

Osnovni pojmovi i definicije

- Entitet je apstraktni pojednostavljeni model skupa subjekata, objekata ili događaja iz stvarnog svijeta koji imaju naglašena zajednička svojstva (atribute) po kojima ih pratimo.
 - Entitet se predstavlja relacijskom tablicom
 - Entitet (tablica) – skup stupaca i redaka. Predstavlja objekte iz realnog svijeta koji imaju naglašena zajednička svojstva.
- Element entiteta je jedan konkretni primjerak apstraktnog objekta kojeg pratimo (jedna instanca entiteta)
 - Redak (n-torka, instanca entiteta) – skup informacija o pojedinom članu.
 - Redovi tablice odgovaraju elementima entiteta
- Atribut (stupac) – svojstvo (karakteristika) entiteta.
 - Stupci tablice odgovaraju pojedinim atributima.

Atributi

- 2 osnovna tipa atributa:
 - Opisni atributi – opisuju neko svojstvo entiteta
 - Identifikatori – jedinstveno određuju konkretni element entiteta
- složeni atribut = grupa atributa koji predstavljaju logički povezana svojstva entiteta (npr adresa = ulica, broj)
- **Primarni ključ** PK je atribut ili skup atributa koji jedinstveno identificiraju svaki element entiteta (redak u tablici). Mora biti card=(1,1)
 - **Jedinstvenost** - U tablici ne mogu postojati dva retka s istom vrijednošću primarnog ključa
 - **Minimalnost** - Ako je primarni ključ složen tj. sastoji se od više atributa, tada se niti jedna njegova komponenta ne može ukloniti a da se ne naruši pravilo jedinstvenosti.
 - **NOT NULL (Obavezni podatak)** - Niti jedna komponenta primarnog ključa ne smije imati NULL vrijednost

Kardinalitet atributa, kardinalitet entiteta u relaciji

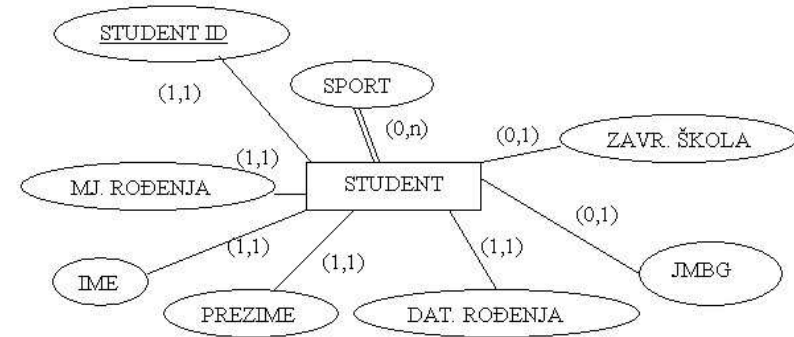
- Kardinalitet atributa govori koliko vrijednosti pojedini atribut daje za opis jednog elementa entiteta.
- Pri utvrđivanju kardinaliteta određuje se donja i gornja granica (minimalni i maksimalni kardinalitet atributa A u entitetu E):
 - Atributi sa minimalnim kardinalitetom jednakim 0 nisu obvezni.
 - Viševrijednosni atributi su atributi (svojstva), koja mogu imati više vrijednosti za pojedini element entiteta. $\text{Card}(A, E) = (0, n)$
- Kod složenih atributa (grupa logički povezanih svojstava entiteta), za donji/gornji kardinalitet uzima se max donjih/gornjih kardinaliteta pojedinih atributa.
- Kardinalitet entiteta u relaciji - definiran je brojem veza pojedinog elementa tog entiteta s elementima entiteta s kojim je relacijski povezan

Kardinalitet atributa, kardinalitet entiteta u relaciji

- Min kardinalitet atributa 0 znači da podatak nije obavezan.
- Min kardinalitet atributa 1 znači da je podatak obavezan (NOT NULL, odnosno REQUIRED).
- Min kardinalitet entiteta u relaciji 0 znači da ne sudjeluju svi elementi entiteta u vezi.
- Min kardinalitet entiteta u relaciji 1 znači da svaki elementi entiteta u vezi sudjeluje u relaciji (odnosno da za svaki redak u tablici entiteta postoji zapis).

- NULL vrijednost – označava nepoznat ili nepostojeći podatak
 - neovisno je o tipu podatka
 - različito od svih ostalih vrijednosti
 - `NULL <> ""` (prazni string)
 - `NULL <> 0`
 - `NULL <> "NULL"`
 - `NOT NULL = REQUIRED`
 - Constraint "NOT NULL" kod kreiranja tablice označava obavezne attribute

ER dijagram

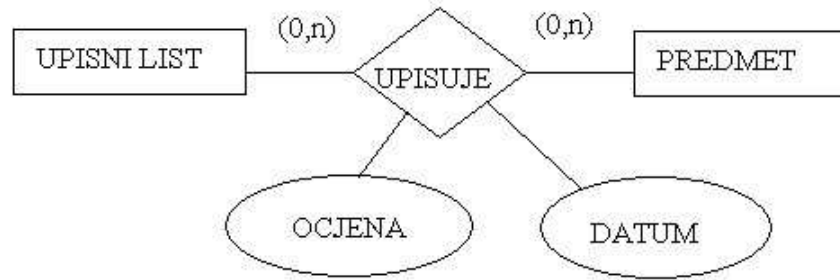


Slika 2. Dijagram entiteta

Dijagram entiteta = grafički prikaz entiteta i njegovih atributa

- Entitet - pravokutni znak u koji se upisuje naziv entiteta - **TABLICA**
- Atribut - ovalni znak u koji se upisuje naziv atributa, povezan s entitetom – **STUPAC U TABLICI**
 - Naziv atributa koji predstavlja primarni ključ se podcrtava.
 - Kardinalitet atributa upisuje na njegovu poveznicu sa entitetom.
 - Viševrijednosni atributi imaju dvostruku poveznicu sa entitetom, koja podrazumijeva kardinalitet (0,n). Prikazuje se posebnom tablicom koja se sastoji od PK entiteta i viševrijednosnog atributa.

ER dijagram



Slika 7. Dijagram relacije više-na- više ($n \times n$).

ER Dijagram (Entity-Relationship) = proširenje grafičkog prikaza entiteta na prikaz međusobnih veza između entiteta

- Veza ili Relacija – romb, odnosno deltoidni znak \diamond u koji se upisuje naziv veze, povezan s entitetima koji sudjeluju u vezi.
 - Atributi veze - Ukoliko postoje, atributi koji su posljedica relacije ucrtavaju se ovalnim simbolom kao i atributi entiteta, te se povezuju sa relacijom iz koje proizlaze. Čest slučaj kod relacija “više prema više”.
 - Kardinalitet veze upisuje se na poveznicu sa entitetom.

Vrste veza

Tipovi veza ovise o kardinalitetu entiteta u vezi

- 1 x 1 / relacija 1 prema 1 / one-to-one
 - Realizira se na 1 od slijedećih načina:
 - U jednu od tablica dodaje se FK na drugu tablicu
 - Tablice se spajaju
- 1 x n / relacija 1 na više / one-to-many
 - "glava" (1) i "stavke" (n)
 - realizira se pomoću stranog ključa u tablici "stavke" (dodaje se ID "glave", primarni ključ iz tablice "glava")
- n x n / relacija više prema više / many-to-many
 - prikazuje se novom tablicom koja osim stranih ključeva koji pokazuju na element entiteta koji su u relaciji može imati i dodatne attribute veze.
 - Najčešće je PK ovakve tablice veze složen od stranih ključeva entiteta koji sudjeluju u relaciji

SQL = Structured Query Language



DDL = Data Definition Language

Definiranje same baze:

- Kreiranje objekata u bazi
- Izmjene objekata u bazi
- Uništavanje postojećih objekata
- Objekti – tablica, atribut, ograničenje, ...

CREATE / ALTER / DROP

DML = Data Manipulation Language

Manipulacija podacima:

- upiti
- dodavanje podataka u bazu
- izmjena podataka
- brisanje podataka

SELECT / INSERT / UPDATE / DELETE

SQL = Structured Query Language

DDL = Data Definition Language

Kreiranje/Izmjena definicije objekta/Uništavanja objekata

Objekti u bazi – tablica, indeks, dodatni atribut u tablici, constraint,...

