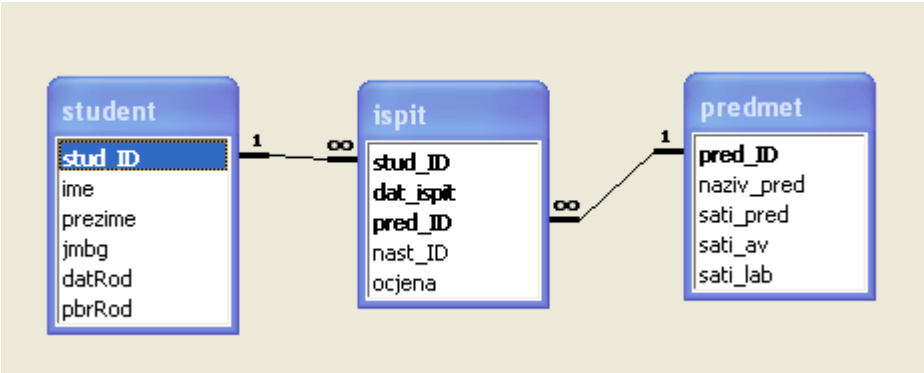
PRIMJERI SQL

Suzana Divić

1. Osnove – projekcija, selekcija, spajanje tablica



1.1. Projekcija

Ukoliko želimo vratiti sve atribute nekog entiteta, možemo korištenjem znaka * koji označava sve atribute:

select * from predmet;
select predmet.* from predmet;
select p.* from predmet p;

ili možemo nabrojiti svaki od atributa

select pred_id, naziv_pred, sati_pred, sati_av, sati_lab from predmet;
select predmet.pred_id, predmet.naziv_pred, predmet.sati_pred, predmet.sati_av, predmet.sati_lab from predmet;
select p.pred_id, p.naziv_pred, p.sati_pred, p.sati_av, p.sati_lab from predmet p;

Kad želimo izlistati samo neke atribute (stupce tablice), govorimo o projekciji.

Primjer 1:
Select ime, prezime, jmbg from student;

Za sve studente (odn. sve retke upisane u tablicu "student"), upit vraca samo oznacene atribute

stud_ID	ime	prezime	jmbg	datRod	pbrRod
10001	Ana	Jurić	1106980401236	11.06.1980	10000
10234	Mario	Klarić	0311981302215	22.05.1981	31000
11010	Ivan	Korkut	0209979502247	02.09.1979	21000
11234	Ante	Jurić	0412979214665	04.12.1979	48000
12244	Marija	Knjaz	1109980325234	11.09.1980	51000
13475	Kristina	Kušan	0202980122201	25.02.1980	31000
21112	Ivan	Županović	0311981376432	18.02.1982	31000
21331	Petar	Šoljić	1211981879356	21.05.1981	22000
22222	Mirta	Lončar	2709980127782	05.11.1987	52000
25367	Davor	Dujmić	2504982561131	24.02.1987	10000
27783	Franjo	Mušnjak	1105981200211	21.05.1981	33000
34211	Petra	Krleža	0506978657631	05.11.1986	10000
34678	Jurica	Domanovac	0606978543121	05.06.1989	10000
34879	Hrvoje	Petrović	1708982012003	17.08.1982	10000
35648	Igor	Ivčić	0203976370211	02.03.1976	21000
35765	Ana	Antolić	0808980452211	20.02.1980	32000
36781	Mislav	Bistričić	0311981356685	24.02.1985	20000
47678	Ljubica	Šaput	2412982700121	24.12.1982	44000
56778	Boris	Kundera	0203976443523	02.03.1976	23000
58445	Jelena	Župan	0209979122134	02.09.1979	21000
0					0

Primjer 2:

Select pred_id, naziv_pred from predmet;

Za sve predmete (odn. sve retke upisane u tablicu "predmet"), upit vraca samo oznacene atribute

pred_ID	naziv_pred	sati_pred	sati_av	sati_lab
1001	matematika	4	3	0
1002	baze podataka	3	2	2
1003	primjena računala	1	0	2
1004	operacijski sustavi	2	2	2
1005	programiranje	2	2	1
1006	računala i procesi	3	2	1
1007	elektronika	3	1	2
1008	algoritmi i strukture podataka	2	1	1
1009	računalna grafika	2	1	1
1010	programske paradigme i jezici	2	2	1
0		0	0	0

1.2. Selekcija

Govorimo o selekciji kad želimo dohvatiti samo neke zapise iz tablice (samo one koji zadovoljavaju neki zadani kriterij)!

Primjer 1:

Select * from student where prezime = "Jurić";

Od svih zapisa u tablici student, upit će vratiti samo one označene

	stud_ID	ime	prezime	jmbg	datRod	plaRod
+	35765	Ana	Antolić	0808980452211	20.02.1980	32000
+	36781	Mislav	Bistričić	0311981356685	24.02.1985	20000
+	34678	Jurica	Domanovac	0606978543121	05.06.1989	10000
+	25367	Davor	Dujmić	2504982561131	24.02.1987	10000
+	35648	Igor	Ivčić	0203976370211	02.03.1976	21000
+	11234	Ante	Jurić	0412979214665	04.12.1979	48000
+	10001	Ana	Jurić	1106980401236	11.06.1980	10000
+	10234	Mario	Klarić	0311981302215	22.05.1981	31000
+	12244	Marija	Knjaz	1109980325234	11.09.1980	51000
+	11010	Ivan	Korkut	0209979502247	02.09.1979	21000
+	34211	Petra	Krleža	0506978657631	05.11.1986	10000
+	56778	Boris	Kundera	0203976443523	02.03.1976	23000
+	13475	Kristina	Kušan	0202980122201	25.02.1980	31000
+	22222	Mirta	Lončar	2709980127782	05.11.1987	52000
+	27783	Franjo	Mušnjak	1105981200211	21.05.1981	33000
+	34879	Hrvoje	Petrović	1708982012003	17.08.1982	10000
+	47678	Ljubica	Šaput	2412982700121	24.12.1982	44000
+	21331	Petar	Šoljić	1211981879356	21.05.1981	22000
+	58445	Jelena	Župan	0209979122134	02.09.1979	21000
+	21112	Ivan	Županović	0311981376432	18.02.1982	31000

1.3. Selekcija i projekcija

→ kad želimo izlistati samo neke zapise iz tablice (samo one koji zadovoljavaju neki zadani kriterij), ali ne sve atribute već samo neke!

Primjer 1:

Select ime, prezime, jmbg from student where prezime = "Jurić";

Promotrimo sliku iz prethodnog primjera – ovdje se vrši ista selekcija kao u prošlom primjeru. Izdvajaju se 2 zapisa koja odgovaraju uvjetu da je za njih prezime Jurić. Za ta dva zapisa ispisat će se samo atribute ime, prezime i jmbg:

	ime	prezime	jmbg
	Ana	Jurić	1106980401236
	Ante	Jurić	0412979214665

Primjer 2:

Select ime, prezime, jmbg from student
where prezime = "Jurić" and year(datRod) = 1985;

Radi se kao u prethodnom primjeru, ali uz dodatak kriterija - želimo vratiti samo one koji su rođeni 1985. godine

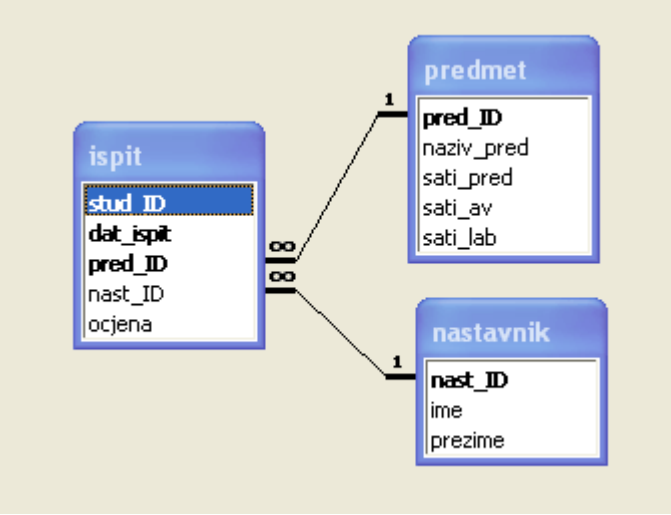
+	35765	Ana	Antolić	0808980452211	20.02.1980	32000
+	36781	Mislav	Bistričić	0311981356685	24.02.1985	20000
+	34678	Jurica	Domanovac	0606978543121	05.06.1989	10000
+	25367	Davor	Dujmić	2504982561131	24.02.1987	10000
+	35648	Igor	Ivčić	0203976370211	02.03.1976	21000
+	11234	Ante	Jurić	0412979214665	04.12.1979	48000
+	10001	Ana	Jurić	1106980401236	11.06.1980	10000
+	10234	Mario	Klarić	0311981302215	22.05.1981	31000
+	12244	Marija	Knjaz	1109980325234	11.09.1980	51000
+	11010	Ivan	Korkut	0209979502247	02.09.1979	21000
+	34211	Petra	Krleža	0506978657631	05.11.1986	10000
+	56778	Boris	Kundera	0203976443523	02.03.1976	23000
+	13475	Kristina	Kušan	0202980122201	25.02.1980	31000
+	22222	Mirta	Lončar	2709980127782	05.11.1987	52000
+	27783	Franjo	Mušnjak	1105981200211	21.05.1981	33000
+	34879	Hrvoje	Petrović	1708982012003	17.08.1982	10000
+	47678	Ljubica	Šaput	2412982700121	24.12.1982	44000
+	21331	Petar	Šoljić	1211981879356	21.05.1981	22000
+	58445	Jelena	Župan	0209979122134	02.09.1979	21000
+	21112	Ivan	Županović	0311981376432	18.02.1982	31000

Za Antu Jurića datRod = '04.12.1979' → ne zadovoljava uvjet year(datRod) = 1980

Zato je rezultat upita ovo:

	ime	prezime	jmbg
	Ana	Jurić	1106980401236
▶			

1.4. Spajanje (join)



Povezivanje tablica (join) koristimo da bismo dohvatili podatke koji se nalaze u različitim tablicama, ali su logički povezani.

Primjer 1:
Dohvatiti sve podatke nastavnika koji su držali ispit zajedno s nazivima predmeta koje su ispitivali.

- Želimo selectirati atribute iz 2 entitea koja nisu direktno povezana vec 'između njih' postoji tablica veze (ispit) => moramo povezati sve 3 tablice da bismo ispravno povezali podatke o nastavniku i predmetu.
- Iz definicije zadatka vidi se popis atributa koje treba dohvatiti:

select nastavnik.*, predmet.naziv_Pred (A)

- u tablici ispit je veza predmeta i nastavnika. Dakle moramo uključiti i tu tablicu da bismo došli do odgovora na pitanje koji nastavnik je povezan uz koji predmet.

from nastavnik n, ispit i, predmet p (B)

- Zbog postojeće veze (FK) očekujemo da u tablici ispit ne može biti unesen ključ nepostojećeg nastavnika ili predmeta (u protivnom bi bio narušen referencijalni integritet baze. RDBMS ne bi dopustio upisivanje takvog zapisa)
- ukoliko ima redaka u tablici ispit za koje je nastavnik NULL, takvi nas zapisi u ovom slučaju ne zanimaju.
- Očekujemo da su nam u tablici predmeta podaci o svim predmetima, te u tablici nastavnika podaci o svim nastavnicima.
- Ne zanimaju nas niti nastavnici niti predmeti za koje ne postoji zapis u tablici ispit, jer ne odgovaraju na naše pitanje o povezanosti profesora i predmeta

where n.nast_id = i.nast_id and i.pred_id = p.pred_id (C)

Dakle:

```
select n.*, p.naziv_pred
from nastavnik n, ispit i, predmet p
where n.nast_id = i.nast_id and i.pred_id = p.pred_id;
```

ili

```
select  n.*, p.naziv_pred
from predmet p inner join  (nastavnik n inner join  ispit i on n.nast_id = i.nast_id)  on
p.pred_id = i.pred_id;
```

ili

```
select  n.*, p.naziv_pred
from  (nastavnik n inner join  ispit i on n.nast_id = i.nast_id)  inner join  predmet p on
p.pred_id = i.pred_id;
```

ili

```
select  n.*, p.naziv_pred
from  nastavnik n inner join  (ispit i inner join  predmet p on p.pred_id = i.pred_id) on
n.nast_id = i.nast_id;
```

itd... (varijacije redosljeda)

Ne može ovo: ... from ispit INNER JOIN (predmet INNER JOIN nastavnik ...)
JER NEMA KLJUČA KOJI SPAJA PREDMET I NASTAVNIK!!!!

