

Curs 10

2018/2019

Programarea aplicațiilor web

CURS

I.	HTML si XHTML (recapitulare)	1 oră
II	CSS	2 ore
III	Baze de date, punct de vedere practic	1 oră
IV	Limbajul de interogare SQL	4 ore
V	PHP - HyperText Preprocessor	8 ore
VI	XML - Extended Mark-up Language si aplicatii	4 ore
VII	Conlucrare intre PHP/MySql, PHP/XML, Javascript/HTML	2 ore
VIII	Exemple de aplicatii	6 ore
	Total	28 ore

Versiunea finala

Proiect

Teme de proiect

- La toate temele **1p** din nota este obtinut de indeplinirea functionalitatii cerute.
- La toate temele forma paginii prezinta importanta (dependenta de dificultatea temei)
- Incepand din 2018/2019 **nu mai exista nota din oficiu** la proiect

PROIECT (final)

- Tema de nota **8** ~~7~~
 - Tema unica pentru fiecare student
 - Baza de date cu care se lucreaza contine minim 20 ~~15~~ de inregistrari in tabelul cel mai "voluminos«
- Tema de nota **9** ~~8~~
 - Conditiiile de la tema de nota 8 **si in plus**
 - Necesitatea conlucrarii intre 2 studenti cu doua teme "pereche"
 - Se accepta ca un student sa realizeze ambele puncte
 - Numar **minim** de pagini dinamice (php+mysql) in aplicatie **4 = 2 X 2**
 - Baza de date cu care se lucreaza contine minim 50 ~~30~~ de inregistrari in tabelul cel mai "voluminos"

PROIECT (final)

- Tema de nota **10 9**
 - Condițiile de la tema de nota 9 **si in plus**
 - Necesitatea conlucrării între 2 studenti cu teme "pereche"
 - Tema se preda/trimite cu macar 1 zi înainte de susținerea ei
 - Numar **minim** de pagini dinamice (php+mysql) in aplicatie **6 = 3 X 2**
 - Baza de date cu care se lucreaza sa contina minim 100 ~~60~~ de inregistrari in tabelul cel mai "voluminos".

PROIECT (final)

- Tema de nota **10+ 10**
 - Condițiile de la tema de nota 10 **si in plus**
 - Numar **minim** de pagini dinamice (php+mysql) in aplicatie **8 = 4 X 2**
 - Baza de date cu care se lucreaza contine minim **300** de inregistrari in tabelul cel mai "voluminos"
 - Necesitatea investigarii posibilitatilor de **imbunatatire** a aplicatiei si adaugarii de functionalitate (**obligatoriu**)
 - nota individuala la proiect va depinde intr-o mica masura (in limita a 1p) de nota minima a colegilor din echipa
 - **+1p la nota de examen**

PROIECT (final)

- proiectul se sustine individual (oral si practic)
- grila de notare la proiect schimbata fata de anii precedenti
- fiecare membru al unei echipe (la temele de nota 10 si 10+) trebuie sa sustina in aceeasi zi proiectul
- nota individuala la proiect va depinde intr-o mica masura (in limita a 1p) de nota medie a colegilor din echipa (numai la temele de 10+)
 - $N-\min(E)=1 \rightarrow -0\text{ p}$
 - $N-\min(E)=2 \rightarrow -0.5\text{ p}$
 - $N-\min(E)=3 \rightarrow -1\text{ p}$

PROIECT (final)

- In caz de necesitate, pentru completarea echipei cadrul didactic poate fi membru al echipelor (9/10/10+). Conditii:
 - metoda de comunicare in echipa sa fie prin email sau direct
 - latentă de raspuns: ~ 1 zi
 - reactiv
 - nota implicita 10 (😊)
 - nu lucreaza noaptea, si in special nu in noaptea dinaintea predarii (😊)
- dezavantaj asumat: "spion" in echipa

PROIECT (final)

- Tema bonus **10+** (>5, in general **offline**)
 - Conditiiile de la tema de nota 10+ **si in plus**
 - Baza de date cu care se lucreaza contine minim **500** de inregistrari in tabelul cel mai "voluminos"
 - Numar **minim** de pagini dinamice (php+mysql) in aplicatie **15 = 5 X 3**
 - Tema care face apel la controlul **sesiunii** client/server
 - Necesitatea utilizarii **Javascript** in **aplicatie** (aplicatie libera dar cu efect tehnic nu estetic)
 - Forma paginii trebuie sa respecte cerintele "F shape pattern"
 - Facilitati in ceea ce priveste nota (**DACA** toate celelalte conditii sunt indeplinite), la alegere:
 - prezenta la laborator – P = **66%**, L = **0%**, E = 33%
 - **+2p la nota de examen**

Exemplu

- 1. Galerie de imagini in care imaginile sunt ordonate dupa categorii.

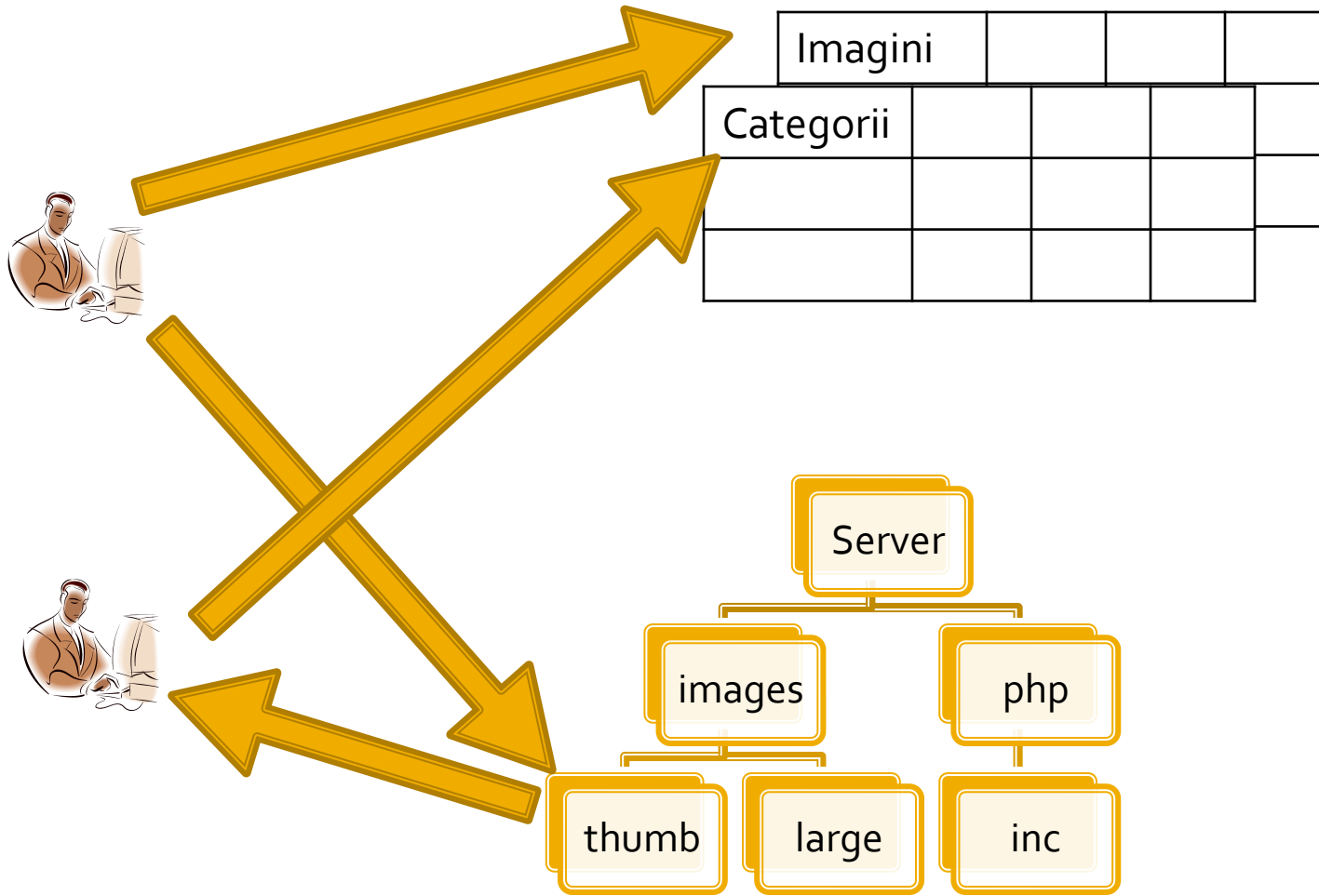


a. aplicatia pentru adaugarea de categorii si afisare a imaginilor (cu alegerea prealabila a categoriei si afisarea listei de imagini format mic)



b. aplicatia pentru adaugare de imaginilor (cu alegerea prealabila a categoriei si generarea prealabila a imaginii format mic)

Exemplu



Teme de proiect

- **Functionalitate**
 - La toate temele **1p** din nota este obtinut de indeplinirea functionalitatii cerute.
 - orice tehnologie, orice metoda, "sa faca ceea ce trebuie"
- **Forma paginii prezinta importanta**
 - dependenta de dificultatea temei
- **Initiativa**
 - **Necesitatea** investigarii posibilitatilor de imbunatatire
- **Cooperare**
 - Necesitatea conlucrarii intre 2/3 studenti cu teme "pereche"

Notare

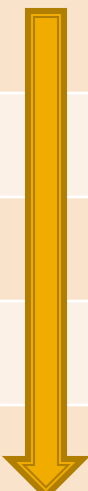
- 1p – functionalitate
 - cadrul didactic va incerca sa foloseasca aplicatia respectiva. Daca "pe dinafara e vopsit gardul" se obtine 1p
- 1p – mutarea site-ului (restaurare backup + setare server) pe un server de referinta
 - server-ul de referinta va fi masina virtuala utilizata la laborator (inclusiv aplicatiile cu pricina)
 - sa va pregatiti pentru situatia in care pe acel server exista si alte baze de date care nu trebuie distruse
 - fiecare student isi pune sursele in directorul propriu, in radacina server-ului. Daca tema depinde de anumite fisiere ale colegului, le cereti inainte
- 1p – cunoasterea codului
 - raspunsul la intrebari de genul: "unde ai facut aceasta"
- Teme "de nota 10"
 - 1p – initiativa. Investigarea posibilitatilor de imbunatatire
 - 1p – intrebari legate de cooperarea cu colegul de echipa
 - 1p – explicatii relativ la functionarea unei anumite secvente de cod

Notare proiect 2018/2019

- grila de notare diferita
 - premiera activitatii individuale
 - mai greu de obtinut note mari
- 1p – functionalitate ✓
- 1p – instalarea aplicatiei pe server-ul CentOS ✓
- numar de pagini dinamice ✓
- numar de inregistrari in baza de date ✓
- planul aplicatiei ✓

Notare 2019

- numar de pagini dinamice ✓
- numar de inregistrari in baza de date ✓
 - se verifica indeplinirea conditiilor corespunzatoare si se realizeaza **de-clasificarea** temei pana cand **ambele** conditii sunt indeplinite

Tema de nota ...	Pagini	Inregistrari
 bonus	$15 = 5 \times 3$	500
10+	$8 = 4 \times 2$	300
10	$6 = 3 \times 2$	100
9	$4 = 2 \times 2$	50
8	$1 = 1 \times 1$	20

Notare 2019

- 1p – functionalitate
- 1p – mutarea **personala** a site-ului (restaurare backup + setare server) pe un server de referinta
 - server-ul de referinta va fi masina virtuala **Centos 7.1** utilizata la laborator (inclusiv aplicatiile cu pricina)
 - sa va pregatiti pentru situatia in care pe acel server exista si alte baze de date care **nu** trebuie distruse
 - fiecare student isi pune sursele in directorul propriu, in radacina server-ului. Daca tema depinde de anumite fisiere ale colegului, le cereti inainte
- 1p – cunoasterea codului
 - raspunsul la intrebari de genul: “unde ai facut aceasta”
- Teme “de nota 10,10+”
 - initiativa. Investigarea posibilitatilor de imbunatatire
 - intrebari legate de cooperarea cu colegul de echipa
 - explicatii relativ la functionarea unei anumite secvente de cod
 - utilizare sesiune, Javascript, F shape pattern

Examen

- probleme
- fiecare student are subiect propriu
- toate materialele permise
- tehnica de calcul **nu** este necesara dar este permisa

Examen

- Oricare din temele de proiect (sau asemenea) poate constitui una din problemele de examen
 - se va cere realizarea planului / structurii logice a aplicatiei (S5)
- Se poate cere scrierea unui cod pentru realizarea anumitor operatii, fara necesitatea corectitudinii tehnice absolute (";", nume corect al functiilor, parametri functie etc.)
- Se poate cere interpretarea unui cod php/MySql cu identificarea efectului

MySql – Recapitulare rapida

Relatii in Bazele de date

Relatii in Bazele de date

- Legaturile intre tabele pot fi
 - One to One
 - One to Many
 - Many to Many
 - Unare (auto referinta)

One to One

- Fiecare tabel poate avea corespondenta **o singura linie (row) sau nici una** de cealalta parte a relatiei
- echivalent cu o relatie “bijectiva”
- analogie cu casatorie:
 - o persoana poate fi casatorita sau nu
 - daca este casatorita va fi casatorita cu o singura persoana din tabelul cu persoane de sex opus
 - persoana respectiva va fi caracterizata de aceeasi relatie “one to one” – primeste simultan un singur corespondent in tabelul initial

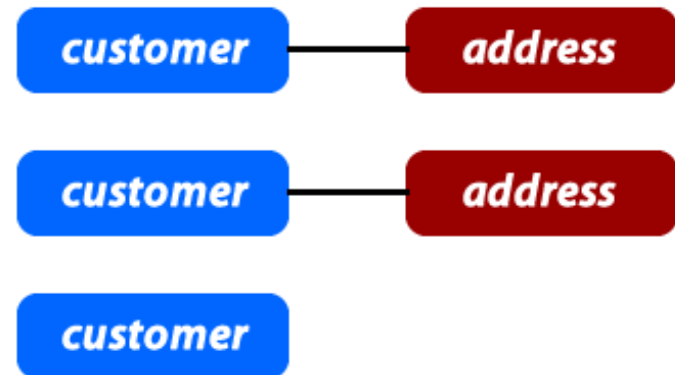
One to One

- de multe ori legaturile "one to one" se bazeaza pe reguli externe
- de obicei se poate realiza usor si eficient gruparea ambelor tabele in unul singur

CUSTOMERS		
customer_id	customer_name	address_id
101	John Doe	301
102	Bruce Wayne	302

ADDRESSES	
address_id	address
301	12 Main St., Houston TX 77001
302	1007 Mountain Dr., Gotham NY 10286

CUSTOMERS		
customer_id	customer_name	customer_address
101	John Doe	12 Main St., Houston TX 77001
102	Bruce Wayne	1007 Mountain Dr., Gotham NY 10286



One to Many

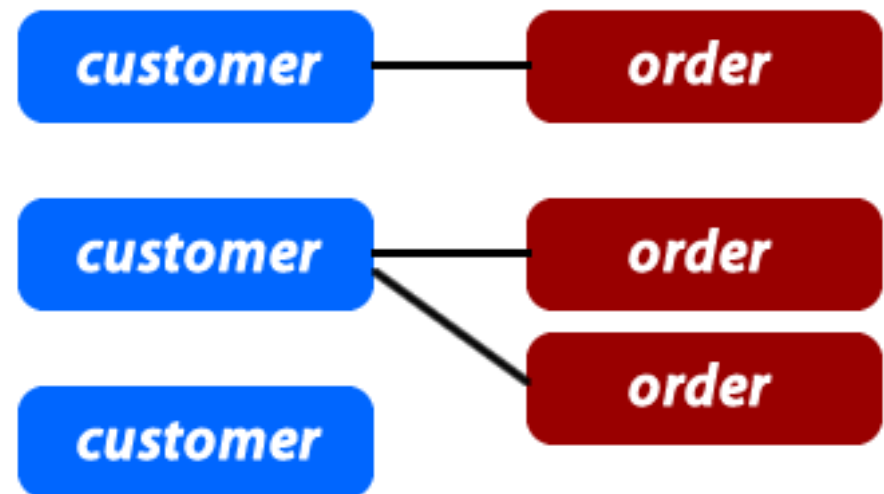
- O linie dintr-un tabel (row), identificata prin cheia primara, poate avea: **nici una, una sau mai multe linii corespondente** in celalalt tabel. In acesta o linie poate fi legata cu o **singura** linie din tabelul primar.
- Analogie cu relatii parinte/copil:
 - fiecare om are o singura mama
 - fiecare femeie poate avea nici unul, unul sau mai multi copii

One to Many, Many to One

- de obicei aceste legaturi se implementeaza prin introducerea cheii primare din tabelul **One** in calitate de coloana in tabelul **Many** (cheie externa – foreign key)

CUSTOMERS	
customer_id	customer_name
101	John Doe
102	Bruce Wayne

ORDERS			
order_id	customer_id	order_date	amount
555	101	12/24/09	\$156.78
556	102	12/25/09	\$99.99
557	101	12/26/09	\$75.00