Naredbe CREATE / ALTER / DROP

Primjeri DDL naredbi (MS Access):

```
create table mjesto  
( pbr number CONSTRAINT pk_pbr PRIMARY KEY,  
  naziv text CONSTRAINT nn_naziv NOT NULL );
```

```
create table vezaFiG  
(f_id integer CONSTRAINT fk_flm REFERENCES f(f_ID),  
 g_id integer CONSTRAINT fk_glm REFERENCES g(g_ID));
```

```
select * into TABLICA_KOPIJA from TABLICA_ORIGINAL;
```

SQL = Structured Query Language

Primjeri DDL naredbi (MS Access):

- alter table student add column godine int;
- alter table student alter column jmbg varchar(13);
- alter table student drop column datRod;

- alter table polaznik add constraint UQ_JMBG UNIQUE (jmbg);
- alter table student DROP CONSTRAINT FK_ROD ;

- drop table mjesto;

Primjer – kreiranje baze narudžbi. Entiteti: kupac, proizvod, narudžba

- Kupci:

create table kupac

```
(k_id          counter (100, 3)  primary key,  
 ime          varchar(100)      not null,  
 prezime      varchar(100)      not null);
```

--AutoNum/sekvenca za PK

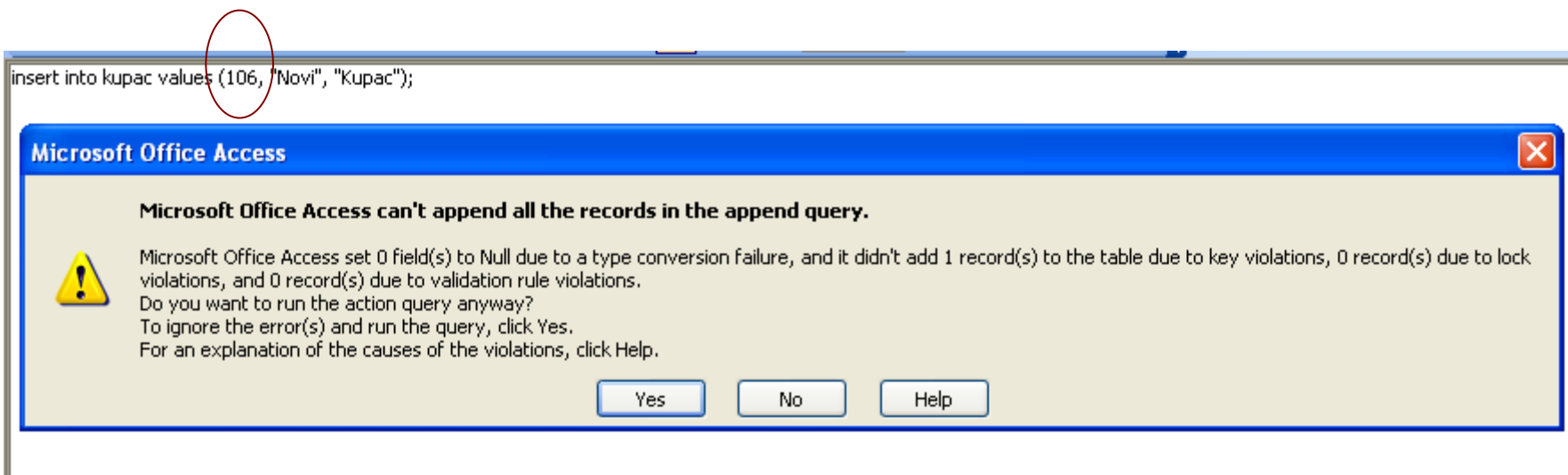
	k_id	ime	prezime
+	100	Pero	Perić
+	103	Ana	Anić
+	106	Mate	Matić
►	(AutoNumber)		

- Proizvodi:

create table proizvod

```
(p_ID          COUNTER (1, 5) PRIMARY KEY,  
 grupa_pr     varchar(20) ,  
 naziv        varchar(50) NOT NULL,  
 CONSTRAINT UQ_GRP_NZ UNIQUE (grupa_pr, naziv));
```

Constraint violations PK:



Zbog toga što u tablici već imamo unesen podatak za primarni ključ $k_ID = 106$, baza nam neće dopustiti upisivanje novog zapisa s istom vrijednosti primarnog ključa

insert into kupac values (107, "Novi", "Kupac");

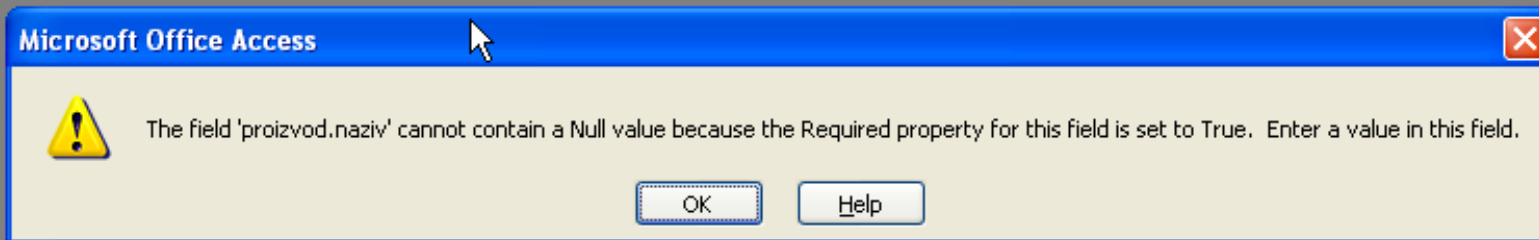
--> ubacivanje novog zapisa uredno prolazi

	k_id	ime	prezime
+	100	Pero	Perić
+	103	Ana	Anić
+	106	Mate	Matić
+	107	Novi	Kupac
►	(AutoNumber)		

Constraint violations NOT NULL, UNIQUE:

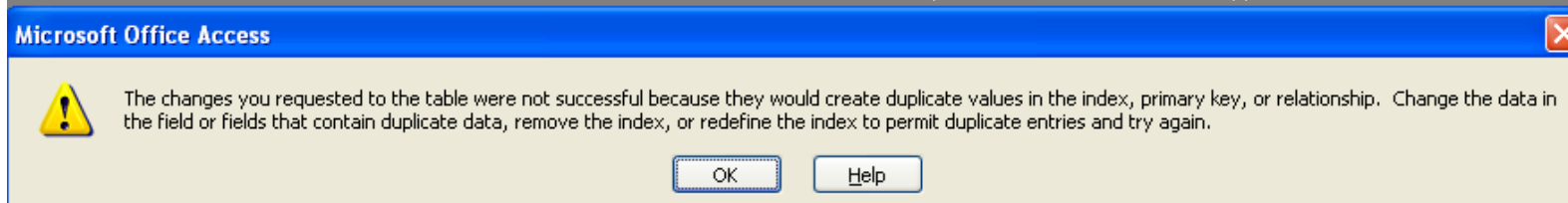
	p_ID	grupa_pr	naziv
+	1	MLJECNI	ABC sir
+	6	MLJECNI	Trajno mlijeko
+	11	MLJECNI	Jogurt
+	16	SUHOMESNATI	Šunka
+	21	SUHOMESNATI	Zimska
+	26	SUHOMESNATI	
*	(AutoNumber)		

Nije dopušten unos NULL vrijednosti u naziv. Regulirano NOT NULL ograničenjem kod kreiranja tablice.



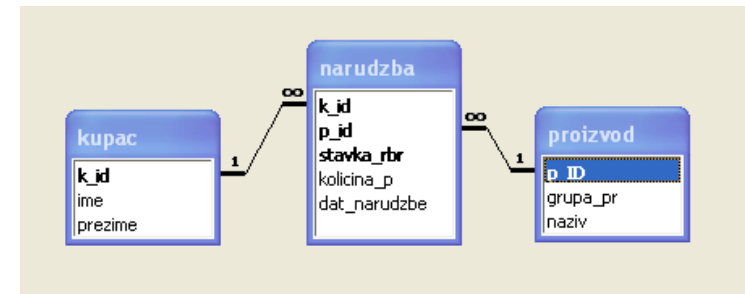
	p_ID	grupa_pr	naziv
+	1	MLJECNI	ABC sir
+	6	MLJECNI	Trajno mlijeko
+	11	MLJECNI	Jogurt
+	16	SUHOMESNATI	Šunka
+	21	SUHOMESNATI	Zimska
+	31	SUHOMESNATI	Zimska
*	(AutoNumber)		

Nije dopušten unos novog zapisa koji ima jednake attribute grupa_pr i naziv kao neki postojeći zapis jer smo kod kreiranja definirali constraint naziva uq_grp_nz koji je UNIQUE (grupa_pr, naziv);



Primjer – kreiranje baze narudžbi. Entiteti: kupac, proizvod, narudžba

- Narudžbe kao veza kupca i proizvoda
 - FK reference na kupca i proizvod
 - Složeni PK (kupac, proizvod i rbr stavke nar.)
 - Relacije generirane SQL naredbom




create table narudzba

```
( k_id          integer CONSTRAINT FKKUP REFERENCES kupac(k_ID),
  p_id          integer CONSTRAINT FKPROIZV REFERENCES proizvod(p_ID),
  stavka_rbr    integer,
  kolicina_p     integer not null,
  dat_narudzbe  datetime NOT NULL,
  CONSTRAINT pk_nar PRIMARY KEY (p_id, k_id, stavka_rbr)
);
```

Kreiranjem FOREIGN KEY CONSTRAINTA naziva fkkup i fkproizv smo SQL-om "povukli veze" između tablica kupac i narudzba i tablica proizvod i narudzba. Kad pogledamo vizualni prikaz u MS Accessu u "Relationships", on odmah po kreiranju tab. narudzba izgleda kao na gornjoj slici.

FK constraint violations:

	k_id	p_id	stavka_rbr	kolicina_p	dat_narudzbe
✎	1	1	1	2	02.11.2010
*			(AutoNumber)		




The dialog box has a blue title bar with the text "Microsoft Office Access" and a close button (X). The main area has a yellow background with a warning icon (exclamation mark in a triangle) on the left. The text reads: "You cannot add or change a record because a related record is required in table 'kupac'." At the bottom, there are two buttons: "OK" and "Help".

U tablici "kupac" nemamo kupca s vrijednosti primarnog ključa 1.

