Cabinet Medical

Nume : Pop

Prenume : Alin

Grupa : 30641

Cuprins

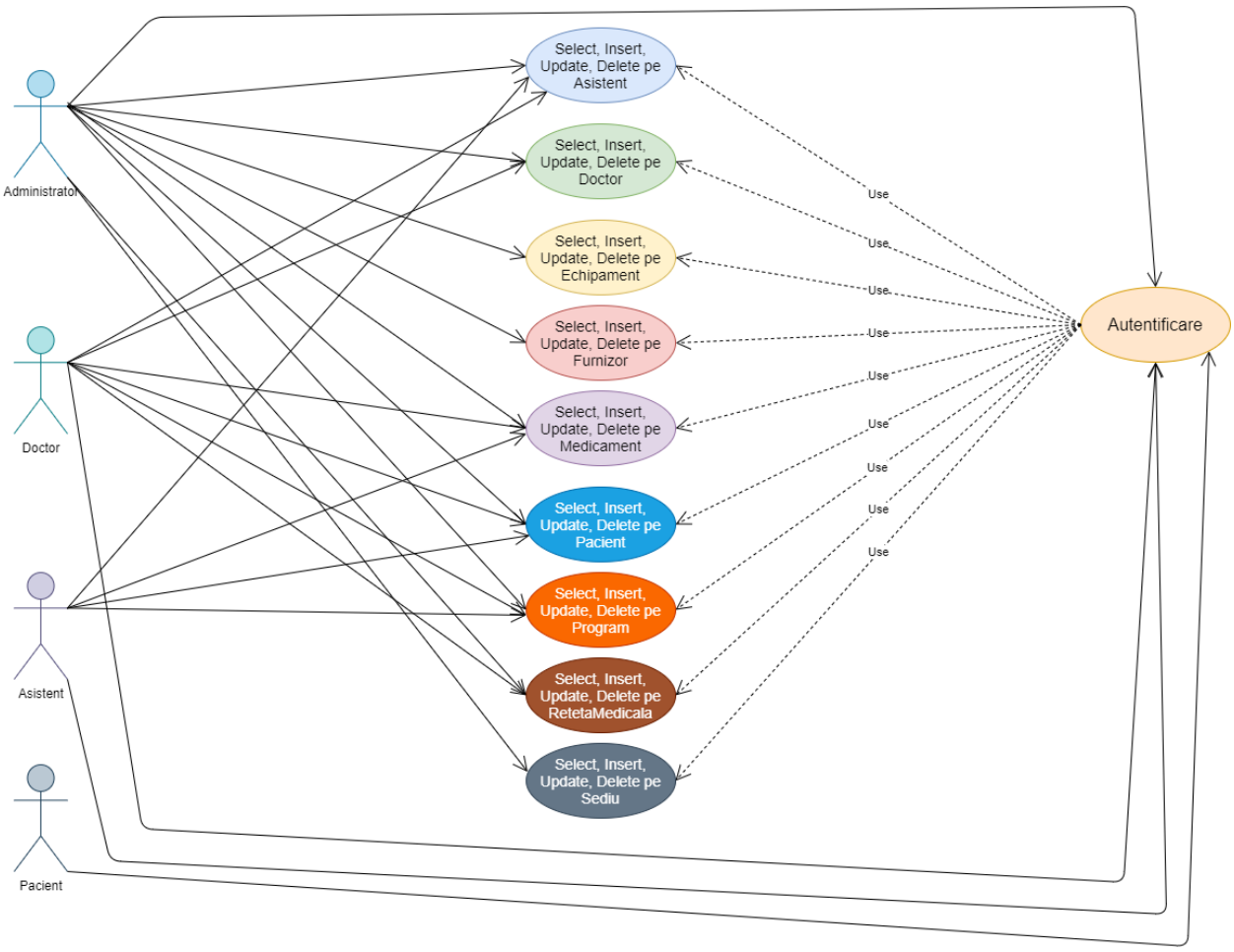
1. Specificatiile proiectului

Acest proiect reprezinta o posibila baza de date pentru un lant de cabinete medicale. Baza de date este compusa din 11 tabele, acestea fiind :

