Univerzitet u Kragujevcu

Fakultet inzenjerskih nauka

BAZE PODATAKA

Projektni zadatak:

RAZVOJ BAZA PODATAKA INFORMACIONOG SISTEMA ZA APOTEKU-IZDAVANJE LEKOVA NA RECEPT

PROFESOR:

Prof. dr Milan D. Erić

STUDENT:

Đorđe Karišić 657/2019

_____

Kragujevac, . . 2022.

SADRŽAJ

1. REZIME

U ovom projektnom zadatku biće objašnjena potreba, i izvršeno modelovanje baze podataka koja odgovara zahtevima skladištenja informacija jedne apoteke.

Struktura baze će predstavljati odnos potrošača-kupca sa radnikom- prodavcem i doktorom, kako je realizovan neki lek koji taj potrošač može kupiti, kao i koje su predispozicije kako bi taj lek mogao bio kupljen.

Pre svega, potrebno je prikazati i objasniti ER i EER dijagram sistema koji je napravljen korišćenjem softverskog okruženja *MySQL Workbench* verzije *8.0* i trial verzije onlajn programa *Lucidchart,* **koja ograničava broj objekata postavljenih na šemi***,* nakon čega će se iz njega izvesti ekvivalentna šema relacione baze podataka, kao i SQL kod koji predstavlja ovu bazu.

2. OPIS POSLOVA I INFORMACIONE POTREBE SISTEMA

Kako bi se omogućio nesmetan, precizan i brz rad jedne apoteke, potrebno je da se obezbedi adekvatna realizacija skladištenja, manipulacije i prikaza podataka.

Zbog ogromne količine podataka koji se javljaju u apotekama, računar je očigledna opcija za skladištenje i održavanje podataka. Efikasnost ovakvog pristupa leži u činjenici da je gotovo nemoguće za zaposleno lice apoteke da ručno barata informacijama, traži pojedine redove po vrednostima i uređuje iste.

Kako bi mogli da iskoristimo računar u ovom sistemu, potrebno je da ga razumemo. Suština ovog projekta jeste predstavljanje model apoteke koja sadrži svoje kupce, zaposlene, inventar, lekove u više različitih pakovanja, inventar svih lekova na stanju, informaciju o tome koji zaposleni je kada i kome izdao lek, i tako dalje. Ovo nam govori da su potrebe realnog sistema jasne, potrebna je dobro struktuirana baza podataka.

3. RELEVANTNI DOKUMENTI U POSMATRANOM REALNOM SISTEMU

Apoteka je zdravstvena ustanova koja nabavlja, skladišti lekove i njima snabdeva stanovništvo.

U apoteci zaposleni farmaceut igra ulogu prodavca, koji kupcu može izdati lek sa (slučaj ovog projekta) ili bez recepta.

Apoteka ima svoj inventar koji sadrži lekove i njihovu količinu. Jedan lek može imati više ambalaža, i svaki pripada nekoj klasi, tj. tipu leka (tableta, kapi, inhalator, sirup...).

U slučaju ovog modela, pre izdavanja leka kupcu, farmaceut mora da ima postojeću tabelu podataka recept(kupac,lek) preko koje može proveriti da li je kupcu izdat recept od strane njegovog lekara, kupac može da naruči (kupi) lek tek kada u sistemu postoji informacija o tome o kojem se leku radi, o kojoj ambalaži, ko je izdao recept, kada je recept izdat, i koji zaposleni je obavio ovaj zadatak.

Svaka osoba ovog sistema deli pojedine atribute sa ostalima, ime, prezime, datum rođenja, username i password kako bi pristupili sistemu. Lekari i doktori imaju posebno polje licenca lekarske prakse, radi preciznije identifikacije.

Može se primetiti da su zadaci, kao što je prethodno navedeno, projektovanja i implementacije baze podataka neminovni.

4. ER DIJAGRAM, ENTITETI, ATRIBUTI I VEZE MEĐU ENTITETIMA

4.1. ENTITETI

Prvi korak modelovanja realnog sistema jeste identifikacija entiteta koji učestvuju u modelu, i njihovih atributa.

4.1.1. OSOBA

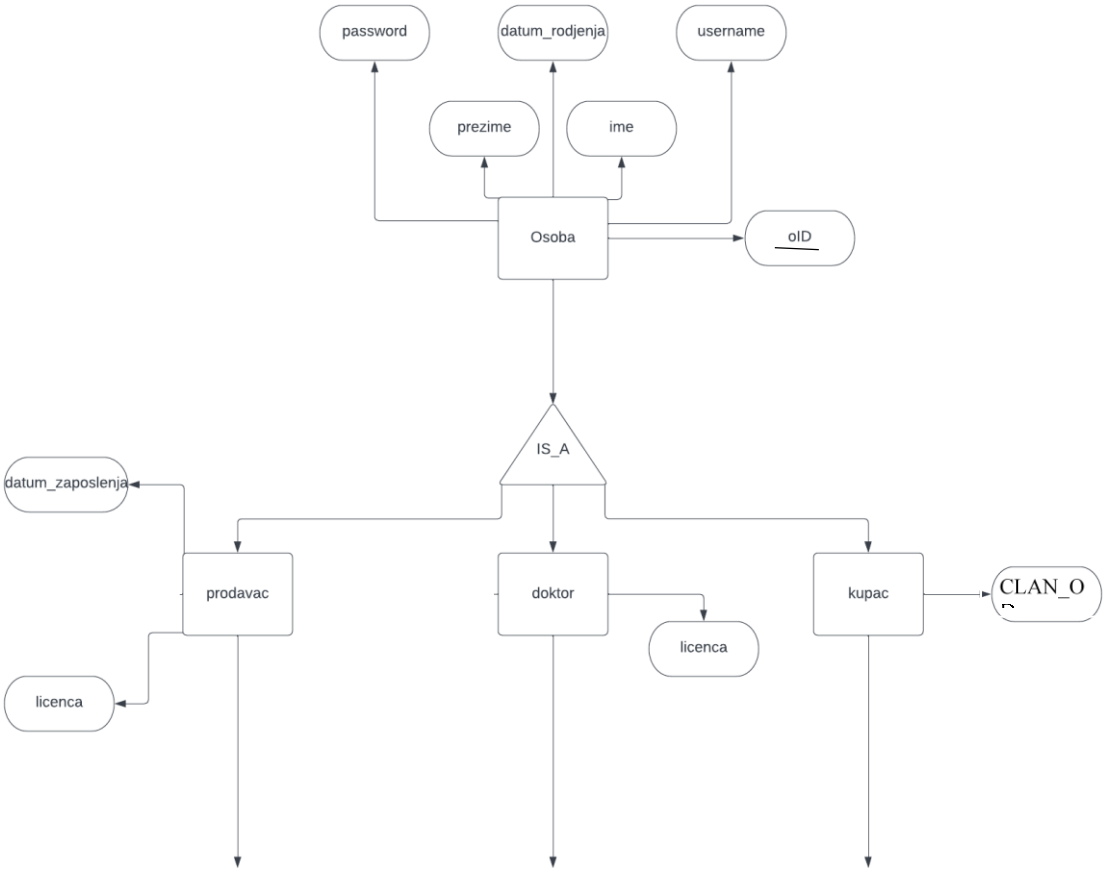
Entitet **Osoba** sadrži:

- ***Identifikator oID***
- ***Ime***
- ***Prezime***
- ***Datum rođenja***
- ***Korisničko ime***
- ***Šifra***

Ovaj entitet sadrži ima sledeće podidentitete, koji nose dodatne atribute:

- ***Prodavac***
- *Datum zaposlenja*
- *Licenca*
- ***Doktor***
- *Licenca*
- ***Kupac***
- *Clan_Od*

Specijalizacijom entiteta Osoba dobićemo entitete Prodavac, Kupac I Doktor.Ovaj pristup je pogodan iz razloga što prodavac, doktor I kupac dele pojedine atribute, kako bismo izbegli ponavljanje istih, dovoljno je da napravimo ovu specijalizaciju.



