Ispit iz Baza podataka
9. srpnja 2018. **(50 bodova)**

Baza podataka **streamservis** sadrži podatke o filmovima (***film)*** i članovima (***clan***) koji koriste *streaming* uslugu (***usluga***) za gledanje filmova. Za člana se bilježi urednost plaćanja korištenih usluga (***clan*.urednost**) kao razlika u broju plaćenih i neplaćenih usluga, te kojem razredu urednosti član pripada (***clan*.razred**). Usluga *streamanja* filma naplaćuje se 5 kuna po primjerku. Kod korištenja usluge, bilježi se datum i vrijeme kada je član pristupio *streaming* usluzi (***usluga*.datVrUsluga**), datum i vrijeme kada ju je platio (***usluga*.datVrUplata**) i je li plaćena (***usluga*.uplaceno**). Ako je usluga plaćena, vrijednost atributa ***usluga*.uplaceno** je 1. Sve do evidencije uplate, za datum i vrijeme uplate, te atribut *usluga.*uplaceno bilježi se NULL vrijednost.

Zadaci **1-5** se odnose na bazu podataka **streamservis** prikazanu na **slici 1**. U zadacima **1-6** se podrazumijeva korištenje SUBP-a PostgreSQL. Na slici **nisu prikazane** sve n-torke sadržane u relacijama.

film

sifF	***nazF***	***trajanje***	***sifZanr***
10001	Monkey Trouble	02:16:33	100
10012	Troubled Waters	01:40:29	103
10026	The Survivor	02:20:00	101
…	…	…	…

clan

sifC	***username***	***urednost***	***razred***	***…***
10	ahorvat	6	super	…
20	inovak	-1	negativan	…
30	bkolar	2	dobar	…
40	mmatic	0	neutralno	…
…	…	…	…	…

zanr

sifZ	***nazZ***
100	komedija
101	akcija
102	drama
…	…

usluga

sifC	***sifF***	***datVrUsluga***	***datVrUplata***	***uplaceno***
30	10117	29.03.2018 17:17:40	02.04.2018 10:10:29	1
30	10001	03.04.2018 17:07:45	20.04.2018 11:21:21	1
10	10012	15.06.2018 16:10:01	NULL	NULL
20	10001	15.06.2018 08:10:00	15.06.2018 18:12:12	1
…	…	…	…	…

Slika 1.

U zadacima **1 i 2** napisati **po jednu SQL naredbu** kojom će se obaviti sljedeće:

1. **(4 boda)** Za svakog člana ispisati šifru, korisničko ime, zaradu koju je donio davatelju usluge u drugom kvartalu tekuće godine. U zaradu uračunati i iznose koje korisnik duguje, ali još nije uplatio. Ako član nije donio nikakvu zaradu, za zaradu ispisati vrijednost 0. Zapise poredati silazno prema zaradi. Bez obzira na trenutak pokretanja, upit treba vratiti tražene podatke za drugi kvartal tekuće godine.

2. **(5 bodova)** Za svaki žanr ispisati šifru žanra, naziv žanra, naslov i šifru najgledanijeg filma u tom žanru, ukupan broj zahtjeva za *streamanjem* tog filma, te ukupan broj *streamanja* tog filma u posljednjih godinu dana. Žanrove za koje nije bilo *streamanja* filmova, ne treba prikazati u ispisu.

3. **(4 boda)** Svi objekti sa slike 1 su kreirani u shemi *public.* U bazi podataka je obavljena sljedeća SQL naredba:
```
REVOKE ALL ON SCHEMA public FROM PUBLIC;
```
Svakom registriranom korisniku (u donjem slučaju korisniku `cpratt`) je naredbama donjeg oblika dodijeljena ovlast za uspostavu korisničke sjednice i spajanja na bazu podataka:
```
CREATE USER cpratt WITH PASSWORD 'JurassicPark' NOINHERIT;
GRANT CONNECT ON DATABASE streamservis TO cpratt;
```
Administrator sustava treba, svim registriranim korisnicima, omogućiti pregled podataka o njihovim neplaćenim uslugama. Napisati niz SQL naredbi kojima će se:
   a) stvoriti shema *visitorS* i uloga *visitorR* te korisnicima kojima se dodijeli uloga *visitorR* omogućiti pregledavanje podataka o neplaćenim uslugama (naslov filma i datum i vrijeme korištenja usluge).
   Sve potrebne objekte administrator treba definirati u shemi *visitorS.*
   b) korisniku cpratt dodijeliti ovlast za korištenje uloge visitorR
   c) Koje SQL naredbe korisnik *cpratt*, nakon što uspostavi korisničku sjednicu, treba obaviti kako bi mogao koristiti dozvole dodijeljene ulozi *visitorR*?

