Ljetni rok iz Baza podataka
(50 bodova)

8. srpnja 2020.

U bazi podataka se evidentiraju podatci o borbama/dvobojima timova igrača (***igrac***, ***tim***, ***igracTim***) u nekoj od tematskih igara (***igra***)**.** Igre se, zbog različitih zahtjeva za opremom, igraju u jednoj od arena (***arena***). Svaka igra može biti sadržana u više paketa (***paket***), a ovisno o predviđenom trajanju igre (***paket.trajanje***, tip INTERVAL), definirane su cijene paketa po igraču (***paket.cijena***). Osnivač tima (***tim.idOsnivac***) je uvijek član tima (***igracTim***) kojeg je osnovao, a igrač može biti i član i osnivač više timova. Relacija ***borba*** sadrži podatke o odigranim, planiranim i borbama koje su u tijeku: početak rezerviranog termina za borbu, vrijeme završetka borbe (NULL ako borba nije završila ili kraj nije poznat); igra i predviđeno trajanje (identificiraju paket), te timovi koji sudjeluju i pobjednik ako je poznat. Pobjednik je, naravno, jedan od timova koji su sudjelovali u dvoboju ili nepoznat (NULL) ako borba nije završila ili je iz nekog razloga neevidentiran.

U bazi podataka postoje mehanizmi koji osiguravaju konzistentnost podataka u tablici ***borba*** (igrač ne igra u isto vrijeme u dva tima, istovremeno u istoj areni se ne odvija više borbi i slično) i o tome ne treba brinuti.

Primarni ključevi relacija su podcrtani.
Slika 1 je ilustrativan prikaz mogućih n-torki u bazi podataka. Prikazane n-torke ne moraju odgovarati u potpunosti podatcima u bazi.

igrac

idIgrac	ime	prezime	datRodjenja
1	Christian	Reeves	30.03.1977
2	Lleyton	Pugh	28.06.1998
3	Tahmina	Preece	14.12.1965
…	…	…	…

tim

idTim	nazivTim	datVrOsnutak	idOsnivac
1	Titanmaya	3.12.2018 15:15:34	1
2	James Names	8.2.2018 12:23:00	3
3	Vileman	1.4.2017 16:00:00	3
4	A Dozendusti	2.4.2017 10:02:03	2
...	...	…	...

igracTim

idIgrac	idTim
1	1
2	1
3	2
4	2
...	…

arena

idArena	nazivArena
1	Primrose View Arena
2	Lakeside Arena
3	Tony's Arena
…	…

igra

idIgra	nazivIgra	idArena
1	Domination	1
2	Deathmatch	1
3	Battle Royale	2
4	Chernobyl Stalker	3
5	Zombie Biohazard	2
...	...	...

paket

idIgra	trajanje	cijena
1	00:30:00	100
1	01:00:00	150
1	02:00:00	250
2	01:00:00	170
2	02:00:00	230
3	02:00:00	200
4	01:00:00	160
5	01:30:00	180
…	…	…

borba

idBorba	pocetakTermin	zavrsetakBorba	idIgra	trajanje	idTim1	idTim2	idTimPobjeda
1	12.04.2019 14:00:00	12.04.2019 15:50:00	3	02:00:00	1	2	1
4	05.03.2020 18:00:00	05.03.2020 18:25:00	1	00:30:00	1	3	3
5	10.04.2020 14:00:00	10.04.2020 16:20:00	3	02:30:00	3	4	3
6	05.04.2020 12:00:00	05.04.2020 12:30:00	1	00:30:00	1	3	3
7	07.04.2020 12:00:00	07.04.2020 12:45:00	4	01:00:00	1	3	1
16	27.04.2020 17:00:00	NULL	3	02:00:00	2	4	NULL
…	…	…	…	…	…	…	…

Slika 1

1. **(4 boda)** Uz pomoć jedne SQL naredbe izvedite sljedeće: za sve arene u kojima se odigrala barem jedna borba čije je stvarno trajanje bilo za barem 25% kraće od predviđenog trajanja ispisati naziv arene i ukupan broj takvih borbi. Ispis poredati silazno po broju takvih borbi, a potom abecedno po nazivu arene.

