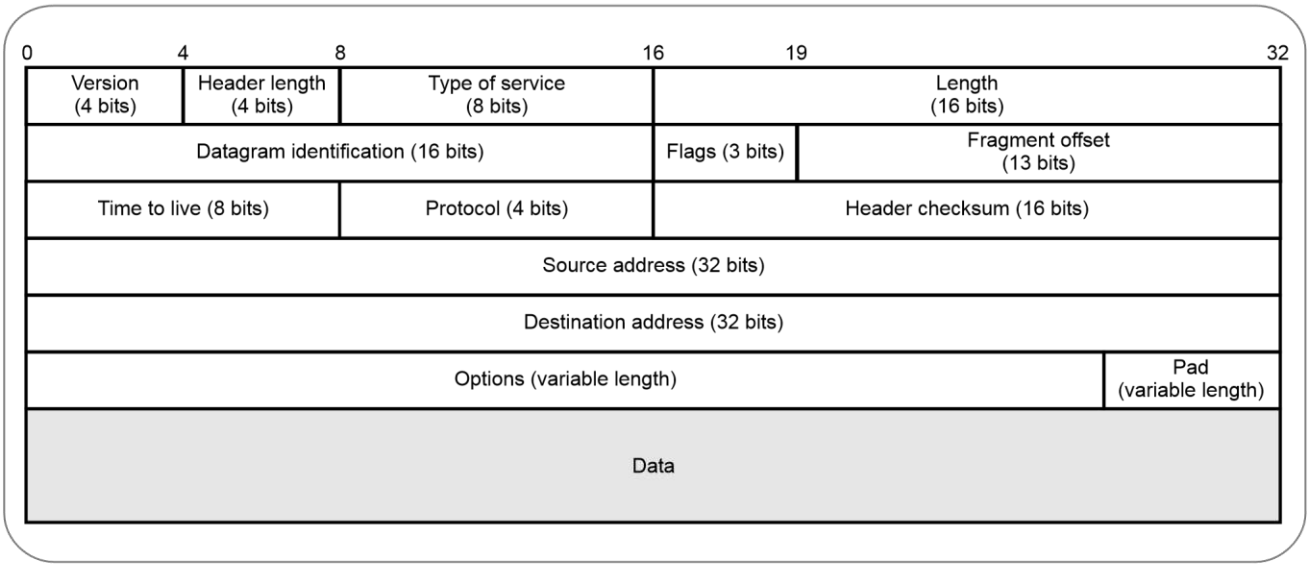


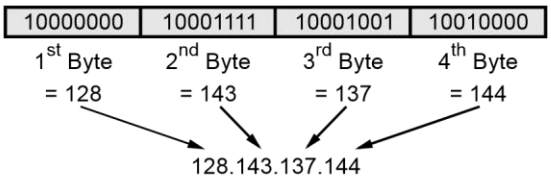
Fig. 3.3.2 IPv4 header format

- **Protocol field** indicates the transport layer protocol that must process the packet's payload at the destination. Common values for this field are 6 for TCP and 17 for UDP
- **Length field** indicates the total length of the entire IPv4 packet (header and payload) in bytes. This implies that an IPv4 packet cannot be longer than 65535 bytes.
- **Fragment offset** is used to reassemble the full datagram. The value in this field contains the number of 64-bit segments (header bytes are not counted) contained in earlier fragments. If this is the first (or only) fragment, this field contains a value of zero.
- **Flags (3 bits) :** Only two of the bits are currently defined : MF(More Fragments) and DF(Don't Fragment).
- **More Fragments (MF) flag** is a single bit in the Flag field used with the Fragment Offset for the fragmentation and reconstruction of packets. The More Fragments flag bit is set, it means that it is not the last fragment of a packet. When a receiving host sees a packet arrive with the MF = 1, it examines the Fragment Offset to see where this fragment is to be placed in the reconstructed packet. When a receiving host receives a frame with the MF = 0 and a non-zero value in the Fragment offset, it places that fragment as the last part of the reconstructed packet.
- **Don't Fragment (DF) flag** is a single bit in the Flag field that indicates that fragmentation of the packet is not allowed. If the Don't Fragment flag bit is set, then fragmentation of this packet is NOT permitted. If a router needs to fragment a packet to allow it to be passed downward to the Data Link layer but the DF bit is set to 1, then the router will discard this packet.
- **Time-to-Live (TTL)** is an 8-bit binary value that indicates the remaining "life" of the packet. The TTL value is decreased by at least one each time the packet is processed by a router (that is, each hop).When the value becomes zero, the router discards or drops the packet and it is removed from the network data flow.
- **Source address** field that contains the IPv4 address of the source host
- **Destination address** field that contains the IPv4 address of the destination host
- **Checksum** that protects only the IPv4 header against transmission errors. Checksum is for the information contained in the header. If the header checksum does not match the contents, the datagram is discarded.

- **Options (variable) :** Encodes the options requested by the sending user.
- **Padding (variable) :** Used to ensure that the datagram header is a multiple of 32 bits.
- **Data (variable) :** The data field must be an integer multiple of 8 bits. The maximum length of the datagram (data field plus header) is 65,535 octets.

IP Address

- An IPv4 address is a 32-bit sequence of ones and zeros. To make the IP address easier to work with, it is usually written as four decimal numbers separated by periods.
- For example, an IP address of one computer is 192.168.1.2. This is called the dotted decimal format.
- Each part of the address is called an octet because it is made up of eight binary digits. For example, the IP address 192.168.1.8 would be 11000000.10101000.00000001.00001000 in binary notation.
- Address 0.0.0.0, 127.0.0.1 and 255.255.255.255 carries special meaning. IP address is divided into a network number and a host number. The network prefix identifies a network and the host number identifies a specific host.
- **Example :**



- The IP address consists of a pair of numbers : **IP address = <network number><host number>**
- Example : Suppose IP address of technicalpublication.org is **129.144.136.146,** then network address is 129.144.0.0 and host number is 136.146.

Classes of IP Address

- IP address is divided into five classes: A, B, C, D, and E. Fig. 3.3.3 shows IP address classes with net ID and host ID.

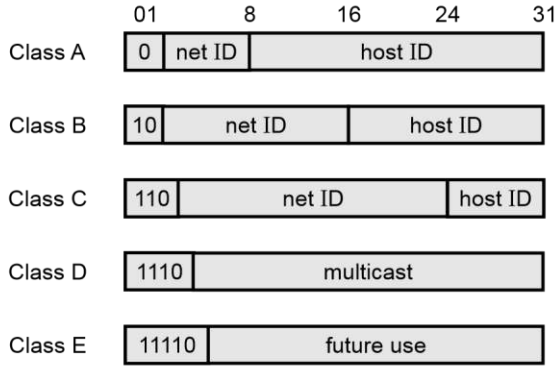


Fig. 3.3.3 IP address classes with net ID and host ID

- The class of an IP address is identified in the most significant few bits. If the first bit is 0, it is a class A address.
- If the first bit is 1 and second is 0, it is a class B address. If the first two bits are 1 and third is 0, it is a class C address.
- Class D is used for what is known as multicasting, transmitting data to multiple computers or routers.
- Class E was reserved for future use, but this has given way to IPv6 instead
- **Range of IP address :**

Class	Starting address	Last Address
Class A	0.0.0.0	127.255.255.255
Class B	128.0.0.0	191.255.255.255
Class C	192.0.0.0	223.255.255.255
Class D	224.0.0.0	239.255.255.255
Class E	240.0.0.0	255.255.255.255

- **Loopback Testing :** The 127 network number isn't used by hosts as a ***logical IP address.*** Instead, this network is used for ***loopback IP addresses,*** which allow for testing.
- Address 0.0.0.0, 127.0.0.1 and 255.255.255.255 carries special meaning.

Public and Private Addresses

- IPv4 addresses are further classified as either public or private. Public IP addresses are ones that are exposed to the Internet. Any other computers on the Internet can potentially communicate with them.

1

- Private IP addresses are hidden from the Internet and any other networks. They are usually behind an IP proxy or firewall device.
- Private Addresses are as follows :

Class	Starting Address	End of range
Class A	10.0.0.0	10.255.255.255
Class B	172.16.0.0	172.31.255.255
Class C	192.168.0.0	192.168.255.255

Subnetting

- Subnetting is the subdivision of logical IP network. By default, all computers are on one subnet or network with no divisions involved.
- Subnet mask of class A, B and C are as follows :

IP address Class	Subnet Mask
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

3.4 TCP/IP Model

- TCP/IP stands for Transmission Control Protocol / Internet Protocol. This model is based on a five-layer model for networking : physical layer, datalink layer, network or internet layer, transport layer and application layer.
- The TCP/IP stack is open. The TCP/IP protocol stack models a series of protocol layers for networks and systems that allows communications between any types of devices. Fig. 3.4.1 shows TCP/IP protocol suite. The reference model was named after two of its main protocols, TCP and Internet Protocol.

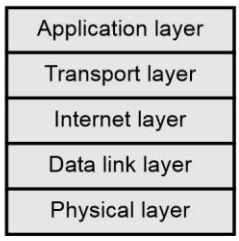


Fig. 3.4.1 TCP/IP protocol suite

- TCP stands for Transmission Control Protocol. TCP breaks messages into packets, hands them off to the IP software for delivery, and then orders and reassembles the packets at their destination
- IP stands for Internet Protocol. It deals with the routing of packets through the maze of interconnected networks to their final destination.
- At the physical and data link layers, the TCP/IP protocols do not define any standards.

Function of TCP/IP Layers :

1. **Application Layer :** Application layer includes all process and services that use the transport layer to deliver data. The original TCP/IP specification described a number of different applications that fit into the top layer of the protocol stack.
   - These applications include Telnet, FTP, SMTP and DNS. TELNET is the Network Terminal Protocol, which provides remote login over the network. FTP is used for interactive file transfer. SMTP delivers the electronic mail.
2. **Transport Layer :** This layer provides communication session management between host computers. Defines the level of service and status of the connection used when transporting data.
   - Transport layer manages connection oriented streams, flow control, reliable transport and multiple transmissions. Application programs send data to the transport layer protocols TCP and UDP.
   - An application is designed to choose either TCP or UDP based on the services it needs. The transport layer provides peer entities on the source and destination hosts to carry on a conversation. Data may be user data or control data.
   - Two modes are available, full-duplex and half duplex. In full-duplex operation, both sides can transmit and receive data simultaneously, whereas in half duplex, a side can only send or receive at one time.

2

3. **Network or Internet Layer :** Packages data into IP datagrams, which contain source and destination address information that is used to forward the datagrams between hosts and across networks. Performs routing of IP datagrams.
   - The Internet network level protocol (IP, ARP, ICMP) handle machine to machine communications. The primary protocol used to move data is the Internet Protocol (IP), which provides fragmentation and addressing services.
   - IP provides a connectionless method of delivering data from one host to another. It does not guarantee delivery and does not provide sequencing of datagrams.
   - It attaches a header to datagram that includes source address and the destination address, both of which are unique Internet addresses.
4. **Network Interface Layer :** It contains two sub-layer : **Data link layer and physical layer.** This layer is also called host to network layer.
   - Host to network layer cannot define any protocol. It is responsible for accepting and transmitting IP datagrams. This layer may consist of a device driver in the operating system and the corresponding network interface card in the machine.
   - It specifies details of how data is physically sent through the network, including how bits are electrically signaled by hardware devices that interface directly with a network medium, such as coaxial cable, optical fiber, or twisted-pair copper wire.

3.4.1 TCP/IP Data Encapsulation and Decapsulation

- The packet is the basic unit of information that is transferred across a network. The basic packet consists of a header with the sending and receiving systems' addresses, and **payload,** with the data to be transferred.
- As the packet travels through the TCP/IP protocol stack, the protocols at each layer either add or remove fields from the basic header.
- When a protocol on the sending system adds data to the packet header, the process is called **data encapsulation.**
- **Decapsulation :** The reverse process of encapsulation (or decapsulation) occurs when data

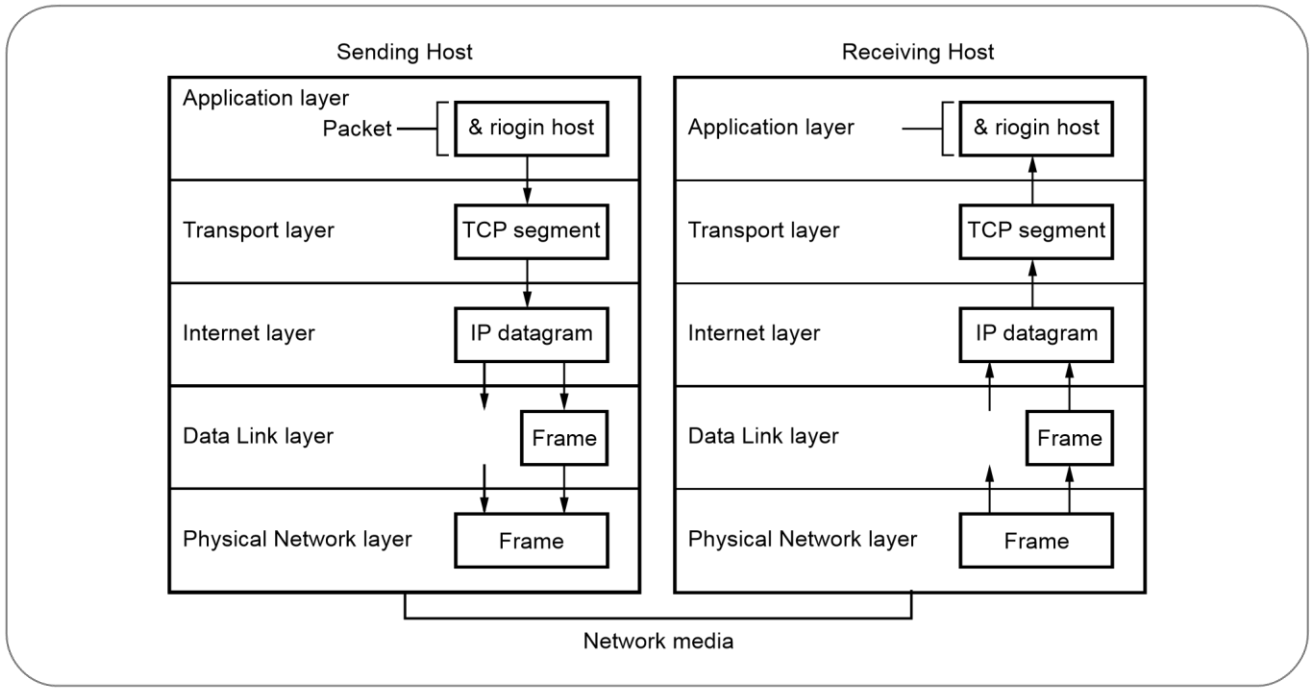


Fig. 3.4.1 Data flow through the TCP/IP stack

e s

3

is received on the destination computer. As the data moves up from the lower layer to the upper layer of TCP/IP protocol stack, each layer unpacks the corresponding header and uses the information contained in the header to deliver the packet to the exact network application waiting for the data.

- Fig. 3.4.1 shows data flow through the TCP/IP stack.
- Each layer uses encapsulation to add the information its peer needs on the receiving system.
- The communication process from any source to any destination can be as follows :
  1. Creation of data at the application layer of the originating sending host.
  2. Segmentation and encapsulation of data as it passes down the protocol stack in the sending host.
  3. Generation of the data onto the media at the network access layer (Internet layer) of the stack.
  4. Transportation of the data through the internetwork, which consists of media and any intermediary devices.
  5. Reception of the data at the Internet layer of the receiving host.
  6. Decapsulation and reassembly of the data as it passes up the stack in the receiving host.
  7. Passing this data to the destination application at the application layer of the receiving host.

3.4.2 Addressing

- An Internet employing TCP/IP protocols uses four levels of addresses : Physical (Link) addresses, Logical (IP) addresses, Port addresses and Specific addresses.
- The **physical address** is the lowest level address and is also referred as link address. The physical address of a node is defined by its LAN or WAN. The physical address is included in the frame by the data link layer.
- **Logical addresses** are necessary for universal communications. It is a 32-bit address which uniquely defines host connected to Internet.
- In TCP/IP architecture, th label a signed to a process is called **port address.** In TCP/IP the port address is of 16-bit.
- The **specific addresses** get changed to corresponding port and logical addresses by the station or host who sends it.

3.4.3 Difference between TCP and IP

- TCP is used to transfer packet data and An Internet Protocol (IP) is responsible for the logical addressing. IP obtains the address and TCP guarantees transfer of that packet data on a particular address.
- TCP breaks messages into packets, hands them off to the IP software for delivery, and then orders and reassembles the packets at their destination. IP deals with the routing of packets through the maze of interconnected networks to their final destination.
- TCP is connection oriented protocol and IP is connectionless protocol.
- When IP forwards datagrams, there is no guarantee that the datagrams will arrive. If they do arrive, they will not necessarily be in the correct order. TCP adds reliability to IP. It also provides security mechanisms.