Ovi upiti nam u pravilu daju 'duplićе' jer tablica ispit ima i vezu prema studentu – dakle ukoliko je nastavnik Pero Perić držao ispit iz baza podataka 13.2.2010 na kojeg je izašlo 15 studenata, u tablici ispit biti će barem 15 zapisa koji će odgovarati tom

nastavniku i predmetu (i više ako je isti nastavnik iz istog predmeta držao ispit i na neke druge datume).

Dodamo li u sve ove upite nakon SELECT još i ključnu riječ DISTINCT, dupli zapisi se neće pojaviti:

```
select DISTINCT n.*, p.naziv_pred
from nastavnik n, ispit i, predmet p
where n.nast_id = i.nast_id and i.pred_id = p.pred_id
```

ili

```
select  DISTINCT n.*, p.naziv_pred
from predmet p inner join  (nastavnik n inner join  ispit i on n.nast_id = i.nast_id)  on
p.pred_id = i.pred_id
```

ili ...

Primjer 2:

Dohvatiti sve nastavnike:

```
select * from nastavnik;
```

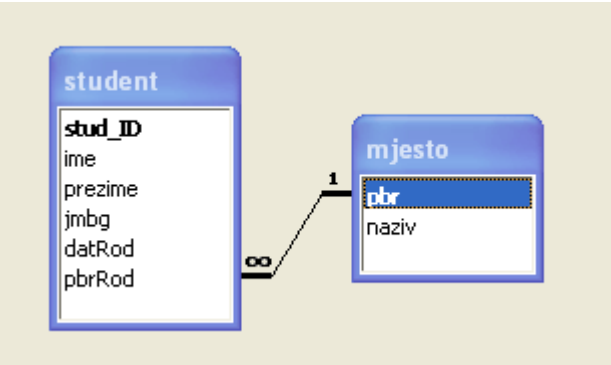
Za ovo nije potrebno spajati tablicu nastavnika s drugom tablicom!

Primjer 3:

Dohvatiti sve nastavnike koji su držali ispit

```
select nastavnik.* from nastavnik INNER JOIN ispit on nastavnik.nast_id = ispit.nast_id;
```

1.5. Spajanje (Join) - Ilustracija razlike INNER i Left/Right OUTER spajanja



Pretpostavimo da mjesto rođenja nije obavezan podatak za studenta. Tamo gdje je podatak nepoznat, vrijednost pbrRod je NULL. Tamo gdje podatak imamo, unesen je ključ iz tablice mjesto.

Inner join tablica student i mjesto **neće** vratiti zapise o studentima za koje nema podatka o pbrRod u tablici student:

```
SELECT student.ime, student.prezime, mjesto.naziv
FROM mjesto INNER JOIN student ON mjesto.pbr = student.pbrRod;
```

Ili

```
SELECT student.ime, student.prezime, mjesto.naziv
FROM mjesto, student WHERE mjesto.pbr = student.pbrRod;
```

Ukoliko želimo dohvatiti podatke ime, prezime, mjesto rođenja o svim studentima ali tako da se naziv mjesta rođenja ispiše za one studente za koje ima podatka, moramo koristiti lijevi ili desni outer join na način da se uzme sve podatke iz tablice student (student LEFT JOIN mjesto ili mjesto RIGHT JOIN student):

```
SELECT student.ime, student.prezime, mjesto.naziv
FROM mjesto RIGHT  JOIN student ON mjesto.pbr = student.pbrRod;
```

```
SELECT student.ime, student.prezime, mjesto.naziv
FROM student LEFT JOIN mjesto ON mjesto.pbr = student.pbrRod;
```

- Kada koristiti LEFT a kada RIGHT JOIN?
  - nema veze s tim što je na dijagramu nacrtano lijevo a što desno!!!
  - Ima veze s tim koja tablica je navedena prva (ona se smatra LIJEVOM tablicom). Pitanje je da li želimo sve podatke iz nje (lijeve) ili sve podatke iz one druge (desne) tablice. (U gornjem primjeru želimo sve podatke iz tablice student.)

1.6. Produkt

Ispisati za svakog studenta koji bi sve predmet mogao slušati: svakog studenta želimo povezati sa svakim predmetom!

```
SELECT student.ime, student.prezime, predmet.naziv_pred
FROM student, predmet
```

Obzirom da nemamo naveden ključ spajanja tablica, spaja se svaki zapis iz tablice student sa svakim zapisom iz tablice predmet.

Ukoliko imamo 2 studenta: Pero Perić i Mara Marić
i 3 predmeta: matematika, fizika i baze podataka

Rezultat upita će biti 2*3 zapisa:

Pero Perić, matematika

Pero Perić, fizika
Pero Perić, baze podataka
Mara Marić, matematika
Mara Marić, fizika
Mara Marić, baze podataka

Da smo imali 100 studenata i 20 predmeta, upit bi vratio 100*20 = 2000 zapisa

1.7. Unija

Primjer 1:

Imamo 2 tablice:
Muskarci = (muskarac_ID, ime, prezime, jmbg)
zene = (zena_ID, ime, prezime, jmbg, datRod)

Želimo dohvatiti sve osobe:

```
select ime, prezime, jmbg from muskarci
union
select ime, prezime, jmbg from zene;
```

Zamislimo da je greškom Pero Perić unesen i u tablicu muškaraca i u tablicu žena. Gornji upit vratit će podatak o Peri Periću samo jednom.

Podatak o Peri Periću biti će izlistan dva puta ukoliko umijesto ključne riječi UNION napišemo UNION ALL:

```
select * from muskarci
union all
select zena_ID, ime, prezime, jmbg from zene;
```

Primjer 2:

Zadatak kao i u primjeru 1. ali želimo imati i informaciju o spolu osobe

```
select ime, prezime, jmbg, 'M' as spol from muskarci
union
select ime, prezime, jmbg, 'Ž' from zene;
```

Za Peru Perića koji je pogreškom unesen u obje tablice, ovdje će i operator union vratiti 2 zapisa jer zapisi koje dobijemo iz prvog i drugog selecta nisu isti – razlikuju se u vrijednosti atributa spol.

2. *Primjeri korištenja izraza i funkcija u select dijelu upita nad jednom tablicom*

Primjer1:

Tablica zaposlenika:

ID_zaposlenog	placa	b_placa_god	bonus
1	10000	12	3000
2	15000	13	20000
3	5000	12	0
4	6000	12	1000

```
SELECT id_zaposlenog, placa * b_placa_god + bonus as  total_godisnje
from zaposlenici
```

id_zaposlenog	total_godisnje
1	123000
2	215000
3	60000
4	73000

Primjer2:

Tablica zaposlenika:

ID_zaposlenog	datum_zaposl
1	01.11.1995
2	15.02.2004
3	11.01.2006
4	07.07.2008
(AutoNumber)	

```
SELECT id_zaposlenog, datum_zaposl,
        year(datum_zaposl) as godina_zaposlenja,
        round((date() - datum_zaposl)/365) as godina_zaposlen
from zaposlenici
```

id_zaposlenog	datum_zaposl	godina_zaposlenja	godina_zaposlen
1	01.11.1995	1995	15
2	15.02.2004	2004	7
3	11.01.2006	2006	5
4	07.07.2008	2008	2

Objašnjenje:

Funkcija round zaokruzuje decimalni broj na cjelobrojnu vrijednost.

Round(1.2) = 1
Round(1.834) = 2

Funkcija date() vraća sistemski datum u formatu datetime

Razlika 2 datuma d2-d1 je broj dana između d1 i d2.

P.S. primjer je za ilustraciju. Točniji rezultat za broj godina koliko je zaposlenik zaposlen bismo dobili pomocu izraza **year(date()) – year(datum_zaposl)**

3. *Primjeri korištenja izraza i funkcija u where dijelu upita nad jednom tablicom*

Primjer 1:

Želimo dohvatiti samo one studente koji su rođeni 1985. godine

select * from student where year(datRodj) = 1985;

Primjer 2:
Tablica zaposlenika:

	ID_zaposlenog	placa	b_placa_god	bonus
	1	10000	12	3000
	2	15000	13	20000
	3	5000	12	0
	4	6000	12	1000

Želimo dohvatiti samo zaposlenike koji zarađuju preko 100.000 kuna godišnje:

```
SELECT *
from zaposlenici
where (placa * b_placa_god + bonus) > 100000;
```

4. *Primjer - Ring struktura*

Tablica povezana sama sa sobom...
Tablica na nekom atributu ima definiran strani ključ koji 'pokazuje' na primarni ključ iste tablice

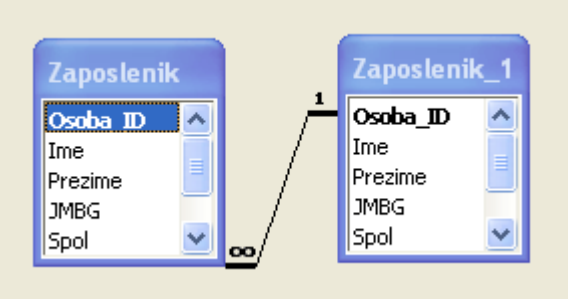
Primjer:

- Tablica zaposlenih, jedan od atributa svakog zaposlenika je informacija o tome tko mu je šef. 1 šef može imati više podređenih, ali svaki zaposlenik ima samo 1 šefa.

	Osoba_ID	Ime	Prezime	JMBG	Spol	Odjel	Sef_ID
+	1	Ana	Anić	1111111111111	Ž	Kadrovska	1
+	2	Pero	Perić	1111111222222	M	Kadrovska	1
+	3	Mate	Matić	2212221212334	M	Tehnika	3
+	4	Mirta	Mirtić	3212734621471	Ž	Kadrovska	1
+	5	Mara	Marić	4238493242342	Ž	Tehnika	3
+	6	Ante	Antić	3249832058349	M	Tehnika	3
	(AutoNumber)						0

- Atribut sef_ID pokazuje na zapis o šefu koji se nalazi u istoj tablici

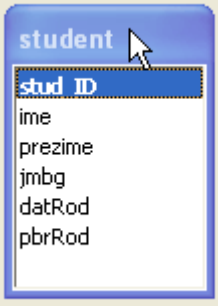
Na *Relationships* dijagramu u Accessu to izgleda ovako:



- Da bismo napisali upit koji koristi istu tablicu za 2 namjene (jednom da bismo dobili podatke o radniku a drugi put da bismo uz podatke o radniku dopisali podatke o njegovom šefu), trebamo koristiti alias naziv:

```
SELECT      Zaposlenik.Ime AS ime_zaposlenika,
            Zaposlenik.Prezime AS prezime_zaposlenika,
            Sef.Ime AS ime_nadredjenog,
            sef.Prezime AS prezime_nadredjenog
FROM Zaposlenik AS Sef INNER JOIN Zaposlenik
      ON Sef.Osoba_ID=Zaposlenik.Sef_ID;
```

	ime_zaposlenika	prezime_zaposlenik	ime_nadredjenog	prezime_nadredjenog
	Ana	Anić	Ana	Anić
	Pero	Perić	Ana	Anić
	Mirta	Mirtić	Ana	Anić
	Mate	Matić	Mate	Matić
	Mara	Marić	Mate	Matić
	Ante	Antić	Mate	Matić



5. *Dodavanje podataka u tablicu – naredba insert*

Imamo tablicu student u kojoj su stud_id, ime i prezime obavezno podaci (required, odnosno not null).

Podatak o datumu i mjestu rođenja za studenta nije obavezan, kao ni jmbg.

Ne možemo ubaciti novi zapis (podatke za novog studenta) ukoliko ne upišemo sve obavezne podatke

Primjer 1:

Želimo u tablicu student dodati zapis o studentu Peri Periću iz Zagreba, rođenom 1.12.1987. godine

Možemo to napraviti na nekoliko načina:

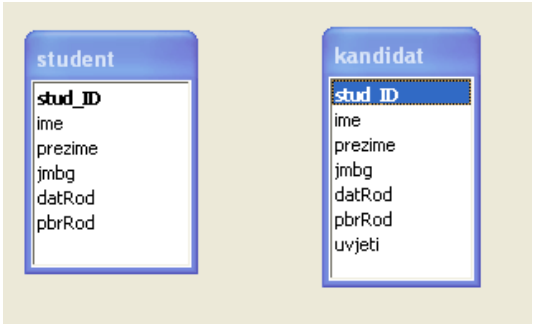
insert into student values (2010623, 'Pero', 'Perić', NULL, '01.12.1987', NULL);

insert into student(ime, stud_id, prezime) values ('Pero', 2010623, 'Perić');

Komentar – kad bi primarni ključ stud_id bio definiran kao sekvenca (Autonumber, COUNTER), ne bi bilo nužno u insert naredbi ubaciti vrijednost stud_id, već bi ga baza sama generirala prilikom upisivanja podataka.

