SPAJANJE RELACIJA

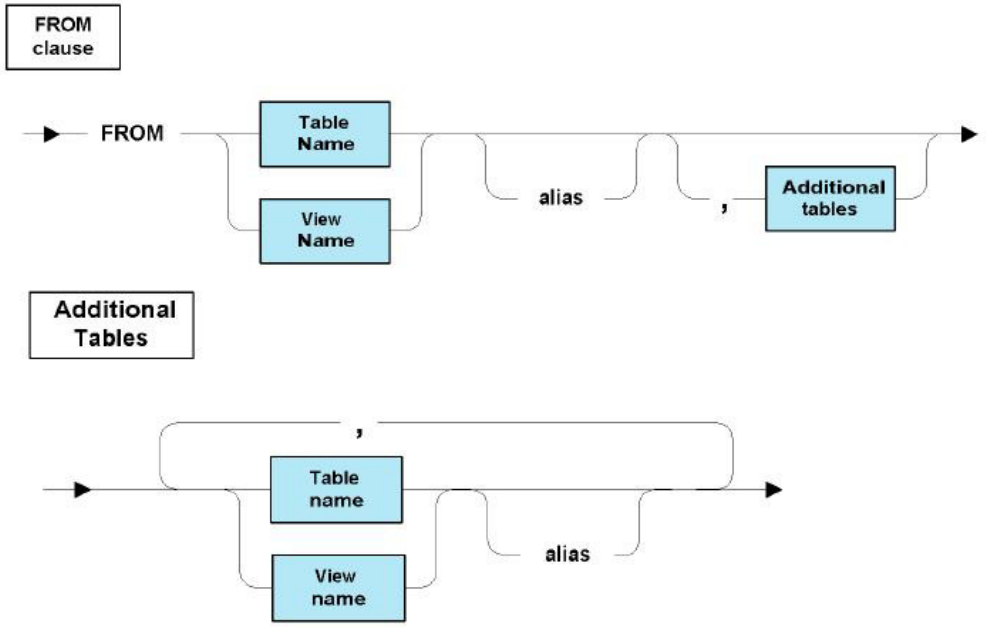
IV auditorne vježbe

Teme

- Spajanje relacija
  - Dekartov proizvod relacija
  - Prirodno spajanje relacija
  - Način spajanja relacija u slučaju postojanja "paralelne" veze
  - Način spajanja relacija u slučaju postojanja "refleksivne" veze
  - Spajanje relacija uz uslov

Spajanje relacija

- U dijelu SQL naredbe koji se naziva FROM *Clause* mogu se navesti relacije (jedna ili više njih) čiji se podaci koriste u upitu.



- U većini slučajeva se više nego jedna relacija navodi u slučajevima kada nad relacijama treba obaviti prirodno spajanje ili spajanje uz uslov (obavljanje Dekartovog proizvoda relacija ima rijetku primjenu u praksi).

Dekartov proizvod relacija

STUDENT

imeStud	pbr
David	75000
Linda	75000
Sandy	71000

MJESTO

pbr	nazMjesto
75000	Tuzla
71000	Sarajevo

```
SELECT * FROM student, mjesto
```

imeStud	pbr	pbr	nazMjesto
David	75000	75000	Tuzla
David	75000	71000	Sarajevo
Linda	75000	75000	Tuzla
Linda	75000	71000	Sarajevo
Sandy	71000	75000	Tuzla
Sandy	71000	71000	Sarajevo

- Zato što je u listi za selekciju naveden znak *, pojavljuju se svi atributi iz obje relacije.
- Rezultat upita je 3*2 n-torki
- Za veliki broj n-torki SUBP troši ogromne računarske resurse za obavljanje ovakvih (beskorisnih) upita

Spajanje relacija u MySQL

table_references:

escaped_table_reference [, ***escaped_table_reference***] ...