Review Question

1. *Show how data flows through different layers in a TCP/IP network, calculate the number of times encapsulation and decapsulation take place.*

3.5 Wireless Local Area Network

- Wireless Local Area Network (WLAN) has data transfer speeds ranging from 1 to 54 Mbps. WLAN signal can be broadcast to cover an area ranging in size from a small office to a large campus. Most commonly, a WLAN access point provides access within a radius of 65 to 300 feet.
- IEEE 802.11 standard for WLANs.
- The 802.11 specifications were developed specifically for Wireless Local Area Networks(WLANs) by the IEEE and include four subsets of Ethernet-based protocol standards: 802.11, 802.11a, 802.11b, and 802.11g.

e e s

4

- 802.11 operated in the 2.4 GHz range and was the original specification of the 802.11 IEEE standards. This specification delivered 1 to 2 Mbps using a technology known as Phase-Shift Keying (PSK) modulation.
- 802.11a operates in the 5 - 6 GHz range with data rates commonly in the 6 Mbps, 12 Mbps, or 24 Mbps range. Because 802.11a uses the Orthogonal Frequency Division Multiplexing (OFDM) standard, data transfer rates can be as high as 54 Mbps.
- The 802.11b standard operates in the 2.4 GHz range with up to 11 Mbps data rates and is backward compatible with the 802.11 standard. 802.11b uses a technology known as Complementary Code Keying (CCK) modulation.
- 802.11g is the most recent IEEE 802.11 draft standard and operates in the 2.4 GHz range with data rates as high as 54 Mbps over a limited distance.
- Fig. 3.5.1 shows WLAN.
- IEEE 802.11 defines the Physical (PHY), Logical Link (LLC) and Media Access Control (MAC) Layers for a wireless LAN.
- Wireless Access Points (APs) is a small device that bridges wireless traffic to your network.

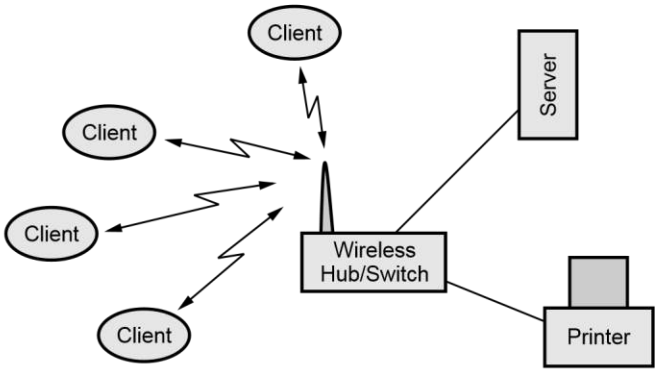


Fig. 3.5.1 Wireless LAN

3.5.1 Hidden and Exposed Terminal Problem

dden Terminal Problem

- Station B can hear both A and C, but A and C can't hear each other.
- ‡It happens when station C attempts to transmit while A is transmitting to B.
- ‡Station "A" is hidden from station C.

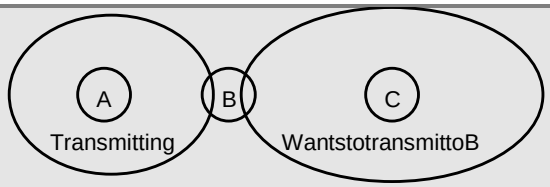


Fig. 3.5.2 Hidden station problem

- **xposed Terminal :**
  Happens when station B is transmitting to A when C attempts to transmit.
- ‡Assuming no interference effect, station C should defer transmitting only if it want to transmit to B.
  ‡Carrier sense provides information about potential collision at the sender, but not at the receiver.

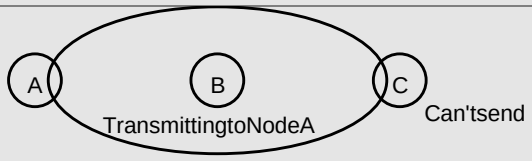


Fig. 3.5.3 Exposed station problem

- To avoid collision, sender senses the carrier before transmission. But collision occurs at the receiver not transmitter

e e s

5

3.5.2 Wireless LAN Protocols : MACA and MACAW

1. Multiple Access with Collision Avoidance (MACA)

- MACA protocol solved hidden, exposed terminal. Send Ready-to-Send **(RTS)** and Clear-to-Send **(CTS)** first. RTS, CTS helps determine who else is in range or busy (Collision avoidance).
- MACA uses exchange of two short messages : Request to Send (RTS), and Clear to Send (CTS).
- They are fixed size. When A wishes to transmit to B, it sends an RTS message. RTS message contains duration of proposed transmission
- If B knows that the channel is free, it responds with a CTS message. (CTS also contains duration of proposed communication).

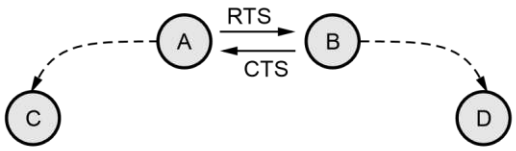


Fig. 3.5.4 MACA

- Any station that hears the RTS message, defers all communication for some time until the associated CTS message has been finished.
- A CTS message defers communication for the duration of the time indicated in the CTS message.
- When A is transmitting data, C can go ahead and access the channel. Node B's CTS message may not be heard by A. Station B found that the channel was already busy. RTS packet might collide.
- If A does not receive a CTS, it times-out and schedules the packet for retransmission. MACA uses the binary exponential back-off algorithm to select the retransmission time. B's CTS message collides at C.
- This would cause C to be unaware of the pending communication between nodes A and B.

2. Multiple Access with Collision Avoidance for Wireless LANs (MACAW)

- Receiver responds with acknowledgment (ACK) after data reception. Other nodes in receiver's range learn that chann l is availabl . Node hearing RTS, but not CTS do not know if transmission will occur.
- MACAW uses Data Sending (DS)packet, sent by sender after receiving CTS to inform such nodes of successful handshake

3.5.3 The IEEE 802.11 MAC Layer

- The Distributed Coordination Function (DCF): Fundamental, contention-based access method. Implemented for use with both ad hoc and *infrastructure* networks.
- DCF is based on Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). Receivers send an ACK if they successfully receive a packet, otherwise the transmitter resends.
- The Point Coordination Function (PCF) : Optional, contention-free access method usable only on infrastructure networks.
- Point Coordination Function (PCF) an optional extension to DCF that provides a time division duplexing capability to accommodate time bounded, connection-oriented services
- DCF utilizes an RTS-CTS exchange to reserve the channel prior to DATA transmission. Successful DATA transmission is followed by an ACK from the recipient.
- PCF requires one node to function as a polling master. This node is called the Point Coordinator (PC).
- Virtual carrier sense implemented using a Network Allocation Vector (NAV).
- The NAV at each node is set to indicate the remaining time before the medium is expected to become idle.
- The NAV is updated based on duration information contained in overheard messages.
- Actual carrier sense combines the NAV state, the node's transmission status and the physical carrier sense indication from the physical layer.

6

3.5.4 Carrier Sense Multiple Access with Collision Avoidance

- Wireless networks cannot use CSMA/CD in the MAC sublayer, since this requires the ability to receive and transmit at the same time - hence the use of CSMA/CA.
- IEEE 802.11 is based on CSMA/CA. Before a node transmits, it first senses the medium for activity. The node is allowed to transmit, if the medium is idle for at least a time period called the DCF inter-frame space (DIFS). Otherwise the device executes a back-off algorithm to defer transmission to a later time!
- This algorithm randomly selects a number of time slots to wait and stores this value in a back-off counter. For every time slot that passes without activity on the network, the counter is decremented and the device can attempt transmission when this counter reaches zero.
- If activity is detected before the counter reaches zero, the device waits until the channel has been idle for a period of DIFS before it continues to decrement the counter value.
- After a successful transmission, receiver device responds with an acknowledgment after waiting for a time period called the Short Interframe Space (SIFS).
- The value of SIFS is smaller than the value of DIFS to ensure that no other device accesses the channel before the receiver can transmit its acknowledgment

3.5.5 Advantages and Disadvantages of WLAN

Advantages of WLAN

1. Very flexible within the reception area
2. Ad-hoc networks without previous planning possible
3. No wiring difficulties (e.g. historic buildings, firewalls)
4. More robust against disasters like, e.g., earthquakes, fire - or users pulling a plug.

Disadvantages of WLAN

1. Typically very low bandwidth compared to wired networks (1-10 Mbit/s)
2. Many proprietary solutions, especially for higher bit-rates, standards take their time (e.g. IEEE 802.11)
3. Products have to follow many national restrictions if working wireless, it takes a vary long time to establish global solutions.

Review Question

1. *Discuss the collision avoidance technique used in wireless LAN. It it possible for collisions to still occur ? Explain.*

3.6 Ethernet

- Ethernet refers to the family of Local-Area Network (LAN) covered by the IEEE 802.3 standard that defines what is commonly known as the CSMA/CD protocol.
- Four data rates are currently defined for operation over optical fiber and twisted-pair cables : 10 Mbps -10Base-T Ethernet, 100 Mbps-Fast Ethernet, 1,000 Mbps-Gigabit Ethernet and 10,000 Mbps-10 Gigabit Ethernet.
- Fig. 3.6.1 shows Ethernet LAN.

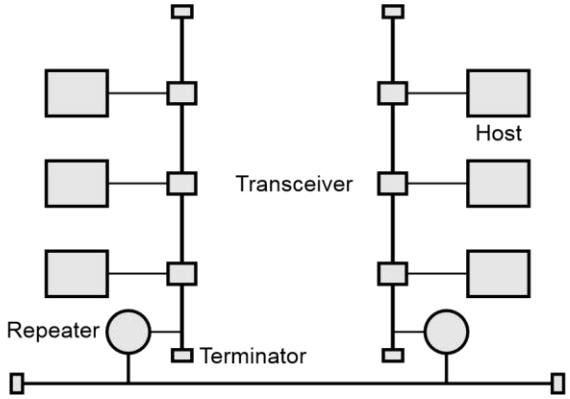


Fig. 3.6.1 Ethernet LAN

- Ethernet uses a communication concept called datagrams to get messages across the network. Ethernet uses a CSMA/CD multiple access algorithm.
- The most common Ethernet technologies today are 10Base2, which uses thin coaxial cable in a bus topology and has a transmission rate of 10 Mbps.

e e s

7

- Ethernet segment is implemented on a coaxial cable of upto 500 met rs. Multipl Ethernet egment joins together by repeaters.
- Ethernet uses Manchester encoding schemes. Any signal placed on the Ethernet by a host is broadcast over the entire network, i.e. signal is propagated in both directions.

MAC addresses

- Every device connected to an Ethernet network has a unique MAC address, assigned by the manufacturer of the network card. Its function is like that of an IP address, since it serves as a unique identifier that enables devices to talk to each other.
- However, the scope of a MAC address is limited to a broadcast domain, which is defined as all the computers connected together by wires, hubs, switches, and bridges, but not crossing routers or Internet gateways.
- MAC addresses are never used directly on the Internet, and are not transmitted across routers.
- Fig. 3.6.2 shows Ethernet frame format.

Preamble (7 Bytes)	Start Frame Delimiter (1 Byte)	Destination Address (6 Bytes)	Source Address (6 Bytes)	Length or Type (2 Bytes)	Data and Padding	CRC (4 Bytes)

Fig. 3.6.2 Ethernet frame format

1. **Preamble** allows the receiver to synchronize with the signal. It is a sequence of alternating 0 and 1.
2. **Start Frame Delimiter (SFD) :** The sequence 10101011, which indicates the actual start of the frame and enables the receiver to locate the first bit of the rest of the frame.
3. Both the **source and destination host address** are identified with a 48 bits address.
4. **Type** field serves as the de-multiplexing key, i.e. it identifies to which of possibly many higher level protocols this frame should be delivered.
5. **Data :** It is a minimum of 46 bytes and a maximum of 1500 bytes. The reason for minimum frame size is that the frame must be long enough to detect a collision.
6. **CRC :** This field contains error detection information.

- Ethernet is a bit oriented protocol. Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet.

3.6.1 Carrier Sense Multiple Access with Collision Detection

- When node has data to transmit, the node first listens to the cable to see if a carrier (signal) is being transmitted by another node. This may be achieved by monitoring whether a current is flowing in the cable.
- The individual bits are sent by encoding them with a 10 clock using Manchester encoding. Data is only sent when no carrier is observed (i.e. no current present) and the physical medium is therefore idle.
- Any node which does not need to transmit, listens to see if other nodes have started to transmit information to it.

e

8

- If two nodes simultaneously try transmit, then both could see an idle physical medium and both will conclude that no other node is currently using the medium. In this case, both will then decide to transmit and a collision will occur.
- The collision will result in the corruption of the frame being sent, which will subsequently be discarded by the receiver since a corrupted Ethernet frame will not have a valid 32-bit MAC CRC at the end.
- If two or more stations have messages to send at the same time and they are separated by significant distances on the bus/channel, each may begin transmitting at roughly the same time without being aware of the other station.
- The signals from each node will superimpose on the channel and is garbled beyond the decoding ability of the receiving station. This is termed as "collision".
- When there is data waiting to be sent, each transmitting node also monitors its own transmission. If it observes a collision, it stops transmission immediately and instead transmits a 32-bit jam sequence.
- The purpose of this sequence is to ensure that any other node which may currently be receiving this frame will receive the jam signal in place of the correct 32-bit MAC CRC, this causes the other receivers to discard the frame due to a CRC error.
- When two or more transmitting nodes each detect a corruption of their own data (i.e. a collision), each responds in the same way by transmitting the jam sequence.

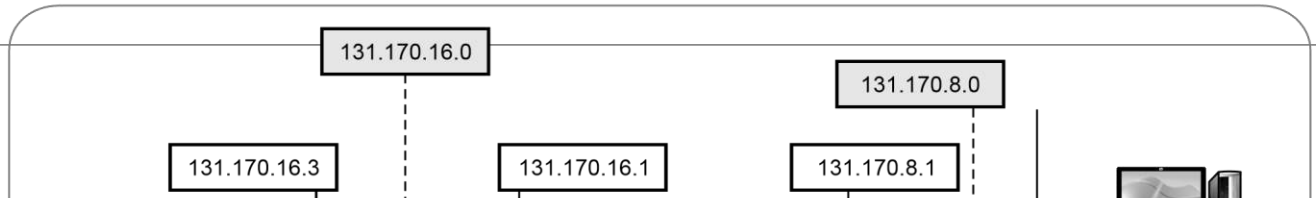
3.7 WiFi

- WiFi means wireless fidelity. It is a wireless technology that uses radio frequency to transmit data through the air. The standard for Wireless Local Area Networks (WLANs). It's actually IEEE 802.11, a family of standards. Wi-Fi is based on the 802.11 standard:802.11a, 802.11b and 802.11g.
- IEEE Created standard, but Wi-Fi Alliance certifies products. The Wireless Ethernet Compatibility Alliance started the Wi-Fi--wireless fidelity--certification program to ensure that equipment claiming 802.11 compliance was genuinely interoperable.
- WiFi systems are the half duplex shared media configurations, where all stations transmit and receive on the same radio channel.
- Wi-Fi combines concepts found in CSMA/CA and MACAW, but also offers features to preserve energy
- The developers of the 802.11 specifications develop a collision avoidance mechanism called the Distributed Control Function (DCF). According to DCF, a WiFi station will transmit only when the channel is clear. All transmissions are acknowledged, so if a station does not receive an acknowledgement, it assumes a collision occurred and retries

ISM Band

- ISM stands for industrial, scientific, and medical. ISM bands are set aside for equipment that is related to industrial or scientific processes or is used by medical equipment.
- Perhaps the most familiar ISM-band device is the microwave oven, which operates in the 2.4-GHz ISM band.
- The ISM bands are license-free, provided that devices are low-power. You don't need a license to set up and operate a wireless network.
- WLAN Architecture : **Ad-Hoc mode :** Peer-to-peer setup where clients can connect to each other directly. Generally not used for business networks.
- Mobile stations communicate to each other directly. It's set up for a special purpose and for a short period of time. For example, the participants of a meeting in a conference room may create an ad hoc network at the beginning of the meeting and dissolve it when the meeting ends.

