Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli

DOKUMENTACIJA UZ PROJEKTNI ZADATAK

''BOLNICA''

TIM 1

Neven Davidović

Marija Ilić

Darjan Juranović

Noel Modrušan

Tin Pritišanac

Smjer : Informatika

Kolegij : Baze podataka I

Mentor : doc. Dr. sc. Goran Oreški

Pula, svibanj, 2022. godina

Sadržaj

1. UVOD 3
2. OPIS POSLOVNOG PROCESA 3
3. ENTITY RELATIONSHIP (ER) DIJAGRAM 4
   3.1. OPIS ER DIJAGRAMA 5
4. RELACIJSKI MODEL (SHEME) 6
5. EER DIJAGRAM (MYSQL) 7
6. TABLICE 8
   6.1. TABLICA odjel 8
   6.2. TABLICA doktor 9
   6.3. TABLICA sos_kontakt 10
   6.4. TABLICA pacijent 11
   6.5. TABLICA medicinske_sestre 11
   6.6. TABLICA dijagnoza 12
   6.7. TABLICA soba 12
   6.8. TABLICA lijek 12
   6.9. TABLICA stanje_lijekova 13
   6.10. TABLICA terapija 13
   6.11. TABLICA prijem 14
   6.12. TABLICA posjeta 14
   6.13. TABLICA oprema 15
   6.14. TABLICA stanje_opreme 15
7. UPITI 16
   7.1. UPIT 1 16
   7.2. UPIT 2 18
   7.3. UPIT 3 20
   7.4. UPIT 4 21
   7.5. UPIT 5 22
   7.6. UPIT 6 24
   7.7. UPIT 7 25
   7.8. UPIT 8 27
   7.9. UPIT 9 28
   7.10. UPIT 10 30
   7.11. UPIT 11 31

7.12. UPIT 12 32
7.13. UPIT 13 33
7.14. UPIT 14 36
7.15. UPIT 15 37
7.16. UPIT 16 37
7.17. UPIT 17 38
8. ZAKLJUČAK 40

1. UVOD

Ovaj projekt se od svoje početne verzije dosta promijenio, naravno na bolje. Kako su nam se znanje i vještine nadograđivale tokom semestra uvijek je postojala preinaka koja je poboljšala samu strukturu i funkciju pojedinih elemenata na izvedbu i rad same baze podataka. Sam proces izrade baze podataka je obavljen preko GitHuba zbog nemogućnosti okupljanja svih članova, što zbog posla što zbog drugih obaveza. Izgradili smo bazu podataka imena ''Bolnica'' te smo nastojali što detaljnije opisati sve poslovne procese i konceptualni model koji je prikazan preko ER dijagrama. Naravno, naša baza je samo djelić one prave, koju smo morali reducirati i izričito specificirati opis poslovanja za potrebe našeg projekta. Krajnji cilj je opisati što se događa kada pacijent dođe u bolnicu, od početnih koraka, prijema i pregleda, do terapije i smještaja u sobu. Tu su također dodatni skupovi entiteta koji su detaljno opisani u konceptualnom modelu. Također, svaka tablica, unosi podataka, upiti, ograničenja i ostalo su detaljno opisani u narednih nekoliko stranica. EER dijagram, odnosno logička shema baze podataka je generirana u MySQL Workbench-u preko Reverse Enginering opcije. Sva imena, prezimena, adrese, OIB-ovi, brojevi mobitel su generirani nasumično pomoću online alata, dok su nazivi opreme i odjela preneseni iz stvarne bolnice.

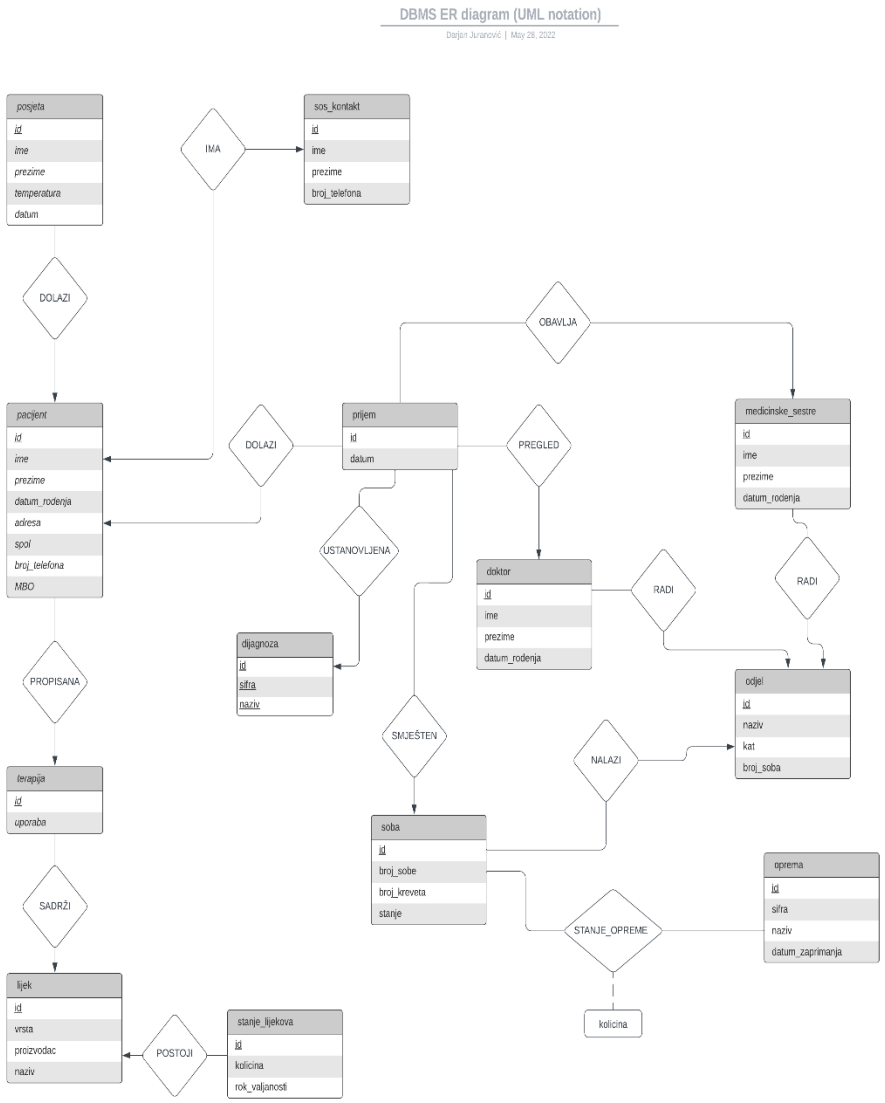
2. OPIS POSLOVNOG PROCESA

U bolnici se prati prijem pacijenta. Medicinska sestra obavlja prijem pacijenta. Za svakog pacijenta se prati ime, prezime, datum_rodenja, adresa, spol, broj_telefona, MBO. Za svaki prijem pacijenta pratimo datum_prijema.

Za pacijenta postoji SOS kontakt. Za svakog sos kontakta pratimo ime, prezime, broj_telefona. Doktor obavlja pregled pacijenta, dok se za svakog doktora prati ime, prezime, datum_rodenja. Nakon pregleda uspostavljena je dijagnoza za koju se prati naziv i sifra. Nakon toga je terapija propisana pacijentu, a za terapiju se prati uporaba. Za propisanu terapiju se izdaje lijek, te se za taj lijek prati vrsta, proizvodac i naziv. Kod stanja lijekova pratimo kolicinu i rok_valjanosti. Pacijent je smješten u sobu na daljnje liječenje te se za sobu prati broj_sobe, broj_kreveta, i stanje. Posjeta dolazi pacijentu te se za nju prati ime, prezime, temperatura, datum, vrijeme_dolaska, vrijeme_odlaska. Sobe u kojima su smješteni pacijenti se nalaze po odjelima. Različita oprema je raspoređena po sobama, a za samu opremu se prati sifra, naziv, datum_zaprimanja. Također postoji stanje opreme koje je povezano sa sobom i opremom te se za to stanje prati količina. Doktori i medicinske sestre rade na različitim odjelima u bolnici. Za medicinsku sestru se prati ime, prezime, datum_rodenja.

3. ENTITY RELATIONSHIP (ER) DIJAGRAM

Sljedeći ER dijagram detaljno i pregledno opisuje sve skupove - entiteta (njihove atribute) kao i skupove - veza između njih. Kardinalnost mapiranja (strelice) predstavljaju koliko drugih entiteta može biti povezano s entitetom preko određenog skupa – veza.



Slika 1/ ER dijagram za našu bazu

3.1. OPIS ER DIJAGRAMA

- **DOKTOR** radi na **ODJELU** (doktor radi na jednom odjelu, dok na tom odjelu može raditi više doktora) – kardinalnost **one** to **many**
- **PACIJENT** ima **SOS_KONTAKT** (pacijent ima jedan sos_kontakt, dok je taj sos_kontakt vezan samo za tog pacijenta) **one** to **one.** (ovo je navedeno ovako radi jednostavnijeg daljnjeg rada)
- **MEDICINSKA_SESTRA** radi na **ODJELU** (medicinska sestra radi na jednom odijelu, dok na tom odjelu može raditi više medicinskih sestara) **one** to **many**
- **SOBA** se nalazi u **ODJELU** (soba je u jednom odjelu, dok odjel može imati više različitih soba) **one** to **many**
- **TERAPIJA** je propisana **PACIJENTU** (pacijent prima jednu terapiju, dok se ta terapija može prepisati većem broju pacijenata) **one** to **many.** (pošto je pacijentu propisana jedna terapija, dok god je ona na snazi lijekovi se neće mijenjati za tu terapiju i zbog toga je kardinalnost mapiranja **one** to **many**)
- **LIJEK** se izdaje na propisanu **TERAPIJU** (lijek se izdaje za određenu terapiju, dok više terapija može imati isti lijek) **one** to **many**
- Za **LIJEK** postoji **STANJE_LIJEKOVA** (za jedan lijek postoji više stanja, pošto se ovisno o terapiji mijenja količina lijekova) **one** to **many**
- **PACIJENT** primljen u **BOLNICU** (pacijent je primljen u bolnicu, dok u tu istu bolnicu može biti primljeno više pacijenata) **one** to **many**
- **MEDICINSKA_SESTRA** obavlja **PRIJEM** (prijem je obavljen od strane medicinske sestre, dok ta ista sestra može obaviti više prijema) **one** to **many**
- **DOKTOR** nakon prijema pregledava **PACIJENTA** (pacijent je pregledan od jednog doktora, dok isti taj doktor može pregledati više pacijenata) **one** to **many**
- Doktor je uspostavio **DIJAGNOZU** nakon **PRIJEMA** (dijagnoza je uspostavljena od doktora, dok isti taj doktor može uspostaviti više dijagnoza) **one** to **many**
- Pacijent je smješten u **SOBU** nakon **PRIJEMA** (pacijent je smješten u sobu, dok u toj sobi može biti više pacijenata) **one** to **many**
- **POSJETA** dolazi **PACIJENTU** (jedan posjetitelj posjećuje pacijenta, dok taj pacijent može imati više posjetitelja) **one** to **many**
- **OPREMA** je raspoređena po **SOBI** (više soba može imati više različite opreme) **many** to **many**
- Dodatni entitet **STAVKA_OPREME** gdje se primarni ključevi **SOBE** i **OPREME** referenciraju kao strani ključ za STAVKU_OPREME

