Napredne računarske aplikacije

Predavanje broj: 07

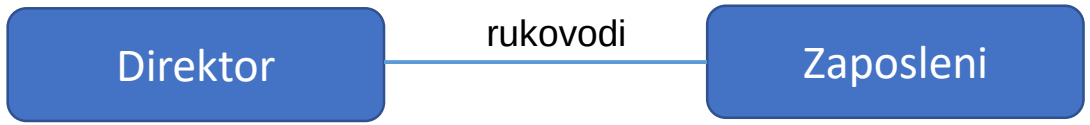
Nastavne teme:

- ✓VEZE
- ✓PODUPITI – UGNJEŽDENI UPITI
- ✓VRSTE SPAJANJA (INNER, LEFT, RIGHT, FULL, OUTER)

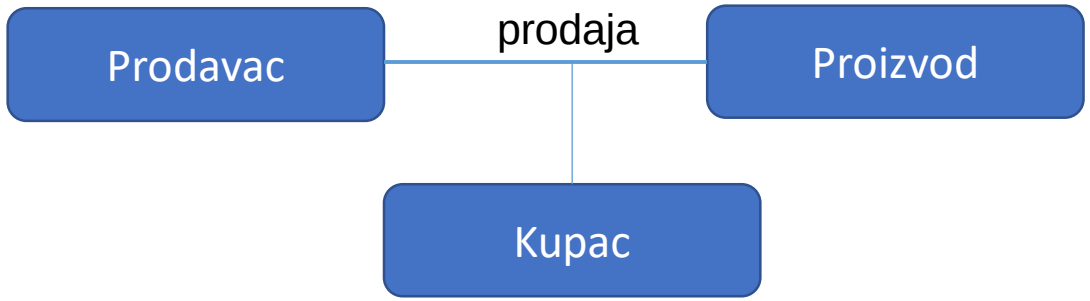
Veze

- Entiteti se mogu povezivati jedan s drugim u veze (relacije).
- Broj entiteta u vezi predstavlja STEPEN VEZE.

Binarna veza:
veza 2 entiteta



Ternarna veza:
veza 3 entiteta



Unarna veza:
isti entitet više puta egzistira u različitim ulogama

Kardinalnost veze

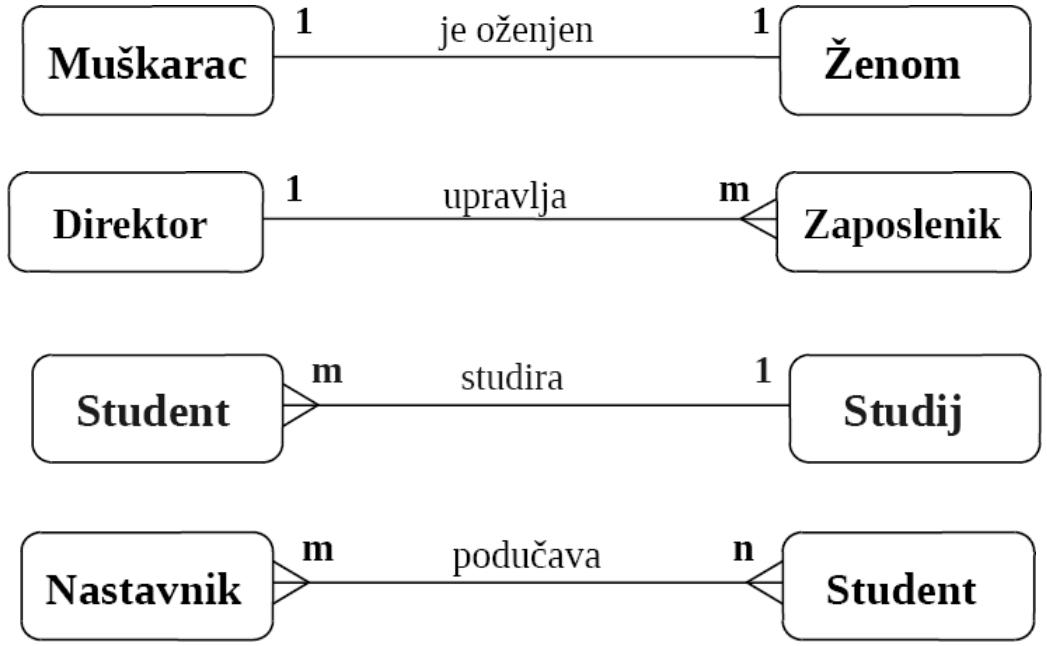
- **Odnos broja povezanim entitetima nazivamo kardinalnost veze**

- Jedan na jedan **(1:1)**

- Jedan na više **(1:m)**

- Više na jedan **(m:1)**

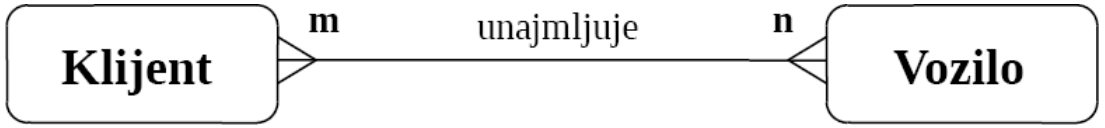
- Više na više **(m:n)**



Razbijanje M:N veza

Veza m:n u ER modelu se može razbiti uvođenjem novog veznog entiteta.

Primer:



Vezu m:n možemo razbiti uvođenjem entiteta najam, koji sadrži atribut datum_najma

Preslikavanje ER modela u relacije

VEZA JEDAN:VIŠE

- Primarni ključ entiteta sa strane veze JEDAN dodaje se kao strani ključ u entitet sa strane veze VIŠE.

VEZA VIŠE:VIŠE

- Doda se novi entitet, koji sadrži primarne ključeve entiteta koji učestvuju u vezi.
- Ti atributi zajedno čine složeni primarni ključ novonastalog entiteta.

Povratne (rekurzivne) veze

- Dodaje se strani ključ jednak primarnom ključu relacije.
- Za povratne veze vredi da je strani ključ jednak primarnom ključu relacije, ali pod drugim imenom

Podupiti

- Podupit je SELECT-rečenica ugnježdena u neku klauzulu ili drugu SELECT rečenicu.
- Podupit se izvršava jednom pre glavnog upita.
- Glavni ili spoljašnji upit koristi rezultat podupita.

Glavni ili spoljašnji upit

```
SELECT lista_kolona1
FROM tabela
WHERE ime_kolone_i_uslov_poređenja
```

Podupit ili unutrašnji upit

```
(SELECT lista_kolona2
FROM tabela);
```

Podupiti

- Podupit se koriste kada se:
  - ❖ sve informacije koje u upitu želimo da prikažemo (kolone) nalaze u jednoj tabeli
  - ❖ kolone preko kojih postavljamo uslove nalaze u drugim tabelama.

- Podupit (subquery) je upit ugrađen (ugnježden) u neki drugi upit.

- Spoljašnji i unutrašnji upit mogu biti povezani po vrednostima više atributa.

- Ako se upoređuju argumenti koji se sastoje od više atributa, oba argumenta moraju imati jednak broj atributa, a upoređuje se prvi sa prvim, drugi sa drugim itd.

- Atributi koji se upoređuju moraju biti istog ili kompatibilnog tipa podataka.

Podupiti

- Rezultat podupita se koristi kao konkretna vrednost.
  - ✓ ako se radi o upitu koji vraća jedan red: single row subquery.

- Ako podupit vraća skup vrednosti, mora se koristiti odredba IN.

- Podupiti se mogu nalaziti u sledećim SQL-klauzulama:

  SELECT,
  WHERE,
  HAVING,
  FROM.

Podupiti

Primer:

```
SELECT last_name, salary, salary*1.05
FROM employees
WHERE salary*1.05 > (SELECT AVG(salary) FROM employees);
```

Primer:

```
SELECT last_name, salary
FROM employees
WHERE department_id  = 20 AND salary IN  (SELECT salary FROM
employees WHERE department_id = 60);
```

Jednoredni podupiti

Jednoredni podupit kao rezultat vraća jedan red. Jednoredni podupiti se koriste sa jednorednim operatorima: >, =, >=, <, <>, <= .

Primer:

Koji radnici u tabeli F_STAFFS su mlađi od radnika koji se preziva Brown? Odgovor se dobija poređenjem datuma rođenja svih radnika sa Brown-ovim datumom rođenja.

Dakle, prvo je potrebno utvrditi kada je rođen Brown:

```
SELECT birthdate
FROM f_staffs
WHERE last_name = ‘Brown’;
```

BIRTHDATE
30-MAR-69

Jednoredni podupiti

Zatim se navedeni upit ugrađuje kao podupit u glavni upit kojim se dobija odgovor na postavljeno pitanje:

```
SELECT id, first_name, last_name, birthdate
FROM f_staffs
WHERE birthdate >
        (SELECT birthdate  FROM f_staffs WHERE last_name = 'Brown');
```

ID	FIRST_NAME	LAST_NAME	BIRTHDATE
12	Sue	Doe	01-JUL-80
9	Bob	Miller	19-MAR-79

Višeredni podupiti

Višeredni upiti se koriste sa višerednim operatorima:

- BETWEEN ... OR, IN, ALL, ANY.

Višeredni podupit kao rezultat vraća više redova i ne može se koristiti sa jednorednim operatorima u glavnom upitu.

Višeredni podupiti

Primer:

Prikazati naslove pesama, izvođače i trajanje pesama na CD-u čiji je identifikacioni broj=91.