ER dijagram tipova entiteta Osoba

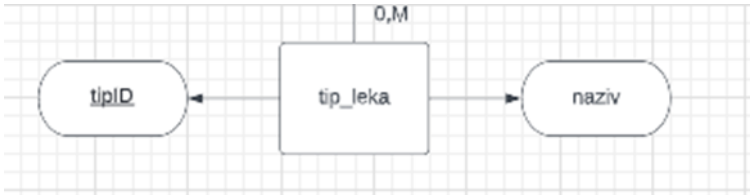
4.1.2. TIP LEKA

Entitet **tip leka** sadrži sledeće atribute:

- ***Identifikator tipID***
- ***Naziv***

Ovaj entitet predstavlja tipove lekova, npr. tableta, kapsula, kapi...

Neophodan je iz razloga što svaki lek mora pripadati nekom tipu, I radi lakše klasifikacije, pravimo ovaj entitet.

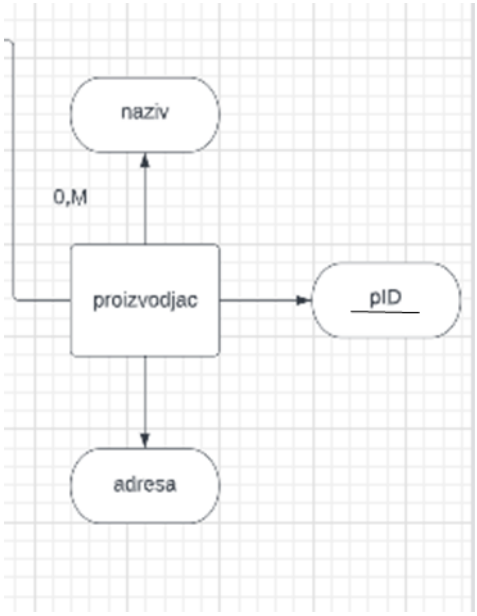


4.1.3. PROIZVOĐAČ

Entitet **proizvođač** sadrži sledeće atribute:

- ***Identifikator pID***
- ***Ime***
- ***Adresa***

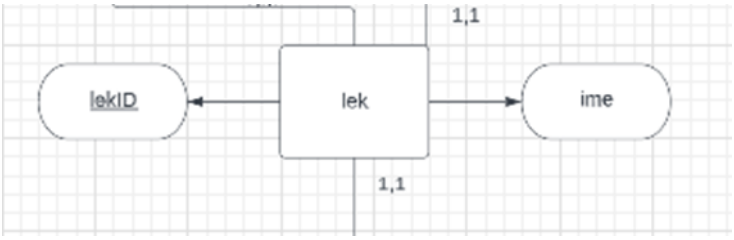
Svaki lek pre prodaje, prvenstveno mora biti napravljen od strane nekog proizvođača. Kako bi imali informaciju o kojem proizvođaču se radi, kao i gde se nalazi, kreiramo ovaj identitet sa ovim atributima.



4.1.4 LEK

Entitet lek ima sledeće atribute:

- **Identifikator LekID**
- **Ime**

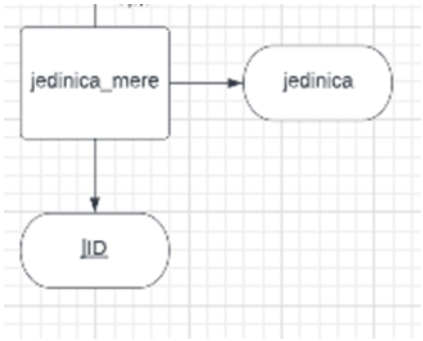


4.1.5. JEDINICA MERE

Entitet **Jedinica mere** sadrži atribute:

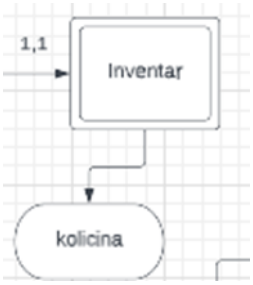
- ***Identifikator jID***
- ***Jedinica***

Kombinacija jID, Jedinica nam govore o mogućim ambalažama leka. Atribut jedinica može imati oblik "*200 ml*" ili "*10 tableta*" itd. Ovaj entitet nam pomaže u kreiranju mogućnosti da jedan lek može imati više ambalaža.



4.1.6. INVENTAR

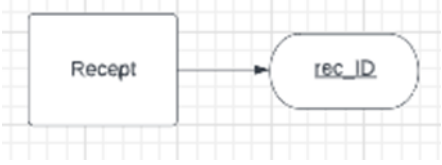
Entitet inventar je slabi entitet koji zavisi od ključa Gerunda **ambalaža_leka**. Ima svoj atribut količina.



4.1.7. RECEPT

Entitet **recept** je entitet, koji ima smisla tek nakon povezivanja sa parametrima više drugih entiteta.

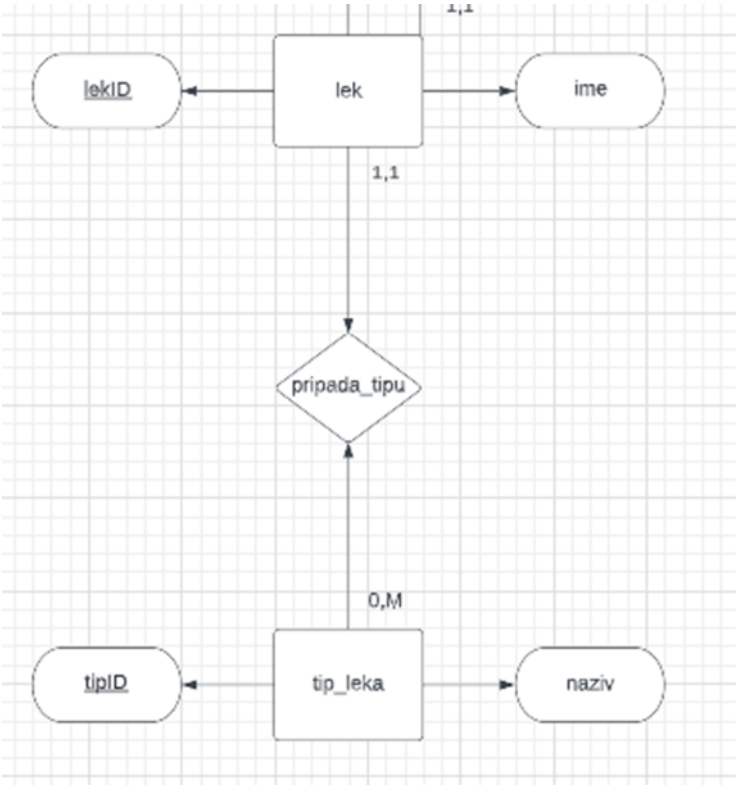
Od svojih atributa ima jedino **Identifikator recID.**