- Wi-Fi network services are as follows :
  1. Distribution and integration
  2. Association, re-association, and disassociation
  3. Authentication and deauthentication
  4. Providing privacy
- **Distribution :** This service is used by mobile stations in an infrastructure network every time they send data. Once a frame has been accepted by an access point, it uses the distribution service to deliver the frame to its destination. Any communication that uses an access point travels through the distribution service, including communications between two mobile stations associated with the same access point.
- **Integration** is a service provided by the distribution system; it allows the connection of the distribution system to a non-IEEE 802.11 network. The integration function is specific to the distribution system used and therefore is not specified by 802.11, except in terms of the services it must offer.
- **Association :** Delivery of frames to mobile stations is made possible because mobile stations register, or associate, with access points. The distribution system can then use the registration information to determine which access point to use for any mobile station.
- **Reassociations :** When a mobile station moves between basic service areas within a single extended service area, it must evaluate signal strength and perhaps switch the access point with which it is associated. Reassociations are initiated by mobile stations when signal conditions indicate that a different association would be beneficial; they are never initiated by the access point. After the reassociation is complete, the distribution system updates its location records to reflect the reachability of the mobile station through a different access point.
- **Disassociation :** To terminate an existing association, stations may use the disassociation service. When stations invoke the disassociation service, any mobility data stored in the distribution system is remov d. Once disassociation is complete, it is as if the station is no longer attached to the network. Disassociation is a polite task to do during the station shutdown process. The MAC is, however, designed to accommodate stations that leave the network without formally disassociating.
- **Authetication/Deauthentication :** Physical security is a major component of a wired LAN security solution. Wired network's equipment can be locked inside offices. Wireless networks cannot offer the same level of physical security, however, and therefore must depend on additional authentication routines to ensure that users accessing the network are authorized to do so. Authentication is a necessary prerequisite to association because only authenticated users are authorized to use the network.
- **Deauthentication** terminates an authenticated relationship. Because authentication is needed before network use is authorized, a side effect of deauthentication is termination of any current association.
- Each 802.11a/b/g device can operate in one of four possible modes:
  1. Master mode (also called AP or infrastructure mode) is used to create a service that looks like a traditional access point.
  2. Managed mode is sometimes also referred to as client mode. Wireless cards in managed mode will join a network created by a master, and will automatically change their channel to match it.
  3. Ad-hoc mode creates a multipoint-to-multipoint network where there is no single master node or AP. In ad-hoc mode, each wireless card communicates directly with its neighbors.
  4. Monitor mode is used by some tools to passively listen to all radio traffic on a given channel. When in monitor mode, wireless cards transmit no data.

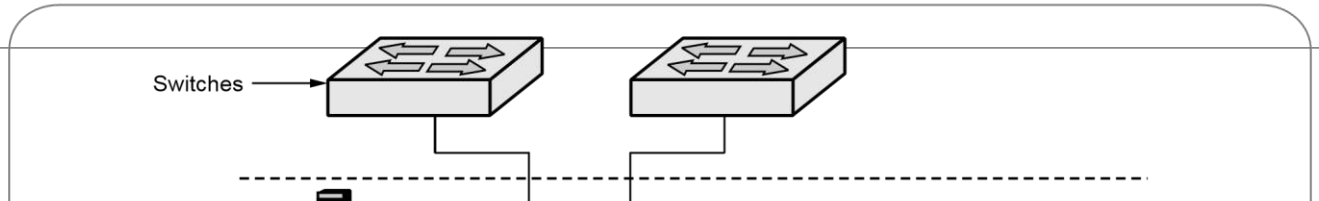

131.170.16.0
131.170.8.0
131.170.16.3
131.170.16.1
131.170.8.1

5. As each route is found, the packet is sent to the next router, called a "hop", and finally delivered to the destination host. If a route is not found, an error message is sent to the source host

daemons, and so continuously update the routing table.

- The routing algorithm is fundamental to dynamic routing. Whenever the topology of a network changes because of growth, reconfiguration, or

•


Switches

e

3

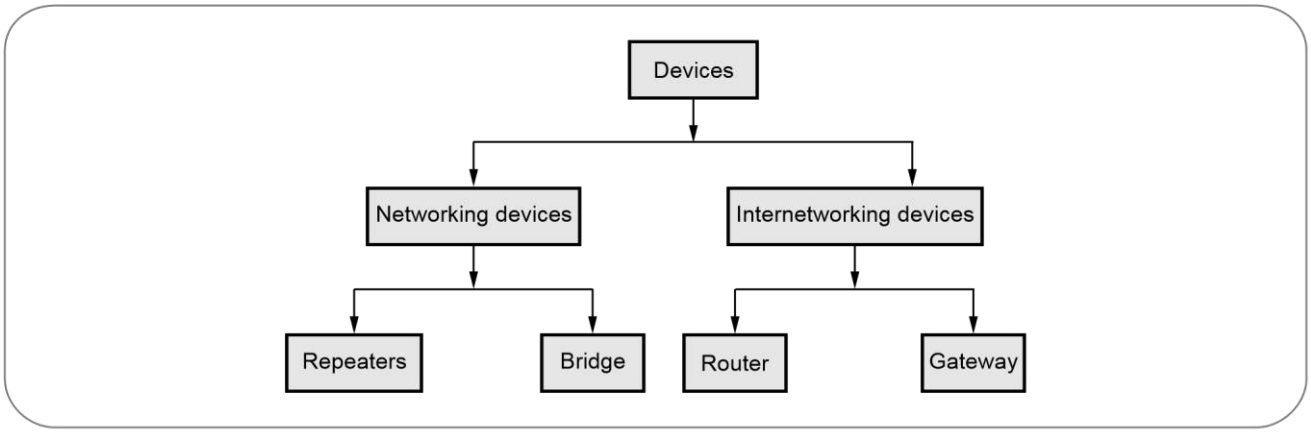


Fig. 3.10.1

3. Circuit Switching

- In circuit switching, a dedicated path is established. Data transmission is fast and interactive. Nodes need not have storage facility.
- Circuit switched communication system involves three phases: circuit establishment (setting up dedicated links between the source and destination); data transfer (transmitting the data between the source and destination); and circuit disconnect (removing the dedicated links).

3.9.1 Difference between Packet switching, Circuit Switching and Message Switching

Packet switching	Circuit switching	Message switching
Packet uses different path.	All the packet uses same path.	Packets are stored and forward
Full use of bandwidth	Circuit id dedicated to the call	Full use of bandwidth
Resource are shared	No sharing of resources	Resource are shared
No waste of bandwidth.	Waste of bandwidth is possible.	No waste of bandwidth.
Suitable for handling interactive traffic.	Not suitable for handling interactive traffic.	Not suitable for handling interactive traffic.
Congestion occurs for per packet.	Congestion occur for per minute	No congestion or very less congestion
No needs of end to end path before data transmission.	Needs an end to end path before the data transmission	No needs of end to end path before data transmission.

Review Question

1. *Compare contrast different types switching methodologies in network communication.*

3.10 Network Components

- The OSI reference model provides a simple representation of how data moves through a network. It can serve as a basis for understanding and characterizing an overall networking strategy. (Refer Fig. 3.10.1.)

3.10.1 Repeater

- Repeaters operate at the physical layer. It forward bits from one LAN segment to another. The basic purpose of a repeater is to extend the distance of LAN.
- A repeater is a network device that is used to regenerate or replicate signals that are weakened or distorted by transmission over long distances and through areas with high levels of electromagnetic interference (EMI).
- Repeaters do not have physical addresses on the network. It is not permitted to create a loop between two network segments by using two repeaters.
- Repeaters do not translate anything. A repeater does not actually connect two LANs; it connects two segments of the same LAN. A repeater is not a device that can connect two LANs of different protocols.
- Repeater is a simple way to solve the Ethernet distance limitations problem.
- **Types of repeaters :**
  1. Single port repeater
  2. Multiport repeater
  3. Smart repeater
  4. Optical repeater

4

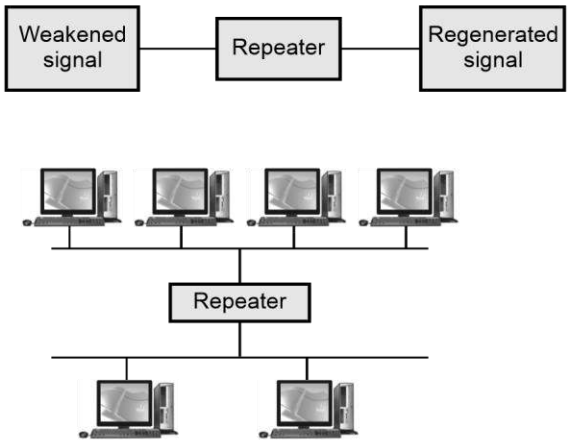


Fig. 3.10.2 Repeater

3.10.2 Bridge

- Bridges operate at the Data Link layer. It has a single input and single output port. A bridge extends the maximum distance of network by connecting separate network segment. A bridge simply passes on all the signals it receives.
- Bridges use MAC addresses to handle traffic flow. A bridge can also filter by MAC address, a feature that makes the bridge more attractive than a repeater. This style of filtering on Ethernet is called transparent bridging.
- Fig. 3.10.3 shows bridge.
- The interface between the bridge and each LAN segment is known as a port. Each LAN attached to a port is called a network segment.
- It operates on Ethernet frames, examining frame header and selectively forwarding frame based on its destination. Bridge isolates collision domains since it buffers frames. When frame is to be forwarded on segment, bridge uses CSMA/CD to access segment and transmit.
- If the frame is addressed to a MAC addr ss on the local side of the bridge, it is not forwarded to the other segment. MAC addresses on the other segment are forwarded. Bridges maintain a MAC address table for both segments they are connected to.
- Bridge performs data link functions such as error detection, frame formatting and frame routing.
- The bridge examines the source address in a frame and places this address in a routing table, to be used for future routing decisions. As traffic passes through the bridge, information about the computer addresses is then stored in the bridge's RAM. The bridge will then use this RAM to build a routing table based on source addresses.
- Bridges maintain filtering tables, implement filtering, learning and spanning tree algorithms. Bridges do not offer protection from broadcast storms.
- Bridges may be either local or remote. Local bridges connect multiple LAN segments within the same local area. Local bridges connect to local transmission media, particularly network backbones. Remote bridges are also known as wide-area bridges. Remote bridges connect multiple LAN segments in different areas
- 802 Bridges : IEEE 802.1 committee has standardized concepts for interconnecting 802-type LANs, they are :
  1. Fixed-routing bridges
  2. Transparent or spanning tree bridges
  3. Source routing bridges
  4. Remote bridges

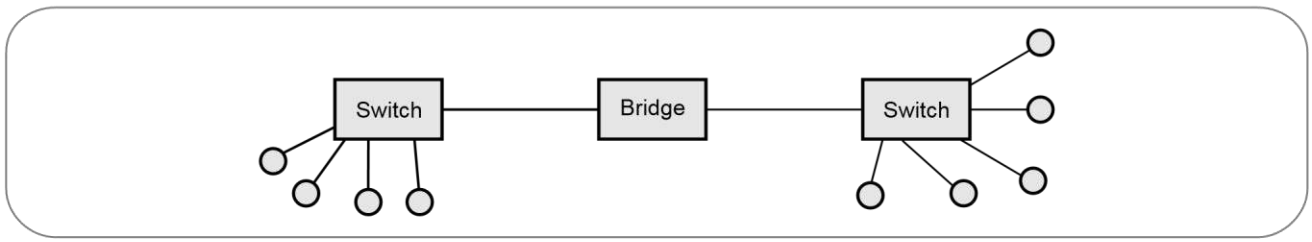


Fig. 3.10.3 Bridge

e

5

Advantages of Bridge

1. Simple to use and install.
2. Bridges are transparent to users.
3. Additional software is not required.
4. Bridges form single logical networks.

Disadvantages of Bridge

1. Bridge suffers from broadcast storms.
2. Fault isolation is not provided.

3.10.3 Router

- Routers operate at the network layer of the OSI Model. They connect networks into internetworks that are physically unified, but in which each network retains its identity as a separate network environment. A router's primary purpose is to find the best path from one network environment to another and forward packets between them.
- The key feature of a router is to determine the shortest path to destination. A router forwards packet by examining protocol address at network layer, look up the address in the routing table, then forward the packet to the next hop. Router uses one or more routing algorithms to calculate the best path through an internetwork.
- Fig. 3.10.4 shows router. A router is a hardware component used to interconnect networks. A router has interfaces on multiple networks.
- Routers often incorporate firewall functions. A router accepts an outgoing packet, removes any LAN headers and trailers, and encapsulates the necessary WAN headers and trailers. Because a router has to make wide area network routing decisions, the router has to dig down into the network layer of the packet to retrieve th network destination address.
- It monitors network traffic and report statistics to a management information base.

Router Features and Functions

- **Static routing :** Technique in which a network administrator programs a router to use a specified paths between nodes.
- **Dynamic routing :** Automatically calculates best path between nodes and accumulates this information in a routing table.
- **Hop :** Term used in networking to describe each trip data take from one connectivity device to another.

3.10.4 Gateway

- Gateway is combination of networking hardware and software that connects two dissimilar kinds of networks. A gateway is protocol converter. It operates in all seven layers of the OSI model.
- A gateway can accept a packet formatted for one protocol (e.g. TCP/IP) and convert it to a packet formatted for another protocol (e.g. Apple Talk) before forwarding it.
- The gateway must adjust the data rate, size and data format. Gateway is generally software installed within a router.
- To process the data, the gateway decapsulates incoming data through the networks complete protocol stack. The outgoing data is encapsulates in the complete protocol stack of the other network to allow transmission.

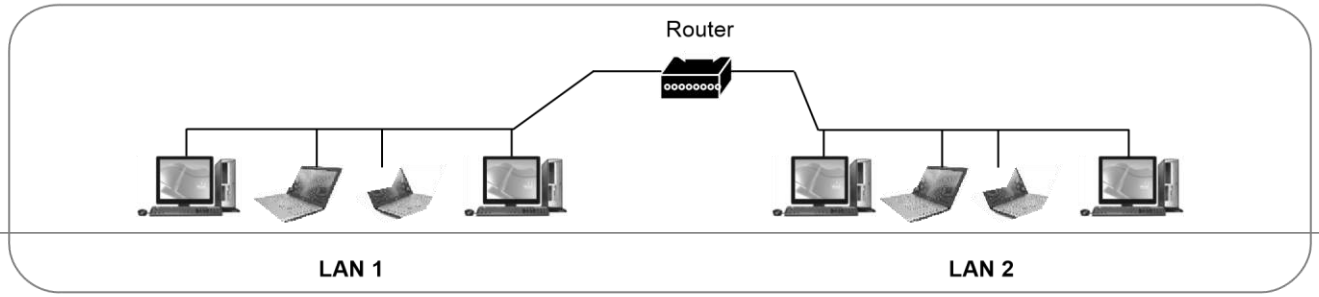


Fig. 3.10.4 Router

6

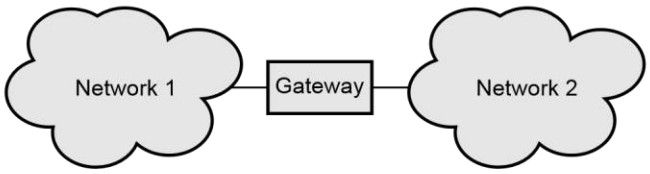


Fig. 3.10.5 Gateway

- A gateway is required to convert data packets from one protocol format to another before forwarding it, as it connects two dissimilar networks.
- **Popular types of gateways include :**
  1) E-mail gateways 2) IBM host gateways
  3) Internet gateways 4) LAN gateways

3.10.5 NIC

- The network interface cards (NICs) are also called as Network Adapter. The primary function of NIC is to allow the computer to communicate on the network. It supports transmitting, receiving and controlling traffic with other computers on the network. NIC operates at physical layer of OSI model.
- A committee of the Institute of Electrical and Electronics Engineers (IEEE) assigns blocks of addresses to each NIC manufacturer. The manufacturers hardwire these addresses into chips on the card by a process known as "burning" the address into the card. With this process, each NIC and therefore each computer has a unique address on a network. This address is called physical address. Its size is 48 bits.
- Physical address is also called Media Access Control (MAC) address.

3.10.6 Hub

- A hub joins multiple computers together to form a single network segment. On this network segment, all computers can communicate directly with each other.
- Direct data packets to all devices connected to the hub. Hub shared bandwidth between all device.
- Each connected LAN referred to as LAN segment. Hubs do not isolate collision domains; node may collide with any node residing at any s gment in LAN.
- Fig. 3.10.6 shows hub device.

