BAZE PODATAKA - IZVANREDNI (DEKANSKI) ROK - 15.09.2020.

1. (3 boda)

Uz pomoć jedne SQL naredbe izvedite sljedeće: za svako deklarirano trajanje paketa (jedinstvena vrijednost atributa *paket.trajanje*) koje je korišteno u barem jednoj završenoj borbi ispisati navedeno trajanje i prosječno stvarno trajanje svih borbi s tim deklariranim trajanjem. Ispis poredati po prosječnoj vrijednosti stvarnog trajanja uzlazno.

Primjer redaka ispisa (podaci ne moraju odgovarati stvarnima):

DeklariranoTrajanje	**ProsjTrajanje**
30 minutes	4.4444 seconds 25 minutes
2 hours	15.77771 seconds 12 minutes 1 hour

```
SELECT paket.trajanje as DeklariranoTrajanje,
     AVG(zavrsetakBorba - pocetakTermin)
               AS prosjTrajanje
FROM paket JOIN borba ON
        (paket.trajanje = borba.trajanje AND zavrsetakBorba IS NOT NULL)
GROUP BY paket.trajanje
ORDER BY prosjTrajanje;
```

2. (5 bodova)

Uz pomoć jedne SQL naredbe izvedite sljedeće: za sve timove koji su u 2019. godini igrali preko pet puta više borbi nego u 2020. Ispisati naziv tima i datum/vrijeme osnutka. N-torke u rezultatu poredati po nazivu tima abecedno. Borbe ne moraju biti završene da bi se brojile u rezultatu.

Primjer retka ispisa (podaci ne moraju odgovarati stvarnima):

NazivTim	**DatVrOsnutak**
Charlieatron	07.11.2016 13:55:27

```
SELECT nazivtim, datVrOsnutak
FROM tim
WHERE (
    SELECT COUNT(idborba) FROM  borba
        WHERE ((idTim1 = idTim OR idTim2 = idTim)
             AND EXTRACT(YEAR FROM pocetakTermin) = 2019)
     ) > 5 * (
    SELECT COUNT(idborba) FROM  borba
         WHERE ((idTim1 = idTim OR idTim2 = idTim)
             AND EXTRACT(YEAR FROM pocetakTermin) = 2020)
     )
ORDER BY nazivtim;
```

3. (**6 bodova**)

Sljedeći zadatak riješite **jednim** SQL upitom.

Za svaku arenu u kojoj se odigrala bar jedna borba, ispišite njezin naziv, naziv tima koji je u toj areni ostvario najveći broj pobjeda i time postavio rekord za broj pobjeda u toj areni. Također ispišite ime i prezime osnivača tog tima te broj navedenih pobjeda. Ukoliko je više timova u nekoj areni ostvarilo jednak maksimalan broj pobjeda, ispišite sve timove.

N-torke u rezultatu poredajte po nazivu arene i nazivu tima uzlazno.

Primjer ispisa:

nazivArena	nazivTim	ime	prezime	brojPobjeda
Apple Corner Arena	A Dozendust	Bradley	Snow	6
Apple Corner Arena	School of the A Dozen Lions	Quinton	Mckee	6
Blond Arena	Sausage Army	Rudi	Fischer	5
Rose Arena	Sausage Army	Rudi	Fischer	3
…	…	…	…	…

Rješenje:

```
SELECT arena.nazivArena, tim.nazivTim, igrac.ime, igrac.prezime, COUNT (*)
      AS brojPobjeda
FROM borba  NATURAL JOIN igra
            NATURAL join arena
            JOIN tim on tim.idTim = borba.idTimpobjeda
            JOIN igrac on tim.idosnivac = igrac.idIgrac
WHERE borba.zavrsetakBorba IS NOT NULL
GROUP BY arena.idArena, arena.nazivArena, borba.idTimPobjeda, tim.nazivTim,
      igrac.ime, igrac.prezime
HAVING COUNT (*) >= ALL (SELECT COUNT (*)
                         FROM borba as borba2 NATURAL JOIN igra
                                              NATURAL JOIN arena as arena2
                         WHERE arena2.idArena = arena.idArena and borba2.zavrsetakBorba
                         IS NOT NULL
                         GROUP BY arena2.idArena, borba2.idTimpobjeda
                         )
ORDER BY nazivArena, tim.nazivTim
```

4. (4 boda)

Zbog fiksnih troškova održavanja određenih arena, postavljena je minimalna cijena paketa za igre u tim arenama. Napisati niz SQL naredbi za kreiranje svih potrebnih objekata kojima će se pri ***unosu*** n-torke ili promjeni vrijednosti atributa ***cijena*** u relaciji ***paket*** spriječiti definiranje cijene paketa niže od 175 HRK za bilo koju igru u areni s nazivom '*Meadow View Arena*'.

