NAPREDNE BAZE PODATAKA

PRIRUČNIK

Sanja Kraljević
Jakob Gračanin

PRIRUČNICI TEHNIČKOG VELEUČILIŠTA U ZAGREBU

MANUALIA POLYTECHNICI STUDIORUM ZAGRABIENSIS

Sanja Kraljević, dipl. ing.
Jakob Gračanin, dipl. ing.

Napredne baze podataka

Priručnik za kolegij 'Napredne baze podataka' koji se održava prema nastavnom planu na preddiplomskim studijima informatike i računarstva Tehničkog veleučilišta u Zagrebu

Zagreb, 2023.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

TEHNIČKO VELEUČILIŠTE U ZAGREBU ZAGREB UNIVERSITY OF APPLIED SCIENCES

izdavač	Tehničko veleučilište u Zagrebu Vrbik 8, Zagreb
za izdavača	prof. dr. sc. Jana Žiljak Gršić
autori	Sanja Kraljević, dipl. ing. Jakob Gračanin, dipl. ing.
recenzenti	Prof. dr. sc. Željko Dobrović, prof. struč. stud., Vojno učilište Tomislav Gligora, struč. spec. ing. techn. inf., v. pred., Veleučilište Velika Gorica
vrsta djela	priručnik Objavljivanje je odobrilo Stručno vijeće Tehničkog veleučilišta u Zagrebu odlukom broj: 5820-7/23
ISBN	978-953-8444-08-1

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Sadržaj

1. TRANSAKCIJE 1
2. PROCEDURE 13
3. FUNKCIJE 23
4. KONTROLA TOKA U UPITIMA I POHRANJENIM ZADATCIMA 28
5. KURSORI I *HANDLERI* 41
6. CTE 56
7. POGLEDI 64
8. OKIDAČI 78
9. KORISNICI I DOZVOLE 94
10. ZAKLJUČAVANJE PODATAKA 102
11. MEHANIZMI POHRANE 109
12. MODELIRANJE 122

POPIS SLIKA 131

LITERATURA 132

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Predgovor

Ovaj priručnik je namijenjen studentima Tehničkog veleučilišta u Zagrebu i kao takav pokriva područje praktičnog dijela baza podataka koje se izvodi kroz kolegij „Napredne baze podataka“ na stručnim preddiplomskom studijima računarstva i informatike. Priručnik je nastao višegodišnjom izradom materijala za vježbe na kolegiju, a zajedno s udžbenikom „Baze podataka“ autora Kramberger T., Duk S. i Kovačević R. čini obaveznu literaturu na kolegiju.

Priručnik je kreiran tako da nudi veliku količinu različitih tipova zadataka odnosno stručnih problema čijim će se praktičnim rješavanjem studenti pripremiti za komponentu laboratorijske vježbe na kolegiju, ali i razviti inženjerski pristup rješavanju problema s kojim će se susresti na radnom mjestu. Zadatci prate stvarne situacije s kojima se susreću razvojni inženjeri i na taj način priprema studente za posao u struci i znanosti. Praktični dijelovi popraćeni su tablicama kako bi se što jednostavnije shvatila materija koju je potrebno izučiti.

Osim za studente Tehničkog veleučilišta u Zagrebu, priručnik je idealan i za čitatelje koji su upoznati s osnovama SQL jezika a koji se žele dodatno profilirati u praktičnom radu s relacijskim bazama podataka, s naglaskom na primjenu SQL jezika. Sustav za upravljanje baza podataka na kojem su napravljeni svi zadatci i na kojem se priručnik zasniva je MySQL koji se odlikuje svojom otvorenošću koda, jednostavnošću instalacije, korištenja i brojnim dodatnim alatima koji su razvijeni od strane zajednice otvorenog koda. Odlikuje ga i široka zajednica korisnika što pojednostavnjuje pretragu interneta u slučaju složenijih problema.

Želimo zahvaliti svima koji su sudjelovali u izradi ovog priručnika, kao i studentima koji su pomogli tražiti pogreške te savjetima poboljšati kolegij, a samim time i priručnik.

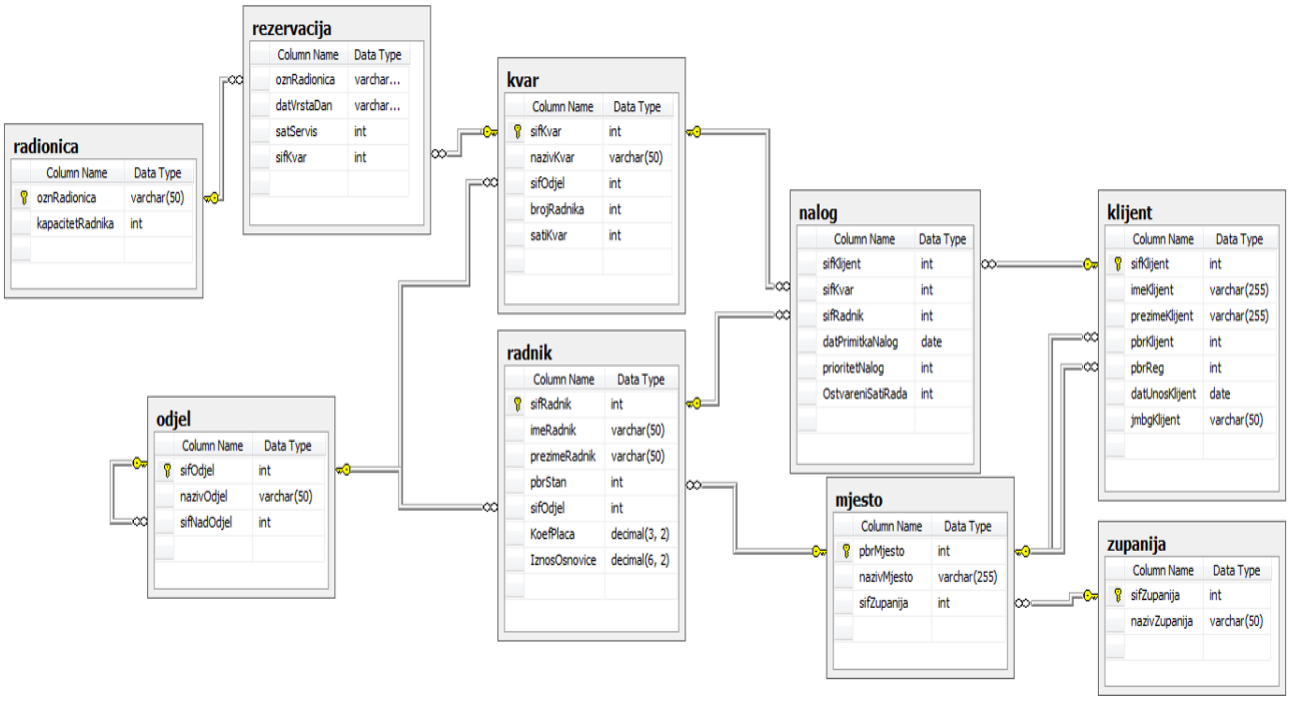
Autori, 2023

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Uvodne napomene

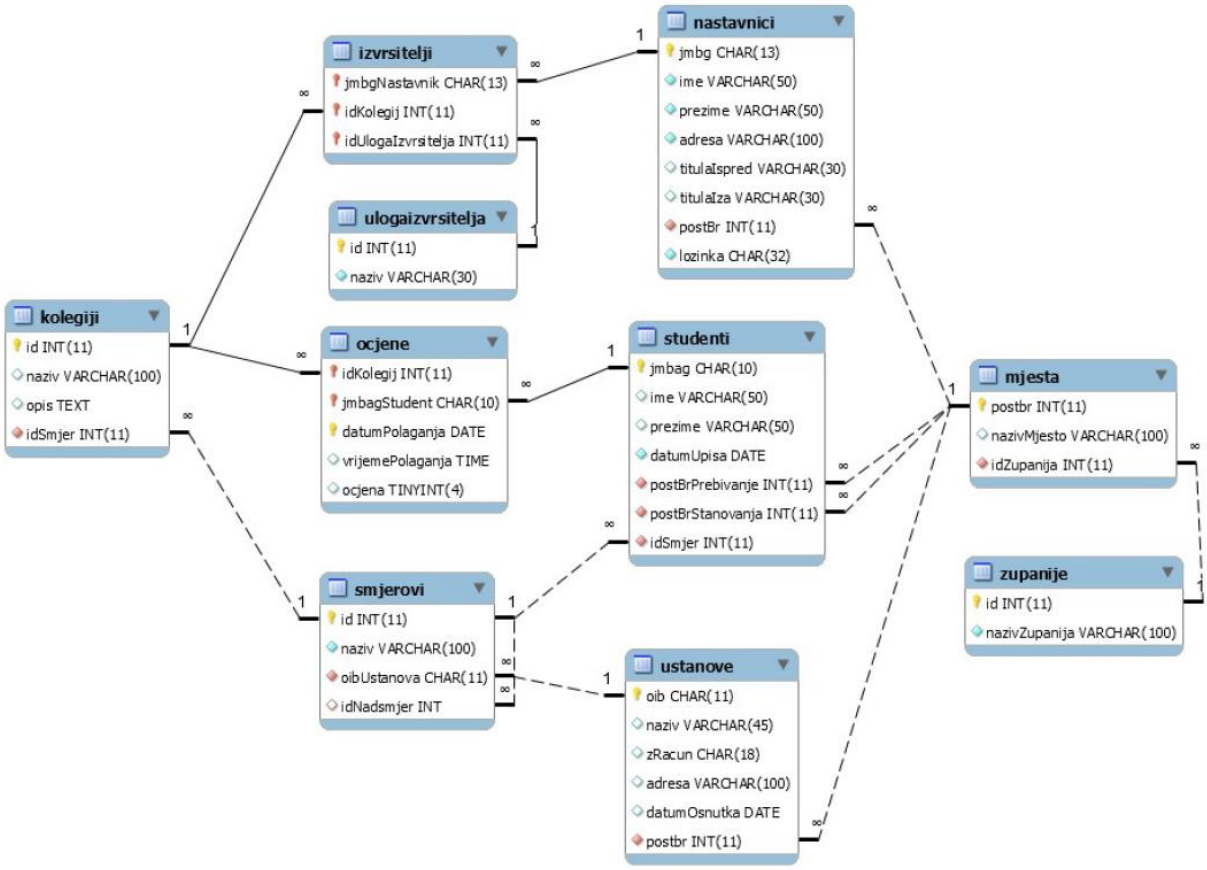
Rješavanjem zadataka iz ovog priručnika studenti se pripremaju za provedbu laboratorijskih vježbi iz kolegija. Zadatci se oslanjaju na teorijsku podlogu obrađenu u udžbeniku „Baze podataka“, a zadani su nad dvjema relacijskim bazama podataka, autoradionica i studenti čiji su shematski prikazi u nastavku ovog poglavlja.

U trenutku pisanja ovog priručnika, baze su smještene na posljednjoj verziji MySQL sustava za upravljanje bazama podataka, a to je MySQL 8.0 (točnije 8.0.32) a pristupa im se korištenjem alata SQLyog i MySQL Workbench. Korisnici ovog priručnika trebaju biti svjesni da se MySQL konstantno razvija a samim time da se unaprjeđuje i mijenja sama SQL sintaksa, posebice DDL i DCL naredbi. Osim sintakse pojedinih naredbi, u odnosu na verzije prije v8.0, a posebice u odnosu na v5.6 promijenjeni su pretpostavljeni mehanizmi pohrane, korišteni algoritmi za sažimanje podataka, strukture tablica i tipovi podataka u metabazama podataka *information_schema*, *performance_schema*, *mysql*, nazivi globalnih varijabli i slično. Stoga se očekuje da će pokretanje ponuđenih rješenja zadataka na istovjetnim skupovima podataka (struktura i podatci u bazama podataka) i istovjetnim verzijama MySQL rezultirati istovjetnim rezultatima koji su predočeni u ovom priručniku, odnosno da će na drugim verzijama MySQL-a dobiveni rezultati minimalno odstupati ili će pak biti potrebno prilagoditi sintaksu samih upita kako bi se zadatak uspješno riješio.



Slika 1. Shema baze autoradionica

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.



Slika 2. Shema baze studenti

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

1. Transakcije

1.1. U bazi autoradionica: Napisati transakciju koja će sve rezervacije koje imaju vrijednost atributa satServis postavljen na vrijednost 1, dodijeliti radionici „R26“. Nakon toga je potrebno radionici „R26“ povećati broj radnika za 1.

Na kraju transakcije, potrebno je potvrditi da su se promjene u bazi dogodile (COMMIT).

Napisati upit/e koje je potrebno pozvati prije i nakon transakcije kako bi se provjerila uspješnost napisanog koda.

```
/* provjera podataka koje će transakcija ažurirati */
SELECT * FROM rezervacija WHERE satServis = 1;
SELECT * FROM radionica WHERE oznRadionica = 'R26';

SET AUTOCOMMIT = 0;
START TRANSACTION;
  UPDATE rezervacija SET oznRadionica = 'R26'
      WHERE satServis = 1;
  UPDATE radionica SET kapacitetRadnika = kapacitetRadnika + 1
      WHERE oznRadionica = 'R26';
COMMIT;
SET AUTOCOMMIT = 1;

/* provjera podataka koje je transakcija ažurirala */
SELECT * FROM rezervacija WHERE satServis = 1;
SELECT * FROM radionica WHERE oznRadionica = 'R26';
```

1.2. U bazi autoradionica: Napisati transakciju koja će svim radnicima s plaćom manjom od 1000 povećati osnovicu za 200 a radnicima s plaćom većom od 5000 osnovicu smanjiti za 100.

Potrebno je potvrditi da se prva promjena dogodila a druga promjena (smanjenje) nije dogodila.

Napisati upit/e koje je potrebno pozvati prije i nakon transakcije kako bi se provjerila uspješnost napisanog koda.

Napomena: iznos plaće potrebno je računati kao umnožak iznosa osnovice i koeficijenta plaće.

```
/* provjera podataka koje će transakcija ažurirati */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT sifRadnik, IznosOsnovice, KoefPlaca, IznosOsnovice * KoefPlaca AS PLACA
  FROM radnik ORDER BY 4;

SET AUTOCOMMIT = 0;
START TRANSACTION;
  UPDATE radnik
  SET IznosOsnovice = IznosOsnovice + 200
  WHERE  IznosOsnovice * KoefPlaca < 1000;
SAVEPOINT tocka;
  UPDATE radnik
  SET IznosOsnovice = IznosOsnovice - 100
  WHERE  IznosOsnovice * KoefPlaca > 5000;
ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;

/* provjera podataka koje je transakcija ažurirala */
SELECT sifRadnik, IznosOsnovice, KoefPlaca, IznosOsnovice * KoefPlaca AS PLACA
  FROM radnik ORDER BY 4;
```

1.3. U bazi autoradionica: Napisati transakciju koja će svim rezervacijama sa šifrom kvara 33 a koje nisu predviđene za petak, umanjiti sat servisa za 1 (uz provjeru da krajnji sat servisa ne smije biti manji od 1). Nakon toga će sve druge rezervacije za petak čija je šifra kvara različita od 33 obrisati.

Potrebno je potvrditi da se prva promjena dogodila a druga promjena (brisanje) nije dogodila.

Napisati upit/e koje je potrebno pozvati prije i nakon transakcije kako bi se provjerila uspješnost napisanog koda.

```
/* provjera podataka koje će transakcija ažurirati */
SELECT * FROM rezervacija
  WHERE sifKvar = '33' AND datVrstaDan <> 'PE' AND satServis > 1;
SELECT * FROM rezervacija
  WHERE sifKvar <> '33' AND datVrstaDan = 'PE';

SET AUTOCOMMIT = 0;
START TRANSACTION;
  UPDATE rezervacija SET satServis = satServis - 1
    WHERE sifKvar ='33' AND datVrstaDan <> 'PE' AND satServis > 1 ;
SAVEPOINT tocka;
  DELETE FROM rezervacija
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    WHERE sifKvar <> '33' AND datVrstaDan ='PE';
ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;

/* provjera podataka koje transakcija jest / nije ažurirala */
SELECT * FROM rezervacija
  WHERE sifKvar = '33' AND datVrstaDan <> 'PE' AND satServis > 1;
SELECT * FROM rezervacija
  WHERE sifKvar <> '33' AND datVrstaDan = 'PE';
```

1.4. U bazi studenti: Napisati transakciju koja će sve ispite iz kolegija čiji je ID 20, povećati za jednu ocjenu (osim ocjene 1 i 5) ako pripadaju smjeru čiji je ID 10. Isto tako smanjiti sve ocjene za 1 ako iz kolegija 30 i smjera 15 (osim ocjena 1 i 2). Potrebno je potvrditi da se promjene u bazi nisu dogodile.

```
/* provjera podataka koje će transakcija ažurirati */
SELECT o.*, k.*
  FROM ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
  WHERE o.idKolegij = 20 AND k.idSmjer = 10;
SELECT o.*, k.*
  FROM ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
  WHERE o.idKolegij = 30 AND k.idSmjer = 15;

SET AUTOCOMMIT = 0;
START TRANSACTION;
  UPDATE ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
    SET o.ocjena=o.ocjena + 1
    WHERE o.idKolegij = 20 AND k.idSmjer = 10 AND o.ocjena BETWEEN 2 AND 4;
  UPDATE ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
    SET o.ocjena=o.ocjena - 1
    WHERE o.idKolegij = 30 AND k.idSmjer = 15 AND o.ocjena NOT IN (1, 2);
  ROLLBACK;
SET AUTOCOMMIT = 1;

/* provjera podataka koje transakcija (ni)je ažurirala */
SELECT o.*, k.*
  FROM ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
  WHERE o.idKolegij = 20 AND k.idSmjer = 10;
SELECT o.*, k.*
  FROM ocjene o INNER JOIN kolegiji k ON o.idKolegij= k.ID
  WHERE o.idKolegij = 30 AND k.idSmjer = 15;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

1.5. U bazi autoradionica: Napisati transakciju koja će radniku koji je napravio najviše naloga povećati koeficijent plaće za 0.5 a radniku koji je napravio najmanje smanjiti koeficijent plaće za 0.2.

Ako više radnika ima isti (najveći) broj naloga, drugi kriterij za povećanje koeficijenta plaće je ukupna suma sati.

Potrebno je potvrditi da se prva promjena dogodila a druga promjena (smanjenje) nije dogodila.

Provjere radi, potrebno je ispisati šifre radnika, broj naloga, sumu sati i koeficijent plaće kako bi se pronašao radnik kojem je potrebno izvršiti povećanje odnosno smanjenje koeficijenta te koliki je dotični iznos prije i poslije izvođenja transakcije.

```
SELECT sifRadnik, COUNT(*) AS br_nal, SUM(OstvareniSatiRada) AS sum_sati,
KoefPlaca
  FROM nalog NATURAL JOIN radnik
  GROUP BY sifRadnik, KoefPlaca
  ORDER BY 2 DESC, 3 DESC;
/* detektirani su radnici sa šifrom 289 (za povećanje koeficijenta) i 150 (za
smanjenje koeficijenta) */
```

sifRadnik	br_nal	sum_sati	KoefPlaca
289	27	155	2.07
514	27	146	0.66
501	26	135	1.32
244	22	117	1.96
379	22	112	0.86
142	21	124	1.92
166	20	124	0.63

...

505	1	10	1.38
281	1	6	2.58
150	1	4	1.84

```
SET AUTOCOMMIT = 0;
START TRANSACTION;
  UPDATE radnik
    SET KoefPlaca=KoefPlaca + 0.5
    WHERE sifRadnik = (SELECT sifRadnik FROM nalog
                       GROUP BY sifRadnik
                       ORDER BY COUNT(*) DESC, SUM(OstvareniSatiRada) DESC
                       LIMIT 0, 1);
SAVEPOINT tocka;
  UPDATE radnik
    SET KoefPlaca=KoefPlaca - 0.2
    WHERE sifRadnik = (SELECT sifRadnik FROM nalog
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
                        GROUP BY sifRadnik
                        ORDER BY COUNT(*), SUM(OstvareniSatiRada) ASC
                        LIMIT 0, 1);
ROLLBACK TO SAVEPOINT tocka;

SET AUTOCOMMIT = 1;

SELECT sifRadnik, COUNT(*) AS br_nal, SUM(OstvareniSatiRada) AS sum_sati,
KoefPlaca
  FROM nalog NATURAL JOIN radnik
  GROUP BY sifRadnik, KoefPlaca
  ORDER BY 2 DESC, 3 DESC;
```

sifRadnik	br_nal	sum_sati	KoefPlaca
289	27	155	2.57
514	27	146	0.66
501	26	135	1.32

...

505	1	10	1.38
281	1	6	2.58
150	1	4	1.84

1.6. U bazi autoradionica: Napisati transakciju koja će svim onim radnicima koji su radili po nalozima koji su zaprimljeni petkom i imali su prosjek ostvarenih sati rada veći od svih naloga koji su zaprimljeni petkom, povećati osnovicu za 30, a radnicima koji nisu radili po nalozima zaprimljenima petkom i pri tome su imali prosjek ostvarenih sati rada manji ili jednak onima koji isto nisu zaprimljeni petkom, osnovicu smanjiti za 10.

Potrebno je potvrditi da se prva promjena dogodila a druga promjena (smanjenje) nije dogodila.

```
/* rješenje pomoću ugniježđenih podupita */
SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE radnik
    SET IznosOsnovice = IznosOsnovice + 30
    WHERE sifRadnik IN
      (SELECT sifRadnik FROM nalog
        WHERE DAYNAME(datPrimitkaNalog) = 'Friday'
        GROUP BY sifRadnik
        HAVING AVG(OstvareniSatiRada) >
          (SELECT AVG(OstvareniSatiRada) FROM nalog
           WHERE DAYNAME(datPrimitkaNalog) = 'Friday'));
SAVEPOINT tocka;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  UPDATE radnik
    SET IznosOsnovice = IznosOsnovice - 10
    WHERE sifRadnik IN
      (SELECT sifRadnik FROM nalog
        WHERE DAYNAME(datPrimitkaNalog) <> 'Friday'
        GROUP BY sifRadnik
        HAVING AVG(OstvareniSatiRada) <=
          (SELECT AVG(OstvareniSatiRada) FROM nalog
            WHERE DAYNAME(datPrimitkaNalog) = 'Friday'));
ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;
```

```
/* rješenje pomoću globalne varijable */
SELECT AVG(OstvareniSatiRada) INTO @prosjekPetkom FROM nalog
  WHERE DAYNAME(datPrimitkaNalog) = 'Friday';
SELECT @prosjekPetkom;

SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE radnik
    SET IznosOsnovice = IznosOsnovice + 30
    WHERE sifRadnik IN
      (SELECT sifRadnik FROM nalog
        WHERE DAYNAME(datPrimitkaNalog) = 'Friday'
        GROUP BY sifRadnik
        HAVING AVG(OstvareniSatiRada) > @prosjekPetkom);
SAVEPOINT tocka;
  UPDATE radnik
    SET IznosOsnovice = IznosOsnovice - 10
    WHERE sifRadnik IN
      (SELECT sifRadnik FROM nalog
        WHERE DAYNAME(datPrimitkaNalog) <> 'Friday'
        GROUP BY sifRadnik
        HAVING AVG(OstvareniSatiRada) <= @prosjekPetkom);
ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

1.7. U bazi autoradionica: Napisati transakciju koja će svim klijentima iza prezimena dodati oznaku ' - VIP' ako su do sada (po nalozima) imali više od 6 popravaka visokog prioriteta (prioriteta 1 ili 2). Ako su imali manje od 3 popravka (prioriteta 1 ili 2) dobivaju oznaku ' - REGULAR'.

Potrebno je potvrditi da se prva promjena dogodila a druga promjena nije dogodila.

Napisati upit/e koje je potrebno pozvati prije i nakon transakcije kako bi se provjerila uspješnost napisanog koda.

```
SELECT nalog.sifKlijent, imeKlijent, prezimeKlijent, COUNT(*)
  FROM nalog JOIN klijent ON nalog.sifKlijent = klijent.sifKlijent
  WHERE prioritetNalog IN (1,2)
  GROUP BY sifKlijent
  HAVING COUNT(*) > 6 OR COUNT(*) <3
  ORDER BY 4 DESC;

SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE klijent
  SET prezimeKlijent = CONCAT(prezimeKlijent, ' - VIP')
  WHERE sifKlijent IN
    (SELECT sifKlijent FROM nalog
      WHERE prioritetNalog IN (1,2)
      GROUP BY sifKlijent
      HAVING COUNT(*) > 6);
SAVEPOINT tocka;
  UPDATE klijent
    SET prezimeKlijent = CONCAT(prezimeKlijent, ' - REGULAR')
    WHERE sifKlijent IN
      (SELECT sifKlijent FROM nalog
        WHERE prioritetNalog IN (1,2)
        GROUP BY sifKlijent
        HAVING COUNT(*) <3) ;
ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;

SELECT nalog.sifKlijent, imeKlijent, prezimeKlijent, COUNT(*)
  FROM nalog JOIN klijent ON nalog.sifKlijent = klijent.sifKlijent
  WHERE prioritetNalog IN (1,2)
  GROUP BY sifKlijent
  HAVING COUNT(*) > 6 OR COUNT(*) <3
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  ORDER BY 4 DESC;
```

1.8. U bazi studenti: Napisati transakciju koja će svim studentima iza prezimena dodati oznaku ' - iznad' i prezime postaviti na velika slova ako im je prosjek ocjena položenih ispita veći ili jednak od prosječne ocjene svih studenata i ako su položili više od 1 ispita.

U suprotnom slučaju prezimenu dodati oznaku ' - ispod' i prezime postaviti na mala slova.

Na kraju transakcije potrebno je opovrgnuti sve promjene. Provjere radi, dok je transakcija još u tijeku, a prije opovrgavanja promjena, provjeriti stanje u bazi za dotične n-torke (studente).

Napisati upit/e koje je potrebno pozvati prije i nakon transakcije kako bi se provjerila uspješnost napisanog koda.

```
SELECT jmbagStudent, ime, prezime
  FROM ocjene JOIN studenti ON ocjene.jmbagStudent = studenti.jmbag
  WHERE ocjena > 1
  GROUP BY jmbagStudent
  HAVING COUNT(*) > 1;


SET AUTOCOMMIT = 0;
BEGIN;
  SELECT AVG(ocjena) INTO @prosjecnaOcjena FROM ocjene;
  SELECT @prosjecnaOcjena; /* U ovom rješenju korištena je globalna varijabla
kako bi se izbjeglo repetativno pisanje istog podupita. Globalne varijable
vidljive su unutar sesije. */
  UPDATE studenti
    SET prezime = UPPER(CONCAT(prezime, ' - iznad'))
    WHERE jmbag IN
      (SELECT jmbagStudent
        FROM ocjene
        WHERE ocjena > 1
        GROUP BY jmbagStudent
        HAVING COUNT(*) > 1 AND AVG(ocjena) >= @prosjecnaOcjena);
  UPDATE studenti
  SET prezime = LOWER(CONCAT(prezime, ' - ispod'))
  WHERE jmbag IN
    (SELECT jmbagStudent
      FROM ocjene
      WHERE ocjena > 1
      GROUP BY jmbagStudent
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      HAVING COUNT(*) > 1 AND AVG(ocjena) < @prosjecnaOcjena);

  SELECT jmbagStudent, ime, prezime
    FROM ocjene JOIN studenti ON ocjene.jmbagStudent = studenti.jmbag
    WHERE ocjena > 1
    GROUP BY jmbagStudent
    HAVING COUNT(*) > 1;

ROLLBACK ;
SET AUTOCOMMIT = 1;

SELECT jmbagStudent, ime, prezime
  FROM ocjene JOIN studenti ON ocjene.jmbagStudent = studenti.jmbag
  WHERE ocjena > 1
  GROUP BY jmbagStudent
  HAVING COUNT(*) > 1;
```

1.9. U bazi autoradionica: Napisati transakciju koja će svim klijentima iza prezimena dodati oznaku ' - VIP' i ime postaviti na velika slova ako su do sada (po nalozima) imali više od 100 sati rada.

Ako su imali manje ili jednako 100 sati rada dobivaju samo oznaku ' -REGULAR' iza prezimena.

Potrebno je odbaciti obje promjene.

```
SET AUTOCOMMIT = 0;
BEGIN;
UPDATE KLIJENT
  SET prezimeKlijent = CONCAT(UPPER(prezimeKlijent), '- VIP')
  WHERE sifKlijent IN
    (SELECT sifKlijent FROM nalog
    GROUP BY sifKlijent
    HAVING SUM(OstvareniSatiRada) > 100);
UPDATE KLIJENT
  SET prezimeKlijent = CONCAT(prezimeKlijent, '- REGULAR')
  WHERE sifKlijent IN
    (SELECT sifKlijent FROM nalog
    GROUP BY sifKlijent
    HAVING SUM(OstvareniSatiRada) > 100);
ROLLBACK;
SET AUTOCOMMIT = 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

1.10. U bazi studenti: Napisati transakciju koja će svim studentima iza prezimena dodati oznaku ' - iznad_5' i prezime postaviti na velika slova ako im je prosjek položenih ispita veći ili jednak od prosječnog i ako su imali barem jednu odličnu ocjenu.

U slučaju da su ispod prosjeka i da imaju više od jedne negativne ocjene, prezimenu dodati oznaku ' - ispod_1' i prezime postaviti na mala slova.

Potrebno je opovrgnuti obje promjene

```
SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE studenti
    SET prezime = UPPER(CONCAT(prezime, '- iznad_5'))
    WHERE jmbag IN
      (SELECT jmbagStudent
        FROM ocjene
        WHERE ocjena > 1
        GROUP BY jmbagStudent
        HAVING AVG(ocjena) > (SELECT AVG(ocjena) FROM ocjene) AND MAX(ocjena) =
5);
  UPDATE studenti
    SET prezime = LOWER(CONCAT(prezime, '- ispod_1'))
    WHERE jmbag IN
      (SELECT jmbagStudent
        FROM ocjene
        GROUP BY jmbagStudent
        HAVING AVG(ocjena) < (SELECT AVG(ocjena) FROM ocjene) )
    AND jmbag IN
      (SELECT jmbagStudent FROM
        ocjene WHERE ocjena=1
        GROUP BY jmbagStudent
        HAVING COUNT(*) > 1);
ROLLBACK;
SET AUTOCOMMIT = 1;
```

1.11. U bazi autoradionica: Napisati transakciju koja će svim klijentima iza prezimena dodati oznaku ' - VIP' i ime postaviti na velika slova ako su do sada (po nalozima) imali između 2 i 4 popravka (prioriteta 1 ili 2) u 2019 i 2020 godini.

Ako su imali manje od 3 popravka u 2020 i da nisu bili prioriteta 1 ili 2 dobivaju oznaku ' - REGULAR'.

Potrebno je potvrditi da se druga promjena dogodila a prva promjena nije dogodila.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE klijent
  SET prezimeKlijent = CONCAT(prezimeKlijent, ' - REGULAR')
   WHERE sifKlijent IN
   (SELECT sifKlijent FROM nalog
    WHERE prioritetNalog NOT IN (1,2) and YEAR(datPrimitkaNalog) = 2020
    GROUP BY sifKlijent
    HAVING COUNT(*) < 3);
SAVEPOINT tocka;
  UPDATE klijent
    SET prezimeKlijent = UPPER(CONCAT(prezimeKlijent, ' - VIP'))
    WHERE sifKlijent IN
      (SELECT sifKlijent FROM nalog
        WHERE prioritetNalog IN (1,2) AND YEAR(datPrimitkaNalog) IN (2019, 2020)
        GROUP BY sifKlijent
        HAVING COUNT(*) BETWEEN 2 AND 4);
ROLLBACK TO SAVEPOINT tocka;
COMMIT;

SET AUTOCOMMIT = 1;
```

1.12. U bazi studenti: Napisati transakciju koja će svim kolegijima kojeg je (pozitivno) položio barem jedan student, iza naziva kolegija dodati oznaku ' - AVG > 3.5' ako su imali prosjek veći od 3.5.

Ako je kolegij (pozitivno) položio barem jedan student, prosjek ocjena na kolegiju je manji ili jednak 3.5, potrebno je dodati oznaku ' - AVG <= 3.5'.

Potrebno je potvrditi samo promjenu dodavanja oznake ' - AVG <= 3.5'.

```
SET AUTOCOMMIT = 0;
BEGIN;
  UPDATE kolegiji
    SET kolegiji.naziv = CONCAT(kolegiji.naziv, "-AVG <=3.5")
    WHERE kolegiji.id IN
      (SELECT idKolegij
        FROM ocjene
        GROUP BY idKolegij
        HAVING AVG(ocjena) <= 3.5)
    AND kolegiji.id IN
      (SELECT idKolegij
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
        FROM ocjene
        WHERE ocjena > 1
        GROUP BY idKolegij
        HAVING COUNT(*) > 1);
SAVEPOINT tocka;
  UPDATE kolegiji
    SET kolegiji.naziv = CONCAT(kolegiji.naziv, "-AVG>3.5")
    WHERE kolegiji.id IN
      (SELECT idKolegij
        FROM ocjene
        GROUP BY idKolegij
        HAVING AVG(ocjena) > 3.5)
    AND kolegiji.id IN
      (SELECT idKolegij
        FROM ocjene
        WHERE ocjena > 1
        GROUP BY idKolegij
        HAVING COUNT(*) > 1);

ROLLBACK TO SAVEPOINT tocka;
COMMIT;
SET AUTOCOMMIT = 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

2. Procedure

2.1. U bazi studenti: Napisati proceduru koja će za zadanu šifru klijenta putem izlaznih parametara vratiti:

- jmbg klijenta,
- broj naloga koji se na njega odnose i
- sumu ostvarenih sati rada po nalozima za tog klijenta.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

Primjer rezultata nakon poziva procedure je u nastavku.

@jmbag		@nalozi	@sati
1110985310615	13B	2	2

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiNalogeSate //
CREATE PROCEDURE vratiNalogeSate(zadaniKlijent INT, OUT jmbgKlij VARCHAR(50),
OUT brNaloga INT, OUT brSati INT)
BEGIN
  SELECT  k.jmbgKlijent, COUNT(*), SUM(n.OstvareniSatiRada)
    INTO jmbgKlij, brNaloga, brSati
    FROM nalog n JOIN klijent k ON n.sifKlijent = k.sifKlijent
    WHERE n.sifKlijent = zadaniKlijent
    GROUP BY n.sifKlijent, k.jmbgKlijent ;
END//
DELIMITER ;
```

```
/* Pozivi procedure */
CALL vratiNalogeSate(1167, @jmbag, @nalozi, @sati);
SELECT @jmbag, @nalozi, @sati;
```

@jmbag		@nalozi	@sati
1110985310615	13B	2	2

```
CALL vratiNalogeSate(1160, @jmbag, @nalozi, @sati);
SELECT @jmbag, @nalozi, @sati;
```

@jmbag		@nalozi	@sati
0907989303508	13B	1	6

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

2.2. U bazi autoradionica: Napisati proceduru koja će za zadanog klijenta vratiti podatke o klijentu, i to:

- jmbg klijenta,
- ime i prezime klijenta (spojeno),
- godinu unosa klijenta u sustav,
- broj naloga koji se na njega odnose i
- sumu ostvarenih sati rada po nalozima za tog klijenta.

Prikazati i sveukupan broj sati za sve klijente preko izlaznog parametra.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiNalogeSate //
CREATE PROCEDURE vratiNalogeSate(zadaniKlijent INT, OUT Total_sati INT)
BEGIN
  SELECT k.jmbgKlijent Jmbg_klijent,
    CONCAT(k.imeKlijent, ' ', k.prezimeKlijent) Ime_prezime,
    YEAR(datUnosKlijent) God_unosa, COUNT(*) Br_naloga,
    SUM(OstvareniSatiRada) Sati_rada
    FROM nalog n NATURAL JOIN klijent k
    WHERE k.sifKlijent = zadaniKlijent;
  -- izračun ukupnog broja sati i pohrana u izlazni parametar
  SELECT SUM(OstvareniSatiRada) INTO Total_sati
    FROM nalog;
END //
DELIMITER ;

/* Poziv procedure i ispis rezultata u izlaznoj varijabli */
CALL vratiNalogeSate(1167, @tot_sati);
```

Jmbg_klijent	Ime_prezime	God_unosa	Br_naloga	Sati_rada
1110985310615	Mirko Teklić	2016	2	2

```
SELECT @tot_sati;
```

@tot_sati
2324

2.3. U bazi autoradionica: Napisati proceduru koja će za zadanu oznaku radionice vratiti:

- oznaku radionice,
- šifru kvarova koji se u njoj popravljaju,

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- naziv kvarova koji se u njoj popravljaju,
- broj radnika i
- broj sati kvara predviđenih za kvarove zadane radionice.

Preko izlaznog parametra treba vratiti ukupan predviđen kapacitet radnika za zadanu radionicu.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiPodatkeRadionice //
CREATE PROCEDURE vratiPodatkeRadionice (zadana_radionica VARCHAR(5), OUT
tot_kapacitet INT )
BEGIN
  SELECT re.oznRadionica, kv.sifKvar, kv.nazivKvar, kv.brojRadnika, kv.satiKvar
    FROM rezervacija re NATURAL JOIN kvar kv
    WHERE re.oznRadionica = zadana_radionica;
  -- izračun ukupnog kapaciteta sati radnika
  SELECT SUM(kapacitetRadnika) INTO tot_kapacitet
    FROM radionica
    WHERE oznRadionica = zadana_radionica;
END//
DELIMITER ;

/* Poziv procedure i ispis rezultata iz izlaznog parametra */
CALL vratiPodatkeRadionice('R10', @tot_kapacitet);
```

oznRadionica	sifKvar	nazivKvar	brojRadnika	satiKvar
R10	11	Zamjena pragova	8	4
R10	15	Popravak oštećenih mjesta na limu	6	4
R10	27	Zamjena blatobrana	7	2
R10	3	Zamjena cerade	3	3
R10	35	Zamjena prednje žarulje	4	4

```
SELECT @tot_kapacitet;
```

@tot_kapacitet
2

2.4. U bazi autoradionica: Napisati proceduru koja će za zadanu šifru odjela i mjesec kojeg je potrebno izostaviti iz dohvata (ispisa) podataka vratiti

- šifru radnika,
- ime i prezime radnika spojeno,
- broj odrađenih naloga,
- prosječne sate rada,

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- koeficijent plaće,
- iznos osnovice te
- plaću na dvije decimale (plaća je umnožak osnovice i koeficijent plaće).

