Napredne računarske aplikacije

Predavanje broj: 08

Nastavne teme:

- ✓PODUPITI
  - ✓Podupiti u WHERE klauzuli
  - ✓Korelisani podupiti
  - ✓Podupiti u FROM klauzuli
- ✓AŽURIRANJE PODATAKA
  - ✓INSERT
  - ✓UPDATE
  - ✓DELETE

Podupiti u WHERE klauzuli

- Ako podupit vraća null vrednost ili nijedan red, spoljni upit uzima rezultat podupita (null) i koristi ovaj rezultat u WHERE klauzuli.

- Spoljni upit neće vratiti nijedan red, pošto poređenje bilo koje vrednosti sa null uvek daje null

Primer:

Ko radi u istom odeljenju kao Luka Bošković?

Podupiti u WHERE klauzuli

- Lukino odeljenje je null, tako da podupit vraća null
- Spoljni upit zamenjuje ovu vrednost u WHERE izrazu (WHERE id_odeljenja = NULL)
- Spoljni upit ne vraća nikakav red, pošto poređenje bilo čega sa null vraća null

```
select ime, prezime
from radnik
where id_odeljenja=(select Id_odeljenja from radnik where ime='Luka')
```

Podupiti u WHERE klauzuli

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

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata, **Id_odeljenja$**>

```
select ime, prezime
from radnik
where plata = ANY
    (select plata from radnik where kvalif='VSS')
and kvalif<>'VSS'
```

I način: Upotreba ANY operatora

```
select ime, prezime
from radnik
where plata IN
    (select plata from radnik where kvalif='VSS')
and kvalif<>'VSS'
```

II način: Upotreba IN operatora

Podupiti u WHERE klauzuli

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

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata, **Id_odeljenja$**>

```
12 • select ime, prezime
13   from radnik
14   where plata<(select plata from radnik where kvalif='VSS')
15   and kvalif<>'VSS'
```

Output

Action Output

#	Time	Action	Message
1	00:30:52	select ime, prezime from radnik where plata<(select plata from radnik where kvalif...	Error Code: 1242. Subquery returns more than 1 row

Podupiti u WHERE klauzuli

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

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata, **Id_odeljenja$**>

I način: Upotreba ANY operatora

```
select ime, prezime
from radnik
where plata < ANY
    (select plata from radnik where kvalif='VSS')
and kvalif<>'VSS'
```

II način: Upotreba ALL operatora

```
select ime, prezime
from radnik
where plata < ALL
    (select plata from radnik where kvalif='VSS')
and kvalif<>'VSS'
```

Da li se dobije isti rezultat?

Podupiti u WHERE klauzuli

```
1 • SELECT plata
2   from radnik
3   where kvalif<>'VSS'
```

Result Grid | Filter Rows:

plata
1300
1000
1200
1000
1000
1000
1100
900
1100

Podupiti u WHERE klauzuli

Rezultat sa primenom ANY operatora

ime	prezime	plata	kvalif
Petar	Vasić	1300	KV
Aleksandar	Marić	1000	KV
Vanja	Kondić	1200	VKV
Jovan	Perić	1000	KV
Mirjana	Dimić	1000	KV
Tomislav	Bogovac	1000	KV
Andrija	Ristić	1100	KV
Slobodan	Petrović	900	KV
Jovan	Perić	1100	KV

Rezultat sa primenom ALL operatora

ime	prezime	plata	kvalif
Aleksandar	Marić	1000	KV
Vanja	Kondić	1200	VKV
Jovan	Perić	1000	KV
Mirjana	Dimić	1000	KV
Tomislav	Bogovac	1000	KV
Andrija	Ristić	1100	KV
Slobodan	Petrović	900	KV
Jovan	Perić	1100	KV

Podsećanje: ALL operator se koristi kada želimo da WHERE izraz spoljnog upita bira redove koji odgovaraju kriterijumu (<, >, =...) **za sve vrednosti koje vrati podupit**.
ALL operator upoređuje vrednosti sa svakom vrednosti vraćenom od strane unutrašnjeg podupita.

Podupiti u WHERE klauzuli

Prikazati prezimena, imena, kvalifikacije, datume zaposlenja i šifre odeljenja radnika koji ne rade u odeljenju 10, ali su se zaposlili **istog datuma kad i neki radnik iz odeljenja 10.**

```
3 •  select ime, prezime, kvalif, dat_zap, id_odeljenja
4    from radnik
5    where id_odeljenja <> 10 and dat_zap = ANY
6          (select dat_zap
7            from radnik where id_odeljenja=10)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime	prezime	kvalif	dat_zap	id_odeljenja
Ivan	Buha	VSS	2003-12-17 00:00:00	20

Podupiti u WHERE klauzuli

Prikazati prezimena, imena, kvalifikacije, datume zaposlenja i šifre odeljenja radnika koji ne rade u odeljenju 10, ali su se zaposlili **istog datuma kad i neki radnik iz odeljenja 10.**

```
3 •  select ime, prezime, kvalif, dat_zap, id_odeljenja
4    from radnik
5    where id_odeljenja <> 10 and dat_zap = SOME
6          (select dat_zap
7            from radnik where id_odeljenja=10)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime	prezime	kvalif	dat_zap	id_odeljenja
Ivan	Buha	VSS	2003-12-17 00:00:00	20

Operator **ANY** može biti zamenjen operatorom **SOME**!

