FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE

DEPARTAMENTUL CALCULATOARE

PROIECT

La Disciplina

INTRODUCERE ÎN BAZE DE DATE

TITLUL LUCRĂRII:

??Platformă de studiu

??PROF. COORDONATOR: COSMINA IVAN

AUTORI:

Păucean Ioana

Cordea Maria  
Apetrei Diana-Andreea

An academic 2024-2025

*grupa 30221*

Cuprins

[1. Introducere 2](#_Toc187581634)

[1.1. Descrierea Generală a Proiectului 2](#_Toc187581635)

[1.2. Scopul și Obiectivele 2](#_Toc187581636)

[2. Tema Proiectului 2](#_Toc187581637)

[2.1. Prezentarea Aplicației 2](#_Toc187581638)

[2.2. Funcționalități și Utilizatori 2](#_Toc187581639)

[3. Descrierea Bazei de Date 3](#_Toc187581640)

[3.1. Structura generala 3](#_Toc187581641)

[3.2. Detalii specifice 5](#_Toc187581642)

[3.3. Nivelul de Normalizare 5](#_Toc187581643)

[4. Elemente de programare a functionalitatilor 6](#_Toc187581644)

[4.1. Proceduri și Triggere.Vederi.Lista completa si exemple 6](#_Toc187581645)

[4.2. Alte exemple de Cod SQL 7](#_Toc187581646)

[5. Interfața Grafică a Utilizatorului (GUI) 7](#_Toc187581647)

[5.1. Descrierea functionalitatilor per tip de utilizator 7](#_Toc187581648)

[5.2. Detalii de implementare funcționalități specifice 7](#_Toc187581649)

[6. Manual de Utilizare 8](#_Toc187581650)

[6.1. Înregistrare și Autentificare 8](#_Toc187581651)

[6.2. Gestionarea Informațiilor per utilizatori 8](#_Toc187581652)

[7. Concluzii și dezvoltari ulterioare 8](#_Toc187581653)

[7.1. Analiza funcționalităților curente- cele mai interesante solutii adoptate 8](#_Toc187581654)

[7.2. Dezvoltari ulterioare 8](#_Toc187581655)

# Introducere

## Descrierea Generală a Proiectului

Proiectul propus vizează dezvoltarea unei aplicații dedicate gestiunii unei platforme de studiu.Această aplicație va integra o interfață grafică intuitivă, care va permite utilizatorilor să acceseze rapid și eficient informațiile necesare pentru desfășurarea activităților academice. Structura aplicației va include module pentru gestionarea datelor referitoare la studenți, profesori, cursuri și activități didactice, asigurând astfel o organizare sistematică a informațiilor.

## Scopul și Obiectivele

Scopul principal al acestui proiect este de a simplifica și eficientiza gestiunea unei platforme de studiu, care să răspundă nevoilor variate ale utilizatorilor facilitând interacțiunii dintre studenți, profesori și administrația universitară. Obiectivele specifice includ dezvoltarea unei baze de date relaționale, care să permită stocarea și gestionarea eficientă a informațiilor despre studenți, profesori,cursuri și alte informații relevante. Aplicația va oferi funcționalități de autentificare, gestionare a activităților didactice și interacțiune între studenți și profesori, contribuind astfel la o experiență educațională mai bine organizată.

# Tema Proiectului

## Prezentarea Aplicației

Aplicația propusă va utiliza un sistem de gestionare a bazelor de date MySQL, care va permite stocarea și manipularea eficientă a datelor. Interfața grafică va fi concepută pentru a oferi o experiență plăcută utilizatorilor, facilitând accesul rapid la informațiile necesare. Utilizatorii vor putea accesa aplicația printr-un proces de autentificare, care va asigura securitatea datelor. Acest proces de autentificare se va realiza prin intermediul e-mail-ului și al parolei.

## Funcționalități și Utilizatori

Se va implementa un sistem de control al accesului bazat pe roluri, care va asigura că fiecare utilizator are acces doar la informațiile și funcționalitățile corespunzătoare rolului său.