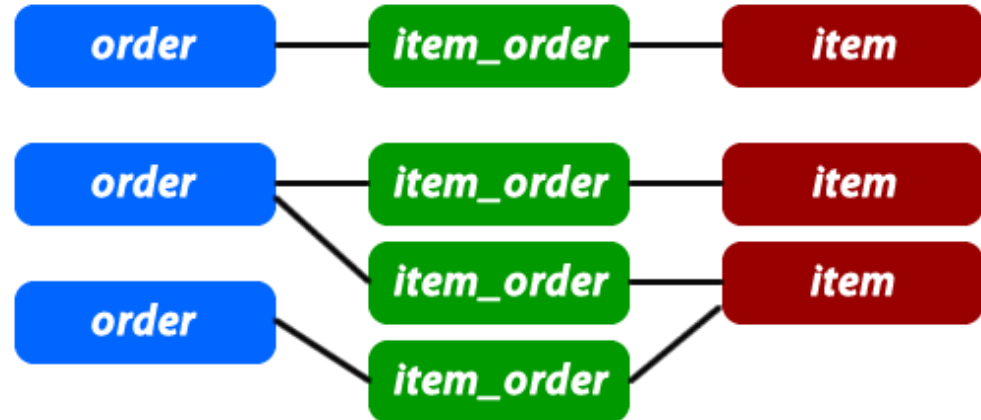
Many to Many

- Fiecare linie (row) din **ambele tabele** implicate in legatura poate fi legat cu **oricate (niciuna, una sau mai multe) linii** din tabelul corespondent.
- Analogie cu relatii de rudenie (veri de exemplu), tabel 1 – barbati, tabel 2 – femei :
 - fiecare barbat poate fi ruda cu una sau mai multe femei
 - la randul ei fiecare femeie poate fi ruda cu unul sau mai multi barbati

Many to Many

- de obicei aceste legaturi se implementeaza prin introducerea unui tabel **suplimentar** (numit tabel **asociat** sau de **legatura**) care sa memoreze legaturile

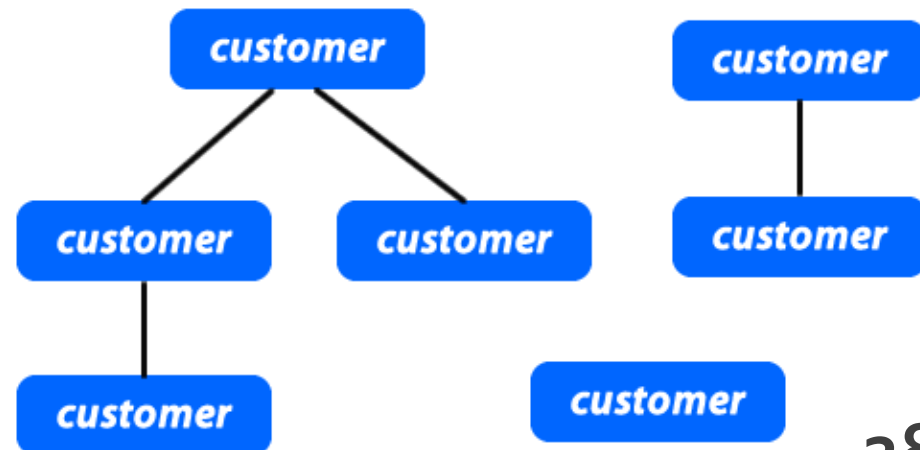
ORDERS				
order_id	customer_id	order_date	amount	
555	101	12/24/09	\$156.78	
556	102	12/25/09	\$99.99	
ITEMS				
item_id	item_name	item_description		
201	Tickle Me Elmo	It wants to be tickled		
202	District 9 DVD	Awesome sci-fi movie		
203	Batarang	It is very sharp		
ITEMS_ORDERS				
order_id	item_id			
555	201			
555	202			
556	202			
556	203			



Self Referencing (unare)

- Un caz particular de legatura "one to many" in care legatura e in interiorul aceluasi tabel
- rezolvarea este similara, introducerea unei coloane suplimentara, cu referinta la cheia primara din tabel
- analogie cu relatii parinte copil cand ambele persoane se regasesc in acelasi tabel

CUSTOMERS		
customer_id	customer_name	referrer_customer_id
101	John Doe	0
102	Bruce Wayne	101
103	James Smith	101



Relatii in Bazele de date

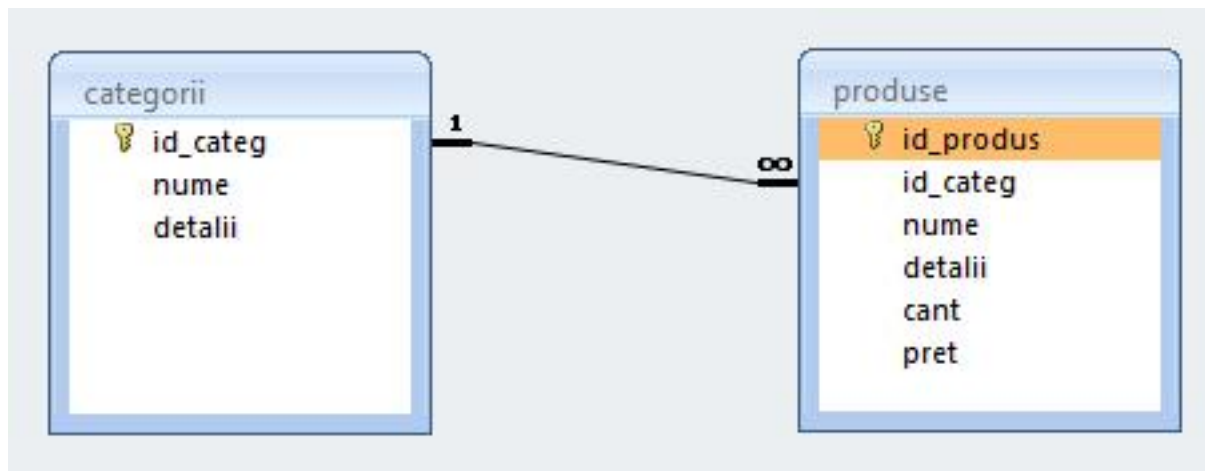
- Respectarea formelor normale ale bazelor de date aduce nenumarate avantaje
- Efectul secundar este dat de necesitatea separarii datelor intre mai multe tabele
- In exemplul utilizat avem doua concepte diferite din punct de vedere logic
 - produs
 - categorie de produs

Relatii in Bazele de date

- In exemplul utilizat avem doua concepte diferite din punct de vedere logic
 - produs
 - categorie de produs
- Cele doua tabele nu sunt independente
- Intre ele exista o legatura data de functionalitatea dorita pentru aplicatie: **un produs va apartine unei anumite categorii de produse**

Relatii in Bazele de date

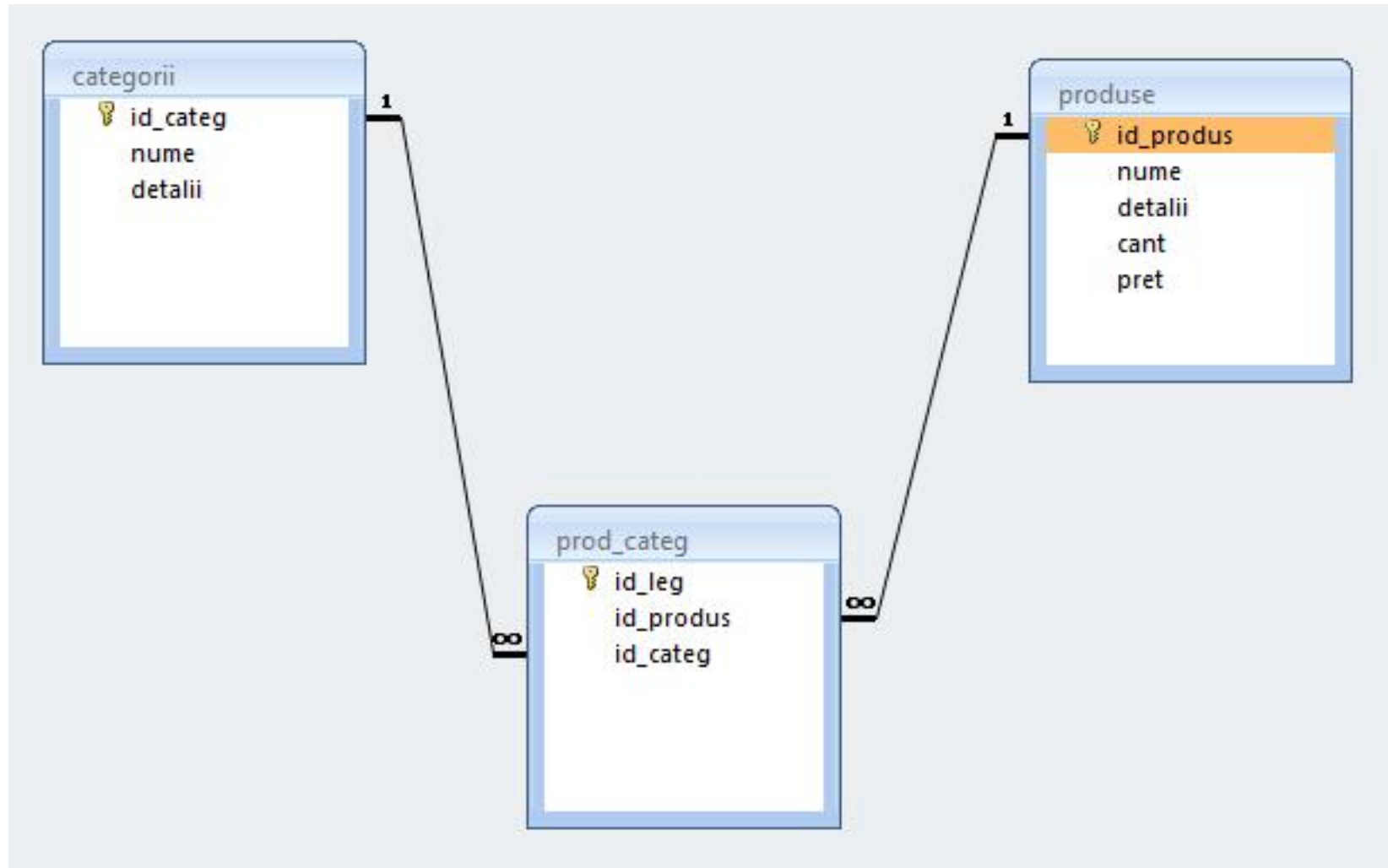
- Legaturile implementata
 - One to Many
 - in tabelul "produse" apare cheia externa (foreign key): "id_categ"



Relatii in Bazele de date

- Daca se doreste o situatie cand un produs poate apartine **mai multor categorii** (o carte cu CD poate fi inclusa si in "papetarie" si in "audio-video")
 - relatia devine de tipul **Many to Many**
 - e necesara introducerea unui tabel de legatura cu coloanele "id_leg" (cheie primara), "id_categorie" si "id_produus" (chei externe)

Relatii in Bazele de date



Relatii

- **Nu** trebuie evitate relatiile
 - Many to Many
 - One to Many
- Prelucrarea cade in sarcina server-ului de baze de date (**RDBMS**)
 - JOIN – **esential** in aplicatii cu baze de date

MySQL – eficienta

- eficienta unei aplicatii web
 - 100% - **toate prelucrarile "mutate" in RDBMS**
 - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySQL
 - 25% **alegerea corecta a tipurilor de date**
 - 25% **crearea indecsilor necesari in aplicatii**
 - 25% **normalizarea corecta a bazei de date**
 - 20% **cresterea complexitatii interogarilor pentru a "muta" prelucrarile pe server-ul de baze de date**
 - 5% **scrierea corecta a interogarilor**

Acces la server-ul MySql din PHP

Acces la server-ul MySQL din PHP

- Bibliotecile corespunzatoare trebuie activate in php.ini – vezi laboratorul 1.
 - mysql
 - mysqli (improved accesul la functionalitati ulterioare MySQL 4.1)
- O baza de date existenta poate fi accesata daca exista un utilizator cunoscut in PHP cu drepturi de acces corespunzatoare – vezi laboratorul 1.
- O baza de date poate fi creata si din PHP dar nu e metoda recomandata daca nu e necesara
 - cod dificil de implementat pentru o singura utilizare
 - necesita existenta unui utilizatori cu drepturi mai mari pentru crearea bazei de date si alocarea de drepturi unui utilizator restrans

Funcții PHP de acces MySQL

- `mysql_query`
 - trimiterea unei interogari SQL spre server
 - resource `mysql_query` (string query [, resource link_identifier])
 - rezultatul
 - SELECT, SHOW, DESCRIBE sau EXPLAIN – resursa (tabel)
 - UPDATE, DELETE, DROP, etc – true/false
- `mysql_fetch_assoc`
 - returneaza o **matrice asociativa** corespunzatoare liniei de la indexul intern (indecsi de tip sir corespunzatori denumirii coloanelor – field – din tabelul de date) si incrementeaza indexul intern sau **false** daca nu mai sunt linii
 - array `mysql_fetch_assoc` (resource result)

Functii PHP de acces MySql

Parcurgerea resurselor rezultat

- `mysql_fetch_assoc`
 - returneaza o **matrice asociativa** corespunzatoare liniei de la indexul intern (indecsi de tip sir corespunzatori denumirii coloanelor – field – din tabelul de date) si incrementeaza indexul intern sau **false** daca nu mai sunt linii
 - array `mysql_fetch_assoc` (resource result)
- `mysql_fetch_row`
 - returneaza o matrice cu indecsi intregi
 - array `mysql_fetch_row` (resource result)

Funcții PHP de acces MySQL

Parcurgerea resurselor rezultat

- `mysql_fetch_array`
 - grupează funcționalitatea `mysql_fetch_assoc` și `mysql_fetch_row`
 - array `mysql_fetch_array` (resource result [, int result_type])
 - `MYSQL_ASSOC`, `MYSQL_NUM`, `MYSQL_BOTH` (implicit)
- `mysql_data_seek`
 - muta indexul intern la valoarea indicată
 - bool `mysql_data_seek` (resource result, int row_number)

Resurse MySQL

- Resursele reprezinta o combinatie intre
 - date structurate (valori + structura) rezultate in urma unor interogari SQL
 - functii de acces la aceste date/structuri
- Analogie cu POO
 - o "clasa speciala" creata in urma interogarii cu functii predefinite de acces la datele respective

Resurse MySQL

Structura

Index intern	Col 1 (tip date)	Col 2 (tip date)
1			
2			
...			

Date

Index intern	Col 1	Col 2
1	Val 11	Val 12	...
2	Val 21	Val 22	...
...

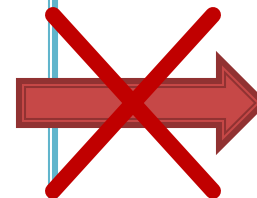
Functii de acces la structura



Functii de acces la date



~~Acces direct~~



Resurse MySQL

- Functiile de acces la structura sunt rareori utilizate
 - majoritatea aplicatiilor sunt concepute pe structura fixa, si cunosc structura datelor primite
 - exceptie: aplicatii generale, ex.: PhpMyAdmin
- Majoritatea functiilor de acces la date sunt caracterizate de acces secvential
 - se citesc in intregime valorile stocate pe o linie
 - simultan se avanseaza indexul intern pe urmatoarea pozitie, pregatindu-se urmatoarea citire

Resurse MySQL

- Functiile sunt optimizate pentru utilizarea lor intr-o structura de control **do {} while()**, sau **while() {}** de control
 - returneaza FALSE cand "s-a ajuns la capat"
- tipic se realizeaza o citire (mysql_fetch_assoc) urmata de o bucla **do {} while()**
 - pentru a se putea introduce cod de detectie probleme rulat o singura data

Exemplu de utilizare

```
$hostname = "localhost";  
$database = "world";  
$username = "web";  
$password = "ceva";  
$conex= mysql_connect($hostname, $username, $password);  
mysql_select_db($database, $conex);
```

```
$query = "SELECT `Code`, `Name`, `Population` FROM `country` AS c ";  
$result = mysql_query($ query, $conex) or die(mysql_error());  
$row_result = mysql_fetch_assoc($ result );  
$totalRows_result = mysql_num_rows($ result );
```

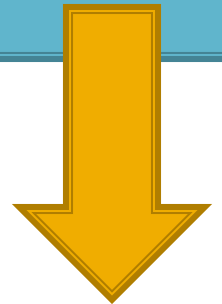
Exemplu de utilizare

```
<?php
do {?>
<tr>
    <td><?php echo $index; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Code']; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Name']; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Population']; ?>&nbsp;  </td>
</tr>
<?php
    $index++;
}
while ($ row_result = mysql_fetch_assoc($ result )); ?>
```

Modificari laborator cu date stocate text

- Codul aplicatiei ramane in mare parte acelasi
- Se modifica doar citirea valorilor pentru popularea matricii \$produse ("antet.php")

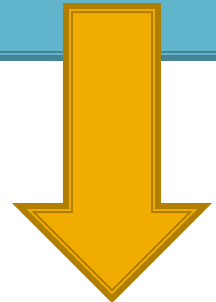
```
$matr=file("produse.txt");  
foreach ($matr as $linie)  
    {  
        $valori=explode("\t",$linie,5);  
        $produse[$valori[0]] [$valori[1]]=array ("descr" => $valori[2], "pret" => $valori[3], "cant" =>  
$valori[4]);  
    }
```



Modificari laborator cu date stocate XML

XML

```
$xml = simplexml_load_file("lista.xml");
if ($xml)
{
foreach ($xml->categorie as $categorie)
    {
    $produse[(string)$categorie["nume"]]=array();
    foreach ($categorie->produs as $prod_cur)
        {
        $produse[(string)$categorie["nume"]][(string)$prod_cur->nume]=array
("descr" => (string)$prod_cur->desc, "pret" => (string)$prod_cur->pret,
"cant" => (string)$prod_cur->cant);
        }
    }
}
```



