Završni ispit iz Baza podataka

27. lipnja 2008.

Zadaci 1. – 4. odnose se na bazu podataka **prodaja** koja sadrži podatke o prodaji računalne opreme u nekoj tvrtci. Bilježe se podaci o djelatnicima, kupcima, proizvodima te o prodajama. Relacija djelatnik sadrži podatke o djelatnicima tvrtke. Za svakog djelatnika evidentira se i djelatnik koji mu je nadređen. Relacije racun i stavkaRacun sadrže podatke o prodanim proizvodima. Jedan racun predstavlja jednu „prodaju“ kojom je kupcu prodan jedan ili više proizvoda. Podaci o proizvodima kupljenim jednim računom nalaze se u relaciji stavkaRacun.
Napomena: Podvučeni atributi čine ključ relacije.

proizvod

sifProizv	INTEGER
naziv	CHAR(255)
cijena	DECIMAL(8,2)

racun

sifRac	INTEGER	šifra računa
sifDjel	INTEGER	šifra djelatnika koji je račun izdao
sifKupac	INTEGER	šifra kupca
datRac	DATE	datum računa
iznos	DECIMAL(8,2)	ukupni iznos računa

kupac

sifKupac	INTEGER
imeKupac	CHAR(30)
prezKupac	CHAR(40)

djelatnik

sifDjel	INTEGER	šifra djelatnika
sifNadDjel	INTEGER	šifra nadređenog djelatnika
imeDjel	CHAR(30)	ime djelatnika
prezDjel	CHAR(40)	prezime djelatnika
login	CHAR(8)	korisničko ime djelatnika

stavkaRacun

sifRac	INTEGER	šifra računa
sifProizv	INTEGER	šifra proizvoda
kolicina	SMALLINT	broj kupljenih proizvoda

1. Napisati po jedan izraz relacijske algebre (**NE SQL UPIT**) kojim će se dobiti:
a) Relacija koja sadrži podatke o djelatnicima i kupcima kojima su djelatnici izdali račun. Relacija sadrži šifru, ime i prezime djelatnika, te šifru, ime i prezime kupca. Za djelatnike koji nisu prodali ništa, relacija sadrži samo jedan zapis u kojem su podaci o kupcu NULL.

$$\pi_{\text{sifDjel,imeDjel,prezDjel, sifKupac, imeKupac, prezKupac}} (\text{djelatnik} *\bowtie (\text{racun} \bowtie \text{kupac}))$$

b) Relacija koja sadrži šifre svih djelatnika koji nikada **nisu** prodali niti jedan proizvod s nazivom 'DVD player'.

$$\pi_{\text{sifDjel}}(\text{djelatnik}) \setminus \pi_{\text{sifDjel}}(\sigma_{\text{naziv = 'DVD player'}} (\text{proizvod} \bowtie \text{stavkaRacun} \bowtie \text{racun}))$$

2. Napisati po jedan SQL upit kojim će se:

a) Za svakog djelatnika koji je nekome nadređen ispisati šifru, ime i prezime te ukupan broj proizvoda koje su prodali njegovi **izravno** podređeni djelatnici. Ispisati samo zapise za koje je ukupan broj prodanih proizvoda veći od 10,000.

```
--1.
SELECT nad.sifDjel, nad.imeDjel, nad.prezDjel, SUM(kolicina)
  FROM djelatnik nad, djelatnik pod, racun, stavkaRacun
 WHERE pod.sifNadDjel = nad.sifDjel
   AND pod.sifDjel    = racun.sifDjel
   AND racun.sifRac = stavkaRacun.sifRac
GROUP BY nad.sifDjel, nad.imeDjel, nad.prezDjel
HAVING SUM(kolicina)>10000
```

b) Ispisati proizvode koji **nisu** prodavani u **tekućoj** godini.

```
-- 2.
```

```
SELECT *
  FROM proizvod
 WHERE sifProizv NOT IN (SELECT sifProizv
                           FROM stavkaRacun, racun
                          WHERE stavkaRacun.sifRac = racun.sifRac
                            AND YEAR(TODAY) = YEAR(datRac))
```

