

3.RT – DIZAJN BAZA PODATAKA (VJEŽBE)

Sadržaj

01-02: IZRADA BAZE PODATAKA U ACCESSU	2
03-04: VJEŽBA 1: ACCESS	2
05-06: DOKUMENTACIJA BAZE PODATAKA.....	3
07-08: VJEŽBA 2: DOKUMENTACIJA BAZE PODATAKA	7
09-10: ER DIJAGRAM 1	7
11-12: ER DIJAGRAM 2 (ZADACI ZA VJEŽBU)	8
13-14: VJEŽBA 3: ER DIJAGRAMI	14
15-16: PRETVARANJE ER DIJAGRAMA U RELACIJSKU SHEMU	15
17-18: NORMALIZACIJA	22
19-20: VJEŽBA 4: RELACIJSKA ALGEBRA	28
21-22: UVOD U SQL	29
23-24: SQL SORTIRANJE I FILTRIRANJE	35
25-26: VJEŽBA 5: SQL 1	36
27-28: SQL JOIN I UPDATE	37
29-30: AGREGATNE FUNKCIJE. TRANSAKCIJE. REFERENCIJALNI INTEGRITET	38
31-32: VJEŽBA 6: SQL 2	45
33-34: SQL I PHP 1.....	46
35-36: SQL I PHP 2.....	52

01-02: IZRADA BAZE PODATAKA U ACCESSU

Ponavljjanje gradiva iz 2. razreda. Izrada BP o školi (tablice, obrasci, upiti, izvještaji).

03-04: VJEŽBA 1: ACCESS

Radi se 2 školska sata. Treba predati u Classroom.

05-06: DOKUMENTACIJA BAZE PODATAKA

P.1. Projektiranje baze podataka o bolnici

U ovom studijskom primjeru bavimo se pojednostavnjenom i donekle neobičnom bolnicom. Projektiranje kreće od specifikacije koja je dobivena utvrđivanjem i analizom zahtjeva, a zatim se odvija u tri uobičajene faze: projektiranje na **konceptualnoj**, **logičkoj** i **fizičkoj** razini.

P.1.1. Specifikacija za bolnicu

Utvrđivanjem i analizom zahtjeva dobili smo sljedeću **specifikaciju**. Ona govori o bolnici, njezinim prostorijama, liječnicima, osoblju i pacijentima te o odgovarajućim odnosima i pravilima.

Pacijenti koji zauzimaju sobe. Pacijent se obično prilikom dolaska u bolnicu smješta u bolničku sobu. Svaka soba može primiti više pacijenata. Konzultanti (stariji kirurzi) bolnice smiju imati i svoje privatne pacijente, koji su smješteni u jednokrevetnim privatnim sobama. Informacije koje treba pamtiti o pacijentu uključuju osobni identifikacijski broj (OIB), prezime, ime, adresu itd.

Medicinske sestre zadužene za sobe. Sestra može ili ne mora biti zadužena za sobu. Pritom jedna sestra može biti zadužena najviše za jednu sobu, no za istu sobu može biti zaduženo više sestara. Sestra je jednoznačno određena svojim OIB-om.

Kirurške operacije koje se obavljaju nad pacijentima. Nad istim pacijentom može se obaviti više kirurških operacija. Informacije su o jednoj operaciji: tip operacije, pacijent, kirurg, datum, vrijeme i mjesto.

Kirurzi koji obavljaju operacije. Jednu operaciju obavlja samo jedan kirurg, a za ostale se prisutne kirurge smatra da asistiraju pri operaciji. Kirurge nadgledaju stariji kirurzi, takozvani konzultanti, koji također mogu obavljati operacije ili asistirati. Informacije su o jednom kirurgu su: OIB, prezime i ime, adresa, broj telefona i tako dalje. Svaki konzultant ima svoju specijalnost.

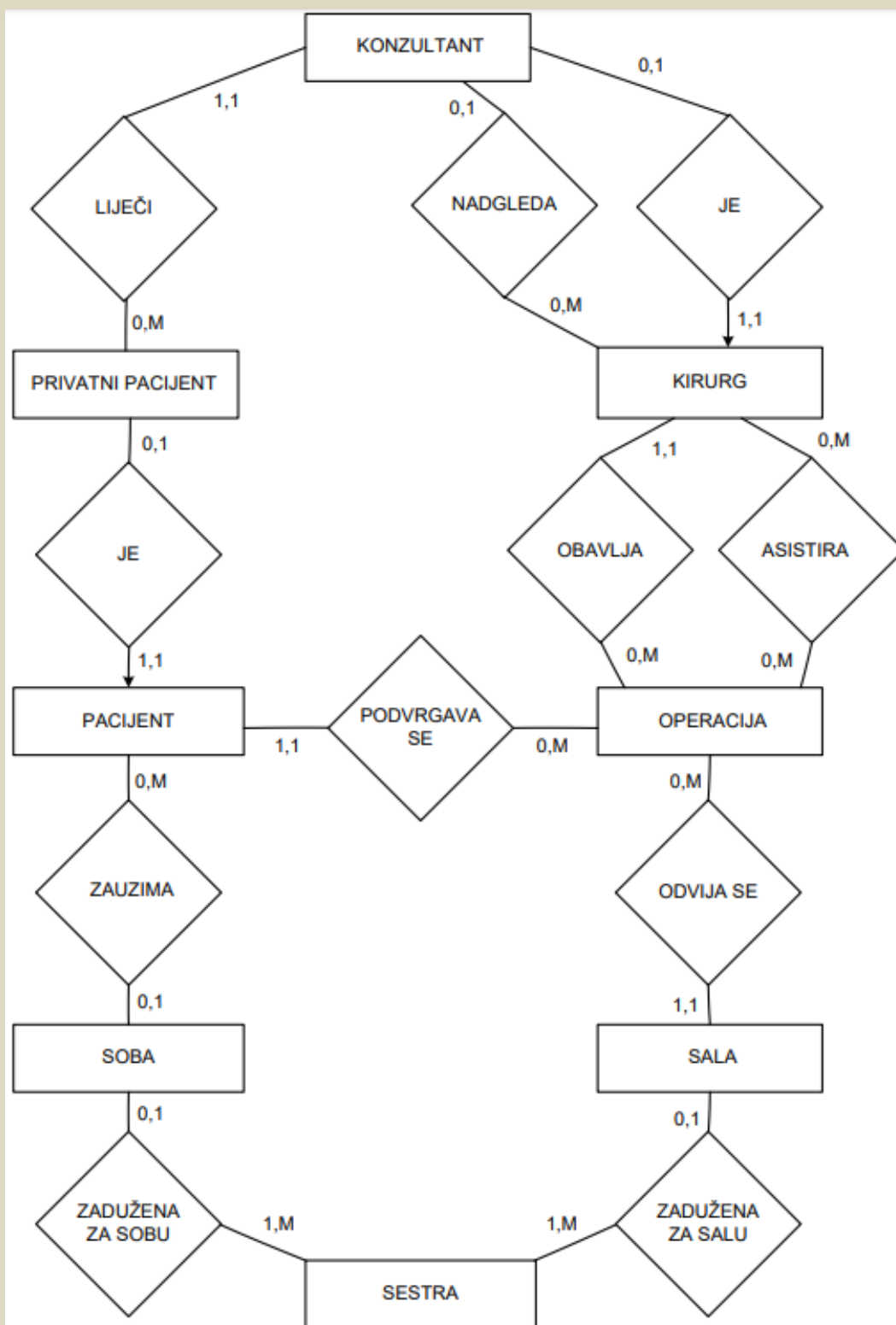
Operacijske sale u kojima se odvijaju operacije. Jedna se operacija odvija samo u jednoj sali, no ista sala može biti poprište mnogih operacija. Svaka sala ima svoju identifikacijsku oznaku. Neke sale su specijalno opremljene za neke vrste operacija.

Medicinske sestre zadužene za sale. Sestra može ili ne mora biti zadužena za salu, no ne može biti zadužena za više od jedne sale. Za jednu salu može biti zaduženo više sestara.

P.1.2 . Konceptualna shema za bolnicu

Početni dio projektiranja baze podataka o bolnici je projektiranje na konceptualnoj razini. Čitanjem prethodne specifikacije otkrivamo elemente od kojih se sastoji konceptualna shema naše baze, dakle **entitete**, **veze** i **atribute**.

Konceptualnu shemu oblikujemo crtanjem **reduciranog Chenova dijagrama** entiteta i veza te sastavljanjem **dodatnog teksta koji prati dijagram**.



Slika P.1: Dijagram s entitetima i vezama za bazu podataka o bolnici

Tip entiteta **KIRURG** ima atribute:

OIB, PREZIME, IME, ADRESA, BROJ TELEFONA.

Tip entiteta **KONZULTANT** je podtip od KIRURG, ima dodatni atribut:

SPECIJALNOST (grana kirurgije u kojoj se specijalizirao).

Tip entiteta **PACIJENT** ima atribute:

OIB, PREZIME, IME, ADRESA, DATUM ROĐENJA, SPOL.

Tip entiteta **PRIVATNI PACIJENT** JE podtip od **PACIJENT**, ima dodatni atribut:

ID PRIVATNE SOBE.

Tip entiteta **SESTRA** ima attribute:

OIB, PREZIME, IME, STRUČNI STUPANJ.

Tip entiteta **SOBA** (misli se na sobu koja nije privatna) ima attribute:

ID SOBE, TIP SOBE, BROJ KREVETA.

Tip entiteta **SALA** ima attribute:

ID SALE, TIP SALE.

Tip entiteta **OPERACIJA** ima attribute:

ID OPERACIJE, TIP OPERACIJE, DATUM, VRIJEME.

Veza **ASISTIRA** ima atribut:

ULOGA (kirurga u operaciji).

Veza **ZADUŽENA ZA SOBU** ima atribut:

DATUM ZADUŽIVANJA.

Veza **ZADUŽENA ZA SALU** ima atribut:

DATUM ZADUŽIVANJA.

Ostale veze nemaju attribute.

Slika P.2: Popratni tekst uz dijagram sa Slike P.1

P.1.3. Relacijska shema i rječnik podataka za bolnicu

Nastavak projektiranja baze podataka o bolnici je projektiranje na logičkoj razini. Služeći se uobičajenim pravilima, konceptualnu shemu sa Slike P.1 i Slike P.2 **pretvaramo u relacijsku shemu**, dakle skup relacija od kojih svaka ima zadano ime, attribute i primarni ključ. Također usput sastavljamo rječnik podataka, dakle popis svih atributa s objašnjenjem njihova tipa i značenja.

Dobivena relacijska shema prikazana je na Slici P.3, a rječnik podataka na Slici P.4.

Vidimo da je svaki tip entiteta iz konceptualne sheme prikazan jednom relacijom u logičkoj shemi. S druge strane, prikaz veze zavisi o njezinoj funkcionalnosti te o obaveznosti članstva njezinih entiteta.

- Veza ZAUZIMA prikazana je stranim ključem ID SOBE u relaciji PACIJENT, jer u njoj PACIJENT ima skoro obavezno članstvo.
- Slično, veza LIJEČI prikazana je stranim ključem OIB KONZULTANTA u relaciji PRIVATNI PACIJENT.
- Također, strani ključevi OIB KIRURGA, OIB PACIJENTA i ID SALE u relaciji OPERACIJA prikazuju veze OBAVLJA, PODVRGAVA SE odnosno ODVIJA SE, za koje OPERACIJA ima obavezno članstvo.
- Veza ASISTIRA zbog svoje je funkcionalnosti M:M morala biti prikazana posebnom relacijom, koja sadrži ključne attribute od KIRURG i OPERACIJA zajedno s dodatnim atributom ULOGA.
- 1:M veze ZADUŽENA ZA SOBU odnosno ZADUŽENA ZA SALU su zbog neobaveznosti članstva prikazane posebnim relacijama. Te relacije sadrže ključne attribute odgovarajućih entiteta i dodatni atribut DATUM ZADUŽIVANJA. Kad bi većina sestara bila zadužena za sobe, tada bi možda bilo bolje vezu ZADUŽENA ZA SOBU prikazati stranim ključem ID SOBE u relaciji SESTRA, no tada bi u istu relaciju morali ugraditi i dodatni atribut DATUM ZADUŽIVANJA.

- 1:M veza NADGLEDA zbog neobaveznosti je članstva prikazana posebnom relacijom. Mogla je biti prikazana i ubacivanjem OIB KONZULTANTA u relaciju KIRURG, no tada bi ubačeni atribut bio prazan za sve kirurge koje nitko ne nadgleda.

Primijetimo da je dobivena relacijska shema već u četvrtoj normalnoj formi, tako da nije potrebno provoditi dodatni postupak normalizacije. To je zato što je polazna konceptualna shema bila zdravo oblikovana.

KIRURG (OIB, PREZIME, IME, ADRESA, BROJ TELEFONA)
 KONZULTANT (OIB, SPECIJALNOST)
 PACIJENT (OIB, PREZIME, IME, ID SOBE, ADRESA, DATUM ROĐENJA, SPOL)
 PRIVATNI PACIJENT (OIB, OIB KONZULTANTA, ID PRIVATNE SOBE)
 SESTRA (OIB, PREZIME, IME, STRUČNI STUPANJ)
 SOBA (ID SOBE, TIP SOBE, BROJ KREVETA)
 SALA (ID SALE, TIP SALE)
 OPERACIJA (ID OPERACIJE, OIB KIRURGA, ID SALE, OIB PACIJENTA, TIP OPERACIJE, DATUM, VRIJEME)
 ASISTIRA (ID OPERACIJE, OIB KIRURGA, ULOGA)
 NADGLEDA (OIB NADGLEDANOG, OIB KONZULTANTA)
 ZADUŽENA ZA SOBU (OIB SESTRE, ID SOBE, DATUM ZADUŽIVANJA)
 ZADUŽENA ZA SALU (OIB SESTRE, ID SALE, DATUM ZADUŽIVANJA)

Slika P.3: Relacijska shema za bazu podataka o bolnici

P.1.4. Fizička shema za bolnicu

Zadnji dio projektiranja baze podataka o bolnici predstavlja projektiranje na fizičkoj razini. Relacijsku shemu naše baze pretvaramo u fizičku shemu tako da građu svake relacije sa Slike P.3 opišemo odgovarajućom SQL naredbom CREATE TABLE. Pritom tipove atributa određujemo u skladu s rječnikom podataka sa Slike P.4. Ako se služimo sintaksom iz MySQL-a, tada dobivena fizička shema izgleda kao na Slikama P.5 i P.6.

Tekst sa Slike P.5 i P.6 treba smatrati početnom inačicom fizičke sheme koja se može dalje dotjerivati. Ta početna inačica osigurava integritet domena za attribute u onoj mjeri koliko to dopuštaju mogućnosti zadavanja tipova u MySQL-u. Također, osigurana je jedinstvenost vrijednosti primarnog ključa u svakoj relaciji.

Primijetimo da shema sa Slike P.5 i P.6 za sada ne osigurava referencijalni integritet, dakle konzistentnu uporabu vrijednosti za strane ključeve. Naime, u našoj bazi postoji vrlo velik broj stranih ključeva:

- ID SOBE u relaciji PACIJENT
- OIB KONZULTANTA u relaciji PRIVATNI PACIJENT
- OIB KIRURGA, ID SALE, i OIB PACIJENTA u relaciji OPERACIJA
- ID OPERACIJE, i OIB KIRURGA u relaciji ASISTIRA
- OIB NADGLEDANOG, i OIB KONZULTANTA u relaciji NADGLEDA
- ID SOBE u relaciji ZADUŽENA ZA SOBU
- ID SALE u relaciji ZADUŽENA ZA SALU.

Automatska provjera svih ovih ključeva ne dolazi u obzir jer bi to previše zakompliciralo fizičku građu baze i degradiralo njezine performanse. Ipak, neke važnije provjere mogle bi se implementirati uvođenjem sekundarnih indeksa i klauzulama FOREIGN KEY.

Zadatak (radite u parovima):

Treba napraviti pripremu za BP knjižnicu. U knjižnici se mogu, osim knjiga, posuditi i digitalni sadržaji (online knjige, igre, računalni programi). Ako si netko želi u njoj nešto posuditi, treba postati njezin član. Knjižnica ima svoje zaposlenike (voditelj, knjižničari, čistači) i svoje korisnike. Knjižnica je velika pa ima nekoliko odjela (dječji, klasični, suvremeni, edukacijski, e-odjel...) koji se nalaze svaki u svojoj prostoriji. U svakoj prostoriji radi po jedan knjižničar koji je zadužen za nju.

