

# **2. Auditorne vježbe**

## **Baze podataka**

***Fakultet elektrotehnike i računarstva***

## U auditornim vježbama obradit će se:

- Upravljanje istodobnim pristupom
- Optimiranje upita
- ER model baze podataka:
  - opis problema → ER model
  - ER model → relacijski modelA
- Sigurnost
- Okidači i pohranjene procedure
- Privremene i virtualne relacije

# 1. Upravljanje istodobnim pristupom

U zadatku se podrazumijeva korištenje PostgreSQL SUBP-a.

Nad bazom podataka su aktivne samo dvije sjednice: A i B.

Korisnici u sjednicama izvode naredbe pomoću interaktivnog alata za izvođenje SQL naredbi (npr. PgAdmin), redom prema brojevima navedenim ispred svake naredbe: korisnik u sjednici A izvede naredbu {1} do kraja, zatim korisnik u sjednici B izvede naredbu {2} do kraja, itd.

Korisnik kojem je zbog zaključavanja dojavljena pogreška, privremeno obustavlja obavljanje daljnjih naredbi **ali ne prekida transakciju**, a drugi korisnik nastavlja s radom dok ne izvede sve svoje naredbe ili dok se i njemu ne dogodi pogreška zbog nemogućnosti postavljanja ključa. Ako se korisniku oslobodi ključ na kojeg čeka, on nastavlja s obavljanjem svojih naredbi.

Za svaku naredbu kojom mogu biti postavljeni ključevi, napišite koja vrsta ključa se postavlja na koji objekt, a ako se ključ ne postavlja ili ne uspije postaviti obrazložite zbog čega. Ako prilikom izvođenja naredbe dođe do pogreške, navedite kojem će korisniku (u kojoj sjednici) pri izvođenju koje naredbe pogreška biti dojavljena te obrazložite uzrok pogreške.

Relacija *pred* kreirana je naredbom:

```
CREATE TABLE pred(  sifPred SMALLINT PRIMARY KEY
                    ,  nazPred CHAR(20) );
```

U relaciju *pred* upisani su predmeti sa šiframa 1 do 50.

# 1. Upravljanje istodobnim pristupom

	Korisnik A		Korisnik B
{1}	BEGIN TRANSACTION;	{2}	BEGIN TRANSACTION;
{3}	SET TRANSACTION ISOLATION LEVEL READ COMMITTED;	{4}	SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
{5}	SELECT * FROM pred WHERE sifPred = 15 FOR SHARE;	{6}	SELECT * FROM pred WHERE sifPred >= 15 FOR SHARE;
{7}	UPDATE pred SET nazPred= 'Predmet 20' WHERE sifPred = 20;	{8}	UPDATE pred SET nazPred = 'Predmet 16' WHERE sifPred = 16;
{9}	COMMIT TRANSACTION;	{10}	COMMIT TRANSACTION;

# 1. Upravljanje istodobnim pristupom

	Korisnik A		Korisnik B
{1}	BEGIN TRANSACTION;	{2}	BEGIN TRANSACTION;
{3}	SET TRANSACTION ISOLATION LEVEL READ COMMITTED;	{4}	SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
{5}	SELECT * FROM pred WHERE sifPred = 15 FOR SHARE;	{6}	SELECT * FROM pred WHERE sifPred >= 15 FOR SHARE;
{7}	UPDATE pred SET nazPred= 'Predmet 020' WHERE sifPred = 20;	{8}	UPDATE pred SET nazPred = 'Predmet 016' WHERE sifPred = 16;
{9}	COMMIT TRANSACTION;	{10}	COMMIT TRANSACTION;
{5}	<b>postavlja</b> ključ za <b>čitanje</b> (shared lock) nad n-torkom sa šifrom 15. Ključ traje do kraja transakcije.	{6}	<b>postavlja</b> ključ za <b>čitanje</b> na n-torke sa šifrom >= 15. Postavljeni ključevi traju do kraja transakcije (do obavljanja naredbe {10}).
{7}	pokušava postaviti ključ za <b>pisanje</b> nad n-torkom sa šifrom 20, ali <b>ne uspijeva</b> jer je nad tom n-torkom u sjednici B naredbom {6} postavljen ključ za čitanje - transakcija čeka dok transakcija B ne otpusti ključ nad n-torkom sa šifrom 20.	{8}	<b>postavlja</b> ključ za <b>pisanje</b> nad n-torkom sa šifrom 16. Ključ za pisanje traje do kraja transakcije.
		{10}	otpušta sve postavljene ključeve
{7}	<b>postavlja</b> ključ za <b>pisanje</b> nad n-torkom sa šifrom 20. Mijenja naziv predmeta u 'Predmet 020'. Ključ za pisanje traje do kraja transakcije.		
{9}	otpušta sve postavljene ključeve		

## 2. Upravljanje istodobnim pristupom

Izvršavanjem sljedeće SELECT naredbe jezika SQL nad bazom podataka *StreamFlix*

```
SELECT trackId, savedProgress
FROM trackView
WHERE profilename = 'curiouswimp'
AND savedProgress BETWEEN '35 min'::INTERVAL AND '45 min'::INTERVAL
ORDER BY trackId;
```

Dobije se sljedeći rezultat:

trackId	savedprogress
6505	00:45:00
10261	00:39:00
45284	00:44:00

Pretpostavimo da su korisnici **Ivo** i **Ana** trenutno jedini korisnici nad bazom te da su uspješno uspostavili paralelne sjednice. Također pretpostavimo da oboje imaju prava izmjene podataka nad svim tablicama u bazi. **Ivo** i **Ana** izvode naredbe kao u tablici dolje naznačenim redoslijedom.

Napomena: Gore navedena SELECT naredba izvršava se i pod rednim brojevima 7, 8 i 11 ali je radi preglednosti u tablici napisan samo početak i završetak naredbe.

## 2. Upravljanje istodobnim pristupom

	Ivo	Ana
1	BEGIN TRANSACTION;	
2	SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
3		BEGIN TRANSACTION;
4		SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
5	INSERT INTO TRACKVIEW (deviceid, viewstartdatetime, profilename, trackid, savedprogress) VALUES ('A0', CURRENT_TIMESTAMP, 'curiouswimp', 10890, '45 MINUTES'::INTERVAL);	
6		UPDATE trackview SET savedProgress = '41 minutes'::INTERVAL WHERE profilename = 'curiouswimp' AND savedProgress = '39 MINUTES'::INTERVAL;
7	SELECT...ORDER BY trackid;	
8		SELECT...ORDER BY trackid
9	DELETE FROM trackviwe WHERE profilename = 'curiouswimp' AND savedProgress = '39 MINUTES'::INTERVAL;	
10		COMMIT;
11	SELECT...ORDER BY trackid;	
12	COMMIT;	

## 2. Upravljanje istodobnim pristupom

- a) Navedite rezultate SELECT naredbi pod rednim brojevima 7 i 8
- b) Da bi korisnik Ivo mogao izvršiti naredbu pod rednim brojem 9 mora čekati završetak Anine transakcije. Objasnite zbog čega. Pretpostavite da je (bez obzira na to što Ivo čeka) Ana nastavila s izvršavanjem svoje transakcije i obavila naredbu pod rednim brojem 10. Što se sada dogodilo s Ivinom transakcijom? Koliko je n-torki obrisala naredba 9?
- c) Navedite rezultate SELET naredbe rednim brojem 11.

7	6505 „00:45:00” 10261 „00:39:00” 10890 „00:45:00” 45284 „00:44:00”
8	6505 „00:45:00” 10261 „00:41:00” 45284 „00:44:00”
b)	Radi se o tome da pisanje blokira pisanje. UPDATE naredba Anine transakcije je promijenila n-torku koja zadovoljava uvjet selekcije DELETE naredbe Ivine transakcije. Ivina transakcija mora čekati dok ishod Anine transakcije ne bude poznat. Nakon što Anina transakcija završi obavlja se naredba pod 9. Naće obrisati niti jednu ntorku jer sada ova s <code>trackid = 10261</code> ne zadovoljava uvjet selekcije.
11	6505 „00:45:00” 10890 „00:45:00” 45284 „00:44:00” 10261 „00:41:00”



### 3. Optimiranje upita

mjerenje

SifMjerenje	datum	OznPostaja	sifMjElem	rbrMjerenje	izmjerenaVrij
1003456	20.05.2015	ZG-M	1	28	12.4
1003457	20.05.2015	ZG-M	2	10	1017.0
5604575	20.05.2015	ZG-M	5	10	6.3
5643216	20.05.2015	ST-M	1	25	17.2
6543808	21.05.2015	HV	3	2	17.6

meteoPostaja

oznPostaja	nazPostaja	NadmVisina
ZG-M	Zagreb-Maksimir	123
ZG-P	Zagreb-Puntijarka	988
ST-M	Split-Marjan	122
ZAV	Zavižan	1594
HV	Hvar	0

mjerniElement

sifMjElem	nazMjElem	sifJedMjere
1	Temperatura zraka	1
2	Tlak zraka	2
3	Temperatura mora	1
4	Količina oborine	3
5	Brzina vjetra	4

jedMjere

sifJedMjere	nazJedMjere	kratJedMjere
1	Celzijev stupanj	°C
2	Hektopaskal	hPa
3	Milimetar	mm
4	Metar u sekundi	m/s
5	Farenhaitov stupanj	°F

Dolje prikazanim naredbama kreirane su relacije **meteoPostaja**, **mjerniElement**, **jedMjere** i **mjerenje**.

```
CREATE TABLE jedMjere (
    sifJedMjere SMALLINT PRIMARY KEY
, nazJedMjere CHAR(50) NOT NULL
, kratJedMjere CHAR(50) NOT NULL
);
```

```
CREATE TABLE mjerniElement (
    sifMjElem SMALLINT PRIMARY KEY
, nazMjElem CHAR(50) NOT NULL
, sifJedMjere SMALLINT NOT NULL
REFERENCES jedMjere(sifJedMjere));
```

```
CREATE TABLE meteoPostaja (
    oznPostaja CHAR(5) PRIMARY KEY
, nazPostaja CHAR(250) NOT NULL
, nadmVisina SMALLINT NOT NULL
);
```

```
CREATE TABLE mjerenje (
    sifMjerenje INTEGER PRIMARY KEY
, datum DATE
, oznPostaja CHAR(5) NOT NULL
REFERENCES meteoPostaja(oznPostaja)
, sifMjElem SMALLINT NOT NULL
REFERENCES mjerniElement(sifMjElem)
, rbrMjerenje SMALLINT NOT NULL
, izmjerenaVrij DECIMAL NOT NULL
, UNIQUE (datum, rbrMjerenje, oznPostaja, sifMjElem));
```