Koristiti tabele D_SONGS, D_CDS i veznu tabelu D_TRACK_LISTINGS.

D_SONGS

ID	TITLE	DURATION	ARTIST	TYPE_CODE
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88

D_CDS

CD_NUMBER	TITLE	PRODUCER	YEAR
90	The Celebrants Live in Concert	Old Town Records	1997
91	Party Music for All Occasions	The Music Man	2000
92	Back to the Shire	Middle Earth Rec.	2002
93	Songs from My Childhood	Old Town Records	1999
94	Carpe Diem	R & B Inc.	2000
95	Here Comes the Bride	The Music Man	2001
96	Graduation Songbook	Tunes Are Us	1998
98	Whirled Peas	Old Town Records	2004

D_TRACK_LISTINGS

SONG_ID	CD_NUMBER	TRACK
45	92	1
46	93	1
47	91	2
48	95	5
49	91	3

Višeredni podupiti

```
SELECT title, artist, duration
FROM d_songs
WHERE id IN (SELECT song_id FROM d_track_listings WHERE
d_track_listings.cd_number = 91);
```

Rezultat podupita:

SONG_ID
47
49

Rezultat glavnog upita:

TITLE	ARTIST	DURATION
Hurrah for Today	The Jubilant Trio	3 min
Lets Celebrate	The Celebrants	8 min

Višeredni podupiti

Isti rezultat se može dobiti i spajanjem tabela:

```
SELECT s.title, s.artist, s.duration
FROM d_songs s, d_track_listings t
WHERE t.song_id = s.id AND t.cd_number = 91;
```

Pravila o podupitima

- Podupiti se navode u zagradi.
- Podupit je uvek na desnoj strani uslova poređenja.
- Spoljašnji (glavni) i unutrašnji upit (podupit) mogu uzeti podatke iz različitih tabela.
- Samo jedna klauzula ORDER BY može biti uključena u jednu SELECT-rečenicu i ona mora biti poslednja klauzula spoljašnjeg upita.
- Podupit ne može imati sopstvene klauzule ORDER BY.
- Jedino ograničenje broja podupita je veličina bafera koji se koristi za upit.

Pravila o podupitima

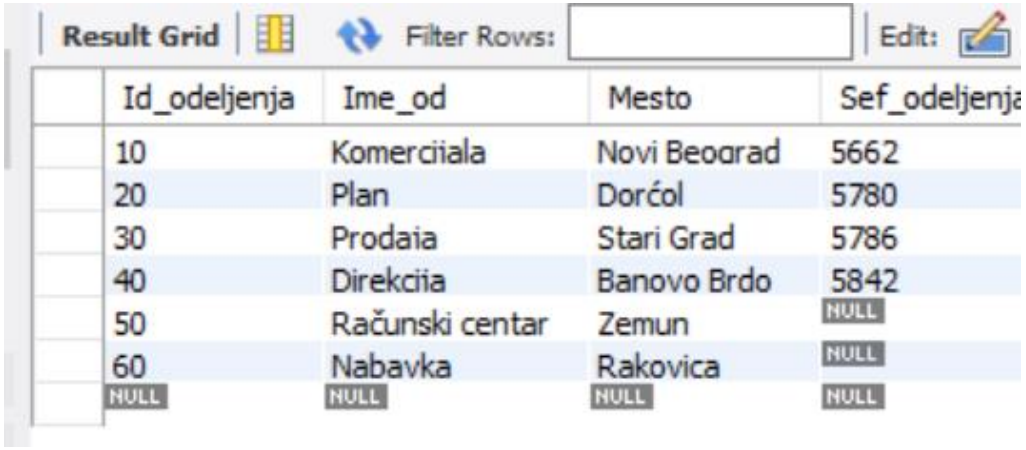
- Tip kolone navedene u podupitu mora biti isti kao tip izraza na levoj strani operatora.
- Povezivanje tabela dinamičkom zamenom rezultata jednog, unutrašnjeg upita u WHERE odredbi drugog, spoljnjeg upita, može se primeniti samo ako su svi podaci koji se prikazuju u spoljnjem upitu iz jedne tabele.

Podupiti

Ranije verzije MySQL nisu podržavale ugnježđene upite, dok novije verzije podržavaju i tu mogućnost.

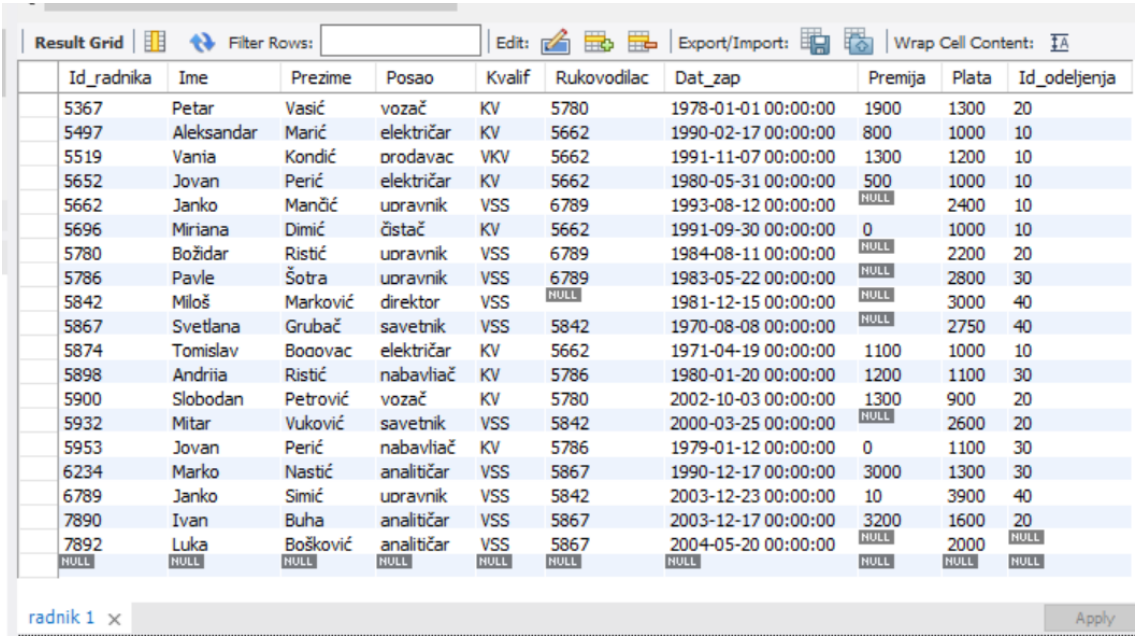
Podupiti - primer

Prikazati spisak zaposlenih koji rade na Dorćolu.



Result Grid | Filter Rows: | Edit:

Id_odeljenja	Ime_od	Mesto	Sef_odeljenja
10	Komerciiala	Novi Beograd	5662
20	Plan	Dorćol	5780
30	Prodaia	Stari Grad	5786
40	Direkciia	Banovo Brdo	5842
50	Računski centar	Zemun	NULL
60	Nabavka	Rakovica	NULL
NULL	NULL	NULL	NULL



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_radnika	Ime	Prezime	Posao	Kvalif	Rukovodilac	Dat_zap	Premija	Plata	Id_odeljenja
5367	Petar	Vasić	vozač	KV	5780	1978-01-01 00:00:00	1900	1300	20
5497	Aleksandar	Marić	električar	KV	5662	1990-02-17 00:00:00	800	1000	10
5519	Vania	Kondić	prodavac	VKV	5662	1991-11-07 00:00:00	1300	1200	10
5652	Jovan	Perić	električar	KV	5662	1980-05-31 00:00:00	500	1000	10
5662	Janko	Mančić	upravnik	VSS	6789	1993-08-12 00:00:00	NULL	2400	10
5696	Miriana	Dimić	čistač	KV	5662	1991-09-30 00:00:00	0	1000	10
5780	Božidar	Ristić	upravnik	VSS	6789	1984-08-11 00:00:00	NULL	2200	20
5786	Pavle	Šotra	upravnik	VSS	6789	1983-05-22 00:00:00	NULL	2800	30
5842	Miloš	Marković	direktor	VSS	NULL	1981-12-15 00:00:00	NULL	3000	40
5867	Svetlana	Grubač	savetnik	VSS	5842	1970-08-08 00:00:00	NULL	2750	40
5874	Tomislav	Bogovac	električar	KV	5662	1971-04-19 00:00:00	1100	1000	10
5898	Andriia	Ristić	nabavliač	KV	5786	1980-01-20 00:00:00	1200	1100	30
5900	Slobodan	Petrović	vozač	KV	5780	2002-10-03 00:00:00	1300	900	20
5932	Mitar	Vuković	savetnik	VSS	5842	2000-03-25 00:00:00	NULL	2600	20
5953	Jovan	Perić	nabavliač	KV	5786	1979-01-12 00:00:00	0	1100	30
6234	Marko	Nastić	analitičar	VSS	5867	1990-12-17 00:00:00	3000	1300	30
6789	Janko	Simić	upravnik	VSS	5842	2003-12-23 00:00:00	10	3900	40
7890	Ivan	Buha	analitičar	VSS	5867	2003-12-17 00:00:00	3200	1600	20
7892	Luka	Bošković	analitičar	VSS	5867	2004-05-20 00:00:00	NULL	2000	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

radnik 1 | Apply

Podupiti - primer

- To se postiže **ugnježđenjem rezultata jednog upita** u **WHERE** klauzulu **drugog upita**.

- **Unutrašnji upit**, treba da iz tabele ODELJENJE pruži odgovor u vidu šifre odeljenja (id_odeljenja) koje je locirano na Dorćolu.

