NEAR EAST UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

INTERACTIVE WEB PAGE DESIGN FOR ONLINE EXAMINATION

Graduation Project

COM – 400

Student: ZAHOOR HUSSAIN

Student No: 20010998

Supervisor: Assist.Prof.Dr.Adil Amirjanov

Nicosia - 2005

ACKNOWLEDGMENT

All praise and glory to almighty ALLAH, the Lord of the universe, who is the entire source of all knowledge and wisdom endowed to mankind. All thanks are due to him who gave me ability and patience throughout my studies for completing this task.

I would like to acknowledge to my parents who has brought all of his efforts to support me, without knowing the return and who has patiently encouraged me to be the best everywhere.

I would also like to thank my project supervisor Prof. Dr. Adil Amirjanov for his intellectual support, encouragement and enthusiasm which made it possible to accomplish this project. I appreciate his most gracious encouragement and very valued constructive criticism throughout my education.

My special thanks goes to NEU educational staff especially to Computer engineering teaching staff for their generosity & special concern of me and all COM students.

ABSTRACT

The aim of this project is to explain Web Designing and making a site for online examination. The web designing has been explained from simple html tags to the design of database for web. The website for online examination includes many pages of html and asp. User can interact with the web site through asp pages i.e. user can register by giving his/her details and able to take exam and see his/her result. Simple JavaScript techniques are used for dynamic time clock and the fields which are compulsory, user can't leave empty those fields. Database is explained and designing simple Student database is discussed. The way user interact with the registered section is that all the question and answer is stored in the database so asp code is used to retrieve the data and put in the browser and some simple addition function are used to sum the correct mark answers

This is simple explanation of online examination system we can go further and further to make it more complex and reliable.

TABLE OF CONTENTS

INTRODUCTION

Internet is worldwide computer network connecting millions of computers. The beauty if html is that it made the internet transparent .Clients Software called Web Browsers communicated with web servers.

The Clients to which they are requesting are called servers. The Servers Contain many pages ad when the client request for the particular page then it is sent.

Explanation of html, JavaScript, ASP, Database and its connection that are used in interactive web design for online examination is done. The First Chapter is about the HTML i.e. Hyper Text Mark up Language which describes simple html elements and tags used to enclose the information in the type of web page that can be viewed by many users. Cascading Style sheet is explained that how to make cascading style sheet and then put into my web pages so that all pages look similar. The form method is explained that how to send information to the new users, well frames is also has an important factor that can divide the pages into many columns containing each column different page or information.

Second Chapter is about the JavaScript. JavaScript is a Scripting language with very Simple Syntax that is developed by Netscape and most popular language. There are many functions of JavaScript that can be used to make website more attractive and well organized. JavaScript can be put regardless of position restriction that is where we need either in body or head or in both. Similar to other programming languages we can declare the variables in JavaScript and by using built in operators we can perform many complex functions .We can also declare functions like other languages in JavaScript and can call at different points in the page whenever we need.

Third Chapter in my project is about the Active Server Page the most vital role in the Web Pages. ASP can contain same thing like html and script but here scripts run the server .The ASP has many functions i.e. it can dynamically edit ,change ,add some content to the web page and it can also retrieve the records contain in the database and put the result on the web page. For the Form type Request.querystring is used for the get method and request. Form is used for the Post method. In the ASP we can also declare

any variables like other programming languages. We can also include the outside asp files to any page to avoid rewriting the same thing like database connection. There are many ASP scripts and Methods that make our web pages dynamic and more attractive.

Fourth Chapter is about the Database Design. Database is most important thing while we talking about the memory. Relational Database is efficient approach to the complex database that we can make the database relations in order to have less memory and fast accessing. By Giving Simple Examples, database relations are explained.

Fifth Chapter is about the ADO. In this chapter explanation of how database connection is established and can be dealt with the database tables using WebPages. All content of ADO is well explained by examples. The final chapter is the explanation of

Step by step the makings of interactive web page design for online examination.

CHAPTER 1

BASICS OF HTML

HTML or Hypertext Markup Language is designed to specify the logical organization of a document, with important hypertext extensions. This choice was made because the same HTML document may be viewed by many different "browsers", of very different abilities. Thus, for example, HTML allows you to mark selections of text as titles or paragraphs, and then leaves the interpretation of these marked *elements* up to the browser.

HTML instructions divide the text of a document into blocks called *elements*. These can be divided into two broad categories -- those that define how the **BODY** of the document is to be displayed by the browser and those that define information `about' the document, such as the **title** or relationships to other documents. [1]

1.1 Elements in HTML Documents

The HTML instructions, along with the text to which the instructions apply, are called HTML ***elements.*** The HTML instructions are themselves called ***tags,*** and look like **<element_name>** that is, they are simply the element name surrounded by left and right angle brackets.

Most elements mark blocks of the document for particular purpose or formatting: the <element_name> tag marks the beginning of such as section. The end of this section is then marked by the *ending* tag </element_name> note the leading slash character "/" that appears in front of the element name in an end tag. End, or stop tags are always indicated by this leading slash character.

1.1.1 Empty Elements

Some elements are empty that is, they do not affect a block of the document in some way. These elements do not require an ending tag. An example is the <HR> element, which draws a horizontal line across the page. This element would simply be entered as <HR>.

1.1.2 Upper and Lower Case

Element names are case insensitive. Thus, the horizontal rule element can be written as any of <hr>, <Hr> or <HR>.

1.1.3 Elements with Attributes

Many elements can have arguments that pass parameters to the interpreter handling this element. These arguments are called attributes of the element. For example, consider the element A, which marks a region of text as the beginning (or end) of a hypertext link. This element can have several attributes. One of them, HREF, specifies the hypertext document to which the marked piece of text is linked. To specify this in the tag for A write:

<A HREF="http://www.zahoor.com/1.html"> Zahoor .

Where the attribute HREF is assigned the indicated value. Note that the A element is not empty, and that it is closed by the tag . Note also that end tags never take attributes the attributes to an element are always placed in the start tag.

1.2 HTML Document Structure

HTML documents are structured into two parts, the HEAD, and the BODY. Both of these are contained within the HTML element this element simply denotes this as an HTML document.

The head contains information about the document that is not generally displayed with the document, such as its TITLE. The BODY contains the body of the text, and is where the document material to be displayed. Elements allowed inside the HEAD, such as TITLE, are not allowed inside the BODY, and vice versa. [1]

1.2.1 Example of Document Structure

```
<HTML>
 <HEAD>
  <TITLE> Example Page </TITLE>
 </HEAD>
 <BODY>
  <h1>Example Project </h1>


   <p>any thing </p>
  <ul>
  <li><A HREF="www.links.com ">more links</A> Links
  <li>Destroy the
     <A HREF="http://searchengines.com">Search</A>Search Eng.    </ul>
```

</BODY>
</HTML>

1.3 CSS

HTML or XHTML only divide a document up into paragraphs, lists, and headings and so on, but does not really say how these things should look. Rather, a browser generally makes some assumptions about how things should look and we are then stuck with those choices.

This could be changed given a way of controlling how different markup elements (like headings, paragraphs, etc) look. This is the role of CSS. Cascading Style Sheets, or CSS, is a language, separate from HTML or XHTML, designed for specifying the layout or formatting properties of the various HTML elements in a document. For example, CSS statements like the following,

```
body {font-family:    Arial, helvetica, sans-serif;
    color:          black;
    background-color: white;
  }
```

Means that, inside the BODY of a document, the desired font is Arial, the desired text color is black, and the desired background color for the page is white. More complicated rules let us control underlining of links, the placement of background images, the widths of margins, the colors of borders around paragraphs or headings, etc.

The style sheet is included into this document using a special link element of the form:

<link rel="stylesheet" href="stylesheet.css" >

Where *stylesheet.css* contains the CSS document.

1.4 HTML Elements

1.4.1 Head

The HEAD contains general information, or *meta*-information, about the document. It is the first thing in any document, lying above the BODY and just after the <HTML> tag starting the document.

The contents of the HEAD are not displayed as part of the document text: the displayed material is found within the BODY. Consequently, only certain mark-up elements can be placed within the HEAD. These are:

BASE-- A record of the original URL of the document: this allows moving the document to a new directory (or even a new site) and having relative URLs access the appropriate place with respect to the original URL.

Usually placed in the HEAD by the server or a server script/program to indicate that a document is searchable.

LINK -- Defines the relationship(s) between this document and another or others. A document can have several LINK elements.

META -- A container for document *metainformation.*

TITLE -- The title of the document. This element is **mandatory** -- ***all*** documents must have a TITLE.

STYLE -- Stylesheet instructions, written in a stylesheet language. Stylesheet instructions specify how the document should be formatted for display. Very few browsers currently support stylesheets.

SCRIPT -- Script program code: for enclosing, within a document, scripting program code that should be run with and that can interact with the document. Example languages are JavaScript and VBScript.