Zbog postojanja stranog ključa u tablici narudzbi s referencom na kupce, ne možemo upisati narudžbu za nepostojećeg kupca

	k_id	p_id	stavka_rbr	kolicina_p	dat_narudzbe
✎	100	3	1	2	02.11.2010
*			(AutoNumber)		



The dialog box has a blue title bar with the text "Microsoft Office Access" and a close button (X). The main area has a yellow background with a warning icon (exclamation mark in a triangle) on the left. The text reads: "You cannot add or change a record because a related record is required in table 'proizvod'." At the bottom, there are two buttons: "OK" and "Help".

Zbog postojanja stranog ključa u tablici narudzbi s referencom na proizvode, ne možemo upisati narudžbu za proizvod koji ne postoji u tablici proizvoda

SQL = Structured Query Language

DML = Data Manipulation Language

Manipulacija podacima:

upiti

dodavanje podataka u bazu

izmjena podataka

brisanje podataka

Primjeri DML upita:

- `select * from student where prezime = "Jurić" ;`
- `insert into student (ime, prezime,...) values ("Jure", "Jurić", ...);`
- `insert into dobavljac`
`select * from kupac where ime in ("Pero", "Mate");`
- `update student set pbr = 10000 where stud_id = 2543;`
- `update student`
`set godiste = year(dat_rod);`
- `delete * from student where pbr = 21000;`
- `delete from student`
`where stud_id not in (select stud_id from ispit where ocjena > 1);`

Upiti: SELECT ... FROM ... join... WHERE...

SELECT (što? koje attribute – lista atributa – atributi odvojeni **zarezom**)

FROM (od kud? iz kojih tablica – lista entiteta – ako nisu dio joina, onda entiteti trebaju biti odvojeni **zarezom**)

JOIN (kako povezujemo te tablice) - **opciono**

WHERE/AND/OR (kriteriji da bi se ispisalo retke: uvj.1 and/or uvj.2 and/or uvj.3..) - **opciono**

Ovako možemo zamisliti:

- odradi se JOIN: možemo zamisliti da nakon toga imamo samo 1 „tablicu“
- odradi se WHERE: selekcija samo onih redaka koji zadovoljavaju postavljene uvjete
- odradi se SELECT samo nekih atributa dobivene „join tablice“
- * umjesto popisa atributa označava da se uzimaju svi atributi
- Minimalno upit mora imati SELECT i FROM dio.

SELECT ...

- ATRIBUTI nabrojani u selectu MORAJU PRIPADATI ENTITETIMA iz "FROM" dijela!
 - Ne možemo dohvatiti podatak o imenu nastavnika ako ne koristimo tablicu nastavnik.
- Osim samih atributa iz entiteta koje koristimo u "FROM" dijelu, možemo u selectu koristiti ugrađene funkcije nad tim atributima ili kreirati vlastite izraze.

- Primjer 1:

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

- Tablica zaposlenika:

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

SELECT ...

➤ Primjer 2:

➤ 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

Select iz jedne tablice

- Entitet E sastoji se od atributa A_1, \dots, A_n : $E = (A_1, \dots, A_n)$. Primjerice Student = (stud_ID, ime, prezime, jmbg, datRod, pbrRod), što odgovara tablici student i njezinim stupcima.
- Upiti koji vraćaju sve attribute entiteta student:
 - Select * from student;
 - Select stud_ID, ime, prezime, jmbg, datRod, pbrRod from student;
 - Select student.* from student;
 - Select s.* from student s; (dodijelili smo alias naziv entitetu)
- Kad nema napisanog WHERE uvjeta, ispisuje se svaki redak iz tablice student
- **Select A1, A3 from E** => vraća samo attribute A1 i A3 iz n-torke atributa (A_1, \dots, A_n)
- **Select A1, A3 from E where UVJET(A2)** => vraća samo attribute A1 i A3 iz n-torke, ali samo za one zapise za koje atribut A2 zadovoljavaju postavljeni uvjet.
 - Uvjet može biti jednostavan ili složen, te može koristiti attribute koje nećemo ispisati.
- **select A1 A2 from E** (bez zareza između atributa)?
 - u Accessu javlja grešku; u nekim drugim bazama izlistalo bi atribut A1 s nazivom atributa A2 umjesto naziva A1

Select iz jedne tablice - primjeri

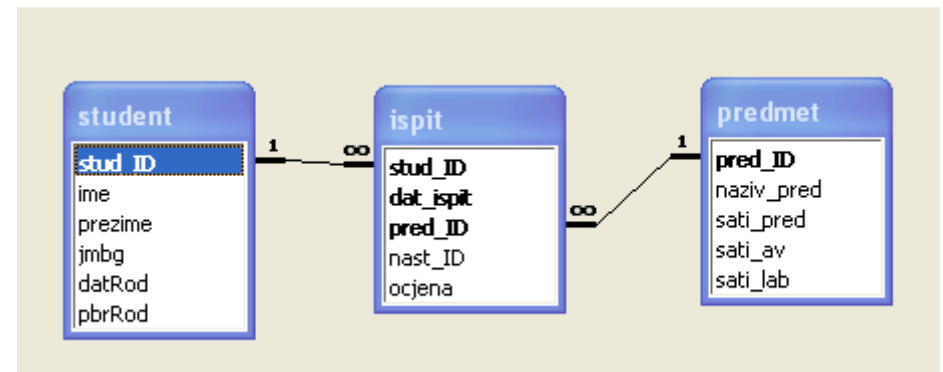
➤ **Select A1, A3 from E** (projekcija)

➤ **Select ime, prezime, jmbg from student**

➤ Za sve studente (odn. sve retke upisane u tablicu "student"), upit vraća samo označene atribute

➤ **Select pred_id, naziv_pred from predmet**

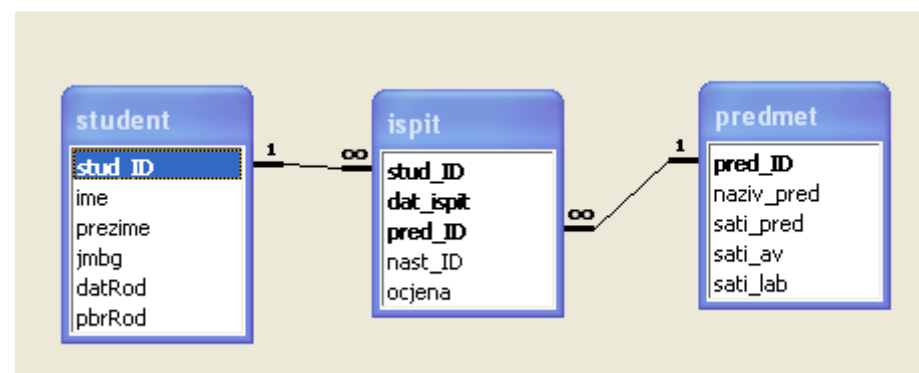
➤ Za sve predmete (odn. sve retke upisane u tablicu "predmet"), upit vraća samo označene atribute



	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

	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

Select iz jedne tablice - primjeri



- **Select * from E**

where UVJET (A1, A2,...) (selekcija)

- **Select * from student**

where prezime = "Jurić";

	stud_ID	ime	prezime	jmbg	datRod	pbrRod
+	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

- **Select A1, A2, A3 from E where UVJET(A2) (selekcija i projekcija)**

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

	ime	prezime	jmbg
	Ana	Jurić	1106980401236
	Ante	Jurić	0412979214665

Select iz jedne tablice - primjeri

➤ **Select A1, A3 from E**

where UVJET(A2) (selekcija i projekcija)

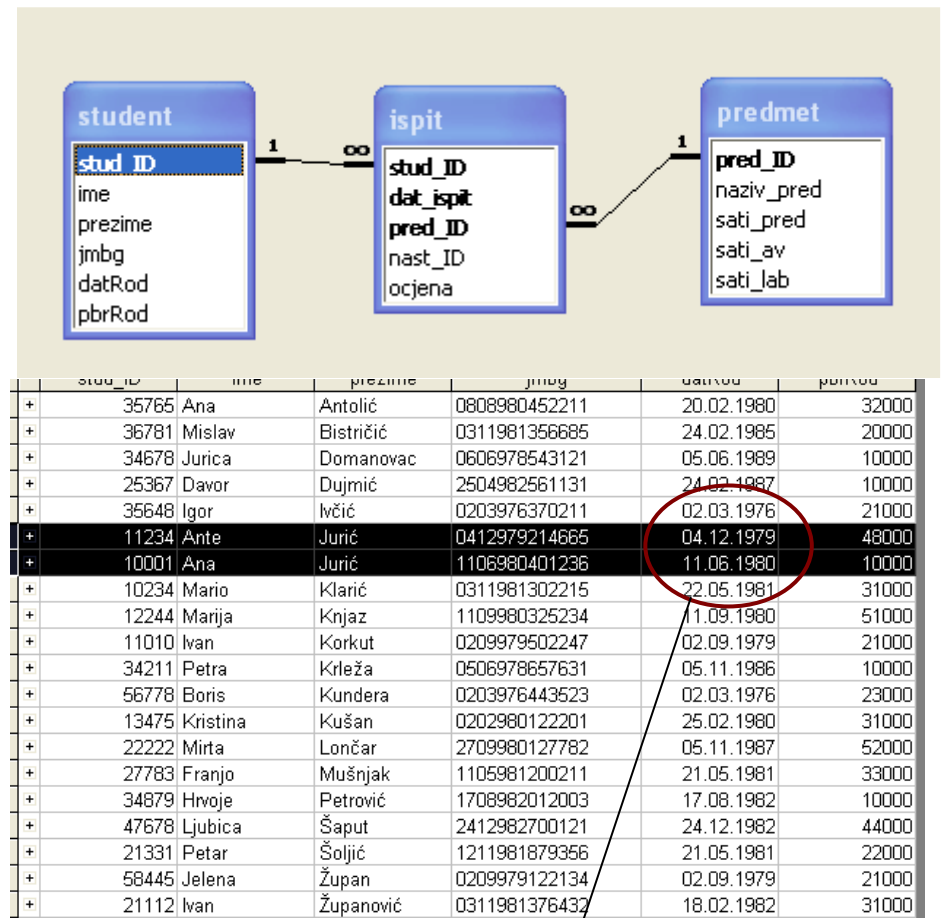
➤ **Select ime, prezime, jmbg**

from student

where prezime = "Jurić"

and year(datRod) = 1980;

	ime	prezime	jmbg
▶	Ana	Jurić	1106980401236



Year (04.12.1979) = 1979 (False)