Primjer 2:

Osim tablice student iz prethodnog primjera, imamo dodatnu tablicu kandidat koja ima sve atribute tablice student uz dodatni atribut uvjeti. Kandidati za koje uvjeti="Da" upisuju fakultet i postaju studenti. Želimo ih prebaciti u tablicu student.



insert into student
select stud_id, ime, prezime, jmbg, datRod, pbrRod
from kandidat where uvjeti = 'Da';

Tablica studenata prije ubacivanja podataka:

stud_ID	ime	prezime	jmbg	datRod	pbrRod
10001	Ana	Jurić	1106980401236	11.06.1980	10000
10234	Mario	Klarić	0311981302215	03.11.1981	31000
11010	Ivan	Korkut	0209979502247	02.09.1979	21000
11234	Ante	Jurić	0412979214665	04.12.1979	48000
12244	Marija	Knjaz	1109980325234	11.09.1980	51000
13475	Kristina	Kušan	0202980122201	02.02.1980	31000
21112	Ivan	Županović	0311981376432	03.11.1981	31000
21331	Petar	Šoljić	1211981879356	12.12.1981	22000
22222	Mirta	Lončar	2709980127782	27.09.1980	52000
25367	Davor	Dujmić	2504982561131	25.04.1982	10000
27783	Franjo	Mušnjak	1105981200211	11.05.1981	33000
34211	Petra	Krleža	0506978657631	05.06.1978	10000
34678	Jurica	Domanovac	0606978543121	05.06.1978	10000
34879	Hrvoje	Petrović	1708982012003	17.08.1982	10000
35648	Igor	Ivčić	0203976370211	02.03.1976	21000
35765	Ana	Antolić	0808980452211	08.08.1980	32000
36781	Mislav	Bistričić	0311981356685	03.11.1981	20000
47678	Ljubica	Šaput	2412982700121	24.12.1982	44000
58445	Jelena	Župan	0209979122134	02.09.1979	
2010623	Pero	Perić		01.12.1987	

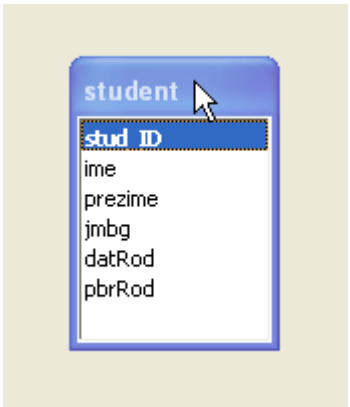
Kandidati:

stud_ID	ime	prezime	jmbg	datRod	pbrRod	uvjeti
536781	Mislav	Bilić	1311981356685		20000	Da
510001	Anka	Matić	2106980401236	11.06.1980	10000	Da
2010623	Pero	Petrić		01.12.1987		Ne
558445	Jelena	Jelić	0209879122134	02.09.1979		Ne
547678	Rade	Šuput	1412982700121	24.12.1982	44000	Ne
510234	Marko	Jozić	1311981302215		31000	Ne

Tablica studenata nakon ubacivanja podataka sadrži i podatke o novim studentima:

stud_ID	ime	prezime	jmbg	datRod	pbrRod
10001	Ana	Jurić	1106980401236	11.06.1980	10000
10234	Mario	Klarić	0311981302215	03.11.1981	31000
11010	Ivan	Korkut	0209979502247	02.09.1979	21000
11234	Ante	Jurić	0412979214665	04.12.1979	48000
12244	Marija	Knjaz	1109980325234	11.09.1980	51000
13475	Kristina	Kušan	0202980122201	02.02.1980	31000
21112	Ivan	Županović	0311981376432	03.11.1981	31000
21331	Petar	Šoljić	1211981879356	12.12.1981	22000
22222	Mirta	Lončar	2709980127782	27.09.1980	52000
25367	Davor	Dujmić	2504982561131	25.04.1982	10000
27783	Franjo	Mušnjak	1105981200211	11.05.1981	33000
34211	Petra	Krleža	0506978657631	05.06.1978	10000
34678	Jurica	Domanovac	0606978543121	05.06.1978	10000
34879	Hrvoje	Petrović	1708982012003	17.08.1982	10000
35648	Igor	Ivčić	0203976370211	02.03.1976	21000
35765	Ana	Antolić	0808980452211	08.08.1980	32000
36781	Mislav	Bistričić	0311981356685	03.11.1981	20000
47678	Ljubica	Šaput	2412982700121	24.12.1982	44000
58445	Jelena	Župan	0209979122134	02.09.1979	
510001	Anka	Matić	2106980401236	11.06.1980	10000
536781	Mislav	Bilić	1311981356685		20000
2010623	Pero	Perić		01.12.1987	
0					0

6. *Ažuriranje podataka – naredba update*



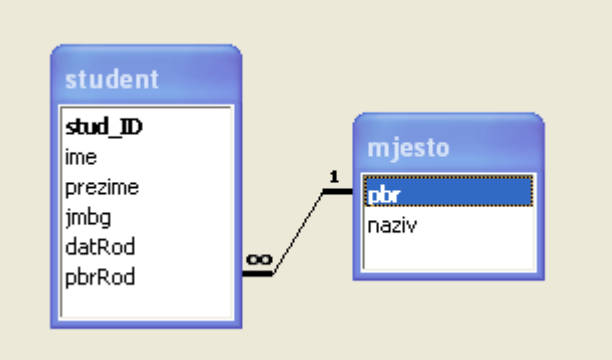
Primjer 1

Imamo tablicu student u kojoj su stud_id, ime i prezime obavezno podaci (required, odnosno not null).
Podatak o datumu i mjestu rođenja za studenta nije obavezan, kao ni jmbg.

Svim studentima kojima je pbrRod = 10000 želimo postaviti prezime u Matić.

```
update student
set prezime = 'Matić'
where pbrRod = 10000;
```

Primjer 2

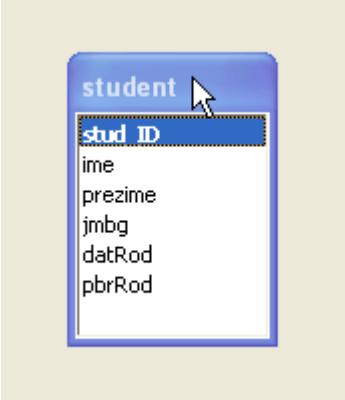


Želimo postaviti ime u „Bjelovarac" svim studentima koji su rođeni u Bjelovaru.

```
update student
set ime = 'Bjelovarac'
where pbrRod in
      (select pbr from mjesto where naziv = 'Bjelovar');
```

7. *Brisanje podataka – naredba delete*

Primjer 1



Želimo iz baze obrisati studenta sa šifrom studenta 12345 (kojem je stud_id = 12345).

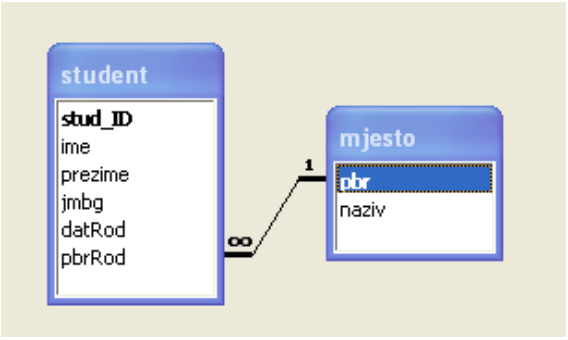
```
delete from student
where stud_id = 12345;
```

Primjer 2

Želimo iz baze obrisati sve studente rođene prije 1950. godine

```
delete from student
where year(pbrRod) < 1950;
```

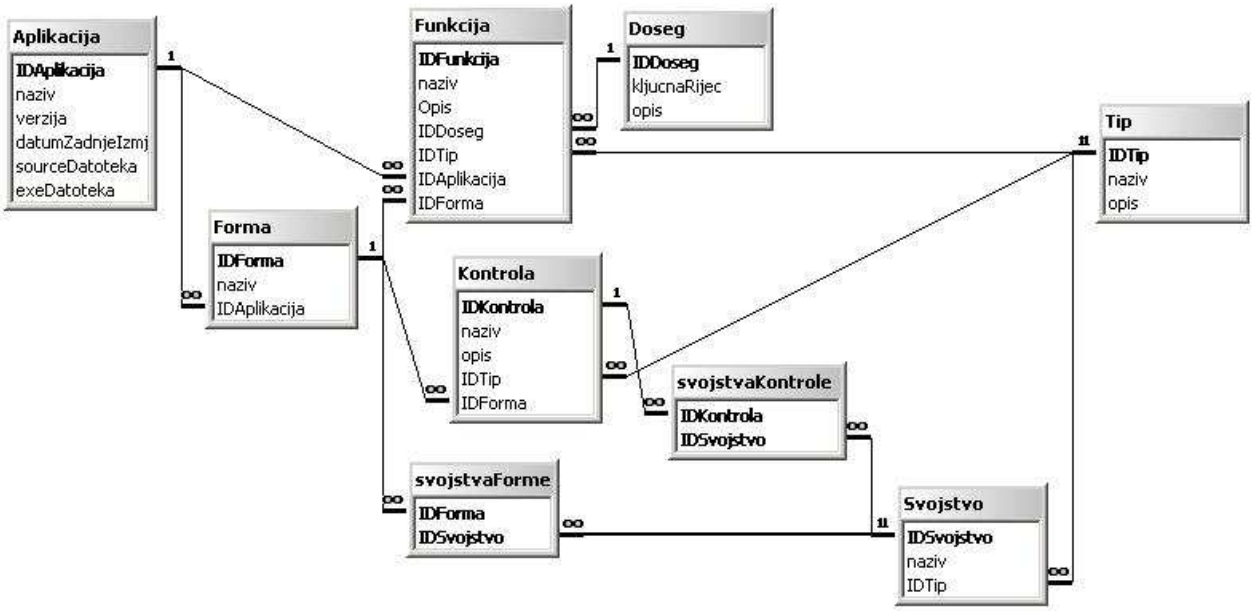
Primjer 3



Želimo obrisati sve studente iz Bjelovara.

```
delete from student
where pbrRod in
      (select pbr from mjesto where naziv = 'Bjelovar');
```

8. *Primjer - Nad bazom prikazanom na slici napraviti SQL upite*



Uputa – iako baza izgleda kompleksno, kod rješavanja svakog od zadataka potrebno se fokusirati samo na jedan manji dio dijagrama.
Dobro pročitati zadatak, zaključiti koje tablice i veze je potrebno koristiti i promatrati samo taj dio dijagrama

a) Izlistati sve aplikacije (IDAplikacija i naziv iz tablice Aplikacija).

```
select IDAplikacija, naziv from aplikacija;
```

b) Izlistati popis aplikacija (sve podatke) s njihovih funkcijama (naziv i opis funkcije) uz pretpostavku da svaka aplikacija ima barem jednu funkciju.

```
select a.*, f.naziv, f.opis
from aplikacija a INNER JOIN funkcija f on a.IDAplikacija = f.IDAplikacija;
```

c) Izlistati sve različite ključne riječi iz tablice Doseg

```
select DISTINCT kljucnaRijec
from Doseg;
```

d) Ispisati nazive svih formi koje pripadaju aplikaciji „Dummy aplikacija“

```
select distinct Forma.naziv
from Forma inner join Aplikacija on Forma.IDAplikacija = Aplikacija.IDAplikacija
where aplikacija.naziv = „Dummy aplikacija“;
```

ili

```
select distinct Forma.naziv
from Forma, Aplikacija
where Forma.IDAplikacija = Aplikacija.IDAplikacija
and aplikacija.naziv = „Dummy aplikacija“;
```

ili

```
select naziv from Forma where idAplikcija IN (select A. idAplikcija from aplikacija A where
A.naziv = „Dummy aplikacija“);
```

e) Ispisati nazive svih formi

```
select naziv
from Forma;
```

f) Izlistati naziv i opis kontrola, te za svaku tip kontrole (naziv i opis)

```
select k.naziv, k.opis, t.naziv as tip_kontrole, t.opis  as opis_tipa_kontrole
from kontrola k INNER JOIN tip t on k.IDtip = t.IDtip;
```

ili

```
select k.naziv, k.opis, t.naziv, t.opis
from kontrola k, tip t
where k.IDtip = t.IDtip;
```

g) Dohvatiti nazive tipova koji se koriste kao tipovi kontrole (jedan naziv ispisati samo jednom)

```
select distinct t.naziv
from tip t INNER JOIN kontrola k on t.IDTip = k.IDTip;
```

ili

```
select naziv from tip
where IDTip in (select IDTip from Kontrola);
```

ili

```
select distinct t.naziv
from Kontrola k, Tip t
```

```
where k.IDTip = t.IDTip;
```

h) Izlistati popis kontrola (naziv i opis) i njihovih svojstava (naziv svojstva).

```
select k.naziv, k.opis, s.naziv as naziv_Svojstva
from kontrola k INNER JOIN (svojstva s INNER JOIN svojstvaKontrole sk on s.IDSvojstvo
= sk.IDSvojstvo) on k.IDKontrola = sk.IDKontrola;
```

ili

```
select k.naziv, k.opis, s.naziv
from kontrola k, svojstva s, svojstvaKontrole sk
where s.IDSvojstvo = sk.IDSvojstvo and k.IDKontrola = sk.IDKontrola;
```