4. **(7 bodova)** Napisati SQL naredbe kojima će se
   a) osigurati konzistentnost atributa ***usluga.datVrUplata*** i ***usluga.uplaceno.*** Atributi su konzistentni ako oba imaju nepoznatu (NULL) vrijednost ili atribut ***uplaceno*** ima vrijednost 1 za poznatu vrijednost atributa ***datVrUplata***.

   b) kreirati **minimalan** broj objekata koji, pri unosu i izmjeni n-torki relacije ***usluga***, osiguravaju konzistentnost atributa ***clan.urednost*** i ***clan.razred.*** Radi jednostavnosti smatrati da se n-torke u ***usluga*** ne brišu.
   Atribut ***urednost*** smanjuje se (za 1) svakim novim zahtjevom člana za *streaming* uslugom, a povećava (za 1) plaćanjem usluge (ažuriranje ***usluga.datVrUplata*** na neku poznatu vrijednost).
   Također, potrebno je osigurati sljedeću povezanost vrijednosti atribut ***razred*** o vrijednosti atributa ***urednost***:
   razred = 'negativan' za negativnu vrijednost atributa ***urednost***, 'neutralan' za vrijednost 0, 'dobar' za ***urednost*** $\in$[1,4], te 'super' za vrijednosti veće ili jednake 5.
   Dodatno, spriječiti unos zapisa u relaciju *usluga* ako član, do te usluge, ima 3 ili više neplaćenih usluga. Korisniku dojaviti poruku „*Prevelik broj neplaćenih usluga!*“.

5. **(6 bodova)** Davatelj usluge odlučio je kazniti neplatiše smanjenjem njihove urednosti, a njezinim povećanjem nagraditi one koji su korištene usluge promptno plaćali. Napišite SQL naredbe kojima će se svim članovima koji imaju ikakva nepodmirena dugovanja stara više od 6 mjeseci njihova urednost smanjiti za 3, a onima koji su sve korištene usluge platili unutar pola sata od završetka gledanja filma (smatrati da ga je počeo gledati u trenutku korištenja usluge ***usluga.datVrUsluga***), urednost uvećati za 5. Atribut **razred** nije potrebno ažurirati.

6. **(4 boda)** Pretpostavite da je kreirana relacija ***autobus*** koja bilježi identifikator autobusa i broj slobodnih sjedala u istom. Tablicom je prikazan sadržaj relacije ***autobus*** u trenutku započinjanja naredbe {1}. Odredite rezultate izvođenja naredbi {5}, {7}, {9} i {12} ako su u početku započinjanja transakcija A, B i C aktivne transakcije s identifikatorima 10 i 12, a zadnja potvrđena transakcija je ona s identifikatorom 11. Pretpostavite da su prije izvođenja naredbe {12}, transakcije s identifikatorima 10 i 12 uspješno potvrdile svoje promjene u relaciji ***autobus***, te da proces koji „čisti" stare n-torke (*garbage collector*) nije aktivan.

autobus			
xmin	**xmax**	**busId**	**brojSjedala**
1	0	1	5
2	0	2	10
3	10	3	7
10	0	3	5
12	0	4	20

	Transakcija A T_{id} = 13		**Transakcija B T_{id} = 14**		**Transakcija C T_{id} = 15**
{1}	`BEGIN TRANSACTION;`	{2}	`BEGIN TRANSACTION;`	{3}	`BEGIN TRANSACTION;` `SET TRANSACTION ISOLATION`