1. Specifikacija
 - a. Budite maštoviti pa dodajte još nekoliko rečenica u opis zadatka (**specifikaciju**). Iz te specifikacije projektantu BP treba biti jasno što se sve i na koji način u knjižnici zbiva.
2. Konceptualna shema
 - a. Pokušajte sada **grafički prikazati** sve što je napisano u specifikaciji.
3. Relacijska shema
 - a. Prema zadacima 1 i 2 napravite **relacijsku shemu** iz koje će se vidjeti koji sve podaci bi se trebali evidentirati u vašoj BP.

07-08: VJEŽBA 2: DOKUMENTACIJA BAZE PODATAKA

Radi se 2 školska sata. Treba predati u Classroom.

09-10: ER DIJAGRAM 1

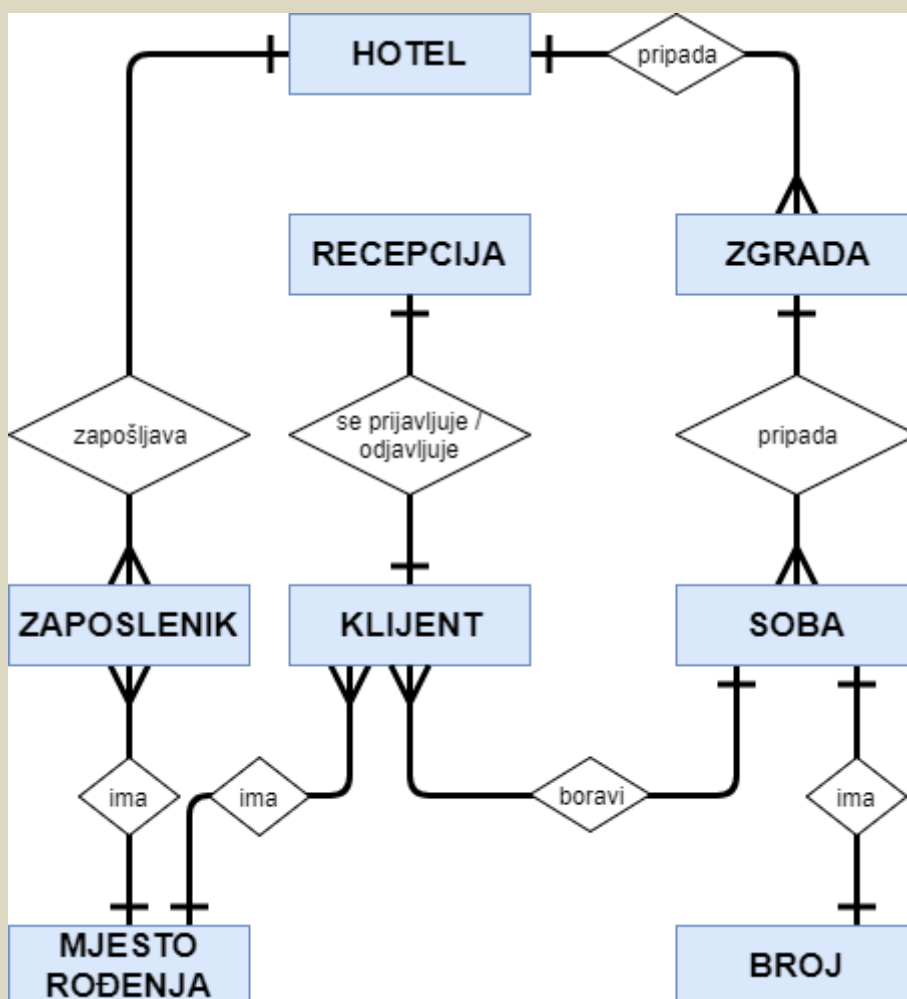
Pogledati prezentaciju i PDF primjer sa zadatkom.

11-12: ER DIJAGRAM 2 (ZADACI ZA VJEŽBU)

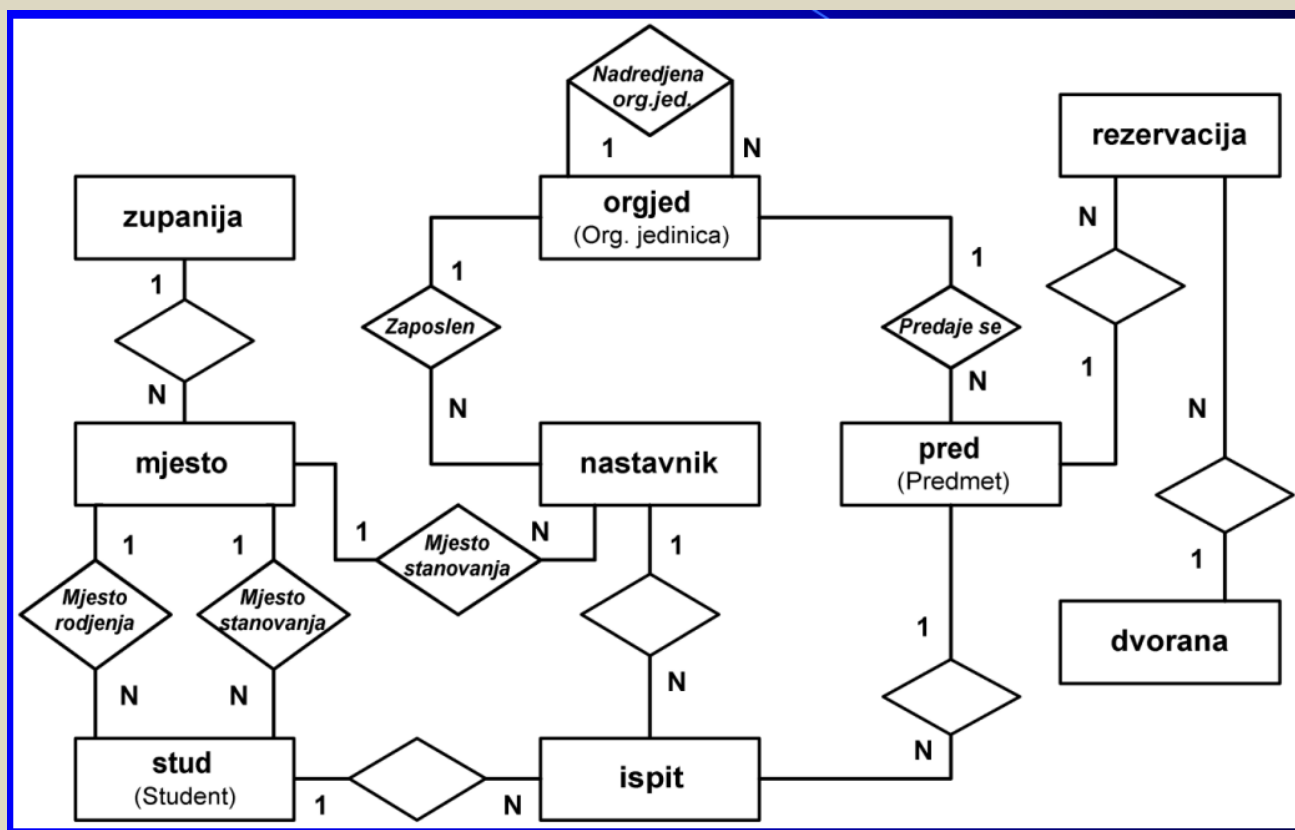
Zadatak 1: Nacrtaj ER dijagram:

Hotel se sastoji od nekoliko zgrada. U svakoj zgradi nalaze se sobe. Svaka soba ima svoj broj. U svakoj sobi može boraviti nekoliko klijenata. U svakoj zgradi je po jedna recepcija. U hotelu su zaposleni zaposlenici. Svaki zaposlenik i svaki klijent ima svoje mjesto rođenja. klijenti se prijavljuju/odjavljuju na recepciji pojedinačno.

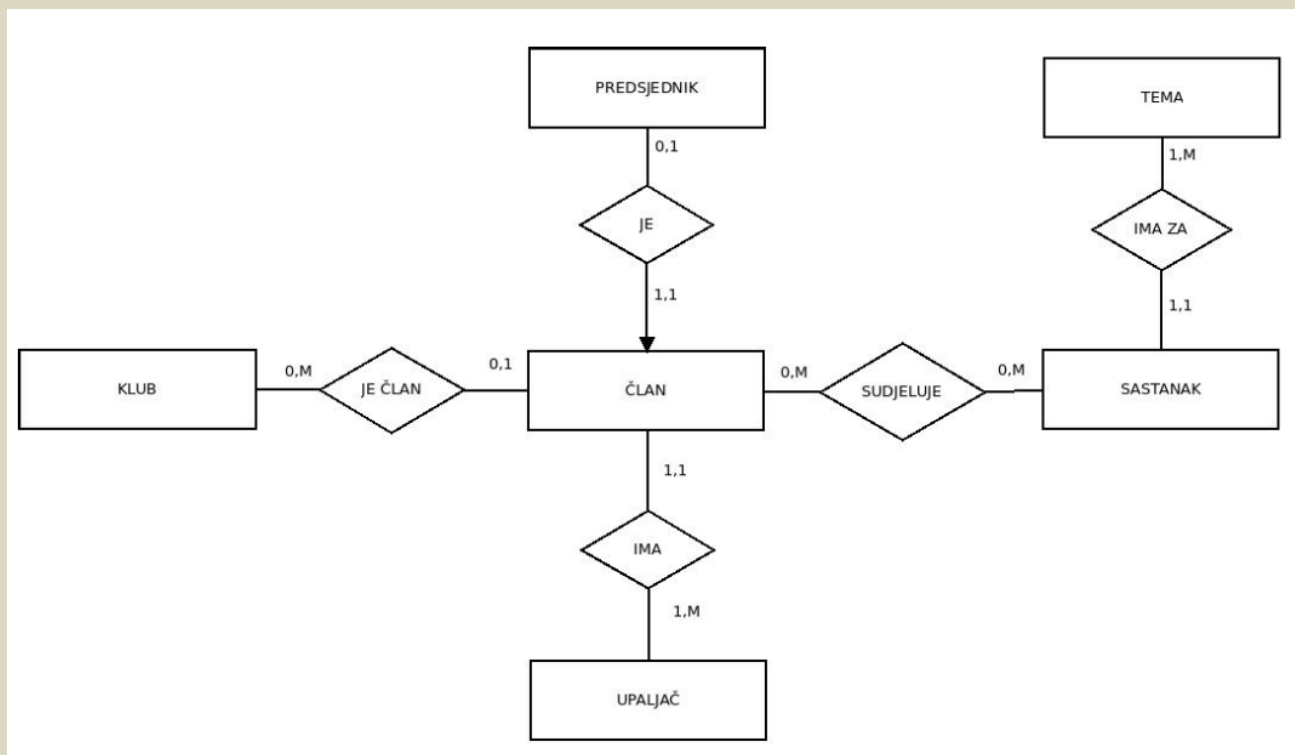
Rješenje:



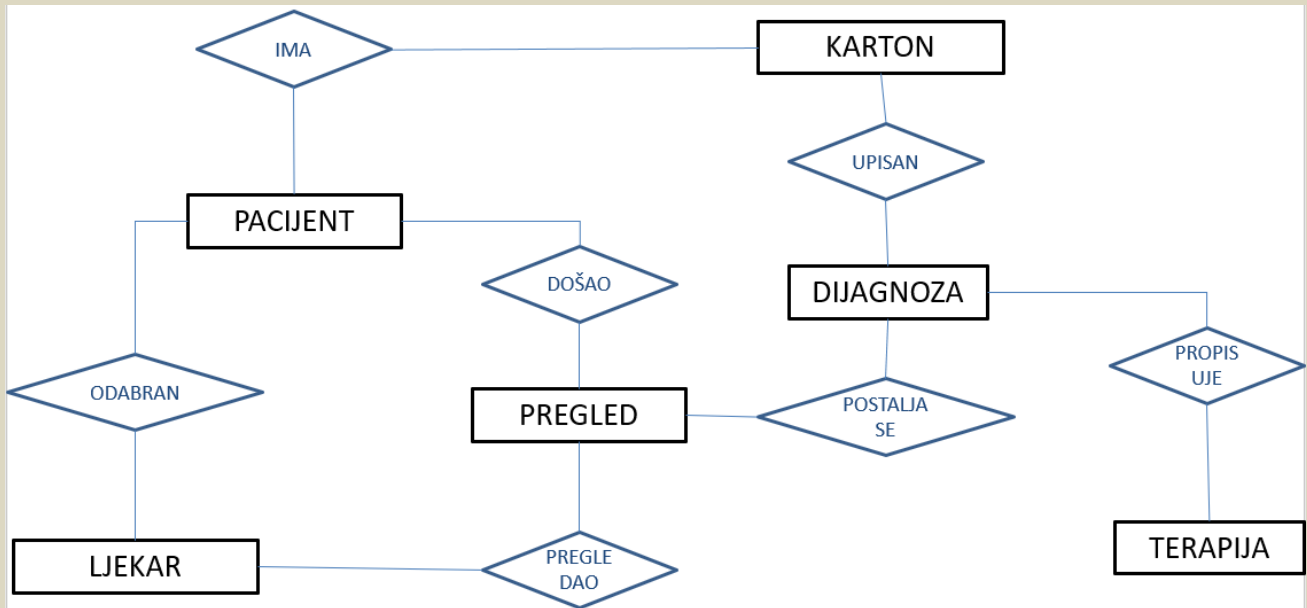
Zadatak 2: Napiši tekst zadatka kojemu bi pripadao ovaj ER dijagram:



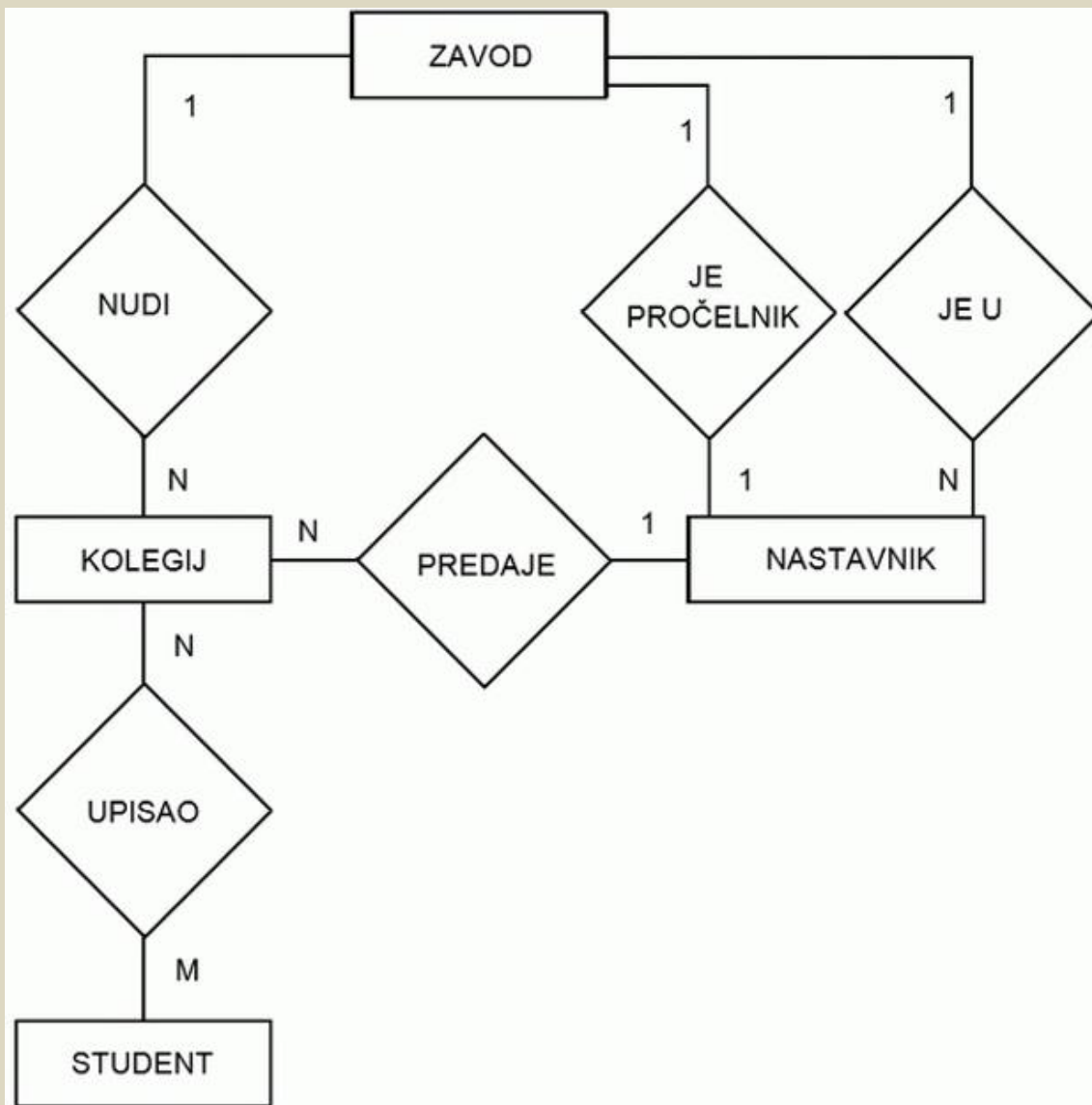
Zadatak 3: Riješi:



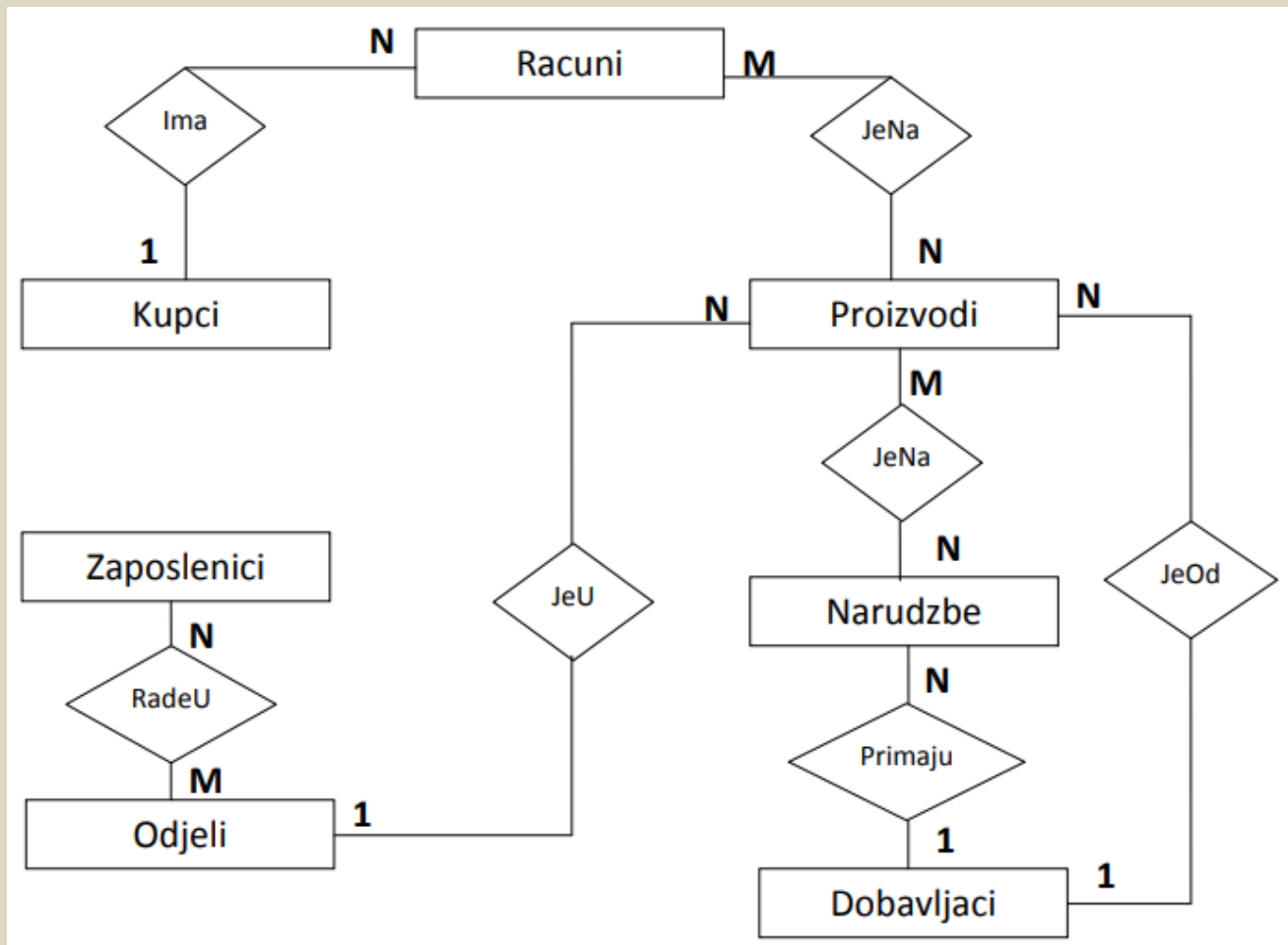
Zadatak 4: Riješi:



Zadatak 5: Riješi:



Zadatak 7: Riješi:



Zadatak 8: Riješi:

- Videoteka JednostavniFilm doo bi htjela unaprijediti svoje poslovanje. Do sada su klijenti sa sobom imali kartica na kojima su pisali podaci o njima i to ime, prezime, datum rođenja, adresa i broj telefona. Na poleđini kartice su se pisali posudbe i to datum posude i datum vraćanja filma. Film je bio opisan nekom šifrom i nazivom. Htjeli bi ubuduće osim naziva filma i pamtiti njegov žanr, te njegovog redatelja.

Analiza zahtjeva:

- Entiteti i atributi su:

- **Klijent** (Ime, Prezime, DatumRođenja, Adresa, BrojTelefona)
- **Film** (Šifra, NazivFilma, NazivRedatelja)
- **Žanr** (NazivŽanra)
- ~~Posudba~~ (~~DatumPosudbe, DatumVraćanja, Film, Klijent~~)

Nije entitet iako ima attribute

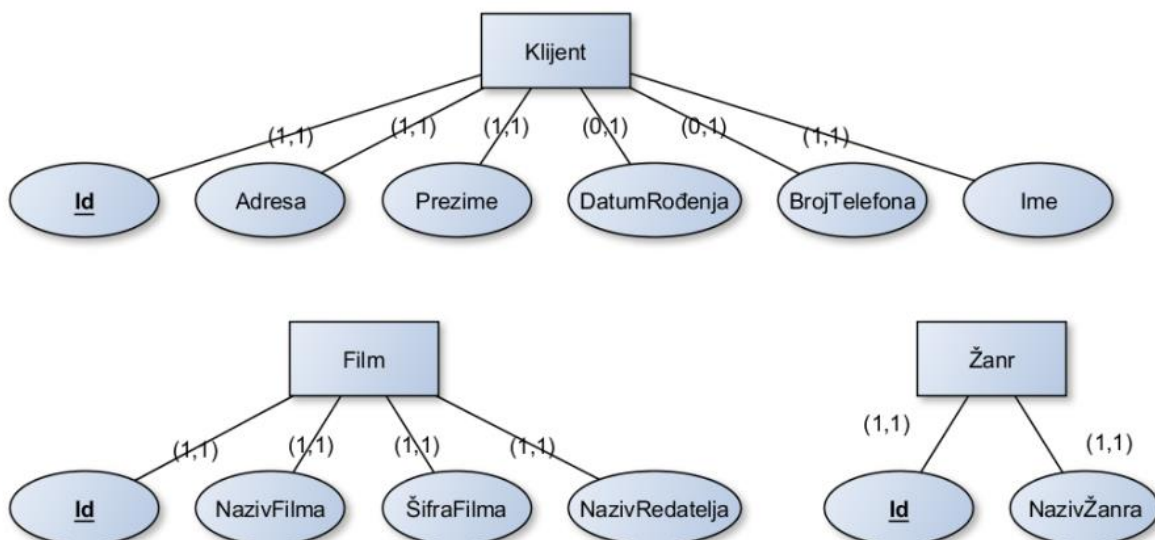
- Veze:

- Klijent **posuđuje** Film => posudba
- Film **pripada** žanru


ER model Dijagram entiteta

- Entiteti i atributi su:

- **Klijent** (Ime, Prezime, DatumRođenja, Adresa, BrojTelefona)
- **Film** (Šifra, NazivFilma, NazivRedatelja)
- **Žanr** (NazivŽanra)





ER diagram

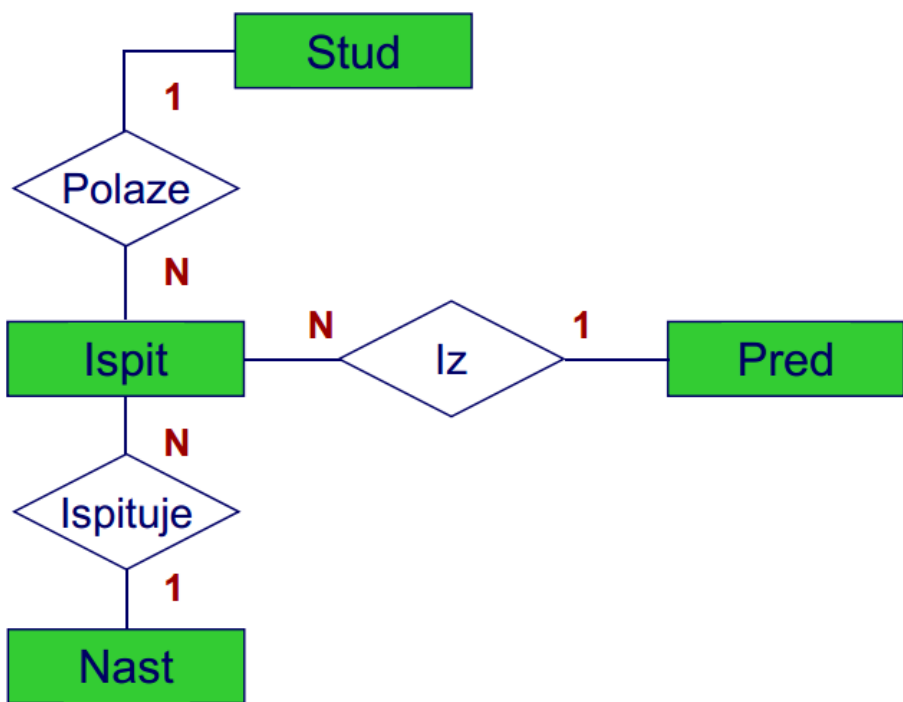
-
- ```

 erDiagram
 Posudba }|--}| Klient : "M"
 Posudba }|--}| Film : "N"
 Pripada }|--}| Film : "N"
 Pripada }|--}| Žanr : "1"
 Posudba }o--o} DatumPosudbeFilma : "(1,1)"
 Posudba }o--o} DatumVraćanjaFilma : "(0,1)"

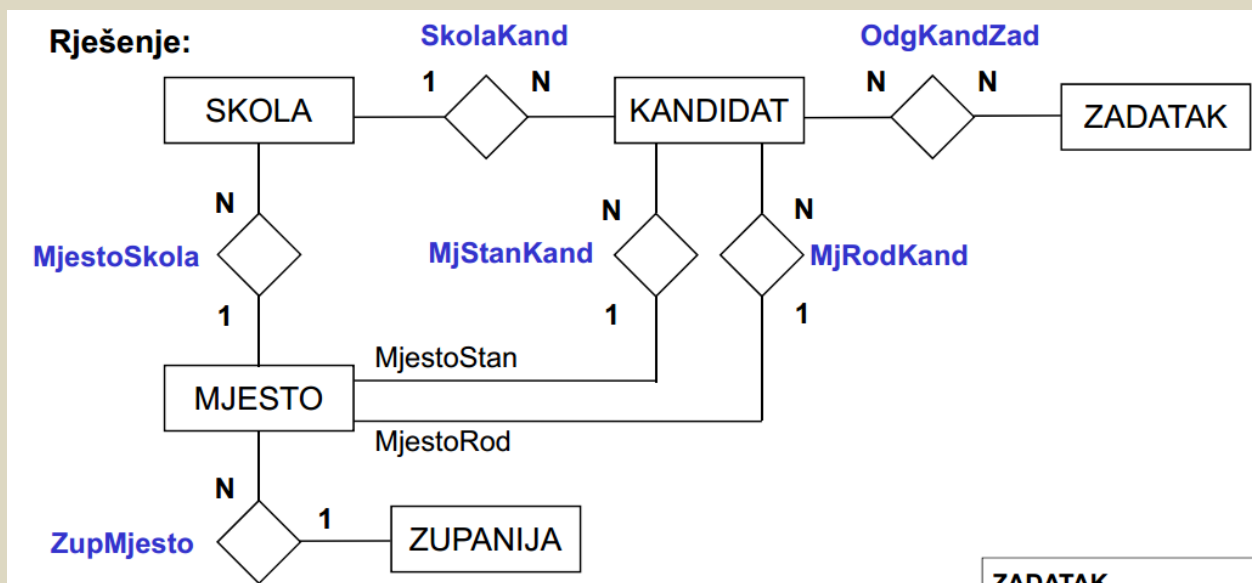
```

[http://e-student.fpz.hr/Predmeti/B/Baze\\_podataka/Materijali/Auditorne\\_vjezbe\\_2.pdf](http://e-student.fpz.hr/Predmeti/B/Baze_podataka/Materijali/Auditorne_vjezbe_2.pdf).

### Zadatak 9: Riješi:



**Zadatak 10:** Riješi:



**Zadatak 11:** U alatu **DRAW.IO** ili nekom sličnom nacrtaj barem dva danas obrađena primjera ER modela.

Besplatan alat za izradu ER dijagrama: **Open ModelSphere (provjerite!)**

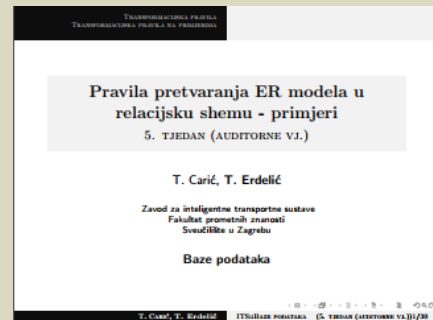
## 13-14: VJEŽBA 3: ER DIJAGRAMI

Radi se 2 školska sata.

## 15-16: PRETVARANJE ER DIJAGRAMA U RELACIJSKU SHEMU

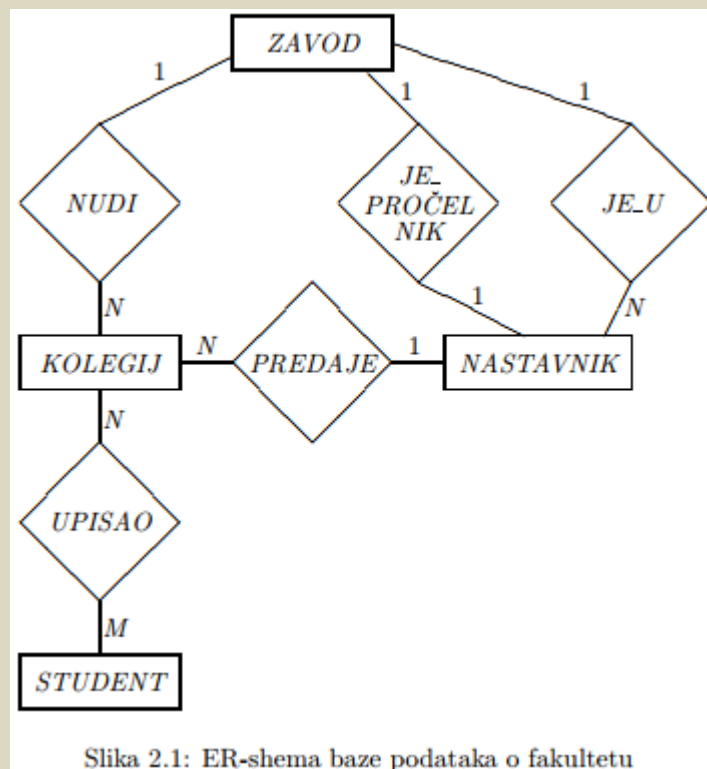
U nastavku objašnjavamo kako se pojedini elementi ER-sheme pretvaraju u relacije. Na taj način pokazat ćemo kako se iz cijele ER-sheme dobiva relacijska shema.

Prvo pogledamo ovu prezentaciju s riješenim primjerom:



### 1. Pretvorba tipova entiteta

Svaki tip entiteta prikazuje se jednom relacijom. Atributi tipa postaju atributi relacije. Jedan primjerak entiteta prikazan je jednom n-torkom. Primarni ključ entiteta postaje primarni ključ relacije.



Npr. tip STUDENT iz naše fakultetske baze podataka sa Slike 2.1 prikazuje se relacijom:

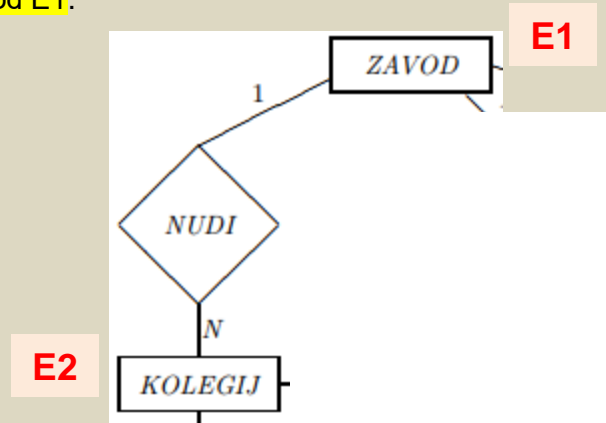
STUDENT (**BR INDEKSA**, IME STUDENTA, ADRESA, SPOL, ...)