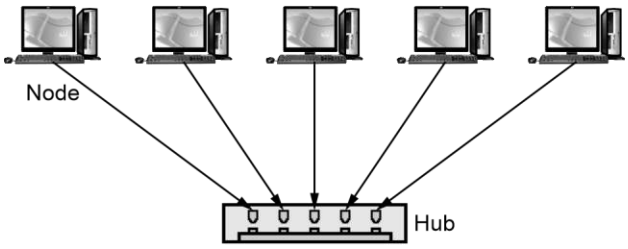


Fig. 3.10.6 Hub

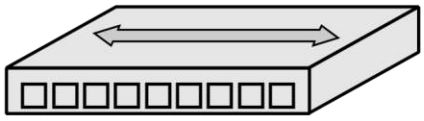


Fig. 3.10.7 Hub icons

- Hubs work at the OSI physical layer to regenerate the network's signal and resend them to other segments. Hubs can also be referred to as repeaters
- Primitive hub can be viewed as a multiport repeater. It regenerates data and broadcasts them to all ports.
- Hubs will always forward every frame out every port, excluding the port originating the frame.
- If any two devices connected to a hub send a frame simultaneously, a collision will occur. Thus, all ports on a hub belong to the same collision domain. A **collision domain** is simply defined as any physical segment where a collision can occur.
- A **broadcast domain** is a logical segmentation of a network, dictating how far a broadcast or multicast frame can propagate.
- Hubs also belong to only one broadcast domain, a hub will forward both broadcasts and multicasts out every port but the originating port.
- A passive hub is just a connector. A passive hub simply combines the signals of network segments. There is no signal regeneration.
- An active hub is actually a multiport repeater. An active hub is that regenerates or amplifies the signals.

e

7

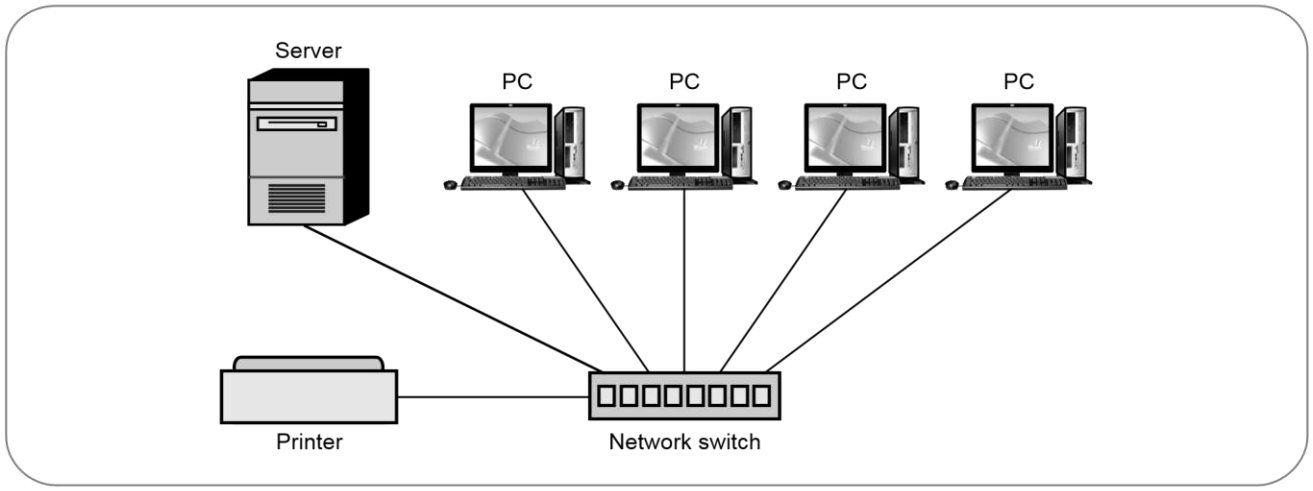


Fig. 3.10.8 switch

Advantages of Hub :

- The most economic way of expanding networks.
- Simple, inexpensive device

Hub limitations

- Cannot connect different Ethernet types
- Hubs do not isolate collision domains

3.10.7 Switch

- A switch is a multiport, Data Link Layer device. It "learns" Media Access Control addresses and stores them in an internal lookup table.
- Temporary, switched paths are created between the frame's originator and its intended recipient, and the frames are forwarded along the temporary path
- Each port, and the device to which it connects, has its own dedicated bandwidth. Fig. 3.10.8 shows switch.

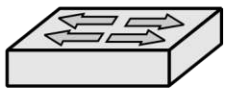


Fig. 3.10.9 Switch icon

- Switches divide a network into several isolated channels. Packets sending from one channel will not go to another if not specify. Each channel has its own capacity and need not be shared with other channels. Fig. 3.10.10 shows switch with channel.
- Use twisted pair or fiber optic cabling, using separate conductors for sending and receiving data.
- Switches usually have a higher port-density, and can perform forwarding decisions at wire speed, due to specialized hardware circuits called Application-Specific Integrated Circuits.
- There are two basic types of switches : **managed and unmanaged.**

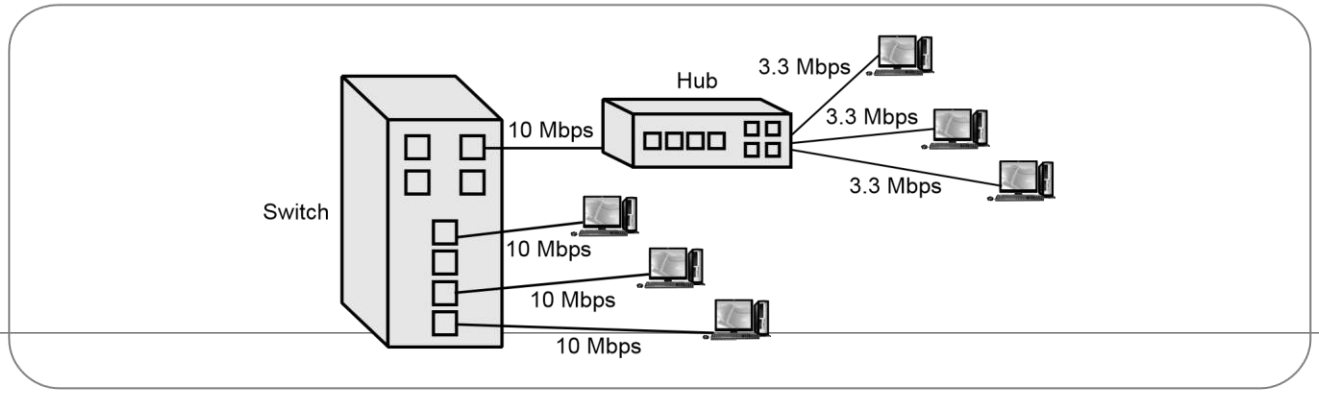


Fig. 3.10.10 Switch with channel

8

- An unmanaged switch works out of the box and does not allow you to make changes. Home-networking equipment typically offers unmanaged switches.
- A managed switch allows you access to program it. This provides greater flexibility to your networking basics because the switch can be monitored and adjusted locally or remotely to give you control over network traffic, and who has access to your network.
- -A switch behaves much like a hub when first powered on. The switch will flood every frame, including unicasts, out every port but the originating port. The switch will then build the MAC-address table by examining the source MAC address of each frame.
- -Switches create more collision domains, which results in fewer collisions. Switches belong to only one broadcast domain

Advantages of Switches

- Reduce the possibility of collision
- Each channel has its own network capacity
- Switches increase available network bandwidth
- Switches reduce the workload on individual computers
- Switches increase network performance

Limitations of Switches

- Device cannot detect collision when buffer full.
- Some higher level protocols do not detect error e.g. UDP

3.10.8 Difference between Hub and Switch

Hub	Switch
Hub operates at physical layer of OSI model	Switch operates at data link layer of OSI model
Hub is a broadcasting device	Switch is a point to point communication device
Hub simply broadcasts the incoming packet.	Switch uses switching table to find out the correct destination
All ports on a hub belong to the same collision domain	Switches create more collision domains, which results in fewer collisions

Two Marks Questions with Answers

Q.1 *List four commonly used topologies to set up a LAN. Suggest a MAC scheme for each.*

Ans. : Topologies are star, ring, bus and mesh. CSMA/CD is used for all topology.

Q.2 *Differentiate between a repeater, a hub and a switch.*

Ans. : Repeater and hub works at physical layer and switch works at data link layer. Repeater is used to extend the LAN. Hub and switch is used to set up a LAN. Repeater regenerate the signal. Hub is broadcasting device. Switch is point-to-point device.

Q.3 *Give the name of a protocol used in each layer of the network.*

Ans. : Application layer uses HTTP, Transport layer uses TCP and UDP. Network layer uses ARP, RARP. Data link layer uses PPP, IEEE 802.2 and physical layer uses Ethernet.

Q.4 *What is meant by encapsulation and decapsulation ?*

Ans. : Encapsulation is used to refer to the process of each layer at the sending computer adding its own header information. Decapsulation is the reverse process of encapsulation, wherein each layer at the receiving computer, interprets the header information sent by its peer layer, takes the required action based on the information and finally removes the header, before passing on the data to the next higher layer.

Q.5 *Define physical and logical topology.*

Ans. : Physical topology defines how the nodes of the network are physically connected. Logical topology dedicated connections between certain selected source-destination pairs using the underlying physical topology.

Q.6 *What is the difference between a passive and an active hub ?*

Ans. : An active hub contains a repeater that regenerates the received bit patterns before sending them out. A passive hub provides a simple physical connection between the attached devices.

Q.7 *Define LAN.*

Ans. : A Local Area Network (LAN) is a data communication system that allows number of independent devices to communicate directly with each other in a limited geographic area.

Q.8 *What are the responsibilities of network layer ?*

Ans. : The network layer is responsible for the source-to-destination delivery of packet across multiple network links

Q.9 *What is Multiple Access ?*

Ans. : If the physical links are shared by more than two nodes, it is said to be Multiple Access.

Q.10 *What is the purpose of Physical layer ?*

Ans. : The physical layer coordinates the functions required to trAns. :mit a bit stream over a physical medium.

Q.11 *What is CSMA/CD ?*

Ans. : Carrier Sense Multiple Access with Collision Detection is a protocol used to sense whether a medium is busy before transmission and it also has the ability to detect whether the packets has collided with another

Q.12 *What is the maximum length of a datagram ?*

Ans. : The maximum length of a datagram is 65,535 bytes.

Q.13 *What is Fragmentation ?*

Ans. : Fragmentation is the division of a datagram into smaller units to accommodate the MTU of a data link protocol.

Q.14 *What is classless addressing ?*

Ans. : In classless addressing variable length blocks are assigned that belong to no class. In this, the entire address space is divided into blocks of different sizes. An organization is granted a block suitable for its purposes.

Q.15 *What is packet switching ?*

Ans. : Packet switching is the dividing of messages into packets before they are sent, transmitting each packet individually, and then reassembling them into the original message once all of them have arrived at the intended destination.

Q.16 *What is data plane in network layer ?*

Ans. : The data plane contains the protocols and mechanisms that allow hosts and routers to exchange packets carrying user data.

Q.17 *What is control plane in network layer ?*

Ans. : The control plane contains the protocols and mechanisms that enable routers to efficiently learn how to forward packets towards their final destination.

Q.18 *What do you mean by TCP ?*

Ans. : TCP guarantees the reliable, in order delivery of a stream of bytes. It is a full-duplex protocol, meaning that each TCP connection supports a pair of byte streams.

Q.19 *Explain the three types of addresses in TCP/IP.*

Ans. : Three types of addresses are used by systems using the TCP/IP protocol : the physical address, IP address and the port address

Q.20 *Define UDP.*

Ans. : User datagram protocol is a Unreliable, connectionless protocol, used along with the IP protocol.

Q.21 *How many bits does IPv6 address contain ? How is it different from IPv4 ?*

30

Ans. :

Sr. No.	IPv4	IPv6
1.	Header size is 32 bits	Header size is 128 bits.
2.	It cannot support auto configuration	Supports auto configuration
3.	Cannot support real time application.	Supports real time application
4.	No security at network layer.	Provides security at network layer.
5.	Throughput and delay is more.	Throughput and delay is less

Q.22 *Differentiate WiFi and WiMax ?*

Ans. :

Parameters	WiFi	WiMax
IEEE standard	802.11	802.16
Typical link length	100 m	10 km
Typical bandwidth	54 Mbps (shared)	70 Mbps (shared)
Typical use	Link a notebook computer to a wired base	Link a building to a wired tower
Wired technology analogy	Ethernet	Co-axial cable
Transport technology support	Wi-Fi supported these technology	ATM, IPv4 and IPv6
Medium access protocol	Wi-Fi uses the CSMA/CA	The MAC layer in WiMAX has been designed to scale from one to up 100s users within one RF channel.
Quality of service	QoS was not considered in the early stage of its implementation.	The base station assigns a QoS class to each connection
Spectrum	Unlicenced	Licensed and unlicenced
Duplexing	TDD	TDD and FDD

❑❑❑

UNIT - V

5 Application Essentials

Syllabus

Creation of simple interactive applications - Simple database applications - Multimedia applications - Design and development of information systems - Personal Information System - Information retrieval system - Social networking applications.

Contents

e

2

5.1 Creation of Simple Interactive Applications

Definition of Interactive Application : An interactive application is a collection of objects intended for performing certain task when user triggers the command.

- The non interactive applications operate without human involvement. For example - Compiler and batch processing applications are non interactive applications. The compiler is a program that converts the high level programs(written in C, C++,PACAL) in the machine language. Similarly the Batch processing is the execution of a series of jobs in a program on a computer without manual intervention.
- The interactive application share a Graphical User Interface(GUI).
- The interactive applications consists of forms on which the components such as buttons, text fields, radio buttons are present.
- In the interactive applications one page may get linked with other pages dynamically. Especially in case of interactive web applications this linking of the pages is through hyperlinks. This linking is based on user input.
- Typical examples of interactive web applications are - online course registration system, online shopping system and so on.

5.1.1 Advantages of Creating Interactive Applications

- It increase the **engagement of users** in the collaborative productions
- Instead of spending lot of time on one way presentation being **shared** across a PowerPoint presentation, an interactive application allows the user to engage with the information used. Files can be easily shared, accessed, edited and saved.
- The interactive applications boosts the communication. The information can be shared using email, print and files. The **intention** of the user to handle that application can be **identified** by the interactive sessions he makes with the application. This h lps in **boosting** the **communication.**
- The documents can be annotated **effective changes** can be made while handling the interactive application. The tools such as 3D modelling, hyperlinks, video links and other applications can be embedded within an interactive application.
- The **interconnectivity** among the interactive applications allows the greater availability. Users can connect to interactive applications using iOS and Android smart devices.
- The **information** can be **made available** to the fingertips by using touch screen technologies. This helps a novice or illiterate user to handle that application with ease and comfort.

5.1.2 User's Perspective about the Interactive Applications

- The users use the interactive application for getting required information. The users always want that the desired information must be neatly arranged on the application. They never prefer chaotic information.
- Poor information architectures make the user to feel frustrated, and confused.
- There are two modes of browsing the information - **known item searching** and **casual browsing.**
  - The **known item searching** is done by the users who know what they are looking for. They know what is the meaning of particular label, how to proceed for getting particular information and from where to leave. They just quickly find the required information and leave.
  - Another category of user make use of **casual browsing.** Such type of users do not know what they are looking for. Sometimes they do not know anything about the product or the services on the corresponding website. They just get the knowledge of the product/service present on the website and leave. Since a **casual approach of searching** is adopted in viewing the information this kind of browsing is called casual browsing.

e

3

5.1.3 Producer's Perspective about the Interactive Applications

- Cost is a major issue while building the information architecture from producer's perspective. One can not predict the exact cost of the information architect. However, depending upon the goals and nature of the application, the investment required for building the information architecture can be predicted. The information architecture must be well designed so that the cost can be negotiated accordingly.
- While building the interactive application architecture by using producer's perspective
  1. The organizational goals and vision must be clear to information architect.
  2. The decision of the contents to be placed on the main page should be taken prior to implementation of the architecture.
  3. The information architecture must be kept away from organizational politics.
  4. The site should be designed so that the intended audience will get the satisfaction of using it.
  5. All the controversial issues must be resolved during the design process, before the application gets built.
- If all the above discussed issues are not solved properly then user will leave bad memories of the application and will not prefer to use such application.

5.1.4 Steps for Creating Simple Interactive Web Applications

Following Steps are followed while creating interactive web applications

1. Understanding Data Items and the Data Dictionary

- Data item identifies the unit of information. It defines how the item can be used.
- A **data dictionary** is a collection of descriptions of the data objects or items in a data model. The data dictionary is **dynamic,** any changes in the data item are effective immediat ly for all applications that include the data item.
- Applications access the data dictionary at runtime and immediately reflect modifications to data item.
- The data dictionary is created for modeling the data items in the interactive applications.

2. Understanding the Table Design

- A relational database table stores the data that an application uses in columns and rows. Each column is a data item, and each row is a record.
- One or more tables can be used in an application.
- To create a table, the required data items are selected. These data items must already exist in the data dictionary. Then assign the key fields as indices for retrieving and updating the data belonging to the tables.
- Various operations that can be performed on these database tables are - create, insert record, delete particular record, and update some record.

3. Understanding Business View Design

- A business view is a selection of data items from one or more tables. After you create a table, use Business View Design to choose only the data items that are required for your application.
- Use appropriate SQL statements to retrieve data from any of the supported database.
- After you define a business view, you can create a form that updates data in an interactive application.

4. Understanding Form Design

- Form design is an important part of the interactive application.
- Applications are composed of forms, and a form is the interface between a user and a table. This interface should present the data logically and contain the functions that are necessary to enter and manipulate data.
- The form design task can be accomplished by placing various components such as text boxes, Labels, Buttons, Check buttons, radio buttons and so on the form at appropriate locations.

e

4

5. Understanding Report Design

- Report Design is used to present the business data stored in the database. Report is basically some kind of template.
- The data is typically presented using batch applications that access the data through business views.
- Each report is comprised of sections, which are the building blocks of all reports. Within the template, you can add, hide, remove, and rearrange sections as needed.
- One can create variations of reports in the interactive applications.

6. Understanding Data Structure Design

- Data structures are a key element of any programming language or environment. A data structure is a list of parameters that passes data among applications and tables or forms.

7. Understanding Event Rules Design

- Event are the activities that occur on the form of an interactive application. Event can be initiated by user or an application.
- Event rules are logic statements that you can create and attach to events.
- The design of event rules is essential to apply the business logic to the interactive applications.
- The event rules can be created for following purposes -
  1. Perform mathematical calculations
  2. Pass data from one field in the form to another field in another form.
  3. Interconnect two forms.
  4. Hide and display the controls using system functions.
  5. Assign the value or an expression to particular field.
  6. Creation of variables or programmer defined field at run time.
  7. Process table input and output, validate data and retrieve record.
- The areas at which th event rules are applicable are -
  1. **Controls :** Controls are reusable components that can be placed on the form. For example push buttons, Radio buttons, text boxes and so on. The event can be triggered from these controls. These events can be handled by performing appropriate actions.
  2. **Form Processing :** Form processing means application of business logic associated with each form. Form processing depends on the occurrence of specific events.
  3. **Table :** You can create database triggers, or rules that you attach to a table by using Table Design Event Rules. The logic that is attached to a table is run whenever any application initiates that database event.