{4}	`UPDATE autobus`
	`SET brojSjedala =`
	`brojSjedala - 1`
	`WHERE busId = 2;`
		{5}	`SELECT xmin, xmax, *`
			`FROM autobus;`
					`LEVEL SERIALIZABLE;`

{7}	`SELECT xmin, xmax, *`
	`FROM autobus;`
		{8}	`COMMIT TRANSACTION;`	{6}	`UPDATE autobus`
					`SET brojSjedala =`
					`brojSjedala - 1`
					`WHERE busId = 1;`

{10}	`COMMIT TRANSACTION;`			{9}	`SELECT xmin, xmax, *` `FROM autobus;`
{12}	`SELECT xmin, xmax, *` `FROM autobus;`			{11}	`COMMIT TRANSACTION;`

7. **(6 bodova)** U bazi su pohranjene tablice r, s i t koje odgovaraju relacijskim shemama: R(A,B,C,F), S(A,D,G), T (E,H,C) (podcrtani atributi su primarni ključevi danih relacija). Optimizator upita raspolaže sljedećim statističkim svojstvima vezanim uz navedene tablice:

r(A, B, C, F)	s(A, D, G)	t(E, H, C)
N(r) = 60000	N(s) = 30000	N(t) = 30000
V(A, r) = 500	V(D, s) = 800	V(H, t) = 1000
V(B, r) = 700	V(G, s) = 500	V(C, t) = 700
V(F, r) = 200		

Obavlja se operacija $\sigma_{D<'d' \wedge G='g' \wedge C<'c'}(r \rhd\lhd s \rhd\lhd t)$.

Nacrtajte **inicijalno** i **konačno** stablo upita nakon provedene heurističke optimizacije. Redoslijed spajanja relacija odredite temeljem procjene broja n-torki u međurezultatima. Navesti sve izraze prema kojima je obavljena procjena broja n-torki u međurezultatima. U stablu upita naznačite očekivani broj n-torki.

8. **(3 boda) a)** Nad relacijom koja sadrži 5000 n-torki izgrađeno je B+ stablo s minimalnom popunjenošću svih čvorova. Stablo je reda 100 i s ***d*** razina. Izračunajte broj razina ***d*** tog stabla.
b) Koliko n-torki sadrži relacija ako se nad njom izgradi B+ stablo s maksimalnom popunjenošću svih čvorova, reda 100 i s 5 razina?

9. **(3 boda)** Napišite definiciju funkcijske zavisnosti te navedite Armstrongove aksiome.

10. **(8 bodova)** Za potrebe natjecanja studenata u inovativnosti definiraju se zadatci. Studenti, na temelju ponuđenih zadataka formiraju projekte na sljedeći način: projekt objedinjuje različite zadatke, pri čemu će svaki zadatak u okviru projekta rješavati po jedan student; isti zadatak može biti sadržan u više različitih projekata; jedan student u okviru jednog projekta može rješavati više različitih zadataka; studenti mogu sudjelovati u različitim projektima, ali u okviru tih projekata ne mogu rješavati iste zadatke koje rješavaju u drugim projektima.

Projekti se prijavljuju na natjecanja koja su različitih vrsta (fakultetsko, sveučilišno, državno, europsko, svjetsko) i evidentiraju se postignuti rezultati (broj bodova). Za opisani problem treba oblikovati ER model baze podataka u kojem će se evidentirati podaci o studentima: JMBAG, ime, prezime, vrsta studija (EIT, RAC, ICT) i datum upisa na studij; projektima: šifra, naziv i kratica naziva, zadatcima: šifra i tekst zadatka; natjecanjima: šifra, naziv, datum održavanja i razina. Natjecanja su hijerarhijski povezana (što nije direktno određeno vrstom natjecanja). Uz to, za natjecanje treba evidentirati državu u kojoj se odvija (oznaka države, naziv države) i mjesto (poštanski broj, naziv mjesta). Voditi računa o tome da mjesta u različitim državama mogu imati iste poštanske brojeve.

