Rad sa Bazama podataka

– SQL SELECT naredba –

08

Računske vežbe

Dr Dušan Stefanović

Dipl. inž. Nikola Vukotić

AKADEMIJA TEHNIČKO-VASPITAČKIH STRUKOVNIH STUDIJA - ODSEK NIŠ

Sadržaj

- Spajanje tabela
- GROUP BY
- HAVING

Spajanje tabela

- Svi SQL upiti koje smo do sada razmatrali su koristili podatke iz samo jedne tabele
- Često se javlja situacija da se tražena informacija nalazi u većem broju tabela
- U takvim situacijama potrebno je izvršiti spajanje vrsta iz različitih tabela i generisanje rezultujuće tabele
- Za pribavljanje podataka iz većeg broja tabela dovoljno je u klauzuli FROM navesti imena tabela iz kojih želimo da pribavimo podatke

Spajanje tabela

- Da bi spajanje tabela bilo uspešno, osim u nekim specijalnim slučajevima, potrebno je da navedemo uslov spoja, odnosno da navedemo kolone na osnovu čijih vrednosti se vrši spajanje vrsta iz različitih tabela
- Spajanje tabela se vrši tako što se najčešće uparuje strani ključ iz jedne tabele sa primarnim ključem koji referencira u drugoj tabeli
- Uslov spajanja može da se zada u okviru WHERE klauzule ili korišćenjem ključne reči JOIN u okviru FROM klauzule.

Spajanje tabela

- **Spoj na jednakost** (equi-join) obezbeđuje spajanje podataka iz dve ili više tabela na osnovu jednakosti odgovarajućih atributa, obično na osnovu primarnih i spoljnih ključeva. Najjednostavniji slučaj navođenja spoja je kada se u WHERE klauzuli specificira uslov spoja po jednakosti
- **Dekartov proizvod** je slučaj kada u WHERE klauzuli ne postoji uslov spoja, a u FROM klauzuli je navedeno više tabela. U tom slučaju nema spajanja vrsta po vrednosti nekog atributa, već se pravi kombinacija svake vrste iz jedne tabele sa svakom vrstom iz druge tabele (u slučaju Dekartovog proizvoda dve tabele)
- **Spoljni spoj** (outer-join) omogućava spajanje dve tabele po vrednosti nekog atributa (kao kod equi-join), ali i uključivanje onih torki (vrsta) iz jedne ili druge tabele (ili iz obe), koje ne zadovoljavaju uslov jednakosti

Spajanje tabela

- SQL standard definiše sledeće tipove spoja:
  - cross join
  - inner join
  - outer join
    - left outer join
    - right outer join
    - full outer join

INNER join

- Unutrašnji spoj (Inner join) predstavlja najčešće korišćeni tip spoja. Ovaj tip spoja, u osnovi, definiše presek vrsta iz tabela koje učestvuju u spoju
- Prilikom spajanja dve tabele (A INNER JOIN B) uzimaju se sve vrste iz tabele A i pronalazi im se odgovarajuća vrsta u tabeli B. Ukoliko vrsta iz tabele A nema odgovarajuću vrstu u tabeli B ne uključuje se u rezultat. Ukoliko vrsti iz tabele A odgovara više vrsta tabele B ona se u rezultatu ponavlja više puta (po jednom za svaku odgovarajuću vrstu u tabeli B)
- Prilikom spajanja tabela treba voditi računa o tome da kolone u različitim tabelama mogu imati ista imena
- U takvim situacijama se koristi sintaksa IME_TABELE.IME_KOLONE

INNER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Tri alternative:
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK
  INNER JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK, SEKTOR
  WHERE RADNIK.BRSEK = SEKTOR.SBROJ;
  ```
  ```

INNER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

```
SELECT RADNIK.MATBR, RADNIK.LIME,
RADNIK.PREZIME, RADNIK.BRSEK, SEKTOR.SBROJ,
SEKTOR.NAZIV
FROM RADNIK, SEKTOR
WHERE RADNIK.BRSEK = SEKTOR.SBROJ;
```

```
SELECT r.MATBR, r.LIME, r.PREZIME, r.BRSEK, s.SBROJ,
s.NAZIV
FROM RADNIK r, SEKTOR s
WHERE r.BRSEK = s.SBROJ
```

