MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE

CATEDRA CALCULATOARE

Proiect de Semestru

la disciplina

Introducere in Baze de Date

Policlinica

An academic: 2020 – 2021

Proiect realizat de Borza Diana - Cristina

Ionescu Maria - Andreea

Niculicioiu Ștefania - Cristiana

Țîrlea Maria - Cristina

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE

CATEDRA CALCULATOARE

Cuprins

1.Introducere

1.1Introducere, argumente, scop si obiective specifice

Proiectul presupune dezvoltarea unei aplicatii care lucreaza cu baze de date pentru gestiunea activitatilor dintr-un lant de policlinici. Scopul acestei aplicatii este simplificarea operatiilor cu baza de date prin oferirea unei interfete grafice pe care personalul policlinicii(administratorii, super-administratorul, asistentii medicali, medicii, inspectorii, receptionerii si expertii contabili) sa o poata utiliza. Aplicatia ofera sprinjin atat pentru interogarea bazei de date cat si pentru manipularea acesteia.

Cu ajutorul aplicatiei utilizatorul de tip administrator poate adăuga, modifica şi şterge informaţii în baza de date legate de ceilalti utilizatori, in timp ce un utilizator de tip super-administrator poate opera inclusiv asupra utilizatorilor de tip administrator.

Un recepționer poate realiza o programare pentru un pacient. Totodată, recepționerul emite un bon fiscal ulterior consultației, cuprinzând fiecare serviciu medical care a fost efectuat.

Un asistent medical poate completa informații în rapoartele pentru analizele medicale corespunzătoare pacienților care au fost înregistrați pentru acestea.

Un medic poate vizualiza pacienții programați / înregistrați, gestiunea raportului medical (istoric, specificare secțiuni, completare informații, parafare).

Angajații de tip recepționer, asistent medical și medic pot vizualiza informațiile furnizate de modulul pentru gestiunea resurselor umane doar în ceea ce privește propria persoană (orarul pentru luna în curs, pentru fiecare zi indicându-se intervalul orar și locația, tratând și situațiile în care nu există nici un program de lucru specificat sau angajatul se află în concediu).

Un inspector poate căuta un angajat (de orice tip), în funcție de parametrii pe care îi indică: nume, prenume, funcție.

Un expert poate vizualiza informații cu privire la profitul realizat de lanțul de policlinici, pentru lunile precedente în care s-au înregistrat activități.

Angajații de tip inspector resurse umane și expert financiar contabil au la dispoziție o secțiune în care pot consulta orarul săptămânal și informații despre concediile efectuate.

Pentru dezvoltarea proiectului au fost folosite:

- **MySQL Workbench 6.2** -pentru crearea bazei de date, popularea initiala, dezvltarea de view-uri, proceduri si triggere si pentru crearea diagramei UML a tabelelor ;
- **Eclipse** –mediu de dezvoltare Java ;
- **mysql-connector-java-8.0.22.jar** –pentru stabilirea conexiuni aplicatiei cu baza de date ;
- **ObjectAid-UML**–pentru realizarea diagramei de clase din Java.

2.Analiza cerintelor utilizatorilor

2.1. Ipotezele bibliotecii (cerinte si constrangeri)

Aplicatia gestioneaza actiunile utilizatorilor unui lant de policlinici, utilizand o baza de date care este supusa urmatoarelor cerinte:

- → Exista 3 tipuri de utilizatori: administrator, super-administrator si angajat (receptioner, asistent medical, medic, inspector resurse umane, expert financiar contabil) ;
- → Orice utilizator este unic identificat printr-un Id, dar mai are si CNP, nume, prenume, adresa, numar de telefon, email, parola, IBAN, numar de contract, data angajarii, salariu negociat si numar de ore lucrate/luna;
- → Pentru un asistent medical se va reține suplimentar tipul (generalist, laborator, radiologie) și gradul (secundar, principal) ;
- → Pentru un medic se va reține suplimentar specialitatea(Radiologie si Imagistica medicala, Cardiologie, Urologie, Diabet si Nutritie, Chirurgie generala, Chirurgie toracica, Nefrologie, Neurochirurgie, Neurologie, Pneumologie, Chirurgie Pediatrica, Gastroenterologie, Chirurgie spinala) sau specialitățile în care își desfășoară activitatea, gradul (specialist, primar) codul de parafă, competențele pe care le deține pentru realizarea unor proceduri ce necesită acreditări speciale , titlul științific(doctorand sau doctor în științe medicale) , postul didactic(preparator, asistent, șef de lucrări (lector), conferențiar, profesor) .Totodată, fiecare medic are negociat un procent din serviciile medicale realizate care îi revine, adiţional față de salariul negociat. Mai mult, medicul are un orar generic caracterizat de un Id unic, data inceput si sfarsit concediu;
- → Aplicaţia va putea fi accesată, pe baza unui proces de autentificare, de către mai multe tipuri de utilizatori, operând în departamentele resurse umane, financiar-contabil sau medical, acestea regasindu-se in tabela “departament”, fiind identificate printr-un Id unic si nume;
- → Lanțul de policlinici oferă pacienților un set de servicii medicale caracterizate de un unic Id, denumire, durata serviciului exprimata in minute si pret;

- → Fiecare unitate medicala este unic identificata de Id, denumire, adresa, profit si cheltuieli, dar are si un program de lucru caracterizat prin ziua la care se referă (zi a săptămânii), zi lucratoare (daca acesta zi a saptamanii este sau nu lucratoare) intervalul orar (momentul de început și momentul de sfârșit);
- → Fiecare pacient care are o programare este unic identificat prin Id, CNP, nume, prenume;
- → Dupa prezentarea la o programare, medicul ofera pacientului un set de investigatii caracterizat de un Id unic si concluzii. Totodata medicul completeaza un raport caracterizat prin Id unic, Id-ul pacientului pentru care s-a intocmit raportul, cnp-ul, numele si prenumele medicului, cnp-ul, numele si prenumele asistentului care s-a ocupat de pacient, rezultat, diagnostic, simptome, recomandari si data raportului;
- → Fiecare programare realizata de receptionist are un Id unic, data, ora si durata (exprimata in minute) programarii, Id-ul pacientului pentru care s-a facut programarea, cnp-ul medicului si Id-ul unitatii la care s-a facut programarea;

2.2.Organizarea structurata tabelar a cerintelor utilizator

Baza de date trebuie sa stocheze urmatoarele informatii:

- ✓ Utilizatorii policlinicii ;
- ✓ Programul de lucru, concediile, orarul pentru fiecare angajat ;
- ✓ Profitul pentru un anumit medic, locație sau specialitate medicală ;
- ✓ Programare si inregistrare pacient ;

Mai mult trebuie sa permita si urmatoarele operatii:

- ✓ Crearea unui nou cont de utilizator;
- ✓ Verificarea existentei unui utilizator;
- ✓ Introducerea unui asistent medical/medic ;
- ✓ Introducerea unitatii medicale pentru un anumit angajat;
- ✓ Adaugarea unei programari;

- ✓ Adaugarea unei serviciu la o programare;
- ✓ Returnarea serviciilor medicale pe care un medic le presteaza si durata acestora;
- ✓ Adaugarea unui raport;
- ✓ Adaugarea unei investigatii ;
- ✓ Afisarea istoricului unui pacient;
- ✓ Returnarea orarului de programari ale unui medic;
- ✓ Introducerea concediului;

2.3.Determinarea si caracterizarea profilurilor de utilizator

Avem 3 tipuri de utilizatori:

1) Administratori, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
    - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
    - ❖ Adăuga, modifica şi şterge informaţii legate de utilizator;
2) Super-adminstratori, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
    - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
    - ❖ Adăuga, modifica şi şterge informaţii legate de utilizator dar si de adminstrator;
3) Angajati, care cuprind 5 sub-categorii:
    I. Asistenti medicali, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
        - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
        - ❖ Vizualizarea informațiilor furnizate de modulul pentru gestiunea resurselor umane doar în ceea ce privește propria persoană;
        - ❖ Completarea informațiilor în rapoartele pentru analizele medicale corespunzătoare pacienților care au fost înregistrați pentru acestea;

II. Medici, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
   - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
   - ❖ Vizualizarea informațiilor furnizate de modulul pentru gestiunea resurselor umane doar în ceea ce privește propria persoană;
   - ❖ Consultarea profitului pe care l-a generat;
   - ❖ Vizualizarea pacienților programați la el pentru ziua calendaristică în curs;

III. Receptioneri, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
   - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
   - ❖ Realizarea unei programari pentru un pacient;
   - ❖ Emiterea unui bon fiscal ulterior unei consultații, cuprinzând fiecare serviciu medical care a fost efectuat;

IV. Inspectori resurse umane, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
   - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
   - ❖ Căutarea unui angajat (de orice tip);
   - ❖ Consultarea orarului săptămânal și informații despre concediile efectuate;

V. Experti financiari contabili, care au urmatoarele posibilitati in aplicatie si asupra bazei de date:
   - ❖ Autentificarea/Deautentificarea si crearea unui cont nou in aplicatie;
   - ❖ Vizualizarea informațiilor cu privire la profitul realizat de lanțul de policlinici;
   - ❖ Consultarea orarului săptămânal și informații despre concediile efectuate;

3.Modelul de date si descrierea acestuia

3.1. Entitati si atributele lor

Tabelele:

Asistent_medical: informatii despre asistentii medicali ai policlinicii. Atribute: cnp_asistent, id_functie, tip, grad.

