PHP/MySQL

PASUL 1 – Introducere

Introducere

Pentru multă lume, principala raţiune de a învăţa un limbaj precum PHP ar fi interacţiunea pe care o oferă cu bazele de date. In acest curs căutăm să explicăm cum se foloseşte PHP şi MySQL, pentru a memora informaţii pe Web şi cum includem toate astea pe site-ul Web. Pentru a parcurge acest curs sunt necesare cel puţin elementele de bază privind folosirea PHP. Se recomandă deci parcurgerea cursului de PHP, înainte de a continua.

De ce îmi trebuie o Bază de Date?

Este, de fapt, surprinzător cât de utilă poate fi o bază de date, mai ales atunci când poate fi folosită într-un site Web. Sunt foarte multe lucruri care se pot face într-un astfel de caz, de la afişarea unor simple liste şi până la producerea integrală a paginilor Web dintr-o bază de date. Câteva exemple cu PHP şi MySQL, folosite împreună, sunt:

- Schimbarea - rotaţia Banner-elor. Pentru a realiza schimbarea pe ecran a imaginilor la diferitele vizitări ale unei pagini şi a asigura astfel un interes mai mare al vizitatorilor, se poate folosi un script PHP care deschide o bază de date şi extrage aleator o imagine sau un set de imagini, pentru a-l include într-un set de pagini Web. Scriptul PHP va contoriza deasemeni numărul de câte ori a fost văzut banner-ul, putând, cu modificări minore, să urmăreasc chiar şi numărul de *click*-uri. Pentru a adăuga, schimba sau edita bannerele, tot ce avem de făcut este să schimbăm baza de date iar scriptul va încărca banner-ul corect în toate paginile site-ului Web.
- Forumurile Web. Sute de forumuri de dezbatere sunt acum realizate - rulează în PHP şi MySQL. Acestea s-au dovedit mult mai eficiente dacât alte sisteme, creind câte o pagină pentru fiecare mesaj şi oferind o largă varietate de opţiuni. Toate paginile forumului pot fi actualizate prin schimbarea unui script PHP.
- Bazele de Date. Un astfel de exemplu (edificator) îl oferă site-urile care dau toate informaţiile dintr-o bază de date. Spre exemplu, Script Avenue este rulat de câteva scripturi, care furnizează toate informaţiile dintr-o bază de date amplă. Toate categoriile de scripturi diferite pot fi accesate într-un singur script exact prin schimbarea URL-ului pentru a accesa diferite părţi ale bazei de date.
- Site-urile Web. Când avem un site de mari dimensiuni - cu multe pagini, şi dorim să modificăm aspectul general sau să schimbăm proiectul, chestiune care ar cere foarte mult timp pentru aducere la zi şi încărcare. Cu PHP şi MySQL întregul site Web se poate reduce la unul sau două scripturi PHP care să acceseze o bază de date MySQL pentru a obţine informaţiile pentru paginile Web. Pentru a actualiza sau a modifica proiectul centrului Web nu trebuie să schimbăm decât o pagină.

De ce e nevoie?

Sunt necesare trei componente pentru a putea rula scripturi PHP care să acceseze baze de date cu MySQL.

1. În primul rând, este nevoie de un server Web. Acesta poate fi sau pe calculatorul personal sau pe o 'gazdă' Web. Oricare pachet - oricare tip de server Web este acceptat şi va lucra cu PHP şi MySQL, dar cel mai recomandat este Apache, care e public şi gratis.
2. PHP trebuie să fie instalat pe server. Dacă nu este deja instalat, puteți să o faceți sau să apelați la administratorul serverului Web, în acest sens. Pachetul de instalare poate fi descărcat - preluat de la http://PHP.net şi este deasemeni public - gratuit. Dacă nu sunteți sigur dacă este sau nu instalat, o să vedeți mai jos cum vă puteți lămuri.
3. În fine, aveți nevoie de MySQL. Acesta este de fapt pachetul de programe pentru gestiunea bazelor de date. Puteți folosi deasemeni multe alte sisteme de baze de date (SQL, Oracle etc.) dar aici discutăm despre PHP/MySQL, deci avem nevoie de MySQL (cu toate că se folosesc comenzi care țin de limbajul SQL ți 'merg' la oricare din sistemele de baze de date SQL). Şi pachetul MySQL este public - gratuit, putând fi preluat din pagina oficială a firmei MySQL. Dar şi în cazul MySQL trebuie să verificăm mai întâi dacă nu cumva este deja instalat.

Dacă nu puteți instala PHP şi MySQL, sau dacă gazda Web nu v-o permite, puteți apela la alt server Web, la altă gazdă. Spre exemplu, Freedom2Surf este un server Web public, care acordă acces liber şi suportă PHP, având MySQL instalat. HostRocket este un alt server excelent care oferă spațiu de până la 300 MO, şi include PHP, MySQL şi multe altele, dar percepe o taxă de abonament de 10 USD pe lună.

Cum aflăm dacă PHP şi MySQL sunt instalate

Există un test simplu, atât pentru PHP cât şi pentru MySQL:

Lansați - deschideți un editor de texte şi scrieți următoarele:

```
<?
phpinfo();
?>
```

salvând fişierul cu un nume precum *phpinfo.php*

Apoi încărcați acest fişier în spațiul web, pe server, apelându-l, după aceea, din navigator. Dacă PHP-ul este deja instalat veți căpăta o pagină plină cu toate detaliile instalării respective. Parcurgeți atunci aceste informații. Dacă găsiți o secțiune despre MySQL atunci înseamnă că MySQL este şi el deja instalat.

