

# Baze podataka

# Predavanja

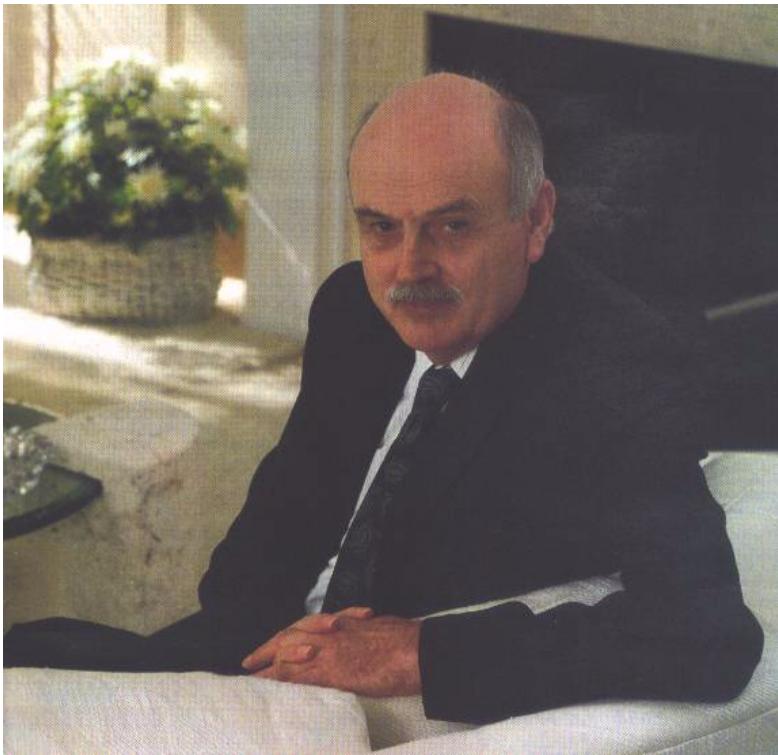
## 2. Relacijski model podataka

# ožujak 2020.



# Relacijski model podataka

- E. F. Codd: "A Relational Model of Data for Large Shared Data Banks", Comm. ACM 13, No. 6, June 1970.



Dr. Edgar Frank Codd (1923-2003)

Ciljevi relacijskog modela podataka:

- osigurati visoki stupanj nezavisnosti podataka
- postaviti temelje za rješavanje problema semantike, konzistentnosti i redundancije podataka (**normalizacija**)
- omogućiti razvoj DML jezika temeljenih na operacijama nad skupovima

# Relacijski model podataka

---

- Važni projekti u ranim 70-tim: jezik ISBL temeljen na relacijskoj algebri, jezici SQUARE i SEQUEL (DBMS System R) temeljeni na relacijskoj algebri i predikatnom računu te Query-By-Example temeljen na predikatnom računu nad domenama
  - razvojem prototipova dokazuje se praktična upotrebljivost relacijskog modela
  - postavljaju se temelji za rješavanje problema implementacije u područjima upravljanja transakcijama, paralelnog pristupa, obnove, optimizacije upita, sigurnosti i konzistentnosti podataka
- Projekti su potaknuli:
  - razvoj strukturiranog upitnog jezika (SQL)
  - razvoj komercijalnih **relacijskih** sustava za upravljanje bazama podataka (RDBMS)
    - Ingres, Oracle, IBM DB2, Informix, ...
    - danas: u upotrebi je nekoliko stotina različitih RDBMS sustava

# Relacijski model podataka

- objekti u relacijskom modelu podataka su RELACIJE

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
51000	Rijeka	2
52100	Pula	4
51300	Delnice	2
42230	Ludbreg	7

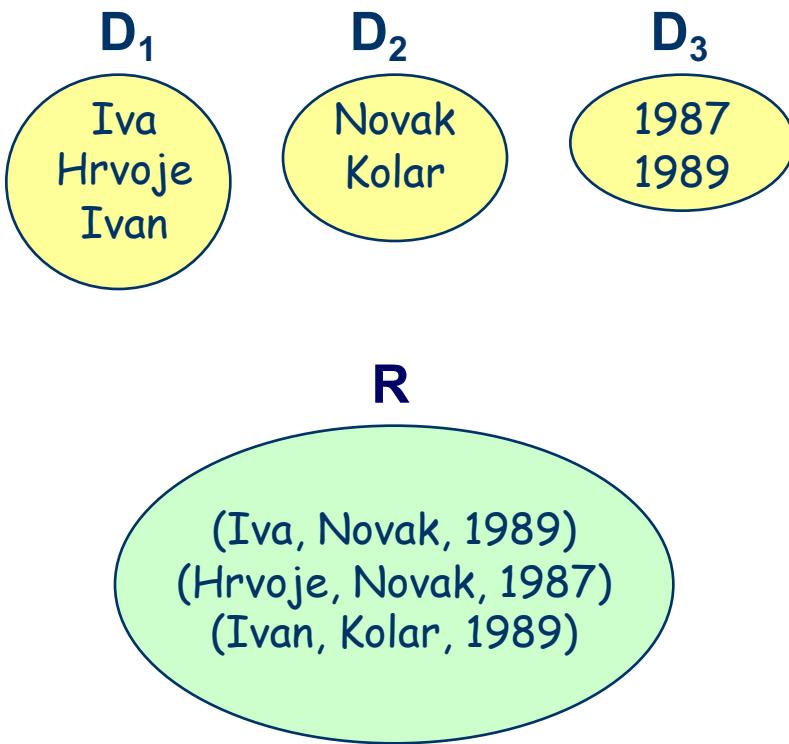
zupanija	
sifZup	nazZup
2	Primorsko-goranska
7	Varaždinska
4	Istarska

- neformalna definicija:** relacija je imenovana dvodimenzionalna tablica
  - atribut je imenovani stupac relacije
  - domena je skup dopuštenih vrijednosti atributa
    - nad istom domenom može biti definiran jedan ili više atributa
  - n-torka (*tuple*) je redak relacije

# Matematička relacija

- Relacija R definirana nad skupovima  $D_1, D_2, \dots, D_n$  je podskup Kartezijevog produkta skupova  $D_1, D_2, \dots, D_n$

$$R \subseteq D_1 \times D_2 \times \dots \times D_n$$



$$D_1 \times D_2 \times D_3$$

(Iva, Novak, 1987)  
(Iva, Novak, 1989)  
(Iva, Kolar, 1987)  
(Iva, Kolar, 1989)  
(Hrvoje, Novak, 1987)  
(Hrvoje, Novak, 1989)  
(Hrvoje, Kolar, 1987)  
(Hrvoje, Kolar, 1989)  
(Ivan, Novak, 1987)  
(Ivan, Novak, 1989)  
(Ivan, Kolar, 1987)  
(Ivan, Kolar, 1989)

# Relacijska shema (formalna definicija)

---

- Neka su zadani atributi  $A_1, A_2, \dots, A_n$ . Relacijska shema R (intenzija) je **imenovani skup atributa**

$$R = \{ A_1, A_2, \dots, A_n \}$$

- radi pojednostavljenja, koristit će se i sljedeća notacija:

$$R = A_1 A_2 \dots A_n$$

- uočite: poredak atributa u shemi relacije je nebitan

$$R = \{ A_1, A_2, A_3 \} \equiv \{ A_3, A_1, A_2 \}$$

- Primjer: relacijska shema MJESTO

$$MJESTO = \{ pbr, nazMjesto, sifZup \}$$

# Relacijska shema (primjer)

---

- Zadani su atributi pbr, nazMjesto, sifZup
- Relacijska shema
  - MJESTO = { pbr, nazMjesto, sifZup }
  - identična je relacijskoj shemi
  - MJESTO = { sifZup, pbr, nazMjesto }

# n-torka (formalna definicija)

---

- Neka je  $R = \{ A_1, A_2, \dots, A_n \}$  relacijska shema; neka su  $D_1, D_2, \dots, D_n$  domene atributa  $A_1, A_2, \dots, A_n$ ; **n-torka t** definirana na relacijskoj shemi R je skup parova oblika *atribut:vrijednostAtributa*  
$$t = \{ A_1:v_1, A_2:v_2, \dots, A_n:v_n \},$$
pri čemu je  $v_1 \in D_1, v_2 \in D_2, \dots, v_n \in D_n$
- Uočite: poredak elemenata n-torke nije bitan
$$\{ A_1:v_1, A_2:v_2, A_3:v_3 \} \equiv \{ A_3:v_3, A_1:v_1, A_2:v_2 \}$$
- Ponekad će se koristiti pojednostavljena notacija: prepostavi li se da poredak vrijednosti atributa odgovara "poretku atributa" u relacijskoj shemi, n-torka se može prikazati na sljedeći način:

$$t = < v_1, v_2, \dots, v_n >$$

# n-torka (primjer)

---

- Zadana je relacijska shema OSOBA = { matBr, ime, prez }, pri čemu su domene atributa:

dom (matBr) = {1234, 1235, 1236, 1237 }

dom (ime) = { Iva, Hrvoje, Ivan }

dom (prez) = { Novak, Kolar }

$t_1 = \{ \text{matBr:1234, ime:Iva, prez:Novak} \}$

$t_2 = \{ \text{matBr:1236, ime:Hrvoje, prez:Novak} \}$

$t_3 = \{ \text{matBr:1237, ime:Ivan, prez:Kolar} \}$

- n-torka  $t_1$  se jednako ispravno može napisati na sljedeći način  
 $t_1 = \{ \text{ime:Iva, prez:Novak, matBr:1234} \}$
- pojednostavljena notacija:

$t_1 = < 1234, Iva, Novak >$

# Relacija (formalna definicija)

---

- Neka je  $R = \{ A_1, A_2, \dots, A_n \}$  relacijska shema; neka su  $D_1, D_2, \dots, D_n$  domene atributa  $A_1, A_2, \dots, A_n$ ; **relacija r** (instanca relacije) definirana na shemi relacije R je **skup n-torki** koje su definirane na relacijskoj shemi R
- kad se želi naglasiti da je relacija r definirana na shemi relacije R, kao oznaka za relaciju koristi se  $r(R)$  ili  $r( \{ A_1, A_2, \dots, A_n \} )$  ili  $r( A_1 A_2 \dots A_n )$
- relacijska shema R: mijenja se relativno rijetko
- instanca relacije r: predstavlja trenutnu vrijednost relacije i često se mijenja (pri unosu/brisaju/izmjeni podataka)

# Relacija (primjer)

- Zadana je relacijska shema STUDENT = { matBr, prez, slika }, pri čemu su domene atributa:

- dom (matBr) = { 100, 102, 107, 111, 135 }
- dom (prez) = { Novak, Kolar, Horvat, Ban }
- dom (slika) = {  ,  ,  }

student(STUDENT) = { { matBr:102, prez:Novak, slika:  },  
                         { matBr:135, prez:Ban, slika:  } }

- IDENTIČNA RELACIJA (poredak n-torki i članova n-torki je nebitan):

student(STUDENT) = { { prez:Ban, matBr:135, slika:  },  
                         { slika: , matBr:102, prez:Novak } }

# Svojstva relacija

- relacija posjeduje ime koje je jedinstveno unutar sheme baze podataka
- atributi unutar relacijske sheme imaju jedinstvena imena (zašto?)
- jedan atribut može poprimiti vrijednost iz samo jedne domene
- u jednoj relaciji ne postoje dvije jednakе n-torke (zašto?)
- redoslijed atributa unutar relacijske sheme je nebitan (zašto?)
- redoslijed n-torki unutar relacije je nebitan (zašto?)

zupanija	
sifZup	nazZup
2	Primorsko-goranska
7	Varaždinska
4	Istarska

=

zupanija	
nazZup	sifZup
Varaždinska	7
Istarska	4
Primorsko-goranska	2

# Relacija (primjer)

$\text{student(STUDENT)} = \{ \{ \text{prez:Ban}, \text{matBr:135}, \text{slika: } \text{  
 $\quad \{ \text{slika: } \text{

- pojednostavljenje prikaza relacije (vizualizacija relacije tablicom)$$

student (STUDENT)		
matBr	prez	slika
102	Novak	<img alt='Icon of a woman holding a smartphone' data-bbox='620 615 660 675"/>
135	Ban	<img alt='Icon of a man holding a clipboard' data-bbox='620 695 660 755"/>

# A-vrijednost n-torke, X-vrijednost n-torke

- Oznaka  $t(A)$  predstavlja vrijednost koju atribut  $A$  poprima u n-torki  $t$ .  $t(A)$  se naziva A-vrijednost n-torke  $t$ .
- Primjer:

$t = \{ \text{matBr:102, prez:Novak, slika:} \text{ } \}$   
  
 $t(\text{pres}) = \text{Novak}$

- Neka je  $X \subseteq R$ . n-torka  $t$  reducirana na skup atributa  $X$  naziva se X-vrijednost n-torke  $t$  i označava s  $t(X)$
- Primjer:

$t = \{ \text{matBr:102, prez:Novak, slika:} \text{ } \}$   
  
 $X = \{ \text{matBr, prez} \} \quad X \subseteq R$   
 $t(X) = t(\{ \text{matBr, prez} \}) = \{ \text{matBr:102, prez:Novak} \}$

# Stupanj i kardinalnost relacije

- stupanj relacije: broj atributa (stupaca) - *degree*
- kardinalnost relacije: broj n-torki (redaka) - *cardinality*

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
51000	Rijeka	2
52100	Pula	4
51300	Delnice	2
42230	Ludbreg	7

stupanj = 3

kardinalnost = 5

Oznake:

$$\deg(mjesto) = 3$$

$$\text{card}(mjesto) = 5$$

# Shema i instanca baze podataka

---

- Shema baze podataka je **skup relacijskih shema**

$$\mathcal{R} = \{ R_1, R_2, \dots, R_n \}$$

- očito, relacijske sheme u jednoj shemi baze podataka moraju imati različita imena

- Instanca baze podataka definirana na shemi baze podataka  $\mathcal{R} = \{ R_1, R_2, \dots, R_n \}$  je **skup instanci relacija**

$$r = \{ r_1(R_1), r_2(R_2), \dots, r_n(R_n) \}$$

- shema baze podataka se relativno rijetko mijenja
- instanca baze podataka se često mijenja

---

# **Operacije u relacijskom modelu podataka**

# Relacijska algebra

---

- Operacije relacijske algebre su:

$\cup$	unija ( <i>union</i> )
$\cap$	presjek ( <i>intersection</i> )
$\setminus$	razlika ( <i>set difference</i> )
$\div$	dijeljenje ( <i>division</i> )
$\pi$	projekcija ( <i>projection</i> )
$\sigma$	selekacija ( <i>selection</i> )
$\times$	Kartezijev produkt ( <i>Cartesian product</i> )
$\rho$	preimenovanje ( <i>renaming</i> )
$\bowtie$	spajanje ( <i>join</i> )
	agregacija, grupiranje

Primjer:  $r_4 = \sigma_{A=x \wedge B=y} (r_1 \cup (r_2 \cap r_3))$

- Karakteristika relacijske algebre - proceduralnost - navodi se redoslijed operacija koje se provode nad relacijama

# Predikatni račun

---

- Operacije se specificiraju navođenjem predikata

$$r = \{ t \mid F(t) \}$$

- $t$  je varijabla koja predstavlja:

- $n$ -torke -  $n$ -torski račun

- rezultat  $r$  je skup  $n$ -torki  $t$  za koje je vrijednost predikata  $F$  istina

