Zadaci za vježbu

(uz predavanje 5 - SQL)

Jednostupčani podupiti + EXISTS

U zadacima iz ovog poglavlja koriste se relacije iz baze podataka **studAdmin.** Detaljnije objašnjenje **studAdmin** baze podataka možete pronaći na web stranicama predmeta.

1. Ispisati vrijednosti svih atributa relacije student za studente koji niti jedan upisani predmet nisu položili s ocjenom 5, a prezime im počinje slovom E.

2. Ispisati različite nazive mjesta u kojima niti jedan student ne stanuje, a barem jedan student je rođen.

3. Ispisati sve podatke studenata koji su rođeni u mjestima u kojima ne stanuje niti jedan student.

4. Ispisati JMBAG, ime, prezime, datum i mjesto rođenja studenata rođenih u Krapinsko-zagorskoj županiji iste godine u kojoj je zaposlen bilo koji aktivan nastavnik.
   Zapise poredati po datumu rođenja tako da se na vrhu popisa nalaze najmlađi studenti, a prezimena i imena studenata s jednakim datumom rođenja poredati abecedno.

5. Za svaku nastavnu grupu iz 2006/2007 akademske godine za koju su se predavanja mogla održati u svakoj dvorani iz B zgrade (prvo slovo oznake dvorane označava zgradu kojoj dvorana pripada) ispisati akademsku godinu, oznaku grupe, deklarirani kapacitet grupe i nazive predmeta za koje je grupa postojala.
   Smatra se da se predavanja za određenu grupu mogu održati u određenoj dvorani ako je kapacitet dvorane veći ili jednak deklariranom kapacitetu grupe (maksimalni broj studenata koji mogu istovremeno prisustvovati nastavi u toj grupi).

6. Ispisati ime, prezime i naziv mjesta stanovanja onih studenata koji su stariji od svih ostalih studenata koji stanuju u istom mjestu. Ukoliko u nekom mjestu stanuje samo jedan student smatra se da je on najstariji (unatoč tome što drugih studenata u tom mjestu nema).

7. Ispisati šifru, ime i prezime za nastavnike koji stanuju u Splitsko-dalmatinskoj županiji i imaju manji koeficijent za plaću od barem jednog nastavnika koji stanuje u Osječko-baranjskoj županiji.

8. Ispisati JMBAG, ime i prezime svih studenata koji su s ocjenom 5 položili predmet kojeg niti jedan drugi student nije položio s istom ocjenom istog datuma.

9. Ispisati broj različitih predmeta koje su studenti položili kod nastavnika s imenom Slaven na datum kad taj isti predmet nije polagao niti jedan drugi student.

10. Za svakog nastavnika čije ime počinje slovom M ispisati: šifru nastavnika, prezime, ime, broj različitih studenata kojima je predavao, broj upisa predmeta koji su rezultirali prolaznom ocjenom, prosječnu ocjenu onih upisa predmeta koji su rezultirali prolaznom ocjenom.

11. Ispisati šifre i nazive županija u kojima je rođen barem jedan student čije prezime počinje slovom P i stanuje barem jedan student čije prezime počinje slovom M.

12. Za studente kojima niti jedna ocjena nije dodijeljena u onaj dan u tjednu na koji su rođeni treba ispisati njihovo ime, prezime i datum rođenja. Dakle, tražene podatke treba ispisati za sve studente koji su rođeni npr. u utorak, a ni jedna ocjena im nije dodijeljena u utorak itd. Za studente za koje nije poznat datum rođenja ne ispisuju se podaci.

13. Ispisati sve podatke o dvoranama u kojima je predavao nastavnik čije ime počinje slovom 'I' i nije predavao niti jedan nastavnik koji više nije u radnom odnosu (odnosno, još je uvijek u radnom odnosu).

14. Ispisati jmbag, ime i prezime studenata za one studente čiji je prosjek pozitivnih ocjena barem za 0.7 veći od prosjeka pozitivnih ocjena ostalih studenata. Za studente koji nemaju pozitivnu ocjenu niti iz jednog predmeta, ne treba ništa ispisivati.

15. Ispisati naziv i poštanski broj za svako mjesto za koje je broj nastavnika koji u njemu stanuju veći od broja studenata koji su u njemu rođeni.

16. Ispisati šifru, ime i prezime nastavnika kod kojih je pozitivnu ocjenu dobilo više studenata nego kod bilo kojeg drugog nastavnika.

(Ne)potrebno korištenje podupita, presjek, razlika

1. Ispisati sve podatke o nastavnicima koji stanuju u županiji koja u nazivu sadrži podniz "zagreb" (bez obzira na velika odnosno mala slova).

2. Ispisati sve organizacijske jedinice koje imaju nadređenu i barem jednu podređenu organizacijsku jedinicu.

3. Ispisati sva mjesta u kojima:

a) stanuje barem jedan nastavnik
b) ne stanuje niti jedan nastavnik

4. Za svakog studenta ispisati jmbag, ime, prezime i prosjek ocjena (za one koji nisu položili niti jedan ispit ispisati null). Poredati po prezimenu studenta uzlazno.

5. Zadane su relacije:

r1

ime	prez
Ivo	Horvat
Ana	Horvat
Ivo	Kolar

r2

ime	prez
Ivo	Horvat
Ana	Kolar

Napišite upite kojima ćete izračunati:

a) r1 ∩ r2

b) r1 \ r2

6. Ovaj zadatak odnosi se na presjek i razliku relacija. Budući da u bazi studAdmin nema unijski kompatibilnih relacija definirajmo dvije nove relacije na temelju relacije orgjed:

- imaNadOrgJed – sadrži org.jedinice koje imaju nadređenu org.jedinicu
- imaPodOrgJed – sadrži org.jedinice kojie imaju barem jednu podređenu org.jedinicu

```
CREATE TABLE imaNadOrgJed (
    sifOrgJed     INTEGER    NOT NULL,
    nazOrgJed     NCHAR(120) NOT NULL,
    sifnadOrgJed  INTEGER
);

CREATE TABLE imaPodOrgJed (
    sifOrgJed     INTEGER     NOT NULL,
    nazOrgJed     NCHAR(120)  NOT NULL,
    sifnadOrgJed  INTEGER
);
```

```
INSERT INTO imaNadOrgJed
SELECT *
  FROM OrgJed
 WHERE sifnadOrgJed IS NOT NULL;

INSERT INTO imaPodOrgJed
SELECT OrgJed.*
  FROM OrgJed
 WHERE sifOrgJed IN (SELECT sifnadOrgJed FROM OrgJed);
```