### 3. Optimiranje upita

mjerjenje						meteoPostaja		
SifMjerenje	datum	Ozn Postaja	sifMj Elem	rbr Mjerenje	izmjerena Vrij	ozn Postaja	nazPostaja	nadm Visina
1003456	20.05.2015	ZG-M	1	28	12.4	ZG_M	Zagreb-Maksimir	123°C

mjerniElement			jedMjere		
sifMjElem	nazMjElem	sifJedMjere	sifJedMjere	nazJedMjere	kratJedMjere
1	Temperatura zraka	1	1	Celzijev stupanj	°C

Optimizatoru su raspoloživi sljedeći podaci: 

N(jedmjere)	= 20
N(mjerniElement)	= 200
N(meteoPostaja)	= 500
N(mjerenje)	= 6.000.000

V(izmjerenaVrij, mjerjenje)	= 2.000.000
V(rbrMjerenje, mjerjenje)	= 500
V(datum, mjerjenje)	= 100
V(nadmVisina, meteoPostaja)	= 450
V(sifJedMjere, mjerniElement)	= 20

Za donju SELECT naredbu potrebno je nacrtati plan izvođenja nakon provođenja heurističke optimizacije. Redoslijed spajanja odrediti na temelju procjene ukupnog broja n-torki u svim međurezultatima. U planu naznačiti očekivani broj zapisa za međurezultate, te korištene metode pristupa. Navesti sve izraze prema kojima je obavljena procjena broja n-torki u međurezultatima.

```

SELECT *
FROM mjerjenje, mjerniElement, jedMjere
WHERE mjerjenje.sifMjElem = mjerniElement.sifMjElem
      AND mjerniElement.sifJedMjere= jedMjere.sifJedMjere
      AND rbrMjerenje = 1
      AND datum < '01.01.2016'
      AND jedMjere.sifJedMjere = 1
    
```

### 3. Optimiranje upita – početni plan

```
SELECT *
```

```
FROM mjerjenje, mjerniElement, jedMjere
WHERE mjerjenje.sifMjElem = mjerniElement.sifMjElem
      AND mjerniElement.sifJedMjere= jedMjere.sifJedMjere
      AND rbrMjerjenje = 1
      AND datum < '01.01.2016'
      AND jedMjere.sifJedMjere = 1
```

mjerjenje

SifMjerenje	datum	Ozn Postaja	sifMj Elem	rbr Mjerjenje	izmjerena Vrij
1003456	20.05.2015	ZG-M	1	28	12.4