Modificari laborator cu date stocate

MySQL

```
$hostname = "localhost";
$database = "tmpaw";
$username = "web";
$password = "test";
$conex= mysql_connect($hostname, $username, $password);
mysql_select_db($database, $conex);
$query = "SELECT * FROM `categorii` AS c";
$result_c = mysql_query($query, $conex) or die(mysql_error());
$row_result_c = mysql_fetch_assoc($result_c);
$totalRows_result = mysql_num_rows($result_c);
do {
    $query = "SELECT * FROM `produse` AS p WHERE `id_categ` = ".$row_result_c['id_categ'];
    $result_p = mysql_query($query, $conex) or die(mysql_error());
    $row_result_p = mysql_fetch_assoc($result_p);
    $totalRows_result = mysql_num_rows($result_p);
    $produse[$row_result_c['nume']] = array();
    do {
        $produse[$row_result_c['nume']][$row_result_p['nume']] = array ("descr" =>
$row_result_p['detalii'], "pret" => $row_result_p['pret'], "cant" => $row_result_p['cant']);
    }
    while ($row_result_p = mysql_fetch_assoc($result_p));
}
while ($row_result_c = mysql_fetch_assoc($result_c));
```

MySQL – eficienta

- eficienta unei aplicatii web
 - 100% - **toate prelucrarile "mutate" in RDBMS**
 - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySQL
 - 25% **alegerea corecta a tipurilor de date**
 - 25% **crearea indecsilor necesari in aplicatii**
 - 25% **normalizarea corecta a bazei de date**
 - 20% **cresterea complexitatii interogarilor pentru a "muta" prelucrarile pe server-ul de baze de date**
 - 5% **scrierea corecta a interogarilor**

Optimizare

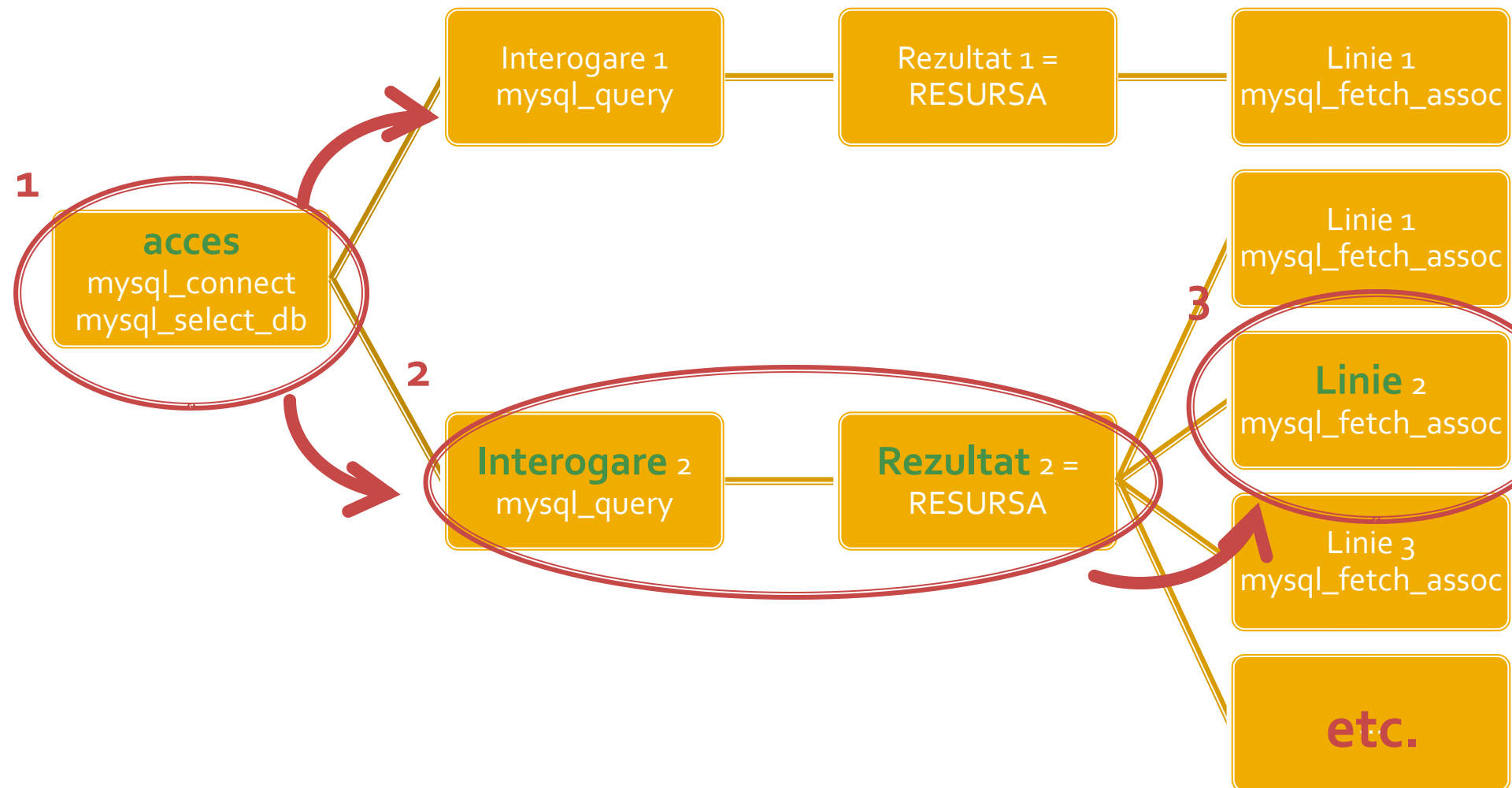
- o singura interogare SQL, unirea tabelelor lasata in baza server-ului MySQL

```
$hostname = "localhost";
$database = "tmpaw";
$username = "web";
$password = "test";
$conex= mysql_connect($hostname, $username, $password);
mysql_select_db($database, $conex);

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
        LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
$result = mysql_query($query, $conex) or die(mysql_error());
$row_result = mysql_fetch_assoc($result);
$totalRows_result = mysql_num_rows($result);

do{
    $produse[$row_result['nume_categ']][$row_result['nume']] = array ("descr" => $row_result['detalii'], "pret"
=> $row_result['pret'], "cant" => $row_result['cant']);
}
while ($row_result = mysql_fetch_assoc($result));
```

Funcții de acces la server-ul MySQL



!! IMPORTANT

PHP > 5.5

PHP 5.5

- Incapand cu versiunea 5.5 a PHP extensia mysql este declarata **depreciata**
 - orice utilizare a unei functii genereaza eroare de tip **E_DEPRECATED**
 - se preconizeaza ca in PHP > 6 aceasta extensie va fi eliminata total
- Alternativele de utilizare sunt
 - extensia mysqli (MySQL Improved)
 - extensia PDO (PHP Data Objects)

Extensia mysqli

- Inafara securitatii sporite ofera acces la facilitatile curente ale server-ului MySQL
 - accesul la interogari predefinite (Prepared Statements) (viteza, securitate)
 - server side
 - client side
 - proceduri stocate pe server (viteza, securitate)
 - interogari multiple
 - tranzactii (integritate)

Extensia mysqli

- Doua modalitati de utilizare
 - procedurala (similar mysql)
 - POO (similar PDO)
- Utilizarea procedurala (aproape) similara cu utilizarea extensiei originale mysql
 - tranzitie facila
 - tranzitie cu mici diferente de parametri

mysqli – Procedural

```
<?php
$mysqli = mysqli_connect("example.com", "user", "password", "database");
$res = mysqli_query($mysqli, "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = mysqli_fetch_assoc($res);
echo $row['_msg'];

$mysql = mysql_connect("example.com", "user", "password");
mysql_select_db("test");
$res = mysql_query("SELECT ' for new developments.' AS _msg FROM DUAL", $mysql);
$row = mysql_fetch_assoc($res);
echo $row['_msg'];
?>
```

- toate functiile mysql au un echivalent mysqli
- majoritatea functiilor au aceeasi parametri in aceeasi ordine
- sunt totusi functii cu mici diferente (Ex: **mysqli_connect**, **mysqli_query**)

mysqli – Programare orientata obiect

```
<?php
$var = new mysqli("example.com", "user", "password", "database");
$res = $var->query ($mysqli, "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = $res->fetch_assoc();
echo $row['_msg'];

$mysqli = mysqli_connect("example.com", "user", "password");
mysqli_select_db("test");
$res = mysqli_query("SELECT ' for new developments.' AS _msg FROM DUAL", $mysqli);
$row = mysqli_fetch_assoc($res);
echo $row['_msg'];
?>
```

Resurse MySQL – mysqli

Structura

Index intern	Col 1 (tip date)	Col 2 (tip date)
1			
2			
...			

Date

Index intern	Col 1	Col 2
1	Val 11	Val 12	...
2	Val 21	Val 22	...
...

Metode

Constructor	query	fetch_assoc
-------------	-------	-------------	------

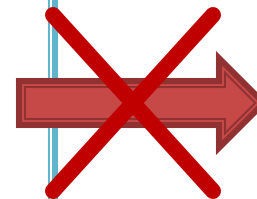
Functii de acces la structura



Functii de acces la date



Acces direct



Metode atasate resursei



Conversia la mysql (obligatorie)

■ exemplul anterior

```
$hostname = "localhost";
$database = "tmpaw";
$username = "web";
$password = "test";
$conex= mysql_connect($hostname, $username, $password);
mysql_select_db($database, $conex);

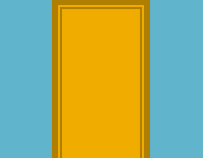

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
        LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
$result = mysql_query($query, $conex) or die(mysql_error());
$row_result = mysql_fetch_assoc($result);
$totalRows_result = mysql_num_rows($result);

do{
    $produse[$row_result['nume_categ']][$row_result['nume']]=array ("descr" => $row_result['detalii'], "pret"
=> $row_result['pret'], "cant" => $row_result['cant']);
}
while ($row_result = mysql_fetch_assoc($result));
```



mysqli (Procedural)

```
//$conex= mysql_connect($hostname, $username, $password);  
//mysql_select_db($database, $conex);  
$conex = mysqli_connect($hostname, $username, $password, $database);  
  
$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p  
        LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";  
//$result = mysql_query($query, $conex) or die(mysql_error());  
$result = mysqli_query($conex, $query);  
  
//$row_result = mysql_fetch_assoc($result);  
$row_result = mysqli_fetch_assoc($result);  
  
//$totalRows_result = mysql_num_rows($result);  
$totalRows_result = mysqli_num_rows($result);  
  
do {  
    $produse[$row_result['nume_categ']][$row_result['nume']] = array ("descr" => $row_result['detalii'], "pret"  
=> $row_result['pret'], "cant" => $row_result['cant']);  
}  
//while ($row_result = mysql_fetch_assoc($result));  
while ($row_result = mysqli_fetch_assoc($result));
```



mysqli (POO)

```
//$conex= mysql_connect($hostname, $username, $password);  
//mysql_select_db($database, $conex);  
//$conex = mysqli_connect($hostname, $username, $password, $database);  
$conex = new mysqli($hostname, $username, $password, $database);  
  
$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p  
LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";  
//$result = mysql_query($query, $conex) or die(mysql_error());  
//$result = mysqli_query($conex, $query);  
$result = $conex->query( $query );  
  
//$row_result = mysql_fetch_assoc($result);  
//$row_result = mysqli_fetch_assoc($result);  
$row_result = $result->fetch_assoc();  
  
//$totalRows_result = mysql_num_rows($result);  
//$totalRows_result = mysqli_num_rows($result);  
$totalRows_result = $result->num_rows;  
  
do {  
    $produse[$row_result['nume_categ']][$row_result['nume']] = array ("descr" => $row_result['detalii'], "pret"  
=> $row_result['pret'], "cant" => $row_result['cant']);  
}  
//while ($row_result = mysql_fetch_assoc($result));  
while ($row_result = $result->fetch_assoc(););
```

MySql

Tipuri de date

MySql – tipuri de date

- numeric
 - intregi
 - BIT (implicit 1 bit)
 - TINYINT (implicit 8 biti)
 - SMALLINT (implicit 16 biti)
 - INTEGER (implicit 32biti)
 - BIGINT (implicit 64biti)
 - real
 - FLOAT
 - DOUBLE
 - DECIMAL – fixed point

MySQL – tipuri de date

- data/timp
 - DATE ('YYYY-MM-DD')
 - '1000-01-01' pana la '9999-12-31'
 - DATETIME ('YYYY-MM-DD HH:MM:SS')
 - '1000-01-01 00:00:00' pana la '9999-12-31 23:59:59'
 - TIMESTAMP ('YYYY-MM-DD HH:MM:SS')
 - '1970-01-01 00:00:00' pana la partial 2037

MySQL – tipuri de date

- sir
 - CHAR (M)
 - sir de lungime constanta M, $M < 255$
 - VARCHAR (M)
 - sir de lungime variabila, maxim M, $M < 255$ ($M < 65535$)
- cantitati mari de date
 - TEXT
 - au alocat un set de caractere, operatiile tin cont de acesta
 - BLOB
 - sir de octeti, operatiile tin cont de valoarea numerica
 - TINYBLOB/TINYTEXT, BLOB/TEXT, MEDIUMBLOB/MEDIUMTEXT, LARGEBLOB/LARGETEXT
 - date 2^8-1 , $2^{16}-1$, $2^{24}-1$, $2^{32}-1 = 4\text{GB}$

MySQL – tipuri de date

- enumerare
 - ENUM('val₁', 'val₂', ...)
 - una singura din cele maxim 65535 valori distincte posibile
 - SET('val₁', 'val₂', ...)
 - niciuna sau mai multe din cele maxim 64 valori distincte
 - echivalent cu "setare de biti" intr-un intreg pe 64 biti cu tabela asociata

Limbas SQL

MySQL – eficienta

- eficienta unei aplicatii web
 - 100% - **toate prelucrarile "mutate" in RDBMS**
 - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySQL
 - 25% **alegerea corecta a tipurilor de date**
 - 25% **crearea indecsilor necesari in aplicatii**
 - 25% **normalizarea corecta a bazei de date**
 - 20% **cresterea complexitatii interogarilor pentru a "muta" prelucrarile pe server-ul de baze de date**
 - 5% **scrierea corecta a interogarilor**

Referinta relativa

- Referinta la elementele unei baze de date se face prin utilizarea numelui elementului respectiv daca nu exista dubii (referinta relativa)
 - daca baza de date este selectata se poate utiliza numele tabelului pentru a identifica un tabel
 - `USE db_name;`
`SELECT * FROM tbl_name;`
 - daca tabelul este identificat in instructiune se poate utiliza numele coloanei pentru a identifica coloana implicata
 - `SELECT col_name FROM tbl_name;`

Referinta absoluta

- In cazul in care apare ambiguitate in identificarea unui element se poate indica descendenta sa pâna la disparitia ambiguitatii
- Astfel, o anumita coloana, `col_name`, care apartine tabelului `tbl_name` din baza de date (schema) `db_name` poate fi identificata in functie de necesitati ca:
 - `col_name`
 - `tbl_name.col_name`
 - `db_name.tbl_name.col_name`

Nume de identificatori permise

- Numele de identificatori pot avea o lungime de reprezentare de maxim 64 octeti cu exceptia Alias care poate avea o lungime de 255 octeti
- Nu sunt permise:
 - caracterul NULL (ASCII 0x00) sau 255 (0xFF)
 - caracterul "/"
 - caracterul "\"
 - caracterul "."
- Numele nu se pot termina cu caracterul spatiu

Nume de identificatori permise

- Numele de baze de date nu pot contine decat caractere permise in numele de directoare
- Numele de tabele nu pot contine decat caractere permise in numele de fisiere
- Anumite caractere utilizate vor impune necesitatea trecerii intre apostroafe a numelui
- Apostroful utilizat pentru nume de identificatori e apostroful invers (**backtick**) “`”
 - pentru a nu aparea confuzie cu variabilele sir
 - nu necesita aparitia apostrofului caracterele alfanumerice normale, “_”, “\$”
- numele rezervate trebuie de asemenea cuprinse intre apostroafe pentru a fi utilizate

Alias

- Orice identificator poate primi un nume asociat
 - **Alias**
 - pentru a elimina ambiguitati
 - pentru a usura scrierea
 - pentru a modifica numele coloanelor in rezultate
- Definirea unui alias se face in interiorul unei interogari SQL si are efect in aceeasi interogare
 - `SELECT `t`.* FROM `tbl_name` AS t;`
 - `SELECT `t`.* FROM `tbl_name` t;`

Alias

- Desi utilizarea cuvintului cheie AS nu este obligatorie, obisnuinta utilizarii lui este recomandata, pentru a evita/identifica alocari eronate
 - `SELECT id, nume FROM produse;` ← doua coloane
 - `SELECT id nume FROM produse;` ← Alias "nume" creat pentru coloana "id"

Alias

- Usurinta scrierii
 - `SELECT * FROM un_tabel_cu_nume_lung AS t WHERE t.col1 = 5 AND t.col2 = 'ceva'`
- Modificarea numelui de coloana, sau crearea unui nume pentru o coloana calculata in rezultate
 - `SELECT CONCAT(ume, " ", prenume) AS nume_intreg FROM studenti AS s;`
 - `SELECT `n1` AS `Nume`, `n2` AS `Nota`, `n3` AS `Numar matricol` FROM elevi AS e;`

