

Curs 8

2016/2017

# Tehnici moderne de proiectare a aplicatiilor web

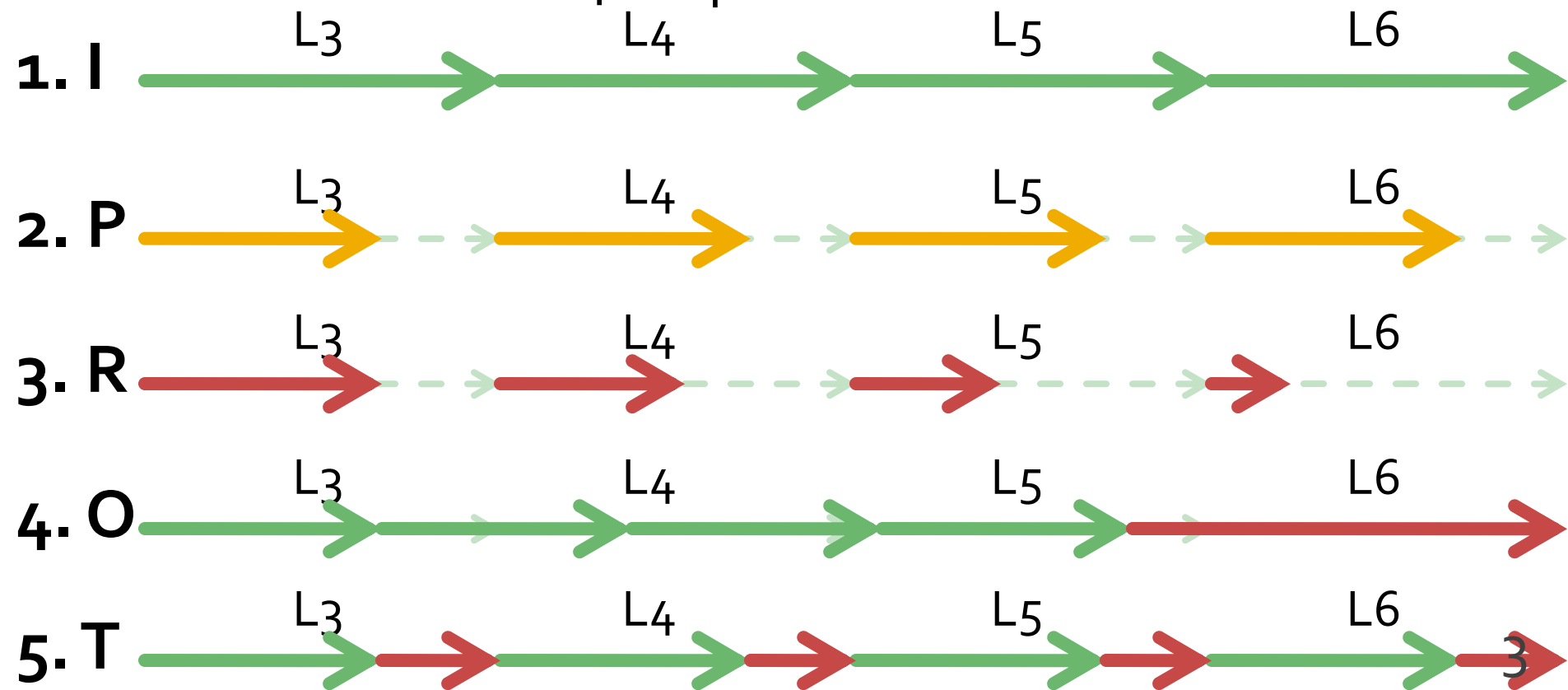
# CURS

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

# ! Important

- Laborator **asincron!**

- recomandat – 4 = Optim



# Laborator 5

# Rezultat

## 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"/>

Trimite

## 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

# Plan aplicatie

- Pe masura ce aplicatia paraseste un fir liniar de executie este necesara introducerea unui plan (graf) al aplicatiei
- Cumparator
  - citirea datelor de pe disc se realizeaza in **antet.php**, comun pentru toate fisierele

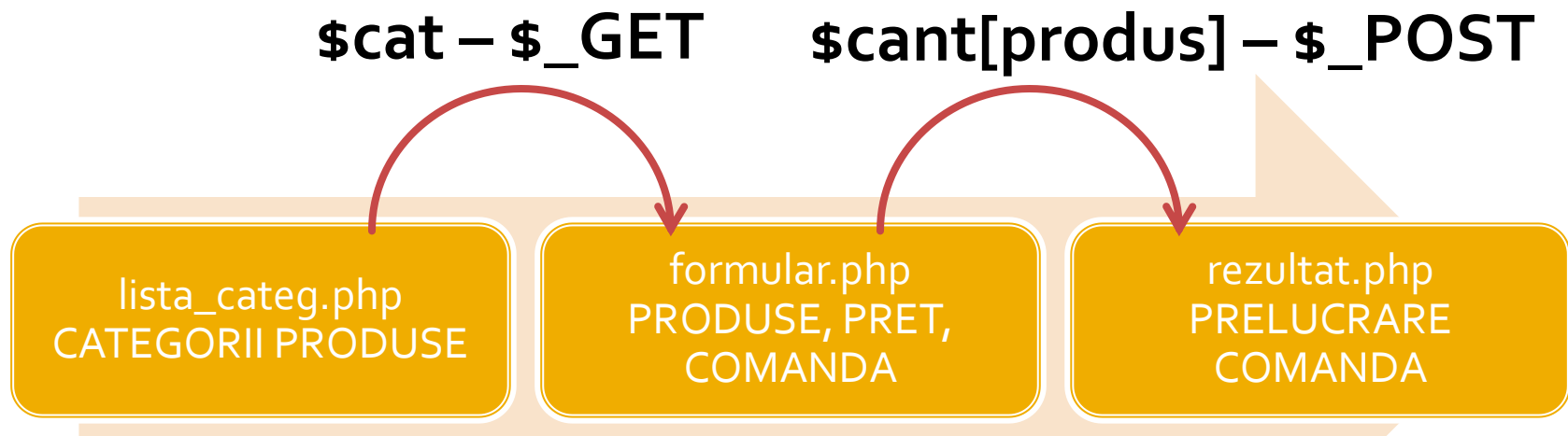
lista\_categ.php  
CATEGORII PRODUSE

formular.php  
PRODUSE, PRET,  
COMANDA

rezultat.php  
PRELUCRARE  
COMANDA

# Plan aplicatie

- Planul aplicatiei trebuie sa cuprinda si informatii relative la:
  - **ce date** se transmit intre diferitele pagini
  - **cum** se transmit datele intre pagini

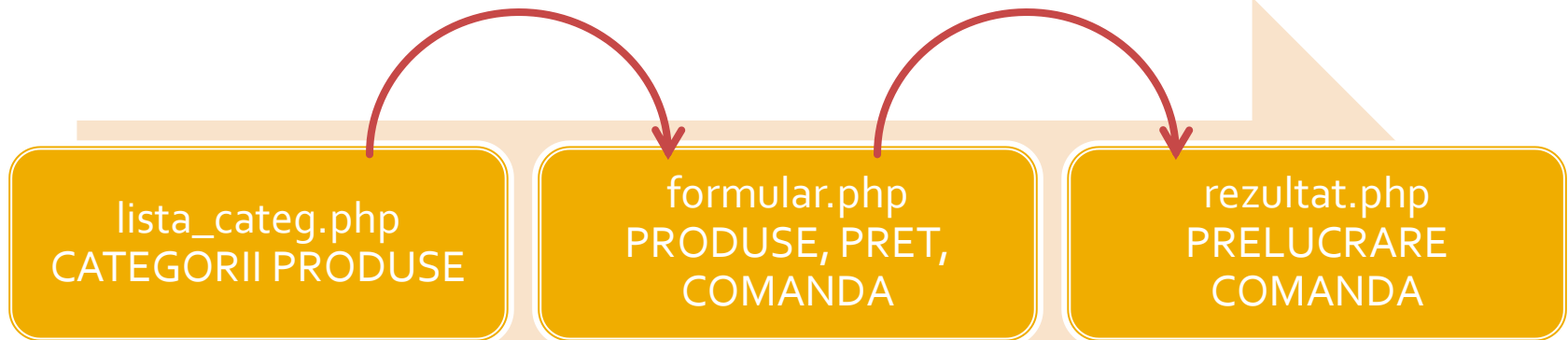


# Plan aplicatie

- Planul aplicatiei – Exemplu
  - lista de categorii va contine “**link-uri active**” deci transmiterea unei singure variabile se face cu **\$\_GET**
  - formularul de comanda transmite date multiple incluse intr-o forma deci transmiterea se face cu **\$\_POST**
- Alegerea \$\_GET/\$\_POST are implicatii:
  - atat in pagina care transmite datele
  - cat si in pagina care le receptioneaza

**\$cat – \$\_GET**

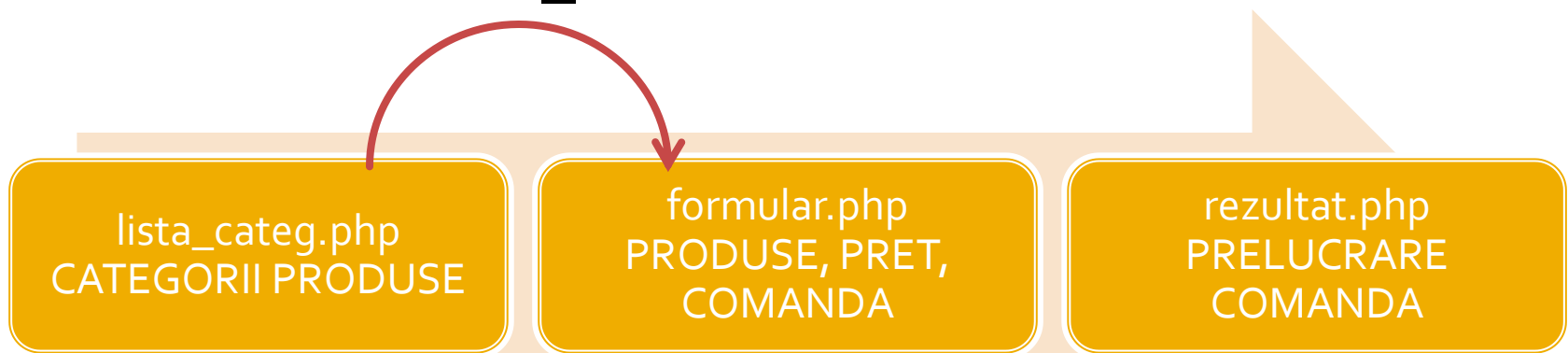
**\$cant[produs] – \$\_POST**





# Link-uri active

- folosite pentru a transmite o **informatie**
- in `lista_categ.php`
  - `<a href="lista_prod.php?categ=<?php echo $cat;?>"> <?php echo $cat;?> </a>`
- are efect in `formular.php`
  - `$_GET['categ']="valoarea $cat corespunzatoare"`  
**\$cat – \$\_GET**