Un utilizator poate avea următoarele roluri în sistem: Student, Profesor, Administrator, și Super-Administrator.  
**Rolurile și funcționalitățile utilizatorilor:**

1. Student

* Căutarea cursurilor și înscrierea la acestea.
* Vizualizarea notelor obținute.
* Vizualizarea grupurilor de studiu și membrilor acestora.
* Trimiterea și citirea mesajelor pe grupuri.
* Vizualizarea și descărcarea activităților curente sau viitoare.
* Posibilitatea de a se înscrie sau de a părăsi un grup de studiu.
* Crearea activităților în grupurile de studiu, stabilind un număr minim de participanți și perioadele disponibile.
* Vizualizarea propriului profil fără a avea dreptul de a-l modifica.

1. Profesor

* Adăugarea și programarea activităților didactice, cum ar fi cursuri, seminare, laboratoare, colocvii și examene.
* Stabilirea ponderilor notelor pentru diferite tipuri de activități și notarea studenților.
* Vizualizarea listelor de studenți.
* Descărcarea cataloagelor.
* Vizualizarea și descărcarea activităților programate curente sau viitoare.

1. Administrator

* Adăugarea, modificarea și ștergerea informațiilor din baza de date legate de utilizatori.
* Gestionarea utilizatorilor: căutare după nume și filtrare după tip.
* Asignarea profesorilor la cursuri și gestionarea informațiilor despre acestea.
* Vizualizarea tuturor studenților înscriși la un anumit curs.

1. Super-Administrator

* Gestionarea administratorilor: adăugare, modificare și ștergere a conturilor acestora.
* Gestionarea utilizatorilor cu toate drepturile de administrator, dar cu un nivel superior de control asupra sistemului.

# Descrierea Bazei de Date

## Structura generala

Structura bazei de date este alcătuită din 15 tabele  
În structura bazei de date proiectate, tabelele sunt interconectate prin următoarele relații:

1. **Tabelul adrese** are o relație One-to-One cu tabela utilizatori prin coloana adresa\_id.
2. **Tabelul utilizatori are relații:**

One-to-One cu tabela studenti prin coloana user\_id.

One-to-One cu tabela profesori prin coloana user\_id.

1. **Tabelul studenti are relații:**

One-to-Many cu tabela catalog prin coloana student\_id.

??Many-to-Many cu tabela activitati\_discipline, relația fiind gestionată prin tabela inscrieri.

Many-to-Many cu tabela grupe\_de\_studiu, relația fiind gestionată prin tabela membri\_grupe\_de\_studiu.

1. **Tabelul profesori are relații:**

Many-to-Many cu tabela discipline relația fiind gestionată prin tabela activitati\_discipline

One-to-Many cu tabela calendar prin coloana profesor\_id.

One-to-Many cu tabela activitati\_discipline prin coloana profesor\_id.

!!!One-to-Many cu tabela catalog asociate prin coloana id\_profesor

1. **Tabelul discipline are relații:**

One-to-Many cu tabela catalog prin coloana disciplina\_id.

Many-to-Many cu tabela profesori relația fiind gestionată prin tabela

One-to-Many cu tabela activitati\_discipline prin coloana disciplina\_id.

One-to-Many cu tabela grupe\_de\_studiu prin coloana disciplina\_id.

1. **Tabelul catalog are relații:**

Oane-to-One cu tabela studenti prin coloana student\_id.

Many-to-One cu tabela discipline prin coloana disciplina\_id.

!!!Many-to-Obe cu tabela profesori asociate prin coloana id\_profesor

1. **Tabelul grupe\_de\_studiu are relații:**

Many-to-Many cu tabela studenti, relația fiind gestionată prin tabela membri\_grupe\_de\_studiu.

One-to-Many cu tabela activitati\_grupe\_de\_studiu prin coloana grupa\_id.

1. **Tabelul membri\_grupe\_de\_studiu are relații:**

Many-to-One cu tabela grupe\_de\_studiu prin coloana grupa\_id.

Many-to-One cu tabela studenti prin coloana student\_id.

1. **Tabelul activitati\_grupe\_de\_studiu are relații:**

Many-to-One cu tabela grupe\_de\_studiu prin coloana grupa\_id.

Many-to-Many cu tabela studenti, relația fiind gestionată prin tabela inscrieri\_activitati\_grupe\_studiu.