escaped_table_reference:

table_reference

| { OJ ***table_reference*** }

table_reference:

table_factor

| ***join_table***

table_factor:

tbl_name [PARTITION (***partition_names***)] [[AS] ***alias***] [***index_hint_list***]

| ***table_subquery*** [AS] ***alias***

| (***table_references***)

join_table:

table_reference [INNER | CROSS] JOIN ***table_factor*** [***join_condition***]

| ***table_reference*** STRAIGHT_JOIN ***table_factor***

| ***table_reference*** STRAIGHT_JOIN ***table_factor*** ON ***conditional_expr***

| ***table_reference*** {LEFT|RIGHT} [OUTER] JOIN ***table_reference join_condition***

| ***table_reference*** NATURAL [{LEFT|RIGHT} [OUTER]] JOIN ***table_factor***

join_condition:

ON ***conditional_expr***

| USING (***column_list***)

index_hint_list:

index_hint [, ***index_hint***] ...

index_hint:

USE {INDEX|KEY} [FOR {JOIN|ORDER BY|GROUP BY}] ([***index_list***])

| IGNORE {INDEX|KEY} [FOR {JOIN|ORDER BY|GROUP BY}] (***index_list***)

| FORCE {INDEX|KEY} [FOR {JOIN|ORDER BY|GROUP BY}] (***index_list***) ***index_list***: ***index_name*** [, ***index_name***] ...

Spajanje relacija u MySQL

- U MySQL, JOIN, CROSS JOIN i INNER JOIN su sintaksno ekvivalentni (mogu zamijeniti jedan drugog)
- U standardnom SQL nisu ekvivalentni (INNER JOIN se koristi sa ON klauzulom, a CROSS JOIN u ostalim slučajevima)
- INNER JOIN i (,) zarez su semantički ekvivalentni u odsutnosti uslova spajanja: oba proizvode Dekartov proizvod tabela
- Međutim, operator zarez (,) ima manji prioritet od INNER JOIN, CROSS JOIN, LEFT JOIN itd.
- Ukoliko se miješa spajanje sa zarezom sa drugim tipovima spajanja kada postoji uslov spajanja, može se pojaviti greška oblika **Unknown column '*col_name*' in 'on clause'**
- *Conditional_expr* korišten u ON kaluzili je bilo koji uslovni izraz u obliku koji može biti korišten u WHERE klauzuli
- Generalno, ON klauzula se treba koristiti za uslovi koji specificiraju spajanje tabela, a WHERE klauzula za uslove dohvata n-torki

Spajanje relacija u MySQL

- Ukoliko ne postoje pripadajuće n-torke u desnoj tabeli ON ili USING dijela u LEFT JOIN, pojaviće se n-torka desne tabele sa svim NULL vrijednostima atributa
- USING(column_list) klauzula imenuje listu atributa koje moraju postojati u obje tabele
- NATURAL [LEFT] JOIN dvije tabele se definiše da bude semantički ekvivalentno INNER JOIN ili LEFT JOIN sa USING klauzulom koja imenuje atribute koji postoje u obje tabele
- RIGHT JOIN radi analogno LEFT JOIN
- STRAIGHT_JOIN je slično JOIN osim što se lijeva tabela uvijek čita prije desne tabele

Spajanje relacija u MySQL

```
SELECT * FROM student INNER JOIN mjesto
    ON student.pbr = mjesto.pbr;
```

imeStud	pbr	pbr	nazMjesto
David	75000	75000	Tuzla
Linda	75000	75000	Tuzla
Sandy	71000	71000	Sarajevo

```
SELECT * FROM student INNER JOIN mjesto USING(pbr);
```

pbr	imeStud	nazMjesto
75000	David	Tuzla
75000	Linda	Tuzla
71000	Sandy	Sarajevo

```
SELECT * FROM student NATURAL JOIN mjesto;
```

pbr	imeStud	nazMjesto
75000	David	Tuzla
75000	Linda	Tuzla
71000	Sandy	Sarajevo

```
INSERT INTO student VALUES ('Robert',72000);
SELECT * FROM student LEFT JOIN mjesto
    ON student.pbr = mjesto.pbr;
```

imeStud	pbr	pbr	nazMjesto
David	75000	75000	Tuzla
Linda	75000	75000	Tuzla
Sandy	71000	71000	Sarajevo
Robert	72000	NULL	NULL