Podupiti u WHERE klauzuli

Prikazati šifre odeljenja, prezimena, imena i godinu zaposlenja radnika koji ne rade u odeljenju 10, ali su se **zaposlili iste godine kad i neki od radnik iz odeljenja 10**.

```
3 •  select id_odeljenja, prezime,ime,EXTRACT(year from dat_zap) as 'Godina zaposlenja'
4    from radnik
5    where EXTRACT(year from dat_zap) = ANY (select EXTRACT(year from dat_zap)
6                                            from radnik where id_odeljenja=10)
7    and  id_odeljenja<>10
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_odeljenja	prezime	ime	Godina zaposlenja
30	Ristić	Andrija	1980
30	Nastić	Marko	1990
40	Simić	Janko	2003
20	Buha	Ivan	2003

Unutrašnji upit će vratiti listu godina zaposlenja radnika iz odeljenja 10.
Spoljnji upit će vratiti svakog zaposlenog koji je zaposlen godine koja odgovara godinama u listi unutrašnjeg upita ali da nije iz odeljenja 10.

Podupiti u WHERE klauzuli

Prikazati prezimena, imena i godinu zaposlenja svakog radnika čija **je godina zaposlenja manja od najmanje jedne godine zaposlenja zaposlenih iz odeljenja 40.**

Prikaz godina zaposlenja radnika iz odeljenja 40.

```
select EXTRACT(year from dat_zap)
from radnik where id_odeljenja=40
```

EXTRACT(year from dat_zap)
1981
1970
2003

Podupiti u WHERE klauzuli

```
select id_odeljenja, prezime,ime,EXTRACT(year from dat_zap) as 'Godina zaposlenja'
from radnik
where EXTRACT(year from dat_zap) < ANY (select EXTRACT(year from dat_zap)
                                        from radnik where id_odeljenja=40)
and id_odeljenja<>40
```

Prikazani primer će vratiti svakog zaposlenog čija je godina zaposlenja manja od najmanje jedne godine zaposlenja radnika iz odeljenja 40

id_odeljenja	prezime	ime	Godina zaposlenja
20	Vasić	Petar	1978
10	Marić	Aleksandar	1990
10	Perić	Jovan	1980
10	Mančić	Janko	1993
10	Dimić	Mirjana	1991
20	Ristić	Božidar	1984
30	Šotra	Pavle	1983
10	Bogovac	Tomislav	1971
30	Ristić	Andrija	1980
20	Petrović	Slobodan	2002
20	Vuković	Mitar	2000
30	Perić	Jovan	1979
30	Nastić	Marko	1990

Podupiti u WHERE klauzuli

Da li se u prethodnom primeru mogao koristiti operator ALL umesto ANY?

Podupiti u WHERE klauzuli

```
select id_odeljenja, prezime,ime,EXTRACT(year from dat_zap) as 'Godina zaposlenja'
from radnik
where EXTRACT(year from dat_zap) < ALL(select EXTRACT(year from dat_zap)
                                       from radnik where id_odeljenja=40)
and id_odeljenja<>40
```

Prikaz godina zaposlenja radnika iz odeljenja 40.

EXTRACT(year from dat_zap)
1981
1970
2003

year(dat_zap)
1970
1971
1978
1979
1980
1981
1983
1984
1990
1991
1993
2000
2002
2003
2004

Pošto nijedan zaposleni nije zaposlen pre 1970, neće se vratiti nijedan red.

Podupiti u WHERE klauzuli

Prikazati podatke o radnicima koji rade u Novom Beogradu.

```
1  SELECT ime, prezime
2  from radnik
3  where id_odeljenja = ANY (select id_odeljenja
4                            from odeljenje
5                            where mesto='Novi Beograd')
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime	prezime
Aleksandar	Marić
Vanja	Kondić
Jovan	Perić
Janko	Mančić
Mirjana	Dimić

Podupiti u WHERE klauzuli

- Pretpostavimo da jedna od vrednosti vraćena sa višerednim podupitom je null, ali ostale nisu
- Ako IN ili ANY se koriste, spoljni upit će vratiti redove koji odgovaraju non-null vrednostima

Prikazati podatke o radnicima koji su rukovodioci.

```
select  id_radnika,prezime,ime
from radnik
where id_radnika = ANY (select rukovodilac from radnik)
```

rukovodilac
5780
5662
6789
NULL
5842
5786
5867

id_radnika	prezime	ime
5662	Mančić	Janko
5780	Ristić	Božidar
5786	Šotra	Pavle
5842	Marković	Miloš
5867	Grubač	Svetlana
6789	Simić	Janko

Podupiti u WHERE klauzuli

```
3 •  select  id_radnika,prezime,ime
4    from radnik
5    where id_radnika <= ALL (select rukovodilac from radnik)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_radnika	prezime	ime

Ako se koristi ALL, spoljni upit ne vraća redove pošto ALL upoređuje red spoljnog upita sa svakom vrednošću vraćenom od podupita, uključujući null;

Poređenje ničega sa null vraća null.

Podupiti u WHERE klauzuli

= ALL, da li može jedna vrednost da bude identična sa svakom vrednošću u skupu vrednosti ?

Podupiti iz različitih tabela

- Više od jednog podupita može vratiti informaciju spoljašnjem upitu.

- Prvi podupit vraća job_id od radnika čiji je id 141 (ST_CLERK).
- Drugi podupit koristi tabelu departments za pronalaženje department_id na location_id 1500 (50)
- Spoljni upit vraća redove iz employees tabele koji odgovaraju za obe vrednosti.

