Curs 8

2018/2019

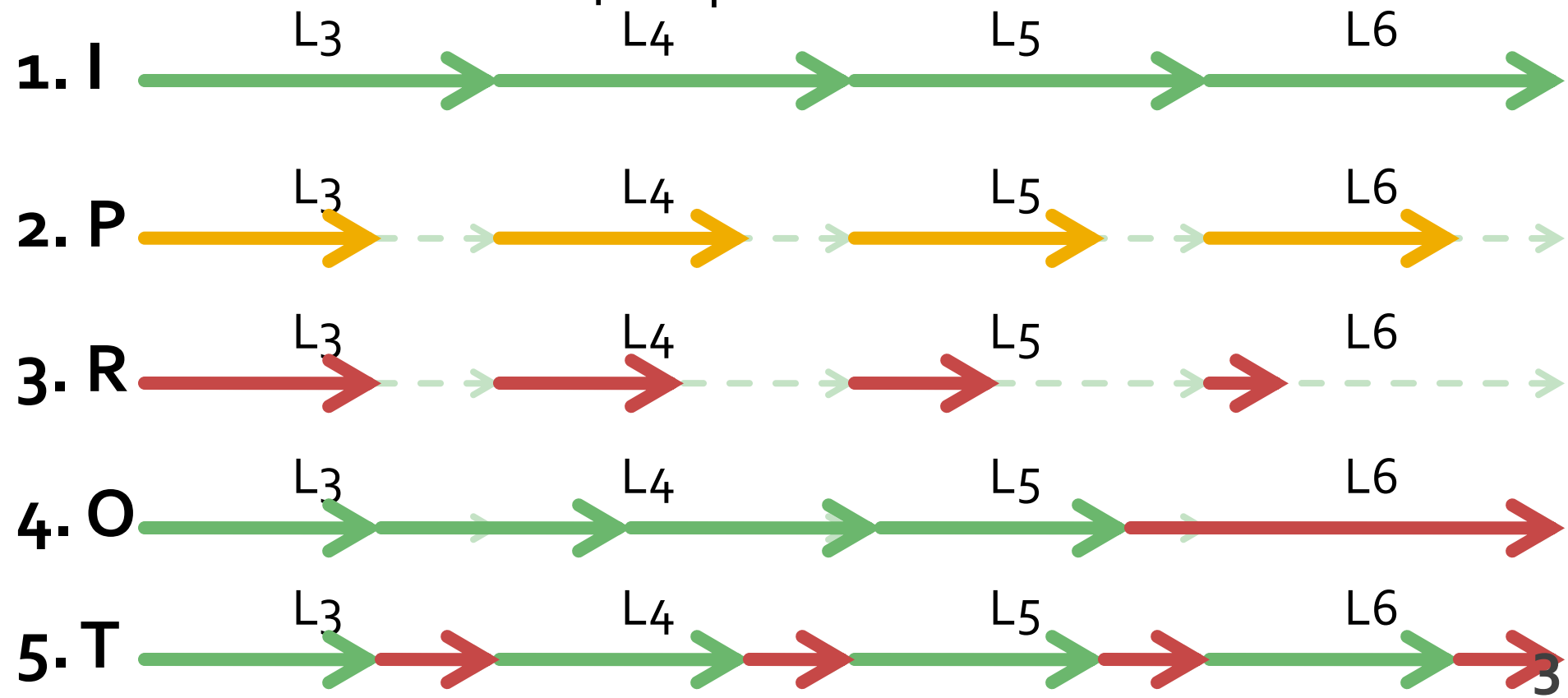
Programarea aplicațiilor web

CURS

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

! Important

- Laborator **asincron!**
  - recomandat – 4 = Optim



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><img src="images/antet.gif"
width="600" height="100" /></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><img src="images/antet.gif"
width="600" height="100" /></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
<h1>Magazin online Firma X SRL</h1>
```

subsol.php

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

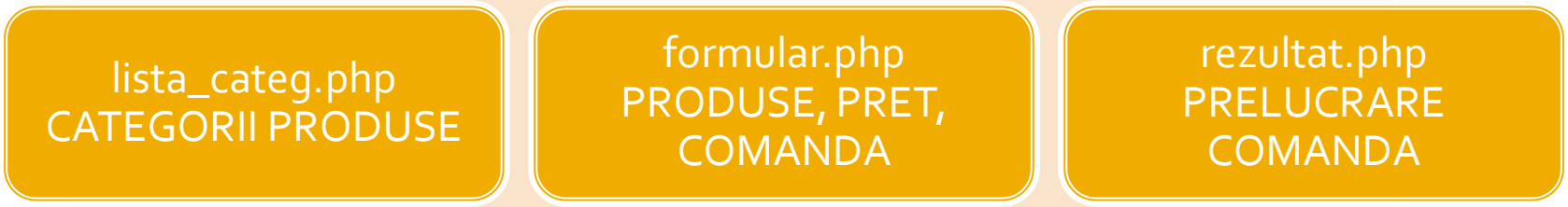
```
<?php require('antet.php');?>
<h2>Lista Produse</h2>
<table border="1">
…
</table>
<?php require('subsol.php');?>
```

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

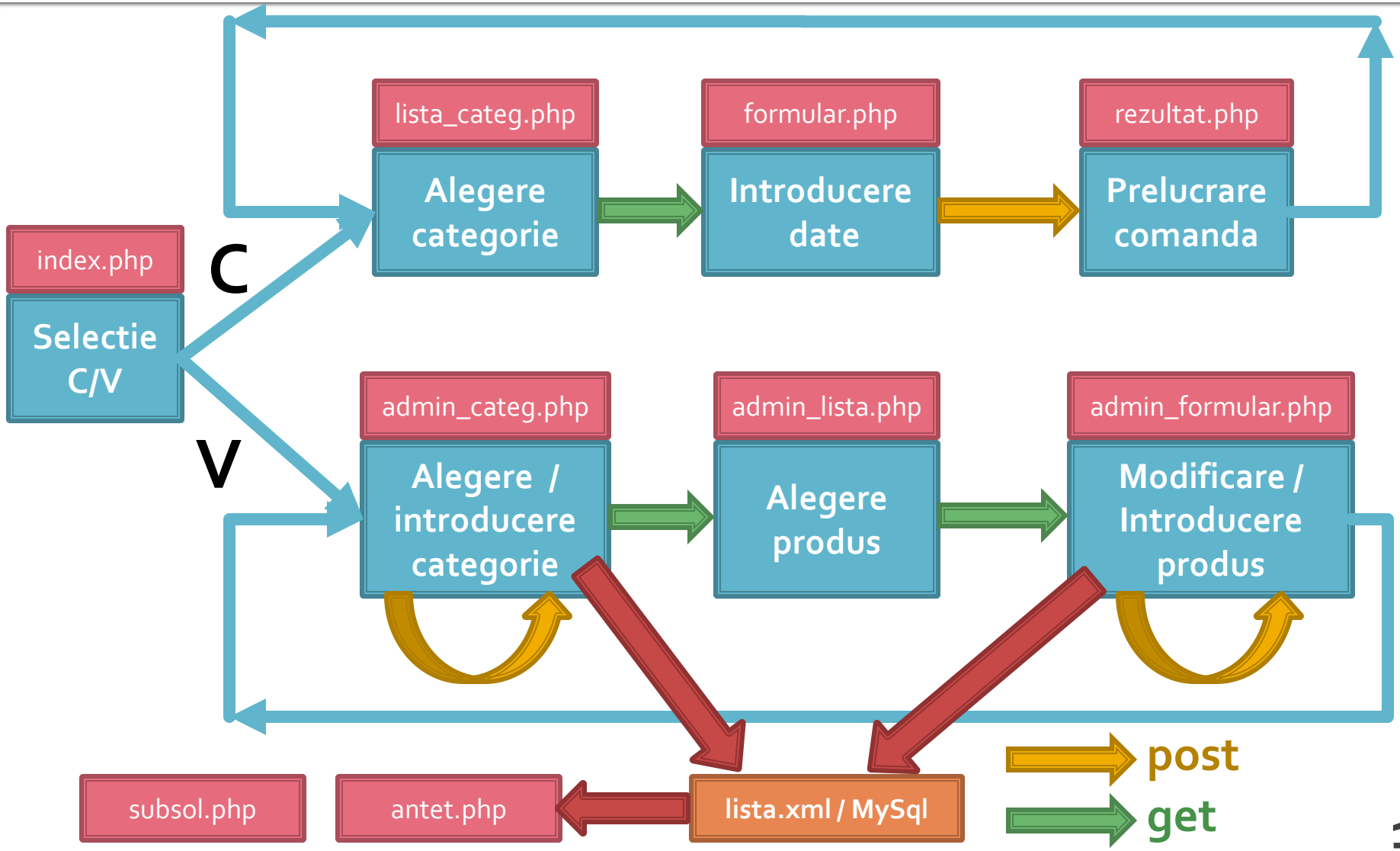
- Pe masura ce aplicatia paraseste un fir liniar de executie este necesara introducerea unui plan (graf) al aplicatiei
- Cumparator
  - citirea fisierului XML (accesarea bazei de date) se realizeaza in antet.php, comun pentru toate fisierele



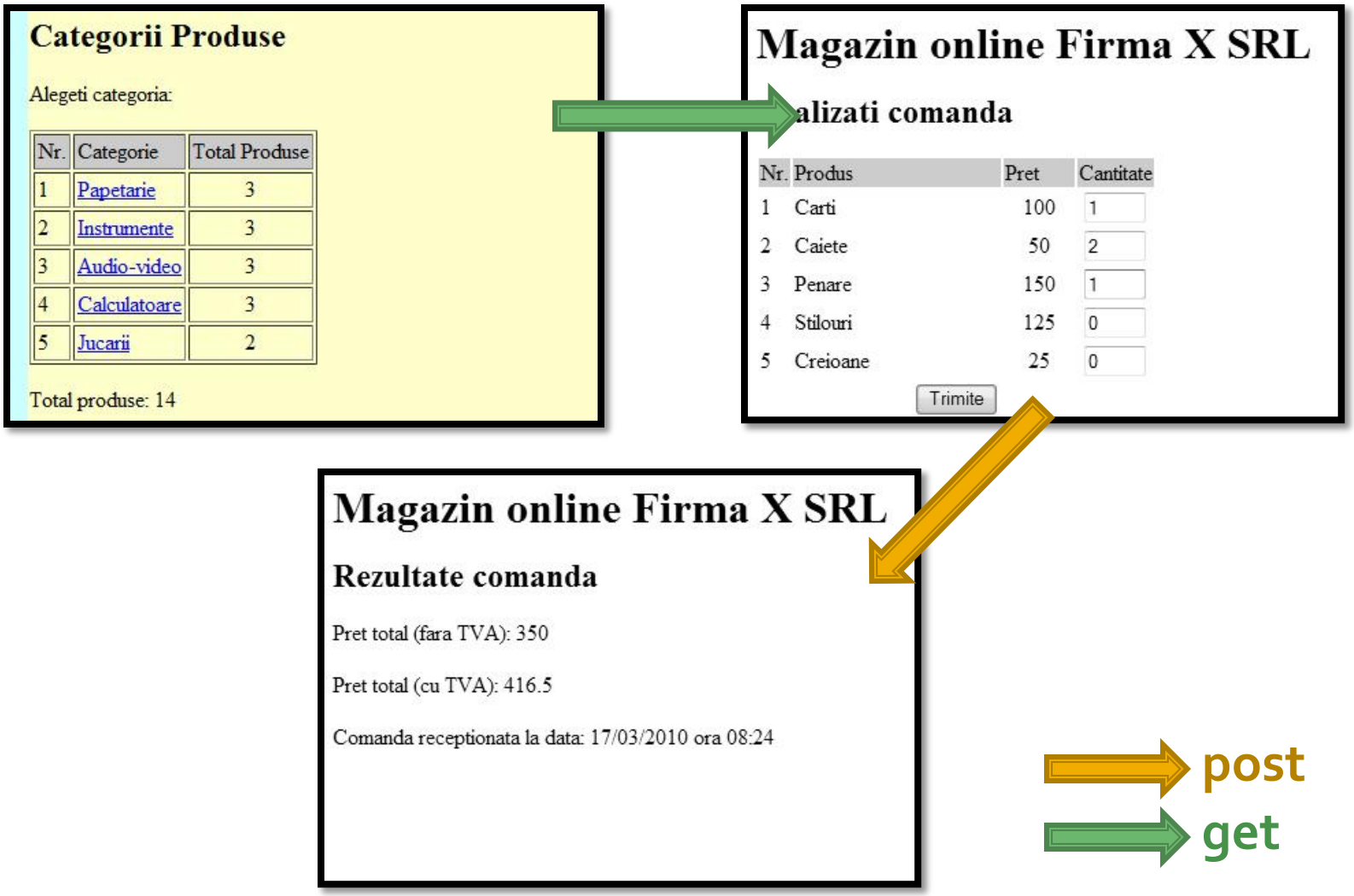
Plan aplicatie – Vanzator

- 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

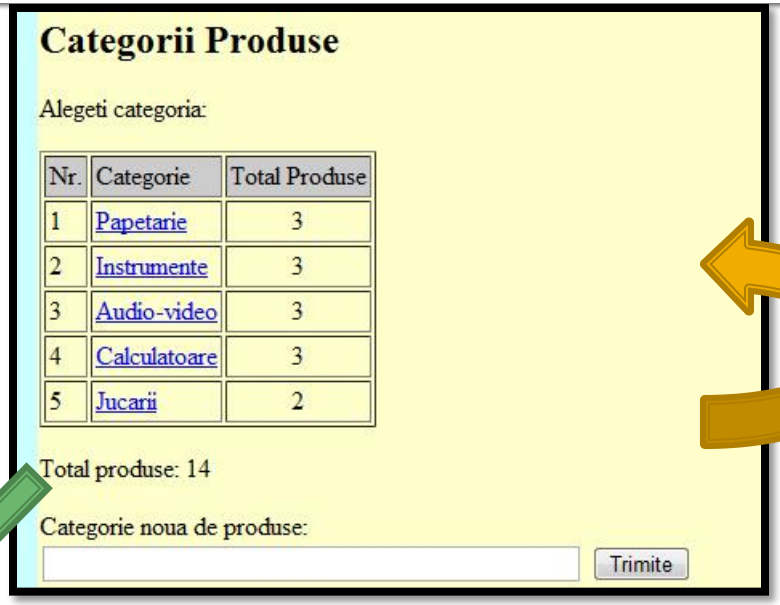
Plan aplicatie

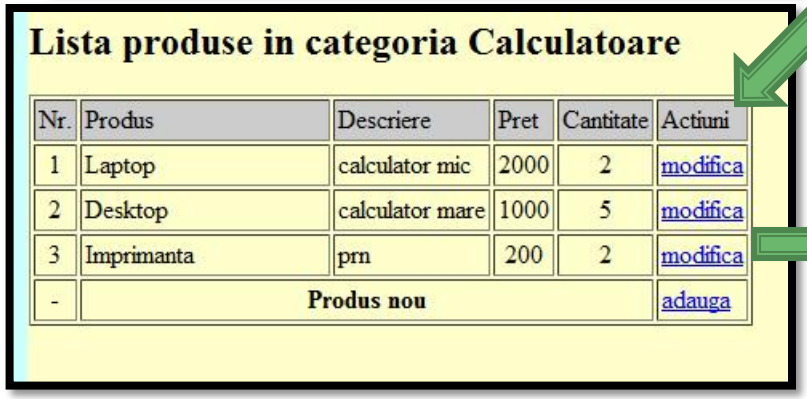


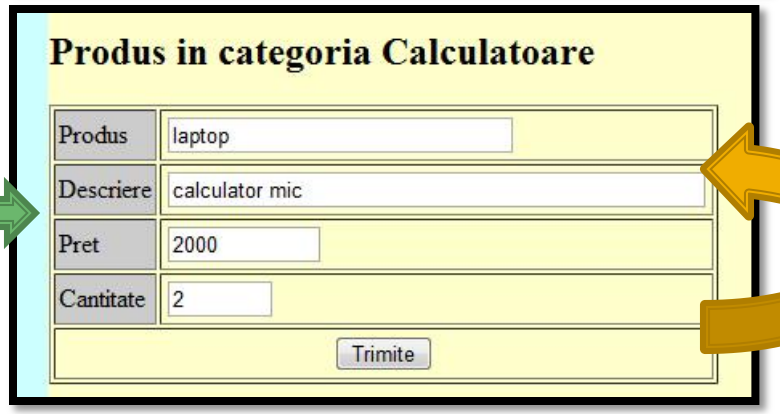
Rezultat (cumparator)



Rezultat (vanzator)







post

get

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

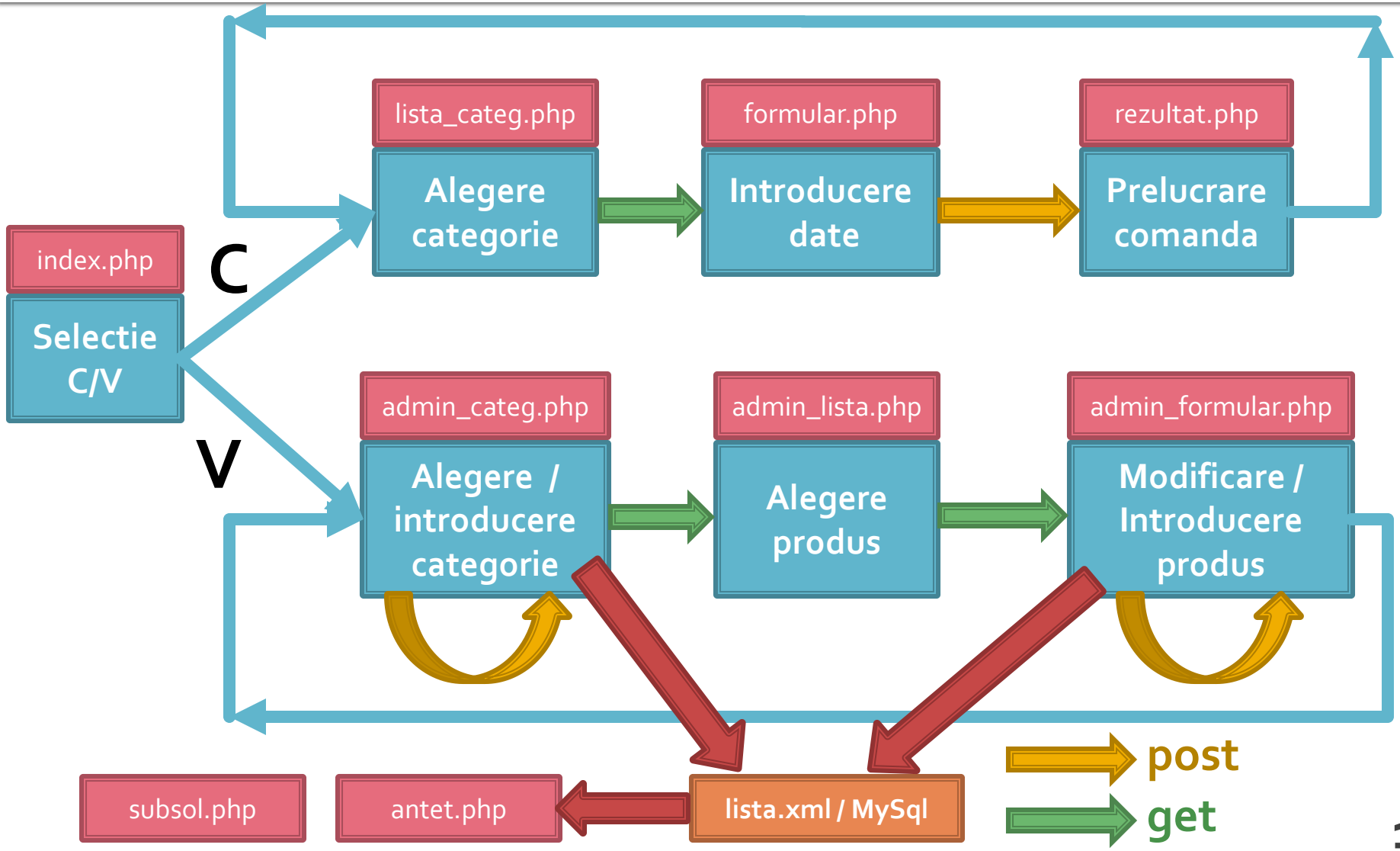
```
<form action="<?php echo $_SERVER['SCRIPT_NAME '];?>" method="post">
<p><input name="date_ok" type="submit" value="Trimite" /></p>
</form>
```

Fisier unic pentru colectare SI prelucrare date

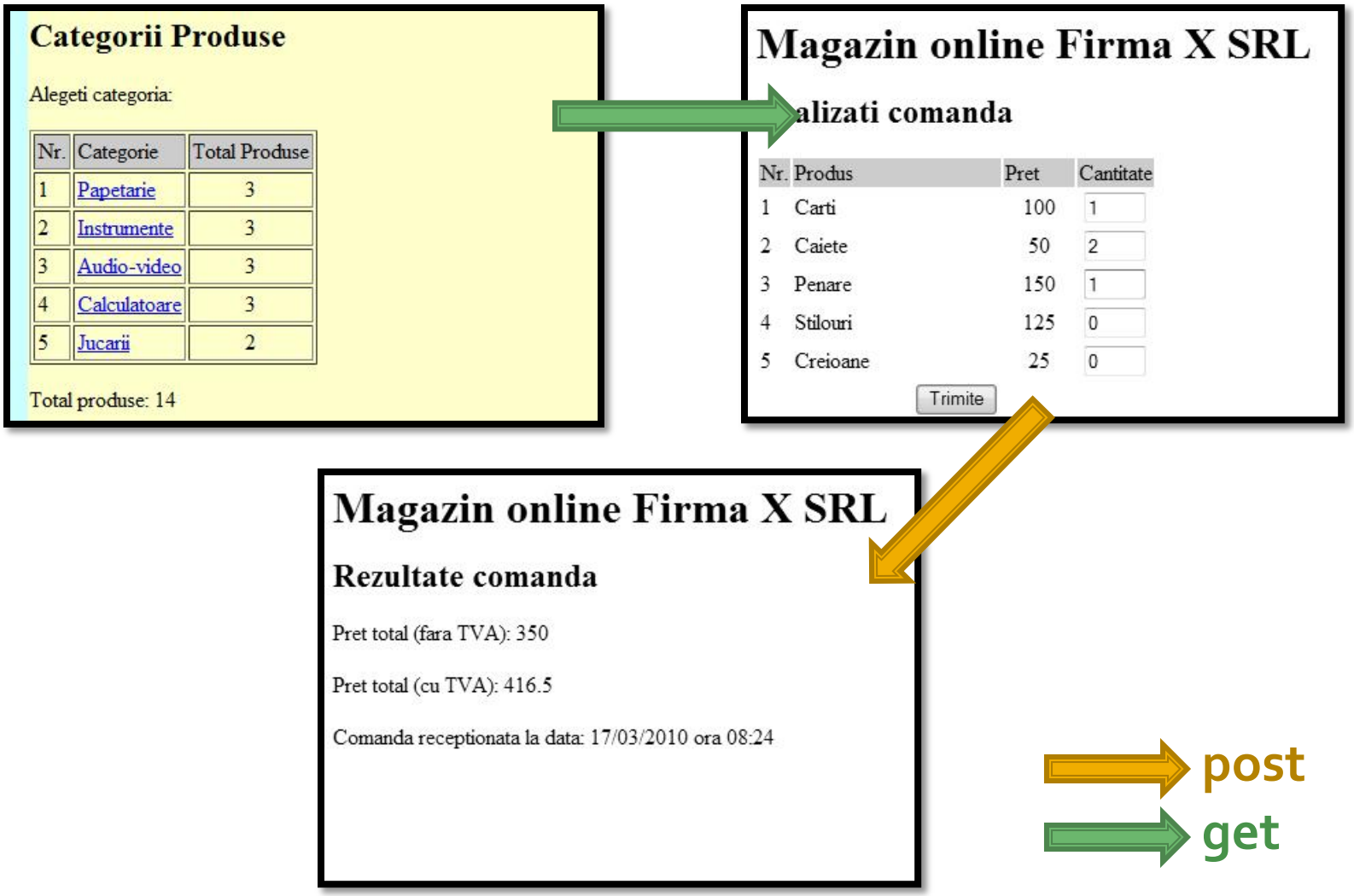
- Detectia existentei datelor se face prin verificarea existentei (isset($variabila)) valorilor introduse
  - eventual pentru un plus de protectie se poate verifica si continutul lor