1.5 Frames

Frames can display more than one HTML document in the same browser window. Each HTML document is called a frame, and each frame is independent of the others.

The disadvantages of using frames are that web developer must keep track of more HTML documents. It is difficult to print the entire page.

1.5.1 The Frameset Tag

The <frameset> tag defines how to divide the window into frames; each frameset defines a set of rows or columns.

The values of the rows/columns indicate the amount of screen area each row/column will occupy.

1.5.2 The Frame Tag

The <frame> tag defines what HTML document to put into each frame. The column can be set to 25% of the width of the browser window. The second column can be set to 75% of the width of the browser window. The HTML document "frame_a.htm" is put into the

first column, and the HTML document "frame_b.htm" is put into the second column. But it's better to put the frames pages asp extensions so that we can change easily. [2]

1.6 Tables

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each row is divided into data cells (with the <td> tag). The letters td stands for "table data," which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

1.7 Forms

A form is an area that can contain form elements. Form elements are elements that allow the user to enter information (like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.) in a form. A form is defined with the <form> tag. <Input>
The most used form tag is the <input> tag. The type of input is specified with the type attribute. The most commonly used input types are explained below.

Text Fields-Text fields are used when you want the user to type letters, numbers, etc. in a form. The Form's Action Attribute and the Submit Button When the user clicks on the "Submit" button, the content of the form is sent to another file. The form's action attribute defines the name of the file to send the content to. The file defined in the action attribute usually does something with the received input.

1.8 Simple Html Tags

Tag	Description
<b>	Defines bold text
 	Breaks the line
<tite>	Defines Title of the page
<i>	Defines italic text
<small>	Defines small text
	Defines strong text
<sub>	Defines subscripted text
<sup>	Defines superscripted text

<table>	Defines table
<frame>	Defines frame page
<pre>	Defines preformatted text

CHAPTER 2

Java Script Fundamentals in Web Design

Java Script is to improve the design, validate forms, and much more. JavaScript was developed by Netscape and is the most popular scripting language on the internet. JavaScript works in all major browsers that are version 3.0 or higher. It is designed to add interactivity to HTML pages. JavaScript is a scripting language - a scripting language is a lightweight programming language which contains lines of executable computer code. A JavaScript is usually embedded directly in HTML pages and it is an interpreted language (means that scripts execute without preliminary compilation). [2]

2.1 Java and JavaScript

Java and JavaScript are two completely different languages. Java (developed by Sun Microsystems) is a powerful and very complex programming language - in the same category as C and C++ while JavaScript was designed to add interactivity to HTML pages. JavaScript is a scripting language - a scripting language is a lightweight programming language which contains is lines of executable computer code. [2]

2.2 Functions of JavaScript

- **JavaScript gives HTML designers a programming tool** - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax. Almost anyone can put small "snippets" of code into their HTML pages.
- **JavaScript can put dynamic text into an HTML page** - A JavaScript statement like this: document. write ("<h1>" + name + "</h1>") can write a variable text into an HTML page.
- **JavaScript can react to events** - A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.
- **JavaScript can read and write HTML elements** - A JavaScript can read and change the content of an HTML element.

- **JavaScript can be used to validate data** - A JavaScript can be used to validate form data before it is submitted to a server, this will save the server from extra processing.

2.3 Syntax of JavaScript into an HTML Page

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

The code above will produce this output on an HTML page:

```
Hello World!
```

2.4 Scripts Position

Scripts in a page will be executed immediately while the page loads into the browser. This is not always what we want. Sometimes we want to execute a script when a page loads, other times when a user triggers an event.

2.4.1 Scripts in the head section: Scripts to be executed when they are called, or when an event is triggered, go in the head section. When you place a script in the head section, you will ensure that the script is loaded before anyone uses it.

2.4.2 Scripts in the body section: Scripts to be executed when the page loads go in the body section. When you place a script in the body section it generates the content of the page.

2.4.3 Scripts in both the body and the head section: You can place an unlimited number of scripts in your document, so you can have scripts in both the body and the head section.

2.4.4 External JavaScript: Sometimes you might want to run the same script on several pages, without writing the script on each and every page. To simplify this you can write a script in an external file, and save it with a .js file extension, you can call this

script, using the "src" attribute, from any of your pages: is: Save the external file as khan.js. [3]

2.5 Variables in JavaScript

Variable can be declared with the var statement:

var strname = *some value*

Variable can also be created without the var statement:

strname = *some value*

2.5.1 Lifetime of Variables

When you declare a variable within a function, the variable can only be accessed within that function. When you exit the function, the variable is destroyed. These variables are called local variables. You can have local variables with the same name in different functions, because each is recognized only by the function in which it is declared.

If variable is declared outside a function, all the functions on the page can access it. The lifetime of these variables starts when they are declared, and ends when the page is closed.

2.6 JavaScript Operators

2.6.1 Arithmetic Operators

Operator	Description	Example	Result
+	Addition	x=2,x+2	4
-	Subtraction	x=2,5-x	3
*	Multiplication	x=4,x*5	20
/	Division	15/5,5/2	3 2.5
++	Increment	x=5,x++	x=6
--	Decrement	x=5,x--	x=4

2.6.2 Assignment Operators

Operator	Example	Is The Same As
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

2.6.3 Comparison Operators

Operator	Description	Example
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

2.6.4 Logical Operators

Operator	Description	Example
&&	and	x=6,y=3 (x < 10 && y > 1) returns true
||	or	x=6,y=3 (x==5 || y==5) returns false
!	not	x=6,y=3!(x==y) returns true

2.6.5 String Operator

A string is most often text, for example "Hello World!" To stick two or more string variables together, use the + operator. [3]

2.7 Functions

A function contains some code that will be executed by an event or a call to that function. A function is a set of statements. Functions can be used within the same script, or in other documents. Functions are defined at the beginning of a file (in the head section), and call them later in the document.

This is JavaScript's method to alert the user.

```
alert("This is a message")
```

In java scripts functions can be created and called with in page.

2.7.1 Function Declaration

To create a function name of the function is defined, any values ("arguments"), and some statements:

```
function myfunction(argument1,argument2,etc)
{
some statements
}
```

A function with no arguments must include the parentheses:

```
function myfunction()
{
some statements
}
```

Arguments are variables used in the function. The variable values are values passed on by the function call.

By placing functions in the head section of the document, make sure that all the code in the function has been loaded before the function is called.

Some functions return a value to the calling expression.

```
function result(a,b)
{
c=a+b
return c
}
```

2.7.2 Calling a Function

A function is not executed before it is called. Function can be called containing arguments as follows,

myfunction(*argument1,argument2,etc*)

or without arguments:

```
myfunction()
```

The return Statement-Functions that will return a result must use the "return" statement. This statement specifies the value which will be returned to where the function was called from. For example a function that returns the sum of two numbers:

```
function total(a,b)
{
result=a+b
return result
}
```

When function is called two arguments must be sent with it:

```
sum=total(2,3)
```

The returned value from the function (5) will be stored in the variable called sum.

JavaScript is Case Sensitive-A function named "myfunction" is not the same as "myFunction". Therefore watch your capitalization when you create or call variables, objects and functions.

2.8 Symbols

Open symbols, like ({ [" ', must have a matching closing symbol, like ' "] }).

2.8.1 White Space

JavaScript ignores extra spaces. white space can be added to the script to make it more readable. These lines are equivalent:

```
name="Hege"
name = "Hege"
```

2.8.2 Break up a Code Line

A code line within a text string can be break up with a backslash. The example below will be displayed properly:

```
document.write("Hello \
World!")
```

A code line can not break up like this:

```
document.write \
("Hello World!")
```

The example above will cause an error.

2.8.3 Insert Special Characters

Special characters (like " ' ; &) can be inserted with the backslash:

```
document.write ("You \& I study \"Happy Birthday\".")
```

The example above will produce this output: You & I study "Happy Birthday".

CHAPTER 3

ACTIVE SERVER PAGERS (ASP)

An ASP file is just the same as an HTML file. An ASP file can contain text, HTML, XML, and scripts. Scripts in an ASP file are executed on the server and this file has the file extension ".asp".

3.1 ASP and HTML

When a browser requests an HTML file, the server returns the file. When a browser requests an ASP file, IIS passes the request to the ASP engine. The ASP engine reads the ASP file, line by line, and executes the scripts in the file. Finally, the ASP file is returned to the browser as plain HTML.

3.2 Functions of ASP

ASP dynamically edit, change or add any content of a Web page Respond to user queries or data submitted from HTML forms Access any data or databases and return the results to a browser customize a web page to make it more useful for individual users.

3.2.1 The advantages of using ASP

Instead of CGI and Perl, those of simplicity and speed Provides security since your ASP code can not be viewed from the browser

Since ASP files are returned as plain HTML, they can be viewed in any browser

Clever ASP programming can minimize the network traffic. [6]

3.3 The Basic Syntax Rule

An ASP file normally contains HTML tags, just like an HTML file. However, an ASP file can also contain server scripts, surrounded by the delimiters <% and %>. Server scripts are executed on the server, and can contain any expressions, statements, procedures, or operators valid for the scripting language you prefer to use.

