Universitatea din București
Facultatea de Matematică și Informatică

GESTIONAREA ACTIVITĂȚII UNUI SPITAL

Profesor Coordonator

Vasile Silviu Laurențiu

Ceaușu Corina-Georgiana

BUCUREȘTI 2020

Cuprins

Capitolul I

Capitolul II

Capitolul I

1. Prezentarea modelului (din lumea reală) și a regulilor acestuia

Spitalul reprezintă o instituție foarte importantă, fiind dotat cu echipamente pentru tratamentul atât medical, cât și chirurgical al pacienților.

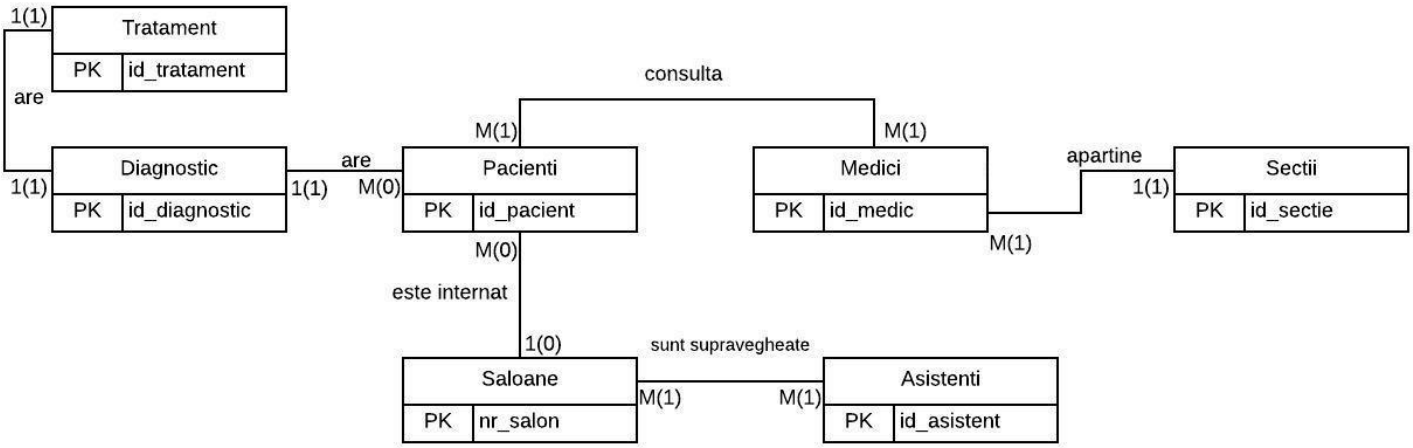
Pentru gestionarea eficientă a unui spital, acesta este împărțit pe **secții**. Secțiile reprezintă jumătăți dintr-un etaj, dotate cu **saloane** pentru pacienți.

Într-o secție pot lucra unul sau mai mulți **medici** iar un salon este gestionat de cel puțin un **asistent**. Aici pot fi internați zero sau mai mulți **pacienți**.

Fiecare pacient poate avea un singur **diagnostic** care deține unul sau mai multe **tratamente**.

2. a) Diagrama entitate-relație

Diagrama entitate-relație (E-R) reprezintă un model neformalizat pentru reprezentarea unor fenomene din lumea reală.



b) Descrierea entităților, atributelor, cheilor, relațiilor și a cardinalităților

Pentru gestionarea mai eficientă a activității, spitalul prezentat în acest proiect este împărțit în secții. Fiecare dintre acestea deține un ID unic („id_sectie”), fiind cheie primară pentru tabelul „Secții”. De asemenea, secțiile au și o denumire pentru a putea fi recunoscute mai ușor.

Un medic lucrează într-o singură secție, nu există medici cărora nu le este asociată o secție, și, prin urmare, între cele două tabele există o relație de tipul „one to many”, fiind legate prin prezența cheii externe „id_sectie” în tabelul „Medici”.