```
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>
        }
```

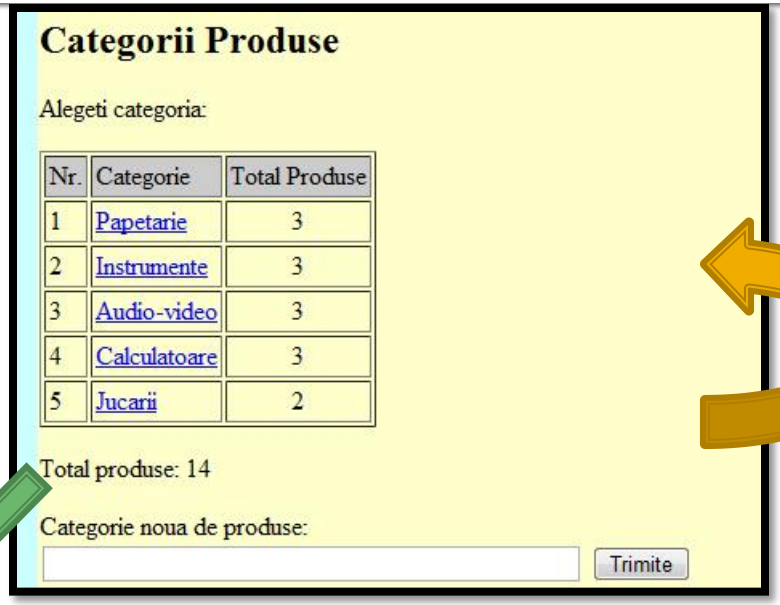
Plan aplicatie

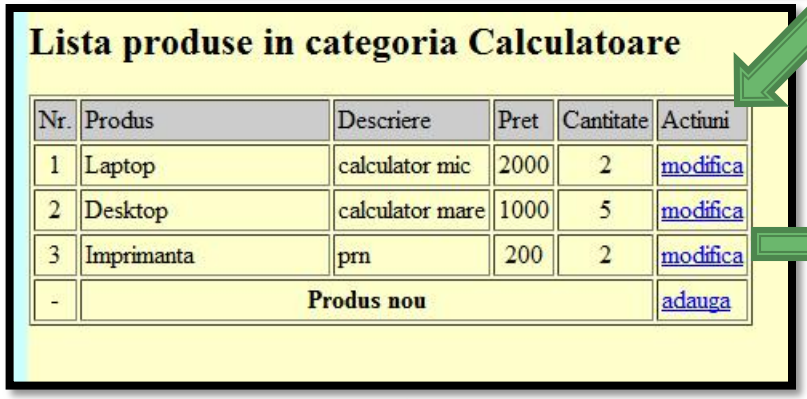


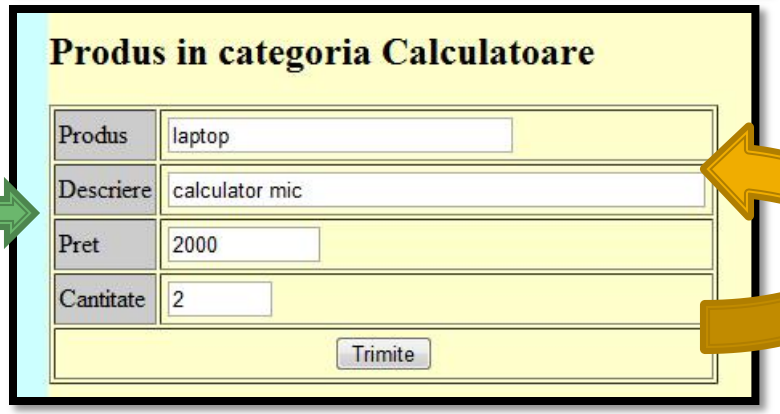
Rezultat (cumparator)



Rezultat (vanzator)







post

get

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 eficienta a datelor

MySql

- Baza de date – instrument pentru stocarea si manipularea informatiei eficient si efectiv
  - datele sunt protejate de corupere sau pierderi accidentale
  - nu se utilizeaza mai multe resurse decat minimul necesar
  - datele pot fi accesate cu performanta acceptabila
- Baze de date relationale
  - model relational (matematic eficient) – Codd ~1970

DBMS, RDBMS

- DBMS – database management system aplicatii incluse in baza de date pentru accesul la informatii
- RDBMS – Relational DBMS. Majoritatea sistemelor de baze de date tind la aceasta titulatura
  - ~300 de reguli trebuie respectate
  - nici un sistem actual nu implementeaza total aceste reguli

Relatii

- Toate sistemele de baze de date sunt caracterizate de:
  - toate informatiile sunt reprezentate intr-o aranjare ordonata **bidimensionala** numita **relatie**
  - toate valorile (atribute) stocate sunt scalare (in orice celula din tabel se stocheaza **o singura** valoare)
  - toate operatiile se aplica asupra unei intregi relatii si rezulta o intreaga relatie
- Terminologii (MySql)
  - tabel – table / recordset / result set
  - linie – record / row
  - coloana – field / column

Relatii, chei

- toate informatiile sunt reprezentate intr-o aranjare bidimensionala numita relatie
  - aranjarile bidimensionale nu sunt ordonate implicit
  - datele trebuie stocate pentru a implementa o relatie in asa fel incat fiecare linie sa fie unica
- cheie candidata
  - exista cel putin o combinatie de atribute (coloane) care pot identifica in mod unic o linie
  - aceste combinatii de atribute se numesc chei candidate

Chei

- Din toate combinatiile de coloane care pot fi utilizate pentru identificarea unica a unei linii se alege **macar** una utilizata intern de RDBMS pentru ordonarea datelor – **cheie primara**
  - Celelelte chei candidate devin **chei alternative** si pot fi folosite pentru eficientizarea prelucrarilor (crearea de "index" dupa aceste chei)
- In cazul in care nu exista o combinatie de coloane utilizabila ca si cheie cu utilitate practica se introduce artificial o cheie, cu numere intregi incrementate automat de DBMS (autoincrement)
  - de multe ori este recomandata o astfel de actiune, numerele intregi fiind mult mai usor de controlat, ordonat, cautat decat alte tipuri de date
  - cheile de tip autoincrement nu e **nevoie** sa contina informatie

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

Eliminarea datelor redundante

Order ID	SalesPerson	Hire Date	Phone	Company Name	Product Name	Quantity
10871	Dodsworth, Anne	15-Nov-1994	452	Bon app'	Alice Mutton	16
10747	Suyama, Michael	17-Oct-1993	428	Piccolo und mehr	Gorgonzola Telino	8
10258	Davolio, Nancy	01-May-1992	5467	Ernst Handel	Chef Anton's Gumbo Mix	65
11007	Callahan, Laura	05-Mar-1994	2344	Princesa Isabel Vinhos	Thüringer Rostbratwurst	10
10421	Callahan, Laura	05-Mar-1994	2344	Que Delicia	Perth Pasties	15
10558	Davolio, Nancy	01-May-1992	5467	Around the Horn	Perth Pasties	18
10431	Peacock, Margaret	03-May-1993	5176	Bottom-Dollar Markets	Alice Mutton	50
10659	King, Robert	02-Jan-1994	465	Queen Cozinha	Gorgonzola Telino	20
10273	Leverling, Janet	01-Apr-1992	3355	QUICK-Stop	Gorgonzola Telino	15
10382	Peacock, Margaret	03-May-1993	5176	Ernst Handel	Chef Anton's Gumbo Mix	32
10949	Fuller, Andrew	14-Aug-1992	3457	Bottom-Dollar Markets	Alice Mutton	6
10285	Davolio, Nancy	01-May-1992	5467	QUICK-Stop	Perth Pasties	36
10867	Suyama, Michael	17-Oct-1993	428	Lonesome Pine Restaur	Perth Pasties	3
10691	Fuller, Andrew	14-Aug-1992	3457	QUICK-Stop	Thüringer Rostbratwurst	40
10354	Callahan, Laura	05-Mar-1994	2344	Pericles Comidas clásic	Thüringer Rostbratwurst	4
10698	Peacock, Margaret	03-May-1993	5176	Ernst Handel	Thüringer Rostbratwurst	12
10962	Callahan, Laura	05-Mar-1994	2344	QUICK-Stop	Perth Pasties	20
10465	Davolio, Nancy	01-May-1992	5467	Vaffeljernet	Thüringer Rostbratwurst	18
10549	Buchanan, Steven	17-Oct-1993	3453	QUICK-Stop	Gorgonzola Telino	55

Eliminarea datelor redundante

Customers Relation

Customer ID	Company Name	Phone
ALFKI	Alfreds Futterkiste	030-0074321
ANATR	Ana Trujillo Emparedados y helados	(5) 555-4729
ANTON	Antonio Moreno Taquería	(5) 555-3932
AROUT	Around the Horn	(171) 555-7788
BERGS	Berglunds snabbköp	0921-12 34 65
BLAUS	Blauer See Delikatessen	0621-08460
BLONP	Blondel père et fils	88.60.15.31
BOLID	Bólido Comidas preparadas	(91) 555 22 82
BONAP	Bon app'	91.24.45.40
BOTTM	Bottom-Dollar Markets	(604) 555-4729
BSBEV	B's Beverages	(171) 555-1212
CACTU	Cactus Comidas para llevar	(1) 135-5555
CENTC	Centro comercial Moctezuma	(5) 555-3392

Invoices Relation

Order ID	Company Name	Phone
10643	Alfreds Futterkiste	030-0074321
10692	Alfreds Futterkiste	030-0074321
10702	Alfreds Futterkiste	030-0074321
10835	Alfreds Futterkiste	030-0074321
10952	Alfreds Futterkiste	030-0074321
11011	Alfreds Futterkiste	030-0074321
10308	Ana Trujillo Emparedados y helados	(5) 555-4729
10625	Ana Trujillo Emparedados y helados	(5) 555-4729
10759	Ana Trujillo Emparedados y helados	(5) 555-4729
10926	Ana Trujillo Emparedados y helados	(5) 555-4729
10365	Antonio Moreno Taquería	(5) 555-3932
10507	Antonio Moreno Taquería	(5) 555-3932
10535	Antonio Moreno Taquería	(5) 555-3932
10573	Antonio Moreno Taquería	(5) 555-3932
10677	Antonio Moreno Taquería	(5) 555-3932

When was she hired?

Order ID	SalesPerson	Hire Date	Phone	Company Name	Product Name
10871	Dodsworth, Anne	15-Nov-1994	452	Bon app'	Alice Mutton
10747	Suyama, Michael	17-Oct-1993	428	Piccolo und mehr	Gorgonzola Telino
10258	Davolio, Nancy	01-May-1992	5467	Ernst Handel	Chef Anton's Gumbo Mix
11007	Callahan, Laura	05-Mar-1994	2344	Princesa Isabel Vinhos	Thüringer Rostbratwurst
10421	Callahan, Laura	05-Mar-1994	2344	Que Delícia	Perth Pasties
10558	Davolio, Nancy	01-May-1992	5467	Around the Horn	Perth Pasties
10431	Peacock, Margaret	03-May-1993	5176	Bottom-Dollar Markets	Alice Mutton

Product ID	Product Name	Unit Price
1	Chai	$18.00
2	Chang	$19.00
3	Aniseed Syrup	$10.00
4	Chef Anton's Cajun Seasoning	$22.00
5	Chef Anton's Gumbo Mix	$21.35
6	Grandma's Boysenberry Spread	$25.00
7	Uncle Bob's Organic Dried Pears	$30.00
8	Northwoods Cranberry Sauce	$40.00
9	Mishi Kobe Niku	$97.00
10	Ikura	$31.00
11	Queso Cabrales	$21.00
12	Queso Manchego La Pastora	$38.00
13	Konbu	$6.00
14	Tofu	$23.25

These are not the same value

Order ID	Product Name	Unit Price	Quantity	Unit Price
10248	Mozzarella di Giovanni	$34.80	5	$174.00
10248	Queso Cabrales	$21.00	12	$168.00
10248	Singaporean Hokkien Fried Mee	$14.00	10	$98.00
10249	Manjimup Dried Apples	$53.00	40	$1,696.00
10249	Tofu	$23.25	9	$167.40

Prima forma normala

- toate valorile sunt scalare

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

- nu toate rezolvarile sunt eficiente

OrderID	CustomerID	Item1	Qty1	Item2	Qty2	Item3	Qty3
1	ANTON	Queso Cabrales	4	Tofu	3	Ravioli Angelo	1
2	BLAUS	Chai	2		0		

Product	Year	TargetJan	ActualJan	TargetFeb	ActualFeb
Aniseed Syrup	2004	$1,000.00	$1,300.00	$0.00	$0.00
Chai	2004	$4,000.00	$2,000.00	$0.00	$0.00
Chang	2004	$3,000.00	$8,022.00	$0.00	$0.00

A doua forma normala

- O relatie este in a **doua** forma normala cand este in **prima** forma normala si suplimentar atributele (valorile de pe coloana) depind de **intreaga cheie** candidata aleasa

	Product Name	SupplierName	Category Name	SupplierPhoneNumber
	Chai	Exotic Liquids	Beverages	(171) 555-2222
	Chang	Exotic Liquids	Beverages	(171) 555-2222
	Guaraná Fantástica	Refrescos Americanas LTDA	Beverages	(11) 555 4640
	Sasquatch Ale	Bigfoot Breweries	Beverages	(503) 555-9931
	Steeleye Stout	Bigfoot Breweries	Beverages	(503) 555-9931
	Côte de Blaye	Aux joyeux ecclésiastiques	Beverages	(1) 03.83.00.68
	Chartreuse verte	Aux joyeux ecclésiastiques	Beverages	(1) 03.83.00.68
	Ipoh Coffee	Leka Trading	Beverages	555-8787
	Laughing Lumberjack Lager	Bigfoot Breweries	Beverages	(503) 555-9931
	Outback Lager	Pavlova, Ltd.	Beverages	(03) 444-2343

A doua forma normala

Product ID	Product Name	Category
1	Chai	Beverages
2	Chang	Beverages
3	Aniseed Syrup	Condiments
4	Chef Anton's Cajun Seasoning	Condiments
5	Chef Anton's Gumbo Mix	Condiments
6	Grandma's Boysenberry Spread	Condiments
7	Uncle Bob's Organic Dried Pears	Produce

Supplier ID	SupplierName	SupplierPhoneNumber
1	Exotic Liquids	(171) 555-2222
2	New Orleans Cajun Delights	(100) 555-4822
3	Grandma Kelly's Homestead	(313) 555-5735
4	Tokyo Traders	(03) 3555-5011
5	Cooperativa de Quesos 'Las Cabras'	(98) 598 76 54
6	Mayumi's	(06) 431-7877
7	Pavlova, Ltd.	(03) 444-2343
8	Specialty Biscuits, Ltd.	(161) 555-4448
9	PB Knäckebröd AB	031-987 65 43

A treia forma normala

- O relatie este in a **treia** forma normala cand este in a **doua** forma normala si suplimentar atributele (valorile de pe coloana) care nu fac parte din cheie sunt **mutual independente**

	Company Name	Address	City	Region	Postal Code
	Exotic Liquids	49 Gilbert St.	London		EC1 4SD
	New Orleans Cajun Delights	P.O. Box 78934	New Orleans	LA	70117
	Grandma Kelly's Homestead	707 Oxford Rd.	Ann Arbor	MI	48104
	Tokyo Traders	9-8 Sekimai	Tokyo		100
	Cooperativa de Quesos 'Las Cabras'	Calle del Rosal 4	Oviedo	Asturias	33007
	Mayumi's	92 Setsuko	Osaka		545
	Pavlova, Ltd.	74 Rose St.	Melbourne	Victoria	3058

A treia forma normala

Company Name	Address	City
Exotic Liquids	49 Gilbert St.	London
New Orleans Cajun Delights	P.O. Box 78934	New Orleans
Grandma Kelly's Homestead	707 Oxford Rd.	Ann Arbor
Tokyo Traders	9-8 Sekimai	Tokyo
Cooperativa de Quesos 'Las Cabras'	Calle del Rosal 4	Oviedo
Mayumi's	92 Setsuko	Osaka
Pavlova, Ltd.	74 Rose St.	Melbourne

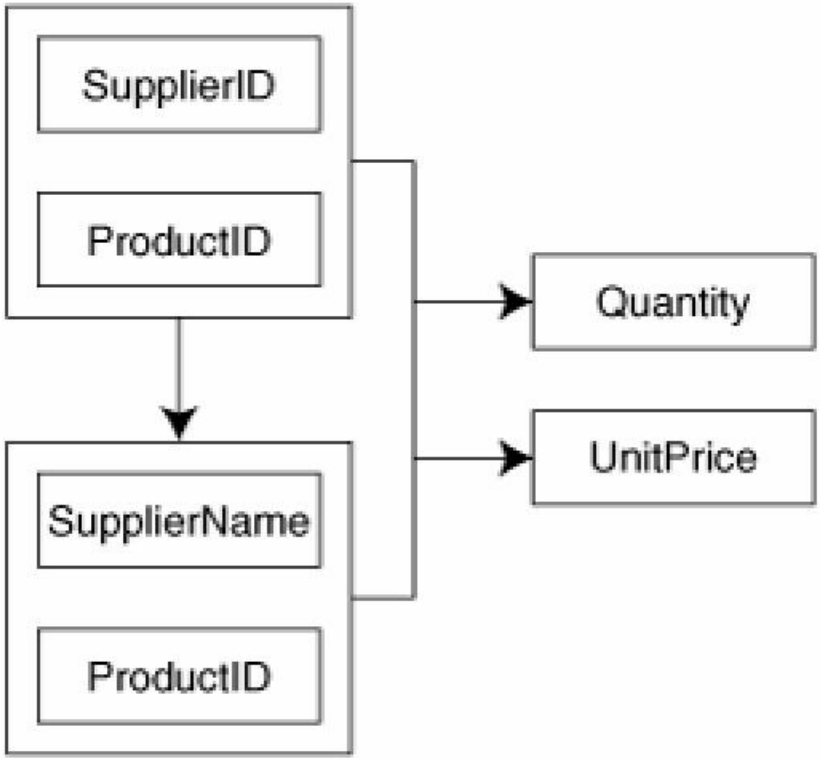
City	Region	Postal Code
Melbourne	Victoria	3058
Ste-Hyacinthe	Québec	J2S 7S8
Montréal	Québec	H1J 1C3
Bend	OR	97101
Sydney	NSW	2042
Ann Arbor	MI	48104
Boston	MA	02134
New Orleans	LA	70117
Oviedo	Asturias	33007

Normalizare suplimentara

- Se tine cont si de eliminarea datelor redundante. Anumite redundante pot fi eliminate prin introducerea de relatii suplimentare
- Forma normala Boyce/Codd cere sa nu existe dependenta functionala intre cheile candidate

Supplier ID	SupplierName	Product	Quantity	Unit Price
5	Cooperativa de Quesos 'Las Cabras'	Queso Cabrales	12	$14.00
20	Leka Trading	Singaporean Hokkien Fried Mee	10	$9.80
14	Formaggi Fortini s.r.l.	Mozzarella di Giovanni	5	$34.80
24	G'day, Mate	Manjimup Dried Apples	40	$42.40
6	Mayumi's	Tofu	9	$18.60
24	G'day, Mate	Manjimup Dried Apples	35	$42.40
19	New England Seafood Cannery	Jack's New England Clam Chowder	10	$7.70
2	New Orleans Cajun Delights	Louisiana Fiery Hot Pepper Sauce	15	$16.80

Normalizare suplimentara



Supplier ID	SupplierName
1	Exotic Liquids
2	New Orleans Cajun Delights
3	Grandma Kelly's Homestead
4	Tokyo Traders
5	Cooperativa de Quesos 'Las Cabras'
6	Mayumi's

SupplierID	ProductID	Quantity	UnitPrice
2	65	15	$21.05
24	53	15	$32.80
8	20	40	$81.00
22	47	16	$9.50
6	14	9	$23.25
28	59	30	$55.00
28	60	40	$34.00
21	46	15	$12.00

MySql – Recapitulare rapida

Relatii in Bazele de date

Relatii in Bazele de date

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

One to One

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

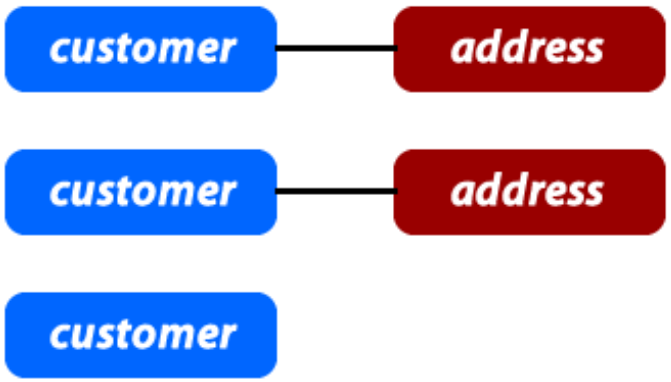
One to One

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

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

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

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



One to Many

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

One to Many, Many to One

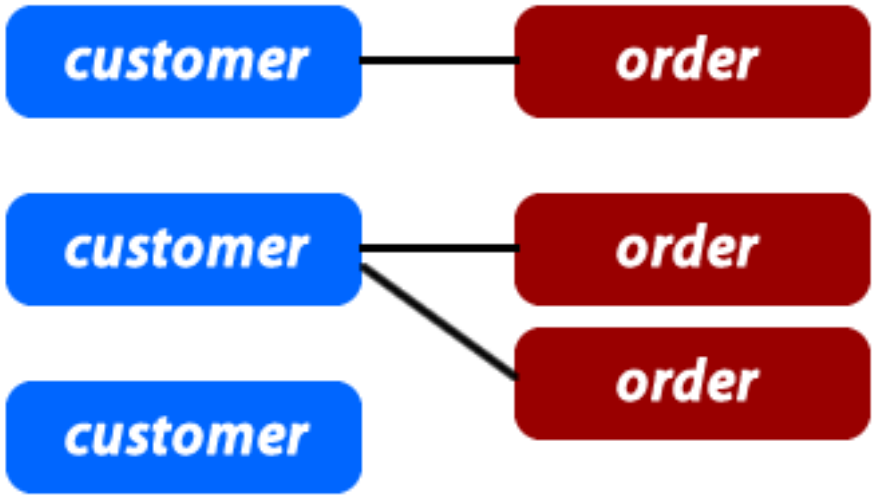
- de obicei aceste legaturi se implementeaza prin introducerea cheii primare din tabelul **One** in calitate de coloana in tabelul **Many** (cheie externa – foreign key)

CUSTOMERS

customer_id	customer_name
101	John Doe
102	Bruce Wayne

ORDERS

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



Many to Many

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

Many to Many

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

ORDERS

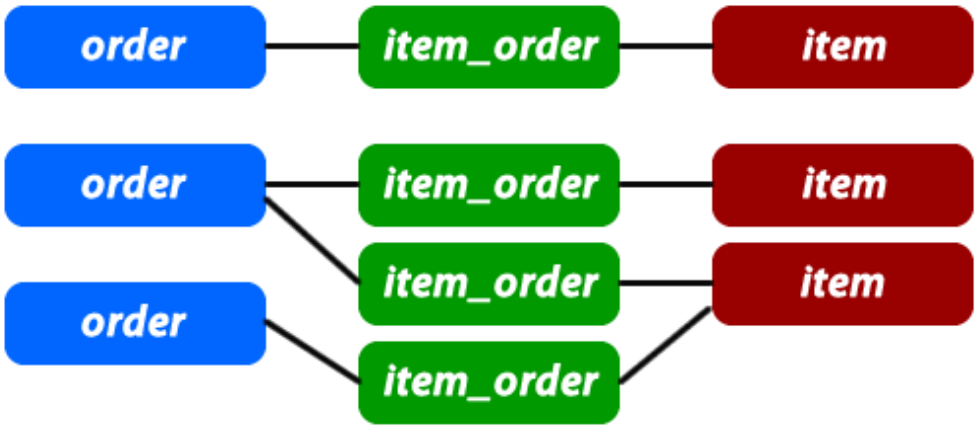
order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99

ITEMS

item_id	item_name	item_description
201	Tickle Me Elmo	It wants to be tickled
202	District 9 DVD	Awesome sci-fi movie
203	Batarang	It is very sharp

ITEMS_ORDERS

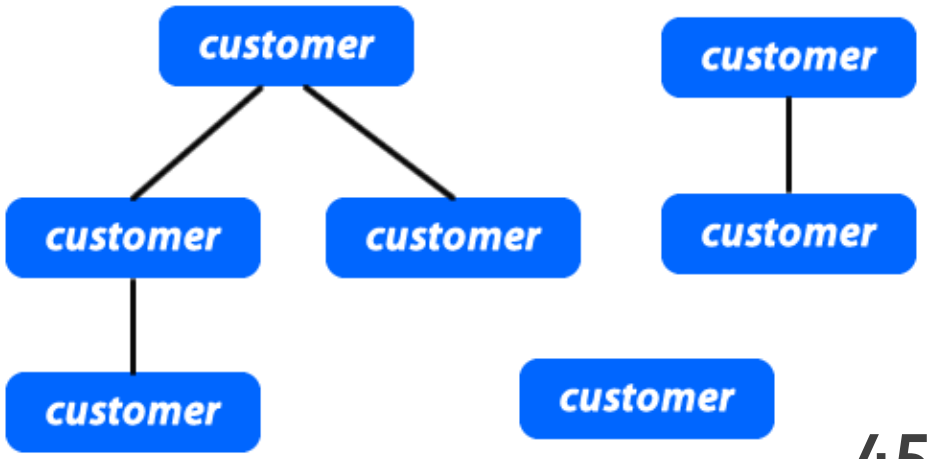
order_id	item_id
555	201
555	202
556	202
556	203



Self Referencing (unare)

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

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



Relatii in Bazele de date

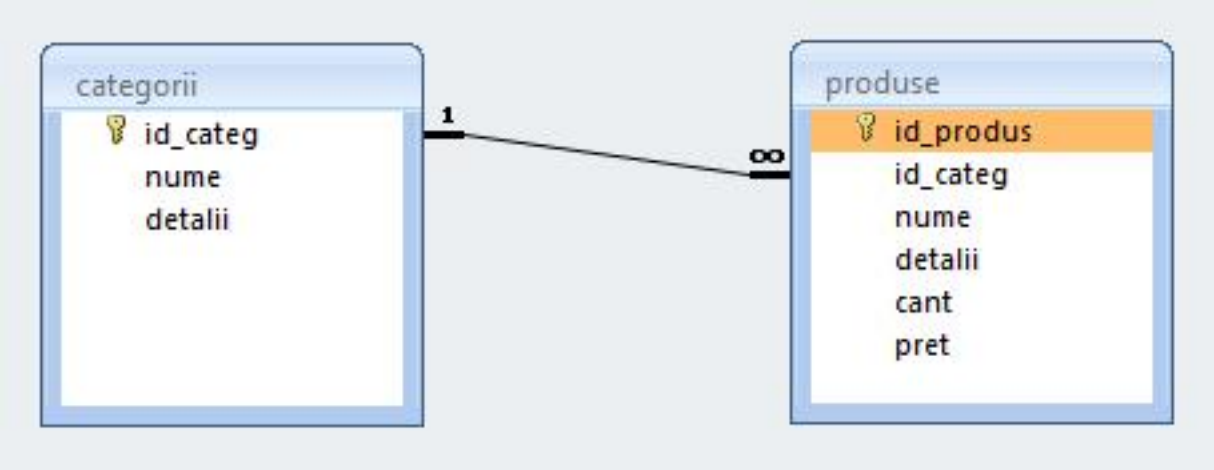
- Respectarea formelor normale ale bazelor de date aduce nenumarate avantaje
- Efectul secundar este dat de necesitatea separarii datelor intre mai multe tabele
- In exemplul utilizat avem doua concepte diferite din punct de vedere logic
  - produs
  - categorie de produs

Relatii in Bazele de date

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

Relatii in Bazele de date

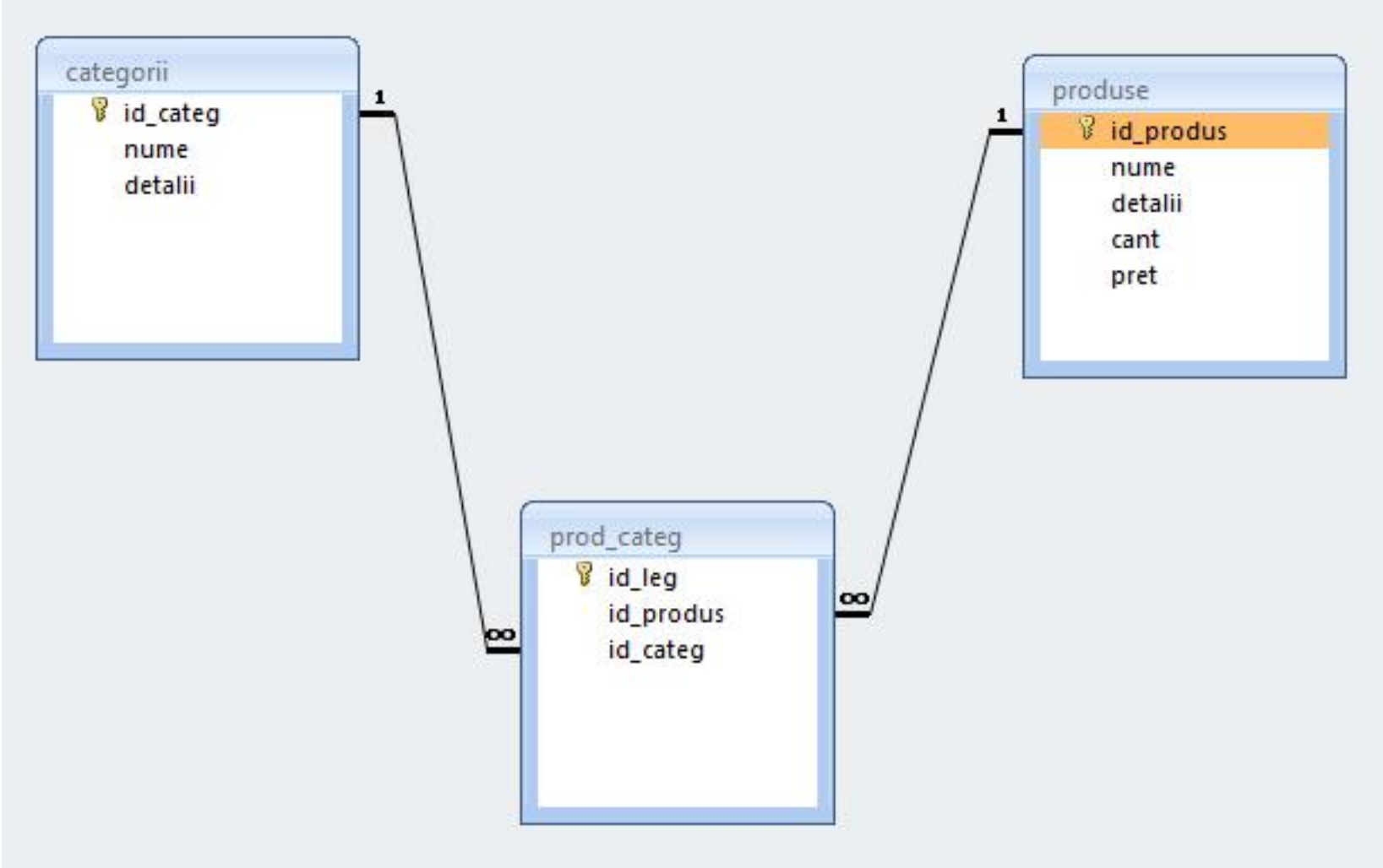
- Legaturile implementata
  - One to Many
  - in tabelul “produse” apare cheia externa (foreign key): “id_categ”



Relatii in Bazele de date

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

Functii PHP de acces MySql

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

Functii PHP de acces MySql
Parcurgerea resurselor rezultat

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

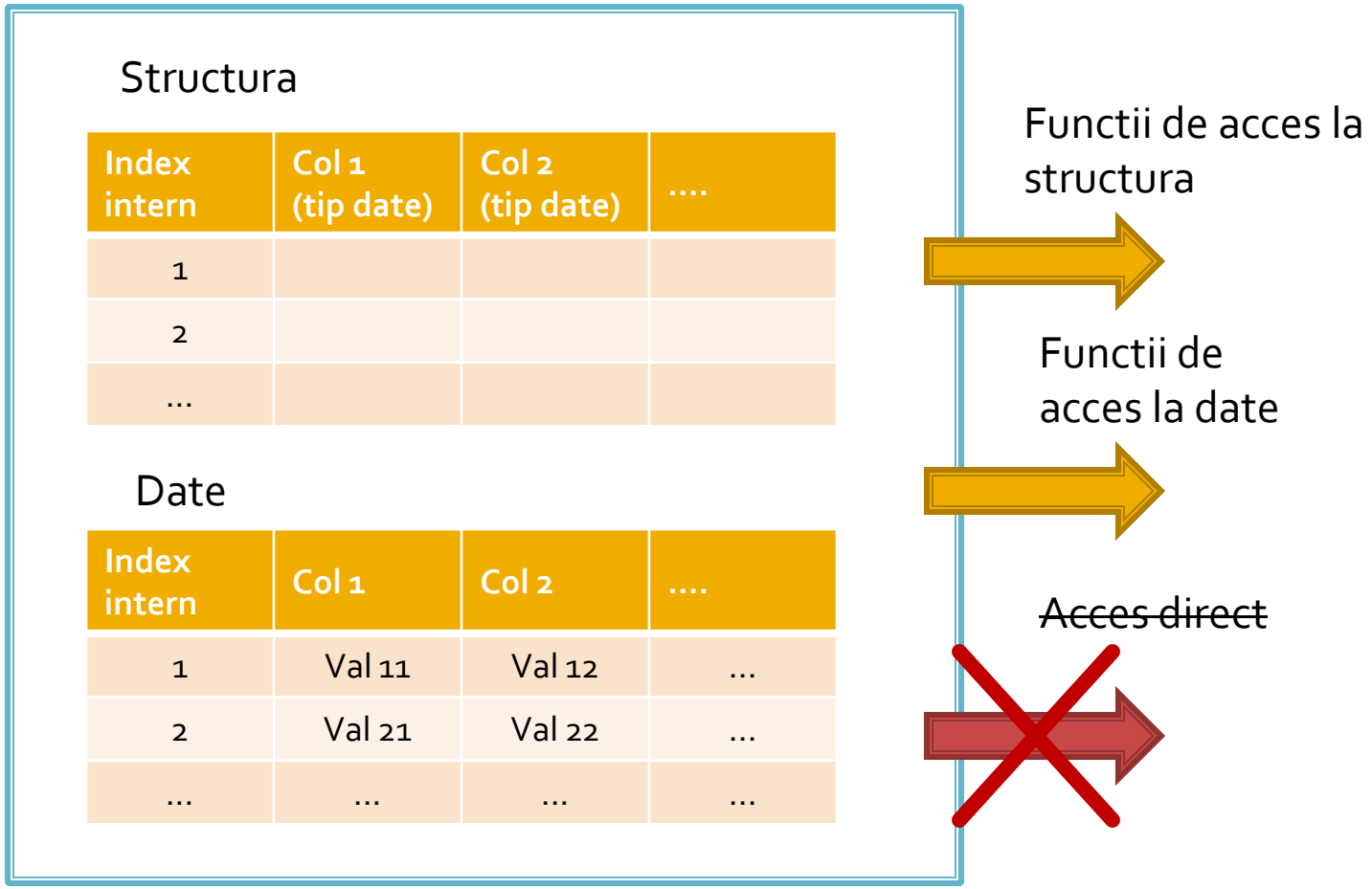
Functii PHP de acces MySql
Parcurgerea resurselor rezultat

- mysql_fetch_array
  - grupeaza functionalitatea mysql_fetch_assoc si 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 indicata
  - 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



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 );
```

