

Curs 9  
2013/2014

# Tehnici moderne de proiectare a aplicatiilor web

MySql

# Laborator 7

# Rezultat (cumparator)

**Magazin Firma**

[Inceput](#) | [Inapoi](#)

## Magazin online Firma X SRL

Alegeti:

- [Cumparator](#)
- [Vanzator](#)

### Categorii Produse

Alegeti categoria:

Nr.	Categorie	Total Produse
1	<a href="#">Papetarie</a>	3
2	<a href="#">Instrumente</a>	3
3	<a href="#">Audio-video</a>	3
4	<a href="#">Calculatoare</a>	3
5	<a href="#">Jucarii</a>	2

Total produse: 14

## Magazin online Firma X SRL

### Realizati comanda

Nr.	Produs	Pret	Cantitate
1	Carti	100	<input type="text" value="1"/>
2	Caiete	50	<input type="text" value="2"/>
3	Penare	150	<input type="text" value="1"/>
4	Stilouri	125	<input type="text" value="0"/>
5	Creioane	25	<input type="text" value="0"/>

## Magazin online Firma X SRL

### Rezultate comanda

Pret total (fara TVA): 350

Pret total (cu TVA): 416.5

Comanda receptionata la data: 17/03/2010 ora 08:24

# Rezultat (vanzator)

**Magazin Firma X**

[Inceput](#) | [Inapoi](#)

## Magazin online Firma X SRL

Alegeti:

- [Cumparator](#)
- [Vanzator](#)

### Categorii Produse

Alegeti categoria:

Nr.	Categorie	Total Produse
1	<a href="#">Papetarie</a>	3
2	<a href="#">Instrumente</a>	3
3	<a href="#">Audio-video</a>	3
4	<a href="#">Calculatoare</a>	3
5	<a href="#">Jucarii</a>	2

Total produse: 14

Categorie noua de produse:

### Lista produse in categoria Calculatoare

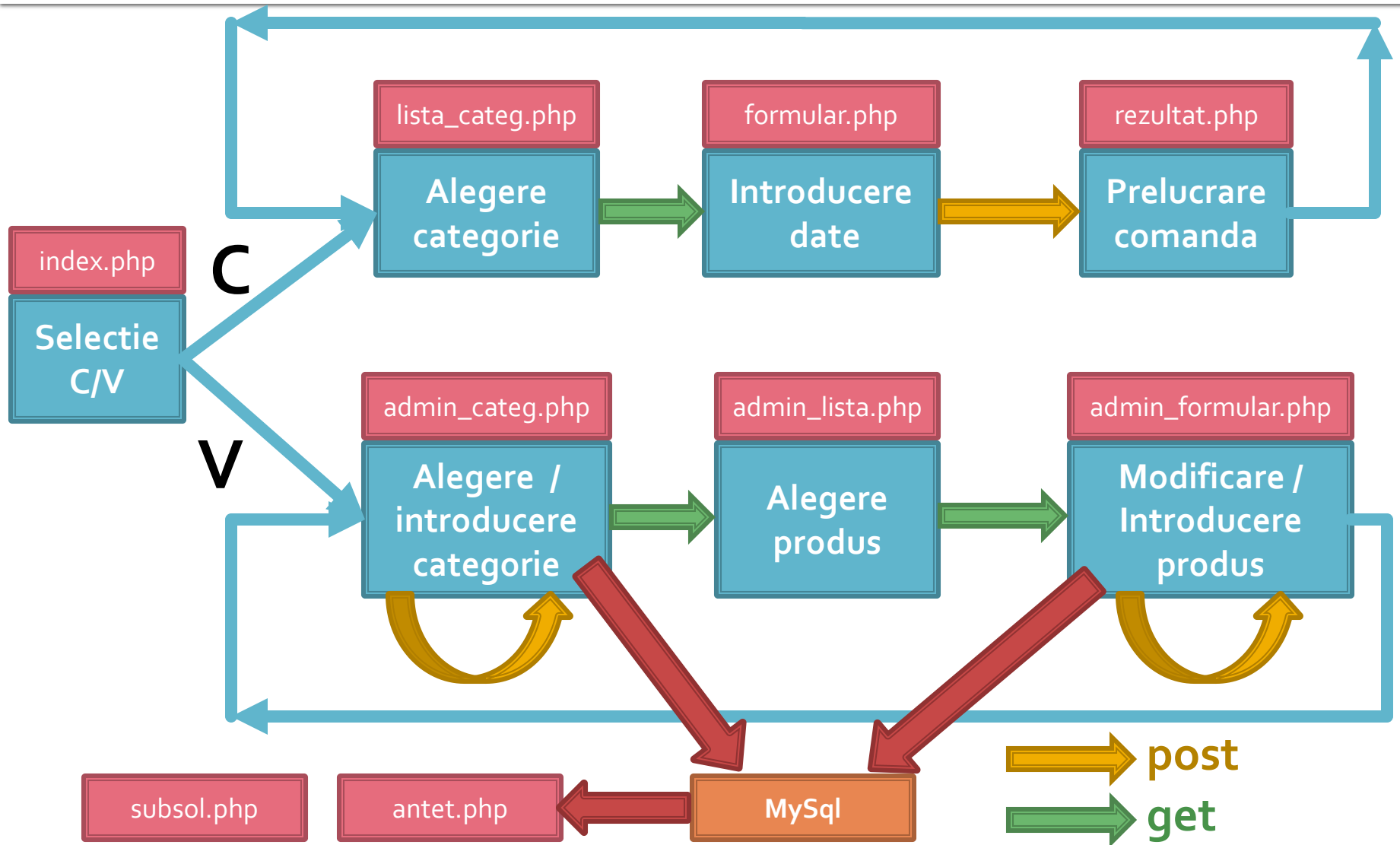
Nr.	Produs	Descriere	Pret	Cantitate	Actiuni
1	Laptop	calculator mic	2000	2	<a href="#">modifica</a>
2	Desktop	calculator mare	1000	5	<a href="#">modifica</a>
3	Imprimanta	prn	200	2	<a href="#">modifica</a>
-	Produs nou				<a href="#">adauga</a>

### Produs in categoria Calculatoare

Produs	<input type="text" value="laptop"/>
Descriere	<input type="text" value="calculator mic"/>
Pret	<input type="text" value="2000"/>
Cantitate	<input type="text" value="2"/>



# Plan aplicatie



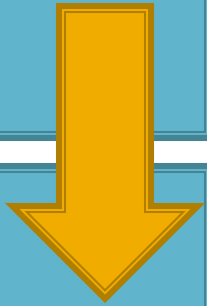
# Plan aplicatie – vanzator

- Deoarece citirea datelor se face in fisierul antet.php (modificat anterior) vor aparea modificari doar la nivelul scrierii datelor noi introduse
- Fisiere
  - admin\_lista.php – nemodificat
  - admin\_categ.php – scrie categorii noi in baza de date: se incuieste cod XML cu cod MySql
  - admin\_formular.php – scrie produse noi / corectii in baza de date: se incuieste cod XML cu cod MySql