Doduše, sudjelovanje entiteta u vezama može zahtijevati da se u relaciju dodaju još neki atributi koji nisu postojali u odgovarajućem tipu entiteta.

## 2. Pretvorba binarnih veza

### 2.1. UBACIVANJE VANJSKOG KLJUČA

Ako tip entiteta E2 ima obavezno članstvo u (N : 1) vezi s tipom E1, tada relacija za E2 treba uključiti primarne attribute od E1.



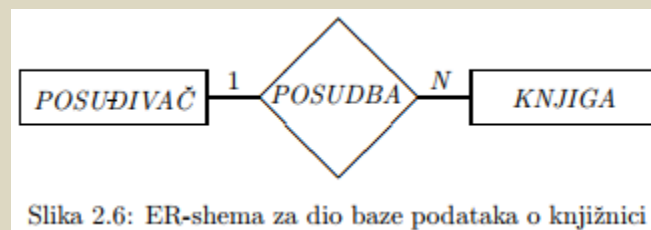
Npr. ako u gornjoj ER-shemi svaki kolegij mora biti ponuđen od nekog zavoda, tada se veza NUDI svodi na to da u relaciju KOLEGIJ (E2) ubacimo ključ relacije ZAVOD (E1):

KOLEGIJ (BR KOLEGIJA, **IME ZAVODA**, NASLOV, SEMESTAR, ...)

Ključ jedne relacije koji je prepisan u drugu relaciju zove se **STRANI (VANJSKI) KLJUČ** (u toj drugoj relaciji).

Dakle, kod veze 1:N vanjski ključ ćemo uvijek staviti u relaciju kod koje je oznaka N.

Ako tip entiteta E2 ima neobavezno članstvo u (N : 1) vezi s tipom E1, tada vezu možemo prikazati ili na prethodni način ili uvođenjem nove relacije čiji atributi su primarni atributi od E1 i E2.



Slika 2.6: ER-shema za dio baze podataka o knjižnici

Kao primjer, promotrimo vezu na Slici 2.6 koja prikazuje posuđivanje knjiga u knjižnici.

POSUĐIVAČ (BR ISKAZNICE, PREZIME, IME, ADRESA, ...)

KNJIGA (REG BROJ, **BR ISKAZNICE**, NASLOV, ...)

Ovdje smo u relaciju KNJIGA dodali BR ISKAZNICE osobe koja je posudila knjigu. Vrijednost atributa BR ISKAZNICE bit će prazna u mnogim n-torkama relacije KNJIGA, tj. za sve knjige koje trenutno nisu posuđene.



## 2.2. UVOĐENJE NOVE RELACIJE

Drugo rješenje zahtijeva **3** relacije:

POSUĐIVAČ (**BR ISKAZNICE**, PREZIME, IME, ADRESA, ...)

KNJIGA (**REG BROJ**, NASLOV, ...)

POSUDBA (**REG BROJ**, **BR ISKAZNICE**)

Samo one knjige koje su trenutno posuđene predstavljene su n-torkom u relaciji POSUDBA.

Uvođenje posebne relacije za prikaz veze je pogotovo preporučljivo ako ona ima neke svoje attribute.

Npr. u relaciju POSUDBA mogli bismo uvesti atribut DATUM VRAĆANJA.

**(N : M) veza se uvijek prikazuje posebnom relacijom** koja se sastoji od primarnih atributa za oba tipa entiteta zajedno s eventualnim atributima veze.

Ako imamo vezu N:M, stvorit ćemo **novu relaciju** koja će imati **2 vanjska ključa** (jedan od prve i jedan od druge tablice).



Npr. veza UPISAO iz fakultetske baze sa Slike 2.1 prikazuje se relacijom:

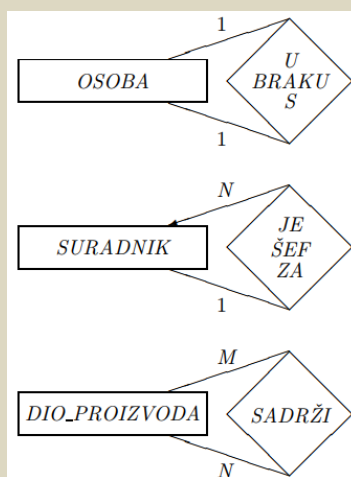
UPISAO (**BR INDEKSA**, **BR KOLEGIJA**, DATUM UPISA) .

Činjenica da je jedan student upisao jedan kolegij prikazuje se jednom n-torkom u relaciji UPISAO.

**Ključ** za UPISAO je očito **složen od dva atributa** (BR INDEKSA i BR KOLEGIJA) jer nijedan od ovih atributa sam nije dovoljan da jednoznačno odredi n-torku u toj relaciji.

### 3. Pretvorba involuiranih veza

Obavlja se slično kao i za binarne veze. Poslužit ćemo se primjerima sa Slike 2.2.



Slika 2.2: Primjeri za involuirane veze

Tip entiteta OSOBA i vezu (1 : 1) U BRAKU S najbolje je (zbog neobaveznosti) prikazati pomoću dvije relacije:

OSOBA (OIB, PREZIME, IME, ADRESA, ...)  
 BRAK (OIB MUŽA, OIB ŽENE, DATUM VJENČANJA)

Tip entiteta SURADNIK i vezu (1 : N) JE ŠEF ZA prikazujemo samo jednom relacijom:

SURADNIK (ID, ID ŠEFA, PREZIME, IME, ...)

Ovo neće uzrokovati mnogo praznih vrijednosti atributa, budući da većina suradnika ima šefa.

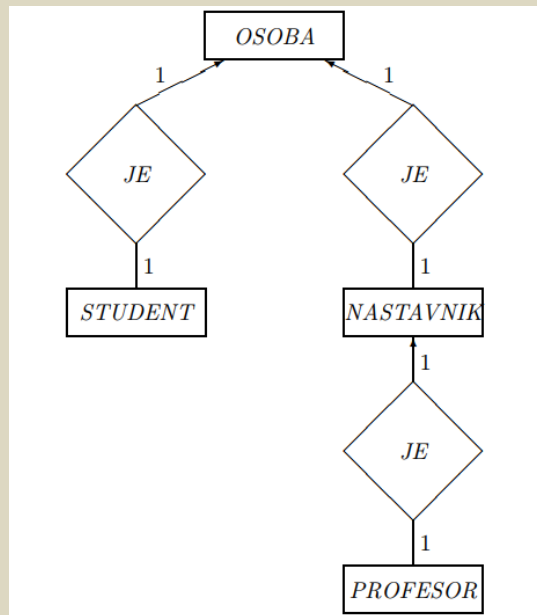
Tip entiteta DIO PROIZVODA i vezu (N : M) SADRŽI moramo prikazati pomoću dvije relacije:

DIO PROIZVODA (BR DIJELA, IME DIJELA, OPIS, ...)  
 SADRŽI (BR DIJELA SLOŽENOG, BR DIJELA JEDNOSTAVNOG, KOLIČINA)

Dodali smo i atribut KOLIČINA koji kaže koliko jednostavnih dijelova ulazi u jedan složeni.

#### 4. Pretvorba podtipova

Podtip se prikazuje posebnom relacijom koja sadrži primarne attribute nadređenog tipa i attribute specifične za taj podtip.



Slika 2.3: Primjer ER-sheme s pod-tipovima entiteta

Npr. hijerarhija tipova sa Slike 2.3 prikazuje se sljedećim relacijama:

OSOBA (OIB, ... atributi **zajednički** za sve tipove osoba ...)

STUDENT (OIB, ... atributi specifični za studente ...)

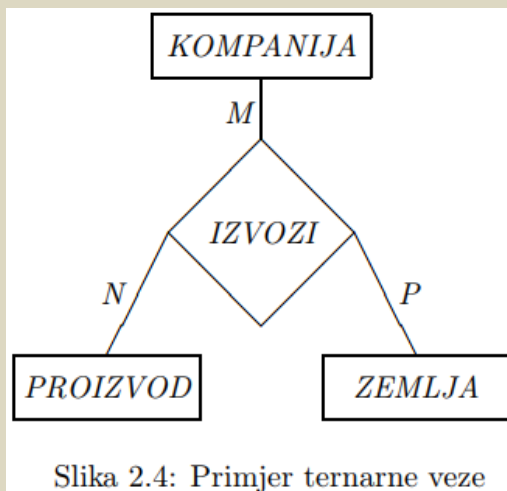
NASTAVNIK (OIB, ... atributi specifični za nastavnike ...)

PROFESOR (OIB, ... atributi specifični za profesore ...)

Veza JE uspostavlja se na osnovu pojavljivanja istog OIB-a u raznim relacijama.

## 5. Pretvorba ternarnih veza

Ternarna veza prikazuje se posebnom relacijom koja sadrži primarne attribute svih triju tipova entiteta zajedno s eventualnim atributima veze.



Za primjer sa Slike 2.4 imamo:

KOMPANIJA (**IME KOMPANIJE**, ...)

PROIZVOD (**IME PROIZVODA**, ...)

ZEMLJA (**IME ZEMLJE**, ...)

IZVOZI (**IME KOMPANIJE**, **IME PROIZVODA**, **IME ZEMLJE**) .

Činjenica da se jedan proizvod jedne kompanije izvozi u jednu zemlju prikazana je jednom n-torkom u relaciji IZVOZI. Kod ternarnih veza čija funkcionalnost nije (M : N : P) broj primarnih atributa je manji.

**ZADATAK ZA DOMAĆU ZADAĆU:**

**Zadatak 1 (za ocjene 2-3):** **Napravite ER dijagram** za dio informacijskog sustava tvrtke koja se bavi izradom EU projekata za vanjske naručitelje. Potrebno je voditi evidenciju o naručiteljima projekta, čija su osnovna obilježja oznaka naručitelja, naziv te adresa, te o naručenim projektima, za koje je potrebno zabilježiti oznaku projekta, naziv i kratki opis projekta. Projekte izrađuju konzultanti (koji su ujedno i zaposlenici tvrtke), pri čemu je svakom projektu dodijeljen točno jedan konzultant, dok pojedini konzultant može raditi na više projekata. Svaki konzultant specijalist je za izradu projekata za određeno područje djelovanja (poput edukacije, poljoprivrede, znanosti i tehnologije itd.). Konzultantov rad naplaćuje se ovisno o broju radnih sati utrošenih na izradu projekta, dok konzultantova satnica (cijena rada po satu) ovisi o području djelovanja za koje se specijalizirao.

**Zadatak 2 (za ocjene 4-5):** **Napravite ER dijagram i relacijsku shemu** za dio informacijskog sustava jedne on-line trgovine. Potrebno je voditi evidenciju o proizvodima koji se nude, a koji su podijeljeni u različite kategorije. Za svaki proizvod bilježi se njegov naziv, slika, proizvođač, broj jedinica kojima se raspolaže, cijena po komadu te podatak o tome je li proizvod raspoloživ ili ne. Nadalje, zahtijeva se vođenje evidencije o kupcima proizvoda i narudžbama koje oni upute. Jedna narudžba može uključivati jedan ili više proizvoda, a za svaku narudžbu treba zabilježiti datum i vrijeme njezina primitka, datum njezine obrade te podatke o tome je li narudžba izvršena te odobrava li se kakav popust na iznos narudžbe. Osim toga, treba pohraniti podatke o broju jedinica proizvoda koje se naručuje, cijeni koju za njih treba platiti te datumu i vremenu njihove isporuke. Kupac naručene proizvode plaća putem kreditne kartice, pri čemu se uz jednog kupca može vezivati više vrsta kreditnih kartica pa stoga uz narudžbu treba također zabilježiti i kojom je kreditnom karticom ona plaćena.

**SISTEMAŠI:**

Odabrani zadatak riješite i pošaljite mi na mail [marko.ljubek@gmail.com](mailto:marko.ljubek@gmail.com) do 11.12.2020.

**PROGRAMERI:**

Odabrani zadatak riješite i pošaljite mi na mail [marko.ljubek@gmail.com](mailto:marko.ljubek@gmail.com) do 18.12.2020.

Izvor 1: <http://jadran.izor.hr/~dadic/EKO/baze-podataka.pdf>

Izvor 2: [https://www.weboteka.net/fpz/Baze%20podataka/Predavanja/A05\\_auditorne.pdf](https://www.weboteka.net/fpz/Baze%20podataka/Predavanja/A05_auditorne.pdf)

## 17-18: NORMALIZACIJA

Normalizacija je postupak uklanjanja nepoželjnih funkcijskih ovisnosti, a provodi se radi postizanja dobrih statičkih i dinamičkih svojstava baze podataka:

- učinkovitost pristupa podacima i jednostavnost njihova pretraživanja
- kontrola nad redundancijom ( višestrukim pojavljivanjem podataka u bazi)
- kontrola nad anomalijama koje se mogu pojaviti kod ažuriranja podataka

Definirano je šest normalnih formi, a u praksi je potrebno i najčešće dovoljno, dovesti sve relacije u treću normalnu formu.

### 1NF – Prva normalna forma

Relacija se nalazi u prvoj normalnoj formi (1NF) ako je svaki njen atribut **atomičan**, što znači da **sadrži samo jednu vrijednost** → ne smije biti više vrijednosti odvojenih npr. zarezom. Dvodimenzionalna tablica ima u svakoj ćeliji samo jednu elementarnu vrijednost, koja se ne može dalje rastaviti bez gubitka smisla informacije. **Ključ uvijek ima jedinstvenu vrijednost, a za svaku njegovu vrijednost može se pojaviti samo jedna vrijednost neključnog atributa:**

**Definicija:** Relacija se nalazi u 1NF ako su svi neključni atributi funkcijski zavisni o ključu relacije.

**Primjer 1:** Čest je slučaj da se cijene artikala vode u Eurima, a prikazuju u Kunama, na način da se množi iznos u Eurima puta tečaj Kune:

Tablica: **Knjige\_0**

| #Naslov       | Cijena u EURIMA | Tečaj Kune |
|---------------|-----------------|------------|
| SQL           | 50,00           | 7,55       |
| Baze podataka | 60,00           | 7,55       |
| DOS           | 30,00           | 7,55       |

Primarni ključ tablice **Knjige\_0** je **Naslov** (# ispred imena označava primarni ključ). Jasno je da cijena svake knjige ovisi o kojoj se knjizi radi, a to nam jednoznačno određuje naslov knjige. Prema tome, neključni atribut **cijena** ovisi o ključu. Tečaj Kuna – Euro, nipošto ne ovisi o naslovu knjige, dakle, tablica **Knjige\_0 nije u prvoj normalnoj formi**. Tablica koja nije u prvoj normalnoj formi ne može se niti zvati relacijom, jer po definiciji relacija treba biti barem u prvoj normalnoj formi.

**Rješenje:** Dekomponirati tablicu Knjiga\_0 u dvije tablice:

Tablica: **Knjige\_1**

| #Naslov       | Cijena u EURIMA |
|---------------|-----------------|
| SQL           | 50,00           |
| Baze podataka | 60,00           |
| DOS           | 30,00           |

Tablica: **Tečajna\_Lista\_1**

| #Datum      | Tečaj Kune |
|-------------|------------|
| 19.01.2018. | 7,55       |

#### Zaključak:

Tablica **Knjige\_1** je u 1NF.

Tablica **Tečajna\_Lista\_1** je u 1NF.


## Primjer 2:

Tablica: **Knjige\_0**

| Kat. oznaka | Naslov        | Broj str. | Izdavač | Pošta | Mjesto | Autori       |
|-------------|---------------|-----------|---------|-------|--------|--------------|
| 1           | Baze podataka | 162       | DRIP    | 10000 | ZAGREB | Varga        |
| 2           | SQL           | 419       | ZNAK    | 10000 | ZAGREB | Vujnović     |
| 3           | ORACLE 7      | 780       | ZNAK    | 10000 | ZAGREB | Hrenić       |
| 4           | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Biluš, Rode  |
| 5           | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Biluš, Kesić |

U tablici **Knjige\_0** upisana je adresa izdavača, ali isto tako prezimena svih autora nekih knjiga. Tablica nije u 1NF jer podaci u stupcu **autori** nisu atomični!