```
SELECT last_name, job_id, salary, department_id
FROM employees
WHERE job_id =
   (SELECT job_id
    FROM employees
    WHERE employee_id = 141)
AND department_id =
   (SELECT department_id
    FROM departments
    WHERE location_id = 1500);
```

Result of 1st subquery

JOB_ID
ST_CLERK

Result of 2nd subquery

DEPARTMENT_ID
50

LAST_NAME	JOB_ID	SALARY	DEPARTMENT_ID
Rajs	ST_CLERK	3500	50
Davies	ST_CLERK	3100	50
Matos	ST_CLERK	2600	50
Vargas	ST_CLERK	2500	50

- U ovom slučaju, pošto je svako odeljenje smešteno samo na jednoj lokaciji, bezbedno je koristiti jednoredni podupit. Ako odeljenje može biti na više lokacija mora se koristiti višeredni podupit.

Podupiti iz različitih tabela

Prikazati podatke o radnicima koji rade na Banovom brdu i zaposleni su posle Mirjane.

```
select ime, prezime, plata
from radnik
where id_odeljenja =
   (select id_odeljenja from odeljenje where mesto='Banovo brdo')
and year(dat_zap)>
  (select year(dat_zap) from radnik where ime='Mirjana')
```

ime	prezime	plata
Janko	Simić	3900

Podupiti iz različitih tabela

Prikazati ime i prezime zaposlenih koji rade u odeljenju u Starom Gradu, učestvuju na projektima i imaju istu kvalifikaciju kao zaposleni Ivan Buha.

```
12 • select ime, prezime
13   from radnik
14   where id_odeljenja=(select id_odeljenja from odeljenje where mesto='Stari Grad')
15   and id_radnika IN (select id_radnika from ucesce)
16   and kvalif=(select kvalif from radnik where ime='Ivan' and prezime='Buha')
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime	prezime
Pavle	Šotra
Marko	Nastić

Multiple-Column podupiti

- Spoljašnji i unutrašnji upit mogu biti povezani po vrednostima više kolona. U tom slučaju, 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. Konačno, napomenimo da atributi koji se upoređuju moraju biti istog ili kompatibilnog tipa podataka.

- Ako koriste više od jedne kolone, u pitanju su **multiple-column** podupiti.

- Multiple-column podupiti mogu biti **pair-wise** upoređivanja ili **non-pair-wise** upoređivanja

Multiple-Column podupiti

Sledeći primer predstavlja upotrebu multiple-column pair-wise podupita za prikaz podataka o zaposlenima čiji su rukovodilac i šifra odeljenja isti kao i rukovodilac i šifra odeljenja zaposlenih Tomislava ili Andrije.

```
select  id_radnika,prezime, ime
from radnik
where (rukovodilac, id_odeljenja) IN (select rukovodilac, id_odeljenja
                                      from radnik  where ime IN ('Andrija','Tomislav'))
and ime not in  ('Andrija','Tomislav')
```

Prvo podupit vraća rukovodilac i id_odeljenja vrednosti za radnike Andriju ili Tomislava.
Ove vrednosti se porede sa rukovodilac kolonom i id_odeljenja kolonom svakog reda u tabeli radnik.
Ako se vrednosti podudare, red se prikaže.

id_radnika	prezime	ime
5497	Marić	Aleksandar
5519	Kondić	Vanja
5652	Perić	Jovan
5696	Dimić	Mirjana
5953	Perić	Jovan

Multiple-Column podupiti

Non-pair-wise multiple-column podupiti takođe koriste više od jedne kolone u podupitu, ali poredi ih jednu po jednu, tako da se upoređivanja izvode u različitim podupitima.

Treba pisati jedan podupit po koloni koje treba uporediti pri izvođenju non-pair-wise multiple-column podupita.

Multiple-Column podupiti

```
select  id_radnika,prezime, ime
from radnik
where rukovodilac IN (select rukovodilac
                      from radnik  where ime IN ('Andrija','Tomislav'))
and id_odeljenja IN (select id_odeljenja
                     from radnik  where ime IN ('Andrija','Tomislav'))
and ime not in  ('Andrija','Tomislav')
```

rukovodilac
5662
5786

id_odeljenja
10
30

prezime	ime	rukovodilac	id_odeljenja
Marić	Aleksandar	5662	10
Kondić	Vanja	5662	10
Perić	Jovan	5662	10
Dimić	Mirjana	5662	10
Perić	Jovan	5786	30

Podupiti + Grupisanje

Prikazati imena odeljenja u kojima su ukupna primanja svih radnika u odeljenju veća od 3000

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata,**Id_odeljenja$**>

ODELJENJE <**#id_odeljenja**, ime_od, mesto, **sef_odeljenja$**>

```
select ime_od
from odeljenje
where id_odeljenja IN (select id_odeljenja from radnik
                       group by id_odeljenja
                       having sum(plata+IFNULL(premija,0))>3000)
```

Podupiti + Grupisanje+Agregatne funkcije

Prikazati imena svih radnika čija su ukupna primanja manja od proseka primanja zaposlenih u odeljenju čije je sedište na Novom Beogradu.