- domene - domenski račun

- rezultat je skup domena  $t$  za koje je vrijednost predikata  $F$  istina

- Primjer:

$$r_4 = \{ t \mid (r_1(t) \vee ((r_2(t) \wedge r_3(t)))) \wedge t(A)=x \wedge t(B)=y \}$$

- Predikatni račun je neproceduralan

- ne navodi se redoslijed operacija

- navode se predikati koje  $n$ -torke (domene) moraju zadovoljavati

# **SQL**

## **Kratki pogled**

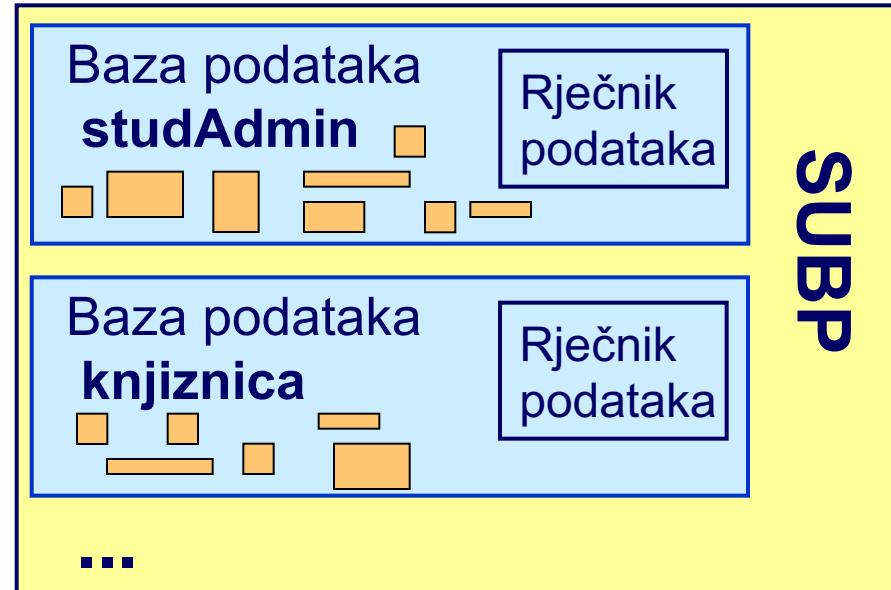
# SQL - Kratki pogled

- SQL (*Structured Query Language*) je temeljen na relacijskom modelu podataka.
- nastao je na temelju jezika SEQUEL
- temelji se na predikatnom računu i relacijskoj algebri
- proglašen standardnim jezikom za relacijske sustave
- objekti u SQL-u su tablice, a ne (formalno definirane) relacije
  - poredek atributa (stupaca) u nekim je slučajevima značajan
  - u tablici ili rezultatu operacija nad tablicama moguća je pojava dvije ili više istih n-torki
    - ipak, postoje načini kako se to može spriječiti

# SQL - Kratki pogled

- kreiranje nove instance baze podataka (kreiranje baze podataka)
  - jedan SUBP može istovremeno upravljati s više baza podataka

```
CREATE DATABASE studAdmin;
```



```
CREATE DATABASE knjiznica;
```



- Rječnik podataka sadrži opise relacijskih shema, integritetskih ograničenja, ...

```
DROP DATABASE knjiznica;
```



# SQL - Kratki pogled

- opisivanje relacijske sheme (kreiranje relacije)
  - kreira praznu relaciju
  - ujedno je moguće definirati i integritetska ograničenja

```
CREATE TABLE mjesto (
    pbr          INTEGER
, nazMjesto    CHAR(30)
, sifZup      SMALLINT
);
```



mjesto		
pbr	nazMjesto	sifZup

```
DROP TABLE mjesto;
```



# SQL - Kratki pogled

- upisivanje novih n-torki u relaciju

```
INSERT INTO mjesto  
VALUES (42000, 'Varaždin', 7);  
  
INSERT INTO mjesto  
VALUES (52100, 'Pula', 4);  
  
INSERT INTO mjesto  
VALUES (42230, 'Ludbreg', 7);
```

mjesto		
pbr	nazMjesto	sifZup



mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

Treba li poredak n-torki u relaciji biti u skladu s redoslijedom upisa?



# SQL - Kratki pogled

- dohvat podataka iz relacije

mjesto

pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

- dohvat podataka o mjestima čija šifra županije ima vrijednost 7

```
SELECT * FROM mjesto  
WHERE sifZup = 7;
```



pbr	nazMjesto	sifZup
42000	Varaždin	7
42230	Ludbreg	7

# SQL - Kratki pogled

- izmjena vrijednosti atributa u relaciji

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

- naziv mjesta s poštanskim brojem 42000 promijeniti u VARAŽDIN

```
UPDATE mjesto  
SET nazMjesto = 'VARAŽDIN'  
WHERE pbr = 42000;
```



mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	VARAŽDIN	7
52100	Pula	4

# SQL - Kratki pogled

- brisanje n-torki iz relacije

mjesto

pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	VARAŽDIN	7
52100	Pula	4

- obrisati mjesta za koje šifra županije ima vrijednost 7

```
DELETE FROM mjesto  
WHERE sifZup = 7;
```



mjesto

pbr	nazMjesto	sifZup
52100	Pula	4



# Relacijska algebra

# Relacijska algebra

---

- **Unarne operacije**

- projekcija, selekcija, preimenovanje
- agregacija, grupiranje

- **Binarne operacije**

- skupovske operacije (*set operations*)
  - temelje se na relacijama kao skupovima n-torki
  - unija, presjek, razlika
- ostale binarne operacije
  - Kartezijev produkt, dijeljenje, spajanje

# Relacijska algebra

---

- obavljanje operacije ne utječe na operande, npr.

$$r_3 = r_1 \cup r_2$$

- obavljanjem prethodne operacije nastaje nova relacija  $r_3$ , a relacije  $r_1$  i  $r_2$  se pri tome ne mijenjaju
- operandi su relacije, a rezultat obavljanja operacije je uvijek relacija. To znači:
  - skup relacija je **zatvoren** s obzirom na operacije relacijske algebre
  - ta činjenica omogućava da se rezultat jedne operacije upotrijebi kao operand u sljedećoj operaciji, što omogućava formiranje složenih izraza

$$r_5 = (r_1 \cup r_2) \times (r_3 \bowtie r_4)$$

# Unija relacija

- Dvije relacije su uniji kompatibilne ukoliko vrijedi:
  - relacije su istog stupnja
  - i
  - korespondentni atributi su definirani nad istim domenama

polozioMatem		
matBr	ime	prez
12345	Ivo	Kolar
13254	Ana	Horvat

polozioProgr		
mbr	prezSt	imeSt
92632	Ban	Jura
67234	Novak	Iva

- relacije su istog stupnja
- dom (matBr) = dom(mbr)
- dom (ime) = dom(imeSt)
- dom (prez) = dom(prezSt)

→ relacije su uniji kompatibilne

- kod ocjene jesu li relacije uniji kompatibilne
  - poredak atributa nije bitan
  - imena atributa nisu bitna

# Unija kompatibilnosti

- dvije relacije koje imaju jednak broj atributa i jednaka imena atributa ne moraju ujedno biti uniji kompatibilne

zrakoplov		pecivo	
oznaka	naziv	oznaka	naziv
B-747	Boeing 747	ZE	Žemlja
A-360	Airbus 360	PR	Perec

- relacije su istog stupnja
  - dom (*zrakoplov.oznaka*)  $\neq$  dom(*pecivo.oznaka*)
  - dom (*zrakoplov.naziv*)  $\neq$  dom(*pecivo.naziv*)
- **relacije NISU uniji kompatibilne**
- notacija *imeRelacije.imeAtributa* se često koristi kada je potrebno razlikovati istoimene attribute različitih relacija

# Skupovske operacije: unija, presjek, razlika

---

- Skupovske operacije (unija, presjek, razlika) mogu se obavljati isključivo nad UNIJSKI KOMPATIBILNIM relacijama

# Unija

- Rezultat operacije  $r_1 \cup r_2$  je relacija čije su n-torce elementi relacije  $r_1$  ili elementi relacije  $r_2$  ili elementi obje relacije.
  - n-torce koje su elementi obje relacije u rezultatu se pojavljuju samo jednom (jer relacija je SKUP n-torki)

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

**polozioBaremJedan =**

**polozioMatem  $\cup$  polozioProgr**

polozioBaremJedan

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
105	Rudi	Kolar
107	Jura	Horvat

studenti koji su položili ili  
Matematiku ili  
Programiranje ili  
oba predmeta

$$r_1 \cup r_2 \equiv r_2 \cup r_1$$

```
SELECT * FROM polozioMatem
UNION
SELECT * FROM polozioProgr;
```

# Presjek

- Rezultat operacije  $r_1 \cap r_2$  je relacija čije su n-torce elementi relacije  $r_1$  i elementi relacije  $r_2$

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

**polozioOba =**

**polozioMatem  $\cap$  polozioProgr**

polozioOba

mbr	ime	prez
102	Ana	Novak
107	Jura	Horvat

studenti koji su  
položili i Matematiku  
i Programiranje

$$r_1 \cap r_2 \equiv r_2 \cap r_1$$

```
SELECT * FROM polozioMatem
INTERSECT
SELECT * FROM polozioProgr;
```

# Razlika

- Rezultat operacije  $r_1 \setminus r_2$  je relacija čije su n-torce elementi relacije  $r_1$  i nisu elementi relacije  $r_2$

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

$$r_1 \setminus r_2 \neq r_2 \setminus r_1$$

polozioSamoMatem = polozioMatem \ polozioProgr

polozioSamoMatem

mbr	ime	prez
100	Ivan	Kolar
103	Tea	Ban

```
SELECT * FROM polozioMatem  
EXCEPT  
SELECT * FROM polozioProgr
```

polozioSamoProgr = polozioProgr \ polozioMatem

polozioSamoProgr

mbr	ime	prez
105	Rudi	Kolar

```
SELECT * FROM polozioProgr  
EXCEPT  
SELECT * FROM polozioMatem
```

# ŠTO AKO SE IMENA KORESPONDENTNIH ATRIBUTA RAZLIKUJU

- Unija, presjek, razlika: u slučajevima kada su relacije unijski kompatibilne, ali se u relacijama koriste različita imena korespondentnih atributa, primjenjuje se sljedeći dogovor (konvencija): kao imena atributa u rezultantnoj relaciji koriste se imena atributa prvog operanda

polozioMatem

mbr	imeSt	prezSt
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

$\text{polozioOba} = \text{polozioMatem} \cap \text{polozioProgr}$

polozioOba

mbr	imeSt	prezSt
102	Ana	Novak
107	Jura	Horvat

# Zadaci za vježbu

---

- zadane su unijski kompatibilne relacije
  - m (mbr ime prez) → studenti koji su položili Matematiku
  - d (mbr ime prez) → studenti koji su položili Dig. logiku
  - p (mbr ime prez) → studenti koji su položili Programiranje
- napisati izraze relacijske algebre koji određuju relacije koje sadrže studente (točnije rečeno n-torke):
  - a) koji su položili sva tri predmeta
  - b) koji su položili ili Matematiku ili Digitalnu logiku, ali ne oba predmeta (*ekskluzivni ili*)
  - c) koji su položili točno jedan (bilo koji) od ta tri predmeta
  - d) koji su položili bilo koja dva predmeta (ali nisu položili treći)

# Dijeljenje (*division*)

- Zadane su relacije  $r(R)$  i  $s(S)$ . Neka je  $S \subseteq R$ . Rezultat operacije  $r \div s$  je relacija sa shemom  $P = R \setminus S$ .  $n$ -torka  $t_r(P)$  se pojavljuje u rezultatu ako i samo ako za  $n$ -torku  $t_r \in r$  vrijedi da se  $t_r(P)$  u relaciji  $r$  pojavljuje u kombinaciji sa svakom  $n$ -torkom  $t_s \in s$

polozen	
mbrSt	sifPred
100	1
100	2
101	1
101	2
101	3
102	2
102	3
103	1
103	2
103	3
104	3

predmet	
sifPred	
1	
2	
3	

studenti koji su položili sve predmete  
sa šiframa u relaciji predmet

**poloziliSve** = **polozen**  $\div$  **predmet**

poloziliSve	
mbrSt	
101	
103	

# Selekcija

---

- Zadana je relacija  $r(R)$ . Neka je  $F$  predikat (formula, uvjet, *condition*) koji se sastoji od operanada i operatora
  - operandi su:
    - imena atributa iz  $R$
    - konstante
  - operatori su:
    - operatori usporedbe:  $<$   $\leq$   $=$   $\neq$   $>$   $\geq$
    - logički operatori:  $\wedge$   $\vee$   $\neg$
- Obavljanjem operacije  $\sigma_F(r)$  dobiva se relacija sa shemom  $R$  koja sadrži one n-torce relacije  $r$  za koje je vrijednost predikata  $F$  istina (*true*)

# Selekcija (primjer)

student

	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000
	107	Ana	Ban	51000

rezultat =  $\sigma_{ime = 'Ana' \vee postBr > 31000}$  (student)

- Za svaku pojedinu n-torku relacije:
  - vrijednosti atributa uvrštavaju se u predikat - uvrštavanjem vrijednosti u predikat dobiva se **sud**
  - onda i samo onda kada je vrijednost dobivenog suda istina (*true*), n-torka se pojavljuje u rezultatu selekcije

'Ivan' = 'Ana'  $\vee$  52000 > 31000  $\rightarrow$  *true*  
'Ana' = 'Ana'  $\vee$  10000 > 31000  $\rightarrow$  *true*  
'Jura' = 'Ana'  $\vee$  21000 > 31000  $\rightarrow$  *false*  
'Ana' = 'Ana'  $\vee$  51000 > 31000  $\rightarrow$  *true*

rezultat

	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	107	Ana	Ban	51000

# SQL - Selekcija

- **SELECT *SELECT List* FROM *table* [WHERE *Condition*]**
- Uvjet (*Condition*) se sastoji od operanada i operatora
  - operandi su:
    - imena atributa iz relacije *table*
    - konstante
  - operatori su:
    - operatori usporedbe: <   <=   =   <>   >   >=
    - logički operatori: AND   OR   NOT
- Vrijednosti svake n-torke iz relacije *table* se uvrštavaju u *Condition* (a to je u stvari predikat). Ako je dobiveni sud istinit (*true*), n-torka se pojavljuje u rezultatu.