Provjerite sadržaj:

```
SELECT * FROM imaNadOrgJed; -- 130 zapisa
SELECT * FROM imaPodOrgJed; -- 18 zapisa
```

a) Napišite upit kojim ćete dobiti organizacijske jedinice koje imaju nadređenu i barem jednu podređenu (kao u zadataku 2) odnosno:

$$\texttt{imaNadOrgJed} \cap \texttt{imaPodOrgJed}$$

b) Napišite upit kojim ćemo dobiti org.jedinice koje imaju nadređenu a nemaju niti jednu podređenu odnosno:

$$\texttt{imaNadOrgJed} \quad \texttt{imaPodOrgJed}$$

c) Napišite upit kojim ćemo dobiti krovne org.jedinice: one koje imaju barem jednu podređenu org.jedinicu a nemaju nadređenu:

$$\texttt{imaPodOrgJed} \setminus \texttt{imaNadOrgJed}$$

Kada ste gotovi s zadatkom obrišite novostvorene tablice s naredbama:

```
DROP TABLE imaNadOrgJed;
DROP TABLE imaPodOrgJed;
```

INSERT, UPDATE i DELETE naredbe

NAPOMENA: Zadaci koji slijede odnose se na studAdmin bazu podataka koju ste dobili u sklopu programske potpore za kolegij i na njoj možete testirati svoja rješenja i točna rješenja zadataka. Kako je u ovom poglavlju riječ o INSERT, UPDATE i DELETE naredbama, treba voditi računa da će te naredbe promijeniti sadržaj baze podataka pa uzastopnim pokretanjem rješenja istog zadatka nećete dobiti isti rezultat (npr. zapis ne možete obrisati ako ste ga obrisali prethodnom naredbom).
Ako na tako izmijenjenoj bazi podataka budete rješavali i zadatke za domaću zadaću, može se dogoditi ispis koji dobijete ne izgleda logično jer sadržaj baze nije onaj početni.
Bazu podataka možete vratiti u početno stanje na dva načina.

1. Možete se poslužiti Skriptom za kreiranje baze podataka koja se nalazi u Materijalima na stranici predmeta.
2. Možete sa CD-a sa programskom potporom ponovno raspakirati izvornu datoteku sa virtualnim računalom koja sadrži bazu podataka u izvornom stanju. U ovom slučaju morate ponovno podesiti spajanje AquaDataStudio SQL editora na virtualno računalo.

1. Kreirajte tablicu ***polozenPredmet***.U novu tablicu potrebno je upisati *ime*, *prezime*, *datum* dobivene ocjene i samu *ocjenu*. Nova tablica sadrži podatke samo za predmet sa *šifrom* 1. Isto tako upisuju se podaci samo za one studente koji su iz tog predmeta dobili odličan ili vrlo dobar.

2. Kreirajte tablicu ***prosjecneOcjene***. U tablicu je potrebno unijeti *šifru predmeta*, *naziv predmeta* i *prosječnu ocjenu* studenta za taj predmet iz studadmin baze.

3. Potrebno je izvršiti određene promjene u tablici gdje se nalazi popis svih upisanih studenata. Neki studenti su rođeni u Zagrebu, ali njihovo mjesto stanovanja više nije Zagreb. Svi drugi studenti (dakle oni koji ne spadaju u grupu spomenutih) trebaju biti izbrisani iz baze.

4. Napišite sql kod za kreiranje tablice ***polozenpredmet*** nakon čega će biti moguće izvesti naredbu kojom upisujemo ime i prezime osoba koje su polozile neki predmet. *Ocjena* i *datum polaganja* se postavljaju na 0 za ocjenu, odnosno null za datum, i ne unose se insert naredbom.
   Primjer naredbe:

   ```
   INSERT INTO polozenpredmet (ime,prezime)
   VALUES
   ('Mima','Maric')
   ```

5. Potrebno je izvršiti promjene u tablici gdje se nalazi popis svih upisanih studenata. Datum rođenja studenta potrebno je povećati za jedan dan (odnosno ako je netko rodjen 1.1.1980. sada treba pisati 2.1.1980.). Ustanovljeno je da korekciju treba izvršiti samo za studente koji se zovu **Tomislav**, **nisu rođeni** u Zagrebu, ali **trenutno žive** u Zagrebu.

6. Potrebno je napraviti izmjenu u tablici **student**. Potrebno je pronaći studente koji imaju najveći broj položenih ispita. Studente koji se nađu na tom popisu treba izbrisati iz baze.

7. Potrebno je izvršiti promjene u tablici gdje se nalazi popis svih mjesta i pripadajućih im županija. Prvo treba pronaći iz koje to županije dolaze nastavnici sa najvećim prosječnim koeficijentom. Kada znamo o kojim je županijama riječ, njihovim šiframa treba dodati još jednu nulu na kraju.

8. Potrebno je stvoriti novu tablicu ***maxgrupa*** koja će sadržavati podatak o ukupnom kapacitetu studenata za svaku pojedinu grupu ako je taj kapacitet veći od 80. Oznaku grupe i pripadajući kapacitet unijeti u tablicu ***maxgrupa***.

9. Potrebno je izvršiti određene promjene u tablici u kojoj se nalazi popis svih upisanih studenata. Svim studentima kojima je ocjena **manja od 5**, potrebno je **povećati ocjenu za jedan**. Isto tako za sve šifre predmeta koje su prikazane jednoznamenkastim brojem potrebno je **dodati još jednu nulu**.
10. Kao i u prethodnom zadatku potrebno je izvršiti određene promjene u tablici u kojoj se nalazi popis svih upisanih studenata. Svim studentima kojima je ocjena manja od 5, potrebno je povećati ocjenu za jedan. Isto tako za sve šifre predmeta koje su prikazane jednoznamenkastim brojem potrebno je dodati još jednu nulu. Zadatak **riješiti tako da se prvo definira lista atributa koje mijenjamo u tablici**.

