SUPORT DE CURS
PHP și MYSQL

LIMBAJUL PHP
Noţiuni introductive

În cazul paginilor statice create cu HTML utilizatorul poate vedea doar imagini şi text. Folosind PHP însă vom putea crea aşa numitele pagini dinamice în care apare elementul de interactivitate: se pot prelua date din formulare realizate în PHP, citi, prelucra şi afişa. De asemenea se poate face interacţiunea cu o bază de date creată pe un server.

Pentru a putea lucra cu PHP trebuie în primul rând să avem acces la un server pe care rulează PHP (acesta poate fi server separat sau server virtual creat pe calculatorul personal). Acest server (cel virtual) poate fi instalat folosind pachetul WAMP care este free pe internet.

Observaţii:

- Codul PHP poate fi scris folosind editorul Notepad;
- În cazul în care lucrăm cu Wamp, fişierul care conţine codul programului php trebuie să aibă extensia php şi să fie salvat în subfolderul WWW din folderul WAMP;
- Lansarea în execuţie se face utilizând un browser (expl Internet Explorer) în care se va scrie în bara de adrese: http://localhost/<nume_fisier.php> (Expl: http://localhost/exemplu.php).

Crearea primului script php:

```
<?php
echo ”Primul script php”;
?>
```

Obs: echo - afişează un mesaj

Combinarea codului HTML cu cod PHP:

Acest lucru se întâmpla foarte des. În interiorul unui script HTML poate exista script PHP delimitat de etichetele *<?php* şi *?>* însă fişierul final trebuie sa fie PHP.

Exemplu:

```
<html>
<body>
Text combinat HTML cu PHP
<?php
echo ”Text scris cu PHP”;
?>
</body>
</html>
```

Comentarii în scripturi:

- un tip de comentariu pentru un rând
// - un alt tip de comentariu pentru un rând

VARIABILE ŞI CONSTANTE ÎN PHP

Variabilele sunt mărimi care pot lua diferite valori. Numele dat unei variabile începe întotdeauna cu simbolul $ urmat de o litera sau caracterul „_”. Nu poate fi precedat niciodată de o cifră.

Definim o variabilă astfel:

$<variabila>=<valoare>;

Observaţie: Tipul de date este specificat prin valoarea atribuită variabilei iniţiale.

Exemplul 1:

```
$a=1; // număr întreg (zecimal)
$b=3.14: // număr real
$c=”exemplu de text”; //şir de caractere
$c=’exemplu de text’; //şir de caractere
$matrice = array('alb','rosu','verde'); //matrice
$a=TRUE; //variabilă booleana
```

Exemplul 2:

```
……..
$a=1;
echo “Numarul  ”, $a,”<br>”;
……..
```

Se va afişa : Numarul 1

În cazul unui şir de caractere definit prin ghilimele duble se pot folosi următoarele secvenţe:

Secventa	**Semnificatie**
\n	rand nou
\r	sfarsit de linie
\t	tab orizontal
\\	backslash
\$	simbol dolar
\"	ghilimele duble

Variabile predefinite

$_SERVER['REMOTE_ADDR'] – afişează adresa IP a vizitatorului
$_SERVER['HTTP_USER_AGENT']- afişează informatii despre browserul folosit
$_SERVER['HTTP_REFERER'] – afişează pagina vizitata anterior
$_SERVER['SERVER_NAME'] – afişează numele serverului
$_SERVER['SCRIPT_NAME'] – afişează numele scriptului

Crearea constantelor

Constantele sunt create cu ajutorul funcţiei *define*.
Expl: *define ("PI", 3.14);*

OPERATORI FOLOSIŢI ÎN PHP

În PHP există următoarele tipuri de operatori:

1. Operatori aritmetici:

Denumire operator	**Operator**
Adunare	+
Scădere	-
Înmulţire	*
Împărţire	/
Modulo (restul împărţirii)	%

2. Operatori de comparaţie:

Denumire operator	**Operator**	**Obs.**	**Expl.**
Egalitate	= =	TRUE dacă a şi b sunt egale	$a= =$b
Identic	= = =	TRUE dacă a şi b sunt egale Şi de acelaşi tip	$a= = =$b
Mai mare ca	>	TRUE dacă a este mai mare decât b	$a>$b
Mai mic ca	<	TRUE dacă a este mai mic decât b	$a<$b
Diferit de	<>	TRUE dacă a diferit de b	$a<>$b
Diferit de	!=	TRUE dacă a diferit de b	$a!=$b
Mai mic sau egal	<=	TRUE dacă a este mai mic sau egal cu b	$a<=$b
Mai mare sau egal	>=	TRUE dacă a este mai mare sau egal cu b	$a>=$b

3. Operatori logici:

Denumire operator	**Operator**	**Obs.**	**Expl.**
NOT	!	TRUE daca $a nu este TRUE	$a!$b
AND	&&	TRUE daca si $a si $b sunt TRUE	$a&&$b
AND	and	TRUE daca si $a si $b sunt TRUE	$a and $b
OR	||	TRUE daca ori $a ori $b sunt TRUE	$a || $b
OR	or	TRUE daca ori $a ori $b sunt TRUE	$a or $b
XOR	xor	TRUE daca ori $a ori $b sunt TRUE dar	$a xor $b

		nu amandoua	

Observaţii: Numerele aleatoare sunt returnate de funcţia ***rand()***
Expl:
……
echo rand(); // se generează un număr aleator
echo rand(1,50) // se generează un număr aleator de la 1 la 50

STRUCTURI DE CONTROL

1. ATRIBUIREA
   Sintaxa este:

   $<nume_variabila>=<valoare>
   unde valoare poate fi o constanta, o variabila sau o expresie

2. DECIZIA
   Sintaxa IF este:

   if <conditie_logica>
   <instructiune1>
   else
   <instructiune2>

   Mod de lucru:
   - se testeaza conditia logica. Daca este adevarata se executa instructiunea 1 dacaeste falsa se executa instructiunea 2

   Sintaxa SWITCH este:

   $variabila=<valoare>
   switch($variabila)
   {
   case <val1>:
      <instructiune1>;
      break;
   case <val2>:

<instructiune2>;
break;
.....
case <valn>:
<instructiunen>;
break;
default:
<alta_instructiune>

Mod de lucru:
- se citeste valoarea unei variabile
- se va executa instructiunea corespunzatoae cazului cand <vali>=<valoare>
- daca nu este corespunde nici un ca se va executa instructiunea din default

3. REPETITIA
Sintaxa WHILE este:

while (<conditie_logica>)
<instructiuni>

Mod de lucru:
/ se testeaza conditia logica. Cit timp este adevarata se executa instructiunile din cadrul structurii

Sintaxa DO ….WHILE este:

do //repeta
<instructiune1>
<instructiune2>
.....
<instructiunen>
while <conditie_logica> //cit timp conditia este adevarata