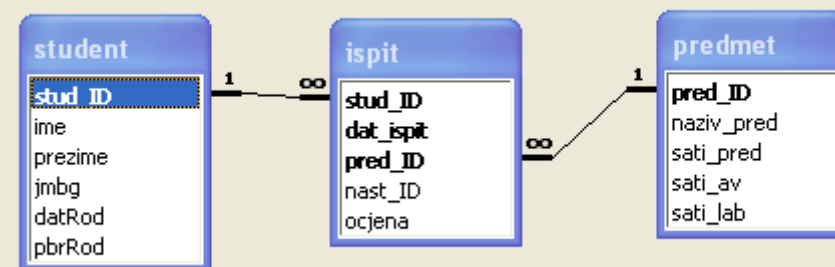
Year (11.06.1980) = 1980 (True)

=> Uvjet je zadovoljen samo za
Anu Jurić, Ante Jurić otpada

Vježbe – upiti nad 1 tablicom.

Predviđeno trajanje 10-ak minuta.

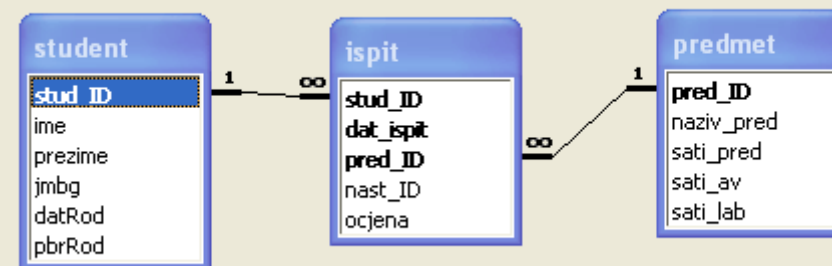
Može u MS Accessu ili na papiru.



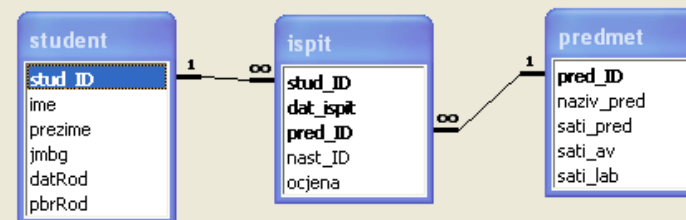
- Ispisati sve studente (sve podatke o njima)
- Ispisati sve nazive predmeta
- Ispisati naziv predmeta, te broj predviđenih sati predavanja za sve predmete
- Ispisati naziv predmeta, te broj predviđenih sati predavanja za predmet baze podataka
- Ispisati naziv predmeta, te broj predviđenih sati predavanja za predmete: baze podataka, uvod u programiranje, matematika.
- Ispisati sve studente koji su rođeni 01.12.1985.
- Ispisati ime i prezime studenata koji imaju pbrRod = 10000
- Ispisati stud_ID, ime i prezime studenata koji su rođeni u srpnju
- Ispisati jmbg, ime i prezime studenata koji su rođeni 1985. godine

Vježbe – upiti nad 1 tablicom.

- Ispisati sve studente (sve podatke o njima)
 - `Select * from student`
- Ispisati sve nazive predmeta
 - `Select naziv_pred from predmet`
- Ispisati naziv predmeta, te broj predviđenih sati predavanja za sve predmete
 - `Select naziv_pred, sati_pred from predmet`
- Ispisati naziv predmeta, te broj predviđenih sati predavanja za predmet baze podataka
 - `Select naziv_pred, sati_pred from predmet where naziv_pred = "baze podataka"`



Vježbe – upiti nad 1 tablicom.



- Ispisati naziv predmeta, te broj predviđenih sati predavanja za predmete: baze podataka, uvod u programiranje, matematika.
 - `Select naziv_pred, sati_pred from predmet where naziv IN ("baze podataka", "uvod u programiranje", "matematika")`
- Ispisati sve studente koji su rođeni 01.12.1985.
 - `Select * from student where datRod = "01.12.1985"`
- Ispisati ime i prezime studenata koji imaju pbrRod = 10000
 - `Select ime, prezime from student where pbrRod = 10000`
- Ispisati stud_ID, ime i prezime studenata koji su rođeni u srpnju
 - `Select stud_id, ime, prezime from student where month(datRod) = 7`
- Ispisati jmbg, ime i prezime studenata koji su rođeni 1985. godine
 - `Select jmbg, ime, prezime from student where year(datRod) = 1985`

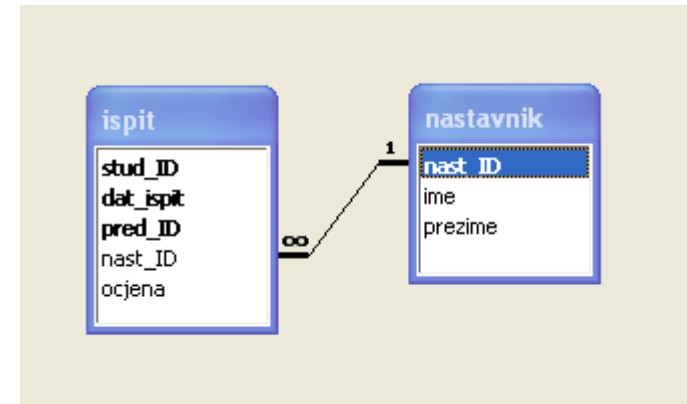
Select iz dvije tablice

- Imamo 2 entiteta, primjerice Student = (stud_ID, ime, prezime, jmbg, datRod, pbrRod), što odgovara tablici student i njezinim stupcima i Ispit = (stud_ID, dat_ispit, pred_ID, nast_ID, ocjena), što odgovara tablici ispita.
- Te dvije tablice su prirodno povezane student.stud_id = ispit.stud_id (svaki zapis iz tablice ispit odnosi se na nekog konkretnog studenta)
 - ...from student INNER JOIN ispit on student.stud_ID = ispit.stud_ID
 - ...from student, ispit where student.stud_ID = ispit.stud_ID
- Neki od upita koji vraćaju sve attribute:
 - Select * from student inner join ispit on...;
 - Select **student.stud_ID**, ime, prezime, jmbg, datRod, pbrRod, **ispit.stud_ID**, dat_ispit, pred_ID, nast_ID, ocjena
from student inner join ispit on...;
 - Select student.stud_ID, student.ime, student.prezime, student.jmbg, student.datRod, student.pbrRod, ispit.stud_ID, ispit.dat_ispit, ispit.pred_ID, ispit.nast_ID, ispit.ocjena
from student inner join ispit on...;
 - Select student.*, ispit.* from student inner join on...;
 - Select s.*, i.* FROM student s inner join ispit i on...;

Select iz dvije tablice

- **Select prezime, jmbg from ispit;** → ne radi, jer su prezime i jmbg atributi entiteta student a ne ispit
- **Select ime, prezime from student inner join ispit on student.stud_id = ispit.stud_id**
- Iako su svi nabrojani atributi iz tablice student, ovo je razlicito od „select ime, prezime from student” jer se u gornjem primjeru selekcija ne radi nad tablicom student vec nad joinom tablica student i ispit.
 - svi nastavnici <> svi nastavnici koji su održali ispit
 - svi studenti <> svi studenti koji su polagali ispit
- Slučajno rezultat oba upita može biti jednak (primjerice kad bi u podacima imali situaciju da je svaki student polagao točno po jedan ispit)
- Da bismo ispisali sve studente/nastavnike dovoljno je selectirati tablicu student/nastavnik
- Da bismo ispisali samo one studente/nastavnike koji su sudjelovali u tablici veze „ispit”, potrebno je napraviti povezivanje (join) s tablicom ispit, iako se traže samo podaci iz tablice student/nastavnik.