4. RELACIJSKI MODEL (SHEME)

odjel (id, naziv, kat, broj_soba)

doktor (id, ime, prezime, datum_rodenja, id_odjel)

sos_kontakt (id, ime, prezime, broj_telefona)

pacijent (id, ime, prezime, datum_rodenja, adresa, spol, broj_telefona, MBO, id_sos_kontakt)

medicinske_sestre (id, ime, prezime, datum_rodenja, id_odjel)

dijagnoza (id, naziv, sifra)

soba (id, broj_sobe, broj_kreveta, stanje, id_odjel)

lijek (id, vrsta, proizvodac, naziv)

stanje_lijekova (id, id_lijek, kolicina, rok_valjanosti)

terapija (id, id_pacijent, id_lijek, uporaba)

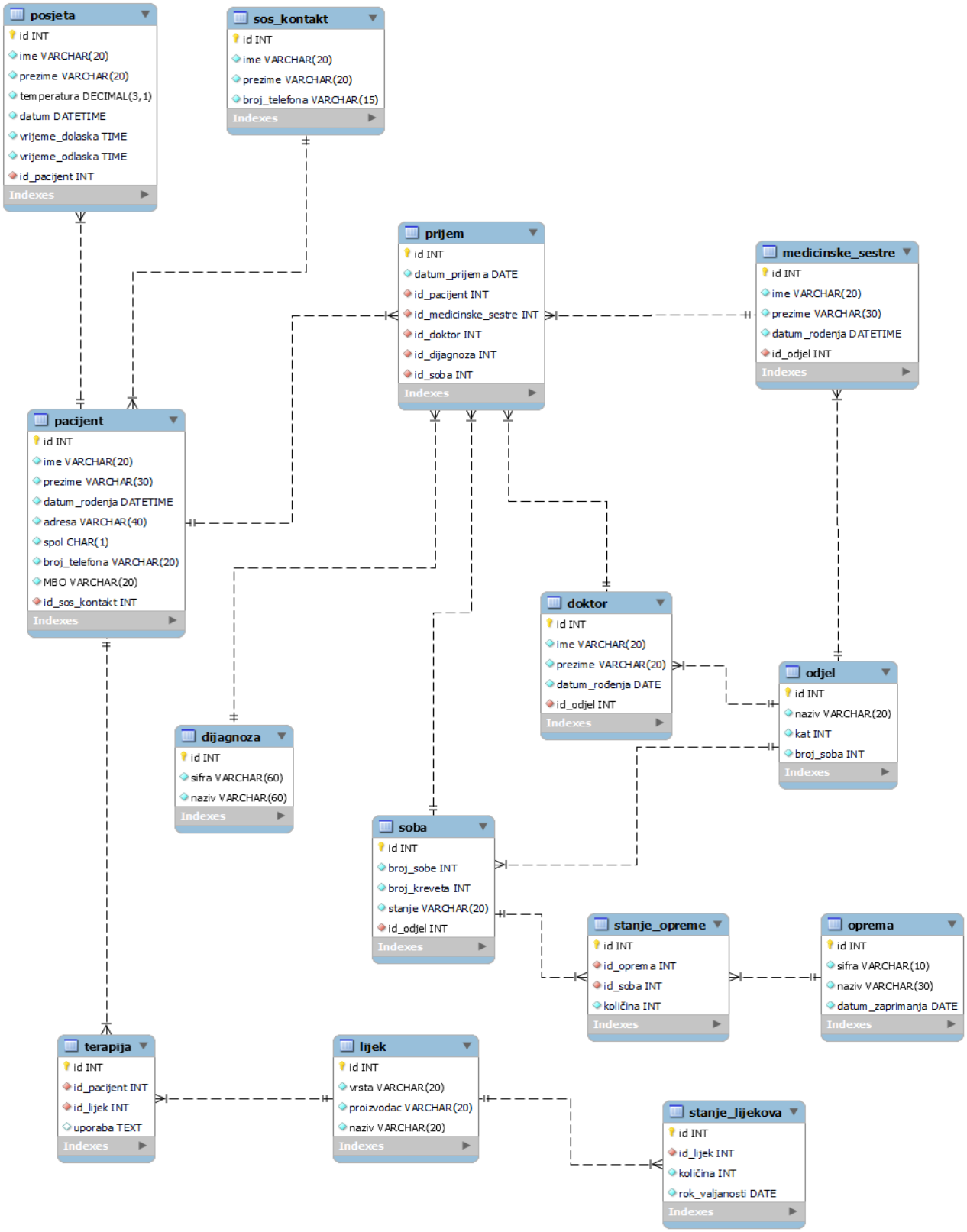
prijem (id, datum_prijema, id_pacijent, id_medicinske_sestre, id_doktor, id_dijagnoza, id_soba)

posjeta (id, ime, prezime, temperatura, datum, vrijeme_dolaska, vrijeme_odlaska, id_pacijent)

oprema (id, sifra, naziv, datum_zaprimanja)

stanje_opreme (id, id_oprema, id_soba, kolicina)

5. EER DIJAGRAM (MYSQL)



Slika 2/EER dijagram, napravljen u MySQL Workbenchu

6. TABLICE

6.1. TABLICA odjel

Tablica **odjel** služi evidenciji postojećih odjela u bolnici. Sadrži atribute: **id**, **naziv**, **kat** i **broj_soba**. Atribut **id** je PRIMARY KEY tablice tipa INTEGER jer se u njega unosi brojčana vrijednost. Služi kako bi smo mogli jedinstveno označiti svaki odjel u bolnici. Iako atribut **id** mora biti jedinstven kako bi opravdao svoju svrhu, nije potrebno staviti ograničenje UNIQUE jer je PRIMARY KEY jedinstven sam po sebi. Atribut **naziv** je tipa VARCHAR koji nam omogućuje dinamičku alokaciju memorije. Odabrala sam tip VARCHAR s obzirom da ne znam unaprijed koliko će znakova pojedini naziv odjela sadržavati. Maksimalna duljina naziva definirana je u zagradi odmah nakon tipa podatka, a iznosi 20 znakova. Unutar atributa **naziv** postavljeno je ograničenje NOT NULL koje ne dopušta unos podataka u bazu podataka bez ovog podatka. Također je postavljeno ograničenje UNIQUE koje osigurava da je svaki naziv odjela u tablici jedinstven. Atribut **kat** je tipa INTEGER, a označava na kojem se katu pojedini odjel u bolnici nalazi. Unutar **kat** atributa također je postavljeno ograničenje NOT NULL kako bi osigurali unos ovog podatka prilikom unošenja podataka unutar baze podataka. Posljednji atribut **broj_soba** označava koliko se soba za smještanje pacijenata nalazi unutar pojedinog odjela, a tipa je INTEGER. Također je obavezan pri unošenju podataka u bazu podataka pa sadrži ograničenje NOT NULL.

```
CREATE TABLE odjel (
id INTEGER PRIMARY KEY,
naziv VARCHAR (20) NOT NULL UNIQUE,
kat INTEGER NOT NULL,
broj_soba INTEGER NOT NULL
);
```

6.2. TABLICA doktor

Tablica **doktor** služi evidenciji zaposlenih doktora u bolnici. Sadrži atribute: **id**, **ime**, **prezime**, **datum_rodenja** i **id_odjel**. Atribut **id** je tipa INTEGER jer u njega unosimo brojčanu vrijednost koja jedinstveno definira svakog pojedinog doktora, što ga ujedno čini i odličnim PRIMARY KEY atributom. Atributi **ime** i **prezime** označavaju ime i prezime svakog pojedinog doktora. Tip podatka VARCHAR koji sam odabrala za ova dva atributa služi dinamičkoj alokaciji memorije, a maksimalna duljina unosa iznosi 20 znakova i definirana je u zagradi odmah nakon tipa podatka. Ograničenje NOT NULL osigurava obavezan unos ova dva podatka prilikom unošenja podataka unutar baze podataka. Atribut **datum_rodenja** je tipa DATE, a označava datum rođenja svakog pojedinog doktora. Ovaj tip podatka omogućuje nam lakši daljnji rad sa datumima prilikom pisanja upita. Također posjeduje ograničenje NOT NULL koje nam garantira obavezan unos ovog podatka u bazu podataka. Atribut **id_odjel** je tipa INTEGER, a označava odjel na kojem je pojedini doktor zaposlen u bolnici. Atribut **id_odjel** je FOREIGN KEY unutar tablice **doktor**, a povezuje tablicu **doktor** sa tablicom **odjel** preko atributa **id** nakon REFERENCES dijela unutar definicije stranog ključa.

```
CREATE TABLE doktor (
id INTEGER PRIMARY KEY,
ime VARCHAR (20) NOT NULL,
prezime VARCHAR (20) NOT NULL,
datum_rodenja DATE NOT NULL,
id_odjel INTEGER NOT NULL,
FOREIGN KEY (id_odjel) REFERENCES odjel (id)
);
```

6.3. TABLICA sos_kontakt

Tablica **sos_kontakt** služi evidenciji SOS kontakata pacijenata. Sadrži atribute **id**, **ime**, **prezime** i **broj_telefona**. Atribut **id** je tipa podatka INTEGER, a ujedno je i PRIMARY KEY jer označava specifičnu brojčanu vrijednost svakog pojedinog SOS kontakta. Atributi **ime** i **prezime** su tipa podatka VARCHAR. Za ova dva atributa odabrala sam tip podatka koji koristi dinamičku alokaciju memorije s obzirom da ne znam unaprijed koja je duljina podataka koji će se unositi unutar njih. Maksimalna duljina unosa unutar atributa **ime** i **prezime** definirana je u zagradi odmah nakon tipa podatka. Ograničenje NOT NULL unutar ova dva atributa osigurava nam obavezan unos ovih podataka prilikom unosa podataka u bazu podataka. Atribut **broj_telefona** tipa je VARCHAR, a označava broj telefona svakog pojedinog SOS kontakta. Odabrala sam VARCHAR tip podatka za ovaj atribut kako bi omogućila dinamičku alokaciju memorije, s obzirom da nisu svi brojevi telefona jednake duljine. Maksimalnu diljinu sam definirala u zagrodi nakon tipa, a iznosi 15 znakova. Drugi razlog iz kojeg sam odabrala VARCHAR tip podatka za ovaj atribut je taj što sam na ovaj način omogućila unos nule na prvo mjesto. Ograničenja koja su stavljena na atribut **broj_telefona** su NOT NULL ograničenjne koje nam osigurava obavezan unos ovog podatka prilikom unošenja podataka u bazu podataka i UNIQUE ograničenje s obzirom da ne postoje dva jednaka broja telefona.

```
CREATE TABLE sos_kontakt (
id INTEGER PRIMARY KEY,
ime VARCHAR (20) NOT NULL,
prezime VARCHAR (20) NOT NULL,
broj_telefona VARCHAR (15) NOT NULL UNIQUE
);
```