Podupiti - primer

Ovaj upit vraća šifru odeljenja koje je smešteno na Dorćolu:

```
SELECT id_odeljenja
FROM ODELJENJE
WHERE mesto='Dorćol';
```

Dobili smo da je šifru odeljenja=20 za odeljenje koje je smešteno na Dorćolu.

Podupiti - primer

Kada dobijemo iz tabele Odeljenje šifru odeljenja smeštenog na Dorćolu (id_odeljenja=20), onda drugim, spoljnim upitom, iz tabele RADNIK tražimo spisak imena zaposlenih u odeljenju gde je id_odeljenja=20.

Ovaj upit vraća imena zaposlenih koji rade u odeljenju 20:

```
SELECT ime, id_odeljenja
FROM RADNIK
WHERE id_odeljenja=20;
```

Podupiti - primer

```
SELECT ime, id_odeljenja
FROM RADNIK
WHERE id_odeljenja = (SELECT id_odeljenja
                      FROM ODELJENJE
                      WHERE mesto='Dorćol');
```

Podupiti - primer

Prikazati ime i posao svih radnika koji rade na Novom Beogradu

```
SELECT ime, posao
FROM RADNIK
WHERE RADNIK.id_odeljenja IN
                (SELECT ODELJENJE.id_odeljenja
                  FROM ODELJENJE
                  WHERE ODELJENJE.mesto='Novi Beograd');
```

Ako podupit vraća više od jednog zapisa, onda u uslovu klauzale WHERE treba napisati IN, a ne znak jednakosti (=).

Znak jednakosti se koristi samo u slučajevima kada je izvesno da podupit vraća samo jedan zapis.

Podupiti - primer

Prikazati id, ime, platu i kvalifikaciju radnika koji imaju istu platu kao bilo koji zaposleni čija je kvalifikacija VSS

```
SELECT id_radnika, ime, plata, kvalif
FROM RADNIK
WHERE plata = ANY (SELECT plata
                   FROM RADNIK
                   WHERE kvalif='VSS');
```

ANY operator vraća TRUE ako bilo koja od vrednosti iz podupita ispunjava uslov u WHERE klazuli spoljašnjeg upita.

Podupiti - primer

Prikazati id, ime, platu i kvalifikaciju zaposlenih koji imaju platu manju od svih zaposlenih čija je kvalifikacija VSS.

```
SELECT id_radnika, ime, plata, kvalif
FROM RADNIK
WHERE plata < ALL (SELECT plata
                   FROM RADNIK
                   WHERE kvalif='VSS');
```

Operater ALL vraća TRUE ako sve vrednosti podupita ispunjavaju uslov.

Ovaj upit vraća TRUE jer sve vrednosti generisane podupitom ispunjavaju uslov postavljen u WHERE klauzuli spoljašnjeg upita.

Podupiti - primer

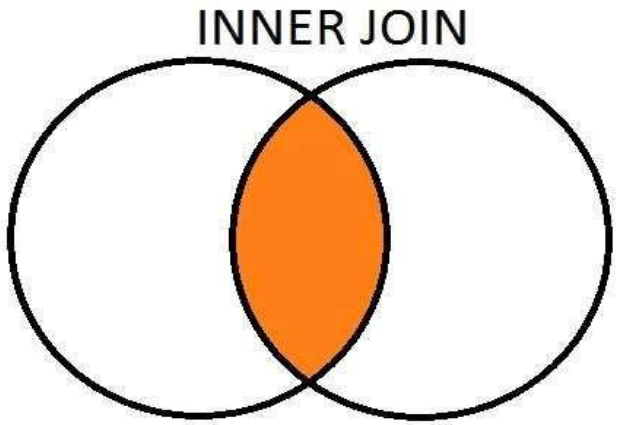
Prikazati sve podatke o odeljenjima u kojima ima zaposlenih radnika.

```
SELECT *
FROM ODELjENJE
WHERE id_odeljenja IN (SELECT id_odeljenja
                       FROM RADNIK
                       WHERE id_odeljenja is not null);
```

Spajanje tabele (JOIN)

- Join je ključna reč u SQL-u, koja označava spoj između dve tabele.
- Dolazi u kompletu sa ključnom rečju ON (ali često i nekim drugim ključnim rečima).
- Kada se u rezultatu spoljnjeg upita kombinuju podaci iz više tabela mora se izvršiti spajanje (JOIN) dveju ili više tabela.
- Spajanje tabela vrši se korišćenjem zajedničkih atributa, tj. atributa koji su definisani nad istim domenima.

Inner JOIN – Unutrašnje spajanje



- Spaja zapise na osnovu jednog ili više zajedničkih polja.
- Vraća zapise u kojima su jednake vrednosti polja preko kojeg (kojih) se vrši spajanje.
- U većini slučajeva spajanje se vrši preko polja primarnog ključa u jednoj tabeli i polja spoljnjeg ključa u drugoj tabeli (u relaciji jedan prema više).
- Ako za neku vrednost primarnog ključa jedne tabele u drugoj tabeli ne postoji ni jedan odgovarajući zapis, taj zapis se ne pojavljuje u rezultatu upita.

Inner JOIN – Unutrašnje spajanje

Prirodno spajanje (uslov spajanja tabela u WHERE klauzuli)

```
SELECT kolone
FROM tabela_1, tabela_2
WHERE uslov_spajanja;
```

INNER JOIN

```
SELECT kolone
FROM tabela_1 JOIN tabela_2
ON uslov_spajanja;
```

Inner JOIN – Unutrašnje spajanje

Da bi se dobila tražena informacija potrebno je spojiti tabele D_CLIENTS i D_EVENTS. Uslov spajanja je jednakost stranog ključa *client_number* u tabeli D_EVENTS sa primarnim ključem *client_number* u tabeli D_CLIENTS.

```
SELECT * FROM d_clients ;
```

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5922	Hiram	Peters	3715832249	hpeters@yahoo.com
5857	Serena	Jones	7035335900	serena@jones.com
6133	Lauren	Vigil	4072220090	lbv@lbv.net

```
SELECT id, name, event_date, cost, client_number
FROM d_events;
```

ID	NAME	EVENT_DATE	COST	CLIENT_NUMBER
100	Peters Graduation	04-MAY-14	8000	5922
105	Vigil wedding	04-APR-28	10000	6133

Inner JOIN – Unutrašnje spajanje

Rešenje:

```
SELECT first_name, last_name, name, event_date
FROM d_clients, d_events
WHERE d_clients.client_number=d_events.client_number;
```

ILI

```
SELECT first_name, last_name, name, event_date
FROM d_clients JOIN d_events
ON d_clients.client_number=d_events.client_number;
```

Inner JOIN – Unutrašnje spajanje

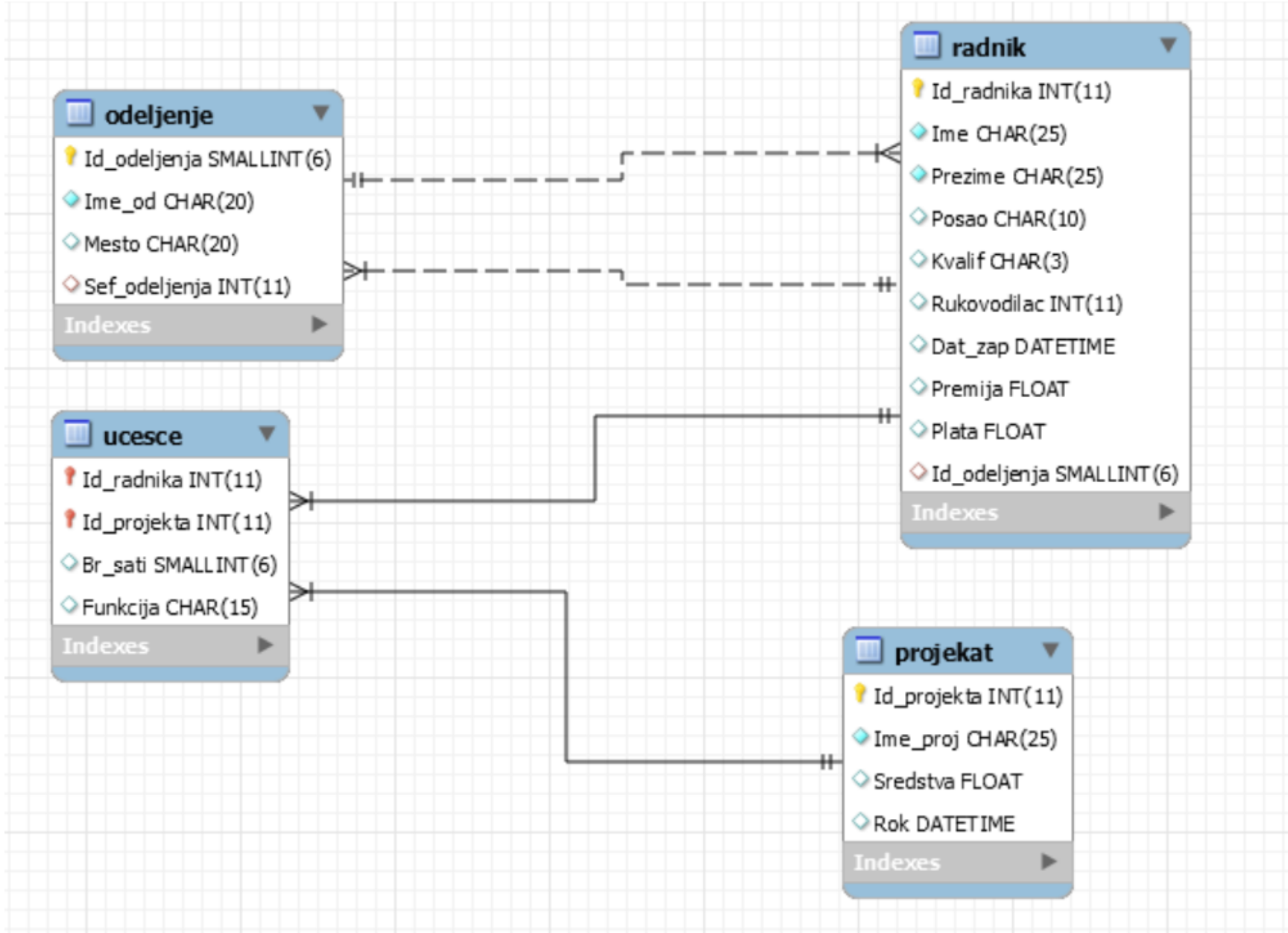
Prikazati spisak imena i prezimena zaposlenih koji rade na Banovom brdu.

```
SELECT ime, prezime
FROM radnik JOIN odeljenje
ON odeljenje.id_odeljenja=radnik.id_odeljenja
WHERE mesto='Banovo brdo’
```

```
SELECT ime, prezime
FROM radnik, odeljenje
WHERE odeljenje.id_odeljenja=radnik.id_odeljenja AND
mesto='Banovo brdo’
```