meteoPostaja

ozn Postaja	nazPostaja	nadm Visina
ZG_M	Zagreb-Maksimir	123°C

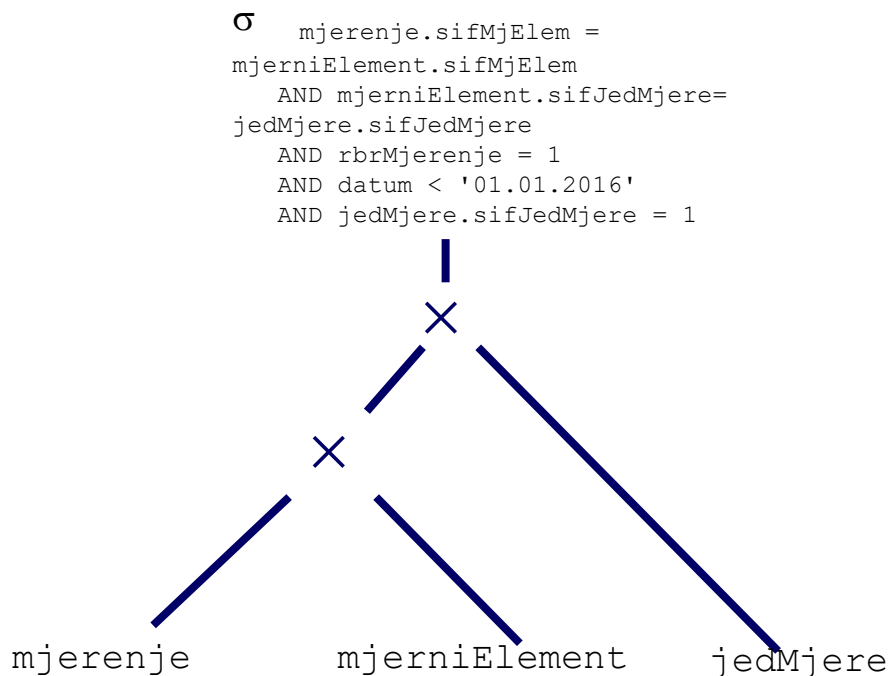
mjerniElement

sifMjElem	nazMjElem	sifJedMjere
1	Temperatura zraka	1

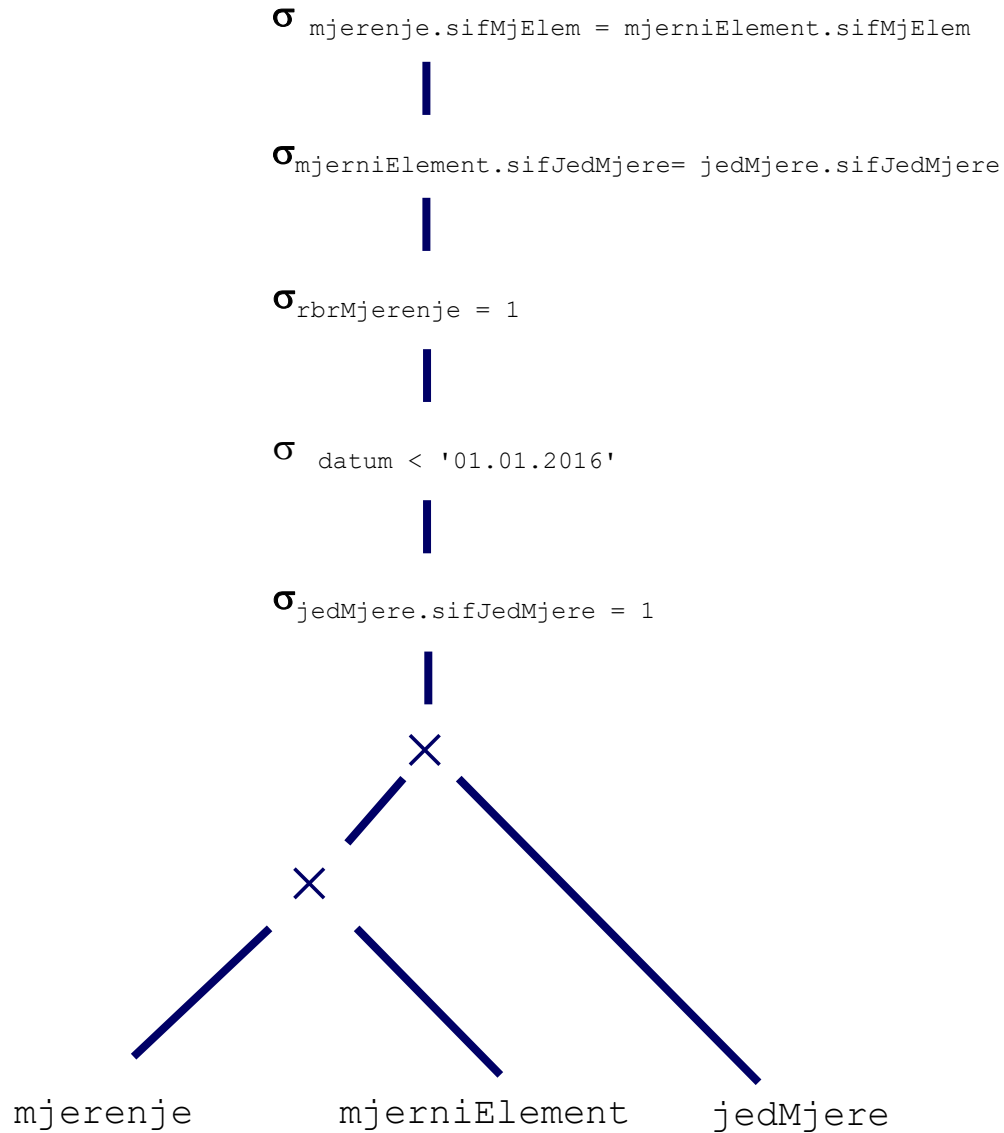
jedMjere

sifJedMjere	nazJedMjere	kratJedMjere
1	Celzijev stupanj	°C

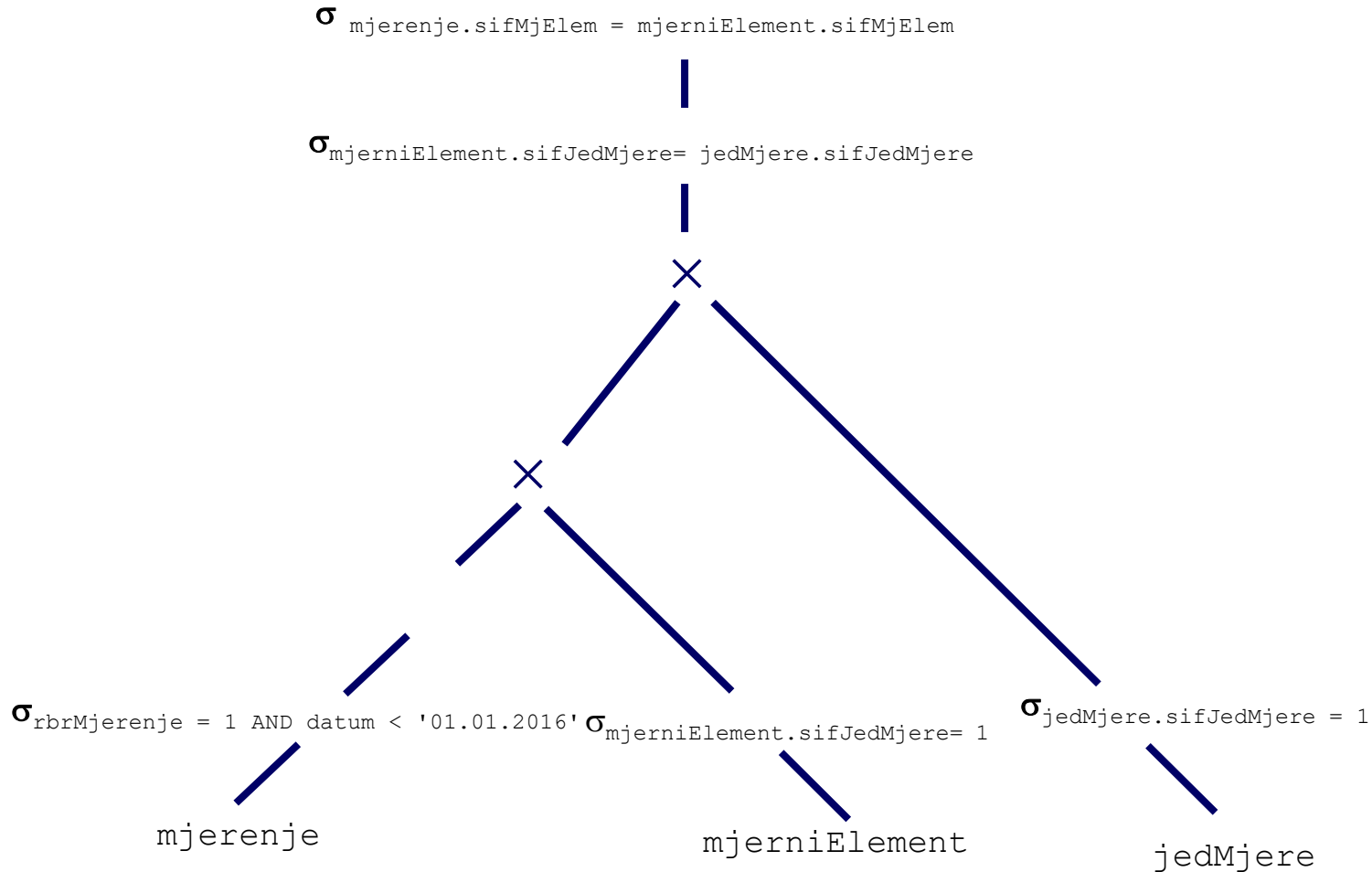
Redoslijed spajanja relacija određen redoslijedom kojim su navedene u FROM dijelu:



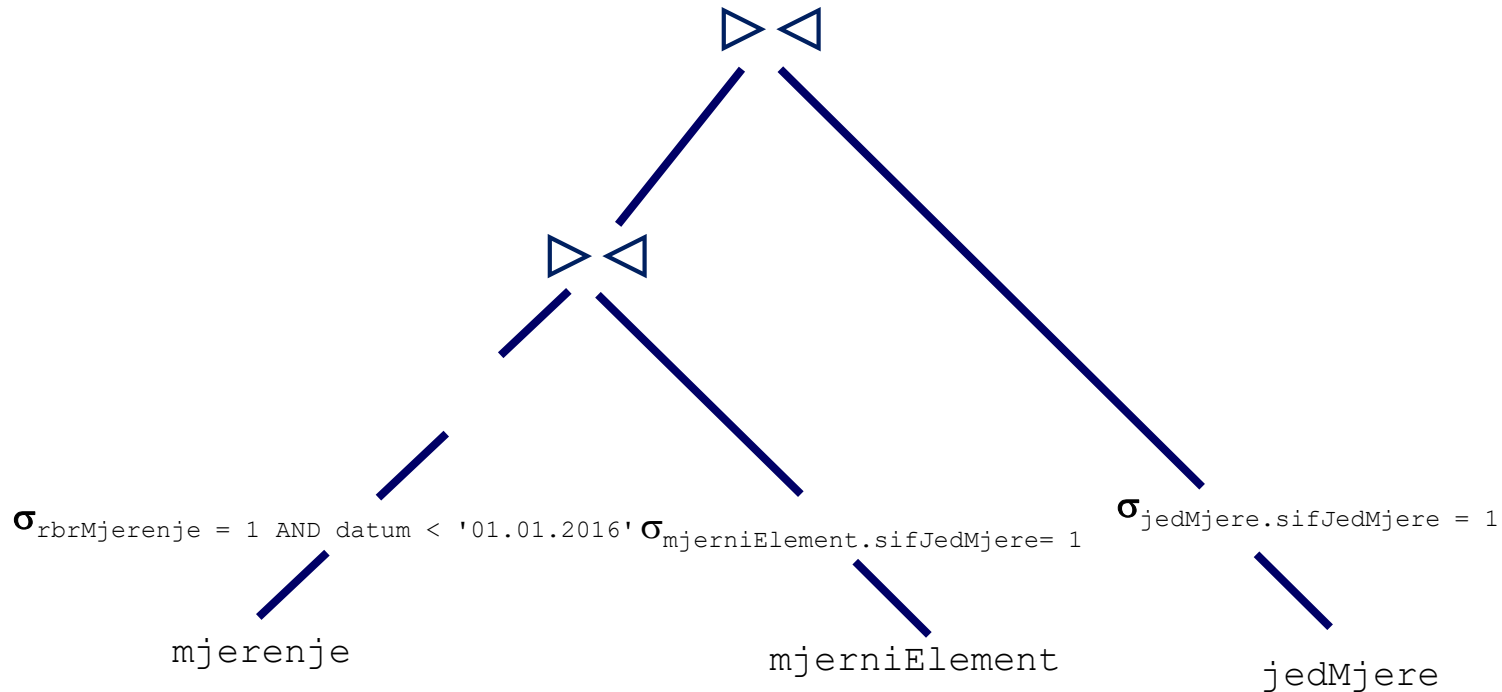
### 3. Optimiranje upita - rastavljanje uvjeta selekcije



### 3. Optimiranje upita - potiskivanje uvjeta selekcije



### 3. Optimiranje upita - kombiniranje operacije selekcije i Kartezijevog produkta



### 3. Optimiranje upita - procjena broja n-torki

```
SELECT *
FROM mjerjenje, mjerniElement, jedMjere
WHERE mjerjenje.sifMjElem = mjerniElement.sifMjElem
      AND mjerniElement.sifJedMjere = jedMjere.sifJedMjere
      AND rbrMjerjenje = 1
      AND datum < '01.01.2016'
      AND jedMjere.sifJedMjere = 1
```

$$\begin{aligned} \text{mjerjenje}_1 &= \sigma_{\text{rbrMjerjenje}=1 \text{ AND datum} < '1.1.2016'}(\text{mjerjenje}) \\ &= N(\text{mjerjenje}_1) / (V(\text{rbrMjerjenje}, \text{mjerjenje}) * 3) \\ &= 6\,000\,000 / (500 * 3) = 4\,000 \end{aligned}$$

$$\begin{aligned} \text{mjerniElement}_1 &= \sigma_{\text{sifJedMjere}=1}(\text{mjerniElement}) \\ &= N(\text{mjerniElement}) / V(\text{jedMjere}, \text{mjerniElement}) \\ &= 200 / 20 = 10 \end{aligned}$$

$$\begin{aligned} \text{jedMjere}_1 &= \sigma_{\text{sifJedMjere}=1}(\text{jedMjere}) \\ &= 1 \end{aligned}$$

Procjena broja n-torki u međurezultatu za različite redoslijede spajanja

$$N(\text{mjerjenje}_1 \bowtie \text{mjerniElement}_1) \leq N(\text{mjerjenje}_1) \leq 4\,000 \quad (\text{mjerjenje}_1 \cap \text{mjerniElement}_1 \text{ je ključ u mjerniElement})$$

$$N(\text{mjerniElement}_1 \bowtie \text{jedMjere}_1) \leq N(\text{mjerniElement}_1) \leq 10 \quad (\text{mjerniElement}_1 \cap \text{jedMjere}_1 \text{ je ključ u jedMjere})$$

$$N(\text{mjerjenje}_1 \bowtie \text{jedMjere}_1) = N(\text{mjerjenje}_1) * N(\text{jedMjere}_1) = 4\,000$$

Redoslijed spajanja:  $\Rightarrow (\text{mjerniElement}_1 \bowtie \text{jedMjere}_1) \bowtie \text{mjerjenje}_1$

N(jedmjere)	= 20
N(mjerniElement)	= 200
N(meteoPostaja)	= 500
N(mjerjenje)	= 6 000 000