Alias

- Eliminarea ambiguitatilor
 - intalnita frecvent la relatii "many to many"
 - `SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`);`
 - tabelele c si p contin ambele coloanele "nume" si "id_categ"
 - modificarea denumirii coloanei "nume" din categorii pentru evitarea confuziei cu coloana "nume" din produse
 - eventual se pot da nume diferite coloanelor "id_categ" pentru a evita ambiguitatea in interiorul clauzei ON (desi si referinta absoluta rezolva aceasta problema)

Interrogari SQL

Interogari

- Interogariile SQL pot fi
 - Pentru definirea datelor, crearea programatica de baze de date, tabele, coloane etc.
 - mai putin utilizate in majoritatea aplicatiilor
 - ALTER, CREATE, DROP, RENAME
 - Pentru manipularea datelor
 - SELECT, INSERT, UPDATE, REPLACE etc.
 - Pentru control/administrare tranzactii/server
- De cele mai multe ori aplicatiile doar manipuleaza datele. Structura este definita in avans de asemenea si administrarea este mai facila cu programe specializate
- Urmatoarele definitii sunt cele valabile pentru **MySql 5.0**

ALTER DATABASE

- ALTER {DATABASE | SCHEMA} [db_name] alter_specification ...
 - alter_specification:
 - [DEFAULT] CHARACTER SET [=] charset_name
 - [DEFAULT] COLLATE [=] collation_name
- Modifica caracteristicile generale ale unei baze de date
- E necesar dreptul de acces (privilegiu) ALTER asupra respectivei baze de date

ALTER TABLE

- ALTER TABLE {table_option [, table_option] ... | partitioning_specification}
 - table_option:
 - ADD [COLUMN] col_name column_definition [FIRST | AFTER col_name]
 - ADD {INDEX|KEY} [index_name] [index_type] (index_col_name,...) [index_option] ...
 - ADD [CONSTRAINT [symbol]] PRIMARY KEY [index_type] (index_col_name,...) [index_option]
 - ...
 - CHANGE [COLUMN] old_col_name new_col_name column_definition [FIRST|AFTER col_name]
 - MODIFY [COLUMN] col_name column_definition [FIRST | AFTER col_name]
 - DROP [COLUMN] col_name
 - DROP PRIMARY KEY
 - DROP {INDEX|KEY} index_name
 - DISABLE KEYS
 - ENABLE KEYS
 - RENAME [TO] new_tbl_name
- permite modificarea unui tabel existent

CREATE DATABASE

- CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name [create_specification...]
 - create_specification:
 - [DEFAULT] CHARACTER SET charset_name
 - [DEFAULT] COLLATE collation_name
- Crearea unei noi baze de date
- Necesara la instalarea unei aplicatii
- Fisierile SQL "backup" contin succesiunea DROP..., CREATE... pentru a inlocui datele in intregime

CREATE INDEX

- CREATE [UNIQUE|FULLTEXT|SPATIAL] INDEX index_name [USING index_type] ON tbl_name (index_col_name,...)
 - index_col_name:
 - col_name [(length)] [ASC | DESC]
- Crearea unui index se face de obicei la crearea tabelului
- Interogarea CREATE INDEX ... se transpune in interogare ALTER TABLE ...

CREATE TABLE

- CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [(create_definition,...)] [table_options] [select_statement]
- CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [() LIKE old_tbl_name ()]
- Interogarea de creare a tabelului este memorata intern de server-ul MySql pentru utilizari ulterioare (in general in ALTER TABLE sa fie cunoscute specificatiile initiale)

CREATE TABLE

- create_definition – coloana impreuna cu eventualele caracteristici (in special chei - indecsi):
 - column_definition
 - | [CONSTRAINT [symbol]] PRIMARY KEY [index_type] (index_col_name,...)
 - | KEY [index_name] [index_type] (index_col_name,...)
 - | INDEX [index_name] [index_type] (index_col_name,...)
 - | [CONSTRAINT [symbol]] UNIQUE [INDEX] [index_name] [index_type] (index_col_name,...)
 - | [FULLTEXT|SPATIAL] [INDEX] [index_name] (index_col_name,...)
 - | [CONSTRAINT [symbol]] FOREIGN KEY [index_name] (index_col_name,...) [reference_definition]
 - | CHECK (expr)
- column_definition – nume si tipul de date (curs 8):
 - col_name type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY] [COMMENT 'string'] [reference_definition]

CREATE TABLE

- Exemple
 - CREATE TABLE test (a INT NOT NULL AUTO_INCREMENT, PRIMARY KEY (a), KEY(b)) SELECT b,c FROM test2;
 - CREATE TABLE IF NOT EXISTS `schema`.`Employee` (
`idEmployee` VARCHAR(45) NOT NULL,
`Name` VARCHAR(255) NULL,
`idAddresses` VARCHAR(45) NULL,
PRIMARY KEY (`idEmployee`),
CONSTRAINT `fkEmployee_Addresses`
FOREIGN KEY `fkEmployee_Addresses` (`idAddresses`)
REFERENCES `schema`.`Addresses` (`idAddresses`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_bin

CREATE TABLE

- `CREATE ... LIKE ...` creaza un tabel fara date pe baza modelului unui tabel existent. Se pastreaza definitiile coloanelor si eventualele chei (index) definite in tabelul anterior
- `CREATE ... SELECT ...` creaza un tabel cu date pe baza modelului si datelor obtinute dintr-un alt tabel existent. Sunt obtinute anumite coloane (`SELECT`) cu tipul lor, dar fara crearea indecsilor
- `CREATE TEMPORARY TABLE` creaza un tabel temporar. Utilizat in cazul interogarilor complexe sau cu numar mare de rezultate

DROP

- `DROP {DATABASE | SCHEMA} [IF EXISTS]`
`db_name`
- `DROP INDEX index_name ON tbl_name`
- `DROP [TEMPORARY] TABLE [IF EXISTS]`
`tbl_name [, tbl_name] ...`
- Trebuie utilizate cu foarte mare atentie aceste interogari, stergerea datelor este ireversibila
- Fisierile SQL "backup" contin succesiunea `DROP...`, `CREATE...` pentru a inlocui datele in intregime

Interrogari SQL

Interogari

- Interogariile SQL pot fi
 - Pentru definirea datelor, crearea programatica de baze de date, tabele, coloane etc.
 - mai putin utilizate in majoritatea aplicatiilor
 - ALTER, CREATE, DROP, RENAME
 - **Pentru manipularea datelor**
 - SELECT, INSERT, UPDATE, REPLACE, DELETE etc.
 - Pentru control/administrare tranzactii/server
- De cele mai multe ori aplicatiile doar manipuleaza datele. Structura este definita in avans de asemenea si administrarea este mai facila cu programe specializate
- Urmatoarele definitii sunt cele valabile pentru **MySql 5.0**

DELETE

- `DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
FROM table_name [WHERE where_condition]
[ORDER BY ...] [LIMIT row_count]`
- Sterge linii din tabelul mentionat si returneaza
numarul de linii sterse
- `[LOW_PRIORITY] [QUICK] [IGNORE]` sunt
optiuni care instruiesc server-ul sa reactioneze
diferit de varianta standard
- Exemplu:
 - `DELETE FROM somelog WHERE user = 'jcole'
ORDER BY timestamp_column LIMIT 1;`

DELETE

- [WHERE where_condition] – folosit pentru a selecta liniile care trebuie sterse
 - In absenta conditiei se sterg **toate liniile** din tabel
- [LIMIT row_count] sterge numai *row_count* linii dupa care se opreste
 - In general pentru a limita ocuparea server-ului (recrearea indecsilor se face “on the fly”)
 - Operatia se poate repeta pana valoarea returnata e mai mica decat row_count
- [ORDER BY ...] precizeaza ordinea in care se sterg liniile identificate prin conditie

INSERT

- INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE] [INTO] tbl_name [(col_name,...)] VALUES ({expr | DEFAULT},...),(...),... [ON DUPLICATE KEY UPDATE col_name=expr, ...]
- INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE] [INTO] tbl_name SET col_name={expr | DEFAULT}, ... [ON DUPLICATE KEY UPDATE col_name=expr, ...]
- INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE] [INTO] tbl_name [(col_name,...)] SELECT ... [ON DUPLICATE KEY UPDATE col_name=expr, ...]

INSERT

- Introduce linii noi intr-un tabel
- Primele doua forme introduc valori exprimate explicit
 - INSERT ... VALUES ...
 - INSERT ... SET ...
- INSERT ... SELECT ... introduce valori rezultate obtinute printr-o interogare SQL
- DELAYED – interogarea primeste raspuns de la server imediat, dar inserarea datelor se face efectiv cand tabelul implicat nu este folosit
 - valabil pentru metodele de stocare MyISAM, Memory, Archive

INSERT

- Exemple
 - `INSERT INTO tbl_name (a,b,c) VALUES (1,2,3), (4,5,6), (7,8,9);`
 - `INSERT INTO tbl_name (col1,col2) VALUES (15,col1*2);`
 - `INSERT INTO table1 (field1,field3,field9) SELECT field3,field1,field4 FROM table2;`

INSERT

- INSERT ... ON DUPLICATE KEY UPDATE ...
- Daca inserarea unei noi linii ar conduce la duplicarea unei chei primare sau unice, in loc sa se introduca o noua linie se modifica linia anterioara
- Exemple
 - INSERT INTO table (a,b,c) VALUES (1,2,3) ON DUPLICATE KEY UPDATE c=c+1;
 - INSERT INTO table (a,b,c) VALUES (1,2,3),(4,5,6) ON DUPLICATE KEY UPDATE c=VALUES(a)+VALUES(b);

REPLACE

- REPLACE [LOW_PRIORITY | DELAYED] [INTO] tbl_name [(col_name,...)] VALUES ({expr | DEFAULT},...),(...),...
- REPLACE [LOW_PRIORITY | DELAYED] [INTO] tbl_name SET col_name={expr | DEFAULT}, ...
- REPLACE [LOW_PRIORITY | DELAYED] [INTO] tbl_name [(col_name,...)] SELECT ...
- REPLACE functioneaza similar cu INSERT
 - daca noua linie nu realizeaza duplicarea unei chei primare sau unice se realizeaza insertie
 - daca noua linie realizeaza duplicarea unei chei primare sau unice se sterge linia anterioara dupa care se insereaza noua linie
- REPLACE e extensie MySql a limbajului SQL standard

UPDATE

- UPDATE [LOW_PRIORITY] [IGNORE] tbl_name SET col_name1=expr1 [, col_name2=expr2 ...] [WHERE where_condition] [ORDER BY ...] [LIMIT row_count]
- Modificarea valorilor stocate intr-o linie
- Exemple
 - UPDATE persondata SET age=15 WHERE id=6;
 - UPDATE persondata SET age=age+1;

SELECT

- SELECT [ALL | DISTINCT | DISTINCTROW]
[HIGH_PRIORITY] [STRAIGHT_JOIN]
select_expr, ... [FROM table_references
 - [WHERE where_condition]
 - [GROUP BY {col_name | expr | position} [ASC | DESC],
... [WITH ROLLUP]]
 - [HAVING where_condition]
 - [ORDER BY {col_name | expr | position} [ASC | DESC],
...]
 - [LIMIT {[offset,] row_count | row_count OFFSET
offset}]
-]

SELECT

- SELECT este **cea mai importanta** interogare SQL.
- Intelegerea setarilor si utilizarea inteligenta a indecsilor stau la baza eficientei unei aplicatii
- E absolut necesara realizarea interogarii in asa fel incat datele returnate sa fie exact cele dorite (prelucrarea sa se realizeze pe server-ul MySql)

SELECT

- `select_expr`: macar o expresie selectata trebuie sa apara
 - identifica ceea ce trebuie extras ca valori de iesire din baza de date
 - pot fi nume de coloana(e)
 - pot fi date de sinteza (rezultate din utilizarea unor functii MySql) – necesara atribuirea unui Alias
 - `SELECT CONCAT(last_name,', ',first_name) AS full_name FROM mytable ORDER BY full_name;`

SELECT

- WHERE where_condition, HAVING where_condition sunt utilizate pentru a introduce criterii de selectie
 - in general au comportare similara si sunt interschimbabile
 - WHERE accepta orice operatori mai putin functii aggregate – de “sumare” (COUNT, MAX)
 - HAVING accepta functii aggregate, dar se aplica la sfarsit, exact inainte de a fi trimise datele clientului, **fara nici o optimizare** – utilizarea este recomandata doar cand nu exista echivalent WHERE

SELECT

- ORDER BY {col_name | expr | position} [ASC | DESC]
 - ordoneaza datele returnate dupa anumite criterii (valoarea unei anumite coloane sau functii).
 - Implicit ordonarea este crescatoare ASC, dar se poate specifica ordine descrescatoare DESC
- GROUP BY {col_name | expr | position}
 - realizeaza gruparea liniilor returnate dupa anumite criterii
 - permite utilizarea functiilor agregate (de sumare)

SELECT

- GROUP BY – functii aggregate
 - AVG(expresie) – mediere valorilor
 - SELECT student_name, AVG(test_score) FROM student GROUP BY student_name;
 - COUNT(expresie), COUNT(*)
 - SELECT COUNT(*) FROM student;
 - SELECT COUNT(DISTINCT results) FROM student;
 - SELECT student.student_name, COUNT(*) FROM student, course WHERE student.student_id=course.student_id GROUP BY student_name;
 - SELECT columnname, COUNT(columnname) FROM tablename GROUP BY columnname HAVING COUNT(columnname)>1
- Cuvantul cheie DISTINCT este utilizat pentru a procesa doar liniile cu valori diferite
 - exemplu: 100 de note (rezultate) la examen
 - COUNT(results) va oferi raspunsul 100
 - COUNT(DISTINCT results) va oferi raspunsul 7 (notele diferite 4,5,6,7,8,9,10)

SELECT

- GROUP BY – functii aggregate
 - MIN(expresie), MAX(expresie) – minim si maxim
 - SELECT student_name, MIN(test_score), MAX(test_score) FROM student GROUP BY student_name;
 - SUM(expresie) – sumarea valorilor
 - SELECT year, SUM(profit) FROM sales GROUP BY year;
- WITH ROLLUP – operatii de sumare super-aggregate (un nivel suplimentar de agregare)

SELECT ... WITH ROLLUP

- `SELECT year, SUM(profit) FROM sales GROUP BY year;`
- `SELECT year, SUM(profit) FROM sales GROUP BY year WITH ROLLUP;`
 - se obtine un total general, linia "super-aggregate" este identificata dupa valoarea NULL a coloanei dupa care se face sumarea

year	SUM(profit)
2000	4525
2001	3010

year	SUM(profit)
2000	4525
2001	3010
NULL	7535

SELECT

- LIMIT [offset,] row_count | row_count
 - se limiteaza numarul de linii returnate
 - utilizat frecvent in aplicatiile web
 - LIMIT 15 – returneaza doar primele 15 linii (1÷15)
 - LIMIT 10,15 – returneaza 15 linii dupa primele 10 linii (11÷25)

JOIN

- Normalizarea si existenta relatiilor intre diversele tabele ale unei baze de date implica faptul ca pentru aflarea unor informatii utilizabile (complete), acestea trebuie extrase **simultan** din mai multe tabele
 - informatie inutilizabila: studentul cu id-ul 253 a luat nota 8 la examenul cu id-ul 35
- Uneori asamblarea informatiilor din mai multe tabele e necesara pentru obtinerea unor rapoarte complexe
 - Exemplu: tabel cu clienti, tabel cu comenzi, tabel cu produse; legatura produse-comenzi e implementata printr-un tabel suplimentar. Raspunsul la intrebarea cate produse x a cumparat clientul y cere tratarea unitara a celor 4 tabele implicate

JOIN

- In general in SQL se poate descrie o astfel de unificare de date intre doua tabele:
 - left_table JOIN_type right_table criteriu_unificare
- JOIN_type
 - JOIN – selecteaza toate liniile compuse in care criteriul este indeplinit pentru ambele tabele
 - LEFT JOIN – compune si selecteaza toate liniile din left_table chiar daca nu este gasit un corespondent in right_table
 - RIGHT JOIN – compune si selecteaza toate liniile din right table (similar)
 - FULL JOIN – compune si selecteaza toate liniile din left_table si right_table fie ca este indeplinit criteriul fie ca nu (nu este implementat in MySql, poate fi simulat)

JOIN

- Clauza JOIN e utilizata pentru a realiza o unificare temporara, dupa anumite criterii, din punct de vedere logic, a doua tabele in vederea extragerii informatiei "suma" dorite
 - left_table [INNER | CROSS] JOIN right_table [join_condition]
 - left_table STRAIGHT_JOIN right_table
 - left_table STRAIGHT_JOIN right_table ON condition
 - left_table LEFT [OUTER] JOIN right_table join_condition
 - left_table NATURAL [LEFT [OUTER]] JOIN right_table
 - left_table RIGHT [OUTER] JOIN right_table join_condition
 - left_table NATURAL [RIGHT [OUTER]] JOIN right_table
 - join_condition: ON conditional_expr | USING (column_list)