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata,**Id_odeljenja$**>

ODELJENJE <**#id_odeljenja**, ime_od, mesto, **sef_odeljenja$**>

```
select ime, prezime
from radnik
where plata+IFNULL(premija,0) < (select avg(plata+IFNULL(premija,0))
                                 from radnik
                                 where id_odeljenja =
                                   (select id_odeljenja
                                    from odeljenje
                                    where mesto='Novi Beograd'))
```

Podupiti + Grupisanje+Agregatne funkcije

Prikazati ime, prezime, datum zaposlenja i primanja zaposlenih koji su angažovani na više od dva projekta.

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata,**Id_odeljenja$**>

UCESCE <**#id_radnika**, **#id_projekta**, br_sati, funkcija >

```
select ime, prezime, dat_zap, plata+IFNULL(premija,0)
from radnik
where id_radnika IN (select id_radnika
                     from ucesce
                     group by id_radnika
                     having count(*)>2)
```

Podupiti + Grupisanje+Agregatne funkcije

Ko su najbolje plaćeni radnici u celom preduzeću?

```
select ime, prezime
from radnik
where plata+IFNULL(premija,0) =
    (select max(plata+IFNULL(premija,0))
     from radnik)
```

- Spoljašnji i unutrašnji upit mogu biti povezani po vrednostima više atributa.
- U tom slučaju ako se upoređuju argumenti oba argumenta moraju imati jednaki broj atributa

Ko su najbolje plaćeni radnici u svakom odeljenju?

```
select ime, prezime
from radnik
where plata+IFNULL(premija,0) IN
    (select max(plata+IFNULL(premija,0)
     from radnik
     group by id_odeljenja)
and id_odeljenja IS NOT NULL
```

```
select ime, prezime
from radnik
where (id_odeljenja, plata+IFNULL(premija,0))  IN
    (select id_odeljenja, max(plata+IFNULL(premija,0))
     from radnik
     group by id_odeljenja)
and id_odeljenja IS NOT NULL
```

Korelisani podupiti

- Korelisani podupit je podupit čiji rezultat zavisi od neke promenljive.
- Ta promenljiva dobija svoju vrednost u nekom spoljašnjem upitu.
- Obrada takvog podupita se ponavlja za svaku vrednost promenljive u upitu, a ne izvršava se svaki put.

Korelisani podupiti

Prikazati imena projekata na kojima rade radnici čija je funkcija organizator.

RADNIK <**#id_radnika**, ime, prezime, posao, kvalif, **rukovodilac$**, dat_zap, premija, plata,**Id_odeljenja$**>

UCESCE <**#id_radnika**, **#id_projekta**, br_sati, funkcija >

```
5 •  SELECT p.ime_proj
6    FROM PROJEKAT p
7    WHERE "organizator" IN ( SELECT funkcija
8    FROM ucesce u
9    WHERE u.id_projekta = p.id_projekta);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime_proj
izvoz

Korelisani podupiti

- Razlika u odnosu na prethodne primere sa podupitima:
  - unutrašnji podupit se ne izvršava pre spoljašnjeg, zato što unutrašnji upit zavisi od promenljive p.id_projekta, čije se značenje menja u odnosu na to kako sistem proverava različite redove tabele PROJEKAT.

- Za prvi red tabele PROJEKAT , id_projekta = 100 generiše se upit
  SELECT funkcija FROM ucesce u WHERE u. id_projekta =100;

- Kao rezultat dobija se skup
  IZVRŠILAC, KONSULTANT, ŠEF, NADZORNIK

- Vrši se obrada spoljnjeg upita (provera da li parametar u WHERE klauzili postoji u dobijenom rezultatu za projekat 100)

- Vrednost ORGANIZATOR ne pripada skupu rezultata podupita.

Korelisani podupiti

- Prelazi se na sledeći red tabele PROJEKAT i pronalazi da je id_projekta jednak 200. Tada imamo podupit:

  SELECT funkcija FROM ucesce u WHERE u.id_projekta=200;

- Kao rezultat dobija se skup (ŠEF, ORGANIZATOR, KONSULTANT, IZVRŠILAC).

- Sada se može izvršiti obrada spoljnjeg upita za projekat 200.

- Kao rezultat toga upita dobija se ime projekta Izvoz jer u skupu postoji vrednost ORGANIZATOR.

- Onda se prelazi na sledeći red tabele PROJEKAT.
- Postupak se ponavlja sve do poslednjeg reda tabele PROJEKAT.

Korelisani podupiti

Prikazati imena radnika koji na projektima imaju više od 2000 sati angažovanja.

```
8  SELECT ime
9  FROM RADNIK AS r
10 WHERE 2000 < ( SELECT SUM(br_sati)
11 FROM ucesce
12 WHERE id_radnika = r.id_radnika);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ime
Mirjana

Podupiti – upotreba EXISTS operatora

U podupitima se može koristiti operator EXISTS. To je unarni logički operator iza kojeg sledi podupit. Njegova vrednost je TRUE ako podupit izdvaja bar jedan zapis.

Da bi operator EXISTS funkcionisao na pravilan način vrlo često se javlja potreba da ugnježdeni upit povežemo (korelišemo) sa spoljašnjim upitom.