c) Ispisati šifre svih proizvoda koji se nalaze na barem dva različita računa.

```
-- 3.
SELECT DISTINCT sifProizv
FROM stavkaRacun sr1
WHERE EXISTS (
            SELECT *
              FROM stavkaRacun sr2
             WHERE sr1.sifRac <> sr2.sifRac
               AND sr1.sifProizv = sr2.sifProizv
      )

SELECT sifProizv
  FROM stavkaRacun
GROUP BY sifProizv
HAVING COUNT(sifRac) >= 2;
```

d) Za svaki proizvod ispisati šifru i naziv te ukupan broj prodanih primjeraka tog proizvoda. Zapise poredati silazno po broju prodanih primjeraka. U listi se moraju nalaziti i proizvodi koji nikada nisu prodani.

```
-- 4.
SELECT sifProizv, naziv, SUM(kolicina) AS brProd
  FROM proizvod LEFT JOIN stavkaRacun
                       ON proizvod.sifProizv = stavkaRacun.sifProizv
 GROUP BY sifProizv, naziv
 ORDER BY brProd DESC
```

3. Pretpostavite da su u bazi podataka **prodaja** kreirane sve relacije te da su definirani samo primarni ključevi. Napisati naredbe kojima će se osigurati sljedeće:

a) Vrijednost atributa sifRac u relaciji stavkaRacun mora biti jedna od postojećih vrijednosti atributa sifRac u relaciji racun. U slučaju brisanja neke n-torke iz relacije racun, automatski treba obrisati i sve odgovarajuće n-torke iz relacije stavkaRacun.

```
ALTER TABLE stavkaRacun ADD CONSTRAINT FOREIGN KEY (sifRac) REFERENCES racun(sifRac) ON DELETE CASCADE;
```

b) Atribut iznos u relaciji racun smije poprimiti samo vrijednosti veće od 0.

```
ALTER TABLE racun ADD CONSTRAINT CHECK iznos > 0;
```

c) Ne smiju postojati dva djelatnika s istim korisničkim imenom.

```
ALTER TABLE djelatnik ADD CONSTRAINT UNIQUE(login);
```

d) Kad god se u relaciju stavkaRacuna **unese** jedna ili više-ntorki, treba ažurirati i odgovarajući iznos u relaciji racun. Pri tome treba osigurati da ukupni iznos na nekom računu ne prijeđe iznos od 10000kn. Ukoliko se to dogodi, treba spriječiti operaciju unosa u relaciju stavkeRacuna i prijaviti poruku „Prevelik iznos računa!“.

```
CREATE PROCEDURE chkInsStavka(p_sifRac LIKE stavkaRacun.sifRac
                            , p_sifProizv LIKE stavkaRacun.sifProizv
                            , p_kolicina LIKE stavkaRacun.kolicina)

  DEFINE p_cijena LIKE proizvod.cijena;
  DEFINE p_iznos LIKE racun.iznos;
```

```
  SELECT cijena INTO p_cijena
    FROM proizvod
   WHERE sifProizv = p_sifProizv;

  SELECT iznos INTO p_iznos
    FROM racun
   WHERE sifRac = p_sifRac;

  IF p_iznos + p_cijena*p_kolicina > 10000 THEN
    RAISE EXCEPTION -746, 0, 'Prevelik iznos računa!'
  ELSE
    UPDATE racun SET iznos = iznos + p_cijena*p_kolicina
     WHERE sifRac = p_sifRac;
  END IF
END PROCEDURE;

CREATE TRIGGER stavkaIns
  INSERT ON stavkaRacun
  REFERENCING NEW AS novaStavka
  FOR EACH ROW (
    EXECUTE PROCEDURE chkInsStavka(novaStavka.sifRac
                                 , novaStavka.sifProizv
                                 , novaStavka,kolicina);
  )
```