8. Understanding System Functions

System Functions are procedures provided by the tool and are usually specific to the type of component being used. For example there are system functions to hide and show fields on an application

5.2 Simple Database Applications

Definition : Database is an organized collection of data. A **database management system (DBMS)** is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data.

Examples :

Well known DBMS software are - MS, ACCESS,Oracle, MySQL, Microsoft SQL Server, Sybase and so on.

Characteristics of Database Applications

Various characteristics of Database management applications are -

1. **Consistency :** DBMS provide greater consistency to the forms of data storage.
2. **Support for Query Language :** DBMS is equipped with query language which makes it

more efficient to retrieve and manipulate the data.

3. **Multiuser Environment :** DBMS allows multiuser environment and allows them to access and manipulate data in parallel without creating any conflicts.
4. **Less Data Redundancy :** DBMS offers less data redundancy.
5. **Relationship among data :** DBMS allows to establish relationship among the data.
6. **Security :** DBMS allows different levels of security features to data.

Advantages of DBMS

There are various advantages of adopting database approach in application system. These are

1. **Reduced Data redundancy**

   The database approach removes the redundancy by integrating the files. But this approach can not completely eliminate redundancy completely but can control the data redundancy.

2. **Data Consistency**

   In this approach all copies of data are kept consistent.

3. **Sharing of data**

   The centralized stored database can be accessed by multiple users or application programs.

4. **Centralized database**

   Data is stored in a single repository and an authorized access to this data is allowed. Only the administrator can give appropriate access rights to the authentic users.

5. **Data Integrity**

   The data integrity provides validity and consistency of stored data.

6. **Improved Security**

   The database approach prevents unauthorized access to data by means of user name, password, and access rights.

7. **Use of standards**

   Certain standards can be enforced to database approach for data formats, naming conventions, documentation standards, access rules and so on.

8. **Backup and recovery**

   The backup and recovery actions can be taken in its current consistent state from inconsistent state.

9. **Increased productivity**

   The database approach allows the programmer to concentrate on specific functionality required by the user.

10. **Increased Concurrency**

    The database approach handles the concurrent data effectively. The loss of information or integrity is avoided in this approach.

11. **Improved maintenance**

    As database approach provide the data independence it simplifies the database application maintenance.

Disadvantages of DBMS

Although there are large number of advantages of Database approach, following are some disadvantages of this approach

1. **Complexity :**

   The database management system is complex to implement

2. **Size :**

   The DBMS consumes large amount of storage space.

3. **Cost :**

   The multi-user database management system is ver y expensive. Also the maintenance cost is involved in database management systems.

5.2.1 Data Models

The data model in Database Applications describe the logical structure of database, relationship between the database stored in database and various constraints on data.

Importance of Data Model

1. End users have different view for data.
2. Data model organizes data for different users.

There are three types of data models that are used commonly -

1. Hierarchical Model :

In this model each entity has only one parent but can have several children. At the top of hierarchy there is only one node called **root**. Refer Fig. 5.2.1.

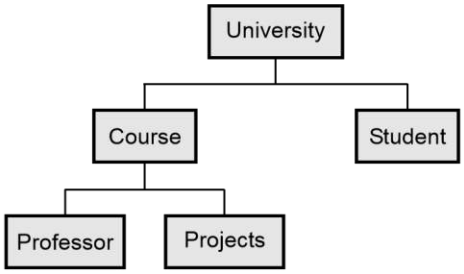


Fig. 5.2.1 Hierarchical Model

This model represents the relationship in 1:N types. That means one university can have multiple courses. One course can have multiple projects and so on.

Advantage

1. This model groups the data into tables and define the relationship between the tables.

Disadvantages

1. For searching any data, we have to start form the root and move downwards and visit each child node. Thus traversing through each node is required.
2. For addition of some information about child node, sometimes the parent information need to be modified.
3. It fails to handle many to many relationship(M:N) efficiently.

   It can cause duplication and data redundancy.

2. Network Model :

This is enhanced version of hierarchical model. It overcome the drawback of hierarchical model. It helps to address M:N relationship. That means, this model is not having single parent concept. Any child in this model can have multiple parents. Refer Fig. 5.2.2.

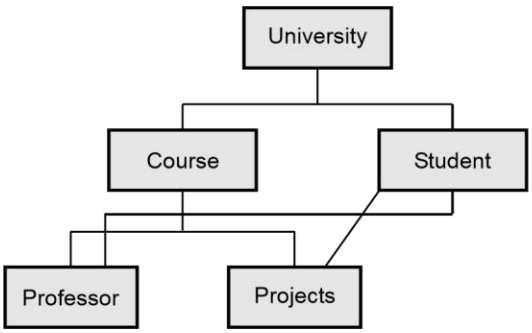


Fig. 5.2.2 Network Model

The **main difference** between network model and hierarchical model is to allow many to many relationship.

Advantages

1. **Capability to handle more Relationships :** Since the network model allows many to many relationship, it helps in modelling the real life situations.
2. **Ease of data access :** The data access is easier and flexible than hierarchical model.
3. **Data Integrity :** In network model every member is associated with some other member in the model.
4. **Conformance to Standards :** The network model structure can be designed as per the standards.

Disadvantages

1. **Complex to implement :** For all the records the pointers need to be maintained, hence the database structure becomes complex.
2. **Complicated Operations :** The simple operations such as insertion, deletion and modification becomes complex due to adjustment of multiple pointer.
3. **Difficult to change structure :** The structural changes are difficult.

3. Relational Model

It is developed by Codd in 1970. In relational model, the data is stored in the form of tables. The database systems based on it are called relational database systems.

7

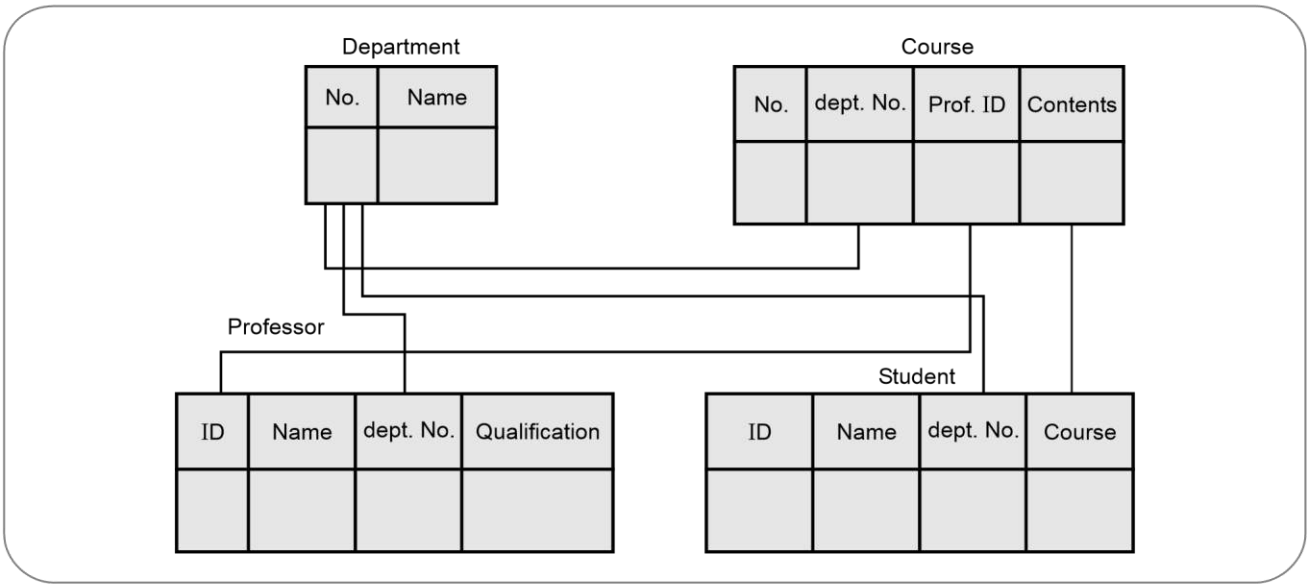


Fig. 5.2.3 Relational Model

These tables are related with each other. Refer Fig. 5.2.3.

Advantages

1. The database design is simple to implement and manage.
2. In this model, the table is an entity which is primarily used. Hence insertion, deletion and retrieval of data becomes easy.
3. The data can be easily linked with other table, using the table attributes.
4. It facilitates the support for SQL languages.
5. It ensures the structural independence.
6. Even after creation of database, new categories of data can be inserted without making changes in the existing application.

Disadvantages

1. It has substantial hardware and software overhead.
2. It can facilitate to poor design and implementation.
3. It can not handle the data in the form of images, audio or video.

4. E-R Model :

This model is basically used to design the relational database systems. The primary purpose of E-R Model is to show the relationship between various data objects.

- The object relationship pair can be graphically represented by a diagram called **Entity Relationship Diagram(ERD)**.
- The ERD was originally proposed by Peter Chen for design of relational database systems.
- Various **components of ERD** are -

Entity

- Drawn as a rectangle.
- An entity is an object that exists and is distinguishable.
- Similar to a record in a programming language with attributes.

Relationship

- Drawn as a diamond.
- An association among several entities.
- Relationships may have attributes.
- Relationships have cardinality (e.g., one-to-many).

Attribute

- Drawn as ellipses.

- Similar to record fields in a programming language.
- Each attribute has a set of permitted values, called the domain.
- Primary key attributes may be underlined.

The typical structure of ER-diagram is illustrated as follows.

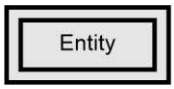
Notations used in ER diagram

Entity

It is an object and is distinguishable it is similar to record.

Weak entity

When this entity is dependant upon some another entity then it is called weak entity.



Attribute

The attributes are properties or characteristics of an entity.

Derived attribute

It is a kind of attribute which is based on another attribute.

Key attribute

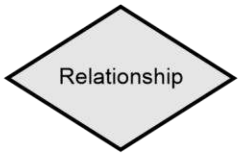
A key attribute is an unique attribute representing distinguishing characteristic of entity. Typically primary key of record is a key attribute.

Multivalued attribute

A multivalued attribute have more than one value.

Relationship

When two entities share some information then it is denoted by relationship.



Notations to show cardinality

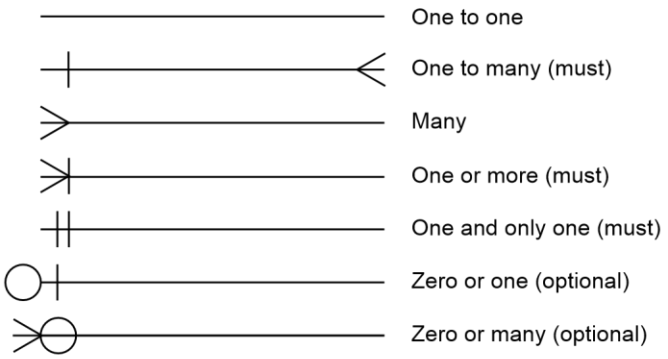


Fig. 5.2.4 Cardinality

Ex. 5.2.1 *Draw an ER diagram for the relationship of teacher and courses. Also specify the association, cardinality and modality.*

Solution :

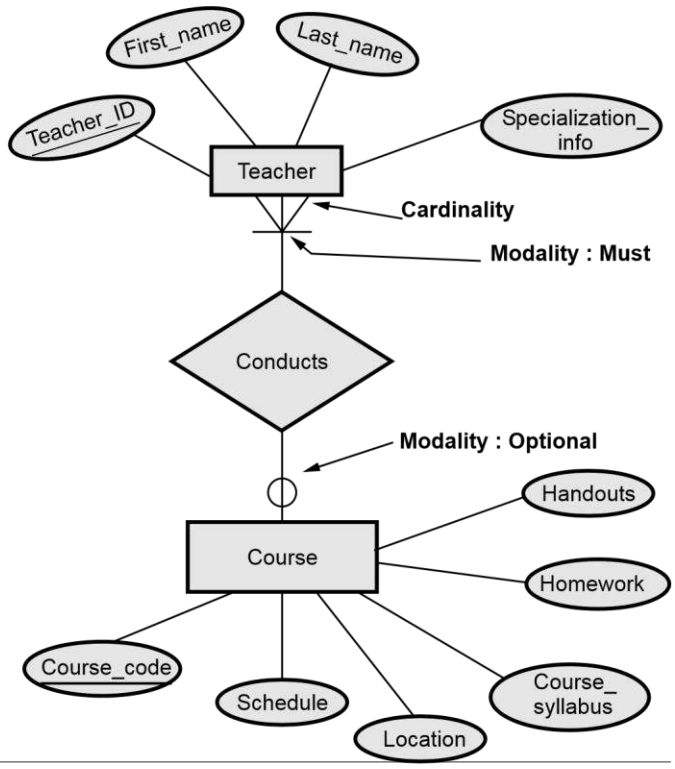


Fig. 5.2.5

Association

In the above ER diagram, a relatiohship **conducts** is introduced. "Teacher" is associated with "Course" by conducting it.

Cardinality

Many Teachers can conduct the single course.

Modality

For conduction of a course, there must be a Teacher.

There may a situation that a Teacher is not conducting any course.

Advantages

1. **Simple to design :** Conceptually creating ER model is conceptually very simple when one knows the various types of entities, attributes and relationships among them.
2. **Visual Representation :** This model gives clear graphical and diagrammatical representation of various entities, attributes and the relationships.
3. **Easy Conversion :** Converting an ER Model to tables is very simple. Also ER model can be easily converted to hierarchical model, network model, or relational model.

Disadvantages

1. **Limited Specification :** Using ER Model only binary relationship can be represented. Thus the model represents minimum specification.
2. **For High Level Design :** This model is popularly used for high level design only.
3. **No Industry Standard :** There is no industry standard notation for developing ER Model.

5.2.2 Architecture of Database Systems

- The database system architecture represents the structure and layout of the data stored in it.The architecture of database system can be from single tier to multitier.
- The multi-tiered system divides the whole system into multiple modules. Each of this individual module can be independently altered or modified.

The most commonly used architecture is 3-tier architecture. Refer Fig. 5.2.6.

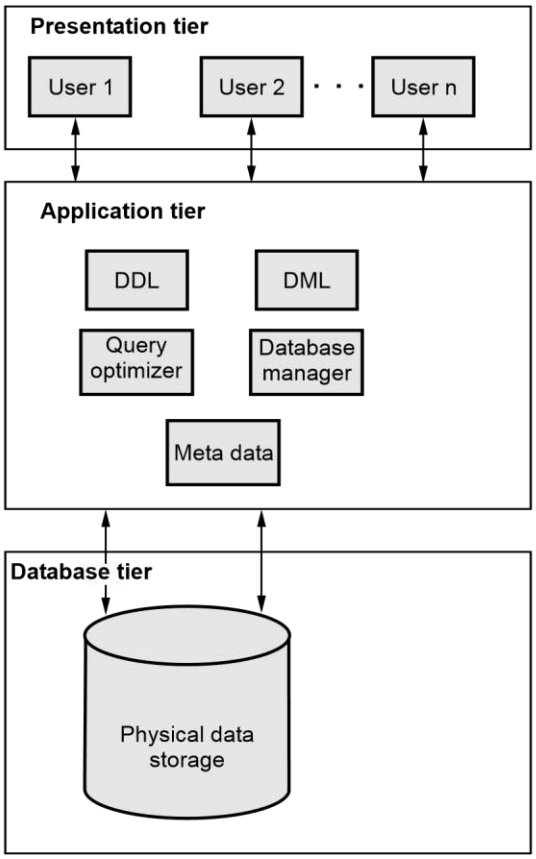


Fig. 5.2.6 Architecture of DB system

Database Tier :

- This is the layer at which the actual database resides.
- In this layer, all the tables, their mappings and the actual data present. When you save you details from the front end, it will be inserted into the respective tables in the database layer, by using the programs in the application layer.
- When the user wants to retrieve the data from this database the database layer fires the queries to get the data from the tables present in the database.

Application Tier :

- This layer sends the requests made by the users of presentation tier to the database tier and returns the response from database tier to presentation tier.

- This layer has all the business logics like validation, calculations and manipulations of data. If this layer sees that the request is invalid, it sends back the message to presentation layer.
- Hence, the application layer sits in the middle and acts as a mediator between the users and the database.

Presentation Tier :

- This layer is made up of the users who uses the database.
- These users have no knowledge of underlying database. At this layer multiple views of the database are provided.