1. **Tabelul inscrieri\_activitati\_grupe\_studiu are relații:**

Many-to-One cu tabela activitati\_grupe\_de\_studiu prin coloana activitate\_id.

Many-to-One cu tabela studenti prin coloana student\_id.

1. **Tabelul mesaje are relații:**

Many-to-One cu tabela grupe\_de\_studiu prin coloana grupa\_id.

Many-to-One cu tabela studenti prin coloana student\_id.

1. **Tabelul** **activitati\_discipline are relații:**

Many-to-One cu tabela discipline prin coloana disciplina\_id.

Many-to-One cu tabela profesori prin coloana profesor\_id.  
Many-to-Many cu tabela studenti, relația fiind gestionată prin tabela inscrieri.  
One-to-One cu tabela calendar prin coloana calendar\_id

1. **??Tabelul calendar are relații:**

Many-to-One cu tabela discipline prin coloana disciplina\_id.

Many-to-One cu tabela profesori prin coloana profesor\_id.  
One-to-One cu tabela calendar prin coloana calendar\_id

1. **Tabelul inscrieri are relații:**

Many-to-One cu tabela studenti prin coloana student\_id.

Many-to-One cu tabela discipline prin coloana disciplina\_id.

1. **Tabelul mesaje\_private are relații:**

Many-to-One cu tabela utilizatori prin coloana id\_destinatar.

Many-to-One cu tabela utilizatori prin coloana id\_expeditor.

## Detalii specifice

## Nivelul de Normalizare

Analiza nivelului de normalizare pentru această schemă implică evaluarea structurii tabelelor și a relațiilor dintre ele. În toate tabelele, atributele iau valori atomice și scalare, fără liste sau seturi de date. De exemplu, în tabela **utilizatori**, atributele precum **user\_id**, **nume** și **email** conțin câte o singură valoare pentru fiecare înregistrare. Astfel, toate relațiile respectă cerințele primei forme normale (FN1).

În cazul tabelelor cu chei primare simple, cum ar fi utilizatori, discipline sau grupe\_de\_studiu,

nu există dependențe parțiale între atributele non-prime și cheia primară, ceea ce îndeplinește cerințele formei normale a doua (FN2). De exemplu, în tabela activitati\_discipline, cheia primară id determină direct toate celelalte atribute, iar relațiile sunt gestionate prin chei externe, cum ar fi disciplina\_id și profesor\_id.

Nu există dependențe tranzitive între atributele non-prime și cheia primară în tabelele cu chei simple. De exemplu, în tabela studenti, cheia primară id determină direct toate atributele, cum ar fi nume, prenume sau grupa\_id, fără dependențe intermediare. Astfel, toate tabelele din schema bazei de date respectă cerințele formei normale a treia (FN3).

În ceea ce privește forma normală Boyce-Codd (BCNF), schema bazei de date este conformă deoarece toate dependențele funcționale sunt determinate de cheile candidate. De exemplu, în tabela utilizatori, dacă CNP este unic și utilizat pentru a identifica utilizatorii, acesta poate fi considerat o cheie candidat, iar toate dependențele funcționale sunt gestionate corect.

# Elemente de programare a functionalitatilor

## Procedur.Vederi.Lista completa si exemple

Proceduri:

* **creare\_grup\_studiu(nume\_disciplina, nume\_grupa)**

Creează un grup de studiu pentru o anumită disciplină, asociat cu un nume specific de grup.

* **inscriere\_student\_grup\_studiu(student\_id, nume\_disciplina)**

Permite înscrierea unui student într-un grup de studiu, verificând dacă acesta este deja înscris la disciplina respectivă.

* **AdaugaActivitate(student\_id, nume\_disciplina, descriere, data\_activitate, ora, durata, min\_participanti, timp\_expirare)**

Permite unui student să adauge o activitate în grupul de studiu, specificând detalii precum data, ora, durata, numărul minim de participanți și timpul de expirare. În cazul în care nu se înscriu suficienți studenți în timpul alocat, activitatea va fi anulată.

* **inscriere\_activitate\_studiu(student\_id, descriere\_activitate, denumire\_disciplina)**

Permite unui student să se înscrie la o activitate dintr-un grup de studiu, cu condiția să facă parte din grupul asociat disciplinei respective.