Inner JOIN – Unutrašnje spajanje – tri tabele

Prikazati za svakog radnika ime, posao i sve informacije o projektu na kome radi.


odeljenje
Id_odeljenja SMALLINT(6)
Ime_od CHAR(20)
Mesto CHAR(20)
Sef_odeljenja INT(11)
Indexes
radnik
Id_radnika INT(11)
Ime CHAR(25)
Prezime CHAR(25)
Posao CHAR(10)
Kvalif CHAR(3)
Rukovodilac INT(11)
Dat_zap DATETIME
Premija FLOAT
Plata FLOAT
Id_odeljenja SMALLINT(6)
Indexes
ucesce
Id_radnika INT(11)
Id_projekta INT(11)
Br_sati SMALLINT(6)
Funkcija CHAR(15)
Indexes
projekat
Id_projekta INT(11)
Ime_proj CHAR(25)
Sredstva FLOAT
Rok DATETIME
Indexes

Inner JOIN – Unutrašnje spajanje –tri tabele

Prikazati za svakog radnika ime, posao i sve informacije o projektu na kome radi.

```
SELECT r.ime, r.posao, p.ime_proj, p.sredstva, p.rok
FROM  radnik r JOIN ucesce u ON r.id_radnika=u.id_radnika
JOIN projekat p ON u.id_projekta=p.id_projekta
ORDER BY r.ime
```

Inner JOIN – Unutrašnje spajanje – tri tabele

Kako se realizuje INNER JOIN sa tri tabele?

table - doctors

docid	dname
1	A.VARMA
2	D.GOMES

table - specialize

spid	desc	docid
1	special1	1
2	special2	2

table - timeschedule

tid	tday	sit_time	docid
1	MON	17:00:00	1
2	WED	08:00:00	1
3	TUE	16:00:00	2
4	FRI	09:00:00	2

Inner JOIN – Unutrašnje spajanje – tri tabele

Kako se realizuje INNER JOIN sa tri tabele?

Tabele doctors – podaci o lekarima (*docid – primarni ključ)*
Tabela *specialize* – podaci o oblastima specijalizacije lekara (*spid – primarni ključ)*
Tabela *timeschedule* - podaci o rasporedu lekara u ordinaciji. (*tid – primarni ključ)*
Definisan je spoljni ključ *docid* u *specialize* i *timeschedule* tabeli koji referencira na primarni ključ *docid* u *doctors* tabeli.

Treba da prikažemo zapise o lekarima koji su specijalizovani za oblast *special1* i po rasporedu su sredom u predviđeno vreme u svojoj ordinaciji.

Inner JOIN – Unutrašnje spajanje – tri tabele

Kako se realizuje INNER JOIN sa tri tabele?

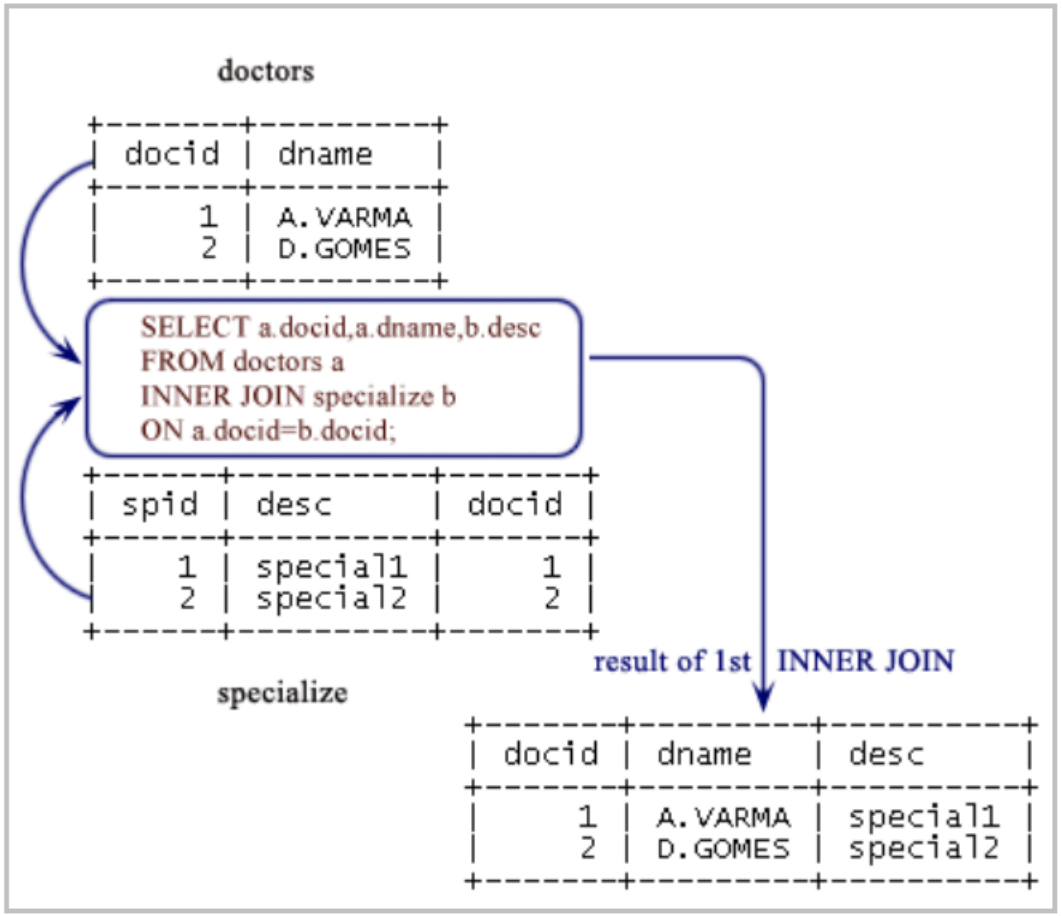
```
SELECT a.docid,a.dname, b.desc,c.tday,c.sit_time
FROM doctors a INNER JOIN specialize b ON a.docid=b.docid
INNER JOIN timeschedule c ON a.docid=c.docid
WHERE a.docid=1 AND c.tday='WED';
```

```
+-------+---------+----------+------+----------+
| docid | dname   | desc     | tday | sit_time |
+-------+---------+----------+------+----------+
|     1 | A.VARMA | special1 | WED  | 08:00:00 |
+-------+---------+----------+------+----------+
```