5.2.3 Data Independence

Definition : Data independence is an ability by which one can change the data at one level without affecting the data at another level. Here level can be **physical, conceptual** or **external.**

- Data independence is one of the important characteristics of database management system.
- By this property, the structure of the database or the values stored in the database can be easily modified by without changing the application programs.
- There are two types of data independence (Refer Fig. 5.2.7)

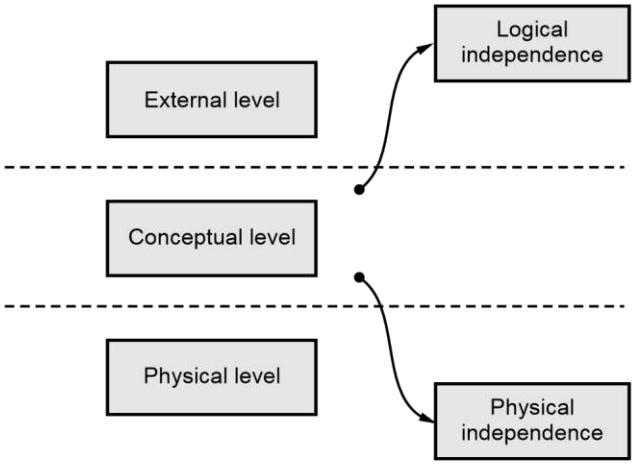


Fig. 5.2.7 Types of data independence

1. Physical Independence :

- This is a kind of data independence which allows the modification of physical schema without requiring any change to the conceptual schema.
- For example - if there is any change in memory size of database server then it will not affect the logical structure of any data object.

2. Logical Independence :

- This is a kind of data independence which allows the modification of conceptual schema without requiring any change to the external schema.
- For example - Any change in the table structure such as addition or deletion of some column does not affect user views.

By these data independence the time and cost acquired by changes in any one level can be reduced and abstract view of data can be provided to the user.

5.2.4 Data Dictionary

Concept: Data dictionary contains information about database itself. The data dictionary thus contains the **metadata** i.e. data about data. Following type of information is stored in data dictionary.

- Definition of database objects such as tables, views, constraints, clusters, procedures, functions, triggers
- Column name
- Data type information
- Amount of space required to store the data object
- Default field values
- Access rights
- Database user names - Schema information
- Last updated or accessed information
- Any other required information.

All this information can be stored in tables, XML files or in spreadsheet.

The data dictionary is updated automatically by the database systems when user issues the corresponding queries.

1

5.2.5 Keys used in Database Applications

Keys are important in relational database in order to establish the relationship between the tables. The keys are also used to access the records in the database table. Following are various types of keys that are used in database systems

1. Primary Key : This is the most important key in database which uniquely identifies the record. It can be a single attribute or combination of attributes. The database designer has to specially assign one of the candidate keys as primary key so that the record can be uniquely identified with the help of it. For example - **Stud_RollNo** is a primary key from student database.

Stud_Roll No.	FirstName	LastName	Address	CourseID
1001	AAA	BBB	Mumbai	C101
1002	XXX	YYY	Pune	M201
1003	PPP	QQQ	Bhopal	E303

2. Candidate Key : A candidate key is a single field or the least combination of fields that uniquely identifies each record in the table. But individual attribute can not identify the record uniquely. For example - **FirstName, LastName, Address** in combination can uniquely identifies the record, but individually FirstName, LastName or Address can not uniquely identifies the record.

Candidate keys

Stud_Roll No.	FirstName	LastName	Address	CourseID
1001	AAA	BBB	Mumbai	C101
1002	XXX	YYY	Pune	M201
1003	PPP	QQQ	Bhopal	E303

3. Foreign Key : A foreign key is generally a primary key from one table that appears as a field in another where the first table has a relationship to the second. For example - consider the following student table in which courseID is a foreign key.

Stud_Roll No	First Name	Last Name	Address	CourseID
1001	AAA	BBB	Mumbai	C101
1002	XXX	YYY	Pune	M201
1003	PPP	QQQ	Bhopal	E303

The **CourseID** from **Course table** will be the primary key. The **Course** table will be -

CourseID	Name
C101	Computer Engineering
M201	Mechanical Engineering
E303	E&TC Engineering
V505	Civil Engineering

4. Composite Key : The Key that consist of two or more attributes that uniquely identify an entity occurrence is called Composite key. For example to identify the Student taking particular course we can uniquely identify such type of record by combining two or more columns from the same table. Hence **Stud_RollNo** and **CourseID** together form a composite key.

Stud_RollNo	CourseID
1001	C101
1002	M201
1003	E303

5.3 Multimedia Applications

The word multi means more than one and media means for conveying information. The multimedia can be defined as -

Definition : Computer-based techniques of text, images, audio, video, graphics, animation, and any other medium where every type of information can be represented, processed, stored, transmitted, produced and presented digitally.

2

Examples

Some of the important programs are listed below in some important categories. So various programs are,

- Maya, Flash, Blender, comes mainly under Graphics Category.
- Interactivity category basically includes, MySQL, AJAX, Flash and Flex and PHP.
- Audio category is of Sound slides, Pro-tools, Adobe Auditions and more.
- Similarly programs in Video Category are, Canopus Edius, i Movie, Flash Video Encoder, Final Cut Pro.
- Text programs are like Word press, InDesign, and Dream weaver.

Components of Multimedia

Components of multimedia are :

1. **Text** can be added for giving emphasis.
2. **Graphics** are added for visual impact. A picture is worth a thousand words. Graphics enhance the presentation.
3. **Voice or audio** enhance presentation by adding persuasion.
4. The **animation** is for attracting attention. A chart can be focused more quickly by adding animation to it.
5. **Video** in multimedia can be used for providing clear cut instructions.

Uses of Multimedia

1. Education

Multimedia is extensively used in the field of education and training. Even in conventional method we use audio visual for imparting education, where charts, models etc. were used. Now a days the classroom need is not limited to that conventional method rather it needs audio and visual media. The multimedia integrates all of them in one system. For the use of multimedia as an education aid the PC contains a high quality display. This all has promoted the development of a wide range of computer based training . The software package named computer aided instruction is available that provides a friendly interactive method of learning.

2. Training

There various systems and intelligent tutoring systems available to train the students in many areas starting from the mathematics of a primary sudden to a difficult surgical process for a medical student.

3. Business

The business application of multimedia includes, product demos, instant messaging. One the excellent applications is voice and live conferencing. A multimedia can make a audience come live. It is widely used in programs. Such a program can be used by a mechanic and people.

The quality of business communication can be enhanced by multimedia. Product promotion, customer information, communication to employee can be done by multimedia.

4. Games and Entertainment

Real life like games can be created by multimedia. Developers use sound,animation,graphics of multimedia to create games.Flight simulator creates real life imaging.

Children can drive cars,play musical instrument, fly aircraft, play golf by multimedia.

5.4 Information System

Definition : An Information System (IS) is a set of interrelated components that collect, manipulate, store, and disseminate data and information and provide a feedback mechanism to meet an objective.

Examples

Some examples of information system are as follows -

1. **Supply Chain Management -** This system is for managing the flow of goods and services that involves the movement and storage of raw materials, work-in-progress inventory and finished goods.
2. **Customer Relationship Management -** Manages communications and marketing initiatives directed at customers.

- t t

3

3. **Geographic Positioning System(GPS) -** It provides driving directions and desired locations.
4. **Enterprise Resource Planning (ERP) -** It is an information system used to integrate the management of all internal and external information across an entire organization.

The parts of information system are -

1. Input 2. Processing 3. Output 4. Feedback

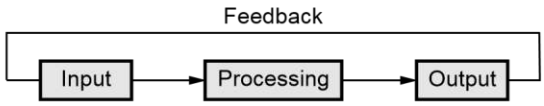


Fig. 5.4.1 Information system

1. Input :

Input is an activity of gathering and capturing raw data. For example - in online student information system, instructor has to submit the detailed information students before the summary of student information can be compiled.

2. Processing :

- Processing means converting or transforming data into useful output.
- Processing can involve making calculations, comparing data and taking alternative actions, and storing data for future use. For example - in tax management system - the tax needs to be calculated from the using the data such as gross salary, insurances, other deductions and so on.
- Processing data into useful information is critical in business settings.
- Processing can be done manually or with computer assistance.
- After processing the data results are typically stored in storage.

3. Output

- Output involves producing of useful information in the form of **documents or reports.**
- For example in-online banking system, the report is getting generated on the current transaction and can be presented to the customer in the form of bank statemen . This statement shows the details of all the transactions and the current available balance in the account.

4. Feedback

- In information system, feedback is information from the system which is used to make changes to input or processing activities.
- For example, errors or problems might make it necessary to correct input data or change a process. Consider a payroll example. Perhaps the number of hours an employee worked was entered as 400 instead of 40.
- Feedback is very important to managers or decision makers in order to modify the system.

5.4.1 Characteristics of Information System

1. **Accessibility :** Information present in the information should be easily accessible by authorized users.
2. **Accurate :** The information must be accurate and error free.
3. **Complete :** Information present in the information system must be complete so that it will satisfy all the queries of its users.
4. **Relevant :** The relevant information is important for the decision maker.
5. **Reliable :** This is very important characteristic of the information system.The reliable information is trusted by its users. The reliability of information depends upon the sources of data collection for the information system.
6. **Secure :** The information system must be secure and prevent any unauthorized access to it.
7. **Simple :** The information system must be very simple to handle and not very complex. It should not present the information system with too much details whereby the decision maker is unable to determine what is really important.
8. **Timely :** The information in the information system must be delivered in timely manner whenever is required.
9. **Economical :** Information must be economical to produce.

t

4

10. **Verifiable :** The information must be verifiable. That means one can check it to make it sure that information available in the information system is correct.

5.4.2 Components of Information System

Various components of information system are -

1. **Hardware :** Computer-based information systems use computer hardware, such as processors, monitors, keyboard and printers.
2. **Software :** These are the programs used to organize, process and analyze data.
3. **Databases :** Information systems work with data, organized into tables and files.
4. **Network :** Different elements need to be connected to each other, especially if many different people in an organization use the same information system.
5. **Procedures :** These describe how specific data are processed and analyzed in order to get the answers for which the information system is designed.

5.4.3 Types

Various types of information system are -

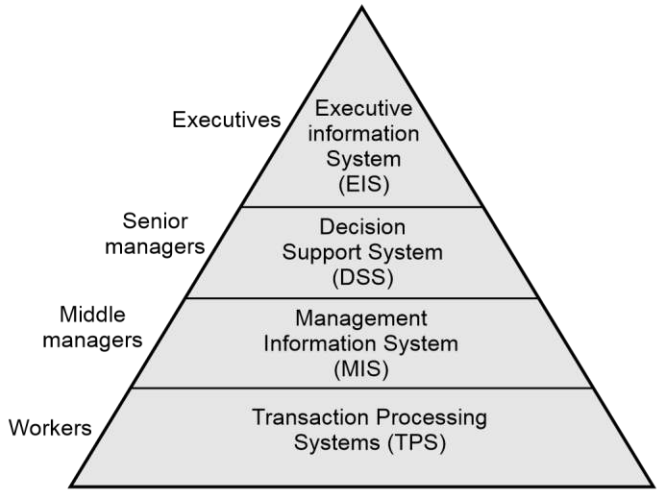


Fig. 5.4.2 Types of information system

1. Transaction Processing System(TPS) :

- The transaction processing system provides way to collect , process, store, display, modify or cancel the transactions.

This sys em allows multiple transactions to ake place simultaneously.

- The data collected by this system is typically stored in databases which can be used to produce reports such as billing, inventory summaries, and manufacturing schedules.
- Examples - Payroll system, order processing systems, Stock control systems.

Properties of Transaction Data

The transaction data must possess ACID properties.

1) Automicity :

The transaction must occure completely or not executed at all.

For example : Withdrawl from one account and deposition in other account completes one complete transaction.

2) Consistency :

The data written in the database must be valid data. This validity must be based on the constraints, triggers and other combinations.

3) Isolation :

If transaction is executed concurrently then it may come to some state. If transaction is executed sequentially then it may come to some another state. The isolation property ensures that these two states are equal.

4) Durability :

A transaction is said to be durable if it is committed and remains in the state even in case of power failure or system crash. To satisfy this property the transactions must be recorded in non - volatile memory.

2. Management Information System (MIS)

- A management information system is information system that uses data collected by Transaction Processing System(TPS).

• t

5

- This data is used for creating reports in such a way that managers can make use of it to make important business decisions.
- Some reports are created to present the summary. These activities are performed to increase the efficiency of managerial activity.
- Some examples of MIS -
  - Sales management systems
  - Inventory control systems
  - Budgeting systems
  - Management Reporting Systems (MRS)
  - Personnel (HRM) systems

3. Decision Support System (DSS)

- The decision support system helps make decisions by working and analyzing data that can generate statistical projections and data models.
- This system gives support rather than replacing a managers judgement while improving the quality of a manager decision.
- The DSS helps solve problems while using external data.
- Some examples of DSS
  - Logistics systems
  - Financial Planning systems
  - Spreadsheet Models

4. Executive Information System(EIS)

- It is a strategic level information system that is present at the top level of the pyramid.
- It helps executives and senior managers analyze the environment in which the organization operates, to identify long-term trends, and to plan appropriate courses of action.
- EIS organizes and presents data and information from both external data sources and internal MIS or TPS in order to support and extend the inherent capabilities of senior executives.
- EIS emphasizes graphical displays and easy-to-use user interfaces.

In recen years, the term EIS has lost populari y in favor of business intelligence.

- Examples of EIS are - Financial Analysis systems

5.5 Design and Development of Information System

The following are **steps** in the information systems development :

1. Feasibility Study

- The aim of a feasibility study is to see whether it is possible to develop a system at a reasonable cost. At the end of the feasibility study a decision is taken whether to proceed or not.
- A feasibility study contains the general requirements of the proposed system.
- It may be that development of a new system is not needed instead an update of the existing is enough.

2. Requirement Analysis

- This is a very important part in the development of an Information System and involves looking at an organization or system and finding out how information is being handled at the moment.
- The stage where users and IT specialists work together to collect and comprehend the business requirements. Based on requirements, both will work on the design and discuss the tasks to be done.
- The requirement analysis document is prepared at the end of this stage.

3. Design

- At this stage the system's blueprint is created.
- The technical architecture is designed which includes telecommunications, hardware and software suited for the system.
- The areas that need to be considered in the design process are listed below :

  1. Outputs 2. Inputs
  3. File design 4. Hardware
  5. Software

• t t

6

• The System design should be done for 1. User interface 2. Data design, and 3. Process design

4. Development and Testing

• Any new system needs to be thoroughly tested before being introduced.

• During this stage the building of the technical architecture, database and programs are executed.

• It is also the stage where the system is tested using the established test scripts and compare the expected outcomes to actual outcomes.

5. Implementation

• The stage where system is in place and is used by the actual workforce.

• User guide manual and training are provided to users.

6. Evaluation

• During this stage system need to be evaluated for any bug from time to time.

7. Maintenance

• This is the stage where system needs to be enhanced or strengthened in order to meet the goals of the organization.

Review Question

1. *What is information system ? What are the steps involved in design and development of an information system ?*

5.6 Personal Information System

• Personal Information management is set of activities in which people perform in order to acquire, organize, maintain, retrieve and use personal information such as documents, web pages, email messages every day to accomplish the assigned task.

• Personnel Information System (PIS) maintains the information about employees in, departments like personal, promotional, postings, qualification, awards, incentives, leave etc. that assists an organization in many ways.

There are various roles in the personal informa ion system such as - employee, manager, customer, student and so on.

• Conceptually PIS is a collection information and methods that help the people to maintain the information of persons.

• This information system can be maintained offline. One can carry this information system in pen drive.

• Examples of personal information system are -
  - Address book system,
  - Personal Notes
  - E-mail Notifications
  - Reminders and Alert Systems
  - Lists
  - Personal File collection system(document, music, photos)
  - Instant messaging systems.

5.6.1 Need for Personal information System

1. This system saves time and efforts in locating the information.
2. Information system is used for easy retrieval of information.
3. The information system organizes the entire information systematically.
4. Using personal information system within an organization means better employee productivity and better team work in the near term.

5.6.2 Functionality of Personal Information System

There are two modes of functionality of personal information system

1. **User Panel :** The user panel is for entering the personal information such as
   i) Profile Details
   ii) Qualification Details
   iii) Employment Details
2. **Administrator Panel :** The administrator panel maintains following activities.
   i) User settings

ii) Profile Master

iii) Qualification Master

iv) Document Upload Master

v) Email Settings

vi) Printer Settings

5.6.3 Benefits of Personal Information System

Following are benefits of personal information system -

1. Personal information system contain the data of all its users.
2. Users can easily search and locate data with personal information management system.
3. Information stored in personal information system is transferrable to other locations and software programs.

