Uvod u relacione baze podataka

Predavanja, sedmica 6

24. novembar 2022.

Sadržaj

Literatura za 6. sedmicu

- C.J.Date: An Introduction to Database Systems, VIII ed, Addison Wesley Inc, 2004. Poglavlje 9
- pisani materijal koji prati predavanja

Napomena

Slajdovi su nastali obradom materijala profesora Nenada Mitića za predmet Uvod u relacione baze podataka

Sadržaj

Uklanjanje entiteta

Postojeći entiteti u tabeli se uklanjaju naredbom DELETE

DELETE FROM <tabela>
WHERE <uslov>

Primer

Primer: Obrisati sve ispite na kojima je polagan predmet Razvoj softvera (id 4005).

```
delete from ispit
where id_predmeta=4005
```

Primer

Primer: Obrisati sve ispite na kojima je polagan predmet Razvoj softvera (id 4005) od strane studenata koji su rođeni pre više od 20 godina.

```
delete from ispit
where id_predmeta=4005
      and indeks in ( select indeks
                      from dosije
                      where datum_rodjenja < current date - 20 years)
```

Promena vrednosti tabele na osnovu sadržaja druge tabele/tabela

Naredba MERGE ažurira ciljnu tabelu koristeći podatke iz druge tabele ili tabela (u nastavku označini kao upit). Sintaksa:

MERGE INTO <ciljna-tabela> [<alias>]?
USING <upit> [<alias>]?
ON <uslov-spajanja-redova-iz-cilja-i-upita>
[WHEN MATCHED [AND <uslov>]* THEN
<akcija1>]*
[WHEN NOT MATCHED [AND <uslov>]* THEN
<akcija2>]*

Promena vrednosti tabele na osnovu sadržaja druge tabele/tabela

<akcija1> može biti

- brisanje, i navodi se samo

  delete

- ažuriranje, kada se koristi sintaksa

  update
  set <izraz-dodele>

Promena vrednosti tabele na osnovu sadržaja druge tabele/tabela

<akcija2> može biti za unos redova i koristi sintaksu

insert [<lista-kolona>]?
values (<lista-vrednosti>)

Primer

Primer: Napraviti tabelu dosije_prosek koja ima tri kolone: indeks studenta, prosek studenta i broj položenih ispita.

```
create table dosije_prosek (

       indeks integer not null,
       prosek float,
       broj_polozenih smallint
)
```

Primer

U tabelu dosije_prosek uneti indekse studenata iz Beograda.

```
insert into dosije_prosek (indeks)
select indeks
from dosije
where mesto_rodjenja='Beograd'
```

Primer

Napisati naredbu za menjanje sadržaja tabele dosije_prosek koja

- za studente o kojima već postoje podaci i imaju prosek bar 8,5 ažurira prosek i broj položenih ispita
- briše podatke o studentima o kojima već postoje podaci u tabeli dosije_prosek, a prosek im je manji od 8,5
- unosi podatke o studentima koji imaju prosek bar 8,5 i o njima nema podataka u tabeli dosije_prosek.

Primer

```
merge into dosije_prosek dp
using ( select d.indeks, count polozeno, avg(ocena*1.0) prosek
        from dosije d join ispit i
             on d.indeks=i.indeks and ocena>5
        group by d.indeks ) as t
on dp.indeks=t.indeks
when matched and t.prosek>=8.5 then
     update
     set (prosek, broj_polozenih)=(t.prosek, t.polozeno)
when matched and t.prosek<8.5 then
     delete
when not matched and t.prosek>=8.5 then
     insert
     values(indeks, prosek, polozeno)
```

Izdvajanje željenog broja redova

Klauzula FETCH FIRST se koristi za zadavanje maksimalnog broja redova koji će se prikazati u rezultatu. Sintaksa:

FETCH FIRST <broj-redova> ROWS ONLY

i zadaje se posle order by klauzule

Primer

Primer: Prikazati prva dva sloga iz tabele dosije.

```
select *
from   dosije
fetch first 2 rows only
```

Izdvajanje željenog broja redova

Klauzula LIMIT se koristi za zadavanje da se prikaže najviše <broj-redova-za-izdvajanje> redova u rezultatu, ali da se ne prikaže prvih <broj-prvih-redova-za-preskakanje> redova. Sintaksa:

LIMIT <broj-redova-za-izdvajanje> OFFSET <broj-prvih-redova-za-preskakanje>

i zadaje se posle order by klauzule

Primer

Primer: Urediti studente po indeksu i prikazati 5 studenata , ali izostaviti prva 3 studenta.

```
select *
from dosije
order by indeks
limit 5 offset 2
```

Sadržaj

Naredbe DDL

- create *objekat ime_objekta* ... - za pravljenje objekta u bazi podataka
- drop *objekat ime_objekta* - za brisanje objekta iz baza podataka
- alter *objekat ime_objekta* - za menjanje objekta u bazi podataka
- *objekat*
  - database
  - schema
  - table
  - view
  - ...

Pravljenje i brisanje baze podataka

Naredba za pravljenje baze podataka

CREATE DATABASE <ime-baze-podataka>
[ALIAS <alias-baze-podataka>]?
[USING CODESET <kod> TERRITORY <teritorija>]?
[PAGESIZE <ceo-broj> [K]?]?

Primeri

Primer: Napraviti bazu podataka stud.

```
create db stud
   using codeset utf-8 territory sp
   pagesize 32768
```

Primeri

Primer: Obrisati bazu podataka stud.

```
drop db stud
```

Pravljenje šeme baze podataka

Naredba za pravljenje šeme baze podataka

CREATE SCHEMA <ime-sheme>
[AUTHORIZATION ime-vlasnika-sheme]?
[CREATE TABLE ...]?
[CREATE VIEW ...]?
[CREATE INDEX ...]?

Primer: Napraviti shemu stud.

```
create schema stud
```

Primer: Obrisati shemu stud.

```
drop schema stud restrict
```

Primer: Napraviti shemu stud i u njoj tabelu semestar sa dve kolone: sk_godina i semestar.

```
create schema stud
create table semestar (
sk_godina smallint,
semestar smallint)
```

```
select *
from stud.SEMESTAR
```

Integritet u RBP

- Pojam *integritet* se u kontekstu baza podataka odnosi na preciznost, punovažnost i korektnost podataka u bazi
- Održavanje integriteta podataka je od najveće važnosti za RSUBP. Zbog toga se u sistemu definišu pravila (tzv. ograničenja integriteta) koja se primenjuju na podatke
- Intuitivno, ograničenje integriteta je logički izraz pridružen bazi za koji se zahteva da njegovo izračunavanje uvek daje vrednost tačno
- Ograničenja se proveravaju pri pravljenju objekata u bazi ili menjanju njihovog sadržaja

Zlatno pravilo

Nijednoj operaciji ažuriranja nije dozvoljeno da ostavi bilo koju relaciju u stanju koje narušava bilo koje od ograničenja te relacije

ili

Nijednoj operaciji ažuriranja nije dozvoljeno da ostavi bilo koju bazu podataka u stanju u kome se neko od ograničenja baze izračunava kao netačno (posledica: pre bilo kakvog stvarnog ažuriranja proverava se važenje ograničenja)

Klasifikacija ograničenja integriteta

Klasifikacija prema tipu ograničenja koje mora da bude ispoštovano u bazi

- Ograničenje baze podataka
- Ograničenje relacije
- Ograničenja atributa
- Ograničenja tipova

Ograničenja tipova

- Ograničenja tipa: definicija skupova vrednosti koji čine dati tip
- Uvek se proveravaju pri pokušaju unosa/promene te nijedna relacija ne može da sadrži vrednost u atributu koja nije odgovarajućeg tipa

Ograničenja atributa

- Ograničenja atributa: ograničenja na skup dozvoljenih vrednosti datog atributa
- Predstavljaju deo definicije atributa

Ograničenja tabele i baze podataka

- Ograničenja relacija : zadaje se ograničenje na vrednost pojedinačne relacije koje se proverava pri ažuriranju te relacije
- Ogranicenja baze: ograničenja koja se odnose na vrednosti koje je dozvoljeno čuvati u bazi (tj. koje se odnose na dve ili više različitih relacija)
- Posebna grupa ograničenja - ograničenja prelaza
- Ograničenje prelaza je ograničenje mogućeg prelaza iz jedne u drugu vrednost

Ograničenja relacije - primer

```
CONSTRAINT REL1
  IF NOT ( IS_EMPTY ( PREDMET ) ) THEN
    COUNT ( PREDMET
      WHERE SIFRA= SIFRA ('R270')) > 0
END IF
```

Ako uopšte postoji neki predmet tada bar jedna od njih mora da ima šifru R270.