Inner JOIN – Unutrašnje spajanje – tri tabele

Korak 1

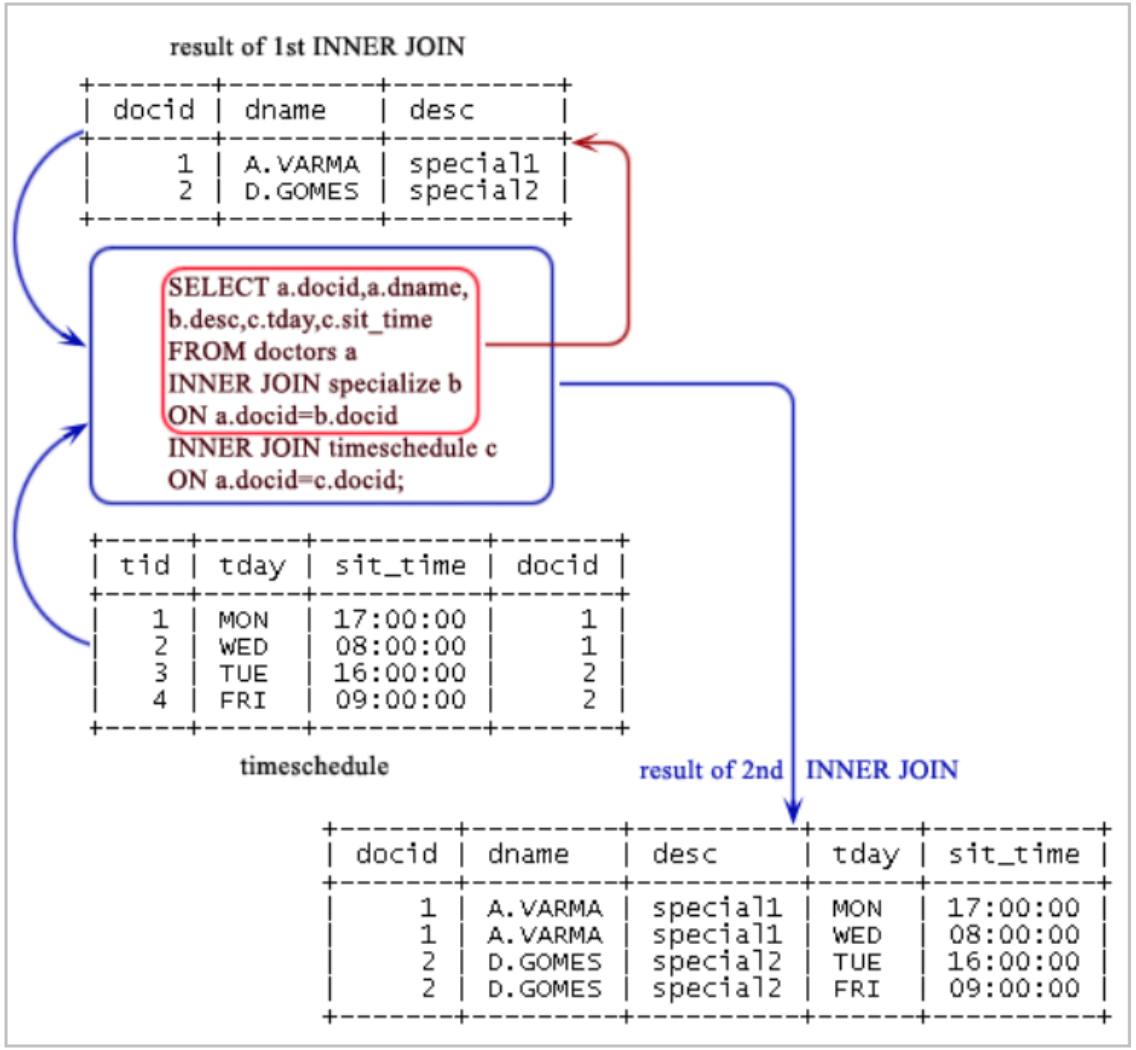
```
SELECT a.docid,a.dname,b.desc
FROM doctors a INNER JOIN specialize b
ON a.docid=b.docid;
```



Inner JOIN – Unutrašnje spajanje – tri tabele

Korak 2

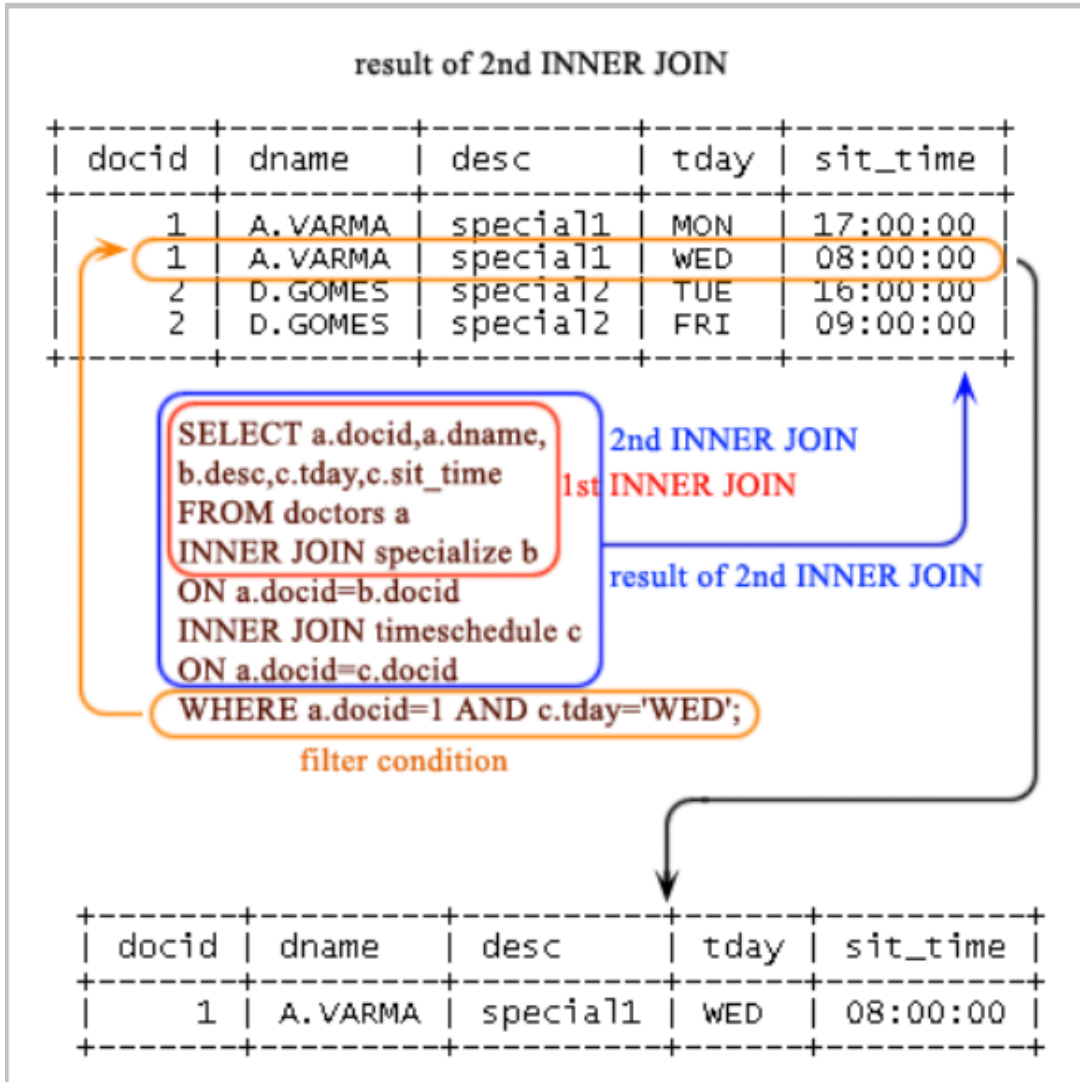
```
SELECT a.docid,a.dname,
b.desc,c.tday,c.sit_time
FROM doctors a
INNER JOIN specialize b
ON a.docid=b.docid
INNER JOIN timeschedule c
ON a.docid=c.docid;
```



Inner JOIN – Unutrašnje spajanje – tri tabele

Korak 3

```
SELECT a.docid,a.dname,
b.desc,c.tday,c.sit_time
FROM doctors a
INNER JOIN specialize b
ON a.docid=b.docid
INNER JOIN timeschedule c
ON a.docid=c.docid
WHERE a.docid=1 AND
c.tday='WED';
```



SELF JOIN – Spajanje tabele sa samom sobom

- Pri prevođenju konceptualnog modela u relacioni model **rekurzivna veza se transformiše u kolonu strani ključ koji ukazuje na primarni ključ iste tabele.**
- Iz takve tabele se mogu dobiti podaci o instancama entiteta povezanih rekurzivnom vezom. Za to je potrebno spojiti tabelu samu sa sobom.
- Spajanje tabele sa samom sobom se naziva **Self Join**.

SELF JOIN – Spajanje tabele sa samom sobom

- **Tabela se može spajati i sa samom sobom (SELF JOIN), kada unutar tabele postoji relacija između pojedinih n-torki.**
- To je slučaj sa tabelom RADNIK jer su neki zaposleni rukovodioci nekim drugim radnicima.
- Ova vrsta spajanja odnosi se na podatke u jednoj tabeli kada unutar objekata jedne tabele postoji relacija 1:1 ili 1:N.
- Spoj tabele sa samom sobom pravi tako što se upitu dodaje duplikat tabele i spajaju se polja iz originala i kopije tabele.

SELF JOIN – Spajanje tabele sa samom sobom

Ako se koristi Self Join, onda će se isti naziv tabele pojaviti dva puta u istoj klauzuli FROM. Zato je kod Self Join-a neophodno koristiti dva različita alijasa za istu tabelu.
Na primer, ako je potrebno prikazati spisak svih zaposlenih iz tabele EMPLOYEES sa imenima njihovih rukovodilaca, onda ta tabela ima dvostuku ulogu:

- S jedne strane, ona sadrži podatke o svakom zaposlenom. Za tu ulogu koristiće se alijas, npr. **emp**;

- S druge strane, ona sadrži podatke o rukovodiocima referenciranim vrednostima stranog ključa iz iste tabele. Za tu ulogu koristiće se drugi alijas iste tabele, npr. **mng**.

SELF JOIN – Spajanje tabele sa samom sobom

Primer:

```
SELECT concat(emp.first_name,' ',emp.last_name) as "Zaposleni",
concat(mng.first_name,' ', mng.last_name) as "Rukovodilac"
FROM employees emp JOIN employees mng
ON mng.employee_id = emp.manager_id;
```

Zaposleni	Rukovodilac
Alexander Hunold	Lex De Haan
Bruce Ernst	Alexander Hunold
Curtis Davies	Kevin Mourgos
Diana Lorentz	Alexander Hunold
Eleni Zlotkey	Steven King
Ellen Abel	Eleni Zlotkey
...	...

Spajanje po jednakosti (Equijoin)

Spajanje tabela gde je uslov spajanja jednakost kolona,

tabela_1.kolona_m= tabela_2.kolona_n,

naziva se ekvi-spajanje (spajanje po jednakosti) odnosno **Equi Join**.