(gledamo samo dio dijagrama koji prikazuje tablice Kontrola, svojstvoKontrole, Svojstvo i veze među njima. Tablica svojstvoKontrole je tablica veze (više prema više) između entiteta Kontrola i entiteta Svojstvo)

i) Napisati naredbu kojom se tablici Aplikacija dodaje atribut datumKreiranja. Atribut nikada ne može biti prazan.

```
alter table aplikacija
add column datumKreiranja datetime NOT NULL;
```

j) Dohvatiti sve IDAplikacija i nazive aplikacija koje nisu mijenjane više od 2 godine, a sourceDatoteka im je nepoznata (nije upisana). Upit mora biti neovisan o vremenu izvođenja.

```
Select IDAplikacija, naziv
From Aplikacija
Where sourcedatoteka is null
and datumZadnjeIzmj < dateserial(year(date())-2, month(date()), day(date()));
```

k) Dohvatiti sve forme zajedno s funkcijama koje im pripadaju, te odgovarajući tip funkcije. Na popisu trebaju biti sve forme (i one koje nemaju definiranu funkciju). Ispisati naziv forme, naziv funkcije i naziv odgovarajućeg tipa.

```
Select frm.IDForma, frm.naziv, fun.naziv, t.naziv
From forma frm left outer join (funkcija fun inner join tip t on fun.IDtip = t.IDtip) on
frm.IDForma = fun.IDForma;
```

l) Napišite naredbe kojima bi se iz tablice svojstvo obrisala svojstva koja se ne koriste kao svojstva formi.

```
delete from svojstvo where IDSvojstvo not im (select IDsvojstvo from svojstvaForme);
```

m) Za aplikaciju naziva „Dummy aplikacija“ postavite opis svih njezinih funkcija u „Dummy funkcija“.

```
update Funkcija
set opis = „Dummy funkcija“
where IDAplikacija in (select IDAplikacija from Aplikacija where naziv = „Dummy
aplikacija“);
```

n) Ispisati sve kontrole koje su povezane s aplikacijom „Dummy aplikacija“ (sve kontrole koje pripadaju formi koja pripada navedenoj aplikaciji).

```
Select distinct k.*
From (kontrola k inner join forma f on k.idForma = f.idForma) inner join Aplikacija A on
A.idAplikacija = f.idAplikacija
where A.naziv = „Dummy aplikacija“;
```

ili

```
Select distinct k.*
From kontrola k, forma f, Aplikacija A
where k.idForma = f.idForma and A.idAplikacija = f.idAplikacija
and A.naziv = „Dummy aplikacija“;
```

ili

```
select *
from kontrola k
where IDForma in
        (select Idforma from forma where IDAplikacija IN
                (select IDaplikacija from aplikacija where naziv = „Dummy aplikacija“));
```

o) Ispisati tipove koji se ne koriste

```
Select t.*
From Tip t
Where IDTip not in
        (select IDTip from Funkcija
        Union
        Select IDTip from Kontrola
        Union
        Select IDTip from Svojstvo);
```

Ili

```
select t.*
from Tip t
where IDTip not in (select IDTip from Funkcija)
and  IDTip not in (select IDTip from Kontrola)
and  IDTip not in (select IDTip from Svojstvo);
```

Objašnjenje: Pregledom slike možemo ustanoviti da je tablica Tip povezana s tablicama Funkcija, Kontrola i Svojstvo. Zaključujemo da neki definirani tip nije u upotrebi ukoliko trenutno niti jedna funkcija nije tog tipa, niti jedno svojstvo nije tog tipa, te niti jedna kontrola nije tog tipa.

p) U kojim sve zadacima je upotrebljena DML naredba?

Samo u zadatku i) gdje smo naredbom
alter table aplikacija add column datumKreiranja datetime NOT NULL;
dodali novi atribut (stupac) datumKreiranja u entitet (tablicu) Aplikacija. Uz novi atribut smo dodali i not null ograničenje.

9. Sortiranje rezultata upita – order by

Upiti kakve smo do sada radili vraćaju 0 ili više zapisa.

Ukoliko želimo rezultat sortirati prema jednom ili više atributa (stvarnih ili izračunatih), koristimo **ORDER BY**.

Sortiranje po nekom atributu može biti

- **ASC** (ascending) uzlazno (od manjih vrijednosti prema većima)
- **DESC** (descending) silazno (od većih vrijednosti prema manjima).

Ukoliko je ispušteno, default je ASC.

Sortirati se može po svim standardnim tipovima podataka (tekst, brojevi, datumi); sve vrijednosti koje možemo međusobno usporediti, možemo po njima i sortirati.

Sortirati možemo i po izračunatim vrijednostima.

Atributi po kojima sortiramo ne moraju nužno biti u select listi.

9.1. Primjeri – order by

Imamo tablicu osoba u kojoj između ostalog imamo atribute ime, prezime, spol i datrod (datum rođenja).

Primjer 1:

Želimo dohvatiti imena i prezimena osoba, ali sortirano od najstarijih prema mlađima:

```
SELECT ime, prezime FROM osoba
ORDER BY datrod
```

Ili

```
SELECT ime, prezime FROM osoba
ORDER BY datrod ASC
```

Primjer 2:

Želimo dohvatiti imena i prezimena osoba ali sortirano od mlađih prema starijima:

```
SELECT ime, prezime FROM osoba
ORDER BY datrod DESC
```

Primjer 3:

Želimo dohvatiti imena i prezimena osoba sortirano po spolu pa po datumu rođenja:

```
SELECT ime, prezime FROM osoba
ORDER BY spol, datrod
```

Primjer 4:

Želimo dohvatiti imena i prezimena osoba sortirano silazno po godištu, uzlazno po spolu i uzlazno po prezimeno:

```
SELECT ime, prezime FROM osoba
ORDER BY YEAR(datrod) DESC, spol, prezime
```

Primjer 5:

Želimo dohvatiti godište, ime, prezime i spol osoba sortirano po prezimenu i imenu:

```
SELECT year(datrod) as godiste, ime, prezime, spol  FROM osoba
ORDER BY prezime, ime
```

Primjer 5:

Neka je u tablici osobe i datum kad smo upisali svaki pojedini zapis u tablicu (dat_upisa).

Želimo dohvatiti ime, prezime i broj dana koji su prošli od upisa osobe u tablicu, sortirano silazno po broju dana koji su prošli od upisa osobe u tablicu:

```
SELECT ime, prezime, date() – dat_upisa as upisan_prije__dana FROM osoba
ORDER BY (date() – dat_upisa)
```

10. Agregiranje izlaznih rezultata

10.1. Agregatne funkcije, group by

Upiti kakve smo do sada radili vraćaju 0 ili više zapisa.

Ukoliko želimo nad rezultatom upita raditi dodatnu analizu zapisa (sumirati vrijednosti nekih atributa, izbrojiti koliko redaka nam je upit vratio i sl) koristimo agregatne funkcije.

Najpoznatije agregatne funkcije su:

- **SUM** - zbrajanje vrijednosti nekog numeričkog atributa
- **AVG** – računanje prosjeka nekog numeričkog atributa
- **MIN** – traženje minimuma
- **MAX** – traženje maksimuma

- **COUNT** – brojanje koliko redaka zadovoljava zadane uvjete

Grupiranje možemo napraviti nad svim zapisima koje naš upit vraća ili možemo napraviti grupiranje po nekom atributu (jednom ili više njih)

PRIMJERI:

Primjer 1: Upit koji vraća popis ocjena (student, predmet, ocjena):

```
select stud_id, pred_id, ocjena from ispit
where ocjena is not null;
```

Mate	Baze	1
Mate	Baze	2
Pero	Baze	5
Stipe	PMA	1
Stipe	PMA	4

Primer 1: Želimo naći ukupnu prosječnu ocjenu (jedinstveni prosjek za sve studente i sve predmete):

```
select avg(ocjena) from ispit where ocjena is not null;
```

prosječna ocjena = (1+2+5+1+4) / 5 = 13/5 = 2,6

2,6

Primjer 2: Želimo naći prosječne ocjene iz svih predmeta za koje imamo podataka (jedinstveni prosjek za sve studente, za svaki predmet posebno):

```
select pred_id, avg(ocjena)
from ispit where ocjena is not null
group by pred_id;
```

Baze	2,67
PMA	2,5

Baze: (1+2+5)/3=8/3=2,67 PMA: (1+4)/2=5/2=2,5

Primjer 3: Želimo naći prosječne ocjene svih studenata koji su izlazili na ispit:

```
select stud_id, avg(ocjena)
from ispit
group by stud_id;
```

Mate	1,5
Pero	5
Stipe	2,5

Mate: (1+2)/2=3/2=1,5
Pero: 5/1=5
Stipe: (1+4)/2 = 5/2 = 2,5

Napomena: Kod nekih baza, group by će i sortirati izlazne rezultate, no to nije generalno pravilo. Ukoliko želimo uz grupiranje sortirati rezultat moramo dodati i order by dio.

Smijemo koristiti samo ono što je u select listi.

Primjer 4: Prošli primjer + dodatno sortirati po stud_id:

```
select stud_id, avg(ocjena)
from ispit
group by stud_id
ORDER BY stud_id;
```

Ovdje ne možemo sortirati po prezimenu, datumu rođenja i sl, jer smo taj podatak grupiranjem „izgubili" iz rezultata.

Primjer 5: Želimo naći prosječne ocjene svih studenata koji su izlazili na ispit, sortirano od boljih prema lošijim prosjecima

```
select stud_id, avg(ocjena)  as prosjek
from ispit
group by stud_id
ORDER BY avg(ocjena) desc;
```

10.2. Agregatne funkcije, group by / having

Grupirani rezultat možemo dodatno filtrirati upotrebom ključne riječi **having.**
Nakon ključne riječi having pišemo uvjete nad zapisima dobivenim grupiranjem.

Primjer 4: Želimo dohvatiti samo studente čija je prosječna ocjena iznad 3,5

```
select stud_id, avg(ocjena)
from ispit
group by stud_id
having avg(ocjena) > 3,5;
```

Mate: (1+2)/2=3/2=1,5; 1,5 > 3,5? FALSE
Pero: 5/1=5; 5 > 3,5? TRUE
Stipe: (1+4)/2 = 5/2 = 2,5; 2,5 > 3,5? FALSE

Mate	1,5
Pero	5
Stipe	2,5

Primjer 5:
Koliko prosječni student ima godina?

```
select avg(year(now) - year(datrod)) as prosjecna_dob
from student
```

Primjer 6:

Koliko prosječni student ima godina, te koliko ukupno imamo studenata?

```
select count(*) as broj_studenata, avg(year(now) - year(datrod)) as prosjecna_dob
from student s
```

Napomena:

Neka imamo u bazi 2 studenta kojima nije unesen datum rođenja.

Takvi studenti ulaze u ukupan broj studenata, ali se njihove godine ne uzimaju u obzir prilikom računanja prosjeka godina.

```
select count(*) as broj_studenata, avg(year(now) - year(datrod)) as prosjecna_dob
from student s where datrod is not null;
```

broj_studenata	prosjecna dob
20	28,95

```
select count(*) as broj_studenata, avg(year(now) - year(datrod)) as prosjecna_dob
from student s where datrod is null;
```

broj_studenata	prosjecna dob
2	

```
select count(*) as broj_studenata, avg(year(now) - year(datrod)) as prosjecna_dob
from student s;
```

broj_studenata	prosjecna dob
22	28,95

Primjer 7:

Želimo prebrojati koliko studenata koji su položili neki ispit ispit s ocjenom većom od 3 živi u kojem gradu, dodatno grupiramo po godištu studenta.

Želimo da naš upit vrati samo ona mjesta, godišta i broj studenata kod kojih je broj studenata veći od 1.

