Računarstvo i informatika

Katedra za računarstvo
Elektronski fakultet u Nišu

Baze podataka (Računske vežbe)

SQL naredbe za ažuriranje podataka

Letnji semestar 2017/2018

Sadržaj

- Naredbe za ažuriranje podataka
- Dodavanje novih podataka
- Modifikacija postojećih podataka
- Brisanje podataka

Naredbe za ažuriranje podataka

- Naredba SELECT omogućava pretraživanje i pribavljanje podataka iz relacione baze podataka.
- U ovoj lekciji ćemo obraditi ostatak DML naredbi odnosno obradićemo naredbe koje omogućavaju modifikaciju podataka u relacionoj bazi podataka.
- Postoje tri moguće operacije za modifikovanje podataka:
  1. **Dodavanje novih podataka** (dodavanje novih vrsta u tabelu)
  2. **Modifikacija podataka** (izmena vrednosti kolona u postojećim vrstama tabele)
  3. **Brisanje podataka** (brisanje vrsta iz tabele)

Naredbe za ažuriranje podataka

- Svi primeri u nastavku će biti razmatrani za slučaj tabele PROJEKAT.
- U nastavku je data CREATE TABLE naredba koja je iskorišćena za kreiranje ove tabele kao i test podaci.

```
CREATE TABLE PROJEKAT
(
    NAZIV VARCHAR(25) NOT NULL,
    LOKPR VARCHAR(15) DEFAULT 'Niš',
    BROJPR NUMBER(3),
    BRS NUMBER(2) NOT NULL,
    CONSTRAINT PROJEKATPK PRIMARY KEY (BROJPR),
    CONSTRAINT NadlezanFK FOREIGN KEY (BRS)
     REFERENCES SEKTOR(SBROJ)
);
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4

Dodavanje novih podataka

- Za dodavanje podataka u tabelu koristi se SQL naredba INSERT...INTO.
- Naredba INSERT...INTO dodaje nove vrste u tabelu relacione baze podataka.
- Vrednosti kolona se definišu zadavanjem vrednosti u obliku konstanti ili korišćenjem rezultata SQL upita.
- U zavisnosti od načina zadavanja vrednosti kolona postoje različiti oblici INSERT..INTO naredbe.

Dodavanje novih podataka

- U nastavku je dat oblik INSERT..INTO naredbe koji se koristi u situacijama kada se vrednosti kolona zadaju korišćenjem konstanti.

INSERT INTO <ime_tabele>

[(<ime_kolone1> [,<ime_kolone2>]...)]

VALUES (<vrednost_kolone1> [{, <vrednost_kolone2>}...]) ;

- Iz definicije možemo zaključiti da je lista kolona opciona.
- Ukoliko je lista kolona izostavljena, u listi vrednosti kolona moraju se navesti vrednosti za svaku kolonu koja postoji u tabeli u koju se dodaje nova vrsta. U tom slučaju **lista vrednosti kolona mora da odgovara redosledu kojim su kolone navedene prilikom kreiranja tabele** (u CREATE TABLE naredbi).
- Takođe, vrednosti kolona, **moraju biti kompatibilne po tipu sa tipovima podataka koji su za kolone navedeni prilikom kreiranja tabele**.

Dodavanje novih podataka

- U nastavku je data SQL naredba kojom se dodaje nova vrsta u tabelu PROJEKAT.
- Dodaju se informacije o projektu čiji je broj 100, zove se ProizvodA, lociran je u Prokuplju i za njega je zadužen sektor čiji je broj 5.

```
INSERT INTO PROJEKAT
VALUES ('ProizvodA', 'Prokuplje', 100, 5);
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Prokuplje	100	5

Dodavanje novih podataka

- Ukoliko se u INSERT...INTO naredbi zadaje lista kolona, moguće je promeniti redosled zadavanja kolona u odnosu na onaj koji je specificiran u CREATE TABLE naredbi.

```
INSERT INTO PROJEKAT
(NAZIV, BROJPR, BRS, LOKPR)
VALUES ('ProizvodB', 101, 4, 'Svrljig');
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Prokuplje	100	5
ProizvodB	Svrljig	101	4

Dodavanje novih podataka

- Prilikom dodavanja nove vrste u tabelu biće proverena sva ograničenja koja su definisana nad tabelom: tip podataka, dužina polja, primarni ključ, NOT NULL, CHECK... **Ukoliko bar jedno ograničenje nije zadovoljeno DBMS će prijaviti poruku o grešci i neće izvršiti naredbu.**
- Lista kolona u pojedinim slučajevima ne mora biti kompletna.
- Iz liste se mogu izostaviti kolone kod kojih nije definisano NOT NULL ograničenje i kolone koje imaju definisano DEFAULT ograničenje.
- Za izostavljene kolone se upisuje podrazumevana vrednost definisana DEFAULT ograničenjem ili se upisuje NULL vrednost ukoliko DEFAULT ograničenje ne postoji.

Dodavanje novih podataka

- U narednom primeru iz liste kolona je izbačena kolona Lokacija.
- Pošto je u tabeli za tu kolona definisano DEFAULT ograničenje, prilikom dodavanja nove vrste u tu kolonu će biti upisana vrednost 'Niš'.
- Da DEFAULT ograničenje ne postoji u kolonu Lokacija bi bila upisana vrednost NULL.