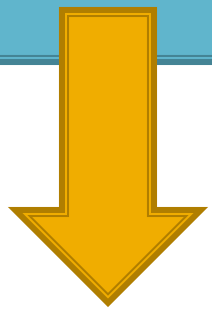
Exemplu de utilizare

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

Modificari laborator cu date stocate text

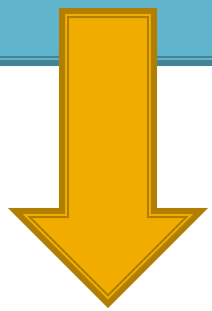
- Codul aplicatiei ramane in mare parte acelasi
- Se modifica doar citirea valorilor pentru popularea matricii $produse (“antet.php”)

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



Modificari laborator cu date stocate XML

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



Modificari laborator cu date stocate MySQL

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

MySql – eficienta

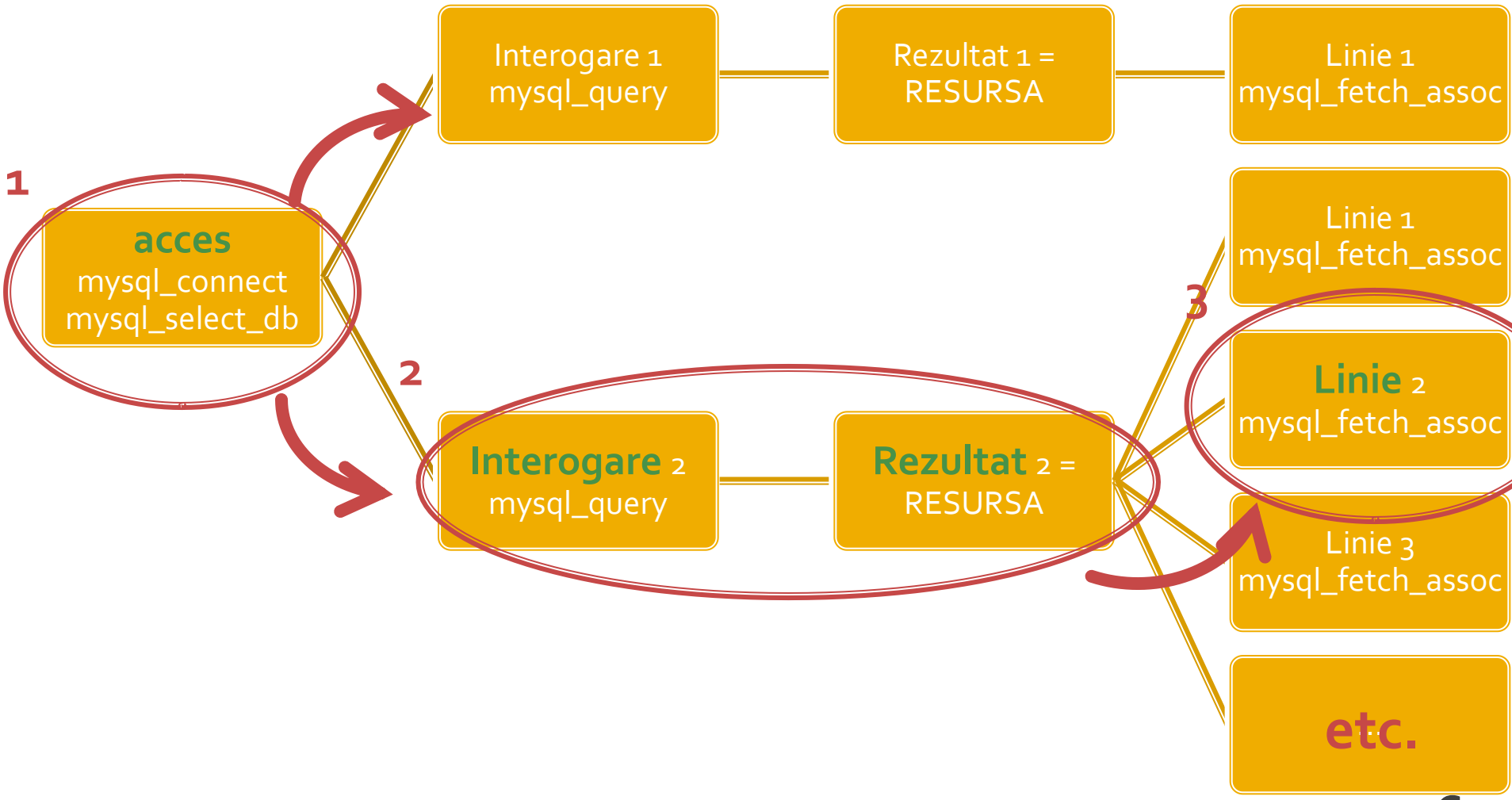
- eficienta unei aplicatii web
  - 100% - **toate prelucrarile "mutate" in RDBMS**
  - PHP **doar** afisarea datelor
- eficienta unei aplicatii MySql
  - 25% **alegerea corecta a tipurilor de date**
  - 25% **crearea indecsilor necesari in aplicatii**
  - 25% **normalizarea corecta a bazei de date**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

Optimizare

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

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

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

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

Functii 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

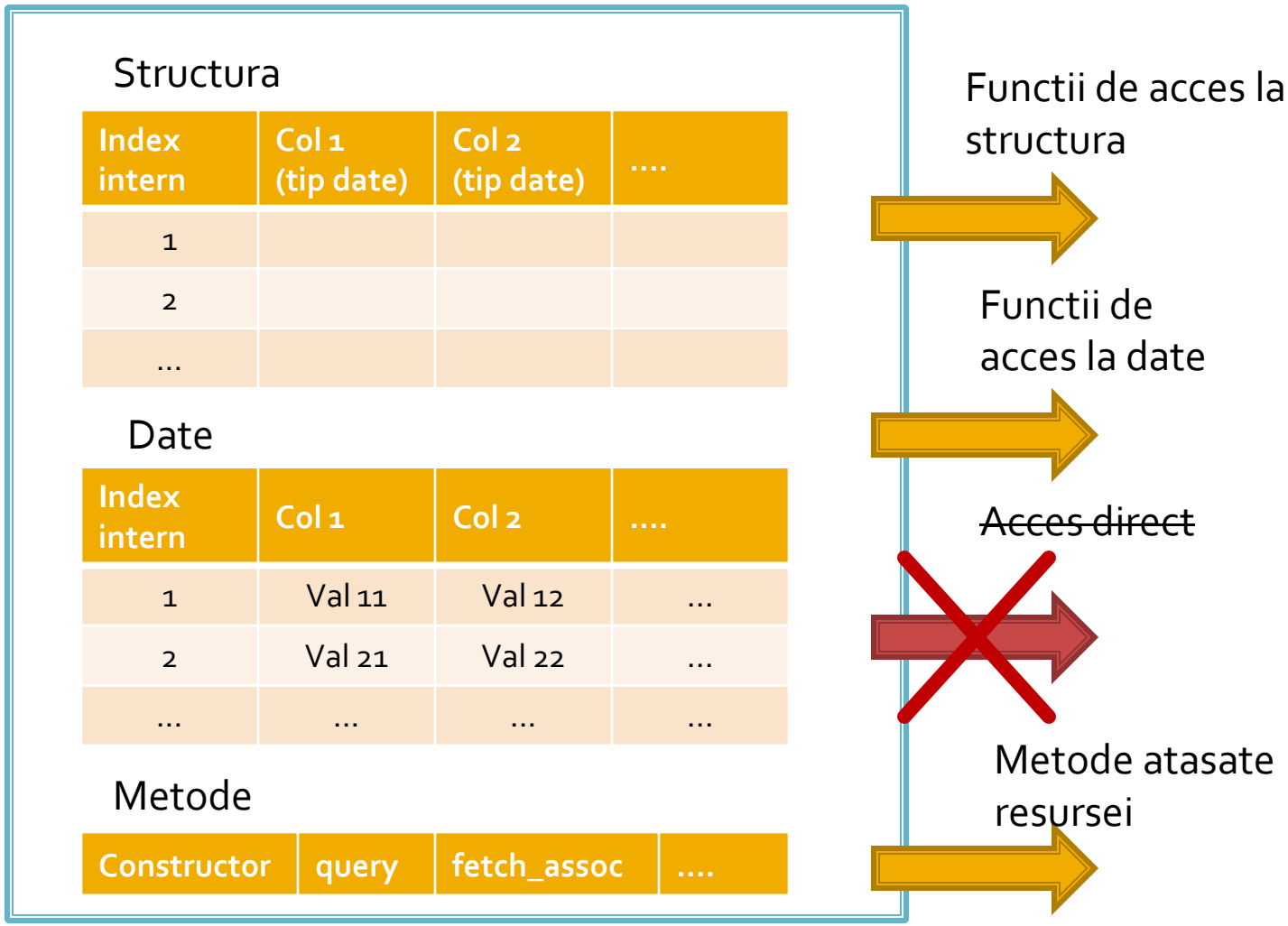
```php
<?php
$mysqli = mysqli_connect("example.com", "user", "password", "database");
$res = mysqli_query($mysqli, "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = mysqli_fetch_assoc($res);
echo $row['_msg'];

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

$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'];
?>
```

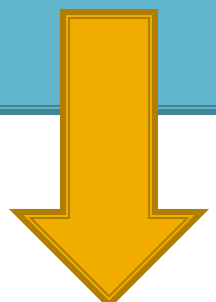
Resurse MySQL – mysqli



Conversia la mysqli (obligatorie)

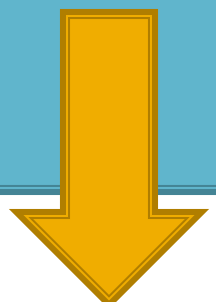
- exemplul anterior

```
$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 = 4GB

MySql – tipuri de date

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

Metode de stocare

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

Limbaj 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 cuvantului 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(nume," ",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)

Interogari SQL

Interogari

- Interogarile 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 aspra 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
- Fisierele 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
- Fisierele SQL “backup” contin succesiunea DROP..., CREATE... pentru a inlocui datele in intregime

Interogari SQL

Interogari

- Interogarile 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 acepta functii aggregate, dar se aplica la sfarsit, exact inainte de a fi trimise datele clintului, **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 aggregate (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 cuvantului 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 SELECT * FROM clienti INNER JOIN comenzi USING (id_client);
```

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 SELECT * FROM clienti INNER JOIN comenzi ON (clienti.id_client=comenzi.id_client);
```

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 SELECT * FROM clienti NATURAL JOIN comenzi;
```

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

LEFT JOIN

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

SQL Query Area

```
1 SELECT * FROM clienti LEFT OUTER JOIN comenzi USING(id_client);
```

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

RIGHT JOIN

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

SQL Query Area

```
1 SELECT * FROM clienti RIGHT OUTER JOIN comenzi USING(id_client);
```

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

SQL Query Area

```
1 SELECT * FROM comenzi RIGHT OUTER JOIN clienti USING(id_client);
```