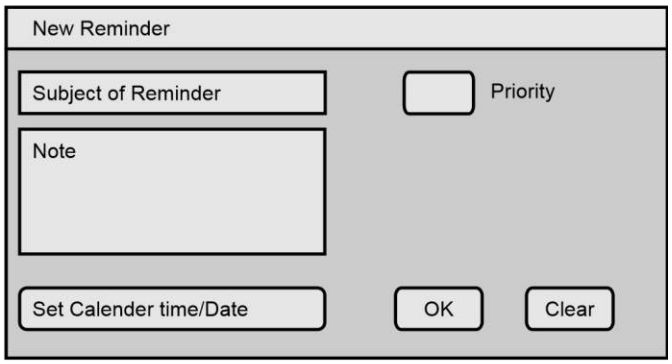
Ex. 5.6.1 *Design simple personal application that gives you reminders for each day. Identify the inputs to be taken, processing to be done, and the output to be produced. What multimedia components can be added to this application ?*

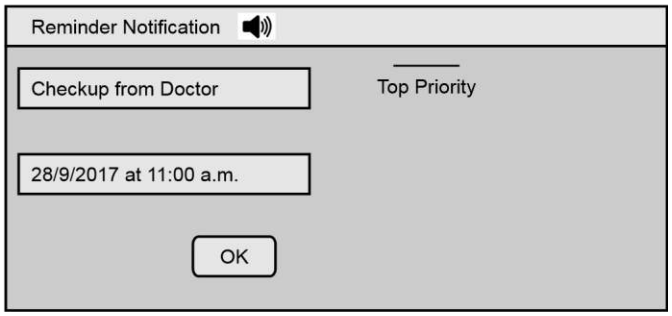
Sol. : The personal application for reminder is a simple and effective application that can be used in busy schedule for reminding the day to day activities.

Features of this application

1. User can set/update date/time of particular event.
2. The history data can be cleared.
3. Priority of task can be set or changed.
4. One can feed to-do list to the application.
5. The meeting schedule can be input to the application. The reminding application will display the schedule one hour prior to actual schedule.
6. The birthdays, anniversaries or important dates can be reminded on particular day-date by flashing images, messages and ringing alarm. User can stop the alarm or press 'remind me after sometime button'.
7. Email data viz, name of person, email address, phone number,and so on can be used by the application to as input. This feature can be se if user permits to do so.
8. The day/date/time can be set according to appropriate time zone of the country.

The sample GUI for per simple personal application that gives you reminders is as follows -





Input : Name of the person, birthdate, anniversary date, meeting timing, purpose of meeting, allotted timings for meeting, to-do list

Processing : Processing involves making calculations, matching the data against system date, matching person name, storing data for future use.

Output : Displaying reminding information on the device, displaying date,ringing alarm, flashing light.

t

8

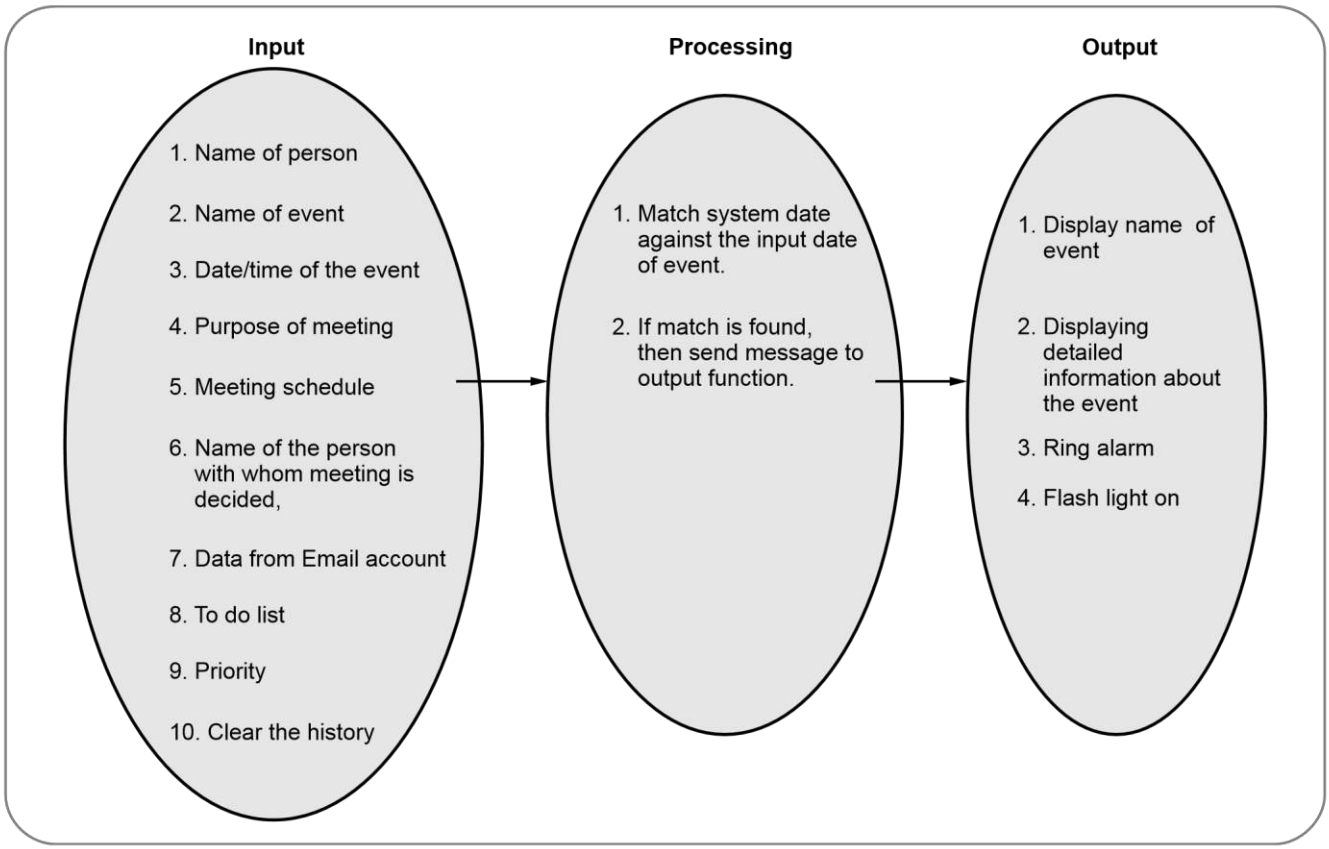


Fig. 5.6.1

Multimedia Components - Various multimedia components that can be added to this application are

1. **Text :** The text is used for typing the input to the system as well as for displaying the name of the event, detailed information about some schedule, to-do list, name of the person and so on.
2. **Graphics :** The attractive graphics flashing as output on matching with date or time of particular event.
3. **Image :** The image/photo of the person(s) can be displayed on the app while reminding the birthdays or anniversaries.
4. **Audio :** Melodious song or ring tone will be ringing for the reminding alarm.
5. **Animation :** Animated image or text can be displayed on the device for reminding app on particular event.

Ex. 5.6.2 : *Design an application for a blood bank which accept blood request and based on the request(blood group) it identify the people from the database and send them message. What are the important features needed for this*

Sol. : Features of Application System

(1) User/donor can see the availability of particular blood group.

(2) User can enter the blood group for making request for blood group.

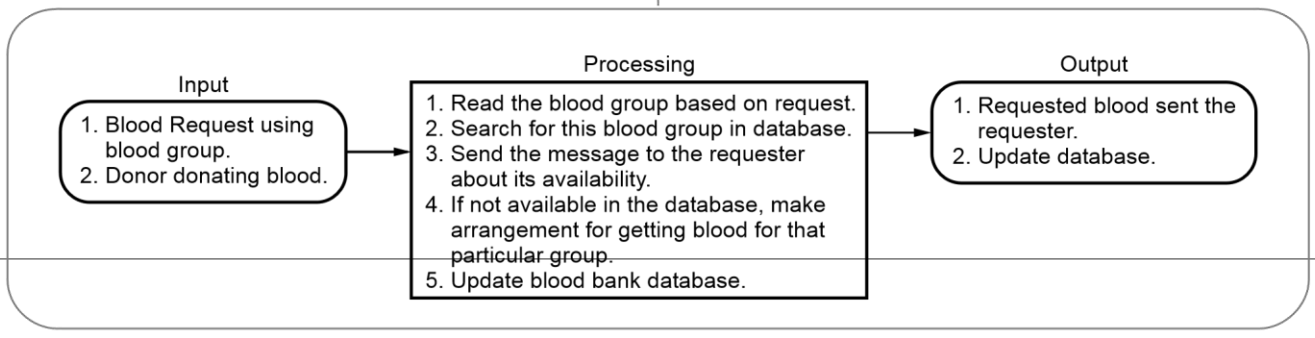


Fig. 5.6.2

(3) User can enter his personal information, such as name, email_address, phone, address to the system.
(4) User can cancel the request within Two Hours from request made.
(5) User can convey its urgency for the need for the blood.
(6) System can send SMS for availability of blood to the requester.
(7) Blood bank can also make a request to donor for supply of blood in case of shortage.
(8) The list of donor is updated after every week.
(9) The list of available blood donor can be sahred with doctors/ hospitals.

5.7 Information Retrieval System (IRS)

Definition of Information Retrieval (IR) : Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources.

- The information retrieval process begins when user submits the query to the system.
- Queries are formal statements of information need. The user queries are matched against the database information.
- Most of the information retrieval systems compute the numeric score on how well each object in the database matches the query and rank the objects according to its value.
- The top ranking objects are presented to the user.
- This process can be iterated if the user wishes to refine the query.

5.7.1 Various Tasks of Information Retrieval

1. Ad-hoc Search

It allows to find relevant documents for an arbitrary text query.

2. Filtering

By this task it is possible to identify the relevant user profile for new document.

3. Classification

The classification permits to identify the relevant lab ls for documents.

4. Question Answering

User can get the appropriate answer to the submitted question by using the information retrieval process.

5.7.2 Important Issues in Information Retrieval Process

1. **Relevance :** The most important responsibility of IR system is to present relevant information to its user.
2. **Evaluation :** Experimental procedures and measures for comparing system output with user expectations during retrieval of information.
3. **User and information need :** Search evaluation is user centered. Keyword queries are often poor descriptions of actual information needs. Interaction and context are important for understanding user intent. Query refinement techniques such as query expansion , query suggestion , relevance feedback improve ranking.

5.7.3 Information Retrieval and Search Engines

A search engine is the practical application of information retrieval techniques to large scale text collections.

Web search engines are best known examples.

For example - Google is a popular search engine.

Search Engine Issues

1. **Performance :** The response time for the search engine must be reduced. Hence in search engines indexes are data structures designed to improve search efficiency
2. **Dynamic Data :** The "collection" for most real applications is constantly changing in terms of updates, additions, deletions. Acquiring or "crawling" the documents is a major task in case of search engine. Updating the indexes while processing queries is also a design issue.
3. **Scalability :** This is the issue that deals with working with millions of users every day,and with many terabytes of documents.
4. **Adaptability :** This issue is related to changing and tuning search engine components such as ranking algorithm, indexing strategy, and interface for different applications.
5. **Spam :** Spam is irrelevant or unsolicited messages sent over the Internet, typically to a large number of users, for the purposes of

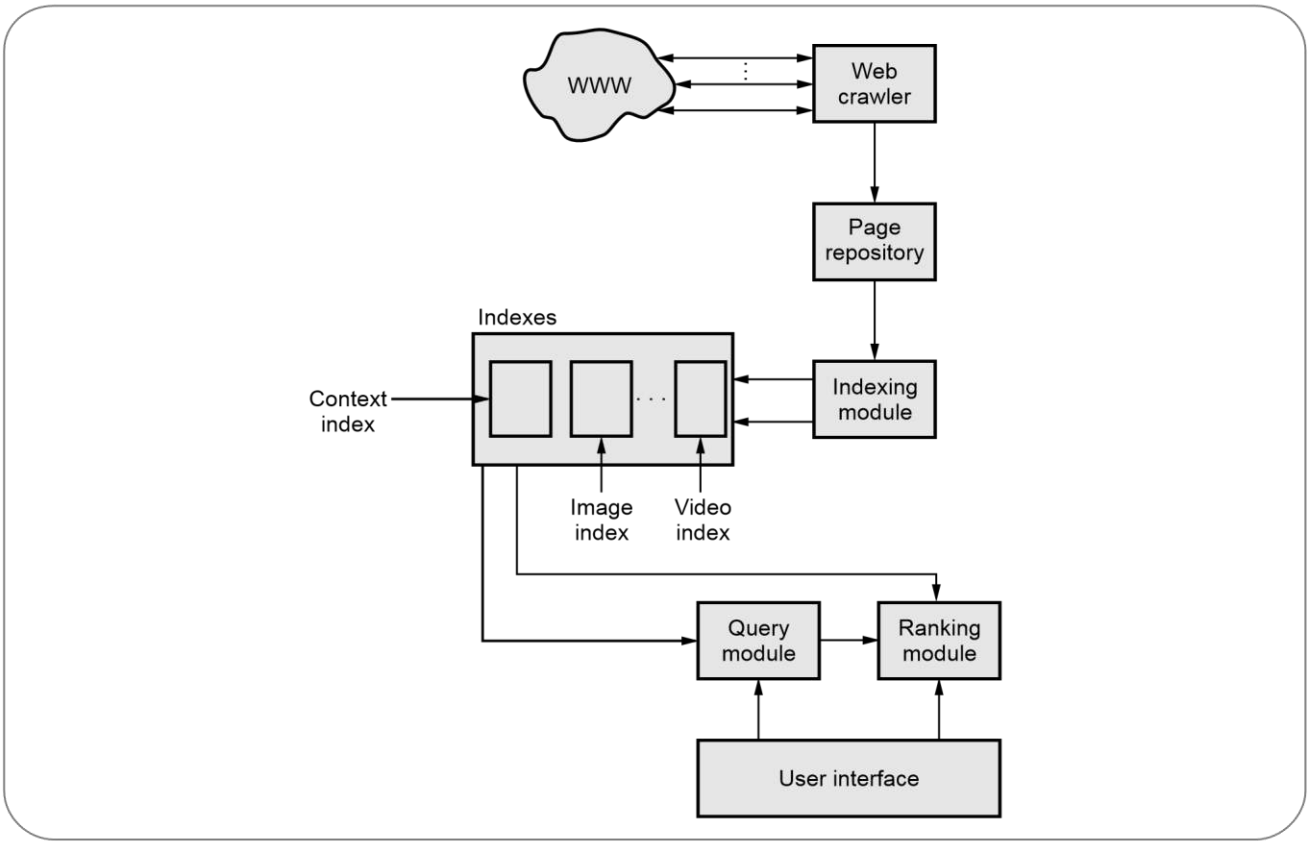


Fig. 5.7.1 General architecture of search engine

advertising, phishing, and spreading malware. Spam is a major issue in web search. It basically affects the efficiency of search engines and effectiveness of the results.

5.7.4 General Architecture of Search Engine

Step 1 : The important module is **Web crawler.** It constantly sends the crawlers to the web for crawling different web sites. It then extracts web pages from **WWW(world Wide Web)** constantly.

Step 2 : The pages brought by web crawler are then stored in **page repository.** The page repository is nothing but a temporary store for these web pages.

Step 3 : The web pages are then sent to **Indexing module.** The indexing module extracts the contents from these web pages. This module extracts the key element contents such as title tag, image tag, description tag, internal links and so on. The set of indices are prepared from the extracted contents present on the web page.

Indexing Process is performed in three steps -

1. **Text acquisition -** Identifies and stores documents for indexing
2. **Text transformation -** Transforms documents into index terms or features.
3. **Index creation -** Takes index terms and creates data structures (indexes) to support fast searching

These indexes can be **content index, image index, video index** and so on. Thus the contents are systematically arranged and ready to submit the search engine when the query is fired. The step 1 to 3 are constantly working whether or not the search engine is in execution. These modules are **query independent modules.**

Step 4 : User Interface module displays the search engine web address(Say Google) using which the user can type the query.

Step 5 : This query is then submitted to **Query module,** which breaks down the query and trnasform

it in a language which the search engine can understand. This query module then collects the results from indexes and pass these results to Ranking module.

Step 6 : The task of **ranking module** is to filter the contents and put them back on the user interface according to their ranking. Based on the popularity of the modules the ranks of the web pages are decided.

Step 7 : Thus the user gets the **result displayed** on the user interface.

If user want to get the result of particular type of information, then the query module need to be changed. And then the ranking module need to be changed so that it can display only the images and not the text contents.

Examples of Popular Search Engines

Search Engine	Description
Google	It is the most popular search engine globally.
Bing	It was launched in 2009 by Microsoft. It is the latest web-based search engine that also delivers Yahoo's results.
Ask	It was launched in 1996 and was originally known as Ask Jeeves. It includes support for match, dictionary, and conversation question.
AltaVista	It is powered by Yahoo technology.
AOL.Search	It is powered by Google.

Review Question

1. *Explain the architecture of generic search system ? If this system is to be used for searching only images, which components would need to be changed and how ?*

5.8 Social Networking Applications

- Social networking applications are **online technologies** that allow users to **communicate** with each other.
- Most popular social networking applications are d veloped with a purpose of **looking for people with common interests** to give them an opportun ty to discuss various topics, videos, and photos, add each other to the Friends category, upload music etc.
- One of the **largest advantages** of social networking application is the opportunity to find your old friends and relatives. Apart from that, social media has a great potential to create virtual jobs.

5.8.1 Types of Social Networking Applications

Following are various types of social networking applications that are most commonly used -

1. Messengers Applications

For the chatting purpose the messenger applications are created. Messenger is a mobile app or web service for instant messaging. In addition to common talks with friends, messengers are actively used in business. They make it possible to provide the client with timely service. The most popular messengers today are undoubtedly WhatsApp, Facebook Messenger,and so on.