ASP scripts are surrounded by <% and %>. To write some output to a browser:

```
<html>
<body>
<% response.write ("Hello World!") %>
```

</body>
</html>

The default language in ASP is VBScript. To use another scripting language, insert a language specification at the top of the ASP page:

<%@ language="javascript" %>
<html>
<body>
<%.....%>
</body>
</html>

The Response Object -The Write method of the ASP Response Object is used to send content to the browser. For example, the following statement sends the text "Hello World" to the browser:

```
<%
response. write("Hello World!")
%>
```

3.4 Forms and User Input

Request.QueryString is used to collect values in a form with method="get". Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar) and has limits on the amount of information to send.

Request.Form is used to collect values in a form with method="post". Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

3.5 ASP Variables

A variable is used to store information. If the variable is declared outside a procedure it can be changed by any script in the ASP file. If the variable is declared inside a procedure, it is created and destroyed every time the procedure is executed.

3.6 Lifetime of Variables

A variable declared outside a procedure can be accessed and changed by any script in the ASP file. A variable declared inside a procedure is created and destroyed every time the procedure is executed. No scripts outside the procedure can access or change the variable. To declare variables accessible to more than one ASP file, declare them as session variables or application variables. [3]

3.6.1 Session Variables

Session variables are used to store information about one single user, and are available to all pages in one application. Typically information stored in session variables are name, id, and preferences.

3.6.2 Application Variables

Application variables are also available to all pages in one application. Application variables are used to store information about all users in a specific application. [4]

3.7 Including Files

We can insert the content of one ASP file into another ASP file before the server executes it, with the #include directive. The #include directive is used to create functions, headers, footers, or elements that will be reused on multiple pages.

Syntax:

<!--#include virtual="somefile.inc"-->

or

<!--#include file ="somefile.inc"-->

Use the virtual keyword to indicate a path beginning with a virtual directory. If a file named "header.inc" resides in a virtual directory named /html, the following line would insert the contents of "header.inc":

<!-- #include virtual ="/html/header.inc" -->

Use the file keyword to indicate a relative path. A relative path begins with the directory that contains the including file. If you have a file in the html directory, and the file "header.inc" resides in html\headers, the following line would insert "header.inc" in your file:

<! -- #include file ="headers\header.inc" --> [4]

Use the file keyword with the syntax (..\) to include a file from a higher-level directory.

3.8 Collections of ASP Objects

3.8.1The Session Object

The Session object is used to store information about, or change settings for a user session. Variables stored in the Session object hold information about one single user, and are available to all pages in one application.

3.8.2Application Object

A group of ASP files that work together to perform some purpose is called an application. The Application object in ASP is used to tie these files together. All users share one Application object. The Application object should hold information that will be used by many pages in the application (like database connection information).

3.9 ASP Scripts

Contents - Holds every item added to the session with script commands

StaticObjects - Holds every object added to the session with the <object> tag, and a given session

Contents.Remove(item/index) - Deletes an item from the Contents collection

Contents.RemoveAll() - Deletes every item from the Contents collection

Properties

CodePage - Sets the code page that will be used to display dynamic content

SessionID - Returns the session id

Timeout - Sets the timeout for the session Method

Abandon - Kills every object in a session object

Lock - Prevents a user from changing the application object properties

Unlock - Allows a user to change the application object properties

The Response Object

The Response Object is used to send output to the user from the server. [3]

3.10 ASP Methods

AddHeader(name, value) - Adds an HTML header with a specified value

Clears- the buffered output. Use this method to handle errors. If Response.Buffer is not set to true, this method will cause a run-time error.

Sends-buffered output immediately. If Response.Buffer is not set to true, this method will cause a run-time error.

Redirects-the user to another url.

Writes- a text to the user.

BinaryRead - Fetches the data that is sent to the server from the client as part of a post request.

Server Object-The Server Object is used to access properties and methods on the server.

Property.

ScriptTimeout - Sets how long a script can run before it is terminated.

Method

CreateObject(type_of_object) - Creates an instance of an object.

Execute (path) - Executes an ASP file from inside another ASP file. After executing the called ASP file, the control is returned to the original ASP file.

CHAPTER 4

Database Design for Web Designing

Databases excel at managing and manipulating structured information. Structured information means that most ubiquitous of databases – the phone book. The phone book contains several items of information – name, address and phone number – about each phone subscriber in a particular area. Each subscriber's information takes the same form. In database parlance, the phone book is a table which contains a record for each subscriber. Each subscriber record contains three fields: name, address, and phone number. The records are sorted alphabetically by the name field, which is called the key field.

Other examples of databases are club membership lists, customer lists, library catalogues, business card files, and parts inventories. The list is, in fact, infinite. Using a database program you can design a database to do anything and increasingly, databases are being used to build Web sites. [5]

4.1 Single and multi-file databases

A database can contain a single table of information or many tables of related information. An order entry system for a business, for example, will consist of many tables:

- an orders table to track each order
- an orders detail table for tracking each item in an order
- a customer table so you can see who made the order and who to bill
- an inventory table showing the goods you have on hand
- a suppliers table, so you can see who you need to re-order your stock from
- A payments table to track payments for orders

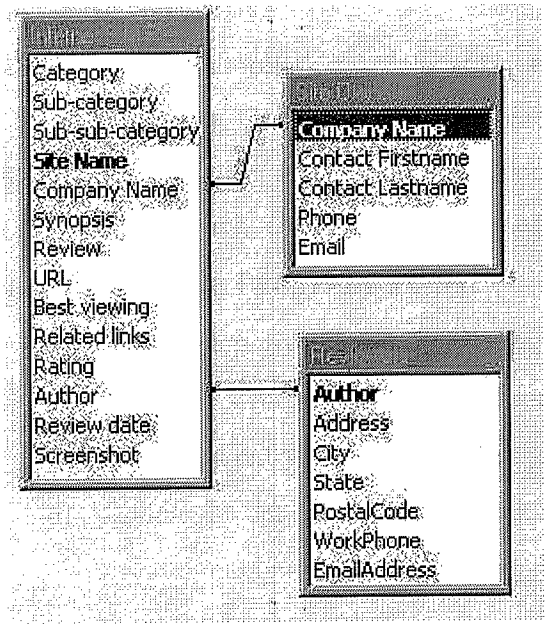
Each of these tables will be linked to one or more of the other tables, so that user can tie information together to produce reports or answer questions about the information that user has in database.

Multi-file databases like this are called relational databases. Its relational databases that provide exceptional power and flexibility in storing and retrieving information.

4.1.1 A multi-file example

Relational databases are made up of two or more tables of information which are connected in some way.

The example below shows a database used to track reviews of Internet sites



There are three tables in the database. The first is the Internet Site Reviews table which includes information about each site, the company or person maintains the site, the review itself, and who wrote the review.

The Site Owners table contains contact details for each person or organization who owns a site listed in the Site Reviews table. The two tables are linked to one another via the Company Name field, which they have in common.

The Reviewers table contains contact details for each person who writes site reviews. It's linked to the Site Reviews table via the Author field.

4.2 Database Programs

To create and maintain a computer database, user need a database program, often called a database management system, or DBMS. Just as databases range from simple, single-table lists to complex multi-table systems, database programs, too, range in complexity.

Some, such as the database component of Microsoft Works, are designed purely to manage single-file databases. With such a product user cannot build a multi-table database. User can certainly create numerous tables for storing different types of

information, but there's no way to link information from one table to another. Such programs are sometimes called **flat-file** databases, or list managers.

Other database programs, called relational database programs or RDBMS, are designed to handle multi-file databases. FileMaker Pro is a relational database that's easy to use and fairly inexpensive.

The most popular relational databases are the offerings from the big three software companies. Lotus, Corel and Microsoft each produces a full-featured relational database application available either as a standalone program or as part of its integrated suite. Lotus has Approach, Corel has Paradox and Microsoft has Access. [5]

4.2.1 Database Program Tools

A database program gives you the tools to:

- design the structure of your database
- create data entry forms so you can get information into the database
- validate the data entered and check for inconsistencies
- sort and manipulate the data in the database
- query the database (that is, ask questions about the data)
- Produce flexible reports, both on screen and on paper, that make it easy to comprehend the information stored in the database.

Most of the more advanced database programs have built-in programming or macro languages, which let you automate many of their functions.

4.3 Relational databases

Things can get more complex when you use the other type of PC database program, called a relational database. With a relational database program user can create a range of databases, from flat-file structures to demanding multi-file systems. If user is using a database in small business, a large organization, or an ambitious school project, user has to likely to need at least some of the features of a more complex relational database.

Whichever type of database program use, the most crucial step in using it is to design database structure carefully. The way user structure data will affect every other action. It will determine how easy it is to enter information into the database; how well the

database will trap inconsistencies and exclude duplicate records; and how flexibly user will be able to get information out of the database.