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiRadnikeOdjela //
CREATE PROCEDURE vratiRadnikeOdjela(sif_Odjela INT, bez_mjeseca INT)
BEGIN
  SELECT agr.sifRadnik Sif, CONCAT(r.imeRadnik, ' ', r.prezimeRadnik) AS
  Ime_prezime, agr.Br_naloga, agr.Pros_sati_rada, r.KoefPlaca AS Koeficijent,
r.IznosOsnovice AS Osnovica,
  ROUND(r.KoefPlaca * r.IznosOsnovice,2) AS Placa
    FROM
      (SELECT sifRadnik, COUNT(*) Br_naloga, avg(OstvareniSatiRada) AS
        Pros_sati_rada
        FROM nalog
        WHERE month(datPrimitkaNalog) <> bez_mjeseca
        GROUP BY sifRadnik
        HAVING COUNT (*) > 5) agr
    JOIN radnik r on agr.sifRadnik = r.sifRadnik
    WHERE r.sifOdjel = sif_Odjela;
END //
DELIMITER ;

/*Poziv procedure*/
CALL vratiRadnikeOdjela(2,7);
```

Sif	Ime_prezime	Br_naloga	Pros_sati_rada	Koeficijent	Osnovica	Placa
142	Alan Ostojčić	21	5.9048	1.92	2200.00	4224.00
330	Vinko Kecman	12	5.4167	1.79	2200.00	3938.00
391	Marko Hazdovac	10	4.2000	2.35	2200.00	5170.00
469	Ivana Dumančić	18	5.6667	2.35	2200.00	5170.00

2.5. U bazi studenti: Napisati proceduru koja će za zadani raspon prosječnih ocjena (prosjecna_ocjena_od, prosjecna_ocjena_do) prikazati po kolegijima sljedeće podatke:

- naziv kolegija,
- id smjera kojem kolegij pripada,
- id kolegija,
- broj položenih ispita i

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- prosječnu ocjenu kolegija.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiKolegije //
CREATE PROCEDURE vratiKolegije (prosjecna_ocjena_od DOUBLE, prosjecna_ocjena_do
DOUBLE)
BEGIN
  SELECT k.naziv, k.idSmjer, agr.*
  FROM
    (SELECT idKolegij , COUNT(*) Br_polozenih, AVG(ocjena) Pros_ocjena_kol
      FROM ocjene
      GROUP BY idKolegij
      HAVING AVG(ocjena) BETWEEN prosjecna_ocjena_od AND prosjecna_ocjena_do)
        agr
  JOIN kolegiji k ON agr.idKolegij = k.id
  ORDER BY Pros_ocjena_kol;
END //
DELIMITER ;

/*Poziv procedure*/
CALL vratiKolegije(2.2, 3.3);
```

naziv	idSmjer	idKolegij	Br_polozenih	Pros_ocjena_kol
Matematička analiza 1	12	23	5	2.4000
Baze podataka	1	1	6	2.5000
Molekularno modeliranje	13	26	4	2.5000
Operacijski sustavi	15	29	6	2.5000
Fizika	3	5	7	2.5714
Tehnička mehanika 2	20	39	5	2.6000
Ekonomika turizma	8	13	8	2.6250
Betonske konstrukcije	4	7	7	2.7143
Uvod u UNIX sustave	1	2	4	2.7500
Uvod u UNIX sustave	2	3	4	2.7500
Osnove elektrotehnike	3	8	4	2.7500
Mjera i integral	11	22	6	2.8333
Izrada web-projekta	15	30	6	2.8333
Poslovno komuniciranje	16	31	8	2.8750
Elementi konstrukcija	10	19	4	3.0000
Mikroekonomija I	18	35	8	3.0000
Tereti u prometu	21	41	6	3.0000
Poslovna etika	5	9	6	3.1667
Javna nabava	6	12	6	3.1667
Refrakcija i binokularni vid I	9	18	8	3.2500
Pogon zrakoplova I	10	20	8	3.2500

2.6. U bazi studenti: Napisati proceduru koja će za zadani smjer prikazati podatke po studentima koji su na ispitima dobili ocjene nedovoljan (1) ili dovoljan (2“ te da im je prosjek tih ocjena veći od 1. Prikazati sljedeće podatke:

- id smjera,
- naziv smjera,

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- jmbag studenta,
- broj ispita,
- prosječnu ocjenu prema zadanom kriteriju.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS vratiPodatkeSmjera //
CREATE PROCEDURE vratiPodatkeSmjera( p_idSmjer INT)
BEGIN
  SELECT sm.id, sm.naziv, agr.jmbagStudent, agr.Br_ispita,
        agr.Pros_ocjena_studenta
  FROM
    (SELECT s.idSmjer, o.jmbagStudent,
        COUNT(*) Br_ispita,
        AVG(ocjena) pros_ocjena_studenta
      FROM ocjene o INNER JOIN studenti s ON o.jmbagStudent=s.jmbag
      WHERE ocjena IN (1 , 2) AND s.idSmjer = p_idSmjer
      GROUP BY s.idSmjer, o.jmbagStudent
      HAVING AVG(ocjena) > 1) agr
  JOIN smjerovi sm ON agr.idSmjer = sm.id
  ORDER BY agr.Pros_ocjena_studenta DESC;
END//
DELIMITER ;

/*Poziv procedure*/
CALL vratiPodatkeSmjera(1);
```

id	naziv	jmbagStudent	Br_ispita	pros_ocjena_studenta
1	Stručni studij informatike	0128051999	1	2.0000
1	Stručni studij informatike	0246022345	2	2.0000
1	Stručni studij informatike	0246024883	2	2.0000

2.7. U bazi studenti: Napisati proceduru koja će za zadani kvartal prikazati podatke po kolegijima. Izuzeti negativne ocjene i kolegije koji su imali samo 1 položeni ispit.

Prikazati sljedeće podatke:

- naziv kolegija i id u zagradi nakon naziva,
- opis kolegija (prvih 100 znakova),
- najmanju ocjenu,
- najveću ocjenu te broj ispita.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Ispis sortirati po šifri kolegija obrnutim redoslijedom.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS Kolegiji_po_kvartalu //
CREATE PROCEDURE Kolegiji_po_kvartalu(zadani_kvartal INT)
BEGIN
  SELECT CONCAT(k.naziv, ' (', k.id, ')')naziv_sifra, LEFT(k.opis, 100) AS
opis_kol, agr.min_ocjena, agr.max_ocjena, agr.br_pol_ispita
  FROM
    (SELECT idKolegij, MIN(ocjena) min_ocjena, MAX(ocjena) max_ocjena,
       COUNT(*) br_pol_ispita
      FROM ocjene
      WHERE ocjena > 1 AND QUARTER(datumPolaganja) = zadani_kvartal
      GROUP BY idKolegij
      HAVING COUNT(*) > 1 ) agr
  JOIN kolegiji k ON agr.idKolegij = k.id
  ORDER BY k.id DESC ;
END//
DELIMITER ;

/* Poziv procedure */
CALL Kolegiji_po_kvartalu(1) ;
```

naziv_sifra	opis_kol	min_ocjena	max_ocjena	br_pol_ispita
Napredne baze podataka (46)	Student se treba upoznati s obj	2	5	3
Tjelesna i zdravstvena kultura 3 (45)	Odgojno-obrazovno područje podm	5	5	5
Tjelesna i zdravstvena kultura 2 (43)	Odgojno-obrazovno područje podm	5	5	5
Planiranje prometne potražnje (42)	Osposobiti studenta za primjenu	3	5	5
Tereti u prometu (41)	Osnovni cilj i zadatak kolegija	2	4	5
Sredstva pomorskog prometa (40)	Cilj kolegija je upoznati stude	4	5	3
Tehnička mehanika 2 (39)	Upoznati studente s teorijskim	2	3	3
Kvantitativne metode u managementu (36)	Upoznavanje sa osnovnim kvantit	2	5	2
Mikroekonomija I (35)	Razumijevanje principa ponude i	2	4	4
Marketing (34)	Marketing kao filozofija poslov	2	5	5
Osnove poduzetništva (33)	Pojmovno određenje poduzetništv	2	5	7
Poslovno komuniciranje (31)	Usvajanje općih znanja potrebni	2	5	2
Izrada web-projekta (30)	Vještina nudi studentima prilik	2	3	2
Operacijski sustavi (29)	Operacijski je sustav skup prog	3	4	2

2.8. U bazi autoradionica: Napisati proceduru koja će za zadani poštanski broj klijenta prikazati sljedeće podatke po klijentima iz tog mjesta:

- ime i prezime klijenta spojeno,
- godina unosa klijenta,
- broj odrađenih naloga,
- sumu ostvarenih sati.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

U ispis uključiti samo one klijente koji imaju više od jednog naloga, a ispis sortirati po sumi sati rada padajući.

Id i naziv županije potrebno je vratiti preko jednog izlaznog parametra.

Napisati smisleni primjer poziva ovog pohranjenog zadatka i ispisa iz izlaznog parametra.

```
DELIMITER //
DROP PROCEDURE IF EXISTS Klijenti_iz_zupanije //
CREATE PROCEDURE Klijenti_iz_zupanije (p_pbrKlij INT, OUT p_zupanija
VARCHAR(50))
BEGIN
  SELECT CONCAT(kl.imeKlijent, ' ', kl.prezimeKlijent) klijent,
YEAR(kl.datUnosKlijent) godina_unosa, agr.br_naloga, agr.sum_sati_rada
  FROM
    (SELECT k.sifKlijent, COUNT(*) br_naloga, SUM(n.OstvareniSatiRada)
sum_sati_rada
      FROM klijent k NATURAL JOIN nalog n
      WHERE k.pbrKlijent = p_pbrKlij
      GROUP BY k.sifKlijent
      HAVING COUNT(*) > 1 ) agr
  JOIN klijent kl ON agr.sifKlijent = kl.sifKlijent
  ORDER BY agr.sum_sati_rada DESC;

  SELECT CONCAT_WS(' ', z.sifZupanija, z.nazivZupanija)
    INTO p_zupanija
    FROM zupanija z JOIN mjesto m ON z.sifZupanija = m.sifZupanija
    WHERE m.pbrMjesto = p_pbrKlij;
END//
DELIMITER ;

/*Poziv procedure i ispis iz izlaznog parametra*/
CALL Klijenti_iz_zupanije(10000, @zupa);
```

klijent	godina_unosa	br_naloga	sum_sati_rada
Ivica Čučković	2016	11	66
Dragan Vidaković	2013	9	57
Jurica Bašić	2017	5	26
Gordan Boršić	2014	6	24
Dominik Hacek	2016	4	22
Antun Ivan Šimić	2011	3	20
Iva Devčić	2016	3	18
Tihomir Fizir	2011	4	18
Lea Kamenečki	2014	2	12
Mijo Poljak	2012	2	12
Juraj Sabljak	2019	2	11
Domagoj Krišto	2015	2	7

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT @zupa AS zadani_pbr;
```

Županija	
21 Grad Zagreb	14B

2.9. U bazi autoradionica: Napisati proceduru koja će za zadani poštanski broj prebivanja studenata prikazati sljedeće podatke o tim studentima:

- JMBAG studenta,
- ime studenta,
- prezime studenta,
- poštanski broj prebivanja,
- poštanski broj stanovanja,
- broj položenih ispita,
- prosječnu ocjenu položenih ispita.

Odabrati studente s više od jednog položenog ispita i prosjekom položenih ispita između 2 i 4.

Potrebno je putem ulazno izlaznog parametra vratiti prosjek svih ocjena svih studenata koji prebivaju u zadanom mjestu.

Ispis sortirati po prezimenu pa imenu.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS studenti_iz_mjesta //
CREATE PROCEDURE studenti_iz_mjesta (INOUT ulaz_izlaz DOUBLE)
BEGIN
  DECLARE tmp DOUBLE;
  SELECT s.jmbag, s.prezime, s.ime, s.postBrPrebivanje, s.postBrStanovanja,
s.agr.br_ocjena, agr.avg_ocjena
    FROM
      (SELECT jmbagStudent , COUNT(*) br_ocjena, AVG(ocjena) avg_ocjena
        FROM ocjene
        WHERE ocjena > 1
        GROUP BY jmbagStudent
        HAVING COUNT(*) > 1 AND AVG(ocjena) BETWEEN 2.00 AND 4.00) agr
    JOIN studenti s ON agr.jmbagStudent = s.jmbag
    WHERE s.postBrPrebivanje = ulaz_izlaz
    ORDER BY 2, 3 DESC;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  SELECT AVG(ocjena)
    INTO ulaz_izlaz
    FROM studenti JOIN ocjene ON jmbag = jmbagStudent
    WHERE postBrPrebivanje = ulaz_izlaz;
  END//
DELIMITER ;

/* Poziv procedure i ispis rezultata*/
SET @var = 32000;
CALL Studenti_iz_Mjesta(@var);
```

jmbag	prezime	ime	postBrPrebivanje	postBrStanovanja	Br_ocjena	avg_ocjena
0036440545	Bedić	Aleksandra	32000	32000	2	3.5000
0283035485	Grgić	Tomislav	32000	32100	2	3.0000
0283021548	Pokupić	Tomislav	32000	32000	2	2.0000

```
SELECT @var AS Prosjek_ocjena_u_mjestu;
```

Prosjek ocjena u mjestu
3.35

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

3. Funkcije

3.1. U bazi studenti: Napisati funkciju koja će za zadanu ocjenu vratiti broj studenata koji su ocijenjeni s tom ocjenom. Ocjena koja se unosi mora biti u opsegu 1 - 5. Ako se kao ulaz u funkciju zada ocjena izvan opsega, funkcija treba vratiti poruku: 'Neispravno unesena ocjena.', a ako je ocjena ispravna, funkcija treba vratiti: 'Ocjenom X ocijenjeno je Y studenata' ('X' i 'Y' zamijeniti s odgovarajućim vrijednostima).

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

Primjeri rezultata su u nastavku.

```
brojOcjena(5)
Ocjenom 5 ocijenjeno je 58 studenata.
```

```
brojOcjena(8)
Neispravno unesena ocjena.
```

```
DELIMITER //
DROP FUNCTION IF EXISTS brojOcjena //
CREATE FUNCTION  brojOcjena(p_ocjena INT) RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
  DECLARE brojOcjena INT DEFAULT NULL;
  IF p_ocjena < 1 OR p_ocjena > 5
    THEN RETURN 'Neispravno unesena ocjena.';
  ELSE
    SELECT COUNT(*) INTO brojOcjena
      FROM OCJENE
      WHERE ocjena =  p_ocjena;
    RETURN CONCAT('Ocjenom ', p_ocjena, ' ocijenjeno je ', brojOcjena,
        ' studenata.') ;
  END IF;
END; //
DELIMITER ;

/* Poziv funkcije */
SELECT brojOcjena(5);
SELECT brojOcjena(8);
```

3.2. U bazi studenti: Napisati funkciju koja će za zadanu šifru kvara ispisati koliko je bilo naloga po toj šifri kvara, prema sljedećem formatu. 'Broj naloga je: n'.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
DELIMITER //
DROP FUNCTION IF EXISTS brojiNaloge //
CREATE FUNCTION brojiNaloge(zadaniKvar INT) RETURNS VARCHAR(50)
DETERMINISTIC
BEGIN
  DECLARE br_naloga INT DEFAULT NULL;
  SELECT COUNT(*) INTO br_naloga
    FROM nalog
    WHERE sifKvar = zadaniKvar;
  RETURN CONCAT('Broja naloga je: ' , br_naloga);
END //
DELIMITER ;

/*Poziv funkcije*/
SELECT brojiNaloge(22) AS Broj;
```

Broj
Broja naloga je: 12

3.3. U bazi studenti: Napisati funkciju koja će za zadanu ocjenu ispisati tu ocjenu, koliko je bilo tih ocjena kao i zadnji datum kada je ta ocjena ostvarena, prema sljedećem formatu:

'Zadana ocjena = x; Broj ocjena = y; Max datum = yyyy-mm-dd'.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP FUNCTION IF EXISTS brojiOcjene //
CREATE FUNCTION brojiOcjene(zadanaOcjena INT) RETURNS VARCHAR(150)
DETERMINISTIC
BEGIN
  DECLARE v_ocjena, v_br_ocjena INT DEFAULT NULL;
  DECLARE v_Max_datum DATE DEFAULT NULL;
  SELECT ocjena, COUNT(*), MAX(datumPolaganja) INTO v_ocjena, v_br_ocjena,
         v_max_datum
    FROM ocjene
    WHERE ocjena = zadanaOcjena
    GROUP BY ocjena;
  RETURN CONCAT('Zadana ocjena = ', v_ocjena, '; Broj ocjena = ', v_br_ocjena, '
; Max datum = ' , v_max_datum);
END //
DELIMITER ;

/*Poziv funkcije*/
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT brojiOcjene(2) AS Izlaz;
```

Izlaz
Zadana ocjena = 2; Broj ocjena = 57 ; Max datum = 2020-02-27

3.4. U bazi autoradionica: Napisati funkciju koja će za zadani prioritet naloga ispisati koliko je bilo takvih naloga i njihov prosječan broj sati (na dvije decimale), prema sljedećem formatu. 'Broj naloga = n; Prosjek sati = x'.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP FUNCTION IF EXISTS brojiNalogeProsjek //
CREATE FUNCTION brojiNalogeProsjek (zadani_prioritet INT) RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
  DECLARE v_br_naloga INT DEFAULT NULL;
  DECLARE v_pros_sati DOUBLE DEFAULT NULL;
  SELECT COUNT(*), ROUND(AVG(OstvareniSatiRada),2) INTO v_br_naloga, v_pros_sati
  FROM nalog
  WHERE prioritetNalog = zadani_prioritet;
  RETURN CONCAT('Broj naloga = ', v_br_naloga, '; Prosjek sati = ',
v_pros_sati);
END//
DELIMITER ;

/*Poziv funkcije*/
SELECT brojiNalogeProsjek(2) as Izlaz;
```

Izlaz
Broj naloga = 86; Prosjek sati = 5.43

3.5. U bazi studenti: Napisati funkciju koja će za zadanu županiju prikazati id županije, naziv županije i broj nastavnika iz te županije, prema formatu u nastavku:

Ispis
Id = 2; Naziv = Zagrebačka županija; Broj nastavnika = 7

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP FUNCTION IF EXISTS brojiNastavnike //
CREATE FUNCTION brojiNastavnike (par_IdZupanije INT) RETURNS VARCHAR(300)
DETERMINISTIC
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
BEGIN
  DECLARE v_idZup, v_broj_nastavnika INT DEFAULT NULL;
  DECLARE v_naz_zup VARCHAR (60) DEFAULT NULL;
  DECLARE izlaz VARCHAR(300);
  SELECT z.id, z.nazivZupanija, COUNT(*) INTO v_idZup, v_naz_zup,
       v_broj_nastavnika
    FROM mjesta m INNER JOIN nastavnici n ON m.postbr =n.postBr
    INNER JOIN zupanije z ON m.idZupanija = z.id
    WHERE m.idZupanija = par_IdZupanije;
  SELECT CONCAT('Id = ', v_idZup, '; Naziv = ', v_naz_zup,
       '; Broj nastavnika = ' , v_broj_nastavnika)
    INTO izlaz;
RETURN izlaz;
END//
DELIMITER ;

/*Poziv funkcije*/
SELECT brojiNastavnike(2) AS Ispis;
```

3.6. U bazi studenti: Napisati funkciju koja će za zadanu godinu i smjer ispisati koliko je bilo upisanih studenata u toj godini na tom smjeru, prema sljedećem formatu: 'Broj upisanih studenata = n; Godina = x'.

Napisati smisleni poziv funkcije.

```
DELIMITER //
DROP FUNCTION IF EXISTS brStudenata //
CREATE FUNCTION brStudenata(p_god INT, p_idSmjera INT) RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
  DECLARE br_studenata INT DEFAULT NULL;
  SELECT COUNT(*) INTO br_studenata
    FROM studenti
    WHERE year(datumUpisa) = p_god AND idSmjer= p_idSmjera ;
  RETURN CONCAT('Broj upisanih studenata: ', br_studenata , ' Godina: ', p_god);
  END //
DELIMITER ;

/* Poziv funkcije */
SELECT brStudenata(2019,5) as Izlaz;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
Izlaz
Broj upisanih studenata: 4 Godina: 2019
```

3.7. U bazi autoradionica: Napisati funkciju koja će za zadani kvartal i šifru kvara ispisati koliko je odrađenih kvarova u tom kvartalu, prema sljedećem formatu: 'Broj odrađenih naloga:=n Kvartal=x'.

Napisati smisleni poziv funkcije.

```
DELIMITER //
DROP FUNCTION IF EXISTS brNaloga //
CREATE FUNCTION brNaloga(p_kvart INT, p_idKvara INT) RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
  DECLARE v_br_naloga INT DEFAULT NULL;
  SELECT COUNT(*) INTO v_br_naloga
  FROM nalog
  WHERE QUARTER(datPrimitkaNalog ) = p_kvart AND sifKvar = p_idKvara ;
  RETURN CONCAT('Broj odrađenih naloga: ', br_naloga , ' Kvartal: ', p_kvart);
END //
DELIMITER ;

/* Poziv funkcije */
SELECT brNaloga(1,1) AS Izlaz;
```

```
Izlaz
Broj odrađenih naloga: 6 Kvartal: 1
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

4. Kontrola toka u upitima i pohranjenim zadatcima

4.1. U bazi autoradionica: Napisati proceduru koja za zadani odjel ispisuje radnike iz tog odjela tako da ih sortira po najvećem prosječnom broju ostvarenih sati iz naloga uz oznaku je li je prosjek sati dotičnog radnika po nalogu iznad/ispod prosjeka tog odjela.

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

Primjer rezultata nakon poziva procedure je u nastavku.

sifRadnik	ImePrezime	prosjekSati	oznaka
142	Alan Ostojčić	5.9048	OverPerformer
469	Ivana Dumančić	5.6667	OverPerformer
330	Vinko Kecman	5.4167	UnderPerformer
391	Marko Hazdovac	4.6364	UnderPerformer

```
DELIMITER //
DROP PROCEDURE IF EXISTS OdjelRadnici //
CREATE PROCEDURE OdjelRadnici(IN zadaniOdjel INT)
BEGIN
  DECLARE ukupanProsjek DECIMAL(6,2);
  SELECT AVG(OstvareniSatiRada)  INTO ukupanProsjek
    FROM nalog NATURAL JOIN radnik
    WHERE sifOdjel = zadaniOdjel;
  SELECT n.sifRadnik, CONCAT(r.imeRadnik, ' ', r.prezimeRadnik) AS ImePrezime,
        AVG(OstvareniSatiRada) AS prosjekSati,
        IF (AVG(OstvareniSatiRada) >= ukupanProsjek, 'OverPerformer',
        'UnderPerformer') AS  oznaka
    FROM nalog n NATURAL JOIN  radnik r
    WHERE r.sifOdjel = zadaniOdjel
    GROUP BY n.sifRadnik, CONCAT(r.imeRadnik, ' ', r.prezimeRadnik)
    ORDER BY 3 DESC;
END;
//
DELIMITER ;

/*Poziv procedure*/
CALL OdjelRadnici(2);
```

4.2. U bazi autoradionica: Napisati funkciju koja će za zadanu šifru radnika ispisati njegovo ime onoliko puta koliko ima znakova u tom imenu (odvojeno crticom).

Nakon zadnjeg ispisa imena ne smije biti crtica.

a) Napisati primjer poziva funkcije za jednog radnika (primjer rezultata je u nastavku)

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

RepeatName(146)
Franjo - Franjo - Franjo - Franjo - Franjo - Franjo

b) Napisati primjer poziva funkcije za sve radnike, sortirano abecedno po imenu (primjer rezultata je u nastavku)

sifRadnik	imeRadnik	RepeatName(sifRadnik)
179	Adrijan	Adrijan - Adrijan - Adrijan - Adrijan - Adrijan - Adrijan - Adrija
142	Alan	Alan - Alan - Alan - Alan
297	Alen	Alen - Alen - Alen - Alen
358	Aljoša	Aljoša - Aljoša - Aljoša - Aljoša - Aljoša - Aljoša
150	Ana	Ana - Ana - Ana
322	Ana	Ana - Ana - Ana
346	Anđelko	Anđelko - Anđelko - Anđelko - Anđelko - Anđelko - Anđelko - Anđelk
130	Antun Ivan	Antun Ivan - Antun Ivan - Antun Ivan - Antun Ivan - Antun Ivan - A
403	Aurelija	Aurelija - Aurelija - Aurelija - Aurelija - Aurelija - Aurelija -
420	Bernard	Bernard - Bernard - Bernard - Bernard - Bernard - Bernard - Bernar
424	Blaž	Blaž - Blaž - Blaž - Blaž
432	Bojan	Bojan - Bojan - Bojan - Bojan - Bojan
260	Božidar	Božidar - Božidar - Božidar - Božidar - Božidar - Božidar - Božida
371	Božidar	Božidar - Božidar - Božidar - Božidar - Božidar - Božidar - Božida
514	Damjan	Damjan - Damjan - Damjan - Damjan - Damjan - Damjan
138	David	David - David - David - David - David
273	Davor	Davor - Davor - Davor - Davor - Davor
501	Davorin	Davorin - Davorin - Davorin - Davorin - Davorin - Davorin - Davori

```
DELIMITER $$
DROP FUNCTION IF EXISTS RepeatName $$
CREATE FUNCTION RepeatName (sifR INT) RETURNS VARCHAR(255)
DETERMINISTIC
BEGIN
  DECLARE n INT DEFAULT 0;
  DECLARE ime VARCHAR(255) DEFAULT NULL;
  DECLARE str VARCHAR(255) DEFAULT NULL;
  DECLARE var INT DEFAULT 0;
  SET var=1;
  SET str='';
  SELECT LENGTH(imeRadnik), imeRadnik INTO n, ime
    FROM radnik
    WHERE sifRadnik = sifR ;
  WHILE var <= n DO
    IF var < n
      THEN SET str = CONCAT(str, ime,' - ');
    ELSE SET str = CONCAT(str, ime);
    END IF;
  SET var = var + 1;
  END WHILE;
  RETURN str;
END $$
DELIMITER ;
/*Poziv funkcije za radnika sa šifrom 146*/
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT RepeatName(146);


/*Poziv funkcije za sve radnike */
SELECT sifRadnik, imeRadnik, RepeatName(sifRadnik) FROM radnik ORDER BY
ImeRadnik;
```

4.3. U bazi autoradionica: Napisati proceduru koja za zadani odjel ispisuje radnike iz tog odjela tako da ih sortira po najvećem prosječnom broju ostvarenih sati iz naloga uz oznaku je li je prosjek sati dotičnog radnika po nalogu iznad/ispod prosjeka tog odjela.

Napisati proceduru koja za zadani broj radnika i zadanu kategoriju plaće ispisuje sve odjele u kojima radi taj broj radnika te primaju plaću u zadanoj kategoriji. Za svaki odjel koji zadovoljava taj kriterij treba ispisati:

- naziv odjela,
- broj radnika,
- kategoriju plaće i
- prosječnu plaću.

Kategorije (A, B, C, D) se formiraju po sljedećim pravilima:

- 'A' - radnici koji imaju plaću manju od 2500
- 'B' - radnici koji imaju plaću između 2500,01 i 3000
- 'C' - radnici koji imaju plaću između 3000,01 i 3500
- 'D' - radnici koji imaju plaću veću od 3500

Sortirati po nazivu odjela rastući.

Napisati smisleni primjer poziva ovog pohranjenog zadatka. Primjer reziltata nakon poziva funkcije je u nastavku.

```
DELIMITER //
DROP PROCEDURE IF EXISTS kateogorijePlacaOdjel //
CREATE PROCEDURE kateogorijePlacaOdjel(IN brRad INT, IN kat CHAR(1))
BEGIN
  SELECT
    nazivOdjel AS 'Naziv odjela',
    COUNT(*) AS 'Broj radnika',
    CASE
      WHEN AVG(KoefPlaca*IznosOsnovice) < 2500 THEN 'A'
      WHEN AVG(KoefPlaca*IznosOsnovice) BETWEEN 2500.01 AND 3000 THEN 'B'
      WHEN AVG(KoefPlaca*IznosOsnovice) BETWEEN 3000.01 AND 3500 THEN 'C'
    ELSE 'D'
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    END
    AS 'Kategorija',
    ROUND(AVG(KoefPlaca*IznosOsnovice),2) AS 'Prosječna plaća'
  FROM radnik NATURAL JOIN odjel
  GROUP BY sifOdjel, nazivOdjel
  HAVING COUNT(*) = brRad AND Kategorija = kat
  ORDER BY 1;
END;
//
DELIMITER ;

CALL kateogorijePlacaOdjel(2, 'D');
```

4.4. U bazi autoradionica: Napisati proceduru koja za odabrani period (godina OD i godina DO) prikazuje troškove autoradionice po godinama i mjesecima. Trošak računati kao umnožak broja radnika koji je radio u mjesecu i prosječne mjesečne plaće.

Također treba osigurati da godina OD bude manja ili jednaka godini DO, da su godine četveroznamenkaste i da je barem jedna u vremenu kada postoji zapis o odrađenom nalogu. Napisati smisleni primjer poziva ovog pohranjenog zadatka.

Potrebno je prikazati podatke prema slici.

Godina	Mjesec	Broj radnika	Prosječna plaća	Trošak
2018	2	4	2536	10144
2018	3	6	2816	16896
2018	4	1	4554	4554
2018	5	4	3174	12696
2018	6	4	3174	12696
2018	7	2	2541	5082
2018	8	2	4037	8074
2018	9	5	3648	18240
2018	10	5	3221	16105
2018	11	4	4719	18876
2018	12	2	1551	3102
2019	1	2	5368	10736
2019	2	3	3879	11637
2019	4	1	3190	3190
2019	5	4	4054	16216
2019	6	2	3564	7128
2019	7	3	3175	9525
2019	8	3	4510	13530
2019	9	2	4554	9108
2019	10	3	4950	14850
2019	11	5	2983	14915
2019	12	1	3036	3036

```
DELIMITER //
DROP PROCEDURE IF EXISTS CostsYearMonth //
CREATE PROCEDURE CostsYearMonth (IN GOD_OD INT, IN GOD_DO INT, OUT MSG
VARCHAR(100))
BEGIN
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  IF
    LENGTH(GOD_OD) = 4 AND LENGTH(GOD_DO)= 4
    AND GOD_OD <= GOD_DO
    AND (GOD_OD IN (SELECT DISTINCT YEAR(datPrimitkaNalog) FROM NALOG)
      OR GOD_DO IN (SELECT DISTINCT YEAR(datPrimitkaNalog) FROM NALOG))
  THEN
    SELECT YEAR(n.datPrimitkaNalog) AS 'Godina',
      MONTH(n.datPrimitkaNalog) AS 'Mjesec',
      COUNT(*) AS 'Broj radnika',
      ROUND(AVG(r.KoefPlaca * r.IznosOsnovice), 0) AS 'Prosječna plaća',
      ROUND(AVG(r.KoefPlaca * r.IznosOsnovice), 0) * COUNT(*) AS 'Trošak'
    FROM nalog n NATURAL JOIN radnik r
    WHERE YEAR(n.datPrimitkaNalog) BETWEEN GOD_OD AND GOD_DO
    GROUP BY YEAR(datPrimitkaNalog), MONTH(datPrimitkaNalog)
    ORDER BY 1, 2;
  ELSE SET MSG = 'Godine nisu dobro definirane / nisu u traženom rasponu';
  END IF;
END;
//
DELIMITER ;

/*Pozivi procedure*/

CALL CostsYearMonth(2004, 2001, @MSG);
SELECT @MSG;
```

@MSG	
Godine nisu dobro definirane / nisu u traženom...	55B

```
CALL CostsYearMonth(2018, 2019, @MSG);
SELECT @MSG;
```

Godina	Mjesec	Broj radnika	Prosječna plaća	Trošak
2018	2	4	2536	10144
2018	3	6	2816	16896
2018	4	1	4554	4554
2018	5	4	3174	12696
2018	6	4	3174	12696
2018	7	2	2541	5082
2018	8	2	4037	8074
2018	9	5	3648	18240
2018	10	5	3221	16105
2018	11	4	4719	18876
2018	12	2	1551	3102
2019	1	2	5368	10736
2019	2	3	3879	11637
2019	4	1	3190	3190
2019	5	4	4054	16216
2019	6	2	3564	7128
2019	7	3	3175	9525
2019	8	3	4510	13530
2019	9	2	4554	9108
2019	10	3	4950	14850
2019	11	5	2983	14915
2019	12	1	3036	3036

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
CALL CostsYearMonth(2004, 2005, @MSG);
SELECT @MSG;
```

@MSG
Godine nisu dobro definirane / nisu u traženom... 55B

4.5. U bazi autoradionica: Napisati proceduru koja će za zadanu šifru kvara preko izlaznih parametara ispisati: naziv kvara, broj naloga i broj **različitih** klijenata koji su imali taj kvar.

Ako nema tražene šifre u nalozima vraća poruku: "Nepostojeća šifra kvara".

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS statistikaPoKvaru //
CREATE PROCEDURE statistikaPoKvaru (IN zadanaSifKvara INT, OUT nazivKvara
VARCHAR(100), OUT brNaloga INT, OUT brKlijenata INT)
BEGIN
  IF zadanaSifKvara IN (SELECT DISTINCT sifKvar FROM nalog)
  THEN
    SELECT k.nazivKvar, COUNT(*) INTO nazivKvara, brNaloga
      FROM nalog n NATURAL JOIN kvar k
      WHERE n.sifKvar = zadanaSifKvara
      GROUP BY k.nazivKvar;
    SELECT COUNT(DISTINCT sifKlijent) INTO brKlijenata
      FROM nalog
      WHERE sifKvar = zadanaSifKvara;
  ELSE
    SELECT 'Nepostojeća šifra kvara.' AS Poruka;
  END IF;
END//
DELIMITER ;

/* Pozivi procedure i ispisi rezultata */
CALL statistikaPoKvaru(27, @naziv_kvara, @broj_naloga,
  @broj_razlicitih_klijenata);
SELECT @naziv_kvara AS "Naziv kvara", @broj_naloga AS "Broj naloga",
  @broj_razlicitih_klijenata AS "Broj različitih klijenata";
```

Naziv kvara	Broj naloga	Broj različitih klijenata
Zamjena blatobrana 18B	16	10

```
CALL statistikaPoKvaru(99, @naziv_kvara, @broj_naloga,
@broj_razlicitih_klijenata );
```

Poruka
Nepostojeća šifra kvara.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

4.6. U bazi studenti: Napisati proceduru koja za zadanu šifru smjera ispisuje studente s tog smjera na način da ih sortira po najvećoj prosječnoj ocjeni i nakon toga po broju ispita (prebrojati sve ocjene) uz oznaku (*flag*) da li je prosjek studenta iznad ili ispod prosjeka tog smjera. Prilikom ispisa, dodatno je potrebno ispisati spojeno ime i prezime studenta. Ispis sortirati silazno po prosječnoj ocjeni te nakon toga po broju ispita.

Ako nema tražene šifre u tablici smjerovi vraća poruku: "Nepostojeća šifra smjera".

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS SmjerStudenti//
CREATE PROCEDURE SmjerStudenti (IN zadaniSmjer INT)
BEGIN
  DECLARE v_AVG_ocj_smjera DECIMAL(6,2);
  IF zadaniSmjer IN (SELECT id FROM smjerovi)
  THEN
    SELECT AVG(o.ocjena) INTO v_AVG_ocj_smjera
      FROM ocjene o INNER JOIN kolegiji k ON o.idKolegij = k.id
      WHERE k.idSmjer = zadaniSmjer;
    SELECT o.jmbagStudent,
        CONCAT(s.ime, ' ', s.prezime) AS ImePrezime,
        COUNT(*) AS BrojIspita,
        AVG(o.ocjena) AS ProsječnaOcjenaStudenta,
        IF (AVG(o.ocjena) > v_AVG_ocj_smjera, 'Above AVG', 'Under AVG') AS Flag
      FROM ocjene o INNER JOIN studenti s ON o.jmbagStudent = s.jmbag
      WHERE s.idSmjer = zadaniSmjer
      GROUP BY o.jmbagStudent, CONCAT(s.ime, ' ', s.prezime)
      ORDER BY 4 DESC, 3 DESC;

  ELSE
    SELECT 'Nepostojeća šifra smjera.' AS Poruka;
  END IF;
END;
//
DELIMITER ;

/* Pozivi procedure */
CALL SmjerStudenti(1);
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

jmbagStudent	ImePrezime	BrojIspita	ProsječnaOcjenaStudenta	Flag
0128050002	Kristina Žikić	1	5.0000	Above AVG
0128050853	Eva Delač	3	3.6667	Above AVG
0128051999	Marina Zrno	3	3.6667	Above AVG
0035177990	Irena Dabić	4	3.2500	Under AVG
0246022345	Ivana Babić	4	3.0000	Under AVG
0246024883	Ivana Jelenčić	3	3.0000	Under AVG

```
CALL SmjerStudenti(99);
```

Poruka
Nepostojeća šifra smjera.

4.7. U bazi autoradionica: Napisati proceduru koja za zadani OIB ustanove u obliku matrice prikazuje nazive smjerova i te koliko je za svaki smjer evidentirano ocjena, posebno za svaku ocjenu (1-5) pojedinačno. Prikazati i prosječnu ocjenu smjera kao i standardnu devijaciju. Ispis sortirati uzlazno po prosjeku smjera.

Ako zadani OIB ne postoji u tablici USTANOVE, procedura vraća poruku: "Nepostojeći OIB ustanove.".

Zadatak je potrebno riješiti pomoću funkcije za sumiranje (ocjena).

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS ocjenePoSmjerovimaUstanove_SUM //
CREATE PROCEDURE ocjenePoSmjerovimaUstanove_SUM (IN zadaniOib CHAR(11))
BEGIN
  IF zadaniOib IN (SELECT oib FROM ustanove)
  THEN
    SELECT s.naziv AS Naziv_smjera,
    SUM(CASE WHEN o.ocjena=1 THEN 1 ELSE 0 END) AS OC_1,
    SUM(CASE WHEN o.ocjena=2 THEN 1 ELSE 0 END) AS OC_2,
    SUM(CASE WHEN o.ocjena=3 THEN 1 ELSE 0 END) AS OC_3,
    SUM(CASE WHEN o.ocjena=4 THEN 1 ELSE 0 END) AS OC_4,
    SUM(CASE WHEN o.ocjena=5 THEN 1 ELSE 0 END) AS OC_5,
    ROUND(AVG(o.ocjena),2) AS AVG_smjera,
    ROUND(STD(o.ocjena),2) AS STD_smjera
      FROM smjerovi s INNER JOIN kolegiji k ON s.id = k.idSmjer
      INNER JOIN ocjene o ON k.id = o.idKolegij
      WHERE s.oibUstanova = zadaniOib
      GROUP BY s.id, s.naziv
      ORDER BY 7 DESC;
  ELSE SELECT 'Nepostojeći OIB ustanove. ' AS Poruka;
  END IF;
END;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
//
DELIMITER ;


/* Pozivi procedure */
CALL ocjenePoSmjerovimaUstanove_SUM ('08814003451');
```

Naziv smjera	OC_1	OC_2	OC_3	OC_4	OC_5	AVG_smjera	STD_smjera
Stručni studij informatike	2	5	2	2	7	3.39	1.5
Stručni studij računarstva	1	1	2	2	2	3.38	1.32
Stručni studij graditeljstv	3	1	2	3	2	3.00	1.48
Stručni studij elektrotehnil	3	3	1	3	1	2.64	1.37

```
CALL ocjenePoSmjerovimaUstanove_SUM ('99999999999');
```

Poruka
Nepostojeći OIB ustanove.