JOIN – Exemplu

- Tabel clienti
 - 4 clienti
- Tabel comenzi
 - client 1 – 2 comenzi
 - client 2 – 0 comenzi
 - client 3,4 – 1 comanda

```
CREATE TABLE `clienti` (  
  `id_client` int(10) unsigned NOT NULL auto_increment,  
  `nume` varchar(100) NOT NULL,  
  PRIMARY KEY (`id_client`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `clienti` (`id_client`,`nume`)VALUES  
(1,'Ionescu'),  
(2,'Popescu'),  
(3,'Vasilescu'),  
(4,'Georgescu');
```

```
CREATE TABLE `comenzi` (  
  `id_comanda` int(10) unsigned NOT NULL auto_increment,  
  `id_client` int(10) unsigned NOT NULL,  
  `suma` double NOT NULL,  
  PRIMARY KEY (`id_comanda`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
INSERT INTO `comenzi` (`id_comanda`,`id_client`,`suma`)VALUES  
(1,1,19.99),  
(2,1,35.15),  
(3,3,17.56),  
(4,4,12.34);
```

INNER JOIN

- INNER JOIN sunt unificarile implicite, in care criteriul (join_condition) trebuie indeplinit in ambele tabele (extensie a cuvintului cheie JOIN pentru evitarea ambiguitatii)
 - OUTER JOIN = {LEFT JOIN | RIGHT JOIN | FULL JOIN} – nu e obligatoriu sa fie indeplinit criteriul in ambele tabele
 - FULL JOIN nu e implementat in MySql, poate fi simulat ca UNION intre LEFT JOIN si RIGHT JOIN
- INNER JOIN sunt echivalente cu realizarea produsului cartezian intre cele doua tabele implicate urmata de verificarea criteriului, daca acesta exista

CROSS JOIN

- In MySql INNER JOIN si CROSS JOIN sunt echivalente in totalitate
 - In SQL standard INNER este folosit in prezenta unui criteriu, CROSS in absenta sa
- INNER (CROSS) JOIN si “,” sunt echivalente cu produsul cartezian intre cele doua tabele implicate in conditiile lipsei criteriului de selectie: fiecare linie a unui tabel este alaturata fiecarei linii din al doilea tabel
 - (un tabel cu M linii si A coloane) CROSS JOIN (un tabel cu N linii si B coloane) → (un tabel cu MxN linii si A+B coloane)

CROSS JOIN

SQL Query Area

```
1 SELECT * FROM clienti JOIN comenzi;  
2 SELECT * FROM clienti, comenzi;  
3 SELECT * FROM clienti INNER JOIN comenzi;  
4 SELECT * FROM clienti CROSS JOIN comenzi;
```

id_client	nume	id_comanda	id_client	suma
1	Ionescu	1	1	19.99
2	Popescu	1	1	19.99
3	Vasilescu	1	1	19.99
4	Georgescu	1	1	19.99
1	Ionescu	2	1	35.15
2	Popescu	2	1	35.15
3	Vasilescu	2	1	35.15
4	Georgescu	2	1	35.15
1	Ionescu	3	3	17.56
2	Popescu	3	3	17.56
3	Vasilescu	3	3	17.56
4	Georgescu	3	3	17.56
1	Ionescu	4	4	12.34
2	Popescu	4	4	12.34
3	Vasilescu	4	4	12.34
4	Georgescu	4	4	12.34

INNER JOIN – criterii

- USING – trebuie sa aiba o coloana cu nume identic in cele doua tabele
 - coloana comuna este afisata o singura data
- ON – accepta orice conditie conditionala
 - chiar daca numele coloanelor din conditie sunt identice, sunt tratate ca entitati diferite (id_client apare de doua ori provenind din cele doua tabele)

SQL Query Area				
1	<code>SELECT * FROM clienti INNER JOIN comenzi USING (id_client);</code>			
id_client	nume	id_comanda	suma	
1	Ionescu	1	19.99	
1	Ionescu	2	35.15	
3	Vasilescu	3	17.56	
4	Georgescu	4	12.34	
1	<code>SELECT * FROM clienti INNER JOIN comenzi ON (clienti.id_client=comenzi.id_client);</code>			
id_client	nume	id_comanda	id_client	suma
1	Ionescu	1	1	19.99
1	Ionescu	2	1	35.15
3	Vasilescu	3	3	17.56
4	Georgescu	4	4	12.34

NATURAL JOIN

- NATURAL JOIN e echivalent cu o unificare INNER JOIN cu o clauza USING(...) care utilizeaza toate coloanele cu nume comun intre cele doua tabele

SQL Query Area			
1	<code>SELECT * FROM clienti NATURAL JOIN comenzi;</code>		
id_client	nume	id_comanda	suma
1	Ionescu	1	19.99
1	Ionescu	2	35.15
3	Vasilescu	3	17.56
4	Georgescu	4	12.34

LEFT JOIN

- Unificare de tip OUTER JOIN
- Se returneaza linia din left_table chiar daca nu exista corespondent in right_table (se introduc valori NULL)
- Cuvantul cheie OUTER este optional

SQL Query Area			
1 SELECT * FROM clienti LEFT OUTER JOIN comenzi USING(id_client);			
id_client	nume	id_comanda	suma
1	Ionescu	1	19.99
1	Ionescu	2	35.15
2	Popescu	NULL	NULL
3	Vasilescu	3	17.56
4	Georgescu	4	12.34

RIGHT JOIN

- Unificare de tip OUTER JOIN
- Se returneaza linia din right_table chiar daca nu exista corespondent in left_table
- Echivalent cu LEFT JOIN cu tabelele scrise in ordine inversa

SQL Query Area				
1 SELECT * FROM clienti RIGHT OUTER JOIN comenzi USING(id_client);				
id_client	id_comanda	suma	nume	
1	1	19.99	Ionescu	
1	2	35.15	Ionescu	
3	3	17.56	Vasilescu	
4	4	12.34	Georgescu	

SQL Query Area				
1 SELECT * FROM comenzi RIGHT OUTER JOIN clienti USING(id_client);				
id_client	nume	id_comanda	suma	
1	Ionescu	1	19.99	
1	Ionescu	2	35.15	
2	Popescu	NULL	NULL	
3	Vasilescu	3	17.56	
4	Georgescu	4	12.34	

JOIN

- `STRAIGHT_JOIN` – forteaza citirea mai intai a valorilor din `left_table` si apoi a celor din `right_table` (in anumite cazuri citirea se realizeaza invers)
- `USE_INDEX`, `IGNORE_INDEX`, `FORCE_INDEX` controlul index-ului utilizat pentru gasirea si selectia liniilor, poate aduce spor de viteza

UNION

- Combina rezultatele mai multor interogari SELECT intr-un singur rezultat general
- SELECT ... UNION [ALL | DISTINCT]
SELECT ... [UNION [ALL | DISTINCT]
SELECT ...]
- Poate fi folosit pentru a realiza FULL JOIN

```
SQL Query Area
1 SELECT * FROM comenzi LEFT JOIN clienti ON (comenzi.id_client=clienti.id_client)
2 UNION
3 SELECT * FROM comenzi RIGHT JOIN clienti ON (comenzi.id_client=clienti.id_client)
4 WHERE comenzi.id_client IS NULL
```

id_comanda	id_client	suma	id_client	nume
1	1	19.99	1	Ionescu
2	1	35.15	1	Ionescu
3	3	17.56	3	Vasilescu
4	4	12.34	4	Georgescu
NULL	NULL	NULL	2	Popescu

Subquery

- O “subinterogare” este o interogare de tip SELECT utilizata ca operand intr-o alta interogare
- O “subinterogare” poate fi privit ca un tabel temporar si tratat ca atare (inclusiv cu JOIN) eventual cu atribuire de nume (Alias) daca este nevoie
- Exemple
 - `SELECT * FROM t1 WHERE column1 = (SELECT column1 FROM t2);`

Subquery

- Subquery – un instrument foarte puternic
- permite selectii in doua sau mai multe etape
 - o prima selectie **dupa un criteriu**
 - urmata de o doua selectie **dupa un alt criteriu** in **rezultatele primei selectii**
 - ... samd
- Exista restrictii asupra tabelelor implicate pentru evitarea prelucrarilor recursive (bucle potential infinite)
 - ex: UPDATE tabel₁ SET ... SELECT ... FROM tabel₁ nu este permis

Subquery

- Subquery – un instrument foarte puternic
- Permite evitarea multor prelucrari PHP si trimiterea lor spre server-ul MySql
 - `INSERT INTO tabel1 ... SELECT ... FROM tabel2` permite inserarea printr-o singura interogare a mai multor linii in tabel1 (in functie de numarul de linii rezultate din tabel2)

Laborator 2 / 2011-2012

- Se recomanda aplicarea exercitiilor din laboratorul 2 / 2011-2012, pentru exemple de interogari, JOIN, subquery, JOIN cu subquery

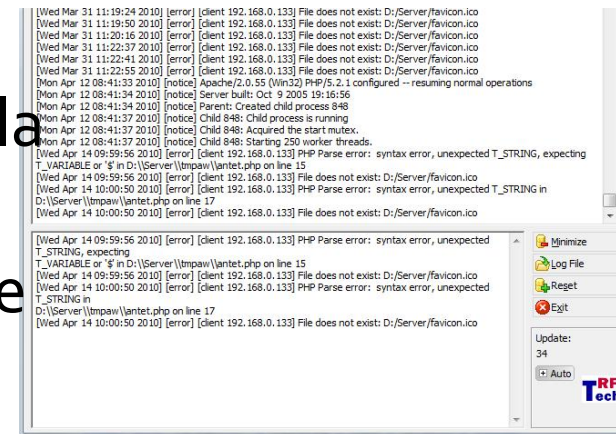
Aspecte practice recomandate in realizarea aplicatiilor web

Metode de lucru recomandate 1

- Daca nu aveti acces simplu la “log-urile” server-ului MySql puteti vedea cum ajung efectiv interogariile la el afisand temporar textul interogarii
 - `$query = "SELECT * FROM `produse` AS p WHERE `id_categ` = ".$row_result_c['id_categ'];
echo $query; //util in perioada de testare`
 - Textul prelucrat de PHP al interogarii va fi afisat in clar pe pagina facand mai usoara depanarea programului
 - Aceste linii **trebuie** eliminate in forma finala a programului ca masura de securitate

Metode de lucru recomandate 2

- Verificarea “log-ului” de erori al server-ului Apache ramane principala metoda de depanare a codului PHP.
- W2000: Utilizarea aplicatiei prezentata la laborator este mai comoda datorita automatizarii dar orice alta varianta este utila
- Centos 7.1:
 - putty → nano /var/log/httpd/error_log
 - <http://192.168.30.5/logfile.php> (nonstandard)

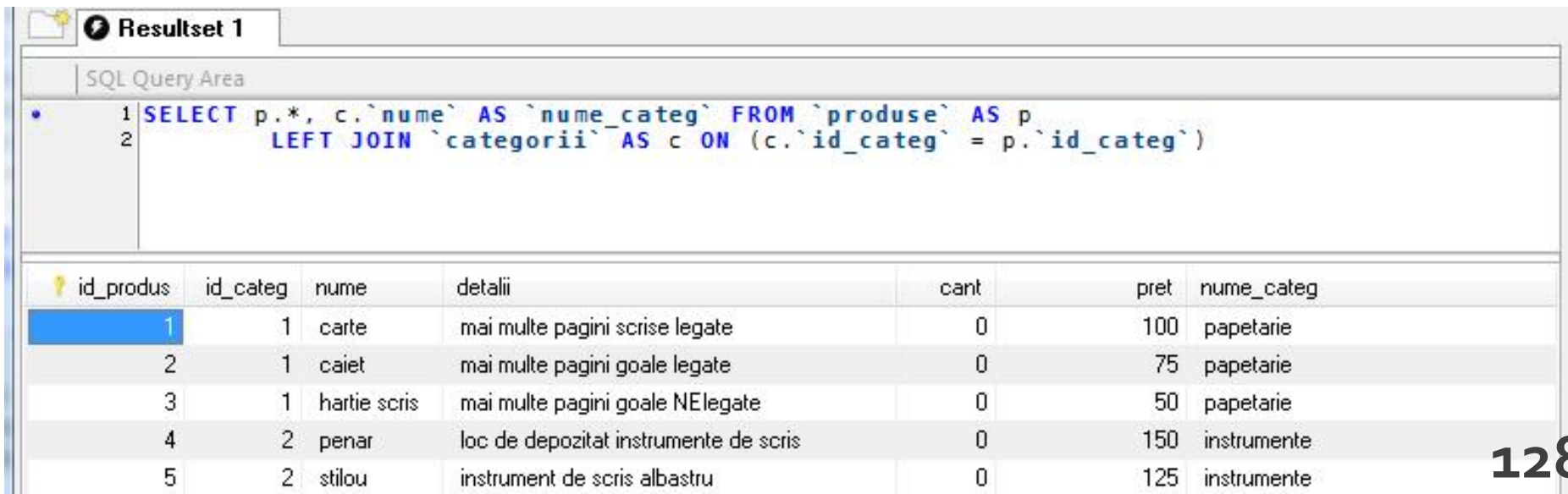


```
[Wed Mar 31 11:19:24 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:19:50 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:20:16 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:22:37 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:22:41 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:22:55 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Mon Apr 12 08:41:33 2010] [notice] Apache/2.0.55 (Win32) PHP/5.2.1 configured -- resuming normal operations
[Mon Apr 12 08:41:34 2010] [notice] Server built: Oct 9 2005 19:16:56
[Mon Apr 12 08:41:34 2010] [notice] Parent: Created child process 848
[Mon Apr 12 08:41:37 2010] [notice] Child 848: Child process is running
[Mon Apr 12 08:41:37 2010] [notice] Child 848: Acquired the start mutex.
[Mon Apr 12 08:41:37 2010] [notice] Child 848: Starting 250 worker threads.
[Wed Apr 14 09:59:56 2010] [error] [client 192.168.0.133] PHP Parse error: syntax error, unexpected T_STRING, expecting
T_VARIABLE or '$' in D:\Server\lmpaw\lntet.php on line 15
[Wed Apr 14 10:00:50 2010] [error] [client 192.168.0.133] PHP Parse error: syntax error, unexpected T_STRING in
D:\Server\lmpaw\lntet.php on line 17
[Wed Apr 14 10:00:50 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico

[Wed Apr 14 09:59:56 2010] [error] [client 192.168.0.133] PHP Parse error: syntax error, unexpected
T_STRING, expecting
T_VARIABLE or '$' in D:\Server\lmpaw\lntet.php on line 15
[Wed Apr 14 09:59:56 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Apr 14 10:00:50 2010] [error] [client 192.168.0.133] PHP Parse error: syntax error, unexpected
T_STRING in
D:\Server\lmpaw\lntet.php on line 17
[Wed Apr 14 10:00:50 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
```

Metode de lucru recomandate 3

- In perioada de definitivare a formei interogarilor MySql este de multe ori benefic sa se utilizeze mai intai **MySql Query Browser/PhpMyAdmin** pentru incercarea interogarilor, urmand ca apoi, cand sunteti multumiti de rezultat, sa transferati interogarea SQL in codul PHP



The screenshot shows a MySQL Query Browser window titled "Resultset 1". The "SQL Query Area" contains the following query:

```
1 SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
2 LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)
```

The results are displayed in a table with the following columns: id_produș, id_categ, nume, detalii, cant, pret, and nume_categ. The first row is highlighted in blue.

id_produș	id_categ	nume	detalii	cant	pret	nume_categ
1	1	carte	mai multe pagini scrise legate	0	100	papetarie
2	1	caiet	mai multe pagini goale legate	0	75	papetarie
3	1	hartie scris	mai multe pagini goale NElegate	0	50	papetarie
4	2	penar	loc de depozitat instrumente de scris	0	150	instrumente
5	2	stilou	instrument de scris albastru	0	125	instrumente

Metode de lucru recomandate 3

MySQL Query Browser - Connection: root@server / tmpaw

File Edit View Query Script Tools Window Help

Transaction Explain Compare

Resultset 1

SQL Query Area

```
1 SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
2 LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)
```

id_produc	id_categ	nume	detalii	cant	pret	nume_categ
1	1	carte	mai multe pagini scrise legate	0	100	papetarie
2	1	caiet	mai multe pagini goale legate	0	75	papetarie
3	1	hartie scris	mai multe pagini goale NElegate	0	50	papetarie
4	2	penar	loc de depozitat instrumente de scris	0	150	instrumente
5	2	stilou	instrument de scris albastru	0	125	instrumente
6	2	creion	instrument de scris gri	0	25	instrumente
7	3	cd	canta	0	50	audio-video
8	3	dvd	vizual	0	100	audio-video
9	3	blue ray	vizual extrem	0	500	audio-video

9 rows fetched in 0.0035s (0.0016s)

Edit Apply Changes Discard Changes First Last Search

1: 1

Metode de lucru recomandate 4

- eficienta unei aplicatii web
 - 100% - **toate prelucrarile "mutate" in RDBMS**
 - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySql
 - 25% **alegerea corecta a tipurilor de date**
 - 25% **crearea indecsilor necesari in aplicatii**
 - 25% **normalizarea corecta a bazei de date**
 - 20% **cresterea complexitatii interogarilor pentru a "muta" prelucrarile pe server-ul de baze de date**
 - 5% **scrierea corecta a interogarilor**

Metode de lucru recomandate 5

- La implementarea unei aplicatii noi (proiect)
 1. Imaginarea planului aplicatiei (ex: S14-S15)
 - "cum as vrea eu sa lucrez cu o astfel de aplicatie"
 - hartie/creion/timp – esentiale
 2. Identificarea datelor/transmisia de date intre pagini
 - get/post/fisier unic colectare-prelucrare
 - baza de date read/write
 3. Identificarea structurii logice a datelor utilizate
 - "clase" de obiecte/fenomene tratate identic
 - se are in vedere scalabilitatea (posibilitatea de crestere a numarului de elemente dintr-o clasa)