INNER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Primer

MATBR	LIME	PREZIME	BRSEK	SBROJ	NAZIV
123456789	Marko	Petrović	5	5	Razvoj
333445555	Sima	Todorović	5	5	Razvoj
999887777	Valentina	Kovačević	4	4	Administracija
987654321	Aleksandra	Petrović	4	4	Administracija
666884444	Velibor	Jovanović	5	5	Razvoj
453453453	Jelena	Janković	5	5	Razvoj
987987987	Stanko	Manojlović	4	4	Administracija
888665555	Jovan	Obradović	1	1	Uprava

LEFT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Levi spoljašnji spoj (Left-outer Join) u osnovi predstavlja prošireni inner-join
- Levi spoljašnji spoj (A LEFT OUTER JOIN B) pored vrsta koje uključuje unutrašnji spoj uključuje i vrste iz tabele A (leve tabele) koje nemaju odgovarajuću vrstu u tabeli B (desnoj tabeli)
- U vrstama koje iz tabele A koje ne mogu da se upare ni sa jednom vrstom iz tabele B, kolone iz tabele B imaju vrednost NULL

LEFT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Tri alternative:
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK
  LEFT OUTER JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK
  LEFT JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```
- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK, SEKTOR
  WHERE RADNIK.BRSEK = SEKTOR.SBROJ(+);
  ```
  ```

LEFT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME, RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK, SEKTOR
  WHERE RADNIK.BRSEK = SEKTOR.SBROJ(+);

- SELECT r.MATBR, r.LIME, r.PREZIME, r.BRSEK, s.SBROJ, s.NAZIV
  FROM RADNIK r, SEKTOR s
  WHERE r.BRSEK = s.SBROJ(+);

LEFT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Primer

MATBR	LIME	PREZIME	BRSEK	SBROJ	NAZIV
123456789	Marko	Petrović	5	5	Razvoj
333445555	Sima	Todorović	5	5	Razvoj
999887777	Valentina	Kovačević	4	4	Administracija
987654321	Aleksandra	Petrović	4	4	Administracija
666884444	Velibor	Jovanović	5	5	Razvoj
453453453	Jelena	Janković	5	5	Razvoj
987987987	Stanko	Manojlović	4	4	Administracija
888665555	Jovan	Obradović	1	1	Uprava
111111111	Borivoje	Veljković	10	(null)	(null)

RIGHT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Desni spoljašnji spoj (Right outer join) funkcioniše kao i left outer join samo je sada uloga tabela promenjena
- Desni spoljašnji spoj (A RIGHT OUTER JOIN B) pored vrsta koje uključuje unutrašnji spoj u rezultat uključuje vrste iz tabele B (desne tabele) koje nemaju odgovarajuću vrstu u tabeli A (levoj tabeli)
- U vrstama koje iz tabele B koje ne mogu da se upare ni sa jednom vrstom iz tabele A, kolone iz tabele A imaju vrednost NULL

RIGHT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Tri alternative:

- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK
  RIGHT OUTER JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```

- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK
  RIGHT JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```

- ```
  SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME,
  RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK, SEKTOR
  WHERE RADNIK.BRSEK(+)= SEKTOR.SBROJ;
  ```
  ```

RIGHT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- SELECT RADNIK.MATBR, RADNIK.LIME, RADNIK.PREZIME, RADNIK.BRSEK, SEKTOR.SBROJ, SEKTOR.NAZIV
  FROM RADNIK, SEKTOR
  WHERE RADNIK.BRSEK(+)= SEKTOR.SBROJ;

- SELECT r.MATBR, r.LIME, r.PREZIME, r.BRSEK, s.SBROJ, s.NAZIV
  FROM RADNIK r, SEKTOR s
  WHERE r.BRSEK(+)= s.SBROJ;

RIGHT OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Primer

MATBR	LIME	PREZIME	BRSEK	SBROJ	NAZIV
888665555	Jovan	Obradović	1	1	Uprava
987987987	Stanko	Manojlović	4	4	Administracija
999887777	Valentina	Kovačević	4	4	Administracija
987654321	Aleksandra	Petrović	4	4	Administracija
453453453	Jelena	Janković	5	5	Razvoj
333445555	Sima	Todorović	5	5	Razvoj
123456789	Marko	Petrović	5	5	Razvoj
666884444	Velibor	Jovanović	5	5	Razvoj
(null)	(null)	(null)	(null)	12	Prodaja

FULL OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Potpuni spoljašnji spoj (Full outer join) predstavlja kombinaciju rezultata koje vraćaju left outer i right outer join
- Potpuni spoljašnji spoj (A FULL OUTER JOIN B) pored vrsta koje uključuje unutrašnji spoj u rezultat uključuje i vrste iz obe tabele (i iz A i iz B) koje nemaju odgovarajuće slogove u drugoj tabeli
- U vrstama koje ne mogu da se upare, nedostajuće kolone imaju vrednost NULL

FULL OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Dve alternative:

- ```
  SELECT RADNIK.MATBR, RADNIK.LIME,
  RADNIK.PREZIME, RADNIK.BRSEK, SEKTOR.SBROJ,
  SEKTOR.NAZIV
  FROM RADNIK
  FULL OUTER JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```

- ```
  SELECT RADNIK.MATBR, RADNIK.LIME,
  RADNIK.PREZIME, RADNIK.BRSEK, SEKTOR.SBROJ,
  SEKTOR.NAZIV FROM RADNIK
  FULL JOIN SEKTOR
  ON RADNIK.BRSEK = SEKTOR.SBROJ;
  ```

FULL OUTER join

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Primer

MATBR	LIME	PREZIME	BRSEK	SBROJ	NAZIV
123456789	Marko	Petrović	5	5	Razvoj
333445555	Sima	Todorović	5	5	Razvoj
999887777	Valentina	Kovačević	4	4	Administracija
987654321	Aleksandra	Petrović	4	4	Administracija
666884444	Velibor	Jovanović	5	5	Razvoj
453453453	Jelena	Janković	5	5	Razvoj
987987987	Stanko	Manojlović	4	4	Administracija
888665555	Jovan	Obradović	1	1	Uprava
111111111	Borivoje	Veljković	10	(null)	(null)
(null)	(null)	(null)	(null)	12	Prodaja

Spajanje tabela

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- SQL upit koji prikazuje imena i prezimena svih radnika koji rade u sektoru ’Razvoj’

- ```
  SELECT r.LIME, r.PREZIME, s.NAZIV
  FROM RADNIK r, SEKTOR s
  WHERE s.NAZIV = 'Razvoj'
  AND r.BRSEK = s.SBROJ;
  ```

Uslov selekcije

Uslov spoja

- Rezultat:

LIME	PREZIME	NAZIV
Marko	Petrović	Razvoj
Sima	Todorović	Razvoj
Velibor	Jovanović	Razvoj
Jelena	Janković	Razvoj
  ```

Spajanje tabela

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji za sve radnike ženskog pola određuje imena projekata na kojima su angažovane
- Podaci se izvlače iz tri tabele: RADNIK, RADI_NA i PROJEKAT
- Potrebno je voditi računa da spojevi između tabela budu definisani na odgovarajući način kako bi dobili željene podatke
- Spoj se najčešće definiše između spoljašnjeg ključa u jednoj tabeli i primarnog ključa koji se referencira u drugoj tabeli (RADI_NA.Mbr i RADNIK.MatBr, RADI_NA.BrPr i PROJEKAT.BrojPr)

Spajanje tabela

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

```
SELECT R.MATBR, R.LIME, R.PREZIME, P.NAZIV
FROM RADNIK R, RADI_NA RN, PROJEKAT P
WHERE R.MATBR = RN.MBR
AND RN.BRPR = P.BROJPR
AND R.POL = 'Ž';
```

MATBR	LIME	PREZIME	NAZIV
453453453	Jelena	Janković	ProizvodX
453453453	Jelena	Janković	ProizvodY
987654321	Aleksandra	Petrović	Informacioni sistem
987654321	Aleksandra	Petrović	Godišnji izveštaj
999887777	Valentina	Kovačević	Reorganizacija
999887777	Valentina	Kovačević	Godišnji izveštaj

Spajanje tabela

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji za sve projekte locirane u Nišu prikazuje broj projekta, broj sektora koji ga izvodi i ime, prezime i datum rođenja rukovodioca