   Primjer retka ispisa (podaci ne moraju odgovarati stvarnima):

nazivArena	**BrojKratkihBorbi**
Primrose View Arena	68

```
SELECT nazivArena, count(*) as brojKratkihBorbi
FROM arena NATURAL JOIN igra NATURAL JOIN borba
WHERE zavrsetakBorba - pocetakTermin <= 0.75 * borba.trajanje
GROUP BY arena.idarena, nazivArena
ORDER BY brojKratkihBorbi DESC, nazivArena ASC;
```

2. **(4 boda**) Uz pomoć jedne SQL naredbe izvedite sljedeće: za sve igrače koji nikada nisu igrali borbu u areni naziva "Giraffe's Arena" ispisati prezime, ime i nazive timova kojem pripadaju. Ispis poredati po prezimenu a potom po nazivu tima.

   Primjer redaka ispisa (podaci ne moraju odgovarati stvarnima):

Prezime	**Ime**	**NazivTim**
Perich	Pero	Hawthornes
Perich	Pero	Human Beings
Reeves	Christian	Wingers

```
SELECT prezime, ime, nazivtim from igrac NATURAL JOIN igractim
      NATURAL JOIN tim
WHERE igrac.idigrac NOT IN
(SELECT idIgrac FROM igrac NATURAL JOIN igracTim
     NATURAL JOIN tim
     JOIN borba ON (idtim1 = idTim OR idtim2 = idtim)
     NATURAL JOIN igra
     NATURAL JOIN arena
     WHERE nazivArena LIKE 'Giraffe''s Arena')
ORDER BY prezime, nazivTim;
```

3. **(6 bodova)** Sljedeći zadatak riješite **jednim** SQL upitom.
   Za svakog igrača ispišite njegov identifikator, ime, prezime, naziv onog tima čiji je član a za čije je borbe ukupno potrošio najviše novaca te taj potrošeni iznos. Uzmite u obzir samo odigrane borbe.
   Ako je igrač s više timova potrošio jednak iznos, ispišite nazive svih timova. Za igrače koji se nikada nisu borili (nisu član niti jednog tima ili nemaju niti jednu završenu borbu) za naziv tima i iznos potrošnje ispišite NULL vrijednost.

   Ispis poredajte abecednim redom uzlazno po prezimenu igrača, a zatim po identifikatoru igrača uzlazno.

   Primjer ispisa:

idIgrac	ime	prezime	nazivTim	maxPotrosnja
398	Leonardo	Abbott	Meaty Meaty	21650.00
472	Waleed	Acevedo	One Girl, A Dozen Shoes	21750.00
43	Lana	Adamson	null	null
402	Tracy	Adamson	The Map Sisters	23275.00
…	…	…	…	…

Rješenje:

```
SELECT igrac.IdIgrac, igrac.ime, igrac.prezime, tim.nazivTim,
 SUM(paket.cijena) as maxPotrošnja
FROM igrac  NATURAL LEFT JOIN igracTim
      NATURAL LEFT JOIN tim
LEFT JOIN borba ON (igracTim.idTim = borba.idTim1 OR
  igracTim.idTim = borba.idTim2) AND
  borba.zavrsetakborba IS NOT NULL
NATURAL LEFT JOIN paket
GROUP BY igrac.IdIgrac, igrac.ime, igrac.prezime, tim.nazivTim
HAVING SUM (paket.cijena) >= ALL (SELECT SUM (paket.cijena)
                                  FROM igracTim iT2
                                  JOIN borba borba2 ON iT2.idTim =
                                  borba2.idTim1 OR it2.idTim = borba2.idTim2
                                  NATURAL JOIN paket
                                  WHERE iT2.idIgrac = igrac.idIgrac AND
                                  zavrsetakBorba IS NOT NULL
                                  GROUP BY iT2.idTim )
ORDER BY igrac.prezime, igrac.IdIgrac
```