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

JOIN

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

UNION

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

SQL Query Area

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

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

Subquery

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

Subquery

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

Subquery

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

Laborator 2 / 2011-2012

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

MySql – Server Windows 2000

Mini – Indrumar practic
Lucru cu bazele de date

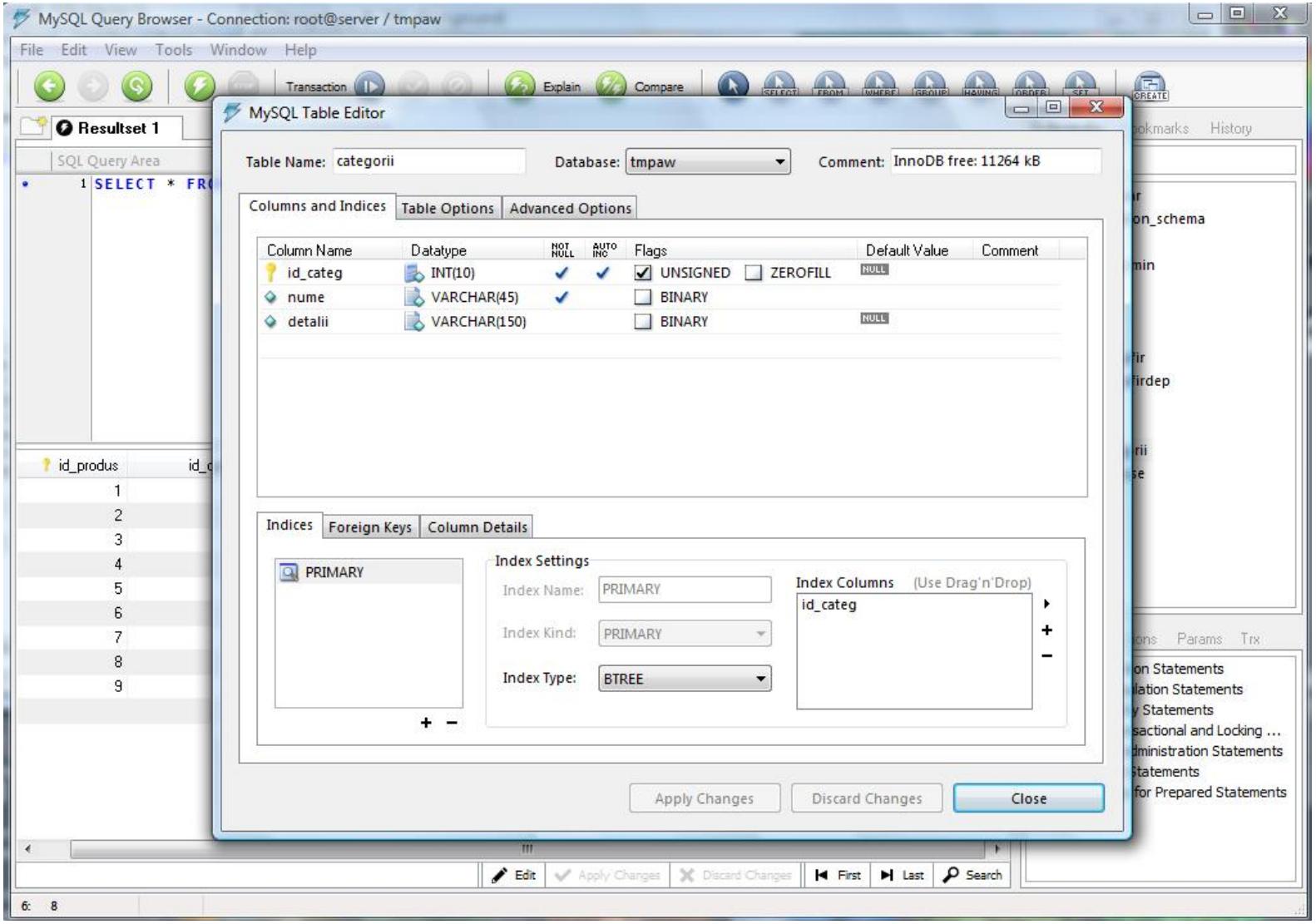
Realizarea bazei de date

- 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

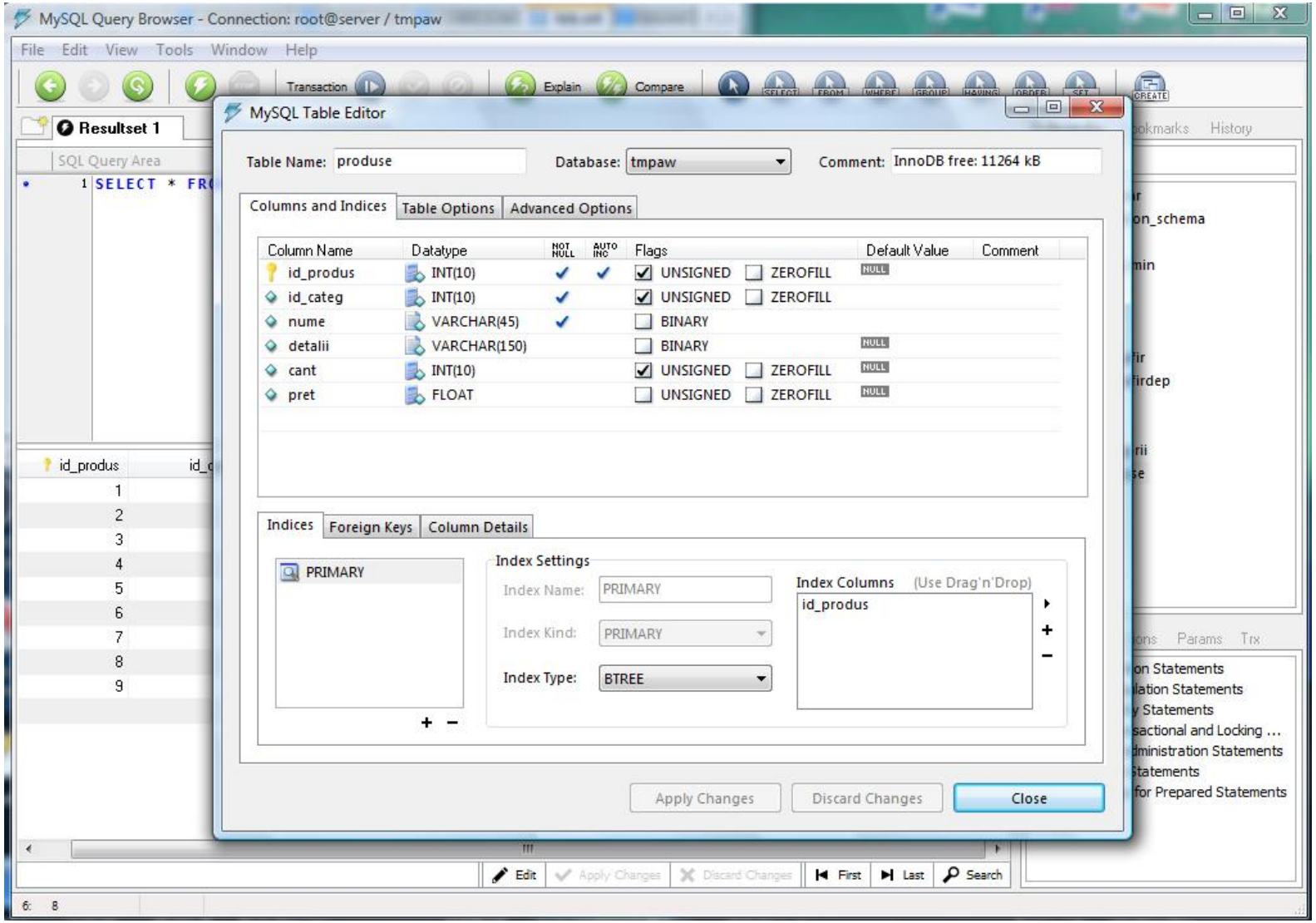
Introducere tabele

- 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



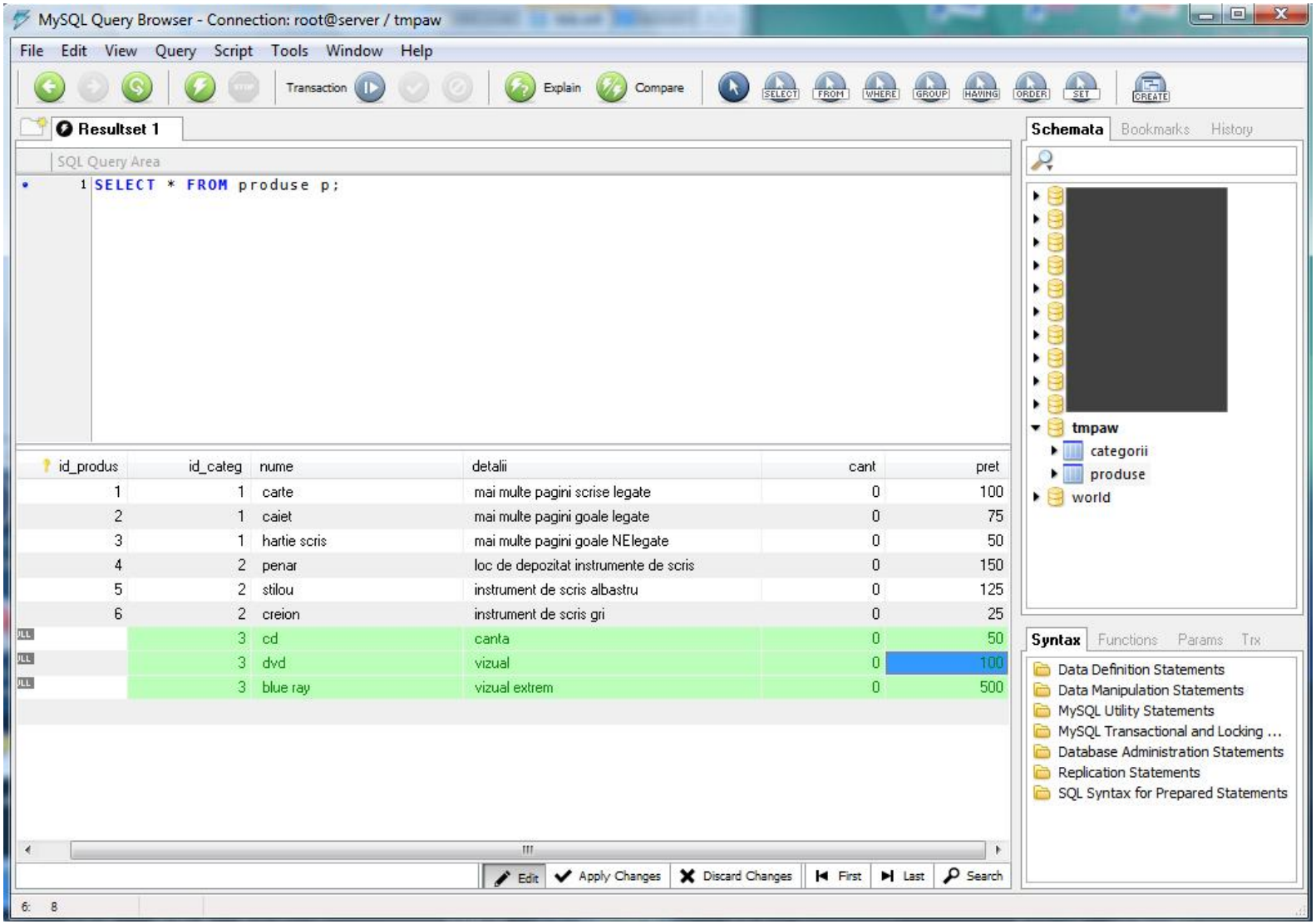
Tabel Produse



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

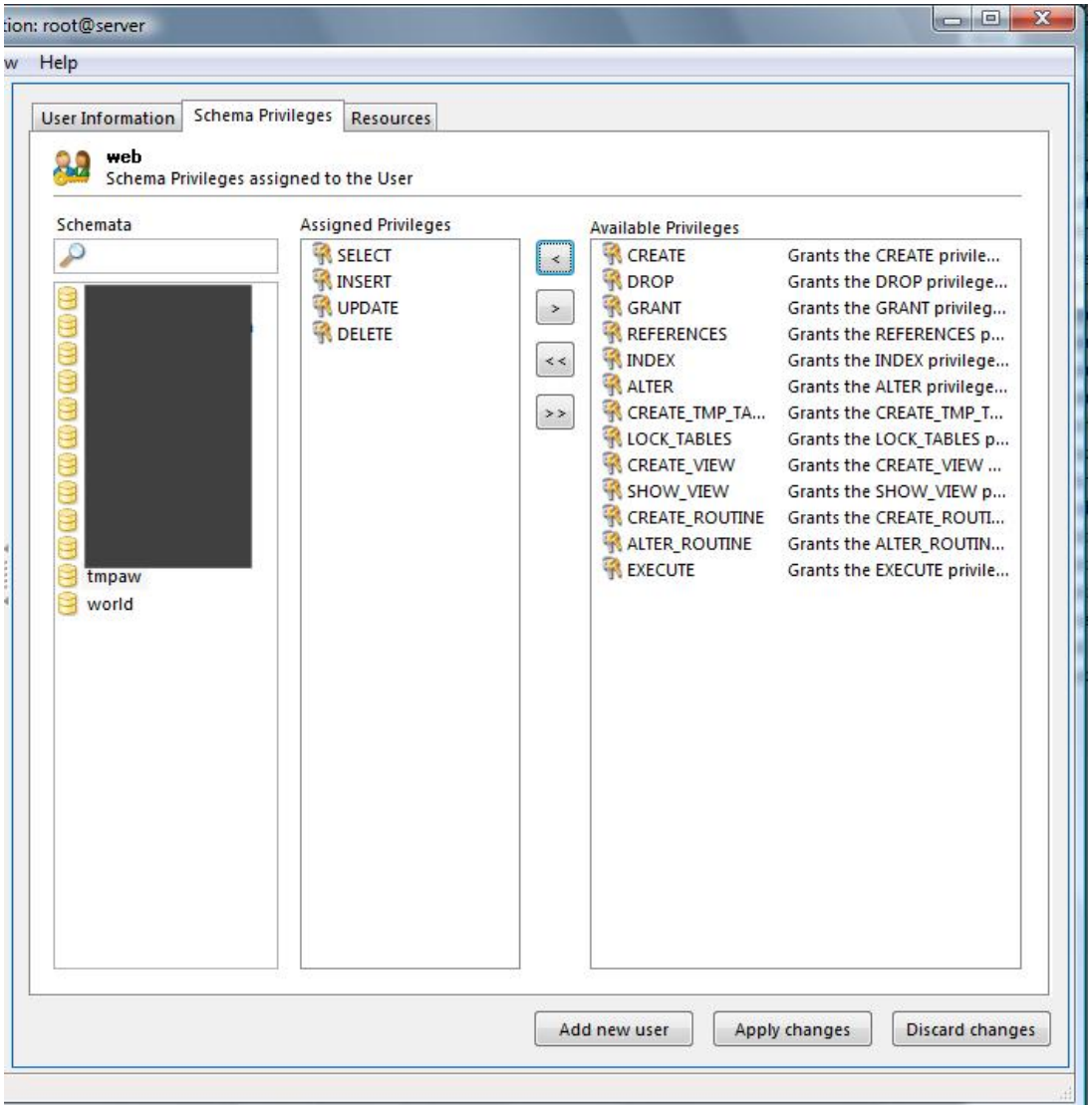
Introducere date initiale



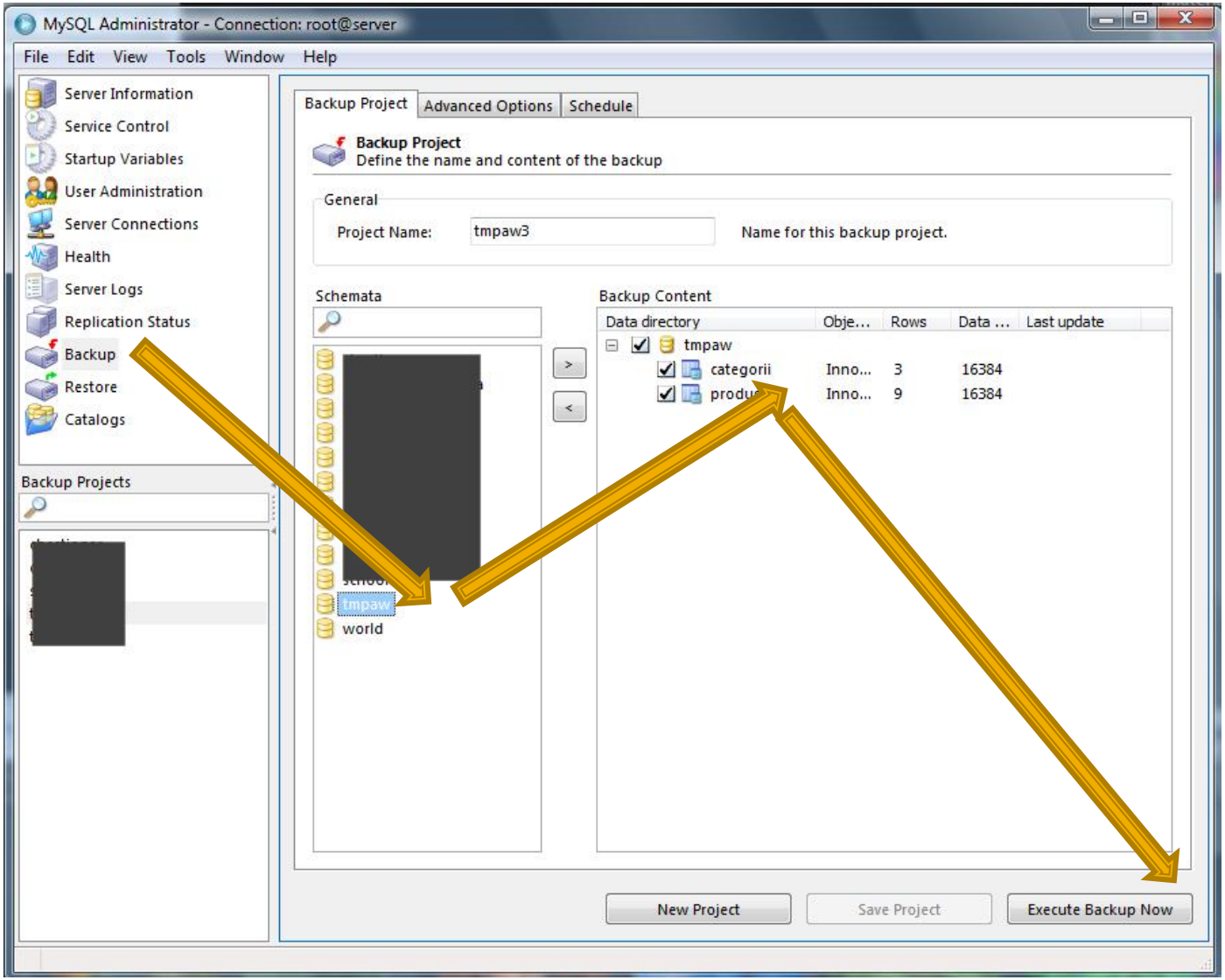
Backup, Restore, drepturi de acces

- 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


MySQL Administrator - Connection: root@server
File Edit View Tools Window Help
Server Information
Service Control
Startup Variables
User Administration
Server Connections
Health
Server Logs
Replication Status
Backup
Restore
Catalogs
Backup Projects
Backup Project
Advanced Options
Schedule
Backup Project
Define the name and content of the backup
General
Project Name:
tmpaw3
Name for this backup project.
Schemata
Backup Content
Data directory
Obje...
Rows
Data ...
Last update
tmpaw
categorii
Inno...
3
16384
Inno...
9
16384
tmpaw
world
New Project
Save Project
Execute Backup Now

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/interogarile SQL necesare pentru crearea bazei de date si popularea ei cu date