6.4. TABLICA pacijent

Sastoji se od atributa id, ime, prezime, datum_rodenja, adresa, spol, broj_telefona, MBO, id_sos_kontakt. Atribut id nam je primarni ključ tipa integer. Atributi ime i prezime su nam tipa varchar sa ograničenjem znakova od 20 za atribut ime i 30 za atribut prezime. Atribut datum_rodenja nam je tipa datetime. Za adresu smo također koristili varchar sa ograničenjem znakova od 40. Atribut spol je tipa char sa ograničenjem 1. Atributi broj_telefona i MBO su tipa varchar sa ograničenjem znakova od 20, na njih je također stavljeno unique ograničenje što znači da su oni jedinstveni. Atribut id_sos_kontakt je tipa integer i on je strani ključ koji povezuje pacijenta s tablicom sos_kontakt. Na atributima ime, prezime, datum_rodenja, adresa, spol, broj_telefona, MBO, i id_sos_kontakt koristio sam ograničenje not null, što znači da je kod tih atributa obavezan upis podataka.

```
CREATE TABLE pacijent(
id INTEGER PRIMARY KEY,
ime VARCHAR(20) NOT NULL,
prezime VARCHAR(30) NOT NULL,
datum_rodenja DATETIME NOT NULL,
adresa VARCHAR (40) NOT NULL,
spol CHAR (1) NOT NULL,
broj_telefona VARCHAR (20) NOT NULL UNIQUE,
MBO VARCHAR (20) NOT NULL UNIQUE,
id_sos_kontakt INTEGER NOT NULL,
FOREIGN KEY (id_sos_kontakt) REFERENCES sos_kontakt (id)
);
```

6.5. TABLICA medicinske_sestre

Sastoji se od atributa id, ime, prezime, datum_rodenja, id_odjel. Atribut id nam je primarni ključ tipa integer. Atributi ime i prezime su nam tipa varchar sa ograničenjem znakova od 20 za atribut ime i 30 za atribut prezime. Atribut datum_rodenja nam je tipa datetime. Atribut id_odjel je tipa integer i on je strani ključ koji povezuje medicinske_sestre s tablicom odjel. Na atributima ime, prezime, datum_rodenja i id_odjel je ograničenje not null, što znači da je kod tih atributa obavezan upis podataka.

```
CREATE TABLE medicinske_sestre(
id INTEGER PRIMARY KEY,
ime VARCHAR (20) NOT NULL,
prezime VARCHAR (30) NOT NULL,
datum_rodenja DATETIME NOT NULL,
id_odjel INTEGER NOT NULL,
FOREIGN KEY (id_odjel) REFERENCES odjel (id)
);
```

6.6. TABLICA dijagnoza

Sastoji se od atributa id, naziv i sifra. Atribut id nam je primarni ključ tipa integer. Atributi naziv i sifra su tipa varchar sa ograničenjem znakova 60 i stavljena su ograničenja not null i unique, što znači da je kod tih atributa obavezan unos i podataka i da su oni jedinstveni

```
CREATE TABLE dijagnoza(
id INTEGER PRIMARY KEY,
naziv VARCHAR (60) NOT NULL UNIQUE,
sifra VARCHAR (60) NOT NULL UNIQUE
);
```

6.7. TABLICA soba

Tablica soba sadrži: integer id koji je primarni ključ u rasponu od 700-800, broj_sobe i broj_kreveta koji su integeri i ne smiju biti NULL, stanje koje je tipa varchar s ograničenjem do 20 znakova i opisuje trenutno stanje sobe (je li slobodna, popunjena ili se ne koristi itd.), te id_odjel koji je integer i strani ključ koji povezuje sobu sa relacijeom odjel.

```
CREATE TABLE soba (
id INTEGER PRIMARY KEY,
broj_sobe INTEGER NOT NULL,
broj_kreveta INTEGER NOT NULL,
stanje VARCHAR (20) NOT NULL,
id_odjel INTEGER NOT NULL,
FOREIGN KEY (id_odjel) REFERENCES odjel (id)
);
```

6.8. TABLICA lijek

Tablica lijek služi za pohranu informacija o svakom lijeku u bolnici.
Sadrži atribute: id tipa integer koji služi kao primarni ključ i u rasponu je od 800-900, vrsta, proizvodac i naziv koji su tipa varchar s ograničenjem do 20 znakova i ne smiju imati NULL vrijednost.

```
CREATE TABLE lijek (
id INTEGER PRIMARY KEY,
vrsta VARCHAR (20) NOT NULL,
proizvodac VARCHAR (20) NOT NULL,
naziv VARCHAR (20) NOT NULL
);
```

6.9. TABLICA stanje_lijekova

Ova tablica služi kao evidencija stanja svih lijekova, pri čemu jedan lijek može biti upisan više puta, no sa različitim količinama i rokom valjanosti. Sadrži atribute: id tipa integer koji služi kao primarni ključ i u rasponu je od 900-1000, id_lijek tipa integer koji je strani ključ koji povezuje ovu tablicu s tablicom lijek, količina tipa integer koji ne smije biti NULL, rok_valjanosti tipa date koji ne smije biti NULL. Na kraju imamo ograničenje gdje rok valjanosti mora biti nakon datuma 15.05.2020.

```
CREATE TABLE stanje_lijekova (
id INTEGER PRIMARY KEY,
id_lijek INTEGER NOT NULL,
količina INTEGER NOT NULL,
rok_valjanosti DATE NOT NULL,
FOREIGN KEY (id_lijek) REFERENCES lijek (id),
CHECK (rok_valjanosti > STR_TO_DATE('15.05.2020.', '%d.%m.%Y.'))
);
```

6.10. TABLICA terapija

Tablica terapija povezuje pacijente sa lijekovima koji su im prepisani, uz kratak opis njihove upotrebe. Sadrži atribute: id tipa integer koji služi kao primarni ključ i u rasponu je od 1000-1100, id_pacijent tipa integer koji je strani ključ koji povezuje ovu tablicu sa tablicom pacijent i ne smije biti NULL, id_lijek tipa integer koji je strani ključ koji povezuje ovu tablicu sa tablicom lijek i ne smije biti NULL i uporaba tipa text pri čemu je omogućen unos dužeg teksta kao opisa (do 65535 znakova).

```
CREATE TABLE terapija (
id INTEGER PRIMARY KEY,
id_pacijent INTEGER NOT NULL,
id_lijek INTEGER NOT NULL,
uporaba TEXT,
FOREIGN KEY (id_pacijent) REFERENCES pacijent (id),
FOREIGN KEY (id_lijek) REFERENCES lijek (id)
);
```

6.11. TABLICA prijem

Tablica prijem sastoji se od atributa id koji je tipa INTEGER te služi kao primarni ključ,datum_prijema koje je tipa DATE. Ostali atributi su strani ključevi koji su tipa INTEGER. Strani ključ id_medicinske_sestre povezuje tablicu medicinske_sestre, strani ključ id_pacijent povezuje prijem sa tablicom pacijent, strani ključ id_doktor povezuje tablicu prijem sa tablicom doctor, id_dijagnoza sa tablicom dijagnoza i id_soba sa tablicom soba.

```
CREATE TABLE prijem(
id INTEGER PRIMARY KEY,
datum_prijema DATE NOT NULL,
id_pacijent INTEGER NOT NULL,
id_medicinske_sestre INTEGER NOT NULL,
id_doktor INTEGER NOT NULL,
id_dijagnoza INTEGER NOT NULL,
id_soba INTEGER NOT NULL,
FOREIGN KEY (id_pacijent) REFERENCES pacijent(id),
FOREIGN KEY (id_medicinske_sestre) REFERENCES medicinske_sestre(id),
FOREIGN KEY (id_doktor) REFERENCES doktor (id),
FOREIGN KEY (id_dijagnoza) REFERENCES dijagnoza(id),
FOREIGN KEY (id_soba) REFERENCES soba(id)
);
```

6.12. TABLICA posjeta

Tablica posjeta sadrži primarni ključ tipa INTEGER, atribute ime i prezime tipa VARCHAR s ograničenjem do 20 znakova, atribut temperature koji je tip NUMERIC gdje je odabrano da iza decimalnog zareza bude ukupno jedan broj. Imamo atribute vrijeme_dolaska i vrijeme_odlaska. Imamo i strani ključ id_pacijent koji veže tablicu posjeta sa tablicom pacijent.

```
CREATE TABLE posjeta(
id INTEGER PRIMARY KEY,
ime VARCHAR(20) NOT NULL,
prezime VARCHAR(20) NOT NULL,
temperatura NUMERIC(3,1) NOT NULL,
datum DATETIME NOT NULL,
vrijeme_dolaska TIME NOT NULL,
vrijeme_odlaska TIME NOT NULL,
id_pacijent INTEGER NOT NULL,
FOREIGN KEY (id_pacijent) REFERENCES pacijent (id)
);
```

6.13. TABLICA oprema

Tablica oprema ima primarni ključ tipa INTEGER te atribute sifra koji je tipa VARCHAR s ograničenjem od 10, naziv koji je isto VARCHAR s ograničenjem 30. Atribut datum_zaprimanja je tipa DATE.

```
CREATE TABLE oprema(
id INTEGER PRIMARY KEY,
sifra VARCHAR(10) NOT NULL UNIQUE,
naziv VARCHAR(30) NOT NULL,
datum_zaprimanja DATE NOT NUL
);
```

6.14. TABLICA stanje_opreme

Unutar tablice oprema imamo atribut id kao primarni ključ te strane ključeve id_soba i id_oprema. Atribut kolicina govori o kolicini opreme. Svi atributi, primarni i strani ključevi su tipa INTEGER.

```
CREATE TABLE stanje_opreme(
id INTEGER PRIMARY KEY,
id_oprema INTEGER NOT NULL,
id_soba INTEGER NOT NULL,
kolicina INTEGER NOT NULL,
FOREIGN KEY (id_oprema) REFERENCES oprema(id),
FOREIGN KEY (id_soba) REFERENCES soba(id)
);
```

7. UPITI

- Marija Ilić (7.1 - 7.5)
- Tin Pritišanac (7.6 – 7.10)
- Neven Davidović (7.11 – 7.13)
- Noel Modrušan (7.14 – 7.17)

7.1. UPIT 1

Količina opreme koju koristi pojedina medicinska sestra.

Medicinska sestra Ivana Ivić odlučila se požaliti ravnatelju bolnice kako joj fali opreme koju mora koristiti za kvalitetno obavljanje posla. Kako bi riješio problem medicinske sestre Ivane, ali i možebitne probleme ostalih medicinskih sestara u bolnici, ravnatelj bolnice je za početak zatražio popis svih medicinskih sestara i količine opreme koja im je dostupna za rad.