V(izmjereniVrij, mjerjenje)	= 2000 000
V(rbrMjerjenje, mjerjenje)	= 500
V(datum, mjerjenje)	= 100
V(nadmVisina, meteoPostaja)	= 450
V(sifJedMjere, mjerniElement)	= 20

```

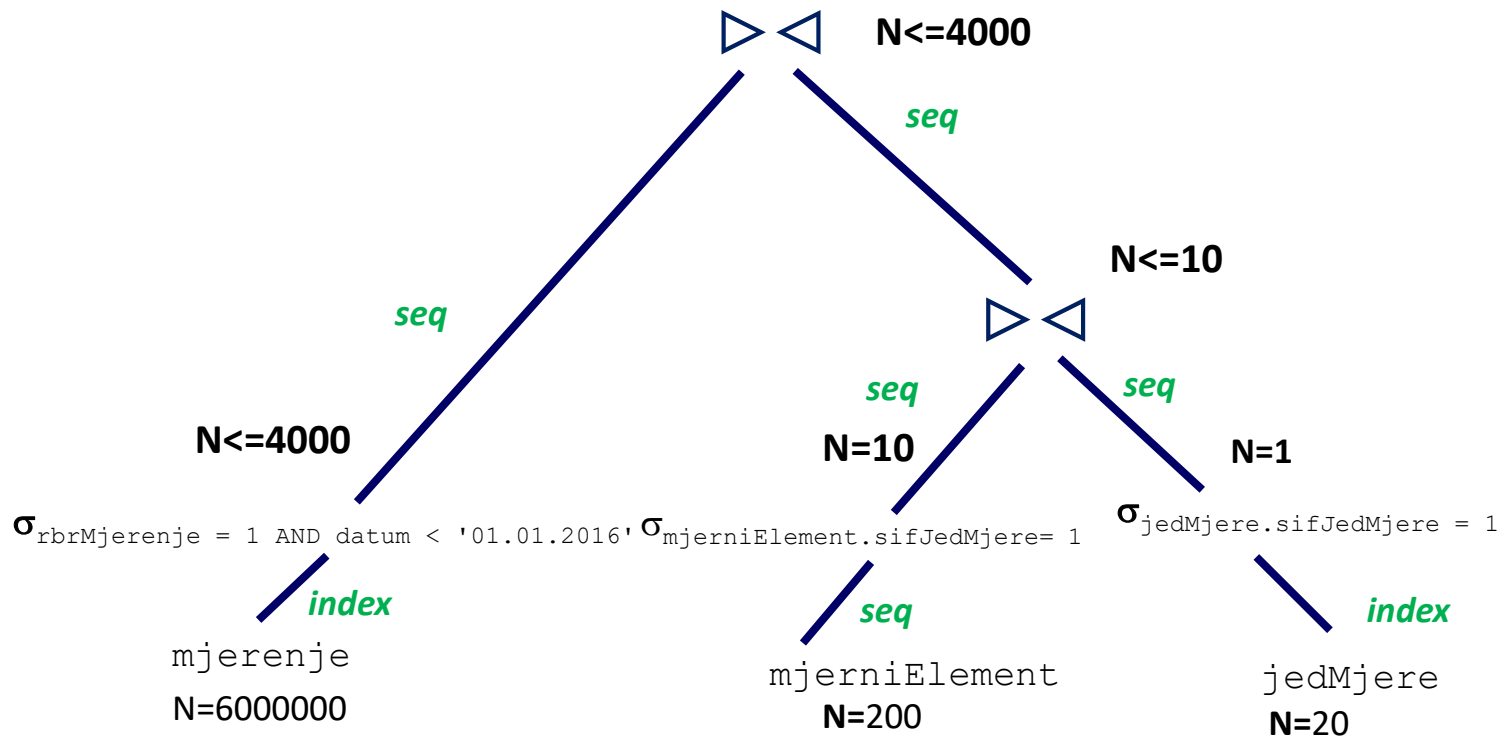
CREATE TABLE jedMjere (
  sifJedMjere  SMALLINT PRIMARY KEY
...
CREATE TABLE mjerniElement (
  sifMjElem    SMALLINT PRIMARY KEY
...
, sifJedMjere  SMALLINT NOT NULL
                REFERENCES
jedMjere(sifJedMjere));

```

```

CREATE TABLE mjerenje (
  sifMjerenje  INTEGER PRIMARY KEY
, oznPostaja   CHAR(5) NOT NULL
                REFERENCES
meteoPostaja(oznPostaja)
, sifMjElem    SMALLINT NOT NULL
                REFERENCES
mjerniElement(sifMjElem)
...
, UNIQUE (datum, rbrMjerenje, oznPostaja,
sifMjElem));

```





## 4. ER model

U zdravstvenoj ustanovi evidentiraju se podaci o osobama koje mogu biti pacijenti ili liječnici (ili oboje) i obavljenim pregledima.

Za svaku osobu se bilježi šifra, ime i prezime, datum i mjesto rođenja, te ulica i kućni broj stanovanja. Za pacijente se dodatno bilježi kategorija osiguranja, dok se za liječnike bilježi datum početka obavljanja prakse.

Za ulice se bilježi šifra i naziv te mjesto u kojem se ulica nalazi, a za mjesto šifra, poštanski broj i naziv. Ulica pripada samo jednom mjestu; ulice jednakog naziva iz različitih mjesta imaju različite šifre.

Za svaki obavljeni pregled evidentira se nad kojim je pacijentom pregled obavljen, koji je liječnik obavio pregled, datum pregleda i trajanje pregleda (broj minuta). Jedan pregled obavlja samo jedan liječnik. Za svaki pregled evidentira se redni broj pregleda, pri čemu svakog dana za svakog liječnika redni brojevi pregleda započinju ponovo s brojem jedan. Za isti datum za istog liječnika ne postoje dva pregleda s istim rednim brojem.

Na jednom pregledu liječnik može pacijentu postaviti više dijagnoza te ga, ovisno o dijagnozi, uputiti na terapije. Za jednu postavljenu dijagnozu na pregledu pacijent se može uputiti na jednu ili više terapija. Na istu terapiju pacijent može biti upućen zbog više dijagnoza postavljenih istim pregledom.

Na različitim pregledima liječnik istom pacijentu za istu dijagnozu može prepisati drugačije terapije.

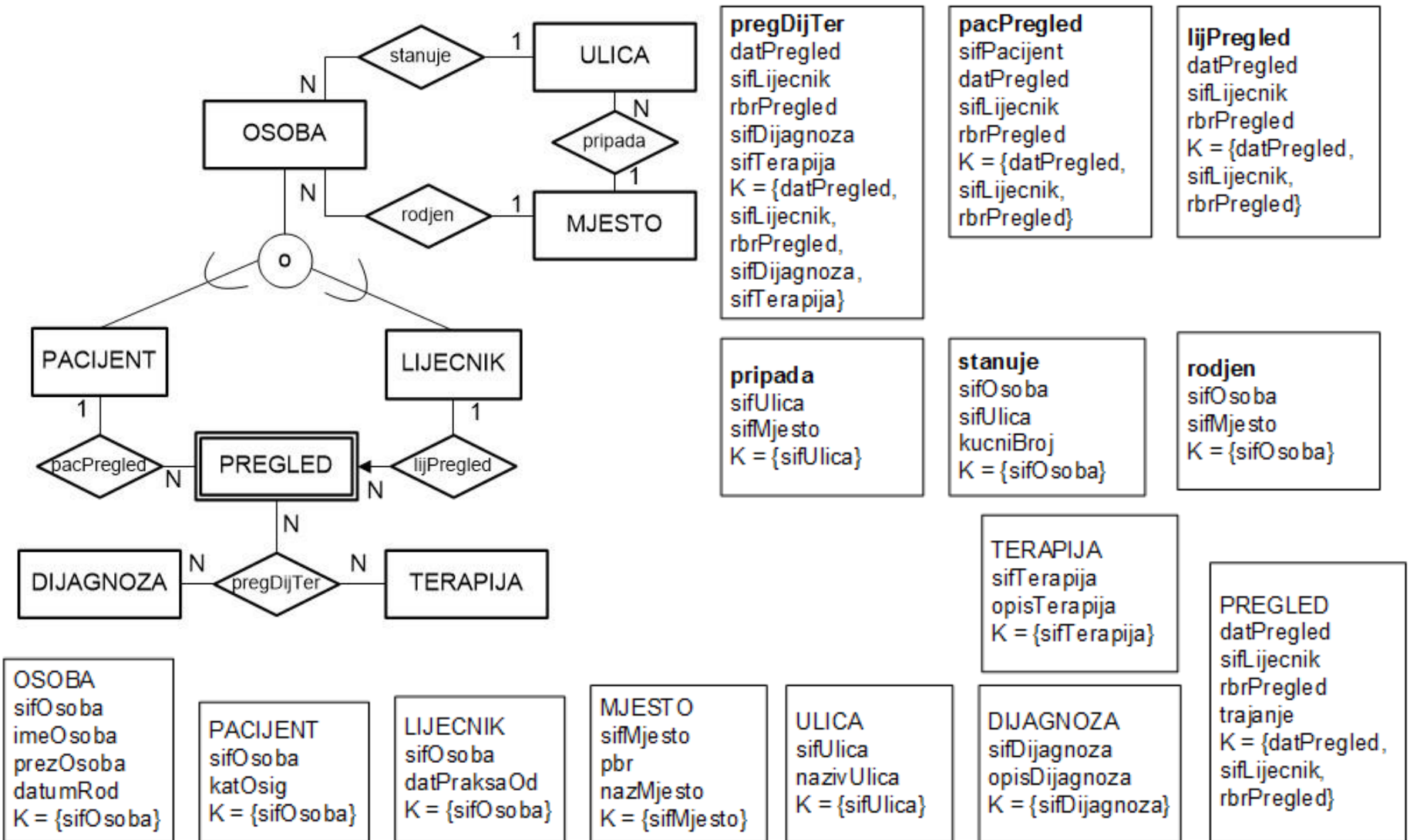
Za dijagnoze se evidentira šifra i naziv.

Za terapiju se evidentira šifra i opis terapije (npr. 1 - Sonoterapija desnog koljena primjenom ultrazvuka, 2- Elektroterapija lijevog ramena jednosmjernom strujom različite frekvencije...).

## 4. ER model

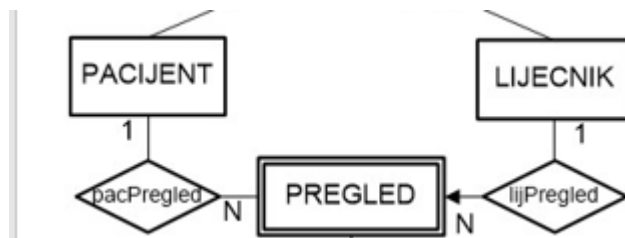
- Nacrtati ER model i opisati entitete i veze (njihove attribute i ključeve). Entitete (osim slabih entiteta) opisati isključivo vlastitim atributima. Sve sheme moraju zadovoljavati 3NF.
- Za segment ER modela koji obuhvaća entitete *pregled*, *pacijent* i *liječnik*, te veze koje postoje među tim entitetima, napisati ekvivalentni relacijski model u obliku SQL naredbi za kreiranje relacija. Naredbe moraju sadržavati definicije integritetskih ograničenja.

## 4. ER model



## 4. ER model – komentar

- Dvije binarne veze prikazano slikom nije moguće pretvoriti u jednu ternarnu vezu između PACIJENT, PREGLED i LIJEČNIK.



- Pregled* je slabi entitet koji je vezan uz jednog liječnika (ključ pregleda sadrži ključ liječnika)
- Pregled je vezan uz jednog pacijenta.