Ograničenje baze podataka - primer

```
CONSTRAINT BAZA1
  FORALL DOSIJE D FORALL ISPIT I
     IS_EMPTY (( D JOIN I )
                WHERE I.INDEKS > 20150000
                AND I.INDEKS = D.INDEKS
                AND GODINA_ROKA=GODINA_ROKA(2015)
```

Ni jedan student upisan na studije 2015 godine ne može da polaže uspit u 2015 godini.

Ograničenja prelaza

Primer: ako baza sadrži podatke o osobama tada su važeća sledeća ograničenja:

- Nije dozvoljeno venčanje već venčanih osoba
- Dozvoljeno je venčati se sa razvedenom osobom
- Osobe koje više nisu žive ne mogu da primaju platu (penziju, ...)
- ...

Koja su ograničenja prelaza u studentskoj bazi podataka?

Kandidat za ključ

Kandidat za ključ relacije R($X_1, X_2, ..., X_n$) predstavlja podskup atributa X te relacije, ako važi:

- Pravilo jedinstvenosti: ne postoje dve torke u relaciji R koje imaju iste vrednosti za X, i
- Pravilo minimalnosti: ne postoji pravi podskup skupa X koji zadovoljava pravilo jedinstvenosti.

Svaka relacija ima bar jednog kandidata za ključ (skup svih atributa ili neki njegov pravi podskup)

Vrste ključeva

- Primarni ključ - jedan od kandidata za ključ
- Alternativni ključevi - ostali kandidati
- Spoljašnji (strani) ključ - skup atributa jedne relacije (R_2) čije vrednosti treba da odgovaraju vrednostima nekog kandidata za ključ neke relacije (R_1)
- Superključ - nadskup kandidata za ključ; poseduje jedinstvenost ali ne i minimalnost

Ključevi - primer

- Relacija DOSIJE
  - primarni ključ: indeks
- Relacija ISPITNI_ROK
  - primarni ključ: (godina_roka, oznaka_roka)
- Relacija ISPIT
  - primarni ključ: (indeks, id_predmeta, godina_roka, oznaka_roka)
  - strani ključ: (godina_roka, oznaka_roka) na relaciju ISPITNI_ROK
  - strani ključ: (indeks) na relaciju DOSIJE

Referencijalni integritet

- Osnovna ideja očuvanja integriteta je da sve vrednosti u tabelama treba da budu usaglašene
- Primer: Student ne može da polaže ispit u ispitnom roku o kome nema podataka u tabeli ispitni_rok
- Problem **referencijalnog integriteta** - problem osiguravanja da baza podataka ne sadrži pogrešne spoljašnje ključeve
- **Referencijalna ograničenja** omogućavaju referencijalni integritet

Referencijalni integritet

- Relacija koja sadrži primarne ključeve (kandidate za ključ) se naziva *roditelj relacija*, a relacija koja sadrži spoljašnje ključeve koji referišu na roditelj relaciju se naziva *dete relacija*.
- Referencijalni integritet: baza podataka ne sme da sadrži neuparene vrednosti spoljašnjih ključeva

Referencijalni ciklus

- $T_n \to T_{n-1} \to T_{n-2} \to ... \to T_1 \to T_n$
- Roditelj tabela i dete tabela ne moraju da budu različite tabele
- Primer: Veza između zaposlenog i nadeđenog u tabeli Zaposleni

Ograničenja osnovnih tabela

- Definicija kandidata za ključeve
  - UNIQUE
  - PRIMARY KEY
  - NOT NULL
- Definicija spoljašnjih ključeva - FOREIGN KEY
- Definicija ograničenja: CHECK (uslovni izraz)

Pravljenje tabele

create table ime-tabele (
def-kolone [, def-kolone]+
[, def-primarni-ključ]?
[, def-strani-ključ]*
[, def-uslov-ogranicenja]*
))

Definicija kolone

Definicija kolone <def-kolone> ima oblik:

<ime-kolone> <tip-podataka>
[NOT NULL]
[[WITH] DEFAULT [<vrednost>]?]
GENERATED ALWAYS AS IDENTITY
[START WITH <vrenost>]?
[INCREMENT BY <vrednost>]?
[MINVALUE <vrednost>]?
[MAXVALUE <vrednost>]?