# admin\_categ.php

```
if (isset($_POST["c_nou"]))
    {//categorie noua introdusa
    $categ_nou=$xml->addChild("categorie");
    $categ_nou->addAttribute("nume", $_POST["nou"]);
    $xml->asXML("lista.xml"); // salvare fisier
    $produse[$_POST["nou"]]=array(); // update matrice produse
    echo "<p>Categoria ".$_POST["nou"]." adaugata!</p>";
    }
```

```
if (isset($_POST["c_nou"]))
    {//categorie noua introdusa
    $query = "INSERT INTO `categorii` (`nume`, `detalii`)VALUES (
    '".$_POST["nou_nume"]."`, '".$_POST["nou_desc"]."`)";
    echo $query; //util in perioada de testare
    $result = mysql_query($query, $conex) or die(mysql_error());
    $record=mysql_insert_id(); //obtinerea id-ului nou
    $produse[$_POST["nou_nume"]]=array(); // update matrice produse
    echo "<p>Categoria ".$_POST["nou_nume"]." adaugata! Are id = ".$record."</p>";
    }
```



# admin\_categ.php

Magazin Firma X SRL

[Inceput](#) | [Inapoi](#)

## Magazin online Firma X SRL

### Categorii Produse

Alegeti categoria:

Nr.	Categorie	Total Produse
1	<a href="#">Papetarie</a>	3
2	<a href="#">Instrumente</a>	3
3	<a href="#">Audio-video</a>	3

Total produse: 9

Categorie noua de produse:

Nume:

Descriere:

Magazin Firma X SRL

[Inceput](#) | [Inapoi](#)

## Magazin online Firma X SRL

```
INSERT INTO `categorii` (`nume`, `detalii`) VALUES ('jucarii', 'pentru copii')
```

Categoria jucarii adaugata! Are id = 4

### Categorii Produse

Alegeti categoria:

Nr.	Categorie	Total Produse
1	<a href="#">Papetarie</a>	3
2	<a href="#">Instrumente</a>	3
3	<a href="#">Audio-video</a>	3
4	<a href="#">Jucarii</a>	0

Total produse: 9

Categorie noua de produse:

Nume:

Descriere:

## Magazin online Firma X SRL

```
INSERT INTO `categorii` (`nume`, `detalii`) VALUES ('jucarii', 'pentru copii')
```

Categoria jucarii adaugata! Are id = 4



# admin\_formular.php

- Pentru inlocuire/adaugare produs apare o tratare diferita a celor doua situatii:
  - Adaugarea de produs face apel la interogarea SQL `INSERT INTO `produse` ...`
  - Modificarea unui produs existent va face apel la interogarea SQL `UPDATE `produse` SET ...`

# admin\_formular.php

```
if (isset($_POST["prod_ant"]))//exista deja acest produs anterior?
    //exista deja acest produs UPDATE
    unset($produse[$_POST['categ']][$_POST['prod_ant']]);//trebuie sters produsul anterior inlocuit
    $query = "UPDATE `produse` SET `nume`='".$_POST["prod"]."', `detalii`='".$_POST["descriere"]."',
`cant`='".$_POST["cantitate"]."', `pret`='".$_POST["pret"]."' WHERE `nume`='".$_POST["prod_ant"].'";
    echo $query;//util in perioada de testare
    $result = mysql_query($query, $conex) or die(mysql_error());
    echo "<p>Produsul '".$_POST["prod"]."' modificat in categoria '".$_POST['categ'].'"!</p>";
}
else
    //NU exista acest produs INSERT
    $query = "INSERT INTO `produse` (`nume`, `detalii`, `pret`, `cant`, `id_categ`) VALUES
('".$_POST["prod"]."', '".$_POST["descriere"]."', '".$_POST['pret']."', '".$_POST['cantitate']."',
(SELECT `id_categ` FROM categorii WHERE `nume` = '".$_POST['categ'].')";
    echo $query;//util in perioada de testare
    $result = mysql_query($query, $conex) or die(mysql_error());
    $record=mysql_insert_id();//obtinerea id-ului nou
    echo "<p>Produsul '".$_POST["prod"]."' adaugat in categoria '".$_POST['categ'].'"! Are id =
".$_record."</p>";
}
$produse[$_POST['categ']][$_POST['prod']] = array("descr" => $_POST['descriere'], "pret" => $_POST['pret'], "cant" =>
$_POST['cantitate']);
```

# rf-opto.etti.tuiasi.ro

← → http://rf-opto.e

- Main
- Courses
- Master
  - RCD
  - IT
- Staff
- Research
- Students

Pagina veche poate fi accesata [aici](#)

English  
Romana  
Pas encore

copyright © 2009 rf-opto  
realizat RF Tech

- php/sql start (backup?)

### Observatii importante

1. Lucrul la disciplina TMPAW este orientat in special spre lucru individual si prog
2. In consecinta, la fiecare laborator exista sectiunea Copy/Paste pentru downl  
punct de plecare pentru laboratorul curent. Exceptie sunt bineinteles sursele  
Copy/Paste | Baza de date initiala.
3. Un rezultat urmarit este obisnuinta lucrului cu anumite accesorii:
  - o Consultarea log-ului de erori a serverului Apache (in principiu cu Logvi
  - o Consultarea manualului electronic PHP pentru detalii de introducere a
  - o Obisnuinta consultarii exemplarelor existente si cautarea unei rezolvari,
  - o PHP) - nu este eficienta din punct de vedere tehnologic reinventare
  - o Utilizarea accesorilor MySql pentru accelerarea operatiilor de backup/r
4. Inainte de a incepe lucrul la proiect verificati ca indepliniti cerintele de la pun
5. Retineti ca 1p din nota la proiect este obtinut prin obtinerea functionalitatii i  
foloseasca baze de date MySql se utilizeaza o alta tehnologie limitata: fisier te
6. Data limita pentru:
  - o sustinere proiect: S14, ora de laborator.
  - o activitatea suplimentara anuntata la curs: S14, inainte de ultimul curs

Poate fi consultata o propunere de [master/cursuri post universitare IT](#)