Bonfiscal: informatii despre datele ce se afla pe un bon fiscal la emiterea acestuia. Atribute: id_generareBon, id_medic, pret_servicii, data_bon.

Competenta: informatii despre competentele medicilor. Atribute: id_competenta, tip_competenta, id_specialitate.

Departament: informatii despre cele 3 departamente ale policlinicii. Atribute: id_departament, nume_departament.

Expert_financiar_contabil: informatii despre expertii policlinicii. Atribute:

Cnp_contabil, id_functie.

Functie: informatii despre functiile pe care un anagajat le poate avea. Atribute: id_functie, denumire_functie.

Inspector_resurse_umane: informatii despre inspectorii policlinicii. Atribute: cnp_inspector, id_functie.

Investigatii: informatii pe care un medic le va scrie in raportul unui pacient. Atribute: id_investigatii, concluzie, id_raport.

Medic: informatii despre medicii policlinicii. Atribute: cnp_medic, id_functie, id_specialitate, grad, cod_parafa, titlu_stiintific, post_didactic, procent_aditional.

Orar_generic: informatii despre orarul unui medic. Atribute: id_orar_generic, id, data_inceput_concediu, data_sfarsit_concediu.

Pacienti: informatii despere pacientii care se prezinta la policlinica. Atribute: id_pacient, cnp, nume, prenume.

Program: informatii despre programul policlinicii. Atribute: nume_zi, zi_lucratoare, ora_inceput, ora_sfarsit.

Programare: informatii despre programarile facute de pacienti. Atribute: id_programare, id_pacient, cnp_medic, data_programare, ora_programare, durata, id_unitate_medicala.

Raport: informatii despre raportul intocmit unui pacient. Atribute: id_raport, id_pacient, cnp_medic, cnp_asistent, id_programare, rezultat, diagnostic, simptome, recomandari, id_investigatie, nume_medic, prenume_medic, nume_asistent, prenume_asistent, dataRap.

Receptioner: informatii despre receptionerii policlinicii. Atribute: cnp_receptioner, id_functie.

Servicii: informatii despre serviciile oferite de policlinici. Atribute: id_serviciu, denumire_serviciu, id_specialitate, durata_min, pret.

Servicii_programare: informatii despre serviciile oferite unui pacient. Atribute: idP, id_programare, id_serviciu.

Specialitatea: informatii despre specialitatile in care un medic isi desfasoara activitatea. Atribute: id_specialitate, denumire_specialitate.

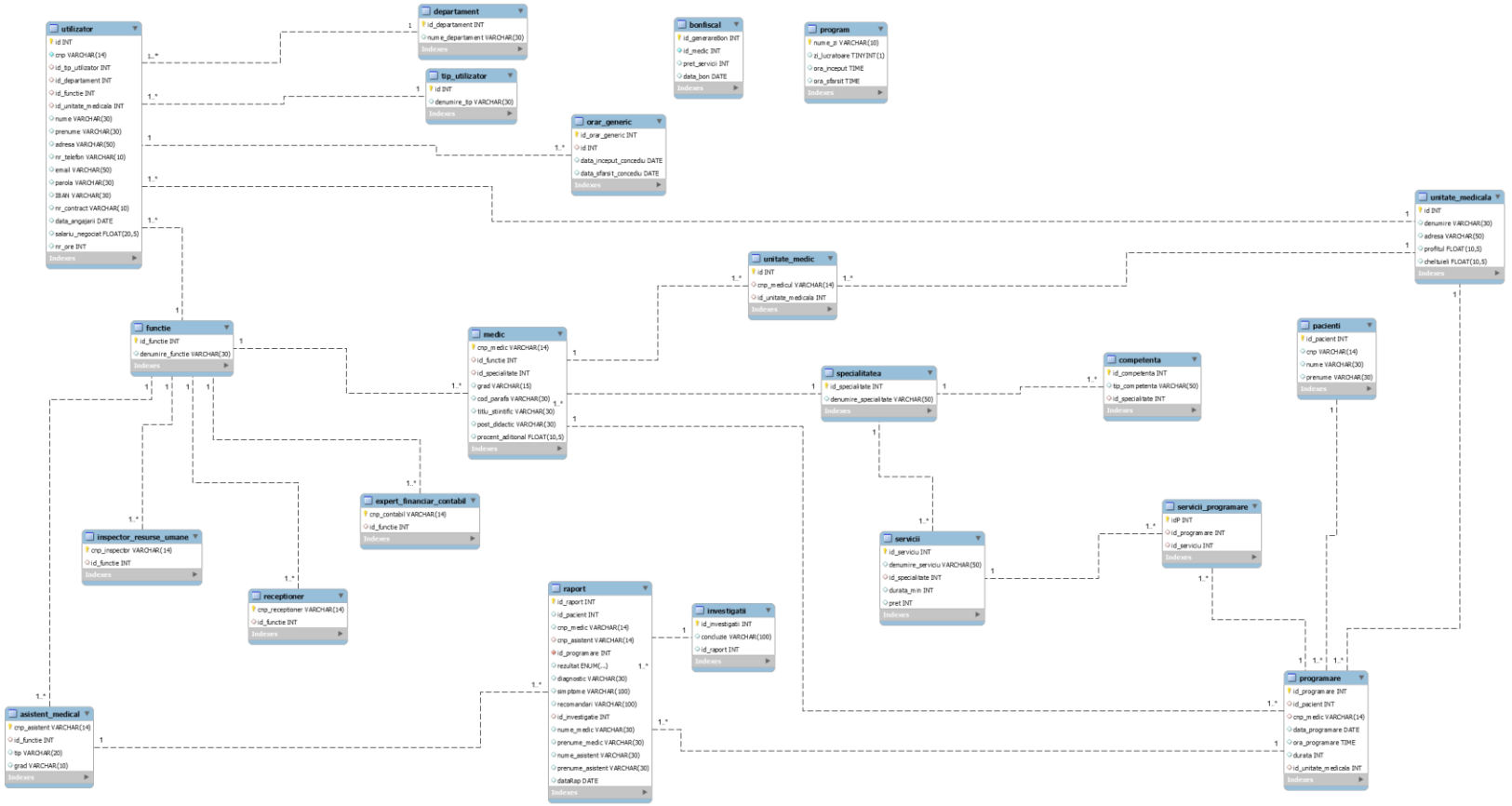
Tip_utilizator: informatii despre tipurile de utilizatori ai aplicatiei. Atribute: id, denumire_tip.

Unitate_medic: informatii despre unitatile medicale la care medicii isi desfasoara activitatea. Atribute: id, cnp_medicul, id_unitate_medicala.

Unitate_medicala: informatii despre unitatile medicale ale policlinicii. Atribute: id, denumire, adresa, profitul, cheltuieli.

Utilizator: informatii despre utilizatorii aplicatiei. Atribute: id, cnp,id_tip_utilizator,id_departament, id_functie, id_unitate_medicala, nume, prenume, adresa, nr_telefon, email, parola, IBAN, nr_contract, data_angajarii, salariu_negociat, nr_ore.