Gestiunea Bazelor de date

Chiar dacă tot ce ține de administrarea bazelor de date poate fi realizat prin scripturi PHP, instalarea pe server a pachetului *PHPMyAdmin* poate aduce un plus de confort. Este un excelent set de scripturi care asigură o interfață administrativă pentru bazele de date MySQL. Si mai este şi public-gratuit.

Cu el puteți adăuga, elimina, edita, salva şi vizualiza bazele de date, fiind deosebit de util pentru depanare.

Scopul acestui curs

Urmărim familiarizarea cursantului cu câteva din elementele de bază ale folosirii PHP împreună cu MySQL. Pentru asta vom urmări mereu un exemplu. Parcurgând cursul de față veți învăța cum să creați un program bazat pe Web pentru a contacta colaboratorii - o variantă de caiet de adrese. Acest program vă permite să memorați numele împreună cu adresa corespunzătoare, adresa e-mail şi numărul de telefon. Veți putea să actualizați înregistrările, să căutați în baza de date. Existând Şi opțiunea de a trimite un mesaj email la toți cei din baza de date (rețineți însă: acest sistem nu va fi folosit pentru expedierea unor mesaje nesolicitate).

După realizarea acestui sistem veți fi, cu siguranță, în măsură să creați aproape orice tip de bază de date pe Web.

PASUL 2 – Construirea unei baze de date

Introducere

Mai înainte de a porni construcția unor script-uri pentru baze de date, ar trebui să avem o bază de date în care să plasăm informații şi din care să citim. In această lecție vom vedea cum creem o bază de date în MySQL şi cum o pregătim pentru date. Ca un exemplu, vom crea o bază de date de tip carte de adrese de contact.

Construcția Bazei de Date

Bazele de date MySQL au un sistem de inițializare standard (un *setup*). Este vorba de o bază de date, compusă din tabele, oarecum separate, conținând diferite câmpuri, etc. chiar dacă sunt parte dintr-o bază de date. Fiecare tabelă conține înregistrări care sunt făcute din câmpuri.

Conectarea la o Bază de Date

Procesul de instalare a unei baze de date MySQL diferă de la o platformă la alta. In esență este vorba peste tot de un nume al bazei de date, un nume de utilizator (cont) şi o parolă. Aceste informații sunt cerute pentru conectarea la baza de date.

Dacă aveți pachetul PHPMyAdmin (sau un program similar) instalat, atunci puteți să intrați acolo şi să vă conectați introducând contul (numele de utilizator) şi parola. Dacă nu aveți aşa ceva, atunci trebuie să faceți întreaga administrare a bazelor de date folosind script-uri PHP (şi o puteți face).

Crearea unei Tabele

Mai înainte de orice altceva cu baza de date, trebuie să vă creați o tabelă. O tabelă este o secțiune a bazei de date pentru memorarea unor informații structurate (legate). Intr-o tabelă vom defini diferite câmpuri care vor fi folosite în acea tabelă. Din cauza acestei construcții, aproape toate centrele cu baze de date trebuie să fie satisfăcute folosind doar o bază de date.

Crearea unei tabele în PHPMyAdmin este simplă, scrieți numele, selectați numărul de câmpuri şi &Quot;apăsați" butonul (*click*). Ajungeți atunci la un ecran *setup* în care trebuie să creați câmpurile pentru baza de date. Dacă folosiți un script PHP pentru a crea propria bază de date, iar completa creere şi inițializarea vor fi făcute într-o singură comandă.

Câmpuri

Există o largă varietate de câmpuri şi de atribute disponibile în MySQL şi vom discuta doar câteva din ele:

Tipul câmpului	**Descriere**
TINYINT	Număr Intreg mic
SMALLINT	Număr Intreg mic
MEDIUMINT	Număr Intreg
INT	Număr Intreg
VARCHAR	Text (maximum 256 caractere)
TEXT	Text

Acestea sunt doar câteva dintre câmpurile disponibile. O căutare pe Internet ne poate furniza lista cu toate tipurile de câmpuri permise.

Crearea unei Tabele cu PHP

Să creem o tabelă din PHP este ceva mai dificil decât cu MySQL. Avem de parcurs următorii paşi:

```
CREATE TABLE nume_tabel {

Fields

}
```

Câmpurile sunt definite după cum urmează:

```
fieldname type(length) extra info,
```

Ultimul câmp introdus nu poate include nici o virgulă.

O să dăm îndată un exemplu complet privind aceste definiri.

Baza de date cu adrese de contact

Baza de date de contact va conține toate informațiile de contact ale cunoscuților introduşi în tabel. Iar informațiile vor putea fi editate şi consulate în internet. Următoarele câmpuri vor fi folosite în baza de date:

Nume	Tipul	Lungimea	Descrierea
id	INT	6	*Un identificator unic pentru fiecare înregistrare*
Nume	VARCHAR	15	*Numele de familie al persoanei*
prenume	VARCHAR	15	*Numele de botez al persoanei*
telefon	VARCHAR	20	*Numărul de telefon*
mobil	VARCHAR	20	*Numărul de telefon mobil*
fax	VARCHAR	20	*Numărul de fax*
email	VARCHAR	30	*Adresa e-mail*
web	VARCHAR	30	*Pagina Web personală*

Poate vă miraţi că am folosit un tip de câmp VARCHAR pentru coloana mumăr_telefon/fax, chiar dacă acestea sunt formate din cifre. Chiar dacă am putea folosi tipul INT, este preferabil să folosim VARCHAR pentru că astfel vor fi permise spaţii şi cratime, precum şi porţiuni de text, la fel ca numerele scrise ca text (exemplu 1800-COMPANY) şi cum nu vom iniția apeluri telefonice de pe Web, totul e în ordine.