Prirodno spajanje (Natural Join)

Ako kolone koje se izjednačavaju pri spajanju tabela, imaju isti naziv u obe tabele, takvo ekvi-spajanje se naziva **prirodno spajanje**, odnosno **Natural Join**.

Na primer:
d_clients.client_number = d_events.client_number

Ako se u klauzuli FROM koristi ključna reč NATURAL JOIN, onda je klauzula sa uslovom spajanja suvišna jer sistem sam traži kolone sa istim nazivima u obe tabele i spaja tabele po jednakosti tih kolona.

Prirodno spajanje (Natural Join)

Primer:

SELECT concat(ime, ' ', prezime) as "Radnik", posao, ime_od FROM radnik NATURAL JOIN odeljenje;

Radnik	posao	ime_od
Aleksandar Marić	električar	Komercijala
Vanja Kondić	prodavac	Komercijala
Jovan Perić	električar	Komercijala
Janko Mančić	upravnik	Komercijala
Mirjana Dimić	čistač	Komercijala
Tomislav Bogovac	električar	Komercijala
Petar Vasić	vozač	Plan
Božidar Ristić	upravnik	Plan
Slobodan Petrović	vozač	Plan
Mitar Vuković	savetnik	Plan
Ivan Buha	analitičar	Plan
Pavle Šotra	upravnik	Prodaja
Andrija Ristić	nabavljač	Prodaja
Jovan Perić	nabavljač	Prodaja
Marko Nastić	analitičar	Prodaja
Miloš Marković	direktor	Direkcija

Prirodno spajanje i klauzula USING

Ako više kolona iz jedne tabele ima iste nazive kao kolone iz druge tabele, onda NATURAL JOIN poredi vrednosti svih parova kolona sa istim nazivom i spaja one redove iz te dve tabele kojima svi parovi kolona sa istim nazivom imaju jednake vrednosti.

Ponekad to nije dobro zato što:

- moguće je da priroda upita koji se kreira ne zahteva izjednačavanje svih tih kolona;
- može da se desi da kolone sa istim nazivom nemaju isti tip podataka pa se nad njima ne može izvršiti operacija poređenja po jednakosti.

Prirodno spajanje i klauzula USING

Da bi se to izbeglo, koristi se običan JOIN sa klauzulom USING.

```
SELECT kolone
FROM tabela_1 JOIN tabela_2
USING (kolona ili kolone);
```

Kolone koje se koriste u klauzuli USING ne treba da imaju kvalifikator (ime tabele ili alijas tabele) bilo gde u SQL rečenici.

Prirodno spajanje i klauzula USING

Primer:

Prikazati imena i prezimena zaposlenih, kao i nazive i identifikacione brojeve odeljenja u kojima rade. Koristiti tabele EMPLOYEES i DEPARTMENTS.

EMPLOYEES

Column Name	Data Type
EMPLOYEE_ID	NUMBER(6,0)
FIRST_NAME	VARCHAR2(20)
LAST_NAME	VARCHAR2(25)
EMAIL	VARCHAR2(25)
PHONE_NUMBER	VARCHAR2(20)
HIRE_DATE	DATE
JOB_ID	VARCHAR2(10)
SALARY	NUMBER(8,2)
COMMISSION_PCT	NUMBER(2,2)
MANAGER_ID	NUMBER(6,0)
DEPARTMENT_ID	NUMBER(4,0)

DEPARTMENTS

Column Name	Data Type
DEPARTMENT_ID	NUMBER(4,0)
DEPARTMENT_NAME	VARCHAR2(30)
MANAGER_ID	NUMBER(6,0)
LOCATION_ID	NUMBER(4,0)

Prirodno spajanje i klauzula USING

```
SELECT last_name, first_name, department_id, department_name
FROM employees JOIN departments
USING (department_id);
```

LAST_NAME	FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	Jennifer	10	Administration
Hartstein	Michael	20	Marketing
Fay	Pat	20	Marketing
Davies	Curtis	50	Shipping
...			

Prirodno spajanje i klauzula USING

ILI

```
SELECT  concat (last_name,' ’,first_name)  AS "Zaposleni",
concat(department_id,' - ’,department_name)  AS "Organizaciona jedinica"
FROM employees JOIN departments
USING (department_id);
```

Zaposleni	Organizaciona jedinica
Whalen, Jennifer	10 - Administration
Hartstein, Michael	20 - Marketing
Fay, Pat	20 - Marketing
Davies, Curtis	50 - Shipping
...	

Prirodno spajanje i klauzula USING

Primer:

Prikazati spisak zaposlenih koji rade u računovodstvu.

```
SELECT concat(ime, ' ', prezime) as Zaposleni, posao, ime_od as OrganizacionaJedinica
FROM radnik JOIN odeljenje
USING (id_odeljenja)
WHERE Ime_od='Komercijala'
```

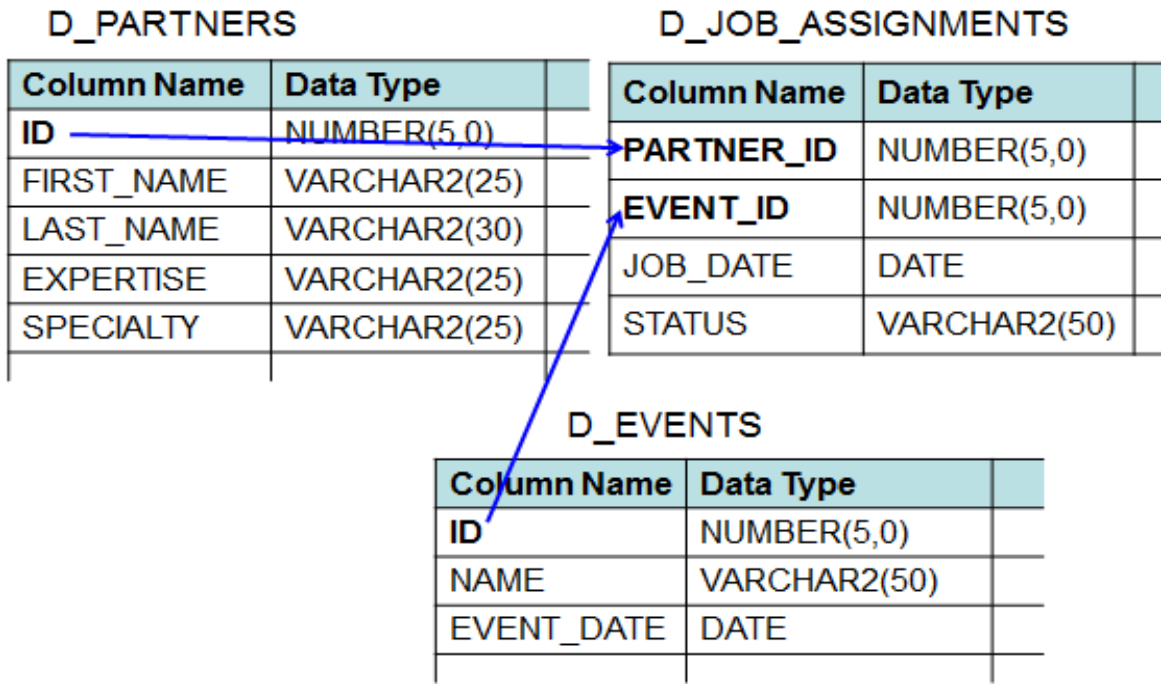
Zaposleni	posao	OrganizacionaJedinica
Aleksandar Marić	električar	Komercijala
Vanja Kondić	prodavac	Komercijala
Jovan Perić	električar	Komercijala
Janko Mančić	upravnik	Komercijala
Mirjana Dimić	čistač	Komercijala
Tomislav Bogovac	električar	Komercijala

Prirodno spajanje i klauzula USING

Primer:

Prikazati spisak DJ-partnera i događaja na kojima rade.

Strukture tabela D_PARTNERS, D_EVENTS i vezne tabele D_JOB_ASSIGNMENTS:



Prirodno spajanje i klauzula USING

Rešenje:

```
SELECT first_name, last_name, name, event_date
FROM (d_partners JOIN d_job_assignments ON d_partners.id = d_job_assignments.partner_id)
JOIN d_events
ON d_job_assignments.event_id = d_events.id;
```

FIRST_NAME	LAST_NAME	NAME	EVENT_DATE
Jennifer	cho	Vigil wedding	28-APR-04

ILI:

```
SELECT first_name, last_name, name, event_date
FROM d_partners , d_job_assignments, d_events
WHERE d_partners.id = d_job_assignments.partner_id AND
        d_job_assignments.event_id = d_events.id;
```

Non-Equijoin

Ako se kod uslova spajanja koriste operatori nejednakosti kao što su <, > ili BETWEEN, onda se uslov spajanja navodi u klauzuli ON.

Primer:

Prikazati sve zaposlene (iz tabele RADNIK) iz odeljenja 20 koji imaju veću platu od nekog ili nekih radnika iz odeljenja 10.

```
SELECT vp.ime, vp.plata, vp.id_odeljenja, mp.ime, mp.plata,  mp.id_odeljenja
FROM radnik vp JOIN radnik mp
ON vp.plata>mp.plata
WHERE vp.id_odeljenja=20 and mp.id_odeljenja=10 ;
```

Non-Equijoin

Tabela RADNIK spojena je sa samom sobom pri čemu su korišćeni alijasi:

vp (veća plata) → zaposleni sa većom platom,

mp (manja plata) → zaposleni sa manjom platom.