**Rješenje:** Rascijepiti neatomične podatke u stupcu **autori** dodavanjem redaka za svakog autora.

Tablica: **Knjige\_0A** (nema primarni ključ)


| Kat. oznaka | Naslov        | Broj str. | Izdavač | Pošta | Mjesto | Autor    |
|-------------|---------------|-----------|---------|-------|--------|----------|
| 1           | Baze podataka | 162       | DRIP    | 10000 | ZAGREB | Varga    |
| 2           | SQL           | 419       | ZNAK    | 10000 | ZAGREB | Vujnović |
| 3           | ORACLE 7      | 780       | ZNAK    | 10000 | ZAGREB | Hrenić   |
| 4           | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Biluš    |
| 4           | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Rode     |
| 5           | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Biluš    |
| 5           | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Kesić    |

Sada imamo atomične podatke te **moramo definirati primarni ključ** relacije. Da bismo ga pravilno odredili, moramo si postaviti pitanje „što određuje atribut **Broj stranica**“? Jasno je da broj stranica knjige ovisi o **Kataloškoj oznaci**. Primarni ključ mora imati jedinstvenu vrijednost, a u slučaju kada su jedan naslov napisala dvojica ili više autora, vrijednost atributa **Kataloška oznaka** se ponavlja.

Kada bi jedan naslov napisao samo jedan autor, tada bi ključ mogao biti jednostavan, činio bi ga samo atribut **Kataloška oznaka**:

Tablica: **Knjige\_1A** (jednostavan ključ)

| #Kat. oznaka | Naslov        | Broj str. | Izdavač | Pošta | Mjesto | Autor    |
|--------------|---------------|-----------|---------|-------|--------|----------|
| 1            | Baze podataka | 162       | DRIP    | 10000 | ZAGREB | Varga    |
| 2            | SQL           | 419       | ZNAK    | 10000 | ZAGREB | Vujnović |
| 3            | ORACLE 7      | 780       | ZNAK    | 10000 | ZAGREB | Hrenić   |
| 4            | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Biluš    |
| 5            | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Biluš    |

Međutim, neke knjige imaju više autora, pa moramo definirati složeni primarni ključ slaganjem atributa **Kataloška oznaka + Autor**.

Tablica: **Knjige\_1B** (složeni ključ)

| #Kat. oznaka | Naslov        | Broj str. | Izdavač | Pošta | Mjesto | #Autor   |
|--------------|---------------|-----------|---------|-------|--------|----------|
| 1            | Baze podataka | 162       | DRIP    | 10000 | ZAGREB | Varga    |
| 2            | SQL           | 419       | ZNAK    | 10000 | ZAGREB | Vujnović |
| 3            | ORACLE 7      | 780       | ZNAK    | 10000 | ZAGREB | Hrenić   |
| 4            | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Biluš    |
| 4            | Vrt           | 420       | LOGOS   | 21000 | SPLIT  | Rode     |
| 5            | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Biluš    |
| 5            | Kolači        | 360       | LOGOS   | 21000 | SPLIT  | Kesić    |

**Zaključak:** Sada svi neključni atributi ovise o ključu pa možemo kazati da smo doveli relaciju u 1NF.

Usprkos tome što smo relaciju doveli u 1NF, uočavamo značajne anomalije, prije svega u **redundantnosti** podataka, što vodi nepotrebnom trošenju medija za spremanje podataka (tvrdi disk) i vrlo je otežano mijenjanje tih podataka. Dakle, nije dovoljno dovesti relaciju u 1NF.



## 2NF – Druga normalna forma

Zavisnost neključnih atributa o dijelovima ključa je uzrok anomalija. Poželjna je potpuna funkcijska zavisnost o cijelom ključu, odnosno o svim dijelovima složenog ključa.

**Definicija:** Relacija se nalazi u 2NF ako su svi neključni atributi potpuno funkcijski zavisni o bilo kojem (mogućem) ključu relacije i to o svim dijelovima ključa.

Relacija **Knjige\_1A** ima jednostavan ključ, dakle čini ga samo jedan atribut i ona samim tim zadovoljava pravilo druge normalne forme. Dakle, možemo reći:

**Relacija je odmah u 2NF ako joj je ključ jednostavan.**

Relacija **Knjige\_1B** nije u 2NF jer neključni atributi **Pošta** i **Mjesto** ne ovise u potpunosti o ključu. Isto tako, pojedini autor određuje **Naslov**, ali ne i **izdavača**, jer isti autor može napisati **Naslov** i za drugog **izdavača**.

**Rješenje:** Napraviti dekompoziciju relacije tako da se neključni atributi koji ovise samo o dijelu ključa premjeste u novu relaciju, skupa s dijelovima ključa koji ih funkcijski određuje.

U našem primjeru, o dijelu ključa **Autor**, ne ovisi **izdavač**, pa prema tome niti atributi koji ovise o izdavaču. **Kataloška oznaka** određuje **autora**, pa ćemo izdvojiti attribute **Kataloška oznaka** i **Autor** u posebnu relaciju:

Tablica: **Knjige\_2**

| #Kat. oznaka | Naslov        | Broj str. | Izdavač | Pošta | Mjesto |
|--------------|---------------|-----------|---------|-------|--------|
| 1            | Baze podataka | 162       | DRIP    | 10000 | ZAGREB |
| 2            | SQL           | 419       | ZNAK    | 10000 | ZAGREB |
| 3            | ORACLE 7      | 780       | ZNAK    | 10000 | ZAGREB |
| 4            | Vrt           | 420       | LOGOS   | 21000 | SPLIT  |
| 5            | Kolači        | 360       | LOGOS   | 21000 | SPLIT  |

Tablica: **Knjige\_Autori\_2**

| #Kat. oznaka | #Autor   |
|--------------|----------|
| 1            | Varga    |
| 2            | Vujnović |
| 3            | Hrenić   |
| 4            | Biluš    |
| 4            | Rode     |
| 5            | Biluš    |
| 5            | Kesić    |

**Zaključak:** Nakon dekompozicije, obje su relacije u 2NF.

### 3NF – Treća normalna forma

Još uvijek nailazimo na redundantnost podataka.

Naime, **Izdavač** jednoznačno određuje **Poštu** i **Mjesto izdavača**. Štoviše, **Mjesto** je funkcijski određeno poštanskim brojem. Anomalija ovakvih podataka se ogleda u nepotrebnom trošenju medija za spremanje, kao i potrebom za promjenama u više redaka poštanskog broja i mjesta kad se izdavač preseli u drugo mjesto.

Relacija se nalazi u 3NF ako nijedan neključni atribut nije tranzitivno ovisan o bilo kojem ključu relacije.

Svaki neključni atribut mora ovisiti o ključu, cijelom ključu i ni o čemu drugom osim o ključu.

U našem primjeru nalazimo tranzitivne zavisnosti: **Izdavač** je određen **Kataloškim brojem**, te **Izdavač** funkcijski određuje **Poštu**. Dakle, **Pošta** ovisi o ključu, ali **tranzitivno** preko **Izdavača**. Nadalje, **Mjesto** ovisi o **Izdavaču** tranzitivno preko **Pošte**. Dvije tranzitivne ovisnosti uklanjamo pouzdanim dekomponiranjem relacije Knjige\_2 u više manjih relacija: **Knjige\_3**, **Izdavači\_3** i **Pošte\_3**:

Tablica: **Knjige\_3**

| #Kat. oznaka | Naslov        | Broj str. | Izdavač |
|--------------|---------------|-----------|---------|
| 1            | Baze podataka | 162       | DRIP    |
| 2            | SQL           | 419       | ZNAK    |
| 3            | ORACLE 7      | 780       | ZNAK    |
| 4            | Vrt           | 420       | LOGOS   |
| 5            | Kolači        | 360       | LOGOS   |

Tablica: **Izdavači\_3**

| Izdavač | Pošta |
|---------|-------|
| DRIP    | 10000 |
| ZNAK    | 10000 |
| LOGOS   | 21000 |

Tablica: **Pošte\_3**

| Pošta | Mjesto |
|-------|--------|
| 10000 | ZAGREB |
| 21000 | SPLIT  |

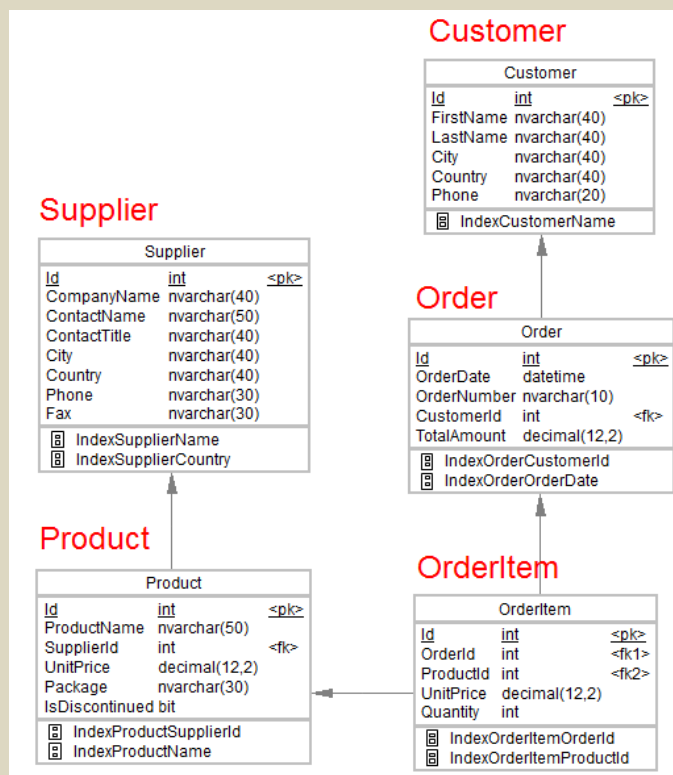
**Zadatak za vježbu:**

Predložite i u bilježnicu napišite relacijsku shemu BP za neku trgovinu koja u sebi sadrži podatke o: trgovini, kupcima, dobavljačima, robi, kategorijama, bankama, stopama PDV-a, računima...

Označite PK i FK.

Schema treba biti normalizirana do 3NF.

Primjer neke BP za trgovinu:



**19-20: VJEŽBA 4: RELACIJSKA ALGEBRA**

---

Radi se 1 školski sat. Piše se na papiru.

## 21-22: UVOD U SQL

Napravit ćemo bazu podataka neke trgovine. Ta baza će se zvati „**trgovina**“.  
U njoj se nalaze ove tablice:

### TRGOVCI

| sifra | prezime   | odjel |
|-------|-----------|-------|
| 1111  | Horvat    | P5    |
| 2222  | Kovač     | P3    |
| 3333  | Bistrović | P4    |
| 4444  | Horvat    | P1    |
| 5555  | Ban       | P5    |

### ODJELI

| sifra | naziv     |
|-------|-----------|
| P1    | Prehrana  |
| P2    | Tehnika   |
| P3    | Meso      |
| P4    | Namještaj |
| P5    | Odjeća    |

### KUPCI-HRV

| sifra | naziv   | rang |
|-------|---------|------|
| 008   | Trgopet | 1    |
| 001   | Maks    | 5    |
| 004   | Elcim   | 3    |
| 002   | Dionea  | 2    |
| 003   | ABC     | 1    |

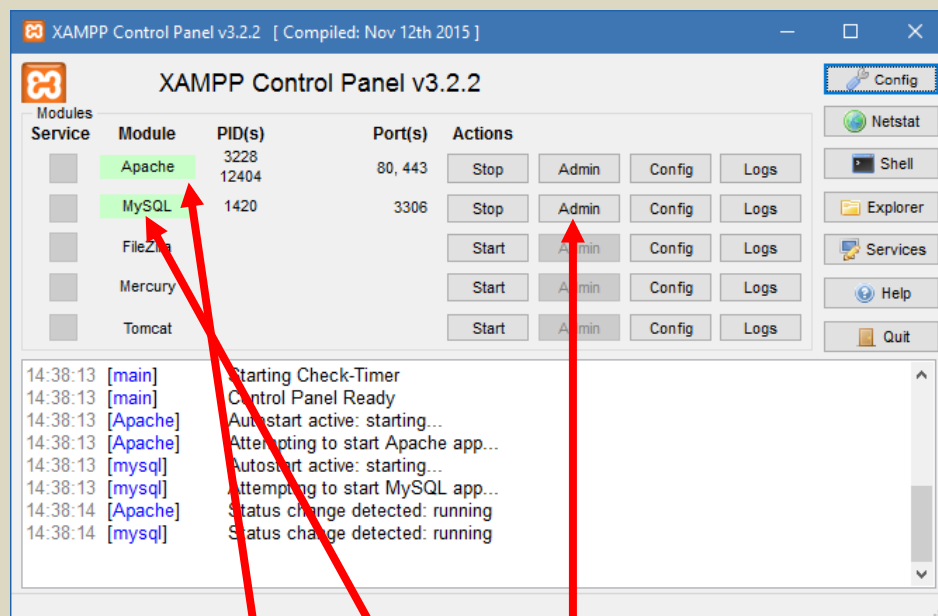
### RAČUNI

| br-racuna | kupac | iznos   | izdao |
|-----------|-------|---------|-------|
| 151       | 001   | 25,00   | 4444  |
| 152       | 004   | 1220,00 | 5555  |
| 153       | 8003  | 115,00  | 4444  |
| 154       | 002   | 47,00   | 3333  |
| 155       | 001   | 9,99    | 2222  |
| 156       | 8004  | 100,00  | 1111  |

### KUPCI-INO

| sifra | naziv   | rang |
|-------|---------|------|
| 8001  | DAX     | 1    |
| 8002  | Hungar  | 2    |
| 8003  | Osteria | 4    |
| 8004  | Vinoa   | 4    |
| 8005  | 3D plus | 3    |

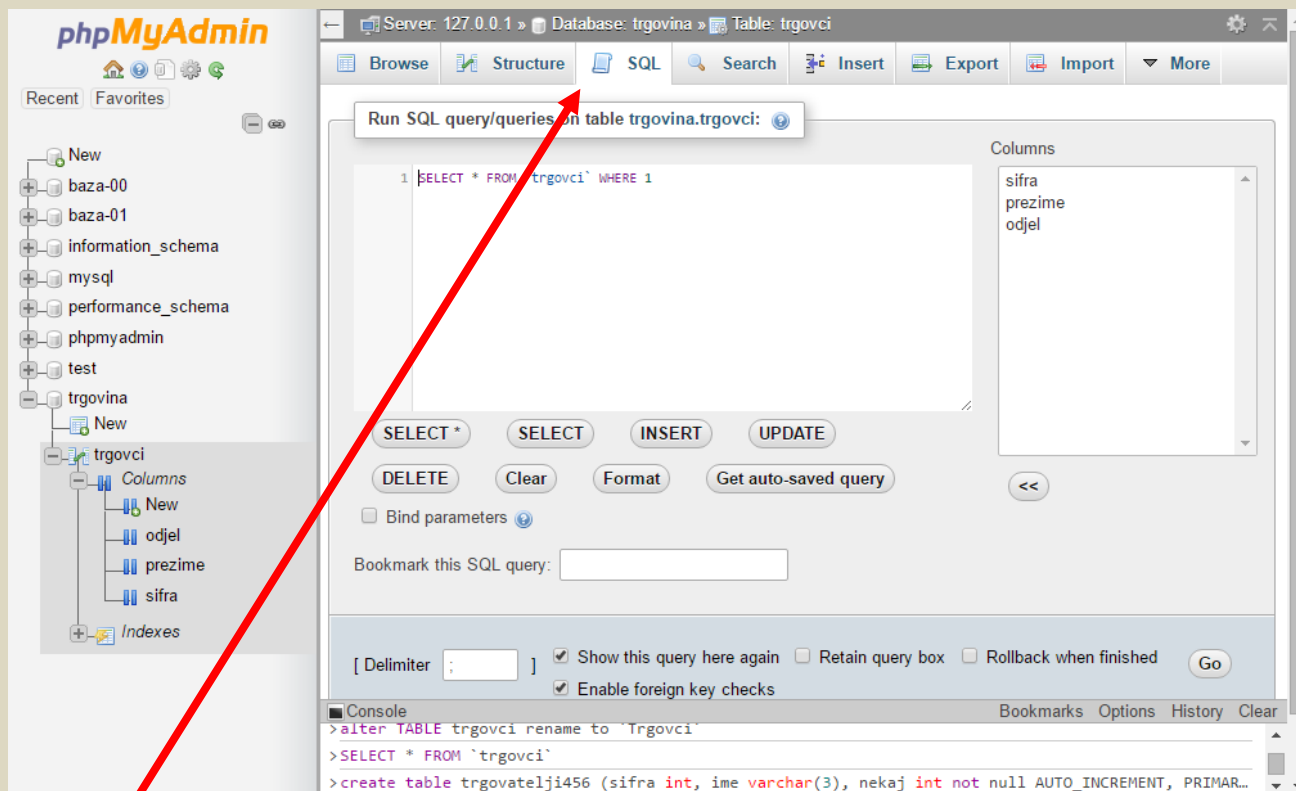
Pokrenemo XAMPP Control Panel:



Da bismo mogli raditi, servisi **APACHE** i **MYSQL** moraju biti pokrenuti!