- ```
  SELECT BROJPR, BRS, LIME,PREZIME, DATRODJ
  FROM PROJEKAT
  INNER JOIN SEKTOR
  ON SBROJ = BRS
  INNER JOIN RADNIK
  ON MATBR = MATBRR WHERE LOKPR = 'Niš';
  ```

- ```
  SELECT BROJPR, BRS, LIME,PREZIME,DATRODJ
  FROM PROJEKAT, SEKTOR, RADNIK
  WHERE SBROJ = BRS
  AND MATBR = MATBRR
  AND LOKPR = 'Niš';
  ```

BROJPR	BRS	LIME	PREZIME	DATRODJ
1	5	Sima	Todorović	08-DEC-55
3	5	Sima	Todorović	08-DEC-55
10	1	Jovan	Obradović	10-NOV-47
30	4	Aleksandra	Petrović	20-JUN-41

Spajanje tabela

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji za svakog radnika prikazuje ime i prezime kao i ime i prezime njegovog šefa
- SELECT X.LIME, X.PREZIME,Y.LIME,Y.PREZIME
  FROM RADNIK X, RADNIK Y
  WHERE X.MATBRS =Y.MATBR;

LIME	PREZIME	LIME_1	PREZIME_1
Marko	Petrović	Sima	Todorović
Sima	Todorović	Jovan	Obradović
Valentina	Kovačević	Aleksandra	Petrović
Aleksandra	Petrović	Jovan	Obradović
Velibor	Jovanović	Sima	Todorović
Jelena	Janković	Sima	Todorović
Stanko	Manojlović	Aleksandra	Petrović
Borivoje	Veljković	Aleksandra	Petrović

GROUP BY i HAVING

- Funkcije agregacije imaju zadatak da omoguće generisanje sumarnih informacija na osnovu podataka u relacionoj bazi podataka
- Klauzula GROUP BY ima zadatak da omogući grupisanje vrsta u rezultujućoj tabeli na osnovu zajedničkih vrednosti kolona
- Time se povećava vrednost funkcija agregacije jer se u kombinaciji sa GROUP BY klauzulom mogu primenjivati na grupe vrsta a ne samo na čitavu rezultujuću tabelu
- Klauzula HAVING se koristi za filtriranje podataka nakon primene klauzule GROUP BY i funkcija agregacije.

GROUP BY i HAVING

- Potrebno je voditi računa, da ukoliko ne postoji GROUP BY klauzula, u SELECT klauzuli nije moguće kombinovati funkcije agregacije sa imenima kolona
- U nastavku je dat SQL upit koji NE MOŽE DA SE IZVRŠI i koji će dovesti do POJAVE GREŠKE
- SELECT Ime, Prezime, SUM(Plata)
  FROM RADNIK;

GROUP BY i HAVING

- Klauzula GROUP BY zahteva od DBMS-a da izvrši sortiranje rezultujuće tabele prema specificiranim kolonama i izvrši grupisanje vrsta koje imaju iste vrednosti za specificirane kolone
- Ukoliko su prisutne funkcije agregacije one će se primeniti na tako dobijene grupe
- Tek uz prisustvo GROUP BY klauzule moguće je u SELECT klauzuli kombinovati imena kolona i funkcije agregacije
- Bitno je da napomenuti da se klauzula GROUP BY izvršava nakon klauzule WHERE odnosno da se grupisanje vrši tek nakon što su određene vrste koje treba da uđu u sastav rezultujuće tabele

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- SELECT LIME, PREZIME, BRSEK
  FROM RADNIK
  GROUP BY BRSEK, LIME, PREZIME;

LIME	PREZIME	BRSEK
Marko	Petrović	5
Sima	Todorović	5
Jelena	Janković	5
Velibor	Jovanović	5
Jovan	Obradović	1
Aleksandra	Petrović	4
Valentina	Kovačević	4
Stanko	Manojlović	4

- SELECT BRSEK, COUNT(*)
  FROM RADNIK
  GROUP BY BRSEK;

BRSEK	COUNT(*)
1	1
5	4
4	3

- SELECT BRSEK, COUNT(*)
  FROM RADNIK
  GROUP BY BRSEK
  HAVING COUNT(*) > 1;

BRSEK	COUNT(*)
5	4
4	3

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Klauzule GROUP BY i HAVING je moguće kombinovati sa WHERE klauzulom
- Pri tome treba voditi računa o redosledu izvršavanja (redosled izvršavanja odgovara redosledu po kome se klauzule rešaju prilikom pisanja SELECT naredbe):

  1. WHERE – primenjuje se predikat koji određuje vrste koje ulaze u sastav rezultujuće tabele

  2. GROUP BY – vrši grupisanje vrsta u rezultujućoj tabeli

  3. HAVING – primenjuje se predikat koji određene vrste koj eće ostati u rezultatu upita

  4. ORDER BY – sortiranje rezultata se vrši tek na kraju

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji prikazuje brojeve sektora u kojima radi više od jednog radnika ženskog pola