Mod de lucru: Se executa instructiunile cat timp conditia este adevarata

Sintaxa FOR este:

for (<contor=valoare_initiala>; <contor=valoare_finala>; <incrementeaza_contor>)
<instructiuni>

Mod de lucru:
- cit timp contorul nu a ajuns la valoarea finala se executa instructiunile si se incrementeaza contorul

Functii pentru siruri de caractere

a) strlen – returnează lungimea unui sir de caractere

Exemplu:

```
<?php
$a=”test”;
echo strlen($a);
?>
```

Afisare:
4

b) strstr – caută un subsir intr-un sir

Exemplu:

```
…….……..……..……..…..
if( strstr( $sir, “PHP” ) = = true )
echo “gasit”;
else
echo “nu am gasit”;
```

c) stristr – identică cu strstr doar că nu tine cont de litere mari sau mici

d) **substr – afişează un subşir într-un şir dat**

Exemplu:

```
<?php
$s="invat php";
echo substr( $s, 1, 2);
?>
```

Afisare:
nv

d) **str_replace – inlocuieste un subsir cu un sir dat**

Exemplu:

```
<?php
$s="invat php";
echo str_replace( "invat", "stiu", $s);
?>
```

Afisare:
stiu php

e) **strtoupper – schimbă tipul literelor (din mici in mari)**

f) **strtolower - schimbă tipul literelor (din mari in mici)**

g) **trim** – **şterge spaţiile dintr-un şir**

h) **nl2br** – **transforma caracterele 'enter' in
**

Exemplu:
```
<?php
echo nl2br( "text afisat pe
3
linii" );
?>
```

Afisare:
text afisat pe
3
linii

e) **explode** – **extrage subşiruri dintr-un sir, fiecare subşir extras punându-l într-un array(matrice)**

Forma generală:
array **explode** (string *$delimiter* , string *$string* [, int *$limit*]

$delimiter – caracterul după care se face delimitarea
Exemplu:

```
<?php
$propozitie = "Exemplu de folosire a functiei explode";
$cuvinte = explode(" ", $propozitie);
echo $cuvinte[0]  ." ". $cuvinte[5];
?>
```

Afişare:
Exemplu explode

e) **implode** – **pune elementele dintr-un array intr-o variabilă**

Forma generală:
string **implode** (string *$delimiter* , array *$components*)

Exemplu:
```
<?php
$array = array('lastname', 'email', 'phone');
$a= implode(",", $array);
echo $a;
?>
```

Afişare:
lastname,email,phone

f) **header** - este functia prin care utilizatorul este redirectionat de pe pagina de procesare, pe pagina initiala (adresa poate fi orice pagina.
Exemplu:

header('location: http://agricolfalticeni.ro');

Funcţii matematice

a) abs(x) - Returneaza valoarea absoluta a lui 'x'
Exemplu:
echo abs(-5); // va returna 5
b) ceil(x) - Returneaza valoarea 'x', rotunjita la întregul imediat superior
Exemplu:
echo ceil(5.3);// va returna 6
c) floor(x) - Returneaza valoarea 'x', rotunjita la întregul imediat inferior
Exemplu:
echo floor(5.6);// va returna 5
d) max(x,y,...) - Returneaza valoarea maxima a unui set de valori
Exemplu:
e*cho max(3,5,7,3,8,4,2);// va returna 8*
e) min(x,y,...) - Returneaza valoarea minima a unui set de valori
f) pow(x,n) - Returneaza numarul 'x', ridicat la puterea specificata 'n'
g) strftime(f) - Return**eaza data curenta, formatata conform continutului parametrului 'f'**
Exemplu:
echo strftime("%A");//va returna Monday
h) sqrt(x) - Returneaza radacina patrata a lui 'x'
i) rand([int_min],int_max) este functia din PHP care genereaza automat numere aleatoare.

Exemplu:
echo rand(2,9);// va afişa un număr aleatoriu

Funcţii data/oră

a) **time() - Întoarce timpul curent măsurat în numărul de secunde de la 1 ianuarie 1970 00:00:00 GMT).**

Exemplu:

echo time(); // returneaza ceva de genul 1268668902

b) **date() – returneaza data serverului**

```
$a = date("j, n, Y");  // 10, 3, 2001

echo $a;
```

FORMULARE

Formularele reprezintă un element principal al site-urilor WEB. Sunt puţine site-uri care nu folosesc formulare.

Folosim formulare pentru a transmite si recepţiona date. Ele reprezintă elementul de interactivitate cu pagina web.

Pentru a folosi un vom folosi tag-urile *<FORM>* şi *</FORM>*. Între aceste tag-uri vor fi introduse toate elementele formularului.

Tag-ul *<FORM>* are doua atribute importante: *ACTION* si *METHOD*.

Prin *ACTION* transmitem browser-ului ce se va înâampla cu datele introduse in formular. Valoarea atributului ACTION poate fi o adresa URL sau poate fi o adresa de email, caz în care datele din formular vor fi trimise prin email la adresa respectiva.

Prin *METHOD*, se precizează metoda de trimitere a datelor. Valoarea implicita a acestui atribut este GET, cu ajutorul acestei metode putand fi trimise cantitati mici de date, cea mai folosita metoda fiind POST.

Elementele unui formular

Elementele unui formular vor fi introduse cu ajutorul tag-ului <INPUT>.
Atributele cele mai importante ale acestui tag sunt urmatoarele:
- TYPE - tipul elementului
- NAME - numele elementului
- VALUE - valoarea elementului

Elementele ale unui formular pot fi:
- campurile de editare
- butoanele radio
- casetele de validare
- casetele pentru upload-ul de fisiere
- listele de selectie
- butoanele submit si reset

Preluarea datelor din formulare

1.

Utilizarea casetelor text pentru introducerea numerelor
PB*1: Se introduc doua numere si se afiseaza suma dintre ele*

Introduceti primul numar: 2

Introduceti al doilea numar 3

Afiseaza_suma | Sterge_numerele

Cod scris în fisierul formular.html	Cod scris în fisierul formular.php
Utilizarea casetelor text pentru introducerea textului PB2: Se introduce un text si se afiseaza textul introdus <form action="f2.php" method="get"> <p> Introduceti textul: <input name="text" type="text"> <input value="Afiseaza_textul"	<html> <Body> <?php $a=$_REQUEST[unu]; $b=$_REQUEST[doi];

type=”SUBMIT”> <input value=”Sterge_textul” type=”RESET”> 	echo “Suma este”, $a+$b; ?> </Body> </html>

2.