# SQL - Lista za selekciju

mjesto

	pbr	nazMjesto	sifZup
42000	Varaždin	7	
52100	Pula	4	

- **SELECT *SELECT List* FROM *table***
- ***SELECT List*** je lista za selekciju: dio SELECT naredbe koji određuje koji će se "stupci" pojaviti u rezultatu

SELECT \* FROM mjesto;

=

```
SELECT mjesto.pbr  
      , mjesto.nazMjesto  
      , mjesto.sifZup  
FROM mjesto;
```

- uz ime atributa može se navesti ime relacije (radi izbjegavanja dvosmislenosti u slučajevima kada se podaci dohvaćaju istovremeno iz više relacija čija se imena atributa podudaraju)  
**imeRelacije.imeAtributa**
- u slučajevima kada takva dvosmislenost ne postoji, ime relacije se može (ali ne mora) ispuštiti

# SQL - Selekcija

student

	matBr	ime	prez	postBr
100	Ivan	Kolar	52000	
102	Ana	Horvat	10000	
105	Jura	Novak	21000	
107	Ana	Ban	51000	

$\sigma_{ime = 'Ana' \vee postBr > 31000} (student)$

```
SELECT * FROM student  
WHERE ime = 'Ana'  
OR postBr > 31000;
```

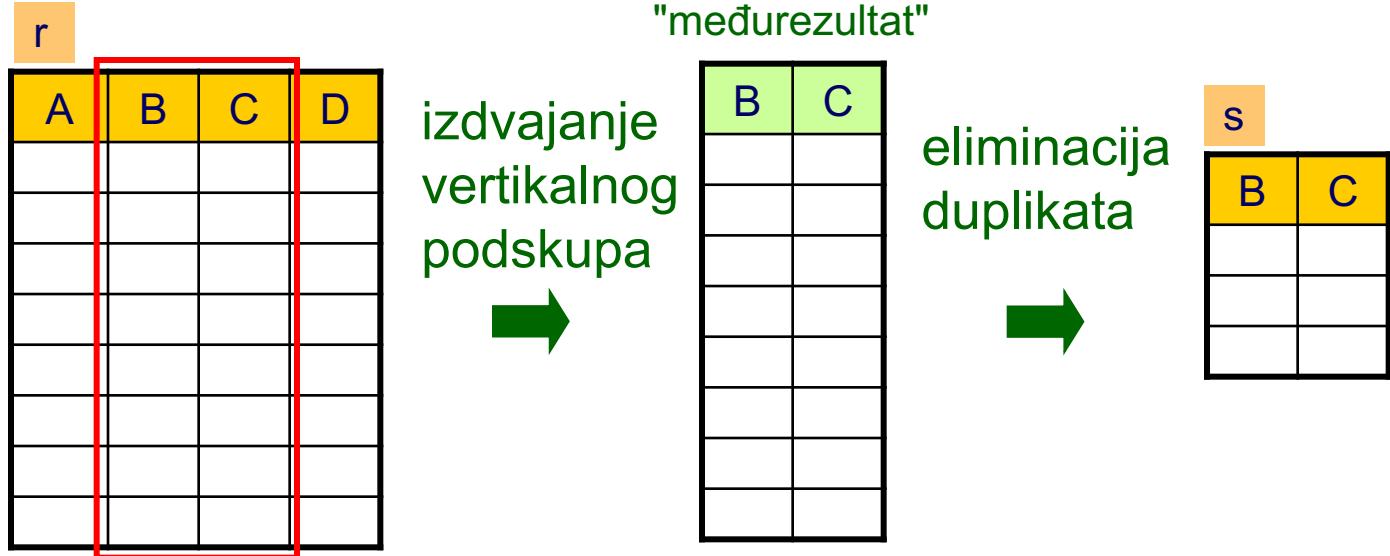


	matBr	ime	prez	postBr
100	Ivan	Kolar	52000	
102	Ana	Horvat	10000	
107	Ana	Ban	51000	

# Projekcija

- Zadana je relacija  $r(R)$ . Neka je skup atributa  $\{ A_1, A_2, \dots, A_k \} \subseteq R$
- Obavljanjem operacije  $\pi_{A_1, A_2, \dots, A_k}(r)$  dobiva se relacija s sa shemom  $\{ A_1, A_2, \dots, A_k \}$  koja sadrži vertikalni podskup relacije  $r$ 
  - $\deg(s) = k$
  - $\text{card}(s) \leq \text{card}(r)$  (jer se eliminiraju duplikati)

$$s = \pi_{B, C}(r)$$



# Projekcija (primjer)

Relacija nastup: u kojim gradovima nastup su nastupali koji tenori kojeg datuma

Traži se: u kojim gradovima su nastupali koji tenori

$$\text{tenorGrad} = \pi_{\text{tenor}, \text{grad}}(\text{nastup})$$

tenor	grad	datum
P. Domingo	London	15.2.1976
P. Domingo	New York	27.3.1981
P. Domingo	London	11.4.1987
J. Carreras	New York	11.4.1987
L. Pavarotti	Sydney	22.6.1992
L. Pavarotti	London	15.2.1976
L. Pavarotti	Sydney	19.1.1993
L. Pavarotti	London	14.7.1993

→ "međurezultat"

tenor	grad
P. Domingo	London
P. Domingo	New York
P. Domingo	London
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London
L. Pavarotti	Sydney
L. Pavarotti	London

tenorGrad →

tenor	grad
P. Domingo	London
P. Domingo	New York
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London

# SQL - Lista za selekciju

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
52100	Pula	4

zupanija	
sifZupanija	nazZup
7	Varaždinska
4	Istarska

- u listi za selekciju se ne moraju navesti svi atributi relacije navedene u FROM dijelu naredbe:

```
SELECT nazMjesto  
      , pbr  
  FROM mjesto;
```



nazMjesto	pbr
Varaždin	42000
Pula	52100

- u listi za selekciju se mogu navesti samo oni atributi koji se nalaze u dosegu SELECT naredbe, tj. atributi relacije koja je navedena u FROM dijelu naredbe:

```
SELECT nazMjesto  
      , pbr  
      , nazZup  
  FROM mjesto;
```

Neispravna naredba

# SQL - Projekcija

- za ispravno obavljanje projekcije nije dovoljno u listi za selekciju samo navesti imena atributa prema kojima se obavlja projekcija:

- primjer koji ujedno pokazuje kako rezultat SQL naredbe ne mora uvijek biti relacija



```
SELECT tenor  
      , grad  
  FROM nastup;
```

Neispravna projekcija

tenor	grad
P. Domingo	London
P. Domingo	New York
P. Domingo	London
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London
L. Pavarotti	Sydney
L. Pavarotti	London

$\pi_{\text{tenor}, \text{grad}}(\text{nastup})$

```
SELECT DISTINCT tenor  
      , grad  
  FROM nastup;
```

Ispravna projekcija

tenor	grad
P. Domingo	London
P. Domingo	New York
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London

# SQL - Projekcija i selekcija

student

	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000
	107	Ana	Ban	51000

$$\pi_{\text{ime}}(\sigma_{\text{ime} = \text{'Ana'} \vee \text{postBr} > 31000}(\text{student}))$$

```
SELECT DISTINCT ime  
FROM student  
WHERE ime = 'Ana'  
    OR postBr > 31000;
```

"medurezultat"

matBr	ime	prez	postBr
100	Ivan	Kolar	52000
102	Ana	Horvat	10000
107	Ana	Ban	51000

ime
Ivan
Ana

# Kartezijev produkt

---

- Zadana je relacija  $r(R)$  i relacija  $s(S)$ , pri čemu je  $R \cap S = \emptyset$ .
- Obavljanjem operacije  $r \times s$  dobiva se relacija  $p(P)$ ,  $P = R \cup S$ .  
n-torce relacije p se dobivaju spajanjem (ulančavanjem) svake n-torce iz relacije r sa svakom n-torkom iz relacije s
  - $\deg(p) = \deg(r) + \deg(s)$
  - $\text{card}(p) = \text{card}(r) \cdot \text{card}(s)$

# Kartezijev produkt (primjer)

student

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban

predmet

sifra	naziv
1	Programiranje
2	Matematika

$$\text{upis} = \text{student} \times \text{predmet}$$

upis

mbr	ime	prez	sifra	naziv
100	Ivan	Kolar	1	Programiranje
100	Ivan	Kolar	2	Matematika
102	Ana	Novak	1	Programiranje
102	Ana	Novak	2	Matematika
103	Tea	Ban	1	Programiranje
103	Tea	Ban	2	Matematika

# SQL - Kartezijev produkt

- **SELECT *SELECT List*  
FROM *table [, table]...*  
[WHERE *Condition*]**
- navede li se u FROM dijelu naredbe više od jedne relacije, obavlja se operacija Kartezijevog produkta navedenih relacija

student

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban

predmet

sifra	naziv
1	Programiranje
2	Matematika

student × predmet

**SELECT \***

**FROM student, predmet;**

**SELECT student.\* , predmet.\***

**FROM student, predmet;**

mbr	ime	prez	sifra	naziv
100	Ivan	Kolar	1	Programiranje
100	Ivan	Kolar	2	Matematika
102	Ana	Novak	1	Programiranje
102	Ana	Novak	2	Matematika
103	Tea	Ban	1	Programiranje
103	Tea	Ban	2	Matematika

# SQL - Kartezijev produkt

- drugačija sintaksa:
- **SELECT *SELECT List***  
**FROM *table* [CROSS JOIN *table*] . . .**  
**[WHERE *Condition*]**

```
SELECT *
  FROM student CROSS JOIN predmet;
```

- Kartezijev produkt triju relacija:

```
SELECT *
  FROM r1 CROSS JOIN r2 CROSS JOIN r3;
```

# Kartezijev produkt

- Što učiniti ukoliko je potrebno obaviti operaciju Kartezijevog produkta nad relacijama  $r(R)$  i  $s(S)$ , u slučaju kada  $R \cap S \neq \emptyset$

r	
A	B
1	a
2	b
3	c

s	
B	C
c	$\alpha$
d	$\beta$

A	B	B	C
1	a	c	$\alpha$
1	a	d	$\beta$
2	b	c	$\alpha$
2	b	d	$\beta$
3	c	c	$\alpha$
3	c	d	$\beta$



NIJE RELACIJA!

→ Potrebno je koristiti operaciju preimenovanja

# Preimenovanje (relacije, atributa)

- Zadana je relacija  $r(\{ A_1, A_2, \dots, A_n \})$ 
  - **preimenovanje relacije:** operacijom preimenovanja  $\rho_s(r)$  dobiva se relacija  $s$  koja ima jednaku shemu i sadržaj kao relacija  $r$
  - **preimenovanje relacije i atributa:** operacijom preimenovanja  $\rho_{s(B_1, B_2, \dots, B_n)}(r)$  dobiva se relacija  $s$  čija shema umjesto atributa  $A_1, A_2, \dots, A_n$  sadrži attribute  $B_1, B_2, \dots, B_n$ , a sadržaj relacije  $s$  je jednak sadržaju relacije  $r$

$$p = r \times \rho_{s(B2, C)}(s)$$

r	A	B
1	a	
2	b	
3	c	

s	B	C
c		α
d		β

p	A	B	B2	C
1	a		c	α
1	a		d	β
2	b		c	α
2	b		d	β
3	c		c	α
3	c		d	β

# SQL - Preimenovanje atributa

- ukoliko se drugačije ne navede, imena stupaca u rezultatu odgovaraju imenima atributa iz liste za selekciju
- implicitna imena stupaca rezultata se mogu promijeniti korištenjem operatora za preimenovanje **AS**

zupanija	sifZupanija	nazZup
7	Varaždinska	
4	Istarska	

```
SELECT sifZupanija AS sifraz  
      , nazZup AS nazz  
  FROM zupanija;
```

sifraZ	nazz
7	Varaždinska
4	Istarska

- rezervirana riječ AS smije se ispustiti

```
SELECT sifZupanija sifraz  
      , nazZup nazz  
  FROM zupanija;
```

# SQL - Preimenovanje atributa

- Primjer u kojem je potrebno koristiti preimenovanje atributa
  - SQL naredba bi bila ispravna i bez preimenovanja, ali tada kao rezultat ne bismo dobili relaciju (jer bi u shemi rezultata postojala dva atributa istog imena)

r	A	B
1	a	
2	b	
3	c	

s	B	C
	c	$\alpha$
	d	$\beta$

$r \times \rho_{s(B2, C)}(s)$

```
SELECT A, r.B, s.B AS B2, C  
FROM r, s;
```

	A	B	B2	C
1	a		c	$\alpha$
1	a		d	$\beta$
2	b		c	$\alpha$
2	b		d	$\beta$
3	c		c	$\alpha$
3	c		d	$\beta$

# Prirodno spajanje (*Natural Join*)

- Prirodno spajanje obavlja se na temelju jednakih vrijednosti istoimenih atributa.
- Zadane su relacije  $r(R)$  i  $s(S)$ . Neka je  $R \cap S = \{ A_1, A_2, \dots, A_n \}$ . Obavljanjem operacije  $r \bowtie s$  dobiva se relacija sa shemom  $R \cup S$  koja sadrži n-torce nastale spajanjem n-torki  $t_r \in r$ ,  $t_s \in s$ , za koje vrijedi  $t_r(A_1) = t_s(A_1) \wedge t_r(A_2) = t_s(A_2) \wedge \dots \wedge t_r(A_n) = t_s(A_n)$ .

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
52100	Pula	4	
42230	Ludbreg	7	

zupanija	sifZup	nazZup
7	Varaždinska	
4	Istarska	

$mjestouZupaniji = mjesto \bowtie zupanija$

mjestouZupaniji	pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7	Varaždinska	
52100	Pula	4	Istarska	
42230	Ludbreg	7	Varaždinska	

što možemo reći o stupnju rezultata?

# SQL - Prirodno spajanje

```
SELECT mjesto.* , zupanija.nazZup  
FROM mjesto, zupanija  
WHERE mjesto.sifZup = zupanija.sifZup;
```

- ili:

```
SELECT mjesto.* , zupanija.nazZup  
FROM mjesto JOIN zupanija  
ON mjesto.sifZup = zupanija.sifZup;
```

# SQL - Prirodno spajanje – jednostavnija sintaksa

```
SELECT mjesto.* , zupanija.nazZup  
FROM mjesto JOIN zupanija  
    USING(sifzup);
```

- ili:

```
SELECT mjesto.* , zupanija.nazZup  
FROM mjesto NATURAL JOIN zupanija;
```

NAPOMENA: ove naredbe specifične su za PostgreSQL!