Script SQL Backup - utilitate

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

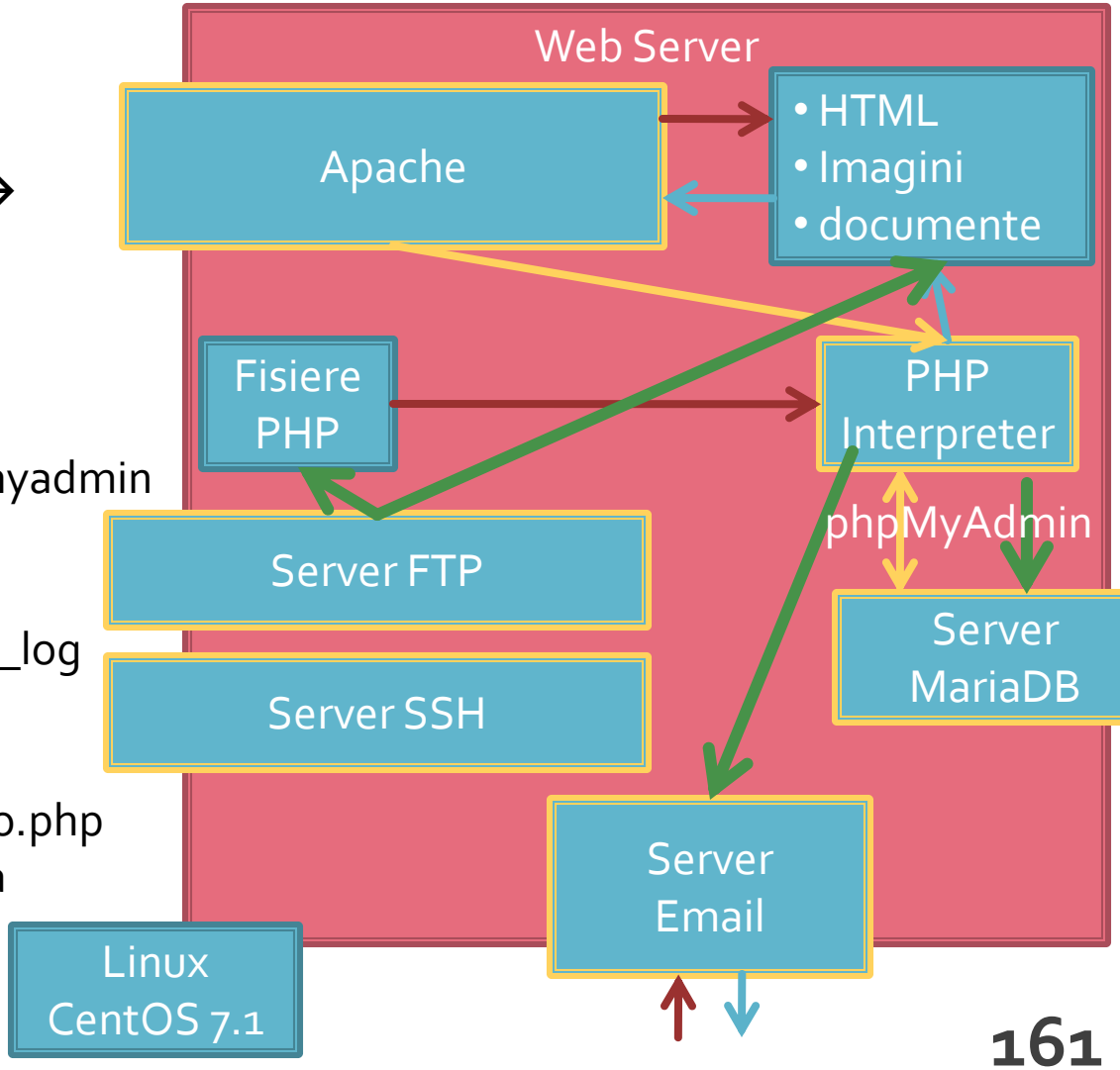
```
CREATE DATABASE IF NOT EXISTS tmpaw;
USE tmpaw;

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

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

MySql – Server Centos 7.1

Mini – Indrumar practic
Lucru cu bazele de date

Utilizare LAMP

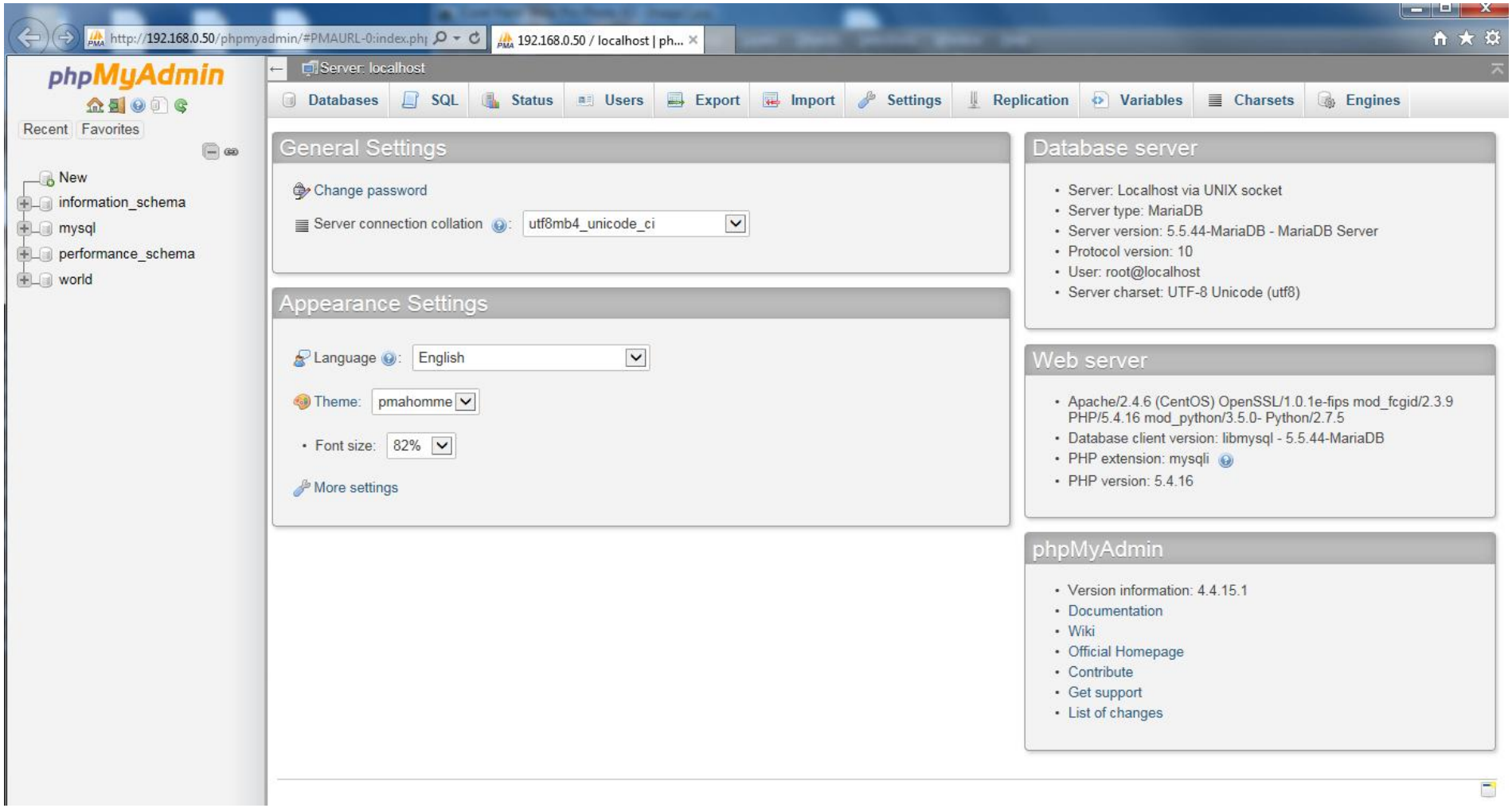
1. login → root:masterrc
2. ifconfig → 192.168.30.5
3. putty.exe → 192.168.30.5 → SSH → root:masterrc (remote login)
4. [alte comenzi linux dorite]
5. FTP → Winscp → SFTP → student:masterrc@192.168.30.5
6. MySql → http://192.168.30.5/phpmyadmin → root:masterrc
7. Apache Error Log →
7a. putty → nano /var/log/httpd/error_log
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

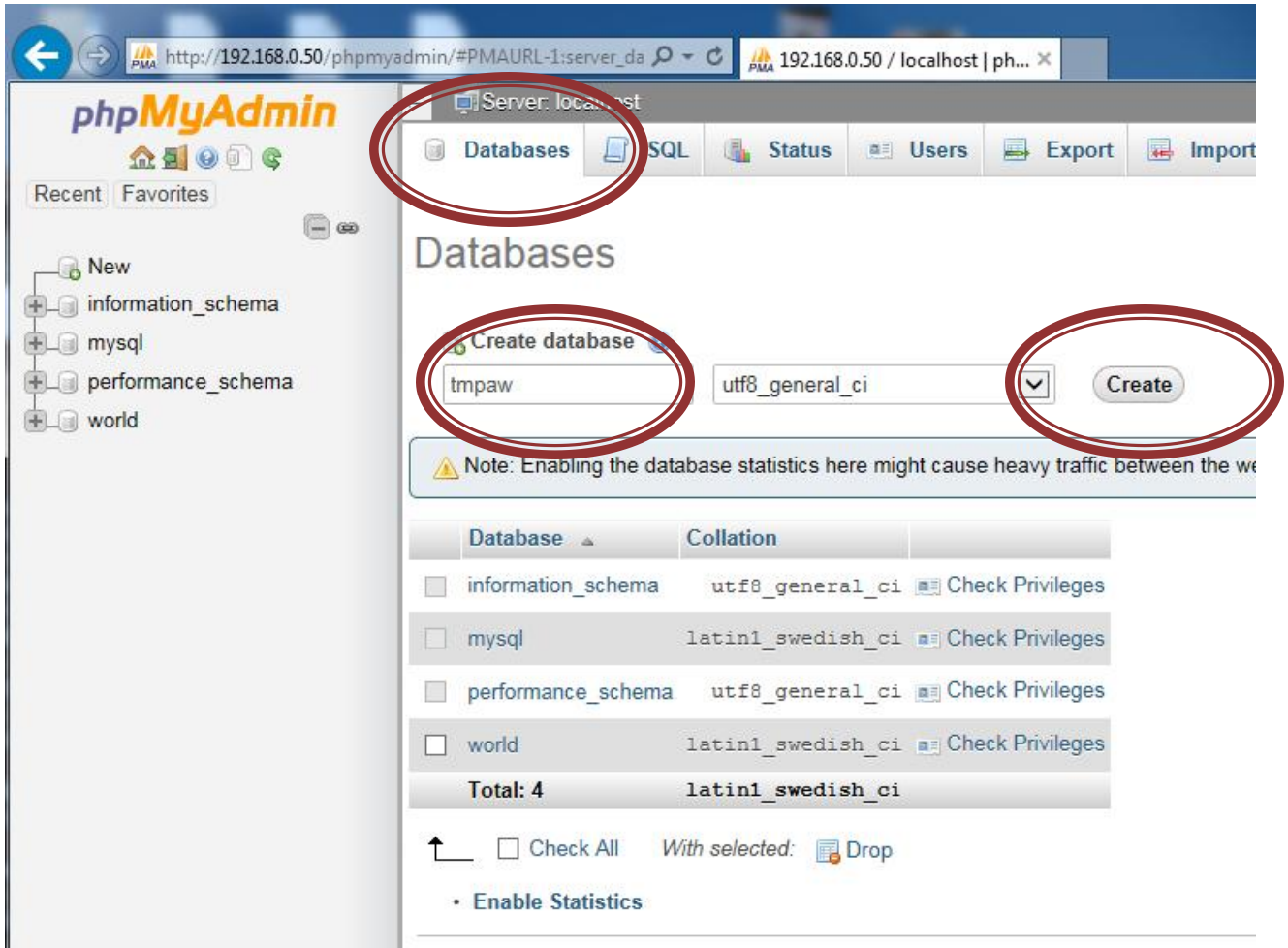
- http://192.168.30.5/phpmyadmin
  - root
  - parola administrator **MySql/MariaDB** (masterrc)

PhpMyAdmin


http://192.168.0.50/phpmyadmin/#PMAURL-0:index.php
192.168.0.50 / localhost | ph...
phpMyAdmin
Recent
Favorites
New
information_schema
mysql
performance_schema
world
Server: localhost
Databases
SQL
Status
Users
Export
Import
Settings
Replication
Variables
Charsets
Engines
General Settings
Change password
Server connection collation:
utf8mb4_unicode_ci
Appearance Settings
Language:
English
Theme:
pmahomme
Font size:
82%
More settings
Database server
Server: Localhost via UNIX socket
Server type: MariaDB
Server version: 5.5.44-MariaDB - MariaDB Server
Protocol version: 10
User: root@localhost
Server charset: UTF-8 Unicode (utf8)
Web server
Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod_fcgid/2.3.9 PHP/5.4.16 mod_python/3.5.0- Python/2.7.5
Database client version: libmysql - 5.5.44-MariaDB
PHP extension: mysqli
PHP version: 5.4.16
phpMyAdmin
