

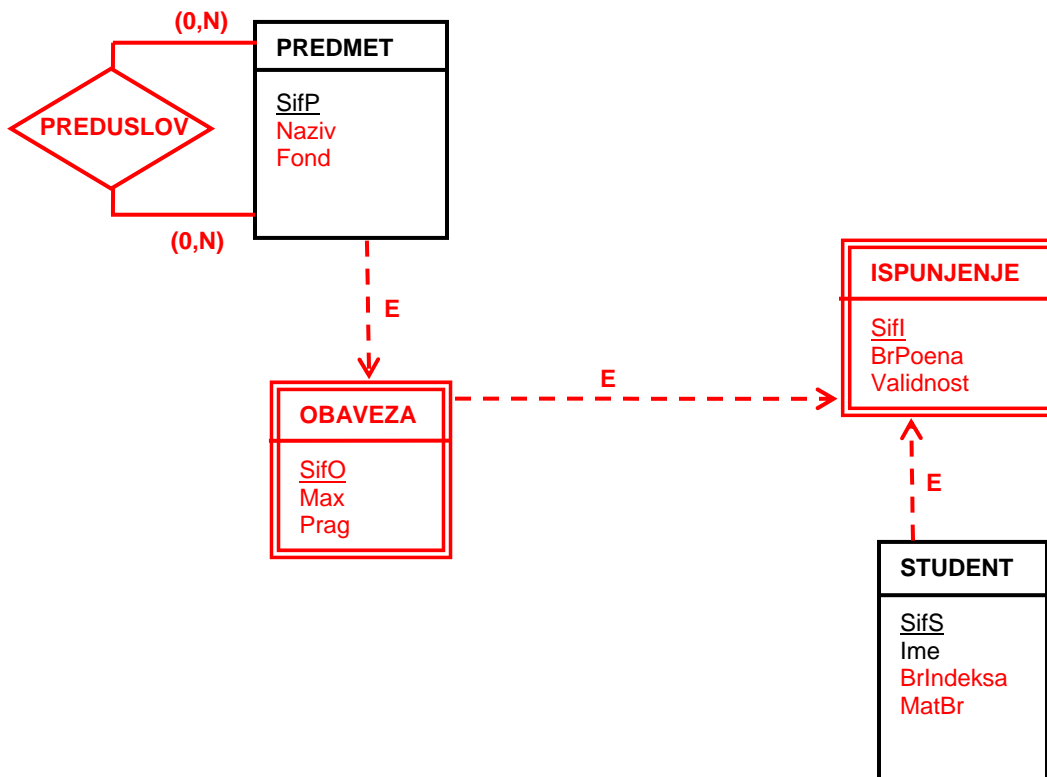
## Baze podataka - kolokvijum

Kolokvijum traje **120** minuta

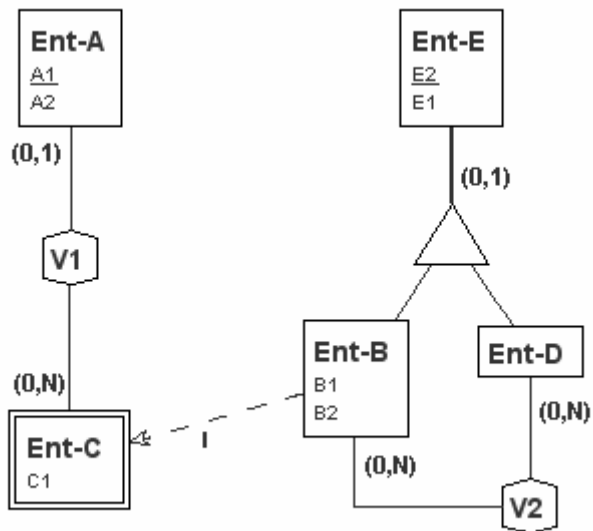
Ime prezime i broj indeksa studenta	Potpis dežurnog	Broj poena