Metode de lucru recomandate 5

- La implementarea unei aplicatii noi (proiect)
 4. Realizarea structurii bazei de date
 - In general un tabel pentru fiecare clasa logica distincta **DAR...**
 - se are in vedere scalabilitatea (daca aplicatia creste sa **NU** apara cresterea numarului de clase/tabele) **SI...**
 - normalizare
 5. Identificarea tipului de date necesar pentru coloane
 - de preferat numerele intregi in orice situatie care presupune ordonare
 - dimensiunea campurilor nu mai mare decat e necesar (poate fi fortata prin atributul "size" in eticheta HTML "input")
 6. Imaginarea formei fizice a paginilor
 - "am mai vazut asa si mi-a placut" (Don't make me think!)
 - investigarea posibilitatii de a introduce functionalitate template

Metode de lucru recomandate 5

- La implementarea unei aplicatii noi (proiect)
 7. Popularea manuala a bazei de date cu date initiale
 - MySql Query Browser (sau PhpMyAdmin) / automat / imprumut
 - programarea individuala a paginilor are nevoie de prezenta unor date
 8. Programare individuala a paginilor
 - In general in ordinea din planul aplicatiei (de multe ori o pagina asigura datele necesare pentru urmatoarea din plan)
 - modul "verbose" activ pentru PHP (adica: `echo $a; print_r($matr)`)
 9. Pregatirea pentru distributie/mutare
 - testare detaliata (eventual un "cobai")
 - eliminarea adaosurilor "verbose"
 - backup
 - generarea unui eventual install/setup

MySQL – eficienta

- eficienta unei aplicatii web
 - 100% - **toate prelucrarile "mutate" in RDBMS**
 - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySQL
 - 25% **alegerea corecta a tipurilor de date**
 - 25% **crearea indecsilor necesari in aplicatii**
 - 25% **normalizarea corecta a bazei de date**
 - 20% **cresterea complexitatii interogarilor pentru a "muta" prelucrarile pe server-ul de baze de date**
 - 5% **scrierea corecta a interogarilor**

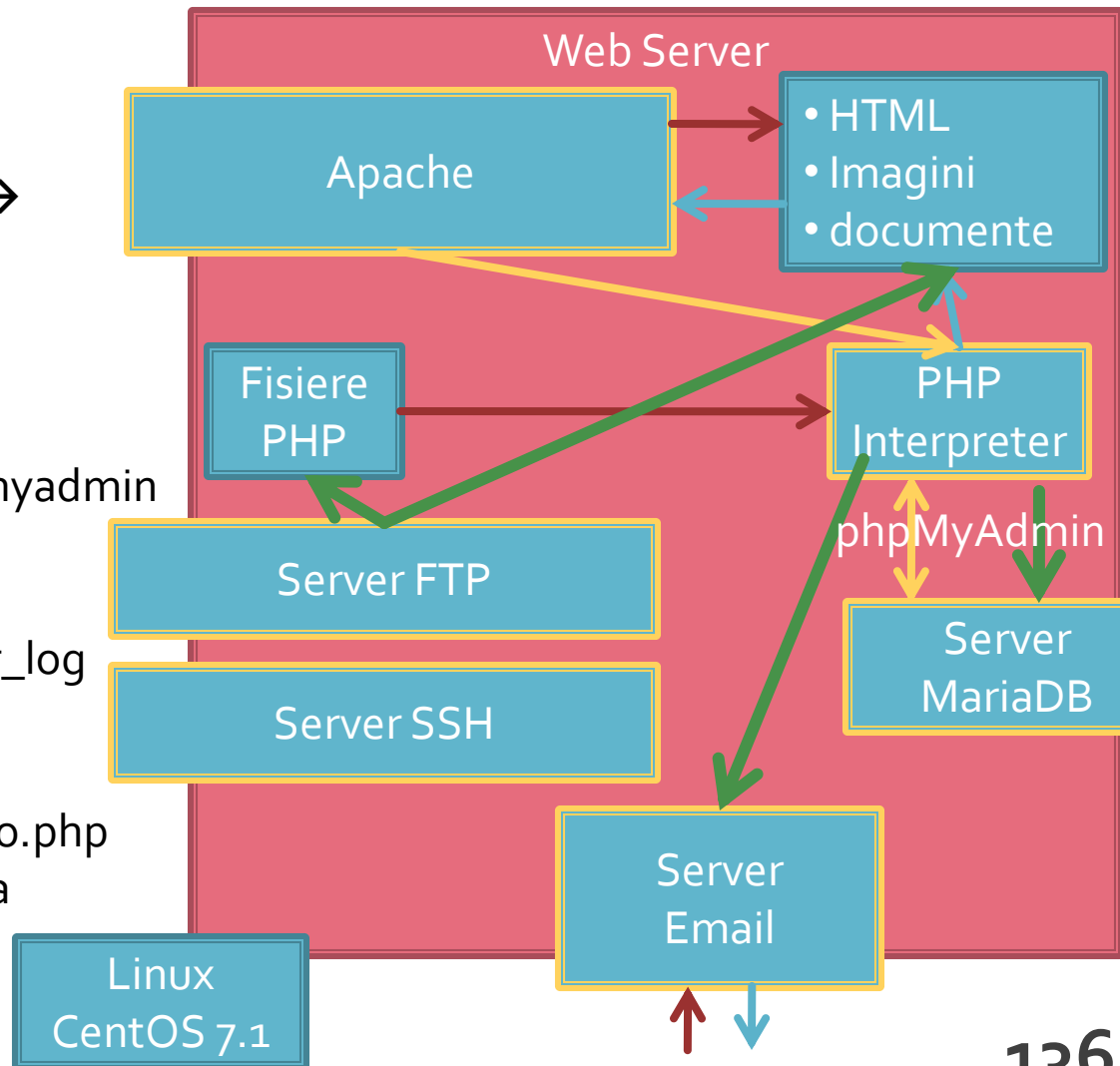
MySql – Server Centos 7.1

Mini – Indrumar practic

Lucru cu bazele de date

Utilizare LAMP

1. login → root:masterrc
2. ifconfig → 192.168.30.5
3. putty.exe → 192.168.30.5 → SSH → root:masterrc (remote login)
4. [alte comenzi linux dorite]
5. FTP → Winscp → SFTP → student:masterrc@192.168.30.5
6. MySql → http://192.168.30.5/phpmyadmin → root:masterrc
7. Apache Error Log →
 - 7a. putty → nano /var/log/httpd/error_log
 - 7b. http://192.168.30.5/logfile.php (nonstandard)
8. PHP info → http://192.168.30.5/info.php
9. daca serviciul DHCP duce la oprirea Apache: `service httpd restart`



PhpMyAdmin

- <http://192.168.30.5/phpmyadmin>
 - root
 - parola administrator **MySql/MariaDB** (masterrc)



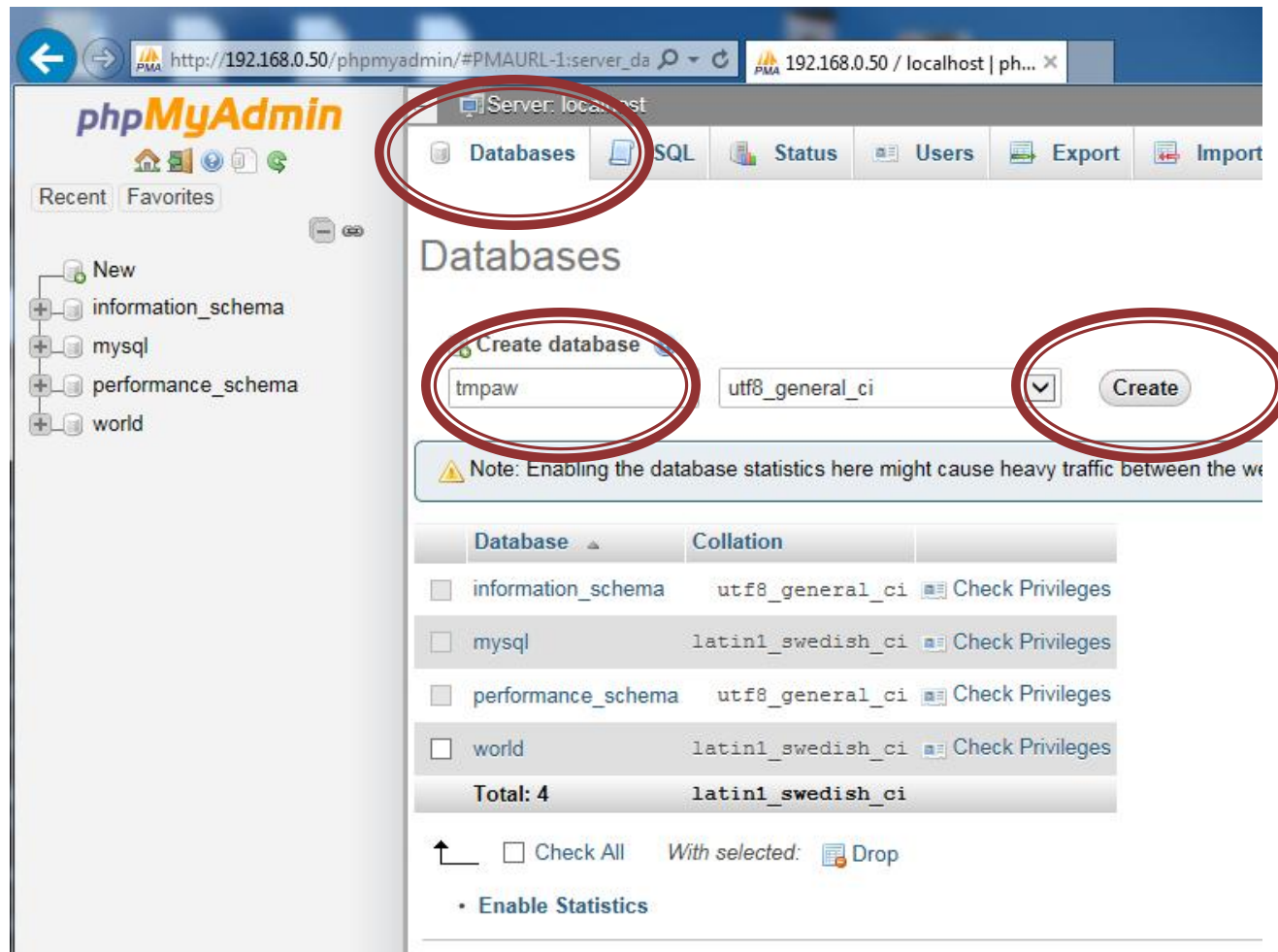
PhpMyAdmin

The screenshot displays the phpMyAdmin web interface in a browser window. The address bar shows the URL `http://192.168.0.50/phpmyadmin/#PMAURL-0:index.php` and the server name `192.168.0.50 / localhost | ph...`. The interface includes a navigation menu on the left with options like "Recent" and "Favorites", and a main content area with several panels:

- General Settings:** Includes a "Change password" link and a "Server connection collation" dropdown menu set to `utf8mb4_unicode_ci`.
- Appearance Settings:** Includes a "Language" dropdown set to "English", a "Theme" dropdown set to "pmahomme", and a "Font size" dropdown set to "82%". A "More settings" link is also present.
- Database server:** Lists server details:
 - Server: Localhost via UNIX socket
 - Server type: MariaDB
 - Server version: 5.5.44-MariaDB - MariaDB Server
 - Protocol version: 10
 - User: root@localhost
 - Server charset: UTF-8 Unicode (utf8)
- Web server:** Lists web server details:
 - Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod_fcgid/2.3.9
 - PHP/5.4.16 mod_python/3.5.0- Python/2.7.5
 - Database client version: libmysql - 5.5.44-MariaDB
 - PHP extension: mysqli
 - PHP version: 5.4.16
- phpMyAdmin:** Lists version and resource information:
 - Version information: 4.4.15.1
 - Documentation
 - Wiki
 - Official Homepage
 - Contribute
 - Get support
 - List of changes

Creare Baza de Date

- Databases → "nume" → Create



The screenshot shows the phpMyAdmin interface. The 'Databases' tab is selected and circled in red. Below it, the 'Create database' form is visible, with the database name 'tmpaw' entered in the first field, 'utf8_general_ci' in the second, and a dropdown menu set to 'utf8_general_ci'. The 'Create' button is also circled in red. A table below the form lists existing databases and their collations.

Database	Collation	
<input type="checkbox"/> information_schema	utf8_general_ci	Check Privileges
<input type="checkbox"/> mysql	latin1_swedish_ci	Check Privileges
<input type="checkbox"/> performance_schema	utf8_general_ci	Check Privileges
<input type="checkbox"/> world	latin1_swedish_ci	Check Privileges
Total: 4	latin1_swedish_ci	

↑ Check All With selected: [Drop](#)

• [Enable Statistics](#)

Creare tabelle in baza de date

- Baza de date (in lista) → Structure → div Create Table → nume/coloane → Go

The screenshot displays the phpMyAdmin web interface. The browser address bar shows the URL `http://192.168.0.50/php`. The main content area shows the 'Database: tmpaw' view. The 'Structure' tab is selected, and a 'Create table' dialog is open. The 'Name' field contains 'categorii' and the 'Number of columns' field contains '3'. The 'Go' button is visible at the bottom right. The left sidebar shows a tree view of databases, with 'tmpaw' selected. The top navigation bar includes buttons for 'Structure', 'SQL', 'Search', 'Query', 'Export', and 'More'. A warning message at the top of the main area states 'No tables found in database.'

Introducere coloane, tabel categorii

- (eventual) Adaugare coloane / Stabilire nume
- Name / Type / Length / Default

The screenshot shows the phpMyAdmin interface for creating a table named 'categorii'. The table name is entered in the 'Table name' field. Below it, the 'Add 1 column(s) Go' button is visible. The table structure is defined by three columns:

Name	Type	Length/Values	Default	Collation
id_categ	INT		None	
nume	VARCHAR	45	None	
detalii	VARCHAR	150	None	

The interface also shows a sidebar with a tree view of databases (information_schema, mysql, performance_schema, tmpaw, world) and a top navigation bar with options like Browse, Structure, SQL, Search, Import, and Privileges. The browser address bar shows the URL: http://192.168.0.50/phpmyadmin/#PMAURL-3:tbl_create.php?d

Introducere coloane

- (eventual) NOT NULL / Index / Auto Increment
 - in functie de “necesitatile” coloanei respective

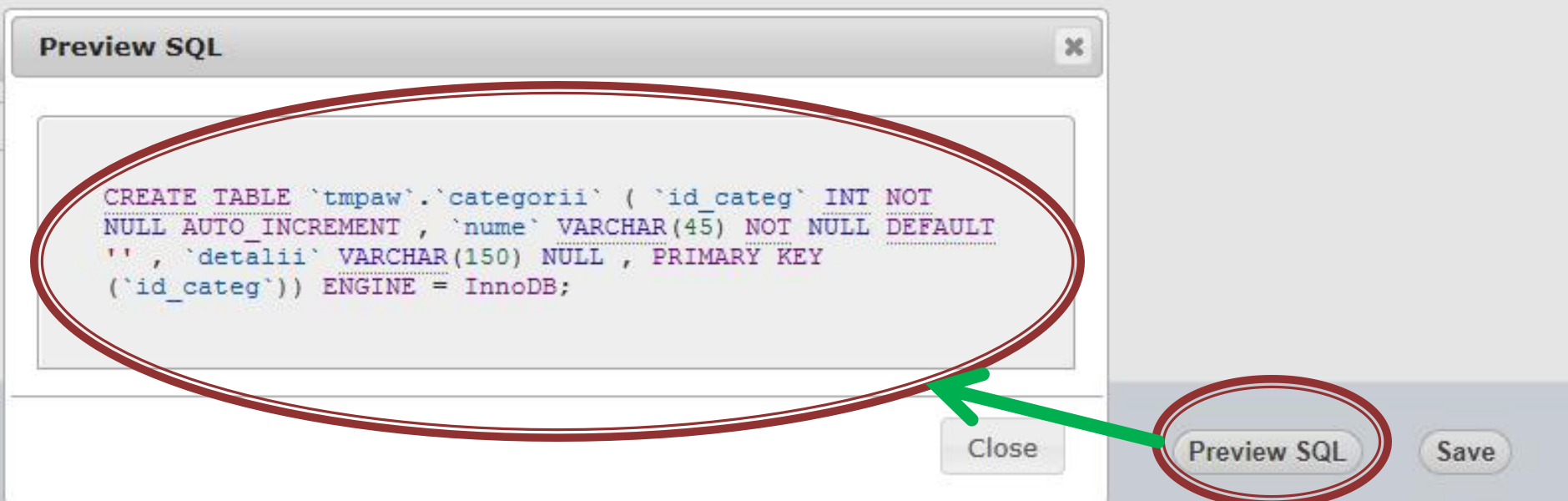
Table name: Add column(s)

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
id_categ	INT		None			<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	
nume	VARCHAR	45	As defined:			<input type="checkbox"/>	---	<input type="checkbox"/>	
detalii	VARCHAR	150	None			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	

Preview SQL

- in aproape toate etapele in PhpMyAdmin
 - exemplu de cod SQL/schelet utilizabil (copy/paste) in aplicatia PHP
 - modificari de finete absente din interfata
 - copy → Sectiune "SQL" in interfata → paste → modificare



Introducere coloane, tabel produse

- New → Nume → Add Columns → ...