Nacrtajte ER model baze podataka. Navedite sheme entiteta i sheme veza (označite ključeve). Svaki entitet opisati **isključivo vlastitim atributima** (osim slabih entiteta, ukoliko postoje u modelu).
Nužno je da sve sheme zadovoljavaju **3NF**.

Rješenja

1. (4 boda)

```
SELECT clan.sifC, username,
   CASE
       WHEN COUNT(datVrUsluga) = 0 THEN 0
       ELSE COUNT(*)*5
     END AS zarada
FROM clan LEFT JOIN usluga ON clan.sifC = usluga.sifC
                      AND EXTRACT (YEAR FROM datvrusluga) = EXTRACT (YEAR FROM CURRENT_DATE)
                      AND EXTRACT (MONTH FROM datvrusluga) IN (4,5,6)
GROUP BY clan.sifC, username
ORDER BY zarada DESC;
```

2. (5 bodova)

```
SELECT film.sifZanr, nazZ, film.sifF, nazF, COUNT(*) as brStreamova,
       ( SELECT COUNT(*)
         FROM usluga u
         WHERE u.sifF = film.sifF
           AND CURRENT_DATE-'1 year'::INTERVAL <= u.datVrUsluga
        ) as brPosudbiGodDana
  FROM film
  JOIN zanr
    ON film.sifZanr = zanr.sifZ
  NATURAL JOIN usluga
GROUP BY film.sifZanr, nazZ, film.sifF, nazF
HAVING COUNT(*) >= ALL ( SELECT COUNT(*)
                         FROM usluga
                         NATURAL JOIN film film2
                        WHERE film2.sifZanr = film.sifZanr
                       GROUP BY film.sifF );
```

3. (4 boda)

a)

```
CREATE SCHEMA visitorS;
CREATE ROLE visitorR;
GRANT USAGE ON SCHEMA visitorS TO visitorR;                --dozvoliti pristup shemi

CREATE VIEW visitorS.zaduzenja (nazivFilm, datVrUsluga) AS
   SELECT nazF, datVrUsluga
    FROM film NATURAL JOIN usluga
              NATURAL JOIN clan
    WHERE datVrUplata IS NULL
      AND username = SESSION_USER;
-- mora biti SESSION_USER, a ne CURRENT_USER jer se USER zbog NOINHERIT mora proglasiti ulogom
   visitorR

GRANT SELECT ON visitorS.zaduzenja TO regKorisnik;
```

b) `GRANT visitorR to cpratt;`
c) `SET ROLE cpratt;`

4. (7 bodova)

