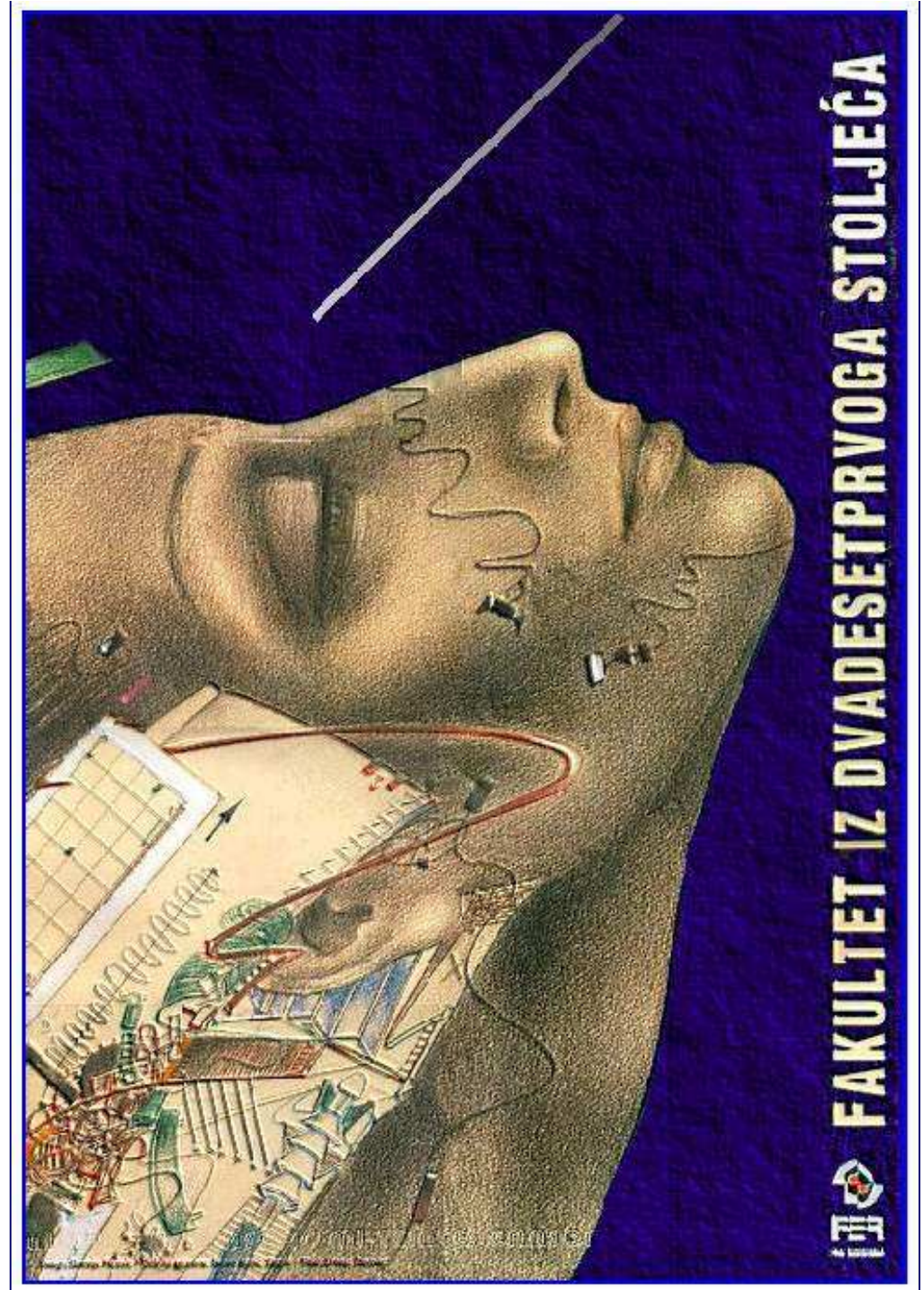


# Baze podataka

lipanj 2009.

Priprema za 3.MI



# Analiza prethodnih međuispita

---

- 1. i 2. MI:
  - SQL + relacijska algebra
  - Integritetska ograničenja
  - Funkcijske zavisnosti
  - B-stablo
  - Indeksi
  - Normalizacija
- Okidači + pohranjene procedure
- Sigurnost
- ER:
  - priča -> ER model
  - ER model -> relacijski model

# Model – telekom operator

**SIM**

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

**korisnik**

sifKorisnik	imeKorisnik	prezKorisnik	brBodova
1	Ana	Kralj	0.00
2	Ivan	Car	0.00
3	Petra	Knez	0.00
...	...	...	