- Sediu : tabela care reprezinta cabinetul medical
- Asistent : tabela care reprezinta asistentii de la un cabinet medical
- Doctor : tabela care reprezinta docotrii de la un cabinet medical
- Pacient : tabela care reprezinta pacientii de la un cabinet medical
- RetetaMedicala : retetele medicale pe care le scrie un doctor
- Medicament : medicamentele pe care le contine o reteta medicala
- Reteta_Medicament : tabela de legatura intre Reteta si Medicament
- Program : reprezinta programul de lucru al unui doctor
- Furnizor : reprezinta furnizorii pentru un cabinet medicat
- Echipament : reprezinta echipamentul pe care il ofera furnizorii
- Furnizor_Echipament : tabela de legatura intre Furnizor si Echipaent

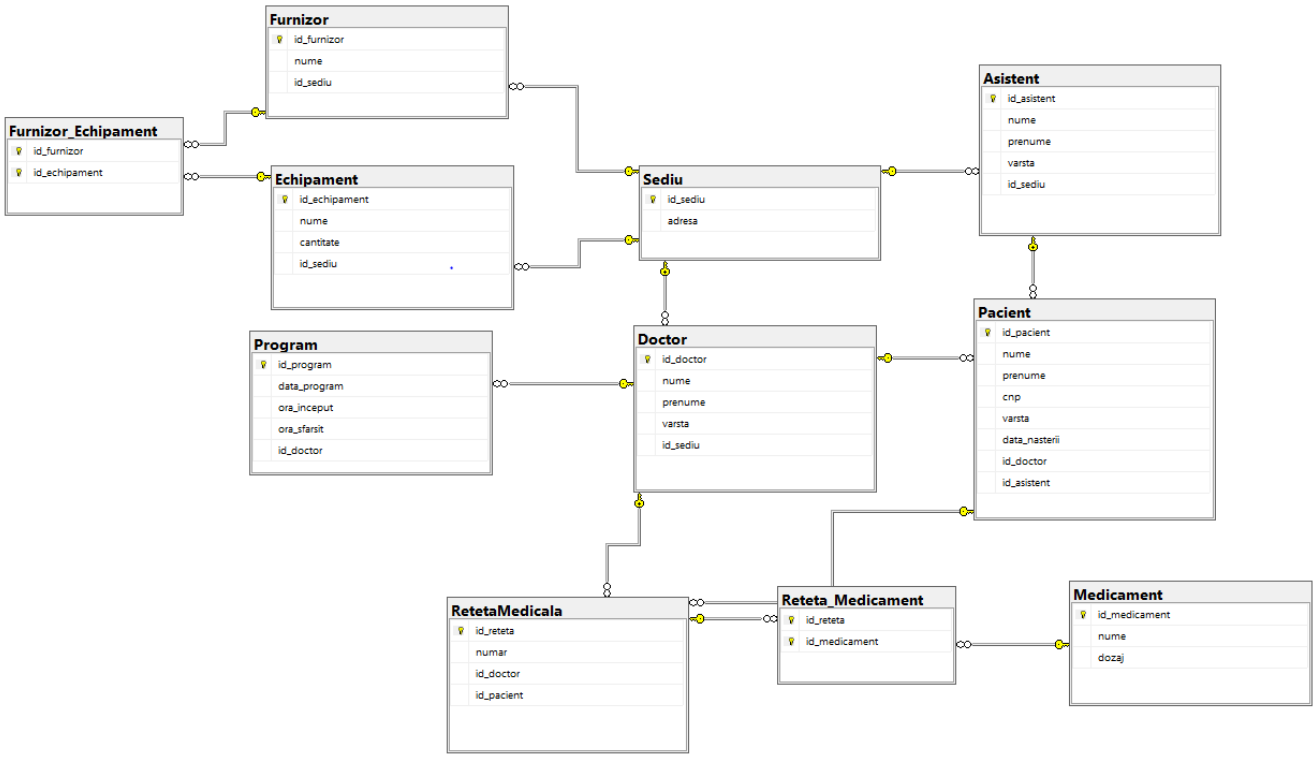
Pentru realizarea proiectului am folosit limbajul de programare SQL, iar proiectul a fost realizat in mediul de lucru Microsoft SQL Server Management Studio 18.

2. Diagrama cazurilor de utilizare



Din diagram cazurilor de utlizate se poate observa ca avem 4 tipuri de utilizatori: Administrator, Doctor, Asistent si Pacient. Fiecare, in functie de repturi, poate sa faca diferite actiuni asupra bazei de date.