Dobijeni rezultati pokazuju da Mitar iz odeljenja 20 ima veću platu nego Jovan, Mirjana i Tomislav iz odeljenja 10 ....

ime	plata	id_odeljenja	ime	plata	id_odeljenja
Mitar	2600	20	Jovan	1000	10
Ivan	1600	20	Jovan	1000	10
Mitar	2600	20	Janko	2400	10
Petar	1300	20	Mirjana	1000	10
Božidar	2200	20	Mirjana	1000	10
Mitar	2600	20	Mirjana	1000	10
Ivan	1600	20	Mirjana	1000	10
Petar	1300	20	Tomislav	1000	10
Božidar	2200	20	Tomislav	1000	10
Mitar	2600	20	Tomislav	1000	10
Ivan	1600	20	Tomislav	1000	10

Spoljašnje spajanje tabela (Outer Join)

Unutrašnje spajanje dve tabele (Inner join) vraća podatke koji zadovoljavaju uslov spajanja.

Ponekad je potrebno prikazati redove jedne ili obe tabele koji se, prema datom uslovu spajanja, ne mogu spojiti ni sa jednim redom druge tabele.

U **rezultat spajanja se uključuju i one n-torke koje ne zadovoljavaju uslov spajanja**, nemaju parnjaka u obe tabele, **ali zadovoljavaju uslov iz WHERE odredbe.**

Spoljašnje spajanje tabela (Outer Join)

Koristi se kod održavanja baze podataka

- da bi se iz tabele uklonili zapisi “siročići” (zapisi koji nisu u relaciji)
- li duplikati podataka, tako što se pravi nova tabela.

Funkcioniše tako što prikazuje:

- sve zapise iz jedne tabele koji zadovoljavaju postavljeni uslov i imaju parnjaka u drugoj tabeli (spajanje po jednakosti),

- sve one zapise iz te tabele za koje u drugoj tabeli koja je član spoja ne postoje odgovarajući zapisi.

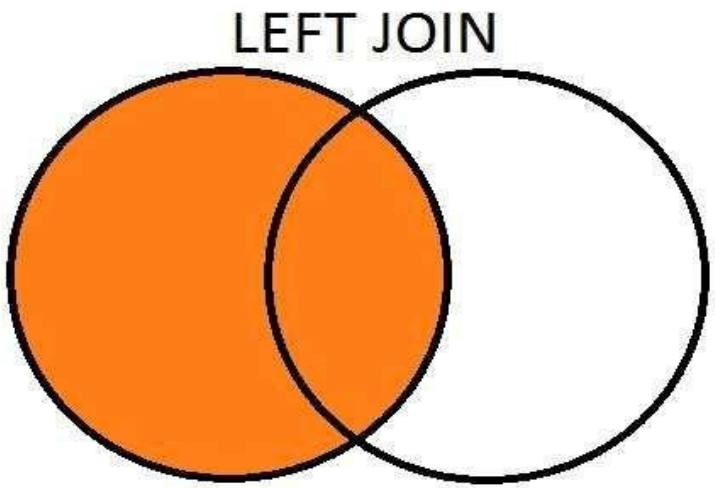
Spoljašnje spajanje tabela (Outer Join)

Spoljašnje spajanje tabela se javlja u tri oblika:

- **LEFT OUTER JOIN (levo spoljašnje spajanje)**
- **RIGHT OUTER JOIN (desno spoljašnje spajanje)**
- **FULL OUTER JOIN (puno spoljašnje spajanje)**

Left JOIN

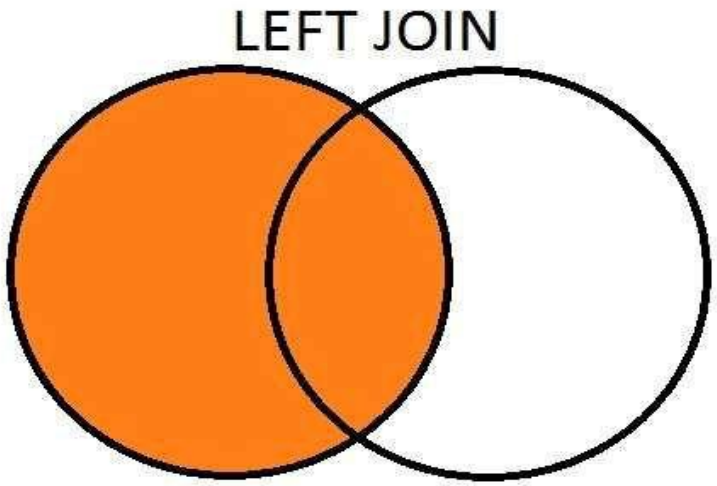
leva_tabela **LEFT JOIN** desna_tabela



- **LEFT JOIN vraća sve zapise iz tabele koju u spoju proglasimo kao "LEVU",** odnosno koja je prva navedena u izrazu za spajanje.
- „LEVA" tabela predstavlja tabelu postavljenu levo od ključne reči LEFT JOIN
- Vraća sve redove leve tabele spojene sa odgovarajućim redovima desne tabele za koje je uslov spajanja zadovoljen i redove iz leve tabele koji nemaju par u desnoj (redovi koji nemaju svoj par spajaju se sa null vrednostima).

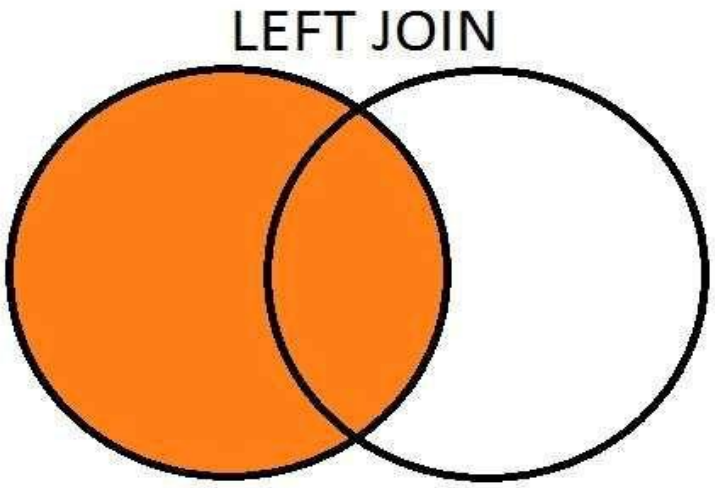
Left JOIN

Prikazati **nazive odeljenja**, ime i posao svakog radnika koji u njima rade, **uključujući i odeljenja** u kojima **nema raspoređenih radnika**.




```
SELECT ime_od, ime, posao from
odeljenje left join radnik
on odeljenje.id_odeljenja=radnik.id_odeljenja
```

Left JOIN



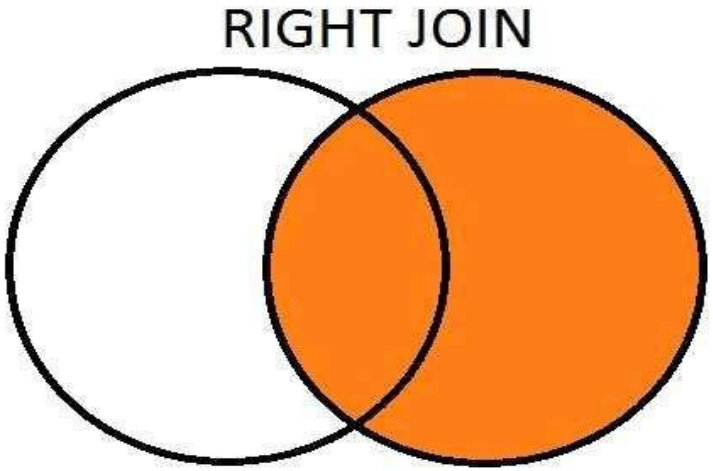
Dobijeni rezultati pokazuju da u odeljenjima Nabavka i Računski centar nema raspoređenih radnika.

To ne bi bilo tako vidljivo da je korišćeno unutrašnje spajanje tabela.

ime_od	ime	posao
Direkcija	Svetlana	savetnik
Direkcija	Janko	upravnik
Komercijala	Aleksandar	električar
Komercijala	Vanja	prodavac
Komercijala	Jovan	električar
Komercijala	Janko	upravnik
Komercijala	Mirjana	čistač
Komercijala	Tomislav	električar
Nabavka	NULL	NULL
Plan	Petar	vozač
Plan	Božidar	upravnik
Plan	Slobodan	vozač
Plan	Mitar	savetnik
Plan	Ivan	analitičar
Prodaja	Pavle	upravnik
Prodaja	Andrija	nabavljač
Prodaja	Jovan	nabavljač
Prodaja	Marko	analitičar
Računski c...	NULL	NULL

Right JOIN

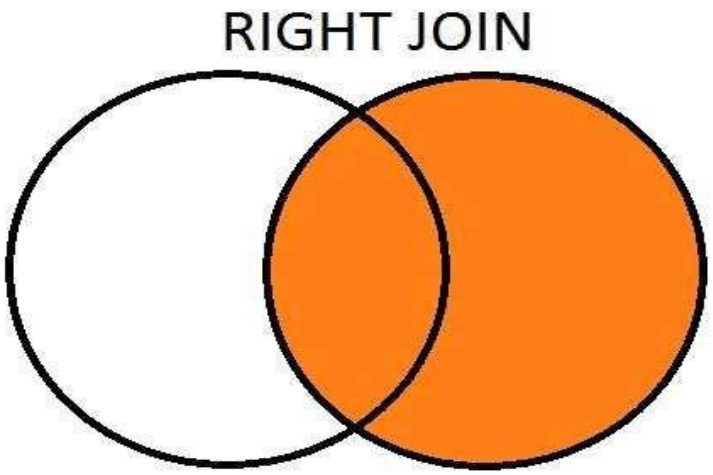
leva_tabela **RIGHT JOIN** desna_tabela



- **RIGHT JOIN vraća sve zapise iz tabele koju u spoju proglasimo ka“DESNU”,** bez obzira na to da li se odgovarajući zapisi nalaze u “levoj” tabeli.
- Vraća sve redove desne tabele spojene sa odgovarajućim redovima leve tabele za koje je uslov spajanja zadovoljen i redove iz desne tabele koji nemaju par u levoj (redovi koji nemaju svoj par spajaju se sa null vrednostima).