4. Vlasnik baze podataka **prodaja** i svih relacija u bazi je korisnik *admin*. Osim korisnika *admin* nitko nema dozvolu pristupa bazi podataka niti objektima baze podataka. Napisati niz SQL naredbi koje treba obaviti korisnik *admin*, a koje će osigurati sljedeće:

a) Korisniku horvat omogućiti pregled svih podataka u relaciji djelatnik, osim korisničkog imena.

```
GRANT CONNECT TO horvat;
GRANT SELECT(sifDjel, sifNadDjel, imeDjel, prezDjel) ON djelatnik TO horvat;
```

b) Korisniku kolar omogućiti pregled i izmjenu svih podataka o kupcima s mogućnošću dodjeljivanja tih dozvola drugim korisnicima.

```
GRANT CONNECT TO kolar;
GRANT SELECT, UPDATE ON kupac TO kolar WITH GRANT OPTION;
```

c) Svim trenutnim i budućim djelatnicima omogućiti pregled, unos, izmjenu i brisanje podataka samo o onim računima koje su sami oni izdali.

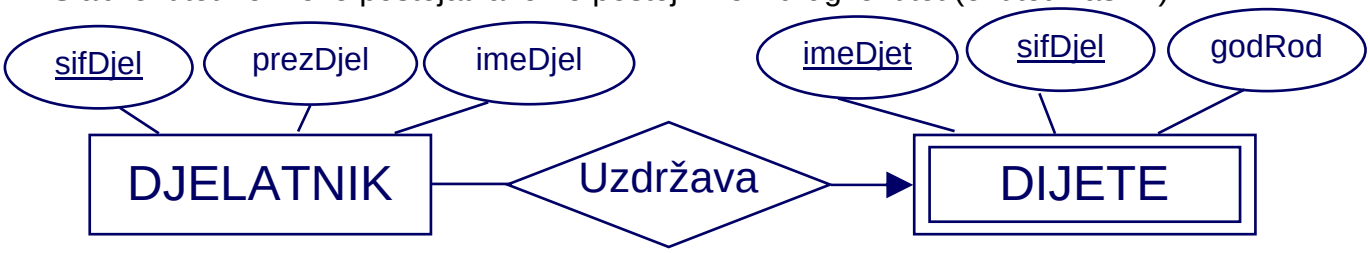
```
CREATE VIEW racunDjel AS
  SELECT * FROM racun
   WHERE sifDjel IN (SELECT sifDjel FROM djelatnik WHERE login = USER)
WITH CHECK OPTION;
GRANT CONNECT TO PUBLIC;
GRANT SELECT, INSERT, UPDATE, DELETE ON racunDjel TO PUBLIC;
```

5. Jedno od važnih svojstava transakcija osigurava da se zadatak kojeg transakcija obavlja mora ili obaviti u cijelosti i samo jednom ili se uopće ne smije obaviti. Navedite naziv tog svojstva transakcije te na vlastitom primjeru objasnite važnost tog svojstva (što bi se moglo dogoditi kad se to svojstvo ne bi poštovalo).