4.2. VEZE

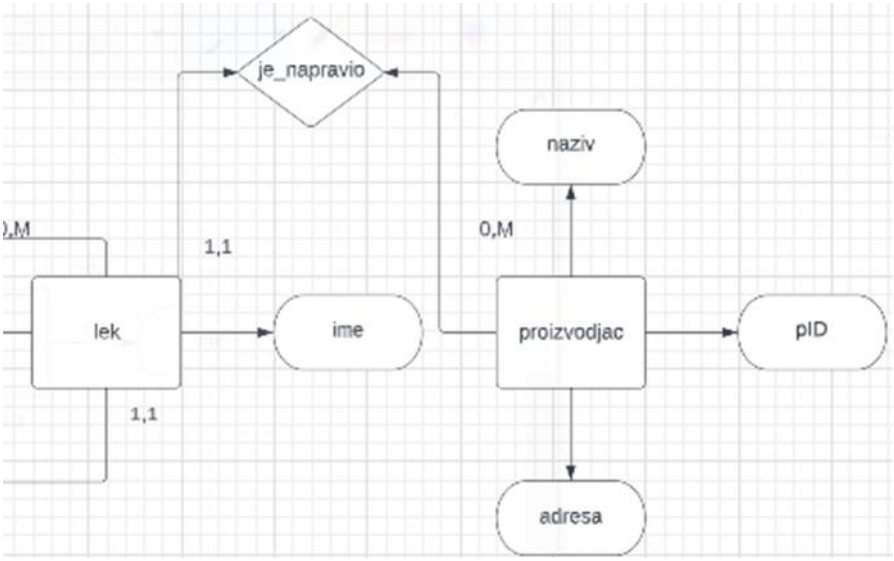
4.2.1. TIP_LEKA-LEK

Jedan lek pripada tačno jednom tipu leka. Jednom tipu leka može pripadati više lekova. Kardinalnost ove veze je **1:M**.



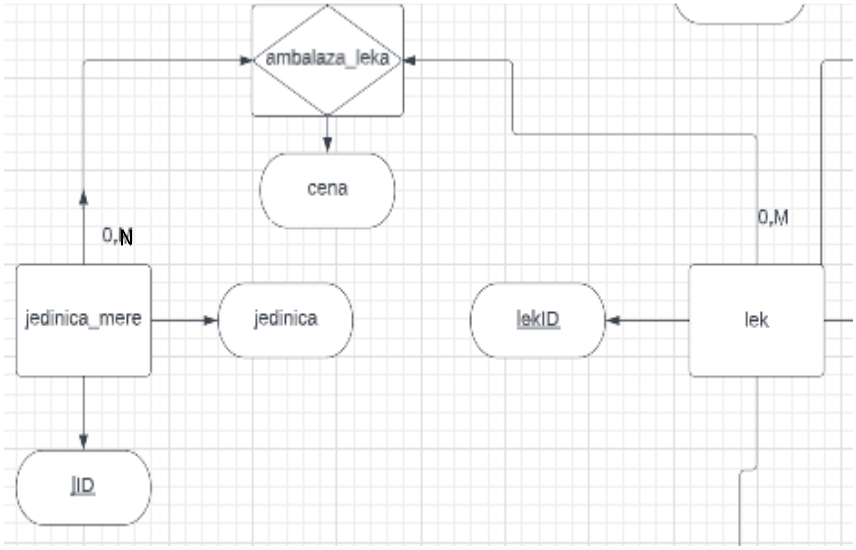
4.2.2. LEK-PROIZVOĐAČ

Svaki lek je napravljen od strane tačno jednog proizvođača. Svaki proizvođač može napraviti od **0 do M** lekova.



4.2.3. LEK-JEDINICA_MERE

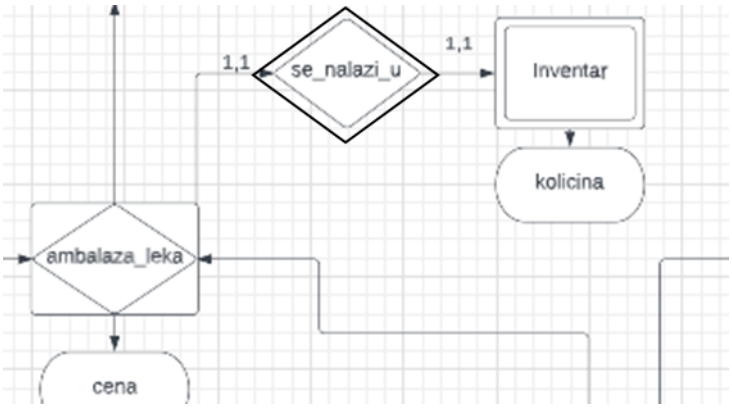
Veza **lek-jedinica_mere** je realizovana preko gerunda ambalaza_leka, kao što je prikazano na sledećoj slici:



Za realizaciju veze između entiteta jedinica_mere i lek, koristimo gerund **ambalaza_leka** koji ima svoj atribut cena, i čiji je ključ kombinacija ključeva **jedinica_mere** i lek.Jedna jedinica mere se može koristiti za više lekova. Jedan lek može postojati u više varijanti što se tiče jedinice mere, npr *50ml, 100ml*, itd, a može biti ni u jednoj, ako još nije upakovan.

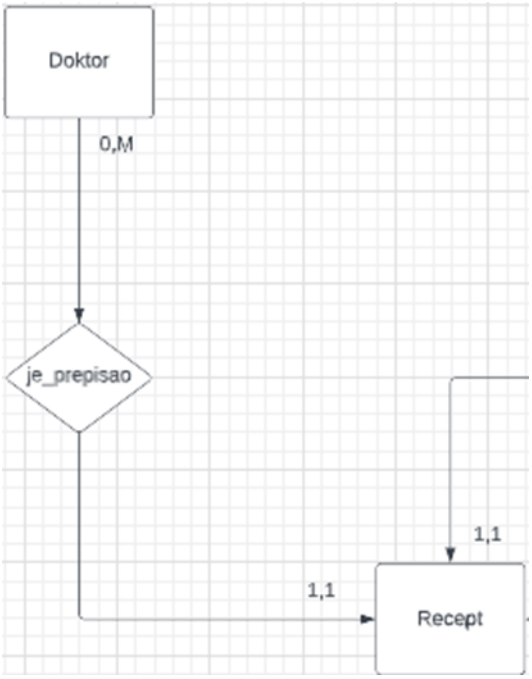
4.2.4. INVENTAR-AMBALAZA_LEKA

Jedna ambalaža leka se može naći samo jednom u inventaru, jedan red inventara sadrži samo jedan lek. Razlog zašto je izabrana ova realizacije je zato što Inventar ima svoj atribut količina, tako da je dovoljno da se jedna ambalaža leka pojavi samo jednom unutar njega, uz element količina.



