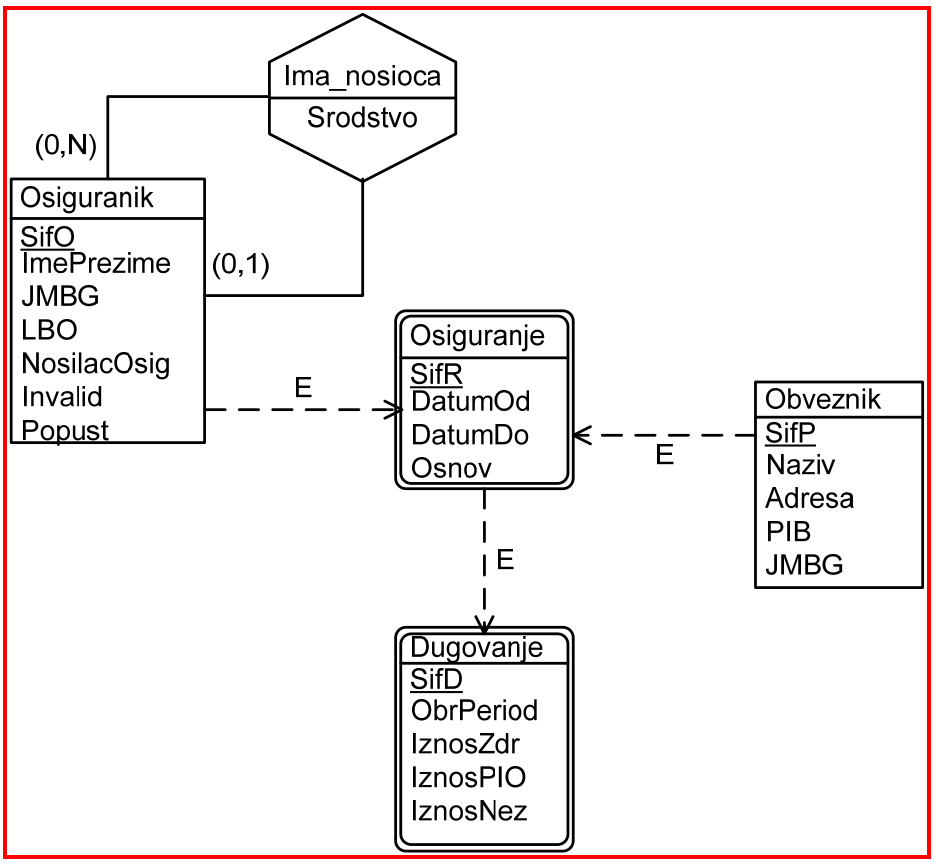
Baze podataka - kolokvijum

Kolokvijum traje **150** minuta

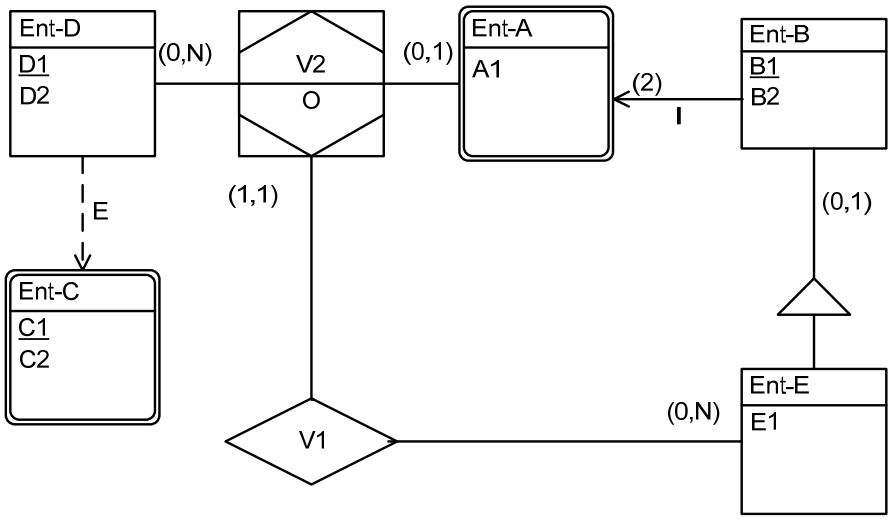
Ime prezime i broj indeksa studenta	Potpis dežurnog	Broj poena

Napomena: Nije dozvoljena upotreba literature.

1. (7) Posmatra se baza podataka za potrebe centralnog registra građana, u kome se čuvaju informacije o dugovanjima i uplatama za potrebe zdravstvenog i socijalnog osiguranja. Osiguranik je fizičko lice koje pored informacije o imenu, prezimenu, matičnom broju (JMBG), adresi ima i informaciju o ličnom broju osiguranika (LBO). Osiguranik može biti sam nosilac osiguranja, ali i ne mora (pri čemu je onda potrebno voditi evidenciju o tome ko je nosilac osiguranja i srodstvu između njih). Ukoliko je osiguranik sa invaliditetom, onda može imati odobren popust. Osiguranik ne mora sve vreme biti pokriven osiguranjem. Odnosno treba voditi evidenciju u toku kog perioda je prijavljen (početak i kraj), koji je osnov osiguranja, kao i tome ko je obveznik osiguranja. Obveznik osiguranja je pravno lice koje ima naziv, adresu, PIB i JMBG. U sistemu se vodi evidencija i o tome da li na osnovu nekog osiguranja, određeni obveznik za određenog osiguranika, za neki obračunski period duguje iznose za zdravstveno osiguranje, penziono i invalidsko osiguranje i nezaposlenost.
Za model prikazan na slici potrebno je ukloniti suvišne (odnosno dodati potrebne) atribute/entitete, a potom dodati potrebne neposredne i posredne odnose.