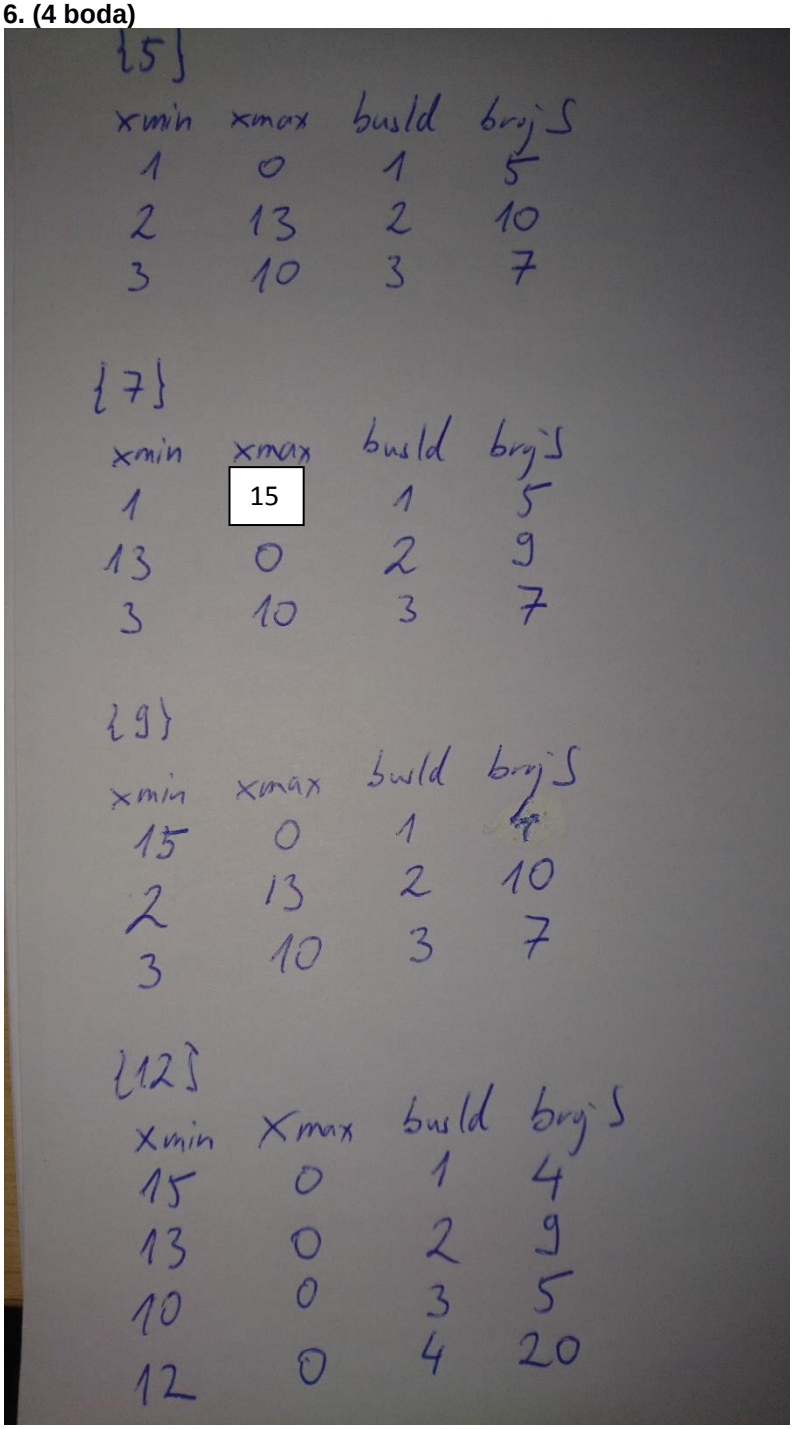
```
ALTER TABLE usluga ADD CONSTRAINT chkUsluga CHECK
                (datVrUplata IS NULL AND uplaceno IS NULL OR
                 datVrUplata IS NOT NULL AND uplaceno = 1);

-- minimalan broj naredbi=2; 1 trigger i 1 function

CREATE OR REPLACE FUNCTION chkInsUpdUplata()
  RETURNS TRIGGER AS $$
DECLARE
  p_dodatak INTEGER DEFAULT 1;
BEGIN
  IF TG_OP = INSERT
  THEN
    IF  (SELECT COUNT(*)
         FROM usluga
```

```
        WHERE sifC = NEW.sifC AND datVrUplata IS NULL) >=3
    THEN RAISE EXCEPTION 'Prevelik broj dugovanja!';
    ELSE
    p_dodatak = -1;   -- uredno novo dugovanje
    END IF;
  END IF;
    -- inace se radi o UPDATE nad usluga (tj. uplati)

    UPDATE clan SET urednost = urednost + p_dodatak
     WHERE sifC = NEW.sifC;

    UPDATE clan SET razred =
               CASE  WHEN urednost<0 THEN 'negativan'
                     WHEN urednost=0 THEN 'neutralan'
                     WHEN urednost BETWEEN 1 AND 4 THEN 'dobar'
                     ELSE 'super'
                   END
      WHERE sifC = NEW.sifC;

  RETURN NEW;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER InsUpdUsluga
   BEFORE INSERT OR UPDATE OF datVrUplata ON usluga
   FOR EACH ROW
     EXECUTE PROCEDURE chkInsUpdUplata();
```