Există încă ceva de care trebuie să avem grije în această bază de date. Câmpul id va fi pus ca PRIMARY, INDEX, UNIQUE şi iniţializat ca *auto_increment* (poziţie ce apare în Extra în PHPMyAdmin). Raţiunea pentru asta este că acesta va fi câmpul identificator (primar sau index) şi deci trebuie să fie unic. Definirea sa ca *auto increment* înseamnă că la adăugarea fiecărei înregistrări, câtă vreme nu specificăm acolo un id, acesta va primi ca valoare următorul număr.

Dacă folosim PHPMyAdmin sau un program de management, putem crea o tabelă numită contacte.

Crearea Tabelei în PHP

Pentru a crea această tabelă vom folosi următoarea secvenţă de comenzi PHP. O parte dintre comenzile de mai jos n-au fost încă discutate, dar le vom explica în detaliu în lecţia următoare.

```
<?
$user="username";
$password="password";
$database="database";
mysql_connect(localhost,$user,$password);
@mysql_select_db($database) or die( "Baza de date nu poate fi selectata");
$query="CREATE TABLE contacts (id int(6) NOT NULL auto_increment,nume
varchar(15) NOT NULL,prenume varchar(15) NOT NULL,telefon
varchar(20) NOT NULL,mobil varchar(20) NOT NULL,fax varchar(20) NOT
NULL,email varchar(30) NOT NULL,web varchar(30) NOT
NULL,PRIMARY KEY (id),UNIQUE id (id),KEY id_2 (id))";
mysql_query($query);
mysql_close();
?>
```

Introduceţi numele bazei de date, contul MySQL şi parola MySQL în poziţiile corespunzătoare din primele 3 linii de mai sus.

PASUL 3 – Introducerea datelor

Pe parcursul precedentelor lecţii am văzut ce vrem să învăţăm în acest curs şi am văzut cum putem crea o bază de date pe care s-o folosim în acest curs. In această lecţie vom vedea cum introducem anumite informaţii în baza noastră de date astfel încât să devină cât mai utilă.

Conectarea la o bază de date

Primul lucru care trebuie făcut, mai înainte de a putea face ceva, este conectare la baza de date MySQL. Acesta este un pas foarte important, pentru că, dacă nu suntem conectați, comenzile către baza de date vor eşua.

Practic, pentru a folosi o bază de date trebuie să precizăm numele de utilizator (*username*), parola (*password*) şi numele bazei de date :

```
$username="nume_utilizator";
$password="parola";
$database="numele_bazei_de_date";
```

Desigur, aici s-ar putea discuta dacă nu-i riscant să păstrăm parola într-un fişier. Nu trebuie să vă alarmați, însă, deoarece sursa PHP este prelucrată de server înainte de a fi trimisă navigatorului, astfel încacirc;t este imposibil pentru orice utilizator să vadă scriptul.

Mai apoi, avem nevoie de o comandă care să lanseze conexiunea la baza de date:

```
mysql_connect(localhost,$username,$password);
```

Această linie spune PHP-ului să se conecteze la serverul de baze de date MySQL la 'localhost' (*localhost* se numeşte serverul pe care rulează PHP-ul. In afara cazului în care *gazda Web* indică altceva, vom folosi *localhost*) folosind numele de cont memorat în $username şi parola din $password.

Mai înainte de a discuta cum lucrăm cu baza de date, vom vedea încă o comandă:

```
mysql_close();
```

Aceasta este o comandă foarte importantă care închide conexiunea cu serverul de baze de date. Scriptul va rula încă, dacă nu includem această comandă, iar prea multe conexiuni MySQL deschise pot cauza probleme serverului web. Este un obicei bun să includem comanda de închidere de mai sus după ce am introdus toate comenzile către baza de date, pentru a menține platforma în bună stare.

Selectarea bazei de date

După ce ne-am conectat la severul de baze de date, trebie să selectăm baza de date pe care vrem s-o folosim. Trebuie să fie o bază de date la care avem acces, cu respectivul nume de utilizator. Se foloseşte următoarea comandă:

```
@mysql_select_db($database) or die( "Baza de date nu poate fi selectata");
```

Aceasta spune PHP-ului să selecteze baza de date specificată în variabila $database (pe care am definit-o mai înainte). Dacă conexiunea nu se poate realiza procesul (execuția script-ului) se opreşte afişând textul:

```
Baza de date nu poate fi selectata
```

Această parte suplimentară 'or die' este bună pentru ieşire, chiar dacă nu asigură decât un minim control al erorii.

Execuția comenzilor

După conectarea la server şi selecția bazei de date dorite, putem începe execuția comenzilor pe server.

Există două moduri de a executa o comandă. Prima revine pur şi simplu la introducerea comenzii în PHP. Asta merge atunci când nu apar rezultate ca urmare a execuției comenzii.

Cea de a doua variantă este să definim comanda ca o variabilă. Asta va atribui variabilei rezultatele operației.

In această lecție vom folosi prima cale, deoarece nu aşteptăm răspuns de la baze de date. Comanda va arăta cam aşa:

```
mysql_query($query);
```

Folosirea acestei forme a comenzii este utilă pentru că repetăm pur şi simplu aceiaşi comandă iarăşi şi iarăşi fără a fi nevoie să memorăm altele. Tot ce avem de făcut este să schimbăm variabila.

Introducerea datelor

Acum ne vom întoarce la baza de date cu adrese de contact, pe care am creat-o în lecția anterioară. Pentru a introduce primele informații în baza de date:

Nume: Golescu
Prenume: Gina
Telefon: 021 011012
Mobil: 0722 112233
Fax: 021 556677
E-mail: golescug@gmail.com
Web: http://www.acces.ro

Toate acestea vor fi introduse cu o singură comandă:
$query = "INSERT INTO contacts VALUES ('','Golescu','Gina','021 011012','0722 112233','021556677','golescug@gmail.com','http://www.acces.ro')";

Pare puțin cam confuz la prima vedere. Să lămurim puțin lucrurile.