Da bismo otvorili **phpMyAdmin** sučelje, trebamo kliknuti ovdje.

Otvorimo phpMyAdmin sučelje. Možemo raditi mišem, ali i pomoću SQL naredbi:

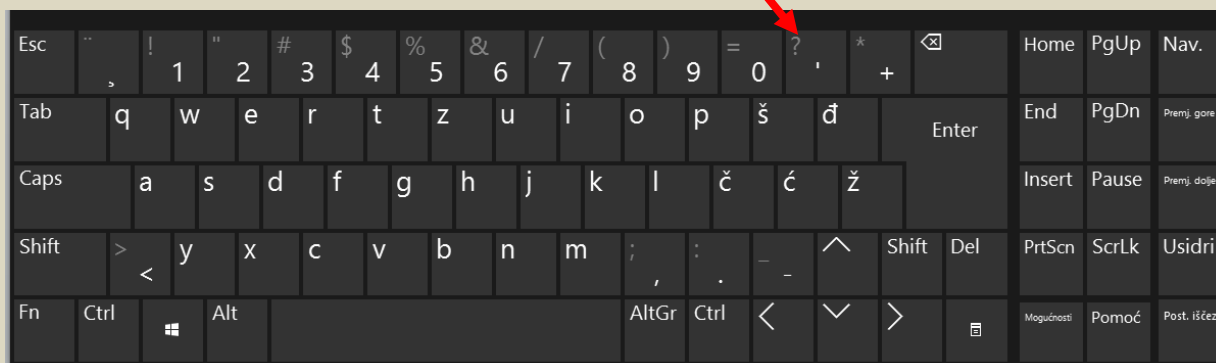


Da bismo mogli pisati naredbe, najprije moramo otvoriti karticu SQL.  
U polje za unos pišemo naredbe.  
Postoje već gotovi predlošci koji nam ubrzavaju pisanje pojedinih naredbi.

### PAZITE NA NAVODNIKE!

Ponekad su potrebni, ponekad nisu.

Uglavnom su to JEDNOSTRUKI NAVODNICI (tipka '?')



## 1. BAZA PODATAKA

### STVARANJE NOVE BAZE PODATAKA

```
CREATE DATABASE trgovina
```

ili

```
CREATE DATABASE trgovina DEFAULT CHARACTER SET cp1250 COLLATE cp1250_croatian_ci;
```

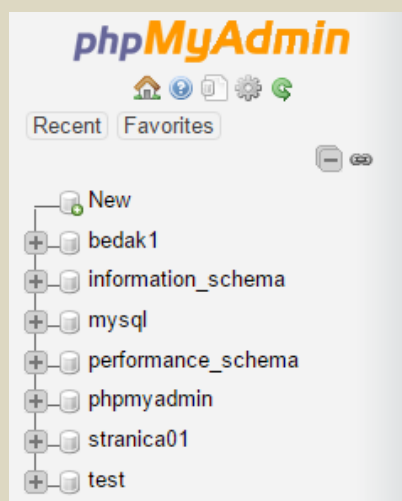
### BRISANJE BAZE PODATAKA

```
DROP DATABASE trgovina
```

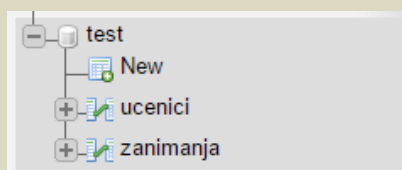
... ..

Kako provjeriti je li naša nova baza stvorena?

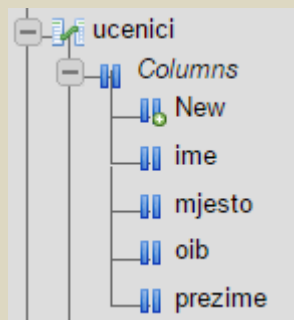
Jednostavno! Na lijevoj strani ekrana u phpMyAdminu vidimo popis svih baza na ovom serveru:



Ako želimo vidjeti koje sve tablice ta baza sadrži, trebamo kliknuti na **plus** pored imena baze:



Ako kliknemo na plus pored imena tablice, vidjet ćemo od kojih stupaca se ta tablica sastoji.



## 2. TABLICE

### STVARANJE NOVE TABLICE

```
CREATE TABLE trgovci
(
 sifra INT,
 prezime VARCHAR(20),
 odjel VARCHAR(2),
 PRIMARY KEY (sifra)
);
```

Napomena: može i ovako:

```
CREATE TABLE trgovina.trgovci...
```

### PREIMENOVANJE TABLICE

```
ALTER TABLE trgovci
 RENAME TO trgovatelji;
```

### BRISANJE TABLICE

```
DROP TABLE trgovatelji;
```

### UNOS PODATAKA U TABLICU

```
INSERT INTO trgovci
 (sifra,prezime,odjel)
VALUES (1,'aaa','111');
```

Da bismo vidjeli sadržaj neke tablice, moramo s lijeve strane kliknuti na njeno ime, a zatim ići na karticu Browse:



Server: 127.0.0.1 » Database: test » Table: zanimanja

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#)

⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are disabled.

✓ Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)

```
SELECT * FROM `zanimanja`
```

☐ Show all | Number of rows: 25 ▼ | Filter rows:

+ Options

| sifra | naziv                   |
|-------|-------------------------|
| 100   | Tehničar za računalstvo |

☐ Show all | Number of rows: 25 ▼ | Filter rows:

### 3. STUPCI

#### DODAVANJE NOVOG STUPCA U POSTOJEĆU TABLICU

```
ALTER TABLE trgovci
 ADD novi_stupac VARCHAR(25);
```

#### PREIMENOVANJE STUPCA

```
ALTER TABLE trgovci
 CHANGE COLUMN novi_stupac prebivaliste VARCHAR(15);
```

#### BRISANJE STUPCA IZ TABLICE

```
ALTER TABLE trgovci
 DROP COLUMN prebivaliste;
```

Za svaki slučaj:

**SPREMAJTE SI SQL SKRIPTE U TXT ILI DOCX DATOTEKU!**

#### PITANJA ZA ZADAĆU

2. Skinite program WAMP ili XAMPP s interneta (<http://www.wampserver.com/en/>) ili (<https://www.apachefriends.org/download.html>). Instalirajte ga na svoje računalo. Pokrenite sve servise u njemu i otvorite phpMyAdmin.
3. Pomoću SQL naredbi napravite bazu „**knjiznica**“ koja se sastoji od tablica **knjiznicari** (id, prezime, oib) i **ucenici** (mat\_br, prezime, razred).
4. Napišite SQL naredbu za:
  - a. Dodavanje novoga stupca 'hobi' u tablicu **ucenici**
  - b. Brisanje stupca 'oib' iz tablice **knjiznicari**
5. Napravite novu bazu „**carina**“, ali bez uporabe SQL-a. Pomoću SQL-a u toj bazi napravite tablicu **zakoni** (br\_zakona, naziv). Bez uporabe SQL-a u tu istu tablicu dodajte nove stupce (datum, potpisao, vrijedi)

## 23-24: SQL SORTIRANJE I FILTRIRANJE

Zadaci:

1. Napravite novu bazu „**farma\_b**“.
2. U toj bazi napravite 3 tablice:
  - a. **radnici** (id, ime, pozicija)
  - b. **zgrade** (sifra, naziv, površina)
  - c. **vrste\_zivotinja** (sifra, naziv\_vrste)
3. Radnici su:
  - a. 987654 Josip traktorist
  - b. 654321 Marija mužačica
  - c. 456654 Ivan sveznalica
4. Zgrade imaju dvije:
  - a. Zgrada1 205 m<sup>2</sup>
  - b. Zgrada2 1333 m<sup>2</sup>
5. Vrste životinja:

```
(
 101, 'krava',
 102, 'koza',
 103, 'slon',
 104, 'emu',
 105, 'noj',
 106, 'klokan',
 107, 'kokoš',
 108, 'šaran',
 109, 'zmija',
 110, 'deva',
 111, 'pura',
 112, 'guska',
 ... ovamo još dodajte 10 životinjskih vrsta ...
)
```
6. Prema SQL skriptama na Edmodu, napravite ove zadatke:
  - a. Sortirajte radnike prema imenu A-Ž
  - b. Dodajte novu zgradu: Zgrada3 855 m<sup>2</sup>
  - c. Ispišite nazive zgrada prema površini 9-0
  - d. Ispišite sve vrste životinja koje počinju slovom K
  - e. Dodajte novi stupac „kategorija“ u tablicu „radnici“ i ostavite ga praznog
  - f. Ispišite sve vrste životinja koje nisu slon, noj ni klokan

Napravite screenshote za zadatke 6. a-f ili nakon svakog uspješnog rješenja pozovite profesora da pogleda.

### PITANJA ZA ZADAĆU

1. Napravite novu bazu „**carina**“, ali bez uporabe SQL-a (pomoću miša). Pomoću SQL-a u toj bazi napravite tablicu **zakoni** (br\_zakona, naziv). Bez uporabe SQL-a u tu istu tablicu dodajte nove stupce (datum, potpisao, vrijedi)

**25-26: VJEŽBA 5: SQL 1**

---

Radi se 2 školska sata.

## 27-28: SQL JOIN I UPDATE

Zadaci:

1. Napravite novu bazu „spoj“.
2. U toj bazi napravite ovakve tablice:

### Zaposlenici

| ID_zaposlenika | Ime  | Prezime |
|----------------|------|---------|
| 0111           | Jura | Jurić   |
| 0112           | Pero | Perić   |
| 0113           | Štef | Štefić  |
| 0114           | Pajo | Patak   |

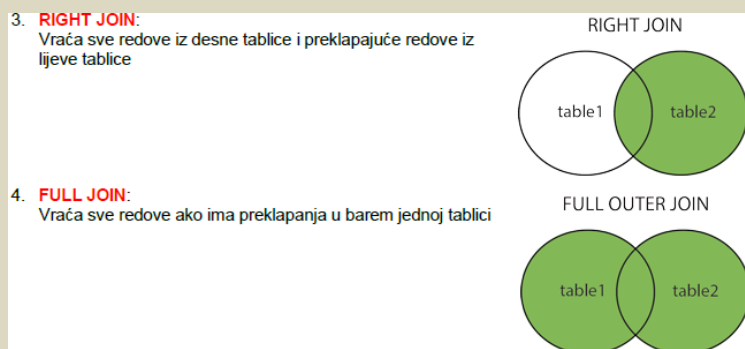
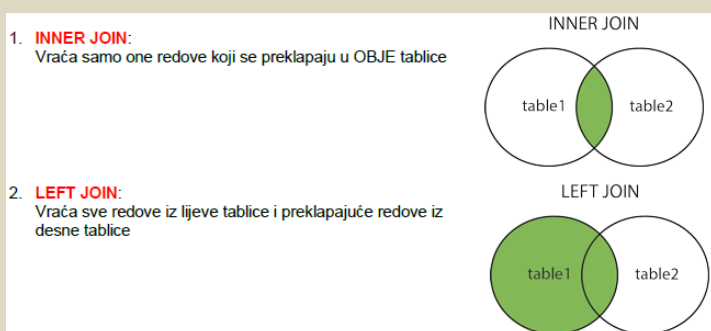
### Narudzbe

| ID_narudzbe | Proizvod | Zaposlenik |
|-------------|----------|------------|
| 234         | Printer  | 0111       |
| 657         | Stol     | 0113       |
| 865         | Ormar    | 0113       |

Pomoć:

```
SELECT prva.naziv,druga.mjesto
FROM prva
xxxx JOIN druga
ON prva.id=druga.id
```

3. Koji zaposlenici su naručili nešto i što su naručili? Oni koji nisu ništa naručivali ne smiju se ispisati.
4. Ispišite sve zaposlenike s njihovim narudžbama (ako nisu ništa naručili, treba se ispisati prazno polje).
5. Ispišite popis svih narudžbi i tko ih je naručio (ako naručitelj postoji).
6. Promijenite ime zaposlenika 0112 u Pedro Pedrić.



Korištene SQL naredbe mi pošaljite izravno u e-mail poruci na [marko.ljubek@gmail.com](mailto:marko.ljubek@gmail.com).

Imamo još zadataka za danas →

## 29-30: AGREGATNE FUNKCIJE. TRANSAKCIJE. REFERENCIJALNI INTEGRITET

Upotrebom SQL naredbi napravi sljedeće zadatke:

1. Napravi bazu podataka „baza25b“.

```
CREATE DATABASE `baza25b` DEFAULT CHARACTER SET cp1250 COLLATE cp1250_croatian_ci;
```

1. Napravi tablicu „mjesto“.
  - a. **pbr** – cijeli broj od 5 znakova / **naziv** – 25 znakova
2. Idi na [ovu](#) web stranicu i iskopiraj dobivene podatke.
3. Pokreni Notepad++. Zalijepi te podatke u novu datoteku. Tu datoteku spremi kao „mjesto1.txt“. U prvi red napišemo naslove stupaca (PBR i MJESTO).

Budući da SQL mora znati **delimitere** (razdjelnike) između pojedinih podataka, **mi moramo sve stupce jednako odvojiti**. To možemo napraviti pomoću razmaka, tabulatora, točke, zareza; dvotočke, točka-zareza...

Moramo malo razmisliti. Pogledajmo ovo:

|    |       |                      |
|----|-------|----------------------|
| 6  | 40313 | Sveti Martin na Muri |
| 7  | 40314 | Selnica              |
| 8  | 40315 | Mursko Središće      |
| 9  | 40317 | Podturen             |
| 10 | 40318 | Dekanovec            |
| 11 | 40319 | Belica               |
| 12 | 40320 | Donji Kraljevec      |
| 13 | 40321 | Mala Subotica        |

Trenutno su nam stupci PBR i NAZIV odvojeni razmakom. Mogli bismo reći da je naš delimiter razmak. Međutim, neka mjesta imaju više riječi u nazivu i između tih riječi se također nalazi razmak! Znači, ne možemo koristiti razmak kao delimiter u ovom primjeru. Moramo nešto drugo smisliti.

Neki od najčešćih delimitera: **\n** (novi red), **\r** (novi red), **\t** (tabulator)...

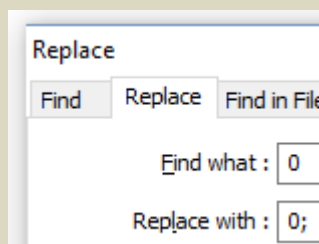
Znamo da brojeve u SQL-u ne moramo pisati pod navodnike. Mogli bismo sve nazive mjesta napisati pod navodnike... No, to bi bilo previše posla i dugo bi trajalo.

I dalje je glavno pitanje:

**KAKO UREDITI DOKUMENT DA SQL TOČNO ZNA  
GDJE POČINJE I ZAVRŠAVA JEDAN STUPAC,  
A GDJE POČINJE I ZAVRŠAVA DRUGI STUPAC?**

Najjednostavniji način (koji nije jako brz, ali nema boljega) jest da između stupaca stavimo neki znak različit od razmaka. Neka to bude **točka-zarez**.

Koristit ćemo Notepadovu funkciju *Find & Replace*:



Tu funkciju ćemo koristiti 10 puta. Zašto? Vidimo da su nam u prvom stupcu samo brojevi. Ti brojevi završavaju znamenkama 0, 1, 2, 3, 4, 5, 6, 7, 8 ili 9. Poslije tih brojeva dolazi razmak kojega mi želimo zamijeniti točka-zarezom. Zato ćemo pod *Find what* napisati znamenku i razmak, a pod *Replace with* ćemo napisati tu istu znamenku i točka-zarez! Taj postupak ćemo ponoviti 10 puta, dok ne pređemo kroz sve znamenke.

Dobili smo ovo:

```
5 40311;Lopatinec
6 40312;Štrigova
7 40313;Sveti Martin na Muri
8 40314;Selnica
9 40315;Mursko Središće
10 40317;Podturen
11 40318;Dekanovec
12 40319;Belica
13 40320;Donji Kraljevec
14 40321;Mala Subotica
```

Sada konačno imamo dobro formatirani dokument kojega možemo iskoristiti za automatsko popunjavanje tablice pomoću SQL-a!