# Prirodno spajanje - relacije bez istoimenih atributa

- Rezultat prirodnog spajanja relacija  $r(R)$  i  $s(S)$  za koje vrijedi da je  $R \cap S = \emptyset$  identičan je rezultatu obavljanja operacije Kartezijevog produkta  $r \times s$

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

zupanija	sifZupanija	nazZup
	7	Varaždinska
	4	Istarska

$mjestouZupaniji = mjesto \bowtie zupanija$

mjestouZupaniji	pbr	nazMjesto	sifZup	sifZupanija	nazZup
	42000	Varaždin	7	7	Varaždinska
	42000	Varaždin	7	4	Istarska
	52100	Pula	4	7	Varaždinska
	52100	Pula	4	4	Istarska
	42230	Ludbreg	7	7	Varaždinska
	42230	Ludbreg	7	4	Istarska

# Spajanje uz uvjet ili $\theta$ - spajanje ( $\theta$ - join)

- Zadane su relacije  $r(R)$  i  $s(S)$  pri čemu je  $R \cap S = \emptyset$ . Neka je  $F$  predikat oblika  $A_i \theta B_j$ , pri čemu je  $A_i \in R$ ,  $B_j \in S$ , a  $\theta$  je operator usporedbe iz skupa operatora  $\{ <, \leq, =, \neq, >, \geq \}$
- Obavljanjem operacije  $r \triangleright\triangleleft_F s$  dobiva se relacija koja sadrži  $n$ -torke iz  $r \times s$  za koje je vrijednost predikata  $F$  istina (true), odnosno:  
$$r \triangleright\triangleleft_F s = \sigma_F(r \times s)$$
što možemo reći o stupnju i kardinalnosti rezultata?
- Umjesto jednostavnog predikata  $A_i \theta B_j$ , može se koristiti složeni predikat doiven primjenom logičkih operatora nad jednostavnim predikatima oblika  $A_i \theta B_j$
- Problem spajanja uz uvjet relacija  $r(R)$  i  $s(S)$  kod kojih je  $R \cap S \neq \emptyset$ , rješava se na jednak način kao kod Kartezijevog produkta (korištenjem operatora preimenovanja)

# Spajanje uz uvjet (primjer)

linija

let	udaljenost
CA-825	700
LH-412	4800
BA-722	15000
CA-311	13000

zrakoplov

tip	dolet
B747	13000
A320	5400
DC-9	3100



mogućnost = linija  $\bowtie$  zrakoplov  
dolet  $\geq$  udaljenost

mogućnost

let	udaljenost	tip	dolet
CA-825	700	B747	13000
CA-825	700	A320	5400
CA-825	700	DC-9	3100
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000



Linije i zrakoplovi koji na  
tim linijama mogu letjeti

# SQL - Spajanje uz uvjet

- Koristi se ekvivalencija

$$r \bowtie_F s = \sigma_F(r \times s)$$

linija  $\bowtie$  zrakoplov  
dolet  $\geq$  udaljenost

linija		zrakoplov	
let	udaljenost	tip	dolet
CA-825	700	B747	13000
LH-412	4800	A320	5400
BA-722	15000	DC-9	3100
CA-311	13000		

```
SELECT *
  FROM linija, zrakoplov
 WHERE dolet >= udaljenost;
```

Kartezijev produkt

Selekcija

Linije i zrakoplovi koji na  
tim linijama mogu letjeti

let	udaljenost	tip	dolet
CA-825	700	B747	13000
CA-825	700	A320	5400
CA-825	700	DC-9	3100
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000

# SQL - Spajanje uz uvjet

- drugačija sintaksa:
- **SELECT *SELECT List***  
**FROM *table* [JOIN *table* ON *joinCondition*]...**  
**[WHERE *Condition*]**

```
SELECT *
  FROM linija JOIN zrakoplov
    ON dolet >= udaljenost;
```

- Spajanje uz uvjet triju relacija:

```
SELECT *
  FROM r1
    JOIN r2
      ON joinCondition
    JOIN r3
      ON joinCondition;
```

# SQL - Spajanje uz uvjet i selekcija

- Kako pronaći linije i zrakoplove koji na tim linijama mogu letjeti, ali samo za one linije na kojima je udaljenost veća od 4000 km

$\sigma_{\text{udaljenost} > 4000}(\text{linija} \bowtie \text{zrakoplov})$   
dolet  $\geq$  udaljenost

```
SELECT *
FROM linija, zrakoplov
WHERE dolet >= udaljenost
AND udaljenost > 4000;
```

```
SELECT *
FROM linija
JOIN zrakoplov
ON dolet >= udaljenost
WHERE udaljenost > 4000;
```

let	udaljenost	tip	dolet
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000

ili

# SQL - Spajanje uz uvjet i projekcija

- Kako pronaći tipove zrakoplova koji se mogu iskoristiti za letove na postojećim linijama

$\pi_{tip}(\text{linija } \bowtie \text{ zrakoplov})$   
dolet  $\geq$  udaljenost

```
SELECT DISTINCT tip  
FROM linija, zrakoplov  
WHERE dolet >= udaljenost;
```

tip
B747
A320
DC-9

ili

```
SELECT DISTINCT tip  
FROM linija  
JOIN zrakoplov  
ON dolet >= udaljenost;
```

# Spajanje s izjednačavanjem (*Equi-join*)

- Spajanje relacija s izjednačavanjem je poseban oblik spajanja uz uvjet u kojem se kao  $\theta$  operator koristi isključivo operator jednakosti (=)

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
52100	Pula	4	
42230	Ludbreg	7	

zupanija	sifZupanija	nazZup
7	Varaždinska	
4	Istarska	

$mjestouZupaniji = mjesto \bowtie zupanija$   
 $sifZup = sifZupanija$

mjestouZupaniji	pbr	nazMjesto	sifZup	sifZupanija	nazZup
42000	Varaždin	7	7	Varaždinska	
52100	Pula	4	4	Istarska	
42230	Ludbreg	7	7	Varaždinska	

- Problem spajanja s izjednačavanjem relacija r(R) i s(S) kod kojih je  $R \cap S \neq \emptyset$ , rješava se na jednak način kao kod Kartezijskog produkta (korištenjem operatora preimenovanja)

# SQL - Spajanje s izjednačavanjem

- Koristi se ekvivalencija

$$r \bowtie s = \sigma_F(r \times s)$$

F

mjesto  $\bowtie$  zupanija

sifZup = sifZupanija

```
SELECT *
  FROM mjesto, zupanija
 WHERE sifZup = sifZupanija;
```

mjesto

pbr	nazMjesto	sifZup
42000	Varaždin	7
52100	Pula	4
42230	Ludbreg	7

zupanija

sifZupanija	nazZup
7	Varaždinska
4	Istarska

ili

```
SELECT *
  FROM mjesto
 JOIN zupanija
 ON sifZup = sifZupanija;
```

# SQL - Prirodno spajanje ≠ Spajanje s izjednačavanjem

- prirodno spajanje se razlikuje od spajanja s izjednačavanjem po tome što se istoimeni atributi iz dviju relacija izbacuju (tako da od svakog ostane samo po jedan)

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

zupanija	sifZup	nazZup
	7	Varaždinska
	4	Istarska

$\text{mjestouZupaniji} = \text{mjesto} \bowtie \text{zupanija}$

```
SELECT mjesto.* , zupanija.nazZup
  FROM mjesto, zupanija
 WHERE mjesto.sifZup = zupanija.sifZup;
```

pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7	Varaždinska
52100	Pula	4	Istarska
42230	Ludbreg	7	Varaždinska

# Agregacija (aggregation)

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Kako izračunati prosjek ocjena na svim ispitima?

prosjek	prosjOcj
	3.625

# Agregacija

---

- Zadana je relacija  $r(R)$ . Neka je atribut  $A \in R$ . Neka je  $\mathcal{AF}$  agregatna funkcija. Rezultat operacije agregacije  $G_{\mathcal{AF}(A)}(r)$  je relacija stupnja 1 i kardinalnosti 1, pri čemu je vrijednost atributa određena primjenom funkcije  $\mathcal{AF}$  nad vrijednostima atributa A u svim n-torkama relacije r. Funkcija  $\mathcal{AF}$  može biti jedna od:
  - COUNT određuje broj pojava (broji sve, eventualni duplikati se također broje)
  - SUM izračunava sumu vrijednosti
  - AVG izračunava aritmetičku sredinu vrijednosti
  - MIN izračunava najmanju vrijednost
  - MAX izračunava najveću vrijednost
- naziv rezultantne relacije i atributa nije definiran operacijom, stoga se najčešće koristi u kombinaciji s operacijom preimenovanja
- također se koriste agregatne funkcije
  - COUNT-DISTINCT, SUM-DISTINCT, AVG-DISTINCT

# Agregacija

ispit

	mbrStud	akGod	nazPred	ocjena
100	2005	Matematika	3	
101	2005	Matematika	5	
102	2005	Matematika	2	
103	2006	Matematika	3	
100	2004	Fizika	5	
101	2006	Fizika	5	
102	2006	Fizika	2	
100	2005	Vjerojatnost	4	

- Prosjek ocjena na svim ispitima (rješenje):

$$\rho_{\text{projek}(\text{projOcj})}(G_{\text{AVG}(\text{ocjena})}(\text{ispit}))$$

projek

projOcj

3.625

```
SELECT AVG(ocjena) AS projOcj  
FROM ispit;
```

projOcj

3.625

# Agregacija (primjeri ostalih agregatnih funkcija)

osoba		
sifra	tezina	visina
101	62	170
103	94	186
105	74	181
107	62	165

$$\rho_{\text{rez1(broj1)}}(G_{\text{COUNT(sifra)}}(\text{osoba}))$$

rez1	broj1
	4

$$\rho_{\text{rez2(broj2)}}(G_{\text{SUM(tezina)}}(\text{osoba}))$$

rez2	broj2
	292

$$\rho_{\text{rez3(broj3)}}(G_{\text{AVG(visina)}}(\text{osoba}))$$

rez3	broj3
	175.5

$$\rho_{\text{rez4(broj4)}}(G_{\text{MAX(visina)}}(\text{osoba}))$$

rez4	broj4
	186

$$\rho_{\text{rez5(broj5)}}(G_{\text{MIN(tezina)}}(\text{osoba}))$$

rez5	broj5
	62

Moguće je odjednom izračunati više agregatnih vrijednosti:

$$\rho_{\text{rez6(broj6, broj7, broj8)}}(G_{\text{MIN(tezina), AVG(visina), MAX(visina)}}(\text{osoba}))$$

rez6	broj6	broj7	broj8
	62	175.5	186

# SQL - Agregatne funkcije

- naziv rezultantnog atributa nije definiran operacijom, stoga se koristi AS operator za preimenovanje

osoba	sifra	tezina	visina
	101	62	170
	103	94	186
	105	74	181
	107	62	165

```
SELECT COUNT(sifra) AS broj1 FROM osoba;
```

broj1  
4

```
SELECT SUM(tezina) AS broj2 FROM osoba;
```

broj2  
292

```
SELECT AVG(visina) AS broj3 FROM osoba;
```

broj3  
175.5

```
SELECT MAX(visina) AS broj4,  
       MIN(tezina) AS broj5  
FROM osoba;
```

broj4	broj5
186	62

# SQL - Agregatne funkcije

- agregatne funkcije s DISTINCT

osoba	sifra	tezina	visina
	101	62	170
	103	94	190
	105	74	170
	107	62	170

```
SELECT COUNT(DISTINCT visina) AS broj1  
FROM osoba;
```

broj1
2

```
SELECT SUM(DISTINCT tezina) AS broj2  
FROM osoba;
```

broj2
230

```
SELECT AVG(DISTINCT visina) AS broj3  
FROM osoba;
```

broj3
180

# Agregacija i grupiranje

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Zadatak: izračunati prosječnu ocjenu za svaki pojedini predmet
  - prosjek za Matematiku
  - prosjek za Fiziku
  - ... i za sve ostale predmete čiji se naziv pojavljuje u relaciji

# Agregacija i grupiranje

## ■ Loše rješenje:

- Za svaki predmet napisati po jedan upit

$$\rho_{\text{prosjek}(\text{prosjOcjMat})}(G_{\text{AVG}(\text{ocjena})}(\sigma_{\text{nazPred} = \text{'Matematika'}}(\text{ispit})))$$

```
SELECT AVG(ocjena) AS prosjOcjMat  
FROM ispit  
WHERE nazPred = 'Matematika' ;
```

prosjOcjMat
3.25

$$\rho_{\text{prosjek}(\text{prosjOcjFiz})}(G_{\text{AVG}(\text{ocjena})}(\sigma_{\text{nazPred} = \text{'Fizika'}}(\text{ispit})))$$

```
SELECT AVG(ocjena) AS prosjOcjFiz  
FROM ispit  
WHERE nazPred = 'Fizika' ;
```

prosjOcjFiz
4

- itd. (za svaki naziv predmeta)
- postoji li bolje rješenje?

# Grupiranje (*grouping*)

---

- Zadana je relacija  $r(R)$ . Neka su atributi  $A_1, A_2, \dots, A_m, B_1, B_2, \dots, B_n$  atributi sheme  $R$ . Opći oblik operacije grupiranja je sljedeći:

$$A_1, A_2, \dots, A_m G_{\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)}(r)$$

- a) određuju se grupe n-torki: u svakoj grupi se nalaze n-torke koje imaju jednake vrijednosti atributa  $A_1, A_2, \dots, A_m$
- b) za svaku grupu n-torki izračunavaju se vrijednosti agregatnih funkcija  $\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)$
- c) za svaku grupu formira se n-torka s vrijednostima atributa  $A_1, A_2, \dots, A_m$  i izračunatim vrijednostima agregatnih funkcija

# Agregacija i grupiranje

ispit

	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4



- Za svaki predmet ispisati prosječnu ocjenu (ispravno rješenje):

$\rho_{\text{projek}}(\text{nazPred}, \text{projOcj})(\text{nazPred} G_{\text{AVG}(\text{ocjena})}(\text{ispit}))$

- grupirati po nazPred
- za svaku grupu izračunati AVG(ocjena)
- za svaku grupu formirati po jednu n-torku s vrijednošću atributa nazPred i izračunatim projekom
- obaviti operaciju preimenovanja

projek

projek	nazPred	projOcj
	Matematika	3.25
	Fizika	4
	Vjerojatnost	4

# Agregacija i grupiranje

- Ispisati prosječnu i najveću ocjenu za svaki predmet i akademsku godinu:

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

}

- u istu grupu ulaze n-torce koje imaju jednake vrijednosti atributa nazPred i akGod

$\rho_{\text{projek1}(\text{nazPred}, \text{akGod}, \text{projOcj}, \text{maxOcj})}(\text{nazPred}, \text{akGod} G_{\text{AVG}(\text{ocjena})}, \text{MAX}(\text{ocjena}))(\text{ispit})$

projek1	nazPred	akGod	projOcj	maxOcj
	Matematika	2005	3.333	5
	Matematika	2006	3	3
	Fizika	2004	5	5
	Fizika	2006	3.5	5
	Vjerojatnost	2005	4	4

# SQL - Grupiranje

- **SELECT *SELECT List***  
**FROM . . .**  
**[WHERE *Condition*]**  
**[GROUP BY *column* [, *column*] . . . ]**

$\rho_{\text{projek1}(\text{nazPred}, \text{akGod}, \text{projOcj}, \text{maxOcj})}$  (nazPred, akGod  $G_{\text{AVG}(\text{ocjena}), \text{MAX}(\text{ocjena})}$  (ispit))

```
SELECT nazPred
      , akGod
      , AVG(ocjena) AS projOcj
      , MAX(ocjena) AS maxOcj
   FROM ispit
 GROUP BY nazPred, akGod;
```

nazPred	akGod	projOcj	maxOcj
Matematika	2005	3.333	5
Matematika	2006	3	3
Fizika	2004	5	5
Fizika	2006	3.5	5
Vjerojatnost	2005	4	4

# SQL - Grupiranje

- svi atributi koji se nalaze u listi za selekciju, a koji nisu argumenti agregatnih funkcija, moraju biti navedeni u GROUP BY dijelu naredbe

```
SELECT nazPred  
      , akGod  
      , mbrStud  
      , AVG(ocjena) AS prosjOcj  
      , MAX(ocjena) AS maxOcj  
  FROM ispit  
 GROUP BY nazPred, akGod;
```

NEISPRAVNO!

ispit

mbrStud	akGod	nazPred	ocjena
100	2005	Matematika	3
101	2005	Matematika	5
102	2005	Matematika	2
103	2006	Matematika	3
100	2004	Fizika	5
101	2006	Fizika	5
102	2006	Fizika	2
100	2005	Vjerojatnost	4

Zašto je to neispravno?

Za svaku grupu se formira samo po jedna n-torka: što s onim grupama u kojima postoji više vrijednosti atributa mbrStud?

nazPred	akGod	mbrStud	prosjOcj	maxOcj
Matematika	2005	100, 101, 102 ?	3.333	5
Matematika	2006	?	3	3
Fizika	2004	?	5	5
Fizika	2006	?	3.5	5
Vjerojatnost	2005	?	4	4

## Baze podataka

# Predavanja

## 3. Nepotpune informacije i NULL vrijednosti

Ožujak, 2020.