4. **(6 bodova)** Pretpostavite da se u relaciju ***borba*** za rezervaciju unose i zapisi za zakazane borbe koje će se tek dogoditi, pri čemu se za atribute ***zavrsetakBorba*** i ***idTimPobjeda*** unose NULL vrijednosti. Ako je unesena odigrana borba, navedeni atributi nemaju NULL vrijednosti, a vrijednost atributa ***idTimPobjeda*** mora imati jednu od vrijednosti (***idTim1***, ***idTim2***).
   a) Jednom SQL naredbom osigurajte konzistenciju vrijednosti atributa ***zavrsetakBorba*** i ***idTimPobjeda*** u skladu s prethodno opisanim pravilima.
   b) Napisati niz SQL naredbi za kreiranje svih potrebnih objekata kojima će se pri ***unosu*** n-torke u relaciju ***borba*** spriječiti unos borbe čiji se termin preklapa s ***predviđenim vremenom*** za neku drugu borbu koja bi se trebala održati u istoj areni. **Predviđeno vrijeme** za neku borbu je od trenutka početka termina pa do isteka vremena naznačenog atributom ***trajanje***. Pretpostavite da je u bazi već kreirana funkcija `provjeriPreklapanje` (vidjeti napomenu niže) koja kao parametre prima datum i vrijeme početka te trajanje prvog, odnosno drugog termina.

   Pri narušavanju opisanog ograničenja korisniku javiti sljedeću grešku: ***'Pogreška: željeni termin se preklapa s postojećim terminom u istoj areni.***'.

   U svim ostalim slučajevima n-torka mora biti uspješno upisana u relaciju ***borba***.

   Napomena: puna definicija funkcije `provjeriPreklapanje` dana je u nastavku.

```
CREATE or replace FUNCTION provjeriPreklapanje(datVrPoc1 TIMESTAMP, trajanje1 INTERVAL,
          datVrPoc2 TIMESTAMP, trajanje2 INTERVAL )
RETURNS SMALLINT AS
$$
DECLARE
datVrKraj1 TIMESTAMP;
datVrKraj2 TIMESTAMP;
preklapanje SMALLINT;
BEGIN
datVrKraj1 = datVrPoc1 + trajanje1;
datVrKraj2 = datVrPoc2 + trajanje2;
IF (datVrPoc1 < datVrKraj2 AND datVrPoc2 < datVrKraj1)
       THEN preklapanje = 1;
       ELSE preklapanje = 0;
END IF;
RETURN preklapanje;
END;
$$ language plpgsql;
```

```
a)
ALTER TABLE borba
ADD CONSTRAINT chkGotovaBorbaPobjednik
CHECK ( (zavrsetakBorba IS NULL AND idTimPobjeda IS NULL)
        OR
         (  zavrsetakBorba IS NOT NULL
           AND
           (idTimPobjeda = idTim1  OR idTimPobjeda = idTim2)
          )
       );
```

```
b)
CREATE or replace FUNCTION preklapanjeUAreni() RETURNS TRIGGER AS
$$
BEGIN
 IF ( (SELECT COUNT(*)
 FROM borba NATURAL JOIN igra
WHERE idArena = (SELECT idArena FROM igra WHERE igra.idIgra = NEW.idIgra)
  AND provjeriPreklapanje(borba.pocetakTermin, borba.trajanje,
                                NEW.pocetakTermin, NEW.trajanje)= 1
) > 0
    )
 THEN
 RAISE EXCEPTION 'Pogreška: željeni termin se preklapa s postojećim terminom u istoj
areni.';
END IF;

RETURN NEW;

END;
$$ language plpgsql;

CREATE TRIGGER insert_Borba
 BEFORE INSERT ON borba
 FOR EACH ROW EXECUTE FUNCTION preklapanjeUAreni();
```

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;
```

Napisati SQL naredbe kojima će administrator sustava:

a) kreirati ulogu *igrac* te toj ulozi omogućiti pregled naziva igre i podatka koliko je puta ta igra odigrana 2019. godine (uzevši u obzir završene borbe čiji je početak bio 2019. godine).
b) ulozi *igrac* omogućiti izmjenu naziva igre u tablici *igra*.
c) korisniku *mwest* dodijeliti ovlasti za uspostavu korisničke sjednice (lozinku postaviti na *mwestPwd*) uz korištenje opcije NOINHERIT. Dodijeliti mu također ovlasti za spajanje na bazu podataka *tagfight*, kao i ovlasti za rad s ulogom *igrac*.
d) Koje SQL naredbe korisnik *mwest* treba obaviti nakon što uspostavi korisničku sjednicu, kako bi mogao koristiti dozvole dodijeljene ulozi *igrac*?

a)