## 4. ER model

- Za segment ER modela koji obuhvaća entitete *pregled*, *pacijent* i *liječnik*, te veze koje postoje među tim entitetima, napisati ekvivalentni relacijski model u obliku SQL naredbi za kreiranje relacija. Naredbe moraju sadržavati definicije integritetskih ograničenja.

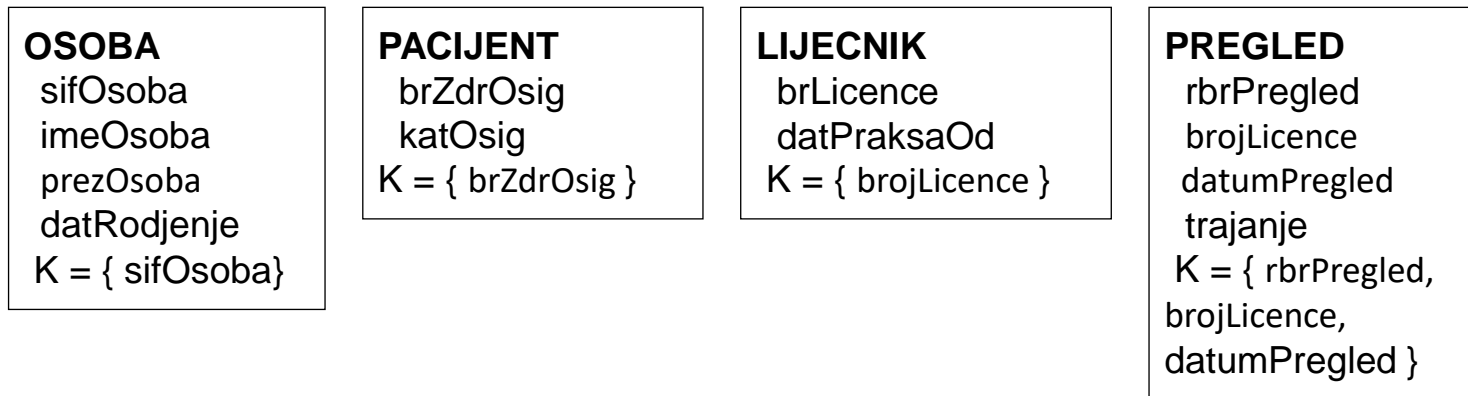
```
CREATE TABLE osoba (...)  
CREATE TABLE pacijent  
  (sifOsoba INTEGER PRIMARY KEY  
   katOsiguranja CHAR(2) NOT NULL,  
   FOREIGN KEY (sifOsoba ) REFERENCES osoba(sifOsoba)  
  );  
CREATE TABLE lijecnik  
  (sifOsoba INTEGER PRIMARY KEY  
   datPraksaOd DATE      NOT NULL,  
   FOREIGN KEY (sifOsoba) REFERENCES osoba(sifOsoba)  
  );  
  
CREATE TABLE pregled  
  (datumPregled DATE  
   , sifLijecnik  INTEGER REFERENCES lijecnik (sifOsoba )  
   , rbrPregled   SMALLINT  
   , trajanje     SMALLINT  
   , sifPacijent  INTEGER REFERENCES pacijent (sifOsoba)  
   , PRIMARY KEY (datumPregled, sifLijecnik, rbrPregled));
```

## 4. ER model (verzija 2)

U zdravstvenoj ustanovi evidentiraju se podaci o osobama koje mogu biti pacijenti ili liječnici (ili oboje) i obavljenim pregledima.

Za svaku osobu se bilježi šifra, ime i prezime, datum i mjesto rođenja te ulica i kućni broj stanovanja.

Za pacijente se dodatno bilježi **broj zdravstvenog osiguranja (identifikator pacijenta)** i kategorija osiguranja, dok se za liječnike bilježi **jedinstveni broj liječničke licence (identifikator liječnika)** i datum početka obavljanja prakse.



Relacijski model:

OSOBA = sifOsoba, imeOsoba, prezOsoba, datRodjenje    PK = {sifOsoba}

PACIJENT = brZdrOsig, katOsog, sifOsoba    PK = {brZdrOsig}    K<sub>A</sub> = {sifOsoba}

LIJEČNIK = brLicence, datPraksaOd, sifOsoba    PK = {brLicence}    K<sub>A</sub> = {sifOsoba}

## 5. Sigurnost BP

U relaciji **ovlasti** (baza podataka **meteoMjerenja**) evidentirane su meteorološke postaje s čijim podacima određeni korisnik može raditi. Vrijednost atributa *login* jednaka je korisničkom imenu (CURRENT\_USER) s kojim korisnik uspostavlja SQL sjednicu. Osim vlasnika baze podataka, nijedan korisnik (uključujući PUBLIC) nema nikakve ovlasti. Napisati niz SQL naredbi:

- a) kojima će se stvoriti uloga (role) *meteorolog* te korisnicima kojima se dodijeli uloga *meteorolog* omogućiti:
- pregledavanje podataka o svim zapisima relacija **meteoPostaja** i **mjerenje**
  - unos, izmjena i brisanje samo onih podataka iz relacije **meteoPostaja** za koje su im dodijeljene ovlasti (tablica **ovlasti**).
  - kojima će administrator sustava stvoriti korisnika *vakula* s proizvoljnom lozinkom i ovlastima za uspostavu korisničke sjednice, spajanje na bazu podataka **meteoMjerenja** i korištenje uloge *meteorolog*.
- b) koje korisnik *vakula*, nakon što uspostavi korisničku sjednicu, treba obaviti kako bi mogao koristiti dozvole dodijeljene ulozi *meteorolog*.

mjerenje

<u>SifMjerenje</u>	<u>datum</u>	<u>Ozn Postaja</u>	<u>sifMj Elem</u>	<u>rbr Mjerenje</u>	<u>izmjerena Vrij</u>
1003456	20.05.2015	ZG-M	1	28	12.4

meteoPostaja

<u>ozn Postaja</u>	<u>nazPostaja</u>	<u>nadm Visina</u>
ZG_M	Zagreb-Maksimir	123°C

ovlasti

<u>login</u>	<u>oznPost</u>
vakula	ZG-M
cacic	ZG-M
sijerkovic	ZG-P
...	...

mjerniElement

<u>sifMjElem</u>	<u>nazMjElem</u>	<u>sifJedMjere</u>
1	Temperatura zraka	1

jedMjere

<u>sifJedMjere</u>	<u>nazJedMjere</u>	<u>kratJedMjere</u>
1	Celzijev stupanj	°C

## 5. Naredbe za kreiranje potrebnih relacija

```
CREATE database meteoMjerenja ;
REVOKE CONNECT ON DATABASE meteoMjerenja FROM PUBLIC;
REVOKE ALL      ON SCHEMA public FROM PUBLIC;
REVOKE ALL ON ALL TABLES IN SCHEMA public FROM PUBLIC;
CREATE TABLE meteoPostaja (
    oznPostaja      CHAR(5) PRIMARY KEY
, nazPostaja       CHAR(250) NOT NULL
, nadmVisina       SMALLINT  NOT NULL
);
CREATE TABLE jedMjere (
    sifJedMjere      SMALLINT PRIMARY KEY
, nazJedMjere       CHAR(50)  NOT NULL
, kratJedMjere      CHAR(50)  NOT NULL
);
CREATE TABLE mjerniElement (
    sifMjElem        SMALLINT PRIMARY KEY
, nazMjElem         CHAR(50)  NOT NULL
, sifJedMjere       SMALLINT  NOT NULL REFERENCES jedMjere(sifJedMjere)
);
CREATE TABLE mjerenje (
    sifMjerenje      INTEGER PRIMARY KEY
, datum            DATE
, oznPostaja        CHAR(5)  REFERENCES meteoPostaja(oznPostaja)
, sifMjElem         SMALLINT REFERENCES mjerniElement(sifMjElem)
, rbrMjerenje       SMALLINT CHECK (rbrMjerenje between 1 AND 3000)
, izmjerenavaVrij   DECIMAL  NOT NULL
, UNIQUE (datum, rbrMjerenje, oznPostaja, sifMjElem)
);
CREATE TABLE ovlasti(
    login           CHAR(10)
, oznPost          CHAR(5) REFERENCES meteoPostaja (oznPostaja)
, PRIMARY KEY (login, oznPost));
```



## 5. ...i punjenje relacija sadržajem1

```
INSERT INTO meteoPostaja VALUES ('ZG-M', 'Zagreb - Maksimir', 123);
INSERT INTO meteoPostaja VALUES ('ZG-P', 'Zagreb - Puntijarka', 988);
INSERT INTO meteoPostaja VALUES ('ST-M', 'Split - Marjan', 122);
INSERT INTO meteoPostaja VALUES ('ZAV', 'Zavižan', 1594);
INSERT INTO meteoPostaja VALUES ('HV', 'Hvar', 0);

INSERT INTO jedMjere VALUES (1, 'Celzijev stupanj', '°C');
INSERT INTO jedMjere VALUES (2, 'Hektopaskal', 'hPa');
INSERT INTO jedMjere VALUES (3, 'Milimetar', 'mm');
INSERT INTO jedMjere VALUES (4, 'Metar u sekundi', 'm/s');

INSERT INTO mjerniElement VALUES (1, 'Temperatura zraka', 1);
INSERT INTO mjerniElement VALUES (2, 'Tlak zraka', 2);
INSERT INTO mjerniElement VALUES (3, 'Temperatura mora', 1);
INSERT INTO mjerniElement VALUES (4, 'Količina oborine', 3);
INSERT INTO mjerniElement VALUES (5, 'Brzina vjetra', 4);

INSERT INTO mjerenje VALUES (1003456, '20.05.2015', 'ZG-M', 1, 28, 12.4);
INSERT INTO mjerenje VALUES (1003457, '20.05.2015', 'ZG-M', 2, 10, 1017.0); INSERT INTO
mjerenje VALUES (5604575, '20.05.2015', 'ZG-M', 5, 10, 6.3);
INSERT INTO mjerenje VALUES (5643216, '20.05.2015', 'ST-M', 1, 28, 17.2);
INSERT INTO mjerenje VALUES (6543808, '21.05.2015', 'HV' , 3, 2, 17.6);

INSERT INTO ovlasti VALUES ('vakula', 'ZG-M');
INSERT INTO ovlasti VALUES ('cacic', 'ZG-M');
INSERT INTO ovlasti VALUES ('sijerkovic', 'ZG-P');
```