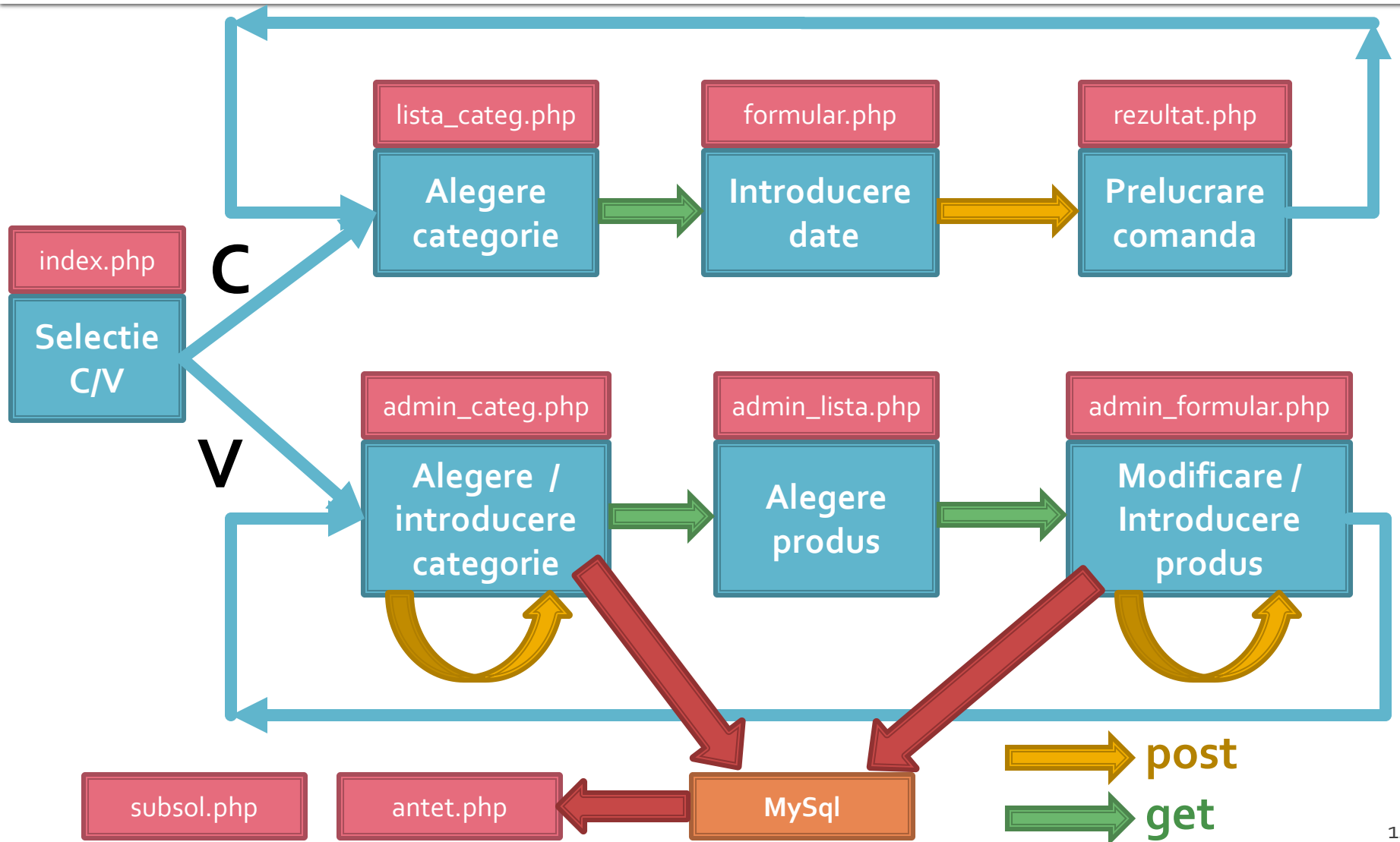
# Final laborator

- Sursele complete ale aplicatiei pot fi obtinute de pe site-ul laboratorului
- Utilizarea MySql in aplicatii asa cum a fost facuta in acest exemplu **nu este optima**
  - Se incarca initial intreaga baza de date intr-o matrice de produse (antet.php)
  - Aceasta metoda nu este eficienta:
    - Server-ul MySql este o aplicatie compilata nativa sistemului de operare pe care ruleaza, in timp ce PHP este un limbaj interpretat
    - Se incarca inutil toate datele chiar si atunci cand nu este necesar (de exemplu cand afisez doar produsele dintr-o categorie sau cand afisez pentru a fi modificate doar detaliile unui produs)