4.8. U bazi autoradionica: Napisati proceduru koja za zadani OIB ustanove u obliku matrice prikazuje nazive smjerova i te koliko je za svaki smjer evidentirano ocjena, posebno za svaku ocjenu (1-5) pojedinačno. Prikazati i prosječnu ocjenu smjera kao i standardnu devijaciju. Ispis sortirati uzlazno po prosjeku smjera.

Ako zadani OIB ne postoji u tablici USTANOVE, procedura vraća poruku: "Nepostojeći OIB ustanove.".

Zadatak je potrebno riješiti pomoću funkcije za prebrojavanje (ocjena).

Napisati smisleni primjer poziva ovog pohranjenog zadatka.

```
DELIMITER //
DROP PROCEDURE IF EXISTS ocjenePoSmjerovimaUstanove //
CREATE PROCEDURE ocjenePoSmjerovimaUstanove_COUNT(IN zadaniOib CHAR(11))
BEGIN
  IF zadaniOib IN (SELECT oib FROM ustanove)
  THEN
    SELECT s.naziv AS Naziv_smjera,
    COUNT(CASE WHEN o.ocjena=1 THEN 1 ELSE NULL END ) AS OC_1,
    COUNT(CASE WHEN o.ocjena=2 THEN 1 ELSE NULL END ) AS OC_2,
    COUNT(CASE WHEN o.ocjena=3 THEN 1 ELSE NULL END ) AS OC_3,
    COUNT(CASE WHEN o.ocjena=4 THEN 1 ELSE NULL END ) AS OC_4,
    COUNT(CASE WHEN o.ocjena=5 THEN 1 ELSE NULL END ) AS OC_5,
    ROUND(AVG(o.ocjena),2) AS AVG_smjera,
    ROUND(STD(o.ocjena),2) AS STD_smjera
      FROM smjerovi s INNER JOIN kolegiji k ON s.id = k.idSmjer
      INNER JOIN ocjene o ON k.id = o.idKolegij
      WHERE s.oibUstanova = zadaniOib
      GROUP BY s.id, s.naziv
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      ORDER BY 7 DESC;
  ELSE SELECT 'Nepostojeći OIB ustanove. ' AS Poruka;
  END IF;
END;
//
DELIMITER ;

/* Pozivi procedure */
CALL ocjenePoSmjerovimaUstanove_COUNT('08814003451');
```

Naziv smjera	OC_1	OC_2	OC_3	OC_4	OC_5	AVG_smjera	STD_smjera
Stručni studij informatike	2	5	2	2	7	3.39	1.5
Stručni studij računarstva	1	1	2	2	2	3.38	1.32
Stručni studij graditeljstv	3	1	2	3	2	3.00	1.48
Stručni studij elektrotehnil	3	3	1	3	1	2.64	1.37

```
CALL ocjenePoSmjerovimaUstanove_COUNT ('99999999999');
```

Poruka
Nepostojeći OIB ustanove.

4.9. U bazi autoradionica: Napisati funkciju koja za zadanog klijenta ispisuje njegovo ime i prezime odvojeno crticom, onoliko puta koliko je taj klijent imao naloga (na kraju ispisa crtica se ne smije pojaviti). Potrebno je koristiti funkciju CONCAT.

Ako zadana šifra klijenta ne postoji u tablici nalog, funkcija vraća poruku: "Nepostojeća šifra klijenta".

a) Napisati primjer poziva funkcije za jednog klijenta.

b) Napisati primjer poziva funkcije za sve klijente sa barem jednim nalogom čije prezime počinje slovom 'M'. Ispisati samo jedinstvene vrijednosti.

```
DELIMITER $$
DROP FUNCTION IF EXISTS ClientName $$
CREATE FUNCTION ClientName(sifK INT) RETURNS VARCHAR(500) DETERMINISTIC
BEGIN
  DECLARE n INT DEFAULT 0;
  DECLARE str, str_kum VARCHAR(500);
  DECLARE var INT;
  SET var = 1;
  SET str = '';
  SET str_kum = '';
  SELECT COUNT(*) INTO n FROM nalog WHERE sifKlijent = sifK;
  SELECT CONCAT(imeKlijent,' ', prezimeKlijent) INTO str
    FROM klijent
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    WHERE sifKlijent = sifK;
  IF sifK IN (SELECT DISTINCT sifKlijent FROM nalog)
  THEN
    WHILE var <= n DO
      IF var < n THEN
        SET str_kum = CONCAT(str_kum, str, ' - ');
      ELSE
        SET str_kum = CONCAT(str_kum, str);
      END IF;
      SET var = var + 1;
    END WHILE;
    RETURN str_kum;
  ELSE RETURN 'Nepostojeća šifra klijenta' ;
  END IF;
END; $$
DELIMITER ;

/* Pozivi funkcije */
SELECT ClientName(1127);
```

ClientName(1127)
Dario Risek - Dario Risek - Dario Risek - Dario Risek - Dario Risek

```
SELECT DISTINCT(ClientName(sifKlijent))
  FROM nalog NATURAL JOIN klijent
  WHERE prezimeKlijent LIKE "M%";
```

(ClientName(sifKlijent))
Niko Marušić
Majo Musić
Iva Mioč - Iva Mioč - Iva Mioč - Iva Mioč - Iva Mioč - Iva Mioč - Iva Mioč -
Perica Mikulec - Perica Mikulec - Perica Mikulec - Perica Mikulec - Perica M
Darko Mrduljaš - Darko Mrduljaš - Darko Mrduljaš - Darko Mrduljaš
Vinko Marković
Majo Meglaj - Majo Meglaj - Majo Meglaj - Majo Meglaj - Majo Meglaj - Majo M
Juraj Maček
Edo Marić
Željka Morić - Željka Morić - Željka Morić - Željka Morić - Željka Morić - Ž

...

4.10. U bazi autoradionica: Napisati funkciju koja za zadanog radnika prikazuje njegovo ime i prezime odvojeno crticom onoliko puta koliko je taj radnik imao naloga (na kraju ispisa crtica se ne smije pojaviti). Potrebno je koristiti funkciju CONCAT_WS.

Ako zadana šifra radnika ne postoji u tablici nalog, funkcija vraća poruku: "Nepostojeća šifra radnika u tablici nalog.".

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Napisati primjer poziva funkcije za jednog radnika.

Napisati primjer poziva funkcije za sve radnike koji pripadaju odjelu 'Alarmi'.

```
DROP FUNCTION IF EXISTS WorkerName;
DELIMITER $$
CREATE FUNCTION WorkerName(p_sifRadnik INT) RETURNS VARCHAR(300) DETERMINISTIC
BEGIN
  DECLARE v_i INT DEFAULT 0;
  DECLARE v_brNaloga INT DEFAULT NULL;
  DECLARE v_imePrezime VARCHAR(30) DEFAULT NULL;
  DECLARE v_ispis VARCHAR(300) DEFAULT NULL;
  IF p_sifRadnik NOT IN (SELECT sifRadnik FROM nalog)
  THEN
    RETURN 'Nepostojeća šifra radnika u tablici nalog.';
  END IF;
  SELECT COUNT(*) INTO v_brNaloga FROM nalog
    WHERE sifRadnik = p_sifRadnik;
  SELECT CONCAT(imeRadnik, ' ', prezimeRadnik) INTO v_imePrezime FROM radnik
    WHERE sifRadnik = p_sifRadnik;
  SET v_ispis = v_imePrezime;
  WHILE v_i < v_brNaloga - 1
  DO
    SET v_ispis = CONCAT_WS(' - ', v_ispis, v_imePrezime);
    SET v_i = v_i + 1;
  END WHILE;
  RETURN v_ispis;
END; $$
DELIMITER ;

/* Pozivi funkcije */
SELECT WorkerName(122);
```

WorkerName(122)
Tin Grabovac - Tin Grabovac - Tin Grabovac - Tin Grabovac - Tin Grabovac - Tin Grabovac - Tin Grabovac - Tin

... (13 puta)

```
SELECT WorkerName(radnik.sifRadnik)
  FROM radnik
  JOIN odjel ON radnik.sifOdjel = odjel.sifOdjel
  WHERE nazivOdjel = "Klima";
```

WorkerName(radnik.sifRadnik)
Nepostojeća šifra radnika u tablici nalog.
Sanja Watz - Sanja Watz - Sanja Watz - Sanja Watz - Sanja Watz - Sanja Watz - Sanja Watz -
Sandra Mihaljević - Sandra Mihaljević - Sandra Mihaljević - Sandra Mihaljević - Sandra Mih.
Nepostojeća šifra radnika u tablici nalog.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

dohvaceno studenata
18

count(*)
18

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

5. Kursori i *handleri*

5.1. U bazi autoradionica: Napisati proceduru koja će ispisati podatke klijenta (ime, prezime i šifru), te uz njega ispisati iznos popusta koji je taj klijent ostvario kumulativno gledajući sve njegove naloge. Popust se dodjeljuje po slijedećem principu:

- 5% popusta za 10 ili više sati kvara,
- 10% popusta za 20 ili više sati kvara,
- 15% popusta za 30 ili više sati kvara,
- 20% popusta za 40 ili više sati kvara.

Ispisati samo one klijente koji imaju pravo na popust. Ispis sortirati uzlazno po broju sati kvara. Osigurat da po završetku procedure privremena tablica više nije dostupna.

Potrebno je koristiti kursor, handler i privremenu tablicu.

Napisati primjer poziva ovog pohranjenog zadatka.

```
DROP PROCEDURE IF EXISTS ispisiPopuste;
DELIMITER //
CREATE PROCEDURE ispisiPopuste()
BEGIN
  DECLARE sif, brSati, popust INT;
  DECLARE ime, prezime VARCHAR(255);
  DECLARE nastavi BOOL;
  DECLARE kur CURSOR FOR
    SELECT klijent.sifKlijent, klijent.imeKlijent, klijent.prezimeKlijent,
        SUM(kvar.satiKvar)
      FROM klijent
        JOIN nalog ON klijent.sifKlijent = nalog.sifKlijent
        JOIN kvar ON nalog.sifKvar = kvar.sifKvar
      GROUP BY klijent.sifKlijent, klijent.imeKlijent, klijent.prezimeKlijent;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET nastavi = FALSE;
  SET nastavi = TRUE;
  DROP TEMPORARY TABLE IF EXISTS tmp;
  CREATE TEMPORARY TABLE tmp (
    sif INT,
    ime VARCHAR(255),
    prezime VARCHAR(255),
    brojSati INT,
    popust INT
  );
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  OPEN kur;
  petlja:LOOP
    FETCH kur INTO sif, ime, prezime, brSati;
    IF nastavi = FALSE THEN
      LEAVE petlja;
    END IF;
    IF brSati >= 40 THEN
      INSERT INTO tmp VALUES(sif, ime, prezime, brSati, 20);
    ELSEIF brSati >= 30 THEN
      INSERT INTO tmp VALUES(sif, ime, prezime, brSati, 15);
    ELSEIF brSati >= 20 THEN
      INSERT INTO tmp VALUES(sif, ime, prezime, brSati, 10);
    ELSEIF brSati >= 10 THEN
      INSERT INTO tmp VALUES(sif, ime, prezime, brSati, 5);
    END IF;
  END LOOP;
  SELECT * FROM tmp ORDER BY brojSati;
  DROP TEMPORARY TABLE IF EXISTS tmp;
END//
DELIMITER ;

/* Poziv procedure */
CALL ispisiPopuste();
```

sif	ime	prezime	brojSati	popust
1471	Božidar	Ježić	10	5
1202	Perica	Mikulec	10	5
1145	Jurica	Bašić	10	5
1414	Tin	Šipek	10	5
1286	Fran	Šef	11	5
1306	Helena	Octenjak	11	5
1127	Dario	Risek	12	5
1498	Janko	Jakopović	12	5

...

1490	Bruno	Hundak	25	10
1230	Predrag	Buljat	26	10
1441	Majo	Meglaj	38	15
1312	Nina	Kućan	49	20
1172	Ninoslav	Novak	50	20

5.2. U bazi autoradionica: Napisati proceduru koja će za zadani prioritet naloga prikazati sve odjele koji su ostvarili više od jednog naloga sa zadanim prioritetom.

Za svaki odjel potrebno je ispisati:

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- id odjela,
- naziv odjela,
- iznos prosječno ostvarenih sati,
- broja naloga,
- tip odjela prema broju odrađenih naloga i to

- UNDER_VOLUME ako ima manje od 20 naloga i
- OVER_VOLUME ako ima više ili jednako 20 naloga.

Zadatak riješiti pomoću kursora i privremene tablice, bez korištenja handlera. Sortirati silazno prema broju naloga.

Napisati primjer poziva procedure.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS odjeli_nalozi$$
CREATE PROCEDURE odjeli_nalozi(prior INT)
BEGIN
  DECLARE c_nazivOd VARCHAR(255) DEFAULT NULL;
  DECLARE c_sifOd, c_brNal, dohvaceno INT DEFAULT NULL;
  DECLARE c_AvgSati DECIMAL(4,2) DEFAULT 0;
  DECLARE i INT DEFAULT 0;
  DECLARE kur CURSOR FOR
  SELECT r.sifOdjel, o.nazivOdjel, AVG(OstvareniSatiRada) AS avg_sati,
        COUNT(*) AS br_naloga
    FROM nalog n NATURAL JOIN radnik r NATURAL JOIN odjel o
    WHERE prioritetNalog = prior
    GROUP BY r.sifOdjel, o.nazivOdjel
    HAVING COUNT(*) > 1
    ORDER BY 4 DESC ;

  DROP TEMPORARY TABLE IF EXISTS tmp;
  CREATE TEMPORARY TABLE tmp(idOd INT(11), nazivOd VARCHAR(50),
        AvgSati DECIMAL(4,2), brNal INT(11), tipOd VARCHAR(50));

  OPEN kur;
    SELECT FOUND_ROWS() INTO dohvaceno;
    WHILE i < dohvaceno DO
      FETCH kur INTO c_sifOd, c_nazivOd, c_AvgSati, c_brNal;
      INSERT INTO tmp (idOd, nazivOd, AvgSati, brNal, tipOd)
        VALUES ( c_sifOd, c_nazivOd, c_AvgSati, c_brNal,
      IF(c_brNal >= 20, 'OVER_VOLUME', 'UNDER_VOLUME'));
      SET i=i+1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    END WHILE;
  CLOSE kur;
  SELECT * FROM tmp;
END; $$
DELIMITER ;

/* Poziv procedure */
CALL odjeli_nalozi(1);
```

idOd	nazivOd	AvgSati	brNal	tipOd
2	Limarija	5.43	23	OVER_VOLUME
3	Elektrika	5.04	23	OVER_VOLUME
6	Mehanika	6.48	21	OVER_VOLUME
4	Vulkanizacija	5.41	17	UNDER_VOLUME
15	Skladište	6.67	15	UNDER_VOLUME
22	Mulde	5.57	14	UNDER_VOLUME
13	Klima	3.64	11	UNDER_VOLUME
19	Balansiranje guma	4.38	8	UNDER_VOLUME
5	Optika	5.33	6	UNDER_VOLUME
24	Laki motori	6.33	3	UNDER_VOLUME
16	Balansiranje	4.67	3	UNDER_VOLUME
11	Elektropokretači	6.33	3	UNDER_VOLUME
20	Amortizeri	2.50	2	UNDER_VOLUME

5.3. U bazi autoradionica: Napisati proceduru koja će svim kvarovima koji imaju manje od 10 odrađenih naloga, smanjiti broj radnika za 1. Potrebno je osigurati da se broj radnika ne može postaviti na manje od 1.

Procedura mora vratiti broj n-torki koje je dohvatila i obradila. U optimalno postavljenim petljama, očekujemo da će ove dvije vrijednosti biti identične odnosno da procedura neće odrađivati nepotrebni broj iteracija.

Zadatak riješiti pomoću kursora i handlera. Napisati primjer poziva procedure.

```
DELIMITER //
DROP PROCEDURE IF EXISTS korigiraj_broj_radnika //
CREATE PROCEDURE korigiraj_broj_radnika()
BEGIN
  DECLARE c_sif, dohvaceno, smanjeno INT;
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
  SELECT sifKvar
    FROM nalog
    WHERE sifKvar IN (SELECT sifKvar FROM kvar WHERE brojRadnika > 1)
    GROUP BY sifKvar
    HAVING COUNT(*) < 10;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag=TRUE;
  SET smanjeno=0;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  OPEN kurs;
  SELECT FOUND_ROWS() INTO dohvaceno;

  petlja:LOOP
    FETCH kurs INTO c_sif ;
    IF flag=TRUE THEN
    LEAVE petlja;
    END IF;
    UPDATE kvar SET brojRadnika = brojRadnika - 1 WHERE sifKvar = c_sif;
    SET smanjeno=smanjeno+1;
  END LOOP;
  CLOSE kurs;

  SELECT dohvaceno AS dohvaceno_rezultata, smanjeno AS smanjeni_br_radnika;
END; //
DELIMITER ;

/* Poziv procedure */
CALL korigiraj_broj_radnika();
```

dohvaceno_rezultata	smanjeni_br_radnika
7	7

5.4. U bazi studenti: Napisati proceduru koja će u bazi podataka označiti sve studente koji imaju veći broj negativnih od broja pozitivnih ocjena. U atributu prezime, nakon prezimena dodati oznaku " n1 = x", gdje je x broj negativnih ocjena.

Navedeno napraviti samo za studente koji imaju barem jedan nepoložen ispit.

Procedura mora vratiti broj n-torki koje je dohvatila i obradila.

Zadatak riješiti pomoću kursora i handlera. Napisati primjer poziva procedure.

```
DELIMITER //
DROP PROCEDURE IF EXISTS mark_students //
CREATE PROCEDURE mark_students()
BEGIN
  DECLARE c_br_nepol, c_br_pol, dohvaceno, markirano INT;
  DECLARE c_jmbag, c_prezime VARCHAR(50);
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
    SELECT o.jmbagStudent, s.prezime,
        COUNT(IF(o.ocjena = 1, 1, NULL)) AS br_nepol,
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
        COUNT(IF(o.ocjena > 1, 1, NULL)) AS br_pol
      FROM ocjene o INNER JOIN studenti s ON o.jmbagStudent = s.jmbag
      GROUP BY o.jmbagStudent, s.prezime
      HAVING br_nepol > 0 AND br_nepol > br_pol;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag=TRUE;
  SET markirano=0;

  OPEN kurs;
  SELECT FOUND_ROWS() INTO dohvaceno;

  petlja:LOOP
    FETCH kurs INTO c_jmbag, c_prezime, c_br_nepol, c_br_pol;
    IF flag=TRUE THEN
      LEAVE petlja;
    END IF;
    UPDATE studenti SET prezime =
      CONCAT(prezime, ' n1 = ', c_br_nepol) WHERE jmbag = c_jmbag;
    SET markirano=markirano+1;
  END LOOP;
  CLOSE kurs;
  SELECT dohvaceno AS dohvaceno_studenata, markirano AS markirano_studenata;
END; //
DELIMITER ;

/* Poziv procedure */
CALL mark_students();
```

dohvaceno_studenata	markirano_studenata
6	6

```
/* Dodatna provjera svih studenata sa barem jednom negativnom ocjenom */
SELECT * FROM studenti JOIN ocjene ON jmbag = jmbagStudent WHERE jmbag IN
  (SELECT jmbagStudent WHERE ocjena = 1);
```

jmbag	ime	prezime	datumUpisa	postBrPrebivanje	postBrStanovanja	idSmjer
0035166854	Krešo	Šišić	2018-09-26	10410	10410	10
0035177990	Irena	Dabić	2019-09-26	10000	10000	1
0035192211	Karla	Brandis	2017-09-28	10000	40000	4
0036460022	Ivan	Matić	2015-09-24	10000	10000	15
0036460031	Dajana	Jerec	2017-09-28	10000	10430	14
0036499965	Lovro	Keglović 1 = 2	2015-09-26	10000	10410	15
0036499965	Lovro	Keglović 1 = 2	2015-09-26	10000	10410	15
0067405522	Dario	Veh	2017-09-29	21000	10430	18
0067408000	Nikola	Mucić	2017-09-28	21000	21000	19
0112034516	Denis	Vrdoljak	2018-09-29	51000	52440	21
0128050853	Eva	Delač	2017-09-30	10000	51000	1

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

5.5. U bazi studenti: Napisati funkciju koja za zadani smjer i oznaku smanjenja 'S' odnosno povećanja 'P', smanjuje ili povećava sve ocjene studentima zadanog smjera. Ocjenu 1 nije moguće smanjiti a ocjenu 5 nije moguće povećati.

Oznake smanjenja odnosno povećanja mogu biti mala ili velika slova 'S' i 's' odnosno 'P' i 'p'.

Funkcija mora vratiti broj studenata kojima je ocjenu dohvatila, smanjila i povećala (u jednom pozivu funkcije očekuju se ili povećanja ili smanjenja).

Zadatak riješiti pomoću kursora i handlera.

Napisati primjer poziva funkcije.

```
DELIMITER //
DROP FUNCTION IF EXISTS CorrectMark //
CREATE FUNCTION CorrectMark (smjer INT, inc_dec CHAR(1)) RETURNS VARCHAR(255)
DETERMINISTIC
BEGIN
  DECLARE dohvaceno, br_smanjenja, br_povecanja INT DEFAULT 0;
  DECLARE c_jmbag VARCHAR(10) DEFAULT NULL;
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
    SELECT jmbag FROM studenti WHERE idSmjer = smjer;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag=TRUE;
  IF UPPER(inc_dec) NOT IN ('S', 'P') THEN
    RETURN ('Neispravna oznaka smanjenja/povećanja.');
  END IF;
  OPEN kurs;
    SELECT FOUND_ROWS() INTO dohvaceno;
    petlja:LOOP
      FETCH kurs INTO c_jmbag ;
      IF flag=TRUE THEN
        LEAVE petlja;
      END IF;
      CASE WHEN UPPER(inc_dec) = 'S'
      THEN
        UPDATE ocjene SET ocjena = ocjena - 1
          WHERE jmbagStudent = c_jmbag and ocjena > 1;
        SET br_smanjenja = br_smanjenja + 1;
      WHEN UPPER(inc_dec) = 'P'
      THEN
        UPDATE ocjene SET ocjena = ocjena + 1
          WHERE jmbagStudent = c_jmbag and ocjena < 5;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
        SET br_povecanja = br_povecanja + 1;
      END CASE;
    END LOOP;
  CLOSE kurs;
  RETURN CONCAT('Dohvaćeno: ', dohvaceno, ' Smanjeno: ',br_smanjenja, '
Povećano: ', br_povecanja);
END; //
DELIMITER ;

/* Pozivi funkcije */
SELECT CorrectMark(1, 's') AS poruka;
```

poruka
Dohvaćeno: 7 Smanjeno: 7 Povećano: 0

```
SELECT CorrectMark(1, 'abc');
```

CorrectMark(1, 'abc')
Neispravna oznaka smanjenja/povećanja.

5.6. U bazi autoradionica: Napisati funkciju koja za zadanu šifru kvara u bazi podataka označava sve klijente čiji je udio ostvarenih sati rada u ukupnim satima rada (za taj kvar) veći od 5%. Klijente označava tako da im iza prezimena doda oznaku ' VIP -> % udio = ' i iznos udjela u %. Provjeriti je li ispravno zadana šifra kvara.

Funkcija mora vratiti broj klijenata koje je dohvatila i obradila.

Zadatak riješiti pomoću kursora i handlera.

Napisati primjer poziva funkcije.

```
DELIMITER //
DROP FUNCTION IF EXISTS MarkVIP //
CREATE FUNCTION MarkVIP (p_sifKvar INT) RETURNS VARCHAR(255) DETERMINISTIC
BEGIN
  DECLARE c_sifKlijent, dohvaceno, oznaceno INT DEFAULT 0;
  DECLARE c_udio DECIMAL(4,2) DEFAULT 0;
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
    SELECT
      sifKlijent,
      SUM(OstvareniSatiRada) /
        (SELECT SUM(OstvareniSatiRada) FROM nalog
         WHERE sifKvar = p_sifKvar)*100 AS UDIO
      FROM nalog
      WHERE sifKvar = p_sifKvar
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      GROUP BY sifKlijent
      HAVING SUM(OstvareniSatiRada) /
        (SELECT SUM(OstvareniSatiRada) FROM NALOG
         WHERE sifKvar = p_sifKvar) * 100 > 5;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag=TRUE;

  IF p_sifKvar IN (SELECT DISTINCT sifKvar FROM nalog) THEN
    OPEN kurs;
      SELECT FOUND_ROWS() INTO dohvaceno;
      petlja:LOOP
        FETCH kurs INTO c_sifKlijent, c_udio ;
        IF flag=TRUE THEN
          LEAVE petlja;
        END IF;
        UPDATE klijent SET prezimeKlijent =
          CONCAT(prezimeKlijent, ' VIP -> % udio= ', c_udio )
          WHERE sifKlijent = c_sifKlijent ;
          SET oznaceno = oznaceno + 1;
      END LOOP;
    CLOSE kurs;
    RETURN CONCAT('Dohvaćeno ', dohvaceno, ', označeno ', oznaceno, ' VIP
klijenata.');
  ELSE
    RETURN CONCAT('Nepostojeća šifra kvara!');
  END IF;
END; //
DELIMITER ;

/* Poziv funkcije */
SELECT MarkVIP(13) AS poruka;
```

poruka
Dohvaćeno 9, označeno 9 VIP klijenata.

sifKlijent	imeKlijent	prezimeKlijent	pbrKlijent	pbrReg	datUnosKlijent	jmbgKlijent
1137	Jure	Ribarić	22000	22000	2012-11-02	2604000392304
1139	Niko	Marušić	48000	48000	2016-09-24	0609000173977
1140	Davor	Vurnek VIP -> % udio= 11.39	20000	20000	2015-10-26	1706000300802
1141	Zoran	Habajec	21000	21000	2017-11-24	0204998301807
1143	Davor	Voras	20000	10000	*(NULL)*	
1144	Zvonimir	Ozimec	21000	21000	2012-09-19	1611987390015
1145	Jurica	Bašić	10000	10000	2017-03-16	1608986361605
1147	Alenka	Vukojević	21000	21000	2018-07-22	2806989311906
1148	Antonijo	Javorina	10000	10000	2011-11-04	3107985330068

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

5.7. U bazi studenti: Napisati proceduru koja će svim studentima koji imaju evidentirane negativne ocjene iz određenog kolegija, dodati iza prezimena oznaku "Kol = K; Br_neg = N; OZNAKA;", pri čemu je:

- K šifra kolegija,
- N broj negativnih ocjena,
- oznaka se definira u ovisnosti o N, i to tako da:
  - za 1 negativnu ocjenu -> 'Još 2 pokušaja'
  - za 2 negativne ocjene -> 'Još 1 pokušaj'
  - za 3 negativne ocjene -> 'Komisijski ispit'
  - za više od 3 negativne ocjene -> 'Ponovni upis kolegija'

Ako student iz više kolegija ima evidentirane negativne ocjene, sve kolegije je potrebno evidentirati iza prezimena.

Procedura mora vratiti broj n-torki (kolegija po studentima) koje je dohvatila.

Zadatak riješiti pomoću kursora i handlera. Napisati primjer poziva procedure.

```
ALTER TABLE studenti
MODIFY COLUMN prezime VARCHAR(500);

DELIMITER //
DROP PROCEDURE IF EXISTS mark_negative //
CREATE PROCEDURE mark_negative()
BEGIN
  DECLARE c_id_kol, c_br_neg, dohvaceno INT;
  DECLARE c_jmbag, c_oznaka VARCHAR(50);
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
    SELECT idKolegij, jmbagStudent, COUNT(*) AS br_neg,
      CASE
        WHEN COUNT(*) = 1 THEN 'Još 2 pokušaja'
        WHEN COUNT(*) = 2 THEN 'Još 1 pokušaja'
        WHEN COUNT(*) = 3 THEN 'Komisijski ispit'
        ELSE 'Ponovni upis kolegija'
      END AS oznaka
      FROM ocjene
      WHERE ocjena = 1
      GROUP BY idKolegij, jmbagStudent;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag=TRUE;
  OPEN kurs;
    SELECT FOUND_ROWS() INTO dohvaceno;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    petlja:LOOP
      FETCH kurs INTO c_id_kol, c_jmbag, c_br_neg, c_oznaka;
      IF flag=TRUE THEN
        LEAVE petlja;
      END IF;
      UPDATE studenti
        SET prezime = CONCAT(prezime, ' Kol = ', c_id_kol, '; Br_neg = ',
c_br_neg, '; ',  c_oznaka, '; ')
      WHERE jmbag = c_jmbag;
    END LOOP;
  CLOSE kurs;
  SELECT dohvaceno AS dohvaceno_kolegij_po_studentima;
END; //
DELIMITER ;


/* Poziv procedure i provjere rezutlata */
/* Unosimo još jednu negativnu ocjenu studentu koji već ima negativnu ocjenu iz
drugog kolegija, kako bi testirati slučaj kada isti student ima negativnu ocjenu
iz više od 1 kolegija */
INSERT INTO ocjene VALUES (42, "0112034516", "2023-14-02", "12:00:00", 1);
SELECT * FROM studenti WHERE jmbag = "0112034516";


CALL mark_negative();
```

dohvaceno kolegija po studentima
19

```
SELECT COUNT(*) FROM studenti WHERE LENGTH(prezime) > 30;
```

count(*)
18

```
/* Za ukupno 18 studenata evidentirano je 19 kolegija sa negativnim ocjenama */


/* Izgled dijela tablice studenti nakon poziva procedure - pretpostavka je da će
nakon upisa podataka, duljina prezimena premašiti 30 znakova */
SELECT * FROM studenti WHERE LENGTH(prezime) > 30;
```

jmbag	ime	prezime
0035166854	Krešo	Šišić Kol = 20; Br_neg = 1; Još 2 pokušaja;
0035177990	Irena	Dabić Kol = 2; Br_neg = 1; Još 2 pokušaja;
0035192211	Karla	Brandis Kol = 7; Br_neg = 1; Još 2 pokušaja;
0036460022	Ivan	Matić Kol = 30; Br_neg = 1; Još 2 pokušaja;
0036460031	Dajana	Jerec Kol = 28; Br_neg = 1; Još 2 pokušaja;
0036499965	Lovro	Keglović Kol = 29; Br_neg = 2; Još 1 pokušaja;
0067405522	Dario	Veh Kol = 35; Br_neg = 1; Još 2 pokušaja;
0067408000	Nikola	Mucić Kol = 38; Br_neg = 1; Još 2 pokušaja;
0112034516	Denis	Vrdoljak Kol = 41; Br_neg = 1; Još 2 pokušaja; Kol = 42; Br_neg = 1; Još 2 pokušaja;
0128050853	Eva	Delač Kol = 1; Br_neg = 1; Još 2 pokušaja;
0246020326	Melita	Rukavina Kol = 8; Br_neg = 1; Još 2 pokušaja;
0246025633	Zvonimir	Kolar Kol = 7; Br_neg = 2; Još 1 pokušaja;
0246031254	Petra	Vujić Kol = 5; Br_neg = 2; Još 1 pokušaja;
0246035265	Katarina	Dujmović Kol = 3; Br_neg = 1; Još 2 pokušaja;
0283021325	Robert	Stanić Kol = 9; Br_neg = 1; Još 2 pokušaja;
0313006332	Karla	Rolj Kol = 13; Br_neg = 2; Još 1 pokušaja;

...

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

5.8. U bazi autoradionica: Napisati proceduru koja će za zadana 2 kvartala ispisati sve odjele koji su ostvarili ili sumu sati između 20 i 50 ili imali više od 5 naloga. Dodatno, u ispisu je potrebno naziv odjela postaviti na velika slova onim odjelima kod kojih je suma sati veća od 50.

Ispis mora sadržavati:

- id odjela,
- naziv odjela,
- sumu ostvarenih sati,
- standardnu devijaciju ostvarenih sati i
- broj naloga.

Sortirati po sumi sati padajući.

Zadatak riješiti pomoću kursora i privremene tablice, bez korištenja handlera.

Napisati primjer poziva procedure.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS nalozi_kvartal$$
CREATE PROCEDURE nalozi_kvartal (p_kvart1 INT, p_kvart2 INT)
BEGIN
  DECLARE c_nazivOdj VARCHAR(255) DEFAULT NULL;
  DECLARE c_quarter, c_brNal, dohvaceno INT DEFAULT NULL;
  DECLARE c_SumaSati, c_StdSati DECIMAL(8,2) DEFAULT 0;
  DECLARE i INT DEFAULT 0;
  DECLARE kur CURSOR FOR
    SELECT QUARTER(datPrimitkaNalog), o.nazivOdjel,
        SUM(OstvareniSatiRada) AS suma_sati,
        ROUND(STD(OstvareniSatiRada),2) AS std_sati, COUNT(*) AS br_naloga
      FROM nalog n NATURAL JOIN radnik r NATURAL JOIN odjel o
      WHERE QUARTER(datPrimitkaNalog) IN (p_kvart1, p_kvart2)
      GROUP BY QUARTER(datPrimitkaNalog), o.nazivOdjel, o.sifOdjel
      HAVING COUNT(*) > 5 OR SUM(OstvareniSatiRada) BETWEEN 20 AND 50
      ORDER BY 1 ASC, 3 DESC;

  DROP TEMPORARY TABLE IF EXISTS tmp;
  CREATE TEMPORARY TABLE tmp (kvartal INT, naziv_odjel VARCHAR(50),
        suma_sati DECIMAL(8,2), st_devijacija DECIMAL(4,2), broj_naloga
INT(11));
  OPEN kur;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  SELECT FOUND_ROWS() INTO dohvaceno;
  WHILE i < dohvaceno DO
    FETCH kur INTO c_quarter, c_nazivOdj, c_SumaSati, c_StdSati, c_brNal;
    IF c_SumaSati > 50 THEN
      SET c_nazivOdj = UPPER(c_nazivOdj);
    END IF;

    INSERT INTO tmp (kvartal, naziv_odjel, suma_sati,
        st_devijacija, broj_naloga)
      VALUES (c_quarter, c_nazivOdj , c_SumaSati, c_StdSati, c_brNal);
    SET i=i+1;
  END WHILE;
  CLOSE kur;
  SELECT * FROM tmp;
END; $$
DELIMITER ;

/* Poziv procedure */
CALL nalozi_kvartal(1,2);
```

kvartal	naziv_odjel	suma_sati	st_devijacija	broj_naloga
1	LIMARIJA	92.00	2.79	17
1	VULKANIZACIJA	72.00	2.74	12
1	MEHANIKA	69.00	2.91	14
1	ELEKTRIKA	52.00	3.03	10
1	Balansiranje guma	46.00	2.92	11
1	Skladište	42.00	1.51	7
1	Mulde	38.00	2.44	10
1	Laki motori	32.00	2.92	6
1	Optika	27.00	1.50	5
1	Klima	22.00	2.50	5
1	Alarmi	20.00	2.36	3
2	ELEKTRIKA	85.00	2.49	15
2	MEHANIKA	72.00	1.92	11
2	LIMARIJA	70.00	2.61	12
2	MULDE	67.00	2.33	12
2	Vulkanizacija	42.00	2.41	11
2	Balansiranje guma	32.00	2.38	7
2	Klima	30.00	3.28	7
2	Skladište	28.00	2.24	4
2	Elektropokretači	21.00	2.45	3

5.9. U bazi autoradionica: Napisati funkciju koja će za zadani period (godina_od, godina_do) označiti sve radnike čiji je udio odrađenih broja naloga u ukupno odrađenim nalozima za zadani period veći od 1%.

Tablici radnik dodati novi atribut *udio_u_nalozima* odgovarajućeg tipa podatka. Radnicima po tom kriteriju u novostvoreni atribut *udio_u_nalozima* upisati izračunatu vrijednost u %. Osigurati da funkcija resetira (postavi na vrijednost NULL) atribut *udio_u_nalozima* prije nego krene upisivati vrijednosti za zadani period.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Ako je godina_od veća od godine_do potrebno je vratiti poruku 'Godina OD je veća od godine DO'.

Funkcija mora vratiti broj klijenata koje je dohvatila i obradila.

Zadatak riješiti pomoću kursora i handlera.