Ma întâi, apare variabila $query căreia îi atribuim o comandă (vezi paragraful precedent). Următoarea parte, adică:

INSERT INTO contacts VALUES

este destul de uşor de înțeles. Ea spune PHP-ului să insereze în tabel numită *contacts* valorile care urmează (scrise între paranteze).

Acolo, între paranteze, avem toate câmpurile de adăugat. Apar toate câmpurile în ordine şi sunt inserate informațiile dintre ghilimele. Spre exemplu:

GOLESCU va fi inserat în al 2-lea câmp, care, în această tabelă este câmpul 'nume' . Poate ați reținut că nu am inserat nimic în primul câmp din baza de date (id). Asta din cauză că acest câmp va fi un câmp index, elementul unic de identificare. Fiecare înregistrare din baza de date va avea un unic ID. Din această cauză, când încărcăm baza de date, vom pune ID pe 'Auto Increment'. Asta înseamnă că, ne-atribuindu-i nici o valoare, el va lua la fiecare înregistrare următoarea valoare din şir. Iar prima înregistrare va avea valoarea ID=1 .

PASUL 4 – Afisarea datelor-

Introducere

Până acum, am creat o bază de date şi am încărcat în ea informații. In această lecție vom vedea cum se realizează o pagină de intrare pentru această bază de date, şi cum se afişează conținutul acesteia.

Introducerea datelor dintr-o pagină HTML

Introducerea datelor din paginile HTML este aproape identică cu inserarea lor din script-uri PHP. Avantajul constă, însă, în faptul că nu apare nevoia schimbării script-ului pentru fiecare câmp de introdus. In plus, putem permite "vizitatorilor" să introducă direct datele lor.

Iată mai jos cum arată o pagină HTML cu celule text pentru introducerea detaliilor corespunzătoare:

```
<form action="insert.php" method="post">
Nume Familie: <input type="text" name="nume"><br>
Prenume: <input type="text" name="prenume"><br>
Telefon: <input type="text" name="telefon"><br>
Mobil: <input type="text" name="mobil"><br>
Fax: <input type="text" name="fax"><br>
E-mail: <input type="text" name="email"><br>
Web: <input type="text" name="web"><br>
<input type="Submit">
</form>
```

Această pagină poate fi, desigur, formatată şi pot apare diferite schimbări - variante. Este doar un formular de pornire, ilustrativ. Va trebui însă editat script-ul din lecţia precedentă, pentru ca, în locul introducerii informaţiilor direct din script în baza de date, să fie folosite variabilele:

```
<?
$username="username";
$password="password";
$database="baza_mea_de_date";

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or die( "Baza de date nu poate fi selectata");

$query = "INSERT INTO contacts VALUES
('','$nume','$prenume','$telefon','$mobil','$fax','$email','$web')";

mysql_query($query);

mysql_close();
?>
```

Acest script trebuie salvat ca fişier cu numele *insert.php*, astfel ca să poată fi apelat de formularul HTML. Treaba va merge întrucât, în loc ca datele să fie introduse local, ele se introduc în formular şi sunt memorate în variabilele care apar precizate acolo şi care sunt transmise apoi PHP-ului.

Putem să adăugăm script-ului un mesaj care să confirme preluarea datelor. Asta face parte din 'oferta' de bază a PHP şi rămâne ca exerciţiu.

Scoaterea Datelor

Acum avem în baza de date cel puţ'in o înregistrare, dacă nu mai multe. Se pune problema cum vizualizăm, cum scoatem aceste date folosind PHP. Cunoaşterea elementelor de programare din PHP este necesară, recomandabilă fiind parcurgerea Mini-cursului de PHP înaintea acestuia.

Prima comandă la care vom apela este comanda SELECT din SQL, folosită într-o cerere MySQL în forma:

```
SELECT * FROM contacts
```

Aceasta este o comandă de bază din MySQL, care va spune script-ului să selecteze toate înregistrările din tabela *contacts*. Dar, de data aceasta, comanda furnizează un rezultat, ea va trebui executată atribuin rezultatele unei variabile:

```
$query="SELECT * FROM contacts";
$rezultat=mysql_query($query);
```

In acest caz, întregul conținut al bazei de date va fi atribuit variabilei cu numele $rezultat (care va fi deci o matrice, un tablou). Mai înainte de a putea scoate aceste date va trebui să le extragem ca variabile simple, separate. Pentru asta trebuie să ştim câte înregistrări avem în tabelă (deci şi în variabila $rezultat).

Numărarea liniilor

Avem o comandă specială în MySQL pentru calculul numărului de linii din tabelă. Este importantă pentru că în baza de date se fac uzual numeroase actualizări, completări, ştergeri.

```
$num=mysql_numrows($rezultat);
```

Astfel, variabila $num va căpăta ca valoare numărul de linii din $rezultat (adică tocmai din baza de date). Acest număr va putea fi folosit în continuare pentru ciclul în care se vor analiza şi desface în variable separate câmpurile fiecărei linii.

Construirea ciclului

Avem de scris un ciclu pentru a selecta din tabel (din rezultat) linie cu linie.... Definim un indice, o variabilă $i care va parcurge (cu incrementare) valorile de la 1 la $num .

```
$i=0;
while ($i < $num) {

OPERATIILE ASUPRA LINIEI

++$i;
}
```

Acesta este un ciclu tipic în PHP şi va executa ansamblul numit OPERATIILE ASUPRA LINIEI de exact $num ori, adică de atâtea ori cât trebuie. La fiecare reluare $i fiind mărit cu o unitate. Astfel $i poate fi folosit şi pentru a preciza numărul liniei care se prelucrează.