```
CREATE VIEW borbe2019 AS

SELECT igra.nazivIgra, COUNT(*) AS brojOdigranihIgara

FROM borba NATURAL JOIN igra

WHERE EXTRACT (YEAR FROM borba.pocetakTermin) = 2019

and zavrsetakBorba IS NOT NULL

GROUP by igra.nazivIgra;


CREATE ROLE igrac;
```

```
--GRANT CONNECT ON DATABASE tagfight TO igrac; ne treba jer nikada neće
uspostavljati korisničku sjednicu niti doći u priliku da se spaja na
bazu podataka

GRANT USAGE ON SCHEMA public TO igrac;

GRANT SELECT ON borbe2019 TO igrac;
```

b)

```
GRANT UPDATE public.igra(nazivIgra) TO igrac;
```

c)

```
CREATE USER mwest WITH PASSWORD 'mwestPwd' NOINHERIT;

GRANT CONNECT ON DATABASE tagfight TO mwest;

GRANT igrac TO mwest;

--GRANT USAGE ON SCHEMA public TO mwest; ne treba mu jer će dobiti preko
igrac
```

d)

```
SET ROLE igrac;
```

6. (3 boda)

a) Koliko razina internih čvorova, ne računajući korijen, ima maksimalno popunjeno B^+-stablo reda n=23 koje sadrži 267 674 n-torki?

b) Navedite glavnu razliku između internog čvora i lista B^+-stabla s obzirom na podatkovne strukture na koje njihove kazaljke pokazuju. Koja je specifičnost zadnje kazaljke lista B^+-stabla?

Rj.

a) maksimalna popunjenost:

=> interni čvorovi: n kazaljki => 23 na čvorove

=> list: n-1 kazaljki na blokove s podatcima => 22

$\log_{23}(267\,674/22) = 3$ 3 razine int. čvorova (uključujući i korijen)

=> 1 razina je korijen, dakle 2 su razine ostalih internih čvorova

Struktura stabla: K-Č-Č-L => Stablo ima 2 razine internih čvorova koji nisu korijen.

b) Kazaljke internog čvora pokazuju na interne čvorove na idućoj (dubljoj) razini stabla ili listove stabla.
Kazaljke lista pokazuju na blokove s podatcima, osim zadnje, koja pokazuje na idući list.

7. (5 bodova) Zadana je relacijska shema *NaruciLijek* gdje se evidentiraju pacijentove narudžbe lijekova. Pacijent u jednom danu može naručiti više različitih lijekova. Isti lijek pacijent može naručiti više puta, ali ne istoga dana. Jedan lijek jednom pacijentu uvijek odobrava jedan liječnik. Različite lijekove mogu odobriti različiti liječnici. Atributi relacijske sheme *NaruciLijek* su:

sifPacijent	šifra pacijenta
imePacijent	ime pacijenta
prezPacijent	prezime pacijenta
sifLijek	šifra lijek
nazLijek	naziv lijek
sifProizv	šifra proizvođača
nazProizv	naziv proizvođača