**poziv**

sifPoziv	sifSim	brPoz	vrPocPoziv	vrKrajPoziv
100	1	555222	2009-06-10 18:00:00	2009-06-10 18:01:15
101	2	555111	2009-06-10 19:00:00	2009-06-10 19:00:15
102	3	888000	2009-06-10 12:00:00	2009-06-10 12:20:00
103	4	888888	2009-06-10 12:20:05	2009-06-10 12:21:15
104	5	123456	2009-06-10 20:00:00	2009-06-10 20:15:00
105	4	123456	2009-06-11 08:00:00	2009-06-11 08:15:00
...	...	...	...	...

# 1. zadatak

- Za sve korisnike ispisati broj poziva i ukupno vrijeme poziva po mjesecima u tekućoj godini. Za "prepaid korisnike" kojima nije poznato ime i prezime ispisati anon:<brMob>

korisnik	mjesec	brpoziva	ukvrijeme
anon:888000	6	2	0 00:16:10
anon:888888	6	1	0 00:15:00
Ana Kralj	6	1	0 00:01:15
Petra Knez	6	1	0 00:20:00
Ivan Car	6	1	0 00:00:15

**dobije se  
sumiranjem  
razlike datetime  
tipova**

# 1. zadatak - rješenje

---

```
SELECT CASE WHEN prezKorisnik IS NULL THEN 'anon:' || broj
          ELSE trim(imeKorisnik) || ' ' || prezKorisnik
        END AS korisnik
      , MONTH(vrPocPoziv) AS mjesec
      , COUNT(*) AS brPoziva
      , SUM (vrKrajPoziv - vrPocPoziv) AS ukVrijeme

FROM poziv JOIN sim
      ON poziv.sifSim = sim.sifSim
LEFT JOIN korisnik
      ON sim.sifKorisnik = korisnik.sifKorisnik
WHERE YEAR(vrPocPoziv) = YEAR(today)
GROUP BY 1, 2
```

## 2. zadatak

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

```
SELECT * FROM poziv2;
```

sifpoziv	korisnik1	korisnik2	vrpocpoziv	vrkrajpoziv
100	Ana Kralj	Ivan Car	2009-06-10 18:00:00	2009-06-10 18:01:15
101	Ivan Car	Ana Kralj	2009-06-10 19:00:00	2009-06-10 19:00:15
102	Petra Knez	anon:888000	2009-06-10 12:00:00	2009-06-10 12:20:00
103	anon:888000	anon:888888	2009-06-10 12:20:05	2009-06-10 12:21:15
104	anon:888888	vanjski	2009-06-10 20:00:00	2009-06-10 20:15:00
105	anon:888000	vanjski	2009-06-11 08:00:00	2009-06-11 08:15:00

## 2. zadatak - rješenje

---

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

SELECT sifPoziv
    , CASE WHEN korisnik.prezKorisnik IS NULL THEN 'anon:' || sim.broj
        ELSE trim(korisnik.imeKorisnik) || ' ' || korisnik.prezKorisnik
        END AS korisnik1
    , CASE WHEN sim2.broj IS NULL THEN 'vanjski'
        WHEN korisnik2.prezKorisnik IS NULL THEN 'anon:' || sim2.broj
        ELSE trim(korisnik2.imeKorisnik) || ' ' || korisnik2.prezKorisnik
        END AS korisnik2
    , vrPocPoziv
    , vrKrajPoziv

FROM poziv JOIN sim
    ON poziv.sifSim = sim.sifSim

LEFT JOIN sim AS sim2
    ON poziv.brPoz = sim2.broj

LEFT JOIN korisnik
    ON sim.sifKorisnik = korisnik.sifKorisnik

LEFT JOIN korisnik AS korisnik2
    ON sim2.sifKorisnik = korisnik2.sifKorisnik
```

### 3. zadatak

---

- Osigurati da vrijeme početka poziva prethodi vremenu kraja poziva.