Podupiti – upotreba EXISTS operatora

Prikazati sve podatke o odeljenjima u kojima ima zaposlenih radnika.

```
SELECT *
FROM ODELJENJE
WHERE  EXISTS ( SELECT id_odeljenja
FROM radnik
WHERE RADNIK.id_odeljenja=ODELJENJE.id_odeljenja)
```

Korelisanje se obavlja uz pomoć uslova RADNIK.id_odeljenja=ODELJENJE.id_odeljenja.

Zahvaljujući dodatnom uslovu spoljašnji i ugnježđeni upit su povezani i ugnježđeni upit vraća šifre odeljenja samo za odeljenja u kojima ima radnika.

U spoljašnjem upitu se za svaku šifru odeljenje uz pomoć operatora EXISTS proverava da li ugnježđeni upit vraća rezultujuću tabelu sa vrstama odnosno da li odeljenje ima radnike ili ne.

Id_odeljenja	Ime_od	Mesto	Sef_odeljenja
10	Komercijala	Novi Beograd	5662
20	Plan	Dorćol	5780
30	Prodaja	Stari Grad	5786
40	Direkcija	Banovo Brdo	5842

Podupiti – upotreba EXISTS operatora

Ukoliko se izostavi uslov korelisanja dobio bi se upit koji NE VRAĆA TRAŽENE PODATKE.

Vraćeni su podaci o svim odeljenjima bez obzira na to da li imaju zaposlene radnike ili ne. Problem je nepostojanje korelacije između spoljašnjeg upita i ugnježdenog upita.

```
SELECT *
FROM ODELJENJE
WHERE EXISTS (SELECT id_odeljenja FROM radnik)
```

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

Kad god se proverava da li u odeljenju ima zaposlenih radnika ili ne, ugnježdeni upit uvek vraća isti rezultat (sve šifre odeljenja iz tabele radnik). Zbog toga EXISTS ima vrednost TRUE kod svakog odeljenja odnosno rezultujuća tabela sadrži podatke o svim odeljenjima.

Podupiti – upotreba EXISTS operatora

```
5 •  SELECT *
6    FROM ODELJENJE
7    WHERE EXISTS (SELECT id_odeljenja FROM radnik
8                  WHERE id_odeljenja IS NOT NULL);
```

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

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

Podupiti – upotreba EXISTS operatora

Modifikacijom SQL upita iz prethodnog primera može se dobiti upit koji prikazuje podatke o odeljenjima u kojima nema zaposlenih radnika.

```
5 •  SELECT *
6    FROM ODELJENJE
7    WHERE NOT EXISTS (SELECT id_odeljenja FROM radnik
8                      WHERE radnik.id_odeljenja=odeljenje.id_odeljenja)
```

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

Id_odeljenja	Ime_od	Mesto	Sef_odeljenja
50	Računski centar	Zemun	NULL
60	Nabavka	Rakovica	NULL

Podupiti – upotreba EXISTS operatora

```
5 •  SELECT *
6    FROM ODELJENJE
7    WHERE NOT EXISTS (SELECT 0 FROM radnik
8                      WHERE radnik.id_odeljenja=odeljenje.id_odeljenja)
```

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

Id_odeljenja	Ime_od	Mesto	Sef_odeljenja
50	Računski centar	Zemun	NULL
60	Nabavka	Rakovica	NULL

Ugnježdeni upit vraća samo konstantu vrednost 0. Ovakvo rešenje se primenjuje često kada nisu neophodni podaci iz ugnježdenog upita već samo želimo da proverimo da li oni postoje ili ne.

Podupiti – upotreba NOT EXISTS operatora

Prikazati podatke o zaposlenima koji su i rukovodioci

```
8 •  SELECT concat(prezime," ",ime) as 'Jeste rukovodilac'
9    FROM RADNIK emp
10   where EXISTS (select rukovodilac from radnik mng
11                 where emp.id_radnika=mng.rukovodilac)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Jeste rukovodilac
Mančić Janko
Ristić Božidar
Šotra Pavle
Marković Miloš
Grubač Svetlana
Simić Janko

Prikazati podatke o zaposlenima koji nisu rukovodioci

```
SELECT concat(prezime," ",ime) as 'Nisu rukovodioci'
FROM RADNIK emp
where NOT EXISTS (select rukovodilac from radnik mng
                  where emp.id_radnika=mng.rukovodilac)
```

Nisu rukovodioci
Vasić Petar
Marić Aleksandar
Kondić Vanja
Perić Jovan
Dimić Mirjana
Bogovac Tomislav
Ristić Andrija
Petrović Slobodan
Vuković Mitar
Perić Jovan
Nastić Marko
Buha Ivan
Bošković Luka

Podupiti – upotreba NOT IN operatora

- Ako se isti upit izvršava sa NOT IN umesto sa NOT EXISTS, rezultat će biti drugačiji.
- Rezultat ovoga upita pokazuje da nema zaposlenih koji nisu takođe i rukovodioci.

```
8 •  SELECT prezime, ime
9    FROM RADNIK
10   where id_radnika NOT IN (select rukovodilac from radnik)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

prezime	ime

- Razlog za ovakav rezultat je zbog NULL vrednosti koja je vraćena od strane podupita.
- Jedan od redova u tabeli radnik nema rukovodioca i to čini da ceo rezultat bude netačan.