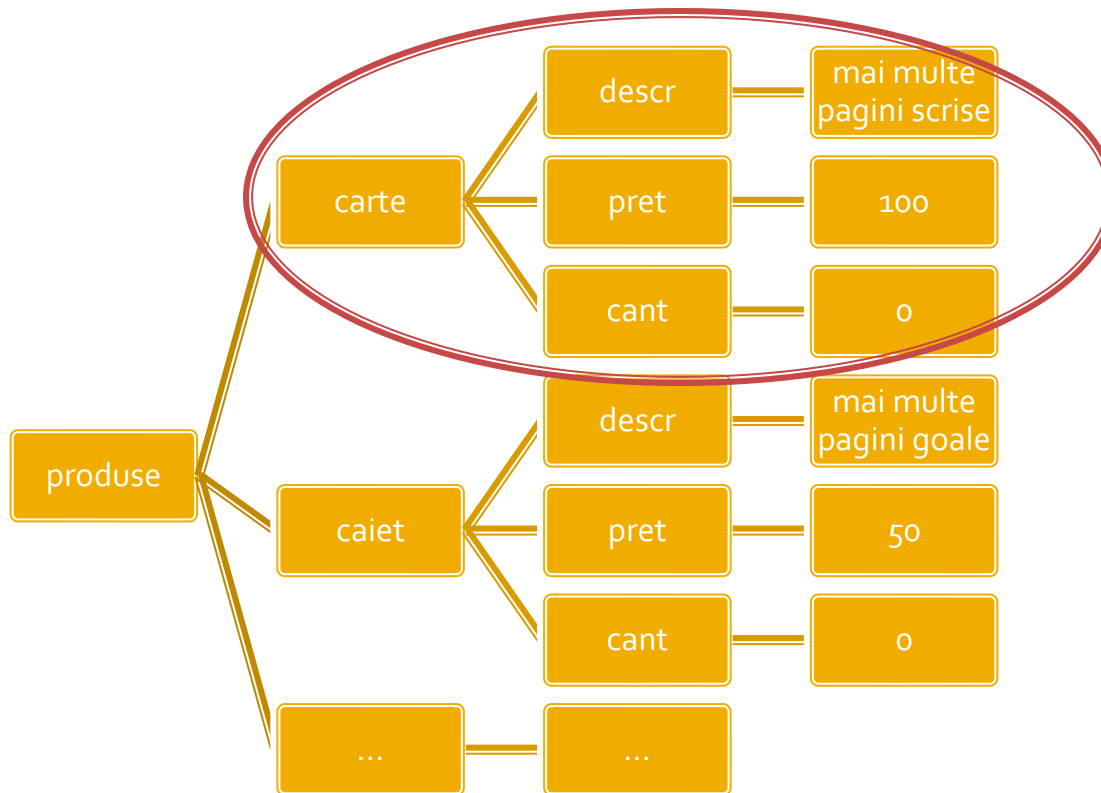
# Laborator 5

- Sa se continue magazinul virtual cu:
  - produsele sunt grupate pe **categorii** de produse
  - sa prezinte utilizatorului o lista de categorii de produse pentru a alege
  - sa prezinte utilizatorului o lista de produse si preturi in categoria aleasa
  - lista de produse si preturi se citeste dintr-un **fișier**
  - se preia comanda si se calculeaza suma totala
- Optional
  - se creaza o pagina prin care vanzatorul poate **modifica** preturile si produsele

# Laborator 5 – Mod de lucru

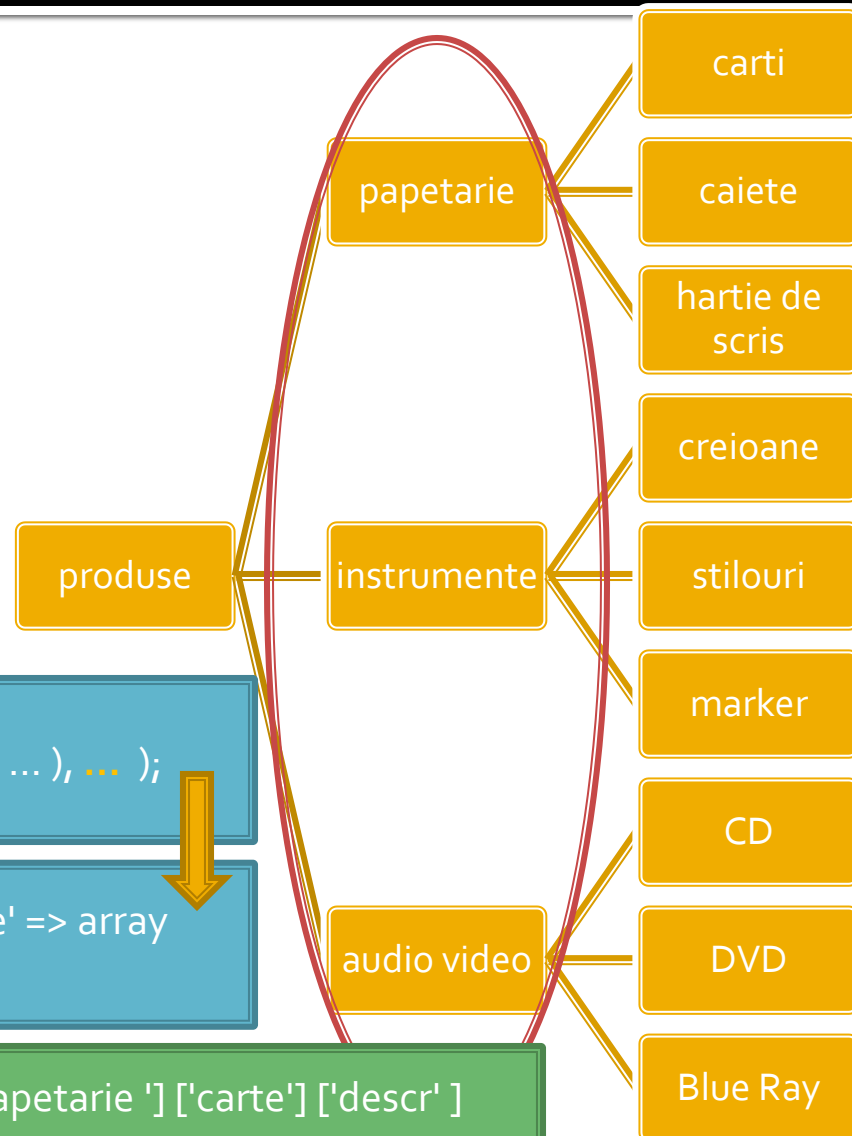
- 1. Se introduce in surse facilitatea template
- 2. Se modifica sursele pentru lucru cu matrici
- 3. Se modifica sursele pentru a citi datele de pe disc (C5 – fisier text)
  - **anterior** se creaza fisierul text sau:
  - **o singura data** se salveaza datele (C5 – S72)
- 4. Se introduce structura suplimentara, categorie
  - se **creaza pagina** de selectie a categoriei, din care se va merge in lista de produse (utilizare \$\_GET – S103)
- 5. Lista de produse si preturi se citeste dintr-un fisier **XML**
- 6. Optional: Se creaza o pagina care sa permita modificarea fisierului text/XML
  - numai pret/descriere, fara adaugare/schimbare produse

# Laborator 4 – Matrice produse



# Laborator 5

- exemplu de grupare
- apare un nivel suplimentar de noduri in arbore
  - deci apare un indice suplimentar in matrice



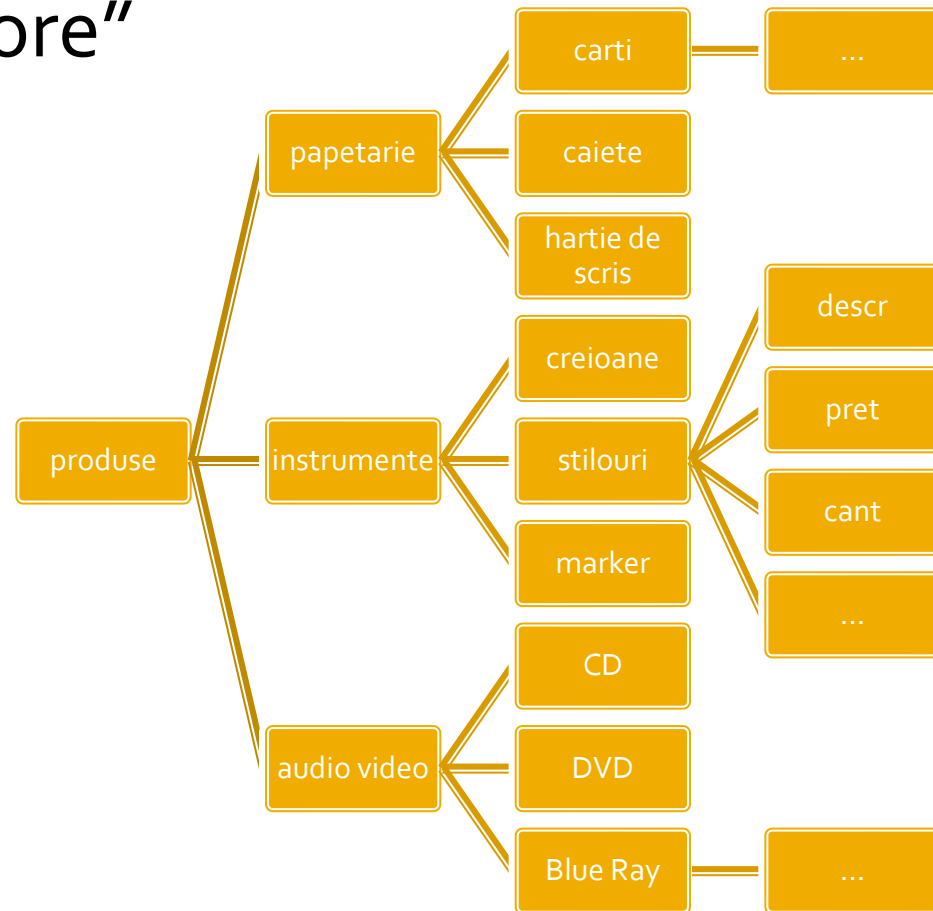
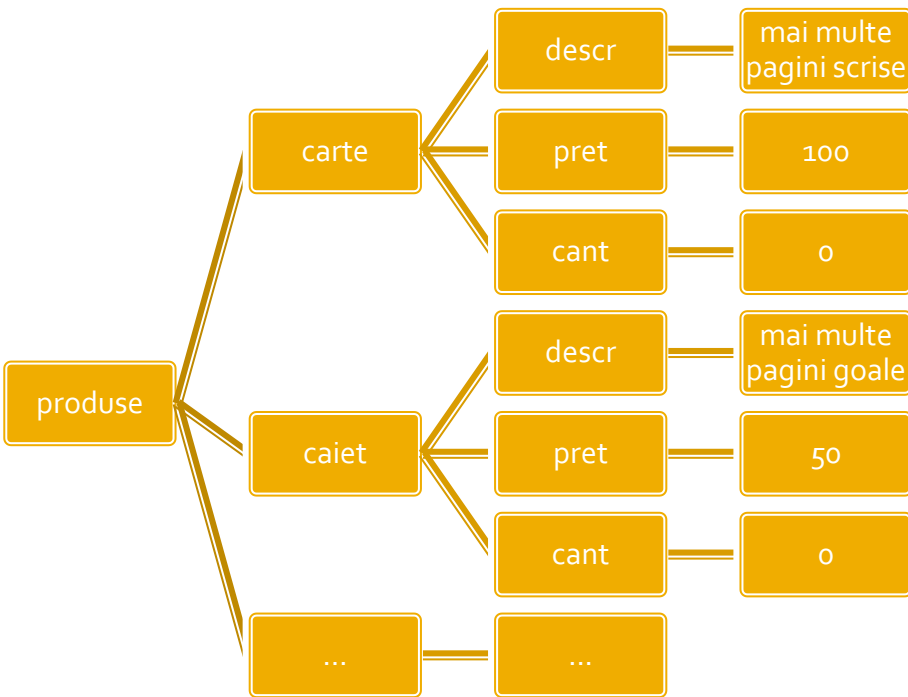
```
$produse = array ( 'carte' => array ("descr" => ... ), ... );
```

```
$produse = array ( 'papetarie' => array ( 'carte' => array ("descr" => ...), ... ) ... );
```

```
$produse['carte']['descr'] ↔ $produse ['papetarie ']['carte']['descr']
```

