



Računarstvo i informatika

Katedra za računarstvo

Elektronski fakultet u Nišu

Baze podataka (Računske vežbe) **SQL SELECT naredba (3)**

Letnji semestar 2017/2018



Sadržaj

- Kombinovanje rezultata upita
- Ugnježdjeni upiti
- ROWNUM
- CONNECT BY



Kombinovanje rezultata upita

- Programski jezik SQL dozvoljava kombinovanje rezultata većeg broj SQL upita korišćenjem operacija za rad sa skupovima:
 - unija (UNION)
 - presek (INTERSECT)
 - razlika (MINUS)
- 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 imati kompatibilne tipove)



Kombinovanje rezultata upita

- Klauzula UNION kombinuje rezultate dva ili više upita u jednu rezultujuću tabelu.
- U rezultujuću tabelu ulaze svi slogovi iz svih rezultujućih tabela.
- Prilikom izvršavanja operacije UNION elimišu se duplikati.
- To znači da se slog koji se pojavljuje u više rezultujućih tabela, u rezultatu unije pojavljuje samo jednom.
- Zbog toga je potrebno voditi računa da rezultujuće tabele koje se spajaju moraju da uključuju attribute neophodne za jednoznačnu identifikaciju slogva.



Kombinovanje rezultata upita

- U nastavku je dat SQL upit koji vraća nazive sektora u kojima rade radnici koji se prezivaju Petrović i Jovanović.

Upit koji vraća podatke o sektorima u kojima rade radnici koji se prezivaju Petrović.

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Petrović';
```

NAZIV
Razvoj
Administracija

Upit koji vraća podatke o sektorima u kojima rade radnici koji se prezivaju Jovanović.

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';
```

NAZIV
Razvoj



Kombinovanje rezultata upita

- Kombinovanjem ova dva upita korišćenjem klauzule UNION dobija se traženi rezultat.

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Petrović'
```

```
UNION
```

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';
```

NAZIV
Administracija
Razvoj

- Sektor Razvoj koji se javlja kao rezultat i jednog i drugog upita u uniji se pojavljuje samo jednom.



Kombinovanje rezultata upita

- U nastavku su dati **pogrešno napisani upiti**, koji ne mogu da se kombinuju jer **nema slaganja po tipu kolona**.

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Petrović'
```

```
UNION
```

```
SELECT DISTINCT SBROJ
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';
```




Kombinovanje rezultata upita

- U nastavku su dati **pogrešno napisani upiti**, koji ne mogu da se kombinuju **jer nema slaganja po broju kolona.**

```
SELECT DISTINCT NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Petrović'
```

```
UNION
```

```
SELECT DISTINCT SBROJ, NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';
```




Kombinovanje rezultata upita

- U nastavku su dati **pogrešno napisani upiti**, koji ne mogu da se kombinuju jer **nema slaganja po redosledu i tipu kolona**.

```
SELECT DISTINCT NAZIV, SBROJ
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Petrović'
```

```
UNION
```

```
SELECT DISTINCT SBROJ, NAZIV
```

```
FROM SEKTOR, RADNIK
```

```
WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';
```



Kombinovanje rezultata upita

- U nastavku je dat SQL upit koji korišćenjem klauzule INTERSECT određuje podatke o radnicima koji rade u sektoru Administracija i koji imaju platu veću od 40000.

Upit koji vraća podatke o radnicima koji rade u sektoru Administracija.

```
SELECT MATBR, LIME, PREZIME  
FROM SEKTOR, RADNIK  
WHERE SBROJ = BRSEK AND NAZIV = 'Administracija'
```

MATBR	LIME	PREZIME
999887777	Valentina	Kovačević
987654321	Aleksandra	Petrović
987987987	Stanko	Manojlović

Upit koji vraća podatke o radnicima koji imaju platu veću od 40000.

```
SELECT MATBR, LIME, PREZIME  
FROM RADNIK  
WHERE PLATA > 40000;
```

MATBR	LIME	PREZIME
987654321	Aleksandra	Petrović
888665555	Jovan	Obradović



Kombinovanje rezultata upita

- Traženi rezultat se dobija kombinovanjem ova dva upita korišćenjem klauzule INTERSECT.

SELECT MATBR, LIME, PREZIME

FROM SEKTOR, RADNIK

WHERE SBROJ = BRSEK AND NAZIV = 'Administracija'

INTERSECT

SELECT MATBR, LIME, PREZIME

FROM RADNIK

WHERE PLATA > 40000;

- U rezultatu se nalaze samo slogovi koji se pojavljuju u rezultujućim tabelama oba upita.

MATBR	LIME	PREZIME
987654321	Aleksandra	Petrović



Kombinovanje rezultata upita

- U nastavku je dat SQL upit koji podake o radnicima koji rade u sektoru Administracija a nemaju platu veću od 40000.
- Upit iz prethodnog primera treba modifikovati tako da se umesto klauzule INTERSECT koristi klauzula MINUS.