5. (6 bodova)

```
UPDATE clan
SET urednost = urednost - 3
WHERE EXISTS (SELECT * FROM usluga
               WHERE usluga.sifC = clan.sifC
                 AND datVrUplata IS NULL
                 AND datVrUsluga < CURRENT_DATE - '6 MONTHS'::INTERVAL
             );


UPDATE clan
SET urednost = urednost + 5
WHERE ( SELECT COUNT(*)
          FROM usluga
          NATURAL JOIN film
         WHERE usluga.sifC = clan.sifC
           AND datVrUplata BETWEEN datVrUsluga
                               AND datVrUsluga + film.trajanje + '30 minutes'::INTERVAL)
       =
       ( SELECT COUNT(*)
         FROM usluga
         WHERE usluga.sifC = clan.sifC );
```

6. (4 boda)



7. (6 bodova)

- Procjena veličine međurezultata nakon potiskivanja selekcije

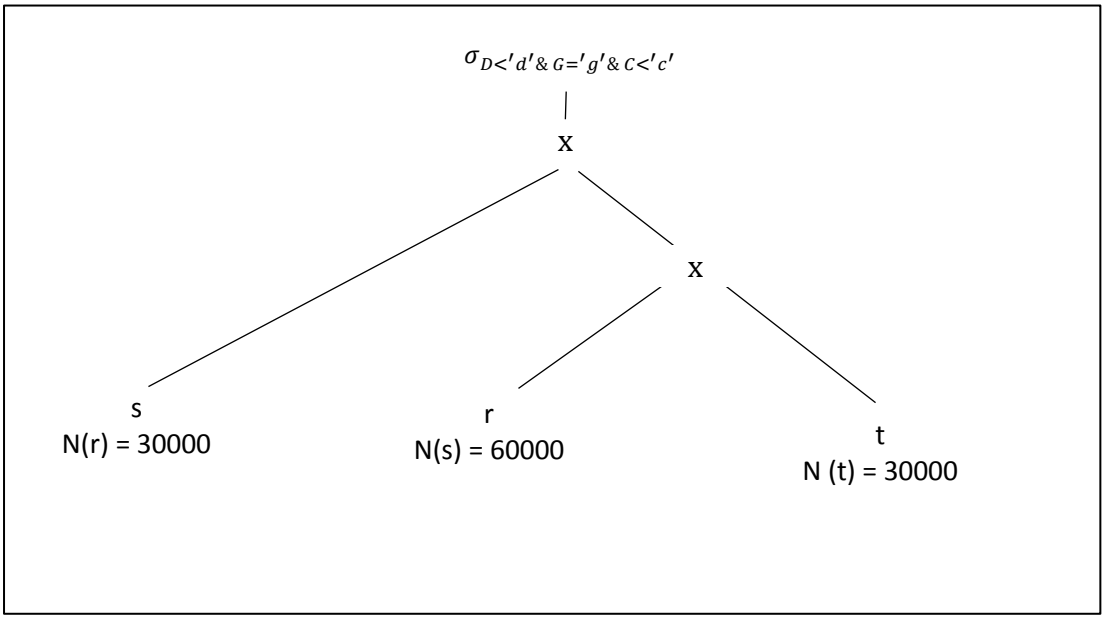
N(s1) = $N(\sigma_{D<'d'\,\&\,G='g'}(s)) = N(\sigma_{D<'d'}(\sigma_{G='g'}(s)))$ = (N(s)/V(G, s))/3 = (30000/500)/3 = 20
N(r1) = $N(\sigma_{C<'c'}(r))$ = N(r)/3 = 60000/3 = 20000
N(t1) = $N(\sigma_{C<'c'}(t))$ = N(t)/3 = 30000/3 = 10000