# Matrici

- adaugare nivel in "arbore"



# Laborator 6+7 MySQL in PHP

# Laborator 6+7

- Sa se continue magazinul virtual cu:
  - produsele sunt grupate pe categorii de produse
  - sa prezinte utilizatorului o lista de grupe de produse pentru a alege
  - sa prezinte utilizatorului o lista de produse si preturi in grupa aleasa
  - lista de produse si preturi se citeste dintr-o baza de date **MySQL**
  - se preia comanda si se calculeaza suma totala
  - **se creaza o pagina prin care vanzatorul poate modifica preturile si produsele**



# Utilizare template - recomandat

- sectiunile repetabile pot fi mutate intr-un fisier separat si introduse cu `require()`
- se identifica zonele comune

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0" align="center">
<tr><td></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

# Utilizare template - recomandat

antet.php

```
<html>
<head>
<title>Magazin online Firma X
SRL</title>
</head>
<body bgcolor="#CCFFFF"><?php
define('PRET_CARTE',100);

//orice cod comun PHP

?><table width="600" border="0"
align="center">
<tr><td></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
<h1>Magazin online Firma X SRL</h1>
```

subsol.php

```
</td></tr>
</table>
</body>
</html>
```

```
<?php require('antet.php');?>
<h2>Lista Produse</h2>
<table border="1">
...
</table>
<?php require('subsol.php');?>
```

# Utilizare template

- antet.php
  - citirea datelor si realizarea matricii \$produse se realizeaza aici
  - acest lucru permite sa se realizeze usor trecerea la alte tehnologii txt → XML → MySql
    - restul fisierelor pot ramane (in mare parte) nemodificate deoarece se bazeaza pe utilizarea matricii \$produse, indiferent cum e ea realizata
- subsol.php
  - se poate utiliza la realizarea interfetei pentru vanzator
  - se salveaza matricea \$produse in formatul necesar tehnologiei utilizate

# Plan aplicatie – Cumparator

- Pe masura ce aplicatia paraseste un fir liniar de executie este necesara introducerea unui plan (graf) al aplicatiei
- Cumparator
  - citirea fisierului XML (accesarea bazei de date) se realizeaza in antet.php, comun pentru toate fisierele

lista\_categ.php  
CATEGORII PRODUSE

formular.php  
PRODUSE, PRET,  
COMANDA

rezultat.php  
PRELUCRARE  
COMANDA

# Plan aplicatie – Vanzator

- Aparitia aplicatiei pentru vanzator
  - introduce un fir paralel de executie cu necesitatea alegerii initiale: cumparator/vanzator
  - aduce posibilitatea scrierii fisierului XML
  - diverse operatii de scriere
    - introducere categorie de produse
    - introducere produs nou intr-o categorie existenta
    - modificare produs existent
  - modificarea fisierului implica 2 actiuni:
    - colectare date
    - prelucrare

# Fisier unic pentru colectare SI prelucrare date

- De multe ori se prefera aceasta varianta
- Permite pastrarea unitara a tuturor operatiilor pentru indeplinirea unei actiuni
  - acces mai simplu
  - usurinta la programare
  - evitarea erorilor: File does not exist: D:/Server/...
- Acelasi fisier e folosit initial pentru a colecta date si apoi, daca se detecteaza prezenta acestora, pentru prelucrarea lor

# Fisier unic pentru colectare SI prelucrare date

- Fisierul de receptie pentru <form> va fi fisierul curent
- se recomanda utilizarea variabilei globale `$_SERVER['SCRIPT_NAME']`
  - flexibilitate la redenumirea fisierelor
- alternativ `$_SERVER['PHP_SELF']` nu este recomandata
  - probleme de securitate
- Sectiunea de colectare date se afiseaza numai in absenta datelor

```
<form action="<?php echo $_SERVER['SCRIPT_NAME'];?>" method="post">  
<p><input name="date_ok" type="submit" value="Trimite" /></p>  
</form>
```

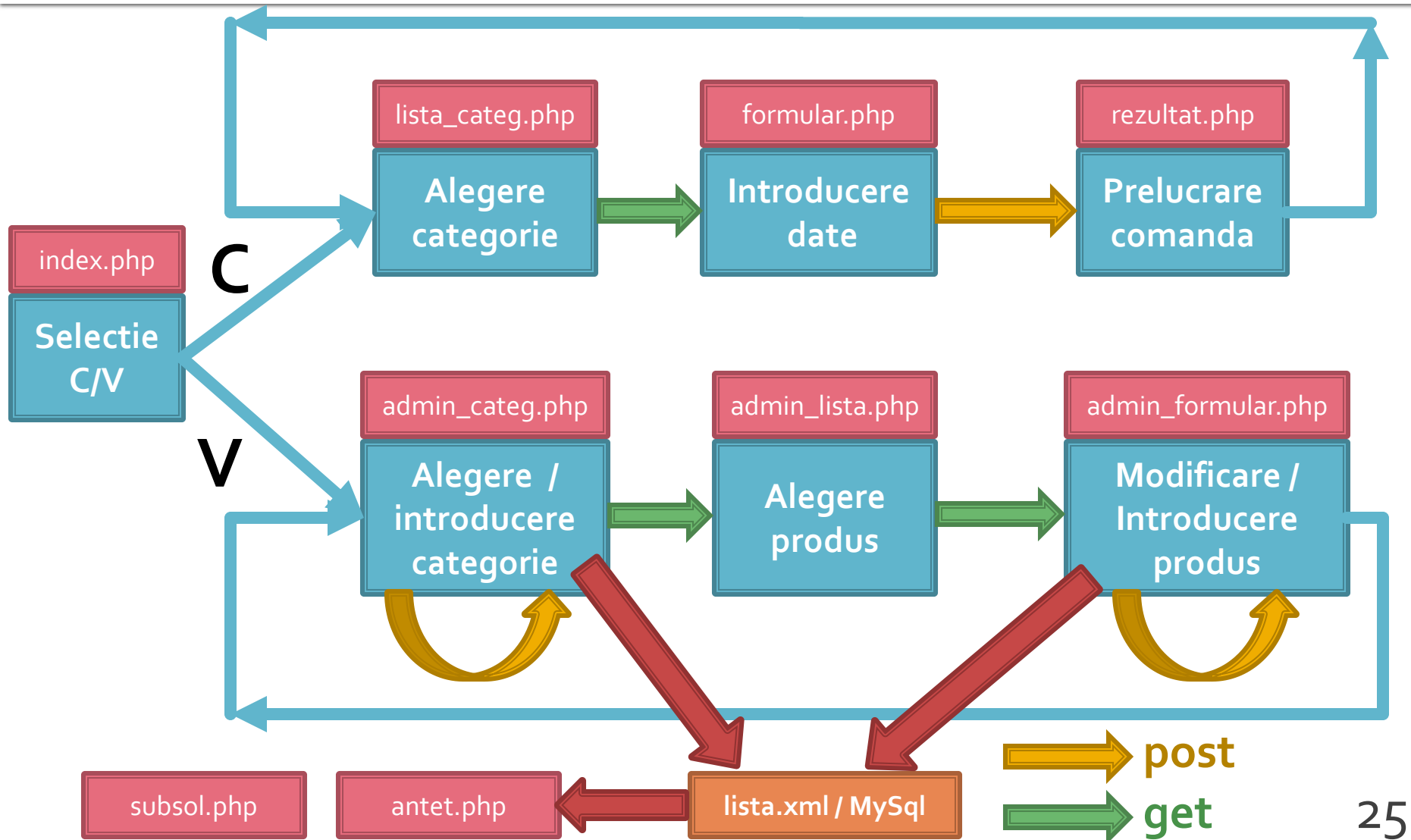
# Fisier unic pentru colectare SI prelucrare date

- Detectia existentei datelor se face prin verificarea existentei ( isset(\$variabila) ) valorilor introduse
  - eventual pentru un plus de protectie se poate verifica si continutul lor

```
if (isset($_POST[" date_ok "]))
{ //date trimise
  if ($_POST[" date_ok "]=="Trimite" )
    { //date trimise de fisierul curent
      //prelucrare
    }
}
else
{
  //colectare date
  <form action="<?php echo $_SERVER['SCRIPT_NAME '];?>" method="post">
  <p><input name="date_ok" type="submit" value="Trimite" /></p></form>
}
```



# Plan aplicatie



# Rezultat (comparator)

## 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

### Finalizati 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"/>



# Laborator 6

- Sa se continue magazinul virtual cu:
  - produsele sunt grupate pe categorii de produse
  - sa prezinte utilizatorului o lista de grupe de produse pentru a alege
  - sa prezinte utilizatorului o lista de produse si preturi in grupa aleasa
  - lista de produse si preturi se citeste dintr-o baza de date **MySQL**
  - se preia comanda si se calculeaza suma totala
  - **se creaza paginile prin care vanzatorul poate modifica preturile, produsele, categoriile**

MySql

# Accesul la metode externe de stocare eficiente a datelor

# Normalizare

- Normalizarea asigura:
  - stocarea eficienta a datelor
  - prelucrarea eficienta a datelor
  - integritatea datelor
- Trei nivele de normalizare
- Eliminarea datelor redundante

OrderID	CustomerID	OrderDate	Items	OrderTotal
1	CACTU	1/1/1999	3 Zaanse koeken, 1 Tarte au sucre	\$89.70
2	BSBEV	1/5/1999	4 Mozzarella di Giovanni	\$139.20
3	SUPRD	5/2/1999	3 Ravioli Angelo, 6 Tofu	\$198.06