```
ALTER TABLE poziv  
    ADD CONSTRAINT CHECK (vrPocPoziv < vrKrajPoziv)
```



## 4. zadatak

- Telekom operator vodi sustav nagrađivanja *postpaid* (poznatih) korisnika. Svakih 30 sekundi poziva se nagrađuje jednim bodom. Sekunde se ne prenose iz jednog u drugi poziv. Ostvariti ažuran broj bodova u tablici korisnik.
- Pretpostavite da postoji pohranjena procedura `getSecFromInterval` koja za zadani interval vraća broj sekundi

- trigger

- Kako odrediti broj bodova?  
Napisati SP!

sek	sek/30	round(sek/30, 0)	a treba nam:
30	1.0	1	1
31	1.033	1	1
45	1.5	2	1
46	1.533	2	1
60	2.0	2	2

# Neka rješenja

---

```
CREATE PROCEDURE div1(a INTEGER, b INTEGER) RETURNING INTEGER
    RETURN ROUND(a/b-0.5, 0);
END PROCEDURE;
```

```
CREATE PROCEDURE div2(a INTEGER, b INTEGER) RETURNING INTEGER
    RETURN a/b;
END PROCEDURE;
```

```
CREATE PROCEDURE div3(a INTEGER, b INTEGER) RETURNING INTEGER
    RETURN TRUNC(a/ b);
END PROCEDURE;
```

```
CREATE PROCEDURE div4(a INTEGER, b INTEGER) RETURNING INTEGER
    RETURN ( (a - mod(a, b)) / b );
END PROCEDURE;
```

## 4. zadatak - rješenje

---

```
CREATE TRIGGER insPoziv  
  INSERT ON poziv  
  REFERENCING NEW AS noviPoziv
```

da li je nužno?

```
  FOR EACH ROW  
    WHEN ((SELECT COUNT(*) FROM sim  
           WHERE sifSim = noviPoziv.sifSim  
           AND sifKorisnik IS NOT NULL) = 1 )  
    (  
      UPDATE korisnik  
        SET brBodova = brBodova  
          + TRUNC( getSecFromInterval(noviPoziv.vrKrajPoziv  
                                     - noviPoziv.vrPocPoziv)/30  
                )  
      WHERE sifKorisnik = (SELECT sifKorisnik  
                           FROM sim  
                           WHERE sifSim = noviPoziv.sifSim)  
    )
```

# Informativno (nije potrebno znati)

---

```
CREATE PROCEDURE getSecFromInterval (i interval day to
    second) RETURNING INTEGER
RETURN 0
    + (cast ( i as interval second(9) to second)
        :: CHAR(20)
    );
END PROCEDURE;
```

## 5. zadatak

---


- Uvedena je novi, složeniji sustav nagrađivanja. Bodovi se dodjeljuju s obzirom na broj poziva obavljen **tog dana**. Što je broj poziva veći, potrebno je manje sekundi da se "osvoji bod". Vrijedi sljedeća tablica:

broj poziva	broj sekundi
0-10	30
11-20	20
21-	10

# Mijenjamo okidač

```
DROP TRIGGER insPoziv ;
CREATE TRIGGER insPoziv
  INSERT ON poziv
  REFERENCING NEW AS noviPoziv

  FOR EACH ROW
  WHEN ((SELECT COUNT(*) FROM sim
        WHERE sifSim = noviPoziv.sifSim
        AND sifKorisnik IS NOT NULL) = 1 )
  (
    UPDATE korisnik
      SET brBodova = brBodova
        + TRUNC( getSecFromInterval(noviPoziv.vrKrajPoziv
        - noviPoziv.vrPocPoziv)/getSekBod(noviPoziv.sifSim) )
      WHERE sifKorisnik = (SELECT sifKorisnik FROM sim
                          WHERE sifSim = noviPoziv.sifSim)
  )
```



umjesto  
"30"

# Kako odrediti pozive na današnji dan?

---

```
MDY(MONTH(vrKrajPoziv)
    , DAY(vrKrajPoziv)
    , YEAR(vrKrajPoziv)) = TODAY
```

AND

## Kako odrediti pozive u kojima sudjeluje sifSim?