Primjer 1



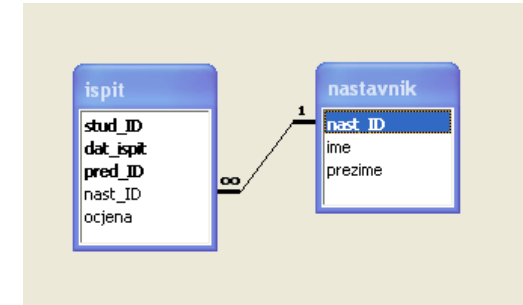
Svi nastavnici:

`select ime, prezime from nastavnik;`

Svi nastavnici koji su održali ispit:

1. `select ime, prezime from nastavnik, ispit where nastavnik.nast_id = ispit.nast_id;`
 - dobivamo nastavnike više puta (po jednom za svaki ispit kojeg su održali). Ne vraća nastavnike koji nisu održali niti jedan ispit.
2. `select ime, prezime from nastavnik INNER JOIN ispit on nastavnik.nast_ID = ispit.nast_ID`
 - → dobivamo nastavnike više puta (po jednom za svaki ispit kojeg su održali). Ne vraća nastavnike koji nisu održali niti jedan ispit.
3. `select ime, prezime from nastavnik LEFT JOIN ispit on nastavnik.nast_ID = ispit.nast_ID`
 - → dobivamo nastavnike više puta (po jednom za svaki ispit kojeg su održali), ali i one nastavnike koji nisu održali niti jedan ispit.

Primjer 1



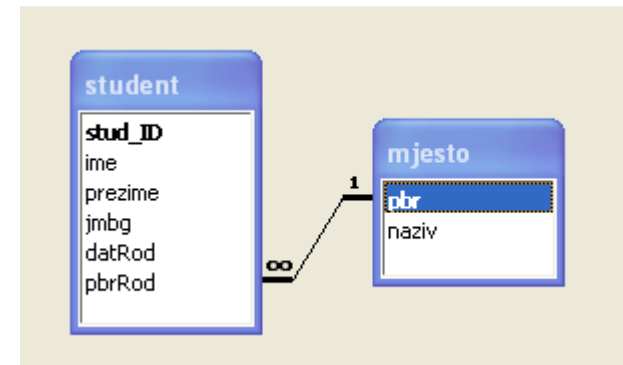
4. `select ime, prezime from nastavnik where nast_id in (select nast_id from ispit)`

- dobivamo samo **po jednom** ispisane one nastavnike koji su držali ispit
- nad jako velikim tablicama bolje izbjegavati jer se `select` u `where` uvjetu odvrti za svaki redak pa često ovakav upit nije optimalan. Za male tablice je OK.

5. `select ime, prezime from nastavnik RIGHT JOIN ispit on nastavnik.nast_ID = ispit.nast_ID`

- dobivamo nastavnike više puta (po jednom za svaki ispit)
- obzirom da ispit ne može biti evidentiran za nastavnika koji nije unesen u tablicu nastavnik (zbog postojećih FK-ova) te ukoliko je atribut `nast_id` u tablici `ispit` 'required', rezultat će biti ekvivalentan onom kojeg vraća `INNER JOIN`
- da nije uspostavljen referencijalni integritet ovih tablica, moglo bi se desiti da je u tablici ispita „nepostojeci“ nastavnik (`nast_id` kojeg nema u tablici nastavnika). U takvom slučaju bi se za takvog nastavnika dobilo (NULL, NULL) kao njegovo ime i prezime za svaki takav redak u tablici ispit
- ukoliko je u tablicu ispit moguće upisati redak bez podatka o nastavniku (`nast_id IS NULL`), za svaki takav redak ćemo kao rezultat select naredbe dobiti (NULL, NULL)

Primjer 2



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.

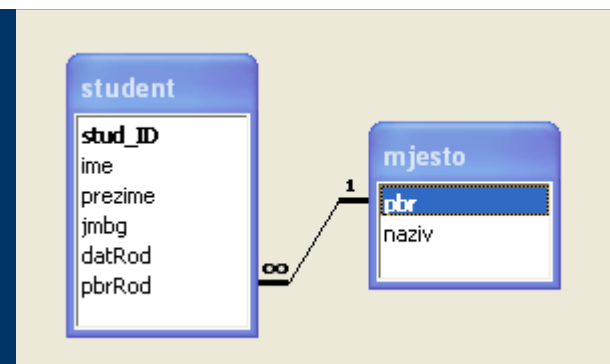
Inner join tablica student i mjesto neće vratiti zapise o studentima za koje nema podatka o pbrRod u tablici student.

Ukoliko želimo ispisati 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;
```

Vježba 2 ~ 15 min



Pretpostavimo da je u tablici student pbrRod obavezan podatak (NOT NULL)

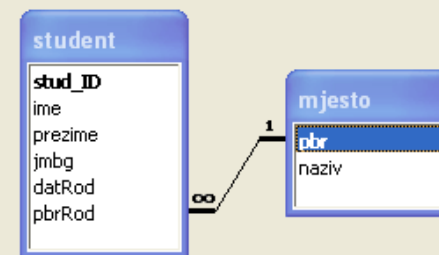
- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente
- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente koji su iz Bjelovara ili iz Trogira

Pretpostavimo da u tablici student pbrRod nije obavezan podatak (smije biti NULL)

- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente
- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente koji su iz Bjelovara ili iz Trogira

Vježba 2

Pretpostavimo da je u tablici student pbrRod obavezan podatak (NOT NULL)



- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente:

```
select ime, prezime, naziv
from mjesto INNER JOIN student on mjesto.pbr = student.pbrRod
```

Ili

```
select ime, prezime, naziv
from student INNER JOIN mjesto on student.pbrRod = mjesto.pbr
```

Ili

```
select ime, prezime, naziv
from student, mjesto
where mjesto.pbr = student.pbrRod
```

- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente koji su iz Bjelovara ili iz Trogira

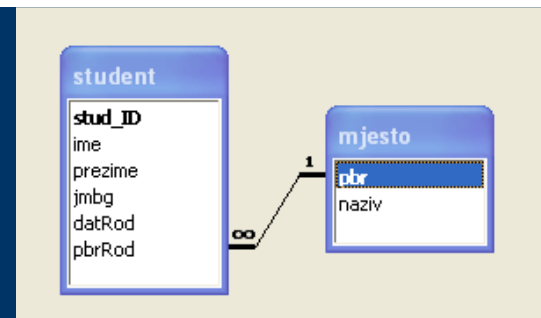
```
select ime, prezime, naziv
from mjesto INNER JOIN student on mjesto.pbr = student.pbrRod
where naziv in ("Bjelovar", "Trogir")
```

Ili (...student inner join mjesto) ili

```
select ime, prezime, naziv
from student, mjesto
where mjesto.pbr = student.pbrRod
and mjesto.naziv in ("Bjelovar", "Trogir")
```

Vježba 2

Pretpostavimo da je u tablici student pbrRod nije obavezan podatak



- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente:

```
select ime, prezime, naziv
from mjesto LEFT JOIN student on mjesto.pbr = student.pbrRod
```

- Ispisati podatke o imenu, prezimenu i nazivu mjesta rođenja za sve studente koji su iz Bjelovara ili iz Trogira

također inner join (jer NULL zapisi nam ovdje nisu od interesa – NULL nije ni Bjelovar ni Trogir)

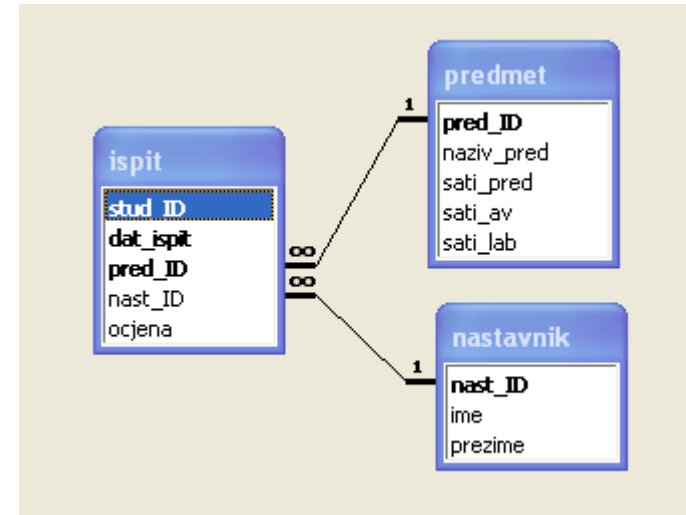
Join nekoliko tablica

...FROM (join1) JOIN tablica 3 on...

...FROM tablica 1 JOIN (join 2) on ...

...FROM t1, t2, t3

WHERE t2.fk_t1 = t1.pk and t2.fk_t3 = t3.pk
(t2 je u veza)



- **Primjer 3**
- Želimo selectirati attribute iz 2 entitea koja nisu direktno povezana vec 'između njih' postoji tablica veze => moramo napraviti join sve 3 tablice da bismo ispravno povezali podatke..
- Zadatak: Ispisati sve podatke nastavnika koji su držali ispit zajedno s nazivima predmeta koje su ispitivali:

Primjer 3 - Ispisati sve podatke nastavnika koji su držali ispit zajedno s nazivima predmeta koje su ispitivali...

- Iz definicije zadatka vidi se popis atributa koje treba ispisati:

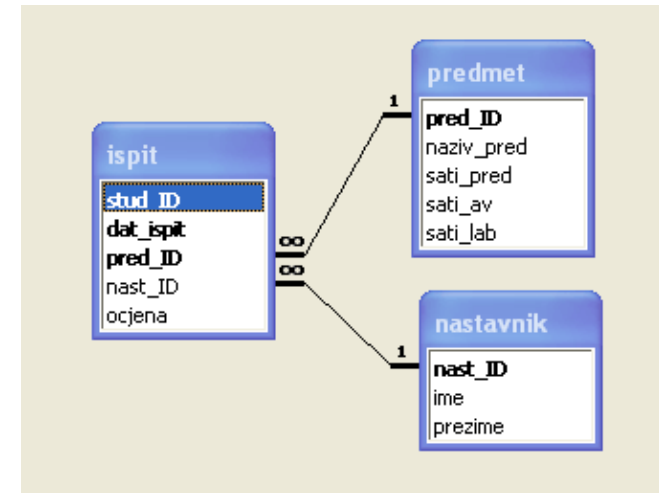
select nastavnik.*, predmet.naziv_Pred (A)

- 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)



Primjer 3 - Ispisati sve podatke nastavnika koji su držali ispit zajedno s nazivima predmeta koje su ispitivali...