The screenshot shows the phpMyAdmin interface for a database named 'tmpaw'. The 'Structure' tab is active, and the table 'produse' is selected. The 'Add 1 column(s)' button is highlighted with a red circle. The 'Table name' field contains 'produse'. The 'Structure' table below shows the following columns:

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	C
id_produs	INT		None			<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	
id_categ	INT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	
nume	VARCHAR	45	As defined:			<input type="checkbox"/>	---	<input type="checkbox"/>	
detalii	VARCHAR	150	None			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	
cant	INT		None			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	
pret	FLOAT		None			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>	

Introducere date initiale (interfata)

- Tabel → Insert → Completare → Go

The screenshot shows the phpMyAdmin interface for a table named 'categorii' in the 'tmpaw' database. The 'Insert' tab is selected, and the 'id_categ' column is highlighted. The 'nume' column contains the value 'papetarie'. The 'Go' button is visible at the bottom right. The 'Continue insertion with' dropdown is set to '1' row.

Column	Type	Function	Null	Value
id_categ	int(11)			
nume	varchar(45)			papetarie
detalii	varchar(150)		☑	

Continue insertion with row

Vizualizare date existente

- Tabel → Browse → salt la pagina (numar de linii pe pagina)

The screenshot displays the phpMyAdmin interface for a MySQL database. The browser address bar shows the URL `http://192.168.0.50/phpmyadmin/`. The interface is in the 'Browse' view for the 'categoriasii' table in the 'tmpaw' database. The 'Browse' tab is highlighted in the top navigation bar. The table structure is shown as follows:

id_categ	nume	detalii
1	papetarie	NULL
2	instrumente	NULL
3	audio-video	NULL

The interface also shows a query editor with the SQL statement `SELECT * FROM `categoriasii`` and a results table with 3 rows. The 'Number of rows' is set to 25, and the 'Filter rows' field is empty. The 'Sort by key' is set to 'None'. The 'Query results operations' section at the bottom includes options for 'Print view', 'Print view (with full texts)', 'Export', 'Display chart', and 'Create view'.

Introducere date initiale (SQL)

- Tabel → SQL → completare → Go

The screenshot shows the phpMyAdmin interface with the following elements:

- Navigation:** The left sidebar shows a tree view of databases. The 'tmpaw' database is selected, and the 'produse' table is highlighted.
- SQL Tab:** The 'SQL' tab is active, showing an SQL query:

```
1 INSERT INTO `produse` (`id_produ`  
2 `id_categ`, `nume`, `detalii`, `cant`, `pret`)  
3 VALUES  
4 (1,1,'carte','mai multe pagini scrise legate',0,100),  
5 (2,1,'caiet','mai multe pagini goale legate',0,75),  
6 (3,1,'hartie scris','mai multe pagini goale NElegate',0,50),  
7 (4,2,'penar','loc de depozitat instrumente de scris',0,150),  
8 (5,2,'stilou','instrument de scris albastru',0,125),  
9 (6,2,'creion','instrument de scris gri',0,25),  
10 (7,3,'cd','canta',0,50),  
11 (8,3,'dvd','vizual',0,100),  
12 (9,3,'blue ray','vizual extrem',0,500);
```
- Buttons:** Below the query editor are buttons for 'SELECT *', 'SELECT', 'INSERT', 'UPDATE', 'DELETE', 'Clear', and 'Format'. The 'Go' button is highlighted in the bottom right corner.
- Columns:** A 'Columns' list on the right shows the table structure: id_produ, id_categ, nume, detalii, cant, pret.

Tabel produse

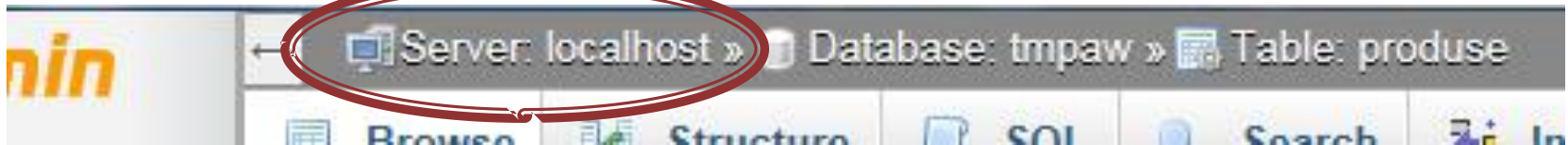
The screenshot shows the phpMyAdmin interface for a database named 'tmpaw'. The 'Structure' tab is selected, displaying the table 'produse'. The table structure is as follows:

id_produs	id_categ	nume	detalii	cant	pret
1	1	carte	mai multe pagini scrise legate	0	100
2	1	caiet	mai multe pagini goale legate	0	75
3	1	hartie scris	mai multe pagini goale NElegate	0	50
4	2	penar	loc de depozitat instrumente de scris	0	150
5	2	stilou	instrument de scris albastru	0	125
6	2	creion	instrument de scris gri	0	25
7	3	cd	canta	0	50
8	3	dvd	vizual	0	100
9	3	blue ray	vizual extrem	0	500

The interface also shows a SQL query: `SELECT * FROM `produse`` and a table with 9 rows. The 'Structure' tab and the 'produse' table name in the left sidebar are circled in red.

Adaugare utilizator

- Server → Users → Add user

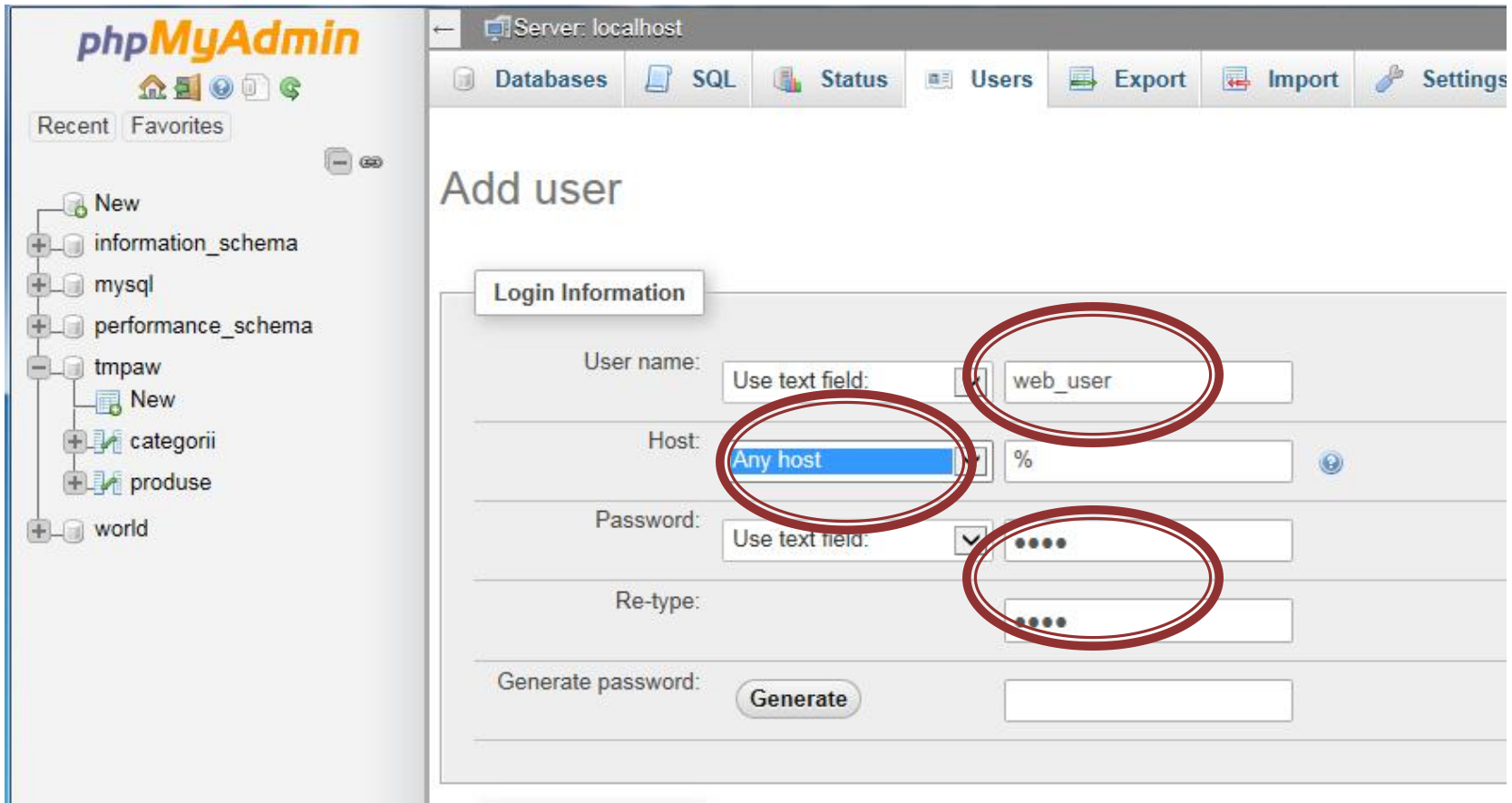


A screenshot of the phpMyAdmin 'Users overview' page. The navigation bar at the top shows 'Server: localhost' circled in red. Below it, the 'Users' tab is also circled in red. The main content area displays a table of users with columns: User name, Host, Password, Global privileges, Grant, and Action. At the bottom, a 'New' button is circled in red, with an 'Add user' link below it.

	User name	Host	Password	Global privileges	Grant	Action
<input type="checkbox"/>	root	127.0.0.1	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	:::1	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	localhost	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	tmpaw.etti	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	web	%	Yes	USAGE	No	Edit Privileges Export

Adaugare utilizator

- Nu e recomandabil/**posibil** sa se utilizeze user-ul MySql "root" pentru aplicatii



The screenshot shows the phpMyAdmin interface for adding a new user. The 'Login Information' section is visible, with the following fields:

- User name: (circled in red)
- Host: (circled in red)
- Password: (circled in red)
- Re-type: (circled in red)

The 'Generate password' section is also visible, with a 'Generate' button and an empty text field.

Drepturi de acces

- Server → Users → Edit Privileges

The screenshot shows the phpMyAdmin interface. The breadcrumb 'Server: localhost' is circled in red. The 'Users' menu item in the top navigation bar is also circled in red. The 'Users overview' table is displayed below, with the 'Edit Privileges' link for the 'web_user' entry circled in red.

	User name	Host	Password	Global privileges	Grant	Action
<input type="checkbox"/>	root	127.0.0.1	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	:::1	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	localhost	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	root	tmpaw.etti	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export
<input type="checkbox"/>	web	%	Yes	USAGE	No	Edit Privileges Export
<input type="checkbox"/>	web_user	%	Yes	USAGE	No	Edit Privileges Export

Drepturi de acces

- Database → nume → Go

The screenshot shows the phpMyAdmin interface for a MySQL server on localhost. The 'Database' tab is selected in the navigation menu. The main content area displays the 'Edit Privileges: User 'web_user'@'%' page. Under the 'Database-specific privileges' section, there is a table with the following structure:

Database	Privileges	Grant	Table-specific privileges	Action
None				
mysql				
tmpaw				
world				

Below the table, there is a text input field labeled 'Add privileges on the following database(s):' with a dropdown menu containing the selected databases: mysql, tmpaw, and world. The 'Database' tab in the navigation menu and the list of databases in the table are circled in red.

Drepturi de acces

- Se alocă drepturile SELECT + INSERT + UPDATE + DELETE asupra bazei de date create

The screenshot displays the phpMyAdmin interface for editing database privileges. The user 'web_user'@'%' is selected for the database 'tmpaw'. The 'Data' section is checked, indicating that SELECT, INSERT, UPDATE, and DELETE privileges are granted. The 'Structure' and 'Administration' sections are unchecked.

Server: localhost

Databases SQL Status Users Export Import Settings Replicati

Database Table

Edit Privileges: User **'web_user'@'%'** - Database **tmpaw**

Database-specific privileges Check All

Note: MySQL privilege names are expressed in English.

Data	Structure	Administration
<input checked="" type="checkbox"/> SELECT	<input type="checkbox"/> CREATE	<input type="checkbox"/> GRANT
<input checked="" type="checkbox"/> INSERT	<input type="checkbox"/> ALTER	<input type="checkbox"/> LOCK TABLES
<input checked="" type="checkbox"/> UPDATE	<input type="checkbox"/> INDEX	<input type="checkbox"/> REFERENCES
<input checked="" type="checkbox"/> DELETE	<input type="checkbox"/> DROP	
	<input type="checkbox"/> CREATE TEMPORARY TABLES	
	<input type="checkbox"/> SHOW VIEW	

Drepturi de acces, verificare

- Nume → Privileges
- Marea majoritate a aplicatiilor **nu** au nevoie de drepturi de acces la structura/administrare

Server: localhost » Database: tmpaw

Structure SQL Search Query Export Import Operations **Privileges** Routing

Users having access to "tmpaw"

User	Host	Type	Privileges	Grant	Action	
<input type="checkbox"/>	root	127.0.0.1	global	ALL PRIVILEGES	Yes	Edit Privileges
<input type="checkbox"/>	root	:::1	global	ALL PRIVILEGES	Yes	Edit Privileges
<input type="checkbox"/>	root	localhost	global	ALL PRIVILEGES	Yes	Edit Privileges
<input type="checkbox"/>	root	tmpaw.etti	global	ALL PRIVILEGES	Yes	Edit Privileges
<input type="checkbox"/>	web_user	%	database-specific	SELECT, INSERT, UPDATE, DELETE	No	Edit Privileges

Check All With selected: Export

Index

- Adaugare index e esentiala pentru viteza
 - exemplu, produse grupate pe categorii, selectia produselor dintr-o categorie se face cu :
 - `SELECT * FROM `produse` WHERE `id_categ` = 1`
- Tabel → Structure → Index / Selectare + Index

The screenshot shows the phpMyAdmin interface for a database named 'tmpaw'. The 'Table: produse' structure is displayed. The table has six columns: id_produs, id_categ, nume, detalii, cant, and pret. The 'id_categ' column is highlighted with a green circle, and the 'Index' icon for this column is circled in red. The 'Structure' tab is also circled in red. The 'categorii' and 'produse' tables in the left sidebar are also circled in red. The 'Index' icon in the bottom toolbar is circled in green.





#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	id_produs	int(11)			No	None	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	id_categ	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
3	nume	varchar(45)	utf8_general_ci		No			Change Drop Primary Unique Index Spatial Fulltext Distinct values
4	detalii	varchar(150)	utf8_general_ci		Yes	NULL		Change Drop Primary Unique Index Spatial Fulltext Distinct values
5	cant	int(11)			Yes	NULL		Change Drop Primary Unique Index Spatial Fulltext Distinct values
6	pret	float			Yes	NULL		Change Drop Primary Unique Index Spatial Fulltext Distinct values

Verificare/Stergere index

- Apasare +Indexes, se deschide lista de indecsi
- Apasare -Indexes, se inchide lista de indecsi

- Indexes

Indexes ⓘ

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
 Edit  Drop PRIMARY		BTREE	Yes	No	id_produ	9	A	No	
 Edit  Drop id_categ		BTREE	No	No	id_categ	9	A	No	

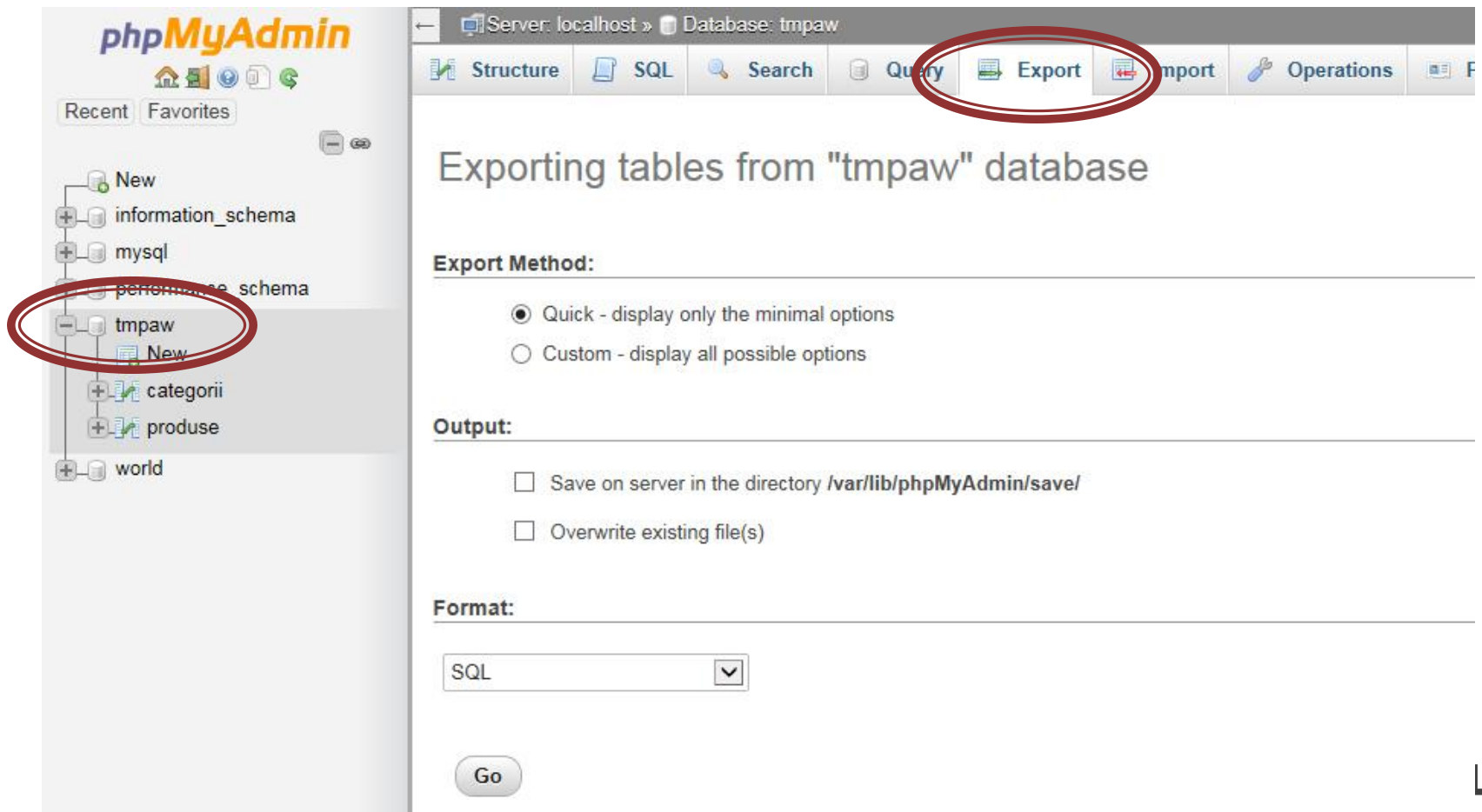
Create an index on columns

Backup, Restore

- Ca și în cazul Windows 2000 facilitatea de Backup realizează un script SQL care conține structura și datele exprimate sub forma de interogări SQL
- O deosebire între PhpMyAdmin și aplicațiile specifice MySQL (aceleși de pe Windows 2000 sau MySQL Workbench) este absența liniilor de creare a bazei de date
 - CREATE DATABASE IF NOT EXISTS tmpaw;
 - USE tmpaw;
- La utilizarea PhpMyAdmin trebuie să se creeze manual înaintea restaurării baza de date

Backup

- Nume (tabel sau baza de date) → Export



The screenshot displays the phpMyAdmin interface for a database named 'tmpaw' on a localhost server. The left sidebar shows a tree view of databases, with 'tmpaw' highlighted and circled in red. The main content area shows the 'Export' menu option also circled in red. Below the menu, the 'Exporting tables from "tmpaw" database' screen is visible, featuring options for 'Export Method' (Quick or Custom), 'Output' (Save on server or Overwrite existing file(s)), and 'Format' (SQL).