4. U SQL upišemo ovu naredbu:

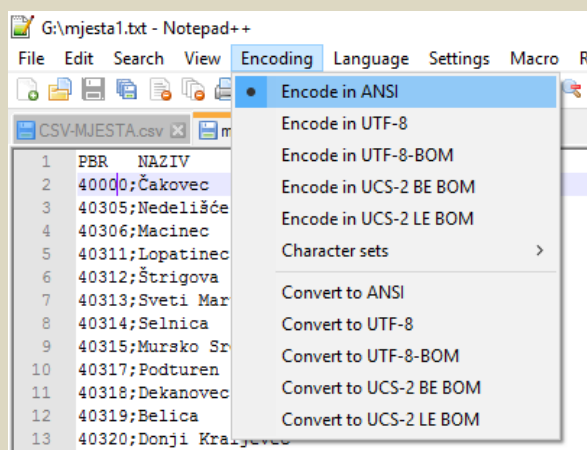
VAMA ĆE BITI NEKA DRUGA PUTANJA

```
LOAD DATA INFILE 'G:/mjesta1.txt' INTO TABLE mjesta
COLUMNS TERMINATED BY ';' ENCLOSED BY ''
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
```

5. Pogledamo kako nam sada izgleda naša tablica u bazi. Nije dobro. Ne ispisuju se hrvatski znakovi.
6. Izbrišemo sve podatke u tablici:

```
DELETE FROM mjesta
```

7. U Notepadu postavimo kodiranje sadržaja u ANSI formatu i još jednom spremimo dokument.



8. Opet u SQL upišemo istu naredbu kakva je u koraku 4.
9. Sada idemo na karticu **Export** u phpMyAdminu. Tu si odaberemo opcije za izvoz podataka iz tablice u novu datoteku i isprobamo!

## AGREGATNE FUNKCIJE I TRANSAKCIJE

Stvaranje baze:

```
CREATE DATABASE `baza29a` DEFAULT CHARACTER SET cp1250 COLLATE
cp1250_croatian_ci;
```

Stvaranje tablice **zaposlenici**:

```
create table zaposlenici
(
 id int not null auto_increment,
 ime varchar(15),
 razred int(1),
 spol varchar(1),
 starost int,
 zarada int,
 primary key (id)
)
```

**zaposlenici.txt:** (napravite si ovakav dokument u Notepadu)

```
IME;RAZRED;SPOL;STAROST;ZARADA
```

```

```

```
Ante;2;M;16;350
```

```
Žan;3;M;17;340
```

```
Ivana;1;Ž;15;1000
```

```
Kevin;1;M;15;140
```

```
Katarina;3;Ž;17;1500
```

```
Boris;2;M;16;990
```

```
Leon;1;M;15;555
```

```
Luka;2;M;16;60
```

```
Martina;4;Ž;18;700
```

```
Renato;4;M;18;800
```

```
Petra;3;Ž;17;1500
```

```
Patrik;2;M;16;200
```

Uvoz podataka iz txt datoteke u tablicu:

```
LOAD DATA INFILE 'G:/zaposlenici.txt' INTO TABLE zaposlenici
COLUMNS TERMINATED BY ';' ENCLOSED BY ''
LINES TERMINATED BY '\n'
IGNORE 2 ROWS
(ime, razred, spol, starost, zarada)
set id=NULL
```



**ZADACI:**

---

Prebroji koliko imamo zaposlenika.

```
SELECT COUNT(*) FROM zaposlenici
SELECT COUNT(*) as Koliko FROM zaposlenici
```

---

Zbroji koliko su svi zaposlenici zajedno zaradili.

```
SELECT SUM(zarada) FROM zaposlenici
```

---

Zbroji koliko su zaradile zaposlenice (Ž).

```
SELECT SUM(zarada) FROM zaposlenici WHERE spol='Ž'
```

---

Koliko su svi zaposlenici prosječno zaradili?

```
SELECT AVG(zarada) FROM zaposlenici
```

---

Koja je najveća i najmanja zarada i tko ju je zaradio?

```
SELECT MAX(zarada) FROM zaposlenici
SELECT ime,zarada FROM zaposlenici WHERE zarada=(SELECT MIN(zarada) FROM
zaposlenici)
SELECT ime,zarada FROM zaposlenici WHERE zarada=(SELECT MAX(zarada) FROM
zaposlenici)
```

## SQL TRANSAKCIJE

**Transakcije** se koriste kada želite više naredbi izvršiti u jednom komadu.

Zapamtite si koju zaradu imaju Kevin i Petra.

Kevin je od svoje zarade uzeo 111 kn i dao ih je Petri:

```
START TRANSACTION;
 UPDATE zaposlenici SET zarada=zarada-111 WHERE ime='Kevin';
 UPDATE zaposlenici SET zarada=zarada+111 WHERE ime='Petra';
 SELECT * FROM zaposlenici
COMMIT;
```

Sada provjerite je li se Kevinova i Petrina zarada promijenila.

## SQL – REFERENCIJALNI INTEGRITET

Napraviti novu bazu **Refint** i u njoj 2 tablice: **marke** i **modeli**.

```
CREATE TABLE marke
(
 idmar int not null auto_increment,
 naziv_marke varchar(10) not null,

 PRIMARY KEY(idmar)
)
```

```
CREATE TABLE modeli
(
 idmod int not null auto_increment,
 marka int not null,
 naziv_modela varchar(10) not null,
 broj_sjedala int(1),

 PRIMARY KEY(idmod),
 FOREIGN KEY(marka) REFERENCES marke(idmar)
 ON DELETE xxxxxxxx ON UPDATE xxxxxxxx
)
```

Upišite barem 4 marke i za svaku marku po barem 2 modela automobila.

Umjesto **xxxxxxx** isprobavajte ove opcije: **RESTRICT**, **CASCADE**, **SET NULL**.

Da biste isprobali kako te naredbe rade, probajte ažurirati ili izbrisati neki podatak iz tablice **modeli**, npr.:

```
DELETE FROM modeli WHERE broj_sjedala=3
```

```
START TRANSACTION;
```

```
UPDATE modeli SET broj_sjedala=broj_sjedala+10
```

```
WHERE marka='1';
```

```
UPDATE modeli SET marka='ABC'
```

```
WHERE marka='2';
```

```
SELECT * FROM modeli
```

```
COMMIT;
```

```
SELECT * FROM marke
```

Primjer za vježbu: <http://www.mysqltutorial.org/mysql-on-delete-cascade/>

## POMOĆ: UNOS DJELOMIČNIH PODATAKA U TABLICU

Ponekad nemate sve podatke koji su potrebni u nekoj tablici, ali svejedno želite napraviti unos. Postoji više načina za to.

Recimo da imate tablicu UCENICI:

```
create table UCENICI (
 id int not null AUTO_INCREMENT,
 ime varchar(15),
 prezime varchar (20),
 primary key (id)
)
```

Stupac **id** smo složili kao AUTO\_INCREMENT, a to znači da će se on automatski povećavati za 1 u svakom novom retku. Zbog toga, mi njega zapravo i ne moramo unositi:

```
insert into UCENICI(id,ime,prezime) values ('Donald','Trump')
```

Međutim, javlja nam se greška:

```
insert into ucenici(id,ime,prezime) values ('Donald','Trump')
```

MySQL said: ?

#1136 - Column count doesn't match value count at row 1

Ako malo bolje pogledamo naredbu, vidimo da smo nakon naziva tablice UCENICI u zagradi napisali 3 stupca, a želimo upisati samo 2 podatka (riječ 'Donald' + riječ 'Trump'). Sustav ne zna u koje stupce bi te pojmove trebao staviti pa zato javlja grešku.

Popravimo sada naredbu na način da maknemo **id** s popisa stupaca:

```
insert into UCENICI(ime,prezime) values ('Donald','Trump')
```

Sada je dobro:

| id | ime    | prezime |
|----|--------|---------|
| 17 | Donald | Trump   |

Vidimo da se **id** dodao automatski, bez našeg unosa.

Isto tako možemo upisati samo jedan podatak:

```
insert into UCENICI(prezime) values ('Krump')
```

| id | ime    | prezime |
|----|--------|---------|
| 17 | Donald | Trump   |
| 18 | NULL   | Krump   |

Vidimo da je podatak dodan, a prazna polja su popunjena znakom NULL.

Podatke u tablicu ne moramo unositi po istom redoslijedu kako su napravljeni stupci:

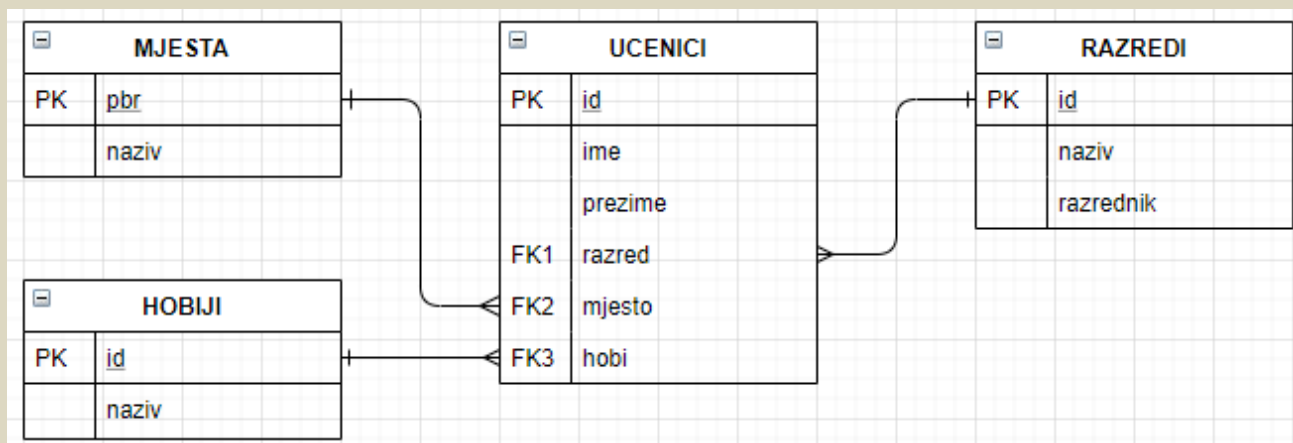
```
insert into UCENICI(prezime,ime) values ('Kennedy','John')
```

| id | ime    | prezime |
|----|--------|---------|
| 17 | Donald | Trump   |
| 18 | NULL   | Krump   |
| 19 | John   | Kennedy |

## ZA VJEŽBU: SQL – UVOZ I KLJUČEVI

Sljedeći zadatak treba napraviti za vježbu.

Zadatak NIJE OBAVEZAN, ali bi bilo dobro da ga pročete jer će sličan zadatak biti na sljedećim vježbama za ocjenu.



Napraviti 4 tablice:

**UCENICI** (id, ime, prezime, razred, mjesto, hobi)

**HOBIJI** (id, naziv)

**MJESTA** (pbr, naziv)

**RAZREDI** (id, naziv, razrednik)

**id** treba staviti na **AUTO\_INCREMENT** !  
id int not null AUTO\_INCREMENT

Svaka tablica treba imati svoj primarni ključ.

Na kraju treba dodati vanjske ključeve ([link](#)).

Zatim treba napraviti tekstualne datoteke:

- popis\_razreda.txt (možete iskoristiti [ovo](#))
- popis\_hobija.txt (možete iskoristiti [ovo](#))
- popis\_mjesta.txt (možete iskoristiti [ovo](#))

... i uvesti podatke iz tih datoteka u pripadajuće tablice!

~~U Word dokument mi stavite:~~

- a) SQL naredbe za stvaranje sve 4 tablice (s naredbama za ključeve).
- b) Screenshote sva 3 txt dokumenta.
- c) SQL naredbe za uvoz tih dokumenata u pripadajuće tablice.
- d) Izgled tablica MJESTA, HOBIJI i RAZREDI nakon uvoza.
- e) Probajte u tablicu UCENICI upisati 3 učenika. Možete li? Ima li kakvih ograničenja? Dvojici pod hobi stavite 'Chess'.
- f) Pomoću naredbe UPDATE svakom učeniku čiji je hobi 'Chess' promijenite ime u Franjo.

Ako netko baš želi, može mi rješenje poslati na e-mail.

**31-32: VJEŽBA 6: SQL 2**

---

Radi se 2 školska sata.

## 33-34: SQL I PHP 1

Danas ćemo naučiti kako povezati web stranicu i bazu podataka.

Trebat će nam 3 stvari:

- XAMPP (ili WAMP) kao virtualni server na kojem će se nalaziti MySQL baza podataka
- Web preglednik (npr. Chrome)
- Text editor (npr. Notepad++)

Koraci:

- Pokrenemo XAMPP i napravimo si malu bazu podataka koju ćemo nazvati **vjetar**. U njoj stvorimo i popunimo jednu tablicu **sportasi**.

```
CREATE DATABASE vjetar DEFAULT CHARACTER SET utf8 COLLATE utf8_croatian_ci;

CREATE TABLE sportasi (
 rb INT AUTO_INCREMENT PRIMARY KEY,
 ime VARCHAR(15),
 prezime VARCHAR(20),
 mjesto VARCHAR(25),
 hobi VARCHAR(20)
);

INSERT INTO sportasi(ime,prezime,mjesto,hobi) VALUES
('Lionel','Messi','Barcelona','nogomet'), ('LeBron','James','Los
Angeles','košarka'), ('Roger','Federer','Bern','tenis'), ('Tiger','Woods','New
York','golf'), ('Usain','Bolt','Kingston Town','trčanje'),
('Ivano','Balić','Zagreb','rukomet'), ('Joža','Bećar','Strmec','nogomet'),
('Janica','Kostelić','Zagreb','skijanje'),
('Goran','Ivanišević','Split','tenis'),
('Cristiano','Ronaldo','Torino','nogomet'),
('Michael','Jordan','Chicago','košarka');
```

- Po defaultu, dok radimo u phpMyAdminu, mi smo prijavljeni kao korisnik pod nazivom **root** i imamo sva moguća prava.
- Stvorit ćemo 3 nova korisnika i svakome dati različita prava. Svaki će korisnik imati i lozinku:

```
CREATE USER 'stari_gazda'@'localhost' IDENTIFIED BY 'stari123';
CREATE USER 'mladi_gazda'@'localhost' IDENTIFIED BY 'mladi123';
CREATE USER 'radnik'@'localhost' IDENTIFIED BY 'radnik123';
```

- Za svaki slučaj, prvo ćemo svim tim korisnicima oduzeti sva prava (naredba REVOKE):

```
REVOKE ALL PRIVILEGES ON vjetar.* FROM 'stari_gazda'@'localhost';
REVOKE ALL PRIVILEGES ON vjetar.* FROM 'mladi_gazda'@'localhost';
REVOKE ALL PRIVILEGES ON vjetar.* FROM 'radnik'@'localhost';
```

- ... a zatim ćemo im dati samo ona prava koja im želimo dati (naredba GRANT):

```
-- STARI GAZDA MOŽE SVE:
GRANT ALL PRIVILEGES ON vjetar.* TO 'stari_gazda'@'localhost';
-- MLADI GAZDA NE MOŽE BRISATI (DELETE i DROP):
GRANT SELECT, INSERT, UPDATE ON vjetar.* TO 'mladi_gazda'@'localhost';
-- RADNIK MOŽE SAMO GLEDATI:
```

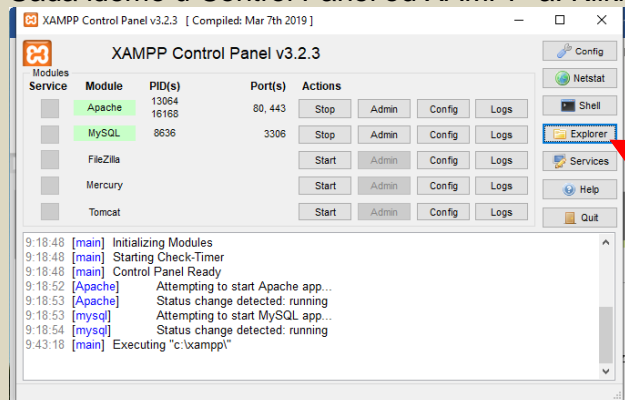
```
GRANT SELECT ON vjetar.* TO 'radnik'@'localhost';
```

6. Još si sada stvorimo jednu tablicu koju ćemo na kraju probati izbrisati:

```
CREATE TABLE bezvezna (
 ime varchar(14),
 nekaj decimal(3,2)
);
```

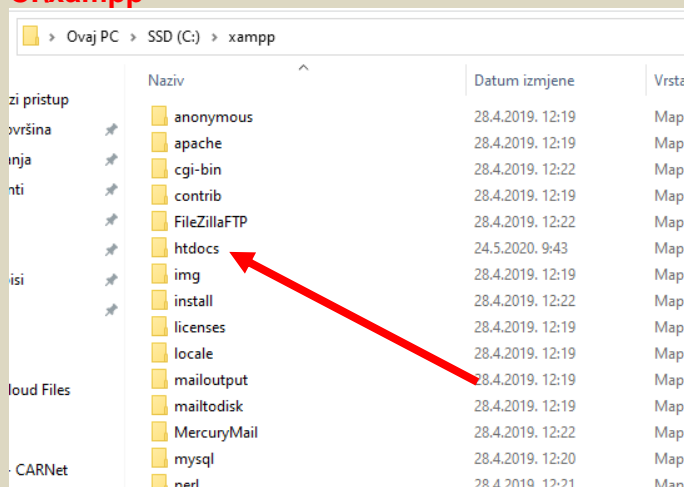
7. S izradom baze smo gotovi.