```
INSERT INTO PROJEKAT
(NAZIV, BROJPR, BRS)
VALUES ('ProizvodC', 6, 5);
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Prokuplje	100	5
ProizvodC	Niš	102	5
ProizvodB	Svrljig	101	4

Dodavanje novih podataka

- Ukoliko se vrednosti kolona zadaju korišćenjem upita, naredba INSERT...INTO ima nešto drugačiji oblik.

INSERT INTO <ime_tabele>
[(<ime_kolone1> [,<ime_kolone2>]...)]
<upit>;

- U ovom slučaju za listu kolona važe ista pravila kao i u prethodnim situacijama. Vrednosti kolona se sada ne zadaju kako konstante već se zadaju kao rezultujuća tabela nekog upita. Redosled i tip kolona u rezultujućoj tabeli upita mora da odgovara redosledu i tipu kolona u listi.

Dodavanje novih podataka

- U nastavku je dat primer INSERT...INTO naredbe koja koristi SQL upit da bi u fiktivnu tabelu PROJEKAT_NEW dodala podatke iz tabele PROJEKAT.

```
CREATE TABLE PROJEKAT_NEW
(
   BROJ NUMBER(3),
   NAZIV VARCHAR(25) NOT NULL,
   LOKACIJA VARCHAR(15)
);

INSERT INTO PROJEKAT_NEW
(BROJ, NAZIV, LOKACIJA)
SELECT BROJPR, NAZIV, LOKPR
FROM PROJEKAT;
```

Dodavanje novih podataka

- Kreiranje nove tabele na osnovu upita koji vraća naredba SELECT.
- Kreirana tabela **nema ograničenja**.

```
CREATE TABLE SEK_INFO
AS SELECT NAZIV, COUNT(*), AVG(PLATA)
    FROM SEKTOR, RADNIK
   WHERE BRSEK = SBROJ
    GROUP BY NAZIV;
```

NAZIV	BROJ_RADNIKA	PROSECNA_PLATA
Administracija	3	31000
Uprava	1	55000
Razvoj	4	32750

Modifikacija podataka

- Za modifikaciju podataka koristi se SQL naredba UPDATE...SET.
- Osnovni oblik ove komande je dat unastavku:

UPDATE <ime_tabele>

SET <ime_kolone> = <izraz> [,<ime_kolone> = <izraz>...]

[WHERE <uslov>];

- SET definiše u kojoj se koloni menja vrednost definisana zadatim izrazom. **Izraz može biti konstantna vrednost, vrednost nekog izraza ili vrednost koju vraća ugnježdeni SQL upit**.
- Ako se navede WHERE klauzula, ažuriranje se vrši samo za kolone koje ispunjavaju vrednost nekog navedenog uslova.

Modifikacija podataka

- U nastavku je dat SQL upit koji lokacije projekta čiji je broj 101 menja na vrednost 'Beograd'.

```
UPDATE PROJEKAT
SET LOKPR = 'Beograd'
WHERE BROJPR = 101;
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Prokuplje	100	5
ProizvodC	Niš	102	5
ProizvodB	Beograd	101	4

- Treba biti jako oprezan prilikom korišćenja UPDATE...SET naredbe. **Ukoliko se u prethodnom primeru izostavi where klauzula ili uslov nije dobro definisan, lako možemo da izmenimo i vrste koje nismo želeli da menjamo, odnosno da izgubimo neke dragocene podatke.**

Modifikacija podataka

- Korišćenjem UPDATE...SET naredbe moguće je istovremeno menjati vrednosti većeg broja kolona.
- U sledećem primeru svi projekti koji u nazivu sadrže reč Proizvod se premeštaju u Niš u nadležnost sektora broj 4.

```
UPDATE PROJEKAT
SET LOKPR = 'Niš', BRS = 4
WHERE NAZIV LIKE '%Proizvod%';
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	4
ProizvodY	Niš	2	4
ProizvodZ	Niš	3	4
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Niš	100	4
ProizvodC	Niš	102	4
ProizvodB	Niš	101	4

Modifikacija podataka

- U nastavku je dat SQL upit kojim se projekat broj tri prbacuje u Beograd u naležnost sektora broj 5.

```
UPDATE PROJEKAT
SET LOKPR = 'Beograd', BRS = 5
WHERE BROJPR= 3;
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	4
ProizvodY	Niš	2	4
ProizvodZ	Beograd	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	4
Godišnji izveštaj	Niš	30	4
ProizvodA	Niš	100	4
ProizvodC	Niš	102	4
ProizvodB	Niš	101	4

Modifikacija podataka

- Projekti za koje je zadužen sektor broj 4 prelaze u nadležnost sektora Razvoj.