2. Live Streaming Applications

It's never been as easy to launch live streaming video from any place as it is today. Now any smartphone owner can do it. Live streaming apps are gaining wide popularity day by day.

Live stream video is interesting for consumers because it's a more of an authentic advertising type, rather than an already edited video clip. For example, the company can show how they manufacture their goods, hold various contests, raffle some gifts and so on.

Different social media apps like Periscope, Meerkat, dominate their social media apps sector.

3. Applications for Inspiration

The Do it yourself(DIY) is a buzzword for the creative users of social networking. For example - Pinterest and Wishbone apps allow grouping images by topics or posting them on a general board.

. i

n

2

4 Lifestyle Applications

Fashion, movies, hobbies, sports, leisure, travel, daily routine - all these activities make life versatile and interesting.

Lifestyle apps conquer the mobile apps market since they simplify some of our daily routines.

This type of application is also widespread. Operator, Foursquare Swarm, Nextdoor - they are the few of the most popular applications.

5. Social Blogging Applications

For those who lover writing and those who have great writing skills make blogs to publish their thoughts on social media publicly or within some group. Boggler, Tumblr are some services that offer the blogging facility.The service allows users to post multimedia and other content to a short-form blog.

6. Business Social Applications

People who are looking for a job or who wants to develop useful contacts will surely find business social media applications more than helpful.

LinkedIn career app is familiar to many people. Today it is literally the largest business network in the Internet. It is oriented directly for business. You can build your professional brand there, look for employees, and read professional news.

7. Anonymous Social Applications

If you don't want to be recognized, but you want to participate in any discussions, then such applications are built for you. For example - Ask.fm app is very popular among teenagers where they can ask each other any questions. All questions are posted in an anonymous way, but if you want, you can show your name.

5.8.2 Features of Social Networking Applications

The desired features of social networking Applications are -

1. Simple User Interface :

A clear, clean and clutter free interface is a must-have - not only for a social networking site. All the popular social networking sites such as Facebook, Twitter or LinkedIn etc. maintain a very simple and minimalistic user interface. The choice of color for the site should be very sober - Preferably use wh te backgrounds and highlight the updates etc. in light color.

2. Prominent Search Facility

Because of the vast amount of information available on social networking sites, a good search functionality is must. The search functionality on the site however needs to go beyond content search and it needs to expand to search in social connections like groups, communities and so on.

Top right corner is the most recommended place for the search box.

3. Personalize user Profile and Experience

One of the key features which distinguishes a social networking site from a regular website is the personalized experience for the user. On a social networking site, users expect the content and flow of the website as per their likes and preferences.

All the features such as user profiles with activity feeds, member update notifications, emails, private messaging between members, ability to become friends with other members and follow members and so on enhance the personalized user experience.

4. Notifications and Real Time Updates

The regular update or real time notification enhances the user engagement in social networking applications.

5. Interactivity

The objective of social networking sites is to facilitate interactions between users by actively involving users. User can share photos, videos, locations, comments among the friends using the social networking applications.

5.8.3 Issues of Social Networking Applications

Various issues regarding social networking applications are -

i

3

1 Spamming

Spamming is when one perso or company sends an unwanted email to another person. Spamming is used for advertising purpose on the social networking applications.

2. Privacy

Privacy concern of the social networking application is raised because user tend to share lot of personal information on social networking sites. There are situations in which users may disclose personal information, sites may not take adequate steps to protect user privacy, and third parties frequently use information posted on social networks for a variety of purposes.

Users of these services need to be careful about data theft and virus attacks.

3. Notifications

Typically positive notifications are sent on social networking application. For example Mr. X and Mr. Y are now friends - such notification in common on Facebook but there is no notification that such and such person is removed from the friend-list. It enhances the positive atmosphere on social networking sites.

4. Unauthorized Access

There is a tendency to make unauthorized access to user's profile using social networking site.

5. Child Safety

The major issue of concern is teenagers and children make misuse of social networking services. For instance, there is a study which suggests that the children are not too far from inappropriate content on YouTube. Overuse of social networking sites also bring the depression and anxiety among the teenagers.

6. Trolling

Social networking sites such as Facebook are occasionally used to emotionally abuse, harass or bully individuals, either by posting defamatory statements or by forwarding private digital photos or videos that can have an adverse impact on the individuals depicted in the videos. Such actions are oft n referred to as "trolling". With a variety of celebrities joining social network ng sites, trolls tend to target abuse towards them.

7. Online Bullying

Online bullying is called cyberbullying is common occurrence due to social networking sites and result in emotional disturbances.

8. Data Mining

Through data mining companies are able to improve sales and profitability using the social networking applications.

5.8.4 Social Impact of Social Networking Applications

Following are key elements that demonstrate the social impact of social networking applications

1. Psychological Effects

The popularity of social networking sites is increasing day by day. People have been spending an excessive amount of time on the Internet in general and social networking sites in specific. This leads to addiction of these sites and cause clinical disorder. Researchers claim that due to social networking sites such as Facebook, person may start feeling lonely.

2. Interpersonal Communication

The purpose of social networking applications is to enhance the interpersonal communication. We can find out our old friends and can be in touch with them with the help of social networking sites. But the negative side is that - we then start believing in this type virtual communication and forget or give least importance to face to face communication or meeting the people.

3. Awareness of Rights

Workers or employees get aware of their rights due to use of social networking applications.

4. Social Anxiety

Due to increasing number of messages and use of social networking applications people get connected continuously. One can not get his/her own space. Sometimes it may create a negative impact on friends, and relatives and thereby the social anxiety gets increased.

n

4

5 Growth of Patents

There has been rapid growth i the number of patent applications that cover new technologies related to social networking. The number of published applications has been growing rapidly since 2003.

Ex. 5.8.1 *Design a simple social-media application to pass on messages to friends whenever two people come in contact with(within range of) one another. Assume that you are using devices with Blue-tooth support to transfer messages.*

Sol. :

1. Introduction

Bluetooth is a wireless technology standard for exchanging data over short distances. The Bluetooth based message transfer application can be used on PC or on Android devices.

The security aspect of this application is taken care by pairing the Bluetooth devices before starting the actual communication.

This is a kind of application in which two people can pass messages to each other when they come in contact(in fact within the range) of each other.

2. Working

Following are the steps that demonstrate the working of this messaging application -

1. The one communicating device checks whether the other device is on or off.
2. If the other device is in off mode then this device makes a request to switch on the device for communication purpose.
3. Perform the scanning of devices which are within the range.
4. Display the list of devices which are available for pairing.
5. Select the device to which you want to pass message.
6. By entering the Security Pin establish the connection.
7. If the device connects - then start sending and receiving messages through chat room.

This working can be demonstrated by following diagram.

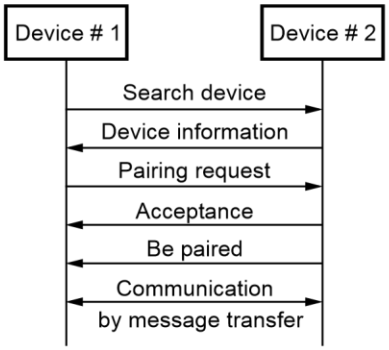


Fig. 5.8.1

3. Graphical User Interface

The sample graphical user interface of this application can be as shown by following sketches.

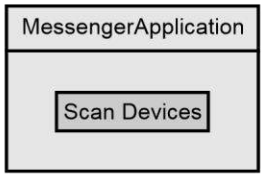


Fig. 5.8.2

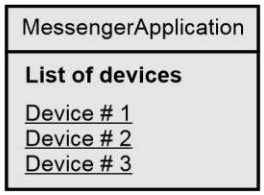


Fig. 5.8.3

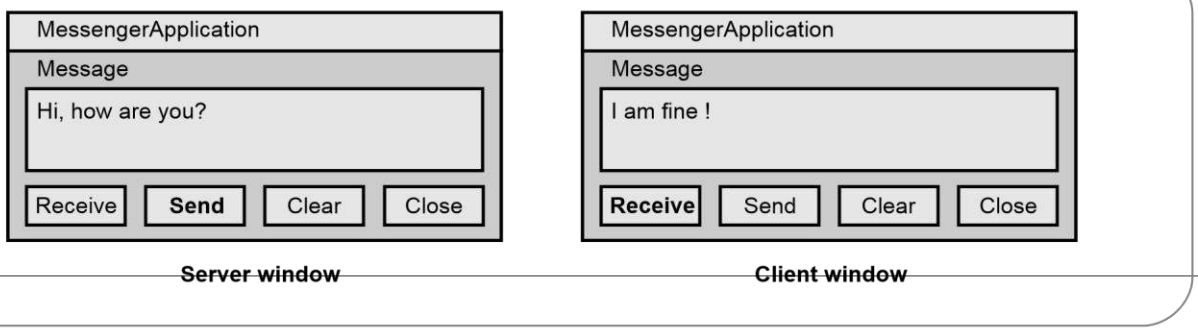


Fig. 5.8.4 Sample GUI

n

5

4 Uses

The messenger application usi g Bluetooth chatting is an innovative approach. This is basically a chat application which can take advantage of wireless technology for messenger application.

Two Marks Questions with Answers

Q.1 *What is interactive application ?*

Ans. :

- An interactive application is a collection of objects intended for performing certain task when user triggers the command.
- Typical examples of interactive web applications are - online course registration system, online shopping system and so on.

Q.2 *Enlist the advantages of interactive applications.*

Ans. : The advantages of interactive application are as follows -

1. Increases engagement of users.
2. Boosts communication by sharing emails, files, and printing files
3. Annotations of documents
4. Interactive applications can be ported on mobile devices to avail the interconnectivity at finger tip.

Q.3 *List the steps involved in a typical application development life cycle.*

Ans. : The steps are as follows -

1. Understand data items and the data dictionary.
2. Understand the table design.
3. Understand business view design.
4. Understand form design.
5. Understand report design.
6. Understand data structure design.
7. Understand event rules design.
8. Understand system function.

Q.4 *Define the terms - Database and DBMS.*

Ans. : Database is an organized collection of data. A Database Management System (DBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data.

Examples : Well known DBMS software are - MS, ACCESS,Oracle, MySQL, Microsoft SQL Server, Sybase and so on.

Q.5 *What are advantages of DBMS ?*

Ans. : The advantages of DBMS are -

1. **Reduced data redundancy :** The database approach removes the redundancy by integrating the files. But this approach can not completely eliminate redundancy completely but can control the data redundancy.
2. **Data consistency :** In this approach all copies of data are kept consistent.
3. **Sharing of data :** The centralized stored database can be accessed by multiple users or application programs.
4. **Data integrity :** The data integrity provides validity and consistency of stored data.
5. **Improved security :** The database approach prevents unauthorized access to data by means of user name, password, and access rights.

Q.6 *What is data model in database ?*

Ans. : The data model in Database Applications describe the logical structure of database, relationship between the database stored in database and various constraints on data.

Q.7 *What is the importance of data model ?*

Ans. :

1. End users have different view for data.
2. Data model organizes data for different users.

Q.8 *Enlist the data models used in database management system*

Ans. : Various types of data models are used in database management system -

1. Hierarchical data model
2. Network data model

3. Relational Model
4. E-R Model

Q.9 *Explain the concept of data independence in DBMS.*

Ans. :

- Data independence is an ability by which one can change the data at one level without affecting the data at another level. Here level can be physical, conceptual or external.
- Data independence is one of the important characteristics of database management system. By this property, the structure of the database or the values stored in the database can be easily modified by without changing the application programs.

Q.10 *What is Data Dictionary ?*

Ans. : Data dictionary contains information about database itself. The data dictionary thus contains the metadata i.e. data about data.

Following type of information is stored in data dictionary - Definition of database objects such as tables, views, constraints, clusters, procedures, functions, triggers,Column name, Data type information,Amount of space required to store the data object

Q.11 *What is primary Key ?*

Ans. : This is the most important key in database which uniquely identifies the record. It can be a single attribute or combination of attributes. The database designer has to specially assign one of the candidate keys as primary key so that the record can be uniquely identified with the help of it. For example - Stud_RollNo is a primary key from student database.

Q.12 *What is multimedia technology ?*

Ans. : Multimedia technology is Computer-based technique of text, images, audio, video, graphics, animation, and any other medium where every type of information can be represented, processed, stored, transmitted, produced and presented digitally.

Q.13 *Enlist two uses of multimedia technologies.*

Ans. :

1. Multimedia is extensively used in the field of education and training.
2. The business application of multimedia includes, product demos, instant messaging. One the excellent applications is voice and live conferencing.
3. Real life like games can be created by multimedia. Developers use sound,animation,graphics of multimedia to create games.
4. Multimedia is used in simulation and modeling. Flight simulator creates real life imaging.

Q.14 *What is information system ?*

Ans. : An Information System (IS) is a set of interrelated components that collect, manipulate, store, and disseminate data and information and provide a feedback mechanism to meet an objective.

Some examples of information system are as follows -

1. Supply Chain management
2. Customer Relationship management
3. Geographic Positioning System(GPS)
4. Enterprise Resource Planning (ERP)

Q.15 *What are parts of information System ?*

Ans. :

- The input, processing, output and feedback are the parts of information system.
- Input is an activity of gathering and capturing raw data. Processing means converting or transforming data into useful output. Output involves producing of useful information in the form of **documents or reports.** In information system, feedback is information from the system which is used to make changes to input or processing activities.

Q.16 *Specify any two desirable characteristics of information system.*

7

Ans. :

1. **Accurate :** The information must be accurate and error free.
2. **Complete :** Information present in the information system must be complete so that it will satisfy all the queries of its users.
3. **Relevant :** The relevant information is important for the decision maker.
4. **Reliable :** This is very important characteristic of the information system. The reliable information is trusted by its users.

Q.17 *Enlist the different types of information system.*

Ans. :

There are four types of information systems -

1. Transaction Processing System (TPS)
2. Management Information System (MIS)
3. Decision support System (DSS)
4. Executive Information system (EIS)

Q.18 *What is transaction processing system(TPS) ?*

Ans. : The transaction processing system provides way to collect , process, store, display, modify or cancel the transactions.

Q.19 *What is MIS ?*

Ans. :

- A management information system is information system that uses data collected by Transaction Processing System(TPS).
- This data is used for creating reports in such a way that managers can make use of it to make important business decisions.

Q.20 *Give examples of Decision Support System(DSS).*

Ans. : Some examples of DSS

- Logistics systems
- Financial planning systems
- Spreadsheet models

Q.21 *Enlist the steps used in design and development of information system.*

Ans. : Following are the steps used in design and development of information system -

1. Feasibility study
2. Requirement analysis
3. Design
4. Development and Testing
5. Implementation
6. Evaluation
7. Maintenance

Q.22 *What is personal information system ?*

Ans. :

- Personal information management is set of activities in which people perform in order to acquire, organize, maintain, retrieve and use personal information such as documents, web pages, email messages every day to accomplish the assigned task.
- Examples of personal information system are -
  - Address book system
  - Personal Notes
  - E-mail Notifications
  - Reminders and Alert Systems
  - Instant messaging systems

Q.23 *Why personal information system is needed ?*

Ans. :

1. This system saves time and efforts in locating the information.
2. Information system is used for easy retrieval of information.
3. The information system organizes the entire information systematically.
4. Using personal information system within an organization means better employee productivity and better team work in the near term.

Q.24 *What information Retrieval System (IRS) ?*

Ans. : Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources.

8

Q.25 *What is search engine ?*

Ans. : A search engine is the practical application of information retrieval techniques to large scale text collections.

For example - Google is a popular search engine.

Q.26 *What is spam ?*

Ans. : Spam is irrelevant or unsolicited messages sent over the Internet, typically to a large number of users, for the purposes of advertising, phishing, and spreading malware. Spam is a major issue in web search. It basically affects the efficiency of search engines and effectiveness of the results.

Q.27 *What is web crawler ?*

Ans. : Web crawler is an important program in search engine which constantly crawls or examines the web for collecting information from web site.

Q.28 *Give examples of popular search engines*

Ans. :

Search Engine	Description
Google	It is the most popular search engine globally.
Bing	It was launched in 2009 by Microsoft. It is the latest web-based search engine that also delivers Yahoo's results.
Ask	It was launched in 1996 and was originally known as Ask Jeeves. It includes support for match, dictionary, and conversation question.
AltaVista	It is powered by Yahoo technology.

Q.29 *Why the social networking applications are created ?*

Ans. : • Social networking applications are online technologies that allow users to communicate with each other.

• Most popular social networking applications are developed with a purpose of looking for people with common interests to give them an opportunity to discuss various topics, videos, and photos, add each other to the Friends category, upload music etc.

Q.30 *List 5 important features of social networking app.*

Ans. : The desired features of social networking app are as follows -

1. Simple User Interface
2. Prominent Search Facility
3. Personalize user Profile and Experience
4. Notifications and Real Time Updates
5. Interactivity

Q.31 *Write any four multimedia applications used in educational domain.*

Ans. :

1) Drawing tools such as photoshop, paint.
2) Presentation tools such as PowerPoint.
3) Ebook Reader such as Kindle.
4) Animation tools such as Adobe Flash.
5) Tools for Simulation Modeling.

❑❑❑