Version information: 4.4.15.1
Documentation
Wiki
Official Homepage
Contribute
Get support
List of changes

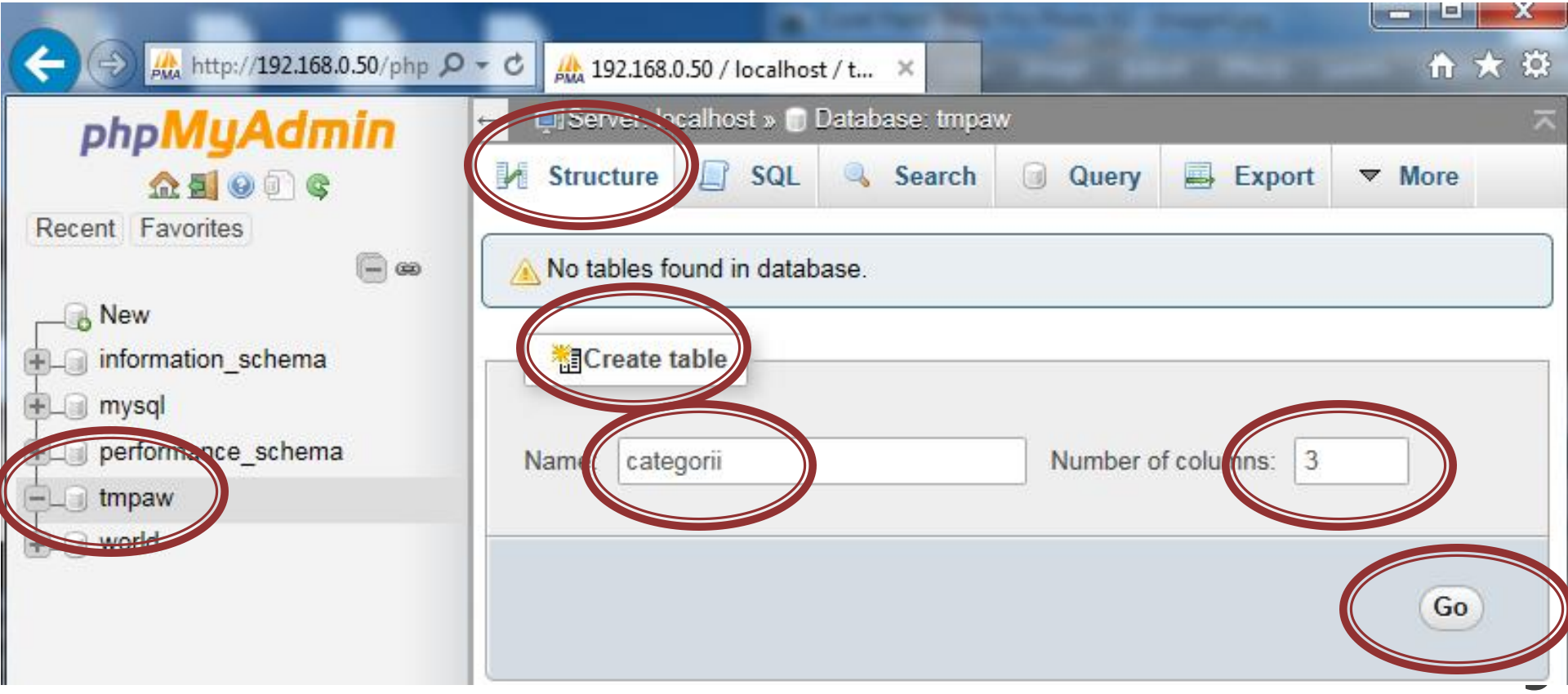
Creare Baza de Date

- Databases →"nume" →Create



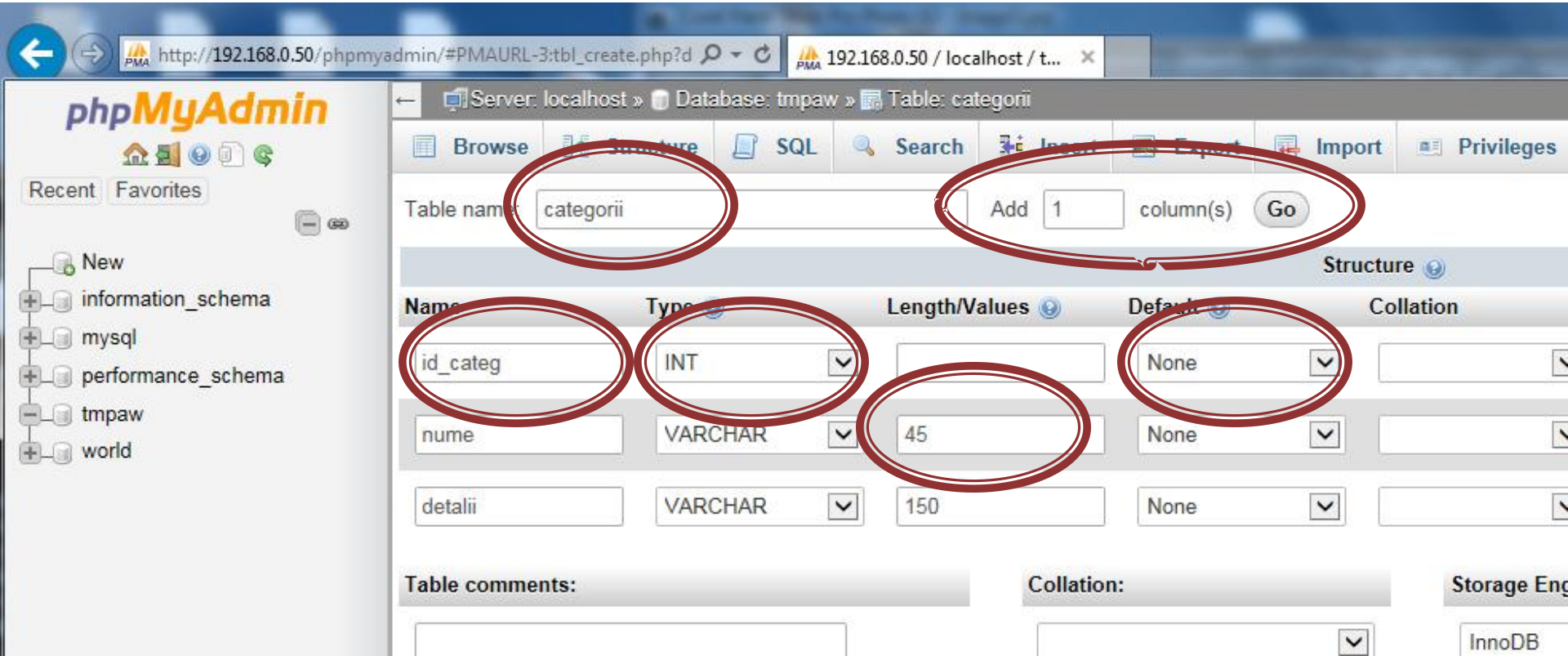
Creare tabele in baza de date

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



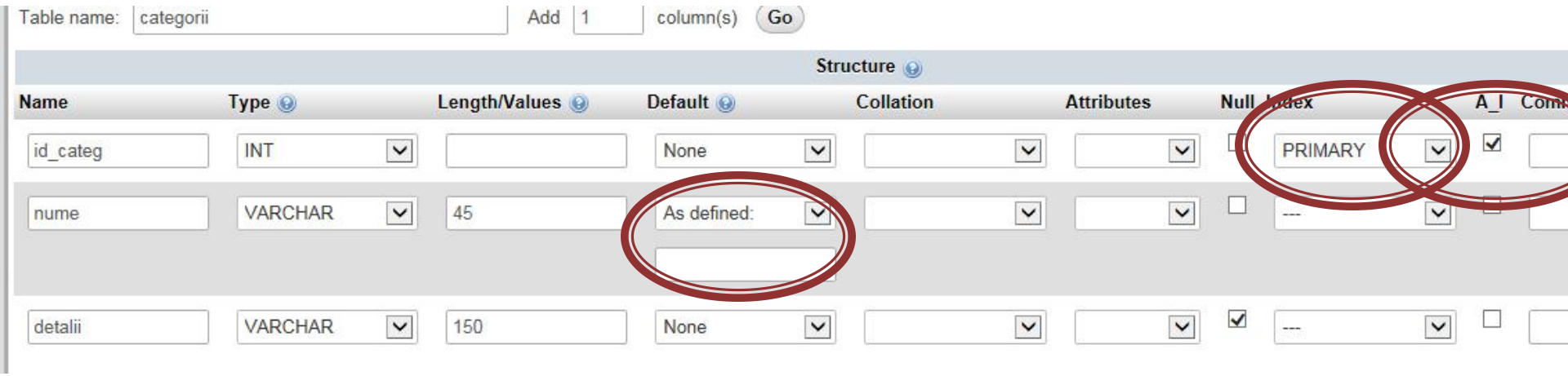
Introducere coloane, tabel categorii

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



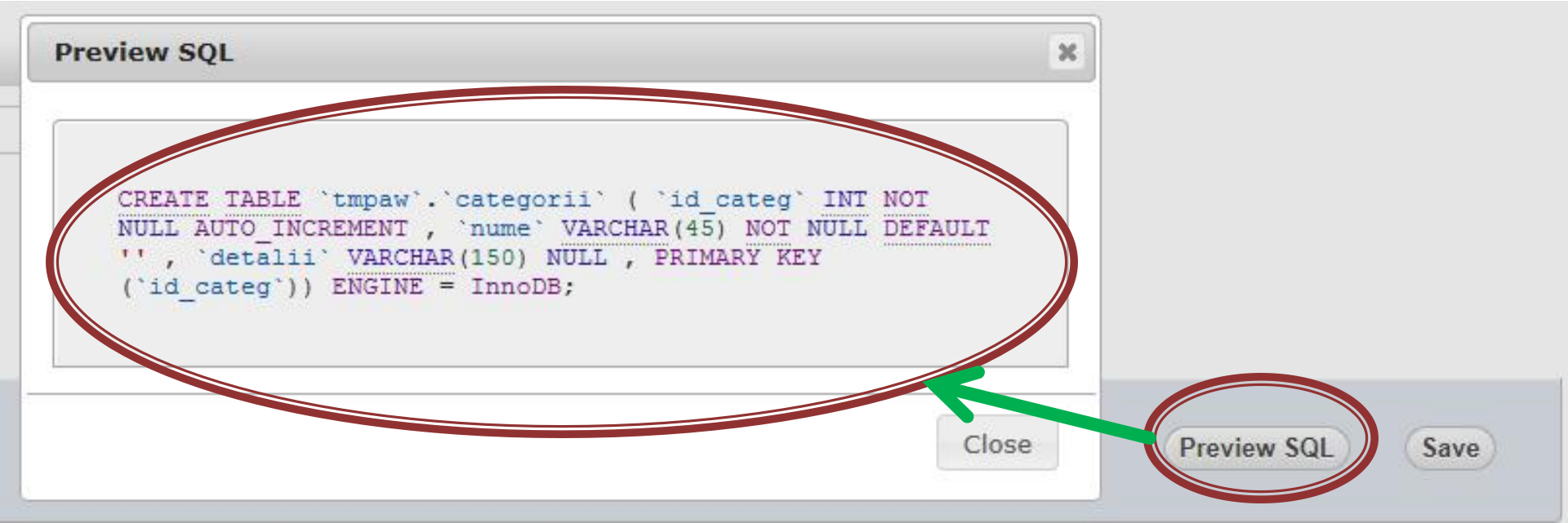
Introducere coloane

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



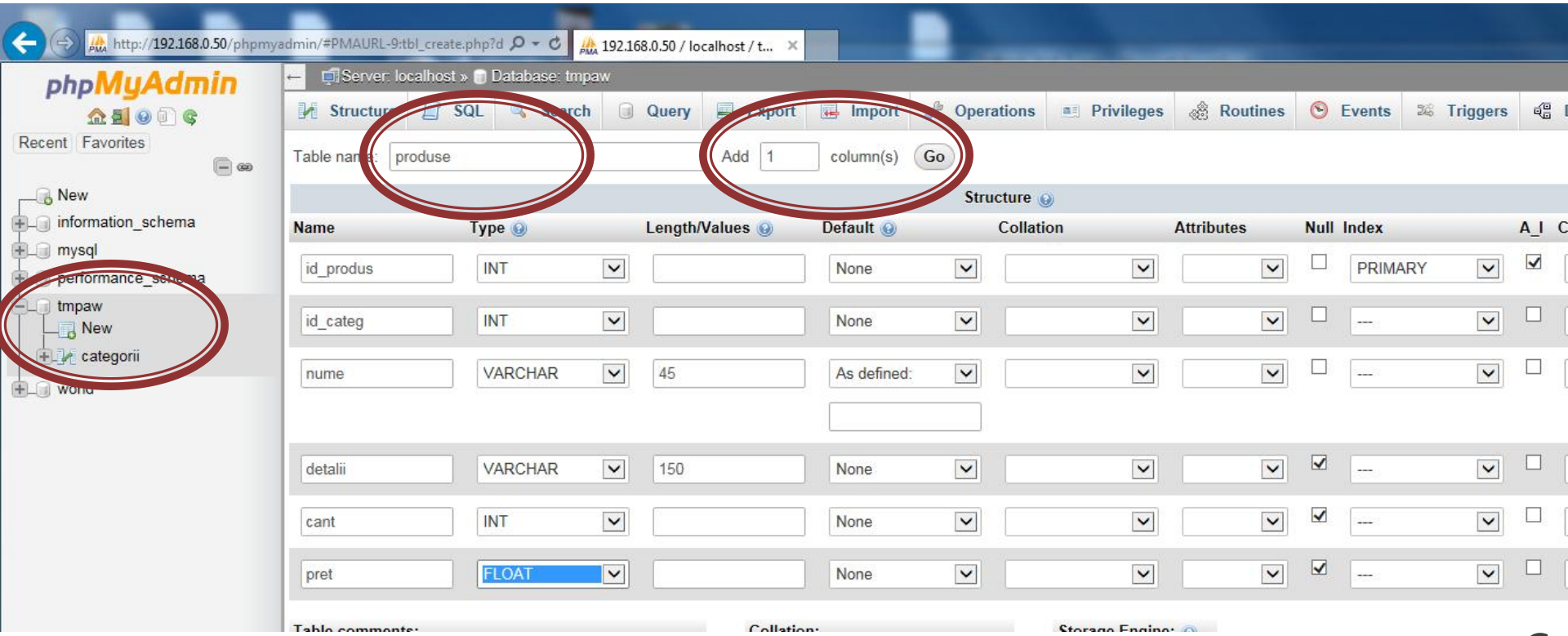
Preview SQL

- 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



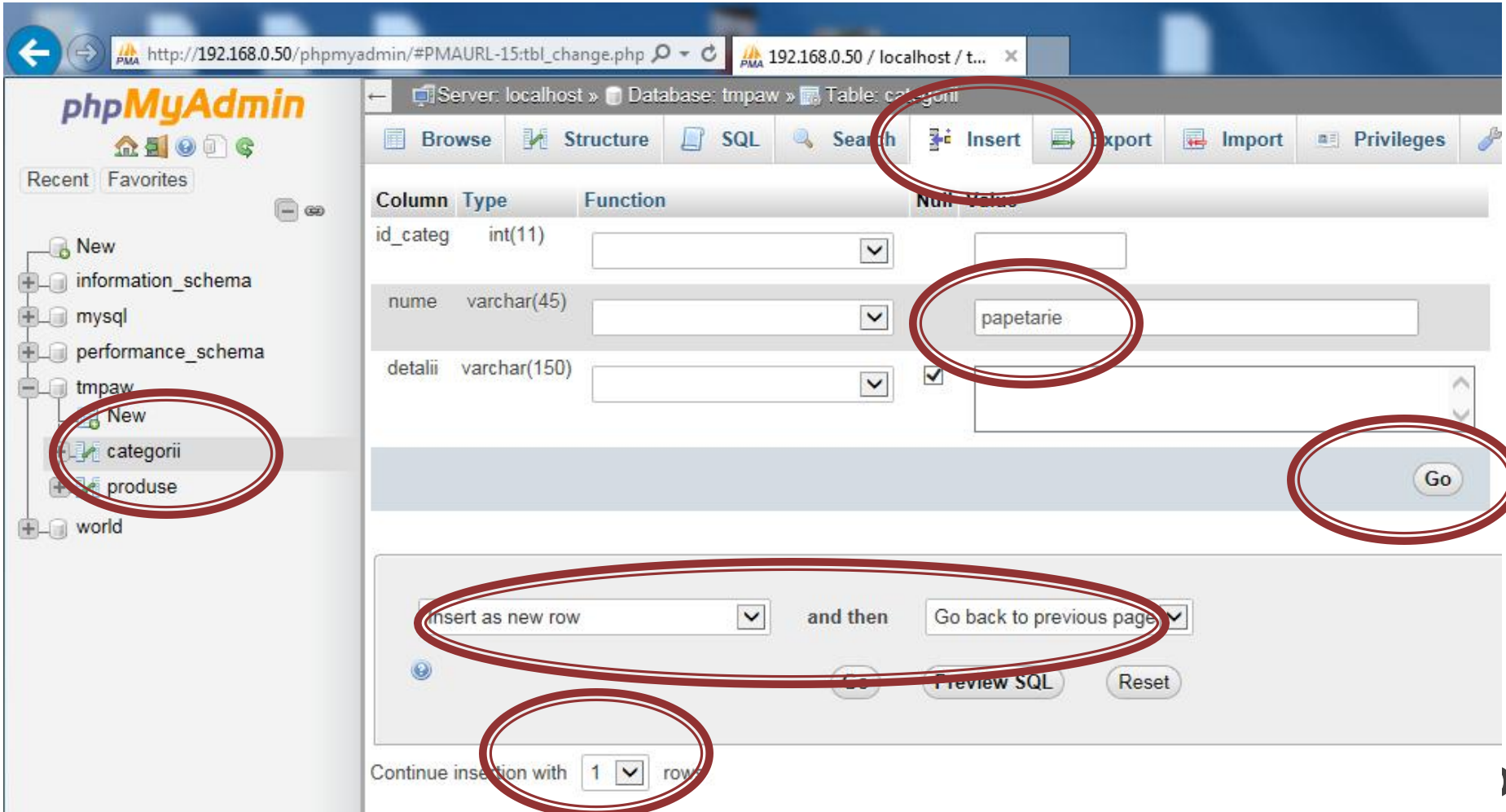
Introducere coloane, tabel produse

- New → Nume → Add Columns → …



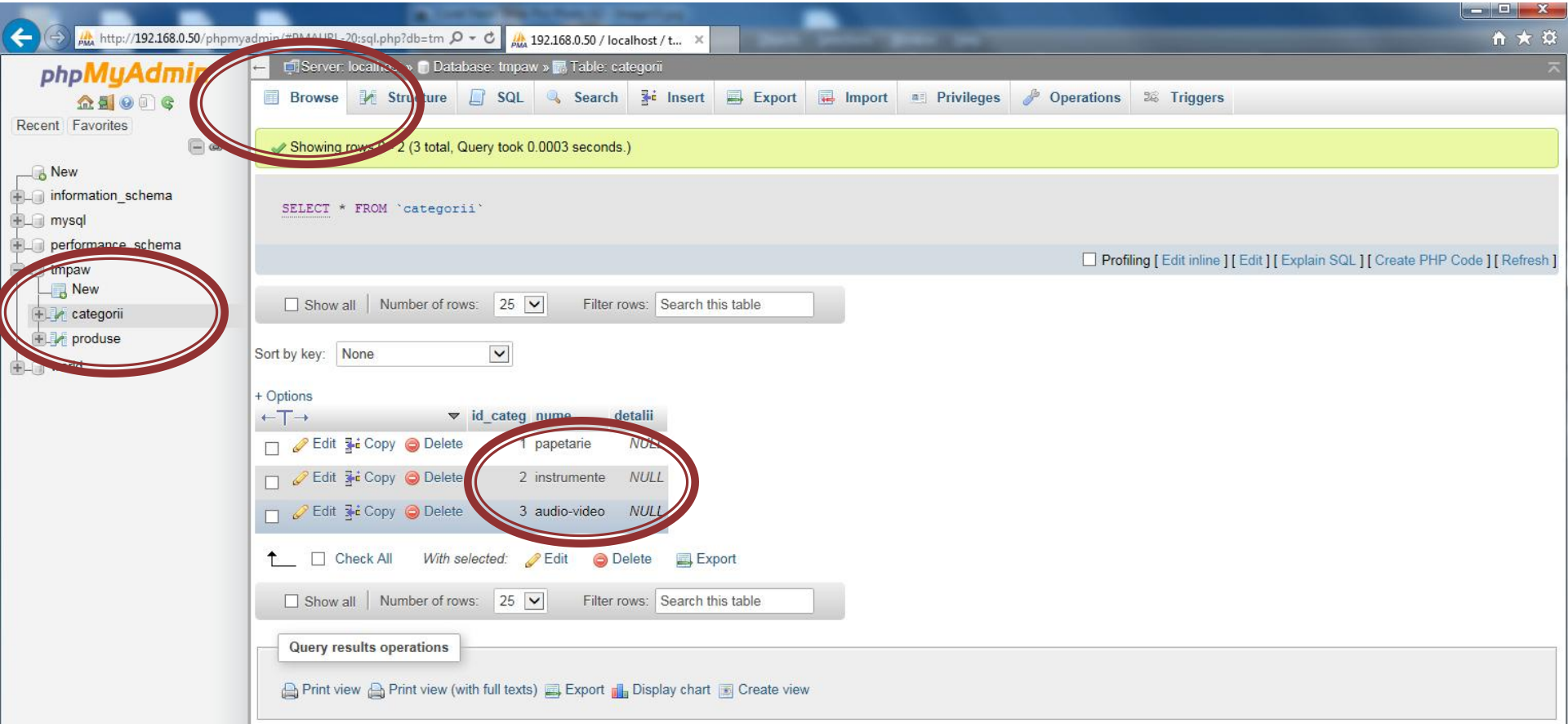
Introducere date initiale (interfata)

- Tabel → Insert → Completare → Go



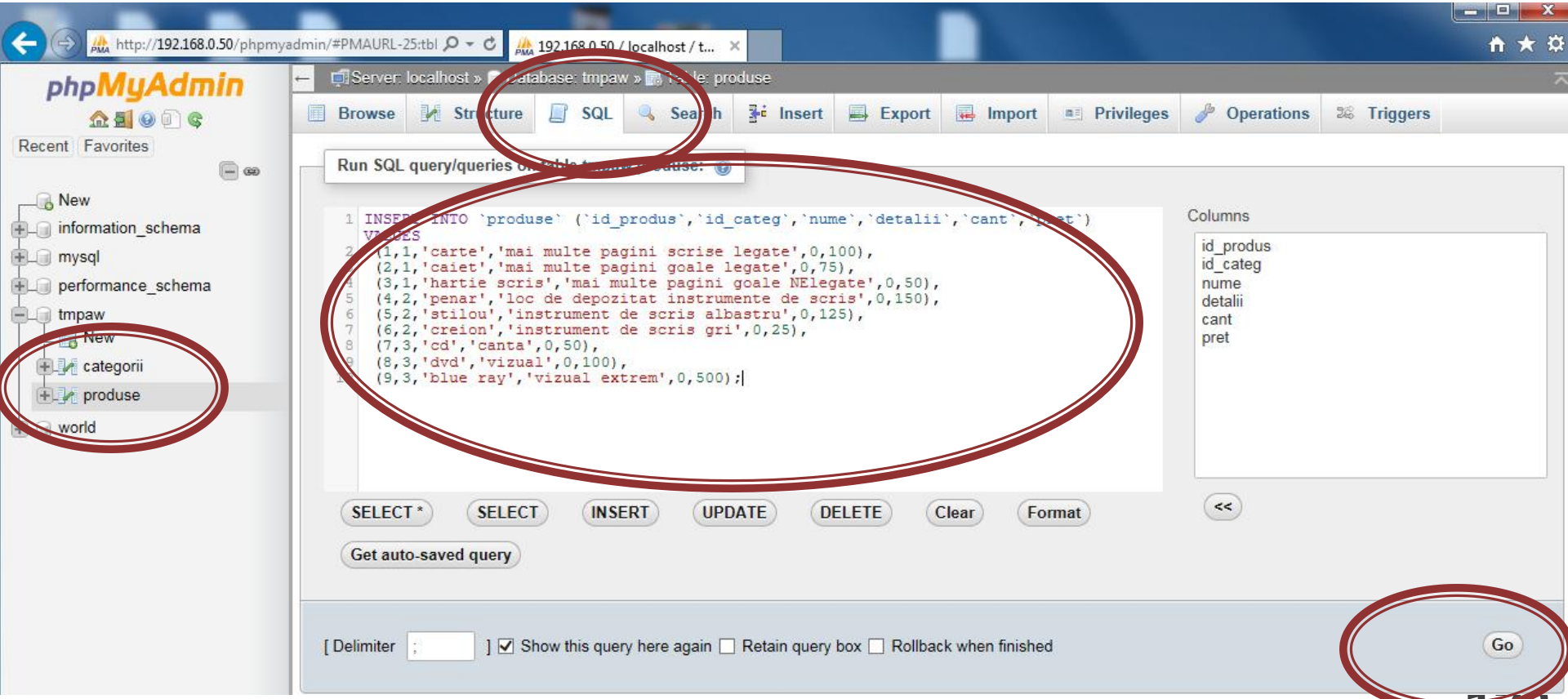
Vizualizare date existente

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

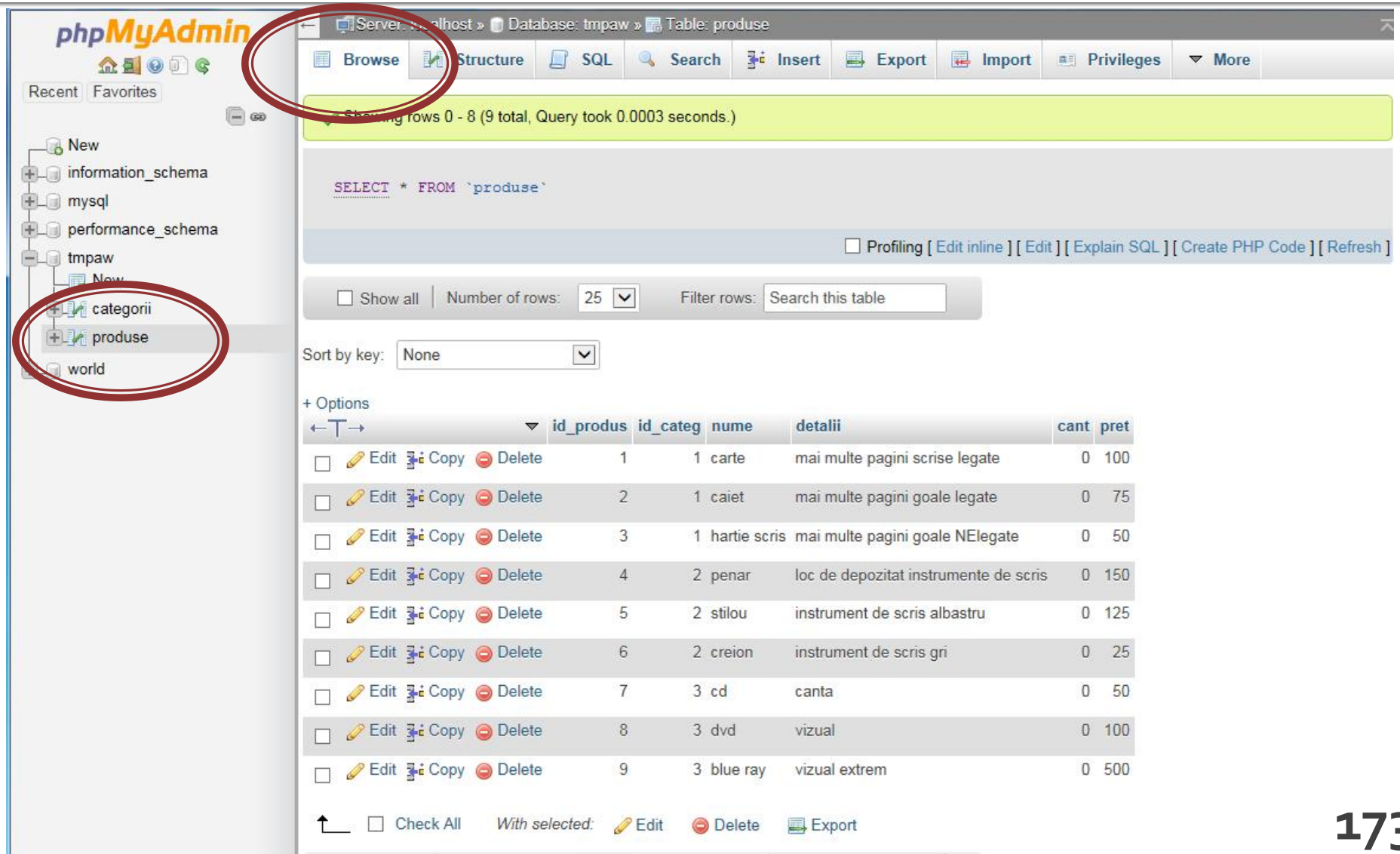


Introducere date initiale (SQL)

- Tabel → SQL → completare → Go

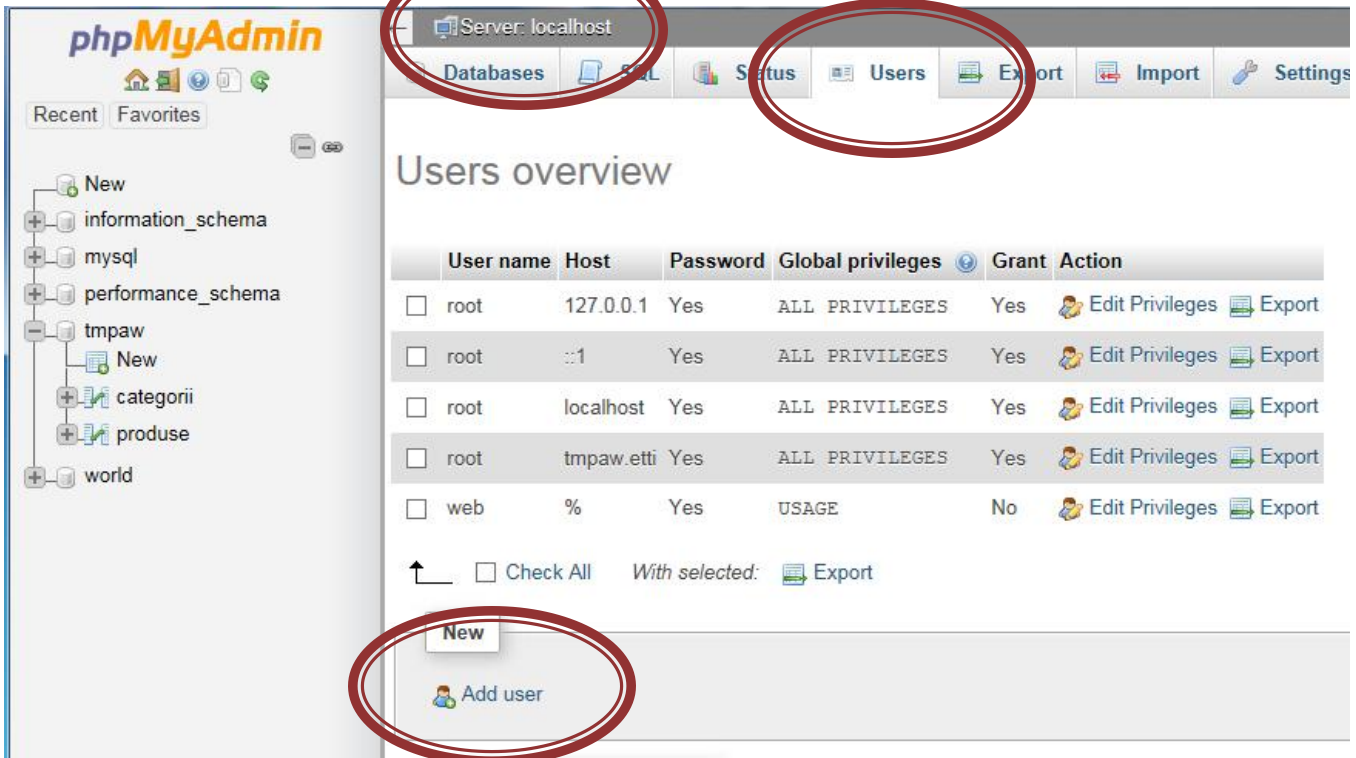


Tabel produse



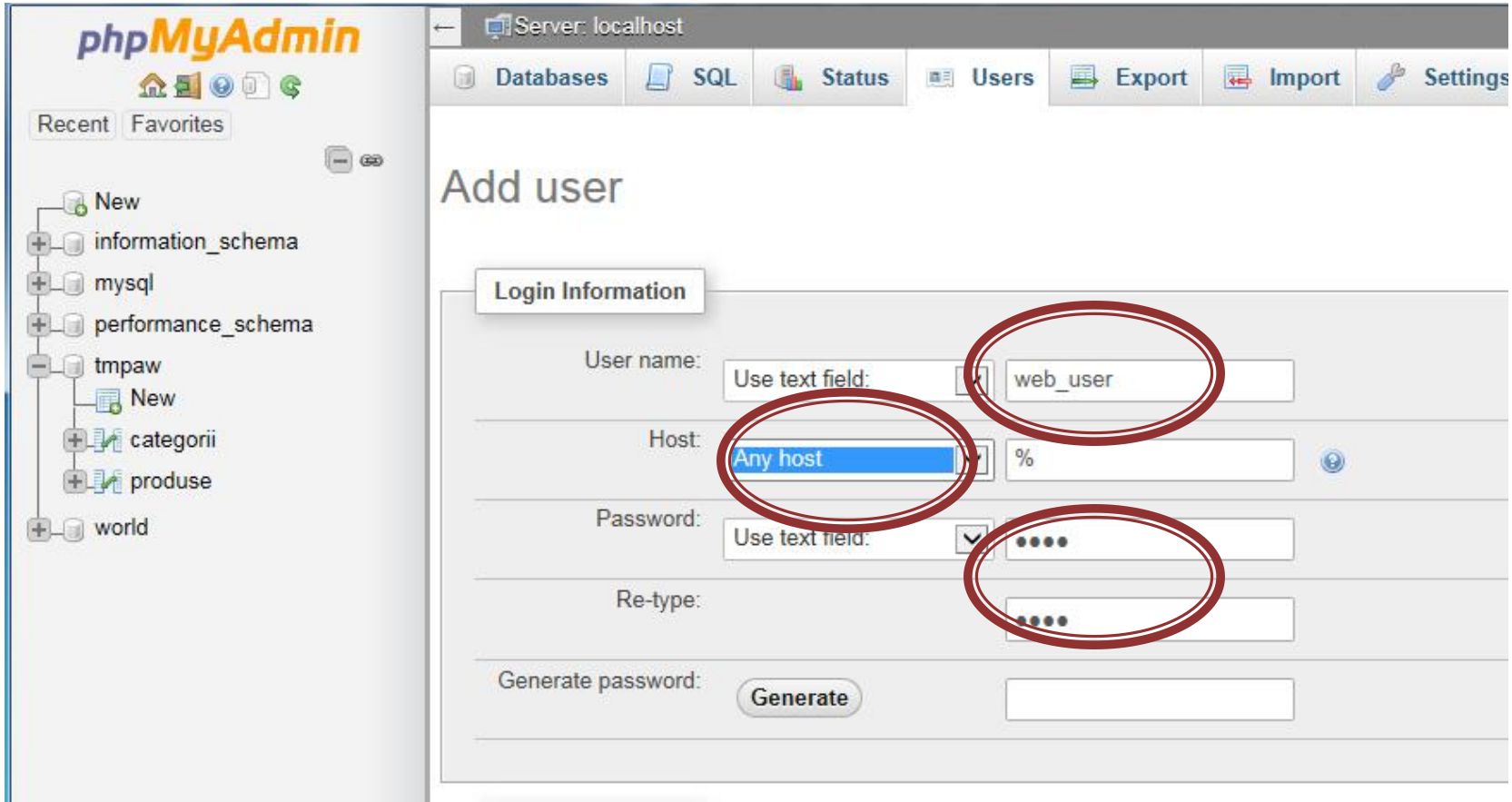
Adaugare utilizator

- Server → Users → Add user



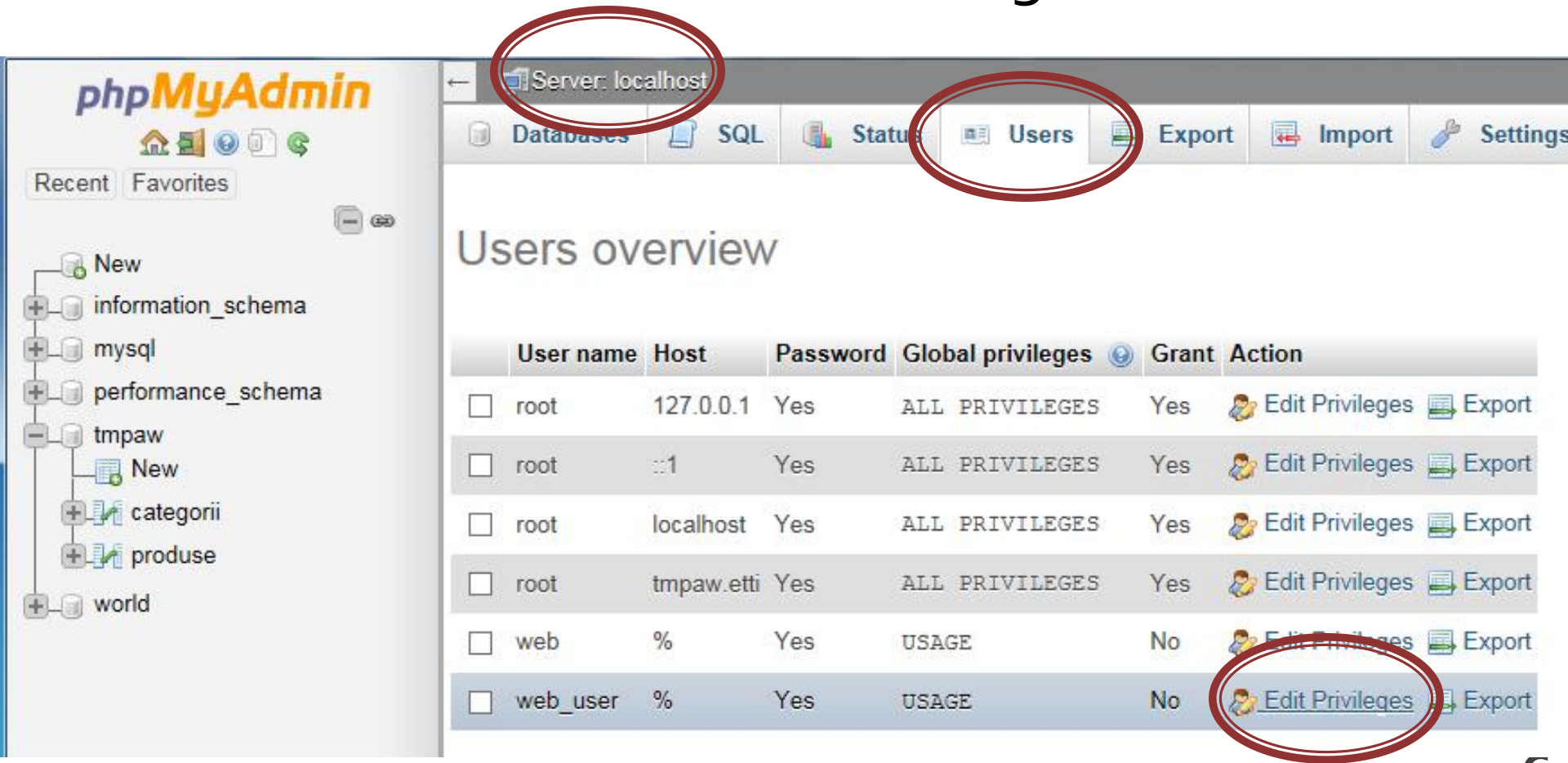
Adaugare utilizator

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



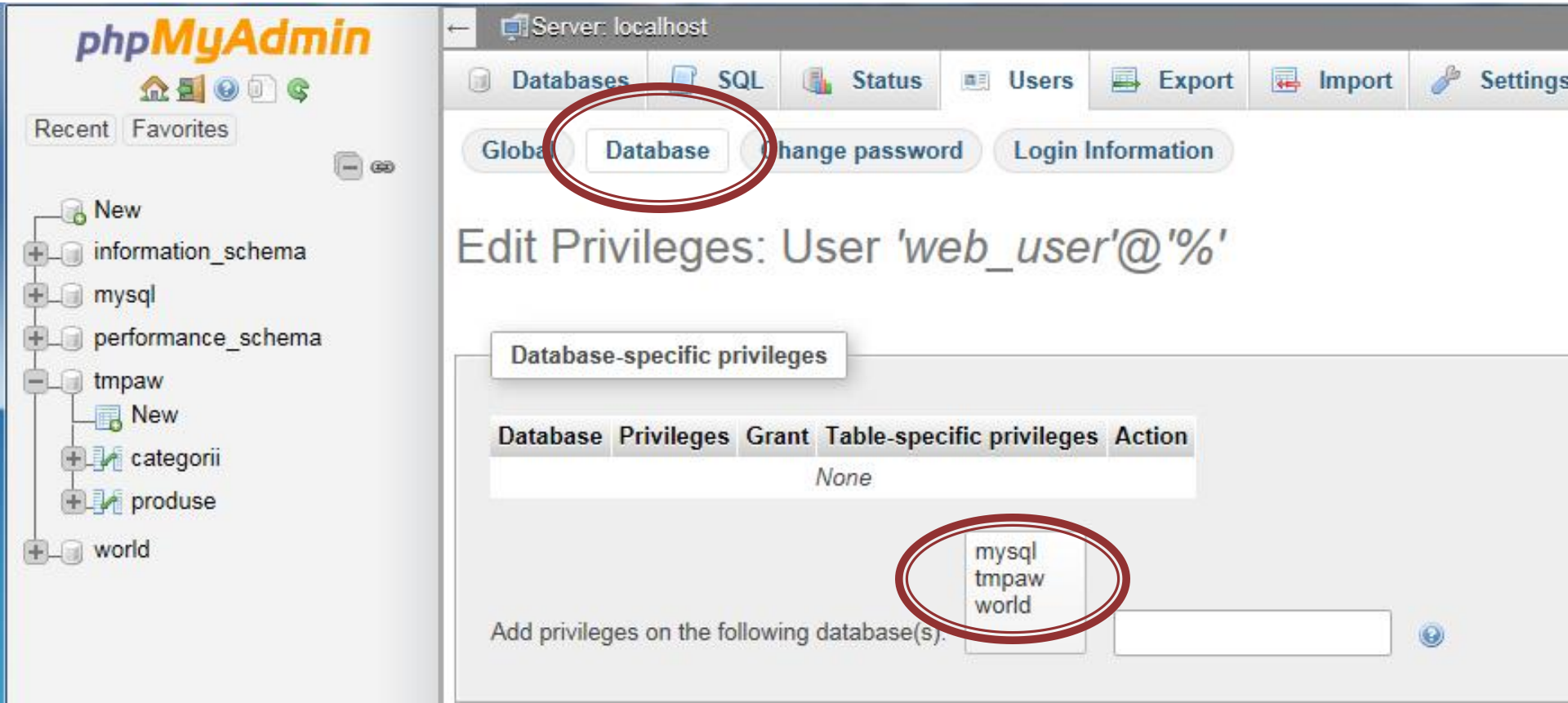
Drepturi de acces

- Server → Users → Edit Privileges



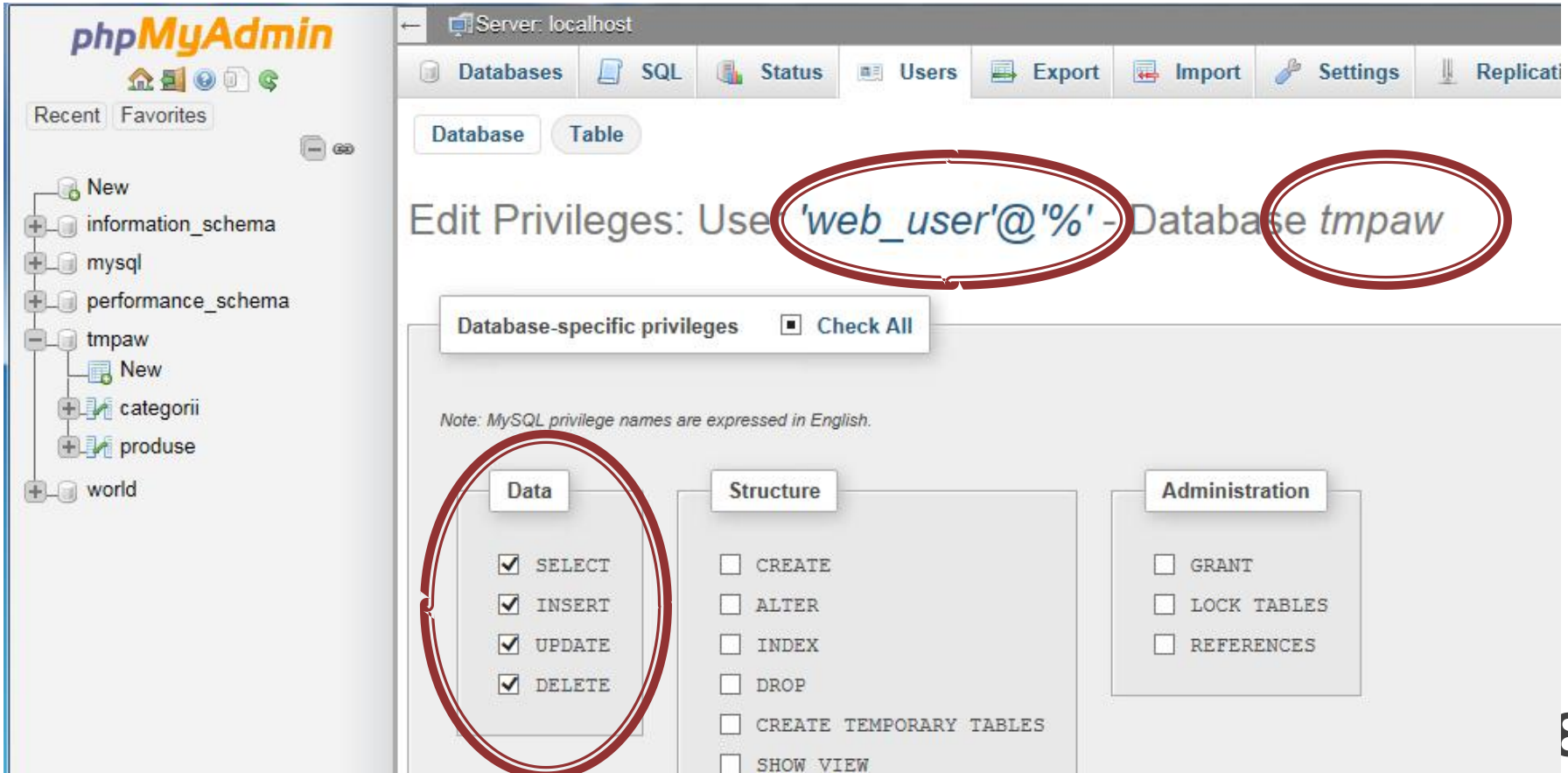
Drepturi de acces

- Database → nume → Go



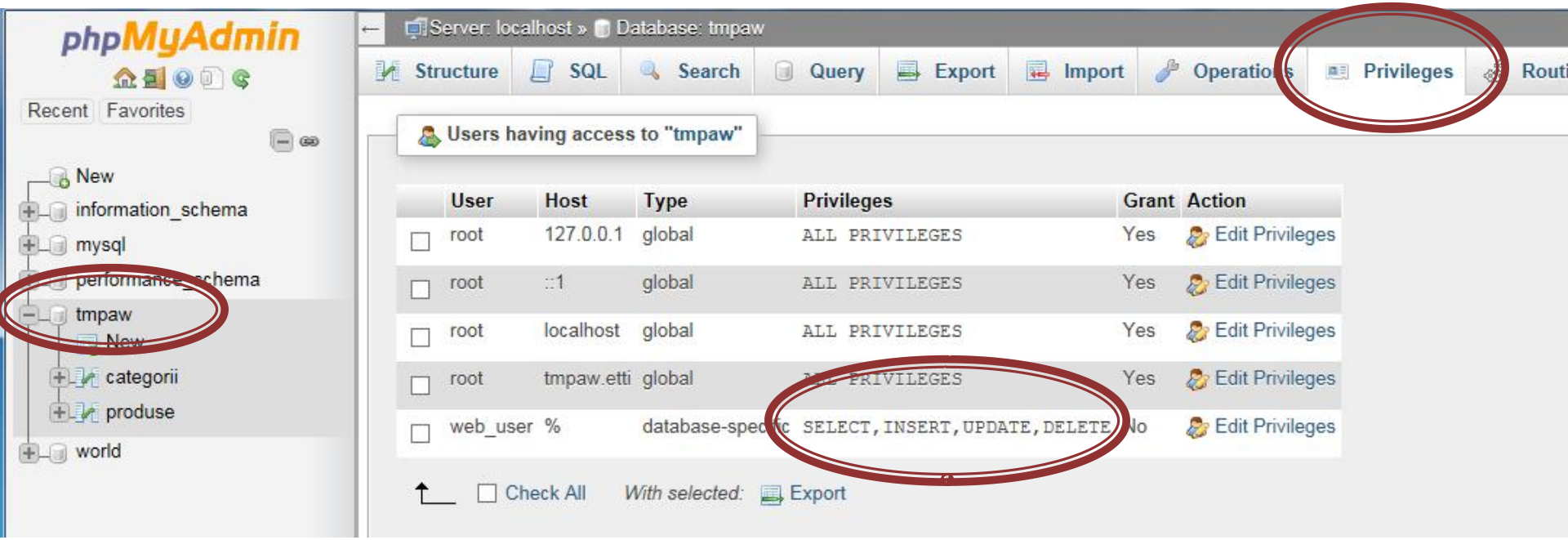
Drepturi de acces

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



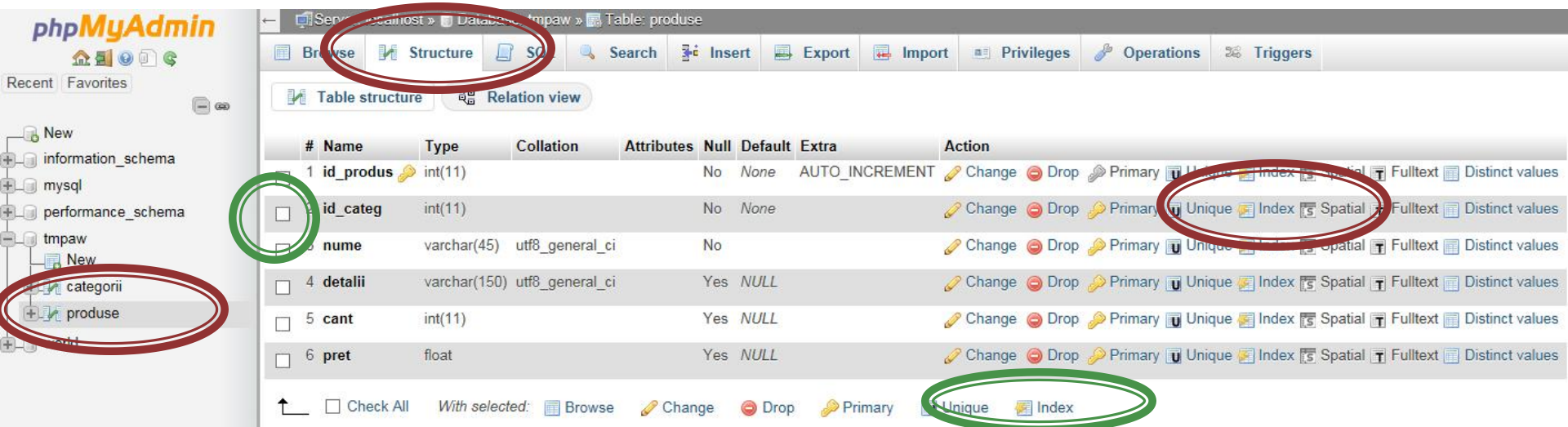
Drepturi de acces, verificare