Utilizarea listelor de selectie

PB*3: Se alege o culoare dintr-o lista si se afiseaza culoarea aleasa*

Cod scris în fisierul formular.html	Cod scris în fisierul formular.php
Utilizarea listelor de selectie PB3: Se alege o culoare dintr-o lista si se afiseaza culoarea aleasa <FORM ACTION=f3.php METHOD=get> <P> <B>Alegeti o culoare:</B> <select name=text> <option value=1>alb <optionvalue=2>negru <optionvalue=3>verde <optionvalue=4>rosu <optionvalue=5>maro </select> <INPUT TYPE=SUBMIT VALUE=Afiseaza_culoarea> <INPUT TYPE=RESET VALUE=Sterge_textul> </FORM>	<html> <Body> <?php $a=$_REQUEST[text]; echo "Ati ales culoarea ", $a; ?> </Body> </html>

3.

Utilizarea butoanelor radio

PB*4: Ce virsta aveti?*

Cod scris în fisierul formular.html	Cod scris în fisierul formular.php
Utilizarea butoanelor radio PB4: Ce virsta aveti? <FORM ACTION=f4.php METHOD=get> <INPUT TYPE=radio NAME=virsta	<html> <Body> <?php

value="copil">SUB 18 ANI value="tinar">INTRE 18-50 value="batrin">PESTE 50 </FORM>	$a=$_REQUEST[virsta]; echo "Sunteti ", $a; ?> </Body> </html>

4.

Utilizarea butoanelor de validate

PB5: *Alegeti produsele dorite*

Cod scris în fisierul formular.html	Cod scris în fisierul formular.php
Utilizarea butoanelor de validate PB5: Alegeti produsele dorite <form action=f5.php method=GET> <input type="checkbox" name="produs1" value="mere" />mere <input type="checkbox" name="produs2" value="pere" />pere <input type="checkbox" name="produs3" value="gutui" />gutui <INPUT TYPE=SUBMIT VALUE=Afiseaza> <INPUT TYPE=RESET VALUE=Reset> </form>	<html> <Body> <?php $a=$_REQUEST[produs1]; $b=$_REQUEST[produs2]; $c=$_REQUEST[produs3]; echo "Produsul selectat este: ", $a,",",$b,",",$c; ?> </Body> </html>

5.

Introducerea comentariilor

PB6: *Introduceti un comentariu*

Cod scris în fisierul formular.html	Cod scris în fisierul formular.php
Introducerea comentariilor PB6: Introduceti un comentariu </p> <FORM ACTION=f6.php METHOD=get> <p> <TEXTAREA NAME=comentariu ROWS=5 COLS=60> </TEXTAREA> <INPUT TYPE=SUBMIT VALUE=Trimite> <INPUT TYPE=RESET VALUE=Sterge_comentariu> </p> <p>SURSA PROGRAMULUI PHP CARE PREIA DATELE DIN ACEST FORMULAR(PB6) </p> </FORM>	<html> <Body> <?php $a=$_REQUEST[comentariu]; echo "Comentariul introdus este: ", $a; ?> </Body> </html>

Module COOKIE

Modulele cookie conţin text pe care il putem stoca pe calculatorul utilizatorului şi să-l citim ulterior.

Spre deosebire de variabilele obişnuite care îşi pierd valoarea în momentul închiderii unui program php, variabilele cookie îşi păstrează valoarea un timp indefinit sau definit de cel care le creează. Pentru a se putea păstra, browserul utilizatorului stochează variabilele cookie în unitatea de hard-disc a utilizatorului.

Modulele cookie sunt transmise atât de pe server către browser cât şi de la browser către server (astfel prin instalarea unui modul cookie înţelegem transmiterea unui modul către utilizator iar prin citirea unui modul cokie înţelegem transmiterea unui modul cookie către server).

1. Instalarea unei modul cookie

Pentru instalare se foloseşte funcţia setcookie cu următoarea sintaxă:

setcookie(numeCookie, valoare, expirare),

unde:
- numeCookie - specifica numele variabilei cookie acre se transmite;
- valoare - specifică valoarea variabilei care se transmite;
- expirare - indică momentul expirarii variabilei cookie (după ora specificata, variabila cookie nu mai este accesibilă).

Observaţii:

- O variabilă cookie se creează înaintea oricărei alte informaţii;
- În general, pentru specificarea momentului expirării se foloseşte funcţia **time()**, care returnează intervalul de timp (exprimat în secunde) scurs de la 1 ianuarie 1970;
- Folosind funcţia setcookie se pot transmite mai multe cookie-uri succesiv (maximum 20 numarul cookie-urilor ce pot fi trimise aceluiasi utilizator);
- argumentul expirare poate lipsi.

Exemplu:

Fişierul *inst_cookie.php:*

```
<?php
setcookie('functie', "inginer", time()+7200);
?>
```

Variabila cookie va fi disponibilă 2 ore (7200 secunde) de la crearea sa

2. Citirea unui modul cookie

După ce varianila cookie s-a instalat, aceasta este accesibilă la următoarea încărcare a paginii prin matricea $_COOKIE.

Valorile variabilelor cookie sunt încarcate automat în matricea $_COOKIE. Astfel pentru citire este suficient să verificăm conţinutul variabilei $_COOKIE[numeCookie].

Exemplu:

```
<?php
if(isset($_COOKIE['functie']))
{
echo $_COOKIE['functie'];
}
?>
```

3. Ştergerea unui modul cookie

Un modul cookie va fi şters automat dacă i se schimbă timpul de expirare. De exemplu:

```
setcookie('functie', " ", time() -7200);
```

SESIUNI

Cu ajutorul sesiunilor PHP retine informaţii de la o pagina la alta.

După crearea unei sesiuni informaţia se păstrează pana la închiderea browser-ului, sau până cand utilizatorul distruge sesiunea curenta.

Sesiunile au următoarele avantaje:
- permit stocarea unui volum mai mare de informaţii comparativ cu cookie-urile
- pot fi folosite chiar daca browserul utilizatorului nu suporta cookie-uri sau daca acestea sunt dezactivate.

Initializarea unei sesiuni se face cu funcţia **session_start()** care trebuie să fie printre primele linii de cod dintr-un script PHP, deoarece apelul acestei ei trebuie făcut înainte de trimitea către browser-ul Web a unui cod HTML.

Exemplul 1: Parolarea unei pagini php utilizând sesiuni

Fisierul de start index.php conţine codul:

```
<?php
include "sesiune.php";
?>
……restul codului fişierul index.php
```

Fişerul sesiune.php conţine codul:

```
<?php
	session_start();
	if(!isset($_SESSION['SESS_LOG']))
	{
		header("location: logare.php");
		exit();
	}
?>
```

Fisierul parola.php conţine codul:

```
<?php
	session_start();
	$login =$_POST['login'];
	if($login=="parola")
{
	session_regenerate_id();
	$_SESSION['SESS_LOG'] = "1";
	session_write_close();
	header("location: index.php");
	exit();
}
else
{
	header("location: logare.php");
	exit();
}
?>
```

Fişierul logare.php conţine codul:

```
<form action="parola.php" method="post">
	<b>Parola :</b>
	<input type="password" name="login" />
	<input type="submit" value="Login" id="loginbutton" name="Submit"/>
</form>
```

Fişierul logout.php conţine codul:

```
<?php
	session_start();
	unset($_SESSION['SESS_LOG']);
?>
```

Exemplul 2: Contor de vizitare – afiseaza de câte ori s-a vizitat o pagină

```
<?php
session_start();
if(!isset($_SESSION['count']))
{
	$_SESSION['count']=1;
}
else
{
	$_SESSION['count']++;
}
echo $_SESSION['count'];

?>
```

Utilizarea protocolului FTP

Protocolul FTP este util pentru realizarea transferurilor de fisiere la distanta. Pentru a putea fi utilizat din PHP va trebui activate optiunea FTP la instalarea serverului PHP.