3.2.Diagrama EER/UML pentru modelul de date complet



3.3 Proceduri

3.3.1. preluareDate(in idU int)

- selecteaza datele despre utilizator in functie de cheia primara id

idU – id-ul utilizatorului

3.3.2. getOrarProgramari(in cnpMedic varchar(14))

- returneaza programarile unui medic, ordonate in functe de data in care au fost inregistrate acestea

cnpMedic - CNP-ul medicului asociat programarilor returnate

3.3.3. inserareConcediu(in idUtilizator int,in dataInceput date,in datasfarsit date)

- insereaza in Orar_Generic date referitoare la concediul unui angajat

idUtilizator – id-ul utilizatorului pentru care se vor insera datele

dataInceput – data la care incepe concediul

datasfarsit – data la care se incheie perioada de concediu

3.3.4. adauga_programare(in cnp_medic varchar(14), in cnp_pacient varchar(14), in nume_pacient varchar(30), in prenume_pacient varchar(30), in nume_medic varchar(30), in prenume_medic varchar(30),in data_prog date, in ora time, in durata int,in idLocatie int)

- insereaza datele specifice unei programari atunci cand aceasta este inregistrata

cnp_medic – CNP-ul medicului

cnp_pacient – CNP-ul pacientului

nume_pacient, prenume_pacient – numele si prenumele pacientului

nume_medic, prenume_medic - numele si prenumele medicului

data_prog – data programarii

ora – ora programarii

durata – durata programarii

idLocatie- id-ul unitatii medicale

3.3.5. adauga_serviciu_prog(in cnp_medic varchar(14), in cnp_pacient varchar(14), in serviciu_med varchar(50), in durata_serv int, out ok int)

- face legatura intre programare si serviciile asociate acesteia, inserand date in tabelul **servicii_programare,** dar si in tabelul **pacienti** in cazul in care pacientul nu se afla deja in baza de date

cnp_medic – CNP-ul medicului

cnp_pacient – CNP-ul pacientului

serviciu_med – denumirea serviciului medical

durata_serv – durata serviciului medical

ok – variablia returnata primeste valoarea 1 daca inserarile au fost efectuate cu cu succes, sau 0 in caz contrar

3.3.6. getServiciiMedic(in cnp_med varchar(14))

- returneaza serviciile oferite de un anumit medic in functie de specialitatea si competentele pe care le detine

cnp_medic – CNP-ul medicului

3.3.7. getDurataServiciiMedic(in denumireServ varchar(50))

- returneaza durata unui serviciu medical

denumireServ – denumirea serviciului medical

3.3.8. adauga_raport(in cnp_medic varchar(14), in cnp_pacient varchar(14), in cnp_asistent varchar(14),in diagnostic varchar(30), in simptome varchar(100), in recomandari varchar(100),in rez varchar(10), out ok int, in nume_med varchar(30), in prenume_med varchar(30), in nume_a varchar(30), in prenume_a varchar(30), in dataR date)

- insereaza datele corespunzatoare unui raport medical

cnp_medic – CNP-ul medicului

cnp_pacient – CNP-ul pacientului

cnp_asistent – CNP-ul asistentului medical

diagnostic – diagnosticul

simptome - simptomele

recomandari – recomandarile medicului

rez – rezultatul analizelor medicale, confirmat de catre asistent

ok – variablia returnata primeste valoarea 1 daca inserarile au fost efectuate cu cu succes, sau 0 in caz contrar

nume_med, prenume_med - numele si prenumele medicului

nume_a, prenume_a - numele si prenumele asistentului

dataR – data in care se inregistreaza raportul medical

3.3.9. adauga_investigatie(in concluzie varchar(100), out ok int)

- insereaza in tabela **investigatii** concluziile medicului in legatura cu investigatiile medicale efectuate in timpul programarii

concluzie - concluzia medicului

ok – variablia returnata primeste valoarea 1 daca inserarile au fost efectuate cu cu succes, sau 0 in caz contrar

3.3.10. getIstoric(in cnp_pacient varchar(14))

-returneaza toate rapoartele medicale anterioare, inregistrate pentru un anumit pacient

cnp_pacient – CNP-ul pacientului

3.3.11. cont_nou(cnp_1 varchar(14),nume_1 varchar(30),prenume_1 varchar(30),adresa_1 varchar(50),tel_1 varchar(10),email_1 varchar(50),iban_1 varchar(30),nr_1 varchar(10),angajare date,tip_util varchar(30),parola_1 varchar(30),salar_neg float(10,5),nr_ore_1 int,tip_functie varchar(30),unit_med varchar(30))

- insereaza in tabelul **utilizator** datele introduse de acesta in momentul crearii unui cont nou pe platforma lantului de policlinici

cnp_1 – CNP-ul utilizatorului

nume_1, prenume_1 – numele, prenumele utilizatorului

adresa_1 – adresa utilizatorului

tel_1 – numarul de telefon

email_1 – adresa de email

iban_1 - IBAN

nr_1 – numar de contact

parola_1 – parola utilizatorului

salar_neg – salarul negociabil

nr_ore_1 – numarul de ore

tip_functie – id-ul functiei

unit_med – id-ul unitatii medicale

3.3.12. insert_asistent(tip_1 varchar(20),grad_1 varchar(10))

- insereaza datele aditionale specifice unui angajat cu functia asistent medical

tip_1 – tipul asistentului medical

grad_1 – gradul asistentului medical

3.3.13. insert_medic(grad_1 varchar(15),cod_1 varchar(30),titlu varchar(30),post_1 varchar(30),procent_1 float,specialitate_1 varchar(100))

- insereaza datele aditionale specifice unui medic

grad_1 – gradul asistentului medical

cod_1 – codul parafei

post_1 – postul didactic ocupat

procent_1 – procentul aditional

specialitate_1 – specialitatea medicului

3.3.14. insert_unitate(nume_unitate varchar(30))

- insereaza in tabela **unitate_medic** unitatile medicale in care un medic ofera servicii in functie de cnp-ul acestuia

nume_unitate – numele unitatii medicale

3.3.15. modificareDate(in idUtilizator int,in salariuModificat float(20,5),in adresaModificata varchar(50), in telefonModificat varchar(10))

- modifica in tabela **utilizator** datele retinute in campurile pentru salariu, adresa si numar de telefon

idUtilizator – id-ul utilizatorului pentru care se vor modifica datele

salariuModificat – noul salariu inregistrat dupa modificare

adresaModificata – noua adresa inregistrata dupa modificare

telefonModificat – noul numar de telefon inregistrat dupa modificare

3.3.16. adaugaBon(in cnpM varchar(14), in pret int,in dataBon date)

- insereaza in tabela bon_fiscal datele corespunzatoare, in urma generarii unui astfel de bon

cnpM – CNP-ul medicului

pret – pretul total al serviciilor pentru care s-a efectuat programarea

dataBon – data emiterii bonului fiscal

3.4. Normalizarea datelor

Definitia formei normale Boyce-Codd (FNBC) este: Fie R o schemă de relație si F multimea de dependențe funcționale asociată. Se spune ca R este în forma normală Boyce-Codd dacă și numai dacă oricare ar fi o dependență netrivială X →Y din F, atunci X este supercheie pentru R.

Baza de date respect acest lucru. Atributele fiecărui tabel nu depind de alte atribute. Fiecare tabel are o singură cheie primară după care sunt identificate înregistrările și este suficientă pentru a identifica în mod unic orice înregistrare din baza de date.

În fiecare tabel avem doar o cheie și toate dependențele au în partea stângă o supercheie (cheia primară a tabelei).

3.5. Interogari MySQL

1. ***Interogari translatate prin algebra relationala***

```
"select id_pacient from pacienti where cnp = " + cnpP + ";"
```

$\Pi_{id\_pacient}\sigma_{cnp=cnpP}(pacienti)$

- Utilizata la parafarea raportului.

```
"Select id_functie from utilizator where id='"+id+"';"
```

$\Pi_{id\_functie}\sigma_{id=id}(utilizator)$

- Utilizata la verificarea functiei utlizatorului in momentul salectarii modulului pentru gestiunea activitatilor operationale.

```
"select IBAN,nr_contract,data_angajarii,salariu_negociat from utilizator where id='"+id+"'"
```

$\Pi_{IBAN,nr\_contract,data\_angajarii,salariu\_negociat}\sigma_{id=id(utilizator)}$

- Utilizata pentru afisarea datelor unui medic.

```
"Select data_inceput_concediu,data_sfarsit_concediu from orar_generic “+where id=’”+id”’;”
```

$\Pi_{data\_inceput\_concediu,data\_sfarsit\_concediu}\sigma_{id=id(orar\_generic)}$

- Uilizata pentru afisarea datelor despre concediu din orarul generic al unui medic.

```
"Select id from utilizator where email='"+email+"' and parola='"+parola+"'"
```

$\Pi_{id}\sigma_{email=email,parola=parola(utilizator)}$

- Uilizata pentru determinarea id-ului unui utilizator in functie de datele de logare.

```
"Select * from utilizator where email='"+email_1+"' and parola='"+password_1+"'"
```

$\Pi_{*}\sigma_{email=email\_1,parola=password\_1(utilizator)}$

- Utilizata la verificarea datelor de inregistrare in aplicatie.