datNarudzba	datum zahtjeva pacijenta za određeni lijek
datOdobrenje	datum odobrenja
sifLijecnik	šifra liječnika koji odobrava lijek
imeLijecnik	ime liječnika
prezLijecnik	prezime liječnika

Odrediti ključ relacijske sheme NaruciLijek 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 NARUCI_LIJEK.

Rješenje upisati u prostor za unos teksta ispod zadatka.

Rješenje:

1NF
NL_1 = sifPacijent, sifLijek, datNarudzba, imePacijent, prezPacijent, nazLijek, sifProizv, nazProizv, datOdobrenje, sifLijecnik, imeLijecnik, prezLijecnik

2NF
PACIJENT = sifPacijent, imePacijent, prezPacijent
NL_1 = sifLijek, nazLijek, sifProizv, nazProizv
ODOBRENJE = sifPacijent, sifLijek, datNarudzba, datOdobrenje
NL_3 = sifPacijent, sifLijek, sifLijecnik, imeLijecnik, prezLijecnik

3NF
LIJEK = sifLijek, nazLijek, sifProizv
PROIZVODJAC = sifProizv, nazProizv
ODOBRAVA = sifPacijent, sifLijek, sifLijecnik
LIJECNIK = sifLijecnik, imeLijecnik, prezLijecnik

NARUCI_LIJEK = PACIJENT, LIJEK, LIJECNIK, ODOBRAVA, ODOBRENJE, PROIZVODJAC

8. (**5 bodova**)

U PostgreSQL SUBP su nad bazom podataka koja se koristi u ispitu aktivne su samo dvije sjednice S1 i S2 u kojima se izvode dole navedene naredbe, prema brojevima navedenim ispred svake naredbe.

Korisnik kojem je zbog zaključavanja dojavljena pogreška, privremeno obustavlja obavljanje daljnjih naredbi **ali ne prekida transakciju**, a drugi korisnik nastavlja s radom dok ne izvede sve svoje naredbe ili dok se i njemu ne dogodi pogreška zbog nemogućnosti postavljanja ključa. Ako se korisniku oslobodi ključ na kojeg čeka, on nastavlja s obavljanjem svojih naredbi.

Za naredbe {4} – {7}, napišite koja vrsta ključa na koji objekt se traži od SUBP, uspijeva li postavljanje ključa i koliko dugo će ključ biti postavljen. Ako se ključ ne traži ili ne uspije postaviti obrazložite zbog čega. Ako prilikom izvođenja naredbe dođe do pogreške, navedite u kojoj sjednici pri izvođenju koje naredbe će pogreška biti dojavljena te obrazložite uzrok pogreške.

Objasnite što se s eventualno postavljenim ključevima događa u naredbama {8} i {9}.

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

	Sjednica S1		**Sjednica S2**

{1}
{2}
	`BEGIN TRANSACTION;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
		{3}
		
		{4}
			`BEGIN TRANSACTION;`
			
			`SELECT * FROM igra WHERE idIgra = 30 FOR SHARE;`