Fiecare medic este identificat de un id („id_medic”) care reprezintă cheia primară și are prezente în tabel atribute care reprezintă informații personale: nume, prenume, vârstă, email, număr de telefon.

Medicii reprezintă o entitate indispensabilă, ei fiind cei care se ocupă direct de pacienți, în tabelul „Pacienți” regăsindu-se cheia externă „id_medic”. Un pacient poate fi consultat de mai mulți medici iar un medic poate trata mai mulți pacienți, deci între cele doua tabele există o relație de tipul „many to many”.

Pacientul este identificat de un id care este cheie primară pentru acest tabel. Ca și la tabelul medicilor, pacienții au afișate în baza de date informații personale: nume, prenume, vârstă, gen, număr de telefon, adresă.

După consultul medicului, pacientului îi este atribuit un singur diagnostic, însă un diagnostic poate fi atribuit mai multor pacienți. Acest lucru este subliniat prin prezența câmpului id_diagnostic în tabelul pacienților și, respectiv, printr-o relație „zero to many” între entitățile „Pacienti” și „Diagnostice”

Tabelul diagnosticelor mai este prevăzut cu un câmp pentru denumirea fiecărui diagnostic și cu un tratament reprezentat prin cheia externă „id_tratament” care este definită drept cheie primară a tabelului „Tratamente”.

Un tratament este specific unui singur diagnostic, însă un diagnostic poate avea unul sau mai multe tratamente, rezultând o relație de tipul „one to many” între cele două tabele.

Pentru a putea fi recunoscute, tratamentele dispun de o denumire, iar pentru a putea fi dozate corespunzător, ele dispun de un atribut numit „cantitate”.

După ce au fost făcute investigații de către medici și stabilit diagnosticul, pacientul poate fi internat într-un salon din spital. Aici se pot interna zero sau mai mulți pacienți, deci relația dintre cele doua tabele este de tipul „zero to many”, în tabelul pacienților existând numarul salonului în care aceștia se află (atributul „nr_salon” este cheie primară pentru tabelul „Saloane” și cheie externă pentru tabelul „Pacienți”). De asemenea, saloanele pot fi pentru femei sau pentru bărbați (atributul „tip”) și au un anumit număr de locuri disponibile (atributul „nr_locuri”).

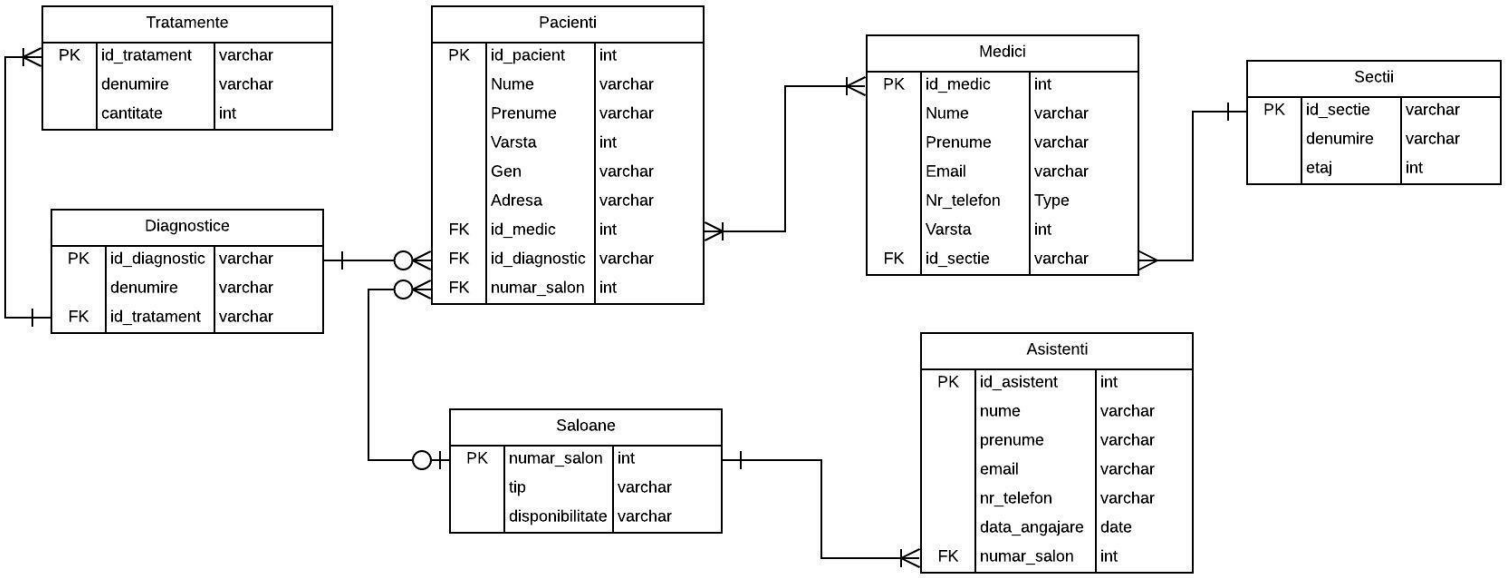
Saloanele reprezintă camere dotate cu paturi în care pacienții vor fi tratați și ținuți sub observație. Un salon este supravegheat de către unul sau mai mulți asistenți, iar un asistent poate supraveghea un singur salon, existând o relație de tipul „one to many”.