2. ***Exemple de interogari din codul java***

```
"select nume,prenume from utilizator where
id_tip_utilizator!='2' and id_tip_utilizator!='1'"
```

- Utilizata la afisarea numelui si prenumelui tuturor angajatilor in afara de administrator/super-administrator.

```
"select cnp,salariu_negociat from utilizator where
id='"+id+"';"
```

- Utilizata la afisarea cnp-ului si al salariului unui utilizator in functie de id.

```
"select pret_servicii from bonFiscal where
id_medic='"+id+"' and month(data_bon)='"+luna+"';"
```

- Utilizata la afisarea pretului unui anumit serviciu de pe bonul fiscal in functie de id-ul medicului care a facut acel serviciu si luna in care acest lucru s-a intamplat.

```
"select procent_aditional from medic where
cnp_medic='"+cnpMedic+"';"
```

- Utilizata la afisarea procentului aditional al medicului in functie de id.

```
"select id from unitate_medicala where
denumire='"+unitate+"';"
```

- Utilizata pentru determinarea id-ului unei unitati medicale in functie de denumirea unitatii medicale.

```
"Select pret from servicii where denumire_serviciu = ?;"
```

- Utilizata pentru afisarea pretului unui serviciu in functie de denumirea fiecarui serviciu in parte.

```
"select ora_programare, ADDTIME(ora_programare,
SEC_TO_TIME(durata*60)) from programare where
data_programare = '" +date + "' and cnp_medic =
'"+cnpM+"';"
```

- Utilizata pentru determinarea orei in care este facuta o programare in functie de data acesteia si cnp-ul medicului caruia ii este asignata.

```
"select medic.cnp_medic from medic,unitate_medic where
medic.cnp_medic=unitate_medic.cnp_medicul and
unitate_medic.id_unitate_medicala='"+idUnitate+"';"
```

- Utilizata pentru determinarea cnp-ului medicului in functie de unitatea medicala in care acesta lucreaza.

```
"select salariu_negociat from utilizator where
id_unitate_medicala='"+idUnitate+"';"
```

- Utilizata pentru determinarea salariului unui utilizator in functie de unitatea medicala in care lucreaza.

```
"select  IBAN, nr_contract, data_angajarii,
salariu_negociat, id_unitate_medicala from utilizator
where id='"+id+"'"
```

- Utilizata pentru afisarea IBAN-ului, numarului contractului, data angajarii, salariului si id-ul unitatii medicale in functie de id-ul utilizatorului.

3.6. Cod MySQL

3.6.1. Cod pentru crearea bazei de date si a tabelelor

```
drop database if exists policlinica;

create database if not exists policlinica;
use policlinica;

drop table if exists asistent_medical;
drop table if exists competenta;
drop table if exists departament;
drop table if exists expert_financiar_contabil;
drop table if exists functie;
drop table if exists inspector_financiar_contabil;
drop table if exists functie;
drop table if exists inspector_resurse_umane;
drop table if exists investigatii;
drop table if exists medic;
drop table if exists orar_generic;
drop table if exists orar_medic;
drop table if exists pacienti;
drop table if exists program;
drop table if exists programare;
drop table if exists raport;
drop table if exists receptioner;
drop table if exists servicii_programare;
drop table if exists specialitatea;
drop table if exists tip_utilizator;
drop table if exists utilizator;
drop table if exists unitate_medicala;


create table unitate_medic
(id int unique not null primary key auto_increment,
cnp_medicul varchar(14),
id_unitate_medicala int);

create table  unitate_medicala
(id int not null primary key auto_increment,
```

```
denumire varchar(30),
adresa varchar(50),
profitul float(10,5),
cheltuieli float(10,5)
);

create table bonFiscal
(
        id_generareBon int not null primary key auto_increment,
   id_medic int not null,
   pret_servicii int,
   data_bon date
);

create table program
(nume_zi varchar(10) not null primary key,
zi_lucratoare bool,
ora_inceput time,
ora_sfarsit time
);

 create table tip_utilizator
(id int not null primary key,
denumire_tip varchar(30)
);

create table departament
(id_departament int not null primary key,
nume_departament varchar(30)
);

create table utilizator
(id int not null primary key auto_increment,
cnp varchar(14) not null,
id_tip_utilizator int,
id_departament int,
id_functie int,
id_unitate_medicala int,
nume varchar(30),
prenume varchar(30),
adresa varchar(50),
nr_telefon varchar(10),
email varchar(50),
```

```
parola varchar(30),
IBAN varchar(30),
nr_contract varchar(10),
data_angajarii date,
salariu_negociat float(20,5),
nr_ore int
);

 create table servicii
(id_serviciu int not null primary key,
denumire_serviciu varchar(50),
id_specialitate int,
durata_min int,
pret int
);

create table inspector_resurse_umane
(cnp_inspector varchar(14) not null primary key,
id_functie int
);

 create table if not exists functie
(id_functie int not null primary key auto_increment,
denumire_functie varchar(30)
);

 create table expert_financiar_contabil
(cnp_contabil  varchar(14) not null primary key,
id_functie int
);

create table specialitatea
(id_specialitate int not null primary key auto_increment,
denumire_specialitate varchar(50)
);

create table competenta
(id_competenta int unique not null primary key auto_increment,
tip_competenta varchar(50),
id_specialitate int
);

create table servicii_programare
```

```
(idP int not null primary key auto_increment,
id_programare int,
id_serviciu int
);

create table orar_generic
(id_orar_generic int primary key,
 id int,
data_inceput_concediu date,
data_sfarsit_concediu date
);

create table receptioner
(cnp_receptioner varchar(14) not null primary key ,
id_functie int
);

create table asistent_medical
(cnp_asistent varchar(14) not null primary key ,
id_functie int,
tip varchar(20),
grad varchar(10)
);

create table medic
(cnp_medic varchar(14) not null primary key,
id_functie int,
id_specialitate int,
grad varchar(15),
cod_parafa varchar(30),
titlu_stiintific varchar(30),
post_didactic varchar(30),
procent_aditional float(10,5)
);

create table raport
(id_raport int not null primary key auto_increment,
id_pacient int,
cnp_medic varchar(14) ,
cnp_asistent varchar(14),
id_programare int unique not null,
rezultat enum('pozitiv','negativ'),
diagnostic varchar(30),
```

```
simptome varchar(100),
recomandari varchar(100),
id_investigatie int,
nume_medic varchar(30),
prenume_medic varchar(30),
nume_asistent varchar(30),
prenume_asistent varchar(30),
dataRap date
);

 create table pacienti
(id_pacient int not null primary key auto_increment,
cnp varchar(14),
nume varchar(30),
prenume varchar(30)
);

 create table programare
(id_programare int unique not null primary key auto_increment,
id_pacient int,
cnp_medic varchar(14),
data_programare date,
ora_programare time,
durata int,
id_unitate_medicala int
);

 create table investigatii
(id_investigatii int not null primary key auto_increment,
concluzie varchar(100),
id_raport int
);




alter table programare
add foreign key(id_unitate_medicala) references unitate_medicala(id);
alter table utilizator
add foreign key(id_departament) references departament(id_departament);
alter table utilizator
add foreign key(id_tip_utilizator) references tip_utilizator(id);
alter table orar_generic
```

```
add foreign key(id) references utilizator(id); ####
alter table utilizator
add foreign key(id_functie) references functie(id_functie);
alter table servicii
add foreign key(id_specialitate) references specialitatea(id_specialitate);
alter table servicii_programare
add foreign key(id_serviciu) references servicii(id_serviciu);
alter table servicii_programare
add foreign key(id_programare) references programare(id_programare);
alter table medic
add foreign key(id_specialitate) references specialitatea(id_specialitate);
alter table competenta
add foreign key(id_specialitate) references specialitatea(id_specialitate);
alter table inspector_resurse_umane
add foreign key(id_functie) references functie(id_functie);
alter table receptioner
add foreign key(id_functie) references functie(id_functie);
alter table asistent_medical
add foreign key(id_functie) references functie(id_functie);
alter table medic
add foreign key(id_functie) references functie(id_functie);
alter table expert_financiar_contabil
add foreign key(id_functie) references functie(id_functie);
alter table raport
add foreign key(cnp_asistent) references asistent_medical(cnp_asistent);
alter table raport
add foreign key(id_investigatie) references investigatii(id_investigatii);
alter table programare
add foreign key(cnp_medic) references medic(cnp_medic);
alter table programare
add foreign key(id_pacient) references pacienti(id_pacient);
alter table raport
add foreign key(id_programare) references programare(id_programare);
alter table utilizator
add foreign key(id_unitate_medicala) references unitate_medicala(id);
alter table unitate_medic
add foreign key(cnp_medicul) references medic(cnp_medic);
alter table unitate_medic
add foreign key(id_unitate_medicala) references unitate_medicala(id);

alter table orar_generic modify id_orar_generic int auto_increment ;
```