- Procjena veličine međurezultata nakon spajanja

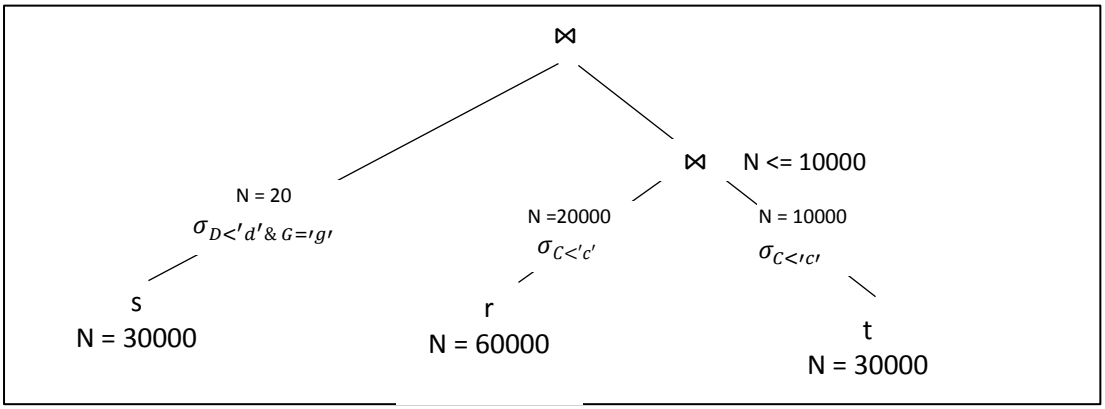
- r1 i s1 - > r1 presjek s1 = ključ u s1
  - N($r1 \rhd\lhd s1$) = N(r1) = 20000
- s1 i t1 - > s1 presjek t1 = prazan skup
  - N($s1 \rhd\lhd t1$) = N(s1)*N(t1) = 20*10000 = 200000
- r1 i t1 - > r1 presjek t1 = ključ u r1
  - N($r1 \rhd\lhd t1$) = N(t1) = 10000

Redoslijed spajanja: $(r \rhd \lhd t) \rhd \lhd s$

Inicijalno stablo upita:



Konačno stablo upita:



8. (3 boda)

a)

$$m = 2 * ceil(n/2)^{d-1}$$
$$5000 = 2 * ceil(100/2)^{d-1}$$
$$\ldots.$$
$$d = log_{50}(5000 * 25)$$
$$d = log_{50}(125000)$$
$$d = log_{50}(50^3)$$
$$d = 3$$

Ovo se može riješiti i „pješke“, bez formule.

- Korijen – najmanje 2 kazaljke
- Interni čvor – ceil(n/2) = 50
- List – ceil((n-1)/2) ≈ ceil(n/2) = 50
- 2*50*50 = 5000 - 3 razine

b)

$$(100^4 * 99) = 9900000000$$

9. (3 boda)