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



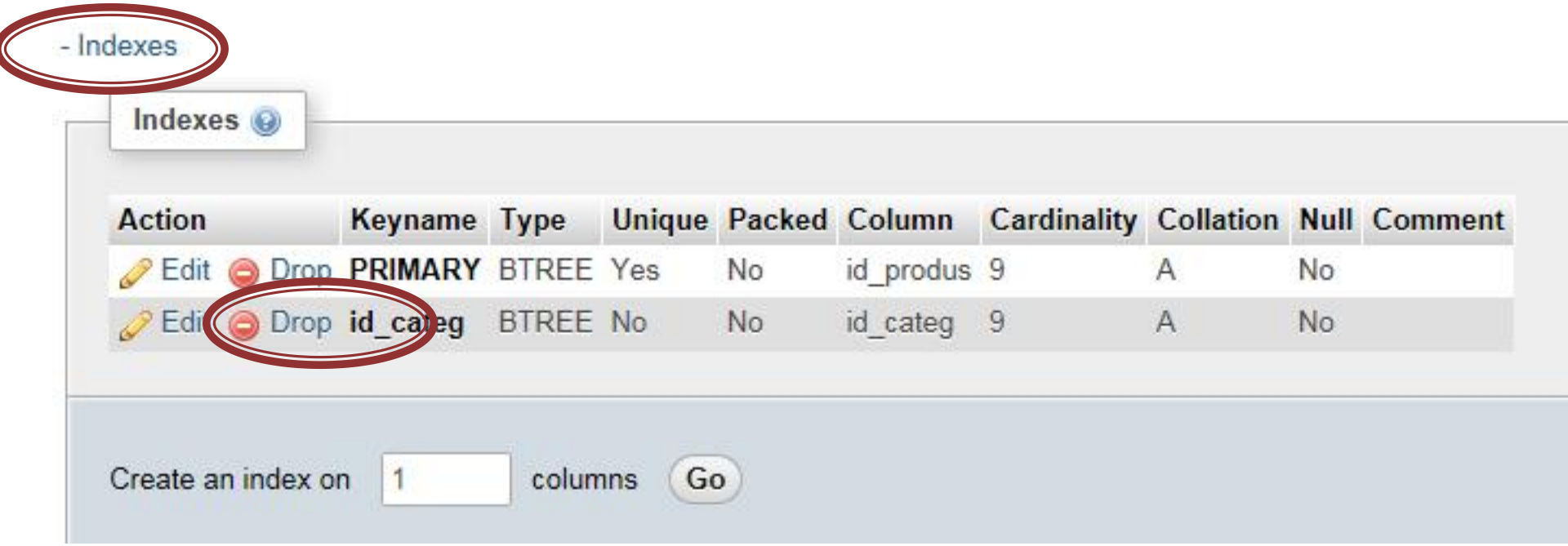
Index

- 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



Verificare/Stergere index

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

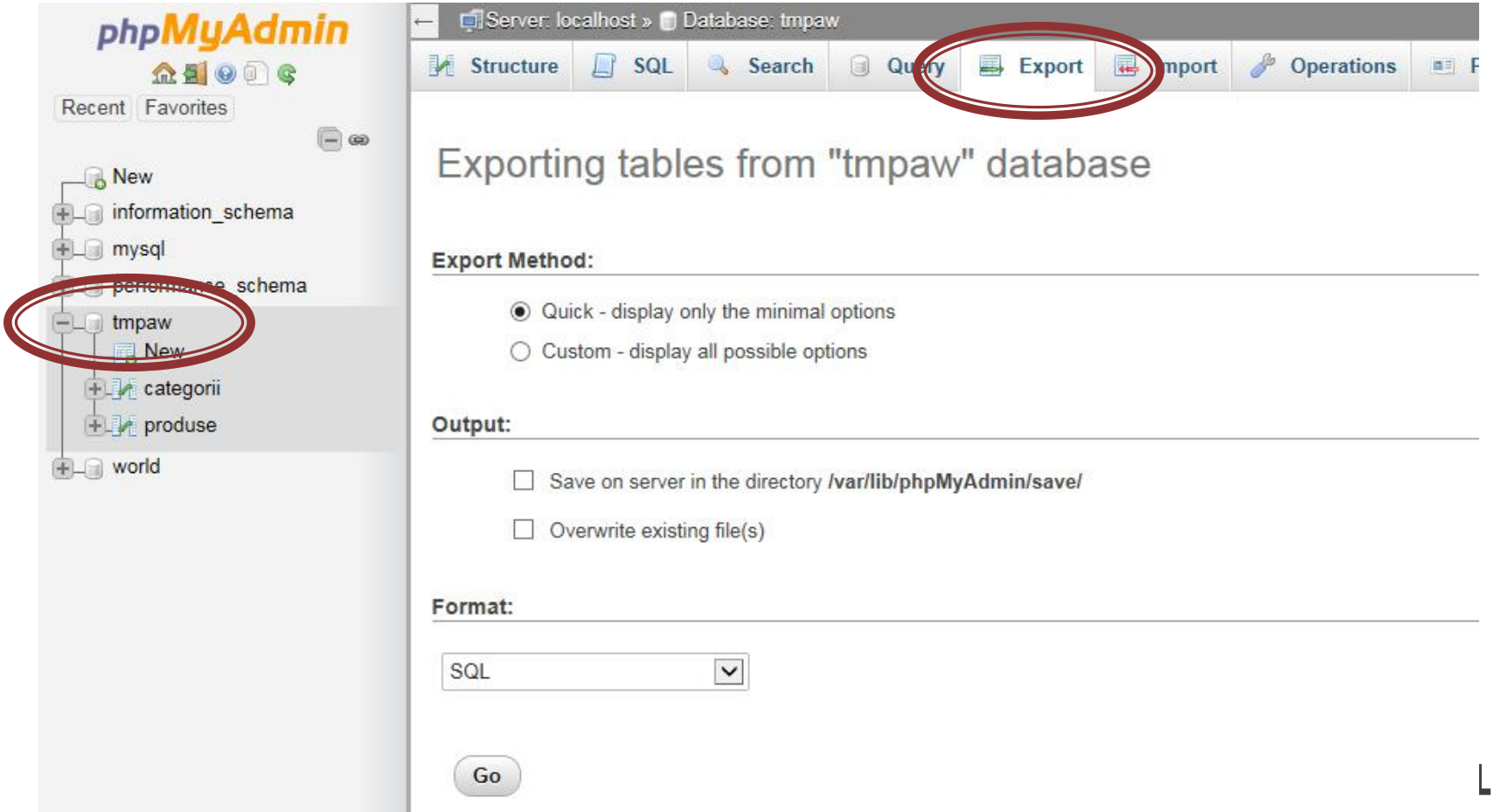


Backup, Restore

- Ca si in cazul Windows 2000 facilitatea de Backup realizeaza un script SQL care contine structura si datele exprimate sub forma de interogari SQL
- O deosebire intre PhpMyAdmin si aplicatiile specifice MySql (aceleasi de pe Windows 2000 sau MySql Workbench) este absenta liniilor de creare a bazei de date
  - CREATE DATABASE IF NOT EXISTS tmpaw;
  - USE tmpaw;
- La utilizarea PhpMyAdmin trebuie sa se creeze manual inaintea restaurarii baza de date

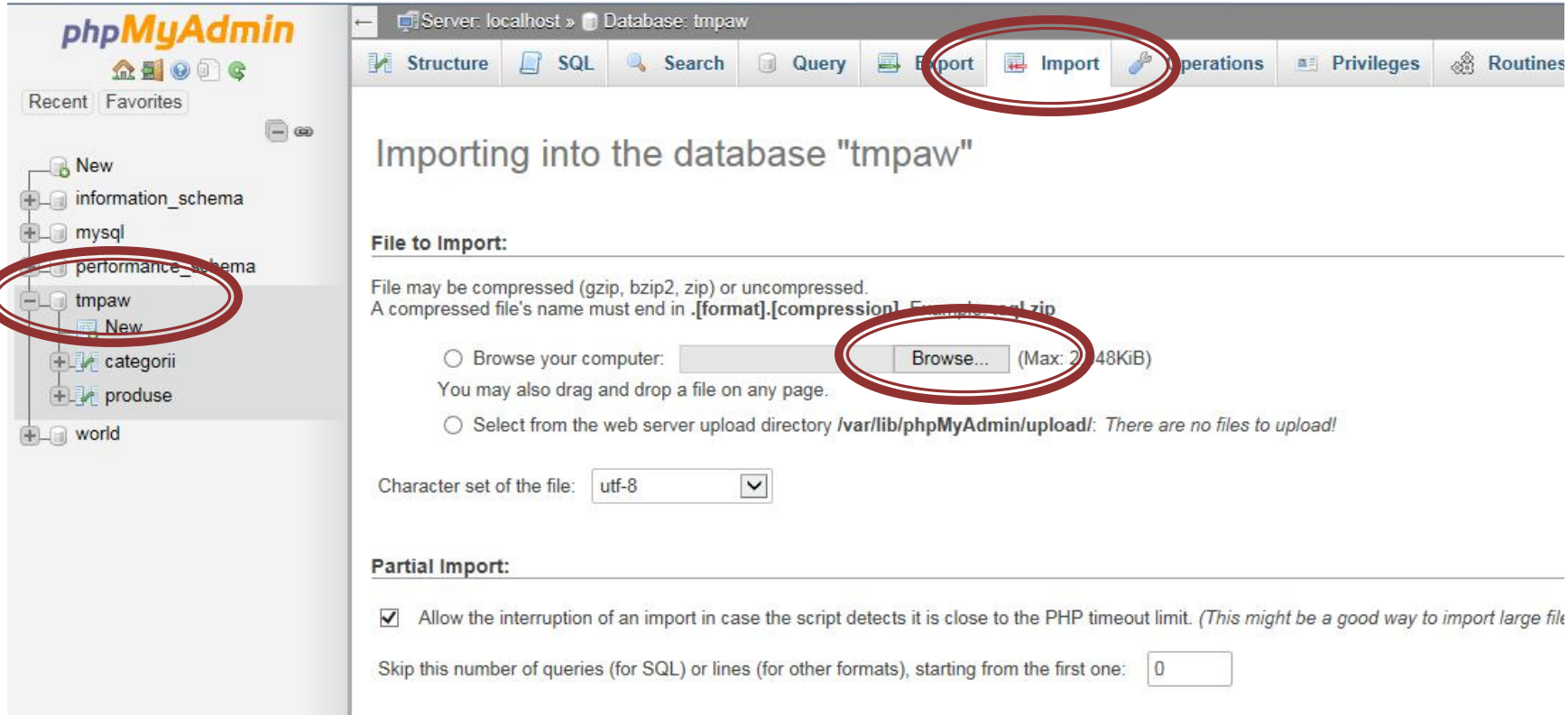
Backup

- Nume (tabel sau baza de date) → Export



Restore

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

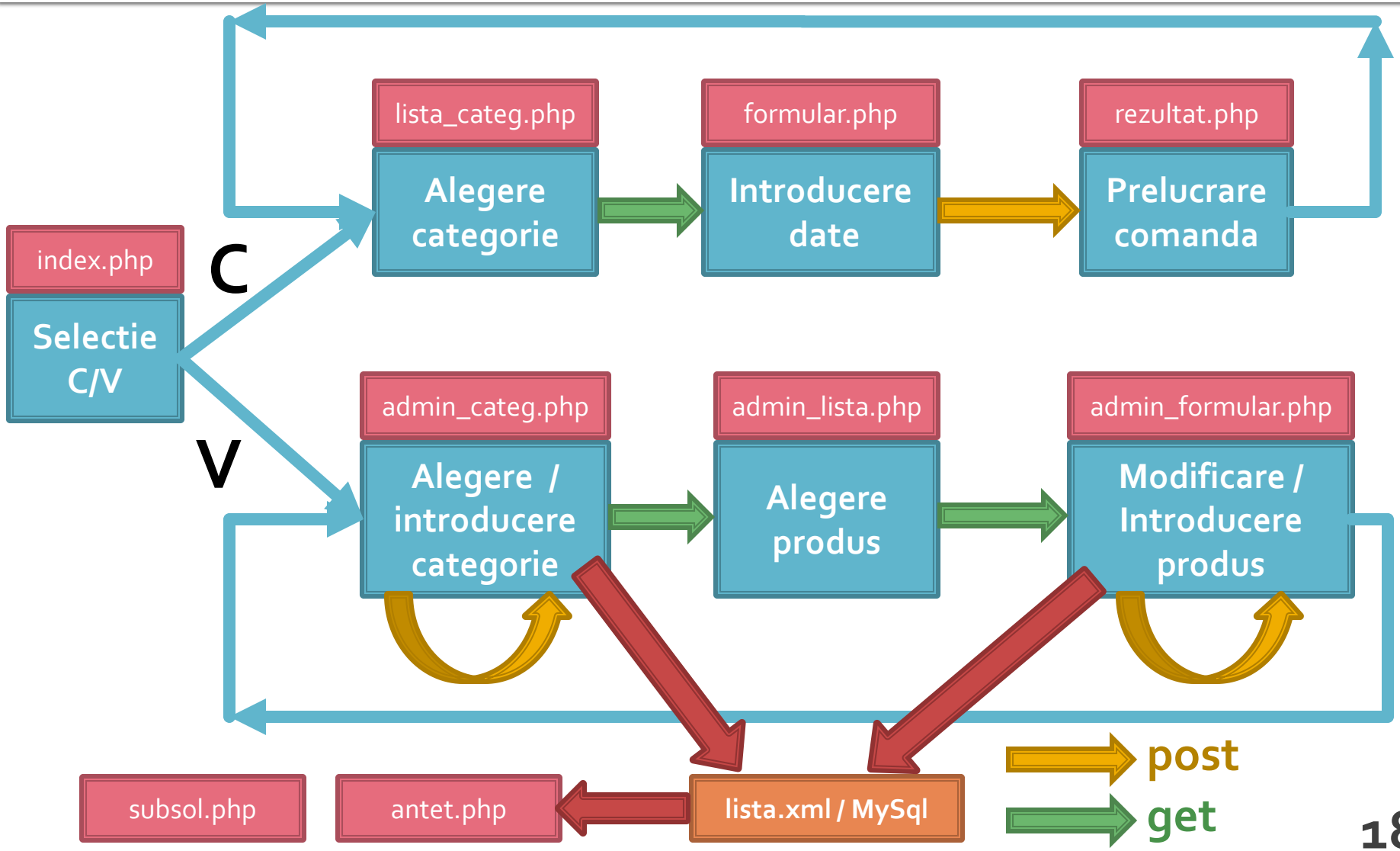


Laborator 6

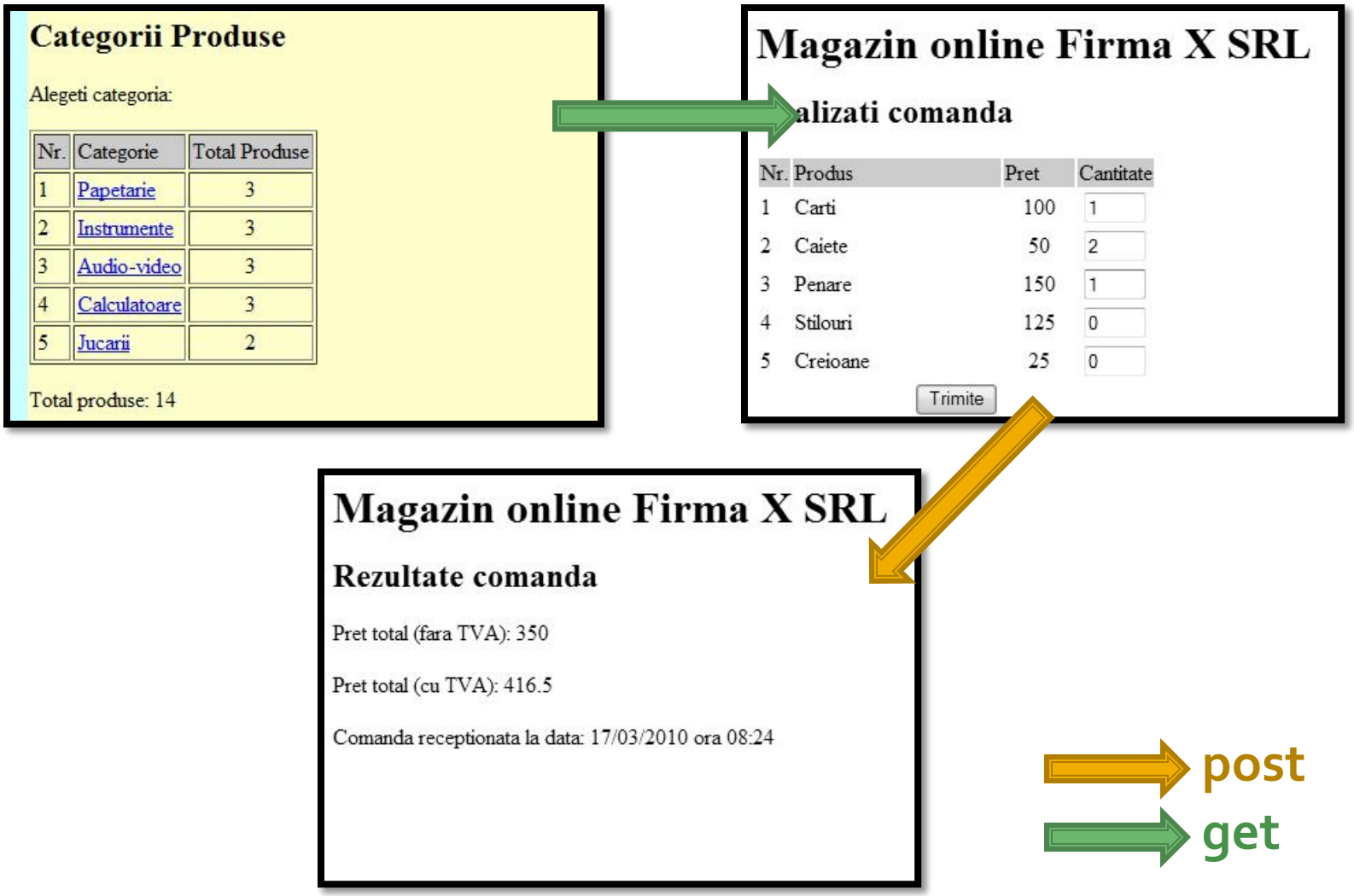
Laborator 6+7

- 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**

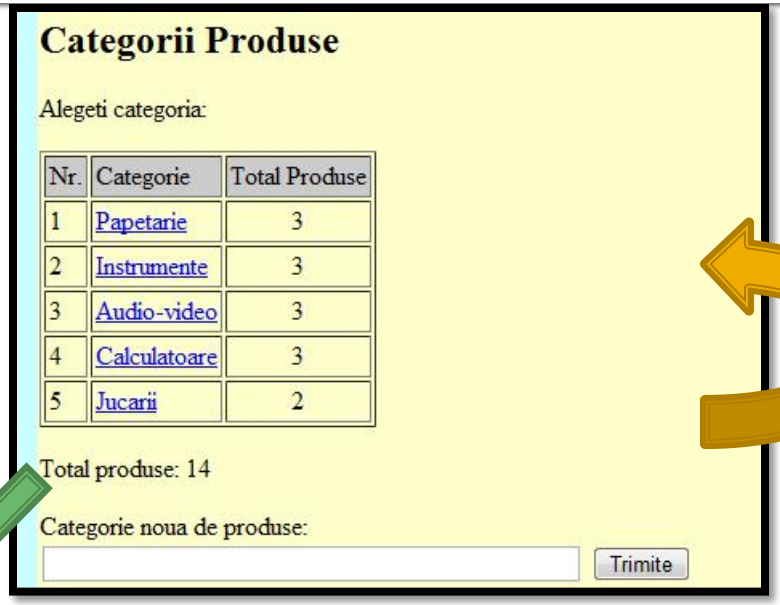
Plan aplicatie

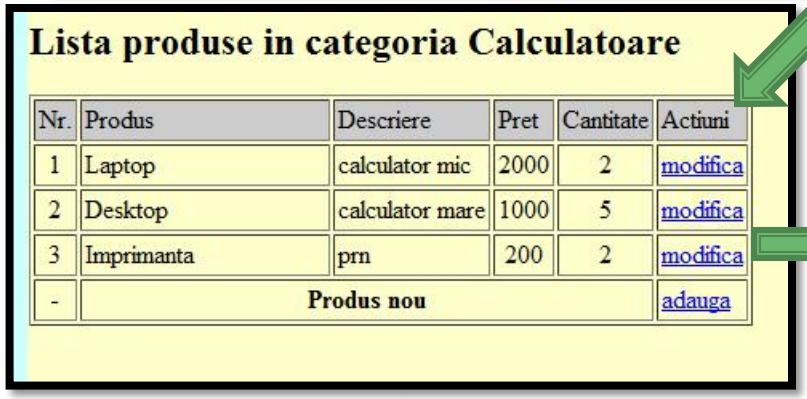


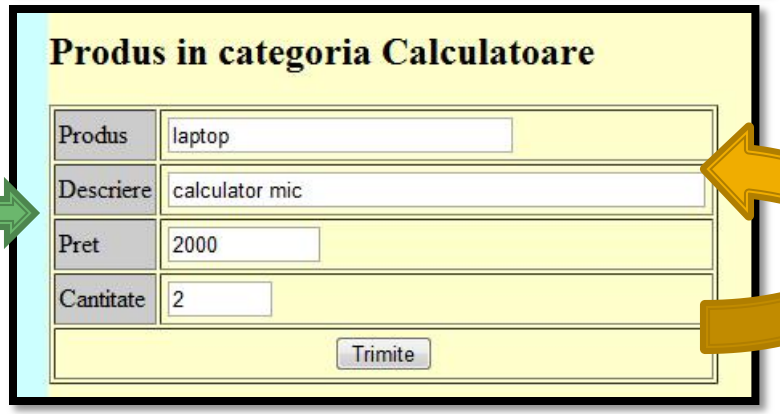
Rezultat (cumparator)



Rezultat (vanzator)



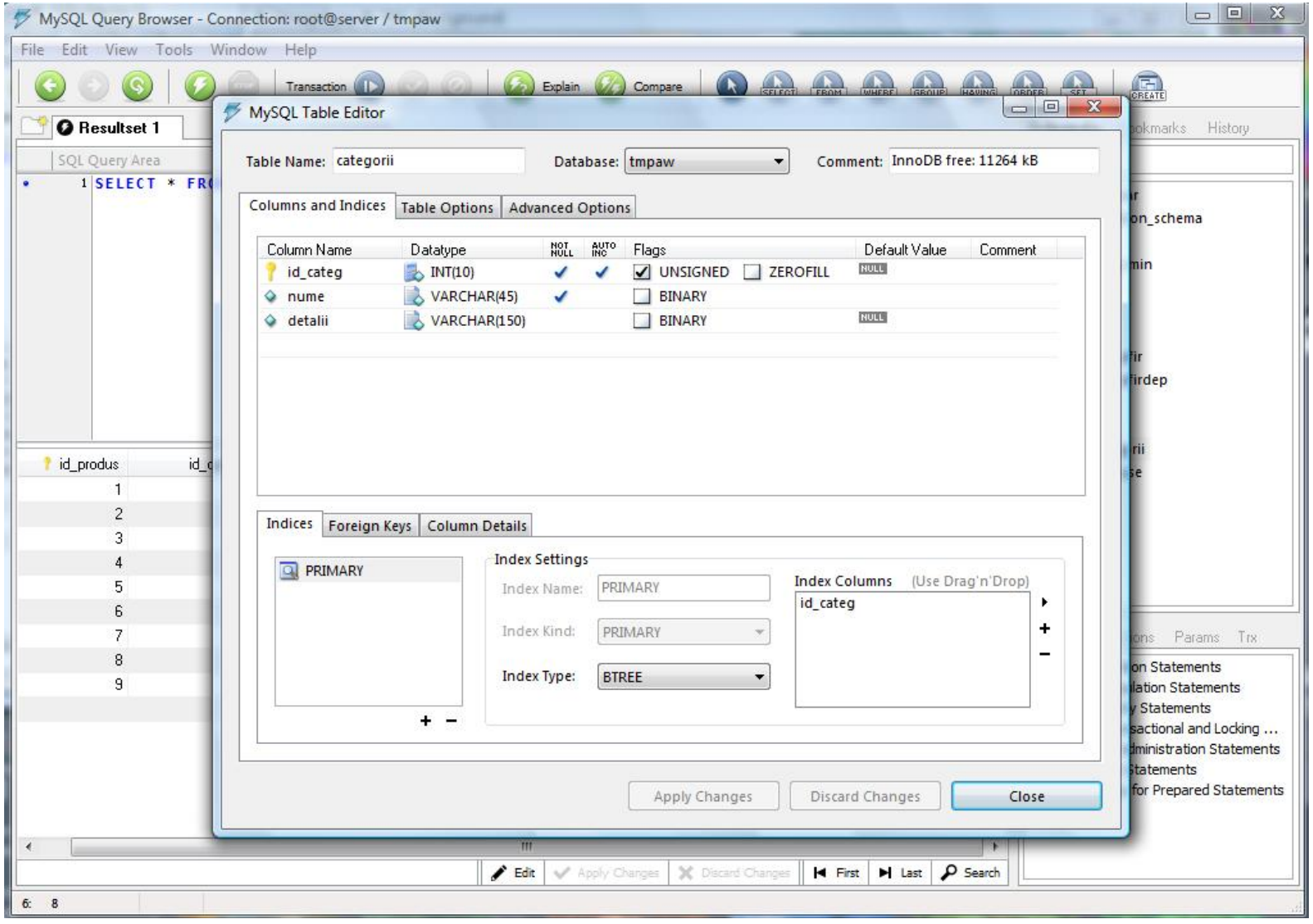




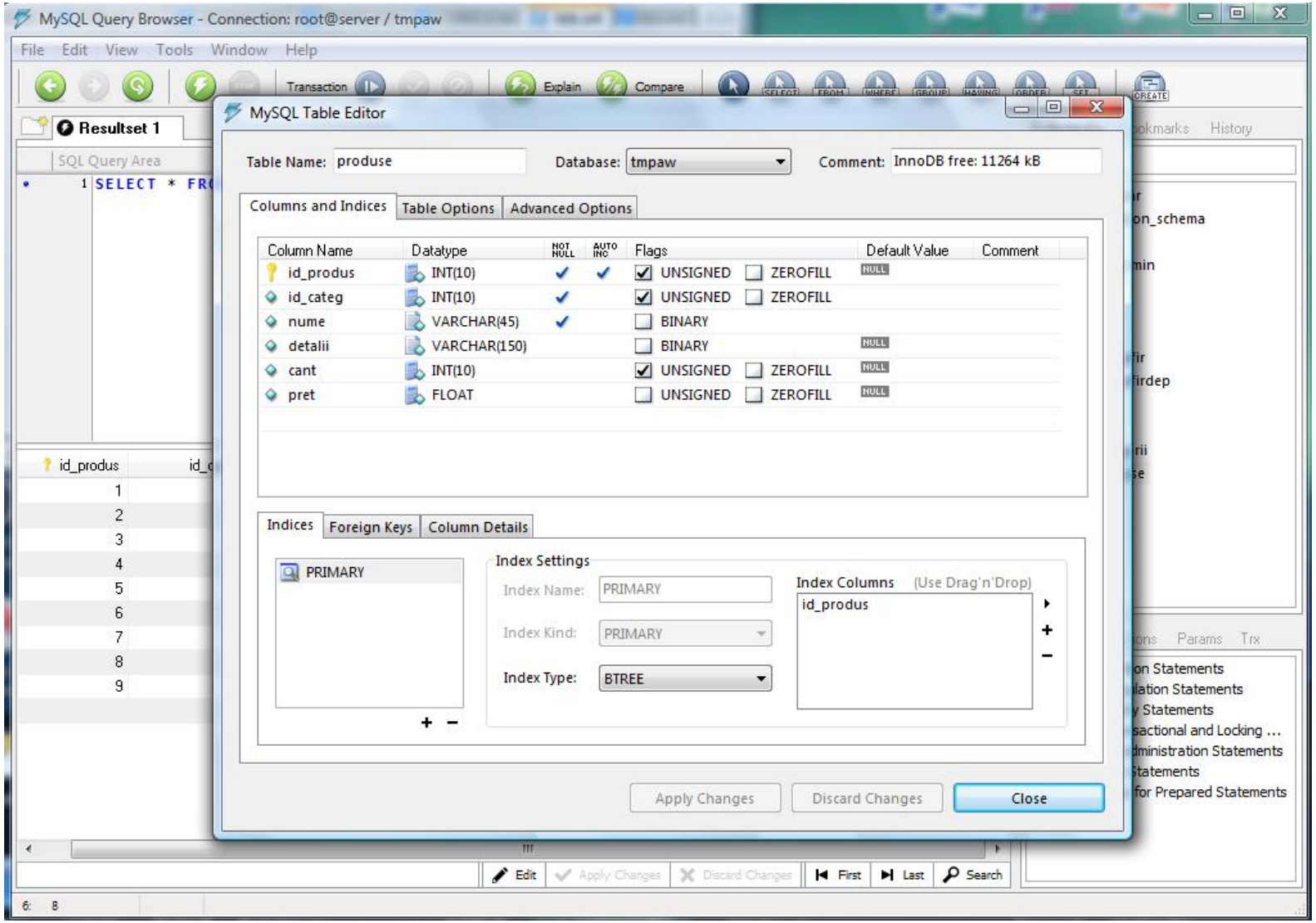
post

get

Tabel Categorii



Tabel Produse



Laborator 6/7 – 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)

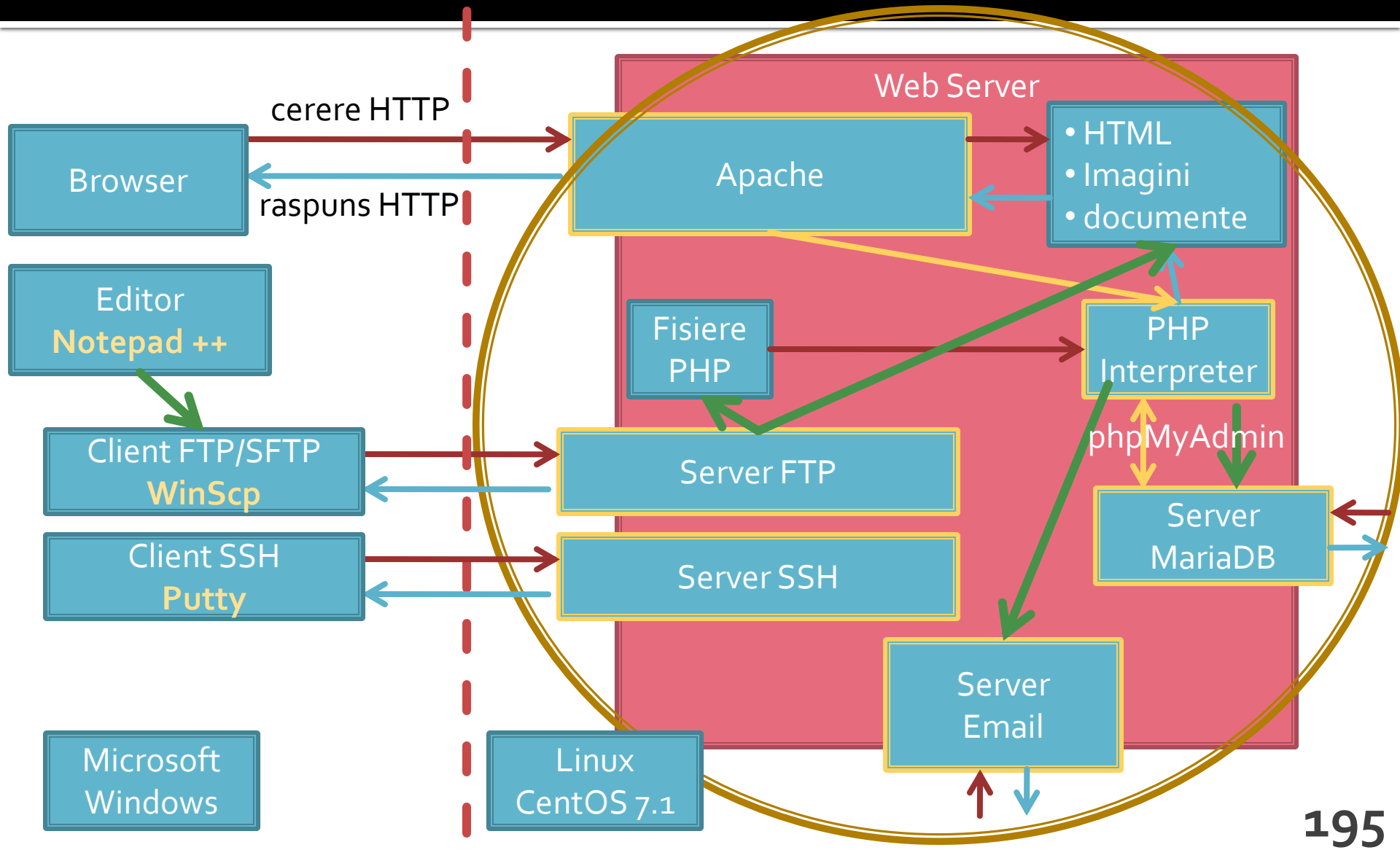
Laborator 6/7 – Mod de lucru

- 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 6/7 – Mod de lucru

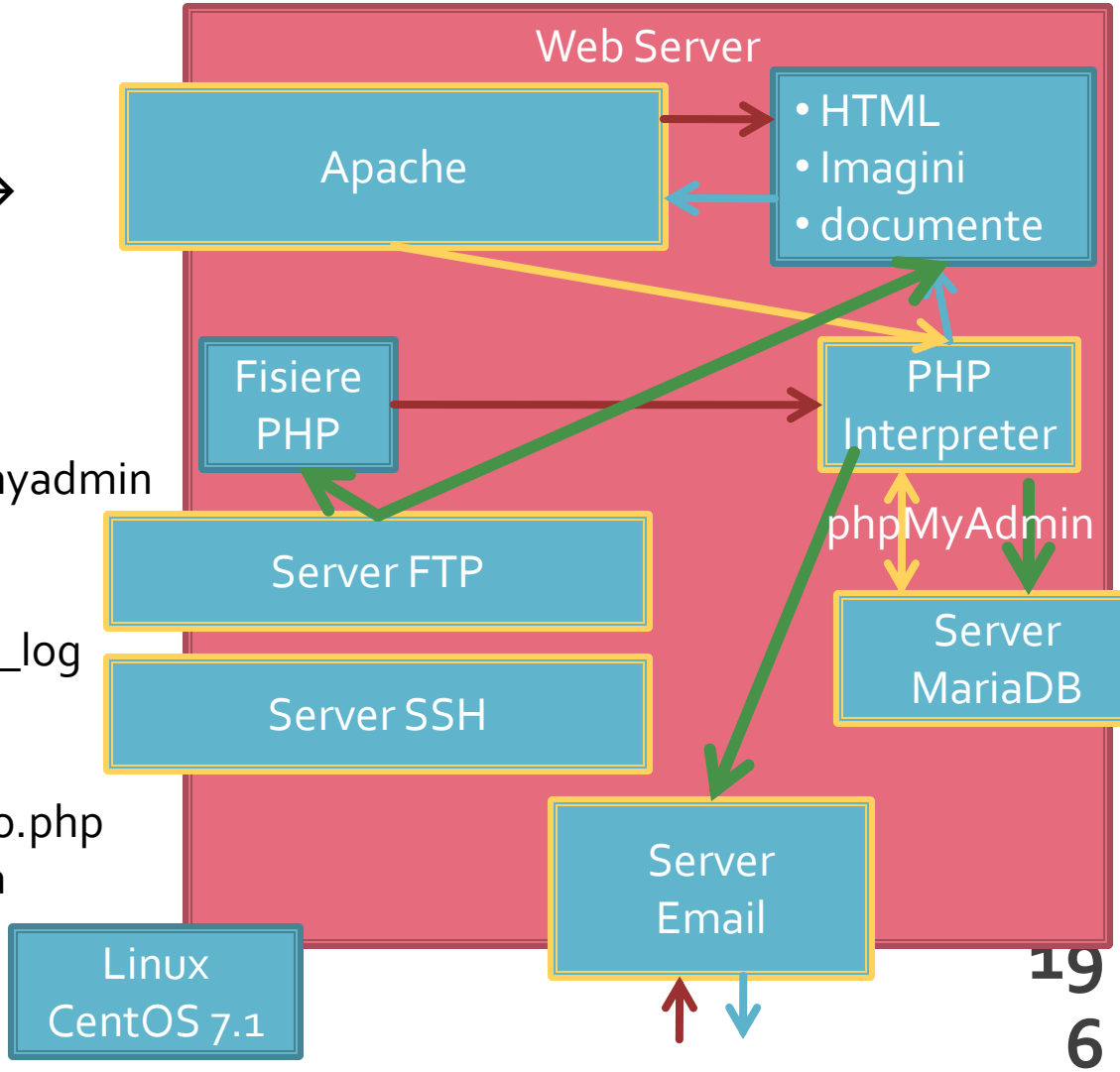
- Se creaza firul de executie paralel pentru vanzator
  - fisierele pentru cumparator reprezinta o buna cale de pornire (Save As, Copy/Paste) pentru 2 din cele 3 fisiere