Rješenja

Jednostupčani podupiti + EXISTS

```
1.
SELECT * FROM student
 WHERE prezimeStudent LIKE 'E%'
   AND JMBAG NOT IN (SELECT DISTINCT JMBAG
                       FROM upisanPredmet
                      WHERE ocjena = 5)
```

Ispravno je i rješenje napisano pomoću koreliranog podupita. Crvenom bojom je označen atribut iz vanjskog upita kojeg koristi podupit.

```
SELECT * FROM student
 WHERE student.prezimeStudent LIKE 'E%'
   AND NOT EXISTS (SELECT upisanPredmet.JMBAG
                     FROM upisanPredmet
                    WHERE upisanPredmet.JMBAG = student.JMBAG
                      AND ocjena = 5);
2.
SELECT DISTINCT nazMjesto
  FROM mjesto
 WHERE pbr NOT IN (SELECT pbrStanStudent FROM student
                    WHERE pbrStanStudent IS NOT NULL)
        AND pbr IN (SELECT pbrRodStudent FROM student);
```

U gornjem rješenju je nepotrebno upotrijebljen podupit za ispitivanje je li u mjestu rođen barem jedan student. Ispravnije je upotrijebiti spajanje relacija mjesto i student uz uvjet mjesto.pbr = student.pbrRod. Ispravnije je rješenje koje slijedi.

```
SELECT DISTINCT nazMjesto
  FROM mjesto, student
 WHERE mjesto.pbr = student.pbrRodStudent
   AND mjesto.pbr NOT IN (SELECT pbrStanStudent FROM student
                           WHERE pbrStanStudent IS NOT NULL);
```

Pokušajte gornji upit izvesti bez navođenja uvjeta `WHERE pbrStanStudent IS NOT NULL` u podupitu. U rezultatu nema niti jedne n-torke. Razlog tome je što određeni broj studenata nema evidentirano mjesto stanovanja odnosno vrijednost atributa stud.pbrStan je NULL. U rezultatu podupita će se pojaviti i NULL vrijednost (onoliko puta koliko ima studenata bez evidentiranog mjesta stanovanja), a ukoliko se u rezultatu ovakvog podupita (***WHERE expression* NOT IN (*subquery*)**) nalazi makar jedna NULL vrijednost, rezultat izračunavanja uvjeta selekcije nikad neće biti *true.*

Ispravno je i rješenje napisano pomoću koreliranog podupita. Crvenom bojom je označen atribut iz vanjskog upita kojeg koristi podupit.

```
SELECT DISTINCT nazMjesto
  FROM mjesto, student
 WHERE mjesto.pbr = student.pbrRodStudent
   AND NOT EXISTS (SELECT *
                     FROM student
                    WHERE student.pbrStanStudent = mjesto.pbr);
3.
SELECT *
  FROM student
 WHERE pbrRodStudent IS NOT NULL
   AND pbrRodStudent NOT IN (SELECT pbrStanStudent
                               FROM student
                              WHERE pbrStanStudent IS NOT NULL);
```

Pokušajte gornji upit izvesti bez navođenja uvjeta `WHERE pbrStanStudent IS NOT NULL` u podupitu. U rezultatu nema niti jedne n-torke. Objašnjenje potražite u prethodnom zadatku.

Ispravno je i rješenje napisano pomoću koreliranog podupita. Crvenom bojom je označen atribut iz vanjskog upita kojeg koristi podupit.

```
SELECT *
  FROM student studentRod
 WHERE pbrRodStudent IS NOT NULL
  AND NOT EXISTS (SELECT *
                    FROM student studentStan
                   WHERE studentStan.pbrStanStudent =
                         studentRod.pbrRodStudent);

4.
SELECT JMBAG, student.imeStudent, student.prezimeStudent
     , student.datRodStudent, mjesto.nazMjesto
  FROM student, mjesto, zupanija
 WHERE student.pbrRodStudent = mjesto.pbr
   AND mjesto.sifZupanija = zupanija.sifZupanija
   AND zupanija.nazZupanija = 'Krapinsko-zagorska'
   AND YEAR (student.datRodStudent) = ANY (SELECT YEAR(datumZaposlenOd)
                                             FROM nastavnik
                                            WHERE datumZaposlenDo IS NULL)
ORDER BY student.datRodStudent DESC, student.prezimeStudent,
         student.imeStudent
```

Prema tekstu zadatka dovoljno je da je barem jedan aktivan nastavnik zaposlen u godini rođenja studenta pa je za ovaj problem ispravno upotrijebiti ANY (ili SOME) ključnu riječ, a ne ALL.

```
5.
SELECT grupa.akGodina, grupa.oznGrupa
     , grupa.kapacitet, predmet.naziv
  FROM grupa, predmetGrupa, predmet
 WHERE grupa.akGodina = predmetGrupa.akGodina
   AND grupa.oznGrupa = predmetGrupa.oznGrupa
   AND predmet.sifPredmet = predmetGrupa.sifPredmet
   AND grupa.akGodina = 2006
   AND grupa.kapacitet  <= ALL (SELECT dvorana.kapacitet
                                  FROM dvorana
                                 WHERE oznDvorana LIKE 'B%')
Obavite sljedeću naredbu:
UPDATE dvorana SET kapacitet = NULL
 WHERE oznDvorana = 'B1';
```

Ponovo izvedite upit naveden kao rješenje ovog zadatka. U rezultatu nema niti jedne n-torke. Razlog tome je što je kapacitet dvorane B1, koja zadovoljava uvjet dohvata u podupitu, postavljen na NULL vrijednost pa će se u skupu rezultata podupita pojaviti i NULL vrijednost. Ukoliko se u rezultatu ovakvog podupita (***WHERE expression relationalOperator* ALL (*subquery*)**) nalazi makar jedna NULL vrijednost, rezultat izračunavanja uvjeta selekcije nikad neće biti *true.*