Asocierea datelor la variabile

In partea pe care am numit-o OPERATIILE ASUPRA LINIEI trebuie să separăm fiecare câmp din linia - înregistrare asociindu-l unei variabile. Folosim pentru asta următoarea secvență:

```
$variable=mysql_result($rezultat,$i,"numele_campului");
```

Astfel, pentru a extrage fiecare componentă din baza noastră de date, vom folosi următoarele instrucțiuni:

```
$nume=mysql_result($result,$i,"nume");
$prenume=mysql_result($result,$i,"prenume");
$telefon=mysql_result($result,$i,"telefon");
$mobil=mysql_result($result,$i,"mobil");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");
```

Nu ne-am ocupat aici de câmpul ID (deşi o puteam face) pentru că nu ne este necesar la afişare datelor.

Combinarea Script-ului

Acum putem scrie script-ul complet pentru scoaterea datelor. In acest script datele nu sunt formatate, adică vor fi afişate toate cu acelaşi *font*, predefinit.

```
<?
$username="username";
$password="password";
$database="baza_noastra_de_date";

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or die( "Baza de date nu poate fi selectata");
$query="SELECT * FROM contacts";
$rezultat=mysql_query($query);

$num=mysql_numrows($rezultat);

mysql_close();

echo "<b><center>Database Output</center></b><br><br>";
```

```
$i=0;
while ($i < $num) {

$nume=mysql_result($rezultat,$i,"nume");
$prenume=mysql_result($result,$i,"prenume");
$telefon=mysql_result($result,$i,"telefon");
$mobil=mysql_result($result,$i,"mobil");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");

echo "<b>$nume $prenume</b><br>Telefon: $telefon<br>Mobil: $mobil<br>Fax:
$fax<br>E-mail: $email<br>Web: $web<br><hr><br>";

++$i;
}

?>
```

PASUL 5 – Alte rezultate-

Introducere

Până acum am văzut cum creem o bază de date - tabelele din care este formată, cum inserăm informațiile şi cum afişăm datele din baza de date. In această lecție vom vedea mai multe moduri de a afişa - scoate date din baza de date.

Formatarea scoaterii

In ultima parte a lecției precedente am scos o listă cu toate persoanele înregistrate în baza de date. De fapt am ajuns să avem definite variabilele care permit scoaterea sau afişarea, încheind cu o comandă destul de simplă de afişare (*echo*) fără a o explica. Desigur, datele din tabel ar trebui prezentate pe ecran tot în forma unui tabel. Iar asta nu-i prea complicat. In fond dacă ştim cum să afişăm (scoatem) fiecare variabilă în parte, atunci tot restul privind formatarea, organizarea ca tabel, etc., este doar legat de cunoaşterea HTML.

Tot ce avem de făcut deci este să folosim PHP pentru scoaterile HTML incluzând variabilele în zonele corecte. Cel mai simplu este să închidem tagul PHP şi să introducem liniile normale din HTML. Oridecâte ori ajungem la o variabilă o vom include folosind o secvență de forma:

```
<? echo "$numele_variabilei"; ?>
```

Putem atunci să folosim un ciclu PHP pentru a repeta o secvență de cod ca cea de mai sus, la includerea variabilelor într-un tablou. Spre exemplu, folosind secvența de cod din lecția precedentă cu ciclul corespunzător, putem formata scoaterile pentru ca să apară într-o tabelă mare:

```
<table border="0" cellspacing="2" cellpadding="2">
<tr>
<th><font face="Arial, Helvetica, sans-serif">Nume</font></th>
<th><font face="Arial, Helvetica, sans-serif">Telefon</font></th>
<th><font face="Arial, Helvetica, sans-serif">Mobil</font></th>
<th><font face="Arial, Helvetica, sans-serif">Fax</font></th>
<th><font face="Arial, Helvetica, sans-serif">E-mail</font></th>
<th><font face="Arial, Helvetica, sans-serif">Website</font></th>
</tr>

$i=0;
while ($i < $num) {

$nume=mysql_result($rezultat,$i,"nume");
$prenume=mysql_result($result,$i,"prenume");
$telefon=mysql_result($result,$i,"telefon");
$mobil=mysql_result($result,$i,"mobil");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");
?>
<tr>
<td><font face="Arial, Helvetica, sans-serif"><? echo "$nume $prenume";
?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><? echo "$telefon"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><? echo "$mobil"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><? echo "$fax"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><a href="mailto:<? echo "$email";
?>">E-mail</a></font></td>
<td><font face="Arial, Helvetica, sans-serif"><a href="<? echo "$web";
?>">Website</a></font></td>
</tr>

<?
++$i;
}


echo "</table>";
```

Această secvență va afișa antetul tabelului, apoi va adăuga o linie suplimentară pentru fiecare înregistrare din baza de date, formatând datele la scoatere.

Dacă sunteți deja familiarizați cu PHP şi HTML, atunci lucrurile sunt probabil destul de clare - de uşor de înțeles. O să explicăm doar una din liniile din tabel, spre exemplu:

```
<a href="mailto:<? echo "$email"; ?>">E-mail</a>
```

care construieşte o legătură email la adresa transmisă de variabila $email . Asta arată una dintre calitățile importante şi utile ale folosirii PHP pentru includerea datelor MySQL . Adică prin astfel de scoateri putem face paginile Web dinamice.