Podupiti – upotreba NOT IN operatora

Šta treba dodati upitu da rezultat bude tačan?

```
SELECT ime, prezime
from radnik
where id_radnika NOT IN (select rukovodilac
                         from radnik
                         where rukovodilac IS NOT NULL)
```

ime	prezime
Petar	Vasić
Aleksandar	Marić
Vanja	Kondić
Jovan	Perić
Mirjana	Dimić
Tomislav	Bogovac
Andrija	Ristić
Slobodan	Petrović
Mitar	Vuković
Jovan	Perić
Marko	Nastić
Ivan	Buha
Luka	Bošković

Podupiti - zapamti

- Kada se koriste u podupitu, IN operator i = ANY imaju potpuno istu ulogu
- Voditi računa o NULL vrednostima u podupitima.
- Ako niste sigurni da li će podupit uključiti null vrednost eliminišite null korišćenjem IS NOT NULL u WHERE izrazu
- NOT IN je jednako sa <> ALL.
- NOT IN nije jednako sa <> ANY
- NOT EXISTS i NOT IN ne vraćaju isti rezultat.
- SQL upiti koji koriste EXISTS uslov su veoma neefikasni jer se podupit izvršava za svaki red u spoljašnjem upitu.

Podupiti u FROM klauzuli

- Mehanizam virtuelnih pogleda omogućava i ugnježđavanje jednog SQL upita u **FROM** klauzuli drugog SQL upita.
- Ugnježđeni SQL upit se u tom slučaju ponaša kao i bilo koja druga tabela iz baze podataka.
- Rezultat podupita se ponaša kao virtuelna tabela iz koje se izdvajaju n-torke (zapisi).
- Nazivi kolona u spoljnom upitu moraju biti podskup skupa naziva kolona u unutrašnjem upitu.

Podupiti u FROM klauzuli

Ko su najbolje plaćeni radnici u svakom odeljenju?

```
SELECT ime, posao, plata, r.id_odeljenja
FROM radnik r,
    (SELECT MAX(plata) AS max_plata, id_odeljenja
     FROM radnik
     GROUP BY id_odeljenja) AS maxplata
WHERE plata=max_plata AND r.id_odeljenja=maxplata.id_odeljenja
```

ime	posao	plata	id_odeljenja
Janko	upravnik	2400	10
Pavle	upravnik	2100	30
Mitar	savetnik	2600	20
Janko	upravnik	3900	40

Podupiti u FROM klauzuli

Prikazati podatke o odeljenjima u kojima trenutno radi bar 2 radnika.

```
5 •  SELECT odeljenje.id_odeljenja, odeljenje.ime_od, tabelaVirt.brojR
6    FROM odeljenje, (select id_odeljenja, count(*) as brojR
7                     from radnik
8                     group by id_odeljenja) as tabelaVirt
9    where odeljenje.id_odeljenja= tabelaVirt.id_odeljenja and  tabelaVirt.brojR>=2
10   order by 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_odeljenja	ime_od	brojR
10	Komercijala	6
20	Plan	5
30	Prodaja	4
40	Direkcija	3

Podupiti u FROM klauzuli

Može i na ovaj način!

```
5 •  SELECT o.id_odeljenja, o.ime_od, count(*) as 'Broj radnika'
6    from odeljenje o, radnik r
7    where o.id_odeljenja=r.id_odeljenja
8    group by o.id_odeljenja, o.ime_od
9    having count(*)>=2
10   order by 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_odeljenja	ime_od	Broj radnika
10	Komercijala	6
20	Plan	5
30	Prodaja	4
40	Direkcija	3

AŽURIRANJE PODATAKA

Sledeće naredbe omogućavaju ažuriranje baze podataka:

- INSERT – unošenje podataka, dodavanje novih zapisa u tabelu,
- DELETE – brisanje podataka, izbacivanje zapisa iz tabele,
- UPDATE – ažuriranje u užem smislu, izmena vrednosti podataka u tabeli.

Uobičajeno se ažuriranje podataka vrši preko aplikacija. U aplikacije su ugrađene procedure za zaštitu integriteta, kao i procedure za realizaciju naredbi INSERT, DELETE i UPDATE.

Naredbe ažuriranja se uvek odnose na jednu tabelu.
Svaka od naredbi vrši promene na nivou redova tabele
Upiti kojim se realizuje ažuriranje baze podataka nazivaju se **AKCIONI UPITI**.

INSERT

Uz **INSERT naredbu** mora se navesti:

- u koju tabelu ubacujemo,
- za koje kolone dajemo vrednosti,
- vrednosti koje ubacujemo.

Postoje tri tipa ovih naredbi:

1. za ubacivanje vrednosti SVIH atributa jedne n-torke,
2. za ubacivanje vrednosti NEKIH atributa jedne n-torke,
3. za ubacivanje podataka iz jedne tabele u drugu.

INSERT

Uz **INSERT naredbu** mora se navesti:

- u koju tabelu ubacujemo,
- za koje kolone dajemo vrednosti,
- vrednosti koje ubacujemo.