# NULL vrijednosti

- Ponekad se dešava da informacije koje treba unijeti u bazu podataka nisu potpune
  - neke informacije trenutno nisu poznate
  - neke informacije uopće ne postoje (nisu primjenjive)
  - neke informacije postoje, ali do njih nije moguće doći
- Informacije koje nedostaju prikazuju se kao poseban oblik podatka: NULL vrijednost

nije primjenjivo  
(vidi datum rođenja)

clanoviKnjiznice

mbr	ime	prez	pbr	datRodj	adresa	zanimanje
100	Maja	Novak	10000	01.5.2015	Ilica 1	NULL
105	Ivo	Kolar	21000	12.3.1973	NULL	odvjetnik
107	James	Bond	NULL	NULL	NULL	tajni agent

nedostupno

trenutno nepoznato

# SQL - Interna pohrana NULL vrijednosti

- NULL vrijednost se interno pohranjuje drugačije od bilo koje druge dopuštene vrijednosti (nije 0, nije 0.0, nije prazan niz, ...)
- Način interne pohrane NULL vrijednosti je nebitan - NULL vrijednost je neovisna od tipa podatka kojeg predstavlja. U SQL naredbama, bez obzira na tip podatka, koristi se "konstanta" **NULL**

```
CREATE TABLE mjesto (
    pbr          INTEGER
, nazMjesto    CHAR(30)
, sifZupanija SMALLINT
);
```

```
INSERT INTO mjesto VALUES (10000, 'Zagreb', NULL);
INSERT INTO mjesto VALUES (10001, NULL, 1);
UPDATE mjesto SET sifZupanija = NULL
WHERE pbr = 10001;
```

# SQL - prikaz NULL vrijednosti

- Način na koji se NULL vrijednost prikazuje korisniku ovisi o programskom alatu koji se koristi:

pgAdmin 4 SQL Client

```
1  SELECT *
2   FROM student
```

	jmbag character (10)	imestudent character (25)	prezimestudent character (25)	oib character (11)	spol character (1)
1	0555000443	Neven	Kolaric	[null]	M
2	0555000950	Mijo	Klasic	[null]	M

Valentina Studio SQL Client

```
1  SELECT * FROM student
```

	jmbag	imestudent	prezimestudent	oib	spol
1	0555000443	Neven	Kolaric	<NULL>	M
2	0555000950	Mijo	Klasic	<NULL>	M

# SQL - Izrazi

- Izraz (*Expression*) se sastoji od
  - imena atributa
  - konstanti
  - operatora      +    -    \*    /      unarni + -
  - zagrada      ( )
- Izrazi se mogu koristiti
  - u listi za selekciju
  - u uvjetu u WHERE dijelu naredbe
  - i drugdje ...

# SQL - Izrazi

- iznosi plaća za svaku osobu su navedeni u kolumnama
- ispisati matični broj, prezime i plaću izraženu u dolarima, za one osobe čija je plaća veća od 1000 eura
- $1 \text{ USD} = 5.6 \text{ KN}, 1 \text{ KN} = 0.136 \text{ EUR}$

dohodak		
mbr	prez	placa
100	Novak	7500
102	Horvat	5600
105	Kolar	9000
107	Ban	4200

```
SELECT mbr  
      , prez  
      , placac/5.6 ?  
FROM dohodak  
WHERE placac*0.136 > 1000;
```

mbr	prez	(expression)
100	Novak	1339.28571429
105	Kolar	1607.14285714

```
SELECT mbr  
      , prez  
      , placac/5.6 AS placacUSD  
FROM dohodak  
WHERE placac*0.136 > 1000;
```

mbr	prez	placaUSD
100	Novak	1339.28571429
105	Kolar	1607.14285714

# NULL vrijednost u izrazima

- Neka je binarni operator  $\alpha \in \{ +, -, *, / \}$ , a X i Y su izrazi
  - ako jedan ili oba operanda X, Y poprimaju NULL vrijednost, tada je rezultat izraza X  $\alpha$  Y također NULL vrijednost

5 + NULL → NULL

NULL - NULL → NULL

NULL \* 0 → NULL

- Neka je unarni operator  $\beta \in \{ +, - \}$ , a X je izraz
  - ako operand X poprima NULL vrijednost, tada je rezultat izraza  $\beta$  X također NULL vrijednost

- NULL → NULL

# SQL - NULL vrijednost u izrazima

bodovi

	mbr	prez	bodLab	bodMI
101	Novak	12	NULL	
103	Ban	NULL	NULL	
107	Horvat	21	66.3	
109	Kolar	NULL	54.3	

```
SELECT mbr
      , prez
      , bodLab + bodMI AS ukupBodova
      , - bodLab AS negBodLab
FROM bodovi;
```

	mbr	prez	ukupBodova	negBodLab
101	Novak	NULL	-12	
103	Ban	NULL	NULL	
107	Horvat	87.3	-21	
109	Kolar	NULL	NULL	

# NULL vrijednost u uvjetima usporedbe

---

- Neka su  $X$  i  $Y$  izrazi, a  $\gamma$  je operator usporedbe
$$\gamma \in \{ <, \leq, =, \neq, >, \geq \}$$
- ako niti jedan od operanada  $X$ ,  $Y$  nije NULL vrijednost, tada je rezultat izraza  $X \gamma Y$  logička vrijednost istina (*true*) ili logička vrijednost laž (*false*)
- ako jedan ili oba operanda  $X$ ,  $Y$  jesu NULL vrijednosti, tada je rezultat izraza  $X \gamma Y$  logička vrijednost nepoznato (*unknown*)

$7 \geq 5$	$\rightarrow$ true
'atlas' > 'zvuk'	$\rightarrow$ false
$-17.8 \leq \text{NULL}$	$\rightarrow$ unknown
$\text{NULL} = \text{NULL}$	$\rightarrow$ unknown
$\text{NULL} \neq \text{NULL}$	$\rightarrow$ unknown

# Operacija selekcije - NULL vrijednosti

- Obavljanjem **operacije selekcije**  $\sigma_F(r)$  dobiva se relacija koja sadrži samo one n-torke relacije r za koje je vrijednost predikata F istina (*true*). To znači da se n-torke za koje je vrijednost predikata F laž (*false*) ili nepoznato (*unknown*) ne pojavljuju u rezultatu

r	A	B
1	20	
2	NULL	
3	60	

$$s = \sigma_{B \leq 50}(r)$$

$20 \leq 50 \rightarrow \text{true}$   
 $\text{NULL} \leq 50 \rightarrow \text{unknown}$   
 $60 \leq 50 \rightarrow \text{false}$

s	A	B
1	20	

r	A	B
1	20	
2	NULL	
3	60	

$$s = \sigma_{B \neq 20}(r)$$

$20 \neq 20 \rightarrow \text{false}$   
 $\text{NULL} \neq 20 \rightarrow \text{unknown}$   
 $60 \neq 20 \rightarrow \text{true}$

s	A	B
3	60	

# SQL - Selekcija i NULL vrijednosti

student

	matBr	prez	postBr
100	Kolar	52000	
102	Horvat	10000	
105	Novak	NULL	
107	Ban	10000	

$\sigma_{\text{postBr} = 10000}$

```
SELECT * FROM student  
WHERE postBr = 10000;
```



	matBr	prez	postBr
102	Horvat	10000	
107	Ban	10000	

$\sigma_{\text{postBr} \neq 10000}$

```
SELECT * FROM student  
WHERE postBr <> 10000;
```



	matBr	prez	postBr
100	Kolar	52000	

- gdje je Novak ?

# SQL - operatori usporedbe IS NULL, IS NOT NULL

student

	matBr	prez	postBr
100	Kolar	52000	
102	Horvat	10000	
105	Novak		NULL
107	Ban	10000	

- U SQL-u nije dopušteno operatore usporedbe  $<$ ,  $\leq$ ,  $=$ ,  $\neq$ ,  $>$ ,  $\geq$  koristiti u kombinaciji s "konstantom" NULL (npr.  $=NULL$ ,  $\neq NULL$ , ...)

```
SELECT * FROM student  
WHERE postBr = NULL;
```

**Neispravna naredba**

```
SELECT * FROM student  
WHERE postBr IS NULL;
```



matBr	prez	postBr
105	Novak	NULL

```
SELECT * FROM student  
WHERE postBr IS NOT NULL;
```



matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
107	Ban	10000

- Rezultat logičkog izraza  $X IS NULL$  ili logičkog izraza  $X IS NOT NULL$  je uvijek ili *true* ili *false*

# Trovalentna logika

- Osnovne logičke operacije - tablice istinitosti u prisustvu logičke vrijednosti *unknown*

AND	true	unknown	false
true	true	unknown	false
unknown	unknown	unknown	false
false	false	false	false

OR	true	unknown	false
true	true	true	true
unknown	true	unknown	unknown
false	true	unknown	false

NOT	
true	false
unknown	unknown
false	true

# Selekcija, logički operatori i NULL vrijednosti

$$s = \sigma_{B \leq 50 \wedge C \neq 300}(r)$$

r	A	B	C
1	NULL	100	
2	20	200	
3	30	300	
4	40	NULL	
5	50	500	
6	60	NULL	

$\text{NULL} \leq 50 \wedge 100 \neq 300 \rightarrow \text{unknown}$  (\*)  
 $20 \leq 50 \wedge 200 \neq 300 \rightarrow \text{true}$   
 $30 \leq 50 \wedge 300 \neq 300 \rightarrow \text{false}$   
 $40 \leq 50 \wedge \text{NULL} \neq 300 \rightarrow \text{unknown}$   
 $50 \leq 50 \wedge 500 \neq 300 \rightarrow \text{true}$   
 $60 \leq 50 \wedge \text{NULL} \neq 300 \rightarrow \text{false}$  (\*\*)

\*  $\text{NULL} \leq 50 \rightarrow \text{unknown}$ ;  $100 \neq 300 \rightarrow \text{true}$ ;  $\text{unknown} \wedge \text{true} \rightarrow \text{unknown}$

\*\*  $60 \leq 50 \rightarrow \text{false}$ ;  $\text{NULL} \neq 300 \rightarrow \text{unknown}$ ;  $\text{false} \wedge \text{unknown} \rightarrow \text{false}$

s	A	B	C
2	20	200	
5	50	500	

# SQL - Logički operatori i NULL vrijednosti

bodovi

mbr	prez	bodLab	bodMI
101	Novak	6	NULL
103	Ban	NULL	NULL
105	Horvat	12	44.0
107	Kolar	NULL	85.0
109	Pevec	20	15.0

- Za prolaz je potrebno barem 10 bodova iz labosa i barem 50 bodova ukupno. Studentima koji nisu dolazili na labos ili izlazili na međuispite upisana je NULL vrijednost. Ispisati studente koji **nisu položili** ispit.

Ovaj upit ne daje zadovoljavajući rezultat

```
SELECT *
  FROM bodovi
 WHERE bodLab < 10
   OR bodMI + bodLab < 50;
```

mbr	prez	bodLab	bodMI
101	Novak	6	NULL
109	Pevec	20	15.0

```
SELECT *
  FROM bodovi
 WHERE bodLab IS NULL
   OR bodMI IS NULL
   OR bodLab < 10
   OR bodLab + bodMI < 50;
```

mbr	prez	bodLab	bodMI
101	Novak	6	NULL
103	Ban	NULL	NULL
107	Kolar	NULL	85.0
109	Pevec	20	15.0

# NULL vrijednosti i skupovi

---

- Neka skup  $S$  sadrži vrijednosti:  $S = \{1, 2, 3, \text{NULL}\}$
- NULL vrijednost je nepoznata, ali može poprimiti i neku od vrijednosti 1, 2 ili 3
  - kardinalnost skupa  $S$  je neodređena (može biti 3 ili 4)
  - narušena je definicija skupa (u skupu nije dozvoljena pojava dviju ili više jednakih vrijednosti)
  - što je logička vrijednost suda  $\text{NULL} \in S \rightarrow \text{unknown}$
  - što je logička vrijednost suda  $4 \in S \rightarrow \text{unknown}$

# NULL vrijednosti i skupovi

---

- Sustavi za upravljanje bazama podataka nisu u stanju međusobno razlikovati NULL vrijednosti, stoga se kao konvencija koristi sljedeći model rukovanja s NULL vrijednostima u skupovima:
  - dopuštena je pojava jedne i samo jedne NULL vrijednosti u skupu
  - element **e** je **kopija** jednog od elemenata u skupu:
    - ako vrijednost elementa **e** nije NULL, a u skupu postoji element s jednakom vrijednošću
    - ili
    - ako vrijednost elementa **e** jest NULL, a u skupu **S** već postoji element s NULL vrijednošću

# Kopija n-torke

---

- elementi relacije su n-torke
- Definicija kopije n-torke:
  - neka su  $t_1$  i  $t_2$  n-torke definirane na shemi  $\{ A_1, A_2, \dots, A_n \}$
  - $t_1 = \langle d_1, d_2, \dots, d_n \rangle$ ,  $t_2 = \langle e_1, e_2, \dots, e_n \rangle$
  - n-torka  $t_1$  je kopija n-torke  $t_2$  ako i samo ako  $\forall i, 1 \leq i \leq n$ , vrijedi:
  - $(d_i = e_i) \vee (d_i \text{ jest } \text{NULL} \wedge e_i \text{ jest } \text{NULL})$
- neformalno: ako su vrijednosti korespondentnih atributa n-torki ili jednake ili su obje NULL

# Kopija n-torke

- Primjer:

osoba

mbr	ime	prez	postBr
100	Ivan	Novak	10000
102	Ana	Horvat	21000
103	Tea	Ban	52000
105	NULL	Kolar	NULL

student

mbr	ime	prez	postBr
100	Ivan	Novak	NULL
102	Ana	Horvat	21000
103	Tea	Ban	21000
105	NULL	Kolar	NULL

nije kopija

jest kopija

nije kopija

jest kopija

# Unija, razlika i presjek - NULL vrijednosti

- unija, razlika i presjek su skupovske operacije: pri usporedbi elemenata (n-torki) treba voditi računa o definiciji kopije n-torke

r	A	B	C
1	a	$\alpha$	
2	b	NULL	
3	NULL	$\gamma$	
4	NULL	NULL	

s	A	B	C
1	a	$\alpha$	
2	b	NULL	
3	c	$\gamma$	
4	NULL	NULL	

$r \cup s$	A	B	C
1	a	$\alpha$	
2	b	NULL	
3	NULL	$\gamma$	
3	c	$\gamma$	
4	NULL	NULL	

$r \cap s$	A	B	C
1	a	$\alpha$	
2	b	NULL	
4	NULL	NULL	

$r \setminus s$	A	B	C
3	NULL	$\gamma$	

$s \setminus r$	A	B	C
3	c	$\gamma$	

# Projekcija - NULL vrijednosti

- pri obavljanju operacije **projekcije** potrebno je u fazi eliminacije duplikata voditi računa o definiciji kopije n-torke

$$s = \pi_{B, C}(r)$$

izdvajanje  
vertikalnog  
podskupa

r	A	B	C
1	a	$\alpha$	
2	b	NULL	
3	NULL	$\gamma$	
4	a	$\alpha$	
5	NULL	NULL	
6	NULL	$\gamma$	
7	NULL	NULL	

eliminacija  
duplikata

"međurezultat"