Pri narušavanju opisanog ograničenja korisniku javiti sljedeću grešku: ***'Pogreška: Cijena paketa za igru (<igra>) je preniska s obzirom da se može koristiti Meadow View Arena'***.

Umjesto <igra> potrebno je ispisati šifru igre za koju se definira paket.

U svim ostalim slučajevima n-torka mora biti uspješno upisana ili promijenjena u relaciji ***paket***.

Rješenje:

```
CREATE FUNCTION chkCijenaPaket() RETURNS TRIGGER AS
$$
   BEGIN
      IF((SELECT nazivArena
            FROM igra
            NATURAL JOIN arena
           WHERE idIgra = NEW.idIgra) = 'Meadow View Arena' AND NEW.cijena<175)
THEN
            RAISE EXCEPTION
                  'Pogreška: Cijena paketa za igru (%) je preniska s obzirom
da se može koristiti Meadow View Arena',
                   NEW.idigra;
      END IF;
      RETURN NEW;
   END;
$$ language plpgsql;

CREATE TRIGGER trigInsUpdPaket
BEFORE INSERT OR UPDATE OF cijena ON paket
FOR EACH ROW EXECUTE FUNCTION chkCijenaPaket();
```

5. (6 bodova)

Administrator baze podataka je nakon kreiranja baze podataka i relacija u shemi *public* obavio sljedeće SQL naredbe:

```
REVOKE CONNECT ON DATABASE tagfight FROM public;
REVOKE ALL ON SCHEMA public FROM public;
```

a) Napisati SQL naredbe kojima će administrator sustava korisniku *rbrown*, osnivaču nekih timova, dodijeliti ovlasti za:
- uspostavu korisničke sjednice (lozinku postaviti na *rbrownPwd*),
- spajanje na bazu *tagfight*,
- pristup objektima sadržanima u shemi *public* bez mogućnosti kreiranja novih objekata,
- pregled svih podataka u tablici *igra* uz mogućnost dodjeljivanja tih istih ovlasti ostalim korisnicima
- izmjenu naziva tima u tablici *tim*
- pregled naziva igara i maksimalnog stvarnog trajanja tih igara ako se uzmu u obzir sve završene borbe

Kreirajte sve objekte potrebne za dodjelu opisanih ovlasti korisniku *rbrown.*

```
CREATE USER rbrown WITH PASSWORD 'rbrownPwd';
GRANT CONNECT ON DATABASE tagfight TO rbrown;
GRANT USAGE ON SCHEMA public TO rbrown;
GRANT SELECT ON public.igra TO rbrown WITH GRANT OPTION;
GRANT UPDATE ON public.tim(nazivTim) TO rbrown;

CREATE VIEW maxTrajanjeBorbi AS
 SELECT igra.nazivIgra, MAX(zavrsetakBorba-pocetakTermin) AS maxTrajanje
   FROM borba NATURAL JOIN igra
 WHERE zavrsetakBorba IS NOT NULL
GROUP by igra.nazivIgra;
GRANT SELECT ON public.maxTrajanjeBorbi TO rbrown;
```

b) Korisniku *rbrown* dodijeliti ovlasti za stvaranje vlastitih shema.

```
GRANT CREATE ON DATABASE tagfight to rbrown;
```

c) Neka je korisnik *rbrown* dodijelio ovlast pregleda podataka tablice *igra* korisnicima *jmorris* i *dmendez*. Napišite **jednu** naredbu kojom će administrator ukinuti dozvole pregleda podataka tablice *tim* korisnicima *rbrown*, *jmorris* i *dmendez*.

```
REVOKE SELECT ON igra FROM rbrown CASCADE;
```

6. (3 boda)