- Se lucreaza cat mai mult la conversia text -> MySQL
  - activitatea se continua la laboratorul 7

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
  - temp <?php echo "a=";echo $a; ?>

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

Depanare

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

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

MySql – Server Windows 2000

Mini – Indrumar practic
Lucru cu bazele de date

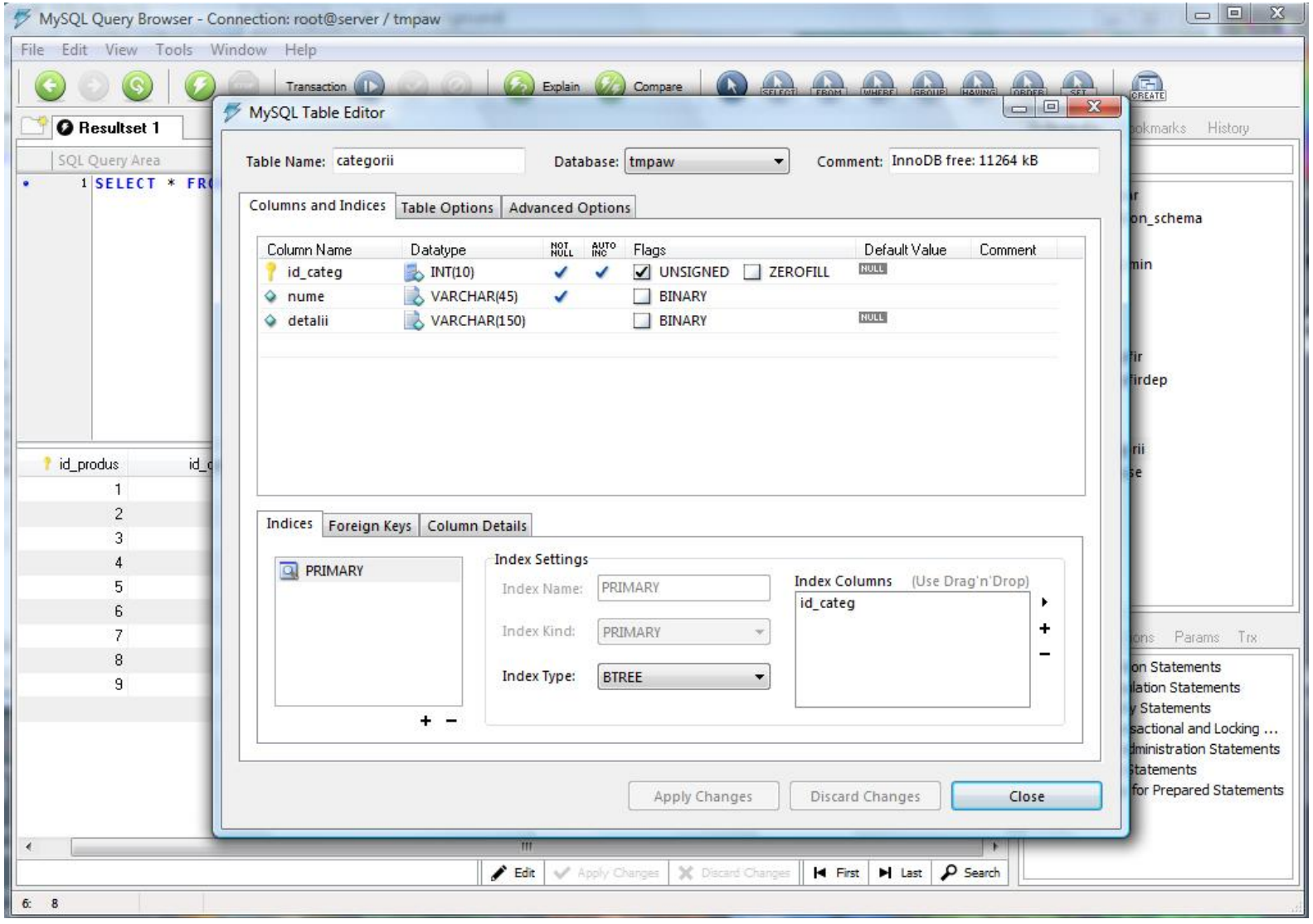
Realizarea bazei de date

- 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

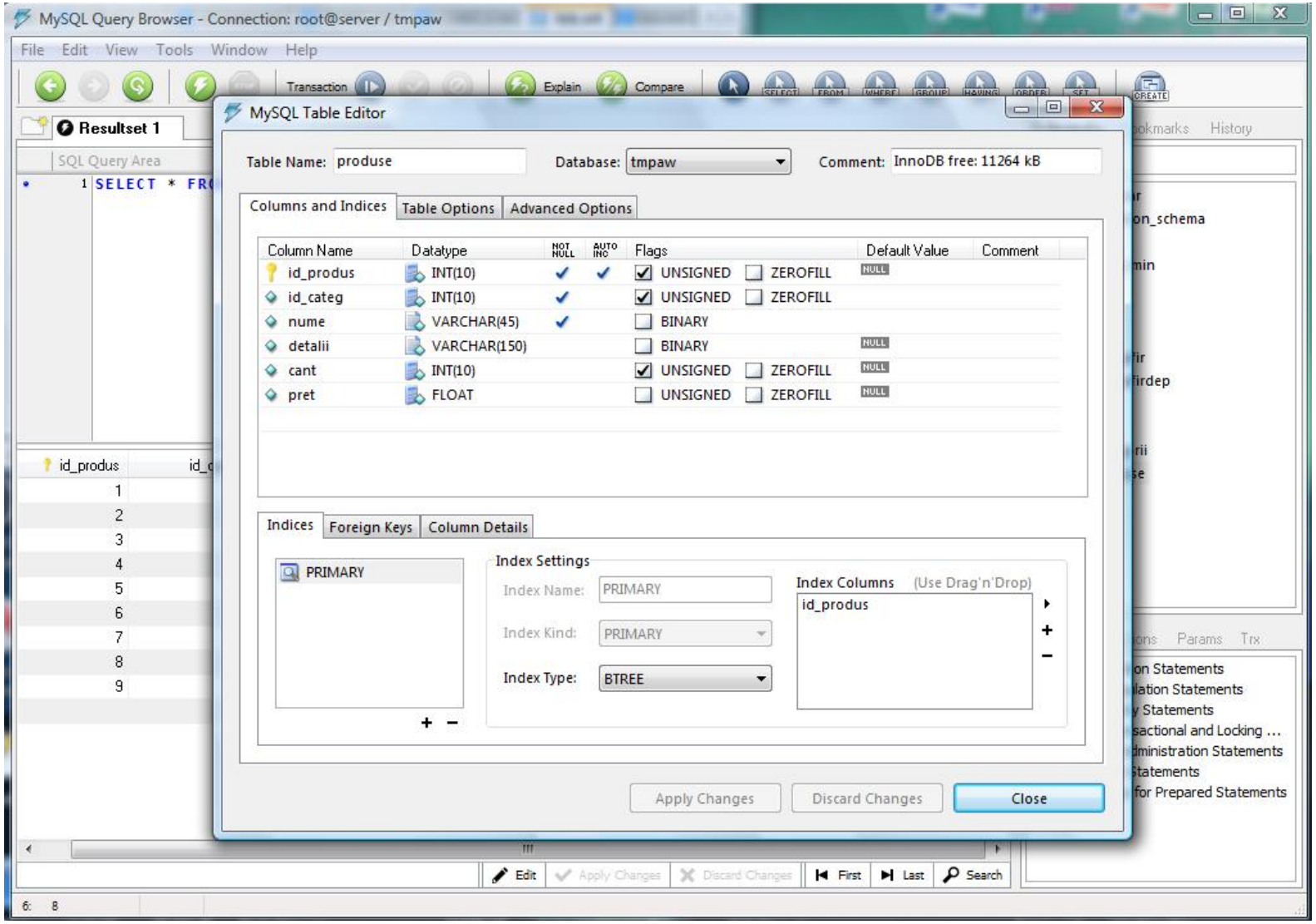
Introducere tabele

- 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



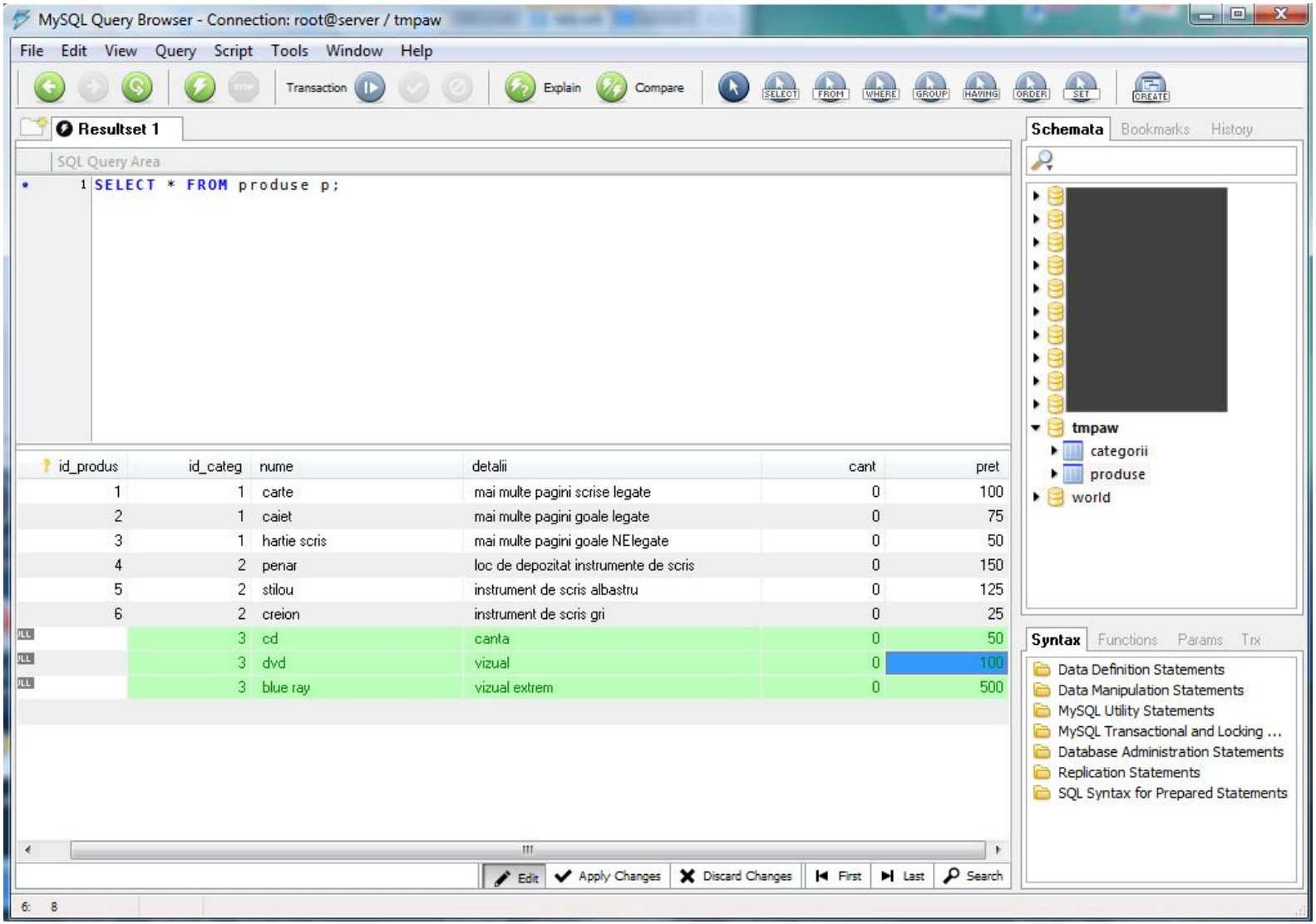
Tabel Produse



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

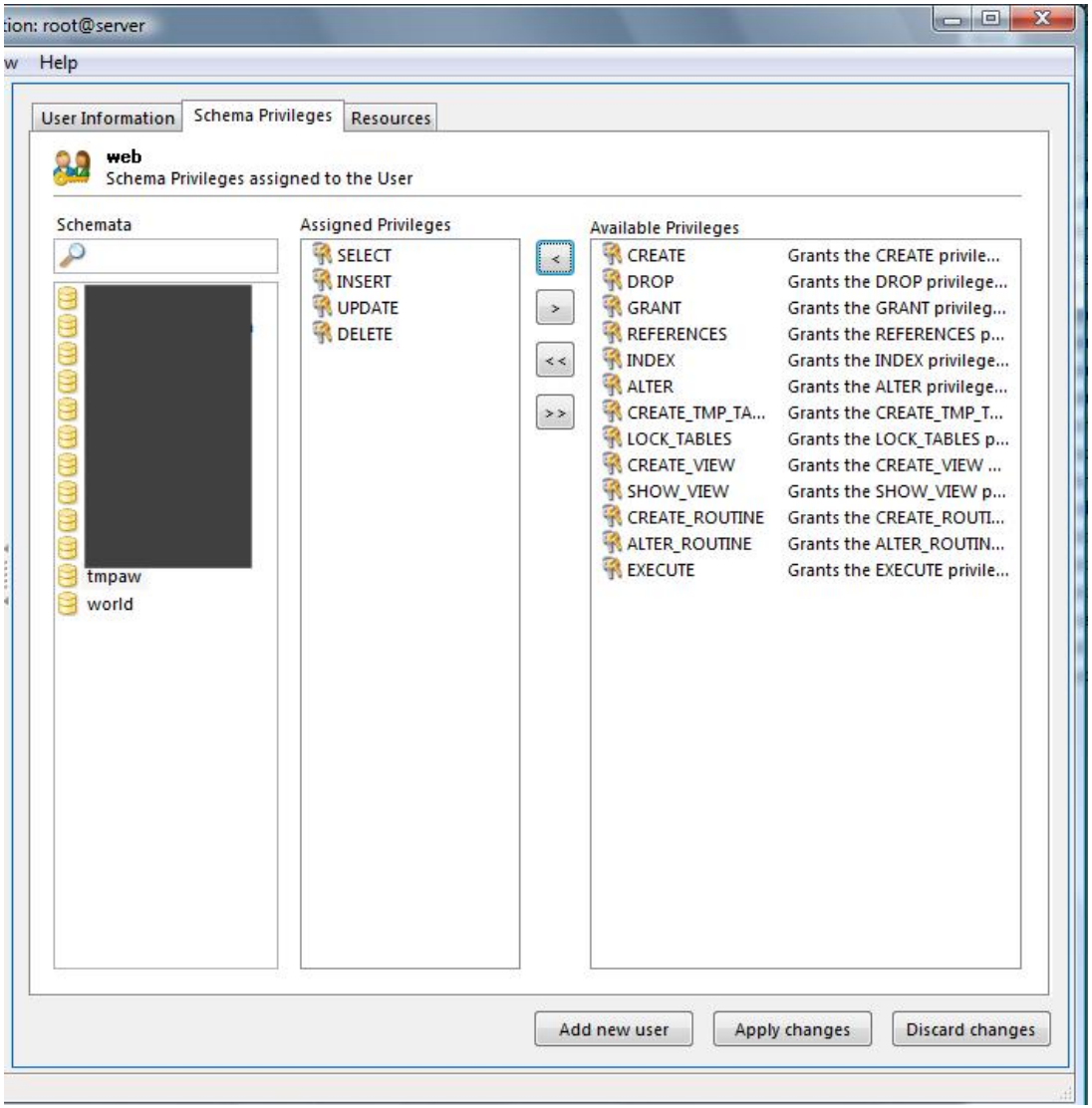
Introducere date initiale



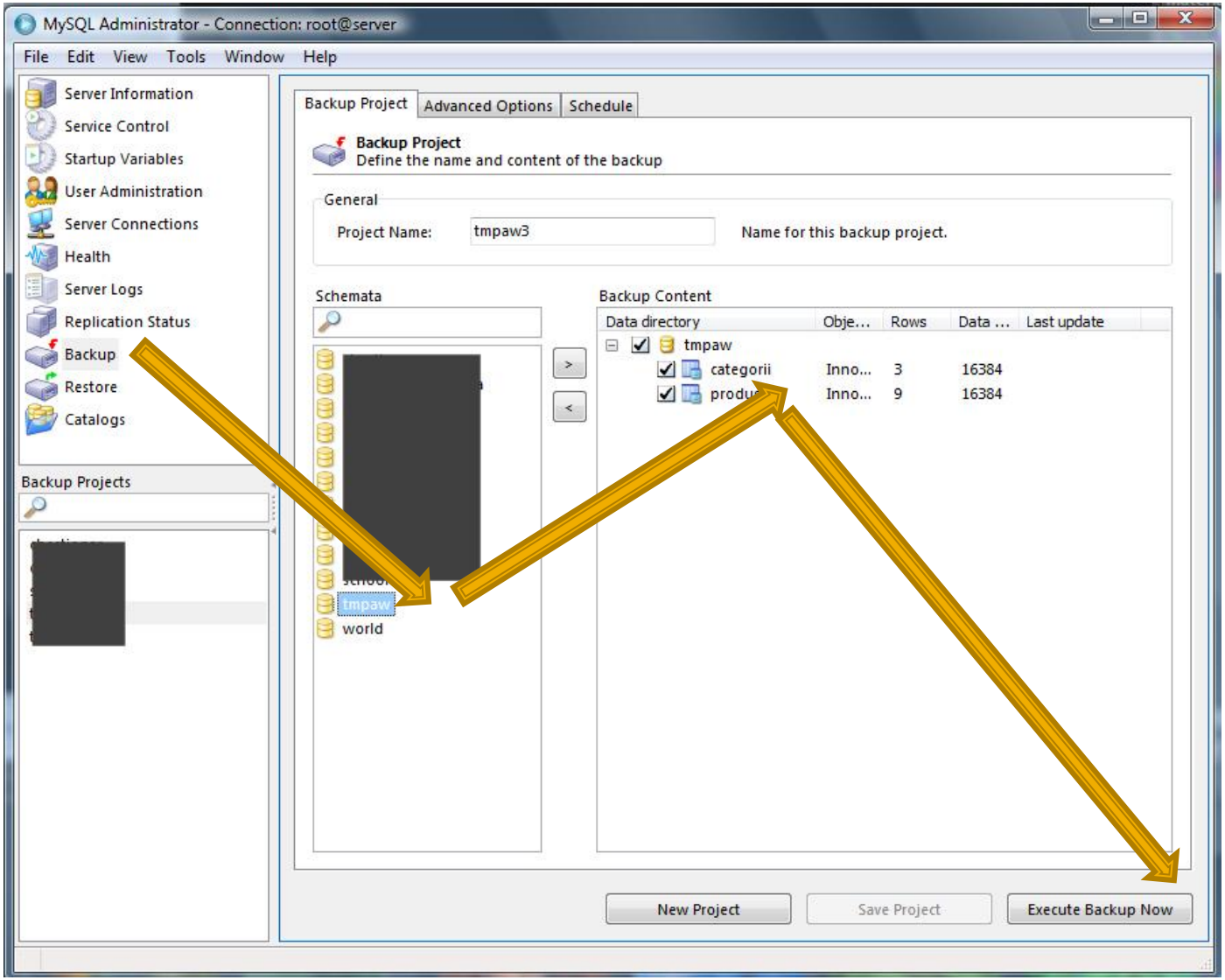
Backup, Restore, drepturi de acces

- 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


MySQL Administrator - Connection: root@server
File Edit View Tools Window Help
Server Information
Service Control
Startup Variables
User Administration
Server Connections
Health
Server Logs
Replication Status
Backup
Restore
Catalogs
Backup Projects
Backup Project
Advanced Options
Schedule
Backup Project
Define the name and content of the backup
General
Project Name:
tmpaw3
Name for this backup project.
Schemata
Backup Content
Data directory
Obje...
Rows
Data ...
Last update
tmpaw
categorii
Inno...
3
16384
Inno...
9
16384
world
New Project
Save Project
Execute Backup Now

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/interogarile SQL necesare pentru crearea bazei de date si popularea ei cu date

Contact

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