Creating an efficient table structure consists of breaking down your fields into simpler and simpler components there's a technical term for this process: **normalization.** For example, user initial structure:

Database design: eliminating redundant information and excluding inconsistencies. [7]

4.4 A sample design process

A sample database design process-Design a database to keep track of students' sports activities and track each activity a student takes and the fee per semester to do that activity. [6]

4.4.1 Step 1: Create an Activities table

Containing all the fields: student's name, activity and cost. Because some students take more than one activity, we'll make allowances for that and include a second activity and cost field. So our structure will be: Student, Activity 1, Cost 1, Activity 2, and Cost 2

Activities Table

Student	Activity1	Cost1	Activity2	Cost2
John Smith	Tennis	$36	Swimming	$17
Jane Bloggs	Squash	$40	Swimming	$17
John Smith	Tennis	$36		
Mark Antony	Swimming	$15	Golf	$47

We can see a glaring problem in the first field. We have two John Smiths. We need to find a way to identify each student uniquely.

4.4.2 Step 2.Uniquely Identify Records

Glaring problem can be fixed by identifying. each student uniquely by giving each one a unique Midland make a new field called ID set this as primary key by making * mark so that duplicate values not to enter.

Our table structure is now: ID, Activity 1, Cost 1, Activity 2, and Cost 2.Using a database program, we can create both table structures and then link them by the common field, ID.

We've now turned our initial flat-file design into a relational database:

Students Table

Student	ID*
John Smith	084
Jane Bloggs	100
John Smith	182
Mark Antony	219

Activities Table

ID*	Activity1	Cost1	Activity2	Cost2
084	Tennis	$36	Swimming	$17
100	Squash	$40	Swimming	$17
182	Tennis	$36		
219	Swimming	$15	Golf	$47

in this structure there is still wrong with the Activities table:

Wasted space. Some students don't take a second activity, and so we're wasting space when we store the data. Eliminate the recurring fields by making new table from the activity table

As each student can only take an activity once, this combination gives us a unique key for each record.

Our Activities table has now been simplified to: ID, Activity, and Cost. Note how the new structure lets students take any number of activities – they're no longer limited to two.

4.4.3 Step 3: Test Sample Data.

Students Table

Student	ID*
John Smith	084
Jane Bloggs	100
John Smith	182
Mark Antony	219

Activities Table

ID*	Activity*	Cost
084	Swimming	$17
084	Tennis	$36
100	Squash	$40
100	Swimming	$17
182	Tennis	$36
219	Golf	$47
219	Swimming	$15
219	Squash	$40

Our final design will thus contain three tables: the Students table (Student, ID), a Participants table (ID, Activity), and a modified Activities table (Activity, Cost).

Students Table

Student	ID*
John Smith	084
Jane Bloggs	100
John Smith	182
Mark Antony	219

Participants Table

ID*	Activity*
084	Tennis
084	Swimming
100	Squash
100	Swimming
182	Tennis
219	Golf
219	Swimming
219	Squash

Activities Table

Activity*	Cost
Golf	$47
Sailing	$50
Squash	$40
Swimming	$15
Tennis	$36

4.5 Database Design in Project

The Database that is used in interactive web design for online examination is Microsoft Access, simple database structure can be created using Ms Access as created student table named database student.mdb

For this creation open the Microsoft access and choose blank database from the starting menu then on prompt of the name, give the name Student and while you saving this database name Student.mdb and save it the same directory as the web page connection to the database going to be.

First look of Ms access dialogue box is shown in figure 4.1, from here select create table in design view.

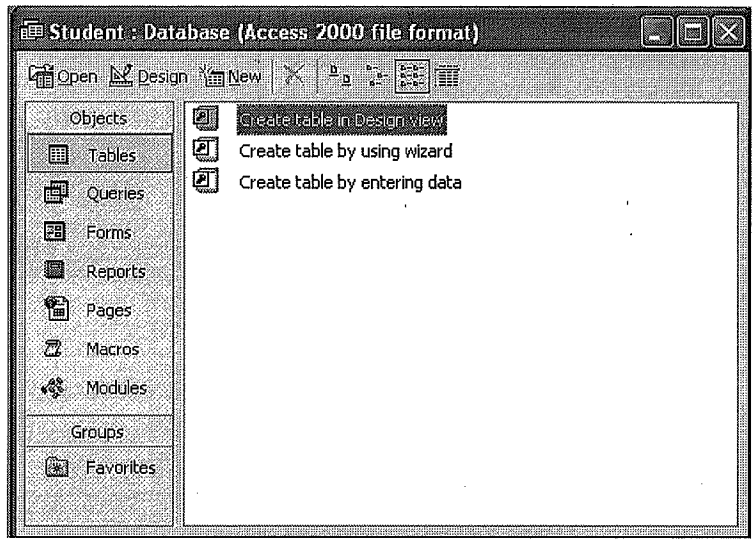


Figure 4.1

Simple example of one of my tables name studentinfo is shown in figure 4.2, fields and their data types are shown.

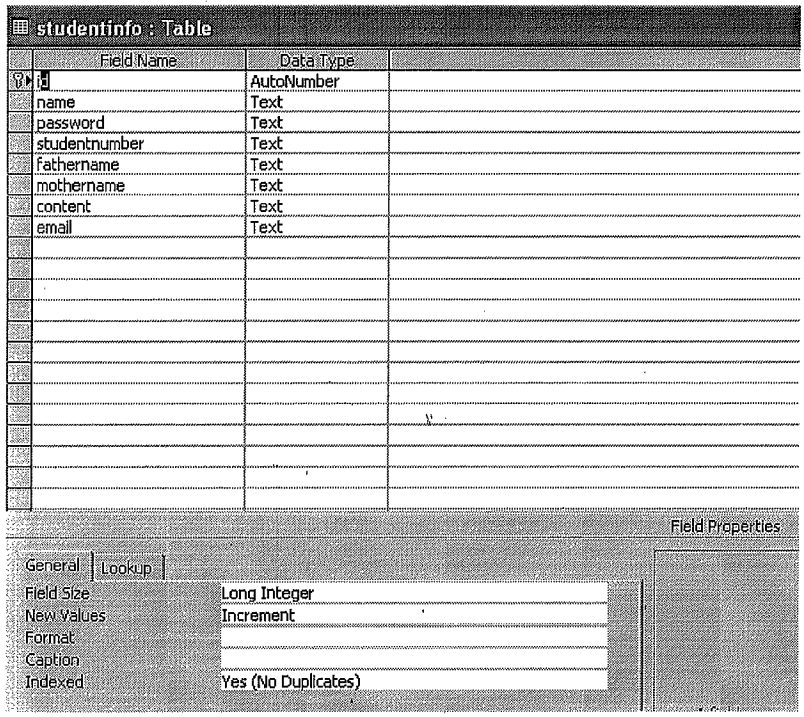


Figure 4.2

Once the entire fields have been created and data types and primary key set save the table as studentinfo.

Now the table has been created some test data can be entered into table. This can be done by double clicking on the new table that already created in the main dialogue box.
By this we can manipulate our database.

CHAPTER 5

ADO

ADO stands for ActiveX Data Objects ADO is a Microsoft technology ADO is a Microsoft Active-X component . ADO is a programming interface to access data in a database

5.1 Accessing a Database from an ASP Page

The common way to access a database from inside an ASP page is to:

- Create an ADO connection to a database
- Open the database connection
- Create an ADO recordset
- Open the recordset
- Extract the data you need from the recordset
- Close the recordset
- Close the connection

5.1.1 Create a DSN-less Database Connection

The easiest way to connect to a database is to use a DSN-less connection. A DSN-less connection can be used against any Microsoft Access database on your web site.

If you have a database called "northwind.mdb" located in a web directory like "c:/webdata/", you can connect to the database with the following ASP code:

```
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
%>
```

5.1.2 Create an ODBC Database Connection

If there is an ODBC database called "northwind" that can be connected to the database with the following ASP code:

```
<%
set conn=Server.CreateObject("ADODB.Connection")
```

```
conn.Open "northwind"
%>
```

With an ODBC connection, connection to any database can be established, on any computer in any network, as long as an ODBC connection is available. [3]

5.2 An ODBC Connection to an MS Access Database

Creation of database connection to a MS Access Database:

- Open the ODBC icon in your Control Panel.
- Choose the System DSN tab.
- Click on Add in the System DSN tab.
- Select the Microsoft Access Driver. Click Finish.
- In the next screen, click Select to locate the database.
- Give the database a Data Source Name (DSN).
- Click OK.

Note that this configuration has to be done on the computer where web site is located. If there is running Personal Web Server (PWS) or Internet Information Server (IIS) on computer, the instructions above will work, but if web site is located on a remote server, you have to have physical access to that server. [3]

5.3 The ADO Connection Object

The ADO Connection object is used to create an open connection to a data source. Through this connection, user can access and manipulate a database.