```
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
```

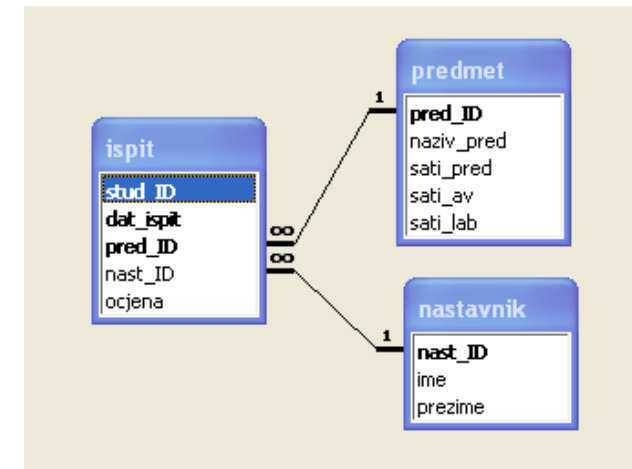
```
select n.*, p.naziv_pred  
from predmet p inner join (nastavnik n inner join ispit i on n.nast_id =  
i.nast_id) on p.pred_id = i.pred_id
```

```
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
```

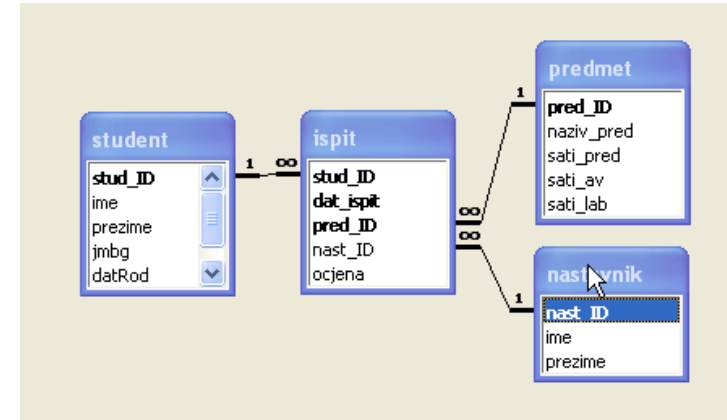
```
select n.*, p.naziv_pred  
from nastavnik n inner join (ispit i inner join predmet p on p.pred_id =  
i.pred_id) on n.nast_id = i.nast_id
```

itd... (varijacije redoslijeda)

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



Primjer 3 - Ispisati sve podatke nastavnika koji su držali ispit zajedno s nazivima predmeta koje su ispitivali...



Ukoliko je n studenata izišlo na neki ispit (isti pred_id, nast_id), dobit ćemo pomoću ovih upita iz tablice ispit za tu vezu (pred_id, nast_id) n zapisa. Obzirom da nas ne zanima podatak koliko je studenata izišlo na te ispite niti veza prema tablici student, već samo (predmet-(ispit)-profesor), možemo upotrebiti **DISTINCT** da bismo ispisali samo one zapise koji su nam korisni (odnosno da ne ispisujemo 'duplice'):

```
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
```

```
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
```

Join

- Kada koristiti INNER a kada L/R OUTER JOIN
 - INNER JOIN upotrebljavamo kada želimo dohvatiti samo uparene zapise, odnosno kada nas ne zanimaju zapisi za koje u jednoj od tablica nema podataka (u smislu da uopće nema zapisa ili da je upisan podatak NULL)
 - INNER JOIN - U tablicama nad kojima je uspostavljen referencijalni integritet (odmah po kreiranju ili kasnije uz provjeru do tada upisanih podataka), ukoliko je ključ po kojem se spaja tablice obavezan podatak (primjer student/mjesto gdje je atribut student.pbrRod NOT NULL), ili ukoliko znamo da postoji barem jedan redak koji će biti uparen (svaka narudžba ima minimalno jednu stavku)
 - L/R JOIN se upotrebljava kad se očekuje da u jednoj od tablica prilikom joina po ključu neće biti nekih podataka !!
 - L/R JOIN koristimo kada želimo ispisati i 'uparene' i 'neuparene' zapise, a takvi zapisi nas također zanimaju
- 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.

Primjer 4 – join tablice i upita

- Želimo ispisati sve studente i ocjenu koju su dobili na ispitu iz baza podataka (ukoliko su polagali baze, u protivnom želimo vratiti ocjena = NULL). Ako je student više puta polagao ispit, ispisujemo više redaka s različitim ocjenama.

Korak 1: trebaju nam svi studenti

```
select s.*, ? from student s LEFT JOIN (?) on s.stud_id = ?
```

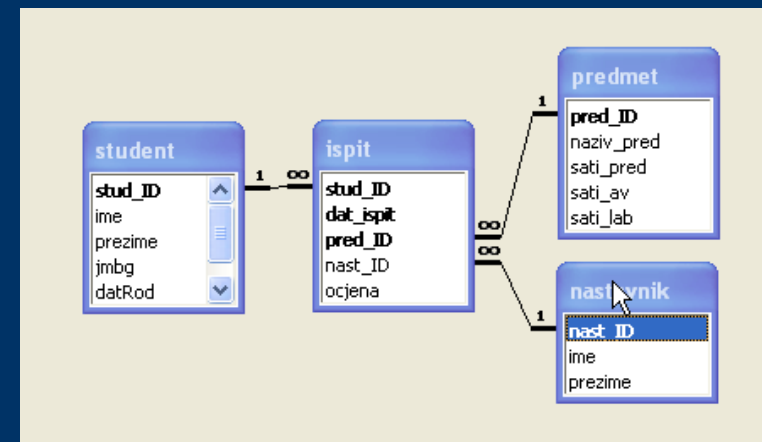
Korak 2: trebaju nam ispiti iz BPOD za pojedine studente

```
select i.stud_id, ocjena  
from predmet p INNER JOIN ispit i on p.pred_id = i.pred_id  
where p.naziv_pred = 'baze podataka'
```

Korak 3: upit pod 2 je ono s čim spajamo studente da bismo dobili rezultat

```
select s.*, X.ocjena from student s LEFT JOIN  
( select i.stud_id, ocjena  
  from predmet p INNER JOIN ispit i on p.pred_id = i.pred_id  
  where p.naziv_pred = 'baze podataka') as X  
on s.stud_id = X.stud_id
```

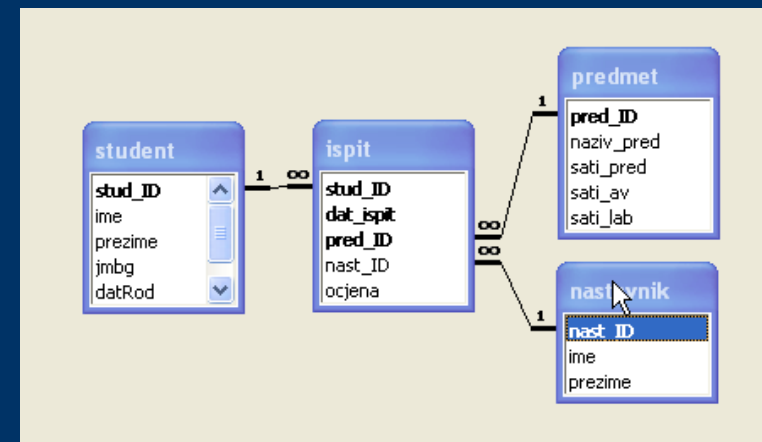

Vježba 3



- Ispisati sve studente koji su prošli baze podataka
- Ispisati predmete koje nitko nije položio
- Ispisati studente koji su pali neki ispit zajedno s datumom ispita, nazivima predmete i imenom i prezimenom nastavnika
- Ispisati ime studenta i nazive predmeta koje su položili za sve studente koji se prezivaju Jurić

Vježba 3

Ispisati sve studente koji su prošli baze podataka



```
select S.*
```

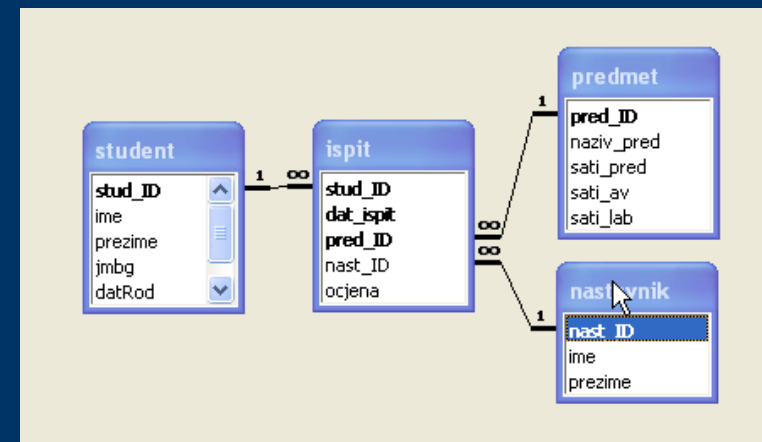
```
FROM student AS s INNER JOIN (predmet AS p INNER JOIN ispit AS i ON p.pred_ID = i.pred_ID) ON s.stud_ID = i.stud_ID
```