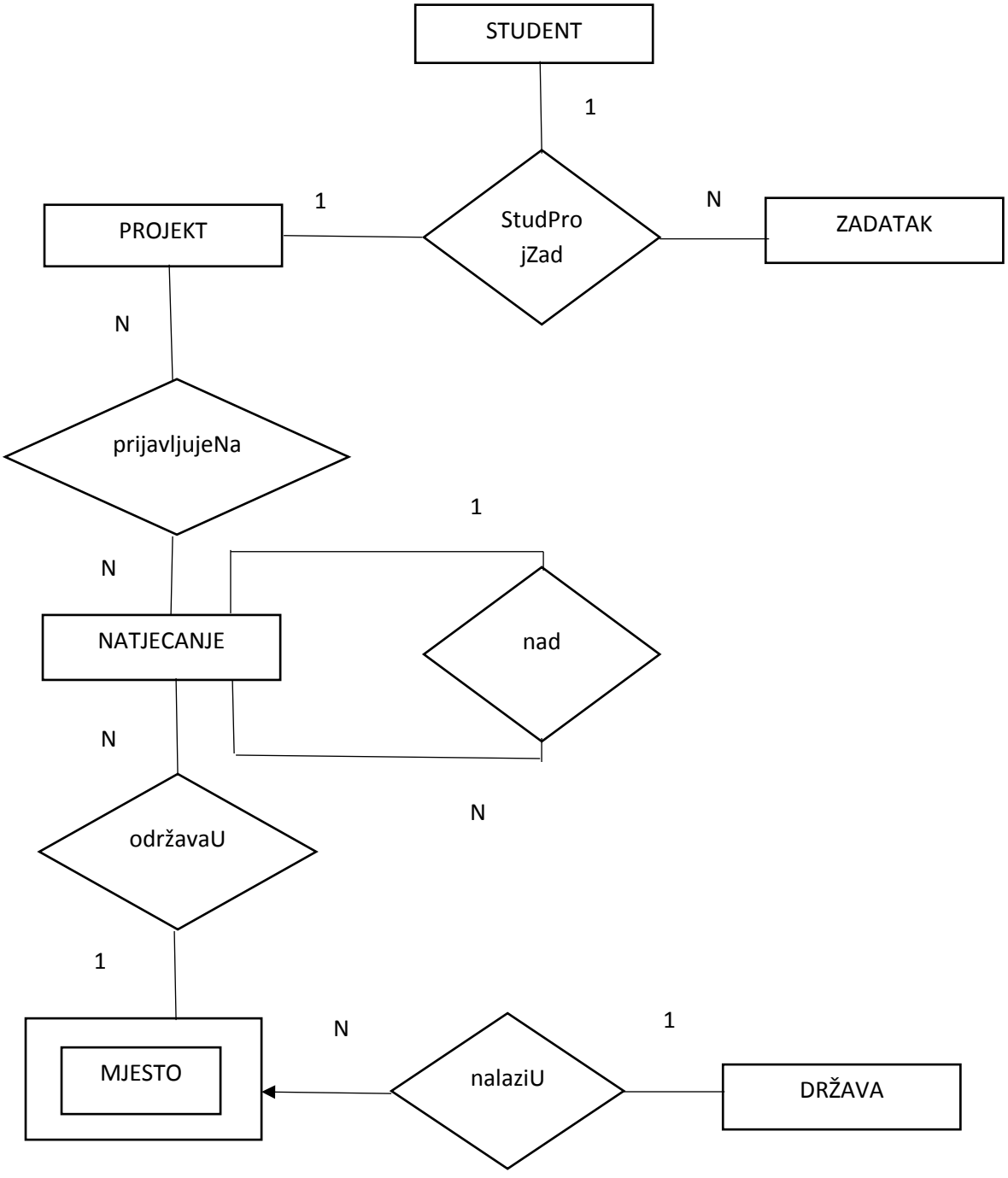
FZ:

Neka je r relacija sa shemom R i neka su X i Y skupovi atributa, X ⊆ R, Y ⊆ R. Funkcijska zavisnost X -> Y vrijedi na shemi R ukoliko u svim dopuštenim stanjima relacije r(R) svaki par n-torki t1 i t2 koje imaju jednake X-vrijednosti, također imaju jednake Y vrijednosti, odnosno: : t1(X) = t2(X) => t1(Y) = t2(Y).

Armstrongovi aksiomi:

- REFLEKSIVNOST
  - Ako je Y ⊆ X, tada vrijedi X -> Y
- UVEĆANJE
  - Ako u shemi R vrijedi X -> Y, tada vrijedi i XZ -> Y
- TRANZITIVNOST
  - Ako u shemi R vrijedi X - > Y i Y -> Z, tada vrijedi i X -> Z

10. (8 bodova)



STUDENT

jmbag

ime

prezime

vrStud

datUpis

PK = jmbag

ZADATAK

sifZad

tekst

PK = sifZad

PROJEKT

sifProjekt

naziv

kratica

PK = sifProjekt

NATJECANJE

sifNatjecanje

naziv

datOdrzavanje

razina

PK = sifNatjecanje

DRŽAVA

oznDrzava

nazDrzava

PK = oznDrzava

MJESTO

pbr

oznDrzava

naziv

PK = { pbr, oznDrzava }

studProjZad

jmbag

sifzad

sifProjekt

PK = { jmbag, sifzad }
PK2 = {sifZad, sifProj}

prijavljujeNa

sifProjekt

sifNatjecanje

PK = { sifProjekt, sifNatjecanje}

održavaU

sifNatjecanje

pbr

oznDrzava

PK = sifNatjecanje

Nad

sifNatjecanje

sifNadNat

PK = sifNatjecanje