



# Zadatci za vježbu

## #10

[Previous step](#)[Submit answer](#)[Next step](#)[Layout ▼](#)[Tutorial content ▼](#)

### 4/4: Kontekstno ovisne dozvole

#### Primjer 1.

Korisnik *postgres* mijenja shemu relacije *student* - dodaje atribut *korisnik*. Vrijednost tog atributa treba postaviti na korisničko ime studenta.

Studentu treba dodijeliti dozvole pregledavanja vlastitih podataka tako da kada student uspostavi SQL sjednicu s vlastitim korisničkim imenom (CURRENT\_USER), temeljem njegovog korisničkog imena su "poznati" i podaci koje smije pregledavati.

Naredbe kojima *postgres* mijenja shemu relacije *student* i postavlja vrijednst atributa *korisnik* za jednog studenta izvedite u **studAdmin** bazi:

```
ALTER TABLE student ADD korisnik VARCHAR (10);
```

```
UPDATE student SET prezimeStudent = 'Kolar', korisnik = 'kolar' WHERE jmbag = '0555005460';
```

Napisati niz SQL naredbi kojima će se svim studentima kojima je dodijeljeno korisničko ime, pomoću uloge *studentUloga* omogućiti pregled svih podataka iz relacije *predmet* te samo vlastitih podataka iz relacije *upisanPredmet*. Navedene dozvole dodijeliti korisniku *kolar*.

[rješenje](#)

```
CREATE VIEW mojPredmet AS
  SELECT upisanPredmet.*
    FROM upisanPredmet, student
   WHERE upisanPredmet.JMBAG = student.JMBAG
      AND korisnik = CURRENT_USER;
```

```
CREATE ROLE studentUloga; --preddefinirano uključuje INHERIT opciju
GRANT USAGE ON SCHEMA PUBLIC to studentUloga;
GRANT SELECT ON predmet TO studentUloga;
GRANT SELECT ON mojPredmet TO studentUloga;
GRANT studentUloga TO kolar;
```

Da bi korisnik *kolar* mogao uspostaviti korisničku sjednicu s bazom **studAdmin** mora imati barem dozvolu CONNECT. Pretpostavlja se da je *kolar* dobio CONNECT u prethodnom koraku tutoriala.

--kolar:

```
SELECT * FROM predmet;
SELECT * FROM mojPredmet;
```

Alternativno rješenje:

--postgres:

```
CREATE VIEW mojPredmet AS
  SELECT upisanPredmet.*
    FROM upisanPredmet, student
   WHERE upisanPredmet.JMBAG = student.JMBAG
      AND korisnik = SESSION_USER;
```

```
CREATE ROLE studentUloga;
GRANT USAGE ON SCHEMA PUBLIC to studentUloga;
GRANT SELECT ON predmet TO studentUloga;
GRANT SELECT ON mojPredmet TO studentUloga;
GRANT studentUloga TO kolar;
```

--kolar:

[Feedback](#)

```
SET ROLE studentUloga --obavezno ako je korisnik kreiran pomoću opcije NOINHERIT
SELECT * FROM predmet;
SELECT * FROM mojPredmet;
```

## Primjer 2.

Pretpostaviti da je u relaciju *nastavnik* dodan atribut *korisnik* koji je jednak korisničkom imenu (CURRENT\_USER) s kojim korisnik uspostavlja SQL sjednicu.

Naredbe kojima *postgres* mijenja shemu relacije *nastavnik* i postavlja vrijednost atributa *korisnik* za dva nastavnika izvedite u **studAdmin** bazi:

```
ALTER TABLE nastavnik ADD korisnik VARCHAR (10);

UPDATE nastavnik SET prezimeNastavnik = 'Horvat', korisnik = 'horvat' WHERE sifnastavnik = 484;
UPDATE nastavnik SET prezimeNastavnik = 'Novak', korisnik = 'novak' WHERE sifnastavnik = 723;
```

Napisati niz SQL naredbi kojima će se svim nastavnicima kojima je dodijeljeno korisničko ime, pomoću uloge *nastavnikUloga* omogućiti pregled vlastitih podataka iz relacije *predmetGrupa*. Navedene dozvole dodijeliti korisniku *horvat*.