Tipovi podataka

- Celobrojni tipovi
  - SMALLINT - 16 bitova
  - INTEGER ili INT - 32 bita
  - BIGINT - 32 bita
- Realni tipovi
  - REAL - jednostruka tačnost
  - DOUBLE - dvostruka tačnost
  - DECIMAL(p, q) - zapis u fiksnom zarezu, gde je p ukupan broj cifara i najveća vrednost za p je 31, a q broj cifara desno od zareza

Niske

- Jednobajtne
  - CHAR(n) ili CHARACTER(n) - niske fiksne dužine, n od 1 do 255
  - VARCHAR(n) - niske promenljive dužine, n od 1 do 32672
  - CLOB(nc) - niske promenjive dužine do n jedinica c (CHAR LARGE OBJECT) gde c može biti
    - K
    - M
    - G (n može biti 1-2)
- Dvobajtne
  - GRAPHIC(n) - niske fiksne dužine, n u opsegu 1 do 127
  - VARGRAPHIC(n) - niske promenljive dužine, n do 16336
  - DBCLOB(n) - velike niske, do 1GiB - 1

Vremenski tipovi

- TIME - vreme
- DATE - datum
- TIMESTAMP - datum i vreme

Korisnički definisani tipovi (KDT)

CREATE DISTINCT TYPE <ime-tipa> AS <izvorni-tip> [WITH COMPARISONS]?

- KDT imaju reprezentaciju kao njihov izvorni tip koji je jedan od ugrađenih tipova
- dostupni su operatori za poređenje dve vrednosti novog tipa, kao i dve funkcjije - za konverziju vrednosti novog tipa u izvorni i obrnuto

Primer: Napraviti korisnički definisan tip prosek nad tipom dec(7,2).

```
create distinct type
                prosek as dec(7,2)
                with comparisons
```

Primer: Napraviti korisnički definisan tip prosek nad tipom dec(7,2).

može

```
select *
from dosije
where prosek(8.5)>prosek(7.5)
```

ne može

```
select *
from dosije
where prosek(8.5)>7.5
```

Primer: Napraviti korisnički definisan tip država čiji je izvorni tip varchar(30).

```
create distinct type
                drzava as varchar(30)
                with comparisons
```

Primer: Napraviti korisnički definisan tip char10 ciji je izvorni tip varchar(10).

```
create distinct type char10 as char(10)
```

Definicija primarnog ključa

[CONSTRAINT <ime>]? PRIMARY KEY (<ime-kolone> [, <ime-kolone>]+)

- kolone primarnog ključa moraju biti definisane sa not null opcijom
- jedan primarni ključ

Definicija stranog ključa

[CONSTRAINT <ime>]? FOREIGN KEY (<ime-kolone> [, <ime-kolone>]*)
REFERENCES <bazna-tabela> [ON DELETE <akcija>]?

- akcije
  - NO ACTION
  - RESTRICT
  - SET NULL
  - CASCADE

Uslov ograničenja

CONSTRAINT <ime> CHECK (<uslov>)

- uslov je logički izraz u kome mogu učestvovati kolone tabele, konstante i izrazi nad njima

Primer: Napraviti tabelu student u shemi stud.

```
create table stud.student (
       indeks            integer     not null,
       korisnik          char10      not null,
       ime               char(10)    not null with default user,
       prezime           varchar(15) not null,
       god_rodjenja      smallint          ,
       mesto_rodjenja    varchar(20)       ,
       tekuci_prosek     prosek      not null with default prosek(0),
       drzava_rodjenja   drzava              with default 'Srbija',
       primary key       (indeks)            ,
       constraint god_indeksa check  (indeks/10000>=2010)
)
```

Promena bazne tabele

ALTER TABLE <ime-tabele>
[ADD <def-kolone>]*
[ADD def-prim-ključa]?
[ADD def-str-ključa]?
[ADD uslov-ograničenja]*
[DROP PRIMARY KEY]?
[DROP FOREIGN KEY<ime-ograničenja>]*
[DROP CHECK<ime-ograničenja>]*
[DROP CONSTRAINT<ime-ograničenja>]*
[ALTER COLUMN <ime-kolone> SET DATA TYPE <tip>]*
[ALTER COLUMN <ime-kolone> SET DEFAULT <vrednost>]*
[ALTER COLUMN <ime-kolone> DROP DEFAULT <tip>]*