```
SELECT MATBR, LIME, PREZIME  
FROM SEKTOR, RADNIK  
WHERE SBROJ = BRSEK AND NAZIV = 'Administracija'  
MINUS
```

```
SELECT MATBR, LIME, PREZIME  
FROM RADNIK  
WHERE PLATA > 40000;
```

MATBR	LIME	PREZIME
987987987	Stanko	Manojlović
999887777	Valentina	Kovačević

- Rezultat operacije MINUS uključuje samo slogove koji se pojavljuju u prvoj rezultujućoj tabeli ali ne i u drugoj.



Ugnježdjeni upiti

- U nastavku je dat SQL upit koji prikazuje podatke o radnicima kojima nakon uvećanja od 5000 plata prelazi 40000.

```
SELECT MATBR, LIME, PREZIME, PLATA + 5000 AS UVECANA_PLATA  
FROM RADNIK
```

MATBR	LIME	PREZIME	UVECANA_PLATA
123456789	Marko	Petrović	35000
333445555	Sima	Todorović	45000
999887777	Valentina	Kovačević	30000
987654321	Aleksandra	Petrović	48000
666884444	Velibor	Jovanović	41000
453453453	Jelena	Janković	30000
987987987	Stanko	Manojlović	30000
888665555	Jovan	Obradović	60000

```
SELECT MATBR, LIME, PREZIME, UVECANA_PLATA  
FROM (SELECT MATBR, LIME, PREZIME, PLATA + 5000 AS UVECANA_PLATA  
      FROM RADNIK)  
WHERE UVECANA_PLATA > 40000;
```

MATBR	LIME	PREZIME	UVECANA_PLATA
333445555	Sima	Todorović	45000
987654321	Aleksandra	Petrović	48000
666884444	Velibor	Jovanović	41000
888665555	Jovan	Obradović	60000



Ugnježdjeni upiti

- U nastavku je dat upit koji prikazuje imena i prezimena radnika koji se nalaze na položaju rukovodioca sektora.

```
SELECT MATBR, LIME, PREZIME
```

```
FROM RADNIK
```

```
WHERE MATBR IN (SELECT MATBRR FROM SEKTOR);
```

MATBR	LIME	PREZIME
333445555	Sima	Todorović
888665555	Jovan	Obradović
987654321	Aleksandra	Petrović

- Identičan upit bi izgledao ovako (razlika je u tome što se skup vrednosti generiše dinamički).

```
SELECT MATBR, LIME, PREZIME
```

```
FROM RADNIK
```

```
WHERE MATBR IN (333445555, 888665555, 987654321);
```



Ugnježdjeni upiti

- U nastavku je dat upit koji korišćenjem klauzule EXISTS određuje podatke o radnicima koji imaju članove porodice.
- Klauzula EXISTS proverava da li ugnježdjeni upit vraća rezultat koji sadrži slogove ili ne.

SELECT MATBR, LIME, PREZIME

FROM RADNIK

WHERE EXISTS (SELECT IME FROM CLAN_PORODICE);

MATBR	LIME	PREZIME
123456789	Marko	Petrović
333445555	Sima	Todorović
999887777	Valentina	Kovačević
987654321	Aleksandra	Petrović
666884444	Velibor	Jovanović
453453453	Jelena	Janković
987987987	Stanko	Manojlović
888665555	Jovan	Obradović

- **PRETHODNI UPIT NIJE DOBAR** zato što vraća podatke o svim radnicima bez obzira da li imaju članove porodice ili nemaju.
- Problem je u ugnježđenom upitu. Ugnježdjeni upit treba da se izvrši za svakog radnika i proveri da li on ima članove porodice.
- U ovom slučaju ugnježdjeni upit uvek vraća isti rezultat (sve članove porodice) bez obzira da li radnik ima članove porodice ili nema.



Ugnježdjeni upiti

- Da bi primenili EXISTS sa ugnježdjenim upitom potrebno je da ugnježdjeni upit povežemo (korelišemo) sa spoljašnjim upitom.

```
SELECT MATBR, LIME, PREZIME
```

```
FROM RADNIK
```

```
WHERE EXISTS (SELECT IME
```

```
FROM CLAN_PORODICE
```

```
WHERE CLAN_PORODICE.MATBRRAD = RADNIK.MATBR);
```

MATBR	LIME	PREZIME
123456789	Marko	Petrović
333445555	Sima	Todorović
987654321	Aleksandra	Petrović

- Zahvaljujući dodatnom uslovu spoljašnji i u gnježdjeni upit su povezani. Ugnježdjeni upit vraća podatke samo za konkretnog radnika.



Ugnježdeni upiti

- Ukoliko želimo da prikazemo podatke o radnicima koji nemaju članove porodice koristićemo klauzulu NOT EXISTS.