Zaključak je da gore navedeno rješenje nije ispravno tj. u podupitu nedostaje uvjet `AND dvorana.kapacitet IS NOT NULL`. Ispravno rješenje je:

```
   SELECT grupa.akGodina, grupa.oznGrupa
        , grupa.kapacitet, predmet.naziv
     FROM grupa, predmetGrupa, predmet
    WHERE grupa.akGodina = predmetGrupa.akGodina
      AND grupa.oznGrupa = predmetGrupa.oznGrupa
      AND predmet.sifPredmet = predmetGrupa.sifPredmet
      AND grupa.akGodina = 2006
      AND grupa.kapacitet  <= ALL (SELECT dvorana.kapacitet
                                     FROM dvorana
                                    WHERE oznDvorana LIKE 'B%'
                                      AND dvorana.kapacitet IS NOT NULL)
```

Iz navedenog je vidljivo kad god jednostupčani podupiti u rezultatu sadržavaju barem jednu NULL vrijednost (npr. atribut relacije naveden u SELECT listi može poprimiti NULL vrijednost), a koriste se u uvjetima selekcije oblika:

- **WHERE *expression relationalOperator* ALL (*subquery*)**
- **WHERE *expression* NOT IN (*subquery*)**

rezultat izračunavanja uvjeta selekcije nikada neće biti *true.*
Čak i kada relacija trenutno ne sadrži nijednu n-torku s NULL vrijednosti tog atributa, izmjenama sadržaja relacije u budućnosti vjerojatno će se takva n-torka pojaviti i nekad 'ispravan' upit prestaje to biti.

Da biste kapacitet dvorane B1 vratili na staru vrijednost izvedite sljedeću naredbu:

```
UPDATE dvorana SET kapacitet = 100
 WHERE oznDvorana = 'B1';
```

6.

```
SELECT student.imeStudent, student.prezimeStudent, nazMjesto
  FROM student, mjesto
 WHERE student.pbrStanStudent = mjesto.pbr
   AND student.datRodStudent =
      (SELECT MIN(drugiStudent.datRodStudent)
         FROM student drugiStudent
        WHERE student.pbrStanStudent = drugiStudent.pbrStanStudent)
```

7.

```
SELECT sifNastavnik, imeNastavnik, prezimeNastavnik
      FROM nastavnik, mjesto, zupanija
      WHERE nastavnik.pbrStanNastavnik = mjesto.pbr
        AND zupanija.sifZupanija = mjesto.sifZupanija
        AND zupanija.nazZupanija = 'Splitsko-dalmatinska'
        AND koef < ANY (SELECT koef
                          FROM nastavnik, mjesto, zupanija
                         WHERE nastavnik.pbrStanNastavnik = mjesto.pbr
                           AND zupanija.sifZupanija= mjesto.sifZupanija
                           AND zupanija.nazZupanija = 'Osječko-
baranjska')
```

8.

```
SELECT DISTINCT student.JMBAG, imeStudent, prezimeStudent
  FROM student, upisanPredmet
 WHERE student.JMBAG = upisanPredmet.JMBAG
   AND ocjena = 5
   AND upisanPredmet.sifPredmet NOT IN
      (SELECT upisanPredmet2.sifPredmet
         FROM upisanPredmet upisanPredmet2
        WHERE upisanPredmet2.JMBAG <> upisanPredmet.JMBAG
          AND upisanPredmet2.datumOcjena=upisanPredmet.datumOcjena
          AND ocjena = 5)
```

Atribut upisanPredmet.sifPredmet ne može poprimit NULL vrijednost pa nije potreban uvjet AND upisanPredmet.sifPredmet IS NOT NULL.

Ispravno je i rješenje napisano pomoću koreliranog podupita. Crvenom bojom su označen atributi iz vanjskog upita koje koristi podupit.

```
SELECT DISTINCT student.JMBAG, imeStudent, prezimeStudent
  FROM student, upisanPredmet upisanPredmet1
 WHERE student.JMBAG = upisanPredmet1.JMBAG
   AND ocjena = 5
   AND NOT EXISTS
        (SELECT upisanPredmet2.sifPredmet
           FROM upisanPredmet upisanPredmet2
          WHERE upisanPredmet1.JMBAG <> upisanPredmet2.JMBAG
            AND upisanPredmet1.sifPredmet = upisanPredmet2.sifPredmet
```

```
            AND upisanPredmet1.datumOcjena = upisanPredmet2.datumOcjena
            AND upisanPredmet1.ocjena = upisanPredmet2.ocjena)
```

9.

Crvenom bojom su označen atributi iz vanjskog upita koje koristi podupit.

```
SELECT COUNT(DISTINCT upisanPredmet.sifPredmet)
  FROM upisanPredmet, predmetGrupa, nastavnik
 WHERE upisanPredmet.sifPredmet = predmetGrupa. sifPredmet
   AND upisanPredmet.akGodina = predmetGrupa.akGodina
   AND upisanPredmet.oznGrupa = predmetGrupa.oznGrupa
   AND predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
   AND nastavnik.ime = 'Slaven'
   AND NOT EXISTS
          (SELECT *
             FROM upisanPredmet upisanPredmet2
            WHERE upisanPredmet.sifPredmet = upisanPredmet2.sifPredmet
              AND upisanPredmet.datumOcjena = upisanPredmet2.datumOcjena
              AND upisanPredmet.JMBAG <> upisanPredmet2.JMBAG);
```