	B	C
	a	$\alpha$
	b	NULL
	NULL	$\gamma$
	a	$\alpha$
	NULL	NULL
	NULL	$\gamma$
	NULL	NULL



s	B	C
	a	$\alpha$
	b	NULL
	NULL	$\gamma$
	NULL	NULL

# Kartezijev produkt - NULL vrijednosti

- pri obavljanju operacije **Kartezijevog produkta** NULL vrijednosti nemaju utjecaja

r	A	B	C
1	a	$\alpha$	
2	b	NULL	
3	NULL	NULL	

s	E	F
1	NULL	
NULL	f	

r × s	A	B	C	E	F
1	a	$\alpha$		1	NULL
2	b	NULL		1	NULL
3	NULL	NULL		1	NULL
1	a	$\alpha$		NULL	f
2	b	NULL		NULL	f
3	NULL	NULL		NULL	f

# Spajanje uz uvjet i spajanje s izjednačavanjem - NULL vrijednosti

- pri obavljanju operacija **spajanja uz uvjet** i **spajanja s izjednačavanjem** potrebno je voditi računa o tome da se spajaju samo one n-torce za koje uvjet spajanja ima logičku vrijednost istina (*true*)

linija	
let	udaljenost
CA-825	700
LH-412	NULL
BA-722	4100
CA-311	13000

zrakoplov	
tip	dolet
B747	NULL
A320	5400
DC-9	3100

mogućnost = linija  $\bowtie$  zrakoplov  
dolet  $\geq$  udaljenost

mogućnost			
let	udaljenost	tip	dolet
CA-825	700	A320	5400
CA-825	700	DC-9	3100
BA-722	4100	A320	5400

# Prirodno spajanje - NULL vrijednosti

- slično, pri obavljanju operacije **prirodnog spajanja** potrebno je voditi računa o tome da se spajaju samo one n-torce za koje uvjet spajanja ima logičku vrijednost istina (*true*)

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
42230	Ludbreg	NULL	
42220	Novi Marof	7	

zupanija	sifZup	nazZup
7	Varaždinska	
NULL	Istarska	

$\text{mjestouZupaniji} = \text{mjesto} \triangleright\triangleleft \text{zupanija}$

mjestouZupaniji	pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7		Varaždinska
42220	Novi Marof	7		Varaždinska

- n-torka  $\langle 42230, \text{Ludbreg}, \text{NULL} \rangle$  neće se spojiti s n-torkom  $\langle \text{NULL}, \text{Istarska} \rangle$  jer je rezultat usporedbe  $\text{NULL}=\text{NULL} \rightarrow \text{unknown}$

# Agregacija - NULL vrijednosti

---

- ako su sve vrijednosti za koje se izračunava agregatna funkcija NULL vrijednosti, ili ako se aggregatna funkcija izračunava za prazan skup vrijednosti
  - rezultat aggregatne funkcije COUNT je nula
  - rezultat ostalih aggregatnih funkcija je NULL
- ako među vrijednostima za koje se izračunava aggregatna funkcija postoje vrijednosti koje nisu NULL vrijednosti
  - aggregatna funkcija se izračunava tako da se NULL vrijednosti zanemaruju (ne uzimaju se u obzir pri izračunavanju)

# Agregacija - NULL vrijednosti

ispit

mbrStud	nazPred	ocjena
100	Matematika	NULL
101	Matematika	4
102	Matematika	3
103	Matematika	3
100	Fizika	NULL
101	Fizika	3

```
SELECT COUNT(mbrStud) AS broj1  
FROM ispit;
```

broj1  
6

```
SELECT COUNT(ocjena) AS broj2  
FROM ispit;
```

broj2  
4

```
SELECT COUNT(ocjena) AS broj3 FROM ispit  
WHERE mbrStud = 100;
```

broj3  
0

```
SELECT COUNT(ocjena) AS broj4 FROM ispit  
WHERE mbrStud = 200;
```

broj4  
0

```
SELECT AVG(ocjena) AS broj5 FROM ispit;
```

broj5  
3.25

```
SELECT AVG(ocjena) AS broj6 FROM ispit  
WHERE mbrStud = 100;
```

broj6  
NULL

```
SELECT AVG(ocjena) AS broj7 FROM ispit  
WHERE mbrStud = 200;
```

broj7  
NULL

# Agregatna funkcija COUNT(\*)

- Agregatna funkcija COUNT(*imeAtributa*)
  - broji n-torke u kojima vrijednost atributa *imeAtributa* nije NULL vrijednost
- Agregatna funkcija COUNT(\*)
  - broji n-torke zanemarujući njihov sadržaj

ispit

	mbrStud	nazPred	ocjena
	100	Matematika	NULL
	101	Matematika	4
	102	Matematika	3
	103	Matematika	3
	100	Fizika	NULL
	101	Fizika	3

```
SELECT COUNT(ocjena) AS brojOcj,  
       COUNT(*) AS brojRedaka  
FROM ispit;
```

brojOcj	brojRedaka
4	6

- Ne postoji agregatna funkcija COUNT(DISTINCT \*)
- ali postoji npr.: COUNT(DISTINCT ocjena)

# Grupiranje - NULL vrijednosti

- pri obavljanju operacije grupiranja, grupiranje n-torki se obavlja tako da se vodi računa o definiciji kopije n-torki
  - ako se grupiranje obavlja prema atributima iz skupa X, tada u istu grupu ulaze one n-torke čije su X-vrijednosti međusobne kopije

```
SELECT akGod, nazPred, AVG(ocjena) AS prosj
      FROM ispit
 GROUP BY akGod, nazPred;
```

ispit

	mbrStud	akGod	nazPred	ocjena
t <sub>1</sub>	100	2005	NULL	3
t <sub>2</sub>	101	NULL	NULL	5
t <sub>3</sub>	102	2005	NULL	2
t <sub>4</sub>	103	2006	Fizika	3
t <sub>5</sub>	100	NULL	NULL	5
t <sub>6</sub>	101	2006	Fizika	5
t <sub>7</sub>	102	2005	NULL	2

$$X = \{ \text{akGod}, \text{nazPred} \}$$

$t_1(X), t_3(X)$  i  $t_7(X)$  su međusobne kopije

$t_2(X)$  i  $t_5(X)$  su međusobne kopije

$t_4(X)$  i  $t_6(X)$  su međusobne kopije

akGod	nazPred	prosj
2005	NULL	2.3333
NULL	NULL	5
2006	Fizika	4

# Vanjsko spajanje - uvod

student

matBr	prez
101	Kolar
102	Horvat
103	Novak

upisanPred

matBr	nazPred
101	Matematika
101	Fizika
101	Programiranje
102	Fizika

upisani = student  $\bowtie$  upisanPred

upisani

matBr	prez	nazPred
101	Kolar	Matematika
101	Kolar	Fizika
101	Kolar	Programiranje
102	Horvat	Fizika

- n-torka **<103, Novak>** neće se pojaviti u rezultatu jer u relaciji **upisanPred** ne postoji niti jedna n-torka koja zadovoljava uvjet spajanja s tom n-torkom

# Vanjsko spajanje - uvod

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
42230	Ludbreg	NULL	
42220	Novi Marof	7	

zupanija	sifZup	nazZup
7	Varaždinska	
4	Istarska	

mjestouZupaniji = mjesto  $\bowtie$  zupanija

mjestouZupaniji	pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7	Varaždinska	
42220	Novi Marof	7	Varaždinska	

- n-torka  $<42230, \text{Ludbreg}, \text{NULL}>$  neće se pojaviti u rezultatu jer u relaciji **zupanija** ne postoji niti jedna n-torka koja zadovoljava uvjet spajanja (a ne može je niti biti, jer sifZup ima NULL vrijednost)

# Lijevo vanjsko spajanje (*Left outer join*)

- sve n-torke relacije **student** će se pojaviti u rezultatu spajanja ako se primijeni operacija **lijevog vanjskog spajanja**

student	matBr	prez
101	Kolar	
102	Horvat	
103	Novak	

upisanPred	matBrSt	nazPred
101	Matematika	
101	Fizika	
101	Programiranje	
102	Fizika	

upisano = student \* $\triangleright\triangleleft$  upisanPred

matBr = matBrSt

upisano		matBr	prez	matBrSt	nazPred
101	Kolar	101		Matematika	
101	Kolar	101		Fizika	
101	Kolar	101		Programiranje	
102	Horvat	102		Fizika	
103	Novak		NULL	NULL	NULL

- n-torkama "lijeve" relacije za koje ne postoji odgovarajuće n-torke u "desnoj" relaciji se kao vrijednosti atributa iz "desne" relacije postavljaju **NULL** vrijednosti

# Lijevo vanjsko spajanje (*Left outer join*)

mjesto	pbr	nazMjesto	sifZupMj
42000	Varaždin	7	
42230	Ludbreg	NULL	
42220	Novi Marof	7	

zupanija	sifZup	nazZup
7	Varaždinska	
4	Istarska	

mjestouZupaniji = mjesto \*▷◁ zupanija

sifZupMj = sifZup

mjestouZupaniji	pbr	nazMjesto	sifZupMj	sifZup	nazZup
42000	Varaždin	7	7	7	Varaždinska
42230	Ludbreg	NULL	NULL	NULL	NULL
42220	Novi Marof	7	7	7	Varaždinska

# SQL - Lijevo vanjsko spajanje (Left outer join)

mjesto	pbr	nazMjesto	sifZupMj
42000	Varaždin	7	
42230	Ludbreg	NULL	
42220	Novi Marof	7	

zupanija	sifZup	nazZup
7	Varaždinska	
4	Istarska	

mjesto \* $\bowtie$  zupanija  
sifZupMj = sifZup

```
SELECT mjesto.* , zupanija.* ←  
FROM mjesto LEFT OUTER JOIN zupanija  
ON sifZupMj = sifZup;
```

ili SELECT \*

pbr	nazMjesto	sifZupMj	sifZup	nazZup
42000	Varaždin	7	7	Varaždinska
42230	Ludbreg	NULL	NULL	NULL
42220	Novi Marof	7	7	Varaždinska

# Desno vanjsko spajanje (*Right outer join*)

- sve n-torce relacije **nastavnik** će se pojaviti u rezultatu spajanja ako se primjeni operacija **desnog vanjskog spajanja**

student	mbrSt	prezSt	temaSt
	101	Horvat	Tranzistori
	103	Novak	Teslini izumi
	105	Kolar	Teorija kaosa

nastavnik	sifNast	prezNast	temaNast
	202	Ban	Teslini izumi
	204	Toplek	Elektrane
	206	Oreb	Teslini izumi
	209	Pernar	Teorija kaosa

$\text{moguciMent} = \text{student} \bowtie \text{nastavnik}$   
temaSt = temaNast

- n-torkama "desne" relacije za koje ne postoji odgovarajuće n-torce u "lijevoj" relaciji se kao vrijednosti atributa iz "lijeve" relacije postavljaju NULL vrijednosti

moguciMent	mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
	103	Novak	Teslini izumi	202	Ban	Teslini izumi
	NULL	NULL	NULL	204	Toplek	Elektrane
	103	Novak	Teslini izumi	206	Oreb	Teslini izumi
	105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

# SQL - Desno vanjsko spajanje (*Right outer join*)

student

mbrSt	prezSt	temaSt
101	Horvat	Tranzistori
103	Novak	Teslini izumi
105	Kolar	Teorija kaosa

nastavnik

sifNast	prezNast	temaNast
202	Ban	Teslini izumi
204	Toplek	Elektrane
206	Oreb	Teslini izumi
209	Pernar	Teorija kaosa

student  $\triangleright\triangleleft^*$  nastavnik

temaSt = temaNast

```
SELECT student.* , nastavnik.*  
FROM student RIGHT OUTER JOIN nastavnik  
ON temaSt = temaNast;
```

ili **SELECT \***

mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
103	Novak	Teslini izumi	202	Ban	Teslini izumi
NULL	NULL	NULL	204	Toplek	Elektrane
103	Novak	Teslini izumi	206	Oreb	Teslini izumi
105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

# Puno vanjsko spajanje (*Full outer join*)

- sve n-torke iz obje relacije će se pojaviti u rezultatu spajanja ako se primjeni operacija **punog vanjskog spajanja**

student

mbrSt	prezSt	temaSt
101	Horvat	Tranzistori
103	Novak	Teslini izumi
105	Kolar	Teorija kaosa

nastavnik

sifNast	prezNast	temaNast
202	Ban	Teslini izumi
204	Toplek	Elektrane
206	Oreb	Teslini izumi
209	Pernar	Teorija kaosa

student\* $\triangleright\triangleleft$ \*nastavnik  
temaSt = temaNast

moguciMent

moguciMent	mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori	NULL	NULL	NULL
	103	Novak	Teslini izumi	202	Ban	Teslini izumi
	NULL	NULL	NULL	204	Toplek	Elektrane
	103	Novak	Teslini izumi	206	Oreb	Teslini izumi
	105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

# SQL - Puno vanjsko spajanje (*Full outer join*)

student

mbrSt	prezSt	temaSt
101	Horvat	Tranzistori
103	Novak	Teslini izumi
105	Kolar	Teorija kaosa

nastavnik

sifNast	prezNast	temaNast
202	Ban	Teslini izumi
204	Toplek	Elektrane
206	Oreb	Teslini izumi
209	Pernar	Teorija kaosa

student\* $\triangleright\triangleleft$ \*nastavnik

temaSt = temaNast

```
SELECT student.* , nastavnik.*  
      FROM student FULL OUTER JOIN nastavnik  
        ON temaSt = temaNast;
```

ili SELECT \*

# Prirodno vanjsko spajanje

---

- kod vanjskog spajanja uz uvjet i vanjskog spajanja s izjednačavanjem u shemi rezultata se pojavljuju **svi atributi obje relacije**
- kod prirodnog lijevog vanjskog spajanja iz sheme rezultata se **izbacuju istoimeni atributi desnog operanda** (jer ionako mogu poprimiti ili vrijednosti jednake vrijednostima korespondentnih atributa lijevog operanda ili NULL vrijednosti)
- kod prirodnog desnog vanjskog spajanja iz rezultata se **izbacuju istoimeni atributi lijevog operanda** (jer ionako mogu poprimiti ili vrijednosti jednake vrijednostima korespondentnih atributa desnog operanda ili NULL vrijednosti)
- kod prirodnog punog vanjskog spajanja potrebno je **u shemi rezultata zadržati sve attribute obje relacije, te primijeniti operator preimenovanja atributa**

# Prirodno vanjsko spajanje

student

mbrSt	prezSt	tema
101	Horvat	Tranzistori
103	Novak	Teslini izumi
105	Kolar	Teorija kaosa

nastavnik

sifNast	prezNast	tema
202	Ban	Teslini izumi
204	Toplek	Elektrane
206	Oreb	Teslini izumi
209	Pernar	Teorija kaosa

student\*▷◁nastavnik

mbrSt	prezSt	tema	sifNast	prezNast
101	Horvat	Tranzistori	NULL	NULL
103	Novak	Teslini izumi	202	Ban
103	Novak	Teslini izumi	206	Oreb
105	Kolar	Teorija kaosa	209	Pernar

student▷◁\*nastavnik

mbrSt	prezSt	sifNast	prezNast	tema
103	Novak	202	Ban	Teslini izumi
NULL	NULL	204	Toplek	Elektrane
103	Novak	206	Oreb	Teslini izumi
105	Kolar	209	Pernar	Teorija kaosa

# Puno prirodno vanjsko spajanje

- to je u stvari **puno vanjsko spajanje s izjednačavanjem** (s preimenovanjem atributa)

student

mbrSt	prezSt	tema
101	Horvat	Tranzistori
103	Novak	Teslini izumi
105	Kolar	Teorija kaosa

nastavnik

sifNast	prezNast	tema
202	Ban	Teslini izumi
204	Toplek	Elektrane
206	Oreb	Teslini izumi
209	Pernar	Teorija kaosa

$\text{student}^* \triangleright \triangleleft^* \rho_{\text{nastavnik}(\text{sifNast}, \text{prezNast}, \text{temaNast})} (\text{nastavnik})$

tema = temaNast

mbrSt	prezSt	tema	sifNast	prezNast	temaNast
101	Horvat	Tranzistori	NULL	NULL	NULL
103	Novak	Teslini izumi	202	Ban	Teslini izumi
NULL	NULL	NULL	204	Toplek	Elektrane
103	Novak	Teslini izumi	206	Oreb	Teslini izumi
105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

## Baze podataka

# Predavanja

## 4. SQL (1. dio)

Ožujak, 2020.