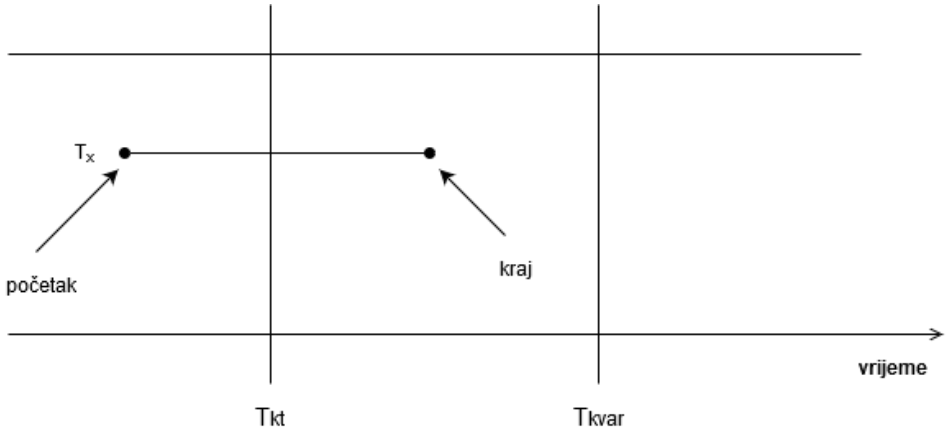
{5}	`UPDATE igra SET nazivIgra = 'Crossside 2' WHERE idIgra = 50;`		
{6}	`SELECT * FROM igra WHERE idIgra >= 20 FOR SHARE;`	{7}	`UPDATE igra SET nazivIgra = 'Bladezone 2' WHERE idIgra = 30;`
{8}	`COMMIT TRANSACTION;`	{9}	`COMMIT TRANSACTION;`

Rješenje

		{4}	traži postavljanje ključa za čitanje (shared lock) na n-torku s idIgra= 30. Ključ se postavlja i traje do kraja transakcije (do obavljanja naredbe {9}).
{5}	traži postavljanje ključa za pisanje (exclusive lock) nad n-torkom s idIgra=50. Ključ se postavlja i traje do kraja transakcije.		
{6}	traži postavljanje ključeva za čitanje nad n-torkama s idIgra>=20. Ključevi se postavljaju i traju do kraja transakcije.	{7}	traži postavljanje ključa za pisanje nad n-torkom s idIgra=30. Ključ se NE postavlja jer je nad tom n-torkom u sjednici S1 naredbom {6} postavljen ključ za čitanje - transakcija čeka dok sjednica 1 ne otpusti ključ nad n-torkom sa šifrom 30.
{8}	otpušta sve u sjednici 1 postavljene ključeve		
		{7}	Već zatraženi ključ za pisanje nad n-torkom s idIgra= 30 se postavlja i traje do kraja transakcije.
		{9}	otpušta sve u sjednici 2 postavljene ključeve

9. **(3 boda)**

U SUBP došlo je do pogreške na razini računalnog sustava u trenutku T_{kvar}. Dnevnik izmjena uspješno je očuvan.



a. Skicirajte (na papiru) transakcije T1, T2, T3, T4 i T5 tako da su za svaku transakciju naznačeni mogući početak i završetak s obzirom na interval između T_{kt} i T_{kvar}, ako je poznato sljedeće:

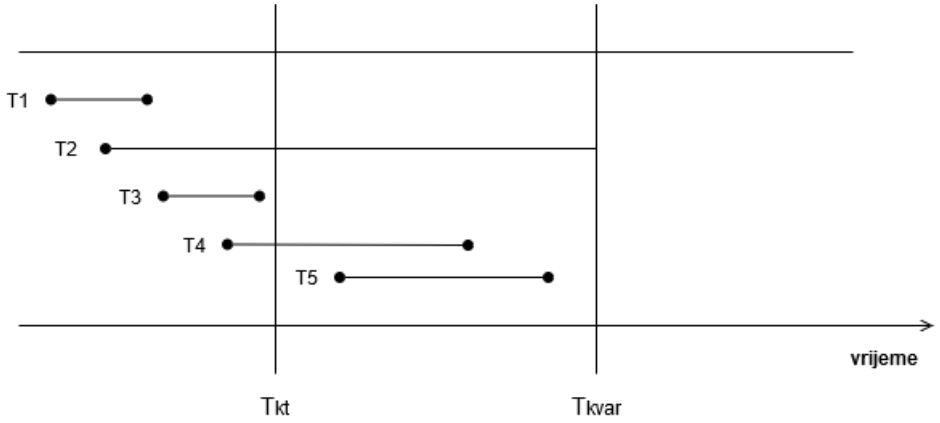
1. Za transakcije T1 i T3 nije potrebno obaviti nikakvu akciju
2. Transakciju T2 potrebno je poništiti
3. Transakcije T4 i T5 potrebno je ponovno obaviti

Između T_{kt} i T_{kvar} nije bilo dodatnih kontrolnih točaka. Primjer skice je na slici.

b. Ako znamo da sustav poštuje svojstvo izdržljivosti transakcija, koji je, prema vašoj skici, posljednji trenutak u kojem je pohranjen sadržaj spremnika dnevnika?

Rješenje:

a. Primjer odgovora



b) Na primjeru sa skice, odgovor je T5 (zbog *write-long-ahead-rule* sadržaj spremnika dnevnika pohranjen je u datoteku dnevnika na disku prije nego li završi procedura potvrđivanja posljednje potvrđene transakcije prije kvara)

10. **(8 bodova)**