```
UPDATE PROJEKAT
SET BRS = (SELECT SBROJ
           FROM SEKTOR
           WHERE NAZIV ='Razvoj')
WHERE BRS = 4;
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Niš	2	5
ProizvodZ	Beograd	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	5
Godišnji izveštaj	Niš	30	5
ProizvodA	Niš	100	5
ProizvodC	Niš	102	5
ProizvodB	Niš	101	5

Brisanje podataka

- Za brisanje podataka iz relacione baze podataka koristi se naredba DELETE.
- U svom osnovnom obliku naredba DELETE ima sledeću sintaksu:

DELETE FROM <ime_tabele>
[WHERE <uslov>];

Brisanje podataka

- U nastavku je data SQL naredba koja iz tabele PROJEKAT briše podatke o svim projektima čiji se naziv završava slovom ‘A'.

```
DELETE
FROM PROJEKAT
WHERE NAZIV LIKE '%A';
```

NAZIV	LOKPR	BROJPR	BRS
ProizvodX	Niš	1	5
ProizvodY	Niš	2	5
ProizvodZ	Beograd	3	5
Reorganizacija	Niš	10	1
Informacioni sistem	Leskovac	20	5
Godišnji izveštaj	Niš	30	5
ProizvodC	Niš	102	5
ProizvodB	Niš	101	5

- Uslov koji navedete u WHERE definiše kriterijume za selekciju torki koje treba obrisati iz zadate tabele. **Ako se ne navede WHERE klauzula, naredba delete briše sve vrste iz tabele čije se ime navede u from klauzuli.**
- Zbog toga treba biti jako oprezan prilikom korišćenja naredbe DELETE. **Ukoliko izostavite uslov ili je uslov neadekvatno definisan može doći do trajnog brisanja podataka koje nismo želeli da obrišemo.**

Brisanje podataka

- U nastavku je data SQL naredba koja briše podatke o svim projektima.

```
DELETE
FROM PROJEKAT;
```

- U nastavku je data SQL naredba koja briše podatke o svim projektima za koje je zadužen sektor broj 1.

```
DELETE
FROM PROJEKAT
WHERE BRS = 1;
```

Brisanje podataka

- U nastavku je data SQL naredba koja briše podatke o svim projektima koji su locirani u Beogradu ili Novom Sadu i za njih je zadužen sektor Administracija.

```
DELETE FROM PROJEKAT
WHERE LOKPR IN ('Beograd', 'Novi Sad')
        AND BRS = (SELECT SBROJ
                   FROM SEKTOR
                   WHERE NAZIV = 'Administracija');
```

Brisanje podataka

- U nastavku je data SQL naredba koja briše podatke o svim projektima na kojima nije angažovan nijedan radnik.

```
DELETE FROM PROJEKAT
WHERE BROJPR NOT IN (SELECT BRPR
                     FROM RADI_NA);
```

```
DELETE FROM PROJEKAT
WHERE NOT EXISTS (SELECT 0
                  FROM RADI_NA
                  WHERE RADI_NA.BRPR = PROJEKAT.BROJPR);
```

Brisanje podataka

- Prilikom korišćenja naredbe DELETE treba voditi računa i o efektima koji mogu da se jave kao posledica postojanja ograničenja stranog ključa.
- Možemo za primer uzmemo tabele RADNIK i SEKTOR iz baze podataka PREDUZEĆE, i da pokušamo da obrišemo podatke o sektoru čiji je naziv 'Administracija'.
- Pretpostavićemo da u tabeli RADNIK postoji strani ključ nad kolonom BRSEK koji referencira kolonu SBROJ u tabeli SEKTOR.

```
DELETE
FROM SEKTOR
WHERE Naziv = 'Administracija';
```

Brisanje podataka

- Izvršavanjem ove naredbe moguće je da se javi jedna od naredne tri situacije:
  1. Ukoliko u tabeli radnik **ne postoje** radnici koji rade u sektoru 'Administracija', iz tabele SEKTOR **biće obrisani podaci** o sektoru čiji je naziv 'Administracija'.
  2. Ukoliko u tabeli radnik **postoje** radnici koji rade u sektoru 'Administracija', DBMS će **prijaviti grešku** i **neće izvršiti brisanje**. Brisanje u ovom slučaju nije moguće jer bi u tabeli RADNIK dobili vrste koje referenciraju nepostojeći sektor čime bi bio narušen referencijalni integritet.
  3. Ukoliko u tabeli radnik **postoje** radnici koji rade u sektoru 'Administracija' i strani ključ je definisan korišćenjem opcije **ON DELETE CASCADE** , iz tabele SEKTOR **biće obrisani podaci** o sektoru čiji je naziv 'Administracija' ali će i iz tabele RADNIK **biti obrisani podaci o radnicima** koji rade u sektoru 'Administracija'. Za razliku od prethodnog slučaja gde će DBMS sprečiti brisanje da ne bi došlo do narušavanja referencijalnog integriteta, **u ovom slučaju DBMS automatski briše sve podatke kod kojih može doći do narušavanja referencijalnog integriteta**. Ovo je još jedan razlog više zbog čega treba biti jako oprezan prilikom korišćenja DELETE naredbe.