BAZE PODATAKA

Predavanje broj: 10

Nastavne teme:

- ✓SKUPOVNI OPERATORI
- ✓ZAŠTITA SIGURNOSTI I INTEGRITETA PODATAKA

SKUPOVNI OPERATORI

Programski jezik SQL dozvoljava kombinovanje rezultata većeg broj SQL upit korišćenjem operacija za rad sa skupovima:

- UNION – unija,
- UNION ALL – unija svih (u rezultatu su moguće duplirane vrste),
- INTERSECT – presek
- EXCEPT – razlika.

SKUPOVNI OPERATORI

- Da bi se dva upita mogla spojiti ovim operatorima moraju biti kompatibilni, tj. liste atributa ili izraza iza SELECT klauzule moraju biti kompatibilne po broju i tipu (moraju da uzimaju vrednosti iz istih domena).

- Rezultujuća tabela će imati iste nazive kolona kao što su nazivi kolona u prvom upitu.

- Svi skupovni operatori imaju isti prioritet, a promenu prioriteta možemo zahtevati upotrebom malih zagrada.

SKUPOVNI OPERATORI

Operatori unije, preseka, razlike i Dekartov proizvod se zasnivaju na teoriji skupova, ali su donekle izmenjeni jer se primenjuju nad relacijama.

SKUPOVNI OPERATORI

- Unija je u suštini relaciona verzija sabiranja.

- Klauzula UNION kombinuje rezultate dva ili više upita u jednu rezultujuću tabelu.

- Rezultati upita koji se kombinuju moraju imati kolone koje se slažu po:
  - broju (isti broj kolona),
  - redosledu (odgovarajuće kolone se nalaze na istim pozicijama)
  - tipu (odgovarajuće kolone moraju da imaju kompatibilne tipove).

SKUPOVNI OPERATORI

- Klauzula UNION kombinuje rezultate dva ili više upita u jednu rezultujuću tabelu.

- Prilikom izvršavanje operacije UNION eliminišu se duplikati.
  - Zapis koji se pojavljuje u više rezultujućih tabela, u rezultatu unije pojavljuje se samo jednom.

- Potrebno je voditi računa da rezultujuće tabele koje se spajaju moraju da uključuju atribute neophodne za jednoznačnu identifikaciju zapisa.

- Unija omogućava dodavanje novih zapisa u relaciju.

SKUPOVNI OPERATORI

- Presek je operacija koja vraća zapise koji su zajednički u obe relacije i kojom se u stvari pronalaze duplikati.

SKUPOVNI OPERATORI

Prikazati ime, broj odeljenja i platu za zaposlene koji rade u odeljenju 20 kao i za zaposlene koji imaju platu manju od 1200 bez obzira u kom odeljenju rade.

```
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE id_odeljenja=20
UNION
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE plata<1200;
```

ime	id_odeljenja	plata
Petar	20	1300
Božidar	20	2200
Slobodan	20	900
Mitar	20	2600
Ivan	20	1600
Aleksandar	10	1000
Jovan	10	1000
Mirjana	10	1000
Tomislav	10	1000
Andrija	30	1100
Jovan	30	1100
Marko	30	975

SKUPOVNI OPERATORI

Prethodni upit vraća prvo radnike koje rade u odeljenju 20, a zatim na taj spisak dodaje radnike koji imaju platu manju od 1200.

Skupovni operator UNION ne prikazuje duplikate.

SKUPOVNI OPERATORI

Prikazati ime, broj odeljenja i platu za zaposlene koji rade u odeljenju 20 kao i za zaposlene koji imaju platu manju od 1200 bez obzira u kom odeljenju rade.

```
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE id_odeljenja=20
UNION ALL
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE plata<1200;
```

ime	id_odeljenja	plata
Petar	20	1300
Božidar	20	2200
Slobodan	20	900
Mitar	20	2600
Ivan	20	1600
Aleksandar	10	1000
Jovan	10	1000
Mirjana	10	1000
Tomislav	10	1000
Andrija	30	1100
Slobodan	20	900
Jovan	30	1100
Marko	30	975

SKUPOVNI OPERATORI

Prethodni upit vraća prvo radnike koje rade u odeljenju 20, a zatim na taj spisak dodaje radnike koji imaju platu manju od 1200.