10.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
SELECT sifNastavnik
     , prezimeNastavnik
     , imeNastavnik
     , (SELECT COUNT(DISTINCT JMBAG)
          FROM upisanPredmet, predmetGrupa
         WHERE upisanPredmet.sifPredmet = predmetGrupa. sifPredmet
           AND upisanPredmet.akGodina = predmetGrupa.akGodina
           AND upisanPredmet.oznGrupa = predmetGrupa.oznGrupa
           AND predmetGrupa.sifNastavnik = nastavnik.sifNastavnik)
     , (SELECT COUNT(JMBAG)
          FROM upisanPredmet, predmetGrupa
         WHERE upisanPredmet.sifPredmet = predmetGrupa. sifPredmet
           AND upisanPredmet.akGodina = predmetGrupa.akGodina
           AND upisanPredmet.oznGrupa = predmetGrupa.oznGrupa
           AND predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
           AND ocjena > 1)
     , (SELECT AVG(ocjena)
          FROM upisanPredmet, predmetGrupa
         WHERE upisanPredmet.sifPredmet = predmetGrupa. sifPredmet
           AND upisanPredmet.akGodina = predmetGrupa.akGodina
           AND upisanPredmet.oznGrupa = predmetGrupa.oznGrupa
           AND predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
           AND ocjena > 1)
       FROM nastavnik
       WHERE imeNastavnik LIKE 'M%'
```

11.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
SELECT DISTINCT sifZupanija, nazZupanija
  FROM zupanija
 WHERE EXISTS (SELECT *
                 FROM mjesto, student
                WHERE mjesto.pbr= student.pbrRodStudent
                  AND prezimeStudent LIKE 'P%'
                  AND mjesto.sifZupanija = zupanija.sifZupanija)
   AND EXISTS (SELECT *
                 FROM mjesto, student
                WHERE mjesto.pbr= student.pbrStanStudent
                  AND prezimeStudent LIKE 'M%'
                  AND mjesto.sifZupanija = zupanija.sifZupanija)
```

ili

```
SELECT DISTINCT sifZupanija, nazZupanija
  FROM zupanija
 WHERE sifzupanija IN (SELECT sifzupanija
                         FROM mjesto, student
                        WHERE mjesto.pbr= student.pbrRodStudent
                          AND prezimeStudent LIKE 'P%' )
   AND sifzupanija IN (SELECT sifzupanija
                         FROM mjesto, student
                        WHERE mjesto.pbr= student.pbrStanStudent
                          AND prezimeStudent LIKE 'M%')
```

12.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
SELECT imeStudent, prezimeStudent, datrodStudent FROM student
 WHERE datrodStudent IS NOT NULL
   AND NOT EXISTS (SELECT * FROM upisanPredmet
                    WHERE student.jmbag = upisanPredmet.jmbag
                      AND WEEKDAY(upisanPredmet.datumOcjena)
                        = WEEKDAY (student.datRodStudent))
```

ili

```
SELECT imeStudent, prezimeStudent, datrodStudent FROM student
 WHERE datrodStudent IS NOT NULL
   AND WEEKDAY (student.datrodStudent) NOT IN
            (SELECT DISTINCT WEEKDAY(upisanPredmet.datumOcjena)
               FROM upisanPredmet
                WHERE student.jmbag = upisanPredmet.jmbag
                  AND upisanPredmet.datumOcjena IS NOT NULL)
```

Uvjet upisanpredmet.datumOcjena IS NOT NULL je dodan, jer može atribut može poprimiti NULL vrijednost. Pogledati objašnjenje u zadatku 5.

13.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
SELECT * FROM dvorana
 WHERE EXISTS (SELECT * FROM predmetGrupa, nastavnik
                WHERE dvorana.ozndvorana = predmetGrupa.oznDvorana
                  AND predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
                  AND nastavnik.imeNastavnik LIKE 'I%')
   AND NOT EXISTS (SELECT * FROM predmetGrupa, nastavnik
                    WHERE dvorana.ozndvorana = predmetGrupa.oznDvorana
                      AND predmetGrupa.sifMastavnik =
                          nastavnik.sifNastavnik
                      AND nastavnik.datumzaposlendo IS NOT NULL)
```

ili

```
SELECT * FROM dvorana
 WHERE oznDvorana IN (SELECT oznDvorana FROM predmetGrupa, nastavnik
                       WHERE predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
                         AND nastavnik.imeNastavnik LIKE 'I%')

   AND oznDvorana NOT IN (SELECT oznDvorana FROM predmetGrupa, nastavnik
                    WHERE predmetGrupa.sifNastavnik =
                          nastavnik.sifNastavnik
                      AND nastavnik.datumZaposlenDo IS NOT NULL)
```

14.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
      SELECT upisanpredmet.jmbag, imeStudent, prezimeStudent
        FROM upisanpredmet, student
       WHERE upisanpredmet.jmbag = student.jmbag
         AND ocjena > 1
       GROUP BY upisanpredmet.jmbag, imeStudent, prezimeStudent
       HAVING AVG(ocjena) > (SELECT AVG(ocjena) FROM upisanpredmet ostali
                              WHERE ostali.jmbag <> upisanpredmet.jmbag
                                AND ocjena > 1) + 0.7
```

15.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
      SELECT pbr, mjesto.nazMjesto
        FROM nastavnik, mjesto
       WHERE nastavnik.pbrstanNastavnik = mjesto.pbr
       GROUP BY  pbr, mjesto.nazMjesto
       HAVING COUNT(*) >
            (SELECT COUNT(*) FROM student, mjesto mjestoDrugo
              WHERE student.pbrRodStudent = mjestoDrugo.pbr
                AND mjesto.pbr = mjestoDrugo.pbr)
```

16.

Crvenom bojom je označen atribut iz vanjskog upita kojeg koriste podupiti.

```
SELECT nastavnik.sifNastavnik, imeNastavnik, prezimeNastavnik
  FROM nastavnik, predmetGrupa, upisanPredmet
 WHERE nastavnik.sifNastavnik = predmetGrupa.sifNastavnik
   AND predmetGrupa.sifPredmet = upisanPredmet.sifPredmet
   AND predmetGrupa.akGodina = upisanPredmet.akGodina
   AND predmetGrupa.oznGrupa = upisanPredmet.oznGrupa
   AND ocjena > 1
   GROUP BY nastavnik.sifNastavnik, imeNastavnik, prezimeNastavnik
   HAVING COUNT(*) > ALL (SELECT COUNT(*)
                            FROM nastavnik nastavnikDrugi
                               , predmetGrupa, upisanPredmet
                           WHERE nastavnikDrugi.sifNastavnik =
                                 predmetGrupa.sifNastavnik
                             AND predmetGrupa.sifPredmet =
                                 upisanPredmet.sifPredmet
                             AND predmetGrupa.akGodina =
                                 upisanPredmet.akGodina
                             AND predmetGrupa.oznGrupa =
                                 upisanPredmet.oznGrupa
                             AND ocjena > 1
                             AND nastavnikDrugi.sifNastavnik <>
                                 nastavnik.sifNastavnik
                          GROUP BY nastavnikDrugi.sifNastavnik
                                 , imeNastavnik, prezimeNastavnik)
```