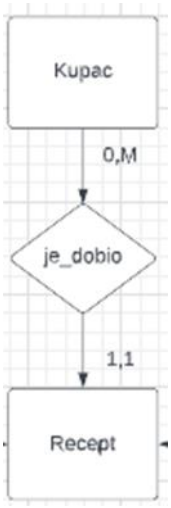
4.2.5. DOKTOR-RECEPT

Jedan doktor može prepisati više recepata, ili ni jedan. Jedan recept može prepisati samo jedan doktor.



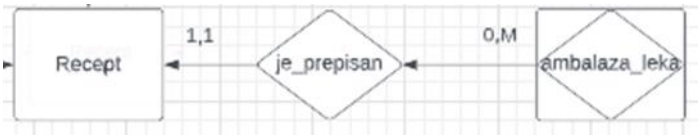
4.2.6. KUPAC-RECEPT

Jedan kupac može dobiti više recepata, ili ni jedan. Jedan recept je prepisan samo jednom kupcu. Ova veza dozvoljava kupcu da dobije više recepata od doktora.



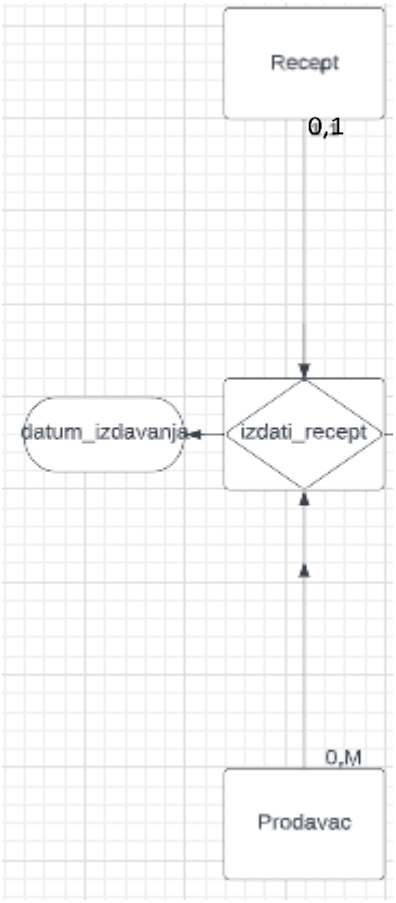
4.2.7. RECEPT-AMBALAZA_LEKA

Jedna ambalaža leka može biti prepisana u više recepata. Jedan recept sadrži samo jednu ambalažu leka.



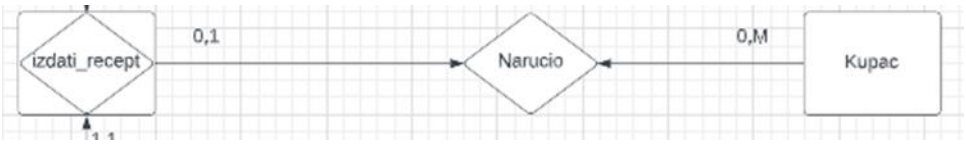
4.2.8. PRODAVAC-RECEPT

Jedan farmaceut (prodavac, zaposleni) može dodati više izdatih recepata, ili ni jedan. Jedan recept može biti izdat od strane ili jednog ili ni jednog prodavca. Jedan prodavac može izdati više recepata, ili ni jedan. Ovaj gerund **izdati_recept** je bitan, jer ukazuje na to da je neki recept, koji je prepisan od strane nekog doktora, **prihvaćen i izdat** od strane zaposlenog za kupca. Izdati_recept ima svoj atribut datum_izdavanja.