Korak 1 - Studente koji su položili neki ispit s ocjenom 4 ili 5 možemo dobiti ovim upitom:

```
select *
from student s inner join ispit i on s.studid = i.stud_id
where ocjena > 3
```

studID	ime	prezime	jmbg	datRod	pbrRod	stud_ID	dat_ispit	pred_ID	nast_ID	ocjena	pbr	naziv
10001	Ana	Jurić	1106980401236		10000	10001	29.12.2001	1003	567	5	10000	Zagreb
10001	Ana	Jurić	1106980401236		10000	10001	11.06.2002	1002	123	5	10000	Zagreb
22222	Mirta	Lončar	2709980127782	27.09.1980	52000	22222	13.04.2001	1004	112	4	52000	Pula
25367	Davor	Dujmić	2504982561131	24.02.1982	10000	25367	14.10.1999	1003	258	4	10000	Zagreb
27783	Franjo	Mušnjak	1105981200211	11.05.1981	33000	27783	02.02.2002	1006	235	4	33000	Virovitica
34211	Petra	Krleža	0506978657631	05.06.1978	10000	34211	19.04.2000	1010	333	5	10000	Zagreb
34678	Jurica	Domanovac	0606978543121	05.06.1978	10000	34678	19.04.2000	1001	235	5	10000	Zagreb

Korak 2 - Želimo dodati i podatak o mjestu rođenja:

```
select * from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr where ocjena > 3
```

Korak 3 – od atributa nas zanimaju mjesto rođenja i godište studenta.

```
select stud_id, year(datRod), m.naziv
from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
where ocjena > 3
```

Dobili smo višestruke zapise za istog studenta tamo gdje je student položio više ispita s ocjenom većom od 3

stud_ID	Expr1	naziv
10001		Zagreb
10001		Zagreb
22222	1980	Pula
25367	1982	Zagreb
27783	1981	Virovitica
34211	1978	Zagreb
34678	1978	Zagreb

Korak 4 – Ukoliko ovo grupiramo, odnosno ukoliko prebrojimo koliko ima stud_id za svako mjesto i godište, takvi zapisi brojit će se višestruko, što ne daje pravu sliku broja studenata...

```
SELECT Year(datRod) AS godiste, m.naziv, count(i.stud_ID)
FROM (student AS s LEFT JOIN mjesto AS m ON s.pbrRod = m.pbr) INNER JOIN ispit AS i ON s.studID = i.stud_ID
WHERE i.ocjena>3
GROUP BY Year(datRod) , m.naziv;
```

godiste	naziv	Expr1002
	Zagreb	2
1978	Zagreb	2
1980	Pula	1
1981	Virovitica	1
1982	Zagreb	1

Ono što bi bilo potrebno napraviti jest izbrojati koliko ima različitih (distinct) stud_ID.

```
SELECT Year(datRod) AS godiste, m.naziv, count(distinct i.stud_ID)
FROM (student AS s LEFT JOIN mjesto AS m ON s.pbrRod = m.pbr) INNER JOIN ispit AS i ON s.studID = i.stud_ID
WHERE i.ocjena>3
GROUP BY Year(datRod) , m.naziv;
```

U većini baza ovo je uobičajen način prebrojavanja različitih vrijednosti, no u MS Accessu count distinct nije implementirano.

Zbog toga radimo zaobilaženje tog problema, odnosno „work-around".

Vratimo se korak unatrag. Upit koji nam daje samo različite studente jest upit iz koraka 3 uz dodatak ključne riječi distinct:

```
select DISTINCT stud_id, year(datRod) as godiste, m.naziv
from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
where ocjena > 3;
```

Taj upit možemo tretirati kao pod-upit za prebrojavanje vrijednosti:

```
select naziv, godiste, count(stud_id)  as broj_studenata
from
(
    select DISTINCT stud_id, year(datRod) as godiste, m.naziv
    from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
    where ocjena > 3
) as X
group by naziv, godiste
```

Dobivamo očekivani rezultat:

naziv	godiste	broj_ studenata
Pula	1980	1
Virovitica	1981	1
Zagreb		1
Zagreb	1978	2
Zagreb	1982	1

Korak 5 – U prethodnom koraku dobili smo kao rezultat broj studenata koji su položili neki ispit s ocjenom većom od 3, grupirano po mjestu rođenja i godištu studenta.

Trebamo dodatno filtrirati samo one zapise kod kojih je dobiveni broj studenata >1.

```
select naziv, godiste, count(stud_id)  as broj_studenata
from
(
    select DISTINCT stud_id, year(datRod) as godiste, m.naziv
    from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
    where ocjena > 3
) as X
group by naziv, godiste
HAVING count(stud_id) > 1;
```

Jedino zapis (Zagreb, 1978, 2) odgovara kriteriju da je atribut broj_studenata > 1.
Zato je rezultat upita:

naziv	godiste	broj_ studenata
Zagreb	1978	2

Za bazu u kojoj je podržano select distinct, rješenje zadatka bi bilo:

```
select year(datRod) as godiste, m.naziv, count(DISTINCT stud_id)
from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
where ocjena > 3
group by year(datRod), m.naziv
having count(distinct stud_id) > 1;
```

No, u Accessu moramo zaobilazno:

```
select naziv, godiste, count(stud_id)  as broj_studenata
from
(
    select DISTINCT stud_id, year(datRod) as godiste, m.naziv
    from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
    where ocjena > 3
) as X
group by naziv, godiste
HAVING count(stud_id) > 1;
```

Napomena:

U HAVING dijelu upita možemo koristiti agregirane vrijednosti kao i atribute po kojima smo grupirali.

Primjerice, mogli smo filtrirati samo zagrepčane:

```
select naziv, godiste, count(stud_id)  as broj_studenata
from
(
    select DISTINCT stud_id, year(datRod) as godiste, m.naziv
    from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
    where ocjena > 3
) as X
group by naziv, godiste
having naziv = 'Zagreb'
```

Upit će raditi, no u praksi je bolje izbjegavati jer je efikasnije filtrirati po atributu kojeg grupiramo u where uvjetu (u našem primjeru, u podselectu dodajemo

```
    and mjesto = 'Zagreb'
    )
```

U ovakvim slučajevima bazi je efikasnije napraviti ranije filtriranje zapisa nego grupirati sve zapise i nakon toga neke od njih filtrirati.

Zbog toga je upit koji radi filtriranje mjesta Zagreb u where uvjetu bolje optimiziran od prethodno navedenog koji obavlja filtriranje u having dijelu.

```
select naziv, godiste, count(stud_id)  as broj_studenata
from
(
        select DISTINCT stud_id, year(datRod) as godiste, m.naziv
        from (student s inner join ispit i on s.studid = i.stud_id) left join mjesto m on s.pbrrod = m.pbr
        where ocjena > 3 and naziv = 'Zagreb'
) as X
group by naziv, godiste;
```

Oba upita vraćaju jednaki rezultat.

11. FUNKCIJE I OPERATORI

Većina baza nudi mogučnosti kreiranja vlastitih funkcija, no nudi i skup standardnih pred-definiranih funkcija koje se mogu koristiti u upitima, jednako kao što nudi i standardne operatore zbrajanja, množenja, dijeljenja, uspoređivanja i sl.

Funkcija kao ulazni parametar može primati 0 ili više vrijednosti (istog ili različitih tipova podataka), neki od ulaznih parametara mogu biti opcioni (ne moramo ih nužno navesti u pozivu funkcije), a uvijek vraća 1 rezultat točno određenog tipa.

Važno nam je znati koliko parametara funkcija prima, kojeg su oni tipa, logički što vraća i kojeg je tipa njezin rezultat.

Također, korisno je provjeriti kako pojedina funkcija tretira NULL vrijednosti ukoliko se koristi nad atributom koji u bazi nije obavezan.

Funkciju nad nekim atributom (atributima) možemo koristiti u svim elementima upita po jednakim pravilima kao i same atribute:

- u select listi
  - select year(datRod), prezime from student
- u where uvjetu
  - select * from student where jmbg like '10%' and year datRod = 1985
- u group by dijelu upita
  - select year(datrod), count(*) from student group by year(datRod)
  - select max(prezime) from student group by year(datRod)
- u order by dijelu upita
  - select * from student order by left(prezime, 1), strreverse(ime)
- u having dijelu upita
  - select year(datrod), max(prezime) from student group by year(datRod) HAVING left(max(prezime), 1) = 'A'
  - select year(datrod), max(datRod) from student group by year(datrod) having month(max(datrod)) <= 10

11.1.1. Datumske funkcije

Sistemski datum/vrijeme – ove funkcije ne primaju nikakav parametar a rezultat je datetime:

- NOW() – trenutno vrijeme dobiveno od operacijskog sustava
- NOW() se evaluira kao vrijeme ″5.11.2010. 10:30:01″ ako je današnji datum 5. studeni 2010. i vrijeme je 10:30:01.
- Vraća datetime. Koristimo kad je potrebno zabilježiti i datum i vrijeme

- DATE ()– trenutni datum dobiven od operacijskog sustava.
- DATE() se evaluira kao ″5.11.2010." ako je današnji datum 5. studeni 2010. Koristimo kad je potrebno evidentirati samo datum a ne i vrijeme.

Funkcije koje kao parametar primaju datum (datetime) a rezultat je integer:

- WEEKDAY – redni broj dana u tjednu za zadani datum (1-nedjelja)
- WEEKDAY(NOW()) = 2 ako je danas ponedjeljak

- DAY – broj dana za zadani datum.
- DAY(#03/21/2010#) = 21, day(#2/21/1923#) = 21

- MONTH – za zadani datum vraća broj mjeseca.
- MONTH(#03/21/2010#) =3, month(#2/21/1923#) = 2

- YEAR – redni broj godine za zadani datum.
- YEAR(#03/21/2010#) = 2010, year(#2/21/1923#) = 1923

Funkcija koje kao parametar prima 3 integera iz kojih "složi" datum:

- DATESERIAL – evaluira datum iz 3 INT vrijednosti (godina, mjesec, dan).
- DATESERIAL(2010, 3, 21) je datumska vrijednost 21.03.2010, odnosno vrijednost #3/21/2010#

Upiti u MS Accessu podrazumijevaju da je zadan US format datuma mm/dd/yyyy.

Primjeri:

SELECT * FROM student WHERE student.datRod=#1/15/1983#;

dohvaća sve studente koji su rođeni 15.1.1983. godine

SELECT * FROM student WHERE year(datRod)= 1983;

dohvaća sve studente koji su rođeni 1983. godine

SELECT * FROM student

WHERE dateserial (year(datRod)+42, month(datRod), day(datrod)) <= date();

dohvaća sve studente koji su navršili 42 godine.

Uzmimo da je student Pero rođen 1.1.1943. godine, a student Mate 15.10.1989.
Razložimo:
year(datRod) 'čupa' godinu rođenja
year(datRod) + 42 je godina u kojoj promatrani student slavi 42. rođendan
month(datRod) i day(datRod) su im mjesec i dan rođenja

Za Matu: Year(datRod) = 1943, Year(datRod) + 42 = 1943+42 = 1985, Month(datRod) = 1, Day(datRod) = 1
Za Peru: Year(datRod) = 1989, Year(datRod) + 42 = 1989+42 = 2031, Month(datRod) = 10, Day(datRod) = 15

Dateserial nam slaže datum iz dobivenih vrijednosti koji daje datum na koji svaki od njih slavi 42. ruđendan:

Za Matu: dateserial(1985, 1, 1) daje datum 1.1.1985
Za Peru: dateserial(2031, 10, 15) daje datum 15.10.2031. godine

Date() nam vraća trenutni datum.

Uvjet
dateserial (year(datRod)+42, month(datRod), day(datrod)) <= date();
nam testira da li datum kad je njihov 42. rođendan već prošao ili nije.

Ako je danas 15.1.2011, Mate će zadovoljiti uvjet (1.1.1985 je manji od 15.1.2011) , a Pero neće (15.10.2031 je veći od 15.1.2011).

11.1.2. Format Property - Date/Time Data Type

Možete podesiti **Format** property na neko od ponuđenih vrijednosti ili možete koristiti custom format za Date/Time atribut.

Preddefinirani formati

Setting	Opis
General Date	(Default) Ako je unesena samo datumska vrijednost, sati se ne prikazuju. Ukoliko je uneseno samo vrijeme, datum se ne prikazuje. Ovaj setting je kombinacija ShortDate i Long Time settingsa.. Npr: 4/3/93, 05:34:00 PM
Long Date	Isto kao Long Date setting u regional settingsima Windowsa. Npr: Saturday, April 3, 1993.
Medium Date	Npr: 3-Apr-93.
Short Date	Kao Short Date setting u regional settingsima Windowsa. Npr: 4/3/93. **Warning** Short Date podrazumijeva da su datumi u 21. stoljeću: 1/1/00 - 12/31/29 su prevedeni kao 01/01/2000 - 12/31/2029. Datumi 1/1/30 - 12/31/99 prevode se kao 20. stoljeće (1930 - 1999).
Long Time	Isti setting kao **Time** u regional settingsima. Npr: 5:34:23 PM.
Medium Time	Npr: 5:34 PM.
Short Time	Npr: 17:34.