TRAŽENO RJEŠENJE:
medicinske_sestre id, **medicinske_sestre** ime_i_prezime, kolicina_opreme

KOD ZA UPIT:

```
CREATE VIEW sestra_soba AS
SELECT medicinske_sestre.id, CONCAT(medicinske_sestre.ime, ' ',
medicinske_sestre.prezime) AS ime_i_prezime, soba.id AS id_sobe
    FROM medicinske_sestre, soba
    WHERE medicinske_sestre.id_odjel=soba.id_odjel;

CREATE VIEW kol_po_sobi AS
SELECT sestra_soba.id, sestra_soba.ime_i_prezime,
COALESCE(kolicina.kol_u_sobi, 0) AS kolicina
    FROM sestra_soba
LEFT JOIN
    (SELECT *, SUM(kolicina) AS kol_u_sobi
        FROM stanje_opreme
        GROUP BY id_soba) AS kolicina ON sestra_soba.id_sobe=kolicina.id_soba;

SELECT kol_po_sobi.id, kol_po_sobi.ime_i_prezime, SUM(kolicina) AS
kolicina_opreme
    FROM kol_po_sobi
    GROUP BY id;
```

OPIS UPITA:

S obzirom da je oprema raspoređena po sobama moramo medicinske sestre povezati sa sobama u kojima obavljaju rad kako bi mogli prebrojati opremu koja je dostupna pojedinoj medicinskoj sestri.

- Unutar prvog upita povezala sam tablicu **medicinske_sestre** i tablicu **soba** pomoću kartezijevog produkta. Kartezijev produkt sam dobila na način da u FROM dijelu navedem obje tablice.
- U WHERE dio upita dodala sam uvjet povezivanja po jednakom **id_odjel** atributu kako bi ispravna imena medicinskih sestra povezala sa ispravnim brojevima soba u kojima obavljaju rad.
- U SELECT dio navela sam atribute koji će mi koristiti za daljnje rješavanje, a to su **id** atribut iz tablice **medicinske_sestre**, **ime_i_prezime** te **id_sobe** iz tablice **soba**.
- Atribut **ime_i_prezime** sam spojila iz dva atributa naredbom CONCAT.
- Od prvog upita kreirala sam pogled pod nazivom **sestra_soba** koristeći naredbu CREATE VIEW.
- Nakon što sam kreirala pogled koji nam govori u kojoj sobi radi koja medicinska sestra, pronašla sam količinu opreme koja se nalazi u svakoj od soba na način da sam tablicu **stanje_opreme** grupirala prema atributu **id_soba** naredbom GROUP BY.
- Unutar SELECT dijela drugog upita, osim cijele tablice **stanje_opreme**, dodala sam i atribut **kol_u_sobi**.
- Atribut **kol_u_sobi** označava količinu opreme koja se nalazi u pojedinoj sobi, a s obzirom da sam već grupirala cijelu tablicu prema atributu **id_soba**, lako sam izračunala i količinu koristeći naredbu SUM.
- Kako bi povezala medicinske sestre sa količinom opreme, drugi sam upit pomoću naredbe LEFT JOIN pridodala pogledu **sestra_soba**. Pri povezivanju tablice preimenovala sam ju u **kolicina**, a povezivanje sam obavila prema uvjetu da su atributi **broj_sobe** iz pogleda **sestra_soba** i **id_soba** iz tablice **kolicina** jednaki.
- U SELECT dio upita navela sam samo potrebne atribute, a to su: **id** i **ime_i_prezime** iz pogleda **sestra_soba** i **kolicina**.
- Atribut **kolicina** rezultat je sumiranja količine opreme iz pridodane tablice **kolicina**, a naredbom COALESCE postigla sam upisivanje 0 gdje se nalazi NULL vrijednost unutar atributa.
- U konačnici sam pomoću naredbe CREATE VIEW kreirala pogled pod nazivom **kol_po_sobi** u kojem se nalaze sve medicinske sestre i količina opreme koju imaju na raspolaganju.
- Kako bi vidjeli točan popis medicinskih sestara i točnu količinu opreme koju imaju na raspolaganju ponaosob (kako se imena i prezimena ne bi ponavljala i ne bi se prepustilo nama da sami zbrajamo opremu po imenima), kreirala sam i treći upit.
- Unutar trećeg upita sam pogled **kol_po_sobi** grupirala prema atributu **id** koji se odnosi na **id** medicinskih sestara i u SELECT dijelu sam, osim atributa koji su potrebni za rješenje, konačno sumirala opremu pomoću funkcije SUM preko atributa **kolicina** i naredbom AS rezultat preimenovala u **kolicina_opreme**.

REZULTAT:

id	ime_i_prezime	kolicina_opreme
500	Ivana Ivić	2
501	Miliana Milić	12
502	Ivana Marić	12
503	Marko Marulić	2
504	Žarka Stanić	12

7.2. UPIT 2

Popis SOS kontakata svih pacijenata sa dijagnozom 'Infarctus myocardii acutus'.

Doktor Noris Grubor sa odjela Kardiologije zamolio je medicinsku sestru da obavijesti sve SOS kontakte pacijenata sa dijagnozom Infarctus myocardii acutus da im je stanje kritično zbog proživljenog stresa uzrokovanog potresom. Kako bi to učinila, medicinska sestra zatražila je popis svih SOS kontakata pacijenata s tom dijagnozom.

TRAŽENO RJEŠENJE:
pacijent ime_i_prezime, **sos_kontakt** sos_ime_i_prezime , **sos_kontakt** broj_telefona

KOD ZA UPIT:

```
CREATE VIEW sa_dijagnozom AS
SELECT prijem.id_pacijent
    FROM prijem, dijagnoza
    WHERE prijem.id_dijagnoza=dijagnoza.id AND dijagnoza.naziv='Infarctus
myocardii acutus';

SELECT CONCAT(pacijent.ime, ' ', pacijent.prezime) AS ime_i_prezime,
CONCAT(sos_pod.ime, ' ', sos_pod.prezime) AS sos_ime_i_prezime,
sos_pod.broj_telefona
    FROM pacijent
RIGHT JOIN sa_dijagnozom AS trazena_dijag ON
pacijent.id=trazena_dijag.id_pacijent
LEFT JOIN sos_kontakt AS sos_pod ON pacijent.id_sos_kontakt=sos_pod.id;
```

OPIS UPITA:

Kako bi uspješno odgovorili na ovaj upit prvo moramo pronaći kojim je pacijentima dijagnosticiran 'Infarctus myocardii acutus', što ćemo učiniti preko tablice **prijem** unutar koje su nam pobrojane dijagnoze svakog pacijenta. Zatim povezujemo tablice **pacijent** (kako bi dobili potrebne podatke o pacijentu) i **sos_kontakt** (kako bi dobili potrebne podatke o SOS kontaktu pacijenta).

- Unutar prvog upita u FROM dijelu napravila sam kartezijev produkt tablica **prijem** i **dijagnoza**.
- U WHERE dio upita navela sam dva uvjeta spajanja. Prvi uvjet spajanja odnosi se na atribut **id** dijagnoze kako bi unutar tablice **prijem** dobili ispravna imena postavljenih dijagnoza, a drugi uvjet spajanja odnosi se na konkretan **naziv** dijagnoze koju tražimo u rezultatu.
- U SELECT dijelu upita navela sam samo atribut **id_pacijent** jer mi je on dovoljan za daljnji rad na upitu.
- Pomoću naredbe CREATE VIEW sam kreirala pogled imena **sa_dijagnozom** od prvog upita.
- Kako bi na temelju **id_pacijent** atributa iz pogleda **sa_dijagnozom** dobila potrebne podatke vezane uz pacijente, pogled **sa_dijagnozom** sam pod imenom **trazena_dijag**, pomoću RIGHT JOIN naredbe povezala sa tablicom **pacijent**. Kao uvjet povezivanja navela sam jednakost atributa **id** iz tablice **pacijent** i atributa **id_pacijent** iz tablice **trazena_dijag**.
- Jedini podaci koji nam sad fale su podaci vezani uz SOS kontakt podjedinog pacijenta. Kako bi i taj dio upita uspješno riješila uključila sam tablicu **sos_kontakt** pomoću naredbe LEFT JOIN pod nazivom **sos_pod**, sa uvijetom da su atributi **id_sos_kontakt** iz tablice **pacijent** i **id** iz tablice **sos_pod** jednaki.
- U SELECT dio upita navela sam atribute **ime_i_prezime** (odnosi se na ime i prezime pacijenta), **sos_ime_i_prezime** (odnosi se na ime i prezime SOS kontakta), te **broj_telefona** (odnosi se na broj telefona SOS kontakta).
- Atribute **ime_i_prezime** i **sos_ime_i_prezime** sam spojila pomoću naredbe CONCAT od odvojenih atributa ime i prezime iz tablica **pacijent** i **sos_pod**.

REZULTAT:

ime_i_prezime	sos_ime_i_prezime	broj_telefona
Sebastijan Milošević	Aden Kotolaš	0997871285
Lucas Perić	Vojmil Novaković	0990409583

7.3. UPIT 3

Doktor s najviše pacijenata.

Ravnatelj bolnice odlučio je dodijeliti nagradu doktoru koji je zadužen za najviše pacijenata.

TRAŽENO RJEŠENJE:
doktor id, **doktor** ime_i_prezime, **doktor** id_odjel, broj_pacijenata

KOD ZA UPIT:

```
CREATE VIEW svi_brojevi_pac AS
SELECT doktor.id, CONCAT(doktor.ime, ' ', doktor.prezime) AS ime_i_prezime,
doktor.id_odjel,
COALESCE(broj_pacijenata, 0) AS broj_pacijenata
    FROM doktor
LEFT JOIN
    (SELECT id_doktor, COUNT(id_pacijent) AS broj_pacijenata
        FROM prijem
        GROUP BY id_doktor) AS izracun ON doktor.id=izracun.id_doktor;

SELECT *
    FROM svi_brojevi_pac
    ORDER BY broj_pacijenata DESC
    LIMIT 1;
```

OPIS UPITA:

Za rješavanje ovoga upita ključna je tablica **prijem** unutar koje je evidentirana dodjela doktora pacijentu prilikom prijema pacijenta u bolnicu.

- Prvi korak koji sam poduzela u svrhu rješavanja ovoga upita je prebrojavanje pacijenata unutar tablice **prijem** za svakog doktora ponaosob.
- Naredbom GROUP BY grupirala sam prema atributu **id_doktor** podatke iz tablice **prijem**.
- Kako bi mogla prebrojati pacijente upotrijebila sam naredbu COUNT na atributu **id_pacijent** unutar SELECT dijela upita i naredbom AS imenovala dobiveni atrinut kao **broj_pacijenata**.
- Također sam u SELECT dio upita navela atribut **id_doktor** koji mi je potreban za povezivanje traženih podataka o doktoru.
- Kako bi dobila podatke o doktoru, naredbom LEFT JOIN dodala sam dobiveni rezultat upita unutar tablice **doktor** pod nazivom **izracun**. Kao uvijet spajanja iskoristila sam jednakost atributa **id** iz tablice **doktor** i atributa **id_doktor** iz tablice **izracun**.
- U SELECT dio upita navela sam iz tablice **doktor** slijedeće atribute: **id**, **ime_i_prezime** i **id_odjel**.
- Atribut **ime_i_prezime** dobila sam spajanjem atributa **ime** i atributa **prezime** iz tablice **doktor** koristeći naredbu CONCAT.

- Kreirala sam pogled s imenom **svi_brojevi_pac** pomoću naredbe CREATE VIEW.
- Na kraju sam kreirala upit u kojem sam u SELECT dijelu navela sve atribute pogleda **svi_brojevi_pac** koju sam navela u FROM dijelu upita.
- Naredbom ORDER BY sam sortirala podatke silazno prema atributu **broj_pacijenata**.
- Naredbom LIMIT sa vrijednošću 1 sam u konačnici dobila doktora s najviše pacijenata.