Zatim napisati niz SQL naredbi kojima će se svim nastavnicima koordinatorima pomoću uloge *koordinatorUloga* omogućiti pregled i izmjena podataka u relaciji *predmetGrupa*, ali samo za one nastavne grupe predmeta koje predaju nastavnici koji pripadaju istoj organizacijskoj jedinici (zavodu) kojoj pripada i nastavnik koordinator. Navedene dozvole dodijeliti korisniku *novak*.

Osigurati da korisnici *horvat* i *novak* pristupaju podacima u relaciji *predmetGrupa* preko pogleda istog imena *grupaZaSve*.

rješenje

--postgres:

```
CREATE VIEW nastavnikGrupa AS
SELECT predmetGrupa.*
  FROM predmetGrupa, nastavnik
 WHERE predmetGrupa.sifNastavnik = nastavnik.sifNastavnik
    AND nastavnik.korisnik = CURRENT_USER;

CREATE VIEW koordinatorGrupa AS
SELECT * FROM predmetGrupa
 WHERE sifNastavnik IN (
    SELECT nastIzIstogOrgJed.sifNastavnik
      FROM nastavnik, nastavnik nastIzIstogOrgJed
    WHERE nastavnik.sifOrgJed = nastIzIstogOrgJed.sifOrgjed
      AND nastavnik.korisnik = CURRENT_USER)
 WITH CHECK OPTION;

CREATE ROLE nastavnikUloga;
CREATE ROLE koordinatorUloga;
GRANT SELECT ON nastavnikGrupa TO nastavnikUloga;
GRANT SELECT, UPDATE ON koordinatorGrupa TO koordinatorUloga;

GRANT nastavnikUloga TO horvat;
GRANT koordinatorUloga TO novak;

CREATE SCHEMA AUTHORIZATION horvat;
CREATE SCHEMA AUTHORIZATION novak;

CREATE VIEW horvat.grupaZaSve AS SELECT * FROM nastavnikGrupa;
CREATE VIEW novak.grupaZaSve AS SELECT * FROM koordinatorGrupa;

GRANT SELECT ON horvat.grupaZaSve TO horvat;
GRANT SELECT, UPDATE ON novak.grupaZaSve TO novak;

--GRANT CONNECT ON DATABASE "studAdmin" TO horvat, novak; --dobili u 1. koraku tutoriala
```

--horvat:

Feedback

```
SELECT * FROM grupaZaSve;
```

Rezultat upita su podaci o nastavnim grupama za samo one predmete koje predaje nastavnik koji je izvršio upit (*horvat*).

--novak:

```
SELECT * FROM grupaZaSve;
```

Rezultat upita su podaci o nastavnim grupama za predmete koje predaju nastavnici koji pripadaju istoj organizacijskoj jedinici (zavodu) kao nastavnik koordinator koji je izvršio upit (*novak*).

```
UPDATE grupaZaSve SET oznDvorana = 'D278'  
WHERE sifPredmet = 33 AND akGodina = 2017  
AND sifNastavnik = 720 AND oznGrupa = 'A-I';
```

```
SELECT *  
FROM grupaZaSve  
WHERE sifPredmet = 33 AND akGodina = 2017  
AND sifNastavnik = 720 AND oznGrupa = 'A-I';
```

Nastavnik koordinator ima još dozvole za izmjenu podataka.

[Question](#)[Playground](#)

Zašto u primjeru 1 korisnik *kolar* u prvoj verziji rješenja ne mora izvršiti naredbu: `SET ROLE studentUloga;`?

Što bi se dogodilo kada bi korisnik *kolar* izvršio navedenu naredbu `SET ROLE studentUloga;` prije dohvata podataka? Koju vrijednost će poprimiti varijabla `CURRENT_USER`?

**a**

INHERIT znači da uloga (automatski) nasljeđuje dozvole eventualnih dodatnih uloga koje su joj dodijeljene, drugim riječima čim uspostavi sjednicu, pored svih ostalih dozvola, *kolar* automatski ima sve dozvole uloge *studentUloga*.

Ne bi se dogodilo ništa loše - i dalje bi mogao obaviti sve. `CURRENT_USER` postaje *studentUloga*.

**b**

Rješenje nije ispravno. *kolar* je trebao obaviti navedenu naredbu.

**c**

Ne znam.