8. Sada idemo u Control Panel od XAMPP-a. Kliknemo na gumb Explorer.



9. Otvorit će nam se novi prozor u Windowsu koji će nas odvesti na putanju na kojoj je instaliran XAMPP, a to je po defaultu (ako niste ništa mijenjali prilikom instaliranja)

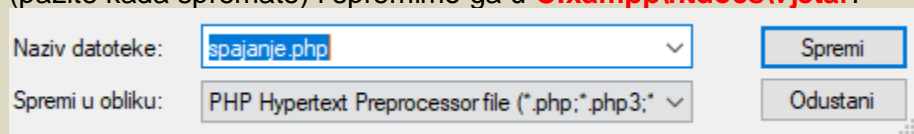
**C:\xampp**



10. Uđemo u mapu **htdocs**. TA MAPA PREDSTAVLJA NAŠ **LOCALHOST**. Znači, ako u Chrome upišemo pojam 'localhost' otvorit će nam se web stranica koja se nalazi u toj mapi.

11. U toj mapi napravimo novu mapu i nazovemo ju **vjetar**. Uđemo u tu mapu.

12. Sada pokrenemo Notepad++ i stvorimo novi PHP dokument. PHP je jezik za izradu dinamičkih web stranica (koje mogu komunicirati s bazama podataka), a unutar PHP-a možemo bez problema koristiti i HTML i CSS i SQL. Taj dokument nazovemo **spajanje.php** (pazite kada spremate) i spremimo ga u **C:\xampp\htdocs\vjetar**!



13. Spajanje web stranice na bazu podataka nije komplicirano i sastoji se samo od nekoliko linija koda u PHP-u:

*localhost* je server    *radnik* je username    *radnik123* je password    *vjetar* je baza

```
<?php
// Povezivanje:
$conn=mysqli_connect('localhost','radnik','radnik123','vjetar');

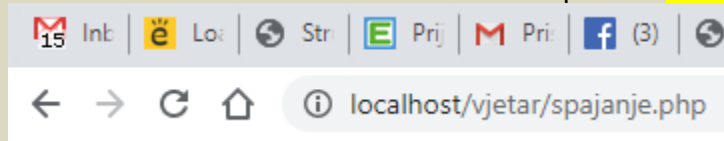
// Za hrvatske znakove:
mysqli_set_charset($conn,"utf8");

// Provjera za grešku:
if(mysqli_connect_errno()){
 echo "Dogodila se greška prilikom spajanja na bazu...";
 echo mysqli_connect_errno();
 die();
}
else{
 echo "Uspješno povezano!
";
}

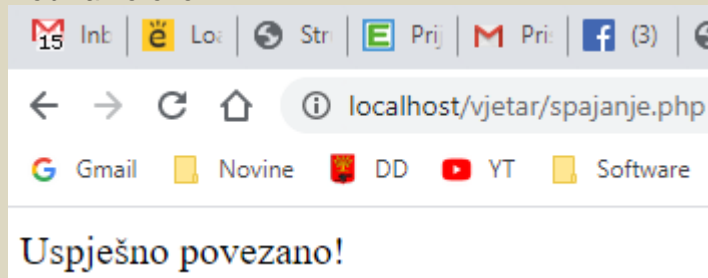
// Zatvaranje veze:
mysqli_close($conn);
?>
```

14. Spremimo tu datoteku i prebacimo se u Chrome.

15. U Chromeovu traku za unos URL adrese upišemo **localhost/vjetar/spajanje.php**



16. Dobivamo ovo:



17. Uspješno smo povezali web stranicu i bazu podataka.

18. Sada se možemo posvetiti izvršavanju SQL upita preko web stranice. Za početak ćemo SQL upit pisati izravno u izvorni kôd web stranice.



**20. PRVI UPIT: Na web stranicu ispisati sve podatke iz tablice 'sportaši'.**

Samo u nastavku datoteke **spajanje.php**, u retku 38 upišemo ovo:

```
// 01. ispis svih sportaša:
$upit = "SELECT * FROM sportasi";
$resultat = mysqli_query($conn, $upit);

if (mysqli_num_rows($resultat) > 0) {
 // ispisujemo jedan po jedan red iz tablice:
 while($row = mysqli_fetch_assoc($resultat)) {
 echo "Redni broj: " . $row["rb"];
 echo "Ime i prezime: " . $row["ime"] . " " . $row["prezime"];
 echo "Mjesto prebivališta: " . $row["mjesto"] ;
 echo "Hobi: " . $row["hobi"];
 echo "
";
 }
}
else {
 echo "0 pronađenih rezultata...";
}
```

\*napomena: PHP jezik je vrlo sličan C++ jeziku.

Na kraju retka dolazi **točka-zarez**

Petlje i funkcije imaju istu sintaksu

Varijable započinju znakom **\$**

Naredba za ispis na ekran je **echo "xyz"**

Da bismo spojili 2 elementa u jednoj **echo** naredbi, između njih stavljamo **točku**

Dobiveni rezultat:

Uspješno povezano!

Redni broj: 1Ime i prezime: Lionel MessiMjesto prebivališta: BarcelonaHobi: nogomet  
Redni broj: 2Ime i prezime: LeBron JamesMjesto prebivališta: Los AngelesHobi: košarka  
Redni broj: 3Ime i prezime: Roger FedererMjesto prebivališta: BernHobi: tenis  
Redni broj: 4Ime i prezime: Tiger WoodsMjesto prebivališta: New YorkHobi: golf  
Redni broj: 5Ime i prezime: Usain BoltMjesto prebivališta: Kingston TownHobi: trčanje  
Redni broj: 6Ime i prezime: Ivano BalićMjesto prebivališta: ZagrebHobi: rukomet  
Redni broj: 7Ime i prezime: Joža BećarMjesto prebivališta: StrmecHobi: nogomet  
Redni broj: 8Ime i prezime: Janica KostelićMjesto prebivališta: ZagrebHobi: skijanje  
Redni broj: 9Ime i prezime: Goran IvaniševićMjesto prebivališta: SplitHobi: tenis  
Redni broj: 10Ime i prezime: Cristiano RonaldoMjesto prebivališta: TorinoHobi: nogomet  
Redni broj: 11Ime i prezime: Michael JordanMjesto prebivališta: ChicagoHobi: košarka

**22. DRUGI UPIT: Na web stranicu ispisati samo sportaše iz Zagreba.**

Sada neka se podaci ispišu u obliku tablice.

Samo u nastavku datoteke **spajanje.php** upišemo ovo:

```
// 02. ispis samo onih sportaša koji su iz Zagreba:
$upit = "SELECT ime, prezime, mjesto FROM sportasi WHERE mjesto='Zagreb'";
$resultat = mysqli_query($conn, $upit);

if (mysqli_num_rows($resultat) > 0) {
 // stvaramo HTML tablicu:
 echo "<table border='1'>";
 echo "<tr>";
 echo "<td>IME</td>";
 echo "<td>PREZIME</td>";
 echo "<td>MJESTO</td>";
 echo "</tr>";
 // ispisujemo podatke u HTML tablicu:
 while($row = mysqli_fetch_assoc($resultat)) {
 echo "<tr>";
 echo "<td>" . $row["ime"] . "</td>";
 echo "<td>" . $row["prezime"] . "</td>";
 echo "<td>" . $row["mjesto"] . "</td>";
 echo "</tr>";
 }
 echo "</table>";
}
else {
 echo "0 pronađenih rezultata...";
}
```

Dobiveni rezultat:

| IME    | PREZIME  | MJESTO |
|--------|----------|--------|
| Ivano  | Balić    | Zagreb |
| Janica | Kostelić | Zagreb |

## 23. TREĆI UPIT:

- Može li korisnik 'radnik' obrisati tablicu 'bezvezna'?
- Može li korisnik mladi\_gazda obrisati tablicu 'bezvezna'?
- Može li korisnik 'stari\_gazda' obrisati tablicu 'bezvezna'?

Da bismo dobili odgovore na ova pitanja, morat ćemo u ovom dokumentu (*spajanje.php*) 3 puta mijenjati pristupne podatke:

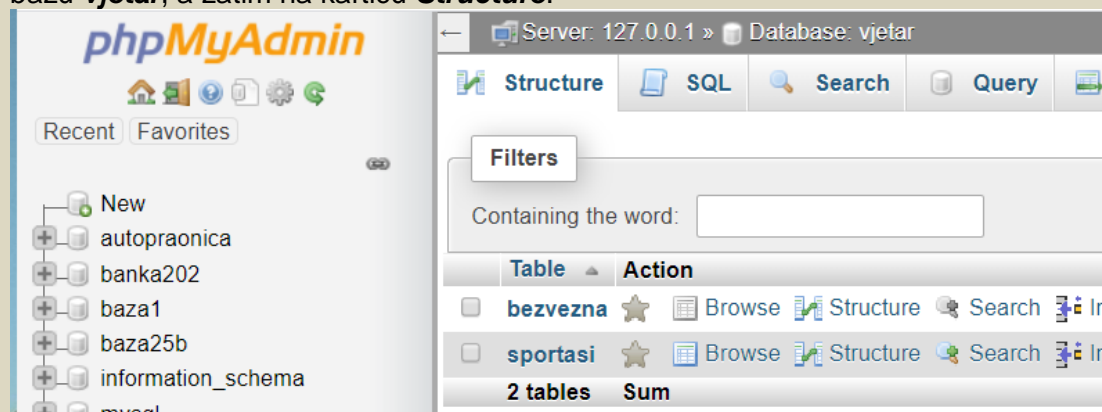
- a) Prvi pokušaj – prijavljujemo se kao **radnik**

```
<?php
// Povezivanje:
$conn=mysqli_connect('localhost','radnik','radnik123','vjetar');
```

Dopišemo sljedeću naredbu i spremimo web stranicu.

```
// 03. brisanje tablice 'bezvezna':
$upit = "DROP TABLE bezvezna";
$resultat = mysqli_query($conn, $upit);
```

Idemo u Chrome i tamo osvježimo (F5) web stranicu *spajanje.php*. Izgleda kao da se ništa novoga nije dogodilo. Prisjetimo se da smo, prilikom davanja prava korisnicima, korisniku 'radnik' dali pravo samo za korištenje naredbe SELECT i nijedne druge. Zbog toga bi tablica 'bezvezna' i nakon izvršavanja gornjeg upita trebala ostati u bazi neoštećena. Kako to provjeriti? Jednostavno – vratimo se u phpMyAdmin, kliknemo na bazu **vjetar**, a zatim na karticu **Structure**:



Vidimo da je tablica 'bezvezna' i dalje u bazi.

- b) Drugi pokušaj – prijavljujemo se kao **mladi\_gazda**

```
<?php
// Povezivanje:
$conn=mysqli_connect('localhost','mladi_gazda','mladil23','vjetar');
```

Idemo u Chrome i tamo osvježimo (F5) web stranicu *spajanje.php*. I dalje izgleda kao da se ništa novoga nije dogodilo. Provjerom u phpMyAdminu vidimo da je tablica 'bezvezna' i dalje u bazi. To je zato jer niti korisniku 'mladi\_gazda' nismo dali pravo na korištenje naredbi DROP i DELETE.

- c) Treći pokušaj – prijavljujemo se kao **stari\_gazda**

```
<?php
// Povezivanje:
$conn=mysqli_connect('localhost','stari_gazda','staril23','vjetar');
```

Idemo u Chrome i tamo osvježimo (F5) web stranicu *spajanje.php*. Provjerom u phpMyAdminu vidimo da je tablica 'bezvezna' konačno obrisana. To je zato jer smo korisniku 'stari\_gazda' dali pravo na korištenje SVIH naredbi. **Na taj način funkcioniraju administratori i obični korisnici na web stranicama.**

**ZADAĆA:** HTML, CSS, C++ i SQL znate... Poigrajte se svojim dosadašnjim znanjem i nadopunite si bazu nekim novim tablicama, a web stranicu probajte čim više obogatiti i uljepšati.

## 35-36: SQL I PHP 2

(nastavak od prošlih vježbi)

Sada znate:

- Služiti se XAMPP-om
- Napraviti SQL bazu podataka
- Napraviti web stranicu u željenoj mapi (C:\xampp\htdocs)
- Povezati web stranicu s bazom
- Dizajnirati web stranicu po željama

### Vaš današnji zadatak:

Izradite SQL bazu o knjižnici. To već znate – napravite npr. tablice UCENICI (id,ime,prezime), KNJIGE (id,naslov,autor,brstr), AUTORI (id,ime,prezime,drzava), KNJIZNICARI (id,ime,prezime), POSUDBE (id,datum,ucenik,knjiznicar,knjiga).

Svaku tablicu popunite s po 5 zapisa.

Budući da znate HTML i CSS, napravite jednu web stranicu (ili više njih povezanih hiperlinkovima!). Web stranicu povežite s bazom i na njoj neka se prikazuju rješenja raznih upita, npr.:

- Ispis svih knjiga u knjižnici
- Ispis svih učenika koji se prezivaju Horvat
- Ispis samo njemačkih autora
- Koliko stranica zajedno imaju imaju knjige čiji naslov počinje slovom A
- Sve posudbe prije 1.6.
- ...

### Treba napraviti 5 upita!

Web stranicu uredite pomoću CSS-a.

Za primjer, ovako sam ja uredio prošlotjedni zadatak:

The screenshot shows a web application with a light blue background. At the top, there's a navigation bar with links like 'Trenutno korišteno neaj charset', 'Uspješno povezano', and 'utf8'. Below this, the main heading is 'ISPIS SVIH SPORTAŠA (pomoću div klase):'. Underneath is a table with 11 rows of athlete data. Below the table, there's another heading 'ISPIS ZAGREBAČKIH SPORTAŠA (pomoću tablice):' followed by a table with 3 columns: IME, PREZIME, and MJESTO. At the bottom, there's a section 'ISPROBAVANJE AGREGATNIH FUNKCIJA:' with a table showing aggregate statistics.

| Redni broj | ime i prezime     | Mjesto prebivališta | Hobi     | Starost |
|------------|-------------------|---------------------|----------|---------|
| 1          | Lionel Messi      | Barcelona           | nogomet  | 28      |
| 2          | LeBron James      | Los Angeles         | košarka  | 24      |
| 3          | Roger Federer     | Bern                | tenis    | 41      |
| 4          | Tiger Woods       | New York            | golf     | 33      |
| 5          | Usain Bolt        | Kingston Town       | boks     | 25      |
| 6          | Ivano Balić       | Zagreb              | rukomet  | 14      |
| 7          | Joža Bačar        | Stmec               | nogomet  | 22      |
| 8          | Janica Kostelić   | Zagreb              | skijanje | 21      |
| 9          | Goran Ivanišević  | Split               | tenis    | 35      |
| 10         | Cristiano Ronaldo | Torino              | nogomet  | 33      |
| 11         | Michael Jordan    | Chicago             | košarka  | 11      |

| IME    | PREZIME  | MJESTO |
|--------|----------|--------|
| Ivano  | Balić    | Zagreb |
| Janica | Kostelić | Zagreb |

| ISPROBAVANJE AGREGATNIH FUNKCIJA: |                        |
|-----------------------------------|------------------------|
| Ukupna starost svih sportaša:     | 287                    |
| Prosječna starost svih sportaša:  | 26.0909                |
| Najmlađi je:                      | 11, a najstariji je 41 |
| Najmlađi se zove:                 | Michael Jordan         |

\* Nije obavezno, ali bilo bi lijepo napraviti **formu** s poljem za unos u kojega se upiše neki pojam, a zatim se taj pojam prenese u SQL upit (u izvornom kôdu web stranice) i upit se izvrši prema njemu. [npr. upišete „Ivan“ i nakon pritiska gumba „Submit“ se ispišu svi autori kojima je ime „Ivan“]

Pomoć za umetanje CSS stilova u echo naredbu:

```
echo <div class="stil">Neki sadržaj</div>
```

Budući da je to dinamička stranica u PHP-u, nećete mi je moći poslati, nego ćete mi poslati:

1. jedan screenshot iz phpMyAdmina iz kojeg se vidi da ste napravili bazu
2. barem jedan screenshot PHP kôda
3. barem jedan screenshot web stranice

Pošaljite mi na e-mail u roku tjedan dana (15.6. ili 22.6.).