RJEŠENJE:

id	ime_i_prezime	id_odjel	broj_pacijenata
200	Krešimira Paspalj	102	9

7.4. UPIT 4

Popis svih lijekova kojima je datum isteka prije 2023. godine, a proizvođač im je Bayer.

Potrebno je napisati narudžbu lijekova za proizvođača Bayer.

TRAŽENO RJEŠENJE:
lijek id, **lijek** vrsta, **lijek** proizvodac, **lijek** naziv

KOD ZA UPIT:

```
CREATE VIEW prije_zadane AS
SELECT *
    FROM stanje_lijekova
    WHERE YEAR (rok_valjanosti) < 2023
    ORDER BY rok_valjanosti ASC;

SELECT lijek_pod.id, lijek_pod.vrsta, lijek_pod.proizvodac, lijek_pod.naziv
    FROM prije_zadane
LEFT JOIN lijek AS lijek_pod ON prije_zadane.id_lijek=lijek_pod.id
HAVING proizvodac='Bayer';
```

OPIS UPITA:

Datum isteka roka svakog lijeka vidljiv je unutar tablice **stanje_lijekova**, a ostali podaci o lijeku koji su nam potrebni da bi izvršili narudžbu proizvođaču nalaze se unutar tablice **lijek**.

- Unutar prvog upita pomoću naredbe YEAR izvršene na atributu **rok_valjanosti** u WHERE dijelu navela sam kako godina isteka roka mora biti manja od 2023. godine.
- Tablica navedena u FROM dijelu upita, na kojoj sam izvršila ovaj uvijet je **stanje_lijekova**, a pomoću ORDER BY naredbe datume sam poredala silazno.
- Pomoću naredbe CREATE VIEW kreirala sam pogled imena **prije_zadane**.
- Kako bi dobila podatke vezane uz lijekove koji su potrebni da bi se izvršila narudžba, povezala sam tablicu **lijek** pomoću LEFT JOIN naredbe pod imenom **lijek_pod** sa

pogledom **prije_zadane**. Uvijet koji sam postavila za povezivanje je jednakost **id_lijek** atributa iz pogleda **prije_zanadne** sa **id** atributom iz tablice **lijek_pod**.

- Naredbom HAVING sam zadala uvijet da se prikazuju samo oni lijekovi koji imaju pod atributom **proizvodac** naziv traženog proizvođača.
- U SELECT dijelu navela sam samo atribute potrebne za izvršavanje narudžbe, a to su **id**, **vrsta**, **proizvodac** i **naziv** iz tablice **lijek_pod**.

RJEŠENJE:

id	vrsta	proizvodac	naziv
800	Anelgetik	Bayer	Aspirin
812	Antibiotik	Bayer	Meropenem

7.5. UPIT 5

Popis pacijenata sa brojem posjeta (odvojeno posjete SOS kontakata i onih koji nisu SOS kontakti).

Na otpusnom listu pacijenta bolnica svakom pacijentu daje podatak koliko je ukupno imao SOS kontakat posjeta i ostalih posjeta. Zbog potresa bolnica se seli u zamjensku zgradu dok oštećena zgrada ne bude ponovno u funkciji. Ravnatelj bolnice naredio je da se svakom pacijentu prije preseljenja izda dokument sličan otpusnom listu koji između ostalih sadrži i te podatke.

TRAŽENO RJEŠENJE:
pacijent id, **pacijent** ime_i_prezime, **pacijent** spol, sos_broj_posjeta, broj_ostalih_posjeta

KOD ZA UPIT:

```
SELECT pacijent.id, CONCAT(pacijent.ime, ' ', pacijent.prezime) AS
ime_i_prezime,
pacijent.spol, COALESCE(sos.sos_broj_posjeta, 0) AS sos_broj_posjeta,
COALESCE(ukupno.ukupna_posjeta, 0)-COALESCE(sos.sos_broj_posjeta, 0) AS
broj_ostalih_posjeta
    FROM pacijent
LEFT JOIN
    (SELECT posjeta.id_pacijent, COUNT(posjeta.id) AS sos_broj_posjeta
        FROM posjeta, sos_kontakt
        WHERE posjeta.ime=sos_kontakt.ime AND
posjeta.prezime=sos_kontakt.prezime
        GROUP BY id_pacijent) AS sos ON pacijent.id=sos.id_pacijent
LEFT JOIN
    (SELECT posjeta.id_pacijent, COUNT(posjeta.id) AS ukupna_posjeta
        FROM posjeta
        GROUP BY id_pacijent) AS ukupno ON pacijent.id=ukupno.id_pacijent;
```

OPIS UPITA:

Za rješavanje ovoga upita koristit ću tri tablice. Unutar tablice **posjeta** možemo provjeriti koliko je posjeta imao pojedini pacijent. U kombinaciji tablica **sos_kontakt** i **posjeta** možemo provjeriti koliko je od tih posjeta bilo od strane SOS kontakata pacijenta, a unutar tablice **pacijent** možemo naći podatke o pojedinom pacijentu.

- Prvi korak je pronaći broj SOS posjeta, a to sam učinila na način da sam u FROM dijelu upita navela tablice **posjeta** i **sos_kontakt** i time kreirala kartezijev produkt.
- U WHERE dijelu upita navela sam dva uvjeta. Prvi uvjet garantira jednakost atributa **ime** u objema tablicama, a drugi uvjet atributa **prezime**.
- Sve skupa sam grupirala prema atributu **id_pacijent** naredbom GROUP BY.
- U SELECT dijelu upita navela sam atribut **id_pacijent** (koji će mi biti potreban za daljnje povezivanje podataka o pacijentu) i atribut **sos_broj_posjeta** koji sam dobila prebrojavajući atribut **id** iz tablice **posjeta** naredbom COUNT.
- Drugi korak je pronaći ukupan broj posjeta. Tablicu **posjeta** grupirala sam naredbom GROUP BY prema atributu **id_pacijent**.
- U SELECT dijelu upita navela sam atribut **id_pacijent** (koji mi je potreban za daljnje povezivanje) i atribut **ukupna_posjeta** koji sam dobila koristeći naredbu COUNT na atributu **id**.
- Kako bi povezala agregirane podatke sa podacima vezanim uz pojedinog pacijenta prvo sam pomoću naredbe LEFT JOIN pod imenom **sos** i uvjetom jednakosti **id** atributa iz tablice **pacijent** sa atributom **id_pacijent** iz tablice **sos** dodala prvi rezultat unutar tablice **pacijent**.
- Zatim sam na isti način dodala i drugi rezultat samo pod nazivom **ukupno**.
- U SELECT dijelu navela sam atribute **id**, **ime_i_prezime** i **spol** iz tablice **pacijent**, te **sos_broj_posjeta** i **broj_ostalih_posjeta**.
- Atribut **ime_i_prezime** objedinila sam koristeći naredbu CONCAT od atributa **ime** i **prezime** iz tablice **pacijent**.
- Kako bi zamjenila NULL vrijendosti sa nulama, u svrhu računanja, koristila sam naredbu COALESCE na atributima **ukupna_posjeta** iz tablice **ukupno** i **sos_broj_posjeta** iz tablice **sos**.
- Atribut **broj_ostali_posjeta** dobila sam oduzimanjem atributa **sos_broj_posjeta** od atributa **ukupna_posjeta** u SELECT dijelu upita.

RJEŠENJE:

id	ime_i_prezime	spol	sos_broj_posjeta	broj_ostalih_posjeta
400	Alen Kolić	M	1	0
401	Ivan Rupčić	M	1	0
402	Lissa Ivić	Ž	1	0
403	Klara Zenzerović	Ž	1	0
404	Sebastijan Milošević	M	1	0
405	Lucas Perić	M	1	0
406	Paola Marić	Ž	1	0
407	Ema Knežević	Ž	0	1
408	Tomi Ivković	M	0	1
409	Stefan Markulinčić	M	0	1

410	Mia Stepančić	Ž	0	1
411	Bruno Marušić	M	0	1
412	Iris Orbanić	Ž	0	1
413	Nora Marjanović	Ž	0	1
414	Borna Karlović	M	0	0
415	Oliver Kinić	M	0	0
416	Toni Belić	M	0	0
417	Petar Dragojević	M	0	0
418	Boris Vlašić	M	0	0
419	Matija Mladenović	M	0	0
420	Filip Perišić	M	0	0

7.6. UPIT 6

Svi pacijenti koji su primljeni prije datuma X kod doktora Y a koji još nisu primili terapiju

Doktorica Debeljak (id:208) otišla je na godišnji odmor 30.08.2022, te je bilo potrebno zbrinuti njene pacijente koji su joj dodjeljeni prije odlaska na godišnji odmor, a kojima ona još nije prepisala terapiju.

KOD ZA UPIT:

```
SELECT pa.*
FROM pacijent as pa, prijem as pr, doktor as d
WHERE pa.id = pr.id_pacijent
    AND pr.id_doktor = d.id
    AND d.id = 208
    AND pr.datum_prijema < STR_TO_DATE('30.08.2022','%d.%m.%Y.')
HAVING pa.id NOT IN (SELECT id_pacijent FROM terapija);
```

OPIS UPITA:

Kako bi ostvarili ovaj upit potrebno je povezati pacijente sa datumima kada su zaprimljeni u bolnicu, te sa doktorom koji ih je zaprimio. Tablica **prijem** povezuje ove tri relacije jer ona sadrži id svakog pacijenta, datum prijema i doktora koji je pacijenta primio.

- Najprije smo naveli sve relacije pod naredbu FROM odvajajući ih zarezom i dodjeljujući im novi alias. Time smo ih spojili kartezijevim produktom.
- Nakon toga smo povezali tablice preko id-jeva dodajući uvjete pod naredbu WHERE pri čemu smo naveli da id-jevi iz tablice pacijent moraju odgovarati id-jevima pacijenata iz tablice prijem, te da id-jevi doktora iz tablice prijem moraju odgovarati id-jevima doktora iz tablice doktor. Te uvjete smo lančano povezali naredbom AND
- Također dodali smo još i uvjet da id doktora mora biti 208, što odgovara doktorici Debeljak, te da datum prijema iz tablice prijem mora biti raniji tj. manji od datuma odlaska doktorice na godišnji, koji smo unijeli funkcijom STR_TO_DATE koja formatira podatak o datumu iz tipa podatka string u tip podatka date.

- Za kraj smo dodali još jedan uvjet naredbom HAVING gdje smo definirali da se id pacijenta ne smije nalaziti unutar skupa id-jeva iz relacije terapija. Za to smo koristili naredbu NOT IN iza koje slijedi drugi upit koji kao rezultat daje id-jeve svih pacijenata koji su primili terapiju.