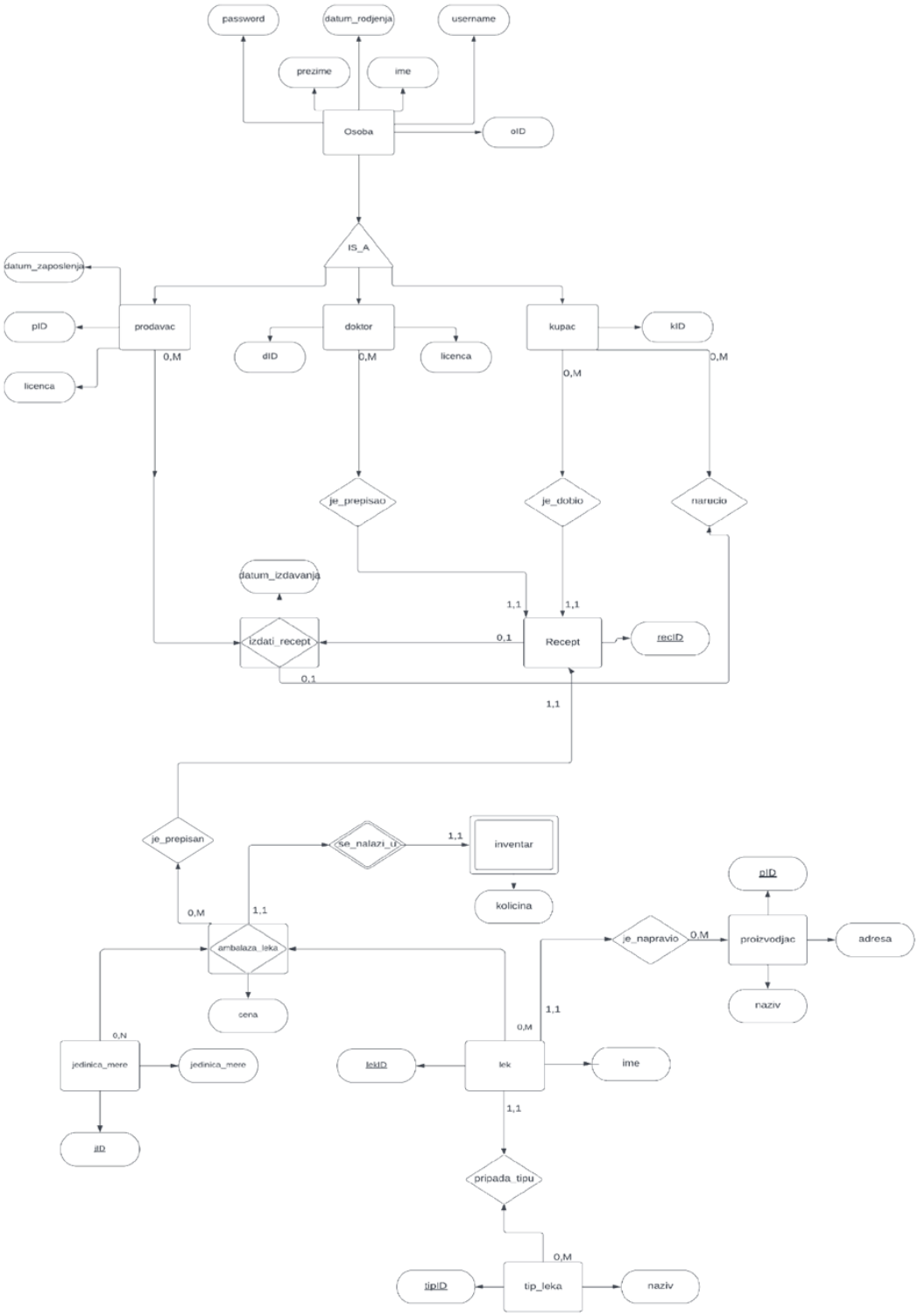


4.2.9. IZDATI_RECEPT-KUPAC

Tek kada postoji izdati recept za neki lek, kupac može da kupi. Tim pristupom osiguravamo da jedan kupac ne može da zloupotrebi jedan recept, i da svaki prepisan recept mora biti primećen i odobren od strane farmaceuta.



4.3. KOMPLETAN ER DIJAGRAM



5. LOGIČKA ŠEMA I MEĐURELACIONA OGRANIČENJA RELACIONE BAZE PODATAKA

Kako bismo dobili logičku šemu relacione baze podataka, potrebno je prevesti sve entitete i veze iz modela objekti-veze u njihov ekvivalentni oblik u relacionom modelu. Elementi će biti prevođeni sledećim redosledom:

- **Entiteti**
- **Identifikacione veze slabih entiteta**
- **Veze nasleđivanja**
- **Gerundi**
- **Preostale veze**

5.1. PREVOĐENJE ENTITETA

Svi tipovi entiteta postaju nezavisne šeme relacija. Ime tipa entiteta postaje ime šeme relacije. Obeležja tipa objekta su obeležja šeme relacije. Identifikator entiteta postaje primarni ključ šeme relacije.

- **osoba(<u>oID</u>, ime, prezime, datum_rodjenja, username, password)**
- **kupac(dan_od)**
- **doktor(licenca_br)**
- **prodavac(licenca_farmaceut, datum_zaposlenja)**
- **tip_leka(<u>tipID</u>, naziv)**
- **proizvodjac(<u>pID</u>, ime_pr, adresa)**
- **lek(<u>lekID</u>, ime_leka)**
- **jedinica_mere(<u>jID</u>, jedinica)**
- **Inventar(kolicina)**
- **recept(<u>recID</u>)**

5.2. IDENTIFIKACIONE VEZE SLABIH ENTITETA

Od slabih entiteta, u ovom sistemu postoji samo Inventar.

Inventar zavisi od ključa gerunda ambalaza_leka.

Gerund ambalaza_leka će biti kasnije objašnjen. Pod pretpostavkom da smo preveli gerund ambalaza_leka i da smo dobili:

- Ambalaza_leka(<u>lek_lekID,jedinica_mere_jID,</u>cena)

Može se odrediti slabi entitet Inventar, koji nasleđuje ključ gerunda ambalaza_leka:

- Inventar(<u>ambalaza_leka_lek_lekID,ambalaza_leka jedinica_mere_jID,</u>kolicina)

5.3. VEZE NASLEĐIVANJA

Tipovi podentiteta nasleduju identifikator tipa nadentiteta. U ovom sistemu, podentiteti će naslediti ključ od nadentiteta.

Prevedeni entiteti su opisani na sledeći način:

- kupac(oID,dan_od)
- doktor(oID,licenca_br)
- prodavac(oID,licenca_farmaceut,datum_zaposlenja)

5.4. GERUNDI

Gerund je tip entiteta koji se ponaša kao veza, ili tip veze koja može da učestvuje u drugim vezama (ponaša se kao entitet). U ovom sistemu postoje gerundi **izdati_recept** i **ambalaza_leka.**

Izdati_recept povezuje recept i prodavca. Govori nam o tome da jedan prodavac može da izda više recepata ili ni jedan, a jedan recept može da izda jedan, ili ni jedan prodavac. Prema tome, veza (0,M)-(0,1) se po pravilu prevodi:

- izdati_recept(prodavac_oID, recept_recID,datum_izdavanja)

Na sličan način, prevodimo i ambalaza_leka; Jedan lek može imati više mernih jedinica, i jedna merna jedinica može označavati više lekova.

- ambalaza_leka(lek_lekID, jedinica_mere_jID,cena)

5.5. PREOSTALE VEZE

- **tip_leka-pripada_tipu-lek:**

Jedan lek pripada jednom i samo jednom tipu leka, a jedan tip leka može imati više lekova ili ni jedan, što znači da ključ tip_leka dodajemo atributu lek.

- lek(lekID, ime_leka,**tipID**)

- **lek-je_napravio-proizvodjac**

Jedan proizvođač može napraviti više lekova, ili ni jedan, ali jedan lek može napraviti jedan i samo jedan proizvođač, kao u prošlom primeru, ključ atributa proizvodjac dodajemo atributu lek.

- Lek (lekID, ime_leka,**tipID**,**pID**)

- **doktor-je_prepisao-recept**

Jedan doktor može da prepiše više recepata ili ni jedan, ali jedan recept može prepisati jedan i samo jedan doktor.

- recept(recID,**oID**)

- **kupac-je_dobio-recept**

Isti princip.

- recept(recID,**doktor_oID,kupac_oID**)

- **lek-je_prepisan-recept**

Jedan lek može biti prepisan u više recepata, a ne mora ni u jednom, dok jedan recept mora da sadrži jedan i samo jedan lek.

- recept(recID,**doktor_oID,kupac_oID,ambalaza_leka_lek_lekID,ambalaza_leka_jedinica_mere_jID**)

- **kupac-narucio-izdati_recept**

Jedan kupac može naručiti više recepata, ili ni jedan, dok jedan recept može naručiti ili jedan, ili ni jedan kupac. Za ovu vezu pravimo novu relaciju **narucio**.Vezu (0,1)-(0,M) prevodimo tako što napravimo novu relaciju koja će naslediti ključne atribute atributa u vezi.

- narucio(izdati_recept_recept_recID,kupac_oID)

Ključ ove relacije je atribut kod kojeg je GG=1.

5.6. KORIGOVANJE RELACIJA RADI POSTOVANJA NORMALNIH FORMI

Svi atributi svih entiteta imaju atomske vrednosti, što znači da sve relacije poštuju 1NF.

Svi atributi su potpuno funkcionalno zavisni od njihovog primarnog ključa, sve relacije su u 2NF npr. Ambalaza_leka(lekID,jID), dve instance ili reda u tabeli ambalaža mogu imati isti lek, a arazličitu jedinicu mere, i suprotno, što znači da jedino kombinacija ta dva atributa potpuno određuje instancu.