Napisati primjer poziva funkcije.

```
ALTER TABLE radnik ADD COLUMN udio_u_nalozima DECIMAL(6,2);


DELIMITER //
DROP FUNCTION IF EXISTS udio_odradenih_naloga //
CREATE FUNCTION udio_odradenih_naloga (p_god_od INT, p_god_do INT) RETURNS
VARCHAR(255) DETERMINISTIC
BEGIN
  DECLARE c_sifRad, dohvaceno, oznaceno INT default 0;
  DECLARE c_udio decimal(4,2) DEFAULT 0;
  DECLARE flag BOOL DEFAULT FALSE;
  DECLARE kurs CURSOR FOR
    SELECT sifRadnik, COUNT(*) /
      (SELECT COUNT(*) FROM nalog
        WHERE YEAR(datPrimitkaNalog) BETWEEN p_god_od AND p_god_do)*100
        AS UDIO
      FROM nalog
      WHERE YEAR(datPrimitkaNalog) BETWEEN p_god_od AND p_god_do
      GROUP BY sifRadnik
      HAVING COUNT(*) /
        (SELECT COUNT(*) FROM nalog
          WHERE YEAR(datPrimitkaNalog) BETWEEN p_god_od AND p_god_do)*100 > 1;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag = TRUE;
  IF p_god_od <= p_god_do THEN
    UPDATE radnik SET udio_u_nalozima = NULL;
    OPEN kurs;
      SELECT FOUND_ROWS() INTO dohvaceno;
      petlja:LOOP
        FETCH kurs INTO c_sifRad, c_udio ;
        IF flag=TRUE THEN
          LEAVE petlja;
        END IF;
        UPDATE radnik SET udio_u_nalozima = c_udio
          WHERE sifRadnik = c_sifRad ;
        SET oznaceno = oznaceno + 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      END LOOP;
    CLOSE kurs;
    RETURN CONCAT('Dohvaćeno ', dohvaceno, ', označeno ', oznaceno, '
radnika.');
  ELSE
    RETURN CONCAT('Godina OD je veća od godine DO');
  END IF;
END; //
DELIMITER ;
```

```
/* Poziv funkcije */
SELECT udio_odradenih_naloga(2018,2019) AS poruka;
```

poruka
Dohvaćeno 27, označeno 27 radnika.

```
SELECT * FROM radnik;
/* Prikaz dijela radnika */
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice	udio_u_nalozima
122	Tin	Grabovac	49000	19	1.38	2200.00	4.41
126	Zdravko	Budišin	53000	27	0.56	2200.00	*(NULL)*
130	Antun Ivan	Jonjić	10000	13	2.69	2200.00	*(NULL)*
134	Želimir	Somrak	33000	8	1.74	2200.00	*(NULL)*
138	David	Petković	35000	18	2.63	2200.00	*(NULL)*
142	Alan	Ostojčić	10000	2	1.92	2200.00	10.29
146	Franjo	Vidaković	20000	17	1.72	2200.00	1.47
150	Ana	Kukec	20000	24	1.84	2200.00	*(NULL)*
154	Žarko	Dubinko	10000	24	0.04	2200.00	*(NULL)*
158	Davorka	Volarić	10000	10	0.69	2200.00	*(NULL)*
162	Želimir	Prester	34550	20	0.34	2200.00	*(NULL)*
166	Nino	Lipec	10000	15	0.63	2200.00	5.88
171	Dinko	Žubrinić	10000	15	0.12	2200.00	*(NULL)*
175	Željko	Kovačević	10000	18	0.70	2200.00	*(NULL)*

…

```
SELECT udio_odradenih_naloga(2023,2018) AS poruka;
```

poruka
Godina OD je veća od godine DO

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

6. CTE

6.1. U bazi autoradionica: Izraditi upit uz korištenje CTE-a (*Common Table Expression*) koji će po mjestima prebivanja radnika sadržavati slijedeće podatke:

- poštanski broj mjesta,
- naziv mjesta,
- broj naloga koji su odrađeni u dotičnom mjestu,
- prosječne sate rada vezane uz mjesta,
- ukupne sate rada,
- te šifru i naziv županije kojoj mjesto pripada.

Ispis sortirati uzlazno po poštanskom broju mjesta.

pbrMjesto	nazivMjesto	br_naloga	pros_sati	sum_sati	sifZupanija	nazivZupanija
10000	Zagreb	291	5.4570	1588	21	Grad Zagreb
20000	Dubrovnik	18	5.2778	95	19	Dubrovačko-neretvanska
21000	Split	41	5.1951	213	17	Splitsko-dalmatinska
31500	Našice	5	7.4000	37	14	Osječko-baranjska
33000	Virovitica	6	5.1667	31	10	Virovitičko-podravska
34000	Požega	9	6.4444	58	11	Požeško-slavonska
48000	Koprivnica	43	5.6512	243	6	Koprivničko-križevačka
49000	Krapina	13	3.7692	49	2	Krapinsko-zagorska
52000	Pazin	1	10.0000	10	18	Istarska

```
/* Unutar CTE-a obavljamo izračun agregiranih podataka i to nad minimalnim
brojem tablica (samo tablica nalog) kako bi se optimizirala brzina upita, te
naknadno u nastavku spajamo sa ostatkom tablica (mjesto i zupanija) za dobivanje
ostalih atributa u ispisu */
WITH ag_mjesto AS
  (SELECT r.pbrStan, COUNT(*) AS br_naloga,
      AVG(OstvareniSatiRada) AS pros_sati,
      SUM(OstvareniSatiRada) AS sum_sati
    FROM nalog n NATURAL JOIN radnik r
    GROUP BY r.pbrStan)
SELECT m.pbrMjesto, m.nazivMjesto, ag.br_naloga, ag.pros_sati, ag.sum_sati,
  m.sifZupanija, z.nazivZupanija
  FROM mjesto m
    JOIN ag_mjesto ag ON m.pbrMjesto= ag.pbrStan
    JOIN zupanija z ON m.sifZupanija= z.sifZupanija
  ORDER BY 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

6.2. U bazi autoradionica: Izraditi upit uz korištenje CTE-a koji će po kvartalima i klijentima sadržavati podatke:

- kvartal,
- šifra klijenta,
- broj naloga,
- prosječan iznos sati po klijentima i kvartalima,
- sumu sati po klijentima i kvartalima,
- ime klijenta,
- prezime klijenta i
- JMBAG klijenta.

Ispis sortirati prema slici (po kvartalu i šifri klijenta uzlazno).

kvartal	sifKlijent	br_naloga	pros_sati	sum_sati	imeKlijent	prezimeKlijent	jmbgKlijent
1	1120	1	8.0000	8	Zdenko	Kolac	2401981330186
1	1132	1	2.0000	2	Nikica	Arunović	1305999330006
1	1145	4	4.7500	19	Jurica	Bašić	1608986361605
1	1147	2	5.5000	11	Alenka	Vukojević	2806989311906
1	1152	1	3.0000	3	Božidar	Tomić	3011993395074
1	1159	1	7.0000	7	Katarina	Dobrina	1011994330169
1	1160	1	6.0000	6	Majo	Musić	0907989303508
1	1165	1	10.0000	10	Željka	Morić	1503989392119
1	1167	1	1.0000	1	Mirko	Teklić	1110985310615
1	1172	5	5.6000	28	Ninoslav	Novak	2105000383309
1	1173	1	2.0000	2	Gordan	Boršić	0607982334016
1	1178	1	2.0000	2	Kristijan	Vrabec	2302993361315
1	1180	5	5.2000	26	Iva	Mioč	0506000340008

```
WITH
  cte_klijent_kvartal AS
     (SELECT QUARTER(datPrimitkaNalog) AS kvartal,
          sifKlijent,
          COUNT(*) AS br_naloga,
          AVG(OstvareniSatiRada) AS pros_sati,
          SUM(OstvareniSatiRada) AS sum_sati
        FROM  nalog
        GROUP BY quarter(datPrimitkaNalog), sifKlijent)
SELECT a.*,  k.imeKlijent, k.prezimeKlijent, k.jmbgKlijent
  FROM klijent k JOIN cte_klijent_kvartal a ON k.sifKlijent = a.sifKlijent
  ORDER BY 1, 2;
```

6.3. U bazi autoradionica: Izraditi pogled uz korištenje CTE-a koji će po kolegijima prikazati sljedeće podatke:

- id kolegija,
- naziv kolegija,
- broj položenih ispita,

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

- najmanju i najveću ocjenu iz kolegija (uzeti samo pozitivne ocjene).

Prikazati samo one kolegije koji su imali više od 3 pozitivna ispita. Sortirati po broju ispita padajući i maksimalnoj ocjeni rastući.

id	naziv	br_ispita	min_ocj	max_ocj
31	Poslovno komuniciranje	8	2	5
18	Refrakcija i binokularni vid I	8	2	5
20	Pogon zrakoplova I	7	3	5
35	Mikroekonomija I	7	2	5
33	Osnove poduzetništva	7	2	5
40	Sredstva pomorskog prometa	6	2	5
12	Javna nabava	6	2	5
13	Ekonomika turizma	6	2	5
22	Mjera i integral	6	2	5
17	Anatomija i fiziologija oka	6	2	5
39	Tehnička mehanika 2	5	2	3
1	Baze podataka	5	2	4
41	Tereti u prometu	5	2	4
28	Energijske tehnologije	5	2	4
43	Tjelesna i zdravstvena kultura 2	5	5	5

```
WITH CTE_KOL AS
  (SELECT idKolegij,
      COUNT(*) AS br_ispita,
      MIN(ocjena) min_ocj,
      MAX(ocjena) max_ocj
    FROM ocjene
    WHERE ocjena > 1
    GROUP BY idKolegij
    HAVING COUNT(*) > 3)
SELECT k.id, k.naziv, ck.br_ispita, ck.min_ocj, ck.max_ocj
  FROM kolegiji k INNER JOIN CTE_KOL ck ON k.id = ck.idKolegij
  ORDER BY 3 DESC, 5;
```

6.4. U bazi autoradionica: Izraditi upit uz korištenje CTE-a koji će po odjelima sadržavati sljedeće podatke:

- šifru i naziv odjela,
- broj naloga odrađenih na odjelu,
- prosječan iznos sati na odjelu,
- ukupnu sumu sati te
- šifru i naziv nadređenog odjela.

Ispis sortirati po nazivu odjela.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

sifOdjel	nazivOdjel	br_naloga	pros_sati	sum_sati	sifNadOdjel	nazNadrOdj
14	Alarmi	11	5.2727	58	3	Elektrika
20	Amortizeri	6	5.1667	31	5	Optika
16	Balansiranje	5	5.2000	26	4	Vulkanizacija
19	Balansiranje guma	27	4.8148	130	5	Optika
3	Elektrika	53	5.4717	290	1	Auto centar kod veselog Zagorca
11	Elektropokretači	6	6.1667	37	3	Elektrika
13	Klima	25	5.1200	128	3	Elektrika
27	Kvačila	5	7.4000	37	6	Mehanika
24	Laki motori	25	4.5600	114	6	Mehanika
2	Limarija	62	5.5161	342	1	Auto centar kod veselog Zagorca
6	Mehanika	54	5.7222	309	1	Auto centar kod veselog Zagorca
22	Mulde	41	5.5610	228	5	Optika
5	Optika	19	5.3158	101	1	Auto centar kod veselog Zagorca
15	Skladište	25	6.4000	160	4	Vulkanizacija
26	Teški motori	4	6.0000	24	6	Mehanika
4	Vulkanizacija	45	5.0000	225	1	Auto centar kod veselog Zagorca
17	Zamjena guma	6	6.1667	37	4	Vulkanizacija
9	Zaštita od korozije	1	10.0000	10	2	Limarija

```
WITH ag_odjel AS
  (SELECT r.sifOdjel,
      COUNT(*) AS br_naloga,
      AVG(OstvareniSatiRada) AS pros_sati,
      SUM(OstvareniSatiRada) AS sum_sati
    FROM nalog n NATURAL JOIN radnik r
    GROUP BY r.sifOdjel)
SELECT o.sifOdjel, o.nazivOdjel, ag.br_naloga, ag.pros_sati, ag.sum_sati,
o.sifNadOdjel, nadredeniOdjel.nazivOdjel AS nazNadrOdj
  FROM odjel o
  NATURAL JOIN ag_odjel ag
  JOIN odjel nadredeniOdjel ON o.sifNadOdjel = nadredeniOdjel.sifOdjel
  ORDER BY 2;
```

6.5. U bazi autoradionica: Izraditi upit uz korištenje CTE-a koji će po godinama i kvarovima sadržavati podatke prema slici. Ispis sortirati po godini i šifri kvara rastući.

god	sifKvar	br_naloga	pros_sati	sum_sati	nazivKvar	sifOdjel	brojRadnika	satiKvar
2011	1	4	6.0000	24	Zamjena blatobrana	6	2	2
2011	2	1	6.0000	6	Zamjena krova	6	4	2
2011	6	2	5.5000	11	Promjena osovine	6	1	2
2011	7	4	6.7500	27	Izmjena amortizera	6	4	2
2011	10	2	4.5000	9	Zamjena vozačkih sjedala	6	4	2
2011	11	1	1.0000	1	Zamjena pragova	7	8	4
2011	12	1	7.0000	7	Zamjena krovnog prozora	7	7	1
2011	13	1	4.0000	4	Zamjena blatobrana	7	4	3
2011	14	1	5.0000	5	Uštimavanje pokretnog krova	7	7	2
2011	17	1	7.0000	7	Ugradnja prednjeg znaka od	7	3	3
2011	18	2	6.0000	12	Zamjena klipa	10	6	4

```
WITH agr AS
  (SELECT YEAR(datPrimitkaNalog) AS god,
      sifKvar,
      COUNT(*) AS br_naloga,
      AVG(OstvareniSatiRada) AS pros_sati,
      SUM(OstvareniSatiRada) AS sum_sati
    FROM  nalog
    GROUP BY YEAR(datPrimitkaNalog), sifKvar)
SELECT a.*, k.nazivKvar, k.sifOdjel, k.brojRadnika, k.satiKvar
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
FROM kvar k JOIN agr a ON k.sifKvar = a.sifKvar
ORDER BY 1, 2;
```

6.6. U bazi autoradionica: Izraditi upit uz korištenje CTE-a koji će prikazati sve odrađene naloge po radnicima kao i podatke odjela kojima radnici pripadaju. Prikazati:

- šifru radnika,
- ime i prezime radnika spojeno,
- broj odrađenih naloga za dotičnog radnika i
- prosječne sate rada po radniku.

U istom retku prikazati i podatke o odjelu, odnosno:

- šifru odjela,
- naziv odjela,
- sveukupan broj naloga na odjelu i
- prosječne sate rada u odjela kojemu radnik pripada.

Sortirati po broju odjela i broju naloga radnika silazno

sifRadnik	ime_prez	BR_NAL_RAD	PROS_SATI_RAD	sifOdjel	nazivOdjel	BR_NAL_ODJEL	PROS_SATI_ODJEL
289	Marina Carek	27	5.7407	3	Elektrika	53	5.4717
514	Damjan Dolar	27	5.4074	6	Mehanika	54	5.7222
501	Davorin Rodinis	26	5.1923	3	Elektrika	53	5.4717
379	Mia Stojić	22	5.0909	4	Vulkanizacija	45	5.0000
244	Zlatko Tomašek	22	5.3182	22	Mulde	41	5.5610
142	Alan Ostojčić	21	5.9048	2	Limarija	62	5.5161
166	Nino Lipec	20	6.2000	15	Skladište	25	6.4000
277	Silvije Szekeres	19	5.3158	5	Optika	19	5.3158
456	Slavica Haban	19	5.8421	22	Mulde	41	5.5610
469	Ivana Dumančić	18	5.6667	2	Limarija	62	5.5161
411	Kristian Damjančić	17	6.3529	6	Mehanika	54	5.7222
334	Sandra Mihaljević	15	5.1333	13	Klima	25	5.1200
232	Dubravko Josipović	15	4.6000	4	Vulkanizacija	45	5.0000
154	Žarko Dubinko	14	3.7143	24	Laki motori	25	4.5600
122	Tin Grabovac	13	3.7692	19	Balansiranje guma	27	4.8148

```
WITH
  CTE_rad AS
    (SELECT sifRadnik,
        COUNT(*) AS BR_NAL_RAD,
        AVG(OstvareniSatiRada) PROS_SATI_RAD
      FROM nalog
      GROUP BY sifRadnik),
  CTE_odjel AS
    (SELECT r.sifOdjel, COUNT(*) AS BR_NAL_ODJEL,
        AVG(OstvareniSatiRada) PROS_SATI_ODJEL
      FROM nalog n NATURAL JOIN radnik r
      GROUP BY r.sifOdjel)
  SELECT  r.sifRadnik,
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    CONCAT_WS(' ',r.imeRadnik,r.prezimeRadnik) ime_prez,
    cr.BR_NAL_RAD,
    cr.PROS_SATI_RAD,
    co.sifOdjel,
    o.nazivOdjel,
    co.BR_NAL_ODJEL,
    co.PROS_SATI_ODJEL
FROM  radnik r
INNER JOIN CTE_rad cr ON r.sifRadnik = cr.sifRadnik
INNER JOIN CTE_odjel co  ON co.sifOdjel = r.sifOdjel
INNER JOIN odjel o ON o.sifOdjel = co.sifOdjel
ORDER BY 3 DESC;
```

6.7. U bazi studenti: Izraditi upit uz korištenje CTE-a koji će po nastavnicima prikazati sljedeće podatke: ime i prezime nastavnika, njegov JMBAG, broj ispita, broj negativnih i pozitivnih ocjena kao i najmanju i najveću ocjenu koju je nastavnik dao i MAX ocjenu. Prikazati samo one nastavnike koji su imali više od 3 ispita (3 ocjene iz kolegija). Sortirati po prosječnoj ocjeni padajući.

ime	prezime	jmbgNastavnik	br_ispita	br_neg	br_poz	MIN_ocj	MAX_ocj	AVG_ocj
Jelena	Nikolić	1907983383326	4	0	4	2	5	4.0000
Stjepan	Kovačević	0608955311207	12	0	12	2	5	3.8333
Nikola	Bačić	1908987173977	12	0	12	2	5	3.8333
Stjepan	Lukić	0804969306801	6	0	6	2	5	3.8333
Morana	Novak	2409952381906	6	0	6	2	5	3.8333
Mila	Ilić	0611973330036	5	0	5	3	5	3.8000
Frano	Vukelić	1212953391826	5	0	5	2	5	3.8000
Sonja	Duvnjak	2306978362104	5	0	5	2	5	3.8000
Željka	Perić	2511949330185	5	0	5	2	5	3.8000
Petra	Jukić	0110959390037	8	0	8	3	4	3.7500
Kristijan	Kolar	0807977310602	8	0	8	2	5	3.7500
Renato	Topić	0610975181955	5	0	5	2	5	3.6000
Ankica	Čeh	1707963340011	4	0	4	2	5	3.5000
Zvonimir	Ivanković	1010962363604	4	0	4	2	5	3.5000
Josipa	Jakšić	0812942370017	6	1	5	1	5	3.5000
Vesna	Ilić	3105981300707	6	1	5	1	5	3.5000
Cvjetko	Stojanovi	2011979362930	4	0	4	2	5	3.5000

```
WITH CTE_NASTAVNICI AS
  (SELECT i.jmbgNastavnik, COUNT(*) AS br_ispita,
      SUM(CASE WHEN o.ocjena = 1 THEN 1 ELSE 0 END) AS br_neg,
      SUM(CASE WHEN o.ocjena > 1 THEN 1 ELSE 0 END) AS br_poz,
      MIN(o.ocjena) AS MIN_ocj,
      MAX(o.ocjena) AS MAX_ocj,
      AVG(o.ocjena) AS AVG_ocj
    FROM ocjene o
      JOIN kolegiji k ON o.idKolegij=k.id
      JOIN izvrsitelji i ON k.id= i.idKolegij
    GROUP BY i.jmbgNastavnik
    HAVING COUNT(*) > 3
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    ORDER BY 7 DESC)
SELECT n.ime, n.prezime, cn.*
 FROM nastavnici n INNER JOIN CTE_NASTAVNICI cn ON n.jmbg=cn.jmbgNastavnik
 ORDER BY 9 DESC;
```

6.8. U bazi studenti: Izraditi upit uz korištenje CTE-a koji će po kolegijima sadržavati podatke prema slici (sortirati padajući po standardnoj devijaciji).

naz_koleg	idKolegij	pros_kolegija	stand_dev	br_1	br_2	br_3	br_4	br_5	Br_ispita	idSmjer	pros_smjera
Marketing destinacije	38	3.3333	1.7	1	2	0	0	3	6	19	3.5455
Betonske konstrukcije	7	2.7143	1.58	3	0	1	2	1	7	4	3.0000
Matematička analiza 1	23	2.4000	1.5	2	1	1	0	1	5	12	2.8750
Kvantitativne metode u ma:	36	3.5000	1.5	0	1	0	0	1	2	18	3.1000
Uvod u UNIX sustave	2	2.7500	1.48	1	1	1	0	1	4	1	3.3889
Poslovna etika	9	3.1667	1.46	1	1	2	0	2	6	5	3.3000
Ekonomika turizma	13	2.6250	1.41	2	3	0	2	1	8	8	2.8182
Fizika	5	2.5714	1.4	2	2	1	1	1	7	3	2.6364
Operacijski sustavi	15	3.5000	1.38	1	0	2	1	2	6	7	3.5000
Osnove poduzetništva	33	4.0000	1.31	0	2	0	1	4	7	17	3.8333
Osnove elektrotehnike	8	2.7500	1.3	1	1	0	2	0	4	3	2.6364
Napredne baze podataka	46	3.3333	1.25	0	1	1	0	1	3	1	3.3889
Sportski marketing	14	3.3333	1.25	0	1	1	0	1	3	8	2.8182
Uvod u kvantnu fiziku	24	3.6667	1.25	0	1	0	1	1	3	12	2.8750
Fizika	4	4.0000	1.22	0	1	0	1	2	4	2	3.3750
Opća kemija	25	4.0000	1.22	0	1	0	1	2	4	13	3.2500
Mikroekonomija I	35	3.0000	1.22	1	2	2	2	1	8	18	3.1000
Izrada web-projekta	30	2.8333	1.21	1	1	3	0	1	6	15	2.6667

```
WITH
  koleg_ocjene AS
    (SELECT idKolegij,AVG(ocjena) pros_kolegija,
        round(STDDEV(ocjena),2) stand_dev,
        SUM(CASE WHEN ocjena = 1 THEN 1 ELSE 0 END) AS br_1,
        SUM(CASE WHEN ocjena = 2 THEN 1 ELSE 0 END) AS br_2,
        SUM(CASE WHEN ocjena = 3 THEN 1 ELSE 0 END) AS br_3,
        SUM(CASE WHEN ocjena = 4 THEN 1 ELSE 0 END) AS br_4,
        SUM(CASE WHEN ocjena = 5 THEN 1 ELSE 0 END) AS br_5,
        COUNT(*) AS Br_ispita
      FROM  ocjene
      GROUP BY  idKolegij),
  smjer_ocjene AS
    (SELECT idSmjer, AVG(o.ocjena) AS pros_smjera
      FROM ocjene o JOIN kolegiji k ON o.idKolegij = k.id
      GROUP BY idSmjer)
SELECT k.naziv AS naz_koleg,  ko.*, k.idSmjer, so.pros_smjera
  FROM kolegiji k
    JOIN koleg_ocjene ko ON ko.idKolegij = k.id
    JOIN smjer_ocjene so ON so.idSmjer = k.idSmjer
  ORDER BY 4 DESC;
```

6.9. U bazi autoradionica: Izraditi upit uz korištenje CTE-a koji će prikazati sve odrađene naloge po radnicima i šiframa kvarova. Prikazati: šifru radnika, ime i prezime radnika

spojeno, broj naloga i prosječno vrijeme po radniku i šifri kvara. U istom retku prikazati šifru i naziv kvara te sveukupan broj naloga po kvaru i prosječne sate po kvaru. Sortirati po broju naloga radnika silazno.

sifRadnik	ime_prez	BR_NAL_R	PROS_SATI_R	sifKvar	nazivKvar	BR_NAL_K	PROS_SATI_K
166	Nino Lipec	18	6.1667	1	Zamjena blatobrana	18	6.1667
514	Damjan Dolar	13	5.9231	10	Zamjena vozačkih sjedala	34	5.0588
330	Vinko Kecman	12	5.4167	14	Uštimavanje pokretnog krova	12	5.4167
501	Davorin Rodinis	12	4.2500	25	Zamjena rashladne tekućine	33	5.2424
334	Sandra Mihaljević	10	5.1000	2	Zamjena krova	26	5.6923
411	Kristian Damjančić	10	7.0000	26	Ravnanje prednje vilice	20	6.4000
289	Marina Carek	9	6.4444	15	Popravak oštećenih mjesta n	11	5.6364
122	Tin Grabovac	9	3.8889	27	Zamjena blatobrana	16	4.0625
289	Marina Carek	9	5.0000	7	Izmjena amortizera	31	4.9032
142	Alan Ostojčić	9	6.0000	12	Zamjena krovnog prozora	18	6.5556
501	Davorin Rodinis	9	7.1111	12	Zamjena krovnog prozora	18	6.5556
379	Mia Stojić	8	5.3750	10	Zamjena vozačkih sjedala	34	5.0588
379	Mia Stojić	8	4.5000	25	Zamjena rashladne tekućine	33	5.2424
391	Marko Hazdovac	8	4.7500	29	Ugradnja sportskog pojasa	8	4.7500
244	Zlatko Tomašek	8	4.3750	7	Izmjena amortizera	31	4.9032
232	Dubravko Josipović	8	3.8750	10	Zamjena vozačkih sjedala	34	5.0588
142	Alan Ostojčić	8	6.2500	25	Zamjena rashladne tekućine	33	5.2424

```
WITH
  CTE_rad AS
    (SELECT sifRadnik, sifKvar,
       COUNT(*) AS BR_NAL_R,
       AVG(OstvareniSatiRada) PROS_SATI_R
      FROM nalog
      GROUP BY sifRadnik, sifKvar),
  CTE_kvar AS
    (SELECT sifKvar, COUNT(*) AS BR_NAL_K, AVG(OstvareniSatiRada) PROS_SATI_K
      FROM nalog
      GROUP BY sifKvar)
  SELECT  r.sifRadnik, concat_ws(' ', r.imeRadnik, r.prezimeRadnik)
    ime_prez, cr.BR_NAL_R, cr.PROS_SATI_R,
    cr.sifKvar, k.nazivKvar, ck.BR_NAL_K, ck.PROS_SATI_K
    FROM radnik r
      INNER JOIN CTE_rad cr  on r.sifRadnik = cr.sifRadnik
      INNER JOIN CTE_kvar ck  on cr.sifKvar = ck.sifKvar
      INNER JOIN kvar k on k.sifKvar = ck.sifKvar
    ORDER BY 3 DESC;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

7. Pogledi

7.1. U bazi autoradionica: Izraditi pogled koji će vraćati popis svih naloga zaprimljenih u drugoj polovini prosinca bilo koje godine. Osim svih podataka iz tablice nalog, pogled mora vraćati i (samo) ime i prezime klijenta.

Ispisati cijeli skup podataka kojeg vraća pogled (napisati poziv pogleda).

```
CREATE OR REPLACE VIEW nalozi_za_iducu_godinu AS
  SELECT nalog.*, imeKlijent, prezimeKlijent
    FROM nalog JOIN klijent ON nalog.sifKlijent = klijent.sifKlijent
    WHERE DAY(datPrimitkaNalog) > 15 AND MONTH(datPrimitkaNalog)=12;


SELECT * FROM nalozi_za_iducu_godinu;
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada	imeKlijent	prezimeKlijent
1217	34	514	2014-12-18	3	8	Lovre	Huk
1235	7	289	2014-12-17	3	9	Lea	Kamenečki
1266	6	142	2016-12-23	4	5	Eduard	Petrović
1312	25	501	2015-12-23	1	1	Nina	Kućan
1343	18	436	2017-12-30	2	5	Davorka	Kralj
1356	6	142	2011-12-31	3	6	Matija	Bujas
1393	27	122	2019-12-30	3	3	Stjepan	Tomić
1414	32	289	2017-12-19	1	7	Tin	Šipek
1421	26	411	2017-12-26	1	7	Ljiljana	Jokić
1441	10	514	2015-12-19	1	9	Majo	Meglaj
1498	34	514	2014-12-27	2	1	Janko	Jakopović
1259	1	166	2011-12-27	1	10	Ljiljana	Bulić

...

7.2. U bazi autoradionica: Izraditi pogled koji će ispisati za svakog klijenta:

- Ime klijenta
- Prezime klijenta
- Naziv županije stanovanja
- Naziv mjesta stanovanja
- Naziv županije u kojoj su registrirali vozilo
- Naziv mjesta u kojem su registrirali vozilo

Koristeći izrađeni pogled potrebno je:

a) Ispisati cijeli skup podataka kojeg vraća pogled

b) Ispisati klijente koji imaju različitu županiju stanovanja i registracije

c) Ispisati klijente koji imaju istu županiju stanovanja i registracije, ali različita mjesta

```
CREATE OR REPLACE VIEW mjesta_i_zupanije_klijenata AS
  SELECT imeKlijent,
      prezimeKlijent,
      z_k.nazivZupanija AS zupanija_klijenta,
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      m_k.nazivMjesto AS mjesto_klijenta,
      z_r.nazivZupanija AS zupanija_registracije,
      m_r.nazivMjesto AS mjesto_registracije
    FROM klijent JOIN mjesto m_k ON klijent.pbrKlijent = m_k.pbrMjesto
    JOIN zupanija z_k ON m_k.sifZupanija = z_k.sifZupanija
    JOIN mjesto m_r ON klijent.pbrReg = m_r.pbrMjesto
    JOIN zupanija z_r ON m_r.sifZupanija = z_r.sifZupanija;

/* a */
SELECT * FROM mjesta_i_zupanije_klijenata;

/* b */
SELECT * FROM mjesta_i_zupanije_klijenata
  WHERE zupanija_registracije != zupanija_klijenta;

/* c */
SELECT *
  FROM mjesta_i_zupanije_klijenata
  WHERE mjesto_klijenta != mjesto_registracije
  AND zupanija_registracije = zupanija_klijenta;
```

imeKlijent	prezimeKlijent	zupanija_klijenta	mjesto_klijenta	zupanija_registracije	mjesto_registracije
Nataša	Cerjan	Virovitičko-podravska	Slatina	Virovitičko-podravska	Pitomača
Karlo	Krsnik	Grad Zagreb	Zagreb-Sloboština	Grad Zagreb	Zagreb
Igor	Bogati	Grad Zagreb	Sveti Ivan Zelina	Grad Zagreb	Zagreb

7.3. U bazi autoradionica: Izraditi pogled koji će kronološki vraćati događaje unesenih klijenata i zaprimljenih naloga. Događaje je potrebno sortirati od najnovijeg prema najstarijem, pa nakon toga uzlazno po prezimenu. Očekujemo ispis kako slijedi u nastavku.

Unesen klijent	imeKlijent	prezimeklijent	datum
Zaprimljen nalog	Helena	Octenjak	2020-02-26
Zaprimljen nalog	Majo	Meglaj	2020-02-19
Zaprimljen nalog	Emil	Sertić	2020-02-18
Zaprimljen nalog	Tomica	Lulić	2020-02-09
Unesen klijent	Nataša	Osredečki	2020-02-09
Zaprimljen nalog	Borislav	Gobin	2020-02-03
Unesen klijent	Helena	Mandić	2020-02-02
Unesen klijent	Mirko	Kadić	2020-01-20
Zaprimljen nalog	Božidar	Cetto	2020-01-18
Zaprimljen nalog	Slavko	Hrvojević	2020-01-16
Zaprimljen nalog	Želimir	Babić	2020-01-13

```
CREATE OR REPLACE VIEW kronologijaNalogKlijenata AS
  SELECT 'Unesen klijent', imeKlijent, prezimeklijent, datUnosKlijent AS datum
    FROM klijent
  UNION ALL
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  SELECT 'Zaprimljen nalog', imeKlijent, prezimeklijent, datPrimitkaNalog AS
datum
      FROM nalog JOIN klijent ON nalog.sifKlijent = klijent.sifKlijent;


SELECT * FROM kronologijaNalogKlijenata ORDER BY datum DESC, prezimeKlijent ASC;
```

7.4. U bazi autoradionica: Izraditi pogled koji će ispisivati za svaki kvar:

- naziv,
- broj sati predviđenih za popravak tog kvara (vrijednost atributa satiKvar),
- prosječnu vrijednost ostvarenih sati rada po nalozima koji se odnose na taj kvar te
- razliku između broja predviđenih sati i prosječno potrošenih sati.

Podatke u pogledu potrebno je sortirati padajući po izračunatoj razlici.

Koristeći izrađeni pogled potrebno je ispisati:

a) sve podatke iz pogleda

b) sve kvarove kod kojih je prosječno potrošeno duplo ili više vremena od predviđenog

```
CREATE OR REPLACE VIEW satnica_kvara AS
  SELECT nazivKvar, satiKvar, AVG(OstvareniSatiRada),
    satiKvar-AVG(OstvareniSatiRada) AS razlika
    FROM kvar JOIN nalog ON kvar.sifKvar = nalog.sifKvar
    GROUP BY nazivKvar, satiKvar
    ORDER BY razlika DESC;

/* a */
SELECT * FROM satnica_kvara;

/* b */
SELECT * FROM satnica_kvara
  WHERE ABS(razlika) > 2 * satiKvar;
```

nazivKvar	satiKvar	AVG(OstvareniSatiRada)	razlika
Zamjena svječica	1	4.7143	-3.7143
Zamjena rashladne tekućine	1	5.2424	-4.2424
Postavljanje ispušnog lonca	2	6.3810	-4.3810
Popravak felgi	2	6.4615	-4.4615
Ugradnja prednjeg znaka od mercedesa	2	7.3333	-5.3333
Zamjena krovnog prozora	1	6.5556	-5.5556
Ugradnja ojačanja	2	10.0000	-8.0000

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

7.5. U bazi studenti: Izraditi pogled koji će ispisivati sortirani popis svih studenata sa njihovim prosjekom ocjena. Popis mora minimalno sadržavati jmbag, ime, prezime i prosječnu ocjenu studenta (zaokruženu na dvije decimale).

Popis je potrebno sortirati po prosjeku i to od većeg prema manjem.

Koristeći izrađeni pogled, potrebno je ispisati:

a) Sve studente iz pogleda koji studiraju na Tehničkom veleučilištu u Zagrebu

b) Sve studente iz pogleda koji studiraju na Međimurskom veleučilištu u Čakovcu te imaju prosjek 5.00

c) Sve studente iz pogleda koji studiraju na Fakultetu elektrotehnike i računarstva te imaju prosjek različit od 5.00

```
/* Napomena: primijetiti da ako je potrebno napraviti selekciju iz pogleda po
određenom atributu (WHERE ustanove.naziv = „…“), da se taj atribut svakako mora
nalaziti u projekciji pogleda (CREATE VIEW ime_pogleda AS SELECT …
ustanove.naziv,….) */

CREATE OR REPLACE ALGORITHM = TEMPTABLE VIEW uspjehStudenata AS
  SELECT jmbag, ime, prezime, ROUND(AVG(ocjena),2) AS prosjek, ustanove.naziv
    FROM studenti JOIN ocjene ON jmbag = jmbagStudent
      JOIN smjerovi ON idSmjer = smjerovi.id
      JOIN ustanove ON oibUstanova = ustanove.oib
    GROUP BY jmbag, ime, prezime
    ORDER BY prosjek DESC;

/* a */
SELECT *
  FROM uspjehStudenata
  WHERE naziv="Tehničko veleučilište u Zagrebu";

/* b */
SELECT *
  FROM uspjehStudenata
  WHERE prosjek = 5.00
  AND naziv="Međimursko veleučilište u Čakovcu";

/* c */
SELECT *
  FROM uspjehStudenata
  WHERE prosjek != 5.00 AND naziv="Fakultet elektrotehnike i računarstva";
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

jmbag	ime	prezime	prosjek	naziv
0036450215	Josip	Pavković	4.00	Fakultet elektrotehnike i računarstva
0036460111	Ozren	Savić	4.00	Fakultet elektrotehnike i računarstva
0036468547	Sandra	Labak	4.00	Fakultet elektrotehnike i računarstva
0036440706	Vicka	Zečević	3.00	Fakultet elektrotehnike i računarstva
0036430552	Armin	Čehajić	3.00	Fakultet elektrotehnike i računarstva
0036450023	Darko	Zidanić	3.00	Fakultet elektrotehnike i računarstva
0036466211	Nada	Marulin	3.00	Fakultet elektrotehnike i računarstva
0108066366	Iva	Perković	3.00	Fakultet elektrotehnike i računarstva
0036435111	Maja	Vuković	2.00	Fakultet elektrotehnike i računarstva

7.6. U bazi studenti: Izraditi pogled koji će ispisivati za sva mjesta, podatke o studentima koji u njima prebivaju te nastavnike koji su iz tog mjesta.

Ispis mora sadržavati kolone:

- Naziv mjesta
- Uloga ('Student' ili 'Nastavnik')
- Ime (studenta / nastavnika)
- Prezime (studenta / nastavnika)
- Jmbag / jmbg

Za dobivanje traženog popisa preporuča se korištenje unije.

a) Ispisati sve podatke iz pogleda.

b) Ispisati samo zapise koji se odnose na grad Zagreb, sortirati prema prezimenu uzlazno.

```
CREATE OR REPLACE VIEW nastavnici_i_studenti_mjesta AS
  SELECT nazivMjesto, 'Student' AS uloga,  ime,  prezime, jmbag AS jmbag_jmbg
    FROM studenti JOIN mjesta ON postBrPrebivanje = postbr
  UNION ALL
  SELECT nazivMjesto, 'Nastavnik' AS uloga, ime, prezime, jmbg AS jmbag_jmbg
    FROM nastavnici JOIN mjesta ON nastavnici.postBr = mjesta.postbr;

/* a */
SELECT * FROM nastavnici_i_studenti_mjesta;

/* b */
SELECT * FROM nastavnici_i_studenti_mjesta
  WHERE nazivMjesto = 'Zagreb'
  ORDER BY prezime;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

nazivMjesto	uloga	ime	prezime	jmbag_jmbg
Zagreb	Student	Ana	Anđal	1191221406
Zagreb	Student	Ivana	Babić	0246022345
Zagreb	Student	Renato	Babojelić	1191219570
Zagreb	Nastavnik	Nikola	Bačić	1908987173977
Zagreb	Nastavnik	Daniela	Bašić	1804949363053
Zagreb	Student	Sonja	Batelić	0246026699
Zagreb	Student	Valentina	Benšek	1191230621
Zagreb	Student	Ksenija	Božić	0246025333
Zagreb	Student	Karla	Brandis	0035192211

7.7. U bazi studenti: Izraditi pogled koji će vraćati popis svih nastavnika u formi:

- titula_ispred ime_i_prezime, titula_iza -> stupac nazvati „Nastavnik“ (Dodatno provjeriti postoji li za određenog nastavnika evidentirana titula ispred i titula iza, kako se nebi u ispisu ispisivali nepotrebni razmaci ili zarezi)
- jmbg -> stupac nazvati „Jmbg_nastavnik“
- na koliko kolegija nastavnik ima ulogu profesora -> stupac nazvati „Profesor“
- na koliko kolegija nastavnik ima ulogu asistent -> stupac nazvati „Asistent“

Vrijednosti trećeg i četvrtog stupca moguće je računati pomoću podupita.

