Proiect Baze de Date

(Baza de date pentru scoala)

* Student: Maxim Claudiu Stefan
* Grupa: 1309B
* Coordonator: Mironeanu Catalin
* Git: <https://github.com/maximclaudiu/Baze_Date>

Aplicatia are ca scop stocarea si organizarea datelor unei

scoli/liceu. Aceasta pastreaza date atat despre angajati, cat si despre elevii respectivei institutiii.

Prin parcurgerea tabelelor, putem afla date despre angajati, cum

ar fi orasul din care provin, titlul locului de munca pe care il ocupa, varsta si multe altele. Despre elevi se pot afla varsta, clasa din care fac parte, orasul de provenienta, dar si notele pe care acestia le primesc la diverse discipline.

Proiectul prezinta o fundatie al unui concept ce poate fi extins

pentru stocarea si manipularea mai multor informatii referitoare la institutia dorita (cartile din biblioteca, echipamentele tehnice, mai multe informatii despre angajati si elevi etc.).

Aplicatia poate fi accesata oricand urmarind link-ul :[Proiect BD](https://proiectbd.herokuapp.com/)

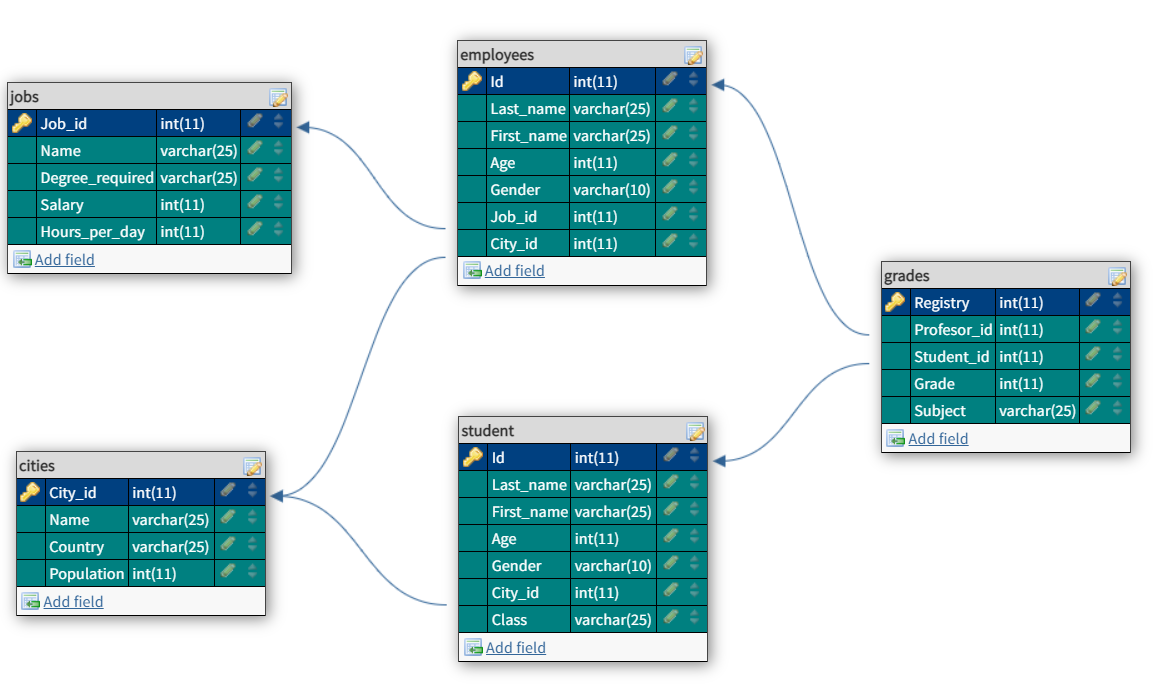
Note: aplicatia este hostata remote pe un server, aceasta se opreste din

rulare daca nu este folosita, drept urmare la prima deschidere, incarcarea dureaza pana intr-un minut. Din aceleasi motive, indexarile autoincrementate in tabel cresc din 10 in 10.