Promena bazne tabele

Problem:
Operation not allowed for reason code "7"on table <ime-tabele>... SQLCODE=-668, SQLSTATE=57007, DRIVER=3.69.56

db2 reorg <ime-tabele>

Primer: Tabeli stud.student dodati ograničenje da student mora biti rođen posle 1950. godine i postaviti da je podrazumevana vrednost kolone korisnik id korisnika koji izvršava naredbu.

```
alter table stud.student
add constraint starost check (god_rodjenja>1950)
alter column korisnik set default user
```

Primer: Uneti podatke u tabelu stud.student.

```
insert into stud.student(indeks,ime,prezime,god_rodjenja,mesto_rodjenja)
values
(20100021, 'Milos'   , 'Peric'      , 1992, 'Beograd' ),
(20100022, 'Marijana', 'Savkovic'   , 1993, 'Kraljevo'),
(20100023, 'Sanja'   , 'Terzic'     , 1991, 'Beograd' ),
(20100024, 'Nikola'  , 'Vukovic'    , 1992,  null     ),
(20100026, 'Zorica'  , 'Miladinovic', 1993, 'Vranje'  ),
(20100027, 'Milena'  , 'Stankovic'  , null,  null     )
```

Indeksi

CREATE [UNIQUE] INDEX <ime-indeksa> ON <tabela> (
<ime-kolone> [<poredak>]? [, <ime-kolone> [<poredak>]?]*)

- mehanizam koji ubrzava pristup redovima tabele
- opcija UNIQUE- dva reda u tabeli nad kojom se pravi indeks ne mogu da imaju iste vrednosti nad indeksnom kolonom / kombinacije vrednosti nad ideksnim kolonama

Primer: Napraviti indeks koji obezbeđuje da ne mogu da postoje dva studenta sa istim imenom.

```
create unique index studime on stud.student(ime)
```

pri pokušaju izvršavanja naredne naredbe javlja se greška: već postoji student Marijana

```
insert into stud.student(indeks,ime,prezime,god_rodjenja,mesto_rodjenja)
values (20100025, 'Marijana', 'Savkovic', 1991, 'Kraljevo')
```

Primer: Napraviti KDT datum i vreme.

```
create distinct type datum as date with comparisons;
create distinct type vreme as time with comparisons;
```

Primer: U shemi stud napraviti tabelu ispit koja pored informacija koje ima tabela ispit sadrži vreme ispita i redni broj prijave.

```
create table  stud.ispit (
       indeks          integer      not null                          ,
       id_predmeta     integer      not null                          ,
       godina_roka     smallint     not null                          ,
       oznaka_roka     char(5)      not null                          ,
       ocena           smallint     not null with default 5           ,
       datum_ispita    datum                                          ,
       vreme_ispita    vreme                                          ,
       rbr_prijave     integer      not null generated always
                                    as identity (start with 1)        ,
       primary key (indeks, id_predmeta, godina_roka,
                    oznaka_roka,rbr_prijave),
       foreign key (godina_roka, oznaka_roka) references ispitni_rok,
       constraint indeks_ref foreign key (indeks) references dosije ,
       foreign key (id_predmeta)  references predmet on delete cascade)
```

Primer: Unos u tabelu stud.ispit.

```
insert into stud.ispit (indeks, id_predmeta, godina_roka, oznaka_roka,
                        ocena, datum_ispita, vreme_ispita)
select indeks, id_predmeta, godina_roka, oznaka_roka,
                        ocena, datum_ispita, time('9.00.00')
from ispit
```

Primer: Napraviti tabelu nalik_na_dosije koja ima sve kolone kao i tabela dosije.

```
create table nalik_na_dosije like dosije
```

Primer: Napraviti korisnički definisan tip stipendija nad tipom decimal(7,2) i funkciju za sabiranje dve vrednosti tog tipa.

```
create distinct type stipendija as dec(7,2) with comparisons;

create function "+" (stipendija,stipendija)
returns stipendija
source sysibm."+" (dec(7,2),dec(7,2));
```

Primer: Napraviti tabelu stipendije.

```
create table stipendije  (
 indeks       integer  not null ,
 jmbg         char(15) not null ,
 ovajmesec    stipendija        ,
 proslimesec  stipendija        ,
 narednimesec stipendija        ,
 primary key  (indeks)
)
```

Primer: Napraviti korisnički definisan tip stipendija nad tipom decimal(7,2) i funkciju za sabiranje dve vrednosti tog tipa.