5.3.1 ADO Recordset

To be able to read database data, the data must first be loaded into a recordset.

5.3.2 Create an ADO Table Recordset

After an ADO Database Connection has been created, it is possible to create an ADO Recordset.

Suppose there is a database named "Northwind", we can get access to the "Customers" table inside the database with the following lines:

```
<%
set conn=Server.CreateObject("ADODB.Connection")
```

```
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
set rs=Server.CreateObject("ADODB.recordset")
rs.Open "Customers", conn
%>
```

5.3.3 Create an ADO SQL Recordset

Access to the data in the "Customers" table using SQL:

```
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
set rs=Server.CreateObject("ADODB.recordset")
rs.Open "Select * from Customers", conn
%>
```

5.4 Extract Data from the Recordset

After a recordset is opened, we can extract data from recordset.

Suppose we have a database named "Northwind", we can get access to the "Customers" table inside the database with the following lines:

```
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
set rs=Server.CreateObject("ADODB.recordset")
rs.Open "Select * from Customers", conn
for each x in rs.fields
  response.write(x.name)
  response.write(" = ")
  response.write(x.value)
next
```

```
%>
```

5.5 The ADO Recordset Object

The ADO Recordset object is used to hold a set of records from a database table.

5.6 ADO Queries

SQL can be used to create queries to specify only a selected set of records and fields to view.

Display Selected Data-Only the records from the "Customers" table that have a "Companyname" that starts with an A (remember to save the file with an .asp extension) can be displayed:

```
<html>
<body>
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
set rs=Server.CreateObject("ADODB.recordset")
sql="SELECT Companyname, Contactname FROM Customers
WHERE CompanyName LIKE 'A%'"
rs.Open sql, conn
%>
<table border="1" width="100%">
 <tr>
 <%for each x in rs.Fields
  response.write("<th>" & x.name & "</th>")
next%>
</tr>
<%do until rs.EOF%>
 <tr>
 <%for each x in rs.Fields%>
  <td><%Response.Write(x.value)%></td>
```

```
<%next
 rs.MoveNext%>
 </tr>
<%loop
rs.close
 conn.close%>
</table>
</body>
</html>
```

5.7 ADO Add Records

SQL INSERT INTO command can be used to add a record to a table in a database.

When the user presses the submit button the form is sent to a file called "demo_add.asp".

The "demo_add.asp" file contains the code that will add a new record to the Customers table: [3]

```
<html>
<body>
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
sql="INSERT INTO customers (customerID,companyname,"
sql=sql & "contactname,address,city,postalcode,country)"
sql=sql & " VALUES "
sql=sql & "('" & Request.Form("custid") & "',"
sql=sql & "'" & Request.Form("compname") & "',"
sql=sql & "'" & Request.Form("contname") & "',"
sql=sql & "'" & Request.Form("address") & "',"
sql=sql & "'" & Request.Form("city") & "',"
sql=sql & "'" & Request.Form("postcode") & "',"
sql=sql & "'" & Request.Form("country") & "')"
```

```
on error resume next
conn.Execute sql,recaffected
if err<>0 then
 Response.Write("No update permissions!")
else
 Response.Write("<h3>" & recaffected & " record added</h3>")
end if
conn.close
%>

</body>
</html>
```

5.8 ADO Update Records

SQL UPDATE command can be used to update a record in a table in a database.

Update a Record in a Table-If the user clicks on the button in the "customerID" column he or she will be taken to a new file called "demo_update.asp". The "demo_update.asp" file contains the source code on how to create input fields based on the fields from one record in the database table. It also contains a "Update record" button that will save your changes:

```
<html>
<body>
<h2>Update Record</h2>
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
cid=Request.Form("customerID")
if Request.form("companyname")="" then
 set rs=Server.CreateObject("ADODB.Recordset")
 rs.open "SELECT * FROM customers WHERE customerID='" & cid & "'",conn
```

```
%>
<form method="post" action="demo_update.asp">
<table>
<%for each x in rs.Fields%>
<tr>
<td><%=x.name%></td>
<td><input name="<%=x.name%>" value="<%=x.value%>"></td>
<%next%>
</tr>
</table>
<br /><br />
<input type="submit" value="Update record">
</form>
<%
else
sql="UPDATE customers SET "
sql=sql & "companyname='" & Request.Form("companyname") & "',"
sql=sql & "contactname='" & Request.Form("contactname") & "',"
sql=sql & "address='" & Request.Form("address") & "',"
sql=sql & "city='" & Request.Form("city") & "',"
sql=sql & "postalcode='" & Request.Form("postalcode") & "',"
sql=sql & "country='" & Request.Form("country") & "'"
sql=sql & " WHERE customerID='" & cid & "'"
on error resume next
conn.Execute sql
if err<>0 then
 response.write("No update permissions!")
else
 response.write("Record " & cid & " was updated!")
end if
```

```
end if
conn.close
%>
</body>
</html>
```

5.9 ADO Delete Records

SQL DELETE command can be used to delete a record in a table in a database.

If the user clicks on the button in the "customerID" column he or she will be taken to a new file called "demo_delete.asp". The "demo_delete.asp" file contains the source code on how to create input fields based on the fields from one record in the database table. It also contains a "Delete record" button that will delete the current record: [3]

```
<html>
<body>
<h2>Delete Record</h2>
<%
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.4.0"
conn.Open "c:/webdata/northwind.mdb"
cid=Request.Form("customerID")
if Request.form("companyname")="" then
 set rs=Server.CreateObject("ADODB.Recordset")
 rs.open "SELECT * FROM customers WHERE customerID='" & cid & "'",conn
 %>
 <form method="post" action="demo_update.asp">
 <table>
 <%for each x in rs.Fields%>
 <tr>
 <td><%=x.name%></td>
 <td><input name="<%=x.name%>" value="<%=x.value%>"></td>
 <%next%>
```

```
 </tr>
 </table>
 <br /><br />
 <input type="submit" value="Delete record">
 </form>
<%
else
 sql="DELETE FROM customers"
 sql=sql & " WHERE customersID='" & cid & "'"
 on error resume next
 conn.Execute sql
 if err<>0 then
  response.write("No update permissions!")
 else
  response.write("Record " & cid & " was deleted!")
 end if
end if
conn.close
%>
</body>
</html>
```

CHAPTER 6

Interactive Web Page Design for Online Examination

In order to run our project we first have to set our IIS server's web directory to the folder where our website pages contain as shown in figure 6.1,

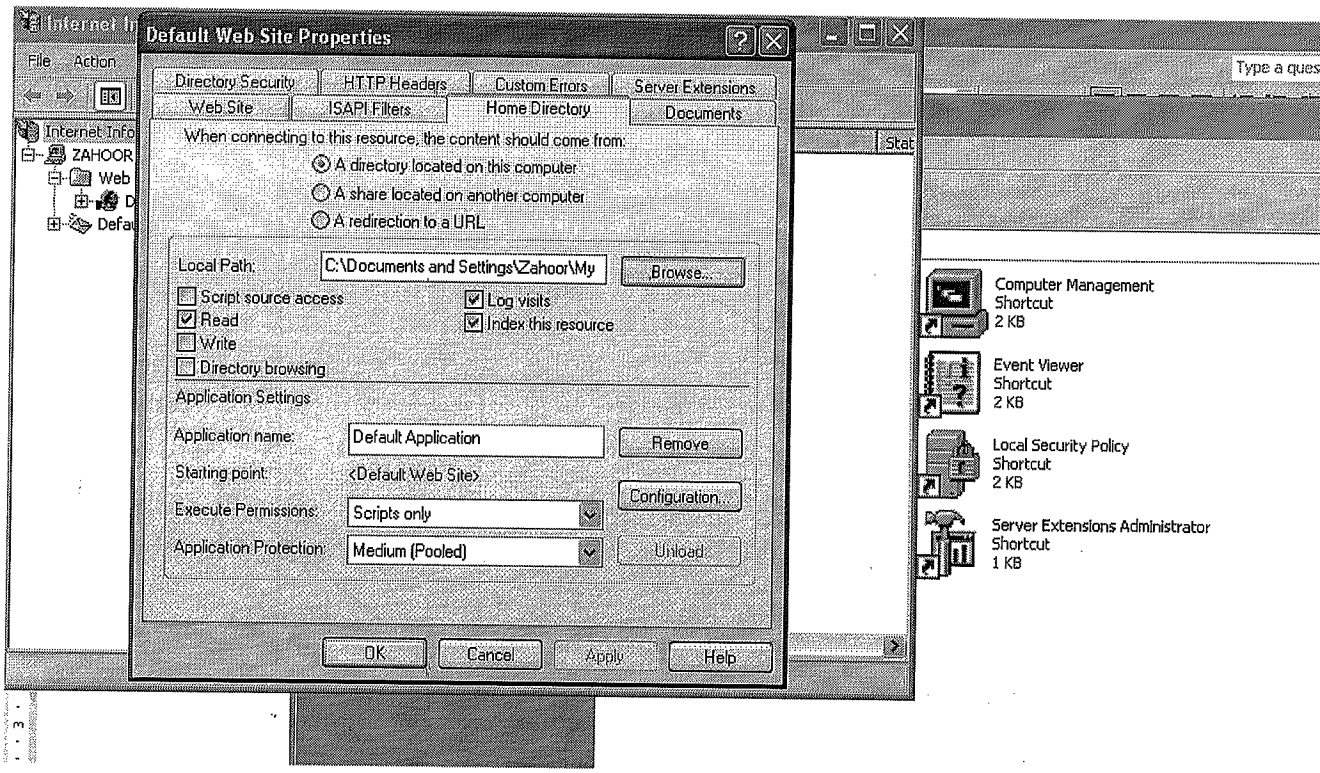


Figure 6.1

Now we can run site on local computer by giving address http://localhost/ in the web browser.

