



Računarstvo i informatika

Katedra za računarstvo Elektronski fakultet u Nišu

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

Letnji semestar 2017/2018







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



- 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 atribute neophodne za jednoznačnu identifikaciju slogva.

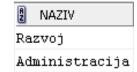


 U nastavku je dat SQL upit koji vraća nazive sektora u kojima rade radnici koji se prezivaju Petrović i Jovanović.

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';

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

**SELECT DISTINCT NAZIV** 

FROM SEKTOR, RADNIK

NAZIV Razvoj

WHERE SBROJ = BRSEK AND PREZIME = 'Jovanović';



• 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.

```
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.

```
FROM SEKTOR, RADNIK

WHERE SBROJ = BRSEK AND PREZIME = 'Petrović'

UNION

SELECT DISTINCT SBROJ, NAZIV

FROM SEKTOR, RADNIK

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



 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';
```



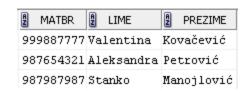
 U nastavku je dat SQL upit koji korišćenjem klauzule INTERSECT određuje podatke o radnicima koji rade u sektoru Administracija i koji imaju platu veću od 40000.

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

SELECT MATBR, LIME, PREZIME

FROM SEKTOR, RADNIK

WHERE SBROJ = BRSEK AND NAZIV = 'Administracija'



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

SELECT MATBR, LIME, PREZIME

FROM RADNIK

WHERE PLATA > 40000;



## 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.



## 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;



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





 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** 



SELECT MATBR, LIME, PREZIME, UVECANA PLATA

FROM (SELECT MATBR, LIME, PREZIME, PLATA + 5000 AS UVECANA\_PLATA

**FROM RADNIK)** 

WHERE UVECANA\_PLATA > 40000;

<u> </u>	MATBR	A	LIME	A	PREZIME	A	UVECANA_PLATA
333	445555	Sir	na	To	dorović		45000
987	654321	Ale	eksandra	Per	trović		48000
666	884444	Ve.	libor	Jo	vanović		41000
888	665555	Jot	7an	0b:	radović		60000



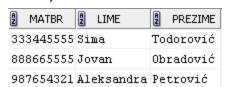


 U nastavku je dat upit koji prikazuje imena i prezimena radnika koji se nalaze na položaju rukovodioca sektora.

**SELECT MATBR, LIME, PREZIME** 

**FROM RADNIK** 

WHERE MATBR IN (SELECT MATBRR FROM SEKTOR);



• 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);





 U nastavku je dat upit koji korišćenjem klauzule EXISTS određuje podatke o radnicima koji imaju članove porodice.

• Klauzula EXISTS proverava da li ugnježdeni upit vraća

rezultat koji sadrži slogove ili ne.

SELECT MATBR, LIME, PREZIME
FROM RADNIK
WHERE EXISTS (SELECT IME FROM CLAN\_PORODICE);

MATBR	2 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 ugnježdenom upitu. Ugnježdeni upit treba da se izvrši za svakog radnika i proveri da li on ima članove porodice.
- U ovom slučaju ugnježdeni upit uvek vraća isti rezultat (sve članove porodice) bez obzira da li radnik ima članove porodice ili nema.





 Da bi primenili EXISTS sa ugnježdenim upitom potrebno je da ugnježdeni upit povežemo (korelišemo) sa spoljašnjim upitom.

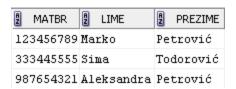
**SELECT MATBR, LIME, PREZIME** 

FROM RADNIK

WHERE EXISTS (SELECT IME

FROM CLAN\_PORODICE

WHERE CLAN\_PORODICE.MATBRRAD = RADNIK.MATBR);



Zahvaljujući dodatnom uslovu spoljašnji i u gnježdeni upit su povezani.
 Ugnježdeni upit vraća podatke samo za konkretnog radnika.





 Ukoliko želimo da prikažemo 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.





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

FROM RADNIK

WHERE (SELECT COUNT(\*)

FROM CLAN\_PORODICE

WHERE RADNIK.MATBR = CLAN\_PORODICE.MATBRRAD) >= 2;

Ugnježdeni upit je opet korelisan sa spoljašnjim upitom.





NAZIV

2 ProizvodX

3 ProizvodY 4 Godišnii izveštai

2

Informacioni sistem

**BROJPR** 20

30



 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 RADI\_NA, RADNIK

WHERE MBR = MATBR

AND PREZIME = 'P

	A	BRPR
1		1
2		2
3		20
4		30

NI DK	3	20	
Petrović');	4	30	





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

SELECT LIME, PREZIME

**FROM RADNIK** 

WHERE EXISTS (SELECT \*

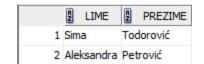
FROM CLAN\_PORODICE

WHERE MATBR = MATBRRAD)

AND EXISTS (SELECT \*

**FROM SEKTOR** 

**WHERE MATBR = MATBRR**);



# 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	2 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:

```
FROM RADNIK
WHERE ROWNUM > [];
```



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

  Baze podataka









- 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;** 

	A	ROWNUM	A	LIME	A	PREZIME	A	PLATA
1		4	Ale	ksandra	Pet	rović		43000
2		2	Sim	a	Too	dorović		40000
3		1	Mai	rko	Pet	rović		30000
4		3	Val	entina	Ko۱	/ačević		25000

Rezultat je neočekivan i POGREŠANI dataka





#### ROWNUM i sortiranje

Pravilno napisan upit ovog tipa koristi ugnježdeni upit koji obavlja sortiranje

SELECT ROWNUM, LIME, PREZIME, PLATA FROM (SELECT LIME, PREZIME, PLATA FROM RADNIK ORDER BY PLATA DESC)

	A	ROWNUM	A	LIME	A	PREZIME	A	PLATA
1		1	Jov	an	Obi	radović		55000
2		2	Ale	ksandra	Pet	rović		43000
3		3	Sim	a	Too	dorović		40000
4		4	Veli	bor	Jov	anović		36000

Koja je razlika?

WHERE **ROWNUM** < 5:

- 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

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;



	2 LIME	PREZIME	A	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

```
ELECT LIME, PREZIME, COUNT(*) AS BR_CLANOVA

ROM 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);
```

	A LIME	2 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';

2 LIME	2 PREZIME	2 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';