Da li može

```
select indeks, ovajmesec + narednimesec+ proslimesec
from   stipendije
```

Da li može

```
select indeks, ovajmesec + narednimesec - proslimesec
from   stipendije
```

Okidači

Okidač (eng. trigger) je niz akcija koje su pridružene određenim događajima, i koje se izvršavaju svaki put kada se takav događaj dogodi.

Okidači

CREATE TRIGGER <ime-okidaca>
<vreme> <radnja> ON <tabela>
[REFERENCING [OLD AS <ime-za-red-pre-promene>]? [NEW AS
<ime-za-red-posle-promene>]?]?
FOR EACH ROW
WHEN(<uslov>)
[BEGIN ATOMIC]?
<akcija1>;
...
<akcijaN>;
[END]?

Okidači

- *vreme*: BEFORE ili AFTER
- *radnja*: INSERT, DELETE, UPDATE [OF lista-kolona]?
- *akcija*: prijava greške, izmena vrednosti koja će biti uneta, menjanje druge tabele ...

Primer: Napisati okidač koji sprečava unos podataka u tabelu ispit ako na novom ispitu ocena za 3 veća od prosečne vrednosti svih prethodnih ocena za isti predmet u istom ispitnom roku za koji se unose ocene.

```
create trigger proveratriger
before insert  on stud.ispit
referencing new as nova
for each row
 when (nova.ocena >
        (select avg(ocena)+3
         from   ispit
         where  id_predmeta=nova.id_predmeta
         and    godina_roka=nova.godina_roka
         and    oznaka_roka=nova.oznaka_roka
        )
     )
 signal sqlstate'70123' ('Proveriti ocenu da li je dobro uneta');
```

Primer: Napisati okidač koji sprečava unos podataka u tabelu ispit ako na novom ispitu ocena za 3 veća od prosečne vrednosti svih prethodnih ocena za isti predmet u istom ispitnom roku za koji se unose ocene.

Pri pokušaju unosa

```
insert into stud.ispit( indeks, id_predmeta, godina_roka, oznaka_roka,
                        ocena, datum_ispita,vreme_ispita)
values
(20140026, 1021, 2015, 'apr', 18,current_date,current_time);
```

javlja se greška:
Application raised error or warning with diagnostic text: "Proveriti ocenu da li je dobro uneta".. SQLCODE=-438, SQLSTATE=70123, DRIVER=3.69.56

Opis sql greške ili stanja

- db2 ? <broj> - vraća odgovarajuću poruku o SQLSTATE
- db2 ? <sqlcode> - vraća odgovarajuću poruku o SQLCODE
- npr. db2 ? sql-438

Primer: Napisati triger koji u slučaju postavljanja šifre predmeta na null postavlja na null i naziv tog predmeta, a ocene u ispitima iz tog predmeta menja na negativnu vrednost.

Potrebne izmene nad tabelom predmet

```
alter table predmet
  alter column sifra drop not null
  alter column naziv drop not null
```

```
select * from ispit where  id_predmeta >3000;
```

Ako se javi greška: Operation not allowed for reason code "7"on table ŠTUDENT.PREDMET".. SQLCODE=-668, SQLSTATE=57007, DRIVER=3.69.56
uraditi: db2 reorg table predmet
pa ponoviti upit

Primer: Napisati triger koji u slučaju postavljanja šifre predmeta na null postavlja na null i naziv tog predmeta, a ocene u ispitima iz tog predmeta menja na negativnu vrednost.

```
create trigger sifratriger
before update of sifra on predmet
referencing new as nova
for each row
when (nova.sifra is null)
      set nova.naziv = null;
```

```
create trigger sifratriger_ispit
after update of sifra on predmet
referencing new as nova
for each row
when (nova.sifra is null)
  UPDATE ispit set ocena=-ocena
  where  id_predmeta=nova.id_predmeta;
```

Primer: Napisati triger koji u slučaju postavljanja šifre predmeta na null postavlja na null i naziv tog predmeta, a ocene u ispitima iz tog predmeta menja na negativnu vrednost.

```
update predmet
set    sifra=null
where  id_predmeta >3000
```