Oblikovati ER model baze podataka za planiranje i upravljanje pričanjem priča o životinjama u zoološkom vrtu. Evidentiraju se podatci o životinjama: šifra, ime, vrsta (lavovi, deve, morski lavovi,...), datum rođenja, datum dolaska u zoo. Vrsta je element klasifikacije životinja. Elementi klasifikacije (razine: koljeno, razred, red, porodica, rod, vrsta) su hijerarhijski povezani (npr. vrsta lavovi ima nadređeni rod mačke, koje imaju nadređenu porodicu sisavci, ...). Za svaki element klasifikacije evidentira se šifra i naziv te klasifikacijska razina. Za klasifikacijsku razinu evidentira se šifra i naziv.

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.

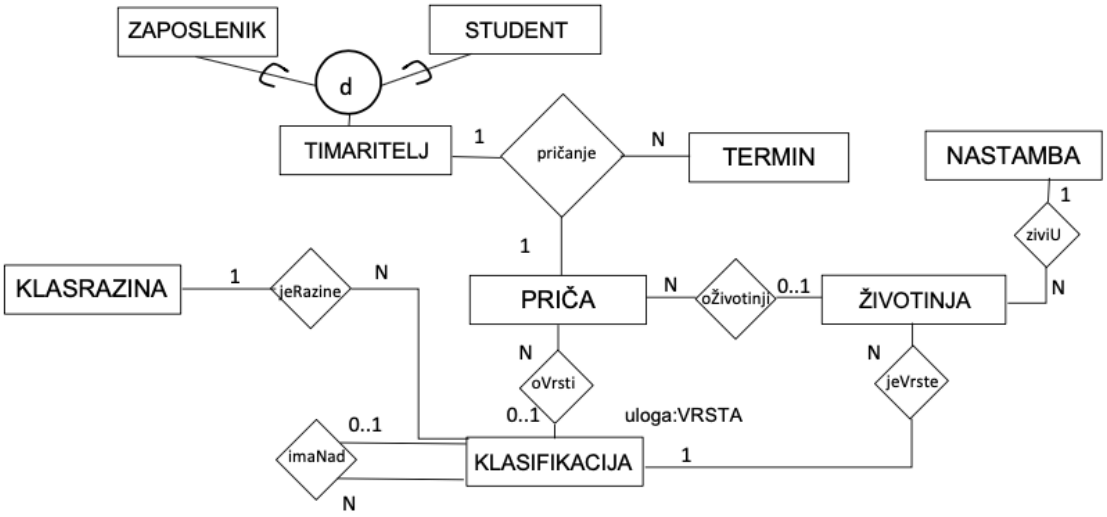
Nekoliko puta dnevno, u točno određenim terminima timaritelji pričaju priče o životinjama. Za termin se evidentira šifra i datum i vrijeme početka. Za timaritelja se evidentira oib, ime i prezime. Timaritelj može biti zaposlenik zoološkog vrta ili student. Za zaposlenika se evidentira datum zaposlenja, za studenta broj sati rada tjedno.

Priče imaju različite teme (način života u prirodnom staništu, dogodovštine iz zoo vrta i sl.). Za priču se evidentira šifra i tema, životinjska vrsta na koju se priča odnosi i životinja (ako se priča odnosi na konkretnu životinju).

Jedan timaritelj može pričati više priča, ali u jednom terminu samo jednu. Jednu priču u jednom terminu priča jedan timaritelj. Jedan timaritelj jednu priču može pričati u različitim terminima.

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 (model nacrtati na papiru)
b) Sheme entiteta i sheme veza **unesite u prostor za slobodni unos teksta ispod teksta zadatka**.



ZIVOTINJA = **sifZivotinja**, imeZivotinja, datRod, datDolazak	imaNad = **sifKlasa**, sifNadKlasa
NASTAMBA = **rbrNastamba**, opisLokacije	jeRazine = **sifKlasa**, sifRazina
KLASIFIKACIJA = **sifKlasa**, nazKlasa	pricanje = **idTermin, sifTimaritelj, sifPrica**
KLASRAZINA = **sifRazina**, nazivRazina	K1 = **idTermin**, **sifTimaritelj**
TERMIN = **idTermin**, datVrijPocPricanja	K2 = **idTermin**, **sifPrica**
PRIČA = **sifPrica**, nazPrica	ziviU = **sifZivotinja**, sifNastamba
TIMARITELJ = **oib**, ime, prezime	jeVrste = **sifZivotinja**, sifKlasa
ZAPOSLENIK = **oib**, datZAposlenja	oVrsti = **sifPrica**, sifKlasa
STUDENT = **oib**, brSatiTjedno	oZivotinji = **sifPrica**, sifZivotinja