Skupovni operator UNION ALL prikazuje i duplikate.

SKUPOVNI OPERATORI

Umesto operatora UNION može se koristiti operator OR.

```
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE plata<1200 OR id_odeljenja=20;
```

Prikazuju se imena radnika koji ili rade u odeljenju 20 ili imaju platu manju od 1200 bez obzira u kom odeljenju rade. Radnicima iz odeljenja 20 pridružuju se radnici iz drugih odeljenja sa platom manjom od 1200 (iz odeljenja 10 i odeljenja 30).

S obzirom da UNION ALL vraća i duplikate, operator OR se ne može koristiti umesto ovog operatora.

ime	id_odeljenja	plata
Petar	20	1300
Aleksandar	10	1000
Jovan	10	1000
Mirjana	10	1000
Božidar	20	2200
Tomislav	10	1000
Andrija	30	1100
Slobodan	20	900
Mitar	20	2600
Jovan	30	1100
Marko	30	975
Ivan	20	1600

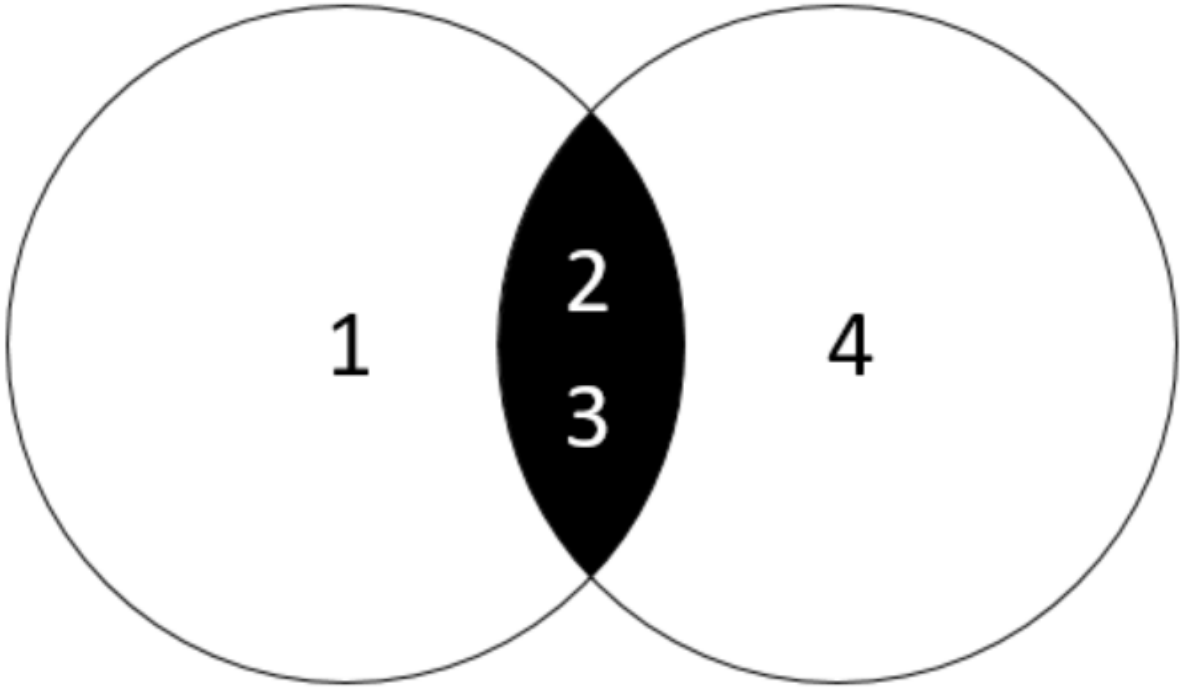
SKUPOVNI OPERATORI

INTERSECT operator je operatot koji vraća samo različite redove rezultata dva ili više upita.

```
(SELECT column_list
FROM table_1)
INTERSECT
(SELECT column_list
FROM table_2);
```

SKUPOVNI OPERATORI

Operator INTERSECT uspoređuje skupove rezultata dva upita i vraća različite redove koji se dobijaju iz oba upita.



SKUPOVNI OPERATORI

Za razliku od operatora UNION, operator INTERSECT vraća presek između dva kruga.