REZULTAT:

id	ime	prezime	datum_rodenja	adresa	spol	broj_telefona	MBO	id_sos_kontakt
407	Ema	Knežević	1995-10-24 00:00:00	Busoler 18	Ž	0957135987	24578946	307
418	Boris	Vlašić	1984-07-28 00:00:00	Puljska cesta 19	M	092548796	815687459	318

7.7. UPIT 7

Popis svih pacijenata i ljudi koji su bili u doticaju s njima (doktori, sestre, posjetitelji) na katu broj 2 na kojem je izbila zaraza nakon datuma X

Na katu broj 2 bolnice izbila je zaraza među pacijentima i potrebno je sastaviti popis svih ljudi (uključujući i pacijente) koji su se nakon određenog datuma našli na ovom katu.

KOD ZA UPIT:

```
CREATE VIEW zarazeni_pacijenti AS
SELECT p.id,p.ime,p.prezime
FROM pacijent as p, prijem, soba, odjel
WHERE p.id = prijem.id_pacijent
  AND prijem.id_soba = soba.id
  AND soba.id_odjel = odjel.id
  AND kat = 2
  AND datum_prijema >= STR_TO_DATE('24.08.2022','%d.%m.%Y.');


-- popis zarazenih pacijenata
SELECT *  FROM  zarazeni_pacijenti

UNION

-- popis svih doktora na temelju zarazenih pacijenata
SELECT DISTINCT doc.id, doc.ime, doc.prezime
FROM doktor as doc, zarazeni_pacijenti as za , prijem
WHERE za.id = prijem.id_pacijent
AND doc.id = prijem.id_doktor

UNION

-- popis svih medicinskih sestara na temelju zarazenih pacijenata
SELECT DISTINCT med.id, med.ime, med.prezime
FROM medicinske_sestre as med, zarazeni_pacijenti as za , prijem
WHERE za.id = prijem.id_pacijent
```

```
AND med.id = prijem.id_medicinske_sestre

UNION

-- popis svih posjeta pacijentima drugog kata na temelju tablice zarazenih
pacijenata
SELECT DISTINCT pos.id, pos.ime, pos.prezime
FROM posjeta as pos, zarazeni_pacijenti as za
WHERE za.id = pos.id_pacijent;
```

OPIS UPITA:

Za ovaj upit najprije je bilo potrebno pronaći sve zaražene pacijente sa drugog kata bolnice nakon zadaog datuma. Za ovaj korak bilo je potrebno povezati relacije **pacijent, prijem, soba i odjel**, budući da relacija prijem povezuje svakog pacijenta sa njegovom sobom, a relacija sobe povezuje svaku sobu sa pripadajućim odjelom, te u tablici odjel nalazimo informaciju na kojem katu bolnice se nalazi koji odjel. Ovime dakle dolazimo do informacije na kojem katu bolnice se nalaze koji pacijenti te kada su primljeni u bolnicu. U sljedećim koracima povezali smo te pacijente sa drugim osobama koje su im dodjeljene kroz tablicu prijem (sestre i doktori) te tablicu posjeta.

- U prvom upitu povezali smo sve relacije **pacijent, prijem, soba i odjel** odvajajući ih zarezom nakon naredbe FROM. Naredbom WHERE povezali smo ih preko pripadajućih id-jeva. Pacijenta i sobu sa prijemom, te sobu i odjel.
- Postavili smo uvjet da se pacijenti moraju nalaziti na 2. katu i da moraju biti primljeni nakon ili na datum 24.08.2022.
- U naredbi SELECT odabrali smo podatke o pacijentima
- Za lakše daljnje korištenje rezultate ovog upita spremili smo u novi pogled zvan zarazeni_pacijenti postavljanjem naredbe CREATE VIEW zarazeni_pacijenti AS ispred upita.
- Budući da su pacijenti povezani sa sestrama i doktorima preko relacije prijem, u sljedećem upitu povezali smo tablice doktor, zaraženi_pacijenti i prijem, također preko id-jeva iza naredbe WHERE te smo iza naredbe SELECT odabrali podatke o doktorima. Također bilo je potrebno dodati i naredbu DISTINCT kako bi eliminirali duplikate, budući da više pacijenata može biti primljeo kod istog doktora.
- Prethodni korak ponovili smo još jednom za novi upit no ovaj puta umjesto tablice doktor povezali smo tablicu medicinske sestre.
- Za kraj povezali smo tablice zaraženi_pacijenti i posjeta navodeći ih iza naredbe FROM i postavljajući uvjet da id zaraženog pacijenta mora odgovarati id-ju pacijenta iz relacije posjeta. Međusobno smo povezali sve ove upite naredbama UNION i tako došli do konačnog rezultata.

REZULTAT:

id	ime	prezime

402	Lissa	Ivić
403	Klara	Zenzerović
205	Jan	Nikolić
207	Božidarka	Obad
503	Marko	Marulić
1203	Issa	Biševac
1204	Elenora	Delfar

7.8. UPIT 8

Količine lijekova po nazivima kojima ističe rok trajanja za manje od godinu dana

Odjel nabave obavezan je unaprijed naručiti nove zalihe lijekova kojima ističe rok za manje od godinu dana, stoga je potrebno sastaviti popis i stanje takvih zaliha lijekova u bolnici.

Lijekovi kojima je već istekao rok trajanja su također u evidenciji sve do njihovog zbrinjavanja i odlaganja, no u ovom slučaju nas zanimaju samo lijekovi sa još uvijek valjanim rokom uporabe koji se trenutno koriste u bolnici.

KOD ZA UPIT:

```
SELECT li.id, li.naziv, st.količina, st.rok_valjanosti,
(SELECT DATEDIFF(rok_valjanosti,(SELECT NOW() FROM DUAL))) as dani_do_isteka
FROM lijek as li
INNER JOIN stanje_lijekova as st ON li.id = st.id_lijek
HAVING dani_do_isteka BETWEEN 0 AND 365
ORDER BY dani_do_isteka ASC;
```

OPIS UPITA:

Informacije koje su nam potrebne za ovaj upit su naziv, količina i rok valjanosti, a nalaze se u dvije tablice, lijek i stanje_lijekova.

- Relaciju lijek navodimo iz naredbe FROM i dodjeljujemo joj alias li
- Spajamo ovu relaciju INNER JOIN-om sa relacijom stanje_lijekova preko id-jeva lijeka
- Iza naredbe HAVING navodimo uvjet korištenjem riječi BETWEEN da dani do isteka moraju biti između 0 i 365 kako bi eliminirali sve lijekove kojima je rok već istekao i kojima će tek isteći ali za više od godinu dana.
- Listu lijekova poredali smo naredbom ORDER BY dani_do_isteka prema rastu naredbom ASC na kraju retka kako bi na početku liste dobili one lijekove kojima rok najranije istječe.
- Za kraj iza naredbe SELECT navodimo podatke id i naziv iz tablice lijek, količina i rok valjanosti iz tablice stanje_lijekova, te dane do isteka valjanosti koje računamo koreliranim podupitom koristeći funkciju DATEDIFF koja prima dva datuma, a kao rezultat vraća broj dana između njih. Za prvi datum navodimo rok_valjanosti, a za

drugi trenutni datum za vrijeme upita koristeći funkciju NOW koja vraća trenutni datum. Ovaj podupit izvršiti će se za svaku n-torku relacije i rezultat će biti prikazan u stupcu dani_do_isteka.

REZULTAT:

id	naziv	količina	rok_valjanosti	dani_do_isteka
812	Meropenem	8756	2022-07-02	38
822	Haloperidol	356	2022-07-05	41
814	Gentamycin	890	2022-09-07	105
820	Heparin	650	2022-09-10	108
804	Ketamine	1201	2022-09-15	113
824	Cyclizine	630	2022-10-07	135
805	Bisoprolol	610	2023-02-16	267
818	Rivaroxaban	980	2023-03-22	301
813	Vancomycin	312	2023-04-12	322

7.9. UPIT 9

Koliko je koji doktor propisao lijekova pacijentima u razdoblju između X i Y i tko su doktori koji su prepisali naviše lijekova pacijentima u tom razdoblju?

KOD ZA UPIT:

```
CREATE VIEW broj_izdavanja_lijeka AS
SELECT doc.id, doc.ime, doc.prezime, COUNT(doc.id) as ukupno_izdano_lijekova
FROM pacijent as pa
INNER JOIN prijem as pr ON pa.id = pr.id_pacijent
INNER JOIN doktor as doc ON doc.id = pr.id_doktor
INNER JOIN terapija as te ON pa.id = te.id_pacijent
WHERE  datum_prijema BETWEEN STR_TO_DATE('22.08.2022','%d.%m.%Y.') AND
STR_TO_DATE('29.08.2022','%d.%m.%Y.')
GROUP BY doc.id
ORDER BY ukupno_izdano_lijekova DESC;

-- ukupno po doktoru
SELECT *
FROM broj_izdavanja_lijeka;

-- doktori sa najviše izdanih lijekova
SELECT *
FROM broj_izdavanja_lijeka
WHERE ukupno_izdano_lijekova = (SELECT MAX(ukupno_izdano_lijekova) FROM
broj_izdavanja_lijeka);
```

OPIS UPITA:

Kako bi povezali doktora sa brojem izdanih terapija (svaka pojedina terapija prepisuje samo jedan lijek!) tj. brojem izdanih lijekova potrebno je povezati tablice pacijent, prijem, doktor, i terapija. Tablica prijem povezuje pacijente i doktore, a tablica terapija povezuje pacijente i lijekove koji su im prepisani. Povezivanjem ovih tablica u konačnici ćemo povezati doktore sa lijekvima.

- Upit krećemo od tablice pacijent koju navodimo nakon naredbe FROM i dodjeljujemo joj kraći alias
- Nakon toga radimo INNER JOIN pacijenta sa prijemom, te doktora sa prijemom preko pripadajućih id-jeva te na kraju povezujemo terapiju sa pacijentom na također isti način.
- Nakon naredbe WHERE naredbom BETWEEN postavljamo uvjet između dva datuma koja unosimo funkcijom STR_TO_DATE koja datum tipa string pretvara u tip podatka date nad kojim se mogu vršiti razne operacije.
- Narebom GROUP BY grupiramo tablicu prema id-ju doktora kako bi se mogla izvršiti funkcija prebrojavanja COUNT.
- Konačni rezultat sortiramo naredbom ORDER BY prema ukupnom broju izdanih lijekova u padajućem smjeru ključnom riječi DESC.
- Za kraj iza naredbe SELECT specificiramo podatke koje želimo u rezultatu, a to su id, ime, prezime doktora iz tablice doktor (doc) i ukupno_izdano_lijekova dobiveno korištenjem funkcije COUNT kojom smo prebrojali prethodno grupirane id-jeve doktora.
- Cjelokupni upit pohranili smo u pogled broj_izdavanja_lijeka naredbom CREATE VIEW broj_izdavanja_lijeka AS prije naredbe SELECT.
- Selektiranjem svih atributa iz pogleda broj_izdavanja_lijeka dolazimo do broja izdanih lijekova po doktoru
- Kako bi dobili doktore koji su izdali najviše lijekova iza naredbe WHERE postavljamo uvjet da ukupno_izdano_lijekova mora biti jednako najvišoj vrijednosti iz tog upita. Tu vrijednost dobivamo korištenjem funkcije MAX u podupitu koja vraća najveću vrijednost stupca ukupno_izdano_lijekova.