* **vizualizare\_membri\_grupa(student\_id, nume\_disciplina)**

Permite unui student să vadă lista completă a membrilor care fac parte dintr-un grup de studiu din care face si el parte.

* **parasire\_student\_grup\_studiu(student\_id, nume\_disciplina)**

Permite unui student să părăsească un grup de studiu asociat unei discipline.

* **adauga\_mesaj\_grup\_studiu(student\_id, nume\_disciplina, mesaj)**

Permite unui student să adauge mesaje în chat-ul unui grup de studiu.

* **vizualizare\_mesaje\_student\_grup\_studiu(student\_id, nume\_disciplina)**

Permite unui student să vizualizeze toate mesajele postate în chat-ul unui grup de studiu.

* **programare\_examen\_colocviu(profesor\_id, nume\_disciplina, tip\_activitate, data\_activitate, ora\_start, durata)**

Permite unui profesor să programeze un examen sau colocviu în calendar, specificând detalii precum data, ora de început și durata activității.

* **programare\_activitate(profesor\_id, nume\_disciplina, tip\_activitate, zi, ora\_start)**

Permite unui profesor să programeze o activitate didactică (curs, seminar, laborator) în calendar, indicând ziua, ora de început și tipul activității.

* **vizualizare\_grupe\_student(student\_id)**

Permite unui student să vadă toate grupurile de studiu la care este înscris.

* **vizualizare\_activitati\_profesor(profesor\_id, data\_aleasa)**

Permite unui profesor să vizualizeze toate activitățile programate pentru ziua curentă sau pentru o dată viitoare specificată.

* **vizualizare\_activitati\_student(student\_id, data\_aleasa)**

Permite unui student să vizualizeze toate activitățile la care este înscris pentru ziua curentă sau pentru o dată viitoare specificată.

* **sugesti\_de\_participanti(grup\_id, student\_id)**

Oferă sugestii de participanți pentru activitățile dintr-un grup de studiu, în funcție de studenții înscriși în grupul respectiv.

* **adauga\_profesor\_la\_activitate(denumire, nume, prenume)**

Permite adăugarea unui profesor la o activitate de studiu, specificând denumirea activității și numele profesorului.

* **sugereaza\_activitate\_grup(student\_id)**

Sugerează unei activități dintr-un grup de studiu la care studentul este înscris, luând în considerare orarul acestuia și activitățile disponibile.

* **afiseaza\_activitati\_studiu(student\_id)**  
  Permite afișarea tuturor activităților din ziua curentă sau din viitor, din grupurile de studiu la care studentul este înscris și la care se poate alătura.
* **afisare\_studenti\_curs(nume\_disciplina)**

Permite unui administrator să afișeze toți studenții înscriși la un anumit curs.

* **cautare\_dupa\_nume\_si\_rol(nume, prenume, rol)**

Permite unui administrator să caute utilizatori pe baza numelui și să filtreze rezultatele după rol. Dacă rolul selectat este „fara rol,” sistemul returnează toți utilizatorii cu acel nume, indiferent de rol.

* **inserare\_administrator(p\_CNP, p\_prenume, p\_nume, p\_adresa\_id, p\_telefon, p\_email, p\_iban, p\_nr\_contract, p\_parola)**

Adaugă un utilizator cu rolul de "administrator" în tabela utilizatori.

* **inserare\_profesor(p\_CNP, p\_prenume, p\_nume, p\_adresa\_id, p\_telefon, p\_email, p\_iban, p\_nr\_contract, p\_parola, p\_departament, p\_ore\_min, p\_ore\_max)**

Adaugă un utilizator cu rolul de "profesor" și îl asociază cu informații suplimentare din tabela profesori.

* **inserare\_student(p\_CNP, p\_prenume, p\_nume, p\_adresa\_id, p\_telefon, p\_email, p\_iban, p\_nr\_contract, p\_parola, p\_an\_studiu, p\_grupa, p\_nr\_ore)**

Adaugă un utilizator cu rolul de "student" și îi asociază informații specifice din tabela studenti.

* **inserare\_disciplina(nume\_disciplina, descriere, data\_inceput, data\_sfarsit)**

Adaugă o disciplină nouă în tabela discipline.