Tehnologii folosite:

1. Pentru interfata:
   1. Html:
      1. Crearea navigatiei si schimbarea paginilor.
      2. Implementarea formularelor de adaugare in tabele.
      3. Afisarea datelor pe ecran sub forma de tabele.
      4. Prezentarea optiunilor de manipulare a datelor intr-un mod cat mai simplist si usor de inteles.
   2. CSS:
      1. Stilizarea paginilor web create.
      2. Confera interfetei un aspect placut si usor de navigat.
   3. JavaScript:
      1. Usureaza munca vizitatorului de a accesa/adauga/sterge informatii in tabelele dorite.
2. Pentru baza de date:
   1. MySql:
      1. Crearea bazei de date si a tabelelor acesteia.
      2. Executare de cod SQL pentru operatiunile de vizualizare, adaugare si stergere.
   2. PHP:
      1. Crearea unei conexiuni la baza de date folosita.
      2. Legatura intre butoanele din interfata si comenzile SQL rulate in fundal.

Structura si inter-relationarea tabelelor:



1. Prima forma normala:
   1. Nu contine mai multe valori pe un camp(ex: “Nume: Popescu, Vlad” -> “Nume: Popescu; Prenume: Vlad”).
   2. Datele de pe o coloana sunt de acelasi tip (ex: int, data, varchar).
   3. Coloanele au nume unice.
2. A doua forma normala:
   1. Se afla in prima forma normala.
   2. Elementele tabelelor sunt dependente doar de cheia primara.

Constrangeri:

* Coloanele Job\_id, City\_id, Id (employees), Id (student) si Registry

sunt primary key-uri in tabelele lor.

* City\_id este foreign key, atat in tabela employees, cat si in tabela

student (one-to-many).

* Job\_id este foreign key in tabela employees (one-to-many).
* Id (employees) si Id (student) sunt foreign key-uri in tabela grades (Profesor\_id, Student\_id) (many-to-many).

Conectarea la baza de date:

Conectarea a fost realizata prin intermediul unei functii php.

Ex: $mysqli= new mysqli ($host, $username, $user\_pass, $database);

Conexiunea este stocata in variabila $mysqli, prin intermediul careia se vor executa comenzile SQL la alegerea utilizatorului.

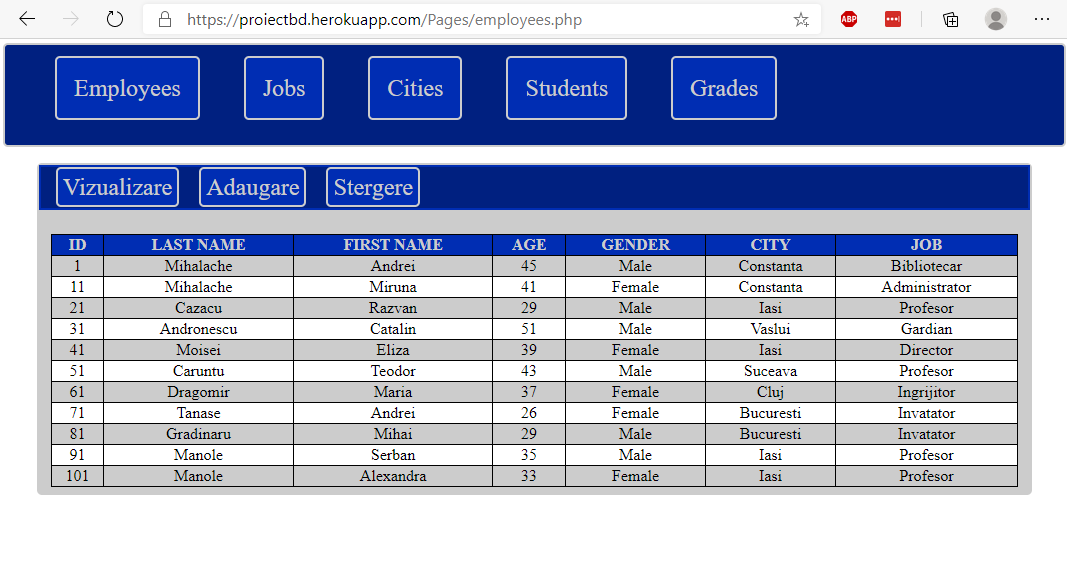


Fig 1. Vizualizarea datelor din tabela.