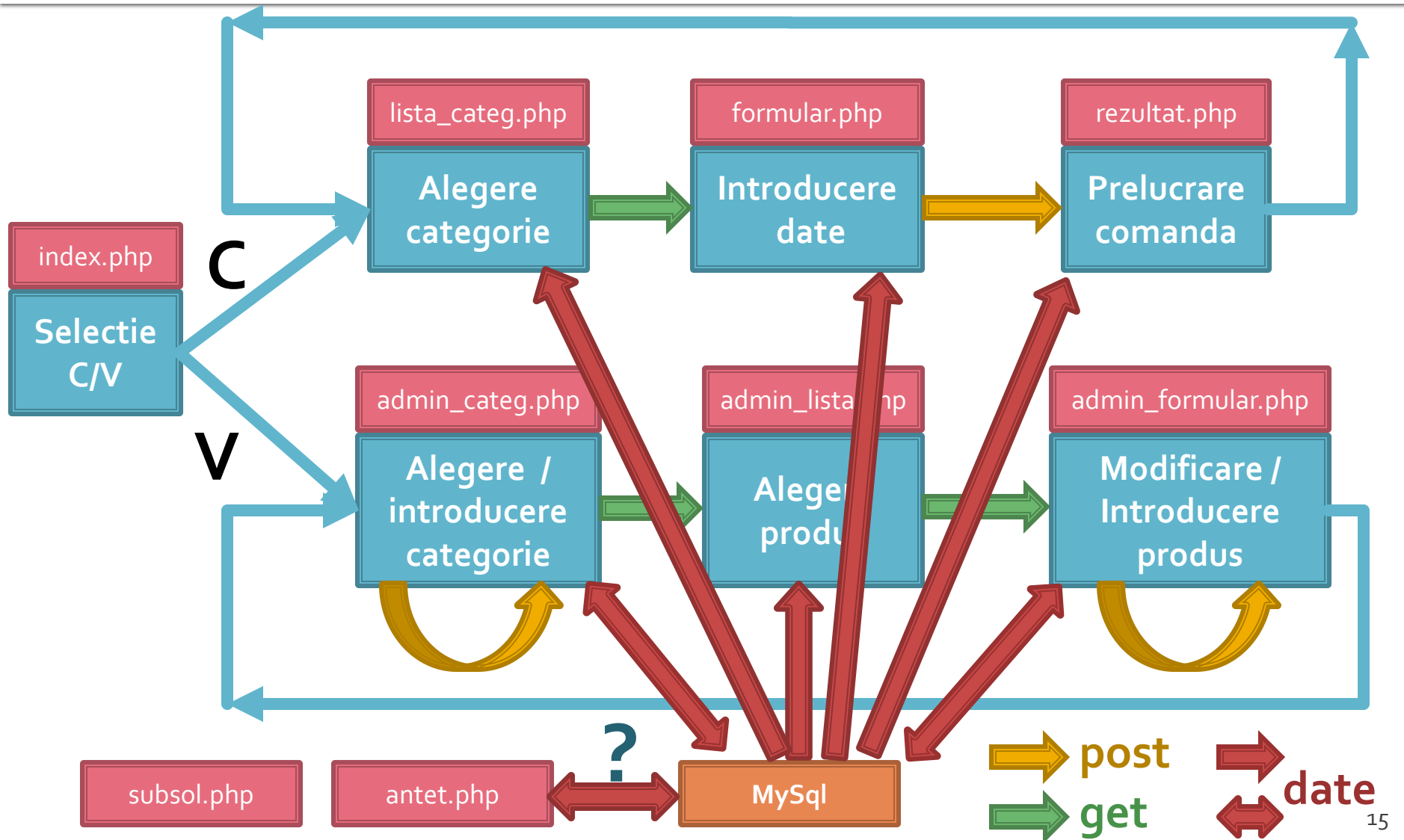
# Final laborator

- Varianta corecta presupune:
  - Citirea datelor in fiecare fisier in parte
  - Selectia datelor necesare pe server-ul MySql (mult mai eficient decat PHP)
  - De multe ori e mai eficienta utilizarea resursei rezultate din interogarea SQL decat crearea unei variabile matriciale suplimentare
    - `$result = mysql_query($query, $conex);`  
`$row_result = mysql_fetch_assoc($result);`  
`..... $row_result['nume'] .....`;

# Plan aplicatie - laborator



# Plan aplicatie - optim



# Activitate suplimentara



# Activitate suplimentara

- Exemplul prezentat in sursele de pe site (laborator) este inefficient
- Suplimentar ascunde o **greseala de logica** care impiedica functionarea corecta a programului
  - programul nu este protejat, nu verifica faptul ca in casuta in care se asteapta numere nu se introduc siruri de text
  - **greseala de logica** presupune utilizatorul **cooperant si educat**, introduce ceea ce se asteapta de la el sa introduca, dar chiar in aceste conditii apare o abatere de la functionarea corecta

# Recompensa activitate suplimentara

- Raspunsul corect va fi recompensat cu:
  - **2p** in plus la nota de laborator (se pot compensa astfel eventuale absente)
  - **2p** in plus la nota de la testarea finala (examen)
- Nota de la proiect
  - Nu este influentata
- Nota finala se obtine prin medie ponderata **dupa** aplicarea suplimentelor amintite mai sus

# Regulament recompensa

- Raspunsul si codul de corectie trebuie trimise individual prin email
- Codul trebuie sa fie functional
- Maxim **2** incercari pentru fiecare student
- Studentii pot discuta intre ei **dar**
- Oricare **doua raspunsuri identice se elimina reciproc**

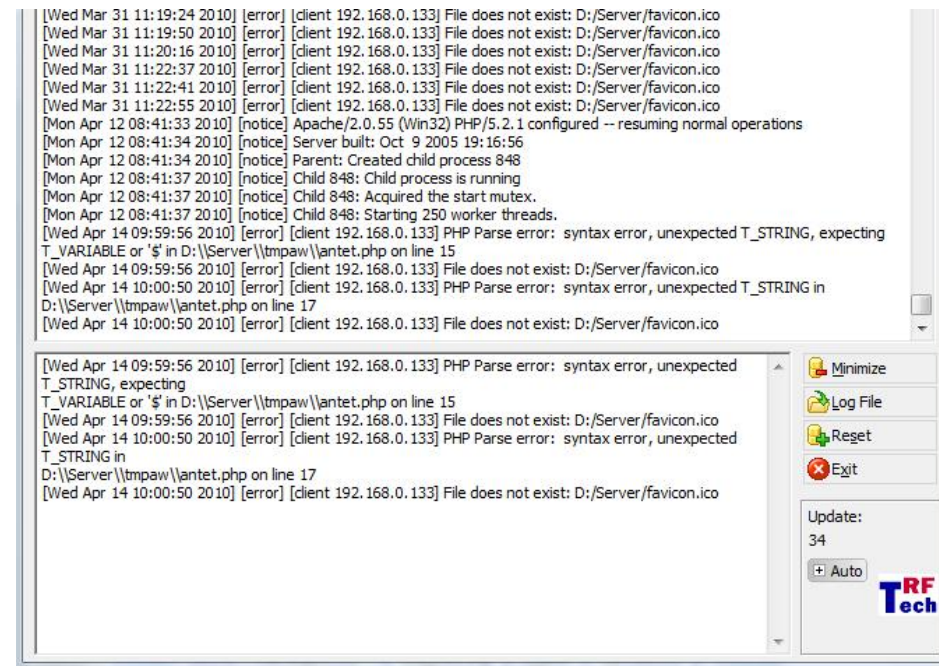
# Aspecte practice recomandate in realizarea aplicatiilor web

# 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. Utilizarea aplicatiei prezentata la laborator este mai comoda datorita automatizarii dar orice alta varianta este utila

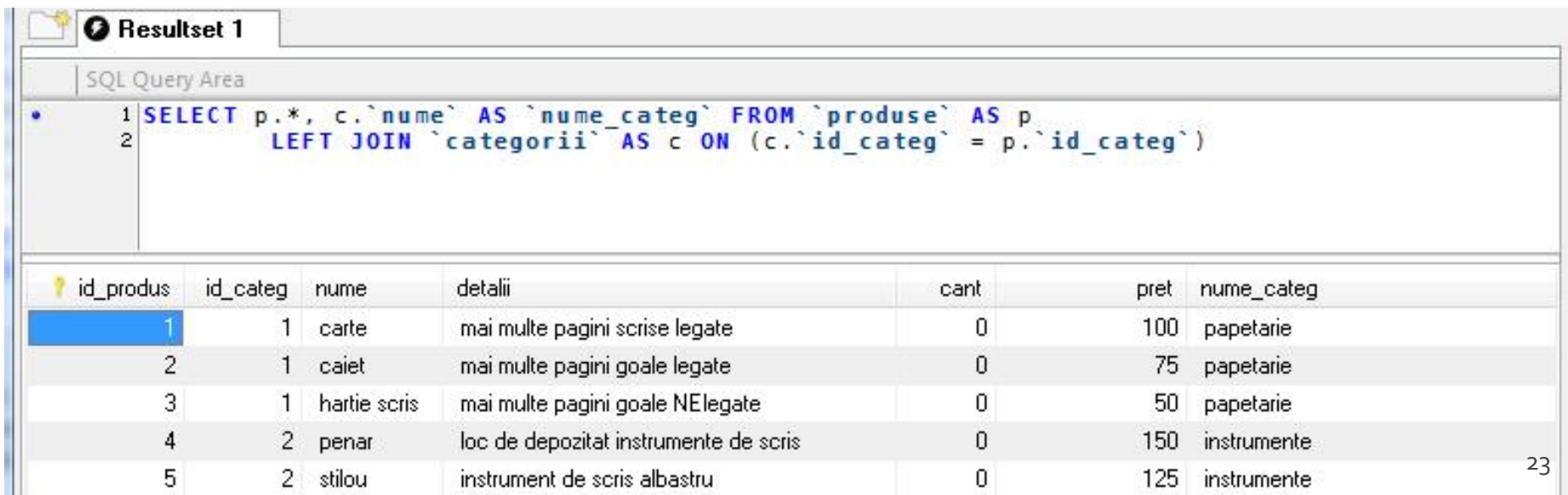


```
[Wed Mar 31 11:19:24 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:19:50 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[Wed Mar 31 11:20:16 2010] [error] [client 192.168.0.133] File does not exist: D:/Server/favicon.ico
[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\\tmpaw\\antet.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\\tmpaw\\antet.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\\tmpaw\\antet.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\\tmpaw\\antet.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** pentru incercarea interogarilor, urmand ca apoi, cand sunteti multumiti de rezultat, sa transferati interogarea SQL in codul PHP



The screenshot shows the MySQL Query Browser interface. At the top, there is a tab labeled "Resultset 1". Below it is the "SQL Query Area" containing 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`)
```