```
(
    sifSim = p_sifSim
OR brPoz = (SELECT broj FROM sim
            WHERE sifSim = p_sifSim)
)
```

# Procedura getSekBod

---

```
CREATE PROCEDURE getSekBod(p_sifSim INTEGER) RETURNING INTEGER
DEFINE br INTEGER;
SELECT COUNT(*)
  INTO br
  FROM poziv
 WHERE MDY(MONTH(vrKrajPoziv), DAY(vrKrajPoziv), YEAR(vrKrajPoziv))
       = TODAY
       AND (
           sifSim = p_sifSim
           OR brPoz  = (SELECT broj FROM sim WHERE sifSim = p_sifSim)
         );
IF (br > 20 ) THEN
  RETURN 10;
ELIF (br > 10) THEN
  RETURN 20;
ELSE
  RETURN 30;
END IF;
END PROCEDURE;
```



# Iznimke

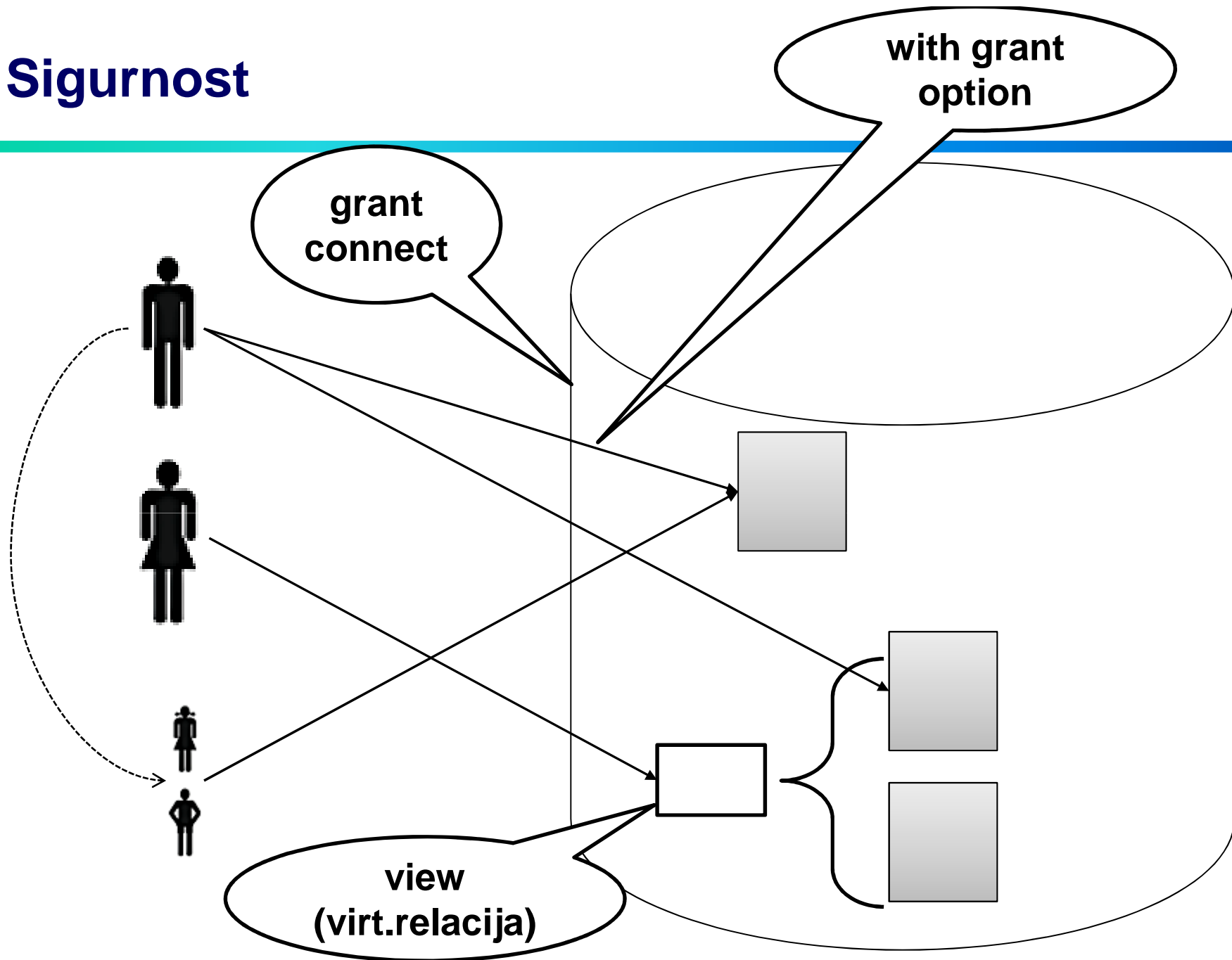
---