Koristeći izrađeni pogled, ispisati samo one nastavnike koji su profesori i asistenti na različitom broju kolegija.

```
CREATE OR REPLACE VIEW uloge_nastavnika AS
  SELECT
    CONCAT(
      IF(titulaIspred = "", "", CONCAT(titulaIspred, " ")),
      ime,
      ' ',
      prezime,
      IF(titulaIza = "", "", CONCAT(", ", titulaIza))
    ) AS nastavnik,
    jmbg AS Jmbg_nastavnik,
    (SELECT COUNT(*)
      FROM izvrsitelji
      JOIN ulogaIzvrsitelja ON idUlogaIzvrsitelja = ulogaIzvrsitelja.id
      WHERE ulogaizvrsitelja.naziv = "profesor"
      AND jmbgNastavnik = jmbg_nastavnik) AS Profesor,
    (SELECT COUNT(*)
      FROM izvrsitelji
      JOIN ulogaIzvrsitelja ON idUlogaIzvrsitelja = ulogaIzvrsitelja.id
       WHERE ulogaizvrsitelja.naziv = "Asistent"
      AND jmbgNastavnik = jmbg_nastavnik) AS asistent
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    FROM nastavnici;


SELECT * FROM uloge_nastavnika
  WHERE profesor != asistent;
```

nastavnik	Jmbg_nastavnik	Profesor	Asistent
mr.sc. Vedran Grubišić	0205951330124	1	0
Lucija Brčić, prof.	0305988391019	1	0
dr.sc. Danijel Tkalčić	0404964361006	0	1
Anto Bošnjak, prof.	0501953210249	1	0
prof. dr.sc. Borna Blažević	0606960383925	1	0
mr.sc. Stjepan Kovačević	0608955311207	2	0
prof. dr.sc. Renato Topić	0610975181955	0	1
dr.sc. Mila Ilić, v. pred.	0611973330036	1	0
prof. dr.sc. Boris Brkić	0708980383302	1	0
prof. dr.sc. Stjepan Lukić	0804969306801	0	1
prof.vis.šk. Anđelka Mađar	0809970383313	1	0

…

7.8. U bazi autoradionica: Izraditi **pogled uz korištenje CTE-a** koji će po odjelima sadržavati podatke prema slici (sortirati uzlazno po nazivu odjela):

sifOdjel	nazivOdjel	br_naloga	pros_sati	sum_sati	sifNadOdjel	nazNadrOdj
14	Alarmi	11	5.2727	58	3	Elektrika
20	Amortizeri	6	5.1667	31	5	Optika
16	Balansiranje	5	5.2000	26	4	Vulkanizacija
19	Balansiranje guma	27	4.8148	130	5	Optika
3	Elektrika	53	5.4717	290	1	Auto centar kod veselog Zagorca
11	Elektropokretači	6	6.1667	37	3	Elektrika
13	Klima	25	5.1200	128	3	Elektrika
27	Kvačila	5	7.4000	37	6	Mehanika
24	Laki motori	25	4.5600	114	6	Mehanika
2	Limarija	62	5.5161	342	1	Auto centar kod veselog Zagorca
6	Mehanika	54	5.7222	309	1	Auto centar kod veselog Zagorca
22	Mulde	41	5.5610	228	5	Optika
5	Optika	19	5.3158	101	1	Auto centar kod veselog Zagorca
15	Skladište	25	6.4000	160	4	Vulkanizacija
26	Teški motori	4	6.0000	24	6	Mehanika
4	Vulkanizacija	45	5.0000	225	1	Auto centar kod veselog Zagorca
17	Zamjena guma	6	6.1667	37	4	Vulkanizacija
9	Zaštita od korozije	1	10.0000	10	2	Limarija

Koristeći izrađeni pogled potrebno je po nadređenim odjelima prikazati ukupan broj naloga i ukupnu sumu sati za one nadređene odjele koji imaju više od 1 naloga. Ispis sortirati uzlazno po šifri nadređenog odjela.

sifNadOdjel	nazNadrOdj	ukupno_naloga	ukupno_sati
1	Auto centar kod veselog Zagorca	233	1267
3	Elektrika	42	223
4	Vulkanizacija	36	223
5	Optika	74	389
6	Mehanika	34	175

```
CREATE OR REPLACE VIEW odjel_nalog AS


/* Unutar CTE-a obavljamo izračun agregiranih podataka i to nad minimalnim
brojem tablica kako bi se optimizirala brzina upita, te naknadno u SELECT-u u
nastavku spajamo sa ostatkom tablica za dobivanje ostalih atributa u ispisu */
  WITH ag_odjel AS
    (SELECT r.sifOdjel,
        COUNT(*) AS br_naloga,
        AVG(OstvareniSatiRada) AS pros_sati,
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
        SUM(OstvareniSatiRada) AS sum_sati
      FROM nalog n NATURAL JOIN radnik r
      GROUP BY r.sifOdjel)


  SELECT o.sifOdjel, o.nazivOdjel, ag.br_naloga, ag.pros_sati, ag.sum_sati,
o.sifNadOdjel, nadredeni_odjel.nazivOdjel AS nazNadOdj
    FROM odjel o
    NATURAL JOIN ag_odjel ag
    JOIN odjel nadredeni_odjel ON o.sifNadOdjel = nadredeni_odjel.sifOdjel
    ORDER BY 2;


/* Provjera svih podataka u pogledu */
SELECT * FROM odjel_nalog;


/* Korištenje pogleda za dobivanje zadanog ispisa po nadređenim odjelima */
SELECT sifNadOdjel, nazNadOdj, SUM(br_naloga) AS ukupno_naloga, SUM(sum_sati) AS
ukupno_sati
  FROM odjel_nalog
  GROUP BY sifNadOdjel, nazNadOdj
  HAVING COUNT(*) > 1;
```

7.9. U bazi studenti izraditi **pogled uz korištenje CTE-a** koji će po kolegijima prikazati sljedeće podatke:

- id kolegija,
- naziv kolegija,
- broj položenih ispita,
- najmanju ocjenu iz kolegija (uzeti samo pozitivne ocjene) i
- najveću ocjenu iz kolegija (uzeti samo pozitivne ocjene).

Prikazati samo one kolegije koji su imali više od 3 pozitivno ocjenjena ispita (ocjene iz kolegija). Sortirati po broju ispita padajući i najvećoj ocjeni rastući.

id	naziv	br_ispita	min_ocj	max_ocj
31	Poslovno komuniciranje	8	2	5
18	Refrakcija i binokularni vid I	8	2	5
20	Pogon zrakoplova I	7	3	5
35	Mikroekonomija I	7	2	5
33	Osnove poduzetništva	7	2	5
40	Sredstva pomorskog prometa	6	2	5
12	Javna nabava	6	2	5
13	Ekonomika turizma	6	2	5
22	Mjera i integral	6	2	5
17	Anatomija i fiziologija oka	6	2	5
39	Tehnička mehanika 2	5	2	3

...

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Koristeći kreirani pogled potrebno je ispisati sve retke gdje je broj položenih ispita između 5 i 7.

id	naziv	br_ispita	min_ocj	max_ocj
20	Pogon zrakoplova I	7	3	5
33	Osnove poduzetništva	7	2	5
35	Mikroekonomija I	7	2	5
12	Javna nabava	6	2	5
13	Ekonomika turizma	6	2	5
17	Anatomija i fiziologija oka	6	2	5
22	Mjera i integral	6	2	5
40	Sredstva pomorskog prometa	6	2	5
39	Tehnička mehanika 2	5	2	3
1	Baze podataka	5	2	4
28	Energijske tehnologije	5	2	4
41	Tereti u prometu	5	2	4
5	Fizika	5	2	5
9	Poslovna etika	5	2	5
15	Operacijski sustavi	5	3	5
21	Linearna algebra 1	5	2	5
30	Izrada web-projekta	5	2	5
34	Marketing	5	2	5
37	Matematika u ekonomiji	5	2	5
38	Marketing destinacije	5	2	5
42	Planiranje prometne potražnje	5	3	5
43	Tjelesna i zdravstvena kultura 2	5	5	5
45	Tjelesna i zdravstvena kultura 3	5	5	5

```
CREATE OR REPLACE VIEW kolegij_ocj AS
/* Unutar CTE-a obavljamo izračun agregiranih podataka i to nad minimalnim
brojem tablica (samo ocjene) kako bi se optimizirala brzina upita, te naknadno u
SELECT-u u nastavku spajamo s ostatkom tablica(tablica kolegiji) za dobivanje
ostalih atributa u ispisu */
  WITH CTE_KOL AS
    (SELECT idKolegij,
        COUNT(*) AS br_ispita,
        MIN(ocjena) min_ocj,
        MAX(ocjena) max_ocj
      FROM ocjene
      WHERE ocjena > 1
      GROUP BY idKolegij
      HAVING COUNT(*) > 3)

  SELECT k.id, k.naziv, ck.br_ispita, ck.min_ocj, ck.max_ocj
    FROM kolegiji k INNER JOIN CTE_KOL ck ON  k.id = ck.idKolegij
    ORDER BY 3 DESC, 5;

-- Kontrolni ispis svih podataka u pogledu
SELECT * FROM kolegij_ocj;

-- Korištenje pogleda za dobivanje zadanog ispisa
SELECT *
  FROM kolegij_ocj
  WHERE br_ispita BETWEEN 5 AND 7;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

7.10. U bazi autoradionica: Izraditi **pogled uz korištenje CTE-a** koji će po godinama i kvarovima sadržavati podatke prema slici (sortirati po godini i šifri kvara):

god	sifKvar	br_naloga	pros_sati	sum_sati	nazivKvar	brojRadnika	satiKvar
2011	1	4	6.0000	24	Zamjena blatobrana	2	2
2011	2	1	6.0000	6	Zamjena krova	4	2
2011	6	2	5.5000	11	Promjena osovine	1	2
2011	7	4	6.7500	27	Izmjena amortizera	4	2
2011	10	2	4.5000	9	Zamjena vozačkih sjedala	4	2
2011	11	1	1.0000	1	Zamjena pragova	8	4
2011	12	1	7.0000	7	Zamjena krovnog prozora	7	1
2011	13	1	4.0000	4	Zamjena blatobrana	4	3
2011	14	1	5.0000	5	Uštimavanje pokretnog krova	7	2
2011	17	1	7.0000	7	Ugradnja prednjeg znaka od Jaguara	3	3
2011	18	2	6.0000	12	Zamjena klipa	6	4
2011	21	2	5.5000	11	Štimanje ventila	2	3
2011	22	1	2.0000	2	Popravak radilice	8	3
2011	24	1	9.0000	9	Zamjena vodene pumpe	7	4
2011	25	4	5.5000	22	Zamjena rashladne tekućine	6	1

...

Koristeći izrađeni pogled potrebno je ispisati za sve vrste kvarova (**i one po kojima nije bilo naloga**) ukupan broj naloga i ukupnu sumu sati.

sifKvar	nazivKvar	ukup_br_naloga	ukup_suma_sati
1	Zamjena blatobrana	18	111
2	Zamjena krova	26	148
3	Zamjena cerade	3	13
4	Zamjena pragova	*(NULL)*	*(NULL)*
5	Učvršćivanje hvata prikolice	4	20
6	Promjena osovine	9	40
7	Izmjena amortizera	31	152
8	Zamjena prednjeg stakla	*(NULL)*	*(NULL)*
9	Zamjena vozačkih i suvozačkih vrata	7	34
10	Zamjena vozačkih sjedala	34	172
11	Zamjena pragova	6	29
12	Zamjena krovnog prozora	18	118
13	Zamjena blatobrana	15	79

```
CREATE OR REPLACE VIEW god_kvar AS


/* Unutar CTE-a obavljamo izračun agregiranih podataka i to nad minimalnim
brojem tablica (samo nalog) kako bi se optimizirala brzina upita, te naknadno u
SELECT-u u nastavku spajamo sa ostatkom tablica (tablica kvar) za dobivanje
ostalih atributa u ispisu */
  WITH agr AS
    (SELECT YEAR(datPrimitkaNalog) AS god, sifKvar,
        COUNT(*) AS br_naloga,
        AVG(OstvareniSatiRada) AS pros_sati,
        SUM(OstvareniSatiRada) AS sum_sati
      FROM  nalog
      GROUP BY YEAR(datPrimitkaNalog), sifKvar)


  SELECT a.*,  k.nazivKvar, k.brojRadnika, k.satiKvar
    FROM kvar k JOIN agr a ON k.sifKvar = a.sifKvar
    ORDER BY 1, 2;


/* Kontrolni ispis svih podataka u pogledu */
SELECT * FROM god_kvar;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
/* Korištenje pogleda za dobivanje zadanog ispisa */
SELECT k.sifKvar, k.nazivKvar, xx.ukup_br_naloga, xx.ukup_suma_sati
  FROM
    (SELECT sifKvar, SUM(br_naloga) AS ukup_br_naloga, SUM(sum_sati) AS
ukup_suma_sati
      FROM  god_kvar
      GROUP BY sifKvar) xx
  RIGHT JOIN kvar k ON xx.sifKvar= k.sifKvar
  ORDER BY 1;
```

7.11. U bazi autoradionica: Izraditi pogled uz korištenje CTE-a koji će prikazati sve odrađene naloge po radnicima i šiframa kvarova. Prikazati sljedeće podatke:

- šifru radnika,
- ime i prezime radnika spojeno,
- broj naloga po kvaru kojeg je radnik odradio i
- prosječno vrijeme po dotičnom radniku i kvaru.

U istom retku prikazati i podatke o kvaru:

- šifru kvara,
- naziv kvara,
- sveukupan broj naloga po kvaru (ne i po radniku) i
- prosječne sate po kvaru (ne i po radniku).

Sortirati po broju naloga radnika silazno.

sifRadnik	ime_prez	BR_NAL_R	PROS_SATI_R	sifKvar	nazivKvar	BR_NAL_K	PROS_SATI_K
166	Nino Lipec	18	6.1667	1	Zamjena blatobrana	18	6.1667
514	Damjan Dolar	13	5.9231	10	Zamjena vozačkih sjedala	34	5.0588
330	Vinko Kecman	12	5.4167	14	Uštimavanje pokretnog krova	12	5.4167
501	Davorin Rodinis	12	4.2500	25	Zamjena rashladne tekućine	33	5.2424
334	Sandra Mihaljević	10	5.1000	2	Zamjena krova	26	5.6923
411	Kristian Damjančić	10	7.0000	26	Ravnanje prednje vilice	20	6.4000
289	Marina Carek	9	6.4444	15	Popravak oštećenih mjesta na	11	5.6364
122	Tin Grabovac	9	3.8889	27	Zamjena blatobrana	16	4.0625
289	Marina Carek	9	5.0000	7	Izmjena amortizera	31	4.9032
142	Alan Ostojčić	9	6.0000	12	Zamjena krovnog prozora	18	6.5556
501	Davorin Rodinis	9	7.1111	12	Zamjena krovnog prozora	18	6.5556
379	Mia Stojić	8	5.3750	10	Zamjena vozačkih sjedala	34	5.0588
379	Mia Stojić	8	4.5000	25	Zamjena rashladne tekućine	33	5.2424
391	Marko Hazdovac	8	4.7500	29	Ugradnja sportskog pojasa	8	4.7500
244	Zlatko Tomašek	8	4.3750	7	Izmjena amortizera	31	4.9032

...

Koristeći kreirani pogled potrebno je ispisati sve retke gdje je prosjek sati radnika po kvaru manji od zajedničkog prosjeka sati po kvaru (to su radnici koji uspješnije odnosno brže rješavaju pojedine kvarove).

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

sifRadnik	ime_prez	BR_NAL_R	PROS_SATI_R	sifKvar	nazivKvar	BR_NAL_K	PROS_SATI_K
501	Davorin Rodinis	12	4.2500	25	Zamjena rashladne tekućine	33	5.2424
334	Sandra Mihaljević	10	5.1000	2	Zamjena krova	26	5.6923
122	Tin Grabovac	9	3.8889	27	Zamjena blatobrana	16	4.0625
142	Alan Ostojčić	9	6.0000	12	Zamjena krovnog prozora	18	6.5556
244	Zlatko Tomašek	8	4.3750	7	Izmjena amortizera	31	4.9032
379	Mia Stojić	8	4.5000	25	Zamjena rashladne tekućine	33	5.2424
232	Dubravko Josipović	8	3.8750	10	Zamjena vozačkih sjedala	34	5.0588
456	Slavica Haban	7	5.1429	26	Ravnanje prednje vilice	20	6.4000
191	Davorka Vidović	6	6.1667	31	Popravak felgi	13	6.4615
146	Franjo Vidaković	6	6.1667	33	Postavljanje ispušnog lonca	21	6.3810
232	Dubravko Josipović	6	4.6667	32	Ugradnja stražje spoilera	17	5.4118

```
CREATE OR REPLACE VIEW rad_odj_kvar AS
  WITH
    CTE_rad AS
      (SELECT sifRadnik, sifKvar,
          COUNT(*) AS BR_NAL_R,
          AVG(OstvareniSatiRada) PROS_SATI_R
        FROM nalog
        GROUP BY sifRadnik, sifKvar),
    CTE_kvar AS
      (SELECT sifKvar, COUNT(*) AS BR_NAL_K, AVG(OstvareniSatiRada) PROS_SATI_K
        FROM nalog
        GROUP BY sifKvar)
  SELECT r.sifRadnik, CONCAT_WS(' ', r.imeRadnik, r.prezimeRadnik) ime_prez,
cr.BR_NAL_R, cr.PROS_SATI_R,
    cr.sifKvar, k.nazivKvar, ck.BR_NAL_K, ck.PROS_SATI_K
  FROM radnik r
    INNER JOIN CTE_rad cr ON r.sifRadnik = cr.sifRadnik
    INNER JOIN CTE_kvar ck ON cr.sifKvar = ck.sifKvar
    INNER JOIN kvar k ON k.sifKvar = ck.sifKvar
  ORDER BY 3 DESC;

-- Kontrolni ispis svih podataka u pogledu
SELECT * FROM rad_odj_kvar;

-- Korištenje pogleda za dobivanje zadanog ispisa
SELECT *
  FROM rad_odj_kvar
  WHERE PROS_SATI_R < PROS_SATI_K;
```

7.12. U bazi studenti: Izraditi pogled uz korištenje CTE-a koji će po kolegijima sadržavati podatke prema slici (sortirati padajući po standardnoj devijaciji):

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

naz_koleg	idKolegij	pros_kolegija	stand_dev	br_1	br_2	br_3	br_4	br_5	Br_ocjena	idSmjer	pros_smjera
Marketing destinacije	38	3.3333	1.7	1	2	0	0	3	6	19	3.5455
Betonske konstrukcije	7	2.7143	1.58	3	0	1	2	1	7	4	3.0000
Matematička analiza 1	23	2.4000	1.5	2	1	1	0	1	5	12	2.8750
Kvantitativne metode u	36	3.5000	1.5	0	1	0	0	1	2	18	3.1000
Uvod u UNIX sustave	2	2.7500	1.48	1	1	1	0	1	4	1	3.3889
Poslovna etika	9	3.1667	1.46	1	1	2	0	2	6	5	3.3000
Ekonomika turizma	13	2.6250	1.41	2	3	0	2	1	8	8	2.8182
Fizika	5	2.5714	1.4	2	2	1	1	1	7	3	2.6364
Operacijski sustavi	15	3.5000	1.38	1	0	2	1	2	6	7	3.5000
Osnove poduzetništva	33	4.0000	1.31	0	2	0	1	4	7	17	3.8333
Osnove elektrotehnike	8	2.7500	1.3	1	1	0	2	0	4	3	2.6364
Napredne baze podataka	46	3.3333	1.25	0	1	1	0	1	3	1	3.3889
Sportski marketing	14	3.3333	1.25	0	1	1	0	1	3	8	2.8182
Uvod u kvantnu fiziku	24	3.6667	1.25	0	1	0	1	1	3	12	2.8750
Fizika	4	4.0000	1.22	0	1	0	1	2	4	2	3.3750
Opća kemija	25	4.0000	1.22	0	1	0	1	2	4	13	3.2500
Mikroekonomija I	35	3.0000	1.22	1	2	2	2	1	8	18	3.1000
Izrada web-projekta	30	2.8333	1.21	1	1	3	0	1	6	15	2.6667
Matematika u ekonomiji	37	3.8000	1.17	0	1	1	1	2	5	19	3.5455

...

Koristeći kreirani pogled potrebno je ispisati sve kolegije na kojima je prosjek kolegija veći od prosjeka smjera.

naz_koleg	idKolegij	pros_kolegija	stand_dev	br_1	br_2	br_3	br_4	br_5	Br_ocjena	idSmjer	pros_smjera
Tjelesna i zdravstvena	45	5.0000	0	0	0	0	0	5	5	16	3.6471
Tjelesna i zdravstvena	44	5.0000	0	0	0	0	0	4	4	10	3.6250
Tjelesna i zdravstvena	43	5.0000	0	0	0	0	0	5	5	1	3.3889
Fizika	4	4.0000	1.22	0	1	0	1	2	4	2	3.3750
Opća kemija	25	4.0000	1.22	0	1	0	1	2	4	13	3.2500
Osnove poduzetništva	33	4.0000	1.31	0	2	0	1	4	7	17	3.8333
Anatomija i fiziologija	17	3.8333	1.07	0	1	1	2	2	6	9	3.5000
Planiranje prometne pot	42	3.8000	0.75	0	0	2	2	1	5	21	3.3636
Matematika u ekonomiji	37	3.8000	1.17	0	1	1	1	2	5	19	3.5455
Osnove građanskog prava	11	3.7500	0.43	0	0	1	3	0	4	6	3.4000
Uvod u kvantnu fiziku	24	3.6667	1.25	0	1	0	1	1	3	12	2.8750
Digitalna logika	27	3.5000	1.12	0	1	1	1	1	4	14	2.7000
Fizika	6	3.5000	1.12	0	1	1	1	1	4	4	3.0000
Statistika	10	3.5000	1.12	0	1	1	1	1	4	5	3.3000
Kvantitativne metode u	36	3.5000	1.5	0	1	0	0	1	2	18	3.1000
Linearna algebra 1	21	3.4000	1.02	0	1	2	1	1	5	11	3.0909
Sredstva pomorskog prom	40	3.3333	1.11	0	2	1	2	1	6	20	3.0000
Sportski marketing	14	3.3333	1.25	0	1	1	0	1	3	8	2.8182
Izrada web-projekta	30	2.8333	1.21	1	1	3	0	1	6	15	2.6667
Osnove elektrotehnike	8	2.7500	1.3	1	1	0	2	0	4	3	2.6364

```
CREATE OR REPLACE VIEW koleg_ocjene AS

  WITH
    koleg_ocjene AS
      (SELECT idKolegij,
          AVG(ocjena) pros_kolegija,
          ROUND(STDDEV(ocjena),2) stand_dev,
          SUM(CASE when ocjena = 1 THEN 1 ELSE 0 END) AS br_1,
          SUM(CASE when ocjena = 2 THEN 1 ELSE 0 END) AS br_2,
          SUM(CASE when ocjena = 3 THEN 1 ELSE 0 END) AS br_3,
          SUM(CASE when ocjena = 4 THEN 1 ELSE 0 END) AS br_4,
          SUM(CASE when ocjena = 5 THEN 1 ELSE 0 END) AS br_5,
          COUNT(*) AS Br_ocjena
        FROM ocjene
        GROUP BY idKolegij),
    smjer_ocjene AS
      (SELECT idSmjer, AVG(o.ocjena) AS pros_smjera
        FROM ocjene o JOIN kolegiji k ON o.idKolegij = k.id
        GROUP BY idSmjer)
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT k.naziv AS naz_koleg, ko.*, k.idSmjer, so.pros_smjera
  FROM kolegiji k
    JOIN koleg_ocjene ko ON ko.idKolegij = k.id
    JOIN smjer_ocjene so ON so.idSmjer = k.idSmjer
  ORDER BY 4 DESC;

-- Kontrolni ispis svih podataka u pogledu
SELECT * FROM koleg_ocjene;

-- Korištenje pogleda za dobivanje zadanog ispisa
SELECT *
  FROM koleg_ocjene
  WHERE pros_kolegija > pros_smjera
  ORDER BY 3 DESC;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

8. Okidači

8.1. U bazi autoradionica: Postaviti okidač koji će prilikom brisanja zapisa iz tablice odjel provjeriti da li je taj odjel referenciran u tablici kvar ili radnik. Ako je referenciran onda će postaviti vrijednosti tih zapisa u tablicama kvar i radnik na NULL.

Napisati primjer naredbe koja će pokrenuti okidač.

```
DELIMITER %%
DROP TRIGGER IF EXISTS provjeriBrisanjeOdjela %%
CREATE TRIGGER provjeriBrisanjeOdjela AFTER DELETE ON odjel FOR EACH ROW
BEGIN
  UPDATE radnik
    SET sifOdjel = NULL
    WHERE sifOdjel = OLD.sifOdjel;
  UPDATE kvar
    SET sifOdjel = NULL
    WHERE sifOdjel = old.sifOdjel;
END %%
DELIMITER ;

/* Okidač će pokrenuti DELETE naredba nad tablicom odjel */
DELETE FROM odjel WHERE sifOdjel = 20;
```

8.2. U bazi autoradionica: Postaviti okidač koji će prilikom unosa novog zapisa u tablicu nalog, novom nalogu dodijeliti sljedeći slobodni prioritet (najveći prioritet uvećan za 1) u tom danu.

Napisati primjer naredbe koja će pokrenuti okidač i dodatno provjeriti učinak.

```
DELIMITER $$
DROP TRIGGER IF EXISTS dodijeliPrioritetNaloga $$
CREATE TRIGGER dodijeliPrioritetNaloga BEFORE INSERT ON nalog FOR EACH ROW
BEGIN
  DECLARE prioritet INT DEFAULT 0;
  SELECT MAX(prioritetNalog) + 1 INTO prioritet FROM nalog
    WHERE datPrimitkaNalog = new.datPrimitkaNalog;
  SET NEW.prioritetNalog = prioritet;
END
$$

DELIMITER ;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
/* Okidač će pokrenuti INSERT naredba nad tablicom nalog */
INSERT INTO nalog VALUES (1174, 26, 456, '2020-01-13', 2, 3);

/* Novom nalogu promijenjena je vrijednost prioriteta sa 2 na 5 */
SELECT * FROM nalog
  WHERE sifKlijent = 1174 AND sifRadnik = 456
  AND datPrimitkaNalog = '2020-01-13';
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1174	26	456	2020-01-13	5	3

8.3. U bazi autoradionica: Napraviti okidač koji će prilikom unosa novog zapisa u tablicu nalog provjeriti ima li radnik na kojeg se nalog odnosi, za taj dan već 8 ili više sati po drugim nalozima. Ako ima, potrebno je datum novog naloga povećati za 1 dan.

Napisati primjer naredbe koja će pokrenuti okidač i dodatno provjeriti rezultat.

```
DELIMITER $$
DROP TRIGGER IF EXISTS provjeraSatiRadnika $$
CREATE TRIGGER provjeraSatiRadnika BEFORE INSERT ON nalog FOR EACH ROW
BEGIN
  DECLARE brSati INT DEFAULT 0;
  SELECT SUM(OstvareniSatiRada) INTO brSati
    FROM nalog
    WHERE sifRadnik = NEW.sifRadnik AND datPrimitkaNalog = NEW.datPrimitkaNalog;
  IF brSati >= 8 THEN
    SET NEW.datPrimitkaNalog = NEW.datPrimitkaNalog + 1;
  END IF;
END
$$
DELIMITER ;

/* Provjera koliko sati ima radnik 456 na dan 26.2.2020. Kasnije ćemo mu
dodavati još naloga. */
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-26";
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-27";
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1306	18	456	2020-02-26	3	4

```
SELECT SUM(OstvareniSatiRada) FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-26";
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

sum(OstvareniSatiRada)
4

```
/* Za isti datum 26.2.2020., dodajemo još 5 sati. Okidač se pokrenuo prilikom
INSERT naredbe ali nije imao potrebu napraviti nikakvu promjenu jer je do tada
radnik imao 5 sati. */
INSERT INTO nalog VALUES (1173, 26, 456, '2020-02-26', 2, 5);
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-26";
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-27";
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1306	18	456	2020-02-26	3	4
1173	26	456	2020-02-26	2	5

```
/* Za isti datum 26.2.2020., dodajemo još 2 sata. Okidač se pokrenuo prilikom
INSERT naredbe te postavio datum za ovaj nalog na 27.2.2020. */
INSERT INTO nalog VALUES (1141, 26, 456, '2020-02-26', 1, 2);
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-26";
SELECT * FROM nalog
  WHERE sifRadnik = 456 AND datPrimitkaNalog = "2020-02-27";
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1306	18	456	2020-02-26	3	4
1173	26	456	2020-02-26	2	5

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1141	26	456	2020-02-27	1	2

8.4. U bazi autoradionica: Postaviti okidač koji će prilikom unosa novog zapisa u tablicu nalog dodijeliti klijentu onog radnika koji je već za njega popravljao taj kvar (jednom ili više puta) i ostvario pri tome najveći broj sati rada (ako je više radnika s istim uvjetom, dodijeliti bilo kojeg od tih radnika).

Ako nema takvog, zapis ostaje nepromijenjen.

Napisati primjer naredbe koja će pokrenuti okidač, i dodatno provjeriti učinak u bazi.

```
DELIMITER $$
DROP TRIGGER IF EXISTS odaberiRadnika $$
CREATE TRIGGER odaberiRadnika BEFORE INSERT ON nalog FOR EACH ROW
BEGIN
  DECLARE sifRad INT DEFAULT 0;
  SELECT sifRadnik INTO sifRad
    FROM nalog
    WHERE sifKlijent = NEW.sifKlijent AND sifKvar = NEW.sifKvar
    GROUP BY sifRadnik, sifKvar
    ORDER BY SUM(OstvareniSatiRada) DESC
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    LIMIT 0, 1;
  IF sifRad <> 0 THEN
   SET NEW.sifRadnik = sifRad;
  END IF;
END
$$
DELIMITER ;

/* Okidač će pokrenuti INSERT naredba nad tablicom nalog */
INSERT INTO nalog VALUES(1172, 15, 122, '2017-12-14', 2, 5);

SELECT * FROM nalog WHERE sifKlijent = 1172 AND datPrimitkaNalog = '2017-12-14';
/* Novounesenom nalogu, dodijeljen je radnik 289. */
```

sifKlijent	sifKvar	sifRadnik	datPrimitkaNalog	prioritetNalog	OstvareniSatiRada
1172	15	289	2017-12-14	2	5

8.5. U bazi autoradionica: U bazi autoradionica: Postaviti okidač koji će prilikom unosa novog radnika u tablicu radnik, novom radniku dodijeliti šifru onog odjela koji ima najmanje radnika. Ako je takvih odjela više, uzeti bilo koji odjel. Osigurati da primarni ključ sifRadnik bude jedinstven - ne smije biti već dodijeljen nekom radniku -u suprotnom slučaju postaviti taj ključ na sljedeću najveću vrijednost.

Napisati primjer naredbe koja će pokrenuti okidač.

```
ELIMITER $$
DROP TRIGGER IF EXISTS dodajRadnika $$
CREATE TRIGGER dodajRadnika BEFORE INSERT ON radnik FOR EACH ROW
BEGIN
  DECLARE v_sifOdjel INT DEFAULT 0;
  SELECT sifOdjel INTO v_sifOdjel
    FROM radnik
    GROUP BY sifOdjel
    ORDER BY COUNT(*)
    LIMIT 0,1;
  /* Provjera valjanosti šifre radnika */
  IF NEW.sifRadnik IN (SELECT sifRadnik from radnik) THEN
    SET NEW.sifRadnik = (SELECT MAX(sifRadnik)+1 FROM radnik);
  END IF;
  SET NEW.sifOdjel=v_sifOdjel;
END
$$
DELIMITER ;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
/* Primjer naredbe koja će pokrenuti okidač - INSERT naredba nad tablicom nalog
*/
INSERT INTO radnik VALUES(518,'Luka','Lukić',21000, 1, 2.40, 2500);
```

8.6. U bazi autoradionica: Postaviti okidač koji će prilikom promjene zapisa u tablici radnik, dodatno obaviti promjenu osnovice plaće radniku na kojeg se promjena odnosi. Potrebno je na temelju broja ostvarenih sati rada:

- Osnovicu povećati za 300 ako radnik ima više od 100 sati
- Osnovicu povećati za 100 ako radnik ima broj sati u opsegu [51 -100] sati
- Osnovica ostaje ista ako radnik ima broj sati u opsegu [0-50] sati

Napisati primjer naredbe koja će pokrenuti okidač te dodatno provjeriti učinak.

```
DELIMITER $$

DROP TRIGGER IF EXISTS korekcijaOsnovice $$
CREATE TRIGGER korekcijaOsnovice BEFORE UPDATE ON radnik FOR EACH ROW
BEGIN
  DECLARE sifRad, osnovica INT DEFAULT 0;
  SELECT x.osnova INTO osnovica
    FROM
      (SELECT sifRadnik,
        CASE
          WHEN SUM(OstvareniSatiRada) > 100 THEN 300
          WHEN SUM(OstvareniSatiRada) > 50
            AND SUM(OstvareniSatiRada) <=100 THEN 100
          ELSE 0
        END AS osnova
        FROM nalog
        GROUP BY sifRadnik) AS x
    WHERE OLD.sifRadnik = x.sifRadnik;

  SET NEW.IznosOsnovice = OLD.IznosOsnovice + osnovica;
END
$$
DELIMITER ;

SELECT * FROM radnik WHERE sifRadnik = 142;
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice
142	Alan	Ostojčić	10000	2	1.92	2200.00

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
/* Primjer naredbe koja će pokrenuti okidač */
UPDATE radnik SET sifOdjel = 18 WHERE sifRadnik = 142;


SELECT * FROM radnik WHERE sifRadnik = 142;
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice
142	Alan	Ostojčić	10000	18	1.92	2500.00

8.7. U bazi studenti: Postaviti okidač koji će prilikom unosa negativne ocjene provjeriti koliko je puta dotični student već polagao kolegij. Ako student za dotični kolegij već ima tri ili više evidentiranih negativnih ocjena, potrebno je upisati u novu tablicu *komisija* novi zapis: idkolegija, jmbag studenta i datum komisije (datum nove negativne ocjene uvećan za jedan mjesec).

Napisati primjer naredbe koja će pokrenuti okidač te dodatno provjeriti učinak.

```
CREATE TABLE komisija (idKolegija INT, jmbg CHAR(10), datumKomisije DATE);


DELIMITER $$
DROP TRIGGER IF EXISTS zaKomisiju $$
CREATE TRIGGER zaKomisiju BEFORE INSERT ON ocjene FOR EACH ROW
BEGIN
  DECLARE jmbag, kolegij, br_neg INT DEFAULT 0;
  SELECT x.* INTO jmbag, kolegij, br_neg FROM
    (SELECT jmbagStudent, idKolegij, COUNT(*) BR_NEG_OC
      FROM ocjene
      WHERE ocjena = 1
      GROUP BY jmbagStudent, idKolegij) AS x
    WHERE NEW.jmbagStudent = x.jmbagStudent AND NEW.idKolegij = x.idKolegij;
  IF NEW.ocjena = 1 AND br_neg > 2
    THEN INSERT INTO komisija
    VALUES (NEW.idKolegij, NEW.jmbagStudent,
        DATE_ADD(NEW.datumPolaganja, INTERVAL 1 MONTH));
  END IF;
END
$$
DELIMITER ;

/* Primjer naredbe koja će pokrenuti okidač */
INSERT INTO ocjene VALUES(29,'0036499965','2018-12-16','10:48:31', 1);
INSERT INTO ocjene VALUES(29,'0036499965','2019-02-05','14:24:04', 1);

SELECT * FROM komisija;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

idKolegija	jmbg	datumKomisije
29	0036499965	2019-03-05

8.8. U bazi autoradionica: Postaviti okidač koji će prilikom unosa novog naloga ažurirati tri nova atributa za radnika koji izvršava taj nalog: broj odrađenih naloga, sumu ostvarenih sati rada i prosječan broj ostvarenih sati rada.

Atribute odgovarajućeg tipa podataka potrebno je dodati u tablicu radnik: brNaloga, sumaSati i avgSati.

Napisati primjer naredbe koja će aktivirati okidač i provjeriti rezultat.