Dodavnje tačno jednog zapisa (reda) u postojeću tabelu:

INSERT INTO <ime_tabele> [(<lista_atributa>)]

VALUES [(<lista_vrednosti>)]

Dodavanje nula, jednog ili više zapisa u postojeću tabelu. n-torke koje se dodaju su rezultat nekog **SELECT** upita:

INSERT INTO <ime_tabele> [(<lista_atributa>)]

<SELECT naredba>

INSERT

Sledeća naredba upisuje jedan zapis u tabelu RADNIK:

```
INSERT INTO RADNIK
VALUES (66,'Milan', 'Milosavljević', NULL, NULL,NULL,NULL,50)
```

Sledeća naredba upisuje jedan zapis u tabelu RADNIK. Redosled atributa u listi INSERT naredbe nije isti kao u definiciji tabele RADNIK. Atributi čije vrednosti nisu zadate dobijaju NULL vrednost.

```
INSERT INTO RADNIK (prezime, ime, id_radnika, id_odeljenja)
VALUES ('Petrov','Milan', 67,20)
```

Sledeća naredba upisuje jedan slog u tabelu ODELJENJE. Atribut ***mesto*** koji postoji u definiciji tabele, a čija vrednost nije zadata, dobija podrazumevanu – ukoliko je postavljena, DEFAULT vrednost ‘Beograd‘.

```
INSERT INTO ODELJENJE(ime_od, id_odeljenja)
VALUES ('Obezbeđenje‘, 70)
```

INSERT

Neka je tabela ZAPOSLENI definisana na isti način kao tabela RADNIK (isti atributi nad istim domenima). Sledeća naredba kopira celokupan sadržan tabele RADNIK u tabelu ZAPOSLENI:

```
INSERT INTO ZAPOSLENI
SELECT *
FROM RADNIK
```

Primena INSERT INTO naredbe za formiranje kopije tabele obavlja se u dva koraka:

- definiše se nova, prazna tabela (naredba CREATE TABLE)
- kopiraju se redovi (naredba INSERT INTO) iz tabele koja se navodi u SELECT rečenici

INSERT

U tabelu UPRAVNIK, koja ima atribute idbr, ime, datzap, brod dodati podatke o svim zaposlenim koji imaju posao upravnika.

```
INSERT INTO UPRAVNIK (idbr, ime, datzap, brod)
SELECT id_radnika, ime, dat_zap, id_odeljenja
FROM RADNIK
WHERE posao='upravnik';
```

```
INSERT INTO UPRAVNIK
SELECT idbr, ime, datzap, brod
FROM RADNIK
WHERE posao='upravnik';
```

INSERT

SELECT .. INTO naredba:

- kreira novu tabelu
- kopira podatke iz postojeće u novu tabelu.

Primer: Kreirati tabelu ZAPOSLENI sa svim istim atributima kao i u tabeli RADNIK i u nju dodati sve zaposlene.

```
SELECT * INTO ZAPOSLENI
FROM RADNIK;
```

Ovim upitom se kreira tabela ZAPOSLENI sa istim atributima kao i u tabeli RADNIK i u nju se dodaju svi zapisi iz tabele RADNIK.

INSERT

Primer: U nepostojeću tabelu KVALIF_VKV sa atributima idbr, ime, brod i u nju smestiti sve zaposlene koji imaju kvalifikaciju VKV.

```
SELECT idbr, ime, brod INTO KVALIF_VKV
FROM RADNIK
WHERE kvalif='VKV';
```

Posle izvršenja ovog upita u bazi se kreira nova tabela KVALIF_VKV, sa atributima: idbr, ime, brod, i u nju su dodati odgovarajući zapisi o radnicima čija je kvalifikacija VKV.

INSERT

SELECT...INTO u MySQL je realizovano na sledeći način:

```
3 • CREATE TABLE kvalif_vkv AS
4   SELECT id_radnika, ime, id_odeljenja
5   FROM radnik
6   WHERE kvalif='VKV';
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:56:36	CREATE TABLE kvalif_vkv AS SELECT id_radnika, ime, id_odeljenja FROM radnik WHERE kvalif=...	1 row(s) affected Records: 1 Duplicates: 0 Warnings: 0	0.313 sec

```
8 • select * from kvalif_vkv
```

Result Grid | Filter Rows: | Export:

id_radnika	ime	id_odeljenja
5519	Vanja	10

INSERT

U tabelu POVIŠICA <idbr, povišica, datzap> dodati podatke o analitičarima i vozačima i dati im povećanje plate od 15%.

```
3 •  CREATE TABLE povisica AS
4    SELECT id_radnika, ime, prezime, round(plata*1.15,0)
5    FROM radnik
6    WHERE posao IN ('analitičar', 'vozač');
7
8 •  select * from povisica
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_radnika	ime	prezime	round(plata*1.15,0)
5367	Petar	Vasić	1495
6234	Marko	Nastić	1495
7890	Ivan	Buha	1840
7892	Luka	Bošković	2300

DELETE

Brisanje zapisa u tabeli može se izvesti pojedinačno ili grupno. Komandom DELETE uvek se briše cela n-torka, a ne samo pojedina vrednost nekog atributa.
Sintaksa naredbe u opštem slučaju glasi:

DELETE
FROM ime tabele
[WHERE lista uslova]

Izbrisati sve podatke iz tabele KVALIF_VKV.