Relacija **igrac** (idigrac, ime, prezime, datrodjenja) sadrži n-torke sa sljedećim vrijednostima atributa idigrac: 1, 4, 13, 42, 85, 99, 137, 169, 227

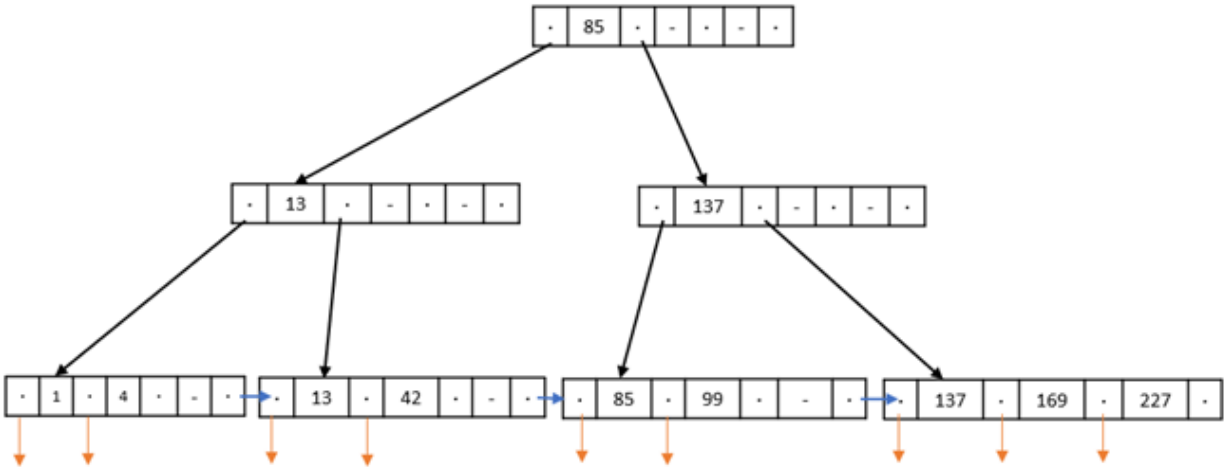
Nacrtajte B^+ stablo reda 4 za atribut idIgrac tako da popunjenost stabla bude minimalna.

Napomena:

- . Rješenje nacrtati na obrascu s vašim imenom i prezimenom.
- Izračunati broj kazaljki za svaku vrstu čvora napišite u prostor za slobodan unos teksta.

MIN: K = 2, Č = $\lceil n/2 \rceil$ = 2, L = $\lceil (n-1)/2 \rceil$ = 2

MAX: K = 4, Č = 4, L = n-1 = 3



7. (5 bodova)

Zadana je relacijska shema VrtniCentar u kojem se prate biljke, njihovi proizvođači te cijene koje vrijede u određenim periodima. Relacijska shema VrtniCentar ima sljedeće atribute:

sifBiljka	šifra biljke - jedinstveni identifikator biljke
nazBiljka	naziv biljke
datCijenaOd	datum kad je postavljena cijena
datCijenaDo	datum do kojeg je vrijedila cijena (null za aktualnu cijenu)
iznCijena	cijena za biljku koja vrijedi od datCijenaOd do datCijenaDo
sifProizvodjac	šifra proizvođača
nazProizvodjac	naziv proizvođača
oznDrzava	jedinstvena oznaka države proizvođača
nazDrzava	naziv države proizvođača

Odrediti ključ relacijske sheme VrtniCentar tako da ona bude u 1NF, a zatim postupno normalizirati relacijsku shemu na 2NF i 3NF.

U svakom koraku navesti funkcijske zavisnosti na temelju kojih se obavlja dekompozicija te označiti ključeve novonastalih relacijskih shema.

Navesti relacijske sheme od kojih se sastoji konačna shema baze podataka VRTNI_CENTAR.

Rezultate upišite u prostor za odgovore ispod teksta zadatka.