(Ne)potrebno korištenje podupita, presjek, razlika

1. Iako daje točne rezultate, ovo je primjer nepotrebnog korištenja podupita:

```
SELECT nastavnik.*
  FROM nastavnik, mjesto
 WHERE nastavnik.pbrStanNastavnik = mjesto.pbr
   AND sifZupanija IN (SELECT sifZupanija
                         FROM zupanija
                        WHERE LOWER(nazZupanija) LIKE '%zagreb%'
                       );
```

Ukoliko je moguće obaviti spajanje (prirodno ili s uvjetom) pri čemu se ne gubi na čitljivosti upita bolje je izbjeći uporabu podupita.
U skladu s tim, ispravno rješenje je:

```
SELECT nastavnik.*
  FROM nastavnik, mjesto, zupanija
 WHERE nastavnik.pbrStanNastavnik = mjesto.pbr
   AND mjesto.sifZupanija = zupanija.sifupanija
   AND LOWER(nazZupanija) LIKE '%zagreb%' ;
```

2. Ovo je primjer gdje je upit moguće napisati i bez podupita, ali se može argumentirati da je takav oblik upita manje čitljiv (razumljiv). Pogledajmo prvo najsloženiju verziju bez podupita u kojoj se organizacijska jedinica spaja i s nadređenom jedinicom (primijetite jedninu) i s podređenim jedinicama (primijetite množinu). Upravo zato jer postoji više podređenih org. jedinica potrebno je napraviti projekciju (DISTINCT) jer bi se u suprotnom organizacijska jedinica pojavila onoliko puta koliko ima podređenih org.jedinica:

```
SELECT DISTINCT OrgJed.*
  FROM OrgJed
  JOIN OrgJed NadOrgJed
    ON OrgJed.sifNadOrgJed = NadOrgJed.sifOrgJed
  JOIN OrgJed PodOrgJed
    ON OrgJed.sifOrgJed = PodOrgJed.sifNadOrgJed;
```

Budući da se u zadatku ne traže nikakvi podaci o nadređenoj organizacijskoj jedinici (već samo da ona postoji) prethodni upit moguće je pojednostavniti tako da se ne spaja s nadređenom već samo zahtijeva da je šifra nadređene org. jedinice poznata (`IS NOT NULL`):

```
SELECT DISTINCT OrgJed.*
  FROM OrgJed
  JOIN OrgJed PodOrgJed
    ON OrgJed.sifOrgJed = PodOrgJed.sifNadOrgJed
 WHERE OrgJed.sifNadOrgJed IS NOT NULL;
```

Konačno, upit se može napisati i korištenjem podupita:

```
SELECT OrgJed.*
  FROM OrgJed
 WHERE sifNadOrgJed IS NOT NULL
   AND sifOrgJed = ANY (SELECT sifNadOrgJed FROM OrgJed);

--ili: sifOrgJed IN (SELECT sifNadOrgJed FROM OrgJed);
```

Valja primijetiti jednu prednost verzije sa spajanjem: moguće je (osim organizacijske jedinice) ispisati podatke i o nadređenoj i/ili podređenoj organizacijskoj jedinici.
U verziji s podupitom to bi se moglo ostvariti dodavanjem podupita u SELECT listu ali to bi već bilo pretjerano i samim tim loše.

3.
a) stanuje barem jedan nastavnik

Upit se može napisati s ili bez podupita:

```
SELECT *
  FROM mjesto
 WHERE pbr IN (SELECT pbrStan FROM nastavnik);

SELECT DISTINCT mjesto.*
  FROM mjesto
  JOIN nastavnik
    ON mjesto.pbr = nastavnik.pbrStanNastavnik;
```

Primijetite da je u verziji bez podupita potrebno obaviti projekciju jer bi se u suprotnom naziv mjesta pojavio u rezultatu onoliko puta koliko ima nastavnika koji stanuju u tom mjestu.

b) ne stanuje niti jedan nastavnik
Ukoliko koristimo NOT IN operator i podupit situacija je drugačija – sljedeći upit je **pogrešan**:

```
SELECT *
  FROM mjesto
 WHERE pbr NOT IN (SELECT DISTINCT pbrStanNastavnik FROM nastavnik);
```

Ovo je dobar primjer kako treba **biti oprezan s podupitima i NULL vrijednostima**. Naime, podupit:
`(SELECT DISTINCT pbrStanNastavnik FROM nastavnik)`
će vratiti skup svih različitih poštanskih brojeva stanovanja nastavnika. Budući da neki nastavnici nemaju evidentiran poštanski broj stanovanja u tom skupu će se pojaviti i jedna NULL vrijednost, npr. :
`{NULL, 10000, 20000, 21265, ...}`

Budući da se izraz "pbr NOT IN {...}" ocjenjuje istinitim samo ako je trenuti pbr **različit od svih** u trenutnom skupu bez obzira koja ja vrijednost trenutnog poštanskog broja ovaj izraz će se ocijeniti kao false ili unknown jer:

```
 pbr <> NULL
```

je uvijek unknown.
Npr. izraz:

```
      23000 NOT IN {NULL, 10000, 22000}
```

nije istinit.
Dakle, treba popraviti upit tako da se u rezultatu podupita **ne pojavljuju** NULL vrijednosti:

```
SELECT *
  FROM mjesto
 WHERE pbr NOT IN (SELECT DISTINCT pbrStanNastavnik
                     FROM nastavnik
                    WHERE pbrStanNastavnik IS NOT NULL);
```

ili, još bolje, napisati upit koristeći spajanje:

```
SELECT mjesto.*
  FROM mjesto
  LEFT JOIN nastavnik
         ON mjesto.pbr = nastavnik.pbrStanNastavnik
 WHERE sifNastavnik IS NULL;
```

Primijetite da u ovom upitu mjesto vanjski spajamo s nastavnikom.
Zašto u ovom slučaju nema projekcije (DISTINCT) i da li bi to što promijenilo?
Što bi se dogodilo ako bi se uvjet "`sifNastavnik IS NULL`" stavio u uvjet spajanja:

```
...
LEFT JOIN nastavnik
         ON mjesto.pbr = nastavnik.pbrStanNastavnik
        AND sifNastavnik IS NULL;
```

a ne u where dio?