# SQL - Uvod

---

- objedinjuje funkcije jezika za definiciju podataka (DDL) i jezika za rukovanje podacima (DML)
- razvoj započeo 70-tih godina
  - IBM San José Research Laboratory (California, USA)
- *Structured Query Language* je standardni jezik relacijskih baza podataka (*database language*)
  - 1986. godine - SQL-86 ili SQL1 (prva verzija standarda)
  - 1992. godine - SQL-92 ili SQL2
  - 1999. godine - SQL:1999
  - 2003. godine - SQL:2003
  - 2008. godine - SQL:2008
  - 2011. godine - SQL:2011
  - 2016. godine – SQL:2016
- proizvođači komercijalnih sustava često ugrađuju i svoje nestandardne DDL i DML naredbe
  - programski kod postaje neprenosiv između različitih SQL sustava
  - otežava se usaglašavanje oko budućih standarda.

# SQL - Uvod

- neproceduralnost - naredbom je dovoljno opisati što se želi dobiti kao rezultat - nije potrebno definirati kako do tog rezultata doći
- u SUBP ugrađeni optimizator upita pronađi najefikasniji način obavljanja upita

zupanija	
sifZup	nazZup
2	Primorsko-goranska
7	Varaždinska
4	Istarska

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
51000	Rijeka	2
52100	Pula	4
42230	Ludbreg	7

- ispisati podatke o mjestima u Varaždinskoj županiji. Rezultate poredati prema nazivu mjesta

```
SELECT mjesto.* FROM mjesto, zupanija
WHERE mjesto.sifZup = zupanija.sifZup
AND nazZup = 'Varaždinska'
ORDER BY nazMjesto;
```

# SQL - Vrste objekata

---

- Baza podataka *Database*
- Relacija (tablica) *Table*
- Atribut (stupac, kolona) *Column*
- Virtualna tablica (pogled) *View*
- Integritetsko ograničenje *Constraint*
- Indeks *Index*
- Pohranjena procedura *Stored Procedure*
- Okidač *Trigger*

# SQL - Identifikatori

---

- Identifikatori (imena objekata) se formiraju iz slova, znaka '\_' i znamenki. Prvi znak od ukupno 128 značajnih (signifikantnih) znakova mora biti slovo ili znak '\_'
- ispravno formirani identifikatori

`stud`

`ispiti2000godine`

`stud_ispit`

`_1mjesec`

- neispravno formirani identifikatori

`_11.mjesec`

`11mjesec`

`stud-ispit`

# SQL - Rezervirane riječi

- SQL je "neosjetljiv" (*case insensitive*) na razliku između velikih i malih slova kada su u pitanju rezervirane riječi (SELECT, UPDATE, DELETE, FROM, WHERE, ...) i identifikatori

```
SELECT * FROM mjesto  
WHERE sifZupanija = 7
```

≡

```
select * FrOm MJesto  
wHERE SIFZupanIJA = 7
```

- Međutim, razlika između velikih i malih slova **postoji** kad su u pitanju nizovi znakova

'Ivan' ≠ 'IVAN'

# SQL - Format naredbi

- SQL je jezik slobodnog formata naredbi (jednako kao C)

```
SELECT * FROM mjesto  
WHERE sifZupanija = 7
```

=

```
SELECT  
*  
FROM  
mjesto  
WHERE  
sifZupanija = 7
```

# SQL - Korištenje komentara

- "blok komentari" (jednako kao u programskom jeziku C)
  - dio teksta omeđen oznakama /\* i \*/

```
/* ovo je komentar koji se  
proteže kroz više redaka teksta */
```

- "linijski komentari"
  - mjesto u retku na kojem se nalaze znakovi -- predstavlja početak komentara koji se proteže do kraja retka

```
-- ovo je komentar  
SELECT * FROM mjesto -- ovo je komentar  
      WHERE pbr = 10000 -- ovo je komentar
```

# SQL - Tipovi podataka

## ▪ INTEGER

- cijeli broj pohranjen u 4 bajta u aritmetici dvojnog komplementa. Dopušteni raspon brojeva određen je intervalom

$$[ -2^{n-1}, 2^{n-1} - 1 ] \quad n=32$$

- dakle, raspon brojeva je:

$$[-2147483648, 2147483647]$$

Konstante:

5	-30000	0	1765723712	NULL
---	--------	---	------------	------

## ▪ SMALLINT

- cijeli broj pohranjen u 2 bajta. Raspon brojeva koji se mogu prikazati je [-32768, 32767]

Konstante:

5	-30000	0	NULL
---	--------	---	------

# SQL - Tipovi podataka

- **CHAR(m)**
  - znakovni niz (*string*) s definiranom duljinom. Npr: CHAR(24).

Konstante:

```
'Ana'      '12345'      NULL  
'Dvostruki navodnik " unutar niza'  
'Jednostruki navodnik '' unutar niza'
```

- pri unosu niza čija je duljina <m, preostala mjesta se pune prazninama (prateće praznine)

- **VARCHAR(m)**
  - znakovni niz (*string*) varijabilne duljine, s unaprijed definiranom maksimalnom duljinom. Npr: VARCHAR(24).
    - nema pratećih praznina
- **uočite:** koriste se **jednostruki** navodnici (drugačije nego u jeziku C)

# SQL - Tipovi podataka

---

- **NCHAR(m)**
  - jednako kao i CHAR tip podatka, ali omogućava ispravno leksikografsko uređenje nizova znakova koji sadrže znakove iz nacionalnih kodnih stranica (*character set*). Koristi se onda kada se predviđa potreba za leksikografskim poretkom nizova znakova u kojima se pojavljuju specifični nacionalni znakovi (Č, Ć, Š, Đ, Ž, ...), npr. za atribut prezime
- **NVARCHAR(m)**
  - kao NCHAR tip podatka ali varijabilne duljine
  - NCHAR i NVARCHAR nisu podržani u PostgreSQL
  - Zastarjeli koncept – danas tipovi CHAR i VARCHAR omogućavaju korištenje nacionalnih kodnih stranica

# SQL - Tipovi podataka

## ▪ REAL

- odgovara tipu podatka `float` u jeziku C (IEEE-754 format prikaza - jednostruka preciznost)

Konstante:

23    -343.23    232.233E3    23.0e-24    NULL

## ▪ DOUBLE PRECISION

- odgovara tipu podatka `double` u jeziku C (IEEE-754 format prikaza - dvostruka preciznost)

Konstante:

23    -343.23    232.233E3    23.0e-302    NULL

# SQL - Tipovi podataka

- **NUMERIC(m, n) ili DECIMAL(m, n)**
  - ekvivalentni tipovi u PostgreSQL-u
  - **m** - ukupni broj znamenki (*precision*)
  - **n** - broj znamenki iza decimalne točke (*scale*,  $n \leq m$ )
  - npr., NUMERIC(15, 3) predstavlja decimalni broj s ukupno najviše 15 znamenki, od toga se najviše 3 znamenke nalaze iza decimalne točke
  - razlikuje se od **float** ili **double** tipa podatka u jeziku C
    - ako se za pohranu broja 1.3 koristi tip podatka NUMERIC(2,1), broj će biti pohranjen **bez numeričke pogreške**
    - ako se za pohranu broja 1.3 koristi tip podatka **float** u jeziku C, u memoriji će se zapravo pohraniti broj 1.2999999523162842 (num. pogreška zbog karakteristika IEEE-754 formata pohrane)

Konstante - primjer za NUMERIC(7, 2):

5	8.1	-12345.67	0	NULL
---	-----	-----------	---	------

# SQL - Tipovi podataka

## ▪ DATE

- podaci ovog tipa se uvijek prikazuju u obliku datuma (npr. 13.12.2019). Ovaj tip podatka omogućava korištenje sljedećih operacija zbrajanja i oduzimanja:
  - **dat1 - dat2** rezultat je podatak tipa INTEGER - broj dana proteklih između *dat2* i *dat1*
  - **dat + cijeliBroj** rezultat je podatak tipa DATE - izračunava koji datum je *cijeliBroj* dana nakon dana *dat*
  - **dat - cijeliBroj** rezultat je podatak tipa DATE - izračunava koji datum je *cijeliBroj* dana prije dana *dat*

Konstante:

'17.2.2017'

'16.07.1969'

NULL

# SQL - Tipovi podataka

Ostali vremenski tipovi podataka - rezolucija  $p \leq 6$  (mikrosekunda)

- **TIMESTAMP [(p)]** – vremenski trenutak = datum + vrijeme

Primjer: '13.03.2017 14:24:54' '13.03.2017 14:24:54.678901'

- **TIME [(p)]** – vrijeme – sati, minute, sekunde na najviše 6 decimala

Primjer: '14:24:54' '14:24:54.678901'

- **INTERVAL [fields] [(p)]** – interval – mogući rasponi: od godina do sekunda na najviše 6 decimala

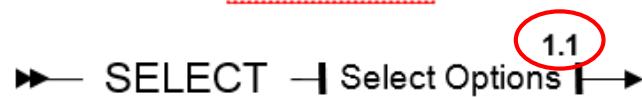
fields - year, month, day, hour, minute, second, year to month, day to hour, day to minute, day to second, hour to minute, hour to second, minute to second

Ako definicija sadrži i **fields** i **p** - **fields** mora sadržavati sekunde, jer se **p** primjenjuje na sekunde.

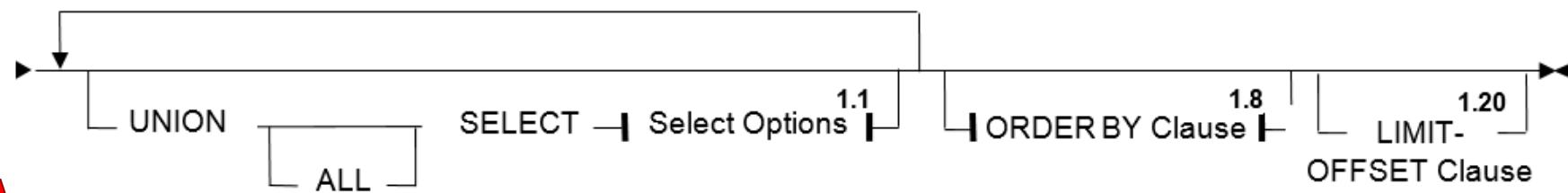
Primjer: '5 YEAR 2 MONTH 1 DAY' – 5 godina 2 mjeseca i 1 dan  
'2 14:24:54' – 2 dana 14 sati 24 minute i 54 sekunde  
'00:4:54' – 4 minute i 54 sekunde

# ***SELECT Statement***

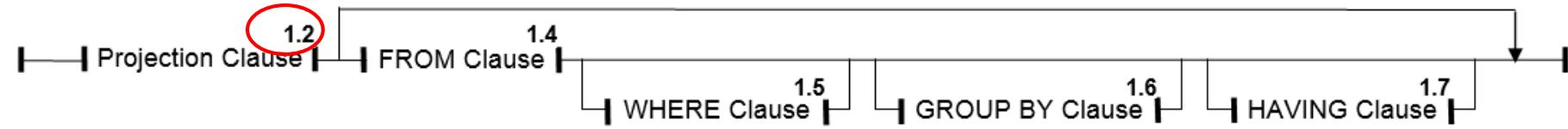
## **1. SELECT Statement**



▪ Sintaksni dijagrami



### **1.1. SELECT Options**



### **1.2. Projection Clause**



# Projection Clause

- Primjeri:

student

matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
105	Kolar	52000
107	Ban	10000

```
SELECT ALL prez  
      , postbr  
FROM student;
```

≡

```
SELECT prez  
      , postbr  
FROM student;
```

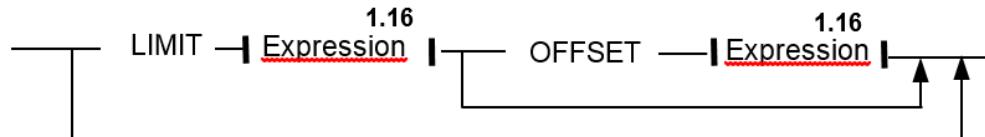
prez	postBr
Kolar	52000
Horvat	10000
Kolar	52000
Ban	10000

```
SELECT DISTINCT prez  
      , postbr  
FROM student;
```

prez	postBr
Kolar	52000
Horvat	10000
Ban	10000

# LIMIT- OFFSET Clause

## 1.20 LIMIT-OFFSET Clause



### Primjeri:

```
SELECT *  
FROM student  
LIMIT 2;
```

matBr	prez	postBr
102	Horvat	10000
105	Kolar	52000

student

matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
105	Kolar	52000
107	Ban	10000

Ne zna se koje dvije n-torce  
će se dobiti kao "prve dvije" -  
poredak n-torki u relaciji (niti u  
SQL tablici) nije definiran

```
SELECT DISTINCT prez  
FROM student  
LIMIT 2 OFFSET 1;
```

prez
Horvat
Ban

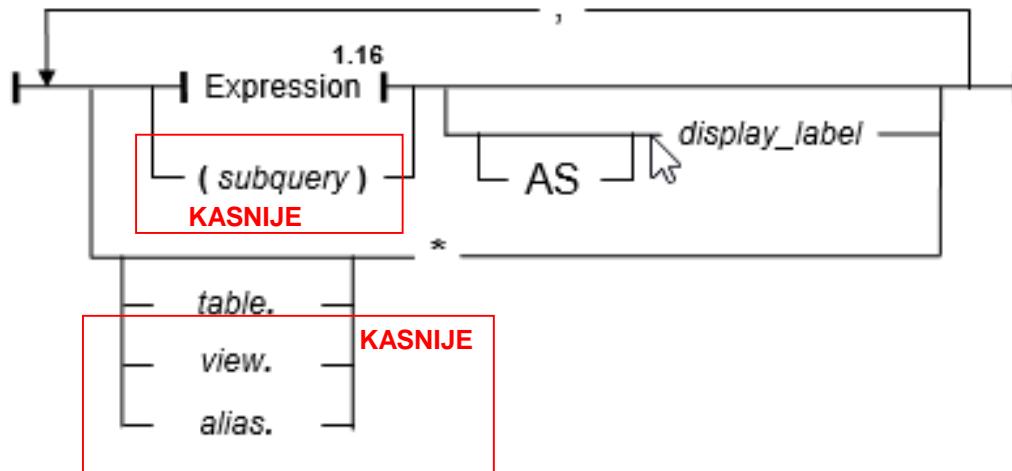
Ne zna se koje su to "prve  
dvije" n-torce nakon jedne  
preskočene

```
SELECT *  
FROM student  
LIMIT 100;
```