Custom Formati

Možete kreirati custom format date/time korištenjem simbola:

Symbol	Description
: (colon)	Separator vremena (separatori se postavljaju u Windows regional settings)
/	Separator datuma.
c	Isto kao General Date format.
d	Dan u mjesecu s 1 ili 2 znamenke (1 do 31).
dd	Dan u mjesecu s 2 znamenke (01 do 31).

ddd	Prva tri znaka dana u tjednu (Ned do Sub).
dddd	Pun naziv dana (Nedjelja do Subota).
ddddd	Kao Short Date format.
dddddd	Kao Long Date format.
w	Dan u tjednu (1 to 7).
ww	Tjedan u godini (1 to 53).
m	Mjesec napisan s 1 ili 2 znamenke, po potrebi (1 to 12).
mm	Mjesec napisan s 2 znamenke (01 to 12).
mmm	Prva tri slova naziva (Sij do Pro).
mmmm	Pun naziv mjeseca (Siječanj do Prosinac).
q	Kvartal (1 do 4).
y	Redni broj dana u godini (1 do 366).
yy	Kratki zapis godine - zadnje 2 znamenke (01 do 99).
yyyy	Godina (0100 do 9999).
h	Sat s 1 ili 2 znamenke, po potrebi (0 do 23).
hh	Sat napisan s 2 znamenke (00 do 23).
n	Minute, 1 ili 2 znamenke (0 do 59).
nn	Minute zapisane s 2 znamenke (00 do 59).
s	Sekunde, kraći zapis (0 do 59).
ss	Sekunde, zapisane s 2 znamenke (00 do 59).
ttttt	Kao Long Time format.
AM/PM	12-satni zapis sata s "AM" / "PM", prema potrebi.
am/pm	12-satni zapis sata s "am" / "pm" (malim slovima), prema potrebi.
A/P	12-satni zapis sata s "A" / "P", prema potrebi.
a/p	12-satni zapis sata s "a" / "p", prema potrebi.
AMPM	12-satni zapis sata s oznakom jutra/popodneva kakva je postavljena u regional settings.

Custom formati se prikazuju u skladu sa postavkama windowsa (regional settings). Ignoriraju se oni Custom formati koji nisu u skladu s regional postavkama.

Napomena Ako primjerice želite dodati zarez ili neki drugi separator u custom format, dodajte ga u navodnicima. Npr: mmm d", "yyyy.

Primjeri custom date/time formata

Postavka	Prikaz
ddd", "mmm d", "yyyy	Mon, Jun 2, 1997
mmmm dd", "yyyy	June 02, 1997
"This is week number "ww	This is week number 22
"Today is "dddd	Today is Tuesday

Korištenje formata u upitima:

Primjer 1 – koji studenti nisu rođeni između 15.1. i 15.6.

1. način – u datum rođenja 'podmetnemo' tekuću godinu i usporedimo

```
SELECT s.*
FROM student AS s
WHERE DateSerial(Year(Now()),Month([datrod]),Day([datrod])) Not Between
DateSerial(Year(Now()),1,15) And DateSerial(Year(Now()),6,15))
```

2. način – složimo datumske varijable od zadanih vrijednosti u godini u kojoj se student rodio year(datrod) i usporedimo

```
SELECT s.*
FROM student AS s
WHERE
datrod not between dateserial(year(datrod), 1, 15) and dateserial(year(datRod), 6, 15)
```

3. način – „Iščupamo" mjesec i dan rođenja vani kao string i radimo usporedbu

```
SELECT s.*
FROM student AS s
WHERE Format([s.datRod],"mmdd") not between "0115" and "0615"
```

Napomena – kod pretvaranja datuma u tekst potrebno je paziti na uspoređivanje – „ddmm" bi dalo pogrešan rezultat

4. način – „pješke", and/or

```
SELECT s.*
FROM student AS s
WHERE month(datRod) = 1 and day(datRod) < 15
OR     month(datRod) = 6 and  day(datRod) > 15
OR month(datRod) >= 7
```

Ili

```
SELECT s.*
FROM student AS s
WHERE  NOT(
month(datRod) = 1 and day(datRod) >= 15
OR     month(datRod) = 6 and  day(datRod) <= 15
OR month(datRod) between 2 and 5
);
```

Primjer 2 – dodavanje/oduzimanje mjeseci, godina, dana,...

Osim do sada za to korištenih funkcija year, month, day, dateserial, možemo koristiti i funkciju DateAdd.

Dodavanje / oduzimanje mjeseci:

select datrod, **DateAdd("m",-1,datrod)** from student
ili
select datrod, dateserial(year(datrod), month(datrod)-1, day(datrod)) from student

Dodavanje / oduzimanje godina:

select datrod, **DateAdd("yyyy",-1,datrod)** from student
ili
select datrod, dateserial(year(datrod)-1, month(datrod), day(datrod)) from student

Primjer 3 – korisni datumi

Prvi u slijedećem mjesecu: DateSerial(Year([givendate]),Month([givendate])+1,1)
Zadnji u prošlom mjesecu: DateSerial(Year([givendate]),Month([givendate]),1)-1
Zadnji u ovom mjesecu: DateAdd("m",1,[givendate])-day([givendate])
Isti dan u prošlom mjesecu: DateAdd("m",-1,[givendate])
Isti datum prošle godine: DateAdd("yyyy",-1,[givendate])

Tjedni

Funkcija **Weekday** vraća broj dana u tjednu (1 do 7). Drugi argument koji se najčešće ispušta predstavlja prvi dan u tjednu (defaultno nedjelja = 1). To možemo iskoristiti da bismo dobili neki dan u prethodnom tjednu (npr prošli ponedjeljak, prošlu nedjelju itd)

1=nedjelja, 2=ponedjeljak, 3=utorak
Prethodni utorak u odnosu na [givendate]:
[givendate]-Weekday([givendate],3)

Radni dani: Weekday([givendate]) < 6
Vraća True(-1) ili False (0)

Neradni dani: Weekday([givendate]) >= 6
Vraća True(-1) ili False (0)

Naziv dana u tjednu može se dobiti funkcijom weekdayname koja prima vrijednosti od 1 (nedjelja) do 7 (subota), kakve vraća funkcija weekday. Vraća #Error za null vrijednosti.

```
SELECT stud_id, datrod, year(datrod), month(datRod), weekday(datRod),
weekdayname(weekday(datrod))
from student;
```

11.2. Rad s tekstualnim atributima

11.2.1. Često korištene string funkcije

- **&** ili **+** rade konkatenaciju (spajanje) stringova u jedan.
  - select ime & ' ' & prezime from student → vraća ime i prezime kao da su jedinstveni atribut

- **LEN**(string) vraća duljinu teksta. Npr LEN('baze') = 4

- **STRREVERSE**(string) vraća obrnuti poredak znakova. Npr STRREVERSE('baze') = 'ezab'

- **LCASE**(string) / **UCASE**(string) pretvaraju sva slova u mala/velika
  - select LCASE(ime), UCASE(prezime), LEN(prezime) from student → vratit će imena napisana malim slovom, a prezimena velikim slovom. Dodatno će se izračunati i duljina svakog prezimena
  - LCASE('Mate Matić') = 'mate matić'
  - UCASE('Mate Matić') = 'MATE MATIĆ

- **MID**(string, start, num) je funkcija koja iz stringa izrezuje podniz duljine num počevši od znaka na poziciji start
  - select mid(ime, 3, 2) from student → za studente Stipe, Mate, Pero, Anamaria vratit će ip, te, to, am

- **LEFT**(string, broj_znakova), **RIGHT**(string, broj_znakova) → vraća zadani broj znakova od početka / kraja stringa
  - **LEFT(string, n) == MID(string, 1, n)**
  - **RIGHT(string, n) == MID(string, len(string) – n + 1, n)**

- **INSTR**(*u_cemu_trazi*, *sto_trazi*) → traži string *sto_trazi* unutar stringa *u_cemu_trazi* i vraća poziciju unutar stringa *u_cemu_trazi* na kojoj je pronađen niz znakova *sto_trazim* ili 0 ako nije pronađen
  - **INSTR ('ana', 'a') = 1**
  - **INSTR('banana', 'ana') = 2**
  - **INSTR('car', 'oko') = 0**
  - **Opciono, možemo precizirati i poziciju u prvom stringu od koje nadalje tražimo: INSTR(3, 'banana', 'ana') = 4**

Primjeri:

SELECT ime, MID(ime, 3, 2) as Mid_3_2, LEFT(ime, 3) as Left_3, RIGHT(ime, 3) as Right_3 FROM student;

ime	Mid_3_2	Left_3	Right_3
sss	s	sss	sss
Ana	a	Ana	Ana
Mario	ri	Mar	rio
Ivan	an	Iva	van
Ante	te	Ant	nte
Marija	ri	Mar	ija
Kristina	is	Kri	ina
Ivan	an	Iva	van
Petar	ta	Pet	tar
Mirta	rt	Mir	rta
Davor	vo	Dav	vor
Franjo	an	Fra	njo
Petra	tr	Pet	tra
Jurica	ri	Jur	ica
Hrvoje	vo	Hrv	oje
Igor	or	Igo	gor
Ana	a	Ana	Ana
Mislav	sl	Mis	lav
Ljubica	ub	Lju	ica
Boris	ri	Bor	ris
Jelena	le	Jel	ena
Suzana	za	Suz	ana

SELECT ime, len(ime) as "len(ime)", :n as N, LEFT(ime, :n) as n_left, MID(ime, 1, :n) as "MID(ime, 1, :n)",
RIGHT(ime, :n) as n_right, MID(ime, IIF(len(ime) > :n, len(ime)-:n +1, 1), :n) as "MID(ime, IIF(len(ime) > :n, len(ime)-:n +1, 1), :n)"
FROM student;

ime	"len(ime)"	N	n_left	"MID(ime, 1, :n)"	n_right	"MID(ime, IIF(len(ime) > :n, len(ime)-:n +1, 1), :n)"
sss	3	6	sss	sss	sss	sss
Ana	3	6	Ana	Ana	Ana	Ana
Mario	5	6	Mario	Mario	Mario	Mario
Ivan	4	6	Ivan	Ivan	Ivan	Ivan
Ante	4	6	Ante	Ante	Ante	Ante
Marija	6	6	Marija	Marija	Marija	Marija
Kristina	8	6	Kristi	Kristi	istina	istina
Ivan	4	6	Ivan	Ivan	Ivan	Ivan
Petar	5	6	Petar	Petar	Petar	Petar
Mirta	5	6	Mirta	Mirta	Mirta	Mirta
Davor	5	6	Davor	Davor	Davor	Davor
Franjo	6	6	Franjo	Franjo	Franjo	Franjo
Petra	5	6	Petra	Petra	Petra	Petra
Jurica	6	6	Jurica	Jurica	Jurica	Jurica
Hrvoje	6	6	Hrvoje	Hrvoje	Hrvoje	Hrvoje
Igor	4	6	Igor	Igor	Igor	Igor
Ana	3	6	Ana	Ana	Ana	Ana
Mislav	6	6	Mislav	Mislav	Mislav	Mislav
Ljubica	7	6	Ljubic	Ljubic	jubica	jubica
Boris	5	6	Boris	Boris	Boris	Boris
Jelena	6	6	Jelena	Jelena	Jelena	Jelena

Napomena - Korištenje parametra u upitu

Ukoliko u upitu napišemo ***:ime_varijable***, MS access nas prije izvođenja upita pita da unesemo vrijednost te varijable.

Što se same sintakse upita tiče, varijablu :ime_varijable koristimo na isti način kao da je atribut zapisan negdje u bazi kojeg smo dohvatili ili vrijednost koju smo u upitu eksplicitno naveli.

11.2.2. Operator LIKE

Da bismo provjerili da li tekst zapisan u nekom polju odgovara zadanom uzorku, koristimo operator LIKE (u nekim bazama MATCHES).

... where *tekstualni_atribut* like '*neki_uzorak*'

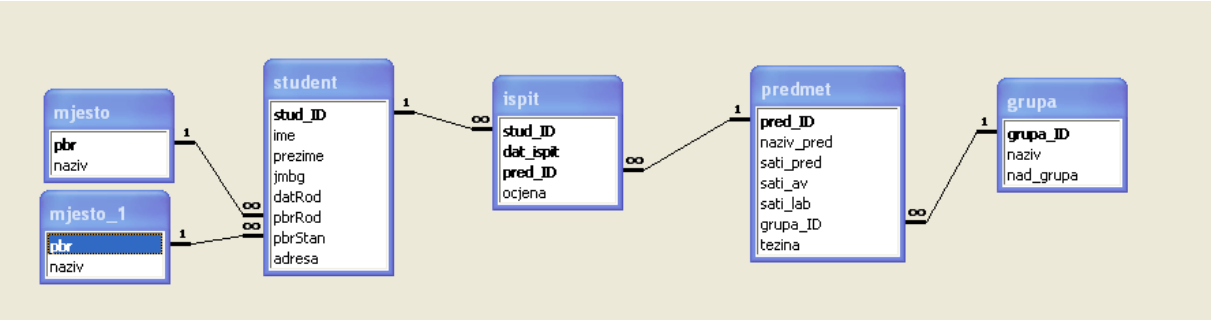
Kod zadavanja uzorka:

- * označava 0 ili više bilo kojih znakova,
- ? označava točno jedan znak (bilo koji).
- [ABC123] označava jedan od nabrojanih znakova (ili A ili B ili C ili 1 ili 2 ili 3)
- [^ABC123] označava bilo koji znak osim jedan od nabrojanih znakova (ne A ni B ni C ni 1 ni 2 ni 3)
- [A-E] označava bilo koji znak iz zadanog intervala (A, B, C, ..., E)
- Ukoliko želimo u uzorku specificirati jedan od specijalnih znakova (*, ?,..), moramo koristiti 'escape' znak \. Primjerice A*\? specificira tekst koji počinje slovom A a završava upitnikom.