Rješenje:
1NF
VC1 = sifBiljka, datCijenaOd, nazBiljka, iznCijena, datCijenaDo, sifProizvodjac, nazProizvodjac, oznDrzava, nazDrzava
2NF
VC2 = sifBiljka, nazBiljka, sifProizvodjac, nazProizvodjac, oznDrzava, nazDrzava
CIJENA = sifBiljka, datCijena, iznCijena, datCijenaDo
3NF
BILJKA = sifBiljka, nazBiljka, sifProizvodjac
VC4 = sifProizvodjac, nazProizvodjac, oznDrzava, nazDrzava
PROIZVODJAC = sifProizvodjac, nazProizvodjac, oznDrzava
DRZAVA = oznDrzava, nazDrzava

VRTNI_CENTAR = BILJKA, CIJENA, PROIZVODJAC, DRZAVA

8. (5 bodova)

Navedite po jedan primjer, koji koristi isključivo podatke iz tablice ***igra,*** kojima ćete pokazati da:

a) **pisanje ne blokira čitanje** kada Postgres za upravljanje istodobnim pristupom koristi isključivo Multiverzijski protokol upravljanja istodobnim pristupom – MVCC

b) **pisanje blokira čitanje** kada Postgres koristi eksplicitno postavljanje ključeva (elemente protokola zasnovanog na zaključavanju)

Svaki od dva primjera treba sadržavati minimalan broj transakcija koje se paralelno odvijaju i minimalan broj SQL naredbi potrebnih za demonstraciju ponašanja. Koristite razine izolacije koje smatrate prikladnima. Naznačite:

- početni sadržaj n-torke/n-torki koje u primjerima koristite
- redoslijed izvršavanja naredbi u transakcijama (globalni redoslijed za sve transakcije koje u primjeru koristite)
- objasnite ulogu svake SQL naredbe koju u primjerima koristite, obavezno navedite za svaku naredbu traži li ona i dodjeljuje li joj se ključ i koje vrste te koliko taj ključ traje
- objasnite zbog čega u primjeru pod a) pisanje ne blokira čitanje, odnosno zbog čega u primjeru pod b) pisanje blokira čitanje.

NAPOMENA: svoje odgovore unesite u <u>prostor za slobodni unos teksta</u> ispod teksta zadatka.

Rješenje

Početno stanje n-torke koja se koristi u primjeru pod a) i pod b)

idIgra	nazivIgra	idArena
10	Crystalrain	2

Obje transakcije koriste defaultnu razinu izolacije – READ COMMITTED, premda u ovim primjerima razina izolacije nije važna jer bi ponašanje bilo jednako i kod bilo koje druge razine.

a)

	T1		T2
{1}	`BEGIN TRANSACTION;`	{2}	`BEGIN TRANSACTION;`
{3}	`UPDATE igra SET idArena = 3` `WHERE idIgra = 10;`		

		{4}	`SELECT *`
			`FROM igra`
			`WHERE idIgra = 10;`

{5}	`COMMIT TRANSACTION;`	{6}	`COMMIT TRANSACTION;`

{3} demonstrira pisanje n-torke. Postavlja se ključ za pisanje (ekskluzivni ključ) nad n-torkom koji traje do kraja transakcije.

{4} demonstrira čitanja iste n-torke koju je T1 netom prije pročitala. Ne traži se niti se postavlja ikakav ključ

Obje naredbe uspješno prolaze – T2 vidi inicijalnu verziju n-torke jer bi se inače radilo o prljavom čitanju koje Postgres ne dozvoljava niti kod jedne razine izolacije.

b)

	T1		T2
{1}	`BEGIN TRANSACTION;`	{2}	`BEGIN TRANSACTION;`
{3}	`UPDATE igra SET idArena = 3` `WHERE idIgra = 10;`		

		{4}	`SELECT *`
			`FROM igra`
			`WHERE idIgra = 10`

{5}	`COMMIT TRANSACTION;`	{6}	`FOR SHARE;`

			COMMIT TRANSACTION;

{3} postavlja ključ za **pisanje** (write/exclusive lock) nad n-torkom sa šifrom 10. Ključ traje do kraja transakcije.
{4} pokušava postaviti ključ za **čitanje** nad n-torkom sa šifrom 10, ali ne uspijeva jer T1 već ima ključ za pisanje (nekompatibilan s ključem za čitanje) nad tom istom n-torkom. Čitanje T2 je blokirano pisanjem T1.
Zbog nekompatibilnosti ključeva – ključ za čitanje nije kompatibilan s ključem za pisanje, T2 ne uspijeva pročitati n-torku koju je T1 promijenila. Odnosno, postavljanjem ključeva koji traju do kraja transakcije postiže se da pisanje blokira čitanje.