Pravila primene:

- Redosled i broj kolona u SELECT listi za upite moraju biti isti.
- Tipovi podataka u odgovarajućim kolonama moraju biti kompatibilni.

SKUPOVNI OPERATORI

MySQL ne podržava INTERSECT operator.

Moguće je realizovati ponašanje INTERSECT operatora pomoću:

- DISTINCT i INNER JOIN
- PODUPITA

SKUPOVNI OPERATORI

MySQL ne podržava INTERSECT operator.

Moguće je realizovati ponašanje INTERSECT operatora pomoću:

- AND operatora
- DISTINCT i INNER JOIN
- PODUPITA

SKUPOVNI OPERATORI

Prikazati podatke o radnicima koji rade u odeljenju 20 i imaju platu manju od 1200 .

```
SELECT ime, id_odeljenja, plata
FROM RADNIK
WHERE plata<1200 AND  id_odeljenja=20;
```

ime	prezime	plata
Slobodan	Petrović	900

SKUPOVNI OPERATORI

Sledeći upit koristi DISTINCT operator i klauzu INNER JOIN za vraćanje različitih redova u obe tabele:

```
SELECT DISTINCT
   id
FROM t1
   INNER JOIN t2 USING(id);
```

Klauzula INNER JOIN vraća redove iz leve i desne tabele.
Operator DISTINCT uklanja duplikate iz prikaza.

SKUPOVNI OPERATORI

Sledeći upit koristi IN operator i podupit da vrati presek dva skupa rezultata.

```
SELECT DISTINCT id
FROM t1
WHERE id IN (SELECT id FROM t2);
```

SKUPOVNI OPERATORI

EXTRACT upoređuje rezultate dva upita i vraća različite redove iz skupa rezultata prvog upita koji se ne pojavljuju u grupi rezultata drugog upita.

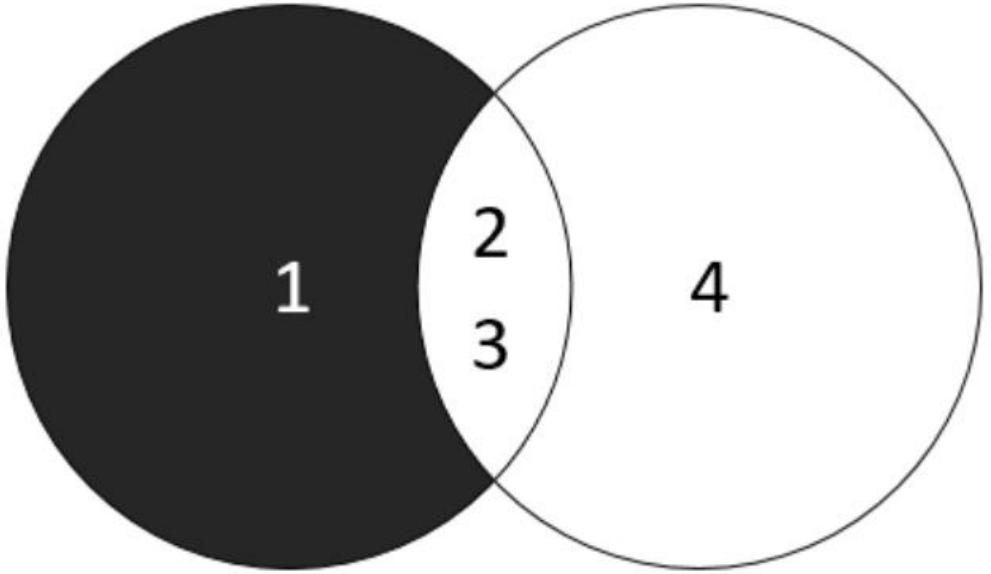
```
SELECT select_list1
FROM table_name1
EXTRACT
SELECT select_list2
FROM table_name2;
```

SKUPOVNI OPERATORI

Osnovna pravila za upit koji koristi EXTRACT operator:

- Broj i redosled kolona u select_list1 i select_list2 moraju biti isti.
- Tipovi podataka iz odgovarajućih kolona u oba upita moraju biti kompatibilni.

SKUPOVNI OPERATORI



id
1
2
3

t1 table

MINUS

id
2
3
4

t2 table

id
1

SKUPOVNI OPERATORI

MySQL ne podržava EXTRACT operator.