## 5. Sigurnost BP

mjerjenje					
<u>SifMjerenje</u>	<u>datum</u>	<u>Ozn Postaja</u>	<u>sifMj Elem</u>	<u>rbr Mjerenje</u>	<u>izmjerena Vrij</u>
1003456	20.05.2015	ZG-M	1	28	12.4

meteoPostaja		
<u>ozn Postaja</u>	<u>nazPostaja</u>	<u>nadm Visina</u>
ZG_M	Zagreb-Maksimir	123°C

mjerniElement		
<u>sifMjElem</u>	<u>nazMjElem</u>	<u>sifJedMjere</u>
1	Temperatura zraka	1

jedMjere		
<u>sifJedMjere</u>	<u>nazJedMjere</u>	<u>kratJedMjere</u>
1	Celzijev stupanj	°C

ovlasti	
<u>login</u>	<u>oznPost</u>
vakula	ZG-M
cacic	ZG-M
sijerkovic	ZG-P
...	...

a) CREATE VIEW vMeteoPostaja AS

```
SELECT *
```

```
FROM   MeteoPostaja
```

```
WHERE  oznPostaja IN (SELECT oznPost
```

```
FROM   ovlasti
```

```
WHERE  login = CURRENT_USER)
```

```
WITH CHECK OPTION;
```

```
CREATE ROLE meteorolog;
```

```
--GRANT CONNECT ON DATABASE meteoMjerenja TO meteorolog; ne treba jer  
nikada neće uspostavljati korisničku sjednicu niti doći u priliku da se  
spaja na bazu podataka
```

```
GRANT USAGE ON SCHEMA public TO meteorolog;
```

```
GRANT SELECT ON meteoPostaja TO meteorolog;
```

```
GRANT SELECT ON mjerjenje TO meteorolog;
```

```
GRANT INSERT, UPDATE, DELETE ON vMeteoPostaja TO meteorolog;
```

## 5. Sigurnost BP

b) `CREATE USER vakula WITH PASSWORD '...je reko';`  
`GRANT CONNECT ON DATABASE meteoMjerenja TO vakula;`  
`GRANT meteorolog TO vakula;`  
`--GRANT USAGE ON SCHEMA public TO vakula; ne treba mu jer je zbog defaultnog INHERIT naslijedio od meteorolog`

c) Ništa zbog defaultnog INHERIT (!)  
Da je LOGIN vakula napravljen sljedećom naredbom:

```
CREATE USER vakula WITH NOINHERIT PASSWORD '...je reko';
```

onda bi virtualna relacija morala biti napravljena sa `SESSION_USER`, te bi *vakula* nakon uspostave korisničke sjednice morao obaviti naredbu:  
`SET ROLE meteorolog;`

## 6. Okidači i procedure

clanak

<i>sif</i> <i>Clanak</i>	<i>naslovClanak</i>	<i>sif</i> <i>Casopis</i>	<i>datum</i> <i>Objava</i>
1	Quantum physics: How to catch a wave	1	08.06.2009
2	How the Fruit Fly Got His Spots	2	27.04.2010
3	Newtonian gravity and the Bargmann algebra	3	11.04.2011

casopis

<i>sif</i> <i>Casopis</i>	<i>naz</i> <i>Casopis</i>	<i>brzda</i> <i>God</i>
1	Nature	12
2	Science	12
3	Astrophysical Journal	6

- Napišite niz SQL naredbi kojima će se u relaciji **clanak** zabraniti izmjena šifre časopisa u kojem je članak objavljen. Pri pokušaju izmjene časopisa u kojem je članak objavljen potrebno je dojaviti poruku

Promjena časopisa nije dozvoljena!

- Napišite jednu naredbu čije će izvođenje pokušati narušiti pravilo o zabrani izmjene šifre časopisa u relaciji **clanak**.

## 6. Okidači i procedure

Naredbe za kreiranje i punjenje relacija **casopis** i **clanak**:

```
CREATE TABLE casopis (  
    sifCasopis    SMALLINT CONSTRAINT pkCasopis PRIMARY KEY,  
    nazCasopis    VARCHAR (150),  
    brIzdGod      SMALLINT  
);
```

```
CREATE TABLE clanak (  
    sifClanak     INTEGER CONSTRAINT pkClanak PRIMARY KEY ,  
    naslovClanak  VARCHAR (150),  
    sifCasopis    SMALLINT CONSTRAINT fkClanakCasopis REFERENCES casopis(sifCasopis) ,  
    datumObjava   DATE);
```

```
INSERT INTO casopis VALUES (1, 'Nature', 12);
```

```
INSERT INTO casopis VALUES (2, 'Science', 12);
```

```
INSERT INTO casopis VALUES (3, 'Astrophisic Journal', 6);
```

```
INSERT INTO clanak VALUES (1, 'Quantum physics: How to catch a wave', 1, '08.06.2009');
```

```
INSERT INTO clanak VALUES (2, 'How the Fruit Fly Got His Spots', 2, '27.04.2010');
```

```
INSERT INTO clanak VALUES (3, 'Newtonian gravity and the Bargmann algebra', 3, '11.04.2011');
```

## 6. Okidači i procedure

```
CREATE FUNCTION dojavipogresku () RETURNS trigger AS
$$
BEGIN
    RAISE EXCEPTION 'Promjena časopisa nije dozvoljena!';
END
$$ LANGUAGE plpgsql;

CREATE TRIGGER clanak_upd
BEFORE UPDATE OF sifCasopis ON clanak
FOR EACH ROW
WHEN (NEW.sifcasopis != OLD.sifCasopis)
EXECUTE FUNCTION dojavipogresku();
```

Naredba kojom se može testirati rješenje:

```
UPDATE clanak
SET sifcasopis = sifcasopis + 1
WHERE sifCasopis < 4;
```

Data Output Explain Messages Query History

```
ERROR: Promjena časopisa nije dozvoljena!
CONTEXT: PL/pgSQL function dojavipogresku() line 3 at RAISE
SQL state: P0001
```

## 7. Okidači i procedure

razinaSt
----------

<i>sifRazinaSt</i>	<i>nazRazinaSt</i>	<i>maksDozvTrajanje</i>
1	Preddiplomska	6
2	Diplomska	4

studij
--------

<i>sifStudij</i>	<i>nazStudij</i>	<i>sifRazinaSt</i>	<i>trajanje</i>
1	Računarstvo	1	6
2	Elektrotehnika	1	5

- Studij ne smije trajati dulje od maksimalnog dozvoljenog trajanja propisanog za razinu studija kojoj pripada. Trebate spriječiti narušavanje navedenog pravila. Pri tom prijaviti poruku:

Nedozvoljeno trajanje studija!

**Napomena:** Podrazumijevati da se vrijednost atributa *maksDozvTrajanje* u relaciji *razinaStudij* ne mijenja.

## 7. Okidači i procedure

Naredbe za kreiranje i punjenje relacija **razinaSt** i **studij**:

```
CREATE TABLE razinaSt (  
    sifRazinaSt      SMALLINT CONSTRAINT pkRazinaSt PRIMARY KEY,  
    nazRazinaSt      VARCHAR(50) NOT NULL,  
    maksDozvTrajanje SMALLINT NOT NULL);  
  
CREATE TABLE studij (  
    sifStudij        INTEGER CONSTRAINT pkStudij PRIMARY KEY,  
    nazStudij        VARCHAR(255) NOT NULL,  
    sifRazinaSt      SMALLINT NOT NULL REFERENCES razinaSt(sifRazinaSt),  
    trajanje         SMALLINT NOT NULL CHECK (trajanje >0));  
  
INSERT INTO razinaSt VALUES (1, 'Preddiplomska', 6);  
INSERT INTO razinaSt VALUES (2, 'Diplomska', 4);  
  
INSERT INTO studij VALUES (1, 'Računarstvo', 1, 6);  
INSERT INTO studij VALUES (2, 'Elektrotehnika', 1, 5);
```



## 7. Okidači i procedure

### Analiza:

- Pravilo „*Studij ne smije trajati dulje od maksimalnog dozvoljenog trajanja propisanog za razinu studija kojoj pripada*” mogle bi narušiti sljedeće operacije:

1. unos n-torke u `studij`

2. izmjena atributa `studij.sifRazinaSt` i/ili `studij.trajanje`

→ Potrebno je implementirati okidače koji će se aktivirati pri obavljanju naredbi

`INSERT` i `UPDATE` u relaciju `studij`.

```
CREATE TRIGGER insStudij
  AFTER INSERT ON studij FOR EACH ROW
  EXECUTE FUNCTION chktrajanjeStudij ();
```

```
CREATE TRIGGER updStudij
  BEFORE UPDATE OF sifRazinaSt, trajanje ON studij FOR EACH ROW
  WHEN (NEW.sifRazinaSt!= OLD.sifRazinaSt OR
        NEW.trajanje    != OLD.trajanje)
  EXECUTE FUNCTION chktrajanjeStudij();
```

## 7. Okidači i procedure

- Je li pravilo narušeno provjeravat ćemo pomoću funkcije `chkTrajanjeStudij` koju ćemo pozivati iz oba okidača.
- Ako studij traje dulje od predviđenog tj.

```
razinaSt.maksDozvTrajanje < studij.trajanje
```

...procedura će zbog narušavanja opisanog pravila dojaviti grešku pomoću iznimke:

```
RAISE EXCEPTION 'Nedozvoljeno trajanje studija!';
```

```
CREATE FUNCTION chktrajanjeStudij() RETURNS TRIGGER AS $$  
BEGIN  
    IF (SELECT maksDozvTrajanje FROM razinaSt  
        WHERE razinaSt.sifrazinaSt = NEW.sifRazinaSt) < NEW.trajanje  
    THEN  
        RAISE EXCEPTION 'Nedozvoljeno trajanje studija!';  
        RETURN NULL;  
    END IF;  
    RETURN NEW;  
END  
$$ LANGUAGE plpgsql;
```

## 7. Okidači i procedure

razinaSt		
sifRazinaSt	nazRazinaSt	maksDozvTrajanje
1	Preddiplomska	6
2	Diplomska	4

studij			
sifStudij	nazStudij	sifRazinaSt	trajanje
1	Računarstvo	1	6
2	Elektrotehnika	1	5

Testiranje okidača za INSERT (i procedure):

```
INSERT INTO studij VALUES (3, 'Računarstvo 2', 1, 4); ✓
```

```
INSERT INTO studij VALUES (4, 'Računarstvo 3', 1, 8);
```

Data Output Explain Messages Query History

ERROR: Nedoovoljeno trajanje studija!