```
ALTER TABLE radnik
  ADD COLUMN brNaloga INT,
  ADD COLUMN sumaSati INT,
  ADD COLUMN avgSati DECIMAL(6,2);


DELIMITER $$
DROP TRIGGER IF EXISTS azurirajTotalZaRadnika $$
CREATE TRIGGER azurirajTotalZaRadnika AFTER INSERT ON nalog FOR EACH ROW
/* Prilikom deklaracije ovog okidača, bitno je izraditi AFTER okidač kako bi se
izračun sva 3 agregirana podatka pokrenuo nakon što i novi nalog bude unesen u
bazu, odnosno da i novi nalog bude uračunat u ukupan broj naloga, sumu
ostvarenih sati rada i prosječan broj sati. */
BEGIN
  DECLARE v_brNaloga INT DEFAULT 0;
  DECLARE v_sumaSati INT DEFAULT 0;
  DECLARE v_avgSati  DECIMAL(6,2) DEFAULT 0;
  SELECT COUNT(*), SUM(OstvareniSatiRada), AVG(OstvareniSatiRada)
    INTO v_brNaloga, v_sumaSati, v_avgSati
    FROM nalog
    WHERE sifRadnik=NEW.sifRadnik;
  UPDATE radnik
    SET brNaloga = v_brNaloga, sumaSati = v_sumaSati, avgSati = v_avgSati
    WHERE sifRadnik = NEW.sifRadnik;
END
$$
DELIMITER ;


/* Primjer naredbe koja će pokrenuti okidač */
INSERT INTO nalog VALUES (1174, 26, 330, '2023-04-01', 2, 3);
/* Prikaz dijela tablice radnik nakon pokretanja INSERT naredbe, vidljivi su
podatci za radnika 330 */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice	brNaloga	sumaSati	avgSati
318	Nino	Gvozdić	10000	9	0.44	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
322	Ana	Juretić	32000	16	0.74	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
326	Sunčica	Pleško	34000	19	0.40	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
330	Vinko	Kecman	10000	2	1.79	2200.00	13	68	5.23
334	Sandra	Mihaljević	10000	13	0.75	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
338	Petar	Petrović	20000	4	0.36	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
342	Tin	Mandić	10000	27	0.56	2200.00	*(NULL)*	*(NULL)*	*(NULL)*

8.9. U bazi autoradionica: Postaviti mehanizam koji će prilikom unosa novog naloga ažurirati tri nova atributa za radnika koji izvršava taj nalog: broj odrađenih naloga, sumu ostvarenih sati rada i prosječan broj ostvarenih sati rada.

Atribute odgovarajućeg tipa podataka potrebno je dodati u tablicu radnik: brNaloga, sumaSati i avgSati.

Zadanu poslovnu logiku potrebno je realizirati unutar procedure, te nakon toga izraditi okidač sa ispravnom aktivirajućom akcijom i vremenom aktiviranja. Pokretanje okidača mora pozvati (i izvršiti) proceduru.

Napisati primjer poziva procedure, te dodatno naredbe koja će aktivirati okidač. Provjeriti rezultat.

```
/* ALTER TABLE radnik
    ADD COLUMN brNaloga INT,
    ADD COLUMN sumaSati INT,
    ADD COLUMN avgSati DECIMAL(6,2); */

DELIMITER $$
DROP procedure IF EXISTS azurirajTotalZaRadnikaProc $$
CREATE procedure azurirajTotalZaRadnikaProc (in zadaniRadnik INT)
BEGIN
  DECLARE v_brNaloga INT DEFAULT 0;
  DECLARE v_sumaSati INT DEFAULT 0;
  DECLARE v_avgSati  DECIMAL(6,2) DEFAULT 0;
  SELECT COUNT(*), SUM(OstvareniSatiRada), AVG(OstvareniSatiRada)
    INTO v_brNaloga, v_sumaSati, v_avgSati
    FROM nalog
    WHERE sifRadnik=zadaniRadnik;
  UPDATE radnik
    SET brNaloga = v_brNaloga, sumaSati = v_sumaSati, avgSati = v_avgSati
    WHERE sifRadnik = zadaniRadnik;
END
$$
DELIMITER ;

/* Primjer poziva procedure */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
CALL azurirajTotalZaRadnikaProc(334);


/* Prikaz dijela tablice radnik nakon poziva procedure, vidljivi su podatci za
radnika 334 */
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice	brNaloga	sumaSati	avgSati
318	Nino	Gvozdić	10000	9	0.44	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
322	Ana	Juretić	32000	16	0.74	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
326	Sunčica	Pleško	34000	19	0.40	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
330	Vinko	Kecman	10000	2	1.79	2200.00	13	68	5.23
334	Sandra	Mihaljević	10000	13	0.75	2200.00	15	77	5.13
338	Petar	Petrović	20000	4	0.36	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
342	Tin	Mandić	10000	27	0.56	2200.00	*(NULL)*	*(NULL)*	*(NULL)*

```
DELIMITER $$

DROP TRIGGER IF EXISTS azurirajTotalZaRadnikaTrigg $$

CREATE TRIGGER azurirajTotalZaRadnikaTrigg AFTER INSERT ON nalog FOR EACH ROW

/* Prilikom deklaracije ovog okidača, bitno je izraditi AFTER okidač kako bi se
izračun sva 3 agregirana podatka pokrenuo nakon što i novi nalog bude unesen u
bazu, odnosno da i novi nalog bude uračunat u ukupan broj naloga, sumu
ostvarenih sati rada i prosječan broj sati. */

  CALL azurirajTotalZaRadnikaProc (NEW.sifradnik) $$

DELIMITER ;


/* Primjer naredbe koja će pokrenuti okidač */

INSERT INTO nalog VALUES (1267, 5, 334, '2023-04-04', 1, 4);


/* Prikaz dijela tablice radnik nakon pokretanja INSERT naredbe, vidljivi su
podatci za radnika 334 */
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice	brNaloga	sumaSati	avgSati
318	Nino	Gvozdić	10000	9	0.44	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
322	Ana	Juretić	32000	16	0.74	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
326	Sunčica	Pleško	34000	19	0.40	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
330	Vinko	Kecman	10000	2	1.79	2200.00	13	68	5.23
334	Sandra	Mihaljević	10000	13	0.75	2200.00	16	81	5.06
338	Petar	Petrović	20000	4	0.36	2200.00	*(NULL)*	*(NULL)*	*(NULL)*
342	Tin	Mandić	10000	27	0.56	2200.00	*(NULL)*	*(NULL)*	*(NULL)*

8.10. U bazi autoradionica: Postaviti okidač koji će prilikom unosa naloga ažurirati tri nova atributa za odjel u kojem se izvršava taj nalog: broj odrađenih naloga, sumu ostvarenih sati rada i prosječan broj ostvarenih sati rada.

Atribute odgovarajućeg tipa podataka potrebno je dodati u tablicu odjel: brNaloga, sumaSati i avgSati.

Napisati primjer naredbe koja će aktivirati okidač i provjeriti rezultat.

```
ALTER TABLE odjel
  ADD COLUMN brNaloga INT,
  ADD COLUMN sumaSati INT,
  ADD COLUMN avgSati DECIMAL(6,2);


DELIMITER $$
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
DROP TRIGGER IF EXISTS azurirajTotalZaOdjel $$
CREATE TRIGGER azurirajTotalZaOdjel AFTER INSERT ON nalog FOR EACH ROW
/* Prilikom deklaracije ovog okidača, bitno je izraditi AFTER okidač kako bi se
izračun sva 3 agregirana podatka pokrenuo nakon što i novi nalog bude unesen u
bazu, odnosno da i novi nalog bude uračunat u ukupan broj naloga, sumu
ostvarenih sati rada i prosječan broj sati. */
BEGIN
  DECLARE v_sifOdjel, v_brNaloga INT DEFAULT 0;
  DECLARE v_sumaSati INT DEFAULT 0;
  DECLARE v_avgSati  DECIMAL(6,2) DEFAULT 0;
  SELECT DISTINCT sifOdjel INTO v_sifOdjel
    FROM radnik
    WHERE sifRadnik=NEW.sifRadnik;
  SELECT COUNT(*), SUM(n.OstvareniSatiRada), AVG(n.OstvareniSatiRada)
   INTO v_brNaloga, v_sumaSati, v_avgSati
    FROM nalog n natural join radnik r
    WHERE r.sifOdjel = v_sifOdjel;
  UPDATE odjel
    SET brNaloga = v_brNaloga, sumaSati = v_sumaSati, avgSati = v_avgSati
    WHERE sifOdjel = v_sifOdjel;
END
$$
DELIMITER ;

/* Primjer naredbe koja će pokrenuti okidač */
INSERT INTO nalog VALUES (1174, 26, 330, '2022-11-29', 2, 3);

/* Prikaz dijela tablice odjel nakon pokretanja INSERT naredbe, vidljivi su
podatci za odjel 2 */
```

sifOdjel	nazivOdjel	sifNadOdjel	brNaloga	sumaSati	avgSati
1	Auto centar kod veselog Zagorca	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*
2	Limarija	1	43	224	5.21
3	Elektrika	1	*(NULL)*	*(NULL)*	*(NULL)*
4	Vulkanizacija	1	*(NULL)*	*(NULL)*	*(NULL)*

8.11. U bazi autoradionica: Postaviti mehanizam koji će prilikom unosa naloga ažurirati tri nova atributa za odjel u kojem se izvršava taj nalog: broj odrađenih naloga, sumu ostvarenih sati rada i prosječan broj ostvarenih sati rada.

Atribute odgovarajućeg tipa podataka potrebno je dodati u tablicu odjel: brNaloga, sumaSati i avgSati.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Zadanu poslovnu logiku potrebno je realizirati unutar procedure, te nakon toga izraditi okidač sa ispravnom aktivirajućom akcijom i vremenom aktiviranja. Pokretanje okidača mora pozvati (i izvršiti) proceduru.

Napisati primjer naredbe koja će aktivirati okidač i provjeriti rezultat.

```
/* ALTER TABLE odjel
    ADD COLUMN brNaloga INT,
    ADD COLUMN sumaSati INT,
    ADD COLUMN avgSati DECIMAL(6,2); */

DELIMITER $$
DROP PROCEDURE IF EXISTS azurirajTotalZaOdjelProc $$
CREATE PROCEDURE azurirajTotalZaOdjelProc (IN zadaniRadnik INT)
BEGIN
  DECLARE v_sifOdjel, v_brNaloga INT DEFAULT 0;
  DECLARE v_sumaSati INT DEFAULT 0;
  DECLARE v_avgSati DECIMAL(6,2) DEFAULT 0;
  SELECT DISTINCT sifOdjel INTO v_sifOdjel
    FROM radnik
    WHERE sifRadnik=zadaniRadnik;
  SELECT COUNT(*), SUM(n.OstvareniSatiRada), AVG(n.OstvareniSatiRada)
    INTO v_brNaloga, v_sumaSati, v_avgSati
    FROM nalog n NATURAL JOIN radnik r
    WHERE r.sifOdjel = v_sifOdjel;
  UPDATE odjel
    SET brNaloga = v_brNaloga, sumaSati = v_sumaSati, avgSati = v_avgSati
    WHERE sifOdjel = v_sifOdjel;
END
$$
DELIMITER ;

CALL azurirajTotalZaOdjelProc(289);
/* Prikaz dijela tablice odjel nakon pokretanja procredure za radnika 289, koji
pripada odjelu 3 - vidljivi su podatci za odjel 3 */
```

sifOdjel	nazivOdjel	sifNadOdjel	brNaloga	sumaSati	avgSati
1	Auto centar kod veselog Zagorca	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*
2	Limarija	1	43	224	5.21
3	Elektrika	1	53	290	5.47
4	Vulkanizacija	1	*(NULL)*	*(NULL)*	*(NULL)*

```
DELIMITER $$
DROP TRIGGER IF EXISTS azurirajTotalZaOdjelTrigg $$
CREATE TRIGGER azurirajTotalZaOdjelTrigg AFTER INSERT ON nalog FOR EACH ROW
/* Prilikom deklaracije ovog okidača, bitno je izraditi AFTER okidač kako bi se
izračun sva 3 agregirana podatka pokrenuo nakon što i novi nalog bude unesen u
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
bazu, odnosno da i novi nalog bude uračunat u ukupan broj naloga, sumu
ostvarenih sati rada i prosječan broj sati. */
  CALL azurirajTotalZaOdjelProc(NEW.sifRadnik) $$
DELIMITER //


/* Primjer naredbe koja će pokrenuti okidač */
INSERT INTO nalog VALUES (1267, 14, 289, '2022-12-15', 2, 3);


/* Prikaz dijela tablice odjel nakon pokretanja INSERT naredbe, vidljivi su
podatci za odjel 3 */
```

sifOdjel	nazivOdjel	sifNadOdjel	brNaloga	sumaSati	avgSati
1	Auto centar kod veselog Zagorca	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*
2	Limarija	1	43	224	5.21
3	Elektrika	1	54	293	5.43
4	Vulkanizacija	1	*(NULL)*	*(NULL)*	*(NULL)*

8.12. U bazi autoradionica: U bazi autoradionica: Postaviti okidač koji će prilikom unosa novog naloga, ažurirati dva nova atributa za radnika kojemu pripada taj nalog (sumu odrađenih sati i kategoriju radnika). Atribute odgovarajućeg tipa podataka potrebno je dodati u tablicu radnik: sumaSati i katRad.

Radnik dobiva, u ovisnosti o broju naloga koje je odradio, kategorije prema sljedećem algoritmu:

- 'D' ako je broj naloga manji od 3,
- 'C' ako je broj naloga između 3 i 10,
- 'B' ako je broj naloga između 11 i 15,
- 'A' ako je broj naloga veći od 15.

Napisati primjer naredbe koja će aktivirati okidač.

```
ALTER TABLE radnik
  ADD COLUMN brNaloga INT,
  ADD COLUMN katRadnik CHAR(1);


DELIMITER $$
DROP TRIGGER IF EXISTS azurirajRadnika $$
CREATE TRIGGER azurirajRadnika AFTER INSERT ON nalog FOR EACH ROW
BEGIN
  DECLARE v_brNaloga INT DEFAULT 0;
  DECLARE v_katRad VARCHAR(1) DEFAULT NULL;
  SELECT
    SUM(OstvareniSatiRada),
    CASE
      WHEN COUNT(*) < 5 THEN 'D'
      WHEN COUNT (*) BETWEEN 5 AND 10 THEN 'C'
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      WHEN COUNT (*) BETWEEN 11 AND 15 THEN 'B'
      ELSE 'A'
    END INTO v_brNaloga, v_katRad
    FROM  nalog
    WHERE sifRadnik = NEW.sifRadnik;
  UPDATE radnik
    SET brNaloga = v_brNaloga, katRadnik  =  v_katRad
     WHERE sifRadnik = NEW.sifRadnik;
END
$$
DELIMITER ;

/* Aktiviranje okidača */
INSERT INTO nalog VALUES (1172, 26, 330, '2017-10-21', 2, 3);
```

sifRadnik	imeRadnik	prezimeRadnik	pbrStan	sifOdjel	KoefPlaca	IznosOsnovice	brNaloga	katRadnik
322	Ana	Juretić	32000	16	0.74	2200.00	*(NULL)*	*(NULL)*
326	Sunčica	Pleško	34000	19	0.40	2200.00	*(NULL)*	*(NULL)*
330	Vinko	Kecman	10000	2	1.79	2200.00	68	B
334	Sandra	Mihaljević	10000	13	0.75	2200.00	*(NULL)*	*(NULL)*
338	Petar	Petrović	20000	4	0.36	2200.00	*(NULL)*	*(NULL)*

8.13. U bazi studenti: Postaviti okidač koji će u slučaju promjene smjera studenta, njegovom prezimenu dodati tekst ' Uvjet: br_neg_ocj= ' i broj negativnih ocjena. Navedeno treba napraviti samo ako je student imao negativnih ocjena.

Osigurati također da se studentu ne može mijenjati njegov JMBAG.

Napisati primjer naredbe koja će pokrenuti okidač.

```
DELIMITER $$
DROP TRIGGER IF EXISTS uvjetniUpis $$
CREATE TRIGGER uvjetniUpis BEFORE UPDATE ON  studenti FOR EACH ROW
BEGIN
  DECLARE v_brNeg INT DEFAULT 0;
  SELECT COUNT(*) INTO v_brNeg
    FROM ocjene
    WHERE ocjena = 1 AND jmbagStudent =  OLD.jmbag;
  IF NEW.idSmjer  <>  OLD.idSmjer AND  v_brNeg > 0
  THEN
    SET NEW.prezime = CONCAT(OLD.prezime,' Uvjet: br_neg_ocj = ', v_brNeg);
  END IF;
  IF NEW.jmbag  <>  OLD.jmbag  THEN
    SET NEW.jmbag = OLD.jmbag ;
  END IF;
END
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
$$
DELIMITER ;


/* Pokretanje okidača */
UPDATE studenti
  SET idSmjer = 4 , jmbag= '9999999999'
  WHERE jmbag='0036499965';


/* Provjera rezultata (upisao se broj negativnih ocjena nakon prezimena, no
JMBAG  se nije promjenio */
SELECT * FROM studenti WHERE jmbag='0036499965';
```

jmbag	ime	prezime	datumUpisa	postBrPrebivanje	postBrStanovanja	idSmjer
0036499965	Lovro	Keglović Uvjet: br_neg_ocj = 2	2015-09-26	10000	10410	4

8.14. U bazi studenti: U tablicu nastavnici dodati novu kolonu 'lozinkaTimestamp'.

Potrebno je osigurati da se nakon svake promjene lozinke nekom od nastavnika, automatski upiše vremenska oznaka promjene lozinke u kolonu 'lozinkaTimestamp' (za tog nastavnika).

Napisati primjer naredbe koja će pokrenuti okidač.

```
ALTER TABLE nastavnici
  ADD COLUMN lozinkaTimestamp DATETIME;


DELIMITER %%
DROP TRIGGER IF EXISTS promjenaLozinke %%
CREATE TRIGGER promjenaLozinke BEFORE UPDATE ON nastavnici FOR EACH ROW
BEGIN
  IF OLD.lozinka <> NEW.lozinka THEN
    SET NEW.lozinkaTimestamp = NOW();
  END IF;
END %%
DELIMITER ;


/* Primjer naredbe koja će aktivirati ciljanu logiku iz okidača */
UPDATE nastavnici SET lozinka = SHA1("DanasJeLijepDan") WHERE jmbg =
"0509959360058";


SELECT * FROM nastavnici WHERE jmbg = "0509959360058";
```

jmbg	ime	prezime	lozinka	lozinkaTimestamp
0509959360058	Leona	Vidović	e236069b1226fbb30c11e2651cffe474	2023-04-15 15:44:36

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

8.15. U bazi studenti: Prilikom upisa ili izmjene ocjene iz nekog kolegija, potrebno je podatke o izmjeni pohraniti u tablicu *ocjene_log*. Potrebno je pohraniti i podatke o korisniku koji je unio ili izmijenio ocjenu te vremenski žig. Potrebno je kreirati tablicu *ocjene_log* sa atributima odgovarajućih tipova podataka.

```
/* U ovom zadatku potrebno je obavljati gotovo istu radnju za unos i izmjenu
podataka, no obzirom da su aktivirajuće operacije različite, biti će potrebno
izraditi dva okidača sa sličnim funkcionalnostima. */

CREATE TABLE ocjene_log (
  idKolegij INT(11) NOT NULL,
  jmbagStudent CHAR(16) COLLATE cp1250_croatian_ci NOT NULL,
  datumPolaganja DATE NOT NULL,
  opis CHAR(20) COLLATE cp1250_croatian_ci NOT NULL,
  ocjenaStara TINYINT(4) DEFAULT NULL,
  ocjenaNova TINYINT(4) DEFAULT NULL,
  korisnik VARCHAR(255)DEFAULT NULL,
  vremenskiZig DATETIME DEFAULT NULL,
  PRIMARY KEY (idKolegij,jmbagStudent,datumPolaganja,opis)
) ENGINE=INNODB DEFAULT CHARSET=cp1250 COLLATE=cp1250_croatian_ci;

/* Okidač za evidentiranje unesenih ocjena */
DELIMITER %%
DROP TRIGGER IF EXISTS insertOcjeneLog %%
CREATE TRIGGER insertOcjeneLog AFTER INSERT ON ocjene FOR EACH ROW
BEGIN
  INSERT INTO ocjene_log  (idKolegij, jmbagStudent, datumPolaganja,
        opis,ocjenaStara, ocjenaNova, korisnik, vremenskiZig)
    VALUES (NEW.idKolegij,
      NEW.jmbagStudent,
      NEW.datumPolaganja,
      'Unos ocjene',
      NULL,
      NEW.ocjena,
      CURRENT_USER(),
      NOW());
END;
%%

DELIMITER ;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
/* Primjer naredbe koja će aktivirati okidač */
INSERT INTO ocjene
  VALUES(2, '0016097990', '2023-02-02', '15:30:00', 1);

/* Okidač za evidentiranje promijenjenih ocjena */
DELIMITER %%
DROP TRIGGER IF EXISTS updateOcjeneLog %%
CREATE TRIGGER updateOcjeneLog AFTER UPDATE ON ocjene FOR EACH ROW
BEGIN
  INSERT INTO ocjene_log  (idKolegij, jmbagStudent, datumPolaganja, opis,
        ocjenaStara, ocjenaNova, korisnik, vremenskiZig)
    VALUES (NEW.idKolegij,
      NEW.jmbagStudent,
      NEW.datumPolaganja,
      'Ažuriranje ocjene',
      OLD.ocjena,
      NEW.ocjena,
      CURRENT_USER(),
      NOW());
END;
%%
DELIMITER ;

/*  Primjer naredbe koja će aktivirati okidač */
UPDATE ocjene
  SET ocjena = 3
  WHERE (idKolegij = 2 AND jmbagStudent = '0016097990' AND datumPolaganja = '2023-
02-02');

SELECT * FROM ocjene_log;
```

idKolegij	jmbagStudent	datumPolaganja	opis	ocjenaStara	ocjenaNova	korisnik	vremenskiZig
2	0016097990	2023-02-02	Ažuriranje ocjene	1	3	root@localhost	2023-04-15 15:06:00
2	0016097990	2023-02-02	Unos ocjene	*(NULL)*	1	root@localhost	2023-04-15 15:05:12

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

9. Korisnici i dozvole

9.1. U bazi autoradionica: Postavite dozvole za prikaz i brisanje slogova u tablicama *ocjene* i *studenti* korisniku „stud_user“ s lokalnog poslužitelja i dajte mu lozinku „loz“, uz ograničenje da u satu može napraviti najviše 10 upita i 15 ažuriranja.

Prikazati postavljene ovlasti. Spojiti se na poslužitelj s novim podacima.

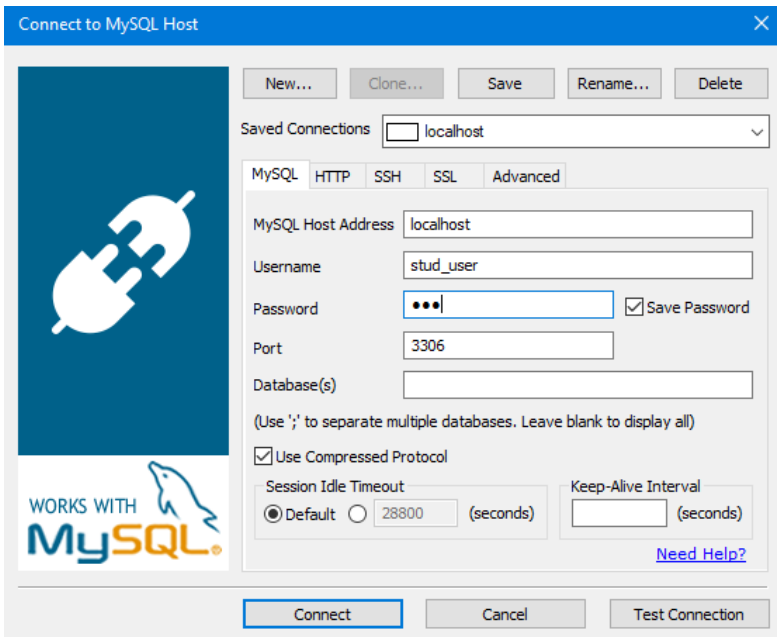
Ukinuti dozvole dodijeljene u ovom zadatku (paziti da se ne ukinu sve dozvole korisnika koje možda nisu dodijeljene u sklopu ovog zadatka). Obrisati stvorenog korisnika.

Riješiti funkcionalnost izrade korisnika i dodjeljivanja istih dozvola ali bez korištenja CREATE / GRANT naredbe već direktnim promjenama (DML) na odgovarajućim tablicama u meta bazi podataka „mysql“.

```
CREATE USER 'stud_user'@'localhost' IDENTIFIED BY 'loz';
GRANT SELECT, DELETE ON studenti.ocjene TO 'stud_user'@'localhost';
GRANT SELECT, DELETE ON studenti.studenti TO 'stud_user'@'localhost';
ALTER USER 'stud_user'@'localhost'
  WITH MAX_QUERIES_PER_HOUR 10 MAX_UPDATES_PER_HOUR 15;

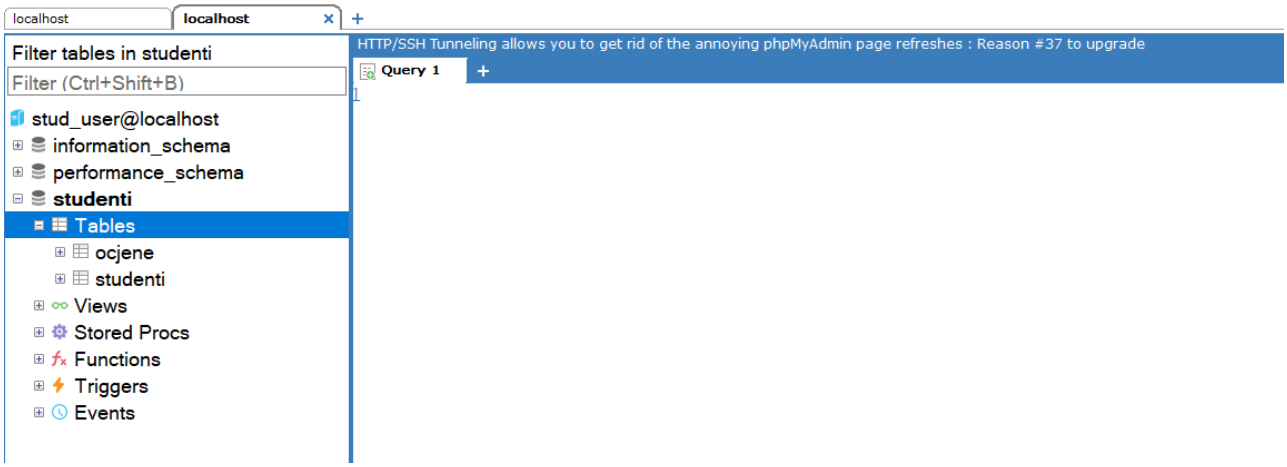
/* Prikaz dozvola i spajanje na poslužitelj kao novi korisnik stud_user */
SHOW GRANTS FOR 'stud_user'@'localhost';
```

```
Grants for stud_user@localhost
GRANT USAGE ON *.* TO `stud_user`@`localhost`
GRANT SELECT, DELETE ON `studenti`.`ocjene` TO `stud_user`@`localhost`
```



Slika 3. Spajanje na poslužitelj kao korisnik stud_user

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.



Slika 4. Pogled na dostupne tablice za korisnika stud_user

```
/* Vratiti se u ulogu 'root' i odraditi ukidanje dozvola */
REVOKE SELECT, DELETE ON studenti.ocjene FROM 'stud_user'@'localhost';
REVOKE SELECT, DELETE ON studenti. studenti FROM 'stud_user'@'localhost';
DROP USER 'stud_user'@'localhost';

/* Rješenje zadatka izravnim upisima i ažuriranjima u meta bazi podataka 'mysql'
*/
INSERT INTO mysql.user
  VALUES
('localhost', 'stud_user', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N',
'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N',
'N', 'N', 'N', '', '', '', '', '0', '0', '0', '0', 'caching_sha2_password',
SHA2('loz',0), 'N', '2023-04-18 13:56:33', NULL, 'N', 'N', 'N', NULL, NULL,
NULL, NULL);

INSERT INTO mysql.tables_priv
  VALUES ('localhost', 'studenti', 'stud_user', 'ocjene', 'root@localhost',
'2018-12-06 11:51:56', 'Select,Delete', ''), ('localhost', 'studenti',
'stud_user', 'studenti', 'root@localhost', '2018-12-06 11:51:57',
'Select,Delete', '');

UPDATE mysql.user
  SET max_questions = 10, max_updates=15
  WHERE user.Host= 'localhost' AND user.User ='stud_user';

/* Ako je potrebno */
FLUSH PRIVILEGES;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

9.2. U bazi autoradionica: Postavite dozvole za ažuriranje i dodavanje slogova u tablicama *radnik* i *nalog* korisniku „voditelj“ s lokalnog poslužitelja i dajte mu lozinku „vod_loz“, uz ograničenje da u satu može napraviti maksimalno 30 upita i 20 ažuriranja.

Prikazati postavljene ovlasti. Nakon toga mu promijenite lozinku u „vod_pass“.

Ukinuti dozvole dodijeljene u ovom zadatku.

```
CREATE USER 'voditelj'@'localhost' IDENTIFIED BY 'loz';


GRANT UPDATE, INSERT ON autoradionica.radnik TO 'voditelj'@'localhost';
GRANT UPDATE, INSERT ON autoradionica.nalog TO 'voditelj'@'localhost';
ALTER USER 'voditelj'@'localhost' WITH MAX_QUERIES_PER_HOUR 30
MAX_UPDATES_PER_HOUR 20;
SHOW GRANTS FOR 'voditelj'@'localhost';


SET PASSWORD FOR 'voditelj'@'localhost'= 'vod_pass';


REVOKE UPDATE, INSERT ON autoradionica.radnik FROM 'voditelj'@'localhost';
REVOKE UPDATE, INSERT ON autoradionica.nalog FROM 'voditelj'@'localhost';
```

9.3. U bazi studenti: Stvorite korisnika „prof“ koji ima pristup poslužitelju baze podataka sa svih adresa i dodijelite mu lozinku „prof_pass“, uz dodatne dozvole i ograničenja:

- da u satu može napraviti maksimalno 10 upita i 15 ažuriranja,
- da u tablici kolegiji:
  - može odabrati atribute: id, naziv;
  - može ažurirati atribute: opis, smjer,
- da u tablici smjerovi:
  - može odabrati atribut id;
  - može ažurirati atribute: naziv, oibUstanova.

Prikazati postavljene ovlasti. Spojiti se na poslužitelj s novim podacima.

Ukinuti dozvole dodijeljene u ovom zadatku.

Obrisati korisnika stvorenog u ovom zadatku.

Riješiti funkcionalnost ali bez korištenja CREATE / GRANT naredbe već direktnim promjenama (DML) na meta bazi podataka "mysql".

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
CREATE USER 'prof'@'%' IDENTIFIED BY 'prof_pass';
GRANT SELECT (id, naziv), UPDATE (opis, idSmjer) ON studenti.kolegiji
      TO 'prof'@'%' ;
GRANT SELECT (id ), UPDATE ( naziv, oibUstanova) ON studenti.smjerovi
      TO 'prof'@'%';
ALTER USER 'prof'@'%'  WITH max_queries_per_hour 10  max_updates_per_hour 15;

SHOW GRANTS FOR 'prof'@'%';

REVOKE SELECT (id, naziv), UPDATE (opis, idSmjer) ON studenti.kolegiji
  FROM 'prof'@'%';
REVOKE SELECT (id), UPDATE (naziv, oibUstanova) ON studenti.smjerovi
  FROM 'prof'@'%' ;
DROP USER 'prof'@'%';

-- Izravni upis u MYSQL meta bazu
INSERT INTO mysql.user
VALUES ('%', 'prof', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N',
 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N',
 'N', 'N', 'N', '', '', '', '', '0', '0', '0', '0', 'caching_sha2_password',
 SHA2('prof_pass',0), 'N', '2023-04-18 13:56:33', NULL,'N','N','N', NULL, NULL,
NULL, NULL );

INSERT into mysql.tables_priv
VALUES ('%', 'studenti', 'prof', 'smjerovi', 'root@localhost', '2018-01-06
        10:21:56', '', 'Select,Update'),
       ('%', 'studenti', 'prof', 'kolegiji', 'root@localhost', '2018-03-06
        11:51:36', '', 'Select,Update');

UPDATE mysql.user
SET max_questions = 10, max_updates=15
WHERE user.Host='%' AND user.User ='prof';
FLUSH PRIVILEGES;

INSERT into mysql.columns_priv
VALUES ('%',  'studenti','prof','smjerovi','id','0000-00-00 00:00:00','Select'),
('%','studenti','prof','smjerovi','naziv','0000-00-00 00:00:00',  'Update'),
('%','studenti','prof','smjerovi','oibUstanova','0000-00-00 00:00:00','Update'),
('%','studenti','prof','kolegiji','id', '0000-00-00 00:00:00','Select'),
('%', 'studenti','prof','kolegiji','naziv','0000-00-00 00:00:00','Select'),
('%','studenti','prof','kolegiji','opis','0000-00-00 00:00:00', 'Update'),
('%', 'studenti', 'prof','kolegiji','idSmjer','0000-00-00 00:00:00','Update');
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
-- Brisanje ovlasti na kolone u MYSQL meta bazi
DELETE FROM mysql.columns_priv WHERE
        (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'smjerovi' AND Column_name = 'id' AND Column_priv = 'Select') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'smjerovi' AND Column_name = 'naziv' AND Column_priv = 'Update') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'smjerovi' AND Column_name = 'oibUstanova' AND Column_priv = 'Update') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'kolegiji' AND Column_name = 'id' AND Column_priv = 'Select') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'kolegiji' AND Column_name = 'naziv' AND Column_priv = 'Select') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'kolegiji' AND Column_name = 'opis' AND Column_priv = 'Update') OR
    (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name =
'kolegiji' AND Column_name = 'idSmjer' AND Column_priv = 'Update');
```

```
-- Brisanje ovlasti na tablice u MYSQL meta bazi
DELETE FROM mysql.tables_priv WHERE
        (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name = 'kolegiji'
        AND Column_priv = 'Select,Update') OR
        (Host = '%'  AND Db = 'studenti'  AND User = 'prof' AND Table_name = 'smjerovi'
        AND Column_priv = 'Select,Update');
```

```
-- Brisanje korisnika u MYSQL meta bazi
DELETE FROM mysql.user WHERE Host = '%' AND user = 'prof';

FLUSH PRIVILEGES;
```

9.4. U bazi studenti: Postavite dozvole za sve baze i objekte korisniku „dekan“ sa svih adresa) i dajte mu lozinku „dekan_pass“, uz dodatnu mogućnost da može dodjeljivati dozvole drugim korisnicima.

Prikazati postavljene ovlasti.

Ukinuti dozvole dodijeljene u ovom zadatku.

```
CREATE USER 'dekan'@'%' IDENTIFIED BY 'dekan_pass';
GRANT ALL PRIVILEGES ON *.* TO 'dekan'@'%' WITH GRANT OPTION;
SHOW GRANTS FOR 'dekan'@'%';
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'dekan'@'%';
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

9.5. U bazi autoradionica: Napisati **proceduru** koja će za zadanu šifru kvara provjeriti postoji li ona u popisu kvarova, te ako postoji vratiti za zadani kvar podatak o nazivu kvara, broju (različitih) klijenata za koje se taj kvar popravljao, te ukupan broj naloga po dotičnom kvaru.

Postavite dozvole za kreiranje i pokretanje procedure korisniku „admin“ s lokalnog poslužitelja i dajte mu lozinku „admin_pass“.

Prikazati postavljene ovlasti. Nakon toga mu promijenite lozinku u „DBadmin_pass“

Spojiti se na poslužitelj s novim podacima te pokrenuti proceduru.

Ukinuti dozvole dodijeljene u ovom zadatku.

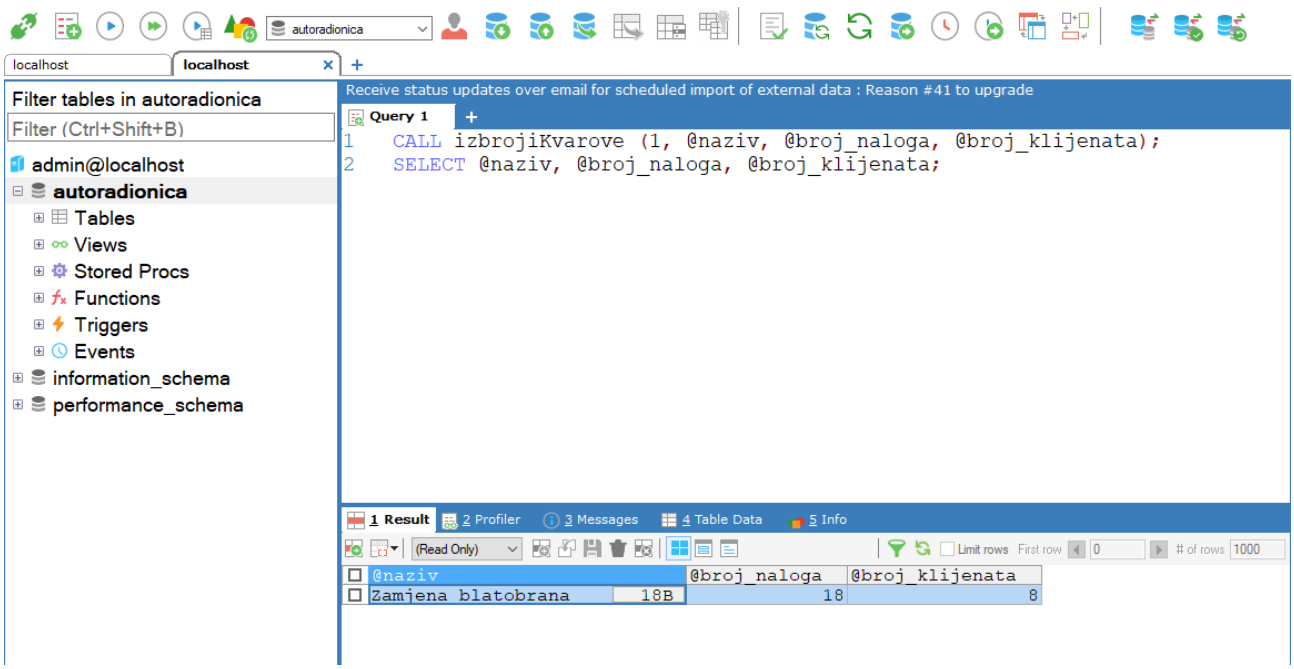
```
/* Izrada procedure */
DELIMITER //
DROP PROCEDURE IF EXISTS izbrojiKvarove //
CREATE PROCEDURE izbrojiKvarove(IN zadanaSifKvara INT, OUT naziv_kv
VARCHAR(100), OUT brNaloga INT, OUT brKlijenata INT)
BEGIN
  IF zadanaSifKvara IN (SELECT DISTINCT sifKvar FROM nalog)
  THEN
    SELECT COUNT(DISTINCT sifKlijent) INTO brKlijenata
      FROM nalog
      WHERE sifKvar =  zadanaSifKvara;
    SELECT  k.nazivKvar, COUNT(*) INTO naziv_kv, brNaloga
      FROM nalog n NATURAL JOIN kvar k
      WHERE n.sifKvar =  zadanaSifKvara
      GROUP BY  k.nazivKvar ;
  ELSE SELECT 'Nepostojeća šifra kvara. ' AS Poruka;
  END IF;
END//
DELIMITER ;

/* Administriranje korisnika i dozvola */
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin_pass';
GRANT CREATE ROUTINE ON autoradionica.* TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE autoradionica.izbrojiKvarove
  TO 'admin'@'localhost';
SET PASSWORD FOR 'admin'@'localhost'='DBadmin_pass';
SHOW GRANTS FOR 'admin'@'localhost';

/* Spojiti se na poslužitelj kao korisnik 'admin' i pokrenuti proceduru */
CALL izbrojiKvarove (1, @naziv, @broj_naloga, @broj_klijenata);
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SELECT @naziv, @broj_naloga, @broj_klijenata;
```



Slika 5. Rezultat nakon što je korisnik *admin* pokrenuo proceduru *izbrojiKvarove*

```
/* Kao korisnik 'root' odraditi ukidanje dozvola */
REVOKE CREATE ROUTINE ON autoradionica.*
  FROM 'admin'@'localhost';
REVOKE EXECUTE ON PROCEDURE autoradionica.izbrojiKvarove
  FROM 'admin'@'localhost' ;
```

9.6. Korisniku „kontrolor“ dodijeliti dozvole za kreiranje funkcija općenito i izvršavanjem funkcije koja za zadanu šifru radnika računa ukupan broj ostvarenih sati rada i vraća taj iznos. Ograničenje na lokaciju s koje „kontrolor“ može izvršavati ove naredbe ne postoji.

Postaviti mu lozinku „ctrl_pass“.

Spojiti se kao novi korisnik i pokrenuti funkciju.

Stvoriti korisnika „kontrolor_limited“ koji ima iste dozvole kao „kontrolor“ ali samo sa poddomene 192.168.10.* uz lozinku „ctrl_pass_limited“.