Selectarea unor date

La fel cu afişarea întregii baze de date, PHP poate fi folosit pentru a selecta date individuale, doar anumite înregistrări, sau înregistrările care verifică anumite criterii. Pentru asta trebuie să folosim o variațiune a cererii SELECT . Pentru afişarea întregii tabele am folosit cererea:

```
SELECT * FROM contacts
```

Dacă vrem însă să selectăm doar pe acele persoane care au prenumele 'Mihai' vom folosi o cerere de forma:

```
SELECT * FROM contacts WHERE prenume='mihai'
```

Ca şi la alte cereri - comenzi SQL, avem de fapt propoziții foarte apropiate de formularea curentă din limba engleză. Intr-un mod asemănător vom putea selecta înregistrările pe baza oricărui câmp din baza de date. Dar se poate selecta o înregistrare folosind mai multe câmpuri, adăugând în formularea cererii clauza:

```
field='value'
```

Fără a intra în prea multe detalii, vom mai spune că putem folosi variabilele pentru a transmite criteriul dorit pentru selecție. Spre exemplu, dacă dintr-un formular de căutare primim a variabilă numită $nume_cautat putem imagina următoarea secvență:

```
$query="SELECT * FROM contacts WHERE nume='$nume_cautat'";
$result=mysql_query($query);
```

Rețineți că la sfârşitul primei linii avem un **'** urmat de **"** , înainte de **;** .

PASUL 6 – Inregistrari si erori-

Introducere

In ultimele două lecţii am văzut cum extragem date din baza de date şi cum le afişăm pe ecran. Acum vom ajunge la aspectele finale ale afişării datelor, prin selectarea elementelor dorite şi controlul erorilor (stoparea mesajelor de eroare) atunci când scoatem date.

Interceptarea Erorilor

Prin scoaterea tuturor informațiilor dintr-o bază de date, este puțin probabil să ajungem la situația când nu mai sunt date. Dar dacă am permis ajustări şi, ştergeri şi actualizări ale înregistrărilor, atunci se prea poate să ajungem la o eroare. Din fericire, cu PHP şi MySQL, avem un mod simplu de a evita o astfel de situație folosind:

```
$num=mysql_numrows($rezultat);
```

unde $rezultat conține rezultatul unei cereri - interogări a bazei de date (precum selectarea tuturor înregistrărilor). Aşa cum am discutat mai sus, aceasta va atribui variabilei $num numărul de linii din rezultat (care s-a utilizat într-un ciclu, în lecția a 4-a). Putem insera în ciclu o comandă de captare/tratare a erorilor folosind o instrucțiune IF :

```
if ($num==0) {
echo "Baza de date nu conține nici o înregistrare";
} else {
Output Loop
}
```

Putem dezvolta ramura asta făcând-o mai prietenoasă. Spre exemplu, oferind o legătură la pagina Add Data, de introducere de informații în baza de date, atunci când ea este vidă.

Ordonarea datelor

Nu numai că putem scoate datele în funcție de conținutul unui câmp, dar putem ordona aceste date pe baza unei reguli aplicată conținutului unei coloane (spre exemplu aranjând utilizatorii în ordine alfabetică). In mod normal, afişarea în urma unei interogări se face în ordinea stabilită de identificatorul ID, pornind de la 1 în sus. Putem însă alege modul de ordonare (sortarea) după oricare coloană din tabel.

Spre exemplu, o ordonare utilă ar putea fi după numele de botez. Asta însemnând în ordine ascendentă (crescătoare, de la A la Z şi de la 1 la 10...). Pentru a obține un astfel de rezultat folosim următoarea cerere:

SELECT * FROM contacts ORDER BY prenume ASC

Putem folosi, desigur şi ordonarea descendentă, specificând DESC în locul lui ASC .

Alte variante cu mysql_numrows şi Sortare

Valoarea care i se atribuie (ca mai sus) variabilei $num este foarte importantă, nu numai pentru cicluri şi captarea erorilor. Un exemplu poate fi scoaterea doar a ultimelor 5 înregistrări adăugate bazei de date. Mai întâi, avem ordonarea naturală, stabilită de ID, (ultima înregistrare având valoarea maximă a ID), dar vom alege ordinea descendentă.

Astfel vom avea înregistrările începând cu cea mai recentă şi terminând cu cea mai veche. Mai trebuie doar să numărăm, afişând doar primele 5.

Desigur, înainte de a începe ciclul de cinci, trebuie să ne asigurăm că $num este mai mare decât 5 . Avem deci o secvență de forma:

```
if ($num<5) {
$to=$num;
}else{
$to=5;
}

$i=0;
while ($i < $to) {
SECVENTA DE COMENZI mysql PENTRU SCOATERE
```

Cu alte cuvinte, dacă avem mai mult de cinci linii în tabel atunci ciclul se va face de la 0 la 5. In caz contrar, dacă sunt mai puțin de 5 linii, ciclul va parcurge exact numărul respectiv de linii.

Câmpul ID

Atunci când, în primele lecții am creat baza de date (cartea de adrese), am inclus un câmp numeric numit id. Pe care l-am stabilit ca *auto_increment* şi i-am dat rolul de câmp primar. Am discutat cum acesta are rolul de identificator unic pentru fiecare înregistrare din baza de date. Acum facem un pas înainte, folosind acest câmp pentru a selecta anumite înregistrări din baza de date.