3. Diagrama bazei de date



4. Descrierea elementelor implementate

4.1. Vederi

Am implementat doua vederi, acestea fiind :

- PacientiiAsistentului care afiseaza pacientii unui asistent
- PacientiiDoctorului care afiseaza pacientii unui doctor

4.2. Proceduri stocate

Am implementat 7 proceduri stocare, acestea fiind :

- **insertAsistent** (@id_asistent int, @nume varchar(50), @prenume varchar(50), @varsta int, @id_sediu int) care insereaza in tabela asistent un asistent nou cu datele transmise ca si parametric
- **viewAsistent** care afiseaza toate datele asistentilor
- **updateAsistent** (@id_asistent int, @nume varchar(50), @prenume varchar(50), @varsta int, @id_sediu int) care schimba datele asistentului cu id-ul dat ca si parametru cu celelalte date transmise prin parametri
- **deleteAsistent** (@id_asistent int) care sterge asistentul care are id-ul egal cu cel transmis ca si parametru
- **doctoriCeFolosescMed** (@nume_medicament varchar(50)) care afiseaza numele si prenumele doctorilor care au scris retete medicale in care au folosit un medicament dat ca si parametru
- **furnizorPentruCluj** (@nume_echipament varchar(50)) care afiseaza id-ul si numele furnizorului care furnizeaza un echipament dat ca si parametru pentru sediile din Cluj-Napoca
- **doctorSuprasolicitat** este o procedura care se va executa saptamanal si nu va lasa ca un doctor sa aiba mai mult de 10 pacienti, pacientii fiind mutati la doctorul cu cei mai putini pacienti

4.3. Triggere

Am implementat 4 triggere DML si 4 triggere DDL

Triggere DML :

- **doctorMax20PAcienti** care face ca atunci cand se insereaza sau modifica datele unui pacient, daca acesta este asignat unui doctor care are deja 20 de pacienti, sa fie repartizat automat doctorului cu cei mai putini pacienti
- **pacientLaAsistentLiber** care face ca atunci cand este inserat un pacient acesta sa fie asignat asistentului cu cei mai putini pacienti
- **stergereDoctorMutarePacienti** care face ca atunci cand se sterge un doctor din baza de date, pacientii acestuia sa fie asignati doctorului cu cei mai putini pacienti
- **noChanges** care nu lasa sa poata fi modificate datele din tabela Sediu

Triggere DDL :

- **ddl_delete_table** care va face ca la stergerea tabelelor din baza de date curenta sa apara un mesaj de avertizare
- **ddl_update_table** care va face ca la modificarea tabelelor din baza de date curenta sa apara un mesaj de avertizare
- **safety** care nu va mai lasa sa se faca modificari in baza de date
- **DDL_Creare_Tabele** care va face ca la crearea unei tabele in baza de date curenta, sa se afiseze numele tuturor tabelelor din baza de date si data crearii pentru fiecare tabela.

4.4. Cursoare

Am implementat doua cursoare :

- Cursor1 care este un cursor de tip scroll, pentru a putea parcurge, pe rand, tabela sediu si tabela Furnizor, pentru a putea lua adresa sediului si numele furnizorului.
- Cursor 2 este un cursor de tip scroll, pentru a putea lua pe rand numarul unei retete medicale si medicamentele pe care le contine aceasta

4.5. Utilizatori

Am implementat 4 tipuri de utilizatori :

- Administrator : are permisiuni depline asupra bazei de date
- Doctor : poate sa faca CRUD pe tabelele de Asistent, Doctor, Medicament, Pacient, Program, RetetaMedicala si sa vizualizeze datele din Echipament, Furnizor si Sediu.
- Asistent : poate sa faca CRUD pe tabelele de Asistent, Medicament, Pacient, Program si sa vizualizeze datele din Doctor, RetetaMedicala, Echipament, Furnizor si Sediu.
- Pacient : poate sa vizualizeze datele din toate tabelele

4.6. Joburi

Am implementat 4 joburi :

- **DailyFullBackupProject** este un job care se realizeaza zilnic, pentru backupul bazei de date
- **MonthlyDiffBackupProject** este un job care se realizeaza o data pe luna, pentru a realiza backupul diferential asupra bazei de date
- **WeeklyLogBackupProject** este un job care se realizeaza saptamanal, pentru a realiza transaction log backup
- **WeeklyDoctorSuprasolicitat** este un job care se realizeaza saptamanal, pentru apelarea procedurii pentru doctori suprasolicitati

4.7. Backup

Am implementat 3 tipuri de backup :

- Full backup, care se face zilnic si este salvat in folderul cu proiectul
- Differential backup, care se realizeaza o data pe luna si este salvat in folderul cu proiectul

- Transactional log backup, care se realizeaza saptamanal si este salvat in folderul cu proiectul