Primjeri:

- **prezime like 'A*ić'** → uvjet zadovoljavaju Anić, Antić, Antunović. Uvjet ne zadovoljava Perić, Matić-Anić, Anteć, Barić
- **ime like '[AB]*a'** → uvjet zadovoljavaju Ana, Anita, Anka, Barbara, Branka. uvjet ne zadovoljava niti jedno ime koje ne počinje na A ili B i koje ne završava s a, npr Ante, Branko, Mate, Zorica
- **LCASE(ime) like** "*a*" → ime koje u sebi sadrži slovo a (veliko ili malo). Uvjet je zadovoljen za, primjerice: Ante, Branko, Anita, Marko, a nije ispunjen npr za Tin, Mirko...
- **UCASE(ime) like** "*a*" → Nikada nije ispunjeno jer ucase sva slova pretvara u velika slova, pa niti jedno od njih nije malo 'a'

Napomena – MS Access upiti nisu osjetljivi na velika i mala slova, što nije generalno pravilo.



Zadatak1: Naći prosječnu ocjenu studenata koji su izlazili na ispite grupirano prema predzadnjem slovu prezimena

Predzadnje slovo prezimena možemo izdvojiti korištenjem funkcijem mid i len ili korištenjem funkcija left i right:
mid(prezime, len(prezime) -1, 1)
ili left(right(prezime, 2), 1)

```
select mid(prezime,  len(prezime) -1, 1) as slovo, avg(ocjena) as prosjek
from student s inner join ispit i on s.stud_id = i.stud_id
group by mid(prezime,  len(prezime) -1, 1);
```

Zadatak 2: Naći prosječnu ocjenu svih studenata koji su izlazili na ispite za svaki predmet, ali samo za one studente kojima ime ne počinje samoglasnikom, a prezime im završava na „ić“ i dugačko je najviše 8 znakova.

```
select p.naziv, avg(ocjena) as prosjek
from (student s inner join ispit i on s.stud_id = i.stud_id) inner join
        predmet p on i.pred_id = p.pred_id
where s.ime not like '[AEIOU]*' and s.prezime like '*ić' and len(s.prezime) <= 8
group by p.naziv;
```

Zadatak 3: Izračunajte za svaku grupu predmeta ukupni broj sati labosa, ukupni broj sati predavanja i ukupni broj sati auditornih vježbi, te sumu svih potrebnih sati (i predavanja i labosa i auditornih vježbi). Rezultat neka bude silazno sortiran po zbroju svih sati, te ukoliko više grupa ima isti ukupni broj sati neka je rezuktat dodatno sortiran po nazivu grupe.

```
SELECT g.naziv, sum(sati_pred) as total_pred, sum(sati_av) as total_av,
  sum(sati_lab) as total_lab, sum(sati_pred + sati_av +sati_lab) as total_sati
FROM grupa g INNER JOIN predmet p ON g.grupa_id=p.grupa_id
group by g.naziv
order by sum(sati_pred + sati_av +sati_lab)  DESC, g.naziv
```

Zadatak 4: Ispisati po abecedi (prvo po prezimenu a onda po imenu) sve studente (ime i prezime) koji su izašli na barem 1 ispit, te njihov prosjek.

```
select ime, prezime, avg(ocjena) as prosjek, count(*) as broj_izlazaka_na_ispit
from student s inner join ispit i on s.stud_id = i.stud_id
group by ime, prezime
order by prezime, ime
```

Napomena:
Što ukoliko u bazi imamo studente koji se jednako zovu?

Ovaj upit će nam ih tretirati kao jednu osobu i izračunat će njihov ukupni prosjek ocjena.
Ukoliko želimo izbjeći takve situacije, u grupiranje trebamo dodati i šifru studenta koja je za svakog studenta jedinstvena (primarni ključ stud_id)

Ukoliko ne želimo da se podatak o stud_id ispiše, možemo ga dodati samo u group by dio:

```
select ime, prezime, avg(ocjena) as prosjek, count(*) as broj_izlazaka_na_ispit
from student s inner join ispit i on s.studid = i.stud_id
group by ime, prezime, stud_id
order by prezime, ime
```

Ukoliko imamo 2 studenta jednakog imena i prezimena, rezultat ovog upita će vratiti 2 zapisa s istim imenom i prezimenom, eventualno s različitim prosjecima.
Obzirom da u rezultatu nije jednoznačno kojem studentu od te dvojice s istim imenom i prezimenom koji prosjek pripada, bolja praksa je dodati stud_id i u select listu:

```
select stud_id, ime, prezime, avg(ocjena) as prosjek, count(*) as broj_izlazaka_na_ispit
from student s inner join ispit i on s.studid = i.stud_id
group by ime, prezime, stud_id
order by prezime, ime
```

Alternativno, možemo napraviti grupiranje samo po stud_id samo koristeći tablicu ispit, pa naknadno join s tablicom student:

```
select stud_ID, ime, prezime,  prosjek, broj_izlazaka_na_ispit
from student s inner join
(
        select stud_id, avg(ocjena) as prosjek, count(*) as broj_izlazaka_na_ispit
        from ispit i
        group by stud_id
) as X
on s.stud_ID = X.stud_ID
```

U ovom slučaju upit X vraća samo 1 redak po studentu (zbog grupiranja po stud_id pdoatku). Zbog toga dodatno grupiranje nakon spajanja X i tablice student nije potrebno (1:1).

11.3. Logički operatori OR, AND, NOT

Uvjet je izraz koji se za određenu vrijednost izračunava kao istinit ili lažan.

Primjerice,
u bazi studenata imamo Peru iz Splita i Matu iz Zagreba
select * from student where pbrRod = 10000

Baza će izračunavajući rezultat upita prolaziti kroz sve retke tablice student i za svakog studenta izračunati da li je (pbrRod = 10000) istina ili laž.
Za Peru će biti: (21000 = 10000) = FALSE i Pero će otpasti iz skupa izlaznih rezultata
Za Matu će biti: (10000 = 10000) = TRUE i zapis o Mati ostaje u izlaznom rezultatu.

Složeni uvjet tvori se od više jednostavnih uvjeta korištenjem logičkih operatora NOT, AND, OR.

- NOT (UVJET X) je zadovoljen ukoliko uvjet X nije zadovoljen
  - NOT (pbr = 10000) isto kao pbr <> 10000
- UVJET1 AND UVJET2 je zadovoljen ukoliko su zadovoljeni i uvjet1 i uvjet2
  - Pbr = 10000 and prezime = "Jurić" – uvjet je TRUE samo za Juriće iz Zagreba
- NOT (UVJET1 AND UVJET2) je zadovoljen ukoliko barem jedan od uvjet1 i uvjet2 nije zadovoljen
  - NOT (Pbr = 10000 and prezime = "Jurić") – uvjet je FALSE samo za Juriće iz Zagreba. True je za Juriće koji nisu iz ZG i za ljude iz ZG koji se ne prezivaju Jurić
- UVJET1 OR UVJET2 je zadovoljen ukoliko je zadovoljen bar jedan od nabrojanih uvjeta (ili uvjet1 ili uvjet2 ili oba)
  - Pbr = 10000 OR prezime = "Jurić" – TRUE za sve ljude iz ZG i za sve koji se prezivaju Jurić
- NOT (UVJET1 OR UVJET2) je zadovoljen ukoliko nije zadovoljen niti jedan od nabrojanih uvjeta (ni uvjet1 ni uvjet2)
  - NOT(Pbr = 10000 OR prezime = "Jurić") – TRUE za sve koji nisu iz Zagreba i ne prezivaju se Jurić

Primjer NOT

Želimo izlistati studente koji nisu iz Zagreba:
select * from student where NOT (pbrRod = 10000)

Baza će izračunavajući rezultat upita prolaziti kroz sve retke tablice student i za svakog studenta izračunati da li je (pbrRod = 10000) istina ili laž.
Za Peru će biti: NOT(21000 = 10000) = NOT(FALSE) = TRUE i Pero ostaje u izlaznom rezultatu
Za Matu će biti: NOT(10000 = 10000) = NOT(TRUE) = FALSE i zapis o Mati će otpasti iz skupa izlaznih rezultata

Primjer AND

Neka imamo u bazi osim Pere (koji je rođen 1991.) i Mate (rođen 1973.) i 2 studenta iz Bjelovara: Martin rođen 1991. i Matija koji je rođen 1983.

Želimo izlistati studente koji su iz Bjelovara i rođeni su 1991. godine:

```
select *
from student s inner join mjesto m on s.pbrRod = m.pbr
where m.naziv = 'Bjelovar' and year(datRod) = 1991
```

Pero i Mate ispadaju već na prvom testu:
(m.naziv = 'Bjalovar') = FALSE
Drugi uvjet (year(datRod) = 1991) ne treba niti provjeravati jer čak i ako je ispunjen, to neće biti dovoljno (FALSE and TRUE) = FALSE.
„AND" izaz će biti istinit samo ukoliko je svaki od navedenih izraza istinit.

Za Martina i Matiju prvi izraz je istinit ('Bjelovar' = 'Bjelovar') = TRUE i testira se drugi uvjet: true and YEAR(datrod) = 1991

Za Martina: TRUE AND (1991 = 1991) = TRUE AND TRUE = TRUE
Za Matiju: TRUE AND (1983= 1991) = TRUE AND FALSE = FALSE

Upit smo mogli zapisati i ovako:

```
select *
from student s, mjesto m
WHERE s.pbrRod = m.pbr
AND m.naziv = 'Bjelovar' and year(datRod) = 1991
```

Primjer OR

Želimo izlistati studente koji su iz ili Bjelovara ili su rođeni 1991. godine.
Dakle dovoljno je da je jedan od tih uvjeta bio zadovoljen da bi zapis o studentu ostao u izlaznim rezultatima.

```
select *
from student s inner join mjesto m on s.pbrRod = m.pbr
where m.naziv = 'Bjelovar' OR year(datRod) = 1991
```

Za Peru i Matu prvom testu: (m.naziv = 'Bjalovar') = FALSE i gleda se drugi uvjet (year(datRod) = 1991) :
Pero: FALSE OR (1991 = 1991)= FALSE OT TRUE = TRUE

Mate : FALSE OR (1973 = 1991)= FALSE OR FALSE = FALSE

Za Martina i Matiju prvi izraz je istinit ('Bjelovar' = 'Bjelovar') = TRUE i nije nužno testirati drugi uvjet YEAR(datrod) = 1991, jer čak i ako taj uvjet nije zadovoljen, ukupni uvjet je zadovoljen. TRUE OR FALSE = TRUE

Dakle, ovdje smo izbacili samo Matu, jer niti je iz Bjelovara niti je rođen 1991. godine.

Upit smo mogli zapisati i ovako:

```
select *
from student s, mjesto m
WHERE s.pbrRod = m.pbr
AND (m.naziv = 'Bjelovar' OR year(datRod) = 1991)
```

Ovdje je potrebno paziti da se složeni uvjet (Bjelovar ili 1991. godište) obavezno stavi u zagrade.
Ukoliko zagrade ispustimo, defaultno su zagrade postavljene ovako:

```
select *
from student s, mjesto m
WHERE (s.pbrRod = m.pbr AND m.naziv = 'Bjelovar' ) OR year(datRod) = 1991
```

pa će ovakav upit vratiti zapise iz joina tablica mjesto i student za koje je naziv mjesta Bjelovar i uz to zapise svih studenata koji su rođeni 1991. uz neki redak iz tablice mjesto (svaki student rođen 1991. će se pojaviti onoliko puta koliko je redaka u tablici mjesto, a ne samo za ono mjesto koje odgovara pbrRod)
Join se primjenjuje samo na prvi uvjet.

Ukoliko nema zagrada, prioriteti AND i OR se tretiraju jednako kao * i +
A and B or C and D or E = (A and B) or (C and D) or E
jednako kako bi se tretiralo A*B + C*D + E

- A or B and C = A or (B and C)
- not A or B and C = (not A) or (B and C)
- A and B or C = (A and B) or C
- not(X or Y) = (not X) and (not Y)

X	Y	X OR Y	NOT (X OR Y)	NOT X	NOT Y	NOT X AND NOT Y
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

- not (X and Y) = (not X) or (not Y)

Uvjet je moguće napisati na više različitih načina.