A. Conectarea la serverul FTP

Pentru conectare se foloseşte funcţia:

ftp_connect(gazda, [port, [timeout]])

unde:
- gazda - este adresa ftp la care ne conectăm
- port – portul pe care se face conectarea (este opţional. Portul implicit este 21)
- timeout – durata maxima căt se aşteaptă până la conectare (opţional. Implicit 90 secunde)

După conectare va trebui să ne identificăm (logăm). Acest lucru îl facem utilizând funcţia:

ftp_login(flux_ftp, nume_utilizator, parola)

Exemplu:

```
<?php
$connect=ftp_connect("ftp.proba.ro");
$result=ftp_login($connect,"user","parola");
if(!$result)
{
	echo "Nu ma pot conecta la serverul ftp";
}
else
	echo "M-am conectat la serverul ftp";
```

```
?>
```

B. Afişarea listei de fişiere dintr-un director

Pentru afişare vom folosi funcţia:

ftp_nlist(flux_ftp, nume_director)

Această funcţie returnează matricea de nume de fişiere din directorul specificat

Exemplu:

```
<?php
$connect=ftp_connect("ftp.proba.ro");
$result=ftp_login($connect,"user","parola");
if(!$result)
{
        echo "Nu ma pot conecta la serverul ftp";
}
else
        echo "M-am conectat la serverul ftp";

$a=ftp_nlist($connect,baza);
foreach($a as $value)
{
        echo $value, "<br>";
}
?>
```

C. Descărcarea unui fişier

Pentru descărcare vom folosi funcţia:

ftp_get(flux_ftp, nume_fişier_local,nume_fişier_la_distanţă,mod)

Fişierul de pe server (fişier_la_distanţă) este salvat local nu numele fisier_local într-un mod specificat: FTP_ASCII sau FTP_BINARY.

Exemplu:

```
<?php

$connect=ftp_connect("ftp.proba.ro");
$result=ftp_login($connect,"user","parola");
if(!$result)
{
        echo "Nu ma pot conecta la serverul ftp";
}
else
        echo "M-am conectat la serverul ftp";
```

```
$result=ftp_get($connect,"fisier.doc","fis1.doc",FTP_ASCII);
ftp_close($connect);

?>
```

D. Încărcarea unui fişier pe server

Se face utilizând funcţia

ftp_put(flux_ftp, nume_fişier_local,nume_fişier_la_distanţă,mod)

LUCRUL CU FIŞIERE IN PHP

Folosind php-ul se pot executa anumite operaţii asupra fişierelor aflate pe server cum ar fi:

fopen() = deschide fisierul indicat
fclose() = inchide fisierul
fread() = citeste continutul fisierului
fwrite() = scrie in fisier
filesize() = indica dimensiunea fisierului

Sintaxa funcţiei fopen() este:
fopen(param1, param2);
param1 = fişierul, calea către fişier sau adresa fişierului care va fi deschis
param2 = modul in care va fi deschis fişierul, si poate avea valorile:

```
Exemplu: fopen('fisier.html',"w")
```

r = fişier deschis doar pentru citire
r+ = fişier deschis doar pentru citire si scriere
w = fişier deschis doar pentru scriere
w+ = fişier deschis pentru citire si scriere iar daca nu exista fişierul îl creează
a = fişier deschis pentru adăugare la sfârşit
a+ = fişier deschis adăugare la sfârşit iar daca nu exista fişierul îl creează
t = fişier deschis in mod text
b = fişier deschis in mod binar
sau combinaţii ale acestora.

Baze de date - noţiuni introductive

Bazele de date sunt colecţii de date, aranjate într-o anumită formă, asupra cărora se pot face diferite operaţii ca:

- Crearea bazei de date;
- Conectarea la baza de date;
- Inserarea datelor in baza de date;
- Stergerea datelor din baza de date;

- Adaugarea sau modificarea datelor;

În general, o bază de date este alcătuită din una sau mai multe tabele, între acestea putându-se stabili diferite relaţii. Acest lucru ofera bazei de date proprietatea de bază de date relaţională.

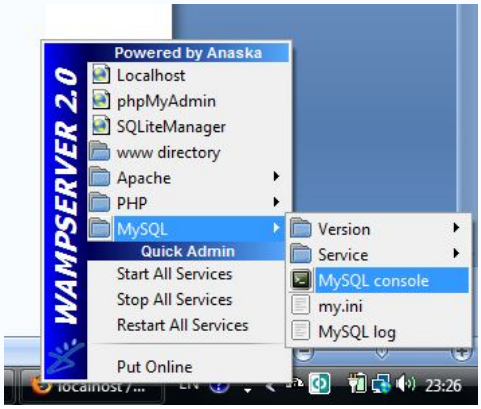
Un tabel este alcătuit din coloane (numite câmpuri) şi rânduri (numite înregistrări).

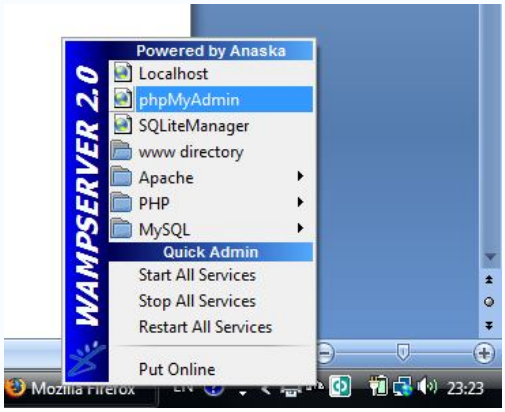
Pentru implementarea unei baze de date este nevoie de un sistem de gestiune a bazelor de date(SGBD). Exemple de astfel de SGBD-uri:

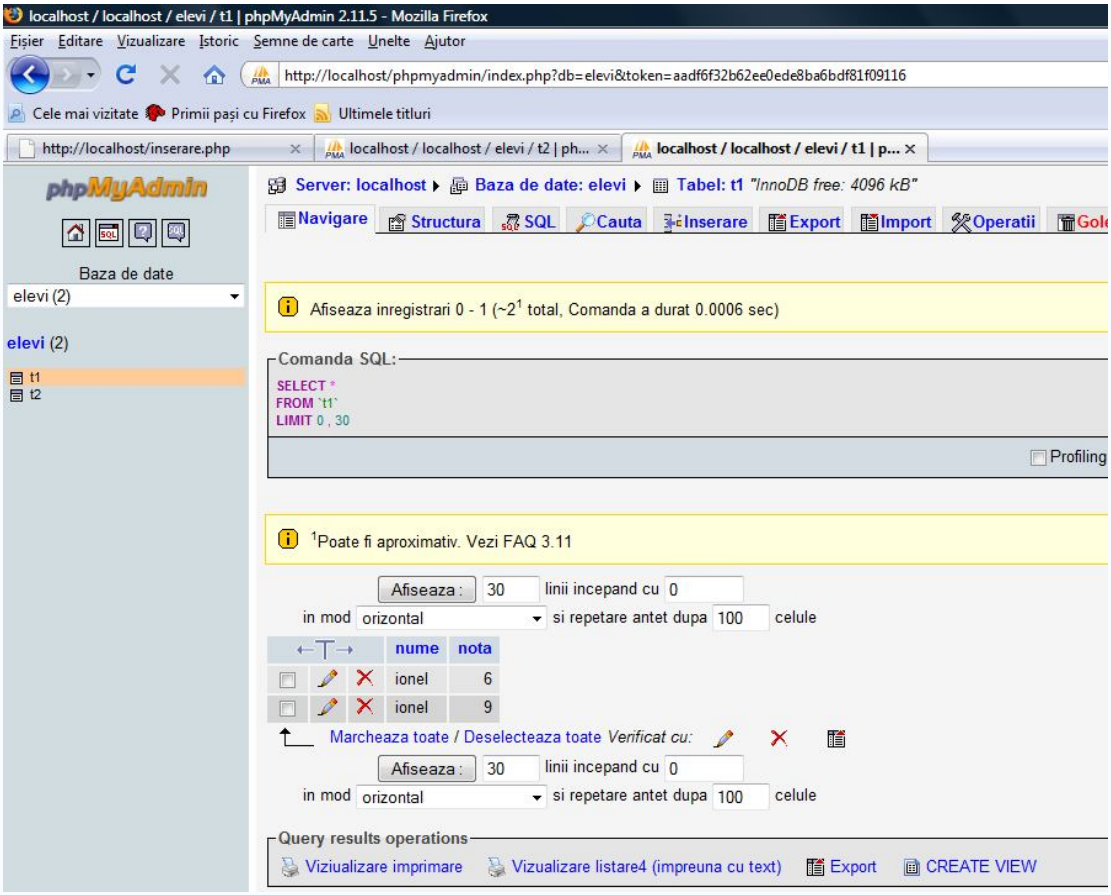
- Microsoft Access;
- Visual Foxpro;
- MySQL;
- Oracle;
- etc.

Vom studia SGBD MySQL. Deşi este folosit foarte des împreună cu limbajul de programare PHP, cu MySQL se pot construi aplicaţii în orice limbaj major. Există multe scheme API disponibile pentru MySQL ce permit scrierea aplicaţiilor în numeroase limbaje de programare pentru accesarea bazelor de date MySQL, cum are fi: C, C++, C#, Borland Delphi, Java, Perl, PHP, Python, FreeBasic, etc., fiecare dintre acestea folosind un tip spefic API. O interfaţă de tip ODBC denumită MyODBC permite altor limbaje de programare ce folosesc această interfaţă, să interacţioneze cu bazele de date MySQL cum ar fi ASP sau Visual Basic. În sprijinul acestor limbaje de programare, unele companii produc componente de tip COM/COM+ sau .NET (pentru Windows) prin intermediul cărora respetivele limbaje să poată folosi acest SGBD mult mai uşor decât prin intermediul sistemului ODBC. Aceste componente pot fi gratuite (ca de exemplu MyVBQL) sau comerciale (http://ro.wikipedia.org/wiki/MySQL).

Administrarea sistemului MySQL se poate face fie din linia de comandă fie folosind aplicaţia PHPMyAdmin.







Cel mai important mod de administrare a unei baze de date în MySQL este însă cel dat de PHP, folosind formulare.

Comenzile uzuale folosite în general atât în MySQL cât si în celelalte SGBD-uri sunt:

- CREATE – crează o bază de date şi/sau un tabel
- DROP - şterge o bază de date şi/sau un tabel
- INSERT – adaugă înregistrări(linii) într-un tabel
- DELETE - şterge înregistrări(linii) într-un tabel
- UPDATE – modifică înregistrările dintr-un tabel
- SELECT – selectează înregistrările dintr-un tabel
- ALTER - modifică proiectul unui tabel după ce acesta a fost creat cu instrucţiunea CREATE TABLE

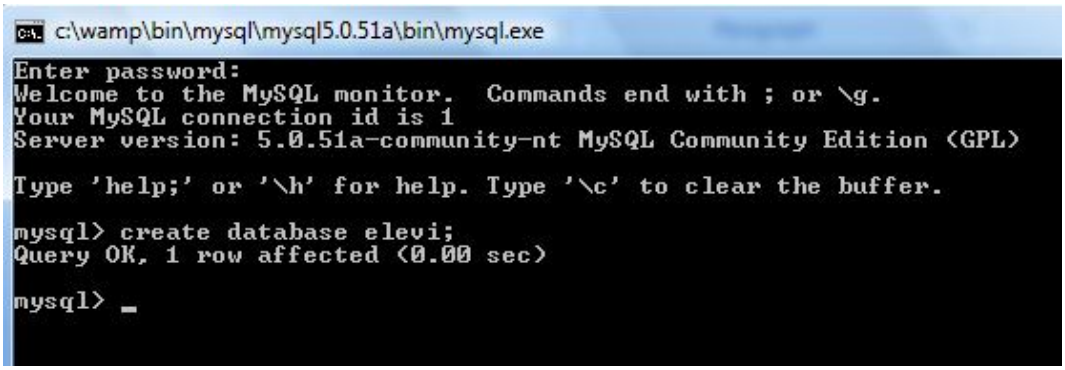
Tipuri de date folosite în MySQL(o parte din ele):

- Int – număr întreg
- Char – secţiune cu lungime fixă de max. 255 caracter
- Varchar – secţiune variabilă de max 255 caractere
- Float – număr real mic
- Double – număr real mare
- Text – şi de maxim 65535 caractere
- Date – data im format an-luna-zi
- Time – ora in format oră-minut-secundă

Crearea unei baze de date în MySQL

Comanda folosită pentru creare unei baze de date, scrisă în linia de comandă a MySQL este:
CREATE DATABASE numele_bazei;
Expl: create database elevi;

Obs: orice comanda scrisă în linia de comandă este terminată de ;