Prirodno spajanje relacija

- Za ispravno obavljanje prirodnog spajanja nužno je navesti uslov spajanja (*Join*) koji se navodi u ON *Clause*. Bez tog uslova rezultat spajanja relacija će uvijek biti Dekartov proizvod.

```
SELECT student.*, mjesto.nazMjesto
FROM student INNER JOIN mjesto
     ON student.pbr = mjesto.pbr
```

imeStud	pbr	nazMjesto
David	75000	Tuzla
Linda	75000	Tuzla
Sandy	71000	Sarajevo

- Oni atributi koji se pojavljuju u obje relacije se u listi za selekciju navode samo jednom, s ciljem da se dobije rezultat koji odgovara strogoj definiciji prirodnog spajanja

Prirodno spajanje relacija

- Konceptualno, može se smatrati da se upiti sa spajanjem relacija obavljaju u tri faze
  - Određuje se Dekartov proizvod relacija
  - Obavlja se selekcija onih n-torki koje zadovoljavaju uslove spajanja
  - Ispisuju se vrijednosti atributa navedenih u listi za selekciju za sve n-torke iz 2. koraka

  Primjer: Dohvat imena studenata, naziva predmeta i ocjena svih studenata čije mjesto stanovanja ima naziv koji počinje slovom Z, a koji su na ispitu dobili ocjenu veću od 3

```
SELECT stud.imeStud, pred.nazPred, ispit.ocjena
FROM stud INNER JOIN mjesto
ON sud.pbrStan = mjesto.pbr
INNER JOIN ispit ON stud.mbrStud = ispit.mbrStud
INNER JOIN pred ON ispit.sifPred = pred.sifPred
     WHERE nazMjesto LIKE 'Z%'
     AND ispit.ocjena > 3
```

Spajanje sa izjednačavanjem (Equi-join)

- Equi-join je spajanje jedne ili više tabela gdje SQL upit koristi WHERE klauzulu i operatore poređenja kako bi se odredili atributi preko kojih se vrši spajanje
- Nedostatak upotrebe equi-join je taj da drugima može biti teško čitati i razumjeti svrhu SQL upita (jer WHERE klauzula postaje složenija)
- Prirodno spajanje relacija preko equi-join je:

```
SELECT student.*, mjesto.nazMjesto
  FROM student, mjesto
       WHERE student.pbr = mjesto.pbr
```

imeStud	pbr	nazMjesto
David	75000	Tuzla
Linda	75000	Tuzla
Sandy	71000	Sarajevo

Način spajanja relacija u slučaju postojanja "paralelne" veze

- Relacija student sada sadrži atribute kojima se opisuje mjesto stanovanja i mjesto rođenja.

STUDENT

imeStud	**pbrRod**	**pbrStan**
David	75000	75000
Linda	75000	71000
Sandy	71000	75000

MJESTO

pbr	**nazMjesto**
75000	Tuzla
71000	Sarajevo

- Ispis studenata u kojem se pojavljuje naziv mjesta rođenja lako se rješava na način kakav je opisan u prethodnom primjeru.

```
SELECT imeStud, pbrRod, nazMjesto FROM student INNER JOIN mjesto
      ON student.pbrRod = mjesto.pbr
```

- Upit postaje nešto složeniji kada se postavi zahtjev za ispisom liste studenata, ali tako da se uz svakog studenta istovremeno ispišu naziv njegovog mjesta rođenja i mjesta stanovanja.

Način spajanja relacija u slučaju postojanja "paralelne" veze

```
SELECT student.*, mjesto.nazMjesto nazMjestoStan,
       mjesto.nazMjesto nazMjestoRod
FROM student INNER JOIN mjesto
    ON pbrRod = mjesto.pbr
    AND pbrStan = mjesto.pbr
```

- Ne vrijedi - rezultat su samo one n-torke iz relacije student u kojima su mjesto rođenja i mjesto stanovanja jednaki.
- Trebalo bi koristiti dvije relacije mjestoRodjenja i mjestoStanovanja, koje ne postoje
- mjesto se mora u upitu pojaviti dva puta, ali jednom u ulozi mjesta rođenja, a jednom u ulozi mjesta stanovanja.
- Da bi se njene uloge razlikovale, obavezno je korištenje alternativnih ili zamjenskih naziva relacija (*ALIAS*).
- Alternativno ime se tada može koristiti u upitu kao da se radi o "pravoj" relaciji.