Moguće je realizovati ponašanje INTERSECT operatora pomoću:

- AND i NOT operatora
- JOIN

SKUPOVNI OPERATORI

Sledeći upit prikazuje prezimena i imena radnika koji rade u odeljenju 10 pri čemu su iz spiska uklonjena preimena i mena svih radnika koji imaju platu veću od 18000.

```
SELECT prezime, ime
FROM RADNIK
WHERE id_odeljenja=10
EXCEPT
SELECT prezime, ime
FROM RADNIK
WHERE plata>18000
```

Sledeći upit prikazuje prezimena i imena radnika koji rade u odeljenju 10 nemaju platu veću od 18000, tj. imaju platu manju od18000.

```
SELECT prezime, ime
FROM RADNIK
WHERE (id_odeljenja =10 AND (NOT(plata>18000)))
```

SKUPOVNI OPERATORI

MySQL ne podržava EXTRACT operator. Međutim, može se koristiti JOIN za realizaciju ovog operatora.

.

```
SELECT
    select_list
FROM
    table1
LEFT JOIN table2
    ON join_predicate
WHERE
    table2.column_name IS NULL;
```

```
SELECT
    id
FROM
    t1
LEFT JOIN
    t2 USING (id)
WHERE
    t2.id IS NULL;
```

Primeri upotrebe skupovnih operatora

Neka su u bazi podataka VISER kreirane i popunjenje sledeće tabele:

```
CREATE TABLE JEDINICA
        (       br_jed# TINYINT PRIMARY KEY ,
                ime_jed CHAR(30) NOT NULL UNIQUE )
        CREATE TABLE RADNIK
         (      id_br#  INT PRIMARY KEY
                CONSTRAINT id_br#_provera CHECK (id_br#>=1 AND id_br#<=99999),
                ime CHAR(15) NOT NULL,
                prezime CHAR(20) NOT NULL,
                kvalif CHAR(3),
                datum_zap SMALLDATETIME,
                posao CHAR(20),
                plata REAL,
                br_jed$ TINYINT CONSTRAINT br_jed$_sk REFERENCES  JEDINICA(br_jed#)
                        ON DELETE NO ACTION
                        ON UPDATE  CASCADE)
```

Primeri upotrebe skupovnih operatora

Zapazimo da atributi ime i prezime iz tabela RADNIK i NASTAVNIK imaju iste domene!

```
CREATE TABLE NASTAVNIK
 (          id_br#  INT PRIMARY KEY
            CONSTRAINT id_br#_provera CHECK (id_br#>=1 AND id_br#<=99999),
            ime CHAR(15) NOT NULL,
            prezime CHAR(20) NOT NULL,
            zvanje CHAR(20),
            datum_zvanje SMALLDATETIME,
            oblast CHAR(50),
            plata REAL,
            br_jed$ TINYINT CONSTRAINT br_jed$_sk REFERENCES JEDINICA(br_jed#)
                        ON DELETE NO ACTION
                        ON UPDATE  CASCADE)
```

Primeri upotrebe skupovnih operatora

Sledeća naredba prikazuje prezimena i imena svih radnika iz tabele RADNIK a zatim na taj spisak dodaje prezimena i imena svih nastavnika iz tabele NASTAVNIK. Broj vrsta u rezultujućoj tabeli je jednak zbiru brojeva vrsta u tabelama RADNIK i NASTAVNIK. Moguće su identične vrste.

```
SELECT prezime, ime
FROM RADNIK
 UNION ALL
SELECT prezime, ime
FROM NASTAVNIK
```

Primeri upotrebe skupovnih operatora

Sledeća izdvaja prvo prezimena i imena radnika, zatim na taj spisak dodaje prezimena i imena nastavnika i na kraju uklanja duplikate.

```
SELECT prezime, ime
FROM RADNIK
UNION
SELECT prezime, ime
FROM NASTAVNIK
```

Primeri upotrebe skupovnih operatora

Sledeća naredba prikazuje prezimena i imena koja se javljaju i u tabeli RADNIK i tabeli NASTAVNIK. Identične vrste su eliminisane. Svako prezime i ime iz rezulatata je prezime i ime bar jednog radnika i bar jednog nastavnika.

```
SELECT prezime, ime
FROM RADNIK
INTERSECT
SELECT prezime, ime
FROM NASTAVNIK
```