MySql – Recapitulare rapida

# Relatii in Bazele de date

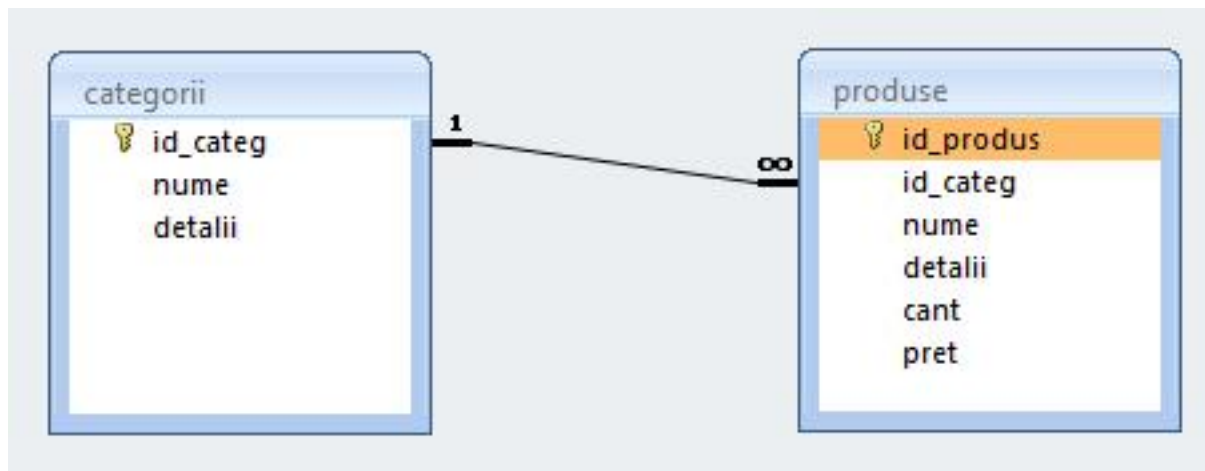
# Relatii in Bazele de date

- 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

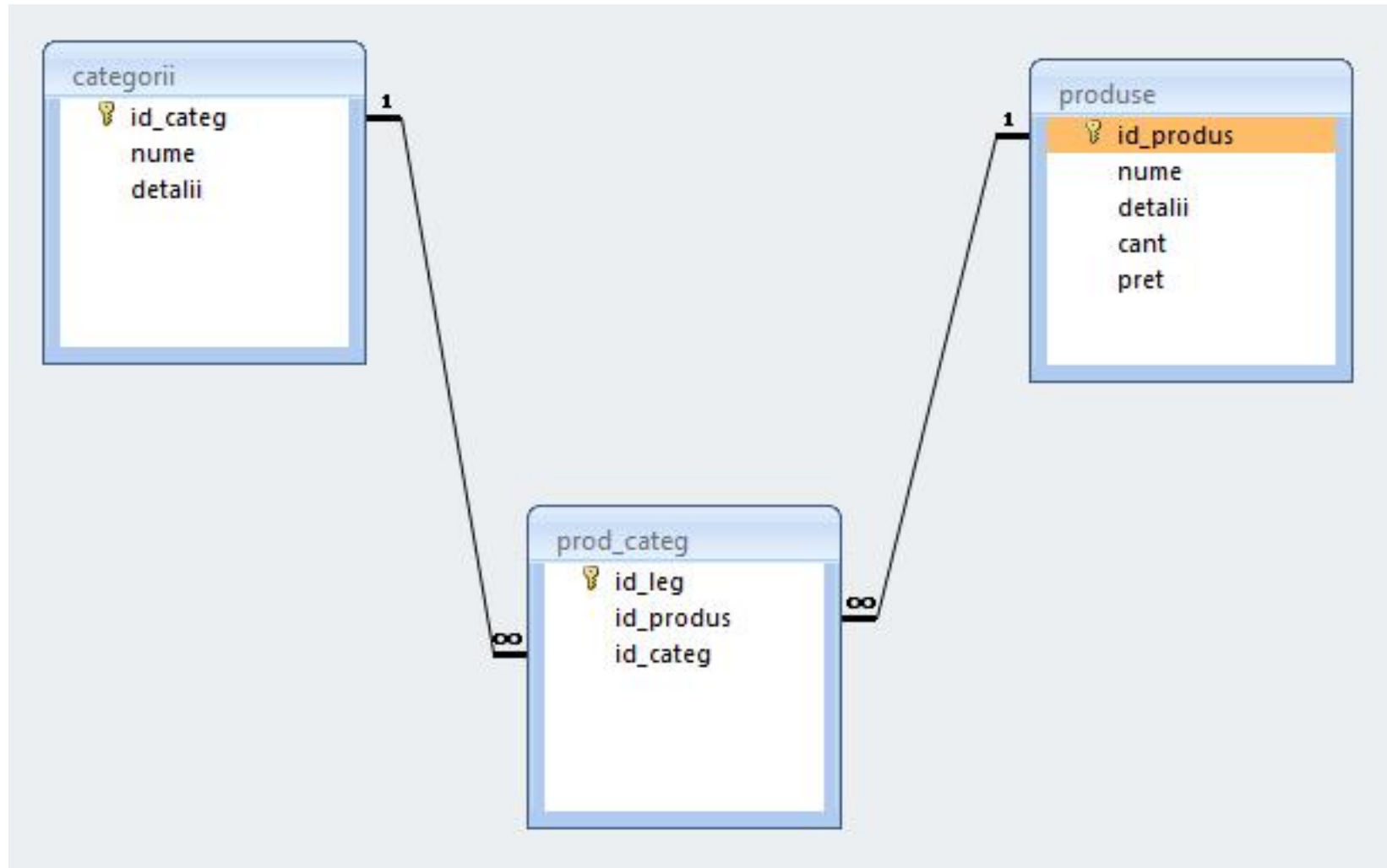
- Legaturile implementata
  - One to Many
  - in tabelul "produse" apare cheia externa (foreign key): "id\_categ"



# 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\_produs" (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 )



# 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 )

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

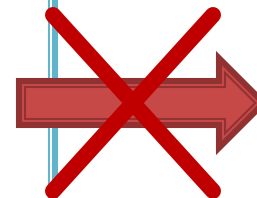
Functii de acces la structura



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

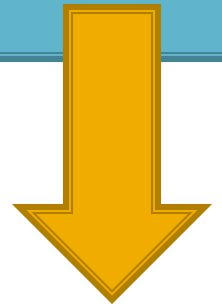
```
<?php
do {?>
<tr>
    <td><?php echo $index; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Code']; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Name']; ?>&nbsp;  </td>
    <td><?php echo $ row_result ['Population']; ?>&nbsp;  </td>
</tr>
<?php
    $index++;
}
while ($ row_result = mysql_fetch_assoc($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

## 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

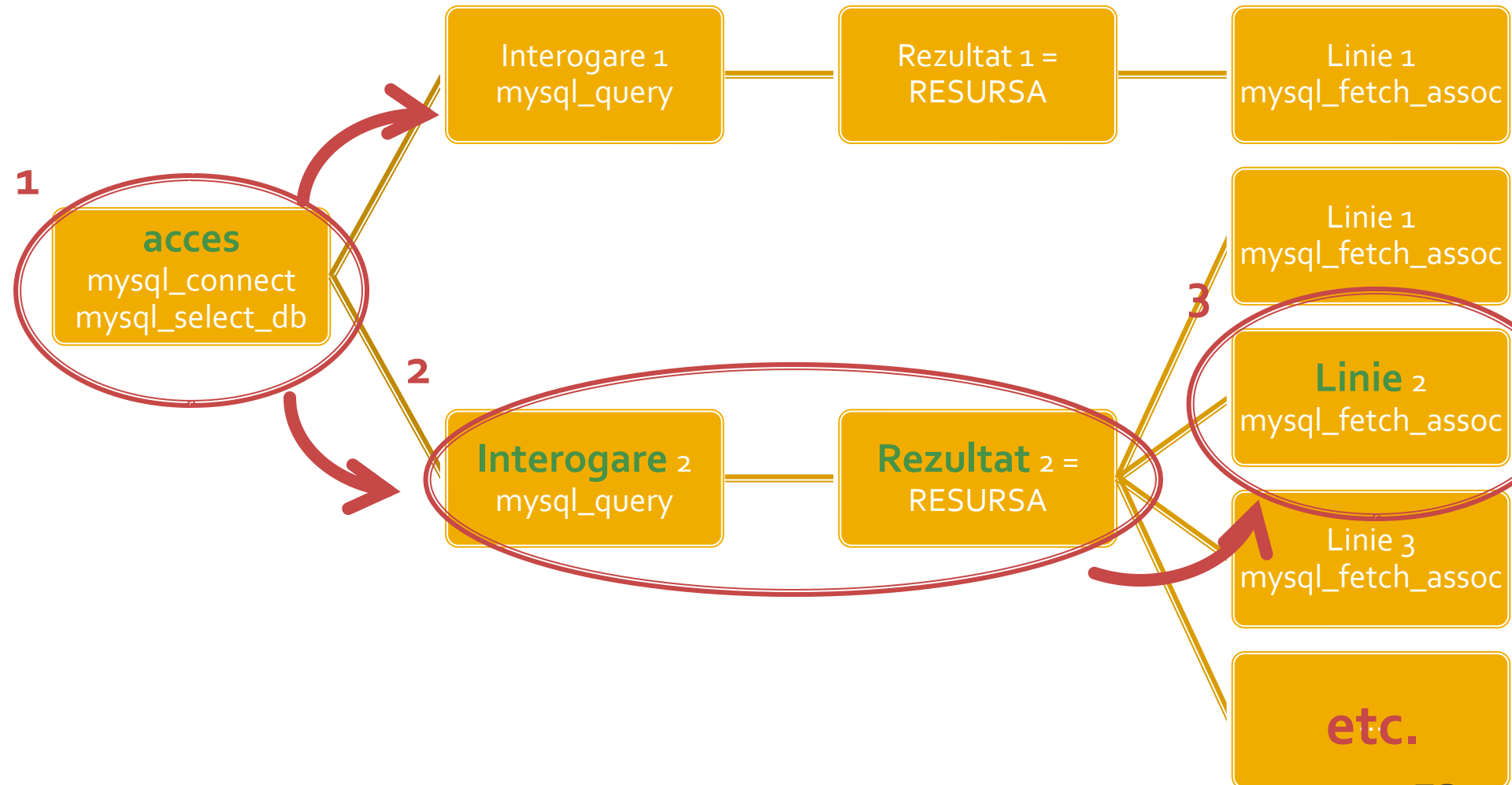
- o singura interogare SQL, unirea tabelelor lasata in baza server-ului MySQL

```
$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 ( "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = $res->fetch_assoc();
echo $row['_msg'];

$mysql = mysqli_connect("example.com", "user", "password");
mysqli_select_db("test");
$res = mysqli_query("SELECT ' for new developments.' AS _msg FROM DUAL", $mysql);
$row = mysqli_fetch_assoc($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	....
-------------	-------	-------------	------

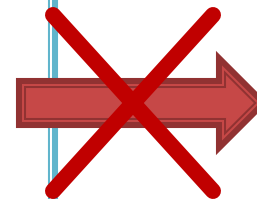
Functii de acces la structura



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)