Način spajanja relacija u slučaju postojanja "paralelne" veze

```
SELECT student.*, mjestoStan.*, mjestoRod.*
  FROM student INNER JOIN mjesto mjestoRod
  ON student.pbrRod = mjestoRod.pbr
  INNER JOIN mjesto mjestoStan
  ON student.pbrStan = mjestoStan.pbr
```

imeStud	pbrRod	pbrStan	pbr	nazMjesto	pbr	nazMjesto
David	75000	75000	75000	Tuzla	75000	Tuzla
Linda	75000	71000	75000	Tuzla	71000	Sarajevo
Sandy	71000	75000	71000	Sarajevo	75000	Tuzla

Način spajanja relacija u slučaju postojanja "refleksivne" veze

- Kada u relaciji postoji refleksivna veza pojedine n-torke iz relacije povezane su sa drugim n-torkema iz iste relacije.
- U konkretnom primjeru, svaka organizacijska jedinica iz relacije orgjed povezana je sa svojom nadređenom organizacijskom jedinicom pomoću atributa sifNadorgjed.

ORGJED

SifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

- U slučajevima kada je potrebno uspoređivati pojedine organizacijske jedinice sa njihovim nadređenim ili podređenim organizacijskim jedinicama, potrebno je obaviti spajanje relacije "same sa sobom".
- Problem je sličan i slično se rješava kao u paralelnoj vezi.

Način spajanja relacija u slučaju postojanja "refleksivne" veze

- **Primjer:** Dohvatiti sve organizacijske jedinice i nazive njihovih nadređenih organizacijskih jedinica

```
SELECT orgjed.sifOrgjed, orgjed.nazOrgjed, orgjed.sifNadorgjed,
nadorgjed.nazOrgjed nadredjenaNaziv
FROM orgjed INNER JOIN orgjed nadOrgjed
     ON orgjed.sifNadorgjed = nadOrgjed.sifOrgjed
```

sifOrgjed	nazOrgjed	sifNadorgjed	nadredjenaNaziv
2	Odjel A 1	Uprava	
3	Odjel B 1	Uprava	
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

- Primjetite da se organizacijska jedinica "Uprava" nije pojavila u rezultatu, jer ne postoji n-torka čija je šifra jednaka NULL vrijednosti

Način spajanja relacija u slučaju postojanja "refleksivne" veze

- **Primjer:** Dohvatiti sve organizacijske jedinice i nazive njihovih podređenih organizacijskih jedinica

```
SELECT orgjed.sifOrgjed, orgjed.nazOrgjed, podorgjed.sifOrgjed
sifPodorgjed, podorgjed.nazOrgjed podredjenaNaziv
FROM orgjed INNER JOIN orgjed podOrgjed
     ON podorgjed.sifNadorgjed = orgjed.sifOrgjed
```

sifOrgjed	nazOrgjed	sifPodorgjed	podredjenaNaziv
1	Uprava	2	Odjel A
1	Uprava	3	Odjel B
2	Odjel A	4	Pododjel X
2	Odjel A	5	Pododjel Y
3	Odjel B	6	Pododjel Z

- Primjetite da se organizacijske jedinice koje nemaju podređene organizacijske jedinice nisu pojavile u rezultatu

Spajanje relacija uz uslov

- Slično prirodnom spajanju, obavlja se i spajanje relacije uz uslov.
- **Primjer:** Ispisuju se avioni i mogući letovi (s obzirom na dolet i udaljenost koju je potrebno preletjeti)

AVION

tip	dolet
B747	6000
DC9	3000
AIRBUS	1800

LET

brLeta	udaljenost
CA-825	7200
A-224	3000
CA-878	4200

```
SELECT * FROM avion INNER JOIN let ON avion.dolet >= udaljenost
```

tip	dolet	brLeta	udaljenost
B747	6000	A-224	3000
B747	6000	CA-878	4200
DC9	3000	A-224	3000

Primjeri upita

- Za svaki položeni ispit ispisati ime i prezime studenta, ime i prezime nastavnika, naziv predmeta i ocjenu koju je student dobio.

```
SELECT imeStud, prezStud, imeNastavnik,
prezNastavnik, nazPred, ocjena
    FROM ispit INNER JOIN stud
    ON ispit.mbrStud = stud.mbrStud
    INNER JOIN nastavnik
    ON ispit.sifNastavnik = nastavnik.sifNastavnik
    INNER JOIN pred ON ispit.sifPred = pred.sifPred
        WHERE ocjena > 1
```