3.6.2. Cod populare baza de date

```
insert into functie(denumire_functie)
values
('Inspector resurse umane'),
('Expert financiar-contabil'),
('Medic'),
('Asistent medical'),
('Receptioner');

insert into departament
values('1','Resurse umane'),
        ('2','Financiar-Contabil'),
        ('3', 'Medical');

insert into tip_utilizator
values ('1','Administrator'),
        ('2','Super-administrator'),
        ('3','Angajat');

 insert into specialitatea
 values (null, 'Radiologie si Imagistica medicala'),
         (null,'Cardiologie'),
         (null,'Urologie'),
         (null,'Diabet si Nutritie'),
         (null, 'Chirurgie generala'),
         (null, 'Chirurgie toracica'),
```

```
 (null, 'Nefrologie'),
(null, 'Neurochirurgie'),
(null, 'Neurologie'),
(null, 'Pneumologie'),
(null, 'Chirurgie Pediatrica'),
(null, 'Gastroenterologie'),
(null, 'Chirurgie spinala');


INSERT INTO servicii(id_serviciu, denumire_serviciu, id_specialitate,durata_min, pret)
 values('1', 'Ecografie', '1', '20', '100'), -- Radiologie
        ('2', 'Endoscopie digestiva', '12', '45', '250'), -- gastroenterologie
        ('3', 'Ecocardiografie', '2', '45', '150'),  -- Cardilogie
        ('4', 'Cardiologie interventionala', '2', '120', '800'), -- Cardiologie
        ('5', 'Bronhoscopie', '10', '30', '250'), -- Pneumologie
        ('6', 'EEG','9', '60', '180'), -- Neurologie
        ('7', 'EMG', '9', '90', '180'), -- Neurologie
        ('8', 'Dializa', '7', '240', '0'), -- Nefrologie
        ('9', 'Chirurgie laparoscopica', '5', '180', '500'), -- Girurgie generala
        ('10', 'Chirurgie toracica', '6', '240', '1000'), -- Chirurgie toracica
        ('11', 'Chirurgie spinala', '13', '240','2500'), -- Chirurgie spinala
        ('12', 'Chirurgie pediatrica', '11', '180', '1500'), -- Chirurgie pediatrica
        ('13', 'Litotritie extracorporeala', '3', '120', '7500'), -- Urologie
        ('14', 'Explorare computer tomograf', '1', '45', '1000'), -- Radiologie
        ('15', 'Imagistica prin rezonanta magnetica', '1', '30', '1700'),  -- Radiologie
        ('16', 'Consiliere nutritionala', '4', '50', '850') ; -- Diabet si nutritie
```

```
insert into competenta
values ('1', 'ecografie', '1'), -- Radiologie
        ('2', 'endoscopie digestiva', '12'), -- gastroenterologie
        ('3', 'ecocardiografie', '2'),  -- Cardilogie
        ('4', 'cardiologie interventionala', '2'), -- Cardiologie
        ('5', 'bronhoscopie', '10'), -- Pneumologie
        ('6', 'EEG','9'), -- Neurologie
        ('7', 'EMG', '9'), -- Neurologie
        ('8', 'dializa', '7'), -- Nefrologie
        ('9', 'chirurgie laparoscopica', '5'), -- Girurgie generala
        ('10', 'chirurgie toracica', '6'), -- Chirurgie toracica
        ('11', 'chirurgie spinala', '13'), -- Chirurgie spinala
        ('12', 'chirurgie pediatrica', '11'), -- Chirurgie pediatrica
        ('13', 'litotritie extracorporeala', '3'), -- Urologie
        ('14', 'explorare computer tomograf', '1'), -- Radiologie
        ('15', 'imagistica prin rezonanta magnetica', '1');  -- Radiologie


insert into unitate_medicala(denumire,adresa)
 values
 ('MedLife Cluj','Str.Unirii'),
 ('MedLife Valcea','Str.Ion Buteanu'),
 ('MedLife Sibiu','Str.Panduri'),
 ('MedLife Alba','Str.Eroilor'),
 ('MedLife Oradea','Str.Florilor'),
 ('MedLife Timisoara','Str.Independentei'),
 ('MedLife Craiova','Str.Bucuriei'),
 ('MedLife Bucuresti','Str.Gerge Enescu');
```

```
insert into program values
('Luni','1','07:00:00','15:00:00'),
('Marti','1','08:00:00','16:00:00'),
('Miercuri','1','09:00:00','17:00:00'),
('Joi','1','10:00:00','18:00:00'),
('Vineri','1','11:00:00','19:00:00'),
('Sambata','0',null,'23:00:00'),
('Duminica','0',null,'23:30:00');
```

```
insert into utilizator values (null,'61',3,1,1,3,'Inspector', 'Popescu','Str. X nr 23', '0755555555','ip@ml.ro','1','ROBTRLRON97984','1','2018-01-01',3000,14);

insert into utilizator values (null,'21',3,3,3,null,'Pop', 'Alex','Str. X nr 23', '0755555555','ap@ml.ro','1','ROBTRLRON97954','2','2018-01-01',5000,14);

insert into medic values ('21', 3, 1, 'Specialist', '21345', 'Doctorand', 'Preparator', 13.00000);

insert into unitate_medic values (null,'21',3);

insert into orar_generic values(null,2,'2022-01-01','2022-02-02');

insert into utilizator values (null,'123', 3, 2, 2,3, 'Contabil','Avram','Str. X nr 23', '0755555555', 'ca@ml.ro', '1','ROBTRLEUR654777','3', '2020-01-01',2456.00000, 12);

insert into utilizator values (null,'13', 3, 3, 4,3, 'Asistent','Mircea','Str. X nr 21', '0755555555', 'am@ml.ro', '1','ROBTRLEUR65417','4', '2020-01-01',2456.00000, 12);

insert into asistent_medical values('13',4,'Radiologie','Primar');

insert into expert_financiar_contabil values('123', 2);

insert into inspector_resurse_umane values('61',1);

insert into utilizator values (null,'51',1,null,null,null,'Admin', 'Pop','Str. X nr 23', '0755555555','admin','1','ROBTRLRON97984','1','2018-01-01',3000,14);

insert into utilizator values (null,'71',2,null,null,null,'Super Admin', 'Dan','Str. X nr 23', '0755555555','sadmin','1','ROBTRLRON97984','1','2018-01-01',3000,14);
```

insert into utilizator values (null,'81',3,3,5,3,'Recptioner', 'Dana','Str. X nr 23', '0755555555','r@ml.ro','1','ROBTRLRON97984','1','2018-01-01',3000,14);

insert into receptioner values ('81',5);

insert into pacienti values(null,'44','Pacient', 'Malina');

insert into programare values (null,1,'21', '2021-01-14','07:00:00',65,3);

insert into servicii_programare values (null, 1, 1);

insert into servicii_programare values (null, 1, 14);