* **inserare\_activitate\_discipline(nume\_disciplina\_in, profesor\_id, activitate\_predata, durata, nr\_max\_participanti, frecventa)**Adaugă o activitate specifică (curs, seminar, laborator) în cadrul unei discipline, asociată unui profesor. Verifică dacă disciplina există înainte de inserare. Dacă nu, generează o eroare.
* **modifica\_ponderi\_disciplina(profesor\_id\_in, nume\_disciplina\_in, pondere\_curs\_in, pondere\_seminar\_in, pondere\_laborator\_in)**  
  Modifică ponderile procentuale ale activităților (curs, seminar, laborator) pentru o disciplină. Permite doar profesorului care predă cursul să modifice aceste valori și verifică dacă suma ponderilor este egală cu 100%.
* **afisare\_informatii\_disciplina(nume\_disciplina\_in)**  
  Afișează informații despre o disciplină specifică, inclusiv profesorii care predau cursuri, seminare și laboratoare, precum și toate detaliile generale din tabelul discipline. Returnează o eroare dacă disciplina nu există.
* **afisare\_informatii\_utilizator\_id(utilizator\_id)**  
  Returnează detalii despre un utilizator specific, în funcție de rolul acestuia: pentru un profesor, afișează disciplinele și activitățile la care participă, pentru un student, afișează disciplinele la care este înscris și profesorii responsabili pentru fiecare tip de activitate. Generază o eroare dacă utilizatorul nu există.
* **actualizeaza\_nota\_student(profesor\_id\_in, student\_id\_in, nume\_disciplina\_in, nota\_curs\_in, nota\_seminar\_in, nota\_laborator\_in)**  
  Actualizează notele unui student pentru o anumită disciplină (curs, seminar, laborator). Dacă studentul nu este înscris la o activitate a profesorului sau disciplina nu există, generează o eroare. Calculează nota finală pe baza ponderilor activităților.
* **afiseaza\_catalog\_student(student\_id\_in)**  
  Returnează catalogul pentru un student specific, afișând disciplina, numele profesorului, notele pe activități (curs, seminar, laborator) și nota finală.
* **afiseaza\_studenti\_la\_disciplina(profesor\_id\_in, disciplina\_id\_in)**  
  Listează toți studenții înscriși la activitățile unei discipline predate de un profesor specific, incluzând numele, prenumele, anul de studiu, grupa și disciplina.
* **inscriere\_student\_la\_disciplina(student\_id\_in, nume\_disciplina\_in)**  
  Inscrie un student la cursul, seminarul și laboratorul unei discipline, în funcție de disponibilitate și fără suprapuneri de orar. Verifică existența disciplinei și actualizează automat tabelul catalog cu studentul și profesorul alocat.
* **vizualizare\_studenti(profesor\_id\_in, nume\_disciplina\_in, activitate\_disciplina\_in)**  
  Afișează lista studenților înscriși la o anumită activitate (curs, seminar, laborator) a unui profesor, împreună cu notele acestora (curs, seminar, laborator și nota finală).
* **actualizare\_utilizator(p\_ID\_utilizator, p\_CNP, p\_prenume, p\_nume, p\_adresa\_id, p\_telefon, p\_email, p\_iban, p\_nr\_contract, p\_rol, p\_parola)**  
  Actualizează informațiile unui utilizator (student, profesor, administrator) pe baza ID-ului utilizatorului, incluzând date personale, adresă, telefon, email, rol și parola.
* **sterge\_utilizator\_cu\_cnp(cnp\_in)**  
  Șterge un utilizator pe baza CNP-ului, împreună cu toate informațiile și activitățile acestuia (în funcție de rolul utilizatorului, fie student, fie profesor), inclusiv înregistrările asociate în tabelele de activități și studenți. Se face actualizarea corespunzătoare a numărului de studenți din activitățile respective.

## Alte exemple de Cod SQL

**Eveniment: VerificaActivitatiPeriodice**  
Acest eveniment verifică toate activitățile de studiu pentru a identifica cele al căror timp de expirare a trecut. Dacă, după expirarea timpului, activitatea nu a atins numărul minim de participanți necesari, aceasta este ștearsă automat. Înainte de ștergere, fiecare student înscris primește un mesaj privat care îl informează că activitatea a fost anulată.