Selecția unei singure înregistrări

Iată cum putem selecta înregistrări din baza de date folosind conținutul unui câmp particular:

```
SELECT * FROM contacts WHERE field='value'
```

Atunci, folosind unicitatea câmpului ID putem selecta orice înregistrare din baza de date, folosind:

```
SELECT * FROM contacts WHERE id='$id'
```

unde $id este o variabilă conținând numărul unei înregistrări. Spre exemplu, dacă dorim să avem o pagină Web generată dinamic dintr-o bază de date cu un singur script PHP, putem scrie script-ul ca să includă pagini Web distincte ca înregistrări ale bazei de date. Atunci, folosind câmpul id, putem selecta fiecare pagină individuală plasând-o la soatere. Putem chiar folosi chiar URL-ul paginii pentru a specifica înregistrarea dorită

```
http://www.centrul_propriu.ro/stiri/items.php?item=5476
```

Iar script-ul PHP să caute înregistrarea care are numărul de ordine (id-ul) care corespunde valorii variabilei $item, care în acest caz este 5476

Legături la o singură înregistrare

Folosind această metodă de alegere a înregistrării folosind URL-ul, înregistrarea poate fi extinsă prin generarea dinamică a URL-ului. Pare cam complicat. Vom vedea cum realizăm o pagină de aducere la zi a bazei de date cu adresele. Cu ideea ca utilizatorul să-şi poată modifica propriile detalii din înregistrare.

Pentru asta, vom include o nouă coloană cuprinzând o legătură Update . Această legătură conducând la o pagină care să permită utilizatorului să actualizeze înregistrarea. Pentru a selecta înregistrarea din acea pagină vom pune:

```
?id=$id
```

Căpătând identificatorul id al înregistrării, secvența aceasta va crea o legătură la fiecare din înregistrări.

PASUL 7 – Actualizare si stergere-

Introducere

Până acum am învățat cum să punem informațiile în baza de date MySQL, cum să vedem aceste informații din baza de date, selectând care din ele vrem să le vedem. In această lecție vom vedea cum facem cele două acțiuni finale, actualizarea bazei de date şi ştergerea unor înregistrări din ea.

Script-ul de actualizare

Am văzut în lecția anterioară cum creem o legătură pentru fiecare înregistrare pentru a ne poziționa în scriptul de actualizare. Prin folosirea variabilei $id , legăturile respective pot transmite valoarea corecta a identificatorului ID către script, astfel ca acesta să poată actualiza baza de date. Vom realiza deci un script de actualizare, care va avea două părți:

Pagina de afişare a actualizării

Prima parte a script-ului de actualizare foloseşte procedeul de selecție a unei singure înregistrări, aşa cum l-am scris în lecția precedentă, adăugând doar câteva elemente HTML pentru a-l face mai util. Mai întâi, ne connectăm la baza de date şi selectăm înregistrarea potrivită.

```
$id=$_GET['id'];
$username="nume_utilizator";
$password="parola";
$database="baza_de_date";
mysql_connect(localhost,$username,$password);

$query=" SELECT * FROM contacts WHERE id='$id'";
$result=mysql_query($query);
$num=mysql_numrows($rezultat);
mysql_close();

$i=0;
while ($i < $num) {
$nume=mysql_result($rezultat,$i,"nume");
$prenume=mysql_result($rezultat,$i,"prenume");
$telefon=mysql_result($rezultat,$i,"telefon");
$mobil=mysql_result($rezultat,$i,"mobil");
$fax=mysql_result($rezultat,$i,"fax");
$email=mysql_result($rezultat,$i,"email");
$web=mysql_result($rezultat,$i,"web");

Zona de cod suplimentar

++$i;
}
```

Unde 'Zona de cod suplimentar' marchează porțiunea din script unde vor apare comenzile de actualizare. Adică formatarea HTML pentru scoatere:

```
<form action="updated.php" method="post">
<input type="hidden" name="ud_id" value="<? echo "$id"; ?>">
Numele de Familie: <input type="text" value="ud_first" value="<? echo
"$nume"?>"><br>
Numele de Botez: <input type="text" value="ud_last" value="<? echo
"$prenume"?>"><br>
Numărul de Telefon: <input type="text" value="ud_phone" value="<? echo
"$telefon"?>"><br>
Numărul de Mobil: <input type="text" value="ud_mobile" value="<? echo
"$mobil"?>"><br>
Numărul de Fax: <input type="text" value="ud_fax" value="<? echo
"$fax"?>"><br>
Adresa E-mail: <input type="text" value="ud_email" value="<? echo
"$email"?>"><br>
Adresa Web: <input type="text" value="ud_web" value="<? echo
"$web"?>"><br>
<input type="Submit" value="Update">
</form>
```

Aşa cum se poate vedea, această secvență construieşte (scoate) un formular standard, dar în locul zonelor goale, aşa cum apăreau în formularul pentru introducerea datelor, de data asta avem conținutul câmpului respectiv din înregistrarea în cauză (care-i de actualizat). Asta îl face mai adaptat scopului, mai comod de folosit.

Actualizarea bazei de date

Tot ce mai avem de făcut este actualizarea efectivă a bazei de date. Asta este o operație simplă care implică o nouă cerere pentru baza de date:

```
$query = "UPDATE contacts SET nume = '$ud_first',prenume = '$ud_last',telefon
= '$ud_phone',mobil = '$ud_mobile',fax = '$ud_fax',email = '$ud_email',web =
'$ud_web' WHERE id = '$ud_id'";
```

Această cerere spune sistemului de gestiune de baze de date să actualizeze tabela *contacts* în acele linii în care ID coincide cu valoarea din $ud_id (care, aşa cum se poate vedea din formularul anterior, a primit valoarea id a înregistrării pe care o actualizăm), modificând următoarele câmpuri cu valorile specificate (care au fost introduse cu ajutorul formularului).

Această cerere poate fi înglobată într-un script simplu:

```
$ud_id=$_POST['ud_id'];
$ud_first=$_POST['ud_first'];
$ud_last=$_POST['ud_last'];
$ud_phone=$_POST['ud_phone'];
$ud_mobile=$_POST['ud_mobile'];
$ud_fax=$_POST['ud_fax'];
$ud_email=$_POST['ud_email'];
$ud_web=$_POST['ud_web'];

$username="nume_utilizator";
$password="parola";
$database="baza_de_date";
mysql_connect(localhost,$username,$password);

$query="UPDATE contacts WHERE id='$ud_id' SET first='$ud_first'
last='$ud_last' phone='$ud_phone' mobile='$ud_mobile' fax='$ud_fax'
email='$ud_email' web='$ud_web'";
mysql_query($query);
echo "Actualizarea s-a făcut";
mysql_close();
```

Asta va actualiza baza de date transmițând şi o confirmare utilizatorului.