Below the query area is a table displaying the results of the query. The table has 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 echivalent) / 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

# 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('val<sub>1</sub>', 'val<sub>2</sub>', ...)

- una singura din cele maxim 65535 valori distincte posibile

- SET('val<sub>1</sub>', 'val<sub>2</sub>', ...)

- niciuna sau mai multe din cele maxim 64 valori distincte
    - echivalent cu "setare de biti" într-un întreg pe 64 biti cu tabela asociată

MySql/PHP

# 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_connect`
  - conectare la server-ul MySQL
  - resource `mysql_connect` ( [string server [, string username [, string password [, bool new\_link [, int client\_flags]]]] )
  - tipic: `mysql_connect($host, $user, $pass)`
  - tipic: `$host="localhost"`
- `mysql_pconnect` – persistent pentru reutilizarea conexiunilor

# Funcții PHP de acces MySQL

- `mysql_select_db`
  - selectarea bazei de date dorita
  - bool `mysql_select_db` ( string `database_name` [, resource `link_identifier`] )
  - resursa este obtinuta in urma unui apel anterior la `mysql_connect` sau `mysql_pconnect`
- `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

# Funcții PHP de acces MySQL

- `mysql_num_rows`
  - indica numărul de linii returnate SELECT de o interogare sau SHOW
  - int `mysql_num_rows` ( resource result )
- `mysql_affected_rows`
  - indica numărul de linii afectate de o interogare INSERT, UPDATE, REPLACE sau DELETE
  - int `mysql_affected_rows` ( [resource link\_identifier] )
- `mysql_insert_id`
  - returnează valoarea unei eventuale coloane autoincrement generate de o interogare INSERT precedentă
  - int `mysql_insert_id` ( [resource link\_identifier] )

# Funcții PHP de acces MySQL

## Parcurgerea resurselor rezultat

- `mysql_fetch_assoc`
  - returnează o **matrice asociativă** corespunzătoare liniei de la indexul intern (indecsi de tip șir corespunzători denumirii coloanelor – field – din tabelul de date) și incrementează indexul intern sau **false** dacă nu mai sunt linii
  - array `mysql_fetch_assoc` ( resource result )
- `mysql_fetch_row`
  - returnează o matrice cu indecsi întregi
  - 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 )

# Exemplu de utilizare

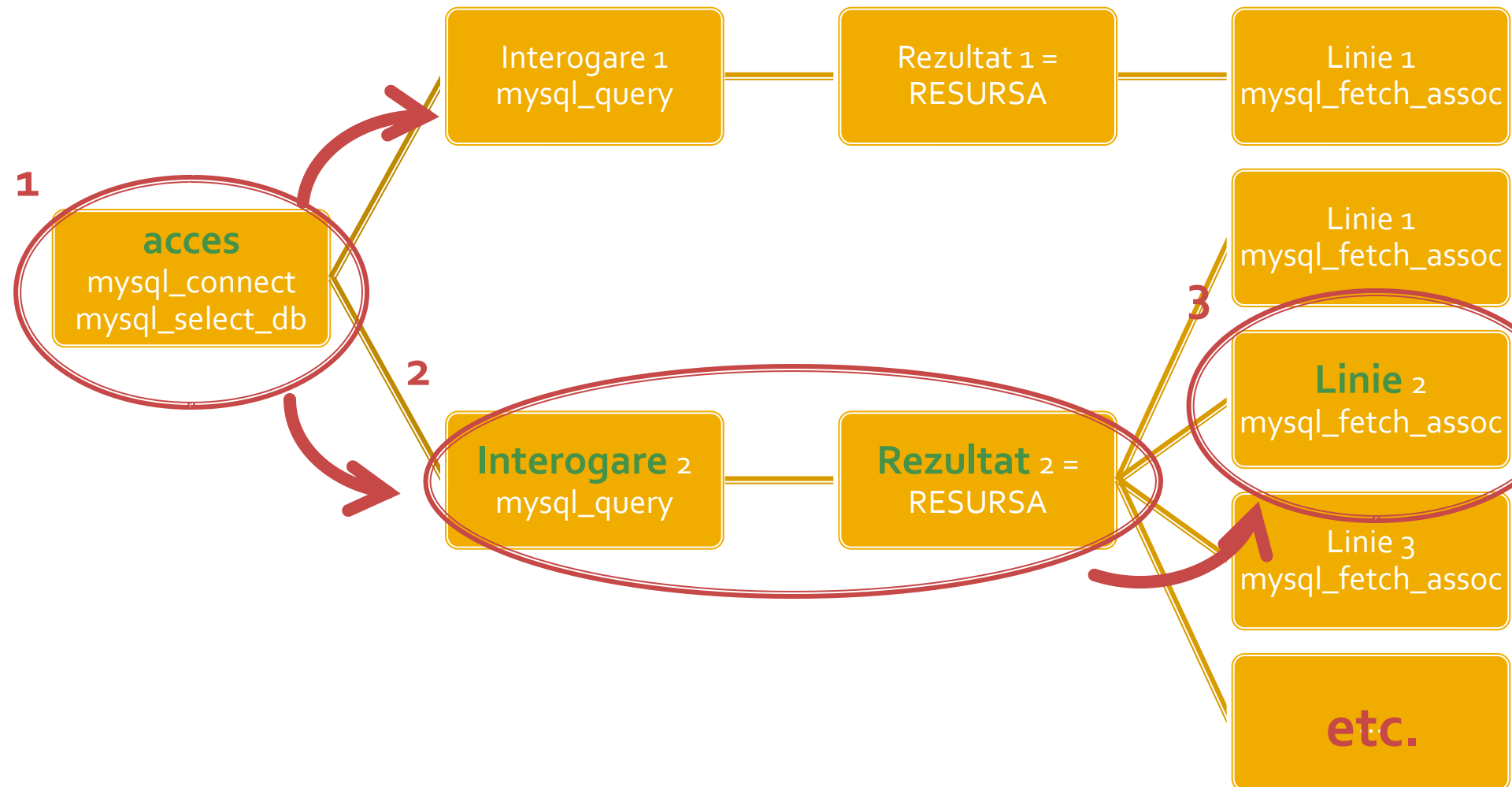
```
$hostname = "localhost";  
$database = "world";  
$username = "web";  
$password = "ceva";  
$conex= mysql_connect($hostname, $username, $password);  
mysql_select_db($database, $conex);
```

```
$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 )); ?>
```

