# Laborator 3

1. Creați un formular web prin intermediul căruia se transmite la server un șir de caractere. Programul va returna numărul de caractere, șirul inversat, șirul scris cu litere mici, șirul scris cu litere mari. După afișare creați un link pentru a intoarce la formular.

2. Creați un formular de login. La completarea câmpurilor utilizator cu “user” și parola cu “password” utilizatorului i se va afișa un mesaj de întâmpinare. În caz contrar, utilizatorului i se va afisa un mesaj de informare.

3. Creati folosind phpmyadmin sau un script php baza de date userapp.

4. Creati folosind phpmyadmin sau un script php tabelul utilizatori (id, username, parola, sex, starecivila, nume, prenume, email).

5. Creați un formular de înregistrare utilizator nou cu câmpurile user și parolă. (inregistrare.php)

6. Sanitizati variabilele transmise prin GET sau POST folosind functia htmlentities. Afisati variabila înainte de sanitizare și după. Introduceți ca data de test utilizatorul <script> si parola &

7. Introduceti în tabelul utilizatori folosind un script php utilizatorul și parola din formular.

8. Conectati-vă la baza de date userapp și verificați dacă utilizatorul și parola se află pe aceeași linie în tabelul utilizatori (parola e reținută în clar).

9. Introduceți în tabelul utilizatori folosind un script php utilizatorul și parola transmise prin formular (parola se reține criptat).

10. Conectati-vă la baza de date userapp și verificați dacă utilizatorul și parola se afla pe aceeasi linie în tabelul utilizatori (parola e reținută criptat).

11. Creați un formular de login în fisierul login.php cu actiune către login.php care să conțină username, parola și butoanele radio pentru ORDER BY NAME.

12. Verificați dacă userul și parola corespunde unui utilizator existent în baza de date.

13. Să se afișeze în fișierul la care se ajunge după login data curentă în formatul “YYYY-mm-dd” și valoarea selectată la ORDER BY NAME.

14. Creați fișierul index.php care să conțină link către inregistrare.php și login.php

15. Creați și restul câmpurilor în formular și refaceți scriptul să însereze și aceste date în baza de date.

# Comenzi MySQL

| Comanda | Descriere |
|---|---|
| **SHOW** DATABASES; | Arată toate bazele de date existente |
| **USE** nume-baza-de-date; | Alegerea bazei de date cu care vrem să lucrăm |
| **SHOW** TABLES; | Arată tabelele existente în baza de date ales |
| **CREATE** DATABASE nume-baza-de-date; | Creează baza de date cu denumirea ”nume-baza-de-date” |
| **CREATE** TABLE nume-tabel (camp1 TEXT); | Creează tabel cu numele ”nume-tabel” cu un câmp numit ”camp1” de tip text |
| **CREATE** TABLE nume-tabel (camp-a TEXT,camp-b INT); | Creează tabel cu numele ”nume-tabel” cu două câmpuri |
| **CREATE** TABLE nume-tabel ( camp1 VARCHAR(128), id INT UNSIGNED NOT NULL KEY); | Creează tabel cu numele ”nume-tabel” cu un câmp numit ”camp1” de tip varchar și unul ”id” de tip int care este camp de index |
| **DROP** TABLE tabel; | Șterge tabelul cu numele ”tabel” |
| **DROP** DATABASE nume-baza-de-date | Șterge baza de date cu numele ”nume-baza-de-date” |
| **TRUNCATE** TABLE tabel; | Golește tabelul cu numele ”tabel” |
| **INSERT** INTO tabel (camp1, camp2, camp3) VALUES (val1, val2, val3); | Introduce în tabelul cu numele ”tabel”, în camp1 val1, în camp2 val2 și în câmp3 val3 *(nu trebuie neapărat să completăm toate câmpurile, numai dacă au atribut de NOT NULL)* |
| **SELECT** * FROM tabel; | Afișează tot conținutul tabelului |
| **SELECT** camp1 FROM tabel; | Afișează conținutul câmpului ”camp1” |
| **SELECT** camp1, camp2 FROM tabel; | Afișează conținutul câmpului ”camp1” și ”camp2” |
| **SELECT** * FROM tabel WHERE camp1=’valoare’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” are valoarea ”valoare” |
| **SELECT** * FROM tabel WHERE camp1 LIKE ‘valoare’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” conține ”valoare” |
| **SELECT** * FROM tabel WHERE camp1 LIKE ‘valoare%’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” se termină cu ”valoare” |
| **SELECT** * FROM tabel WHERE camp1 LIKE ‘%valoare’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” începe cu ”valoare” |
| **SELECT** * FROM tabel WHERE camp1=’val1’ AND camp2 LIKE ‘val2’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” are valoarea de ”val1” și ”camp2” conține ”val2” |
| **SELECT** * FROM tabel WHERE camp1!=’valoare’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” este diferit de ”valoare” |
| **SELECT** * FROM tabel WHERE camp1 NOT LIKE ‘%valoare’; | Afișează toate înregistrările din tabel unde câmpul ”camp1” nu începe cu ”valoare” |