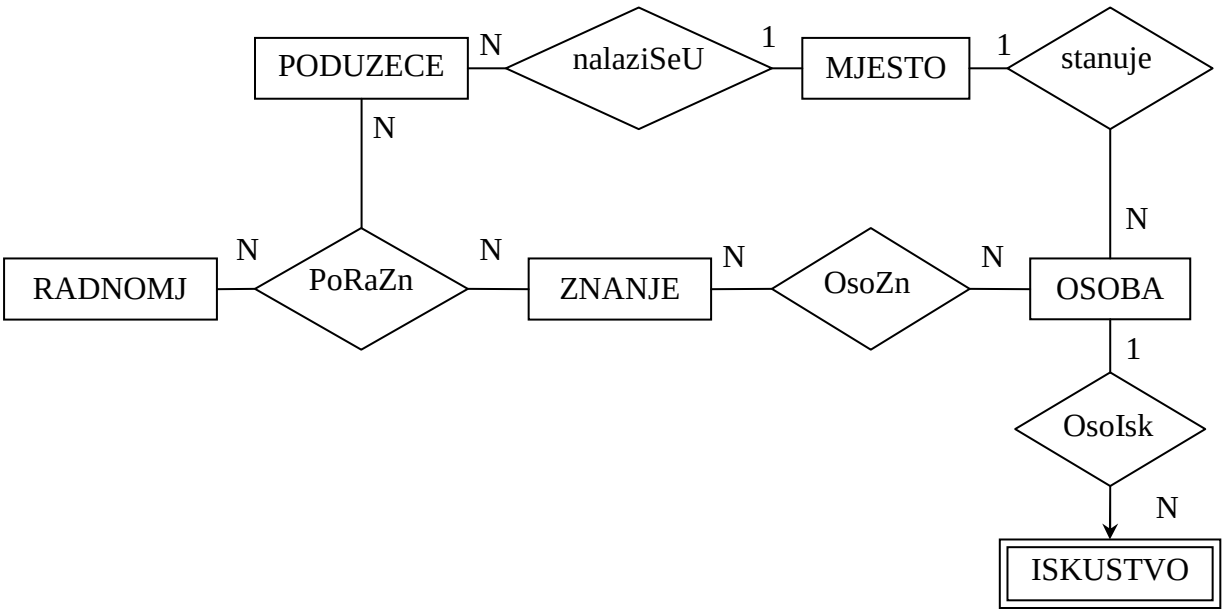
Svojstvo je atomarnost.
Primjer: Prebacivanje novca sa jednog računa na drugi. Prvo se stanje na jednom računu umanji, a zatim se uveća na drugom računu. Kada se ne bi osiguravalo svojstvo atomarnosti, tada bi se moglo dogoditi da se, zbog prekida rada sustava, samo smanji stanje na prvom računu.

6. Što su *slabi entiteti*? Navedite primjer **identifikacijski** slabog entiteta (Objasnite zašto je taj entitet identifikacijski slab).

Slabi entitet ne može postojati ako ne postoji i neki drugi entitet (entitet vlasnik).



7. Agencija za zapošljavanje prikuplja podatke o poduzećima (šifra poduzeća, naziv poduzeća, adresa poduzeća, poštanski broj, naziv mjesta), radnim mjestima (šifra radnog mjesta, naziv radnog mjesta) i potrebnim znanjima (šifra znanja, tekst koji opisuje znanje).
Svako poduzeće za svako radno mjesto za koje traži zaposlenika evidentira jedno ili više potrebnih znanja. Različita poduzeća za isto radno mjesto mogu zahtijevati različita potrebna znanja. Za potrebno znanje za neko radno mjesto u nekom poduzeću evidentira se potrebna razina znanja kao cijeli broj iz intervala [1,5].
Za osobu koja traži posao evidentiraju se osnovni podaci (JMBG, prezime, ime, adresa osobe, poštanski broj, naziv mjesta), znanja koja osoba posjeduje, uz oznaku razine [1-5] te podaci o prethodnim radnim iskustvima (redni broj radnog iskustva koji za svaku osobu počinje od 1, opis radnog iskustva).
Nacrtati ER model i opisati entitete i veze. Entitete, osim slabih entiteta, opisati isključivo vlastitim atributima. Sve sheme moraju zadovoljavati 3NF.



PODUZECE: sifP nazivP	**MJESTO**: pbr nazivMj	**RADNOMJ**: sifRM nazivRM	**ZNANJE**: sifZn opisZn	**OSOBA**: JMBG ime prezime	**ISKUSTVO**: JMBG rbr OpisIsk
PoRaZn: sifP sifRM sifZn razina	**OsoZn**: JMBG sifZn razina	**nalaziSeU**: sifP pbr adresaP	**stanuje**: JMBG pbr adresaO	**OsoIsk**: JMBG rbr	