REZULTATI:

id	ime	prezime	ukupno_izdano_lijekova
200	Krešimira	Paspalj	5
208	Jelena	Debeljak	4
201	Noris	Grubor	2
207	Božidarka	Obad	1

id	ime	prezime	ukupno_izdano_lijekova
200	Krešimira	Paspalj	5

7.10. UPIT 10

Kako bi ustvrdili koje kategorije lijekova se u bolnici najviše koriste napravili smo upit koji će prebrojati koliko je pacijenata primilo koju vrstu lijeka uključujući i lijekove koji nisu prepisani niti jednom pacijentu.

KOD ZA UPIT:

```
CREATE VIEW pacijenti_po_lijeku AS
SELECT vrsta as vrsta_lijeka, COUNT(vrsta) as broj_pacijenata
FROM pacijent as pa
INNER JOIN terapija as te ON pa.id = te.id_pacijent
INNER JOIN lijek as li ON li.id = te.id_lijek
GROUP BY vrsta_lijeka;

SELECT DISTINCT li.vrsta as vrsta_lijeka, COALESCE(broj_pacijenata, 0) as
broj_pacijenata
FROM lijek as li
LEFT JOIN pacijenti_po_lijeku ON li.vrsta = pacijenti_po_lijeku.vrsta_lijeka
ORDER BY broj_pacijenata DESC, vrsta_lijeka ASC;
```

OPIS UPITA:

Tablica terapija povezuje svakog pacijenta sa lijekom koji mu je prepisan. Jedan pacijent može imati više terapija, tj. prepisanih lijekova. Budući da tablica terapija sadrži samo id lijeka, a ne i podatke o njemu, na to moramo povezati još i tablicu lijek koja sadrži podatak o vrsti lijeka koji je prepisan.

Potom ćemo vršiti grupiranje i prebrojavanje prema vrsti lijeka te ćemo to spojiti sa relacijom koja sadrži sve vrste lijekova u bolnici kako bi u rezultat uključili i lijekove koji nisu prepisani niti jednom pacijentu.

- Upit započinjemo navođenjem relacije pacijent iza naredbe FROM te mu dodjeljujemo kraći alias pa.
- Naredbama INNER JOIN povezujemo relacije terapija i lijek sa pacijentom prema pripadajućim id-jevima relacija.
- Grupiramo rezultat prema vrsti lijeka koristeći naredbu GROUP BY
- U naredbi SELECT odabiremo stupce prikaza, a to su vrsta lijeka i broj pacijenata dobiven prebrojavanjem stupca vrsta funkcijom COUNT.
- Ovim upitom dobili smo sve vrste lijekova koji su prepisani nekim pacijentima i broj pacijenata. Rezultat smo pohranili u pogled pacijenti_po_lijeku naredbom CREATE VIEW pacijenti_po_lijeku AS ispred naredbe SELECT
- Kako bi u rezultat uključili i lijekove koji nisu prepisani niti jednom pacijentu postavljamo novi upit.

- Tablica lijek sadrži popis svih lijekova u bolnici, stoga ćemo iz nje selektirati stupac vrsta, no budući da bolnica ima mnogo lijekova različitog naziva no iste vrste, potrebno je koristiti naredbu DISTINCT kako bi se eliminirali duplikati.
- U naredbi FROM navodimo tablicu lijek
- Sada ćemo ovu tablicu koja sadrži vrste svih lijekova, spojiti sa prethodnom tablicom pohranjenom u pogledu pacijenti_po_lijeku koristeći naredbu LEFT JOIN, a spajanje ćemo vršiti prema stupcu vrsta
- Za kraj rezultat ćemo sortirati prema broju pacijenata padajuće naredbom DESC, te potom prema vrsti rastuće naredbom ASC.
- Potrebno je u naredbu SELECT još uključiti stupac koji sadrži broj pacijenata, no kako bi izbjegli NULL vrijednosti u rezultatu koristiti ćemo funkciju COALESCE koja će iz zadanog seta odabrati prvi rezultat koji nije jednak NULL. Time će vrste lijekova koje nemaju niti jednog pacijenta biti evidentirane kao 0 umjesto NULL.

REZULTAT:

vrsta_lijeka	broj_pacijenata
Antibiotik	7
Anelgetik	4
Antiaritmik	2
Antikoagulans	1
Antiemetik	0
Sedativ	0

7.11. UPIT 11

Prikaz svih medicinskih sestara i pacijenta za koje su one zadužene, poredane po atributu id_pacijent. Cilj je bio prikazati sve pacijente i sve medicinske sestre koje su zadužene za njih te sobe i uz sobe id odjela na kojemu se i sobe i pacijenti, a i sestre nalaze.

```
SELECT m.id as ID ,CONCAT(m.ime ,' ', m.prezime)as Ime_i_prezime,pac.id,
pac.ime as ime_pacijenta,pac.prezime as prezime_pacijenta,
s.broj_sobe,s.id_odjel
  FROM medicinske_sestre as m, prijem as p,pacijent as pac, soba as s
  WHERE m.id=p.id_medicinske_sestre AND p.id_pacijent=pac. id AND
p.id_soba=s.id
  ORDER BY pac.id DESC;
```

OPIS UPITA:

Ovaj upit smo napravili tako da smo u SELECT dijelu odabrali attribute id, ime i prezime iz tablice medicinske_sestre. Atribute ime i prezime smo spojili naredbom CONCAT u jedan atribut te smo ga preimenovali u Ime_i_prezime, te smo iz tablice pacijent uzeli attribute

ime, prezime. Iz sobe smo uzeli attribute broj_sobe i id_odjel. Te smo te 4 tablice spojili u FROM dijelu što bi odgovaralo njihovom kartezijevom produktu.
Da bi pridružili pacijente medicinskim sestrama i sobe te odjele pacijentima i medicinskim sestrama morali smo to sve napraviti preko tablice prijem.
Tako smo u WHERE dijelu upita odabrali da se tablica medicinske_sestre i prijem spoje po primarnom ključu iz tablice medicinske sestre, a sve ostale kombinacije se ne prikazuju. To smo isto napravili sa pacijentima i sobama koje smo također preko primarnog ključa povezali sa prijemom.
Na kraju smo to poredali prema atributu pacijent.id silazno.

REZULTAT:

ID	Ime_i_prezime	id	ime_pacijenta	prezime_pacijenta	broj_sobe	id_odjel
504	Žarka Stanić	420	Filip	Perišić	3	102
504	Žarka Stanić	419	Matija	Mladenović	3	102
504	Žarka Stanić	418	Boris	Vlašić	3	102
504	Žarka Stanić	417	Petar	Dragojević	5	102
504	Žarka Stanić	416	Toni	Belić	5	102
504	Žarka Stanić	415	Oliver	Kinić	5	102
504	Žarka Stanić	414	Borna	Karlović	5	102
502	Ivana Marić	413	Nora	Marjanović	4	102
502	Ivana Marić	412	Iris	Orbanić	4	102
502	Ivana Marić	411	Bruno	Marušić	4	102
502	Ivana Marić	410	Mia	Stepančić	4	102
504	Žarka Stanić	409	Stefan	Markulinčić	3	102
504	Žarka Stanić	408	Tomi	Ivković	3	102
502	Ivana Marić	407	Ema	Knežević	1	102
502	Ivana Marić	406	Paola	Marić	1	102
501	Miliana Milić	405	Lucas	Perić	9	101
501	Miliana Milić	404	Sebastijan	Milošević	9	101
503	Marko Marulić	403	Klara	Zenzerović	6	100
503	Marko Marulić	402	Lissa	Ivić	6	100
500	Ivana Ivić	401	Ivan	Rupčić	4	100
500	Ivana Ivić	400	Alen	Kolić	4	100

7.12. UPIT 12

-- Prikaz broja pacijenata dodijeljenih svakoj medicinskoj sestri gdje se može vidjeti opseg posla svake sestre te njeno opterećenje na odjelu. Prikazuju se medicinske sestre koje imaju broj pacijenata veći od 5 .

```
  CREATE VIEW Medicinari AS
SELECT  p.id_medicinske_sestre,m.ime,m.prezime, COUNT(*) as broj_pacijenata
FROM prijem as p, medicinske_sestre as m
     WHERE m.id=p.id_medicinske_sestre
     group by p.id_medicinske_sestre
```

```
    ORDER BY broj_pacijenata DESC;

-- prikaz broja pacijenata po medicinskoj sestri
SELECT * from Medicinari;

--prikaz medicinskih sestara koje imaju pridružen broj pacijenata koji je veći od 5
SELECT * from Medicinari WHERE broj_pacijenata>5;
```

OPIS UPITA:

Da bi znali id medicinske sestre u select dijelu odabiremo atribut id, te atribute ime i prezime, te unutar FROM dijela spajamo dvije tablice prijem i medicinske_sestre i dobivamo kartezijev produkt. Da bi dobiveni rezultat imao smisla te se medicinske sestre povezale sa tablicom prijem unutar odabiremo uvjet WHERE, gdje id medicinske sestre odgovara id medicinske sestre unutar tablice prijem. Uz pomoć agregacijske funkcije COUNT zbrajamo sve medicinske sestre koje su grupirane po atributu id medicinske sestre.Poredali smo dobiveni rezultat uz pomoć naredbe ORDER BY silazno prema broju pacijenata.

Tako dobivenu tablicu smo sporemili kao pogled te je sada unutar nje moguće doći do podatka koliko je medicinska sestra obavila prijema i koji su pacijenti o kojima ona brine.

Sve sestre koje brinu o više od 5 pacijenata trebalo bi rateretiti.

REZULTAT:

- Prikaz pogleda Medicinari

id_medicinske_sestre	ime	prezime	broj_pacijenata
504	Žarka	Stanić	9
502	Ivana	Marić	6
500	Ivana	Ivić	2
501	Miliana	Milić	2
503	Marko	Marulić	2

- Prikaz medicinskih sestara koje su upisale i brinu se za više od 5 pacijenata

id_medicinske_sestre	ime	prezime	broj_pacijenata
504	Žarka	Stanić	9
502	Ivana	Marić	6

7.13. UPIT 13

-- Cilj nam je ovim upitom bio prikazati broj zaposlenih po godinama starosti, te izračunati prosječnu starost osoblja, najstariju i najstarije osobe, prikaz zaposlenika koji imaju više godina od prosječne starosti zaposlenih.

```
CREATE VIEW godine AS
SELECT *, CURDATE() as sadasnji_datum, TIMESTAMPDIFF(YEAR, datum_rodenja,
CURDATE())AS age FROM doktor
UNION
SELECT*, CURDATE() as sadasnji_datum,  TIMESTAMPDIFF(YEAR, datum_rodenja,
CURDATE())AS age FROM medicinske_sestre
ORDER BY datum_rodenja DESC;


-- koliko g ima najstariji zaposlenik
SELECT  MAX(age)
  FROM godine;
-- broj zaposlenika
SELECT  COUNT(*) as broj_zaposlenika
  FROM godine;
-- godine najmlađeg zaposlenika
SELECT MIN(age) FROM godine;
--prosječan broj godina svih zaposlenih
SELECT AVG(age) FROM godine;
-- PRVA tri najstarija zaposlenika unutar bolnice
SELECT * FROM GODINE ORDER BY age DESC LIMIT 3 ;

-- saznaj koji zaposlenici imaju više godina od prosječne dobi
SELECT *
  FROM godine
  WHERE age >(SELECT AVG(age) FROM godine);
```

OPIS UPITA

Unutar SELECT dijela smo odabrali sve iz tablica doktor i medicinske_sestre te smo još dodali funkciju CURDATE() koja vraća sadašnji datum i nju smo usporedili sa datumom rođenja sa funkcijom TIMESTAMPDIF. To smo napravili za obije tablice te smoih tada povezali naredbom UNION.