- ```
  SELECT BRSEK, COUNT(*)
  FROM RADNIK
  WHERE POL = 'Ž'
  GROUP BY BRSEK
  HAVING COUNT(*) > 1;
  ```

BRSEK	COUNT(*)
4	2
  ```

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- Klauzulu GROUP BY je moguće primeniti istovremeno na veći broj kolona
- Pri tome su kriterijum za formiranje grupa zajedničke vrednosti u specificiranim kolonama
- Prilikom formiranja grupa vodi se računa i o redosledu po kome su kolone za grupisanje navedene (kao da se formiraju grupe sa podgrupama u okviru njih)

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji za svaki sektor prikazuje prosečnu zaradu po polovima
- SELECT BRSEK, POL, AVG(*) AS PROSEK_PLATA
  FROM RADNIK
  GROUP BY BRSEK, POL
  ORDER BY AVG(PLATA);

BRSEK	POL	PROSEK_PLATA
5	Ž	25000
4	M	25000
4	Ž	34000
5	M	35333.333333333333333333333333333333333
1	M	55000

- Prilikom korišćenja klauzule GROUP BY, sve kolone koje su navedene u klauzuli SELECT a na koje nije primenjena neka funkcija agregacije, MORAJU BITI NAVEDENE U GROUP BY KLAUZULI
- U suprotnom SQL upit neće moći da se izvrši!

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji prikazuje imena sektora i broj radnika koji rade u njima

```
SELECT S.SBROJ, S.NAZIV, COUNT(*) AS BROJ_RADNIKA
FROM SEKTOR S
INNER JOIN RADNIK R
ON S.SBROJ=R.BRSEK
GROUP BY S.SBROJ, S.NAZIV;
```

SBROJ	NAZIV	BROJ_RADNIKA
5	Razvoj	4
4	Administracija	3
1	Uprava	1

GROUP BY i HAVING

- RADNIK(Lime, Sslovo, Prezime, Pol, Adresa, Plata, Matbr, DatRodj, *BrojSekt*)
- SEKTOR(Naziv, Sbroj, *MatbrR*, DatPost)
- PROJEKAT(Naziv, BrojPr, Lokacija, *BrojSek*)
- CLAN_PORODICE(*MatBrRad*, Ime, Pol, Srodstvo, DatRodj)
- RADI_NA(*Mbr*, *BrPr*, Sati)
- LOK_SEK(*BrS*, Lokacija)

- U nastavku je dat SQL upit koji za sve projekte koji imaju više od dva angažovana radnika prikazuje broj projekta, ime projekta i broj radnika koji na njemu rade

```
SELECT BROJPR, NAZIV, COUNT(*)
FROM PROJEKAT, RADI_NA
WHERE BROJPR = BRPR
GROUP BY BROJPR, NAZIV
HAVING COUNT(*) > 2;
```

BROJPR	NAZIV	COUNT(*)
20	Informacioni sistem	4
10	Reorganizacija	3
2	ProizvodY	3

Pitanja?