# Funcții de acces la server-ul MySQL



# 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	...
...	...	...	...

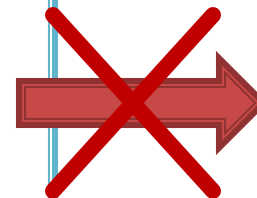
Functii de acces la structura



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 )); ?>
```

# 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 excepția 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 spațiu

# 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**) “`” (langa “1” pe tastatura)
  - 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)

# Metode de stocare

- Metoda de stocare a datelor nu e o caracteristica a server-ului ci a fiecarui tabel in parte
- Exemplu ulterior CREATE: "ENGINE = InnoDB"
- MySql suporta diferite metode de stocare, fiecare cu avantajele/dezavantajele sale
- Implicit se foloseste metoda MyISAM, dar la instalarea server-ului (laborator 1) o anumita selectie poate schimba valoarea implicita in InnoDB
- **Alegerea metodei de stocare potrivita are implicatii majore asupra performantei aplicatiei**

# Metode de stocare

- MyISAM
- InnoDB
- Memory
- Merge
- Archive
- Federated
- NDBCLUSTER
- CSV
- Blackhole
- Example

# Metode de stocare

## ■ MyISAM

- metoda de stocare implicita in MySql
- performanta ridicata (resurse ocupate si viteza)
- posibilitatea cautarii in intregul text (index FULLTEXT)
- blocare acces la nivel de tabel
- **nu** accepta tranzactii
- **nu** accepta FOREIGN KEY
  - probleme relative la integritatea datelor

## ■ InnoDB

## ■ Memory

# Metode de stocare

- **MyISAM**
- **InnoDB**
  - devine metoda de stocare implicita in MySql daca la instalare se alege model tranzactional
  - performanta medie (resurse ocupate si viteza)
  - blocare acces la nivel de linie
  - **nu** accepta index FULLTEXT (posibilitatea cautarii in intregul text, index FULLTEXT apare doar **MySql 5.6 ->** )
  - **accepta** tranzactii
  - **accepta** FOREIGN KEY
    - probleme mai putine la integritatea datelor prin constrangeri intre tabele
- **Memory**



# Metode de stocare

- MyISAM
- InnoDB
- **Memory**
  - metoda de stocare recomandata pentru tabele temporare
  - performanta maxima (viteza – datele sunt stocate in RAM)
    - **la oprirea server-ului datele se pierde**, tabelul este pastrat dar va fi fara nici o linie
  - **nu** accepta tipuri de date mari (BLOB, TEXT) – maxim 255 octeti
  - **nu** accepta index FULLTEXT
  - **nu** accepta tranzactii
  - **nu** accepta FOREIGN KEY