4.
Nepotrebno korištenje podupita u select listi:

```
SELECT jmbag, imeStudent, prezimeStudent,
      (SELECT avg(ocjena) FROM upisanPredmet
        WHERE jmbag = student.jmbag AND ocjena>1) AS PO
  FROM student
ORDER BY prezimeStudent;
```

Primjetiti da je podupit u SELECT listi koreliran s vanjskim upitom:

```
... WHERE jmbag = student.jmbag ...
```

Korektnije bi bilo napisati:

```
 ... WHERE upisanPredmet.jmbag = student.jmbag ...
```

iako je i izvorna verzija točna (pogledati u predavanjima pravila kako odrediti na koju se relaciju atribut odnosi).
Podupit u select listi je bolje izbjegavati kad nije potreban:

```
SELECT student.jmbag, imeStudent, prezimeStudent, AVG(ocjena) AS PO
  FROM student
  LEFT JOIN upisanPredmet
         ON student.jmbag = upisanPredmet.jmbag
        AND ocjena > 1
GROUP BY student.jmbag, imeStudent, prezimeStudent
ORDER BY prezimeStudent;
```

Što bi se dogodilo da u gornjem upitu nismo koristili vanjsko spajanje (odnosno da smo izostavili LEFT iz gornjeg upita)?

5.
a) Pogrešno rješenje:

```
SELECT *
  FROM r1
 WHERE ime  IN (SELECT ime  FROM r2)
   AND prez IN (SELECT prez FROM r2)
```

Također **pogrešno** rješenje:

```
SELECT *
  FROM r1
 WHERE ime  IN (SELECT ime  FROM r2)
    OR prez IN (SELECT prez FROM r2)
```

Ispravno rješenje:

```
SELECT *
  FROM r1
 WHERE EXISTS (
               SELECT *
                 FROM r2
                WHERE r2.ime = r1.ime
                  AND r2.prez = r1.prez
               )
```

b) Pogrešno rješenje:

```
SELECT *
  FROM r1
 WHERE ime  NOT IN (SELECT ime  FROM r2)
   AND prez NOT IN (SELECT prez FROM r2)
```

Također **pogrešno** rješenje:

```
SELECT *
  FROM r1
 WHERE ime  NOT IN (SELECT ime  FROM r2)
    OR prez NOT IN (SELECT prez FROM r2)
```

Ispravno rješenje:

```
SELECT *
  FROM r1
 WHERE NOT EXISTS (
                   SELECT *
                     FROM r2
                    WHERE r2.ime = r1.ime
                      AND r2.prez = r1.prez
                   )
```

6.
a)
Točno rješenje:

```
SELECT *
  FROM imaNadOrgJed
 WHERE EXISTS (SELECT *
         FROM imaPodOrgJed
        WHERE imaPodOrgJed.sifOrgJed = imaNadOrgJed.sifOrgJed
          AND imaPodOrgJed.nazOrgJed = imaNadOrgJed.nazOrgJed
          AND (   imaPodOrgJed.sifNadOrgJed = imaNadOrgJed.sifNadOrgJed
               OR imaPodOrgJed.sifNadOrgJed IS NULL
              AND imaNadOrgJed.sifNadOrgJed IS NULL
              )
    );
```

POGREŠNO rješenje (iako u ovom slučaju vraća iste zapise):

```
SELECT *
  FROM imaNadOrgJed
 WHERE sifOrgJed    IN (SELECT sifOrgJed FROM imaPodOrgJed)
   AND nazOrgJed    IN (SELECT nazOrgJed FROM imaPodOrgJed)
   AND sifNadOrgJed IN (SELECT sifNadOrgJed FROM imaPodOrgJed);
```

Naime, za presjek se mora koristiti EXISTS ukoliko relacija ima više od jednog atributa i **nije logički ispravno** (vidi prethodni zadatak) pokušati presjek izračunati na način se za svaki atribut pojedinačno pokušava odrediti da li se nalazi u drugoj relaciji (vidjeti definiciju **istih** n-torki). Dakle, pogrešne su ovakve konstrukcije kod računanja **presjeka** relacija r1 i r2:

```
...
  AND r1.atr1 IN (SELECT atr1 FROM r2)
  AND r1.atr2 IN (SELECT atr2 FROM r2)
  ...
  AND r1.atrN IN (SELECT atrN FROM r2)
...
```

b)
Točno rješenje:

```
SELECT *
  FROM imaNadOrgJed
 WHERE NOT EXISTS (SELECT *
          FROM imaPodOrgJed
         WHERE imaPodOrgJed.sifOrgJed = imaNadOrgJed.sifOrgJed
           AND imaPodOrgJed.nazOrgJed = imaNadOrgJed.nazOrgJed
           AND ( imaPodOrgJed.sifNadOrgJed = imaNadOrgJed.sifNadOrgJed
              OR imaPodOrgJed.sifNadOrgJed IS NULL
             AND imaNadOrgJed.sifNadOrgJed IS NULL
               )
      );
```