Fiind angajați ai unei instituții, asistenții dispun în baza de date de informații personale și de contact: nume, prenume, email, număr de telefon, dată angajare.

Toate tabelele se află în strânsă legătură pentru buna gestionare a activității spitalului.

3. a) Diagrama conceptuală

Schema conceptuală a bazei de date corespunde unei reprezentări unice (pentru toţi utilizatorii) şi abstracte a datelor, descriind ce date sunt stocate în baza de date şi care sunt asocierile dintre acestea.



b) Descrierea constrângerilor de integritate

- **Tabelul „Secții”**

  În acest tabel, denumirea secției nu poate să lipsească, deci pentru ea vom avea o constrângere de tipul „not null”.

- **Tabelul „Medici”**

  Numele și prenumele unui medic, cât și datele de contact (numărul de telefon și emailul) nu pot fi nule – constrângere „not null”. În plus, datele de contact trebuie să fie unice fiecărui doctor – constrângere de tipul „unique”.

- **Tabelul „Pacienți”**

  Similar tabelului medicilor, numele, prenumele și datele de contact ale unui pacient nu pot fi nule, rezultând constrângeri de tipul „not null” iar constrângerea „unique” este utilizată pentru numărul de telefon și pentru email.

- **Tabelul „Diagnostice”**

  Denumirea diagnosticului este esențială, deci nu trebuie să lipsească. Vom folosi aici o constrângere de tipul „not null”.

- **Tabelul „Tratamente”**

  Pentru atributul „denumire” vom seta o constrângere „not null”.

- **Tabelul „Saloane”**

  Pentru că saloanele sunt împărțite în 2 categorii: saloane pentru femei și saloane pentru bărbați, tipul salonului trebuie specificat, deci pentru atributul „tip” vom adăuga constrângerea „not null”.

- **Tabelul „Asistenți”**

  Similar medicilor, fiind angajați ai instituției, asistenților nu trebuie să le lipsească numele, prenumele și datele de contact (număr de telefon și email) – constrângere „not null”. Datele de contact sunt unice fiecărui asistent – constrângere „unique”.

c) Schemele relaționale

Un medic nu poate fi angajat fără să lucreze într-o secție. Dacă o secție este eliminată din spital, vor fi eliminați și medicii din baza de date care aparțin secției respective, folosind comanda „`ON DELETE CASCADE`”.

În cazul în care este șters un medic, atributul asociat din tabelul pacienților va deveni nul, folosind comanda „`ON DELETE SET NULL`”. Același lucru se va întâmpla și legat de cheile externe „id_diagnostic” și „numar_salon” în cazul în care acestea vor fi șterse din tabelele din care provin.