Primjeri upita

- Ispisati imena i prezimena nastavnika koji su negativno ocijenili studente koji su rođeni u mjestu čiji naziv počinje slovom T, a završava samoglasnikom. Svaki takav nastavnik u listi treba da se pojavi samo jednom.

```
SELECT DISTINCT imeNastavnik, prezNastavnik
    FROM nastavnik INNER JOIN ispit
    ON ispit.sifNastavnik = nastavnik.sifNastavnik
    INNER JOIN stud ON ispit.mbrStud = stud.mbrStud
    INNER JOIN mjesto ON stud.pbrRod = mjesto.pbr
          WHERE ocjena = 1
          AND nazMjesto LIKE 'T%'
          AND nazMjesto RLIKE '[aeiou]$'
```

Primjeri upita

- Ispisati ime i prezime studenata, te nazive županija rođenja i stanovanja za sve studente rođene u maju mjesecu 1982 godine.

```
SELECT imeStud, prezStud, zupanijaR.nazZupanija,
zupanijaS.nazZupanija
    FROM stud INNER JOIN mjesto mjestoR
    ON stud.pbrRod = mjestoR.pbr
    INNER JOIN mjesto mjestoS
    ON stud.pbrStan = mjestoS.pbr
    INNER JOIN zupanija zupanijaR
    ON mjestoR.sifzupanija = zupanijaR.sifZupanija
    INNER JOIN zupanija zupanijaS
    ON mjestoS.sifZupanija = zupanijaS.sifZupanija
          WHERE MONTH(datRodStud) = 5
          AND YEAR(datRodStud) = 1982
```

Primjeri upita

- Ispisati ime i prezime nastavnika i naziv mjesta stanovanja, te naziv mjesta rođenja studenata, za sve nastavnike koji su ispitivali studente koji nisu rođeni u istom mjestu u kojem stanuje nastavnik, ali se mjesta nalaze u istoj županiji.

```
SELECT imeNastavnik, prezNastavnik,
mjestoN.nazMjesto, mjestoS.nazMjesto
     FROM ispit INNER JOIN nastavnik
     ON ispit.sifNastavnik = nastavnik.sifNastavnik
     INNER JOIN stud ON ispit.mbrStud = stud.mbrStud
     INNER JOIN mjesto mjestoN
     ON nastavnik.pbrStan = mjestoN.pbr
     INNER JOIN mjesto mjestoS
     ON stud.pbrRod = mjestoS.pbr
          WHERE mjestoS.pbr <> mjestoN.pbr
          AND mjestoS.sifZupanija = mjestoN.sifZupanija
```

Primjeri upita

- Ispisati ime i prezime nastavnika i koeficijent za platu za sve nastavnike zaposlene u organizacijskoj jedinici čija neposredno nadređena organizacijska jedinica u nazivu sadrži riječ "matike"

```
SELECT imeNastavnik, prezNastavnik, koef
FROM nastavnik INNER JOIN orgjed OJ
ON nastavnik.sifOrgjed = OJ.sifOrgjed
INNER JOIN orgjed NJ
ON OJ.sifNadOrgjed = NJ.sifOrgjed
    WHERE NJ.nazOrgjed LIKE '%matike%'
```

Primjeri upita

- Ispisati šifre i nazive nadređenih organizacionih jedinica na čijim podređenim organizacionim jedinicama se predaju predmeti na kojima je upisano više od 20 studenata i iz kojih je dodjeljena bar jedna negativna ocjena ove godine. Svaka takva organizaciona jedinica se u ispisu treba pojaviti samo jednom

```
SELECT DISTINCT NJ.sifOrgjed, NJ.nazOrgjed
FROM orgjed NJ INNER JOIN orgjed PJ ON
PJ.sifNadorgjed = NJ.sifOrgjed
INNER JOIN pred ON pred.sifOrgjed = PJ.sifOrgjed
INNER JOIN ispit ON pred.sifPred = ispit.sifPred
    WHERE pisanoStud > 20
    AND ocjena = 1
    AND YEAR(datIspit) = YEAR(CURRENT_DATE)
```