Server: localhost » Database: tmpaw

Structure SQL Search Query **Export** Import Operations F

Exporting tables from "tmpaw" database

Export Method:

- Quick - display only the minimal options
- Custom - display all possible options

Output:

- Save on server in the directory `/var/lib/phpMyAdmin/save/`
- Overwrite existing file(s)

Format:

SQL

Go

Restore

- Se creaza in avans baza de date
- Nume → Import → Browse (alegere fisier backup)
- fisierele SQL pot fi compresate gzip, bzip2, zip

The screenshot displays the phpMyAdmin web interface. On the left sidebar, the database 'tmpaw' is selected and circled in red. The main content area shows the 'Import' page for the 'tmpaw' database. The 'Import' button in the top navigation bar is also circled in red. The 'File to Import:' section includes a text input field, a 'Browse...' button (circled in red), and a file size limit of 2048KiB. Below this, there are radio buttons for 'Browse your computer' and 'Select from the web server upload directory'. The 'Character set of the file:' dropdown is set to 'utf-8'. The 'Partial Import:' section has a checked checkbox for 'Allow the interruption of an import in case the script detects it is close to the PHP timeout limit.' and a text input field for 'Skip this number of queries (for SQL) or lines (for other formats), starting from the first one:' with the value '0'.

MySql – Server Windows 2000

Mini – Indrumar practic Lucru cu bazele de date

Realizarea bazei de date

- Se recomanda utilizarea utilitarului **MySQL Query Browser** sau un altul echivalent pentru crearea scheletului de baza de date (detalii – laborator 1)
- Se initializeaza aplicatia cu drepturi depline (“root” si parola)
 - se creaza o noua baza de date:
 - in lista “Schemata” – Right click – Create New Schema
 - se activeaza ca baza de date curenta noua “schema” – Dublu click pe numele ales

Introducere tabele

- Introducere tabel – Click dreapta pe numele bazei de date aleasa – Create New Table
- se defineste structura tabelului
 - nume coloane
 - tip de date
 - NOT NULL – daca se accepta ca acea coloana sa ramana fara date (NULL) sau nu
 - AUTOINC – daca acea coloana va fi de tip intreg si va fi incrementata automat de server (util pentru crearea cheilor primare)
 - Default value – valoarea implicita care va fi inserata daca la introducerea unei linii noi nu se mentioneaza valoare pentru acea coloana (legat de optiunea NOT NULL)

Tabel Categorii

The screenshot shows the MySQL Table Editor interface for a table named 'categorii' in the 'tmpaw' database. The table is currently empty. The editor is configured with the following settings:

- Table Name:** categorii
- Database:** tmpaw
- Comment:** InnoDB free: 11264 kB

The **Columns and Indices** tab is active, showing the following columns:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
id_categ	INT(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL	NULL	
nume	VARCHAR(45)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> BINARY		
detalii	VARCHAR(150)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> BINARY	NULL	

The **Indices** tab is also active, showing a primary index named 'PRIMARY' of kind 'PRIMARY' and type 'BTREE'. The index is defined on the 'id_categ' column.

Buttons at the bottom of the editor include 'Apply Changes', 'Discard Changes', and 'Close'.

Tabel Prognose

The screenshot shows the MySQL Table Editor window for a table named 'produse' in the 'tmpaw' database. The table has a comment 'InnoDB free: 11264 kB'. The 'Columns and Indices' tab is active, displaying the following table structure:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
id_producs	INT(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL	NULL	
id_categ	INT(10)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL		
nume	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/> BINARY		
detalii	VARCHAR(150)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> BINARY	NULL	
cant	INT(10)	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL	NULL	
pret	FLOAT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL	NULL	


The 'Indices' tab is also active, showing a PRIMARY index with the following settings:

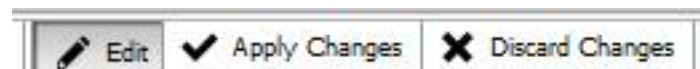
- Index Name: PRIMARY
- Index Kind: PRIMARY
- Index Type: BTREE
- Index Columns: id_producs

The background shows the MySQL Query Browser interface with a resultset table containing 9 rows of data:

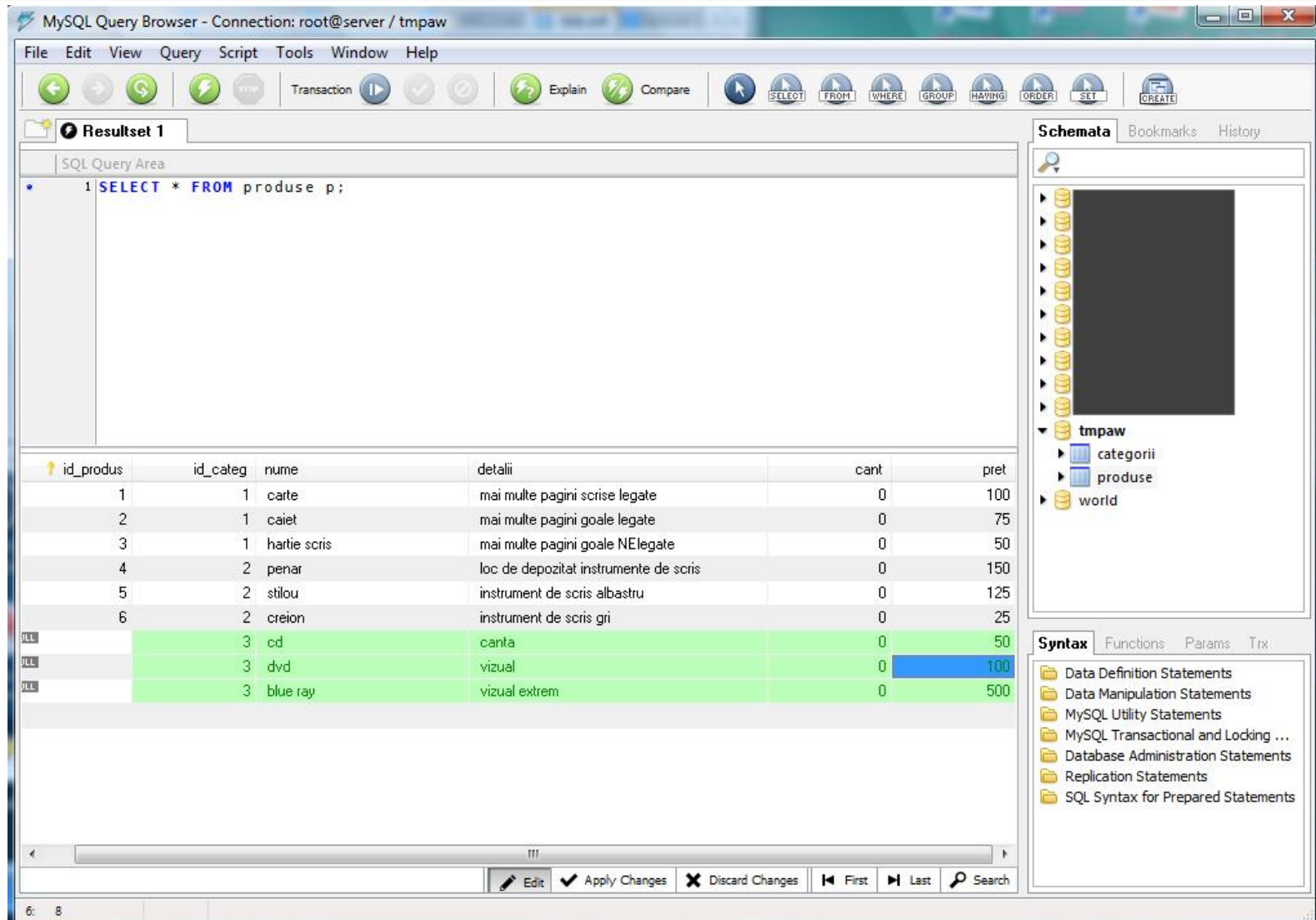
id_producs	id_categ
1	
2	
3	
4	
5	
6	
7	
8	
9	

Introducere date initiale

- Dublu click pe tabel → In zona "SQL Query Area" se completeaza interogarea de selectie totala
 - SELECT * FROM produse p;
- Executia interogarii SQL
 - Meniu → Query → Execute
 - Bara de butoane 
- Lista rezultata
 - initial vida
 - poate fi editata – butoanele "Edit", "Apply Changes", "Discard Changes" din partea de jos a listei



Introducere date initiale



The screenshot displays the MySQL Query Browser interface. The main window shows a query result set for the query `SELECT * FROM produse p;`. The result set contains 9 rows of data, with the last three rows highlighted in green. The columns are `id_produs`, `id_categ`, `nume`, `detalii`, `cant`, and `pret`.

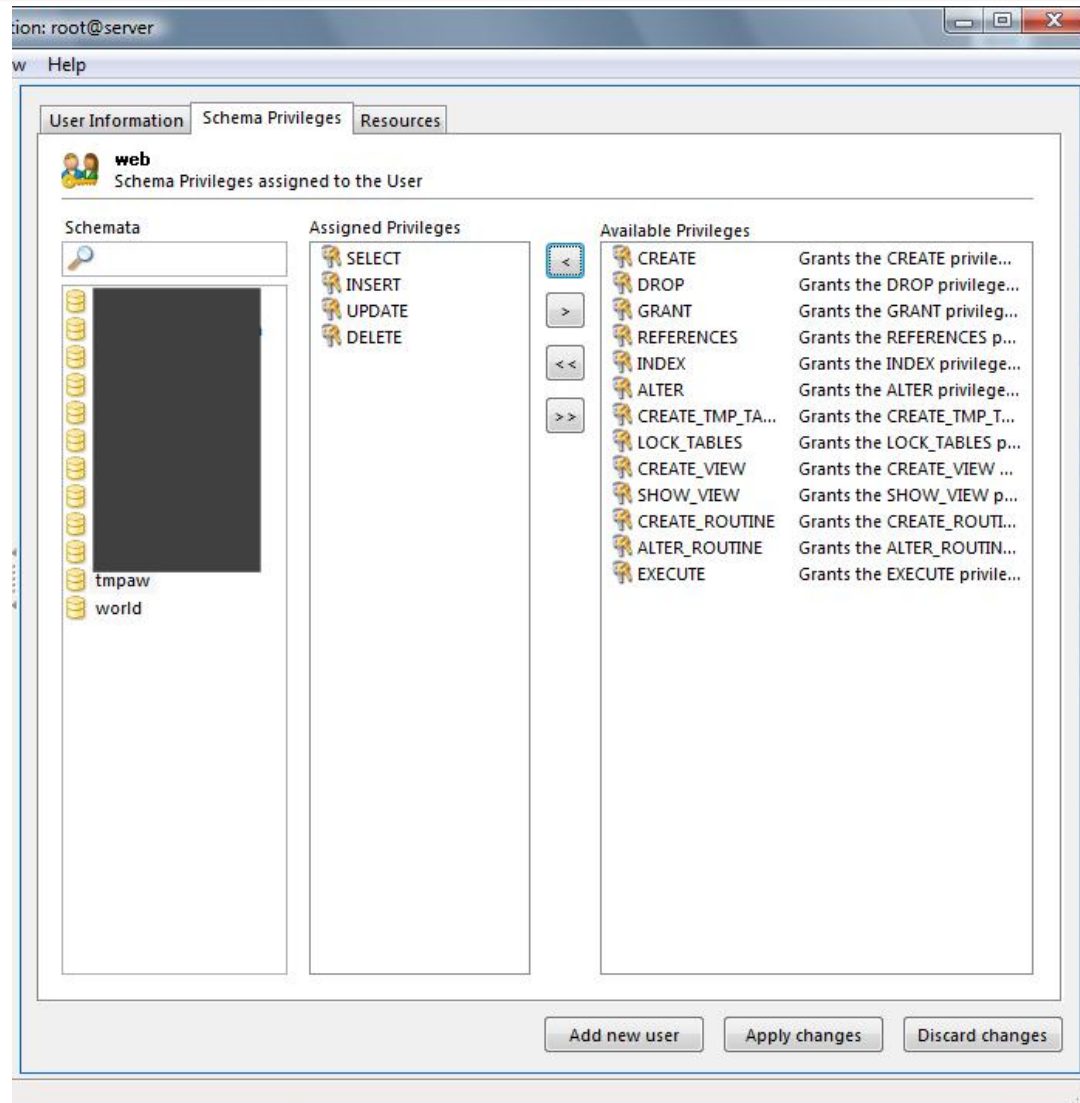
id_produs	id_categ	nume	detalii	cant	pret
1	1	carte	mai multe pagini scrise legate	0	100
2	1	caiet	mai multe pagini goale legate	0	75
3	1	hartie scris	mai multe pagini goale NElegate	0	50
4	2	penar	loc de depozitat instrumente de scris	0	150
5	2	stilou	instrument de scris albastru	0	125
6	2	creion	instrument de scris gri	0	25
7	3	cd	canta	0	50
8	3	dvd	vizual	0	100
9	3	blue ray	vizual extrem	0	500

The interface also shows a 'Schemata' panel on the right, displaying a tree view of the database structure. The 'tmpaw' database is expanded, showing 'categorii' and 'produse' tables. The 'Syntax' panel at the bottom right lists various SQL statement categories.

Backup, Restore, drepturi de acces

- Se recomanda utilizarea utilitarului **MySQL Administrator** sau un altul echivalent (detalii – laborator 1)
- Se initializeaza aplicatia cu drepturi depline (“root” si parola)
- Se creaza un utilizator limitat (detalii – laborator 1)
- Se aloca drepturile “SELECT” + “INSERT” + “UPDATE” asupra bazei de date create (sau mai multe daca aplicatia o cere)

Drepturi de acces



Backup

The screenshot shows the MySQL Administrator interface for configuring a backup project. The window title is "MySQL Administrator - Connection: root@server". The main area is titled "Backup Project" and has three tabs: "Backup Project", "Advanced Options", and "Schedule".

General

Project Name: Name for this backup project.

Schemata

The Schemata list on the left includes: school, tmpaw, and world. The tmpaw schema is selected and highlighted in blue.


Backup Content

Data directory	Obj...	Rows	Data ...	Last update
<input checked="" type="checkbox"/> tmpaw				
<input checked="" type="checkbox"/> categorii	Inno...	3	16384	
<input checked="" type="checkbox"/> produse	Inno...	9	16384	

At the bottom of the window, there are three buttons: "New Project", "Save Project", and "Execute Backup Now".

Yellow arrows indicate the workflow: from the "Backup" icon in the left sidebar to the "tmpaw" schema in the Schemata list, then to the "Backup Content" table, and finally to the "Execute Backup Now" button.

Restaurarea bazei de date

- Din **MySql Administrator**
 - Sectiunea Restore → "Open Backup File"
- Din **MySql Query Browser**
 - Meniu → File → Open Script
 - Executie script SQL
 - Meniu → Script → Execute
 - Bara de butoane 
- Scriptul SQL rezultat contine comenzile/interogariile SQL necesare pentru crearea bazei de date si popularea ei cu date

Contact

- Laboratorul de microunde si optoelectronica
- <http://rf-opto.etti.tuiasi.ro>
- rdamian@etti.tuiasi.ro