The diagram 6.2 represents in short the order of the pages,

6.1 Flow Diagram for online Examination

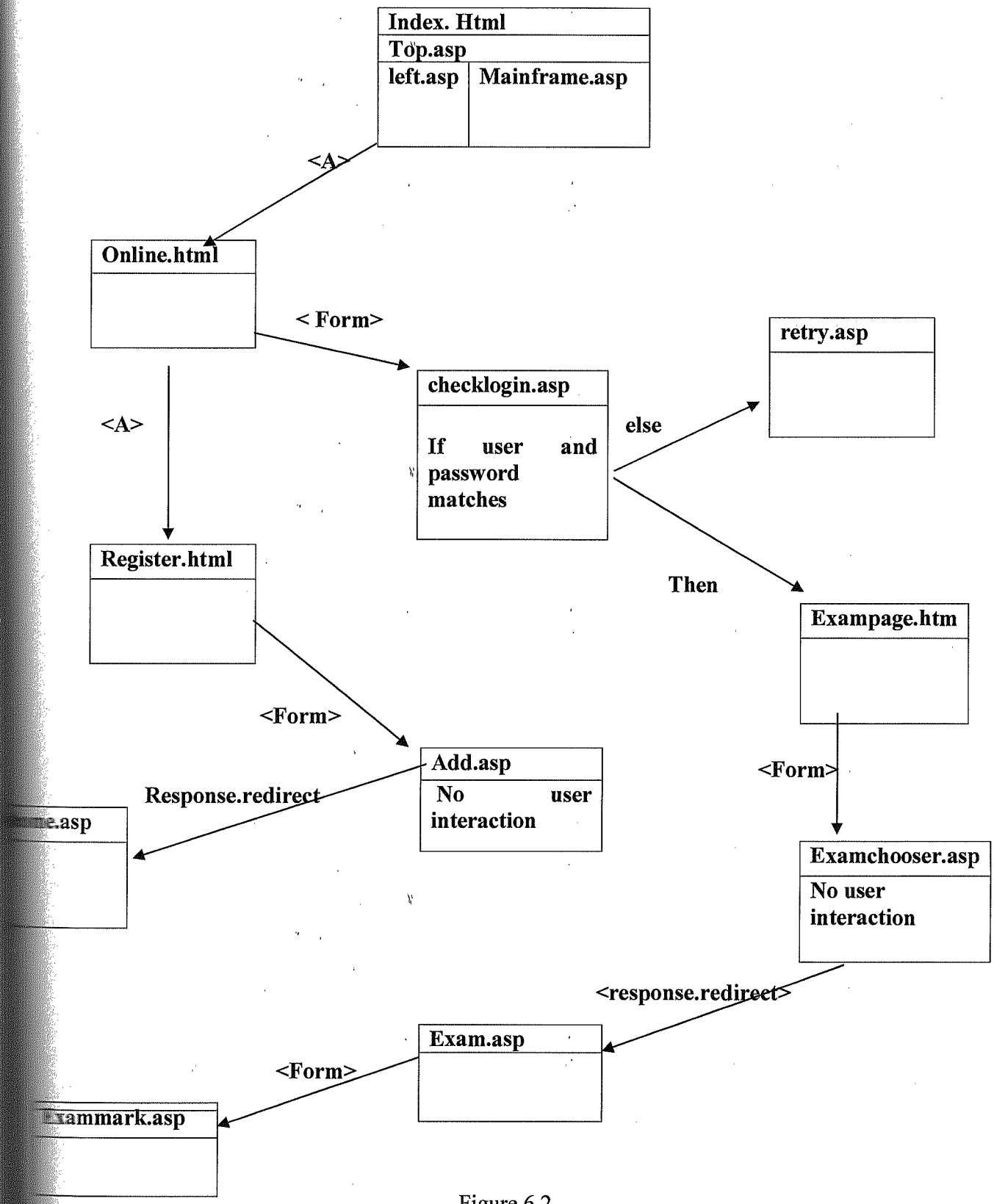

Index. Html
Top.asp
left.asp
Mainframe.asp
<A>
Online.html
< Form>
retry.asp
checklogin.asp
<A>
else
If user and password matches
Register.html
Then
Exampage.htm
<Form>
Add.asp
No user interaction
<Form>
Response.redirect
Examchooser.asp
No user interaction
<response.redirect>
Exam.asp
<Form>
ammark.asp

Figure 6.2

The first page is default page named here **index.html.**

6.2 Index.html

The very first page in my site where simple html and hyperlink is used. Frame tag is used to divide the page into three parts.

The top frame is addressing to the top.asp, which contains simple html tables and left frame links to the left.asp page where simple information is stored using tables and linking is given. The main frame points to the main.asp page which contains the information about the site and that is done by putting information in table elements.

When we click the link on the online examination this link refers to the page named online.html

Simple JavaScript examples are used to show the clock in the top.asp page. JavaScript is case sensitive so the code of JavaScript is as follows

```
<script type="text/javascript">
var timer = null
function stop()
{
clearTimeout(timer)
}
function start()
{
var time = new Date()
var hours = time.getHours()
var minutes = time.getMinutes()
minutes=((minutes < 10) ? "0" : "") + minutes
var seconds = time.getSeconds()
seconds=((seconds < 10) ? "0" : "") + seconds
var clock = hours + ":" + minutes + ":" + seconds
document.forms[0].display.value = clock
timer = setTimeout("start()",1000)
}
</script>
```

In order to execute the JavaScript in the body section we have to call the function as follows

onload="start()" onunload="stop()

where to display that clock we should use the form tag and input field where the clock is displayed.

<input name="display" size="10">

In Online.html page simple three text field is used and form method "post" is used that transfer information entered to the page named checklogin.asp

6.3 Checklogin.asp

```
<%dim a, b
    dim objConn, dbpath, result, query, mydata,estab
  a=request("T1")
   b=request("T2")
   dbpath="C:\Project\Student.mdb"
 estab="Driver={Microsoft Access Driver (*.mdb)}; DBQ="& dbpath
  set objConn = server.createobject("ADODB.Connection")
  objConn.Open estab
  query="select content from studentinfo where name='"& a &"' and password ='"& b
&"'"
  set mydata = objConn.Execute(query)
   if mydata.eof then
response.redirect "retry.asp"
  else
     result = mydata(0)
  response.redirect "Exampage.html"
     end if
  mydata.close
 %>
```

It initiates the variable a,b as shown ,the % sign showing that scripting is used .The field has names in online.html i.e. Name has name T1 and Password has name T2 so same name refers here so that it contain the information that is entered, After assigning these two values to variables it established the connection to the database as shown by the syntax. if it matches with database fields it will take us to the page "exampage.html" else it redirects us to the page named '**retry.asp**".

If user is not registered he/she can register by just clicking the new user hyperlink, this will divert to the page name register.htm.The form method is used to send information filled in the fields as shown figure 6.3.