```
SELECT MATBR, LIME, PREZIME
```

```
FROM RADNIK
```

```
WHERE NOT EXISTS (SELECT 0
```

```
FROM CLAN_PORODICE
```

```
WHERE CLAN_PORODICE.MATBRRAD = RADNIK.MATBR);
```

- **Obratite pažnju da ugnježdeni upit vraća samo konstantu vrednost 0.** Ovakvo rešenje se primenjuje jako često kada nam nisu neophodni podaci iz ugnježdenog upita već samo želimo da proverimo da li oni postoje.



Ugnježdeni upiti

- U nastavku je dat SQL upit koji određuje i prikazuje podatke o radnicima koji imaju više od dva člana porodice.



SELECT LIME, PREZIME

FROM RADNIK

WHERE (**SELECT COUNT**(*)

FROM CLAN_PORODICE

WHERE RADNIK.MATBR = CLAN_PORODICE.MATBRRAD) >= 2;

 LIME	 PREZIME
Marko	Petrović
Sima	Todorović

- Ugnježdeni upit je opet korelisan sa spoljašnjim upitom.



Ugnježdjeni upiti

- Naći listu projekata u koje je uključen radnik sa imenom Petrović kao radnik ili rukovodilac sektora koji izvodi projekte.

SELECT DISTINCT NAZIV

FROM PROJEKAT

WHERE BROJBR IN (SELECT BROJPR

FROM PROJEKAT, SEKTOR, RADNIK

WHERE BRS = SBROJ

AND MATBR = MATBRR

AND PREZIME = 'Petrović')

OR BROJPR IN (SELECT BRPR

FROM RADN_A, RADNIK

WHERE MBR = MATBR

AND PREZIME = 'Petrović');

NAZIV
1 Informacioni sistem
2 ProizvodX
3 ProizvodY
4 Godišnji izveštaj

BROJPR
1 20
2 30

BRPR
1 1
2 2
3 20
4 30



Ugnježdjeni upiti

- Prikazati imena i prezimena svih rukovodioca sektora koji imaju bar jednog člana porodice.

```
SELECT LIME, PREZIME  
FROM RADNIK  
WHERE EXISTS (SELECT *  
               FROM CLAN_PORODICE  
               WHERE MATBR = MATBRRAD)  
AND EXISTS (SELECT *  
            FROM SEKTOR  
            WHERE MATBR = MATBRR);
```

	LIME	PREZIME
1	Sima	Todorović
2	Aleksandra	Petrović



Oracle i ROWNUM u upitima

- U Oracle-u u rezultatima svakog upita postoji pseudo kolona pod nazivom ROWNUM koja sadrži redni broj reda u rezultujućoj tabeli. Redni broj počinje od 1.
- Oracle dodeljuje redni broj redu posle filtriranja uslovom selekcije iz **WHERE** klauzule
- Naziv kolone ROWNUM se može koristiti kao i nazivi svih ostalih kolona

```
SELECT ROWNUM, LIME, PREZIME  
FROM RADNIK  
WHERE ROWNUM < 5;
```

	ROWNUM	LIME	PREZIME
1	1	Marko	Petrović
2	2	Sima	Todorović
3	3	Valentina	Kovačević
4	4	Aleksandra	Petrović



Oracle i ROWNUM u upitima

- Oracle DBMS dodeljuje ROWNUM brojeve redovima prilikom **PRIBAVLJANJA!**
- Ilustrativan primer:

```
SELECT *  
FROM RADNIK  
WHERE ROWNUM > 1;
```

1	LIME	2	SSLOVO	3	PREZIME	4	MATBR	5	DATRODJ	6	POL	7	PLATA	8	ADRESA	9	MATBRS	10	BRSEK
---	------	---	--------	---	---------	---	-------	---	---------	---	-----	---	-------	---	--------	---	--------	----	-------

1. Neće vratiti ni jedan red u rezultatu!!!
2. Prvi red koji treba pribaviti ima ROWNUM 1 i ne zadovoljava uslov, posledica je da red nije pribavljen i ROWNUM i dalje ostaje 1
3. Sledeći red koji treba pribaviti takođe dobija ROWNUM 1 i neće biti pribavljen, a ROWNUM i dalje ostaje 1 i tako do kraja ni jedan red ne zadovoljava uslov i nema redova u rezultatu upita



ROWNUM i sortiranje

- Ukoliko se u upitu rezultati sortiraju po nekom kriterijumu i redni brojevi kolona u rezultatu će biti izmešani zato što su dodeljeni redovima prilikom pribavljanja a to je **PRE** sortiranja!
- Česta greška prilikom pisanja SQL upita tipa *top-N* (prvih N redova u nekom sortiranom rezultatu)
- Može izgledati da upit koji vraća imena, prezimena i iznos plate za prvih 5 radnika koji imaju najveće plate u preduzeću treba da izgleda:

```
SELECT ROWNUM, LIME, PREZIME,  
PLATA  
FROM RADNIK  
WHERE ROWNUM < 5  
ORDER BY PLATA DESC;
```

	ROWNUM	LIME	PREZIME	PLATA
1	4	Aleksandra	Petrović	43000
2	2	Sima	Todorović	40000
3	1	Marko	Petrović	30000
4	3	Valentina	Kovačević	25000

- Rezultat je neočekivan i **POGREŠAN!**



ROWNUM i sortiranje

- Pravilno napisan upit ovog tipa koristi ugnježdjeni upit koji obavlja sortiranje

```
SELECT ROWNUM, LIME, PREZIME, PLATA  
FROM (SELECT LIME, PREZIME, PLATA  
      FROM RADNIK  
      ORDER BY PLATA DESC)  
WHERE ROWNUM < 5;
```

	ROWNUM	LIME	PREZIME	PLATA
1	1	Jovan	Obradović	55000
2	2	Aleksandra	Petrović	43000
3	3	Sima	Todorović	40000
4	4	Velibor	Jovanović	36000



- Koja je razlika?
- U unutrašnjem upitu će ROWNUM vrednosti biti izmešane zato što su generisane pre sortiranja. Ova ROWNUM kolona se **NE KORISTI!**
- Spoljašnji upit pribavlja redove iz rezultata ugnježdenog upita i u tom procesu se generišu nove vrednosti za ROWNUM kolonu i te vrednosti se koriste u spoljašnjem upitu






Primer upita

- Odrediti ime i prezime radnika koji ima najviše članova porodice

```
SELECT LIME, PREZIME  
FROM (SELECT LIME, PREZIME, COUNT(*) AS BR_CLANOVA  
      FROM RADNIK, CLAN_PORODICE  
      WHERE RADNIK.MATBR = CLAN_PORODICE.MATBRRAD  
      GROUP BY LIME, PREZIME  
      ORDER BY BR_CLANOVA DESC)  
WHERE ROWNUM = 1;
```

	 LIME	 PREZIME
1	Sima	Todorović




	 LIME	 PREZIME	 BR_CLANOVA
1	Sima	Todorović	3
2	Marko	Petrović	3
3	Aleksandra	Petrović	1



Primer upita

- Problem sa prethodnim upitom je što vraća samo jedan iako ima više radnika koji zadovoljavaju uslov
- Alternativno rešenje

```
SELECT LIME, PREZIME, COUNT(*) AS BR_CLANOVA  
FROM RADNIK, CLAN_PORODICE  
WHERE RADNIK.MATBR = CLAN_PORODICE.MATBRRAD  
GROUP BY LIME, PREZIME  
HAVING COUNT(*) = (SELECT MAX(COUNT(*))  
FROM RADNIK, CLAN_PORODICE  
WHERE RADNIK.MATBR = CLAN_PORODICE.MATBRRAD  
GROUP BY LIME, PREZIME);
```

	 LIME	 PREZIME	 BR_CLANOVA
1	Sima	Todorović	3
2	Marko	Petrović	3



CONNECT BY

- Naredni SQL upit prikazuje organizacionu strukturu preduzeća.

```
SELECT LIME, PREZIME, LEVEL  
FROM RADNIK  
CONNECT BY PRIOR MATBR = MATBRS  
START WITH LIME = 'Jovan';
```

RZ	LIME	RZ	PREZIME	RZ	LEVEL
	Jovan		Obradović		1
	Sima		Todorović		2
	Marko		Petrović		3
	Velibor		Jovanović		3
	Jelena		Janković		3
	Aleksandra		Petrović		2
	Valentina		Kovačević		3
	Stanko		Manojlović		3

- Klauzula START WITH određuje čvor od koga se kreće sa formiranjem stabla.
- Klauzula PRIOR određuje uslov povezivanja. Povezivanje ide od radnika na višem nivou (od šefa) ka radniku na nižem nivou (ka podređenom).



CONNECT BY




- Ako u klauzuli PRIOR promenimo redosled atributa u uslovu, promeniće se i smer povezivanja.

SELECT LIME, PREZIME, LEVEL

FROM RADNIK

CONNECT BY PRIOR MATBRS = MATBR

START WITH LIME = 'Stanko';

 LIME	 PREZIME	 LEVEL
Stanko	Manojlović	1
Aleksandra	Petrović	2
Jovan	Obradović	3