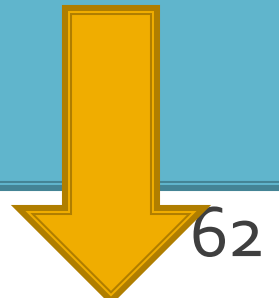
```
//$conex= mysql_connect($hostname, $username, $password);
//mysql_select_db($database, $conex);
$conex = mysqli_connect($hostname, $username, $password, $database);

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
        LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
//$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('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

# 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)

# 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
    - incepand cu MySql 5.6.4 este introdus index FULLTEXT
  - **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 pierd**, 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

# 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 conditile 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 tabel<sub>1</sub> SET ... SELECT ... FROM tabel<sub>1</sub> 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

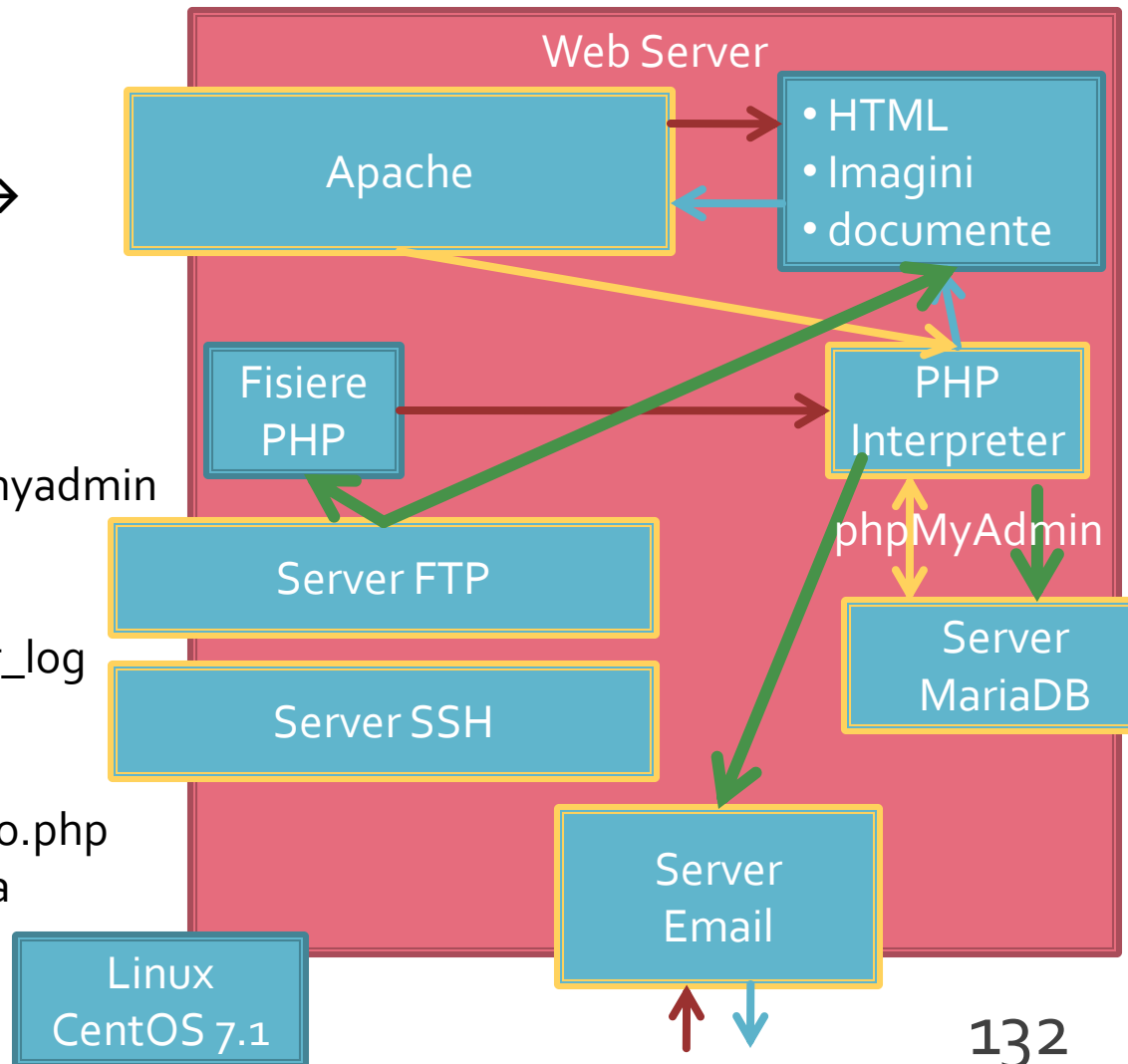
MySql – Server Centos 7.1

# 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)



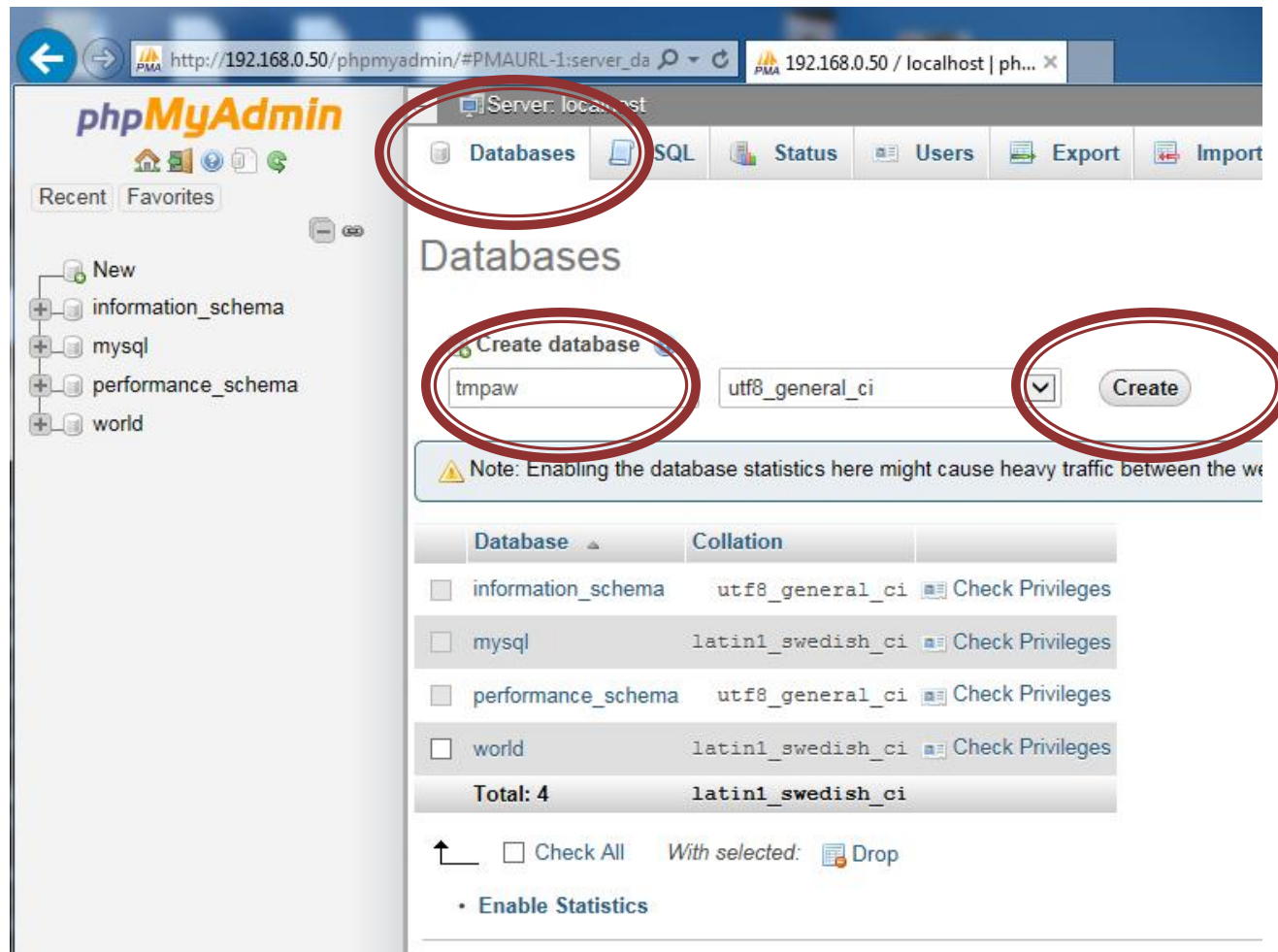
# PhpMyAdmin

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`. The interface includes a navigation menu on the left with options like 'Databases', 'SQL', 'Status', 'Users', 'Export', 'Import', 'Settings', 'Replication', 'Variables', 'Charsets', and 'Engines'. The main content area is divided into 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, and 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, and PHP version: 5.4.16.
- phpMyAdmin:** Lists version information: 4.4.15.1, and links to Documentation, Wiki, Official Homepage, Contribute, Get support, and 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' and the collation 'utf8\_general\_ci' entered. 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	<a href="#">Check Privileges</a>
<input type="checkbox"/> mysql	latin1_swedish_ci	<a href="#">Check Privileges</a>
<input type="checkbox"/> performance_schema	utf8_general_ci	<a href="#">Check Privileges</a>
<input type="checkbox"/> world	latin1_swedish_ci	<a href="#">Check Privileges</a>
<b>Total: 4</b>	<b>latin1_swedish_ci</b>	

↑  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 circled in red. Below it, a message states 'No tables found in database.' The 'Create table' button is also circled in red. The 'Name' field contains the text 'categorii' and is circled in red. The 'Number of columns' field contains the number '3' and is circled in red. The 'Go' button at the bottom right is circled in red. The left sidebar shows a tree view of databases, with 'performance\_schema' and 'tmpaw' circled in red.



# 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' in the 'tmpaw' database. The table name 'categorii' is circled in red. The 'Add 1 column(s) Go' button is also circled in red. The table structure is displayed below, with columns 'id\_categ', 'nume', and 'detalii' listed. The 'id\_categ' column is of type 'INT' and has a 'None' default value, all circled in red. The 'nume' column is of type 'VARCHAR' with a length of '45' and a 'None' default value, with the length '45' circled in red. The 'detalii' column is of type 'VARCHAR' with a length of '150' and a 'None' default value. The interface includes a sidebar with 'Recent' and 'Favorites' sections, and a top navigation bar with 'Browse', 'Structure', 'SQL', 'Search', 'Import', and 'Privileges' options.

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

# Introducere coloane

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

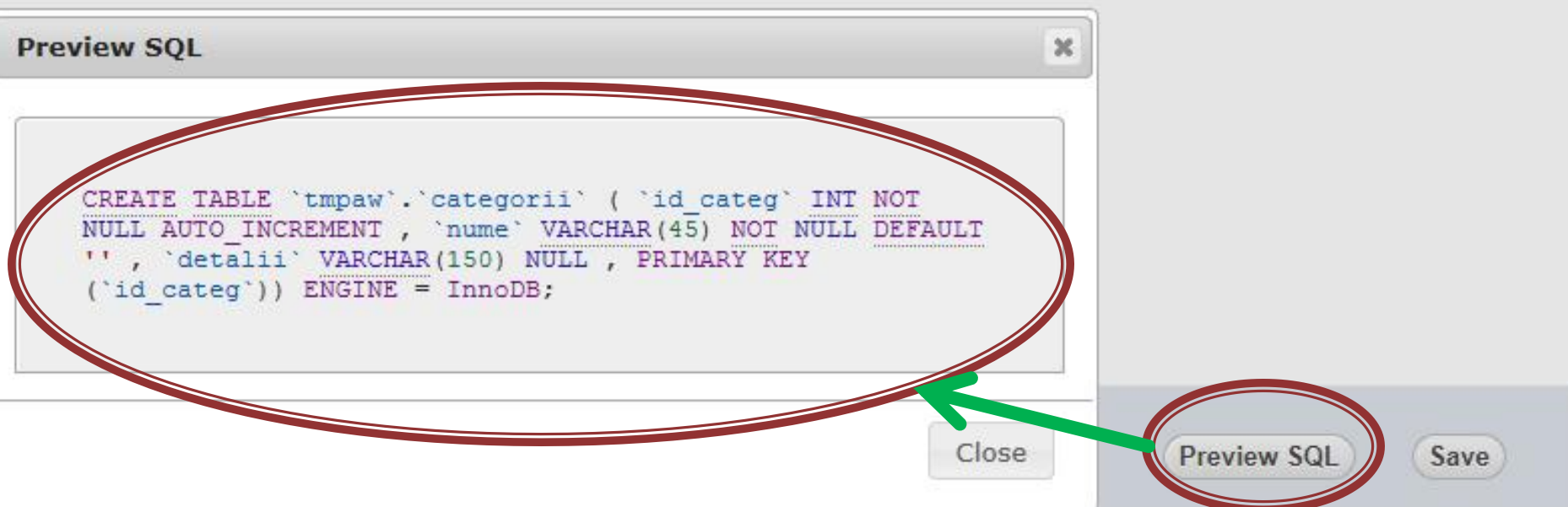
Table name:  Add  column(s)

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 table structure is displayed below, with columns: id\_produc, id\_categ, nume, detalii, cant, and pret. The 'Type' column for 'pret' is set to 'FLOAT'.

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	C
id_produc	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 displays 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 'insert as new row' dropdown is set to 'insert as new row', and the 'and then' dropdown is set to 'Go back to previous page'. 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 shows the phpMyAdmin interface for a database named 'tmpaw'. The 'categoriasii' table is selected in the sidebar. The 'Browse' button is highlighted with a red circle. The table structure is shown with 3 rows. The table data is as follows:

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

The 'Query results operations' section at the bottom shows options for Print view, Export, Display chart, and Create view.

# Introducere date initiale (SQL)

- Tabel → SQL → completare → Go

The screenshot shows the phpMyAdmin interface for a MySQL database named 'mpaw'. The 'produse' table is selected in the left sidebar. The 'SQL' tab is active, and the 'Go' button is highlighted. The SQL query editor contains the following INSERT statement:

```
1 INSERT INTO `produse` (`id_produș`, `id_categ`, `nume`, `detalii`, `cant`, `pret`)
2 VALUES
3 (1,1,'carte','mai multe pagini scrise legate',0,100),
4 (2,1,'caiet','mai multe pagini goale legate',0,75),
5 (3,1,'hartie scris','mai multe pagini goale NElegate',0,50),
6 (4,2,'penar','loc de depozitat instrumente de scris',0,150),
7 (5,2,'stilou','instrument de scris albastru',0,125),
8 (6,2,'creion','instrument de scris gri',0,25),
9 (7,3,'cd','canta',0,50),
10 (8,3,'dvd','vizual',0,100),
11 (9,3,'blue ray','vizual extrem',0,500);
```

The 'Columns' list on the right shows the table structure:

Columns
id_produș
id_categ
nume
detalii
cant
pret

At the bottom of the interface, the 'Go' button is highlighted, indicating the execution of the query.

# Tabel produse

The screenshot shows the phpMyAdmin interface for a database named 'tmpaw'. The 'Table: produse' view is active, displaying the table structure and data. The 'Structure' tab is selected, and the 'produse' table is highlighted in the left sidebar. The table contains 9 rows of data, including columns for 'id\_produs', 'id\_categ', 'nume', 'detalii', 'cant', and 'pret'.

Showing rows 0 - 8 (9 total, Query took 0.0003 seconds.)

```
SELECT * FROM `produse`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP Code ] [ Refresh ]

Show all | Number of rows: 25 | Filter rows: Search this table

Sort by key: None

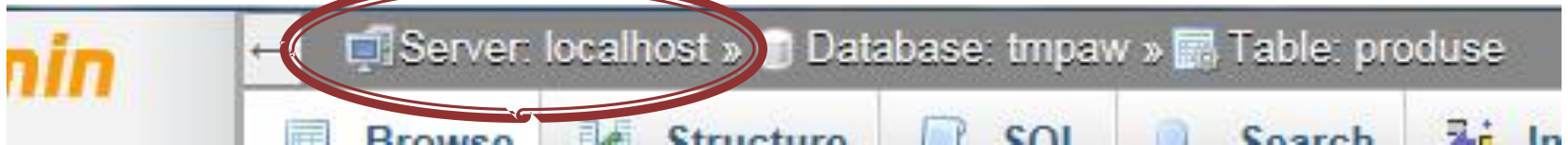
+ Options		id_produs	id_categ	nume	detalii	cant	pret
<input type="checkbox"/>	Edit Copy Delete	1	1	carte	mai multe pagini scrise legate	0	100
<input type="checkbox"/>	Edit Copy Delete	2	1	caiet	mai multe pagini goale legate	0	75
<input type="checkbox"/>	Edit Copy Delete	3	1	hartie scris	mai multe pagini goale NElegate	0	50
<input type="checkbox"/>	Edit Copy Delete	4	2	penar	loc de depozitat instrumente de scris	0	150
<input type="checkbox"/>	Edit Copy Delete	5	2	stilou	instrument de scris albastru	0	125
<input type="checkbox"/>	Edit Copy Delete	6	2	creion	instrument de scris gri	0	25
<input type="checkbox"/>	Edit Copy Delete	7	3	cd	canta	0	50
<input type="checkbox"/>	Edit Copy Delete	8	3	dvd	vizual	0	100
<input type="checkbox"/>	Edit Copy Delete	9	3	blue ray	vizual extrem	0	500

Check All | With selected: Edit Delete Export



# Adaugare utilizator

- Server → Users → Add user

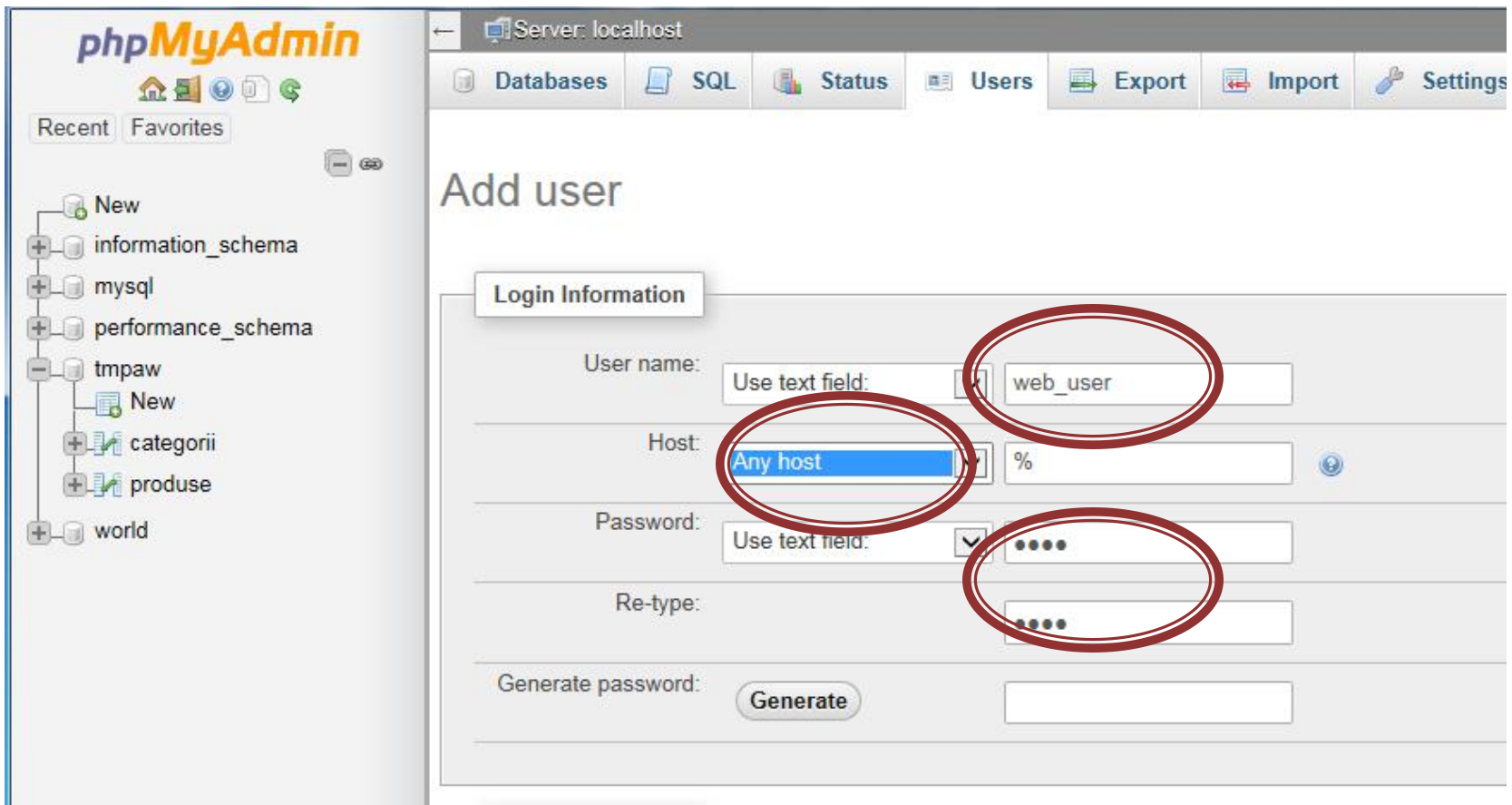


A screenshot of the phpMyAdmin 'Users overview' page. The navigation bar at the top shows 'Server: localhost' circled in red. Below it, the 'Users' menu item is also circled in red. The main content area displays a table of users with columns for 'User name', 'Host', 'Password', 'Global privileges', '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: web\_user
- Host: Any host
- Password: [masked]
- Re-type: [masked]

Red circles highlight the 'User name', 'Host', 'Password', and 'Re-type' fields, indicating the information being entered for the new user.

# 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 shown below, with the 'Edit Privileges' link for the 'web\_user' user circled in red.

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

# Drepturi de acces

- Database → nume → Go

The screenshot shows the phpMyAdmin interface for a MySQL server on localhost. The left sidebar displays a tree view of databases, including 'information\_schema', 'mysql', 'performance\_schema', 'tmpaw', and 'world'. The main content area is titled 'Edit Privileges: User 'web\_user'@'%' and features a navigation bar with buttons for 'Global', 'Database', 'Change password', and 'Login Information'. The 'Database' button is circled in red. Below the navigation bar, there is a section for 'Database-specific privileges' with a table showing a list of databases. The table has columns for 'Database', 'Privileges', 'Grant', 'Table-specific privileges', and 'Action'. The table content is currently empty, showing 'None'. At the bottom, there is a text input field labeled 'Add privileges on the following database(s):' with a dropdown menu containing 'mysql', 'tmpaw', and 'world', which is also circled in red.

Server: localhost

Databases SQL Status Users Export Import Settings

Global Database Change password Login Information

Edit Privileges: User 'web\_user'@'%'

Database-specific privileges

Database	Privileges	Grant	Table-specific privileges	Action
None				

Add privileges on the following database(s):

mysql  
tmpaw  
world

# Drepturi de acces

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

The screenshot shows the phpMyAdmin interface for editing 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 'Structure' tab is selected for the table 'produse'. The table structure is as follows:

#	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





The 'id\_categ' column is highlighted with a green circle, and the 'Index' button in the 'Action' column for 'id\_categ' is circled in red. The 'Structure' tab is also circled in red. The 'produse' table is selected in the left sidebar, also circled in red.