**Napomena:** Nije dozvoljena upotreba literature.

1. (7) Posmatra se deo baze podataka za potrebe nastavničkih servisa. U sistemu se vodi evidencija o predmetima i obavezama koje studenti moraju da ispune u okviru tih predmeta. Za studenta se prati broj indeksa, ime i matični broj. Za predmet su pored naziva, fonda časova, obaveza koje je student dužan da ispuni, definisani i predmeti koji su preduslov za polaganje tog ispita. Svaka obaveza ima definisan maksimalni broj poena koje je moguće ostvariti kao i neophodan prag poena kako bi obaveza bila priznata kao ispunjena, takođe se evidentira se i koliko je poena student ostvario. Student ima pravo na nadoknadu obaveze. Student takođe može da proizvoljan broj puta ponovi neku od obaveza kako bi ostvario veći broj poena. Pri ponovnom ispunjenju obaveze, prethodno ispunjenje obaveze se ne briše iz evidencije, već se označava kao nevalidno. Za model prikazan na slici potrebno je ukloniti suvišne (odnosno dodati potrebne) attribute/entitete, a potom dodati potrebne neposredne i posredne odnose (slabe entitete nije potrebno duplo zaokruživati).



2. (3) Model entiteta i odnosa, prikazan na slici, prevesti u šemu relacione baze podataka, uz naznaku svih stranih ključeva zaokruživanjem.




---

Odgovor:

Ent-A: (A1, A2)

Ent-B: (E2, B1, B2)

Ent-C: (E2, C1)

Ent-D: (E2)

Ent-E: (E2, E1)

V1: (A1, E2)

V2: (E2B, E2D)

---

3. Dat je deo šeme relacione baze podataka za potrebe kompanije za distribuciju robe. Pri tome se u tabeli PRODAJA evidentira svaka pojedinačna prodaja neke robe koju je određeni distributer obavio:

ROBA (SifR, Naziv, Opis);

DISTRIBUTER (SifD, Naziv, Adresa);

PRODAJA (SifN, Datum, Kolicina, Cena, SifD, SifR);

a.(4) Napisati SQL upit koji vraća isti rezultat kao i sledeći upit relacionog računa domena:

$$\{ \langle SifD \rangle \mid \exists Naziv, Adresa (\langle SifD, Naziv, Adresa \rangle \in distributor \\ \wedge \exists SifNA, DatA, CenaA, SifRA (\langle SifNA, DatA, 120, CenaA, SifD, SifRA \rangle \in prodaja \\ \wedge \neg \exists SifNB, DatB, KolB, SifDB, CenaB (\langle SifNB, DatB, KolB, CenaB, SifDB, SifRA \rangle \in prodaja \\ \wedge CenaA > CenaB)) \} \}$$


---

Odgovor:

```
SELECT DISTINCT D.SifD
FROM DISTRIBUTER D, PRODAJA P
WHERE D.SifD = P.SifD
      AND P.Kolicina = 120
      AND NOT EXISTS ( SELECT *
                       FROM PRODAJA P2
                       WHERE P.SifR = P2.SifR
                          AND P.Cena > P2.Cena
                       );
```

-- tabela DISTRIBUTER nije morala da se koristi, jer se u rezultatu koristi samo SifD

**b.** (6) Sastaviti SQL skript koji za sve distributere koji su prodali određenu robu vrši njihovo rangiranje na osnovu ukupne količine te robe koju su oni prodali. Drugim rečima, svi distributeri koji su prodali neku robu, dobiće rang na osnovu toga koliko su te robe prodali. Skript treba da vrati šifru distributera, šifru robe i rang tog distributera za tu robu. Distributer koji je prodao najviše određene robe ima rang 1. Isti rang imaju distributeri koji su prodali istu količinu robe. Distributer koji nikada nisu prodali određenu robu ne dobijaju rang za tu robu. (Napomena: nije dozvoljena upotreba funkcije DENSE\_RANK)

---

Odgovor:

```
CREATE VIEW UkupnaProdaja (SifD, SifR, Ukupno)
AS SELECT SifD, SifR, SUM(Kolicina)
   FROM PRODAJA
   GROUP BY SifD, SifR;
```

```
SELECT U1.SifD, U1.SifR, COUNT(DISTINCT U2.Ukupno)
FROM UkupnaProdaja U1, UkupnaProdaja U2
WHERE U1.SifR = U2.SifR
      AND U1.Ukupno <= U2.Ukupno
GROUP BY U1.SifD, U1.SifR;
```