```
where p.naziv_pred = "baze podataka" and ocjena > 1
```

- Svaki zapis o ispitu odnosi se na nekog konkretnog studenta (stud_id) i neki konkretni predmet (pred_id)
- Prvo uz te podatke o ispitu "dopišemo" informacije o predmetu temeljem pred_id
- Nakon toga uz te podatke "dopišemo" informacije o studentu koji ga je polagao temeljem stud_id iz tablice student
- Možemo zamisliti da nakon ovog koraka imamo 1 veliku tablicu sa svim potrebnim informacijama

Vježba 3

Ispisati sve studente koji su prošli baze podataka



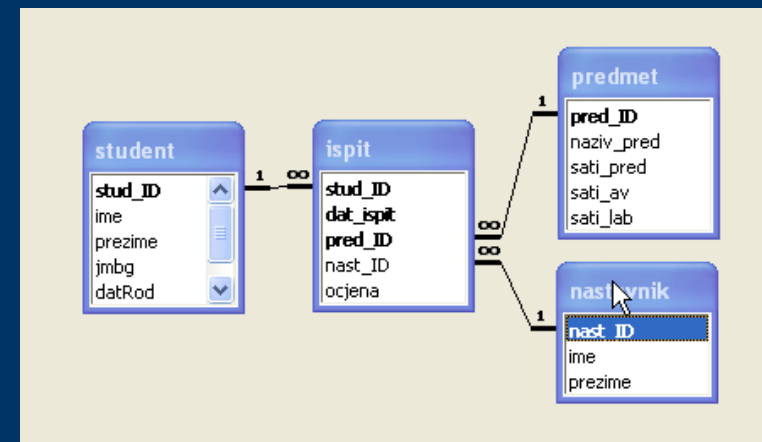
Od dobivenih redaka uzimaju se prema "WHERE uvjetu" samo oni naziva predmeta "baze podataka" s pozitivnom ocjenom

s.stud_ID	ime	prezime	jmbg	datRod	pbrRod	i.stud_ID	dat_ispit	i.pred_ID	nast_ID	ocjena	p.pred_ID	naziv_pred	sati_pred	sati_av	sati_lab
34211	Petra	Krleža	0506978657631	05.06.1978	10000	34211	03.05.2000	1008	452	3	1008	algoritmi i strukture p	2	1	1
25367	Davor	Dujmić	2504982561131	25.04.1982	10000	25367	03.05.2000	1008	452	3	1008	algoritmi i strukture p	2	1	1
10001	Ana	Jurić	1106980401236	11.06.1980	10000	10001	29.12.2001	1002	123	5	1002	baze podataka	3	2	2
11234	Ante	Jurić	0412979214665	04.12.1979	48000	11234	03.05.2000	1007	123	2	1007	elektronika	3	1	2
34678	Jurica	Domanovac	0606978543121	05.06.1978	10000	34678	19.04.2000	1001	235	5	1001	matematika	4	3	0
12244	Marija	Knjaz	1109980325234	11.09.1980	51000	12244	13.09.1999	1001	567	2	1001	matematika	4	3	0
22222	Mirta	Lončar	2709980127782	27.09.1980	52000	22222	13.04.2001	1004	112	4	1004	operacijski sustavi	2	2	2
12244	Marija	Knjaz	1109980325234	11.09.1980	51000	12244	22.11.2000	1004	235	1	1004	operacijski sustavi	2	2	2
25367	Davor	Dujmić	2504982561131	25.04.1982	10000	25367	14.10.1999	1003	258	4	1003	primjena računala	1	0	2
10001	Ana	Jurić	1106980401236	11.06.1980	10000	10001	29.12.2001	1003	567	5	1003	primjena računala	1	0	2
21112	Ivan	Županović	0311981376432	03.11.1981	31000	21112	20.01.2000	1005	333	2	1005	programiranje	2	2	1
34211	Petra	Krleža	0506978657631	05.06.1978	10000	34211	19.04.2000	1010	333	5	1010	programske paradigme	2	2	1
27783	Franjo	Mušnjak	1105981200211	11.05.1981	33000	27783	02.02.2002	1006	235	4	1006	računala i procesi	3	2	1
47678	Ljubica	Šaput	2412982700121	24.12.1982	44000	47678	11.11.2001	1009	768	3	1009	računalna grafika	2	1	1

Nakon toga uzimaju se samo oni stupci koji pripadaju tablici student

stud_ID	ime	prezime	jmbg	datRod	pbrRod
10001	Ana	Jurić	1106980401236	11.06.1980	10000

Vježba 3



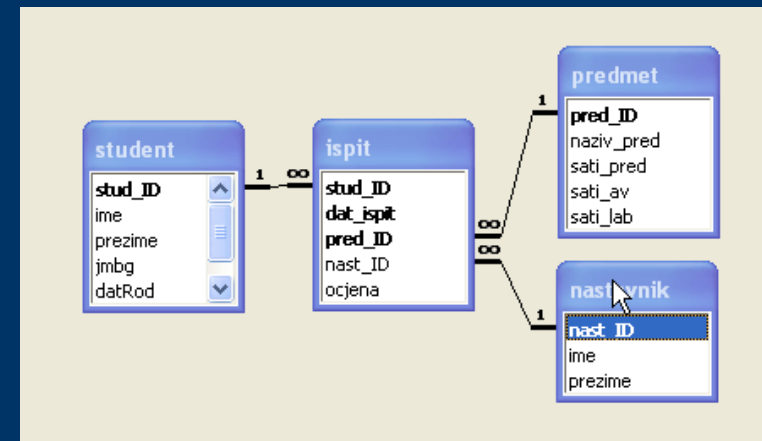
- Ispisati predmete koje nitko nije položio

```
select * from predmet
where pred_id not in (select pred_id from ispit where ocjena > 1)
```

- Ispisati studente koji su pali neki ispit zajedno s datumom ispita, nazivima predmete i imenom i prezimenom nastavnika

```
SELECT Student.*, Ispit.dat_ispit, Predmet.naziv_pred, Nastavnik.ime,
        Nastavnik.prezime
FROM (student INNER JOIN (predmet INNER JOIN ispit ON Predmet.pred_ID =
        Ispit.pred_ID) ON Student.stud_ID = Ispit.stud_ID) INNER JOIN
        nastavnik ON Ispit.nast_ID = Nastavnik.nast_ID
WHERE Ispit.ocjena=1;
```

Vježba 3



- Ispisati ime studenta i nazive predmeta koje su položili za sve studente koji se prezivaju Jurić

```
SELECT student.ime, predmet.naziv_pred
FROM predmet INNER JOIN (student INNER JOIN ispit ON
    student.stud_ID = ispit.stud_ID)
    ON predmet.pred_ID = ispit.pred_ID
WHERE student.prezime="Jurić" AND ispit.ocjena>1;
```

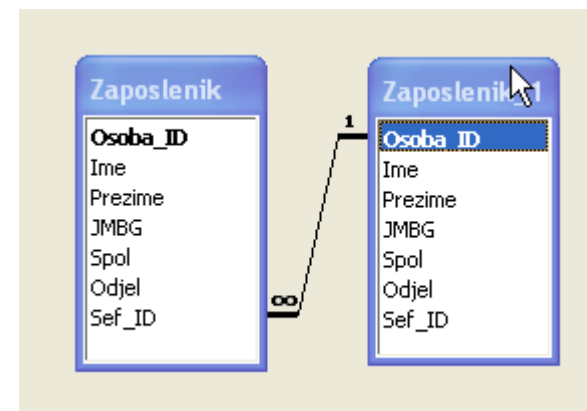
RING Struktura

- 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:



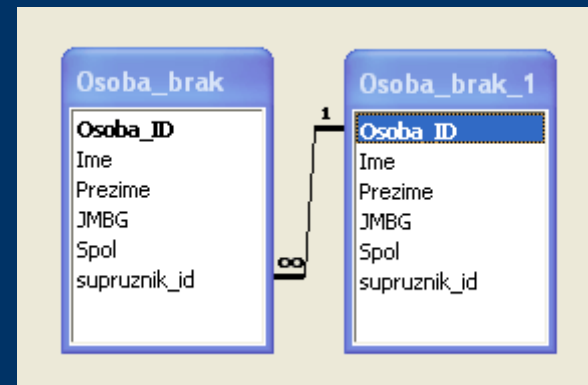
RING Struktura

- 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;
```

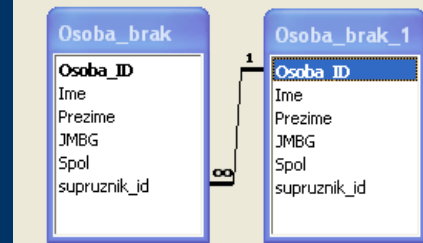
	ime_zaposlenika	prezime_zaposlenik	ime_nadredjenog	prezime_nadredjenog
	Ana	Anić	Ana	Anić
	Pero	Perić	Ana	Anić
	Mirta	Mirtiće	Ana	Anić
	Mate	Matić	Mate	Matić
	Mara	Marić	Mate	Matić
	Ante	Antić	Mate	Matić

Vježba 4



- osoba = (osoba_ID, ime, prezime, jmbg, spol, supruznik_id)
- Supruznik_ID 'pokazuje' na zapis o supružniku (mužu ili ženi) ukoliko je osoba u važećem braku.
- Ukoliko osoba nije u braku, podatak nije unesen (supruznik_ID is NULL)
- Spol ima ograničenje (check constraint) – smije se upisati samo 'M' ili 'Ž'. Podatak o spolu je obavezan pri unosu (not null).
- Zadatak1: dohvatiti podatke o imenu i prezimenu svih udanih žena, uz njih dohvatiti ime i prezime supruga.
- Zadatak2: dohvatiti podatke o svim ženama, uz one koje su udane dohvatiti i ime i prezime supruga.