I) DELETE * FROM KVALIF_VKV;
II) DELETE KVALIF_VKV.* FROM KVALIF_VKV;
III) DELETE FROM KVALIF_VKV;

Ako nije naveden uslov (WHERE), brišu se svi podaci iz tabele. Daleko efikasnija od naredbe DELETE za brisanje svih podataka iz tabele je naredba **TRUNCATE**. **Ova naredba briše sve podatke iz tabele, ali ne i definiciju tabele.**

TRUNCATE TABLE KVALIF_VKV**;**

DELETE

Izbrisati podatke o radnicima koji su se zaposlili pre 1990.– te godine.

```
DELETE
FROM RADNIK
WHERE year(datzap) < 1990;
```

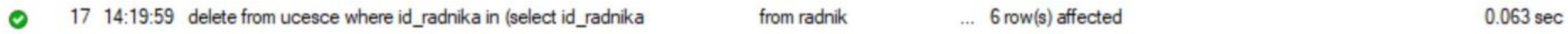
Isključiti radnike koji rade u odeljenju Stari Grad.

```
DELETE
FROM RADNIK
WHERE id_odeljenja= (SELECT id_odeljenja
                     FROM ODELJENJE
                      WHERE mesto= ‘Stari Grad');
```

DELETE

Radnike iz odeljenja 20 isključiti sa svih projekata.

```
delete
from ucesce
where id_radnika in (select id_radnika
                     from radnik
                     where id_odeljenja=20)
```



DELETE

Ukoliko se vrši brisanje velike grupe podataka preporuka je sprovođenje logičnog redosleda koraka.

Za slučaj DBMS sistem koji ne podržava transakcije, ako se izbrišu podaci koje nismo želeli, gotovo nemoguće ih je vratiti.

Pre naredbe DELETE pogodno najpre pomoću SELECT upita izdvojiti podatke koje treba obrisati.

Dobijeni skup podataka pažljivo pregledati i analizirati.

Ukoliko se utvrdi da su to upravo podaci koje treba obrisati, onda jednostavno umesto odredbe SELECT u upitu upisati odredbu DELETE i izvršiti operaciju brisanja.

DELETE

Radnike čija je premija nula isključiti sa svih projekata.

```
DELETE *
FROM UCESCE
WHERE id_radnika IN
                    (SELECT id_radnika
                     FROM RADNIK
                     WHERE premija=0);
```

```
3 •  SELECT *
4    FROM UCESCE
5    WHERE id_radnika IN (SELECT id_radnika
6                                 FROM RADNIK
7                                 WHERE premija=0);
8
```

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

Id_radnika	Id_projekta	Br_sati	Funkcija
5696	200	2000	ŠEF
5696	300	2000	IZVRŠILAC
5953	100	1000	IZVRŠILAC
5953	300	1000	IZVRŠILAC

UPDATE

Modifikacija postojećih podataka (ažuriranje u užem smislu) izvodi se komandom UPDATE - SET.

```
UPDATE ime tabele
   SET atribut1=vrednost1[, atribut2=vrednost2, ...]
   WHERE [ lista uslova ]
```

Naredbom UPDATE - SET može se menjati vrednost jednog ili više atributa unutar jedne n-torke.
Naredbom UPDATE - SET može se menjati i vrednost jednog atributa unutar više n-torki.

UPDATE

Svim konsultantima smanjiti platu za 25% i ukinuti premiju.

```
UPDATE RADNIK
SET plata= plata*0.75, premija=NULL
WHERE id_radnika IN
        (SELECT id_radnika
         FROM UCESCE
          WHERE funkcija='konsultant');
```

UPDATE

Zaposlenima u odeljenju smeštenom na Novom Beogradu povećati platu 30%.

```
UPDATE RADNIK
    SET plata=plata*1.3
    WHERE id_odeljenja IN ( SELECT id_odeljenja
                            FROM ODELJENJE
                            WHERE mesto='Novi Beograd');
```

NAPOMENA

- Prilikom upotrebe naredbi za ažuriranje neophodno je biti veoma oprezan jer one menjaju stanje baze, tj. vrednosti podataka.

- Zbog toga se preporučuje da se najpre pomoću obične SELECT naredbe izdvoje n-torke na koje bi se primenile naredbe za ažuriranje, pa kada posle analize rezultata nepobitno utvrdimo da se radi o željenoj grupi n-torki, onda kreirati odgovarajuću naredbu UPDATE ili DELETE

PREPORUKA

- Umesto brisanja (DELETE) učiniti podatke nedostupnim.

- Ovo se jednostavno može postići dodavanjem jedne dodatne kolone u svaku tabelu, sa nazivom Obrisan. U ovu kolonu sistem automatski upisuje neku podrazumevanu (DEFAULT) vrednost, na primer: NE.

- Kada želimo da “obrišemo“ neke podatke, onda umesto DELETE upita uradimo UPDATE i za izabrane zapise u kolonu Obrisan upišemo vrednost DA.

PREPORUKA

Isključiti sa svih projekata radnike čija je premija nula :

```
UPDATE UCESCE
SET Obrisan = “DA“
WHERE id_radnika IN (SELECT id_radnika
                     FROM RADNIK
                     WHERE premija=0);
```

U upitima za rad sa podacima bi postojala klauzula WHERE Obrisan= “NE“ ili WHERE Obrisan= “DA“

Ako već postoji neki WHERE uslov za selekciju redova onda se na njega dodaje AND Obrisan =“NE“.