Zajedničko u oba primjera je da se kod izmjene postavlja ključ za pisanje na objekt koji se mijenja i ključ traje do kraja transakcije.
Razlika između čistog MVCC protokola i implementacije koja koristi eksplicitno zaključavanje kod čitanja je što MVCC kod čitanja ne zaključava objekt koji pokušava pročitati dok u drugom slučaju se taj ključ eksplicitno traži i postavlja ako je moguće (u primjeru pod b) nije moguće). Zbog toga u trenutku kad se pokuša obaviti čitanje u primjeru pod a) je to moguće jer čitanje ne traži nikakav ključ nad tim objektom, dok u primjeru pod b) to nije moguće jer postoji ključ za pisanje nad tim objektom.

9. (4 boda)

U bazi podataka koja se koristi u ovoj provjeri definirana su ograničenja primarnih i stranih ključeva.
Uz indekse koje PostgreSQL automatski kreira, dodatno je nad relacijom kreiran indeks:

```
CREATE index igra_naziv ON igra (nazivIgra)
```

Optimizator upita raspolaže sljedećim statističkim podacima:

N(igra) = 120 V(nazivIgra, igra) = 60	N(arena) = 20 V(nazivArena, arena) = 20	N(paket) = 300 V(cijena, paket) = 250 V(trajanje, paket) = 4 V(idIgra, paket) = 90

Napomena: navedeni statistički podatci ne odgovaraju stvarnim statističkim podatcima u bazi na Edgaru.

Izvodi se upit:

```
SELECT *
    FROM paket, igra, arena
     WHERE igra.idArena = arena.idArena
    AND igra.idIgra = paket.idIgra
    AND nazivIgra = 'Dreampoint'
    AND cijena <  200
    AND trajanje = '00:30:00' :: interval
```

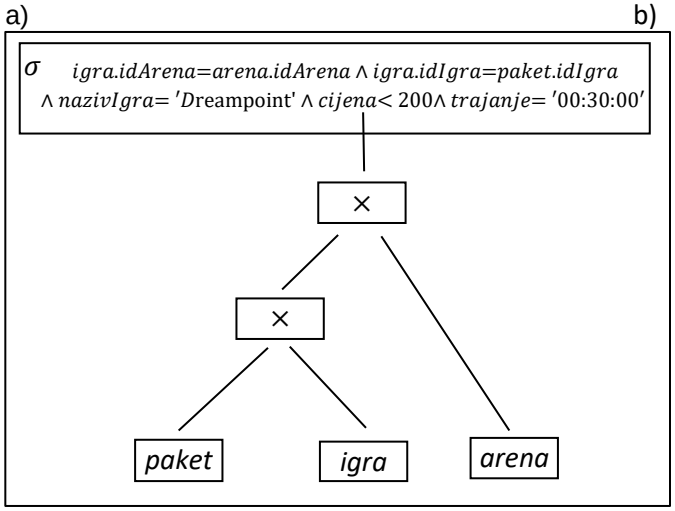
a) Nacrtajte (na papir) stablo upita za **početni plan** izvođenja upita pri čemu je redoslijed spajanja tablica određen redoslijedom kojim su tablice navedene u FROM dijelu SELECT naredbe.
b) Nacrtajte (na papir) stablo upita **nakon** provedene heurističke optimizacije. Dovoljno je nacrtati samo konačno stablo upita. Redoslijed spajanja relacija odrediti temeljem procjene broja n-torki u rezultatima spajanja. Navesti sve izraze prema kojima je obavljena procjena broja n-torki u međurezultatima i u konačnom rezultatu. U stablu upita naznačiti očekivani broj n-torki u svim međurezultatima te korištene metode pristupa