3.6.3. Cod proceduri

####---------- INSERARE IN PROGRAMARI ----------####

```
DELIMITER //
CREATE PROCEDURE adauga_programare(in cnp_medic varchar(14), in cnp_pacient
varchar(14), in nume_pacient varchar(30), in prenume_pacient varchar(30), in nume_medic
varchar(30), in prenume_medic varchar(30),in data_prog date, in ora time, in durata int,in
idLocatie int)

BEGIN
set @cnp_med = null, @id_p = null;
select cnp_medic into @cnp_med from medic where  id_functie = '3' and medic.cnp_medic =
cnp_medic;

select id_pacient into @id_p  from pacienti where cnp = cnp_pacient; -- cazul in care pacientul a
mai fost programat
if (@cnp_med is not null and @id_p is null ) then    -- cazul in care pacientul se prezinta pentru
prima data la clinica; se introduc datele acestuia in BD
begin
insert into pacienti values (null, cnp_pacient, nume_pacient, prenume_pacient);
end;
end if;

select id_pacient into @id_p from pacienti where cnp = cnp_pacient;

if(@cnp_med is not null and @id_p is not null) then
begin
```

```
INSERT INTO
programare(id_programare,id_pacient,cnp_medic,data_programare,ora_programare,durata,id_un
itate_medicala) VALUES (null, @id_p, @cnp_med,data_prog, ora, durata,idLocatie);
end;
end if;

END;
//
DELIMITER ;
```

###-------- INSERARE LEGATURI INTRE SERVICII SI PROGRAMARE ------###

```
DELIMITER //
CREATE PROCEDURE adauga_serviciu_prog(in cnp_medic varchar(14), in cnp_pacient
varchar(14), in serviciu_med varchar(50), in durata_serv int, out ok int)

BEGIN
set ok = 0;
set @cnp_med = null, @id_p = null, @id_serv = null, @durata = null, @id_prog = null;
select cnp_medic into @cnp_med from medic where  id_functie = '3' and medic.cnp_medic =
cnp_medic;
select id_pacient into @id_p from pacienti where cnp = cnp_pacient;
select id_serviciu into @id_serv from servicii where denumire_serviciu = serviciu_med;
select durata_min into @durata from servicii where denumire_serviciu = serviciu_med;
select max(id_programare) into @id_prog from programare ;

if (@cnp_med is not null and @id_p is null ) then     -- cazul in care pacientul se prezinta pentru
                                                        prima data la clinica; se introduc datele acestuia in BD
begin
insert into pacienti values (null, cnp_pacient, nume_pacient, prenume_pacient);
end;
end if;

select id_pacient into @id_p from pacienti where cnp = cnp_pacient;

if(@cnp_med is not null and @id_p is not null and @id_serv is not null and @durata is not null
and @id_prog is not null) then
begin
INSERT INTO servicii_programare VALUES (null, @id_prog, @id_serv);
set ok = 1;
```

```
end;
end if;

END;
//
DELIMITER ;
```

##---------- RETURNARE SERVICII OFERITE DE UN MEDIC ----------######

```
DELIMITER //

CREATE PROCEDURE getServiciiMedic(in cnp_med varchar(14))
BEGIN
set @id_spec = null;
select id_specialitate into @id_spec  from medic where id_functie = '3' and cnp_medic =
cnp_med;
select denumire_serviciu from servicii where id_specialitate = @id_spec;
END ;
//
DELIMITER ;
```

######---------- RETURNARE DURATA UNUI SERVICIU ----------######

```
DELIMITER //

CREATE PROCEDURE getDurataServiciiMedic(in denumireServ varchar(50))
BEGIN
select durata_min from servicii where denumire_serviciu = denumireServ;
END ;
//
DELIMITER ;
```

########---------- ADAUGARE UNUI RAPORT ----------######

```
DELIMITER //
CREATE PROCEDURE adauga_raport(in cnp_medic varchar(14), in cnp_pacient varchar(14),
in cnp_asistent varchar(14),in diagnostic varchar(30), in simptome varchar(100), in recomandari
varchar(100),in rez varchar(10), out ok int, in nume_med varchar(30), in prenume_med
varchar(30), in nume_a varchar(30), in prenume_a varchar(30), in dataR date)

BEGIN
```

```
set ok = 0;
set @cnp_med = null, @id_p = null, @id_prog = null, @cnpA = null, @id_rap = null;
select cnp_medic into @cnp_med from medic where  id_functie = '3' and medic.cnp_medic =
cnp_medic;
select id_pacient into @id_p  from pacienti where cnp = cnp_pacient;
select max(id_programare) into @id_prog from programare where id_pacient = @id_p and
cnp_medic = @cnp_med;
select cnp_asistent into @cnpA from asistent_medical where asistent_medical.cnp_asistent =
cnp_asistent;

if(@cnp_med is not null and @id_p is not null and @id_prog is not null) then
begin
INSERT INTO raport(id_raport, id_pacient,cnp_medic, cnp_asistent,id_programare,rezultat,
diagnostic, simptome, recomandari,nume_medic,prenume_medic,nume_asistent,
prenume_asistent,dataRap)
VALUES (null, @id_p, @cnp_med, @cnpA, @id_prog,rez, diagnostic, simptome, recomandari,
nume_med, prenume_med,  nume_a, prenume_a, dataR);
select max(id_investigatii) into @id_inv from investigatii;
select max(id_raport) into @id_rap from raport;
UPDATE policlinica.investigatii SET id_raport = @id_rap WHERE id_investigatii = @id_inv;
update policlinica.raport set id_investigatie = @id_inv where id_raport = @id_rap;
set ok = 1;
end;

end if;

END;
//
DELIMITER ;
```

######## ---------- **INSERARE INVESTIGATII** ---------- ########

```
DELIMITER //
CREATE PROCEDURE adauga_investigatie(in concluzie varchar(100), out ok int)
BEGIN
set ok = 0;
if(concluzie is not null) then
begin
INSERT INTO investigatii(id_investigatii, concluzie ) VALUES (null,concluzie);
set ok = 1;
end;
end if;
```

```
END;
//
DELIMITER ;
```

##########---------- ISTORIC ----------##########

```
DELIMITER //
CREATE PROCEDURE getIstoric(in cnp_pacient varchar(14))
BEGIN

set @id_p = null;
select id_pacient into @id_p  from pacienti where cnp = cnp_pacient;
if(@id_p is not null) then
begin
select id_raport, dataRap, nume_medic, prenume_medic,nume_asistent, prenume_asistent,
simptome,diagnostic,recomandari, rezultat from raport where id_pacient = @id_p;
end;
end if;

END;
//
DELIMITER ;
```

#######---------- INSERARE DATE UTILIZATOR ----------##########

```
DELIMITER //
CREATE PROCEDURE cont_nou(cnp_1 varchar(14),nume_1 varchar(30),prenume_1
varchar(30),adresa_1 varchar(50),tel_1 varchar(10),email_1 varchar(50),iban_1 varchar(30),nr_1
varchar(10),angajare date,tip_util varchar(30),parola_1 varchar(30),salar_neg
float(10,5),nr_ore_1 int,tip_functie varchar(30),unit_med varchar(30))

BEGIN
set @id_util=null;
select @id_util:=id from tip_utilizator where denumire_tip=tip_util;
select @id_functie:=id_functie from functie where denumire_functie=tip_functie;
select @id_unit:=id from unitate_medicala where denumire=unit_med;
if (tip_functie='Inspector resurse umane') then
```

```
set @id_dep=1;
else
if (tip_functie='Expert financiar-contabil') then
set @id_dep=2;
else
set @id_dep=3;
end if;
end if;
if (tip_util='Administrator' or tip_util='Super-administrator') then
begin
set @id_functie=null;
set @id_dep=null;
end;
end if;
if (tip_functie='Medic') then      #daca este medic inserez tot mai putin
departamentul/departamentele
insert into utilizator
(cnp,nume,prenume,adresa,nr_telefon,email,iban,nr_contract,data_angajarii,parola,salariu_negoc
iat,nr_ore,id_tip_utilizator,id_departament,id_functie,id_unitate_medicala)
values
(cnp_1,nume_1,prenume_1,adresa_1,tel_1,email_1,iban_1,nr_1,angajare,parola_1,salar_neg,nr_
ore_1,@id_util,@id_dep,@id_functie,null);
end if;
if (tip_functie<>'Medic') then
insert into utilizator
(cnp,nume,prenume,adresa,nr_telefon,email,iban,nr_contract,data_angajarii,parola,salariu_negoc
iat,nr_ore,id_tip_utilizator,id_departament,id_functie,id_unitate_medicala)
values
(cnp_1,nume_1,prenume_1,adresa_1,tel_1,email_1,iban_1,nr_1,angajare,parola_1,salar_neg,nr_
ore_1,@id_util,@id_dep,@id_functie,@id_unit);
end if;

END //
DELIMITER ;
```