U ovoj relaciji nema tranzitivnih funkcionalnih zavisnosti, što je najbolje objasniti na primeru entiteta Recept. Recept se sastoji od svog primarnog ključa recID, i atributa ključa ambalaže lekID+jID, doktor_oID i kupac_oID. Ključ recID određuje ove atribute

u ovom entitetu, dok one, iako su primarni ključevi za druge entitete, u ovom ne određuju ništa, što znači da je ključ recID potpuni ključ ove relacije, i da svi elementi zavise isključivo od njega, i ni od jednog elementa više.

5.7. MEĐURELACIONA OGRANIČENJA

kupac[oID]⊆osoba[oID]

doktor[oID] ⊆osoba[oID]

prodavac[oID] ⊆osoba[oID]

lek[tipID] ⊆tip_leka[tipID]

lek[pID] ⊆proizvodjac[pID]

ambalaza_leka[lekID] ⊆lek[lekID]

ambalaza_leka[jID] ⊆jedinica_mere[jID]

inventar[lekID] ⊆ambalaza_leka[lekID]

inventar[jID] ⊆ambalaza_leka[jID]

recept[lekID] ⊆ambalaza_leka[lekID]

recept[jID] ⊆ambalaza_leka[jID]

recept[doktor_oID] ⊆doktor[oID]

recept[kupac_oID] ⊆kupac[oID]

izdati_recept[oID] ⊆prodavac[oID]

izdati_recept[recID] ⊆recept[recID]

narucio[oID] ⊆kupac[oID]

narucio[recID] ⊆izdati_recept[recID]

5.8. KOMPLETNA LOGIČKA ŠEMA

S={

Osoba(oID,ime,prezime,username,password,datum_rodjenja)

Kupac(oID,dan_od)

Prodavac(oID,datum_zaposlenja,licenca_farmaceut)

Doktor(oID,licenca_br)

Tip_leka(tipID,naziv)

Jedinica_mere(jID,jedinica)

Proizvodjac(pID,ime_p,adresa)

Lek(lekID,ime,tipID,pID)

Ambalaza_leka(lekID,jID,cena)

Recept(recID,lekID,jID,doktor_oID,kupac_oID)

Izdati_recept(recID,oID,datum_izdavanja)

Narucio(recID,oID)

Inventar(lekID,jID,kolicina)

}

I={

kupac[oID]⊆osoba[oID]

doktor[oID] ⊆osoba[oID]

prodavac[oID] ⊆osoba[oID]

lek[tipID] ⊆tip_leka[tipID]

lek[pID] ⊆proizvodjac[pID]

ambalaza_leka[lekID] ⊆lek[lekID]

ambalaza_leka[jID] ⊆jedinica_mere[jID]

inventar[lekID] ⊆ambalaza_leka[lekID]

inventar[jID] ⊆ambalaza_leka[jID]

recept[lekID] ⊆ambalaza_leka[lekID]

recept[jID] ⊆ambalaza_leka[jID]

recept[doktor_oID] ⊆doktor[oID]

recept[kupac_oID] ⊆kupac[oID]

izdati_recept[oID] ⊆prodavac[oID]

izdati_recept[recID] ⊆recept[recID]

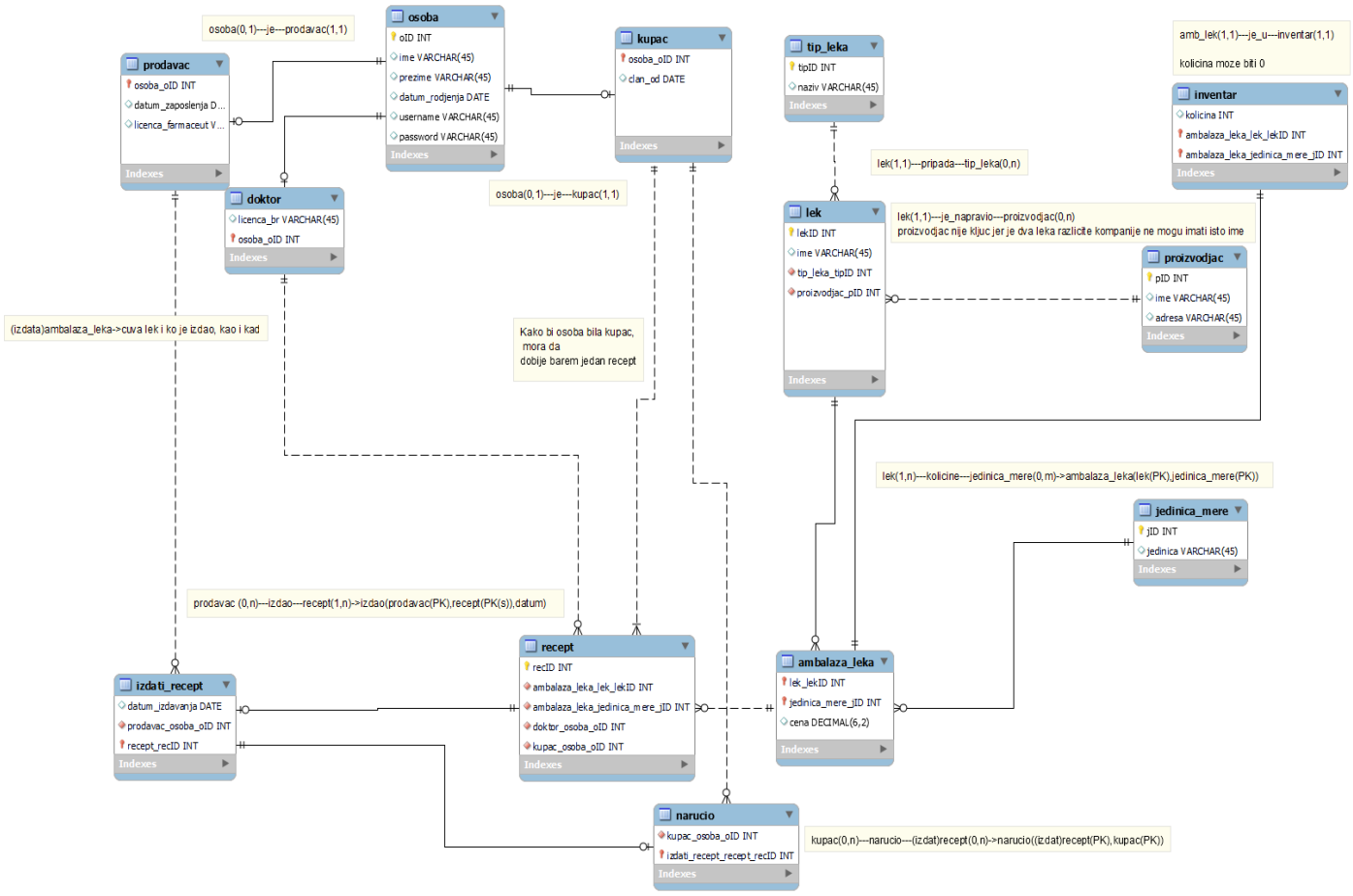
narucio[oID] ⊆kupac[oID]

narucio[recID] ⊆izdati_recept[recID]

}

6. FIZIČKA ŠEMA RELACIONE BAZE PODATAKA

Fizicka šema predstavlja najniži nivo apstrakcije podataka u bazi. Pomoću fizičke šeme se opisuje realan sistem. Fizička šema je identična logičkoj, sa par olakšavajućih promena. Fizička šema će biti predstavljena preko softverskog alata MySQLWorkbench.