Pogrešno rješenje (objašnjenje analogno onom za presjek u a) dijelu zadataka):

```
SELECT *
  FROM imaNadOrgJed
 WHERE sifOrgJed    NOT IN (SELECT sifOrgJed FROM imaPodOrgJed)
   AND nazOrgJed    NOT IN (SELECT nazOrgJed FROM imaPodOrgJed)
   AND sifNadOrgJed NOT IN (SELECT sifNadOrgJed FROM imaPodOrgJed);
```

c)
Točno rješenje:

```
SELECT * FROM imaPodOrgJed
WHERE NOT EXISTS (SELECT *
      FROM imaNadOrgJed
     WHERE imaPodOrgJed.sifOrgJed = imaNadOrgJed.sifOrgJed
       AND imaPodOrgJed.nazOrgJed = imaNadOrgJed.nazOrgJed
       AND ( imaPodOrgJed.sifNadOrgJed = imaNadOrgJed.sifNadOrgJed
          OR imaPodOrgJed.sifNadOrgJed IS NULL
         AND imaNadOrgJed.sifNadOrgJed IS NULL
            )
   );
```

Pogrešno rješenje (objašnjenje analogno onom za presjek u a) dijelu zadataka):

```
SELECT *
  FROM imaPodOrgJed
 WHERE sifOrgJed     NOT IN (SELECT sifOrgJed FROM imaNadOrgJed)
   AND nazOrgJed     NOT IN (SELECT nazOrgJed FROM imaNadOrgJed)
   AND sifNadOrgJed  NOT IN (SELECT sifNadOrgJed FROM imaNadOrgJed);
```

INSERT, UPDATE i DELETE naredbe

1. Nova tablica treba imati strukturu:

ime	prezime	datumOcjena	ocjena

```
INSERT INTO polozenPredmet
SELECT imeStudent, prezimeStudent, datumOcjena, ocjena
  FROM upisanPredmet, student
 WHERE upisanPredmet.jmbag=student.jmbag
   AND upisanPredmet.sifPredmet=1
   AND upisanPredmet.ocjena>3
```

2. Obratiti pažnju na redosljed u listi atributa! Nova tablica ima strukturu:

sifpredmet	naziv	avg(ocjena)

```
INSERT INTO prosjecneOcjene (sifra, predmet, prosjecna)
SELECT upisanPredmet.sifPredmet
     , predmet.naziv
     , avg(ocjena)
  FROM upisanPredmet, predmet
 WHERE upisanPredmet.sifPredmet = predmet.sifPredmet
 GROUP BY upisanPredmet.sifPredmet, predmet.naziv
```

3. Obratiti pažnju na podupit. U podupitu se pronalazi jmbag za studente koje treba izbrisati iz baze.

```
DELETE FROM upisanPredmet
 WHERE jmbag NOT IN
    (SELECT jmbag FROM student
      WHERE pbrRodStudent = 10000
        AND pbrStanStudent <> 10000)
```

4. Nova tablica mora imati četiri atributa. Ime i prezime ne može biti null i zato se mora obavezno navesti u insert naredbi. Ocjenu pak ne moramo navoditi zbog ključne riječi DEFAULT 0. Za datum ocjene će se upisati null ako se ne navede u insert naredbi, jer nije postavljen NOT NULL kao za atribute ime i prezime.

```
CREATE TABLE polozenPredmet (
  ime     NCHAR(30) NOT NULL
, prez    NCHAR(50) NOT NULL
, ocjena  SMALLINT  DEFAULT 0
, datumOcjena DATE);
```

5. Koristi se ključna riječ SET za mijenjanje nekog numeričkog tipa podatka.

```
UPDATE student
   SET datrod = datrod + 1
 WHERE ime = 'Tomislav'
   AND pbrRodStudent <> 10000
   AND pbrStanStudent = 10000
```

6. Da bi riješili ovaj problem potrebno je napisati dva podupita. Prvo grupiramo po jmbag-u da bi izbrojali koliko ocjena je neki student sakupio i onda taj broj uspoređujemo sa svakom n-torkom.

```
DELETE FROM student
WHERE jmbag IN(SELECT jmbag
                 FROM upisanPredmet
                GROUP BY jmbag
               HAVING COUNT(ocjena)
                   >=ALL (SELECT COUNT(ocjena)
                            FROM upisanPredmet
                           GROUP BY jmbag))
```

7. Koristimo ključnu riječ SET da bi dodali nulu na šifru županije. Prije toga je bilo potrebno pronaći u kojem to mjestu stanuju profesori sa najvećim prosječnim koeficijentom (što je riješeno u dva podupita). Na kraju pronađeno mjesto treba povezati sa šifrom županije.

```
UPDATE mjesto SET sifZupanija=sifZupanija*10
WHERE pbr IN(SELECT pbrStanNastavnik FROM nastavnik
              GROUP BY pbrStanNastavnik
             HAVING AVG(koef)
                 >= ALL (SELECT AVG(koef)
                           FROM nastavnik
                          GROUP BY pbrStanNastavnik))
```

8. Novonastala tablica ima sljedeću strukturu:

oznakagrupe	maxkapacitet

Prvo se riješi podupit koji grupira po oznakama grupe i radi sumiranje kapaciteta, uz postavljanje uvjeta na kraju. Select mora odgovarati listi atributa ispod INSERT naredbe!

```
INSERT INTO maxgrupa (oznakagrupe, maxkapacitet)
SELECT ozngrupa, SUM(kapacitet)
  FROM grupa
 GROUP BY ozngrupa
HAVING SUM(kapacitet)>80
```

9. Rješenje se sastoji iz dva dijela, odnosno, koriste se when, then i else ključne riječi za svaki atribut kojeg želimo mijenjati. Potrebno je paziti na redosljed navođenja atributa!

```
UPDATE upisanpredmet
   SET ocjena = CASE
                WHEN ocjena<5 THEN ocjena + 1
                ELSE ocjena
                END
, sifpredmet = CASE
               WHEN sifpredmet<10 THEN sifpredmet*100
               ELSE sifpredmet*10
               END
```

10. Kao i u prethodnom zadatku rješenje se sastoji iz dva dijela, odnosno, koriste se when, then i else ključne riječi za svaki atribut kojeg želimo mijenjati. Za razliku od prethodnog zadatka ovdje je navedena lista atributa ispod UPDATE naredbe.

```
UPDATE upisanpredmet SET
(ocjena, sifpredmet)=
(CASE WHEN ocjena<5 THEN ocjena + 1
 ELSE ocjena
 END
, CASE WHEN sifpredmet<10 THEN sifpredmet*100
  ELSE sifpredmet*10
   END)
```