#####---------- INSERARE DATE ASISTENT MEDICAL ----------##########

```
DELIMITER //
CREATE PROCEDURE insert_asistent(tip_1 varchar(20),grad_1 varchar(10))
BEGIN
select @id_func:=MAX(id_functie) from functie where denumire_functie='Asistent medical';
select @cnp_asist:=cnp from utilizator where id_functie=@id_func;
insert into asistent_medical(cnp_asistent,id_functie,tip,grad)
```

```
values
(@cnp_asist,@id_func,tip_1,grad_1);
END //
DELIMITER ;
```

###########---------- INSERARE DATE MEDIC ----------###########

```
DELIMITER //
CREATE PROCEDURE insert_medic(grad_1 varchar(15),cod_1 varchar(30),titlu
varchar(30),post_1 varchar(30),procent_1 float,specialitate_1 varchar(100))
begin
select @id_func:=MAX(id_functie) from functie where denumire_functie='Medic';
select @id_spec:=id_specialitate from specialitatea where denumire_specialitate=specialitate_1;
select @cnp_med:=cnp from utilizator where id_functie=@id_func;
insert into
medic(cnp_medic,id_functie,id_specialitate,grad,cod_parafa,titlu_stiintific,post_didactic,procent
_aditional)
values
(@cnp_med,@id_func,@id_spec,grad_1,cod_1,titlu,post_1,procent_1);
end //
DELIMITER ;
```

#####---------- INSERARE LEGATURA DINTRE MEDICI SI UNITATI ----------#####

```
DELIMITER //
CREATE PROCEDURE insert_unitate(nume_unitate varchar(30))
begin
select @id_util:=MAX(id) from utilizator;    #ultimul id introdus in baza de date
select @id_functie:=id_functie from utilizator where id=@id_util;    #id_functiei al ultimei pers
introdusa
if (@id_functie=3) then     #daca utlima pers introdusa este medic
begin
select @cnp_medic:=cnp from utilizator where id=@id_util;    #cnp-ul pers
select @id_unit:=id from unitate_medicala where denumire=nume_unitate;
insert into unitate_medic(cnp_medicul,id_unitate_medicala)
values
(@cnp_medic,@id_unit);
end;
end if;
end //
DELIMITER ;
```

#######---------PRELUARE DATE UTILIZATOR------#####

```
DELIMITER //
CREATE PROCEDURE preluareDate(in idU int)
begin
   select cnp,adresa,nr_telefon,
   email,IBAN,nr_contract,data_angajarii,
  salariu_negociat,nr_ore from utilizator
   where utilizator.id=idU;
end;
// DELIMITER ;
call preluareDate('1');
```

#####-------PRELUARE ORAR IN FUNCTIE DE PROGRAMARI MEDIC--------####

```
DELIMITER //
CREATE PROCEDURE getOrarProgramari(in cnpMedic varchar(14))
BEGIN
select programare.data_programare, programare.ora_programare, programare.durata,
unitate_medicala.denumire
from programare, unitate_medicala
where programare.cnp_medic=cnpMedic and
programare.id_unitate_medicala=unitate_medicala.id
order by programare.data_programare;
END;
// DELIMITER  ;
```

###############----------ADAUGARE CONCEDIU----------########

```
DELIMITER //
CREATE PROCEDURE inserareConcediu(in idUtilizator int,in dataInceput date,in datasfarsit
date)
BEGIN
select id into @existaOrar from orar_generic where id=idUtilizator;
if( @existaOrar is not null ) then
begin
 update orar_generic set data_inceput_concediu=dataInceput where id=idUtilizator;
 update orar_generic set data_sfarsit_concediu=dataSfarsit where id=idUtilizator;
end;
else
begin
insert into orar_generic(id,data_inceput_concediu,data_sfarsit_concediu) values
(idUtilizator,dataInceput,dataSfarsit);
```

```
end;
end if;
end;
// DELIMITER ;
```

#################----------ADAUGA BON FISCAL-------#################

```
DELIMITER //
CREATE PROCEDURE adaugaBon(in cnpM varchar(14), in pret int,in dataBon date)
BEGIN
 select id into @idMedic from utilizator where cnp=cnpM;
 if( @idMedic is not null) then
 insert into bonFiscal(id_medic,pret_servicii,data_bon) values(@idMedic,pret,dataBon);
 end if;
 END ;
 // DELIMITER ;
```

##########----------MODIFICARE DATE UTILIZATOR----------#####

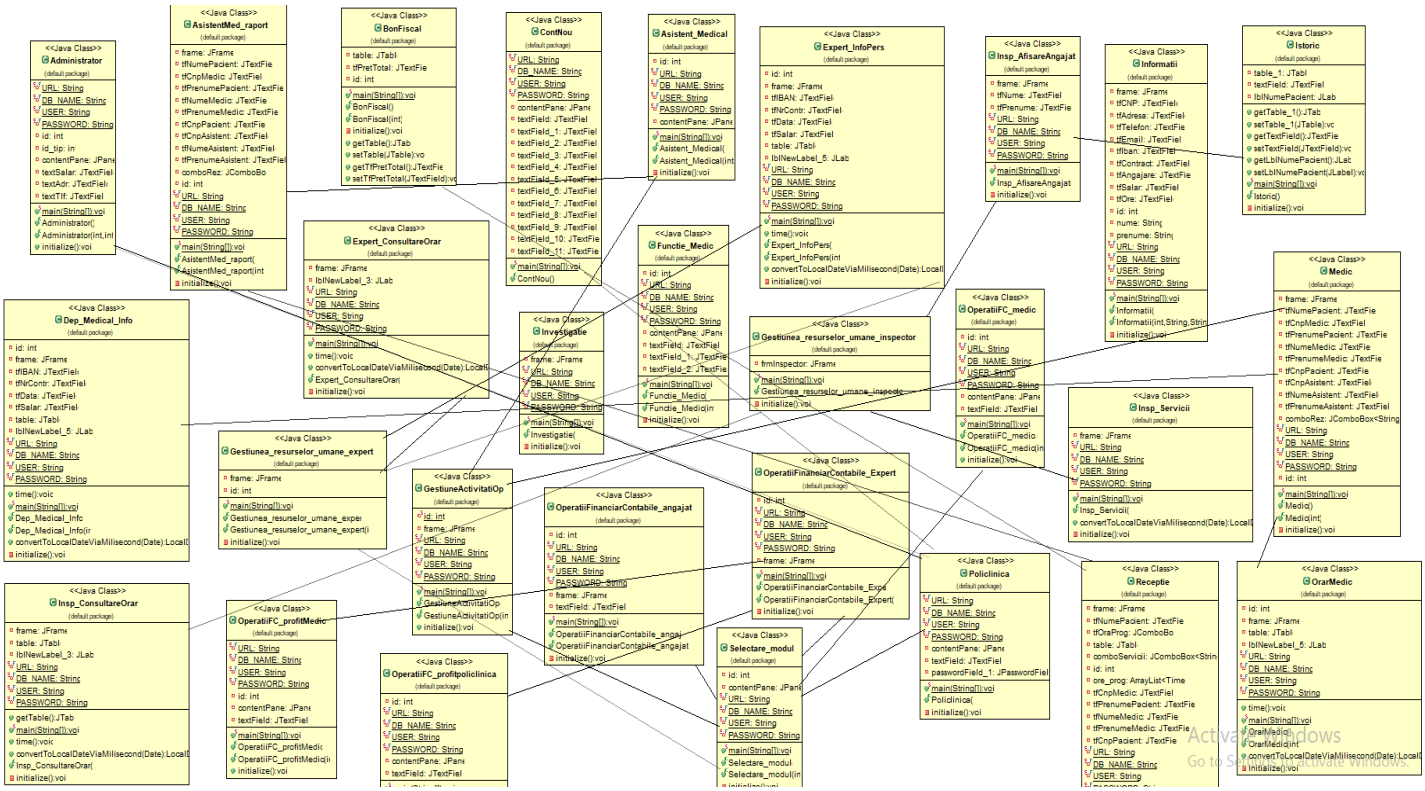
```
DELIMITER //
CREATE PROCEDURE modificareDate(in idUtilizator int,in salariuModificat float(20,5),in
adresaModificata varchar(50), in telefonModificat varchar(10))
BEGIN
select id into @existaUtilizator from utilizator where id=idUtilizator;
if( @existaUtilizator is not null ) then

begin
 update utilizator set salariu_negociat = salariuModificat where id=idUtilizator;
 update utilizator set adresa=adresaModificata where id=idUtilizator;
 update utilizator set nr_telefon=telefonModificat where id=idUtilizator;
end;

end if;
END;
// DELIMITER ;
```