7. *PROJEKTOVANA BAZA PODATAKA SA TESTNIM PODACIMA U SQL SERVERU*

```
-- MySQL Script generated by MySQL Workbench

-- Mon Jun 20 15:46:38 2022

-- Model: New Model    Version: 1.0

-- MySQL Workbench Forward Engineering


SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';


-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------


-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;


-- -----------------------------------------------------

-- Table `mydb`.`osoba`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `mydb`.`osoba` ;
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`osoba` (
  `oID` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `ime` VARCHAR(45) NULL,
  `prezime` VARCHAR(45) NULL,
  `datum_rodjenja` DATE NULL,
  `username` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  PRIMARY KEY (`oID`))
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`kupac`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`kupac` ;

CREATE TABLE IF NOT EXISTS `mydb`.`kupac` (
  `osoba_oID` INT UNSIGNED NOT NULL,
  `clan_od` DATE NULL,
  PRIMARY KEY (`osoba_oID`),
  INDEX `fk_kupac_osoba_idx` (`osoba_oID` ASC) ,
  CONSTRAINT `fk_kupac_osoba`
    FOREIGN KEY (`osoba_oID`)
    REFERENCES `mydb`.`osoba` (`oID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`prodavac`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`prodavac` ;

CREATE TABLE IF NOT EXISTS `mydb`.`prodavac` (
  `osoba_oID` INT UNSIGNED NOT NULL,
  `datum_zaposlenja` DATE NULL,
  `licenca_farmaceut` VARCHAR(45) NULL,
  PRIMARY KEY (`osoba_oID`),
  INDEX `fk_kupac_osoba_idx` (`osoba_oID` ASC) ,
  CONSTRAINT `fk_kupac_osoba0`
    FOREIGN KEY (`osoba_oID`)
    REFERENCES `mydb`.`osoba` (`oID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`tip_leka`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`tip_leka` ;

CREATE TABLE IF NOT EXISTS `mydb`.`tip_leka` (
  `tipID` INT NOT NULL,
```

```
  `naziv` VARCHAR(45) NULL,
  PRIMARY KEY (`tipID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`proizvodjac`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`proizvodjac` ;

CREATE TABLE IF NOT EXISTS `mydb`.`proizvodjac` (
  `pID` INT NOT NULL AUTO_INCREMENT,
  `ime` VARCHAR(45) NULL,
  `adresa` VARCHAR(45) NULL,
  PRIMARY KEY (`pID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`lek`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`lek` ;

CREATE TABLE IF NOT EXISTS `mydb`.`lek` (
  `lekID` INT NOT NULL AUTO_INCREMENT,
  `ime` VARCHAR(45) NULL,
  `tip_leka_tipID` INT NOT NULL,
  `proizvodjac_pID` INT NOT NULL,
```

```
  PRIMARY KEY (`lekID`),
  INDEX `fk_lek_tip_leka1_idx` (`tip_leka_tipID` ASC) ,
  INDEX `fk_lek_proizvodjac1_idx` (`proizvodjac_pID` ASC) ,
  CONSTRAINT `fk_lek_tip_leka1`
    FOREIGN KEY (`tip_leka_tipID`)
    REFERENCES `mydb`.`tip_leka` (`tipID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_lek_proizvodjac1`
    FOREIGN KEY (`proizvodjac_pID`)
    REFERENCES `mydb`.`proizvodjac` (`pID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`jedinica_mere`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`jedinica_mere` ;

CREATE TABLE IF NOT EXISTS `mydb`.`jedinica_mere` (
  `jID` INT NOT NULL AUTO_INCREMENT,
  `jedinica` VARCHAR(45) NULL,
  PRIMARY KEY (`jID`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`ambalaza_leka`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`ambalaza_leka` ;

CREATE TABLE IF NOT EXISTS `mydb`.`ambalaza_leka` (
  `lek_lekID` INT NOT NULL,
  `jedinica_mere_jID` INT NOT NULL,
  `cena` DECIMAL(6,2) NULL,
  PRIMARY KEY (`lek_lekID`, `jedinica_mere_jID`),
  INDEX `fk_ambalaza_leka_lek1_idx` (`lek_lekID` ASC) ,
  INDEX `fk_ambalaza_leka_jedinica_mere1_idx` (`jedinica_mere_jID` ASC) ,
  CONSTRAINT `fk_ambalaza_leka_lek1`
    FOREIGN KEY (`lek_lekID`)
    REFERENCES `mydb`.`lek` (`lekID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ambalaza_leka_jedinica_mere1`
    FOREIGN KEY (`jedinica_mere_jID`)
    REFERENCES `mydb`.`jedinica_mere` (`jID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`doktor`
-- -----------------------------------------------------
```

```
DROP TABLE IF EXISTS `mydb`.`doktor` ;

CREATE TABLE IF NOT EXISTS `mydb`.`doktor` (
  `licenca_br` VARCHAR(45) NULL,
  `osoba_oID` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`osoba_oID`),
  INDEX `fk_doktor_osoba1_idx` (`osoba_oID` ASC) ,
  CONSTRAINT `fk_doktor_osoba1`
    FOREIGN KEY (`osoba_oID`)
    REFERENCES `mydb`.`osoba` (`oID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`recept`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`recept` ;

CREATE TABLE IF NOT EXISTS `mydb`.`recept` (
  `recID` INT NOT NULL AUTO_INCREMENT,
  `ambalaza_leka_lek_lekID` INT NOT NULL,
  `ambalaza_leka_jedinica_mere_jID` INT NOT NULL,
  `doktor_osoba_oID` INT UNSIGNED NOT NULL,
  `kupac_osoba_oID` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`recID`),
```

```
  INDEX `fk_recept_ambalaza_leka1_idx` (`ambalaza_leka_lek_lekID` ASC,
`ambalaza_leka_jedinica_mere_jID` ASC) ,
  INDEX `fk_recept_doktor1_idx` (`doktor_osoba_oID` ASC) ,
  INDEX `fk_recept_kupac1_idx` (`kupac_osoba_oID` ASC) ,
  CONSTRAINT `fk_recept_ambalaza_leka1`
    FOREIGN KEY (`ambalaza_leka_lek_lekID` , `ambalaza_leka_jedinica_mere_jID`)
    REFERENCES `mydb`.`ambalaza_leka` (`lek_lekID` , `jedinica_mere_jID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_recept_doktor1`
    FOREIGN KEY (`doktor_osoba_oID`)
    REFERENCES `mydb`.`doktor` (`osoba_oID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_recept_kupac1`
    FOREIGN KEY (`kupac_osoba_oID`)
    REFERENCES `mydb`.`kupac` (`osoba_oID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`izdati_recept`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`izdati_recept` ;

CREATE TABLE IF NOT EXISTS `mydb`.`izdati_recept` (
```

```
  `datum_izdavanja` DATE NULL,
  `prodavac_osoba_oID` INT UNSIGNED NOT NULL,
  `recept_recID` INT NOT NULL,
  PRIMARY KEY (`recept_recID`),
  INDEX `fk_izdao_prodavac1_idx` (`prodavac_osoba_oID` ASC) ,