Vježba 4



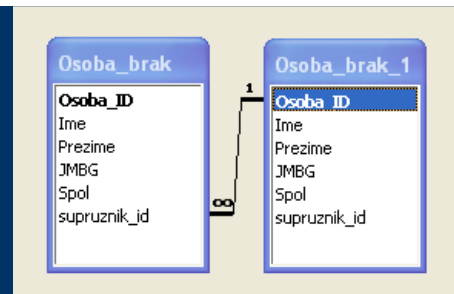
- Primjer podataka

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

- Kad kliknemo na “+” access će nam pokazati podatke na kojeg strani ključ tog zapisa pokazuje (u ovom slučaju to će biti zapis o supružniku iz iste tablice)

[illegible]

Vježba 4



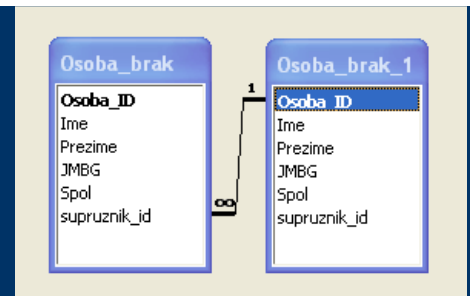
- Zadatak 1: dohvatiti podatke o imenu i prezimenu svih udanih žena, uz njih dohvatiti ime i prezime suprug.
- Upit možemo složiti ovako (spajanje imena i prezimena u jedan atribut je samo u svrhu demonstracije):

```
SELECT zena.Ime & ' ' & zena.Prezime AS suprug,  
       muz.Ime & ' ' & muz.Prezime AS suprug  
FROM Osoba_brak AS zena INNER JOIN Osoba_brak as  
     muz ON zena.supruznik_id=muz.osoba_ID  
WHERE zena.Spol="Ž";
```

- Rezultat:

	suprug	suprug
	Ana Anić	Pero Perić
	Mirta Mirtiće	Ante Antić
►		

Vježba 4



- Zadatak 2: dohvatiti podatke o svim ženama, uz one koje su udane dohvatiti i ime i prezime supruga.
- Upit možemo složiti ovako:

```
SELECT zena.*, muz.Ime & ' ' & muz.Prezime AS suprug
FROM Osoba_brak AS zena LEFT JOIN Osoba_brak as muz
    ON zena.supruznik_id=muz.osoba_ID
WHERE zena.Spol="Ž";
```

- Rezultat:

	Osoba_ID	Ime	Prezime	JMBG	Spol	supruznik_id	suprug
	1	Ana	Anić	111111111111	Ž	2	Pero Perić
	4	Mirta	Mirtiće	3212734621471	Ž	6	Ante Antić
	5	Mara	Marić	4238493242342	Ž		
►	(AutoNumber)						

dodatak

WHERE UVJET

- Where dio SQL upita sastoji se od 1 ili više uvjeta odvojenih logičkim operatorima AND/OR
- Uvjet je ispunjen ili nije, odn. je istina ili laž, odnosno je TRUE ili FALSE
- RDBMS će kao rezultat vratiti samo one zapise za koje je uvjet istinit
- U where uvjetu se također može koristiti funkcije ili složene izraze
- "WHERE" dio ne mora postojati u upitu

- Primjeri jednostavnih uvjeta:
 - `predmet.naziv = „baze podataka“`
 - `student.prezime = „Jurić“`
 - `zaposlenik.placa > 5000`
 - `mjesto.pbr = 10000`
 - `ispit.ocjena > 3`
 - `zaposlenik.bonus > zaposlenik.placa`
 - `zaposlenik.placa * 12 + zaposlenik.bonus > 150000`
 - `student.ime & " " & student.prezime = „Ante Jurić“`
 - `student.ime & " " & student.prezime in ("Ante Jurić", "Mario Klarić")`

WHERE UVJET

- Primjeri složenih uvjeta:
 - student.prezime = „Jurić” and ispit.ocjena = 1
 - *kad želimo izlistati što je Jurić pao*
 - pred_naziv = „baze podataka” or nastavnik.prezime = „Matić”
 - *kad želimo izlistati i baze i predmete koje predaje Matić, bez obzira da li Matić predaje i baze ili ne*
 - student.prezime = „Jurić” and ispit.ocjena = 1 or nastavnik.prezime = „Matić”
 - *kad želimo izlistati i kolegije koje je Jurić pao i kolegije koje predaje Matić*
 - student.ime & " " & student.prezime in ("Ante Jurić", "Mario Klarić") OR pbrRod = 21000
 - *kad želimo izlistati sve studente iz Splita i uz njih nabrojanu dvojicu bez obzira da li su iz Splita ili ne*

AND / OR / NOT

- Logicki operatori: 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

AND / OR / NOT

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

- $A \text{ or } B \text{ and } C = A \text{ or } (B \text{ and } C)$
- $\text{not } A \text{ or } B \text{ and } C = (\text{not } A) \text{ or } (B \text{ and } C)$
- $A \text{ and } B \text{ or } C = (A \text{ and } B) \text{ or } C$

- $\text{not}(X \text{ or } Y)$
 $= (\text{not } X) \text{ and } (\text{not } Y)$

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

- $\text{not } (X \text{ and } Y)$
 $= (\text{not } X) \text{ or } (\text{not } Y)$

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

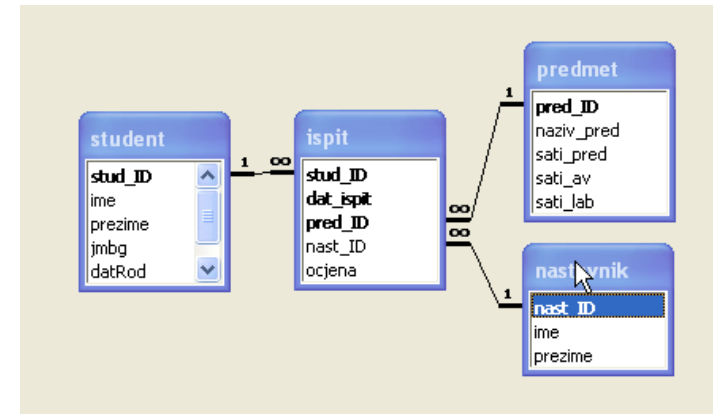
- Uvjet je moguće napisati na više različitih načina

Paziti!

Neka imamo inner join napisan ovako:

```
select * from student s, ispit i, predmet p
where s.stud_id = i.stud_id and i.pred_id = p.pred_id and i.ocjena = 5
```

- Upit vraća studente koji su dobili 5 iz nekog ispita zajedno s informacijama o tom ispitu i predmetu



Česta pogreška, želimo prepraviti upit da vraća one koji su dobili 4 ili 5 i napišemo ovo

```
select * from student s, ispit i, predmet p
where s.stud_id = i.stud_id and i.pred_id = p.pred_id and i.ocjena= 5 or ocjena= 4
```

- Zbog prioriteta and/or, Inner join se primjenjuje samo na uvjet i.ocjena= 5 a uz to se vraćaju svi zapisi produkta ove tri tablice za koje je ocjena = 4

MORA BITI ZAGRADA!!!

alternativno: ... and ocjena IN (4,5)

```
select * from student s, ispit i, predmet p
where s.stud_id = i.stud_id and i.pred_id = p.pred_id and (ocjena=5 or ocjena=4)
```

Datumske funkcije

Sistemske 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("21.03.2010")** = 21
- **MONTH** – za zadani datum vraća broj mjeseca. **MONTH("21.03.2002")** = 3
- **YEAR** – redni broj godine za zadani datum. **YEAR("21.03.2010")** = 2010

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".

MDB AutoNumber – Counter(start, step)

```
create table kupac
( kID          counter (100, 3) primary key,
  ime          varchar(100) not null,
  prezime     varchar(100) not null
);
```

	k_id	ime	prezime
	100	Pero	Perić
	103	Ana	Anić
	106	Mate	Matić
►	(AutoNumber)		

Kad imamo PK tipa counter, onda možemo napraviti insert novog zapisa bez tog podatka, a baza će sama pridijeliti slijedeću vrijednost: max trenutni kID + korak kojeg smo definirali (u gornjem primjeru 3)

```
insert into kupac (ime, prezime) values ('Mara', 'Marić');
```

- Dodaje novi zapis (**109**, 'Mara', 'Marić') u tablicu kupac

Da PK nije 'autoNumber' nego npr integer, morali bismo i taj podatak insertirati (minimalno za insert treba definirati sve obavezne attribute).