    - probleme relative la integritatea datelor

MySql

# 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' on the 'id\_categ' column. The index settings are:

- Index Name:** PRIMARY
- Index Kind:** PRIMARY
- Index Type:** BTREE
- Index Columns:** id\_categ

The **Apply Changes**, **Discard Changes**, and **Close** buttons are visible at the bottom of the editor.

# Tabel Prognose

The screenshot shows the MySQL Table Editor interface for a table named 'produse' in the 'tmpaw' database. The table is currently empty. 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	

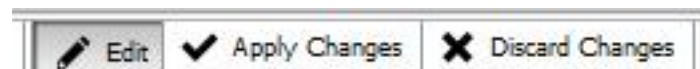
The 'Indices' tab is also active, showing a primary index named 'PRIMARY' on the 'id\_producs' column. The index settings are:

- Index Name: PRIMARY
- Index Kind: PRIMARY
- Index Type: BTREE
- Index Columns: id\_producs

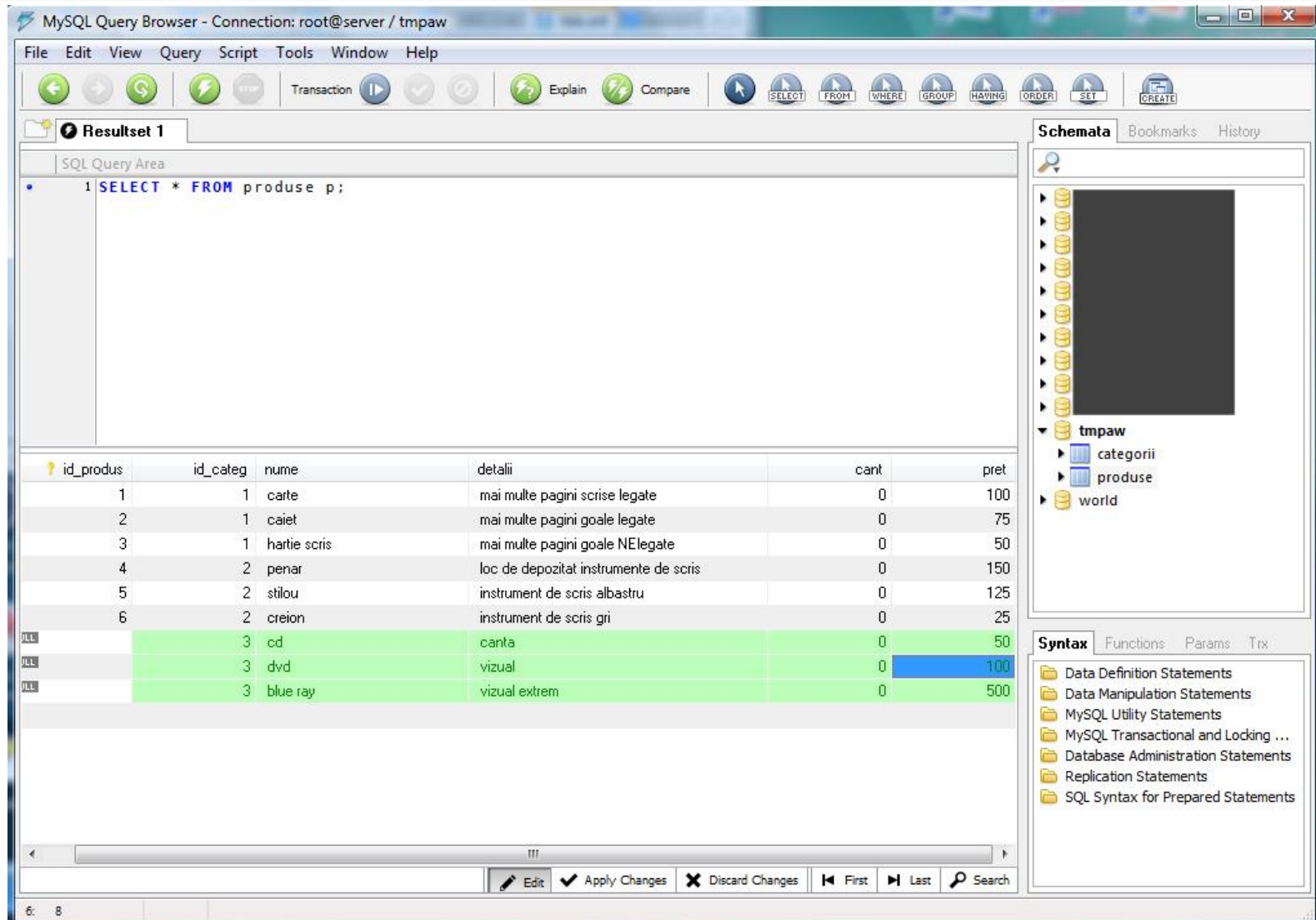
The background shows the MySQL Query Browser interface with a query area containing a partial SQL statement: `1 SELECT * FROM`. The status bar at the bottom indicates '6: 8'.

# 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 the SQL Query Area with the query: `1 SELECT * FROM produse p;`. Below the query area, the Resultset 1 is displayed as a table with the following data:

id_produș	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
ALL	3	cd	canta	0	50
ALL	3	dvd	vizual	0	100
ALL	3	blue ray	vizual extrem	0	500

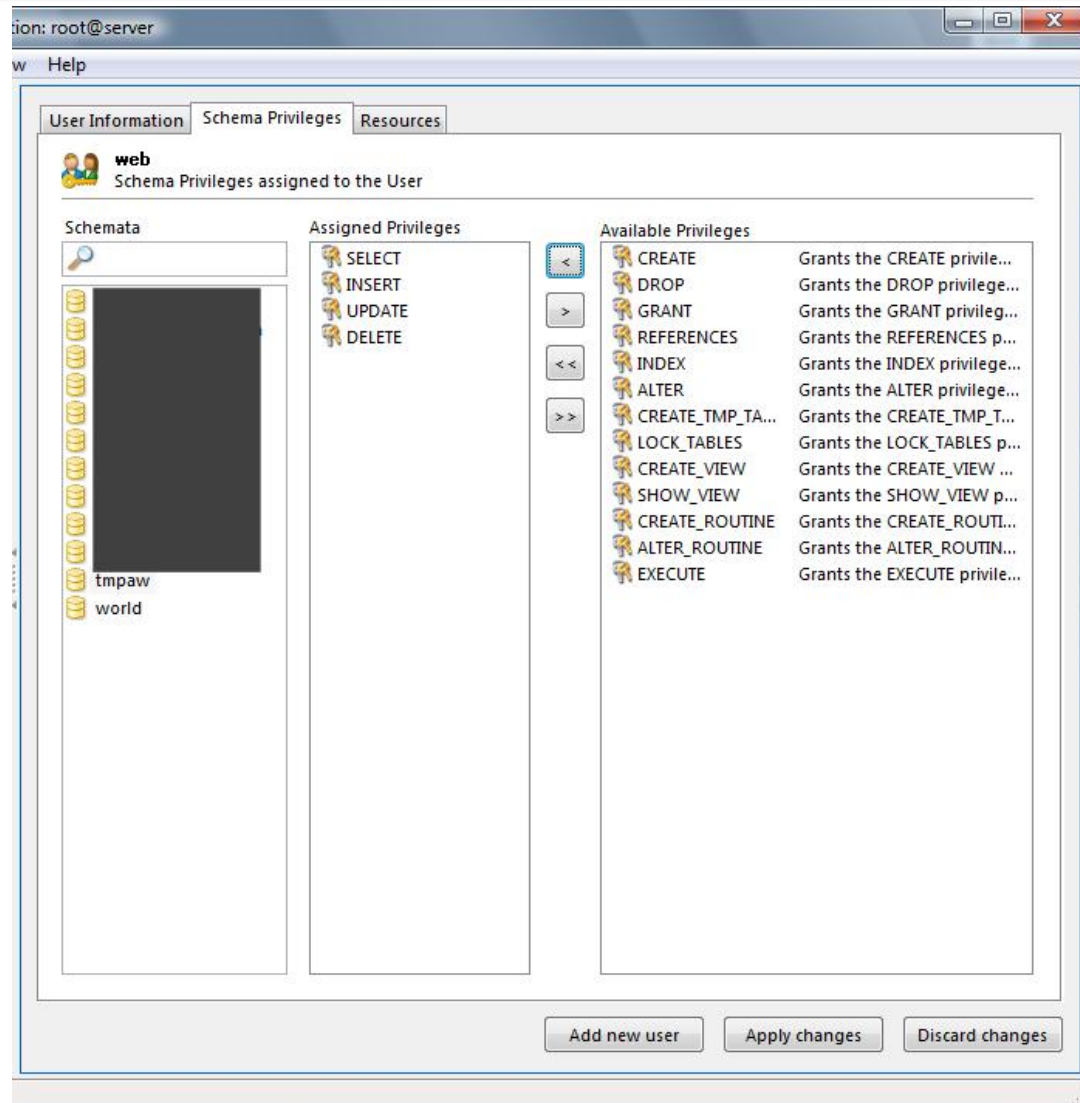
The interface also includes a menu bar (File, Edit, View, Query, Script, Tools, Window, Help), a toolbar with various icons, and a right-hand panel with tabs for Schemata, Bookmarks, and History. The Schemata tab shows a tree view of the database structure, including the 'tmpaw' database with 'categorii' and 'produse' tables. The Syntax tab is also visible, showing a list of 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: one arrow points from the "Backup" icon in the left sidebar to the "Backup Project" tab; another arrow points from the "tmpaw" schema in the Schemata list to the "Backup Content" table; and a third arrow points from the "Execute Backup Now" button to the right side of the window.

# 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

# Script SQL Backup - utilitate

- Poate fi folosit ca un model extrem de bun pentru comenzile necesare pentru crearea programatica (din PHP de exemplu) a bazei de date