Dacă se va elimina un tratament, atributul său din tabelul diagnosticelor va deveni nul, prin comanda „`ON DELETE SET NULL`”.

Un asistent trebuie să supravegheze un salon. În eventualitatea eliminării unui salon, se vor șterge din baza de date asistenții asociați acestuia prin comanda „`ON DELETE CASCADE`”.

Capitolul II

1. Crearea tabelelor

Pentru crearea bazei de date, am folosit „Oracle SQL Developer”, iar codul pentru crearea fiecarui tabel este după cum urmează:

```
CREATE TABLE Saloane (
  numar_salon int,
  tip varchar(10) not null,
  nr_locuri int,
  PRIMARY KEY (numar_salon)
);

CREATE TABLE Asistenti (
  id_asistent int,
  nume varchar(30) not null,
  prenume varchar(30) not null,
  email varchar(50) not null unique,
  nr_telefon varchar(10) not null unique,
  data_angajare date,
  numar_salon int,
  PRIMARY KEY (id_asistent),
  FOREIGN KEY (numar_salon) references Saloane(numar_salon)
  ON DELETE CASCADE
);

CREATE TABLE  Sectii  (
   id_sectie varchar(20),
   denumire varchar(100) not null,
   etaj int,
  PRIMARY KEY (id_sectie)
);

CREATE TABLE Medici (
   id_medic int,
   Nume varchar(30) not null,
   Prenume varchar(30) not null,
   email varchar(50) not null unique,
   nr_telefon varchar(10) not null unique,
   Varsta int,
   id_sectie varchar(20),
   PRIMARY KEY (id_medic),
   FOREIGN KEY (id_sectie) references Sectii(id_sectie)
   ON DELETE CASCADE
);

CREATE TABLE Tratamente (
  id_tratament varchar(10),
  denumire varchar(100) not null,
  cantitate int,
  PRIMARY KEY (id_tratament)
);
```

```
CREATE TABLE Diagnostice (
  id_diagnostic varchar(10),
  denumire varchar(200) not null,
  id_tratament varchar(10),
  PRIMARY KEY (id_diagnostic),
  FOREIGN KEY (id_tratament) references Tratamente(id_tratament)
  ON DELETE SET NULL
);

CREATE TABLE Pacienti (
  id_pacient int,
  Nume varchar(30) not null,
  Prenume varchar(30) not null,
  Varsta int,
  Gen varchar(10),
  Nr_telefon varchar(10) not null unique,
  Adresa varchar(40),
  id_medic int,
  id_diagnostic varchar(10),
  numar_salon int,
  PRIMARY KEY (id_pacient),
  FOREIGN KEY (id_medic) references Medici(id_medic),
  FOREIGN KEY (id_diagnostic) references Diagnostice(id_diagnostic),
  FOREIGN KEY (numar_salon) references Saloane(numar_salon)
  ON DELETE SET NULL
);
```

2. ***Introducere date***

   În fiecare tabel se vor introduce date după modelul următor:

- **Tabelul „Secții”**

```
INSERT INTO sectii
 VALUES
('CH-GEN','Chirurgie Generala',1);
```

- **Tabelul „Medici”**

```
INSERT INTO medici
 VALUES
(1,'Badea','Nicolae','badeanicolae@gmail.com','0742765401',42,'CH-GEN');
```

- **Tabelul „Pacienti”**

```
INSERT INTO pacienti
 VALUES
(1, 'Ionescu','Mihai',62,'masculin','0726671928','Bulevardul Mihail
Kogalniceanu, nr. 15',1,'HER-AB',3);
```

- **Tabelul „Diagnostice”**

```
INSERT INTO diagnostic
 VALUES
('HER-AB','Herniile peretelui abdominal','OP');
```

- **Tabelul „Tratamente”**

```
INSERT INTO tratamente
 VALUES
('OP','Operatie',NULL);
```