| Comanda | Descriere |
|---|---|
| **SELECT** * FROM tabel ORDER BY camp2 ASC; | Afișează înregistrările din tabel în ordine crescătoare/ alfabetică a câmpului ”camp2” |
| **SELECT** * FROM tabel ORDER BY camp2 DESC; | Afișează înregistrările din tabel în ordine descrescătoare a câmpului ”camp2” |
| **SELECT** count (*) FROM tabel WHERE camp1=val1; | Afișează câte înregistrări sunt în total în tabel al căror ”camp1” are valoarea ”val1” |
| **SELECT** SUM(camp1) FROM tabel; | Afișează suma elementelor din câmpul ”camp1” din tabelul ”tabel” |
| **SELECT** AVG(camp1) FROM tabel; | Afișează media aritmetică a elementelor din câmpul ”camp1” din tabelul ”tabel” |
| **SELECT** CONCAT(camp1, camp2) FROM tabel; | Afișează valorile coloanelor ”camp1” și ”camp2” concatenate |
| **SELECT** LENGTH(camp1) FROM tabel; | Afișează câte caractere conține fiecare valoare din coloana ”camp1” |
| **SELECT** MAX(camp1) FROM tabel; | Afișează numărul cel mai mare din coloana ”camp1” |
| **SELECT** MIN(camp1) FROM tabel; | Afișează numărul cel mai mic din coloana ”camp1” |
| **SELECT** * FROM tabel LIMIT 6,4; | Afișează începând de la a 6-a linie încă 4 înregistrări |
| **DELETE** FROM tabel WHERE camp1=val1; | Șterge toate înregistrările din tabelul ”tabel” unde câmpul ”camp1” are valoarea de ”val1” |
| **UPDATE** tabel SET camp1=val1 WHERE id=nr; | Înlocuiește în tabelul ”tabel” în înregistrarea care în câmpul ”id” are valoarea ”nr”, valoarea din câmpul ”camp1” cu valoarea ”val1” |
| **ALTER** TABLE tabel ADD camp3 TEXT; | Adaugă la tabelul ”tabel” un câmp nou denumit ”camp3” de tip TEXT |
| **ALTER** TABLE tabel CHANGE camp1 campa INT; | Redenumește câmpul ”camp1” în ”campa” din tabelul ”tabel” și setează tipul INT |
| **ALTER** TABLE tabel MODIFY camp1 VARCHAR(100); | Modifică tipul câmpului ”camp1” în VARCHAR |
| **ALTER** TABLE tabel DROP camp1; | Șterge coloana ”camp1” din tabelul ”tabel” |
| **GRANT** ALL ON baza_de_date.* TO ‘user’ IDENTIFIED BY ‘parola’ | Setează toate drepturile de accesare a bazei de date ”baza_de_date” de către utilizatorul ”user” identificat cu parola ”parola” |
| **SELECT** * FROM tabel GROUP BY camp1 | Afișează înregistrările din tabelul ”tabel1”, grupate după ”camp1” |
| mysql - u username -p | Deschide linia de comandă MySQL |

- Exemplele de sus sunt folosite pentru accesul bazelor de date din linia de comandă (command prompt).
- Alte comenzi: **BACKUP**, **DESCRIBE**, **EXIT** (CTRL+C), **HELP** (\h, \?), **LOCK**, **QUIT**, **RENAME**, **SOURCE**, **STATUS** (\s), **UNLOCK**, **UPDATE**.