11.4. Ostale korisne funkcije

Funkcija ROUND radi zaokruživanje decimalnog broja na specificirani broj znamenaka. Prvi parametar je broj kojeg zaokružujemo, drugi (opcioni) je na koliko decimalnih znamenaka zaokružujemo. Ukoliko ga ispustimo, radi se cjelobrojno zaokruživanje.

Primjer:

```
SELECT datRod, now, now- datRod,
        round(now- datRod, 2), round(now- datRod, 1),  round(now- datRod)
from student;
```

datRod	now	br_dana	"round(now- datRod, 2)"	"round(now- datRod, 1)"	"round(now- datRod)"
11.06.1980	15.01.2011 14:54:36	11175,62125	11175,62	11175,6	11176
03.11.1981	15.01.2011 14:54:36	10665,62125	10665,62	10665,6	10666
02.09.1979	15.01.2011 14:54:36	11458,62125	11458,62	11458,6	11459
04.12.1979	15.01.2011 14:54:36	11365,62125	11365,62	11365,6	11366
11.09.1991	15.01.2011 14:54:36	7066,62125	7066,62	7066,6	7067
25.02.1987	15.01.2011 14:54:36	8725,62125	8725,62	8725,6	8726
03.11.1981	15.01.2011 14:54:36	10665,62125	10665,62	10665,6	10666
12.12.1981	15.01.2011 14:54:36	10626,62125	10626,62	10626,6	10627
27.09.1980	15.01.2011 14:54:36	11067,62125	11067,62	11067,6	11068
20.02.1982	15.01.2011 14:54:36	10556,62125	10556,62	10556,6	10557
11.05.1981	15.01.2011 14:54:36	10841,62125	10841,62	10841,6	10842
05.06.1978	15.01.2011 14:54:36	11912,62125	11912,62	11912,6	11913
05.06.1978	15.01.2011 14:54:36	11912,62125	11912,62	11912,6	11913
17.08.1982	15.01.2011 14:54:36	10378,62125	10378,62	10378,6	10379
02.03.1976	15.01.2011 14:54:36	12737,62125	12737,62	12737,6	12738
15.02.1980	15.01.2011 14:54:36	11292,62125	11292,62	11292,6	11293
03.11.1981	15.01.2011 14:54:36	10665,62125	10665,62	10665,6	10666
24.12.1982	15.01.2011 14:54:36	10249,62125	10249,62	10249,6	10250
02.03.1976	15.01.2011 14:54:36	12737,62125	12737,62	12737,6	12738
02.09.1979	15.01.2011 14:54:36	11458,62125	11458,62	11458,6	11459
18.10.1985	15.01.2011 14:54:36	9220,62125	9220,62	9220,6	9221
10.06.1979	15.01.2011 14:54:36	11542,62125	11542,62	11542,6	11543
05.06.1983	15.01.2011 14:54:36	10086,62125	10086,62	10086,6	10087
19.01.1981	15.01.2011 14:54:36	10953,62125	10953,62	10953,6	10954
29.07.1978	15.01.2011 14:54:36	11858,62125	11858,62	11858,6	11859
08.01.1989	15.01.2011 14:54:36	8042,62125	8042,62	8042,6	8043

Funkcija STRCOMP radi uspoređivanje 2 stringa:

- String1 < String2, STRCOMP vraća -1
- String1 = String2, STRCOMP vraća 0
- String1 > String2, STRCOMP vraća 1
- String1 ili String2 su NULL, STRCOMP vraća NULL

Primjer:
Želimo provjeriti da li imamo unesen redak o studentu s jednakim imenom i prezimenom

```
select * from student where strcomp(ime, prezime) = 0
```

stud_ID	ime	prezime	jmbg	datRod	pbrRod	pbrStan	adresa
10029	Ana	Ana		01.01.1988	10000	10000	

Funkcija IsNULL prima kao parametar numerički ili tekstualni izraz i vraća TRUE ukoliko je vrijednost parametra NULL, FALSE ako nije null.

```
SELECT s.ime, s.prezime, s.br_tel, IsNull(br_tel)
FROM student AS s;
```

ime	prezime	br_tel	Expr1003
Ana	Jurić	+385912222222	0
Mario	Klarić	012222222	0
Ivan	Korkut	3333333	0
Ante	Jurić	012345	0
Marija	Knjaz	012345	0
Kristina	Kušan		-1
Ivan	Županović	012345	0
Petar	Šoljić		-1
Mirta	Lončar		-1
Davor	Dujmić		-1
Franjo	Mušnjak		-1
Petra	Krleža		-1

Funkcija IIF vraća propisanu vrijednost u ovisnosti o specificiranom uvjetu (kao IF-THEN-ELSE kojeg možemo koristiti u upitu)

IIF (uvjet, vrijednost ako je uvjet ispunjen, vrijednost ako uvjet nije ispunjen)

Primjer 1:
Znamo grupirati studente po poštanskom broju:

```
SELECT pbrRod, count(*)
FROM student
GROUP BY pbrRod
```

No, sada bismo htjeli zbrojiti koliko je onih koji su rodom iz Zagreba a koliko je ostalih.

```
SELECT pbrRod, IIF(pbrRod=10000, 'ZG', 'Nije ZG'), count(*)
FROM student
GROUP BY pbrRod, IIF(pbrRod=10000, 'ZG', 'Nije ZG')
```

pbrRod	Expr1001	Expr1002
	Nije ZG	1
10000	ZG	7
21000	Nije ZG	5
22000	Nije ZG	1
23000	Nije ZG	1
31000	Nije ZG	4
32000	Nije ZG	1
35000	Nije ZG	2
44000	Nije ZG	1
48000	Nije ZG	1
51000	Nije ZG	2
53000	Nije ZG	1

Sada možemo grupirati samo temeljem IIF vrijednosti i dobiti broj studenata:

```
SELECT IIF(pbrRod=10000, 'ZG', 'Nije ZG'), count(*)
FROM student
GROUP BY IIF(pbrRod=10000, 'ZG', 'Nije ZG')
```

Expr1000	Expr1001
Nije ZG	20
ZG	7

Primjer 2: Koliko je studenata rođeno u Zagrebu ili Splitu, a koliko u ostalim mjestima?

Možemo koristiti ugnježđeni IIF
```
IIF (pbrRod = 10000, 'ZG/ST', IIF(pbrRod = 21000, 'ZG/ST', 'ostali'))
```

Upit
```
SELECT pbrRod, IIF (pbrRod = 10000, 'ZG/ST', IIF(pbrRod = 21000, 'ZG/ST', 'ostali')),
count(*)
FROM student AS s
GROUP BY pbrRod,IIF (pbrRod = 10000, 'ZG/ST', IIF(pbrRod = 21000, 'ZG/ST', 'ostali'))
```

Grupira retke u ZG/ST i ostale:

pbrRod	Expr1001	Expr1002
	ostali	1
10000	ZG/ST	7
21000	ZG/ST	5
22000	ostali	1
23000	ostali	1
31000	ostali	4
32000	ostali	1
35000	ostali	2
44000	ostali	1
48000	ostali	1
51000	ostali	2
53000	ostali	1

Traženi rezultat dobit ćemo tako da prebrojimo samo po IIF vrijednosti:

```
SELECT IIF (pbrRod = 10000, 'ZG/ST', IIF(pbrRod = 21000, 'ZG/ST', 'ostali')), count(*)
FROM student AS s
GROUP BY IIF (pbrRod = 10000, 'ZG/ST', IIF(pbrRod = 21000, 'ZG/ST', 'ostali'))
```

Expr1000	Expr1001
ostali	15
ZG/ST	12

Napomena – ugniježđeni IIF je ovdje u svrhu demonstracije. Zadatak se može se riješiti i jednim IIF s uvjetom pbrRod in (10000, 21000).

Funkcija SWITCH omogućava postavljanje vrijednost temeljem zadanih uvjeta (slično kao da imamo više ugniježdenih IF naredbi ili CASE naredbu)

SWITCH (uvjet1, vrijednost1, uvjet2, vrijednost2,..., uvjet n, vrijednost n)

Što je ekivalentno:

```
If uvjet 1 then
        Vrijednost1
Else
        If uvjet 2 then
                Vrijednost 2
        Else
                ...
...
End if
```

Primjer 1:

```
SELECT s.ime, s.prezime,
switch(ime ='Ana', 'to je naša Ana', ime = 'Mate', 'to je naš Mate', prezime = 'Tomac', 'to je naš Tomac', prezime = 'Dujmić', 'to je naš Dujmić', TRUE, 'svi ostali')
FROM student AS s;
```

ime	prezime	Expr1002
Ana	Jurić	to je naša Ana
Mario	Klarić	svi ostali
Ivan	Korkut	svi ostali
Ante	Jurić	svi ostali
Marija	Knjaz	svi ostali
Kristina	Kušan	svi ostali
Ivan	Županović	svi ostali
Petar	Šoljić	svi ostali
Mirta	Lončar	svi ostali
Davor	Dujmić	to je naš Dujmić
Franjo	Mušnjak	svi ostali
Petra	Krleža	svi ostali
Jurica	Domanovac	svi ostali
Hrvoje	Petrović	svi ostali
Igor	Ivčić	svi ostali
Ana	Antolić	to je naša Ana
Mislav	Bistričić	svi ostali
Ljubica	Šaput	svi ostali
Boris	Kundera	svi ostali
Jelena	Župan	svi ostali
Ivan	Marić	svi ostali
Zvonimir	Deranja	svi ostali
Marta	Kurjak	svi ostali
Ana	Tomac	to je naša Ana
Anita	Ćupić	svi ostali
dsas	dsfsfd	svi ostali
Ana	Ana	to je naša Ana

Primjetite u primjeru studenta Ana Tomac: obzirom da je prvi uvjet ime='Ana' zadovoljen, rezultat je postavljen u 'to je naša Ana'. Za taj redak zadovoljeno je i prezime = Tomac' (no obzirom da je ranije uvjet zadovoljen uvjet na prezime se niti ne ispituje) a zadnji uvjet TRUE je zadovoljen za sve retke.
Ono što napišemo nakon TRUE ispisat će se kao rezultat za sve one zapise za koje niti jedan od ranijih uvjeta nije zadovoljen.

Primjer 2:
Izbrojati koliko je studenata iz Zagreba ili Splita, a koliko je ostalih studenata.

Ovako bismo mogli postavili vrijednost po kojoj možemo grupirati studente:
switch (pbr in (10000, 21000), 'Zagreb ili Split', TRUE, 'ostali')

Upit

SELECT stud_id, ime, prezime, pbrRod, switch (pbrRod in (10000, 21000), 'Zagreb ili Split', TRUE, 'ostali')
FROM student AS s;

Vraća zapise o studentima s oznakom grupe kojoj pripadaju (ZG/ST grupa ili ostali):

stud_id	ime	prezime	pbrRod	Expr1004
10001	Ana	Jurić	10000	Zagreb ili Split
10002	Mario	Klarić	31000	ostali
10003	Ivan	Korkut	21000	Zagreb ili Split
10004	Ante	Jurić	48000	ostali
10005	Marija	Knjaz	51000	ostali
10006	Kristina	Kušan	53000	ostali
10007	Ivan	Županović	31000	ostali
10008	Petar	Šoljić	22000	ostali
10009	Mirta	Lončar	51000	ostali
10010	Davor	Dujmić	10000	Zagreb ili Split
10011	Franjo	Mušnjak	35000	ostali
10012	Petra	Krleža	10000	Zagreb ili Split
10013	Jurica	Domanovac	10000	Zagreb ili Split
10014	Hrvoje	Petrović	10000	Zagreb ili Split
10015	Igor	Ivčić	21000	Zagreb ili Split
10016	Ana	Antolić	32000	ostali
10017	Mislav	Bistričić	21000	Zagreb ili Split
10018	Ljubica	Šaput	44000	ostali
10019	Boris	Kundera	23000	ostali
10020	Jelena	Župan	21000	Zagreb ili Split
10021	Ivan	Marić	35000	ostali
10022	Zvonimir	Deranja	21000	Zagreb ili Split
10023	Marta	Kurjak	31000	ostali
10024	Ana	Tomac	31000	ostali
10025	Anita	Ćupić	10000	Zagreb ili Split
10026	dsas	dsfsfd		ostali
10029	Ana	Ana	10000	Zagreb ili Split
0			0	

Ukoliko izbrojimo studente grupirano po tom izrazu, dobit ćemo ono što tražimo:

```
SELECT switch (pbrRod in (10000, 21000), 'Zagreb ili Split', TRUE, 'ostali'), count(*)
FROM student AS s
GROUP BY switch (pbrRod in (10000, 21000), 'Zagreb ili Split', TRUE, 'ostali');
```

Expr1000	Expr1001
ostali	15
Zagreb ili Split	12