```
DROP FUNCTION IF EXISTS brSati;
DELIMITER //
CREATE FUNCTION brSati(sifraRadnika INT) RETURNS INT DETERMINISTIC
BEGIN
  DECLARE ukupnoSati INT DEFAULT 0;
  SELECT SUM(OstvareniSatiRada) INTO ukupnoSati
    FROM nalog WHERE sifRadnik = sifraRadnika;
```

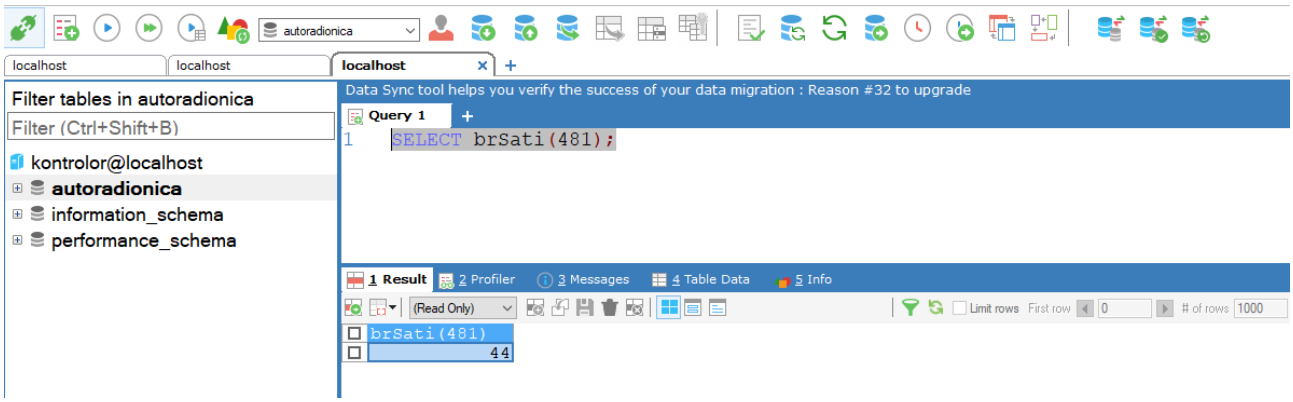
© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  RETURN ukupnoSati;
END; //
DELIMITER ;

/* Administriranje dozvola */
CREATE USER 'kontrolor'@ IDENTIFIED BY 'ctrl pass';
GRANT CREATE ROUTINE ON radionica.* TO 'kontrolor';
GRANT EXECUTE ON FUNCTION brSati TO 'kontrolor';
FLUSH PRIVILEGES;

CREATE USER 'kontrolor_limited'@'192.168.10.%' IDENTIFIED BY
'ctrl_pass_limited';
GRANT CREATE ROUTINE ON radionica.*
  TO 'kontrolor_limited'@'192.168.10.%';
GRANT EXECUTE ON FUNCTION brSati
  TO 'kontrolor_limited'@'192.168.10.%';
FLUSH PRIVILEGES;


/* Spojiti se kao korisnik 'kontrolor' i pokrenuti funkciju */
SELECT brSati(481);
```



Slika 6. Korisnik *kontrolor* pokreće funkciju *brSati*

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

10. Zaključavanje podataka

10.1. U bazi studenti: Zaključati tablicu OCJENE zbog ČITANJA, a tablicu KOLEGIJi zbog pisanja.

U drugoj sesiji pokusajte izracunati prosjek ocjena za sve kolegije (idkolegija, prosjek).

U trećoj sesiji izračunajte to isto ali dodajte i naziv kolegija (idkolegija, naziv kolegija, prosjek).

Komentirati stanja u drugoj i trećoj sesiji.

Otključajte tablice i pogledajte rezultate u ostalim sesijama.

```
-- U PRVOJ SESIJI
LOCK TABLES ocjene READ, kolegiji WRITE;

-- U DRUGOJ SESIJI
SELECT idKolegij, AVG(ocjena) AS prosjek
  FROM ocjene
  GROUP BY idKolegij;
/* Upit se uspješno izvršio */

-- U TREĆOJ SESIJI
SELECT ocjene.idKolegij, kolegiji.naziv, AVG(ocjene.ocjena) AS prosjek
  FROM ocjene join kolegiji on ocjene.idKolegij=kolegiji.id
  GROUP BY ocjene.idKolegij, kolegiji.naziv;
/* Upit se nije uspješno izvršio, odnosno izvršavanje je na čekanju. Upit će se
izvršiti nakon timeouta (neuspješno) odnosno uspješno nakon što prva sesija
skine postavljene ključeve */

-- U PRVOJ SESIJI nakon pokretanja 2. i 3. sesije
UNLOCK TABLES;
```

10.2. U bazi autoradionica: Napisati transakciju koja će zaključati n-torke u tablici KLIJENT koji po mjestu registracije vozila pripadaju Zagrebu (10000), te po nazivu dana u tjednu vraća broj prijavljenih klijenata. Transakciju napisati na način da druge sesije/transakcije ne mogu mijenjati podatke o tim klijentima, ali ih mogu čitati.

U drugoj sesiji, a dok još prva sesija nije zatvorila transakciju (COMMIT):

a) pokušati pročitati podatke o klijentima za Zagreb (po registraciji),

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

b) pokušati promijeniti podatke o tim klijentima tako da se prezime postavi na velika slova.

Komentirati akcije u drugoj sesiji.

Završiti transakciju u prvoj sesiji.

```
/* Prva sesija */
BEGIN WORK;
  SELECT DAYNAME(datUnosKlijent), COUNT(*) AS BrkKlijenata
    FROM klijent
    WHERE pbrReg = 10000
    GROUP BY DAYNAME(datUnosKlijent) FOR SHARE;

/* Druga sesija */
SELECT *
  FROM klijent
  WHERE pbrReg=10000;
UPDATE klijent
  SET prezimeKlijent = UPPER(prezimeKlijent)
  WHERE pbrReg = 10000;
/* Druga sesija ima mogućnost čitanja podataka (SELECT) ali ne i pisanja
(UPDATE) sve dok prva sesija naredbom COMMIT ne otpusti postavljeni ključ za
čitanje nad tablicom klijent */

/* Prva sesija */
COMMIT;
```

10.3. U bazi autoradionica: Napisati transakciju koja će zaključati n-torke u tablici RADNIK i ODJEL koje pripadaju odjelu naziva "Vulkanizacija", te vratiti broj radnika u tom odjelu.

Transakciju napisati na način da druge sesije/transakcije ne mogu mijenjati podatke o tim radnicima i tom odjelu, ali ih mogu čitati.

U drugoj sesiji, a dok još prva sesija nije završila transakciju (COMMIT):

a) pročitati podatke o radnicima za taj odjel (i naziv odjela)

b) promijeniti podatke o prezimenu na način da se prezime postavi na velika slova radnicima koji pripadaju odjelu sa šifrom 4

c) promijeniti podatke o odjelu na način da se odjelu „Vulkanizacija“ naziv odjela postavi na velika slova.

Komentirati akcije u drugoj sesiji.

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Završiti transakciju u prvoj sesiji.

```
/* Prva sesija */
BEGIN WORK;
  SELECT COUNT(*) AS BrRadnika
    FROM radnik r JOIN odjel o ON r.sifOdjel=o.sifOdjel
    WHERE o.nazivOdjel = 'Vulkanizacija' FOR SHARE;

/* Druga sesija */
  SELECT o.nazivOdjel, r.*
    FROM radnik r JOIN odjel o ON r.sifOdjel=o.sifOdjel
    WHERE o.nazivOdjel = 'Vulkanizacija';

  UPDATE radnik
    SET prezimeRadnik = UPPER(prezimeRadnik)
    WHERE sifOdjel= 4;

  UPDATE odjel
    SET nazivOdjel = UPPER(nazivOdjel)
    WHERE nazivOdjel = 'Vulkanizacija';

/* Prva sesija */
COMMIT;
```

10.4. U bazi studenti: Napisati transakciju koja će zaključati n-torke u tablici STUDENTI koje pripadaju smjeru 'računarstvo', te vratiti broj studenta na tom smjeru.

Transakciju napisati na način da druge sesije/transakcije ne mogu niti mijenjati ni čitati podatke o tim studentima.

U drugoj sesiji pokušati promijeniti podatke o tim studentima na način da se prezime postavi na velika slova.

```
/* Prva sesija */
BEGIN WORK;
  SELECT COUNT(*) AS BR_STUDENATA
    FROM STUDENTI
    WHERE idSmjer IN (SELECT id FROM smjerovi WHERE naziv = "računarstvo")
          FOR UPDATE;


/* Druga sesija */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  UPDATE STUDENTI
    SET ime = UPPER(ime)
    WHERE idSmjer IN (SELECT id FROM smjerovi WHERE naziv = "računarstvo");

/* Prva sesija */
COMMIT;
```

10.5. U bazi studenti: Napisati transakciju koja će zaključati n-torke u tablici OCJENE i KOLEGIJI koje pripadaju kolegiju (bez obzira na smjer) 'Baze podataka', te vratiti naziv kolegija i broj položenih ispita iz tog kolegija.

Transakciju napisati na način da druge sesije/transakcije ne mogu niti mijenjati ni čitati podatke o tim ocjenama i kolegiju.

U drugoj sesiji pokušati promijeniti ocjene na način da ih se poveća za 1 (ako nisu već 5).

```
/* Prva sesija */
BEGIN WORK;
  SELECT k.naziv, COUNT(*) AS BrPolozenih
    FROM ocjene o join kolegiji k on o.idKolegij=k.id
    WHERE k.naziv = 'Baze podataka' and ocjena > 1 FOR UPDATE;

/* Druga sesija */

 UPDATE ocjene
 SET ocjena=ocjena + 1
 WHERE ocjena BETWEEN 1 AND 4
 AND idKolegij IN (SELECT id FROM kolegiji WHERE naziv='Baze podataka');

/* Prva sesija */
COMMIT;
```

10.6. Napisati transakciju koja će zaključati n-torke u tablici STUDENTI koji po godini upisa pripadaju drugoj polovini 2017 godine, te po nazivu mjeseca vraća broj upisanih studenata.

Transakciju napisati na način da druge sesije/transakcije ne mogu mijenjati podatke o tim studentima, ali ih mogu čitati.

U drugoj sesiji, a prije nego je prva završila sa radom (COMMIT):

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

a) pročitati podatke o tim studentima

b) promijeniti podatke tim studentima na način da se ime postavi na velika slova.

```
/* Prva sesija */
BEGIN WORK;
  SELECT monthname(datumUpisa), COUNT(*) AS BrStudenata
    FROM studenti
    WHERE YEAR(datumUpisa) = 2010 AND MONTH(datumUpisa) > 6
    GROUP BY monthname(datumUpisa) FOR SHARE;

/* Druga sesija */
SELECT *
  FROM studenti
  WHERE YEAR(datumUpisa) = 2010 AND MONTH(datumUpisa) > 6;

UPDATE studenti
  SET ime = UPPER(ime)
  WHERE YEAR(datumUpisa) = 2010 AND MONTH(datumUpisa) > 6;

/* Prva sesija */
COMMIT;
```

10.7. U bazi studenti: Napisati transakciju koja će zaključati n-torke u tablici STUDENTI i SMJEROVI koje pripadaju smjerovima naziva "Stručni studij informatike" i "Stručni studij računarstva", te vratiti broj studenata po tim smjerovima te njihove udjele u ukupnom broju studenata.

Transakciju napisati na način da druge sesije/transakcije ne mogu mijenjati podatke o tim studentima i tim smjerovima, ali ih mogu čitati.

U drugoj sesiji, a prije nego prva otpusti ključeve, pokušati:

a) pročitati podatke o studentima tih smjerova (i naziv smjera)

b) promijeniti podatke o prezimenu na način da se prezime postavi na velika slova

c) promijeniti podatke o smjeru na način da se naziv smjera postavi na velika slova.

```
/* Prva sesija */
BEGIN WORK;
  SELECT sm.naziv,
    COUNT(*) AS BrRadnika,
    ROUND(COUNT(*) / (SELECT COUNT(*) FROM studenti)*100,2) AS UDIO
  FROM studenti st JOIN smjerovi sm ON st.idSmjer=sm.id
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  WHERE sm.naziv IN ('Stručni studij informatike' , 'Stručni studij
računarstva')
  GROUP BY sm.naziv FOR SHARE;


/* Druga sesija */
SELECT *
  FROM studenti st JOIN smjerovi sm ON st.idSmjer=sm.id
  WHERE sm.naziv IN ('Stručni studij informatike' , 'Stručni studij
računarstva');


UPDATE studenti
  SET prezime= UPPER(prezime)
  WHERE idsmjer IN (SELECT id FROM smjerovi WHERE naziv IN ('Stručni studij
informatike' , 'Stručni studij računarstva'));


UPDATE smjerovi
  SET naziv = UPPER(naziv)
  WHERE naziv IN ('Stručni studij informatike' , 'Stručni studij
računarstva');


/* Prva sesija */
COMMIT;
```

10.8. Napisati transakciju koja će zaključati n-torke u tablici RADNIK i ODJEL koje pripadaju odjelima 'Optika' i 'Mehanika', te vratiti naziv odjela i broj radnika u tim odjelima.

Transakciju napisati na način da druge sesije/transakcije ne mogu niti mijenjati ni čitati podatke o tim radnicima i odjelima.

U drugoj sesiji, a prije nego prva otpusti ključeve, pokušati promijeniti KoefPlace tako da se poveća za 0.1.

```
/* Prva sesija */
BEGIN WORK;
  SELECT o.nazivOdjel, COUNT(*) AS BrRadnika
    FROM radnik r  NATURAL JOIN odjel o
    WHERE o.nazivOdjel IN ('Optika', 'Mehanika')
    GROUP BY o.nazivOdjel FOR UPDATE;


/* Druga sesija */
SELECT *
  FROM radnik r  NATURAL JOIN odjel o
  WHERE o.nazivOdjel IN ('Optika', 'Mehanika');
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
UPDATE radnik
  SET KoefPlaca= KoefPlaca + 1
  WHERE  sifOdjel IN
    (SELECT sifOdjel FROM  odjel WHERE nazivOdjel IN('Optika', 'Mehanika') );

/* Prva sesija */
COMMIT;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

11. Mehanizmi pohrane

11.1. U sklopu baze podataka, izraditi tablicu *muzeji* koja će opisivati podatke o muzejima: id, naziv muzeja, id vrste muzeja, id mjesta (lokacije muzeja), izložbena površina i datum otvaranja. Odabrati / promijeniti mehanizam pohrane podataka koji će omogućiti:

a) Zaključavanje podataka u tablici na razini retka (napisati primjer zaključavanja redaka zbog pisanja za sve muzeje otvorene poslije 2001 godine i da im je izložbena površina od 1000 do 2500 m2, uključeno)

b) Zaključavanje podataka u tablici na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja tablice zbog čitanja, otključati zaključano)

c) Postavljanje stranog ključa naziva *str_kljuc* na atribut *id_mjesto* koji referencira na primarni ključ *postbr* u tablici mjesta.

d) Što kraće vrijeme pretrage podataka, podršku za full text indeks. Ako je potrebno, obaviti potrebne predradnje.

e) Prikazati naziv baze, tablice i mehanizam pohrane za tablicu *muzeji*.

```
CREATE TABLE muzeji (
  id INT,
  naziv VARCHAR(100),
  id_vrsta INT,
  id_mjesto INT,
  izl_povrsina INT,
  dat_otvaranja DATE);

-- a) ------------------------------------------------------------------------
ALTER TABLE muzeji ENGINE ARCHIVE;

SELECT *
  FROM muzeji
  WHERE YEAR(dat_otvaranja) > 2001
  AND izl_povrsina BETWEEN 1000 AND 2500
  FOR UPDATE;

-- b) ------------------------------------------------------------------------
ALTER TABLE muzeji ENGINE MYISAM;
LOCK TABLE muzeji READ;
UNLOCK TABLES;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
-- c) --------------------------------------------------------------------------
ALTER TABLE muzeji ENGINE INNODB;

ALTER TABLE studenti.muzeji
  ADD CONSTRAINT str_kljuc
  FOREIGN KEY (id_mjesto)
  REFERENCES studenti.mjesta (postbr);

-- d) --------------------------------------------------------------------------
ALTER TABLE muzeji ENGINE MYISAM;

/* Izvršavanje prethodne naredbe rezultirat će pogreškom:

Error Code: 3776. Cannot change table's storage engine because the
table participates in a foreign key constraint.

Stoga je prvo potrebno obrisati strani ključ jer ih MyISAM mehanizam pohrane ne
podržava.*/

ALTER TABLE studenti.muzeji
  DROP FOREIGN KEY str_kljuc;

ALTER TABLE muzeji ENGINE MYISAM;

-- e) --------------------------------------------------------------------------
SELECT table_schema, table_name, ENGINE
  FROM information_schema.tables
  WHERE table_schema = 'studenti' AND table_name = 'muzeji';
```

11.2. U bazi autoradionica, izraditi tablicu *odjel_radnik* koja će opisivati podatke o trenutnoj i prošloj pripadnosti radnika odjelu: *sifRadnik, sifOdjel, datum_od, datum_do, ts_azuriranja* (timestamp). Odabrati / promijeniti mehanizam pohrane podataka koji će omogućiti:

a) Zaključavanje podataka u tablici na razini retka (napisati primjer zaključavanja zbog čitanja za sve n-torke koje imaju *datum_DO* prazan ili je prisutnost radnika u nekom odjelu bila manja od 30 dana.

b) Zaključavanje podataka u tablici na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja tablice zbog pisanja, otključati zaključano).

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

c) Postavljanje stranog ključa naziva *str_klj_rad* na atribut *sifRadnik* koji referencira na primarni ključ *sifRadnik* u tablici radnik i stranog ključa naziva *str_klj_odj* na atribut *sifOdjel* koji referencira na primarni ključ *sifOdjel* u tablici odjel.

d) Prikazati naziv baze, tablice i engine za tablice: *radnik, odjel, odjel_radnik.*

```
CREATE TABLE odjel_radnik (
  sifRadnik INT,
  sifOdjel INT,
  datum_OD DATE,
  datum_DO DATE,
  ts_azuriranja TIMESTAMP);

-- a) -------------------------------------------------------------------------
ALTER TABLE odjel_radnik
  ENGINE INNODB;

SELECT *
  FROM odjel_radnik
  WHERE datum_DO IS NULL
  OR (datum_DO IS NOT NULL AND DATEDIFF(datum_DO, datum_OD) < 30)  FOR SHARE;

-- b) -------------------------------------------------------------------------
ALTER TABLE odjel_radnik ENGINE MEMORY;
LOCK TABLE odjel_radnik WRITE;
UNLOCK TABLES;

-- c) -------------------------------------------------------------------------
ALTER TABLE odjel_radnik ENGINE INNODB;

ALTER TABLE autoradionica.odjel_radnik
  ADD CONSTRAINT str_klj_rad
    FOREIGN KEY (sifRadnik)
    REFERENCES autoradionica.radnik (sifRadnik);

ALTER TABLE autoradionica.odjel_radnik
  ADD CONSTRAINT str_klj_odj
    FOREIGN KEY (sifOdjel)
    REFERENCES autoradionica.odjel (sifOdjel);

-- d) -------------------------------------------------------------------------

SELECT table_schema, TABLE_NAME, ENGINE
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  FROM information_schema.tables
  WHERE table_schema = 'autoradionica'
    AND TABLE_NAME IN ('odjel', 'radnik',  'odjel_radnik');
```

TABLE_SCHEMA	TABLE_NAME	ENGINE
autoradionica	odjel	InnoDB
autoradionica	odjel_radnik	InnoDB
autoradionica	radnik	InnoDB

```
/* Za prikaz naziva tablice, mehanizma pohrane i sličnih podataka moguće je
koristiti i: */
SHOW TABLE STATUS WHERE NAME IN ('odjel', 'radnik',  'odjel_radnik');
```

11.3. U sklopu baze radionica, izraditi tablicu *doprinosi_po_zup* koja po svim županijama prikazuje broj radnika i doprinos od radnika (računati ga kao 25% od plaće svih radnika u toj županiji). Nova tablica mora sadržavati zapise o svim županijama, bez obzira postoji li u njoj zapis o mjestu ili zaposlenom radniku. Novu tablicu izraditi iz (na temelju postojećih) tablica u bazi te automatski napuniti i aktualnim podatcima. Odabrati / promijeniti mehanizam pohrane podataka koji će omogućiti:

a) Zaključavanje podataka u bazi podataka na razini retka. Napisati primjer zaključavanja zbog čitanja za sve n-torke koje imaju manje od 10 radnika ili doprinos manji od 1000kn. Prikazati novi mehanizam pohrane podataka za tu tablicu. Navesti koje se vrste datoteka stvaraju na disku (pomoću SHOW VARIABLES WHERE Variable_Name LIKE "%dir").

b) Zaključavanje podataka u bazi podataka na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja zbog pisanja, nakon toga ih otključati). Navesti koje se vrste datoteka stvaraju na disku.

c) Što kraće vrijeme pretrage podataka, podršku za full text index. Navesti koje se vrste datoteka stvaraju na disku.

```
CREATE TABLE doprinosi_po_zup AS
  SELECT z.nazivZupanija,
    COUNT(r.sifRadnik) AS brRadnika,
    ROUND(SUM(r.KoefPlaca*IznosOsnovice*0.25),0) AS doprinos
  FROM zupanija z
    LEFT JOIN mjesto m ON z.sifZupanija = m.sifZupanija
    LEFT JOIN radnik r ON m.pbrMjesto  = r.pbrStan
  GROUP BY  z.nazivZupanija
  ORDER BY 2 DESC;

/* Kontrolni ispis iz novostvorene tablice */
SELECT * FROM doprinosi_po_zup;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SHOW VARIABLES WHERE Variable_Name LIKE "%dir" ;

/* Datoteke su pohranjene na lokaciji
C:\ProgramData\MySQL\MySQL Server 8.0\Data\autoradionica
ili za MySQL stariju verziju (5.7)
C:\ProgramData\MySQL\MySQL SERVER 5.7\DATA\autoradionica
ili
C:\UwAmp\bin\database\mysql-5.7.11\data\autoradionica
ili...

za InnoDB stvorena je datoteka:
doprinosi_po_zup.ibd  */
```

Variable_name	Value
basedir	C:\Program Files\MySQL\MySQL Server 8.0\
character_sets_dir	C:\Program Files\MySQL\MySQL Server 8.0\share\charsets\
datadir	C:\ProgramData\MySQL\MySQL Server 8.0\Data\
innodb_data_home_dir	
innodb_doublewrite_dir	
innodb_log_group_home_dir	.\
innodb_temp_tablespaces_dir	.\#innodb_temp\
innodb_tmpdir	
lc_messages_dir	C:\Program Files\MySQL\MySQL Server 8.0\share\
plugin_dir	C:\Program Files\MySQL\MySQL Server 8.0\lib\plugin\
replica_load_tmpdir	C:\WINDOWS\SERVIC~1\NETWOR~1\AppData\Local\Temp
slave_load_tmpdir	C:\WINDOWS\SERVIC~1\NETWOR~1\AppData\Local\Temp
tmpdir	C:\WINDOWS\SERVIC~1\NETWOR~1\AppData\Local\Temp

```
 -- a) ------------------------------------------------------------------------
ALTER TABLE doprinosi_po_zup ENGINE ARCHIVE;

SELECT *
  FROM doprinosi_po_zup
  WHERE brRadnika < 10 OR doprinos < 1000 LOCK IN SHARE MODE;

SHOW VARIABLES WHERE Variable_Name LIKE "%dir" ;

/* Na disku su stvorene datoteke (potražiti ih na disku na ranije pronađenoj
lokaciji C:\ProgramData\MySQL\MySQL Server 8.0\Data\autoradionica):
doprinosi_po_zup.sdi
doprinosi_po_zup.ARZ */

SHOW TABLE STATUS WHERE NAME = 'doprinosi_po_zup';

-- b) ------------------------------------------------------------------------
ALTER TABLE doprinosi_po_zup ENGINE MEMORY;

/* Stvorene datoteke:
doprinosi_po_zup.sdi  */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
LOCK TABLE doprinosi_po_zup WRITE;
UNLOCK TABLES;

-- c) ------------------------------------------------------------------------
ALTER TABLE doprinosi_po_zup ENGINE MYISAM;

/* Stvorene datoteke:
doprinosi_po_zup.sdi
doprinosi_po_zup.MYI
doprinosi_po_zup.MYD */
```

11.4. U bilo kojoj od baza (autoradionica ili studenti) izraditi tablicu *praksa* koja svakom radniku dodjeljuje sve studente kao potencijalne praktikante. Tablica sadrži sljedeće podatke: šifra radnika, ime i prezime radnika spojeno, jmbag studenta, njegovo ime i prezime spojeno te prosjek studenta. Odabrati / promijeniti mehanizam pohrane podataka u tablici *praksa* koji će omogućiti:

a) Zaključavanje podataka u bazi podataka na razini retka (napisati primjer zaključavanja zbog čitanja za sve n-torke koje imaju veći prosjek od 2.5). Prikazati novi mehanizam pohrane za tu tablicu. Navesti koje se vrste datoteka stvaraju na disku (pomoću SHOW VARIABLES WHERE Variable_Name LIKE "%dir").

b) Zaključavanje podataka u bazi podataka na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja tablice zbog pisanja, nakon toga otključati podatke). Navesti koje se vrste datoteka stvaraju na disku (provjeriti na disku).

c) Promijeniti mehanizam pohrane tablice praksa na odgovarajući kako bi radnici mogli jednostavno sa diska izvesti xls / csv za odabir studenata. Navesti koje se vrste datoteka stvaraju na disku. Možete li jednostavno iz te datoteke pročitati podatke?

```
CREATE TABLE praksa AS
  SELECT rs.*, oc.pros_ocj
  FROM
    (SELECT r.sifRadnik,
      CONCAT(r.imeRadnik,' ', r.prezimeRadnik) radnik,
      s.jmbag, CONCAT(s.ime,' ', s.prezime) student
      FROM studenti.studenti s
      JOIN autoradionica.radnik r ON s.postBrStanovanja = r.pbrStan
      WHERE r.pbrStan = 21000) rs
  INNER JOIN
    (SELECT o.jmbagStudent, AVG(o.ocjena) pros_ocj
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
      FROM studenti.ocjene o
      GROUP BY 1) oc ON rs.jmbag=oc.jmbagStudent
  ORDER BY 1, 2, 5 DESC;

/* Stvorenu datoteku pronaći na C:\ProgramData\MySQL\MySQL Server
8.0\Data\autoradionica: (za InnoDB) praksa.ibd */

-- a) ------------------------------------------------------------------------
ALTER TABLE praksa ENGINE ARCHIVE;

SELECT * FROM studenti.praksa
WHERE pros_ocj > 2.5 LOCK IN SHARE MODE;

/* Stvorene datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\autoradionica:
praksa.sdi
praksa.ARZ */

SHOW TABLE STATUS WHERE NAME = 'praksa';

-- b) ------------------------------------------------------------------------
ALTER TABLE praksa ENGINE MYISAM;

/* Stvorene datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\autoradionica:
praksa.sdi
praksa.MYI
praksa.MYD */

LOCK TABLE praksa WRITE;
UNLOCK TABLES;

-- c) ------------------------------------------------------------------------
ALTER TABLE praksa ENGINE CSV;

/* Pokušaj promjene na CSV rezultirati će slijedećom porukom (u idućem koraku
potrebno je za sve atraibute definirati da ne smiju poprimiti NULL vrijednsot
ERROR Code: 1178
The storage engine for the TABLE doesn't support nullable columns
*/

ALTER TABLE praksa
  MODIFY sifRadnik INT NOT NULL,
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
  MODIFY radnik VARCHAR(101) NOT NULL,
  MODIFY jmbag  CHAR(10) NOT NULL,
  MODIFY student VARCHAR(101) NOT NULL,
  MODIFY pros_ocj DECIMAL(7,4) NOT NULL;

ALTER TABLE praksa ENGINE CSV;

/* Stvorene datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\autoradionica:
praksa.CSM
praksa.CSV
Da, iz datoteke praksa.CSV moguće je jednostavno (vizualno) pročitati podatke i
pripremiti popis studenata za radnike.
*/

SHOW TABLE STATUS WHERE NAME = 'praksa';
```

11.5. U sklopu baze autoradionica, izraditi tablicu *kvarovi_po_god* koja po godinama i nazivima kvarova prikazuje broj naloga i prosječne ostvarene sate. Odabrati samo one retke koji imaju više od jednog naloga. Sortirati po broju kvarova padajući i godinama rastući. Odabrati / promijeniti mehanizam pohrane podataka koji će omogućiti:

a) Zaključavanje podataka u bazi podataka na razini retka. Napisati primjer zaključavanja zbog čitanja za sve n-torke koje imaju između 10 i 20 naloga ili prosjek sati između 6 i 7. Prikazati novi mehanizam pohrane podataka za tu tablicu. Navesti koje se vrste datoteka stvaraju na disku (pomoću SHOW VARIABLES WHERE Variable_Name LIKE "%dir").

b) Zaključavanje podataka u bazi podataka na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja zbog pisanja, nakon toga ih otključati). Navesti koje se vrste datoteka stvaraju na disku.

c) Nema ograničenja na veličinu baze. Navesti koje se vrste datoteka stvaraju na disku.

```
CREATE TABLE kvarovi_po_god AS
  SELECT YEAR(n.datPrimitkaNalog) AS godina, k.nazivKvar,
    COUNT(*) AS br_kvarova, ROUND(AVG(n.OstvareniSatiRada),2) AS pros_sati
  FROM nalog n NATURAL JOIN kvar k
  GROUP BY YEAR(n.datPrimitkaNalog), k.nazivKvar
  HAVING COUNT(*) > 1
  ORDER BY 3 DESC, 1;
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SHOW VARIABLES WHERE Variable_Name LIKE "%dir";

-- a) ---------------------------------------------------------------------
ALTER TABLE kvarovi_po_god ENGINE ARCHIVE;

SELECT *
  FROM kvarovi_po_god
  WHERE br_kvarova BETWEEN 10 AND 20  OR pros_sati BETWEEN 6 AND 7
  FOR SHARE;

SHOW VARIABLES WHERE Variable_Name LIKE "%dir" ;

/* Stvorene datoteke:
kvarovi_po_god.sdi
kvarovi_po_god.ARZ */

SHOW TABLE STATUS WHERE NAME = 'kvarovi_po_god';

-- b) ---------------------------------------------------------------------
ALTER TABLE kvarovi_po_god  ENGINE MyISAM;

/* Stvorene datoteke:
kvarovi_po_god.sdi
kvarovi_po_god.MYI
kvarovi_po_god.MYD */

LOCK TABLE kvarovi_po_god WRITE;
UNLOCK TABLES;

-- c) ---------------------------------------------------------------------
ALTER TABLE kvarovi_po_god ENGINE ARCHIVE;

/* Stvorene datoteke:
kvarovi_po_god.sdi
kvarovi_po_god.ARZ
*/

SHOW TABLE STATUS WHERE NAME = 'kvarovi_po_god';
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

11.6. U bazi autoradionica potrebno je izraditi tablicu *poklon_bon* koja sadrži podatke o bonovima (kuponima) kod automehaničara. Korisnici bonova su nastavnici iz Zagreba i daju se za sve nadodjele u radionici (Kartezijev produkt). Tablica sadrži sljedeće podatke: jmbg nastavnika, ime i prezime nastavnika spojeno, šifru nadodjela i naziv nadodjela. Odabrati / promijeniti mehanizam pohrane podataka koji će omogućiti:

a) Zaključavanje podataka u bazi podataka na razini retka (napisati primjer zaključavanja zbog čitanja za sve n-torke koje pripadaju ili nadodjelu 'Elektrika' ili 'Optika'. Prikazati novi mehanizam pohrane za tu tablicu. Navesti koje se vrste datoteka stvaraju na disku (pomoću SHOW VARIABLES WHERE Variable_Name LIKE "%dir").

b) Zaključavanje podataka u bazi podataka na razini tablice, ali ne i na razini retka (napisati primjer zaključavanja tablice zbog čitanja, nakon toga otključati podatke). Navesti koje se vrste datoteka stvaraju na disku.

c) Promijeniti mehanizam pohrane tablice klijent na odgovarajući kako bi nastavnici dobili xls/csv za odabir nadodjela. Ako je potrebno, obaviti potrebne predradnje. Navesti koje se vrste datoteka stvaraju na disku. Možete li jednostavno iz te datoteke pročitati podatke?

```
CREATE TABLE poklon_bon AS
  SELECT n.jmbg, CONCAT(n.ime,' ', n.prezime) nastavnik,
    o.sifNadOdjel, o.nazivOdjel
  FROM studenti.nastavnici n,
    (SELECT DISTINCT od.sifNadOdjel, nod.nazivOdjel
      FROM autoradionica.odjel od JOIN autoradionica.odjel nod
      ON od.sifNadOdjel = nod.sifOdjel) o
  WHERE n.postBr = 10000
  ORDER BY 1, 3;

SELECT * FROM poklon_bon;

-- a) ------------------------------------------------------------
ALTER TABLE poklon_bon ENGINE ARCHIVE;

SELECT * FROM studenti.poklon_bon
  WHERE nazivOdjel IN ('Elektrika','Optika') LOCK IN SHARE MODE;

SHOW VARIABLES WHERE Variable_Name LIKE "%dir"

/* Stvorene su datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\studenti:
poklon_bon.sdi
poklon_bon.ARZ */
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
SHOW TABLE STATUS WHERE NAME = 'poklon_bon';



-- b) -------------------------------------------------------------
ALTER TABLE poklon_bon ENGINE MYISAM;

/* Stvorene su datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\studenti:
poklon_bon.sdi
poklon_bon.MYI
poklon_bon.MYD   */

LOCK TABLE poklon_bon READ;
UNLOCK TABLES;

-- c) -------------------------------------------------------------
ALTER TABLE poklon_bon  ENGINE CSV;

/* Nakon ove naredbne javlja se greška jer svi atributi tablice koja je na CSV-u
moraju biti NOT NULL
-- ERROR Code: 1178. The storage engine for the TABLE doesn't support nullable
columns
Zato je potrebno atribute definirati kao NOT NULL. */

ALTER TABLE poklon_bon
  MODIFY nastavnik VARCHAR(101) NOT NULL,
  MODIFY sifNadOdjel INT(11) NOT NULL,
  MODIFY nazivOdjel VARCHAR(50) NOT NULL;

ALTER TABLE poklon_bon  ENGINE CSV;

/* Stvorene su datoteke na C:\ProgramData\MySQL\MySQL Server
8.0\Data\autoradionica:
poklon_bon.CSM
poklon_bon.CSV

Da, iz datoteke poklon_bon.CSV moguće je jednostavno (vizualno) pročitati
podatke.*/

SHOW TABLE STATUS WHERE NAME = 'poklon_bon';
```

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

11.7. Izraditi tablice jednake strukture *(ID, name, created_at)* temeljene na mehanizmima pohrane (InnoDB, MyISAM i CSV).

Napuniti tablice sa 1000 – 10000 redaka (generirati pomoću procedure / Excel tablice / PHP skripte, …).

Sve tablice neka sadrže jednak broj redaka i iste podatke.

Detektirati koje vrste datoteka se stvaraju kod InnoDB, MyISAM i CSV mehanizama (.csm, .frm, .ibd, .myd, .myi, .csv).