```
CREATE DATABASE IF NOT EXISTS tmpaw;
USE tmpaw;

DROP TABLE IF EXISTS `categorii`;
CREATE TABLE `categorii` (
  `id_categ` int(10) unsigned NOT NULL auto_increment,
  `nume` varchar(45) NOT NULL,
  `detalii` varchar(150) default NULL,
  PRIMARY KEY (`id_categ`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `categorii` (`id_categ`,`nume`,`detalii`) VALUES
(1,'papetarie',NULL),
(2,'instrumente',NULL),
(3,'audio-video',NULL);
```

# Indicatii examinare

# Teme de proiect

- La toate temele **1p** din nota este obtinut de indeplinirea functionalitatii cerute.
- La toate temele forma paginii prezinta importanta (dependentă de dificultatea temei)

# PROIECT (final)

- Tema de nota 7 (>6)
  - Tema unica pentru fiecare student
- Tema de nota 8 (>6)
  - Conditiiile de la tema de nota 8 **si in plus**
  - Necesitatea conlucrarii intre 2 studenti cu doua teme "pereche"



# PROIECT (final)

- Tema de nota 9 (>5)
  - Condițiile de la tema de nota 8 **si in plus**
  - Necesitatea conlucrării între **3 studenți** cu trei teme “pereche”
  - Tema se predă/trimitte cu macar **1 zi** înainte a sustinerii ei
  - Baza de date cu care se lucreaza sa contina minim **50** de inregistrari in tabelul cel mai "voluminos".
- Tema de nota 10 (>5)
  - Condițiile de la tema de nota 9 **si in plus**
  - Baza de date cu care se lucreaza contine minim **300** de inregistrari in tabelul cel mai "voluminos"
  - Necesitatea investigării posibilitatilor de **imbunatatire** a aplicatiei si adaugarii de functionalitate
  - nota individuala la proiect va depinde intr-o mica masura (in limita a 1p) de nota medie a colegilor din echipa

# 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 9 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 si 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 fiecarei echipe. Conditii:
  - metoda de comunicare in echipa sa fie prin email sau direct
  - latenta 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 de nota 10+ (>5, in general offline)
  - Conditiiile de la tema de nota 10 **si in plus**
  - Baza de date cu care se lucreaza contine minim 400 de inregistrari in tabelul cel mai "voluminos"
  - 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 prezenta la laborator (DACA toate celelalte conditii sunt indeplinite – P = 66%, L = 0%, E = 33%)

# Exemplu

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

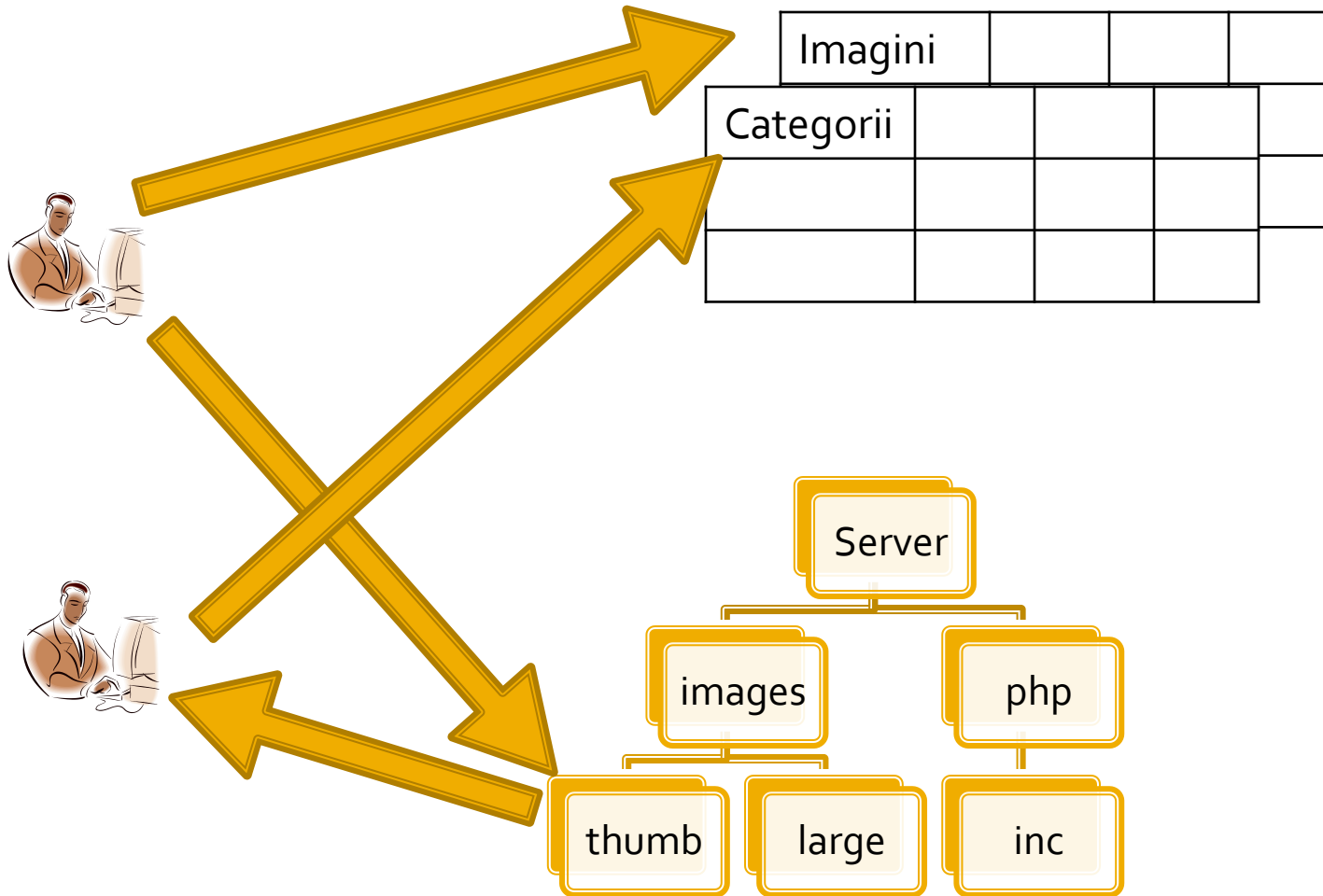


a. aplicatia pentru adaugarea de categorii si afisare a imaginilor (cu alegerea prealabila a categoriei si afisarea listei de imagini format mic)



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
  - 1p – explicatii relativ la functionarea unei anumite secvente de cod



# 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
- 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

# Contact

- Laboratorul de microunde si optoelectronica
- <http://rf-opto.etti.tuiasi.ro>
- [rdamian@etti.tuiasi.ro](mailto:rdamian@etti.tuiasi.ro)