  INDEX `fk_izdao_recept1_idx` (`recept_recID` ASC) ,
  CONSTRAINT `fk_izdao_prodavac1`
    FOREIGN KEY (`prodavac_osoba_oID`)
    REFERENCES `mydb`.`prodavac` (`osoba_oID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_izdao_recept1`
    FOREIGN KEY (`recept_recID`)
    REFERENCES `mydb`.`recept` (`recID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `mydb`.`narucio`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`narucio` ;

CREATE TABLE IF NOT EXISTS `mydb`.`narucio` (
  `kupac_osoba_oID` INT UNSIGNED NOT NULL,
  `izdati_recept_recept_recID` INT NOT NULL,
  PRIMARY KEY (`izdati_recept_recept_recID`),
```

```
  INDEX `fk_narucio_kupac1_idx` (`kupac_osoba_oID` ASC) ,
  INDEX `fk_narucio_izdati_recept1_idx` (`izdati_recept_recept_recID` ASC) ,
  CONSTRAINT `fk_narucio_kupac1`
    FOREIGN KEY (`kupac_osoba_oID`)
    REFERENCES `mydb`.`kupac` (`osoba_oID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_narucio_izdati_recept1`
    FOREIGN KEY (`izdati_recept_recept_recID`)
    REFERENCES `mydb`.`izdati_recept` (`recept_recID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`inventar`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`inventar` ;

CREATE TABLE IF NOT EXISTS `mydb`.`inventar` (
  `kolicina` INT NULL,
  `ambalaza_leka_lek_lekID` INT NOT NULL,
  `ambalaza_leka_jedinica_mere_jID` INT NOT NULL,
  PRIMARY KEY (`ambalaza_leka_lek_lekID`, `ambalaza_leka_jedinica_mere_jID`),
  INDEX `fk_inventar_ambalaza_leka1_idx` (`ambalaza_leka_lek_lekID` ASC,
`ambalaza_leka_jedinica_mere_jID` ASC) ,
  CONSTRAINT `fk_inventar_ambalaza_leka1`
```

```
    FOREIGN KEY (`ambalaza_leka_lek_lekID` , `ambalaza_leka_jedinica_mere_jID`)
    REFERENCES `mydb`.`ambalaza_leka` (`lek_lekID` , `jedinica_mere_jID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;


-- -----------------------------------------------------
-- Data for table `mydb`.`osoba`
-- -----------------------------------------------------
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`osoba` (`oID`, `ime`, `prezime`, `datum_rodjenja`, `username`, `password`) VALUES (1, 'Djordje', 'Karisic', '2000-10-18', 'djordje34', 'djordje34');
INSERT INTO `mydb`.`osoba` (`oID`, `ime`, `prezime`, `datum_rodjenja`, `username`, `password`) VALUES (2, 'DJordje', 'DJordjevic', '1999-01-01', 'asdasdasd123', 'dasdasd431');
INSERT INTO `mydb`.`osoba` (`oID`, `ime`, `prezime`, `datum_rodjenja`, `username`, `password`) VALUES (3, 'Gavrilo', 'Gavrilovic', '1389-06-13', 'gavrilo123', '123gravilo');


COMMIT;



-- -----------------------------------------------------
-- Data for table `mydb`.`kupac`
-- -----------------------------------------------------
```

```
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`kupac` (`osoba_oID`, `clan_od`) VALUES (1, '2010-01-01');

COMMIT;


-- -----------------------------------------------------
-- Data for table `mydb`.`prodavac`
-- -----------------------------------------------------
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`prodavac` (`osoba_oID`, `datum_zaposlenja`, `licenca_farmaceut`) VALUES (3, '2000-01-01', '8658234');

COMMIT;


-- -----------------------------------------------------
-- Data for table `mydb`.`tip_leka`
-- -----------------------------------------------------
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`tip_leka` (`tipID`, `naziv`) VALUES (1, 'antibiotik');

COMMIT;
```

```
-- -----------------------------------------------------

-- Data for table `mydb`.`proizvodjac`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`proizvodjac` (`pID`, `ime`, `adresa`) VALUES (1, 'Galenika', 'D. Maksimovic 34');


COMMIT;




-- -----------------------------------------------------

-- Data for table `mydb`.`lek`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`lek` (`lekID`, `ime`, `tip_leka_tipID`, `proizvodjac_pID`) VALUES (1, 'Hemomicin', 1, 1);


COMMIT;




-- -----------------------------------------------------

-- Data for table `mydb`.`jedinica_mere`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`jedinica_mere` (`jID`, `jedinica`) VALUES (1, '3 tablete po 500mg');
```

```
COMMIT;


-- -----------------------------------------------------
-- Data for table `mydb`.`ambalaza_leka`
-- -----------------------------------------------------
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`ambalaza_leka` (`lek_lekID`, `jedinica_mere_jID`, `cena`) VALUES (1, 1, 200);

COMMIT;


-- -----------------------------------------------------
-- Data for table `mydb`.`doktor`
-- -----------------------------------------------------
START TRANSACTION;
USE `mydb`;
INSERT INTO `mydb`.`doktor` (`licenca_br`, `osoba_oID`) VALUES ('874593295', 2);

COMMIT;


-- -----------------------------------------------------
-- Data for table `mydb`.`recept`
-- -----------------------------------------------------
```

```
START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`recept` (`recID`, `ambalaza_leka_lek_lekID`, `ambalaza_leka_jedinica_mere_jID`, `doktor_osoba_oID`, `kupac_osoba_oID`) VALUES (1, 1, 1, 2, 1);


COMMIT;




-- -----------------------------------------------------

-- Data for table `mydb`.`izdati_recept`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`izdati_recept` (`datum_izdavanja`, `prodavac_osoba_oID`, `recept_recID`) VALUES ('2022-01-09', 3, 1);


COMMIT;




-- -----------------------------------------------------

-- Data for table `mydb`.`narucio`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`narucio` (`kupac_osoba_oID`, `izdati_recept_recept_recID`) VALUES (1, 1);


COMMIT;
```

```
-- -----------------------------------------------------

-- Data for table `mydb`.`inventar`

-- -----------------------------------------------------

START TRANSACTION;

USE `mydb`;

INSERT INTO `mydb`.`inventar` (`kolicina`, `ambalaza_leka_lek_lekID`, `ambalaza_leka_jedinica_mere_jID`) VALUES (10, 1, 1);


COMMIT;
```

1. SQL Dates
2. SQL EER Model
3. Prodaja lekova na recept
4. Generalizacija