# 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

# 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

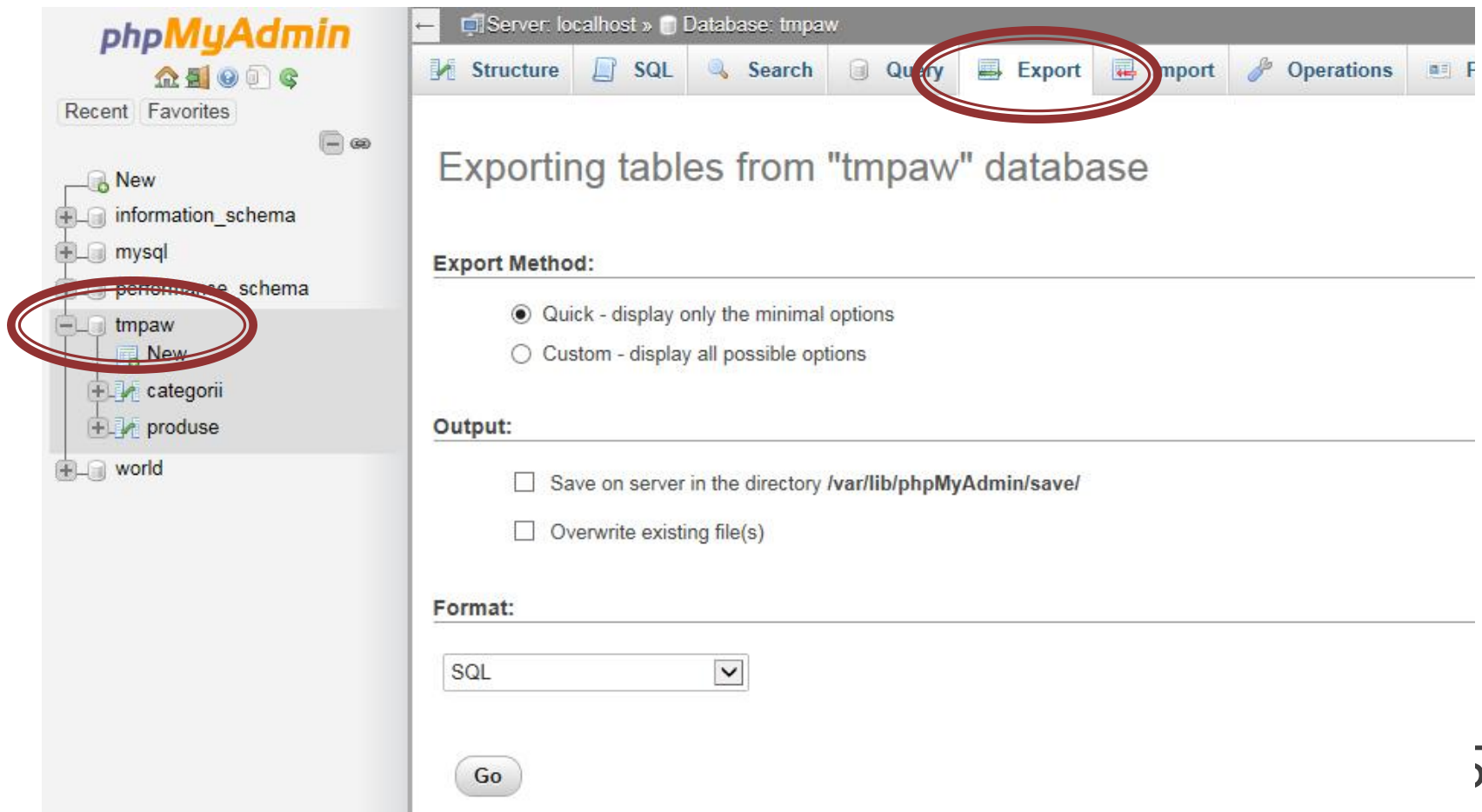
```
CREATE DATABASE IF NOT EXISTS tmpaw;  
USE tmpaw;
```

```
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);
```

# Backup

- Nume (tabel sau baza de date) → Export



The screenshot displays the phpMyAdmin interface. On the left sidebar, the database 'tmpaw' is selected and circled in red. The main panel shows the 'Export' tab selected in the top navigation bar, also circled in red. The main content area is titled 'Exporting tables from "tmpaw" database' and contains the following options:

- 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 (selected in the dropdown menu)

A 'Go' button is located at the bottom of the form.

# Restore

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

The screenshot shows the phpMyAdmin interface. On the left sidebar, the database 'tmpaw' is selected and circled in red. The main content area is titled 'Importing into the database "tmpaw"'. The 'Import' button in the top navigation bar is also circled in red. Under the 'File to Import:' section, the 'Browse...' button is circled in red. The 'Character set of the file:' 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' on the 'id\_categ' column. The index settings are:

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

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

id_produ	id_c
1	
2	
3	
4	
5	
6	
7	
8	
9	



# Tabel Prognose

The screenshot shows the MySQL Table Editor interface 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	

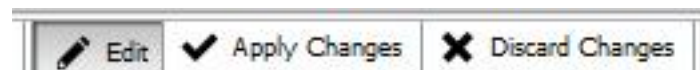
The 'Indices' tab is also active, showing a PRIMARY index with the following settings:

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

The 'Apply Changes', 'Discard Changes', and 'Close' buttons are visible at the bottom of the dialog.

# 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 shows the MySQL Query Browser interface. The SQL Query Area contains the query: `1 SELECT * FROM produse p;`. The result set is displayed as a table with the following data:

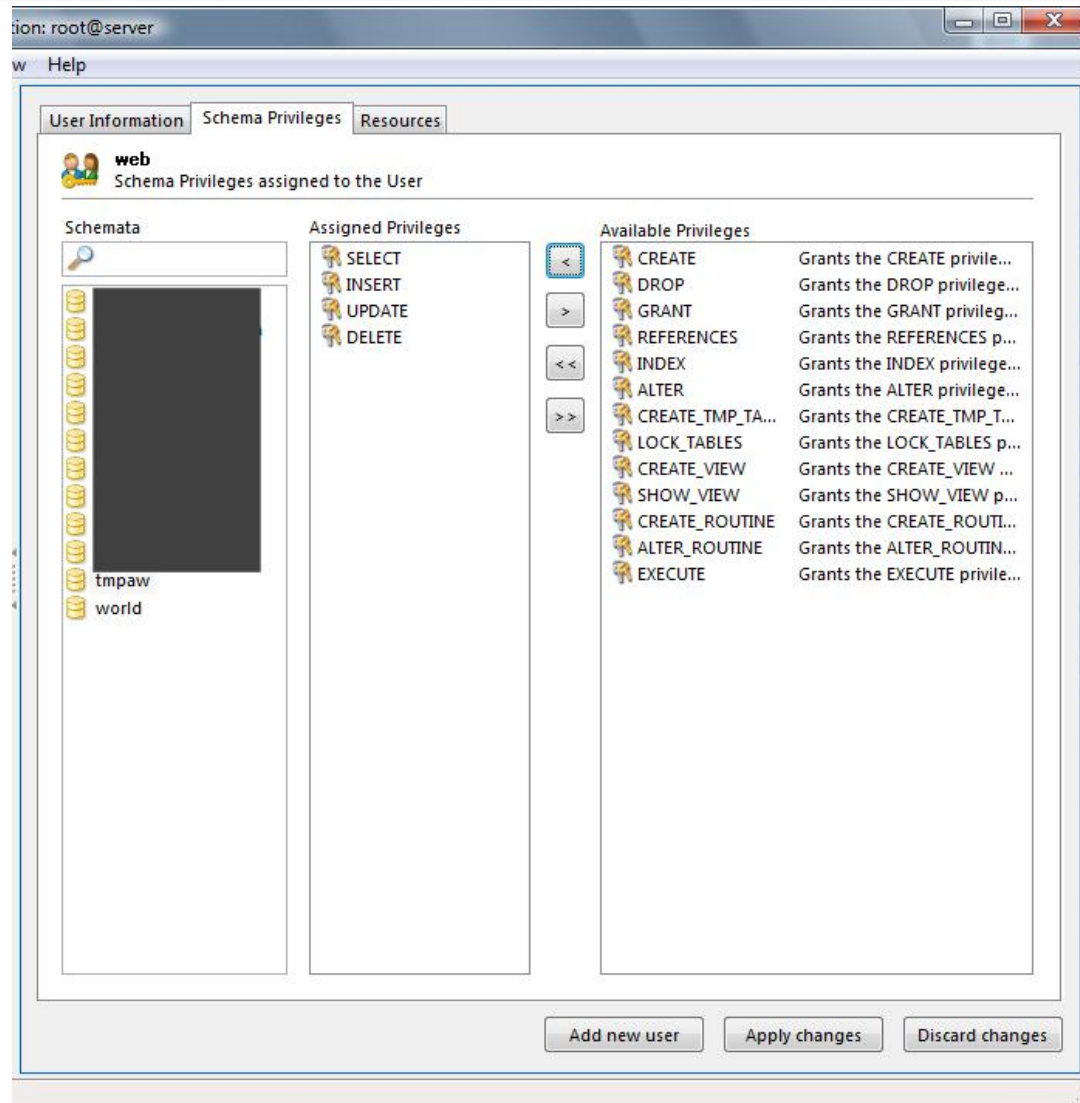
id_produș	id_categ	nume	detalii	cant	pret
1	1	carte	mai multe pagini scrise legate	0	100
2	1	caiet	mai multe pagini goale legate	0	75
3	1	hartie scris	mai multe pagini goale NElegate	0	50
4	2	penar	loc de depozitat instrumente de scris	0	150
5	2	stilou	instrument de scris albastru	0	125
6	2	creion	instrument 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 shows a Schemata panel on the right with a tree view containing 'tmpaw', 'categorii', 'produse', and 'world'. A 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

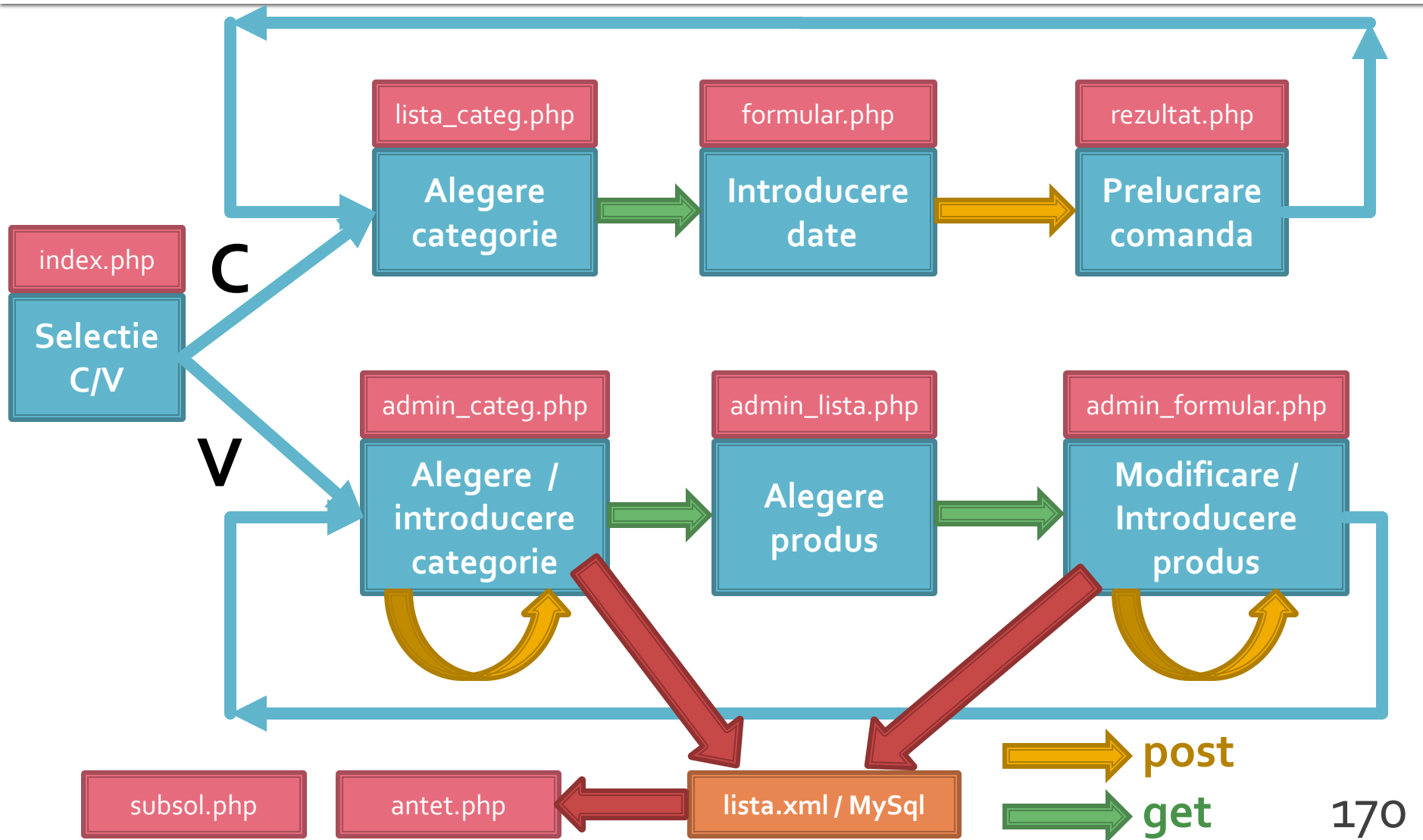
# Laborator 6



# Laborator 6+7

- Sa se continue magazinul virtual cu:
  - produsele sunt grupate pe categorii de produse
  - sa prezinte utilizatorului o lista de grupe de produse pentru a alege
  - sa prezinte utilizatorului o lista de produse si preturi in grupa aleasa
  - lista de produse si preturi se citeste dintr-o baza de date **MySQL**
  - se preia comanda si se calculeaza suma totala
  - **se creaza o pagina prin care vanzatorul poate modifica preturile si produsele**