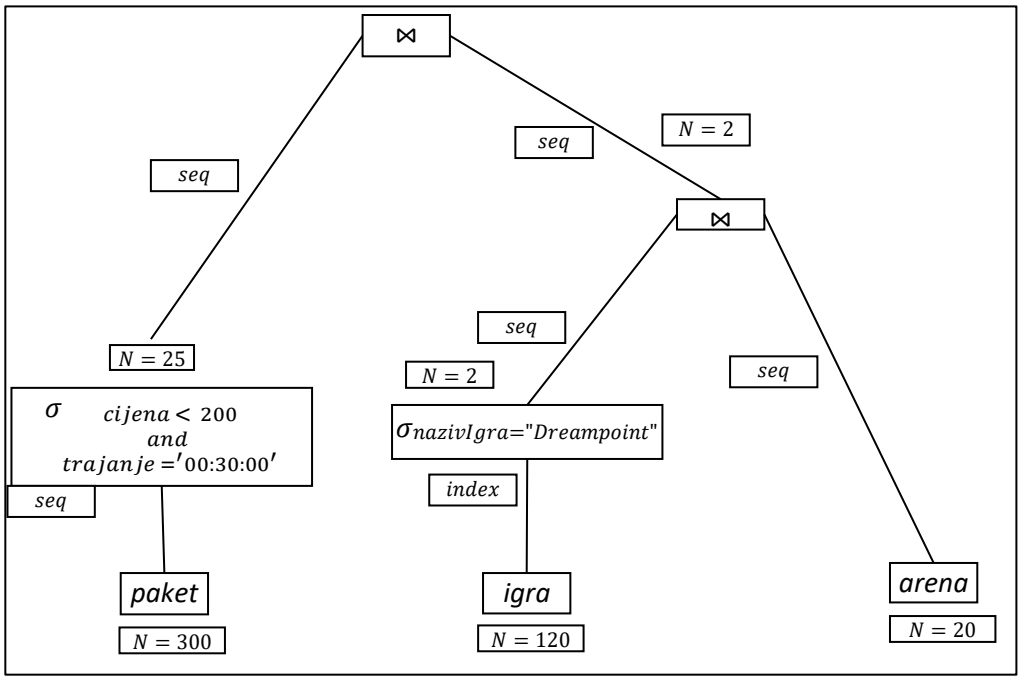
Napomena:

- Postupak i izračun za procjenu broja n-torki **unesite u prostor za slobodni unos teksta ispod teksta zadatka.**

Rj.

a) b)





potiskivanje selekcije `nazivIgra = 'Dreampoint'` na **igra**:

N(igra') = N(igra)/ V(nazivIgra, igra) = 120/60 = 2

potiskivanje selekcije `cijena < 200 and trajanje = '00:30:00'` na **paket**:

N(paket') = N(paket)/(3* V(trajanje, paket)) = 300/(3*4) = 25

Međurezultati spajanja:

igra' + arena (fk u igra) => N(igra') = 2 -> prvo se spajaju igra i arena

igra' + paket' (fk u paket) => N(paket') = 25

arena + paket' (Kartezijev) => 20 * 25 = 500

10. (8 bodova)
Oblikovati ER model baze podataka za planiranje i upravljanje hranjenjem životinja u zoološkom vrtu. Evidentiraju se podatci o životinjama: šifra, ime, porodica, vrsta, datum rođenja, datum dolaska u zoo. Za porodicu se evidentira šifra i naziv (sisavci, ptice, ribe, vodozemci, gmazovi, beskralježnjaci). Za sisavce se dodatno evidentira težina, za ptice raspon krila, dok za životinje koje pripadaju ostalim razredima nema dodatnih podataka. Za vrste (lavovi, vukovi, ...) evidentira se šifra i naziv te porodica kojoj pripada. Svaka životinja smještena je u jednoj nastambi. U nastambi može biti više životinja, iste ili različitih vrsta. Za nastambu se evidentira redni broj i opis lokacije unutar zoološkog vrta.