Primeri upotrebe skupovnih operatora

Sledeća naredba prikazuje sva različita prezimena i imena koja se odnose isključivo na radnike. Nema identičnih vrsta. Prezimena i imena u rezultatu su uređena po abecedi.

```
SELECT prezime, ime
FROM RADNIK
EXCEPT
SELECT prezime, ime
FROM NASTAVNIK
```

Zaštita sigurnosti i integriteta podataka

Zaštita sigurnosti podataka – zaštita od neovlašćenog ažuriranja i korišćenja. Najrasprostranjeniji princip je ograničavanje prava pristupa. Obično se svakom korisniku dodeljuju privilegije koje određuju koje operacije korisnik može da obavi nad objektima (tabelama) baze podataka. Tabela koju korisnik kreira je njegova tabela, on je njen vlasnik (owner). Drugi korisnik ima ona prava korišćenja te tabele koja mu je dodelio vlasnik naredbom GRANT.

Zaštita integriteta podataka – zaštita od slučajnog i/ili pogrešnog ažuriranja. Narušavanje integriteta može nastati prilikom paralelnog izvršavanja transakcija. Transakcija baze podataka je niz operacija nad bazom koji odgovara jednoj logIčkoj jedinici posla u realnom sistemu. Savremeni DBMS-ovi veoma uspešno upravljaju transakcijama.

Naredbe za kontrolu pristupa podacima

GRANT: naredba dodele
REVOKE: naredba uklanjanja

Kreiranje korisnika se obavlja istovremeno sa dodelom jednog od opštih prava:

```
GRANT OpštePravo TO Korisnik
        IDENTIFIED BY Lozinka;
```

Korisnik: javni podatak za najavu korisnika sistemu za upravljanje BP
Lozinka: tajni naziv, kojom korisnik kompletira postupak najave

Naredbe za kontrolu pristupa podacima

OpštePravo: može biti jedno od sledećih:

- CONNECT: sva dodeljena posebna prava nad tabelama, plus kreiranja pogleda nad tim tabelama
- RESOURCE: prethodna prava plus prava kreiranja osnovnih tabela
- DBA(Data Base Administrator): neograničena prava nad BP, koja ima korisnik koji je administrator BP

Naredbe za kontrolu pristupa podacima

Korisnik se najčešće kreira sa CONNECT pravom

Opšte pravo iznad CONNECT može se dodeliti sa naredbom u formi:

```
GRANT OpštePravoTO Korisnik;
```

Administrator BP može ukloniti neko opšte pravo nekom korisniku naredbom:

```
REVOKE OpštePravo FROM Korisnik;
```

Ako se uklone RESOURCE i DBA pravo, korisniku ostaje CONNECT pravo. Uklanjanjem CONNECT prava, uklanja se i korisnik i on se više ne može najavljivati za rad sa BP.

Posebna prava

Posebna prava se odnose na tabele i poglede u BP, kao i na dozvoljene radnje nad njima.

Sintaksa naredbe dodele posebnog prava:

```
GRANT {Pravo,…} | ALL
    ON ImeTabele | ImePogleda
    TO {Korisnik,…} | PUBLIC
    [WITH GRANT OPTION];
```

Jedno ili više prava mogu se dodeliti jednom ili više korisnika

WITH GRANT OPTION: navedeni korisnici mogu dalje dodeljivati sva navedena prava

PUBLIC- obuhvata sve postojeće korisnike

ALL - obuhvata sva prava

Posebna prava

Posebno pravo nad tabelom ili pogledom:

- SELECT: pravo upita
- INSERT: pravo ubacivanja novih redova
- UPDATE [ListaKolona]: Pravo izmene svih kolona ili samo navedenih
- DELETE: pravo uklanjanja redova
- REFERENCES [ListaKolona]: pravo referisanja kolona preko stranih ključeva

Posebna prava

Naredba za dodelu prava GRANT ima sledeći opšti oblik:

GRANT (ALL | (<SQL_naredba> ON <ime_tabele>) TO
<ime_korisnika>

Opcijom ALL vlasnik daje sva prava drugom korisniku.