Tu tablicu smo spremili kao pogled te nad njom izvršili upite MAX da saznamo broj godina najstarijeg zaposlenika, uz pomoć COUNT funkcije smo pobrojali broj zaposlenika, uz pomoć MIN funkcije saznali smo broj godina najmlađeg zaposlenika, uz pomoć funkcije SUM saznali smo prosječan broj godina unutar tablice godine. Da bi saznali koji zaposlenici imaju više od prosječne dobi izvršili smo podupit gdje smo prvo našli prosječnu dob pa odabrali uvjet gdje se prikažu zaposlenici sa većim brojem godina.

REZULTATI PRETRAŽIVANJA:

- Prikaz zaposlenika prema broju godina

id	ime	prezime	datum_rođenja	id_odjel	age
503	Marko	Marulić	2000-03-13 00:00:00	100	22
501	Miliana	Milić	1999-02-11 00:00:00	101	23
502	Ivana	Marić	1998-02-01 00:00:00	102	24

500	Ivana	Ivić	1992-02-11 00:00:00	100	30
504	Žarka	Stanić	1992-02-05 00:00:00	102	30
207	Božidarka	Obad	1990-06-03 00:00:00	100	31
208	Jelena	Debeljak	1988-03-28 00:00:00	102	34
203	Jasmina	Mejak	1985-06-03 00:00:00	101	36
201	Noris	Grubor	1982-05-03 00:00:00	100	40
200	Krešimira	Paspalj	1978-01-11 00:00:00	102	44
206	Gabrijel	Popović	1976-05-01 00:00:00	101	46
202	Šime	Ljubić	1973-12-09 00:00:00	101	48
204	Mauro	Tomasov	1970-07-24 00:00:00	100	51
209	Rudolf	Kolar	1967-09-16 00:00:00	101	54
205	Jan	Nikolić	1962-08-22 00:00:00	100	59

- Prikaz broja godina najstarijeg zaposlenika

MAX(age)
59

- Prikaz broja zaposlenika

broj_zaposlenika
15

- Godine najmlađeg zaposlenika

MIN(age)
20

- Prosječan broj godina

AVG(age)
38.1333

- Prva tri najstarija zaposlenika unutar bolnice

id	ime	prezime	datum_rođenja	id_odjel	age
205	Jan	Nikolić	1962-08-22 00:00:00	100	59
209	Rudolf	Kolar	1967-09-16 00:00:00	101	54
204	Mauro	Tomasov	1970-07-24 00:00:00	100	51

- Zaposlenici koji imaju više godina od prosječne dobi zaposlenika

id	ime	prezime	datum_rođenja	id_odjel	sadasnji_datum	age
201	Noris	Grubor	1982-05-03 00:00:00	100	2022-05-29	40
200	Krešimira	Paspalj	1978-01-11 00:00:00	102	2022-05-29	44
206	Gabrijel	Popović	1976-05-01 00:00:00	101	2022-05-29	46
202	Šime	Ljubić	1973-12-09 00:00:00	101	2022-05-29	48
204	Mauro	Tomasov	1970-07-24 00:00:00	100	2022-05-29	51
209	Rudolf	Kolar	1967-09-16 00:00:00	101	2022-05-29	54
205	Jan	Nikolić	1962-08-22 00:00:00	100	2022-05-29	59

7.14. UPIT 14

```
-- prikaži vrstu lijeka i proizvođača lijeka kojeg je koristio Alen Kolić
```

Ovaj upit prikazuje vrstu lijeka i proizvođača lijeka kojeg je koristio pacijent sa poznatim imenom i prezimenom (Alen Kolić). Pomoću SELECT naredbe dohvatili smo iz tablice lijek naziv lijeka i proizvođača lijeka. Napravio sam INNER JOIN između tri tablice (lijek, terapija i pacijent). Tablica terapija nam služi jer povezuje tablicu lijek i tablicu pacijent zato što sadrži njihove id-eve. I na kraju rezultat se filtrira na temelju imena i prezimena zadanog pacijenta koristeći WHERE.

```
SELECT lijek.vrsta, lijek.proizvodac
    FROM lijek
    INNER JOIN terapija
    ON lijek.id = terapija.id_lijek
    INNER JOIN pacijent
    ON terapija.id_pacijent = pacijent.id
    WHERE ime = 'Alen' AND prezime = 'Kolić';
```

REZULTAT:

vrsta	proizvodac
Antibiotik	Bayer

7.15. UPIT 15

```
-- prikaži sos_kontakte i pacijente koji imaju isto prezime
```

Pomoću SELECT naredbe dohvatio sam imena i prezimena iz tablice sos_kontakt i sva imena i prezimena iz tablice pacijent. Zatim u FROM djelu koristio sam LEFT OUTER JOIN između tablice sos_kontakt i tablice pacijent preko prezimena koristeći naredbu USING.

```
SELECT sos_kontakt.ime, sos_kontakt.prezime, pacijent.ime, pacijent.prezime
      FROM sos_kontakt LEFT OUTER JOIN pacijent USING(prezime);
```

REZULTAT:

ime	prezime	ime	prezime
Zora	Kunstl	NULL	NULL
Stanko	Sutarić	NULL	NULL
Issa	Biševac	NULL	NULL
Elenora	Delfar	NULL	NULL
Aden	Kotolaš	NULL	NULL
Vojmil	Novaković	NULL	NULL
Ljubica	Topić	NULL	NULL
Alan	Bačić	NULL	NULL
Silvana	Cindrić	NULL	NULL
Vilim	Kovačević	NULL	NULL
Ranka	Marković	NULL	NULL
Draženka	Čerkez	NULL	NULL
Senka	Jelić	NULL	NULL
Matej	Herceg	NULL	NULL
Melita	Petrović	NULL	NULL
Renata	Lončar	NULL	NULL
Marica	Kovač	NULL	NULL
Tomica	Radić	NULL	NULL
Dragica	Marušić	Bruno	Marušić
Lucija	Anić	NULL	NULL

7.16. UPIT 16

```
--prikaži ime, prezime i datum rodenja najstarije osobe kojoj je dijagnosticirana 'Angina pectoris'
```

Ovaj upit nam služi kako bi saznali ime, prezime i datum_rodenja osobe koja je najstarija i dijagnosticirana joj je 'Angina pectoris'. Unutar SELECT naredbe dohvatio sam iz tablice pacijenta ime, prezime i datum_rodenja. Napravio sam INNER JOIN između tri tablice (pacijent, prijem i dijagnoza). Tablica prijem nam služi jer povezuje tablicu pacijent i tablicu

dijagnoza zato što sadrži njihove id-eve. Zatim sam pomoću WHERE naredbe selektirao traženu dijagnozu. Koristeći ORDER BY i ASC sam poredao da najstarija osoba bude prva i pomoću naredbe LIMIT 1 dobio sam traženi rezultat.

```
SELECT pacijent.ime, pacijent.prezime, datum_rodenja
   FROM pacijent
   INNER JOIN prijem
   ON pacijent.id = prijem.id_pacijent
   INNER JOIN dijagnoza
   ON prijem.id_dijagnoza = dijagnoza.id
   WHERE naziv = 'Angina pectoris'
   ORDER BY datum_rodenja ASC
   LIMIT 1;
```

REZULTAT:

ime	prezime	datum_rodenja
Ivan	Rupčić	1962-02-23 00:00:00

7.17. UPIT 17

```
-- prikaži sve zaposlene medicinske sestre i doktore unutar bolnice. Unutar pogleda moguće je izabrati koji radnici rade na kojim odjelima, koliko je ukupno radnika na svakom odjelu.
```

Unutar SELECT dijela odabrao sam iz tablice medicinske_sestre atribute id, ime, prezime, a iz tablice odjel naziv odjela, te sam spojio tablicu odjel sa tablicom medicinske_sestre pomoću funkcije INNER JOIN koja je povezala atribut id iz tablice odjel i atribut id_odjel iz tablice medicinske sestre koji služi kao strani ključ. Isto sam napravio i za liječnike te sam ih povezao sa naredbom UNION. Dobivene retke smo dobili ispod medicinskih sestara. To sam spremio kao pogled te sam nad tim pogledom izvršio upit kako bi saznao zaposlenike odjela Kardiologije.

```
 CREATE VIEW zaposleni_na_odjelu AS
SELECT m.id, m.ime, m.prezime, o.naziv FROM medicinske_sestre as m
INNER JOIN odjel as o ON o.id=m.id_odjel
UNION
SELECT d.id, d.ime,d.prezime, o.naziv FROM doktor as d
INNER JOIN odjel as o ON o.id= d.id_odjel
ORDER BY naziv DESC;

-- odaberi sve koji su zaposleni na odjelu kardiologije
SELECT *
FROM zaposleni_na_odjelu
WHERE naziv='Kardiologija';
```

REZULTAT

- tablica zaposlenih po odjelima poredanih po odjelima

id	ime	prezime	naziv
502	Ivana	Marić	Ortopedija
504	Žarka	Stanić	Ortopedija
200	Krešimira	Paspalj	Ortopedija
208	Jelena	Debeljak	Ortopedija
500	Ivana	Ivić	Kardiologija
503	Marko	Marulić	Kardiologija
201	Noris	Grubor	Kardiologija
204	Mauro	Tomasov	Kardiologija
205	Jan	Nikolić	Kardiologija
207	Božidarka	Obad	Kardiologija
501	Miliana	Milić	Intenzivno liječenje
202	Šime	Ljubić	Intenzivno liječenje
203	Jasmina	Mejak	Intenzivno liječenje
206	Gabrijel	Popović	Intenzivno liječenje
209	Rudolf	Kolar	Intenzivno liječenje

- Zaposleni na kardiologiji

id	ime	prezime	naziv
500	Ivana	Ivić	Kardiologija
503	Marko	Marulić	Kardiologija
201	Noris	Grubor	Kardiologija
204	Mauro	Tomasov	Kardiologija
205	Jan	Nikolić	Kardiologija
207	Božidarka	Obad	Kardiologija

8. ZAKLJUČAK

Smatramo da naša baza podataka dobro funkcionira u ovakvom obliku ali isto tako svi znamo da ima još mnogo mjesta za napredak. Sama tema bolnica je dosta opširna stoga smo koristili bazične tablice i podatke. Usprkos svemu tome smatramo da smo kreirali solidnu bazu podataka jedne bolnice te ostvarili sve ciljeve koji su bili definirani u ovom projektnom zadatku.

Za izradu ovog projekta koristili smo:

- za komunikaciju: Discord

- za pisanje SQL skripte: MySQL Workbench

- za kreiranje ER dijagrama: Lucidchart

- za izradu dokumentacije: GitHub