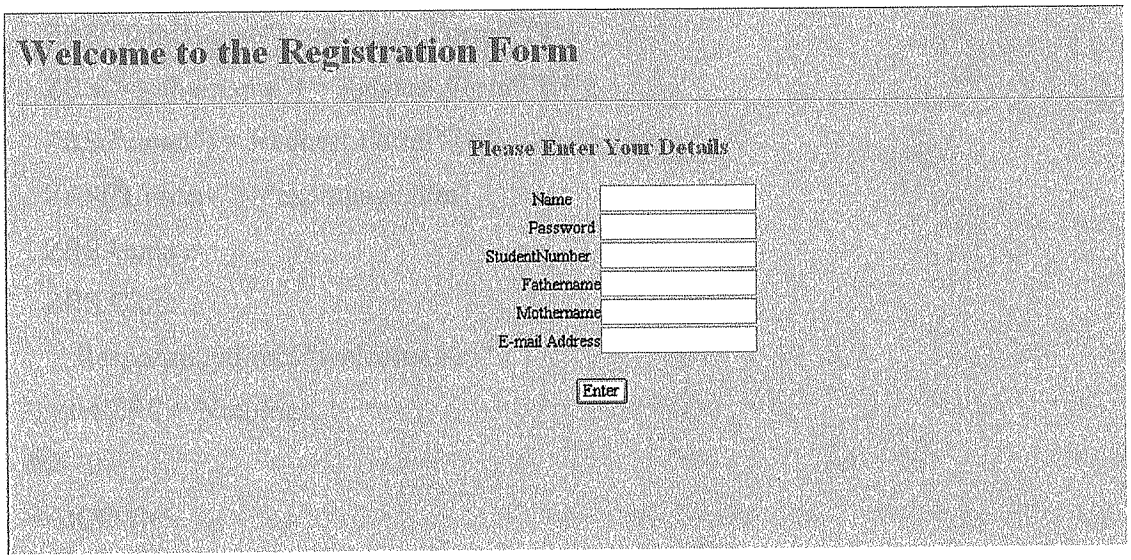


Figure 6.3

After Filling the fields when user submit the form ,it then enter the details in the database named Student by interacting with the Add. ASP Page that has no directly interaction with user. The Code of **Add.asp** is as follows.

```
<!-- #include file="DataStore.asp" -->
<!-- #include file="adovbs.inc" -->
<HTML>
<HEAD>
<TITLE>Adding User</TITLE>
</HEAD>
<BODY>
<%
 Dim ZRS, ID
 Set ZRS = Server.CreateObject ("ADODB.Recordset")
 ZRS.Open "studentinfo", strConnect, adOpenStatic, adLockOptimistic, adCmdTable
 ZRS.MoveLast
 ID = ZRS("id") + "1"
 ZRS.AddNew
 ZRS("name") = Request.Form("T1")
 ZRS("password") = Request.Form("T2")
 ZRS("studentnumber") = Request.Form("T3")
```

```
ZRS("fathername") = Request.Form("T4")
ZRS("mothername") = Request.Form("T5")
ZRS("email") = Request.Form("T6")
ZRS("Content")="exampage.htm"
ZRS.Update
ZRS.Close
   Response.redirect "welcome.asp"
  Set ZRS = Nothing
%>
</BODY>
</HTML>
```

"exampage.html" contains the select field where we can select the desired exam. The selection done as follows

e.g. from the list box user chooses java exam and submit then this information is verified by another page that user has no interact named "**examchooser.asp**" the code of that page is as follows

6.4 Examchooser.asp

```
<%@ LANGUAGE="VBScript"%>
<%
Dim x
x=request.Form("paper")
%>
<%
if x="Java" then
Response.Redirect "exam.asp?exam=1"
%>
<%
else
if x="Software" then
Response.Redirect "exam.asp?exam=2"
%>
<%
else
if x="Circuit" then
Response.Redirect "exam.asp?exam=3"
%>
<%
```

```
else if x="C++" then
Response.Redirect "exam.asp?exam=4"
%>
<%
end if
end if
end if
end if
%>
```

First it assigns the value that is submitted by the exampage.html to the variable x and search for the value in the if else statement if it matches with anyone it will divert to that page address.e.g, if user select the java option then it will diverted to the exam. asp but with the option 1 that specifying exam in the database. the code of **exam.asp** is as follows

```
<HTML>
<HEAD>
<TITLE>
Examination Page
</TITLE>
<link rel="stylesheet" type="text/css" href="style.css">
</HEAD>
<%
Dim dbpath,strProv,Conn,RS
 dbpath="C:\Project\Student.mdb"
  strProv="Driver={Microsoft Access Driver (*.mdb)}; DBQ="& dbpath
  set Conn = server.createobject("ADODB.Connection")
  Conn.Open strProv

'now find the exam specified by the URL command line
'e.g. exam.asp?exam=1 means show exam 1
set RS = Conn.Execute("SELECT * FROM [Exam] WHERE [Examid]=" &
request.querystring("Exam"))
%>
<p align="center" ><font size="6" color="#CC0000">
<b><% =rs("Examname") %></b></p>
<hr>
<% =rs("Examheader") %> </p>

<form action="exammark.asp?Exam=<% =request.querystring("Exam") %>"
method="post">

<font size="3" color="#000000">


<%
```

```
rs.close
'now start the main loop to find all the questions
set RS = Conn.Execute("SELECT * FROM [question] WHERE [Examid]=" &
request.querystring("Exam") & " ORDER BY [Number]")
if not RS.EOF then
        RS.movefirst
        do
        %><font size="4" color="#CC0000"><% response.write "<b>" & rs("Number")
& ". " & rs("Question") & "</b><p>" & chr(13)
        %></font><%
                'now display the available options
                set ORS = Conn.Execute("SELECT * FROM [option] WHERE
[QuestionID]=" & RS("QuestionID") )
                if not ORS.EOF then
                        ORS.movefirst
                        do
          response.write "<input type=Radio Name=""Question" & RS("QuestionID") &
""" Value=""Answer" & ORS("OptionID") & """>" & ORS("Option") & "<br>" &
chr(13)
                                ORS.movenext
                        loop until ORS.EOF
                end if
                response.write "<p>"
                RS.movenext
        loop until RS.EOF
end if
conn.close
%>
</p>
<hr>

<input type=submit value="Submit"><font size="5" color="#cc0000"> Click To See
Your Result .....</font>
</form>
</font></font>
</font>
</BODY></HTML>
```

According to this code, First of all database connection is established so that information contained in the database records can be put into the browser.

After making the connection it searches for the records that is sent by the examchoose.asp page it checks the exam id if its 1 then java exam should be taken.

Because in the database the id of java exam is 1.To put the question and then choices we respectively select the records in the database of table question and option.

The page is shown in figure 6.4.

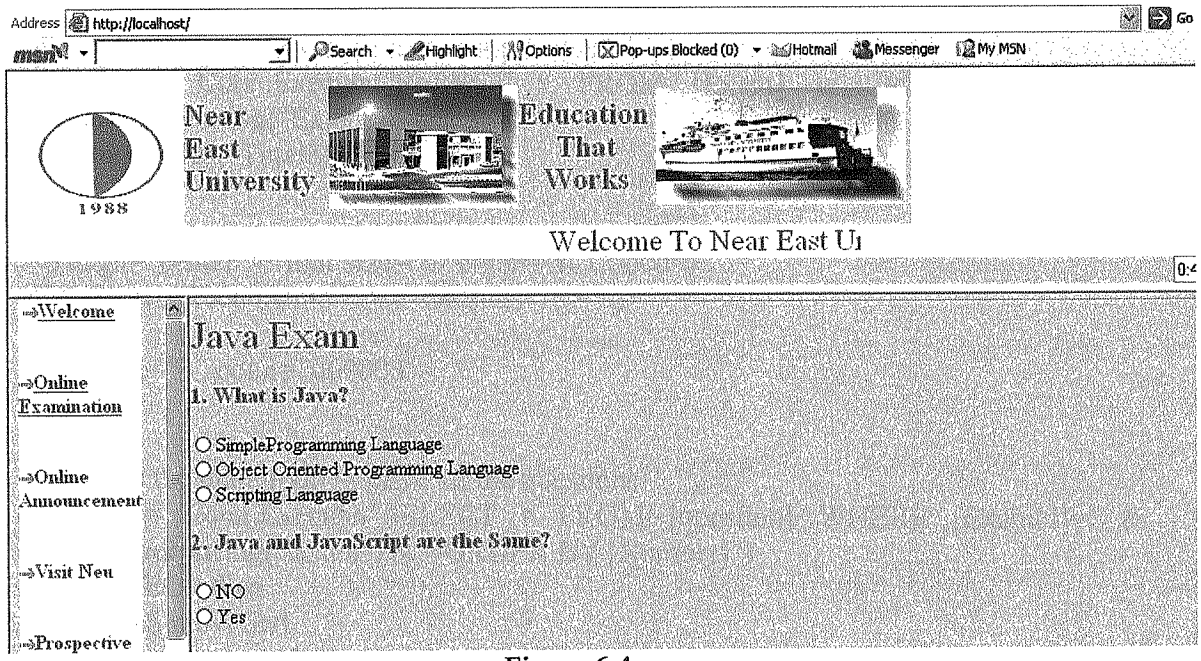


Figure 6.4

While Answering to all the question and submiteed then page goes to another page named **exammark.asp**