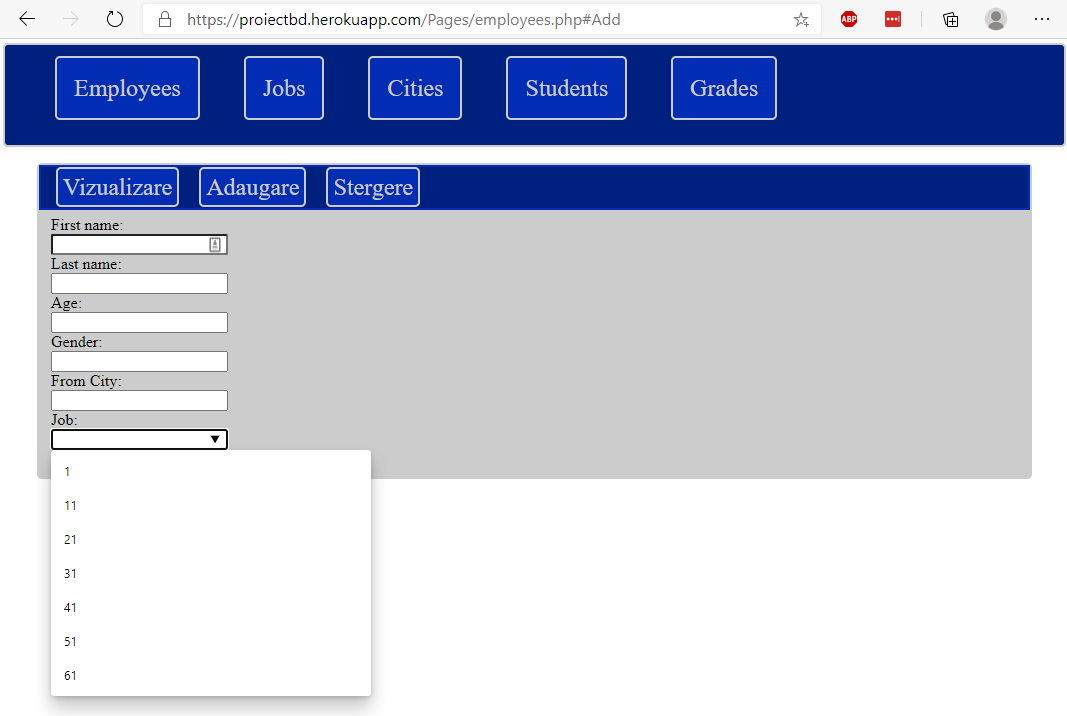


Fig 2. Adaugarea unei inregistrari in tabela si demonstrarea selectiei unei chei straine.

Exemple cod vizualizare:

1. $sql = "SELECT \* FROM ". basename($\_SERVER["REQUEST\_URI"], ".php");
2. $result= $mysqli->query($sql);
3. if ($result->num\_rows>0){
4. while ($row = $result->fetch\_assoc()) {
5. echo " <tr> <td>" . $row["Id"]. "</td>

<td>" . $row["Last\_name"]. "</td>

<td>" . $row["First\_name"]. "</td>

<td>" . $row["Age"]. "</td>

<td>" . $row["Gender"]. "</td>

<td>" . $row2["Name"]. "</td>

<td>" . $row3["Name"]. "</td> </tr>";

}}

* 1. 1. crearea comenzii sql pentru selectarea datelor din tabela. Paginile web au aceeasi denumire cu tabelele din baza, basename($\_SERVER["REQUEST\_URI"], ".php") returneaza acest nume fara ‘.php’ de la final, pentru a obtine o comanda sql valida
  2. 2. rularea scriptului prin intermediul conexiunii php.
  3. 3. daca numarul de rezultate obtinute este mai mare de 0 (zero).
  4. 4. variabila $row pastreaza, sub forma unui array, rezultatele de pe o linie din tabela. (ex: echo $row["Id"];)
  5. 5. returnam un string ce va fi interpretat de html ca fiind o noua linie in tabela de afisare, unde “$row[“X”]” este valoarea aflata in tabela pe coloana “X”, la linia curenta (din 3 si 4:cat timp mai exista date in result, mai bine spus, nu au fost afisate toate liniile din tabela, $row primeste alt array de valori specific unei anumite linii din tabela).

Exemplu cod stergere:

1. $sql = "DELETE FROM ". $\_POST['page']." WHERE Id = '".$\_POST['delete'] ."'";
2. $result= $mysqli->query($sql);
   1. 1. Crearea comenzii de stergere a unei inregistrari din tabela, specifica id-ului primit prin “$\_POST”.
   2. 2. executarea comenzii sql definita precedent prin intermediul conexiunii php.