```
CREATE PROCEDURE dojaviPogresku(tekstPogreske varchar(255))  
    RAISE EXCEPTION -746, 0, tekstPogreske;  
END PROCEDURE;
```

```
CREATE TRIGGER ...
```

```
... WHEN (...)  
    dojaviPogresku('abc');
```

# Sigurnost



# Sigurnost

---

- Vlasnik baze podataka i svih relacija u bazi je korisnik admin. Nitko nema dozvole pristupa bazi podataka niti objektima baze podataka. Napisati niz naredbi koje treba obaviti korisnik admin, a koje će osigurati sljedeće:

# horvat

---

- **horvatu** omogućiti pregled podatka o svim pozivima, simovima i korisnicima. Osim toga, omogućiti i izmjenu imena i prezimena korisnika

```
GRANT CONNECT TO horvat;
```

```
GRANT SELECT ON sim, poziv, korisnik TO horvat;
```

```
GRANT UPDATE (imeKorisnik, prezKorisnik) ON korisnik TO horvat;
```

# novak

---

- **novaku** omogućiti pregled i izmjenu svih podataka o simovima s mogućnošću dodjeljivanja tih dozvola drugim korisnicima.

```
GRANT CONNECT TO kolar;
```

```
GRANT SELECT, UPDATE ON sim TO kolar WITH GRANT OPTION;
```

# car

---

- **caru** omogućiti pregled svih podataka u bazi ali samo za *prepaid* korisnike

```
CREATE VIEW ppSim (sifSIM, broj, sifKorisnik ) AS
    SELECT sifSIM, broj, sifKorisnik FROM sim
    WHERE sifKorisnik IS NULL;

CREATE VIEW ppPoziv (sifPoziv , sifSIM, brPoz, vrPocPoziv,
    vrKrajPoziv) AS
    SELECT sifPoziv, sifSIM, brPoz, vrPocPoziv, vrKrajPoziv
    FROM poziv
    WHERE sifSim IN (SELECT sifSim FROM sim WHERE sifKorisnik IS NULL);
-- WITH CHECK OPTION ??

GRANT CONNECT TO car;
GRANT SELECT ON ppSim TO car;
GRANT SELECT ON ppPoziv TO car;
-- dobro je:

CREATE PRIVATE SYNONYM car.sim FOR ppSim;
CREATE PRIVATE SYNONYM car.poziv FOR ppPoziv;
```

# ER

---

U zdravstvenoj ustanovi evidentiraju se podaci o liječnicima, pacijentima i obavljenom pregledima.

Za pacijenta se evidentira šifra, ime, prezime, datum rođenja, mjesto rođenja, te adresa i mjesto stanovanja. Za mjesto se evidentira poštanski broj i naziv.

Čuvaju se podaci svakog obavljenog pregleda: 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. Za liječnika se evidentira šifra, ime i prezime.

Na jednom pregledu može biti postavljeno više dijagnoza (npr. peludna alergija i migrena).

Dijagnoze imaju šifru i naziv.

Zbog svake postavljene dijagnoze na jednom pregledu pacijent je upućen na barem jednu ili više terapija. Na istu terapiju pacijent može biti upućen zbog više dijagnoza.

Za terapiju se evidentira šifra i opis terapije (npr. 1-masaža, 2-aromaterapija...).

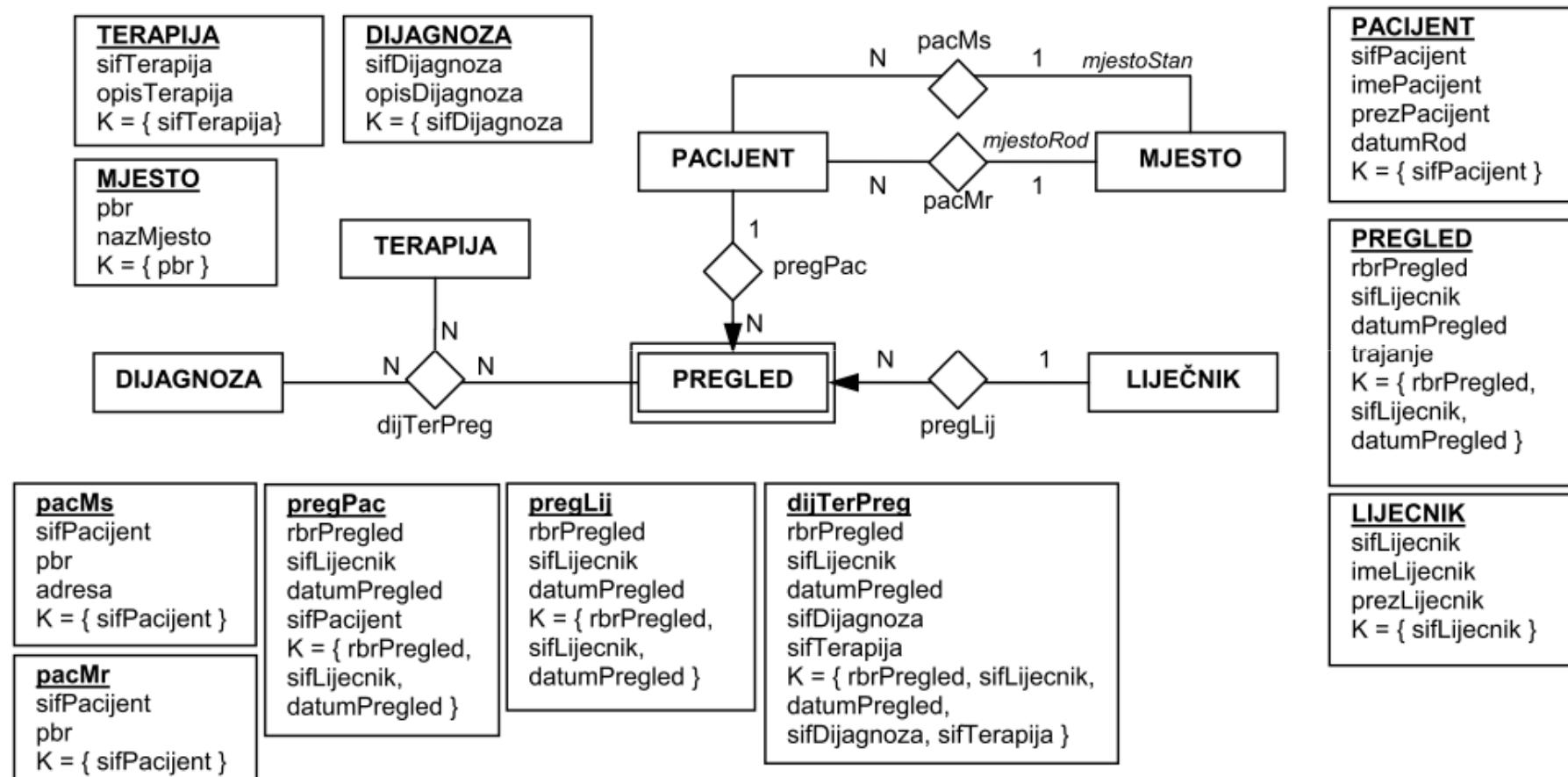
# ER - zadatak

---

- **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 mjesto, te veze koje postoje među tim entitetima, napisati ekvivalentni relacijski model u obliku SQL naredbi za kreiranje relacija s ugrađenim općim pravilima integriteta.**



# ER – rješenje model



# ER – rješenje SQL

---

```
CREATE TABLE mjesto (pbr          INTEGER
                     , nazMjesto CHAR(20)
                     , PRIMARY KEY (pbr));

CREATE TABLE pacijent (sifPacijent INTEGER
                      , imePacijent CHAR(20)
                      , prePacijent CHAR(20)
                      , datumRod    DATE
                      , pbrRod       INTEGER
                      , pbrStan      INTEGER
                      , adresa       CHAR(60)
                      , PRIMARY KEY (jmbg)
                      , FOREIGN KEY (pbrRod) REFERENCES mjesto(pbr)
                      , FOREIGN KEY (pbrStan) REFERENCES mjesto(pbr));
```

```
CREATE TABLE pregled (rbrPregled  SMALLINT
                     , sifLijecnik  INTEGER
                     , datumPregled DATE
                     , trajanje     SMALLINT
                     , sifPacijent  INTEGER
                     , PRIMARY KEY (rbrPregled,sifLijecnik,datumPregled)
                     , FOREIGN KEY (sifPacijent)
                       REFERENCES pacijent(sifPacijent));
-- može se izostaviti jer nije dio segmenta
-- , FOREIGN KEY (sifLijecnik)
--   REFERENCES lijecnik(sifLijecnik)
```