CONTEXT: PL/pgSQL function chktrajanjestudij() line 6 at RAISE  
SQL state: P0001

```
INSERT INTO studij VALUES (4, 'Računarstvo 3', 1, 6); ✓
```

## 7. Okidači i procedure

Testiranje okidača za UPDATE (i procedure)

razinaSt		
sifRazinaSt	nazRazinaSt	maksDozvTrajanje
1	Preddiplomska	6
2	Diplomska	4

```
UPDATE studij SET trajanje = 2 WHERE sifStudij = 1; ✓  
UPDATE studij SET trajanje = 8 WHERE sifStudij = 1;
```

[Data Output](#) [Explain](#) [Messages](#) [Query History](#)

ERROR: Nedoovoljeno trajanje studija!  
CONTEXT: PL/pgSQL function chktrajanjestudij() line 6 at RAISE  
SQL state: P0001

```
UPDATE studij SET sifRazinaSt = 2 WHERE sifStudij = 1; ✓  
UPDATE studij SET sifRazinaSt = 2 WHERE sifStudij = 2;
```

2	Elektrotehnika	1	5
---	----------------	---	---

[Data Output](#) [Explain](#) [Messages](#) [Query History](#)

ERROR: Nedoovoljeno trajanje studija!  
CONTEXT: PL/pgSQL function chktrajanjestudij() line 6 at RAISE  
SQL state: P0001

Razmislite: isti okidač za INSERT i UPDATE?

```
CREATE TRIGGER insUpdStudij  
  AFTER INSERT OR UPDATE ...
```

## 8. Okidači i procedure

clan

<i>brClanlisk</i>	<i>prezime</i>	<i>ime</i>	<i>datRod</i>
1001	Horvat	Ivan	05.12.1979.
1002	Novak	Jura	28.10.1978.
1003	Jozić	Jozo	01.07.1951.
1004	Herceg	Ante	12.09.1977.

posudba

<i>brClanlisk</i>	<i>sifPrim</i>	<i>datPosudba</i>	<i>datPovrat</i>
1001	55	30.08.2012.	31.08.2012.
1002	55	02.09.2012.	NULL
1002	57	02.09.2012.	03.09.2012.
1004	58	04.09.2012.	06.09.2012.
1004	56	06.09.2012.	NULL

- Napisati funkciju **brojPosudbi (int)** koja će za dani broj članske iskaznice vratiti broj trenutno posuđenih primjeraka.
- Napisati SQL naredbu (ili više njih) koja će uz pomoć gore napisane funkcije spriječiti unos nove posudbe ako član kod sebe već ima tri posuđena primjerka. Pri tom treba dojaviti prikladnu poruku - npr.  
**Prekoračen dozvoljeni broj posudbi!**

## 8. Okidači i procedure

Naredbe za kreiranje i punjenje relacije **posudba**:

```
CREATE TABLE posudba (  
    brClanIsk    INTEGER,  
    sifPrim      SMALLINT,  
    datPosudba  DATE,  
    datPovrat   DATE,  
    CONSTRAINT pkPosudba PRIMARY KEY (brClanIsk, sifPrim, datPosudba)  
);
```

```
INSERT INTO posudba VALUES (1001, 55, '30.08.2012', '31.08.2012');  
INSERT INTO posudba VALUES (1002, 55, '02.09.2012', NULL);  
INSERT INTO posudba VALUES (1002, 57, '02.09.2012', '03.09.2012');  
INSERT INTO posudba VALUES (1004, 58, '04.09.2012', '06.09.2012');  
INSERT INTO posudba VALUES (1004, 56, '06.09.2012', NULL);
```

## 8. Okidači i procedure

- Funkcija za “brojanje” posuđenih primjeraka:

```
CREATE FUNCTION brojPosudbi (p_brClanIsk INTEGER) RETURNS integer AS
$$
    BEGIN
        RETURN
            ( SELECT COUNT(*)
              FROM posudba
              WHERE brClanIsk = p_brClanIsk
                AND datPovrat IS NULL);
    END
$$ LANGUAGE plpgsql;
```

- Testiranje funkcije:

```
SELECT brojPosudbi (1001); -- =0
SELECT brojPosudbi (1002); -- =1
SELECT brojPosudbi (1004); -- =1
```

posudba			
<i>brClanIsk</i>	<i>sifPrim</i>	<i>datPosudba</i>	<i>datPovrat</i>
1001	55	30.08.2012.	31.08.2012.
1002	55	02.09.2012.	NULL
1002	57	02.09.2012.	03.09.2012.
1004	58	04.09.2012.	06.09.2012.
1004	56	06.09.2012.	NULL

## 8. Okidači i procedure

- Sprječavanje nove (četvrte) posudbe, ako član ima 3 nevraćena primjerka  
→ okidač na akciju INSERT nad relacijom **posudba**

```
CREATE FUNCTION dojavipogreskuPosud () RETURNS trigger As
$$
BEGIN
    RAISE EXCEPTION 'Prekoračen dozvoljeni broj posudbi!';
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER posudbaIns
AFTER INSERT ON posudba
FOR EACH ROW
WHEN (brojPosudbi(NEW.brClanIsk) > 3)
EXECUTE PROCEDURE dojavipogreskuPosud();
```



## 8. Okidači i procedure

- Testiranje okidača (i procedure):

```
INSERT INTO posudba VALUES (1004, 57, CURRENT_DATE, NULL);
```

```
INSERT INTO posudba VALUES (1004, 58, CURRENT_DATE, NULL);
```

Zbog `AFTER INSERT` u trenutku kada se provjerava `WHEN`, n-torka čiji je `INSERT` aktivirao okidač `JE unesena u posudba`, i zbog toga je dobro u uvjetu staviti `> 3`.

Ove dvije `INSERT` naredbe će se uspješno obaviti jer nije ispunjen uvjet okidača:

```
WHEN (brojPosudbi(newPosudba.brClanIsk) > 3).
```

```
INSERT INTO posudba VALUES (1004, 59, CURRENT_DATE, NULL);
```

Pokušaj obavljanja treće `INSERT` naredbe završit će pogreškom – aktivirat će se procedura dojadi `PogreskuPosud` i signalizirati pogrešku pomoću naredbe `RAISE EXCEPTION`.

**ERROR: Prekoračen dozvoljeni broj posudbi!**

- Što kada bi okidač umjesto `AFTER INSERT` bio **BEFORE** `INSERT`?
  - U trenutku kada se provjerava `WHEN`, n-torka čiji je `INSERT` aktivirao okidač još nije unesena u **posudba**, i zbog toga treba u uvjet staviti `> 2` (a ne `> 3`) jer bi u suprotnom bila dozvoljena 4, a ne 3 posuđena primjerka

# Model – telekom operator

SIM	sifSIM	broj	sifKorisnik	status
	1	555111	1	1
	2	555222	2	1
	3	555333	3	1
	4	888000	4	1
	5	888888	NULL	1
	...	...	...	...

korisnik	sifKorisnik	imeKorisnik	prezKorisnik
	1	Ana	Kralj
	2	Ivan	Car
	3	Petra	Knez
	4	Boris	Križanović
	...	...	...

poziv	sifPoziv	sifSim	brPoz	vrPocPoziv	vrKrajPoziv
	100	1	555222	10.05.2019 08:00:00	10.05.2019 08:01:15
	101	2	555111	10.05.2019 10:00:00	10.05.2019 10:00:15
	102	3	888000	10.05.2019 12:00:00	10.05.2019 12:20:00
	103	4	888888	10.05.2019 12:20:05	10.05.2019 12:21:15
	104	5	123456	10.05.2019 20:00:00	10.05.2019 20:15:00
	105	1	123456	11.05.2019 08:00:00	11.05.2019 08:15:00
	106	1	555222	10.06.2019 18:00:00	10.06.2019 18:01:15
	107	2	555111	10.06.2019 19:00:00	10.06.2019 19:00:15
	108	3	888000	10.06.2019 12:00:00	10.06.2019 12:20:00
	...	...	...	...	...

```

create table korisnik (
    sifKorisnik          integer PRIMARY KEY,
    imeKorisnik          varchar(20)    not null,
    prezimeKorisnik      varchar(20)    not null
);

create table sim (
    sifSim              integer PRIMARY KEY ,
    broj                varchar(10) not null,
    sifKorisnik integer,
    status              integer          not null,
    FOREIGN KEY (sifKorisnik) REFERENCES korisnik(sifKorisnik)
);

create table poziv (
    sifPoziv integer PRIMARY KEY,
    sifSim    integer          not null ,
    brPoz     varchar(10) not null,
    vrPocPoziv timestamp       not null,
    vrKrajPoziv timestamp,
    FOREIGN KEY (sifSim) REFERENCES sim(sifSim)
);

insert into korisnik values (1, 'Ana', 'Kralj');
insert into korisnik values (2, 'Ivan', 'Car');
insert into korisnik values (3, 'Petra', 'Knez');
insert into korisnik values (4, 'Boris', 'Križanović');

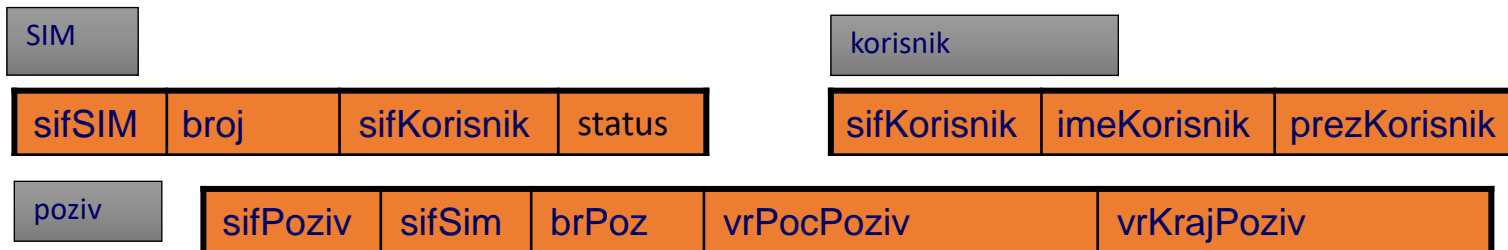
insert into sim values (1, '555111', 1,      1);
insert into sim values (2, '555222', 2,      1);
insert into sim values (3, '555333', 3,      1);
insert into sim values (4, '888000', 4,      1);
insert into sim values (5, '888888', null,    1);

insert into poziv values (100, 1, '555222', '10.05.2019 08:00:00', '10.05.2019 08:01:15');
insert into poziv values (101, 2, '555111', '10.05.2019 10:00:00', '10.05.2019 10:00:15');
insert into poziv values (102, 3, '888000', '10.05.2019 12:00:00', '10.05.2019 12:20:00');
insert into poziv values (103, 4, '888888', '10.05.2019 12:20:05', '10.05.2019 12:21:15');
insert into poziv values (104, 5, '123456', '10.05.2019 20:00:00', '10.05.2019 20:15:00');
insert into poziv values (105, 1, '123456', '11.05.2019 08:00:00', '11.05.2019 08:15:00');
insert into poziv values (106, 1, '555222', '10.06.2019 18:00:00', '10.06.2019 18:01:15');
insert into poziv values (107, 2, '555111', '10.06.2019 19:00:00', '10.06.2019 19:00:15');
insert into poziv values (108, 3, '888000', '10.06.2019 12:00:00', '10.06.2019 12:20:00');