Stergerea înregistrărilor

Ultima parte a acestei lecții privește modul cum ştergem o înregistrare dim baza de date. Ca şi cu pagina de actualizare, vom construi o pagină pentru a şterge una sau mai multe linii din baza de date. Trebuie să-i transmitem poziția (ID-ul) înregistrării, printr-un URL, spre exemplu:

```
delete.php?id=9
```

Scriptul care va face asta, numit *delete.php*, este aproape identic cu cel de actualizare a bazei de date, cu excepția comenzii MySQL (modului în care este construită cererea). In locul comenzii SQL UPDATE , vom folosi:

```
DELETE FROM contacts WHERE id='$id'
```

Ciclurile

La acest punct este momentul să menționăm şi un alt mod de folosire a ciclurilor cu o bază de date. Putem folosi un ciclu pentru a executa un şir de cereri. Spre exemplu, dacă trebuie să schimbăm - extragem toate înregistrările dintr-o bază de date în care apare ca prenume Serban pentru a realiza un Website www.serban.ro:

```
Partea Standard de Conectare la Baza de Date

$query=" SELECT * FROM contacts WHERE prenume='Serban'";
$rezultat=mysql_query($query);
$num=mysql_numrows($rezultat);

$i=0;
while ($i < $num) {
$id=mysql_result($rezultat,$i,"id");
$query1="UPDATE contacts SET web='http://www.serban.ro' WHERE id='$id'";
mysql_query($query);
++$i;
}

mysql_close();
```

PASUL 8 – Incheierea Scriptului-

Introducere

Pe parcursul acestui curs am văzut cum să folosim PHP-ul pentru a interacționa cu un sistem de baze de date MySQL (sau SQL) şi cum să folosim comenzile uzuale (cele mai necesare). Am exemplificat, cu modul de realizare a unei baze de date de tip carte de adrese (ca sistem de management a contactelor ?). In această lecție finală, vom vedea câteva "trucuri" MySQL şi vom scrie versiunea definitivă (finală) a script-ului construit pe parcursul acestui curs.

Pentru economia de timp

Atunci când construim un script complex folosind bazele de date, apare des secvența de conectare la baza de date. De aceea, pentru a simplifica lucrurile, se poate fie să creem un fişier cu numele de utilizator şi parola sau chiar un fişier de conectare. Spre exemplu, un fişier "username/password" poate fi construit cu numele:
dbinfo.inc.php

cuprinzând în el:

```
<?
$username="numele_de_utilizator_al_bazei_de_date";
$password="parola";
$database="nume_baza_de_date";
?>
```

In care precizăm datele concrete potrivite (numele_de_utilizator_al_bazei_de_date, parola, nume_baza_de_date). Atunci în fişierele php vom folosi, chiar la început, următoarea secvență, care va include scriptul de mai sus:

```
include("dbinfo.inc.php");
```

sau, atunci când acesta se află în alt director:

```
include("/[traseul complet]/dbinfo.inc.php");
```

Atunci, vom putea folosi în continuare variabilele $username, $password şi $database în script-ul nostru, fără a avea nevoie să le definim de fiecare dată. De asemeni, dacă vom modifica cândva aceste informații, spre exemplu trecând pe alt server web, tot ce va fi de schimbat va fi în acest unic fişier.

Aceeaşi schemă o putem folosi pentru conectarea la baza de date, plasând şi comanda de conectare în fişier. Atunci va trebui, însă, să ne asigurăm de închiderea conexiunii, pentru a nu avea probleme cu serverul MySQL.

Căutarea

Se poate realiza şi o căutare limitată în baza de date folosind funcţia specială din MySQL. Adică prin folosirea funcţiei LIKE , în forma:

SELECT * FROM nume_tabela WHERE nume_camp LIKE '%$string%'

Asta însemnă că LIKE va spune bazei de date să folosească posibilităţile proprii de căutare. Semnele % au semnificaţia că orice alte date pot să apară în poziţia lor şi variabila $string va conţine cererea de căutare. Adică putem avea acolo un cuvânt, sau un număr, spre exemplu:

LIKE '%pian%'

ceea ce va conduce la scoaterea liniilor care includ cuvântul *pian* în câmpul specificat.

Similar, putem renunţa la unul din semnele % astfel ca să precizăm poziţia şirului de caractere:

LIKE 'pian%'

Astfel vom putea selecta doar liniile în care câmpul specificat începe cu prefixul *pian*, caz în care, spre exemplu, expresia următoare va fi evitată:

Un pian se află pe scenă.

Finalizarea Script-ului

Pe parcursul acestui mini-curs am prezentat diferite porţiuni de cod ale script-ului pentru construcţia bazei de date *contacts*. Puteţi descărca script-ul complet ca un fişier comprimat zip, pentru a examina întreaga aplicaţie (vezi legăturile).

Concluzii

Acum, încheind acest curs, ar trebui să ştiţi să folosiţi PHP şi MySQL împreună pentru a crea o bază de date - accesibilă pe Web, precum şi pentru a scrie programe de acces la baza de date. Folosirea bazelor de date pe Web deschide noi şi mari posibilităţi de lucru în Internet şi poate face mult mai puternic un centru Web, economisind timpul de actualizare, permiţând utilizatorilor să interacţioneze (să răspundă) şi multe altele.