# Tipuri de date MySQL

| Tip | Descriere |
|---|---|
| Text | |
| CHAR(lungime) | Un câmp cu lungime fixă de la 0 la 255 de caractere. |
| VARCHAR(lungime) | Un câmp cu lungime variabilă până la 65535 caractere. |
| TINYTEXT(lungime) | Un șir cu lungime maximă de 255 de caractere |
| TEXT(lungime) | Un șir cu o lungime maximă de 65.535 caractere. |
| MEDIUMTEXT(lungime) | Un șir cu o lungime maximă de 16.777.215 caractere |
| LONGTEXT(lungime) | Un șir cu o lungime maximă de 4.294.967.295 caractere |
| Numere | |
| TINYINT(lungime) | Interval de la -128 la 127. Sau de la 0 la 255 unsigned |
| SMALLINT(lungime) | Interval de la -32.768 la 32.767. Sau de la 0 la 65.535 unsigned |
| MEDIUMINT(lungime) | Interval de la -8.388.608 la 8.388.607. Sau de la 0 la 16.777.215 unsigned |
| INT(lungime) | Interval de la -2.147.483.648 la 2.147.483.647. Sau de la 0 la 4.294.967.295 unsigned |
| BIGINT(lungime) | Interval de la -9.223.372.036.854.775.808 la 9.223.372.036.854.775.808. Sau de la 0 la 18.446.744.073.709.551.615 unsigned |
| FLOAT(lungime,decimale) | Număr mic cu zecimale. Se recomandă folosirea FLOAT fără parametri opționali. |
| DOUBLE(lungime,dec.) | Număr mare cu zecimale |
| DECIMAL(lungime,dec.) | Un tip DOUBLE care permite un număr fix de zecimale. |
| Dată și timp | |
| DATE | O dată stocată ca YYYY-MM-DD. interval de la 1000-01-01 la 9999-12-31 |
| DATETIME | Dată și oră, afișat în format YYYY-MM-DD HH:MM:SS |
| TIMESTAMP | Un timestamp generat de obicei automat. Date între ianuarie 1970 până la ianuarie 2038, în format ca DATETIME |
| TIME | Ora în format HH:MM:SS |
| Date binare | |
| TINYBLOB(lungime) | Până la 255 bytes |
| BLOB(lungime) | Până la 64kB |
| MEDIUMBLOB(lungime) | Până la 16MB |
| LONGBLOB(lungime) | Până la 4GB |
| BINARY(lungime) | Până la 255 bytes |
| BYTE(lungime) | Până la 255 bytes |
| VARBINARY(lungime) | Până la 64kB |
| Liste predefinite | |
| ENUM | O singură alegere dintr-o listă predefinită |
| SET | Zero, una sau mai multe înregistrări dintr-o listă predefinită, până la 64 de înregistrări. |

## MySQL caracteristici ale coloanelor

| Caracteristici | Descriere |
|---|---|
| UNSIGNED | Acceptă doar valori pozitive |
| NOT NULL | Specifică faptul că o coloană nu poate avea valoarea nula |
| AUTO_INCREMENT | Se incrementează automat (nu este recomandat) |
| INDEX | Pentru a crea indexarea unei coloane |
| PRIMARY KEY | Identifica fiecare rând al tabelei. Se poate crea o singură cheie primară pentru fiecare tabel |
| UNIQUE | Specifică o coloană sau o combinație de coloane a cărei valori trebuie să fie unice pentru toate rândurile din tabel |
| DEFAULT CURRENT TIMESTAMP | La înserare se completează cu data și ora actuală. Coloana trebuie să fie de tip timestamp |
| on update CURRENT_ TIMESTAMP | La modificare înregistrare se modifică automat la data și ora actuală |
| DEFAULT NULL | Valoarea implicită este NULL |
| CHARACTER SET utf8 | Setează seturi de caractere folosite. În loc de utf8 se pot folosi alte seturi de caractere. |
| COLLATE utf8_romanian _ci | Specifică codificarea/limba. Acesta se foloseste în comparații între string-uri și la ordonare |
| COMMENT 'comentariu' | Adaugă un comentariu la coloană. |

## MySQL indexarea tabelelor

| Comanda | Descriere |
|---|---|
| ALTER TABLE tabel ADD INDEX (camp1(20)); | Creează un index în tabelul "tabel" pentru câmpul "camp1", având lungime de 20 |
| CREATE INDEX camp1 ON tabel (camp1(20)); | Creează un index în tabelul "tabel" pentru câmpul "camp1", având lungime de 20 |

## Cursor MySQL

| Caracteristici | Descriere |
|---|---|
| mysql> | Stare de așteptare comandă |
| -> | Stare de așteptare pentru următoarea linie din comandă |
| '> | Stare de așteptare pentru a continua completarea șirului de caractere care a început cu ' |
| "> | Stare de așteptare pentru a continua completarea șirului de caractere care a început cu " |
| /*> | Stare de așteptare pentru a continua completarea șirului de caractere care a început cu /* |

- Fiecare comandă se termină cu ";".
- Anularea unei comenzi se face cu \c+Enter.