În continuare va trebui selectată baza de date (există posibilitatea să avem mai multe baze de date). Acest lucru se face cu comanda USE numele_bazei_de_date;

Expl: use elevi;

O bază de date poate conţine una sau mai multe tabele. Pentru crearea unei tabele vom folosi comanda:

CREATE TABLE nume tabela (nume_camp1 tip_camp1, nume_camp2 tip_camp2,…….nume_campn tip_camp_n);
Expl:
Create table t1 (nume text, nota int);

Vizualizarea bazelor de date se face cu comanda SHOW DATABASES;

Alte comenzi:

SHOW TABLES; - afiseaza tabelele existente in baza curenta
SHOW COLUMNS FROM nume_tabela; - afiseaza informatii despre coloanele unui table
DROP TABLE nume tabel; - sterge tabelul numit 'tabel_unu'
DROP DATABASE numele_bazei; - sterge baza de date cu numele 'numele_bazei'

Pentru crearea unei tabele cu ajutorul scriptului PHP va trebui mai intâi sa ne conectăm la baza de date (vom folosi fisierul conexiune.php). Pentru exemplificare vom crea urmatorul tabel:

nume	nota
Ionescu	8
Popescu	7
Vasilescu	8
Georgescu	9

```
<?php
include “conexiune.php”
$table=”create table t1 (nume varchar(20), nota float)  type MyISAM”;
if(mysql_query($table))
{
 echo “Tabelul a fost creat”;
}
else
{
 echo “Tabelul nu a fost creat”;
}
mysql_close($conexiune);
?>
```

Observaţii:

Asupra coloanelor se pot aplica unele restrictii ca:
UNSIGNED-nu vor mai fi valori negative ci vor incepe de la 0.
AUTO_INCREMENT functioneaza cu orice tip intreg. La fiecare rand nou adaugat in baza de date numarul asociat va fi incrementat.
NULL- fara valoare (diferit de spatiu sau zero).
NOT NULL - orice inregistrare va fi considerata ceva.
Expl:

```
.....
$table="CREATE TABLE t1(id int(3) NOT NULL AUTO_INCREMENT, nume varchar(20) NOT NULL , nota float NOT NULL)
.....
```

Conectarea la o baza de date în MySQL

Pentru conectarea la o baza de date vom folosi urmatorul script (pentru a-l putea utiliza şi în alte aplicaţii il vom salva cu numele conexiune.php):

```
<?php
$host="localhost"; //host-ul
$user="root"; //userul
$password="pass"; //parola
$database="t1"; //baza de date
$conexiune=mysql_connect($host,$user,$password)
or die ("Nu ma pot conecta la baza de date");

$bazadate=mysql_select_db($database,$conexiune)
or die ("Nu gasesc baza de date");
?>
```

înlocuind bineînţeles hostul, user-ul şi parola dacă este cazul

- mysql_connect() - este functia prin care ne conectam la serverul MySQL

- $conexiune - va avea o valoarea TRUE sau FALSE functie de rezultatul conectarii la serverul MySQL folosind functia mysql_connect()

- mysql_select_db - este functia care stabileste baza de date la care ne vom conecta

Puteti modifica variabilele din fisierul conexiune.php in functie de configurarile propriei baze de date:
$hostname=adresa serverului, de cele mai multe ori este localhost
$username=username-ul de conectare la baza de date
$password= parola de conectare la baza de date
$database=numele bazei de date

mysql_close($conexiune); - inchide baza de date

Inserarea datelor într-o tabelă din MySQL

Pentru inserarea datelor într-o tabela din MySQL vom utiliza un formular scris în HTML şi un fişier PHP.

Instrucţiunea folosită pentru inserare este INSERT, cu următoarea sintaxă:

INSERT INTO nume_tabel (coloana1, coloana2,..., coloanan) values ('valoare1','valoare2',...,'valoaren');

Exemplificând:

INSERT INTO nume_tabel (camp1, camp2, camp3) VALUES (valoarea1, valoarea2, valoarea3);

Această instrucţiune va introduce in tabelul cu numele 'tabel', in 'camp1' 'valoarea1', in 'camp2' 'valoarea2' si in 'camp3' 'valoarea3'. Tabelul arată astfel:

camp1	**camp2**	**camp3**
valoarea1	valoarea2	valoarea3

Observaţie: Se poate omite una din coloane la inserare.

INSERT INTO nume_tabel (camp1, camp2) VALUES (valoarea1, valoarea2);

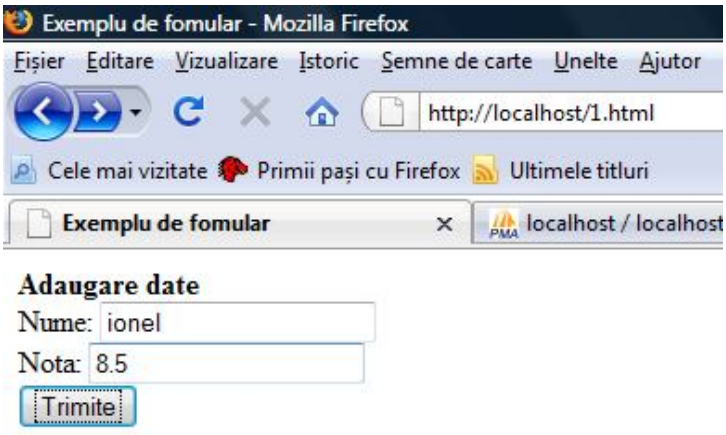
sau:
INSERT INTO tabel VALUES (valoarea1, valoarea2, valoarea3);

daca introducem valori in toate campurile tabelului

Exemplu de introducere a datelor in tabela t1 din baza de date elevi folosind un formular (tabela t1 are câmpurile *nume* şi *nota*):

```
<html>
<head><title>Exemplu de fomular</title>
</head>
<body>
<b>Adaugare date</b>
<form method="POST" action="inserare.php">
Nume: <input type="text" name="nume"><br>
Nota: <input type="text" name="nota"><br>
<input type="submit" value="Trimite">
</form>
</body>
</html>
```

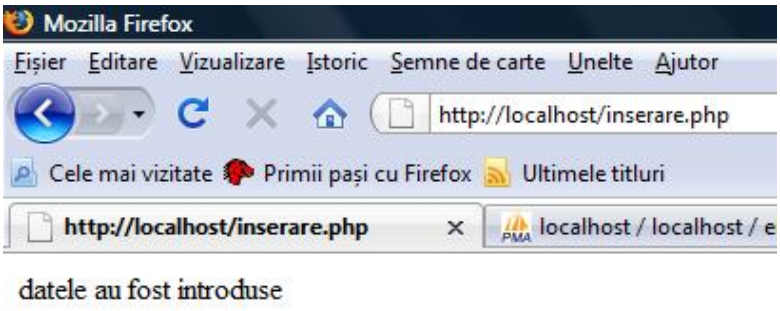
Rezultatul este:




```
/* fisierul inserare.php */

<?php
include "conexiune.php";

$id=$_POST['id'];
$nume=$_POST['nume'];
$nota=$_POST['nota'];

$query="INSERT INTO t1 (nume, nota) VALUES ('$nume','$nota')";
if (!mysql_query($query)) {
die(mysql_error());
} else {
echo "datele au fost introduse";
}
```

```
mysql_close($conexiune);
?>
```