```

## 9a. Privremene tablice

- Napisati naredbe kojima će se kreirati i napuniti privremena relacija *top3*. Relacija *top3* treba sadržavati ime, prezime i broj poziva koje je ostvarilo troje korisnika s najvećim brojem poziva (uzeti u obzir samo brojeve čiji je vlasnik poznat).



## 9a. Privremene tablice

- Napisati naredbe kojima će se kreirati i napuniti privremena relacija *top3*. Relacija *top3* treba sadržavati ime, prezime i broj poziva koje je ostvarilo troje korisnika s najvećim brojem poziva (uzeti u obzir samo brojeve čiji je vlasnik poznat).

```
CREATE TEMP TABLE top3 (  
    ime VARCHAR(20)  
    , prezime VARCHAR(20)  
    , brojPoziva INTEGER);  
  
INSERT INTO top3  
SELECT imeKorisnik, prezimeKorisnik, COUNT(*) AS ukPoziva  
    FROM poziv  
    NATURAL JOIN sim  
    NATURAL JOIN korisnik  
    GROUP BY imeKorisnik, prezimeKorisnik, korisnik.sifKorisnik  
    ORDER BY ukPoziva DESC  
    LIMIT 3
```




## 9a. Privremene tablice

- Napisati naredbe kojima će se kreirati i napuniti privremena relacija *top3*. Relacija *top3* treba sadržavati ime, prezime i broj poziva koje je ostvarilo troje korisnika s najvećim brojem poziva (uzeti u obzir samo brojeve čiji je vlasnik poznat).

```
CREATE TEMP TABLE top3 (ime, prezime, brojPoziva)
AS
SELECT imeKorisnik
      , prezimeKorisnik
      , COUNT(*) AS ukPoziva
FROM poziv
NATURAL JOIN sim
NATURAL JOIN korisnik
GROUP BY imeKorisnik, prezimeKorisnik, korisnik.sifKorisnik
ORDER BY ukPoziva DESC
LIMIT 3
```

## 9a. Privremene tablice




```
SELECT * FROM top3;
```

	<b>ime</b> character varying (20) 	<b>prezime</b> character varying (20) 	<b>brojpoziva</b> bigint 
1	Ana	Kralj	3
2	Ivan	Car	2
3	Petra	Knez	2

## 9a. Privremene tablice

```
INSERT INTO poziv VALUES (109, 4, '555222', '12.06.2019 18:00:00', '10.06.2019 18:01:15');
INSERT INTO poziv VALUES (110, 4, '555111', '12.06.2019 19:00:00', '10.06.2019 19:00:15');
INSERT INTO poziv VALUES (111, 4, '888000', '13.06.2019 12:00:00', '10.06.2019 12:20:00');

SELECT * FROM top3;
```

	<b>ime</b> character varying (20) 	<b>prezime</b> character varying (20) 	<b>brojpoziva</b> bigint 
1	Ana	Kralj	3
2	Ivan	Car	2
3	Petra	Knez	2



## 9b. Virtualne tablice

```
CREATE VIEW top3 (ime, prezime, brojPoziva)
AS
SELECT imeKorisnik
      , prezimeKorisnik
      , COUNT(*) AS ukPoziva
FROM poziv
NATURAL JOIN sim
NATURAL JOIN korisnik
GROUP BY imeKorisnik, prezimeKorisnik, korisnik.sifKorisnik
ORDER BY ukPoziva DESC
LIMIT 3;

SELECT * FROM top3;
```

	ime character varying (20)	prezime character varying (20)	brojpoziva bigint
1	Ana	Kralj	3
2	Ivan	Car	2
3	Petra	Knez	2

## 9b. Virtualne tablice

```
INSERT INTO poziv VALUES (109, 4, '555222', '12.06.2019 18:00:00', '10.06.2019 18:01:15');  
INSERT INTO poziv VALUES (110, 4, '555111', '12.06.2019 19:00:00', '10.06.2019 19:00:15');  
INSERT INTO poziv VALUES (111, 4, '888000', '13.06.2019 12:00:00', '10.06.2019 12:20:00');  
  
SELECT * FROM top3;
```

	ime character varying (20)	prezime character varying (20)	brojpoziva bigint
1	Boris	Križanović	4
2	Ana	Kralj	3
3	Ivan	Car	2

ERROR: cannot update view  
"top3"  
DETAIL: Views containing  
GROUP BY are not  
automatically updatable.

```
UPDATE top3 SET brojPoziva = 5 WHERE ime = 'Boris' and prezime = 'Križanović';
```

## 10. Virtualne tablice

- Kreirati virtualnu relaciju *poziv2* u kojoj će se *sifSim* i *brPoz* zamijeniti s imenima sudionika, ako su poznata. Za *prepaid* korisnike ispisati *anon:<broj>*, a za vanjske korisnike *vanjski*.

SIM

sifSIM	broj	sifKorisnik	status
--------	------	-------------	--------

korisnik

sifKorisnik	imeKorisnik	prezKorisnik
-------------	-------------	--------------

poziv

sifPoziv	sifSim	brPoz	vrPocPoziv	vrKrajPoziv
----------	--------	-------	------------	-------------

```
SELECT * FROM poziv2;
```

sifPoziv	korisnik1	korisnik2	vrPocPoziv	vrKrajPoziv
100	Ana Kralj	Ivan Car	10.05.2019 08:00:00	10.05.2019 08:01:15
101	Ivan Car	Ana Kralj	10.05.2019 10:00:00	10.05.2019 10:00:15
102	Petra Knez	anon:888000	10.05.2019 12:00:00	10.05.2019 12:20:00
103	anon:888000	anon:888888	10.05.2019 12:20:05	10.05.2019 12:21:15
104	anon:888888	vanjski	10.05.2019 20:00:00	10.05.2019 20:15:00
105	anon:888000	vanjski	11.05.2019 08:00:00	11.05.2019 08:15:00
...	...	...	...	...

```

CREATE VIEW poziv2(sifPoziv, korisnik1, korisnik2,
                    vrPocPoziv, vrKrajPoziv) AS

SELECT sifPoziv
, CASE WHEN korisnik1.prezimeKorisnik IS NULL THEN 'anon:' || sim1.broj
      ELSE korisnik1.imeKorisnik || ' ' || korisnik1.prezimeKorisnik
END
, CASE WHEN sim2.broj IS NULL THEN 'vanjski'
      WHEN korisnik2.prezimeKorisnik IS NULL THEN 'anon:' || sim2.broj
      ELSE korisnik2.imeKorisnik || ' ' || korisnik2.prezimeKorisnik
END
, vrPocPoziv
, vrKrajPoziv
FROM poziv JOIN sim AS sim1
      ON poziv.sifSim = sim1.sifSim
LEFT JOIN sim AS sim2
      ON poziv.brPoz = sim2.broj
LEFT JOIN korisnik AS korisnik1
      ON sim1.sifKorisnik = korisnik1.sifKorisnik
LEFT JOIN korisnik AS korisnik2
      ON sim2.sifKorisnik = korisnik2.sifKorisnik

```

```
SELECT * FROM poziv2;
```

	<b>sifpoziv</b> integer	<b>korisnik1</b> text	<b>korisnik2</b> text	<b>vrpocpoziv</b> timestamp without time zone	<b>vrkrajpoziv</b> timestamp without time zone
1	105	Ana Kralj	vanjski	2019-05-11 08:00:00	2019-05-11 08:15:00
2	104	anon:888888	vanjski	2019-05-10 20:00:00	2019-05-10 20:15:00
3	101	Ivan Car	Ana Kralj	2019-05-10 10:00:00	2019-05-10 10:00:15
4	110	Boris Križano...	Ana Kralj	2019-06-12 19:00:00	2019-06-10 19:00:15
5	107	Ivan Car	Ana Kralj	2019-06-10 19:00:00	2019-06-10 19:00:15
6	100	Ana Kralj	Ivan Car	2019-05-10 08:00:00	2019-05-10 08:01:15
7	106	Ana Kralj	Ivan Car	2019-06-10 18:00:00	2019-06-10 18:01:15
8	109	Boris Križano...	Ivan Car	2019-06-12 18:00:00	2019-06-10 18:01:15
9	108	Petra Knez	Boris Križano...	2019-06-10 12:00:00	2019-06-10 12:20:00
10	111	Boris Križano...	Boris Križano...	2019-06-13 12:00:00	2019-06-10 12:20:00
11	102	Petra Knez	Boris Križano...	2019-05-10 12:00:00	2019-05-10 12:20:00
12	103	Boris Križano...	anon:888888	2019-05-10 12:20:05	2019-05-10 12:21:15