# Accesarea MySQL din PHP

connection.php

```
<?php
$servername=”localhost”;
$username=”username”;
$password=”password”;
$dbname=”myDB”;
?>
```

Conectare la o bază de date MySQLi - procedural

```
<?php
require_once ‘connection.php’;
//creare conexiune
$conn = mysqli_connect($servername,$username,$password,$dbname);
//verificare conexiune
if (!$conn) die (”Connection failed: “ . mysqli_connect_error());
echo “Connected successfully”;
?>
```

Conectare la o bază de date MySQLi - object oriented

```
<?php
require_once ‘connection.php’;
//creare conexiune
$conn = new mysqli($servername,$username,$password,$dbname);
//verificare conexiune
if (!$conn) die (”Connection failed: “ . $conn->connect_error);
echo “Connected successfully”;
?>
```

Query la o bază de date MySQLi - procedural

```
<?php
$sql = “SELECT id, nume FROM Guests”;
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result)>0) {
  while($row = mysqli_fetch_assoc($result))
    {
    echo “id: “ . $row[”id”];
    echo “Nume: “ . $row[”nume”] . “</br>”;
    }}
else echo “0 results”;
?>
```

Query la o bază de date MySQLi - object oriented

```
<?php
$sql = “SELECT id, nume FROM Guests”;
$result = $conn->query($sql);
if ($result->num_rows > 0) {
  while($row = $result->fetch_assoc())
    {
    echo “id: “ . $row[”id”];
    echo “Nume: “ . $row[”nume”] . “</br>”;
    }}
else echo “0 results”;
?>
```

**Înserare în baza de date MySQLi - procedural**

```
<?php $sql = “INSERT INTO Guests (nume, email) VALUES (’John’, ‘john@example.com’)”;
if (mysqli_query($conn,$sql)) {
  echo “New record created successfully”;
} else echo “Error: “ . $sql . “</br>” . mysqli_error($conn); ?>
```

**Înserare în baza de date MySQLi - object oriented**

```
<?php $sql = “INSERT INTO Guests (nume, email) VALUES (’John’, ‘john@example.com’)”;
if ($conn->query($sql)===TRUE) {
  echo “New record created successfully”;
} else echo “Error: “ . $sql . “</br>” . $conn->error; ?>
```

**Ștergere date din baza de date MySQLi - procedural**

```
<?php $sql = “DELETE FROM Guests WHERE id=3”;
if (mysqli_query($conn,$sql)) {
  echo “Record deleted successfully”;
} else echo “Error deleting record: “ . mysqli_error($conn); ?>
```

**Ștergere date din baza de date MySQLi - object oriented**

```
<?php $sql = “DELETE FROM Guests WHERE id=3”;
if ($conn->query($sql)===TRUE) {
  echo “Record deleted successfully”;
} else echo “Error deleting record: “ . $conn->error; ?>
```

**Actualizare date în baza de date MySQLi - procedural**

```
<?php $sql = “UPDATE Guests SET nume=’Steve’ WHERE id=3”;
if (mysqli_query($conn,$sql)) {
  echo “Record updated successfully”;
} else echo “Error updating record: “ . mysqli_error($conn); ?>
```

**Actualizare date în baza de date MySQLi - object oriented**

```
<?php $sql = “UPDATE Guests SET nume=’Steve’ WHERE id=3”;
if ($conn->query($sql)===TRUE) {
  echo “Record updated successfully”;
} else echo “Error updating record: “ . $conn->error; ?>
```

**Deconectare bază de date MySQLi - procedural**

```
<?php
mysqli_close($conn);
?>
```

**Deconectare bază de date MySQLi - object oriented**

```
<?php
$conn->close();
?>
```