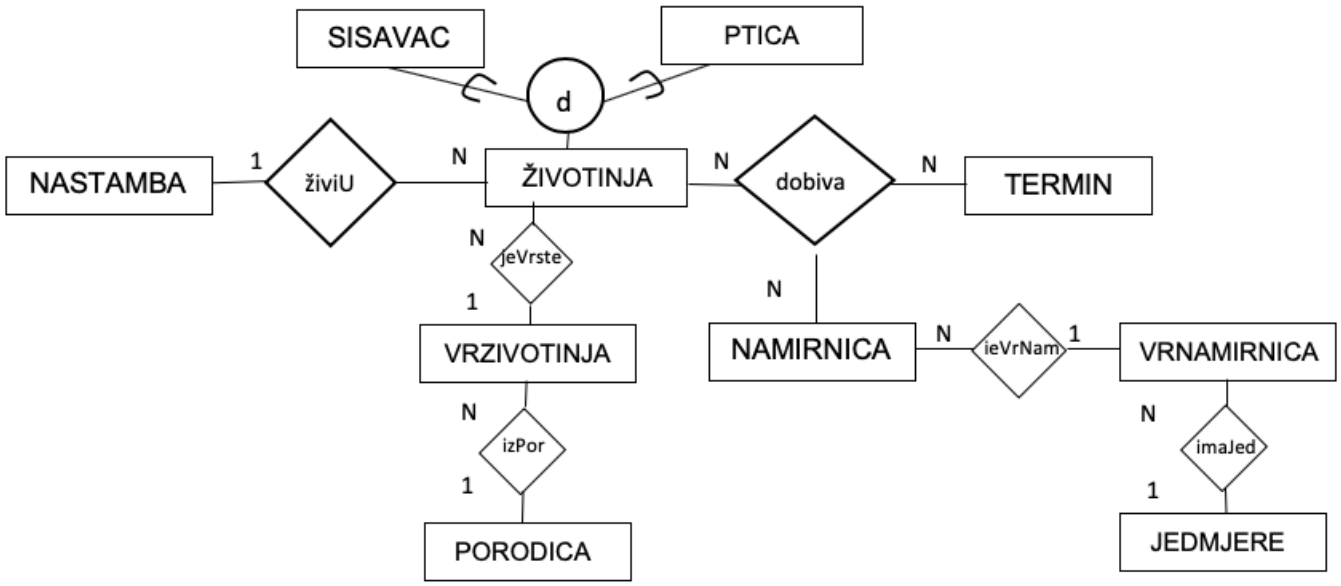
Hranjenje životinja se organizira nekoliko puta dnevno, u unaprijed određenim vremenskim terminima. Jedna životinja u jednom terminu dobiva više različitih namirnica, a istu namirnicu može dobiti u različitim terminima. Jednu namirnicu u jednom terminu dobiva više životinja. Za svaki termin hranjenja određene životinje evidentira se količina ponuđene namirnice, izražena u jedinici mjere za tu vrstu namirnice.
Termin hranjenja ima svoj identifikator (serijski broj), datum i vrijeme početka hranjenja.
Namirnica je opisana šifrom, nazivom, šifrom vrste namirnice. Vrsta namirnice (povrće, voće, meso, mlijeko, žitarice, ...) opisana je šifrom, nazivom te oznakom jedinice mjere. Jedinica mjere je definirana jedinstvenom oznakom i opisom.

Potrebno je nacrtati ER dijagram (bez atributa) i označiti spojnosti entiteta u vezama. Sve entitete, osim slabih, opišite isključivo vlastitim atributima. Definirati relacijske sheme svih entiteta i svih veza, označiti ključeve i provjeriti zadovoljavaju li 3NF.

a) Nacrtajte ER model baze podataka **na dobivenom papiru**

b) U prostor za slobodan unos teksta ispod teksta zadatka navedite sheme entiteta i sheme veza.

Rješenje:



ZIVOTINJA = **sifZivotinja**, imeZivotinja, datRod, datDolazak	VRZIVOTINJA = **sifVrZivotinja**, nazVrZivotinja
NASTAMBA = **rbrNastamba**, opisLokacije	PORODICA = **sifPorodica**, nazPorodica
SISAVAC = **sifZivotinja**, tezinaZivotinja	dobiva = **idTermin**, **sifZivotinja, sifNamirnica,** kolicina
PTICA = **sifZivotinja**, rasponKrila	
TERMIN = **idTermin**, datVrijPocHranjenja	ziviU = **sifZivotinja**, sifNastamba
NAMIRNICA = **sifNamirnica**, nazNAmirnica	jeVrste = **sifZivotinja**, sifVrZivotinja
VRNAMIRNICA = **sifVrNamirnica**, nazVrNamirnica	izPor = **sifVrZivotinja,** sifPorodica
JEDMJERE = **oznJedMjere**, opisJedMjere	imaJed = **sifVrNamirnica**, oznJedMjere