**Trigger: insert\_grupa\_after\_disciplina\_insert**  
Acest trigger este activat după ce o nouă disciplină este inserată în tabela discipline. Trigger-ul creează automat o nouă intrare în tabela grupe\_de\_studiu, asociind disciplina nou adăugată cu o grupă de studiu, care are un câmp nume\_grupa setat la NULL.

# Interfața Grafică a Utilizatorului (GUI)

## Descrierea functionalitatilor per tip de utilizator

## Detalii de implementare funcționalități specifice

# Manual de Utilizare

## Înregistrare și Autentificare

## 6.2. Gestionarea Informațiilor per utilizatori

# Concluzii și dezvoltari ulterioare

## Analiza funcționalităților curente- cele mai interesante solutii adoptate

În cadrul bazei de date platforma\_de\_studiu, una dintre cele mai ingenioase metode adoptate a fost implementarea procedurilor stocate. Acestea sunt structuri de cod SQL care permit executarea unor operațiuni complexe direct în baza de date, fără a fi necesar ca logica să fie executată pe aplicația clientului. Procedurile au fost scrise astfel încât să poată fi apelate exclusiv din interfața aplicației, oferind mai multe avantaje:

* **Centralizarea logicii în baza de date**

Prin definirea procedurilor stocate, logica aplicației este centralizată în baza de date, asigurându-se astfel că regulile de business sunt respectate uniform. Aceste proceduri pot manipula datele într-un mod eficient și coerent, fără a depinde de logica implementată pe fiecare interfață a utilizatorului.

* **Reducerea complexității aplicației client**  
  Aplicația client nu trebuie să se preocupe de detaliile implementării logicii de afaceri. Procedurile stocate preiau această responsabilitate, ceea ce duce la o aplicație mai simplă și mai ușor de întreținut. Astfel, apelarea unui singur procedeu stocat poate înlocui mai multe interogări SQL complexe.
* **Optimizarea performanței**  
  Procedurile stocate sunt pre-compilate și optimizate de serverul de baze de date. Acest lucru reduce timpul de procesare al interogărilor și îmbunătățește performanța aplicației, în special când sunt necesare operațiuni multiple sau complexe asupra unui volum mare de date.

## Dezvoltari ulterioare

O platformă de studii bazată pe o bază de date poate evolua continuu pentru a sprijini atât utilizatorii (studenți, profesori, administratori), cât și instituțiile educaționale. Dezvoltările ulterioare ar trebui să includă atât îmbunătățiri tehnologice cât și funcționalități care să îmbunătățească interacțiunea cu utilizatorii și să crească eficiența administrativă. Câteva idei pentru dezvoltarea ulterioară sunt:

* **Integrarea unui sistem de gestionare a conținutului (CMS)**

Pentru gestionarea materialelor educaționale:

* + Un CMS integrat ar permite profesorilor și administratorilor să încarce și să administreze conținutul educațional direct din platformă (de exemplu, documente PDF, video, lecții interactive).
  + Profesorii ar putea să își creeze propriile lecții și să le editeze în mod dinamic, iar materialele să fie corelate automat cu disciplinele și activitățile corespunzătoare.
* **Generarea automată de rapoarte și statistici:**

Implementarea unui sistem de generare automată a rapoartelor de performanță pentru profesori și studenți (de exemplu, rapoarte de progres, rezultate la examene, participare la activități).

* **Alocarea automată a resurselor educaționale:**

Să se implementeze un sistem care să aloce automat resursele educaționale necesare în funcție de activitățile planificate (de exemplu, să stabilească sălile de curs, să aloce echipamente pentru laboratoare, etc.).

* **Înscriere la examene și colocvii:**

O adăugire importantă ar fi posibilitatea ca studenții să se poată înscrie nu doar la cursuri, ci și la examene și colocvii. Acest proces ar include selectarea examenelor disponibile pentru fiecare disciplină, cu posibilitatea de a vizualiza datele și locațiile acestora. Studenții ar putea, de asemenea, să se retragă de la examene până la o anumită dată limită.