- Pravo SELECT znači da drugi korisnik može da gleda sve ili samo specificirane atribute ukazane tabele.
- Pravo DELETE znači da drugi korisnik može da briše slogove specificirane tabele.
- Pravo UPDATE daje drugom korisniku pravo da ažurira sve ili samo specificirane atribute ukazane tabele.
- Prava CREATE TABLE, CREATE VIEW i CREATE INDEX daju pravo korisniku da kreira tebele, poglede i indekse u datoj bazi podataka.
- Pod određenim uslovima, korisnik može sva ili manja prava od onih koje je dobio proslediti nekom sledećem korisniku.

Primeri naredbi za dodelu prava

1. Sledeća naredba korisniku korisničkog imena **Marko** dodeljuje sva prava nad bazom podataka PREDUZECE.

```
USE PREDUZECE
GRANT ALL TO Marko
```

2. Sledeća naredba korisniku korisničkog imena **Janko** dodeljuje pravo kreiranja tabela. Pretpostavlja se da je otvorena baza podataka PREDUZECE.

```
GRANT CREATE TABLE TO Janko
```

3. Sledeća naredba korisniku korisničkog imena **Janko** dodeljuje pravo da gleda atribute **id_radnika**, **ime** i **prezime** u tabeli RADNIK. Pretpostavlja se da je otvorena baza podataka PREDUZECE.

```
GRANT SELECT(id_radnika, ime, prezime) ON RADNIK TO Janko
```

Primeri naredbi za dodelu prava

4. Sledeća naredba korisniku korisničkog imena **Janko** dodeljuje pravo da menja atribute **posao** i **plata** u tabeli RADNIK. Pretpostavlja se da je otvorena baza podataka PREDUZECE.

```
GRANT UPDATE(posao, plata) ON RADNIK TO Janko
```

5. Sledeća naredba korisniku korisničkog imena **Janko** dodeljuje pravo da briše slogove u tabeli ODELJENJE. Pretpostavlja se da je otvorena baza podataka PREDUZECE.

```
GRANT DELETE ON ODELJENJE  TO Janko
```

Primeri naredbi za oduzimanje prava

Oduzimanje prava vrši se naredbom REVOKE koja ima sledeći opšti oblik:

REVOKE (**ALL** | (<SQL_naredba> **ON** <ime_tabele>) **FROM**
<ime_korisnika>

1. Sledeća naredba korisniku korisničkog imena **Marko** oduzima sva dodeljena prava nad bazom podataka PREDUZECE.

   USE PREDUZECE

   REVOKE ALL FROM Marko

2. Sledeća naredba korisniku korisničkog imena **Janko** oduzima pravo da ažurira atribute ***posao*** i ***plata*** u tabeli RADNIK. Pretpostavlja se da je otvorena baza podataka PREDUZECE.

   REVOKE UPDATE (posao, plata) ON RADNIK FROM Janko

Primer

Neka u RBP Biblioteka imamo tri korisnika koji treba da imaju sledeća prava:

Sef: prava ažuriranja podataka o oblastima, naslovima, autorima, autorstvu i knjigama, plus prava upita nad svim podacima

Radnik: prava ažuriranja podataka o članovima, držanju knjiga, pozajmicama i rezervacijama, plus prava upita nad svim podacima

Clan: prava upita nad podacima o oblastima, naslovima, autorima i autorstvu

Primer

Kreiranje korisnika

```
GRANT CONNECT TO Sef IDENTIFIED BY LozinkaSefa;
GRANT CONNECT TO Radnik IDENTIFIED BY LozinkaRadnika;
GRANT CONNECT TO Clan IDENTIFIED BY LozinkaClana;
```

Primer

Davanje prava upita

```
GRANT SELECT ON Oblast, Naslov, Autor, Je_Autor TO PUBLIC;
GRANT SELECT ON Knjiga, Clan, Drzi, Pozajmica, Rezervacija TO Sef,
Radnik;
```

Davanje prava ažuriranja

```
GRANT INSERT, UPDATE, DELETE ON Clan, Drzi, Pozajmica,
Rezervacija TO Radnik;
GRANT INSERT, UPDATE, DELETE ON Oblast, Naslov, Autor, Je_Autor,
Knjiga TO Sef;
```