2. (3) Model entiteta i odnosa, prikazan na slici, prevesti u šemu relacione baze podataka, uz naznaku svih stranih ključeva zaokruživanjem. (Napomena: idenstifikaciona zavisnost je prikazana punom linijom, a egzistencijalna isprekidanom)

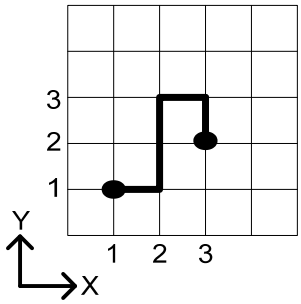


Odgovor:

Ent-A: (B11, B12, A1)	V1: –
Ent-B: (B1, B2)	V2: (B11, B12, D1, O, B1)
Ent-C: (C1, C2, D1)	
Ent-D: (D1, D2)	
Ent-E: (B1, E1)	

3. Dat je deo šeme relacione baze podataka prevoza. U bazi se vodi evidencija o linijama prevoza i njihovim usputnim stanicama. Celokupna mapa evidentirana je u tabeli Lokacija, poput mreže kvadratnog oblika (područje podeljeno na blokove identične veličine). Svaka lokacija se evidentira samo jednom i pri tome se koordinate čuvaju kao celobrojne vrednosti.

LOKACIJA (SifK, X, Y);
LINIJA (SifL, NazivL, BrStanica);
STANICA (SifS, NazivS, SifK, RedBr, SifL);
OBJEKAT (SifO, NazivO, SifK);



Linija:
Slavija,1,1,br1
Terazije,2,1,br2
Moskva,2,2,br3
Trg,2,3,br4
Muzej,3,3,br5
Park,3,2,br6

a.(7) Napisati SQL skript kojim se formira tabela STANICA ukoliko je poznato da je naziv stanice niz od 25 karaktera koji mora biti definisan i ima podrazumevanu vrednost "Prazno". Svaka stanica ima svoj redni broj u okviru linije prevoza na kojoj se nalazi. Treba obezbediti da za svaku lokaciju kroz koju linija prevoza prolazi mora da postoji odgovarajuća stanica kao i da linija prevoza mora biti poligonalna linija.

Odgovor:

```
CREATE TABLE Stanica
( SifS INT PRIMARY KEY,
  NazivS CHAR(25) NOT NULL DEFAULT = 'Prazno',
  SifK INT NOT NULL REFERENCES Lokacija ON UPDATE CASCADE,
  RedBr INT NOT NULL CHECK (RedBr > 0),
  SifL INT NOT NULL REFERENCES Linija ON UPDATE CASCADE,
  UNIQUE(SifL, RedBr),
);

CREATE ASSERTION PoligonalnaLinija
CHECK ( NOT EXISTS ( SELECT *
                       FROM Stanica S, Lokacija L
                      WHERE S.SifK = L.SifK AND S.RedBr > 1
                        AND NOT EXISTS (SELECT *
                                          FROM Stanica S2, Lokacija L2
                                         WHERE S2.SifL = S.SifL
                                           AND S2.RedBr = S.RedBr – 1
                                           AND S2.SifK = L2.SifK
                                           AND ( ( (L2.X-L.X) IN (-1, 1)
                                                   AND L2.Y = L.Y)
                                                 OR
                                                 ( (L2.Y-L.Y) IN (-1, 1)
                                                   AND L2.X = L.X)
                                               )
                                       )
                    )
          );


CREATE ASSERTION SveStanice
CHECK  (NOT EXISTS ( SELECT L.SifL
                       FROM Linija L, Stanica S
                      WHERE L.SifL = S.SifL
                      GROUP BY L.SifL, L.BrStanica
                      HAVING L.BrStanica <> COUNT(S.SifS)
                    )
          );
```

b. (3) Sastaviti iskaz relacione algebre koji vraća parove šifara linija koje imaju bar jednu zajedničku lokaciju.

Odgovor:

$$\pi_{S1.SiS,\,S2.SifS}(\text{Stanica S1} \underset{\substack{S1.SifS<S2.SifS \\ S1.SifK=S2.SifK}}{\infty} \text{Stanica S2}) \rightarrow RESENJE(SifS1, SifS2)$$

---

c. (5) Sastaviti SQL skript koji vraća šifre i nazive onih linija kod kojih je broj usputnih stanica za jedan veći od minimalnog rastojanju između lokacija početne i krajnje stanice.

Odgovor:

```
CREATE VIEW LinijaStanice(SifL, NazivL, BrStanica, PocX, PocY, KraX, KraY)
   AS    SELECT L.SifL, L.NazivL, L.BrStanica, L1.X, L1,Y, L2.X, L2.Y
         FROM Linija L, Stanica S1, Stanica S2, Lokacija L1, Lokacija L2
         WHERE L.SifL = S1.SifL AND L.SifL=S2.SifL
            AND S1.RedBr = 1 AND S2.RedBr = L.BrStanica
            AND S1.SifK = L1.SifK AND S2.SifK = L2.SifK;

SELECT SifL, NazivL
FROM LinijaStanica
WHERE BrStanica = 1 + (CASE WHEN PocX>KraX THEN PocX-KraX ELSE KraX-PocX END) +
                      (CASE WHEN PocY>KraY THEN PocY-KraY ELSE KraY-PocY END);
```