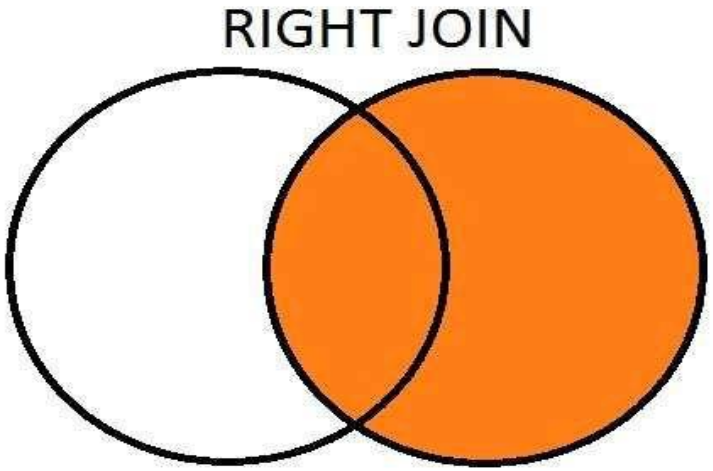
Right JOIN

Prikazati nazive odeljenja, ime i posao svakog radnika koji u njima rade, **uključujući i radnike koji nisu raspoređeni ni u jednom odeljenju**.




```
SELECT ime_od, ime, posao
from odeljenje right join radnik
on odeljenje.id_odeljenja=radnik.id_odeljenja
```

Right JOIN



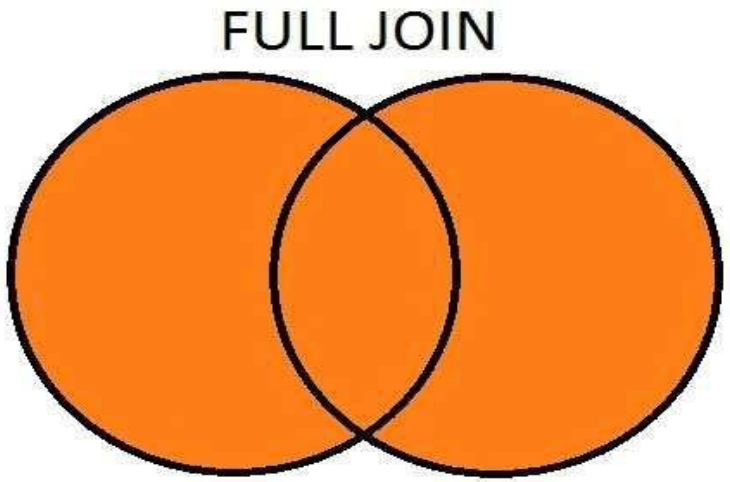
Dobijeni rezultati pokazuju da radnik Luka nije raspoređen ni u jedno odeljenje.

To ne bi bilo tako vidljivo da je korišćeno unutrašnje spajanje tabela.

ime_od	ime	posao
Komercijala	Aleksandar	električar
Komercijala	Vanja	prodavac
Komercijala	Jovan	električar
Komercijala	Janko	upravnik
Komercijala	Mirjana	čistač
Plan	Božidar	upravnik
Prodaja	Pavle	upravnik
Direkcija	Miloš	direktor
Direkcija	Svetlana	savetnik
Komercijala	Tomislav	električar
Prodaja	Andrija	nabavljač
Plan	Slobodan	vozač
Plan	Mitar	savetnik
Prodaja	Jovan	nabavljač
Prodaja	Marko	analitičar
Direkcija	Janko	upravnik
Plan	Ivan	analitičar
NULL	Luka	analitičar

FULL JOIN

leva_tabela **FULL JOIN** desna_tabela



- Vraća sve redove leve tabele spojene sa odgovarajućim redovima desne tabele za koje je uslov spajanja zadovoljen, redove iz leve tabele koji nemaju par u desnoj, kao i redove iz desne tabele koji nemaju par u levoj (redovi koji nemaju svoj par spajaju se sa null vrednostima).

- MySQL ne podržava FULL JOIN. Realizuje se kombinovanjem LEFT i RIGHT JOIN primenom UNION skupovnog operatora.

Prikazati imena radnika koji nemaju premiju i rade na projektu izvoz. Iz prikaza isključiti savetnike.

```
SELECT  ime FROM  radnik
WHERE  id_radnika IN
   (SELECT  id_radnika  FROM  ucesce
     WHERE  id_projekta IN
      (SELECT  id_projekta   FROM    projekat
            WHERE ime_proj = 'izvoz’))
AND premija IS NULL  AND posao <> 'savetnik'
```

Prikazati imena radnika sa Novog Beograda koji rade na projektima uvoz i izvoz

```
SELECT ime FROM radnik
WHERE id_odeljenja IN
  (SELECT id_odeljenja FROM odeljenje WHERE mesto='Novi
  Beograd’)
AND  id_radnika IN
 (SELECT id_radnika FROM ucesce
    WHERE id_projekta IN
       (SELECT id_projekta FROM projekat WHERE ime_proj IN ('uvoz',
'izvoz')))
```

Prikazati imena radnika koji imaju istog rukovodioca kao Petar

```
SELECT ime FROM radnik
where rukovodilac IN
  (select rukovodilac from radnik  where  ime='Petar')
and ime<>'Petar’
```

Ko su najbolje plaćeni radnici u svakom odeljenju?

```
SELECT id_odeljenja, ime, plata
FROM radnik WHERE plata IN
    (SELECT max(plata) FROM radnik
      GROUP BY id_odeljenja)
AND id_odeljenja IS NOT NULL
```

Prikaži imena radnika koji su se poslednji zaposlili u svakom odeljenju.

```
SELECT id_odeljenja, ime
FROM radnik WHERE dat_zap IN
        (SELECT max(dat_zap)  FROM radnik
        GROUP BY id_odeljenja)
AND id_odeljenja IS NOT NULL
```

Prikazati ime, posao i platu zaposlenih u odeljenju 10, koji imaju isti posao kao zaposleni u odeljenju Plan.

```
SELECT ime, posao, plata
FROM RADNIK
WHERE Id_odeljenja = 10 AND posao IN
        (SELECT posao FROM radnik WHERE
          Id_odeljenja =  (SELECT Id_odeljenja FROM odeljenje
                  WHERE ime_od = 'plan'))
```

Prikazati ime i ukupna primanja svih zaposlenih koji imaju isti posao kao Slobodan.

```
SELECT ime, plata + IFNULL(premija,0) AS 'Ukupna primanja’
FROM RADNIK
WHERE posao IN
                (SELECT posao FROM radnik WHERE Ime = 'Slobodan')
```

Prikazati ime i kvalifikaciju svih radnika koji rade na istim projektima kao Marko

```
SELECT ime, kvalif FROM RADNIK
WHERE Id_radnika IN
    (SELECT id_radnika FROM ucesce WHERE id_projekta in
            (SELECT id_projekta FROM ucesce WHERE id_radnika in
                (SELECT id_radnika FROM radnik WHERE ime='Marko')))
```

Prikazati ime i primanja svih zaposlenih čija su primanja veća od prosečnih primanja u preduzeću.

```
SELECT ime, (plata + IFNULL(premija,0)) AS 'UKUPNA PRIMANJA’
FROM RADNIK
WHERE plata + IFNULL(premija,0)>
   (SELECT AVG(plata + IFNULL(premija,0)) FROM radnik)
```

Prikazati imena i poslove radnika, kao i broj i imena projekata na kojima rade uključujući i projekte na kojima ne radi ni jedan radnik

```
SELECT ime, posao, projekat.Id_projekta, Ime_proj
FROM radnik JOIN ucesce ON
radnik.Id_radnika=ucesce.Id_radnika
RIGHT JOIN projekat ON
projekat.Id_projekta=ucesce.Id_projekta
```

Prikazati imena i poslove radnika samo za radnike koji ne rade ni na jednom projektu.

```
SELECT radnik.ime, radnik.posao, ucesce.Id_projekta
FROM radnik LEFT JOIN ucesce
ON ucesce.Id_radnika = radnik.Id_radnika
WHERE ucesce.Id_projekta IS NULL
```

Prikazati ime i primanja radnika i njihovih neposrednih rukovodilaca za one radnike koji imaju veća primanja od svojih neposrednih rukovodilaca.

```
SELECT R.Ime AS 'Ime radnika', R.plata+ifnull(R.premija,0) AS
'Primanja radnika', R1.Ime AS 'Ime rukovodioca',
R1.plata+ifnull(R1.premija,0) AS 'Primanja rukovodioca’
    FROM RADNIK R INNER JOIN RADNIK R1
    ON R.Rukovodilac = R1.ID_radnika
    WHERE
    (R.plata+ifnull(R.premija,0))>(R1.plata+ifnull(R1.premija,0))
```