4. Detalii de implementare

4.1. Structura de clase in java. Diagrama UML



Proiectul nostru este impartit in mai multe interfete grafice prin care am demonstrat functionalitatea codului. Pentru a face legatura intre acestea am folosit metoda de transmitere a variabilelor locale pentru fiecare caz specific, in parte. Astfel, oriunde am naviga prin aplicatie, programul va tine cont de datele prin care ne inregistram, deschizandu-se in continuare doar ferestrele speciale pentru fiecare utilizator. De asemenea, orice date pe care le punem direct din aplicatie se vor retine in baza de date.

- Fisierul “Administrator.java” permite unui utilizator de tip administrator /super-administrator vizualizarea unei interfete pentru modificare datelor celorlalti utilizatori.

- Fisierul “Asistent_medical.java” permite alegerea gradului si tipului pentru un asistent medical.

- Fisierul “AsistentMed_raport.java” permite completarea unui raport medical de catre un utilizator de tip asistent medical.

- Fisierul “BonFiscal.java” ofera vizualizarea bonului fiscal, generat dupa o programare.

- Fisierul “ContNou.java” deschide o interfata prin care unui nou utilizator ii este permis sa se inregistreze in baza de date.

- Fisierul “Dep_Medical.java” permite vizualizarea informatiilor utilizatorilor care fac parte din departamentul medical.

- Fisierul “Expert_ConsultareOrar.java” deschide o interfata prin care utilizatorul de tip expert-contabil poate vizualiza la alegere orarul de lucru al oricarui altui utilizator.

- Fisierul “Expert_InfoPers.java” deschide o interfata prin care utilizatorul de tip expert-contabil isi poate vizualiza proprile informatii.

- Fisierul “Functie_Medic.java” permite alegerea datelor suplimentare pentru un utilizator de tip medic.

- Fisierul “Gestiunea_resurselor_umane_expert.java” deschide o interfata interfata prin care utilizatorul de tip expert-contabil poate alege ce isi doreste sa vada dintre informatiile persoanle si consultarea orarului celorlati utilizatori.

- Fisierul “Gestiunea_resurselor_umane_inspector.java” deschide o interfata prin care utilizatorul de tip inspector resurse umane poate alege operatia pe care isi doreste sa o desfasoare in continuare.

- Fisierul “GestiuneActivitatiOp.java” deschide o interfata pentru vizualizarea tipului de utilizatori din departamentul medical.

- Fisierul “Informatii.java” ofera vizualizarea datelor personale ale utilizatorilor de catre inspectorul de resurse.

- Fisierul “Insp_AfisareAngajat.java” ofera inspectorului de resurse alegerea oricarui angajat caruia vrea sa ii vada informatiile.

- Fisierul “Insp_ConsultareOrar.java” ofera inspectorului de resurse umane psoibilitatea consultarii orarului pentru orice angajat.

- Fisierul “Insp_Servicii.java” ofera inspectorului de resurse umane psoibilitatea adaugarii concediilor pentru angajati.

- Fisierul “Investigatie.java” ofera posibilitatea medicului de a adauga investigatiile efectuate la un raport medical.

- Fisierul “Istoric.java” ofera posibilitatea medicului de a vizualiza istoricul pacientului consultat.

- Fisierul “Medic.java” deschide interfata necesara pentru generarea raportului medical al unui pacient de catre un medic.

- Fisierul “OperatiiFC_medic.java” deschide o interfata pentru vizualizarea profitului pentru un utilizator de tip medic.

- Fisierul “OperatiiFC_profitMedic.java” ofera vizualizarea profitului unui medic pentru un utilizator de tip expert contabil.

- Fisierul “OperatiiFC_profitpoliclinica.java” ofera vizualizarea profitului unei policlinici pentru un utilizator de tip expert contabil.

- Fisierul “OperatiiFinanciarContabile_angajat.java” ofera vizualizarea salariului pentru orice angajat.

- Fisierul “OperatiiFinanciarContabile_Expert.java” ofera unui expert contabil posibilitatea sa aleaga intre a vizualiza profitul unui medic sau al unei policlinici.

- Fisierul “OrarMedic.java” ofera posibilitatea vizualizarii orarului unui medic care este facut in functie de programari.

- Fisierul “Policlinica.java” ofera posibilitatea inregistrarii utilizatorilor in aplicatie.

- Fisierul “Receptie.java” deschide o interfata pentru a adauga o noua programare de catre receptionist.

- Fisierul “Selectare_modul.java” ofera posibilitatea alegerii modulului dorit de catre utilizator.

4.2. Manual de utilizare/instalare

Pentru utilizarea aplicatiei este nevoie sa utilizam programul Eclipse deoarece acesta faciliteaza conectarea la baza de date a aplicatiei. Pentru acest lucru trebuie sa generam o legatura intre MySql si Eclipse.

La lansarea in executie a aplicatiei va rula interfata de “Log in” care permite inregistrarea unui utilizator, iar dupa o autentificare reusita va aparea o pagina principal in care utilizatorul isi poate alege modulul dorit, fiecare utilizator, in functie de tipul acestuia, avand anumite facilitati. De altfel, pentru un utilizator de tip administrator/super-administrator se va deschide direct fereastra de modificare a datelor utilizatorilor. In functie de tipul de utilizator cu care suntem inregistrati avem diferite functionalitati asupra programului. De exemplu, daca suntem inregistrati cu un utilizator de tip expert contabil putem naviga prin toate functiile puse la dispozitie de modulul de “Operatii financiar contabile”.

Pentru a avea utilizatori in baza de date este necesar sa ii generam prin interfata de “ContNou”.

4.3. Elemente de securizare a aplicatiei

Aplicatia prezinta un nivel de securitate in momentul conectarii la baza de date dar si in momentul logarii propriu zise.

In functie de modul de conectare ales (administrator/super-administrator, inspector resurse umane, expert financiar contabil, receptioner, medic si asistent medical) sunt disponibile urmatoarele operatii:

- Administrator/super-administrator: poate accesa baza de date pentru a face anumite modificari asupra datelor personale ale angajatilor.

- Inspector resurse umane: are posibilitatea de vizualizare a tuturor datelor angajatiilor si orarele lor de functionare, precum si posibilitatea de a genera concedii. De asemenea, isi poate vizualiza si salsariul in modulul de operatii financiar contabile.

- Expert financiar contabil: are posibilitatea de vizualizare a profitului medicilor sau al policlinicii, precum poate vizualiza si orarele de functionare ale angajatilor sau datele lui personale.

- Receptioner: are posibilitatea de vizualizare a informatiilor lui personale, precum poate si adauga noi programari in cadrul policlinicii.

- Medic: are posibilitatea de vizualizare a informatiilor lui personale, precum si de generare a unor rapoarte medicale si poate vizualiza si date despre pacienti, precum un istoric medical. De asemenea, acesta isi poate consulta profitul.

- Asistent medical: are posibilitatea de vizualizare a informatiilor lui personale, precum poate si adauga un rezultat al analizelor in raportul medical. De asemenea, isi poate consulta salariul.

5. Concluzii. Limitari si dezvoltari ulterioare

Toate cerintele au fost respecate si indeplinite, acestea ducand la un mediu placut prin care utilizatorii pot interactiona cu o policlinica moderna si la un mod convenabil de gestiune a datelor de catre administratori.

Ca dezvoltare ulterioara mentionam adaugarea unei functionalitati care sa faciliteze procedura de concedieri. De asemenea, pe baza acestei functionalitati vom putea genera o interfata care sa arate toti angajatii care nu mai lucreaza in mod activ in sistemul de policlinici, precum si o interfata prin care putem genera si alte concedieri.