6.5 Exammark.asp

```
<HTML>
<HEAD>
<TITLE>
Exam
</TITLE>
<script type="text/javascript">
function printpage()
{
window.print()
}
</script>
<link rel="stylesheet" type="text/css" href="style.css">
</HEAD>
<BODY BGCOLOR="#CCFFCC">

<%
Dim dbpath,strProv,Conn,RS
 dbpath="C:\Project\Student.mdb"
   strProv="Driver={Microsoft Access Driver (*.mdb)}; DBQ="& dbpath
   set Conn = server.createobject("ADODB.Connection")
   Conn.Open strProv
```

```
'now find the quiz specified by the URL command line
'e.g. exammark.asp?exam=1 means mark exam 1
set RS = Conn.Execute("SELECT * FROM [Exam] WHERE [Examid]=" &
request.querystring("Exam"))
totalmark = RS("TotalMark")
%>

<p align="center"><font size="6" color="#CC0000">
<b><% =rs("Examname") %> - Results</b></p>
<hr>
<font size="3" color="000000">
<%
rs.close
'now start the main loop to mark all the questions
set RS = Conn.Execute("SELECT * FROM [question] WHERE [Examid]=" &
request.querystring("Exam") & " ORDER BY [Number]")
if not RS.EOF then
        RS.movefirst
        do
                %><font size="4" color="#CC0000"><%response.write "<b>" &
rs("Number") & ". " & rs("Question") & "</b><br>" & chr(13)
                %></font><%
                'now check the users answer
                answer = request.form("Question" & rs("QuestionID"))
                if len(answer)>0 then
                        'now get rid of the answer, we just want the number on the end
                        answernum = right(answer,len(answer)-6)
                        set ORS = Conn.Execute("SELECT * FROM [option] WHERE
[OptionID]=" & answernum )
                        if not ORS.EOF then
                                ORS.movefirst
                                response.write " Your Answer : " & ORS("Option") &
"<br>"
                                if ORS("tick") then 'show a tick
                                        response.write "<img src=""ASPtick.gif"">"
                                else    'show a cross
                                        response.write "<img src=""ASPcross.gif"">"
                                end if
                                response.write " " & ORS("mark") & " : " &
ORS("Response") & chr(13)
                                summark=summark + ORS("mark")
                        end if
                else
                        response.write "You did not answer this question.."
                end if
```

```
                response.write "<p>"
                RS.movenext
        loop until RS.EOF
end if
%></font></font>
<font size="6" color="#CC0000">
<% if summark<60 then
response.write "Bad Performance, Grade FF"
end if
if summark<=60 and summark>=70 then
response.write  "Congratulation Passed, Grade DD"
end if
if summark=80 then
response.write "Good,Grade BB"
end if
if summark>=90 then
response.write "Excellent, Grade AA"
end if

%>
<%response.write "<hr>You have scored " & summark & "/" & totalmark & "."
conn.close
%>
</font></font>
</p>
<form>
<input type="button" value="Print this page" onclick="printpage()">
</form>
</BODY></HTML>
```

The code checks the submitted required information and then shows the result on this page the code is as well explained in the comments.And on behalf of marks obtained simple putitting if then statement comments are shown to the user tio improve their performance or appreciate.

Similarly we can make many exams by just putting entries to the database and make them available to the webpages through database connection.

6.6 DATABASE

We use the database and list of tables are shown in the figure 6.5.

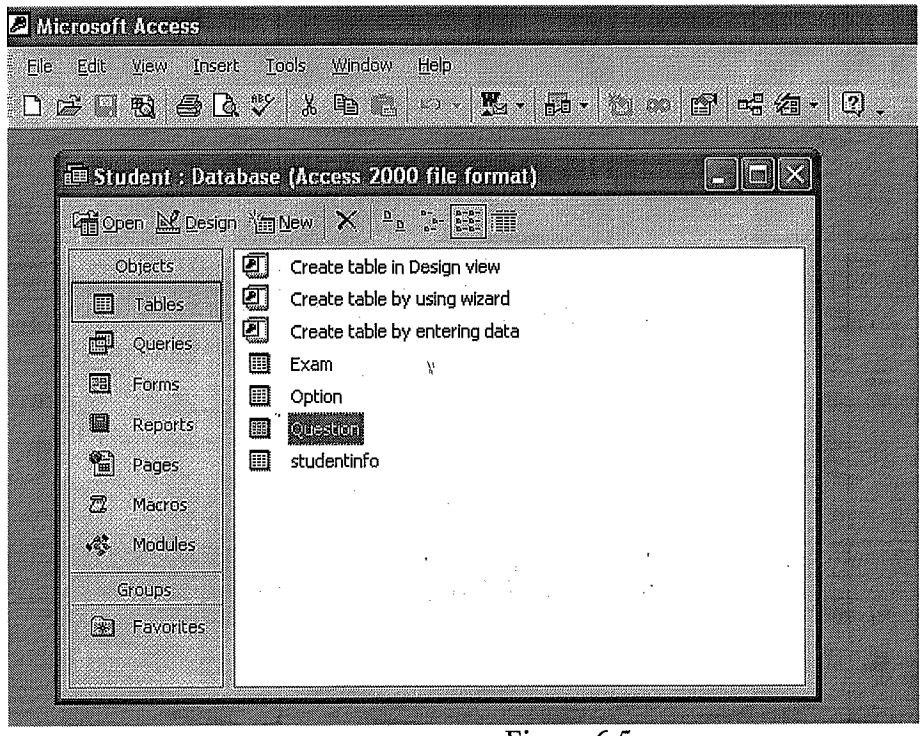


Figure 6.5

This student database contains total 4 tables.

Studeninfo:

Studeninfo has 8 fields and field names and their types are shown in the figure 6.6.

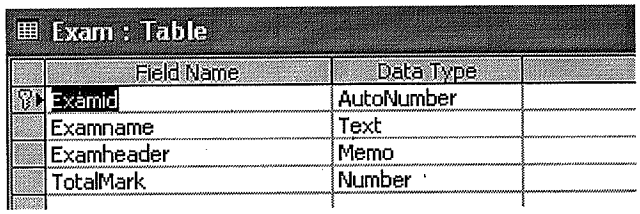
studentinfo : Table

Field Name	Data Type
id	AutoNumber
name	Text
password	Text
studentnumber	Text
fathername	Text
mothername	Text
content	Text
email	Text

Figure 6.6

Where id is autonumber and we set this field as autonnumber in order not to have unique primary key.

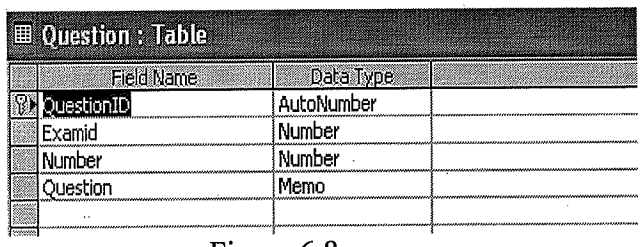
Similarly other table fields are shown as below



Exam : Table

Field Name	Data Type
Examid	AutoNumber
Examname	Text
Examheader	Memo
TotalMark	Number

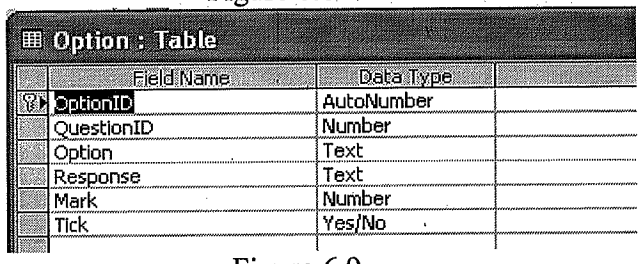
Figure 6.7



Question : Table

Field Name	Data Type
QuestionID	AutoNumber
Examid	Number
Number	Number
Question	Memo

Figure 6.8



Option : Table

Field Name	Data Type
OptionID	AutoNumber
QuestionID	Number
Option	Text
Response	Text
Mark	Number
Tick	Yes/No

Figure 6.9

We can add many exams into our website by just making entries into our database that looks more easy and attractive.

CONCLUSION

Today the web pages are the vital source for the information, trading, education, tourism and for many more fields. There are millions of web pages containing different kind of information for different categories. The idea behind is how to use that source for informative and less time consuming purposes. In online examination system example, put the same thing in practical by making interactive web design for online examination. This all could be possible by appropriate use of html, asp, JavaScript and database techniques. In online examination simple html tags are used and asp is used to retrieve information stored in the database while JavaScript gives more dynamic effect to the system just like to print the user result etc. This is the example of online examination systems where user can login if he/she is registered or for new user they can register their details and take examination for informative purpose or accessing his/her abilities in order to appear in online examination before the real examination.

Interactive web design for online examination can be more complex and reliable systems ,example the register user can take the examination only once and records of the previous exam result is saved in account of that user, the main ideas is that whatever the complexity of the system the basic techniques remain the same .Web pages has their own importance which can not be neglected and in order to make more attractive and effective web pages the techniques of html, asp JavaScript and database gradually getting complex.

REFERENCES

[1] Ian Graham "Introduction to Html"

(www.utoronto.ca)

[2] Deitel "Internet & World Wide Web How to Program"

Deitel Deitel Goldberg

[3] "www.w3schools.com"

[4] Eric A. Smith "Active Server Pages BIBLE"

[5] Database "http://databases.about.com/ "

[6] David Buser, John Kauffman, Chris ullman "Beginning Asp 3.0"

By Shroff Publisher

[7] John Carter "Database Design And Programming"

Mc GrawHill Edition