Rezultatul este:



Afisarea datelor dintr-o tabelă din MySQL

Afişarea datelor dintr-o tabelă se face utilizând comanda SELECT (în linia de comandă) cu una din următoarele structuri, în funcţie de ceea ce ne-am propus:

SELECT * FROM tabel; - afişează tot ce există în tabela tabel

SELECT camp1 FROM tabel; - afişează conţinutul câmpului 'campul1' din tabelul 'tabel'

SELECT camp1, camp2, ….., campn FROM tabel –afişează conţinutul mai multor câmpuri din tabelul 'tabel'

SELECT * FROM tabel WHERE camp1 = 'val1'; - afişează toate câmpurile cu valoarea 'val1'

SELECT camp1, camp2 FROM tabel WHERE camp2 LIKE 'val2'; - afişează toate înregistrările în care 'camp2' este asemănător cu 'val2'

SELECT camp1, camp2 FROM tabel WHERE camp2 LIKE 'val2%'; - afişează toate înregistrările în care 'camp2' începe cu 'val2'

SELECT camp1, camp2 FROM tabel WHERE camp2 LIKE '%val2'; - afişează toate înregistrările în care 'camp2' se termină cu 'val2'

SELECT * FROM tabel WHERE camp1=val1 AND camp2 LIKE '%val2%'; - afişează toate câmpurile care conţin 'val1' şi se asemănă cu 'val2'

SELECT camp1, camp2,….campn FROM tabel WHERE camp1 != val3; - afişează toate câmpurile care diferă de 'val3'

SELECT camp1 FROM tabel ORDER BY camp1 ASC; - afişează conţinutul câmpului 'camp1' în ordine crescătoare

SELECT camp1, camp2 FROM tabel ORDER BY camp1 ASC, camp2 DESC; - afişează conţinutul camp1 în ordine crescătoare şi camp2 în ordine descrescătoare.

SELECT count(*) FROM tabel; - afişează câte înregistrări sunt în tabel

SELECT camp1 FROM tabel GROUP BY camp1 ORDER BY camp1 ASC; - afişează conţinutul camp1 grupat după 'camp1'

SELECT * FROM tabel LIMIT 0,3; - afişează din tabel începând de la prima înregistrare încă 3.

Observaţie: Comanda SELECT este una din cele mai folosite comenzi în MySQL

Pentru afisarea datelor dintr-o tabela din MySQL utilizând PHP vom utiliza sintaxa:

```
$afisare=mysql_query("SELECT * FROM nume_tabel");
```

Pentru afisarea fiecărui rand din tabel vom folosi o bucla *while* si comanda **mysql_fetch_row**.

Exemplu de afişare a datelor dintr-o tabela (vom lua drept exemplu tabela t1 pe care am folosit-o şi în capitolul anterior):

```
<?php
include "conexiune.php";
$afisare=mysql_query("SELECT * FROM t1");
echo "<table border=2>";
echo "<tr><td>Nume</td><td>Nota</td></tr>";
while ($row=mysql_fetch_row($afisare)) {
echo "<tr><<td>$row[1]</td><td>$row[2]</td></tr>";
}
echo "</table>";

mysql_close($conexiune);
?>
```

Observaţie1:

Comanda SELECT permite şi anumite restricţii date prin clauza WHERE. Exemplu dacă dorim să afisăm toatele numele care au nota 9:

```
<?php
include "conexiune.php";
$sql=mysql_query("SELECT * FROM t1 WHERE nota=9");

echo "<table border=2>";
echo "<tr><td>ID</td><td>Nume</td><td>Nota</td></tr>";
while ($row=mysql_fetch_row($sql)) {
echo "<tr><td>$row[0]</td><td>$row[1]</td><td>$row[2]</td></tr>";
}
echo "</table>";

mysql_close($conexiune);
?>
```

Observaţia2: Funcţia **mysql_num_rows($afisare)** returnează numarul de linii continute de baza de date.

Modificarea datelor dintr-o tabelă din MySQL

Pentru modificarea conţinutului unei înregistrări fără a şterge acea înregistrare vom utiliza sintaxa:

UPDATE tabel SET coloana1='noua valoare a coloanei 1', coloana2='noua valoare a coloanei 2' …. coloanan=’noua valoare a coloanei n’ WHERE conditii;

Se vor modifica înregistrările care respectă condiţia dată în WHERE.
Exemplu: Fie tabela t1 care are urmatoarea structura:

nume	nota
ionel	9
ionela	5
Ionescu	7

Daca dorim sa modificam nota persoanei cu numele Ionescu vom scrie(în linia de comandă):

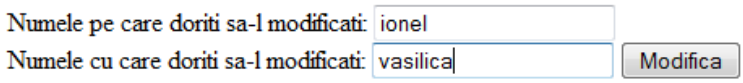
UPDATE t1 SET nota=10 where nume='Ionescu';

Rezultatul este:

nume	nota
ionel	9
ionela	5
Ionescu	10
ionel	10

Modificarea înregistrărilor folosind un formular şi un fişier php:

Expl.: În tabela t1 de mai sus, dorim să modificam numele unei personae folosind un formular sub forma:



Dacă modificarea se face vom afişa: *Modificare efectuată*, altfel *Modificare neefectuată* rezultatul fiind:

nume	nota
vasilica	9
ionela	5
Ionescu	10
vasilica	10

Cele două fişiere(html şi php) vor fi:
Update.html

```
<html>
<body>
<form action="update.php" method="post">
Numele pe care doriti sa-l modificati: <input type="text" name="numev"><br>
Numele cu care doriti sa-l modificati: <input type="text" name="numen">
<input type="Submit" value="Modifica">
</form>
</body>
</html>
```

Update.php

```
<?php
include "conexiune.php";
$numev=$_REQUEST['numev'];
$numen=$_REQUEST['numen'];
```

```
$query="UPDATE t1 SET nume='$numen' WHERE nume='$numev'";

$checkresult = mysql_query($query);

if ($checkresult) {
echo "Modificare efectuata";
} else {
echo "Modificare neefectuata";
}
mysql_close();
?>
```

Observaţie: Modificarea se va face în toate înregistrările tabelului

Stergerea înregistrărilor dintr-o tabelă din MySQL

Pentru a şterge anumite înregistrări dintr-o tabelă vom utiliza sintaxa:

DELETE FROM tabel WHERE conditii;

Se vor şterge înregistrările care respectă condiţia dată în WHERE.