# Plan aplicatie



# Rezultat (comparator)

### 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

#### Finalizati 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

 post  
 get

# 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"/>

 post  
 get

# 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' button is highlighted, indicating that the changes are ready to be saved.

# 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 editor displays 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 active, showing a PRIMARY index named 'PRIMARY' of kind 'PRIMARY' and type 'BTREE'. The index columns are 'id\_producs'. The 'Apply Changes' button is highlighted.

# Laborator 6 – Mod de lucru

- Se continua lucrul la aplicatie (L5)
- Se recomanda laboratorul **asincron** – S2
- Se poate folosi fisierul cu surse cpypaste.txt  
(site-<http://rf-opto.etti.tuiasi.ro>)

# Laborator 6 – Mod de lucru

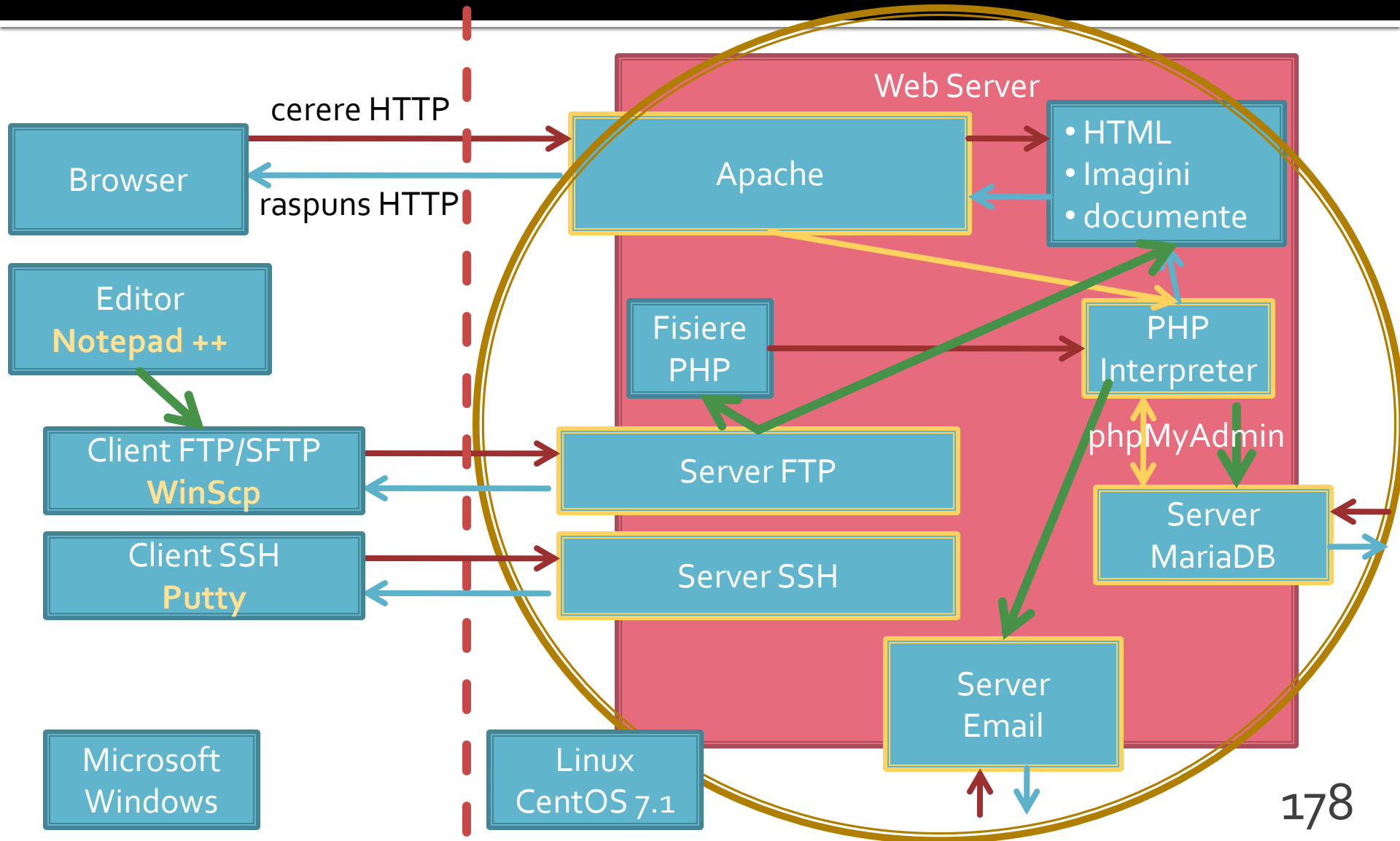
- Se ia o decizie relativ la relatia dintre produse si categorii (S63-S67)
  - One to Many
  - Many to Many
- Se creaza cele 2(3) tabele corespunzatoare
- Se populeaza cu date
- Se actualizeaza planul aplicatiei pentru a corespunde cu aplicatia proprie
  - nume de fisiere, tipuri de transfer a datelor



# Laborator 6 – Mod de lucru

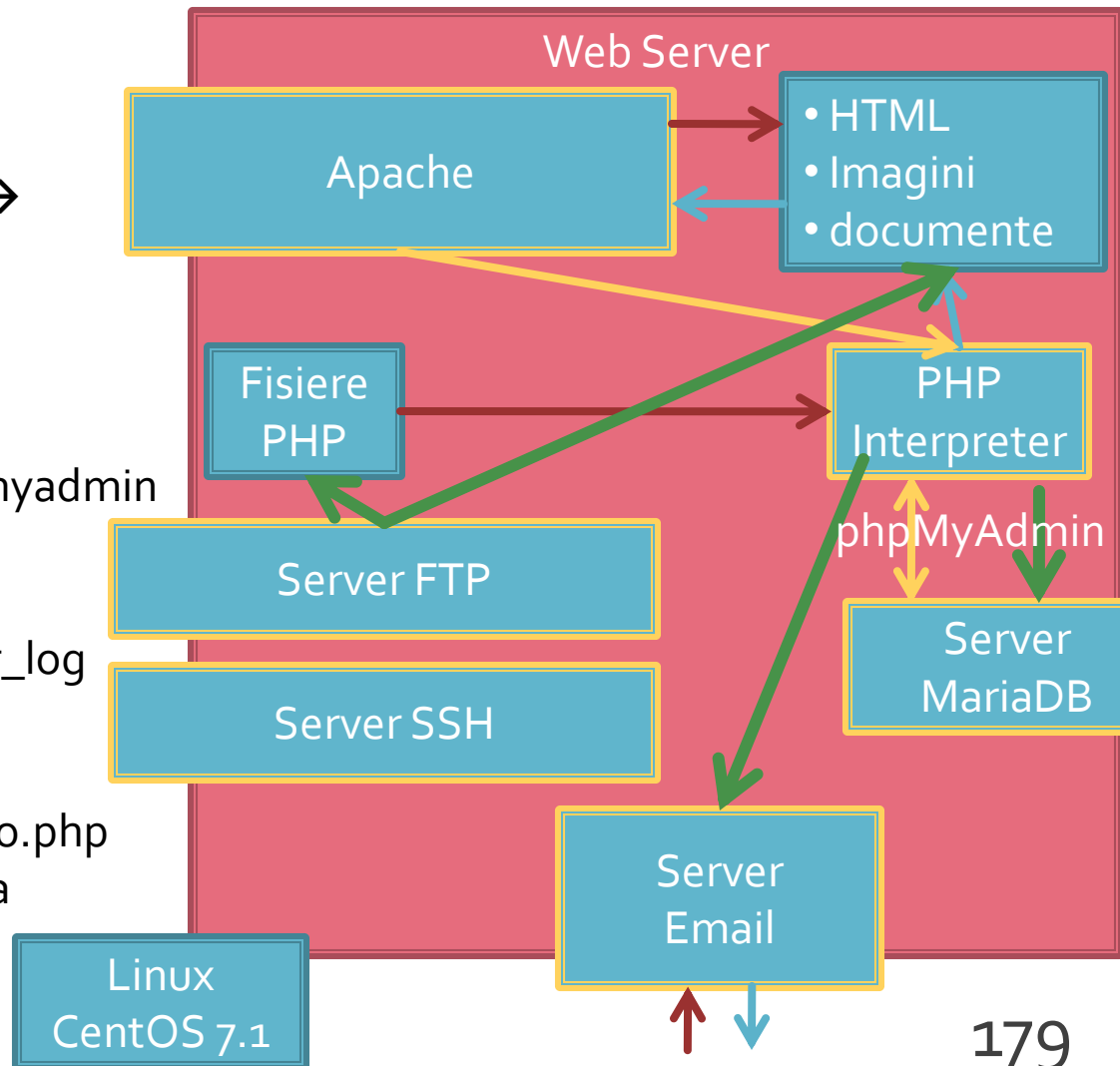
- Se creaza firul de executie paralel pentru vanzator
  - fisierele pentru cumparator reprezinta o buna cale de pornire (Save As, Copy/Paste) pentru 2 din cele 3 fisiere
- Se lucreaza cat mai mult la conversia text -> MySQL
  - activitatea se continua la laboratorul 7

# Utilizare LAMP



# 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`



# Faza de verificare/depanare

- Se recomanda utilizarea posibilitatii vizualizarii matricilor
  - In fisierul care receptioneaza datele
  - temporar pina la definitivarea codului
- utilizarea de cod "verbose" (manual) in etapele initiale de scriere a surselor PHP poate fi extinsa si la alte tipuri de date
  - singura (aproape) metoda de depanare(debug) in PHP
  - `<p>temp <?php echo "a=";echo $a; ?> </p>`

```
echo "<pre>";  
print_r($_POST);  
echo "</pre>";
```

# Depanare

```
echo "<pre>";  
print_r($_POST);  
echo "</pre>";
```

```
<p>temp <?php echo  
"a=";echo $a; ?> </p>
```

# Contact

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