Usporediti veličinu zapisa na disku za InnoDB, MyISAM i CSV tablice s jednakom količinom podataka (istim podatcima).

```
CREATE TABLE test1 (
  ID INT NOT NULL,
  NAME VARCHAR(40) NOT NULL,
  CREATED_AT TIMESTAMP NOT NULL)
  ENGINE=InnoDB DEFAULT CHARSET=cp1250 COLLATE=cp1250_croatian_ci;

CREATE TABLE test2 (
  ID INT NOT NULL,
  NAME VARCHAR(40) NOT NULL,
  CREATED_AT TIMESTAMP NOT NULL)
  ENGINE=MyISAM  DEFAULT CHARSET=cp1250 COLLATE=cp1250_croatian_ci;

CREATE TABLE test3 (
  ID INT NOT NULL,
  NAME VARCHAR(40) NOT NULL,
  CREATED_AT TIMESTAMP NOT NULL)
  ENGINE=CSV DEFAULT CHARSET=cp1250 COLLATE=cp1250_croatian_ci;

-- izrada procedure za punjenje tablica
DROP PROCEDURE IF EXISTS puni_tablice;
DELIMITER $$
CREATE PROCEDURE puni_tablice()
BEGIN
  DECLARE var INT;
  SET var=1;
  WHILE var<=1000 DO
    INSERT INTO test1
      VALUES (var, concat('ime_prezime','-',var), CURRENT_TIMESTAMP());
    INSERT INTO test2
      VALUES (var, concat('ime_prezime','-',var), CURRENT_TIMESTAMP());
```

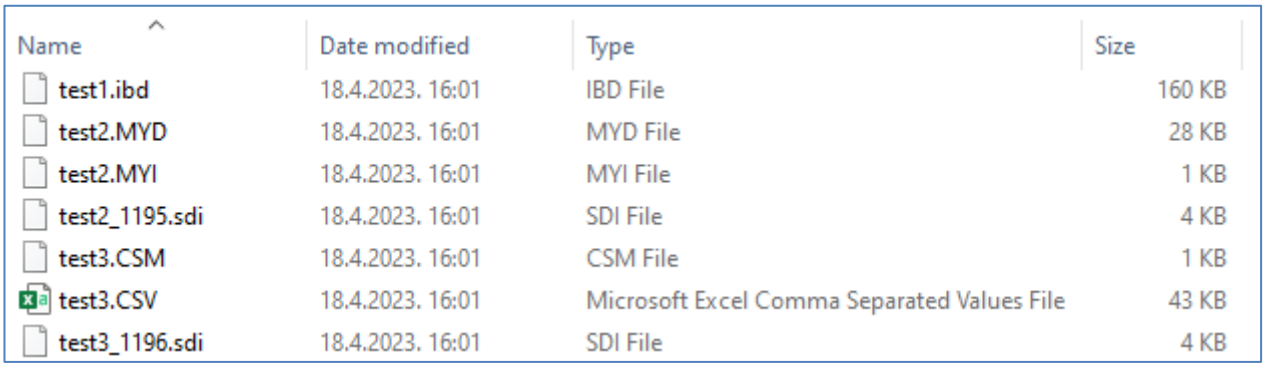
© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

```
    INSERT INTO test3
      VALUES (var, concat('ime_prezime','-',var), CURRENT_TIMESTAMP());
  SET var = var+1;
  END WHILE;
END;
$$
DELIMITER ;

-- punjenje tablica
CALL puni_tablice();

SHOW VARIABLES WHERE Variable_Name LIKE "%dir";
```

Na disku vidljivo je da će podatci pohranjeni na InnoDB mehanizmu pohrane zauzeti najviše diskovnog prostora.



Name	Date modified	Type	Size
test1.ibd	18.4.2023. 16:01	IBD File	160 KB
test2.MYD	18.4.2023. 16:01	MYD File	28 KB
test2.MYI	18.4.2023. 16:01	MYI File	1 KB
test2_1195.sdi	18.4.2023. 16:01	SDI File	4 KB
test3.CSM	18.4.2023. 16:01	CSM File	1 KB
test3.CSV	18.4.2023. 16:01	Microsoft Excel Comma Separated Values File	43 KB
test3_1196.sdi	18.4.2023. 16:01	SDI File	4 KB

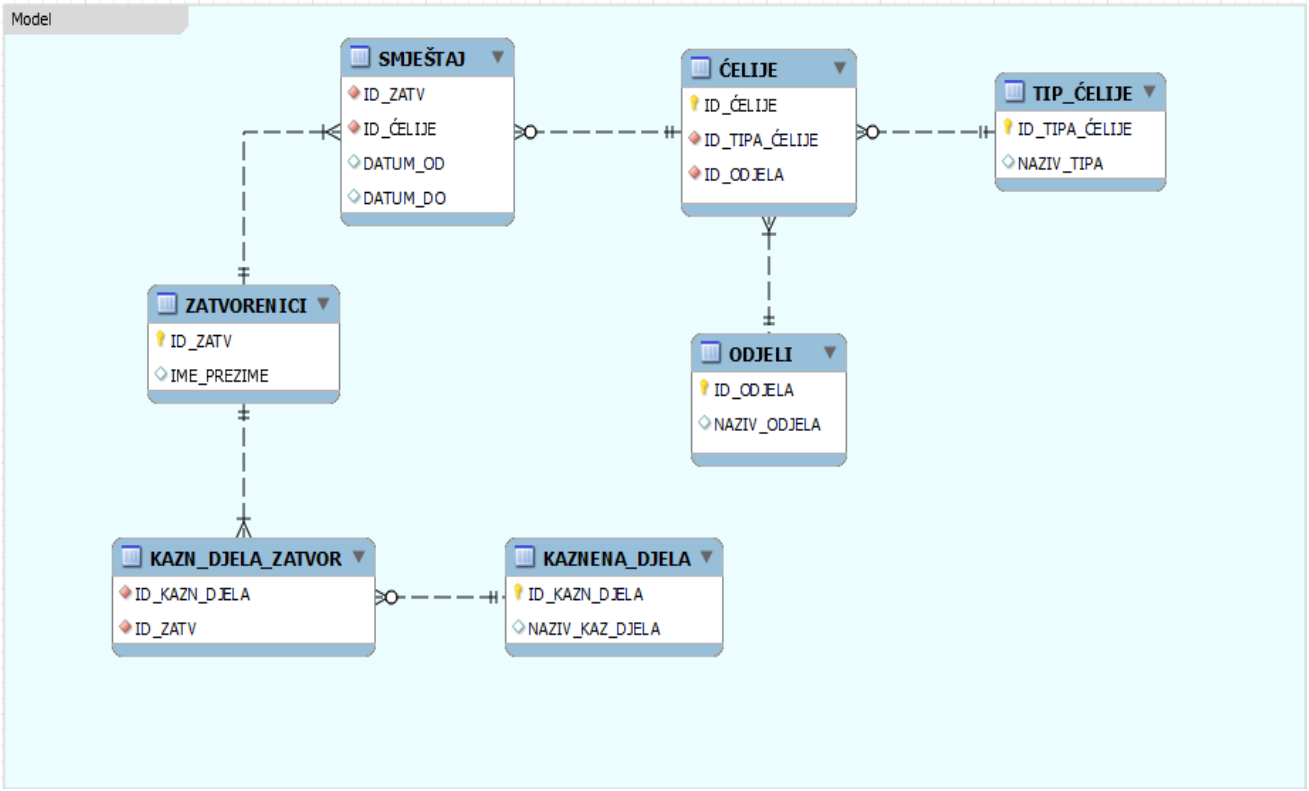
Slika 7. Prikaz kreiranih datoteka za različite mehanizme pohrane

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12. Modeliranje

12.1. Koristeći IE notaciju (*Information Engineering*) izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Smještaj zatvorenika u ćelijama
- Vrste (tipovi) ćelija
- Pripadnost ćelija odjelima
- Učinjena kaznena djela zatvorenika

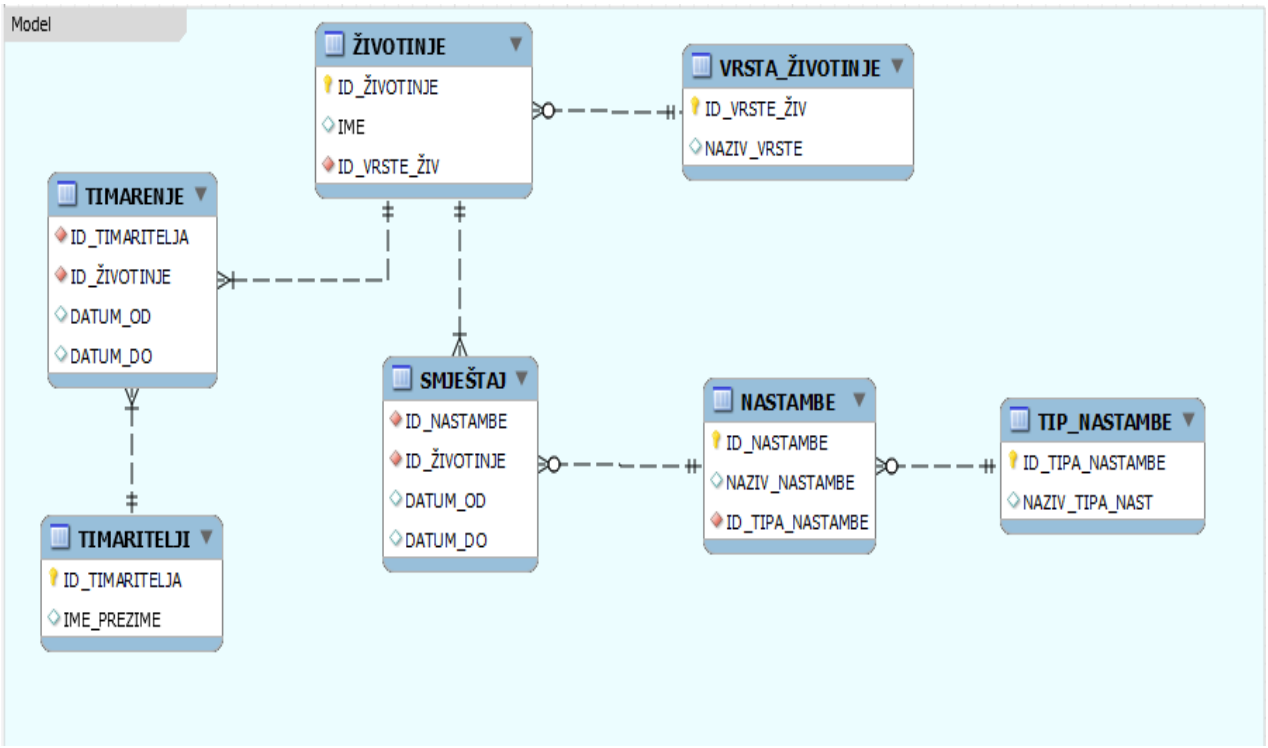


Slika 8. Segment modela podataka - Zatvorenici

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.2. Koristeći IE notaciju (*Information Engineering*) izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Životinje u ZOO vrtu
- Pripadnost životinje vrsti životinja
- Smještaj životinja po nastambama
- Nastambe i tipovi nastambi
- Zaduženost timaritelja za životinje

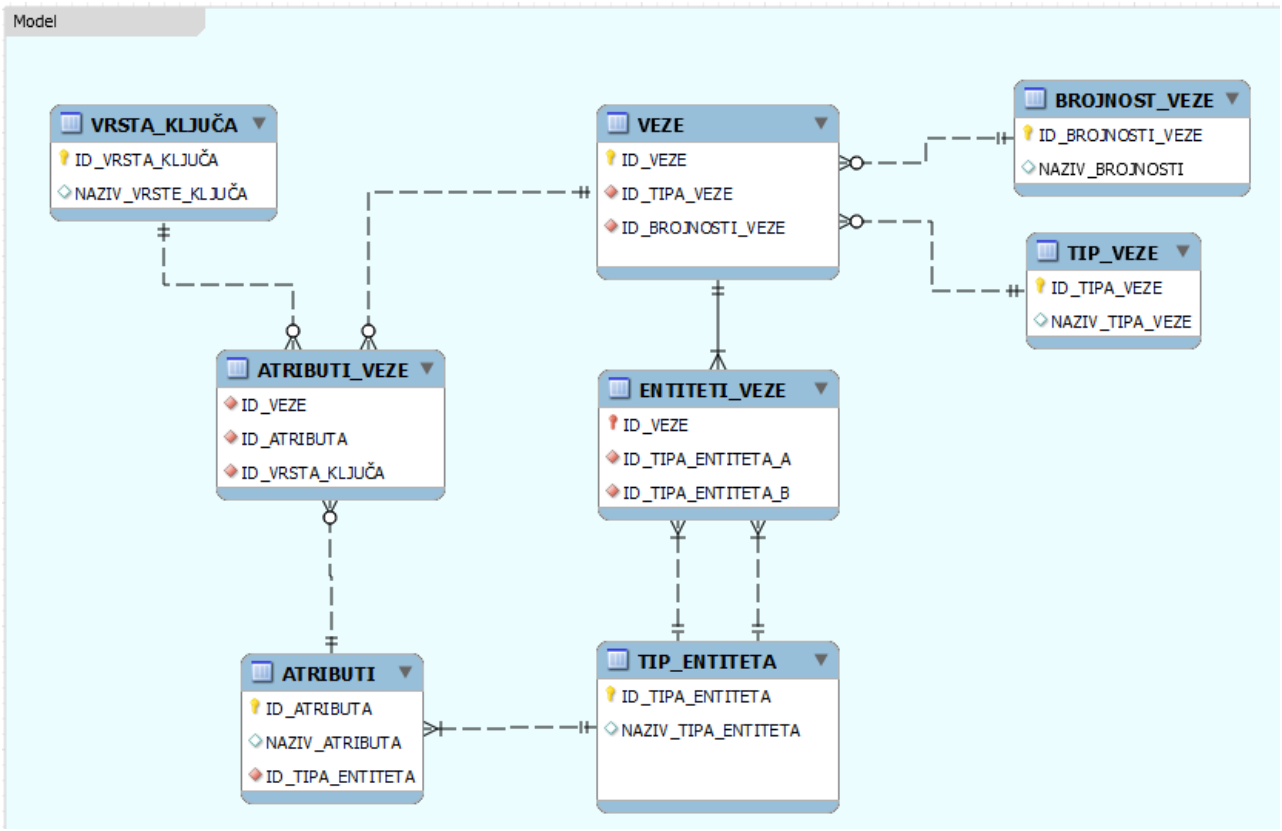


Slika 9. Segment modela podataka – ZOO

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.3. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Tipovi entiteta meta modela baze podataka i njihove veze
- Pripadnost atributa tipovima entitetima
- Atributi veze i vrste ključeva
- Tipovi veza i brojnost veze

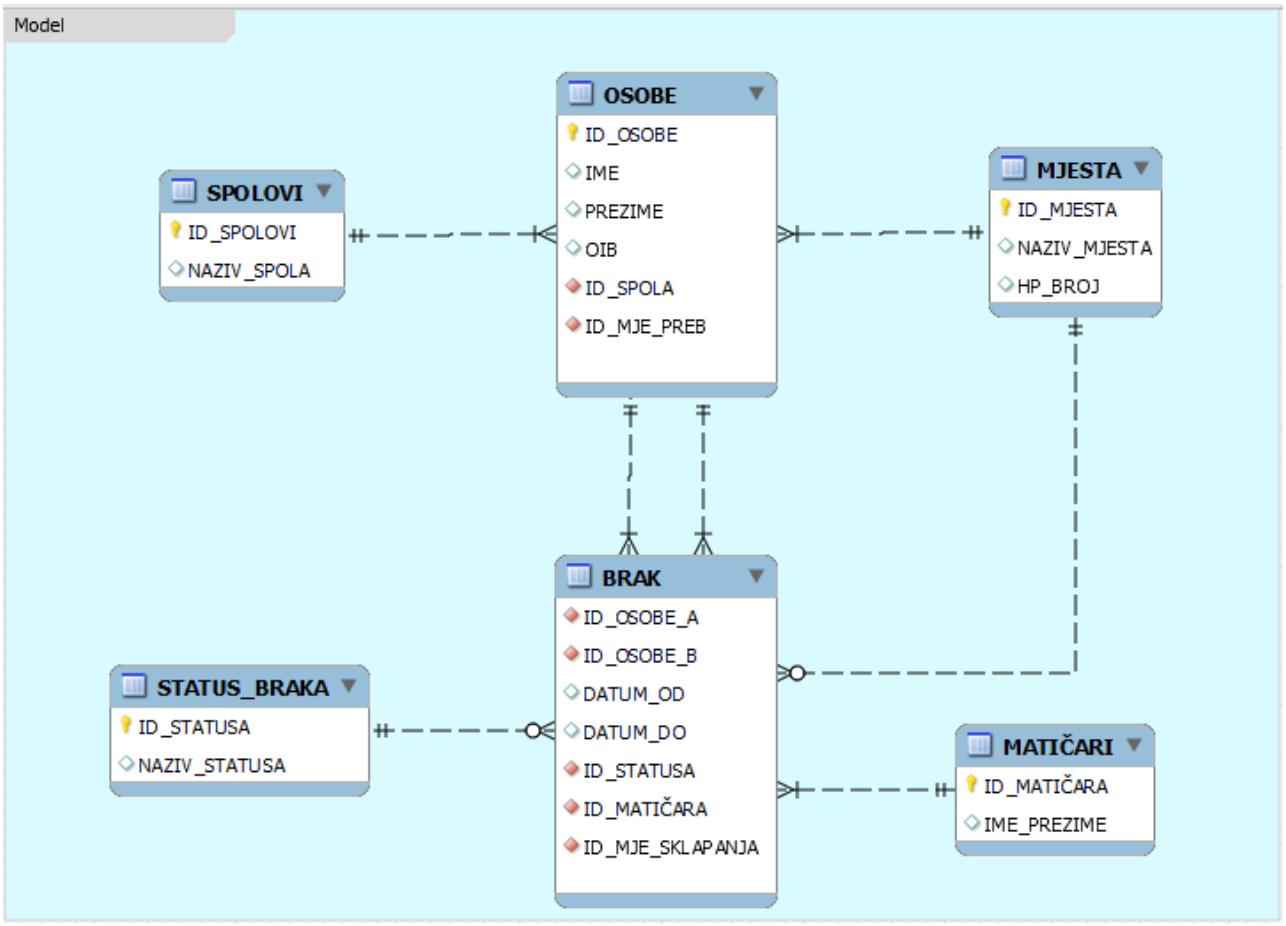


Slika 10. Segment modela podataka - Meta model podataka

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.4. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Osobe u RH
- Sklopljen brak između dvije osobe
- Spol osobe
- Status braka
- Službenik / matičar koji je ozakonio brak
- Mjesto prebivališta osobe
- Mjesto sklapanja braka

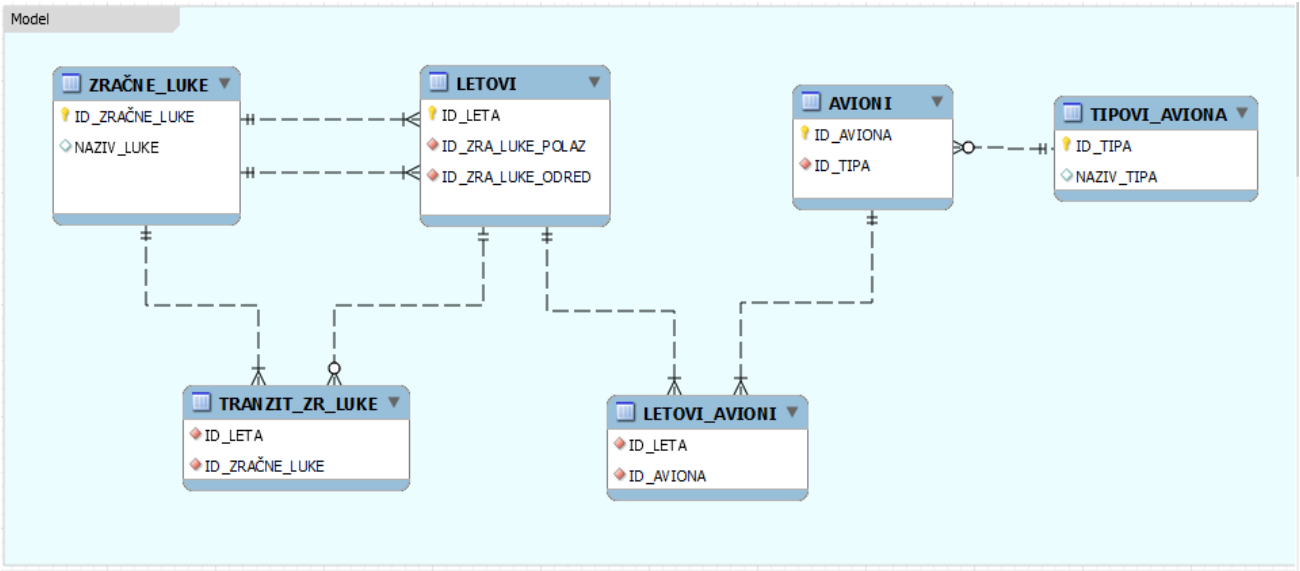


Slika 11. Segment modela podataka - Matica vjenčanih

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.5. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Flota aviona jedne aviokompanije
- Tipovi aviona
- Zračne luke kao polazište, odredište pojedinih letova
- Tranzitne luke na letu (zračne luke slijetanja / polijetanja)

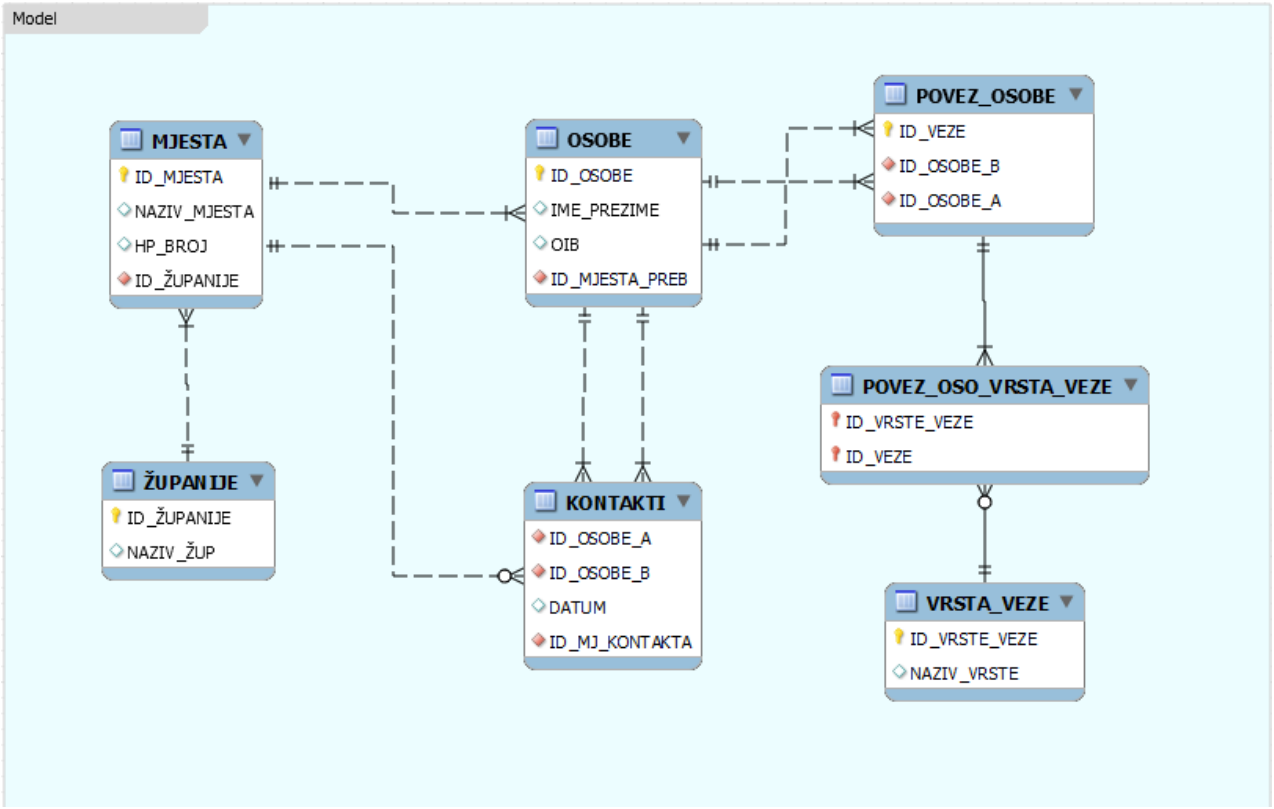


Slika 12. Segment modela podataka - Zračna luka

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.6. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja:

- Osobe i tipovi veza između njih
- Mjesta rođenja osoba
- Pripadnost mjesta županijama
- Ostvareni kontakti COVID (suspektno)
- Mjesta ostvarenog kontakta

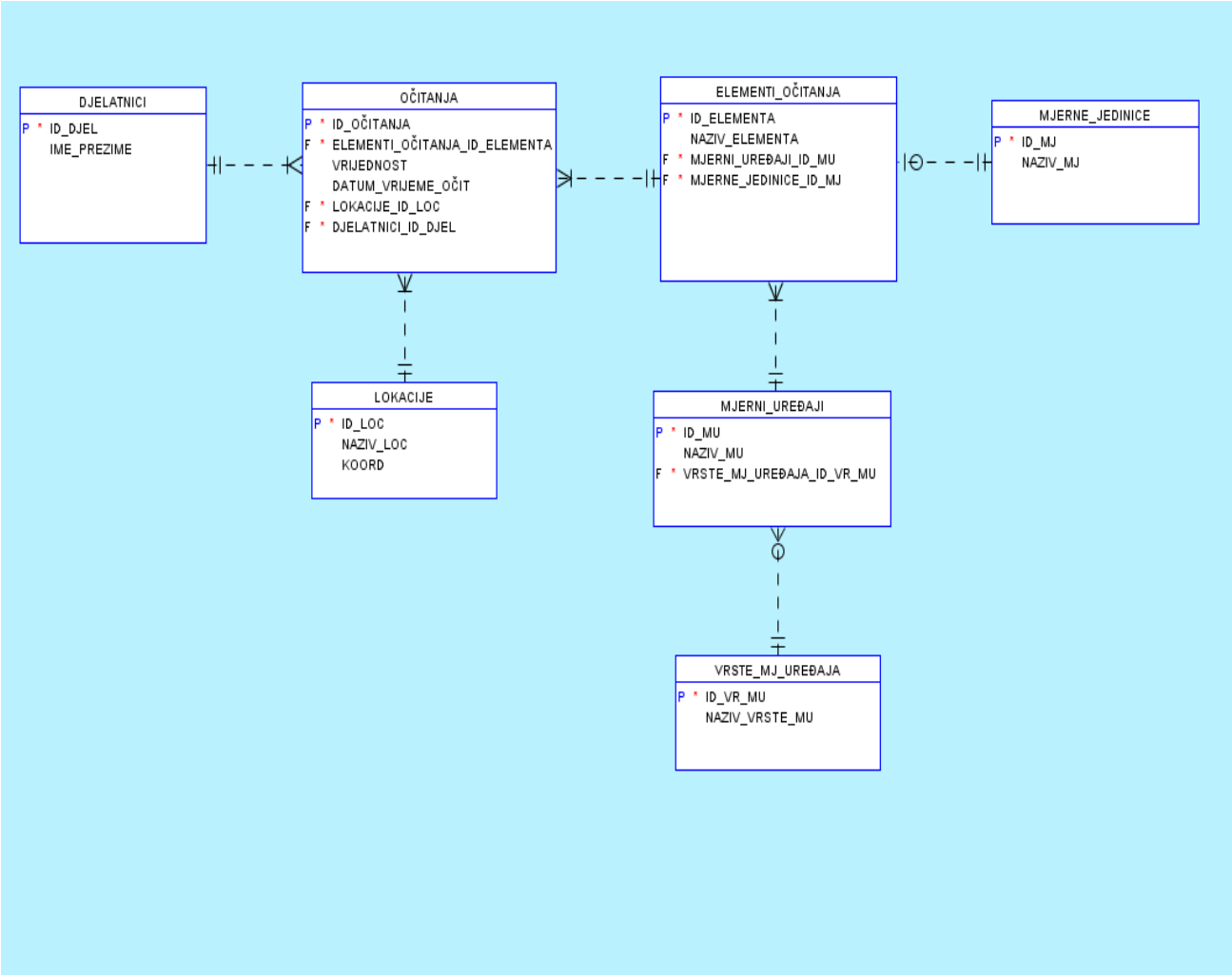


Slika 13. Segment modela podataka – COVID kontakti

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.7. Koristeći IE notaciju izraditi segment logičkog modela podataka „Očitanja meteo službe“ (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja *(oznake: P=Primarni ključ, F=Strani ključ):*

- Djelatnici i učinjena očitanja
- Lokacije očitanja
- Elementi očitanja
- Mjerni uređaji
- Vrste mjernih uređaja
- Mjerne jedinice

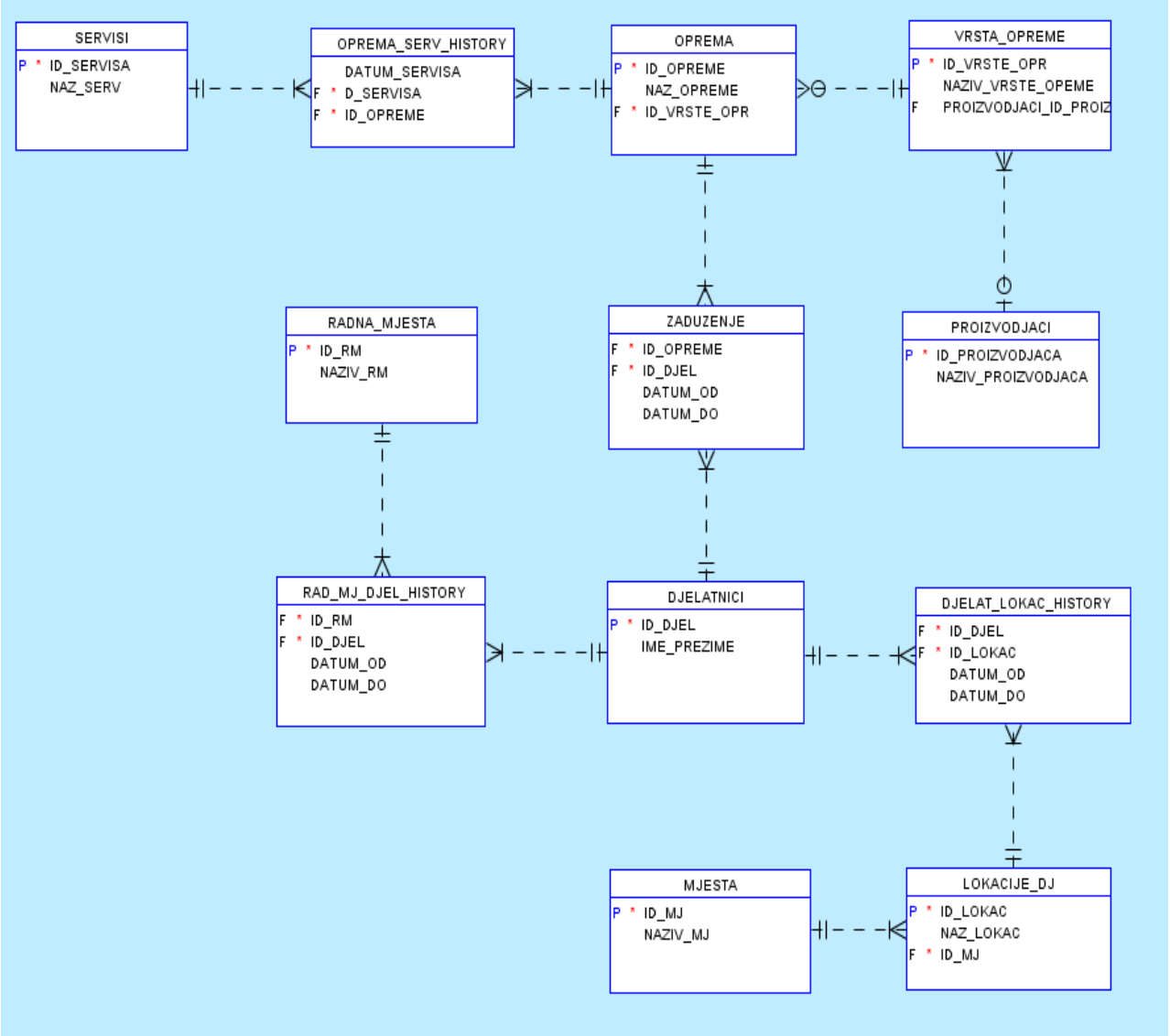


Slika 14. Segment modela podataka – Meteorološka služba

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.8. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja *(oznake: P=Primarni ključ, F=Strani ključ):*

- Djelatnici i radna mjesta
- Lokacije radnih mjesta
- Zaduženja djelatnika IT opremom
- Vrste IT opreme i proizvođači
- Servisiranje opreme u servisima

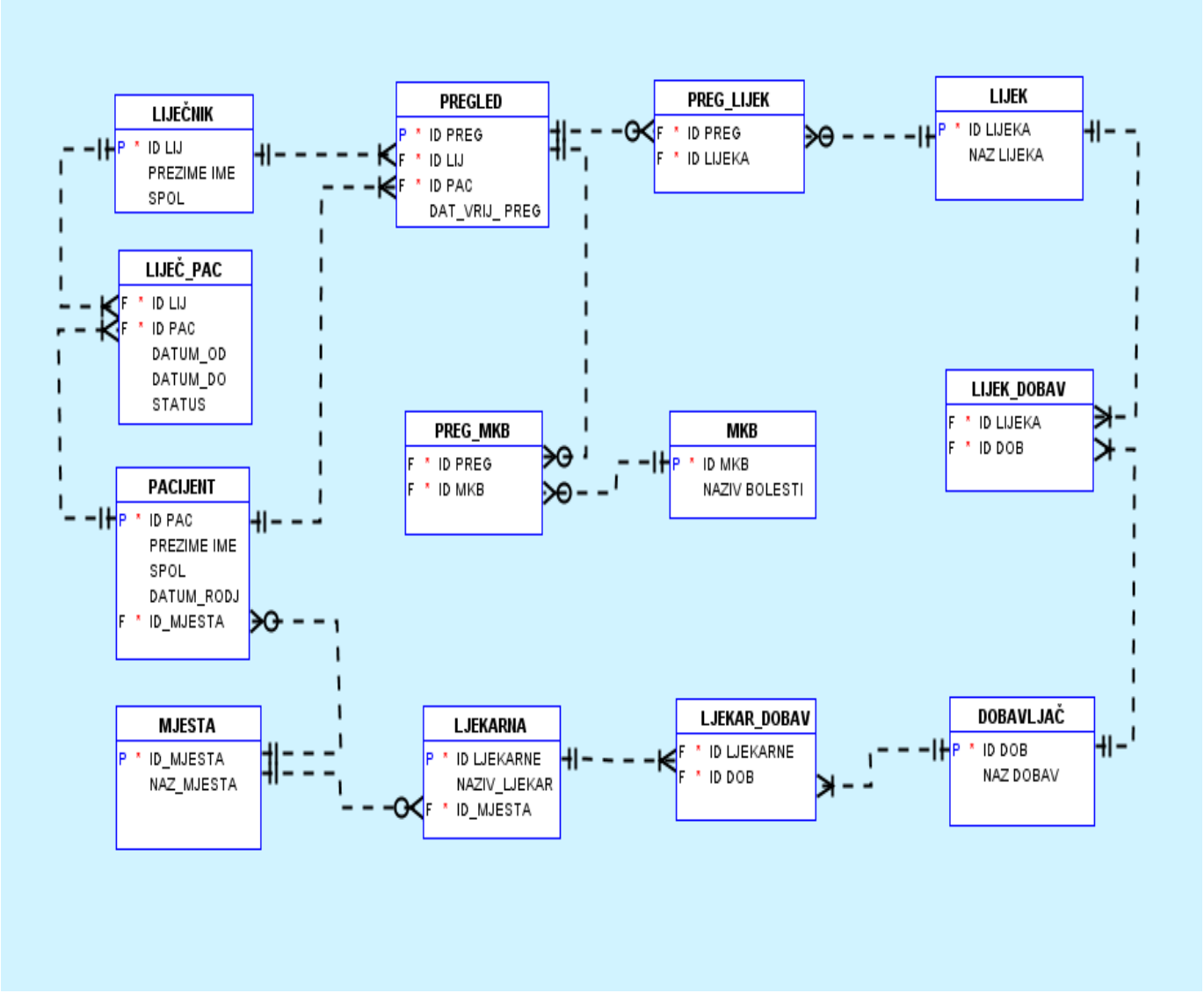


Slika 15. Segment modela podataka – Zaduženje opremom

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

12.9. Koristeći IE notaciju izraditi segment logičkog modela podataka (relacijski model) koji treba omogućiti prikazivanje sljedećih znanja: *(oznake: P=Primarni ključ, F=Strani ključ):*

- Pripadnost pacijenata liječnicima (Primarna zdravstvena zaštita)
- Obavljeni pregledi
- Ustanovljene bolesti
- „Prepisani“ lijekovi
- Ljekarne i dobavljači lijekova
- Mjesta prebivališta pacijenata
- Lokacije ljekarni (mjesta)



Slika 16. Segment modela podataka – Primarna zdravstvena zaštita

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Popis slika

Slika 1. Shema baze autoradionica 6
Slika 2. Shema baze studenti 7
Slika 3. Spajanje na poslužitelj kao korisnik stud_user 94
Slika 4. Pogled na dostupne tablice za korisnika stud_user 95
Slika 5. Rezultat nakon što je korisnik admin pokrenuo proceduru izbrojiKvarove 100
Slika 6. Korisnik kontrolor pokreće funkciju brSati 101
Slika 7. Prikaz kreiranih datoteka za različite mehanizme pohrane 121
Slika 8. Segment modela podataka - Zatvorenici 122
Slika 9. Segment modela podataka - ZOO 123
Slika 10. Segment modela podataka - Meta model podataka 124
Slika 11. Segment modela podataka - Matica vjenčanih 125
Slika 12. Segment modela podataka - Zračna luka 126
Slika 13. Segment modela podataka – COVID kontakti 127
Slika 14. Segment modela podataka – Meteorološka služba 128
Slika 15. Segment modela podataka – Zaduženje opremom 129
Slika 16. Segment modela podataka – Primarna zdravstvena zaštita 130

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.

Literatura

1. Gračanin, J.: „Napredne baze podataka“, materijali za predavanja i vježbe, Tehničko veleučilište u Zagrebu, 2014 - 2023.
2. Kraljević, S.: „Napredne baze podataka“, materijali za predavanja i vježbe, Tehničko veleučilište u Zagrebu, 2009 - 2023.
3. Kramberger, T; Duk, S.; Kovačević, R.: „Baze podataka“, udžbenik za kolegije „Baze podataka“ i „Napredne baze podataka“, Tehničko veleučilište u Zagrebu, 2018.
4. MySQL dokumentacija: „MySQL Reference Manuals“, URL: https://dev.mysql.com/doc/

© Autori: Sanja Kraljević, dipl. ing., Jakob Gračanin, dipl. ing.