Exemplu: Fie tabela t1 care are urmatoarea structura:

nume	nota
ionel	9
ionela	5
Ionescu	10

Daca dorim sa stergem persoana cu numele Ionescu vom scrie(în linia de comandă):

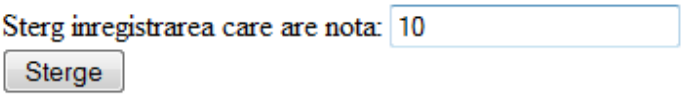
DELETE FROM t1 WHERE nume='Ionescu';

Rezultatul este:

nume	nota
ionel	9
ionela	5

Ştergerea înregistrărilor folosind un formular şi un fişier php:

Expl.: În tabela t1(iniţială) de mai sus, dorim să ştergem înregistrarea care are nota 10 folosind un formular sub forma:



Dacă modificarea se face vom afişa: *Ştergere efectuată*, altfel *Ştergere neefectuată* rezultatul fiind:

nume	nota
ionel	9
Ionela	5

Cele două fişiere(html şi php) vor fi:
delete.html

```
<html>
<body>
<form action="delete.php" method="post">
Sterg inregistrarea care are nota: <input type="text" name='nota'><br>
<input type="Submit" value="Sterge">
</form>
</body>
</html>
```

delete.php

```
<?php
include "conexiune.php";
$nota=$_REQUEST['nota'];

$query="DELETE FROM t1 WHERE nota='$nota'";

$checkresult = mysql_query($query);

if ($checkresult) {
echo "Stergere efectuata";
} else {
echo "Stergere neefectuata";
}
mysql_close();
?>
```

EXEMPLU DE APLICAȚIE IN CARE SE LUCREAZĂ CU BAZE DE DATE ȘI TABELE DIN MYSQL UTILIZÂND PHP

Cerințe:

1. Creare tabela
2. Inserare date
3. Afisare tabela
4. Modificare date
5. Stergere date

Vom utiliza urmatoarele fisiere:


update.php

afisare.html

afisare.php

conexiune.php

creare.html

creare.php

inserare.html

inserare.php

meniu.htm

sterge.html

sterge.php

update.html

1. conexiune.php

```
<?php
$host="localhost";
$user="root";
$database="ion";
```

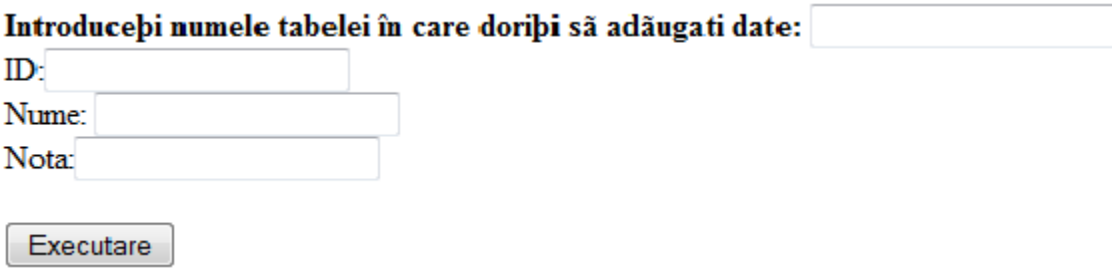
```
$conexiune=mysql_connect($host,$user)
or die("nu s-a contectat");
$bazadate=mysql_select_db($database,$conexiune)
or die("nu s-a conectat la baza de date");
?>
```

2. creare.html

creare.php

```
<html>
<body>
<?php
include "conexiune.php";
$a=$_POST['unu'];

$query="create table $a (Id int, Nume text, Nota int) ";
if(mysql_query($query))
{
echo "Tabelul a fost creat";
}
else
{
echo "Tabelul nu a fost creat ";
}
mysql_close();
?>
<br>
</body>
</html>
```

inserare.html

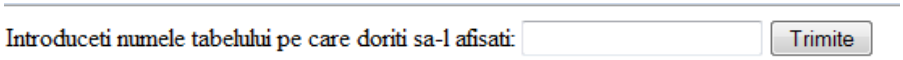


inserare.php

```
<html>
<body>
<?php
include "conexiune.php";
$a=$_POST['unu'];
$id=$_POST['id'];
$nume=$_POST['nume'];
$nota=$_POST['nota'];
$noi="insert into $a (id , Nume , Nota) VALUES ('$id', '$nume' , '$nota')" ;
if(!mysql_query($noi))
{
die(mysql_error());
}
else
{
echo "datele au fost introduse pt: ", $a;
}
```

```
mysql_close($conexiune);
?>
<br><br>
  </body>
</html>
```

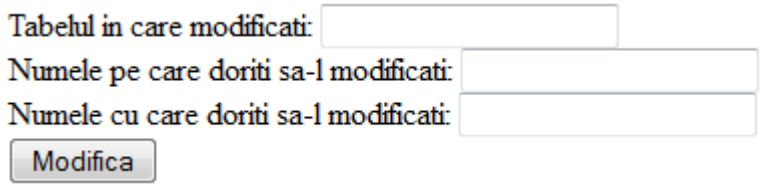
afisare.html



afisare.php

```
<?php
include "conexiune.php";

$a=$_POST['nume'];
$afisare=mysql_query("SELECT * FROM $a");
echo "<table border=3>";
echo "<tr><td>Id</td><td>Nume</td><td>Nota</td></tr>";
while($row=mysql_fetch_row($afisare))
{
echo "<tr>
        <td>$row[0]</td>
        <td>$row[1]</td>
        <td>$row[2]</td>
     </tr>";
}
echo"</table>";
mysql_close($conexiune);
?>
```

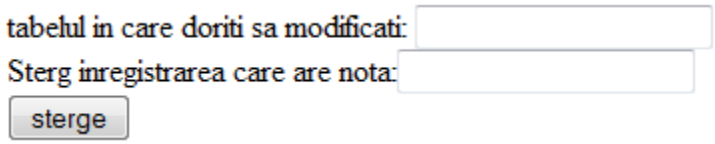
modificare.html



modificare.php

```
<?php
include "conexiune.php";
$a=$_POST[doi];
$nume1=$_POST[nume1];
$nume2=$_POST[nume2];
$query="update $a set nume='$nume2' where nume='$nume1'";
$checkresult=mysql_query($query);
if($checkresult)
{
echo "modificare efectuata";
}
else
{
echo "Modificare neefectuata";
}
mysql_close();
?>
```

stergere.html



stergere.php

```
<?php
include "conexiune.php";
$a=$_POST[unu];
$nota=$_POST['nota'];

$query="DELETE FROM $a WHERE nota='$nota'";
$checkresult=mysql_query($query);

if($checkresult)
{
echo "Stergere efectuata";
}else
{
echo "stergere neefecuata";
}
mysql_close();
?>
```