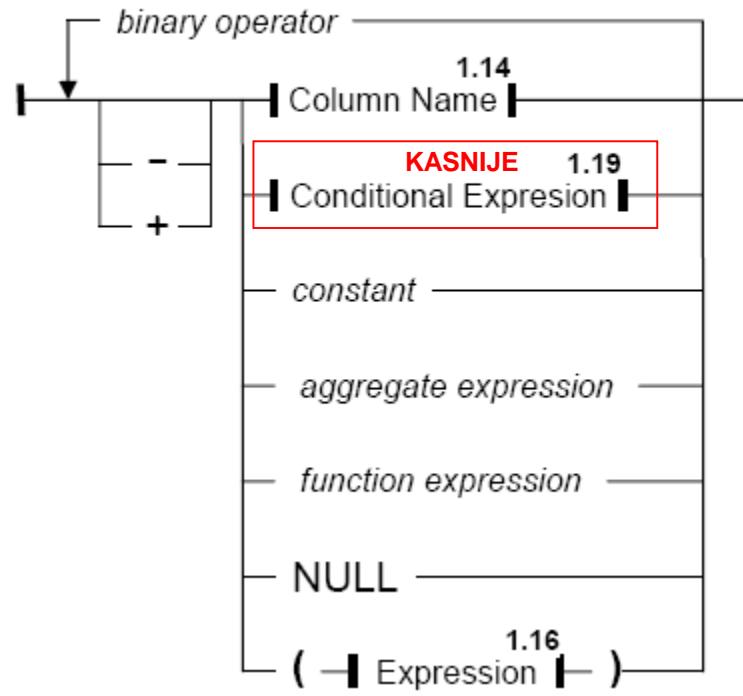
matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
105	Kolar	52000
107	Ban	10000

# SELECT List

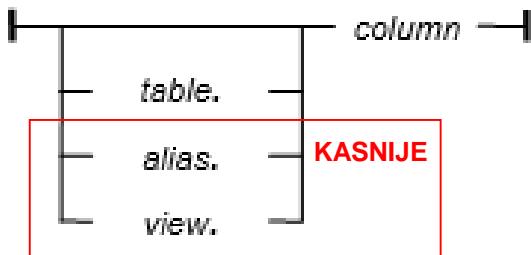
## 1.3 SELECT List



## 1.16. Expression



## 1.14. Column Name



# SELECT List

- Primjer:

mjesto

pbr	nazMjesto	sifZup
42000	Varaždin	7
52100	Pula	4

```
SELECT mjesto.pbr, *, pbr, mjesto.*  
FROM mjesto
```

pbr	pbr	nazMjesto	sifZup	pbr	pbr	nazMjesto	sifZup
42000	42000	Varaždin	7	42000	42000	Varaždin	7
52100	52100	Pula	4	52100	52100	Pula	4

U ovom primjeru rezultat nije relacija!

# Izraz (*Expression*)

## 1.16. Expression

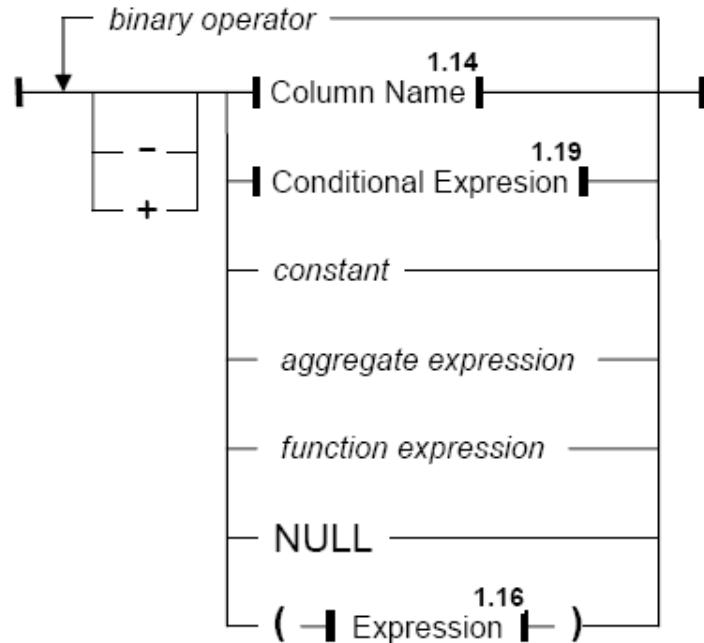
- Unarni operatori

+      -

- Binarni operatori

+      -      \*      /

|| ulančavanje nizova znakova  
(nadovezivanje, konkatenacija)



- Redoslijed obavljanja operacija u složenim izrazima određuje se prema istim pravilima kao u programskom jeziku C (implicitni redoslijed obavljanja se može promijeniti upotrebom okruglih zagrada)
- Konverzija tipova podataka tijekom evaluacije izraza obavlja se prema sličnim pravilima kao u programskom jeziku C

# Izraz (primjeri)

- unarni, binarni operatori i konstante

```
CREATE TABLE bodovi (
    mbr      INTEGER
,  ime      VARCHAR(10)
,  prez     VARCHAR(10)
,  bodLab   INTEGER
,  bodMI    NUMERIC(4,1));
```

```
SELECT mbr,
       bodLab + bodMI,
       (bodLab + bodMI) / 100
  FROM bodovi;
```

```
SELECT mbr, - bodLab
  FROM bodovi;
```

```
SELECT mbr, ime || prez
  FROM bodovi;
```

```
SELECT mbr || '-' || ime ||
          ' ' || prez
  FROM bodovi;
```

bodovi

mbr	ime	prez	bodLab	bodMI
100	Ana	Novak	12	67.2
107	Ivo	Ban	17	54.3

expression

mbr	?column?	?column?
100	79.2	0.792
107	71.3	0.713

mbr	?column?
100	-12
107	-17

mbr	?column?
100	AnaNovak
107	IvoBan

?column?
100-Ana Novak
107-Ivo Ban

# Eksplicitna pretvorba tipova podataka

- Dvije ekvivalentne sintakse:  
CAST ( expression AS type )  
expression::type

```
CREATE TABLE skSati (
    rbr          INTEGER
    , skSatOd   TIMESTAMP
    , skSatDo   TIMESTAMP);
```

skSati

rbr	skSatOd	skSatDo
1	14.03.2017 08:15:00.012345	14.03.2017 09:00:00.012345
2	14.03.2017 09:15:00.012345	14.03.2017 10:00:00.012345

```
SELECT CAST (skSatOd AS TIME(0)) vrijeme
      , skSatOd::TIMESTAMP(0)      datumIVrijeme
      FROM skSati
```

vrijeme	datumIVrijeme
08:15:00	14.03.2017 08:15:00
09:15:00	14.03.2017 09:15:00

# Funkcije (*function expression*)

---

- ABS
- MOD
- ROUND
- SUBSTRING
- UPPER
- LOWER
- TRIM
- CHAR\_LENGTH
- OCTET\_LENGTH
- EXTRACT
- CURRENT\_DATE
- CURRENT\_TIME
- CURRENT\_TIMESTAMP
- CURRENT\_USER

# Funkcije (*function expression*)

- **ABS (*num\_expression*)**

- računa absolutnu vrijednost izraza

*num\_expression* – mora biti numerički tip podatka (INTEGER, NUMERIC, FLOAT, ...)

*rezultat funkcije* – tip podatka ovisi o tipu podatka ulaznog argumenta

- **MOD (*dividend, divisor*)**

- računa ostatak dijeljenja djeljenika i djelitelja (djelitelj ne smije biti 0)
  - argumenti ne moraju biti cijelobrojni

*dividend (djeljenik)* – numerički tip podatka (INTEGER, DECIMAL, NUMERIC, FLOAT, ...)

*divisor (djelitelj)* – numerički tip podatka (INTEGER, DECIMAL, NUMERIC, FLOAT, ...)

*rezultat funkcije* – tip podatka ovisi o tipu podatka ulaznih argumenta

# Funkcije (*function expression*)

- **ROUND (*expression[, rounding\_factor]*)**
  - zaokružuje vrijednost izraza (*expression*)
  - ako se ne navede *rounding\_factor*, uzima se da je njegova vrijednost 0

***expression*** (*izraz koji se zaokružuje*) –

  numerički tip podatka (INTEGER, NUMERIC, FLOAT, ...)

***rounding\_factor*** (*preciznost na koju se vrši zaokruživanje*) – cijelobrojni tip podatka

***rezultat funkcije*** – tip podatka ovisi o tipu podatka ulaznog argumenta (*expression*) i o *rounding\_factor*

# Funkcije (*function expression*)

## ▪ **SUBSTRING (*source\_string FROM start\_position [FOR length]*)**

- vraća podniz zadanog niza
- ako se *length* ne navede vraća se podniz koji počinje na *start\_position*, a završava gdje i niz *source\_string*

***source\_string*** – zadani niz čiji se podniz traži funkcijom  
mora biti izraz tipa niza znakova

***start\_position*** – broj koji predstavlja poziciju prvog znaka podniza u zadanim nizu  
*source\_string*;  
mora biti izraz cijelobrojnog tipa

***length(duljina)*** – broj znakova koje funkcija treba vratiti počevši od *start\_position*;  
mora biti izraz cijelobrojnog tipa

# Funkcije (*function expression*)

---

- **UPPER (expression)**

- sva mala slova (a-z) koja se pojavljuju u zadanom nizu *expression* zamjenjuje odgovarajućim velikim slovima (A-Z)

- **LOWER (expression)**

- sva velika slova (A-Z) koja se pojavljuju u zadanom nizu *expression* zamjenjuje odgovarajućim malim slovima (a-z)

**expression** – zadani niz nad kojim se vrši pretvorba slova  
mora biti izraz tipa niza znakova

# Funkcije (*function expression*)

- **TRIM (*expression*)**
  - funkcija vraća niz znakova koji nastaje tako da se s početka i kraja niza *expression* izbace sve praznine
- **CHAR\_LENGTH (*expression*)**
  - funkcija vraća broj znakova u zadanom nizu *expression* ignorira prateće (*trailing*) praznine
- **OCTET\_LENGTH (*expression*)**
  - funkcija vraća broj byte-ova zadatog niza *expression* uključujući i prateće praznine

***expression*** – mora biti izraz tipa niza znakova

# Funkcije (*function expression*)

---

- **CURRENT\_USER**
  - funkcija vraća *login* korisnika koji je trenutno prijavljen za rad sa bazom podataka
- **CURRENT\_DATE**
  - funkcija vraća podatak tipa DATE koji predstavlja današnji datum (dobiven iz operacijskog sustava)
- **CURRENT\_TIME**
  - funkcija vraća podatak tipa TIME **s vremenskom zonom** koji predstavlja trenutno vrijeme. Podatak je oblika:  
hh:mm:ss.xxxxxx – npr. 13:36:56.225091+01:00
- **CURRENT\_TIMESTAMP**
  - funkcija vraća podatak tipa TIMESTAMP koji predstavlja (*current*) datum i vrijeme **s vremenskom zonom**. Podatak je oblika:
  - DD.MM.YYYY HH:MI:SS.xxxxxx npr. "2018-03-15 13:39:19.89626+01"

# Funkcije (*function expression*)

- **EXTRACT (field FROM source)**
  - funkcija vraća  
redni broj godine, mjeseca, dana, tjedna **za datum** sadržan u **source**  
sata, minute, sekunde u danu **za vrijeme** sadržano u **source**
- Za **field**
  - = **dow**
    - funkcija vraća redni broj dana u tjednu za zadani datum  
(0 – nedjelja, 1 – ponedjeljak, 2 – utorak, itd...)
  - = **doy**
    - funkcija vraća redni broj dana u godini za zadani datum

**field** – *year, month, day, hour, minute, second, week  
dow, doy,*

**source** – izraz tipa TIMESTAMP ili INTERVAL

# Funkcije (primjeri) – matematičke funkcije

```
CREATE TABLE upl_ispl (
    rbr      INTEGER
, racun   INTEGER
, datum   DATE
, iznos   NUMERIC(9,2));
```

```
SELECT rbr, ABS(iznos)
  FROM upl_ispl;
```

```
SELECT rbr, ROUND(iznos, 1)
  FROM upl_ispl;
```

```
SELECT rbr, MOD(iznos, 10)
  FROM upl_ispl;
```

upl\_ispl

rbr	racun	datum	iznos
1	123456	22.02.2007	-120.00
2	878341	23.02.2007	173.47

rbr	?column?
1	120.00
2	173.47

Ispisuje apsolutne vrijednosti iznosa

rbr	?column?
1	-120.0
2	173.5

Ispisuje iznose zaokružene na jednu decimalu

rbr	?column?
1	0
2	3.47

Ispisuje ostatak dijeljenja iznosa sa 10

# Funkcije (primjeri) – funkcije s nizovima

```
CREATE TABLE student (
    jmbag  VARCHAR(10)
,  ime    VARCHAR(25)
,  prezime VARCHAR(25));

```

student

jmbag	ime	prezime
0036368145	Tomislav	Babić
0036369296	Linda	Jurić

Ispisati jmbag i inicijale studenata

```
SELECT jmbag
,  SUBSTRING(ime FROM 1 FOR 1) || '.' || 
    SUBSTRING(prezime FROM 1 FOR 1) || '.'
FROM student;
```

jmbag	?column?
0036368145	T . B.
0036369296	L . J.

Ispisati imena velikim slovima, a prezimena malim slovima

```
SELECT UPPER(ime)
,  LOWER(prezime)
FROM student;
```

?column?	?column?
TOMISLAV	babić
LINDA	jurić

# Funkcije (primjeri) – funkcije s nizovima

```
CREATE TABLE student (
    jmbag CHAR(10)
, ime CHAR(10)
, prezime CHAR(10));
```

student

jmbag	ime	prezime
0036368145	Tomislav	Ban
0036369296	Linda	Kekez

Ispisati imena i prezimena studenata iz kojih su izbačene praznine

```
SELECT ime, prezime
, TRIM(ime) ||
TRIM(prezime)
, ime ||
prezime
FROM student;
```

ime	prezime	?column?	?column?
Tomislav○	Ban○	TomislavBan	TomislavBan
Linda○	Kekez○	LindaKekez	LindaKekez

Neočekivano ponašanje

```
SELECT TRIM(ime) || ' ' || TRIM(prezime) AS imeiprezime
FROM student;
```

imeiprezime
Tomislav Ban
Linda Kekez

Ispisati korisničko ime i broj znakova koji ga čine

```
SELECT CURRENT_USER
, CHAR_LENGTH(CURRENT_USER) ;
```

?column?	?column?
postgres	8

S obzirom da neke operacije nad tipom podataka CHAR() daju neočekivani rezultat, u svim primjerima i zadatcima za nizove znakova koristit će se isključivo VARCHAR() tip podatka.

# Funkcije (primjeri) – funkcije s nizovima

```
CREATE TABLE student (
    jmbag  VARCHAR(10)
,  ime    VARCHAR(25)
,  prezime VARCHAR(25)) ;
```

student

jmbag	ime	prezime
0036368145	Tomislav	Božanić
0036369296	Linda	Kekez

Ispisati broj znakova u prezimenu i broj znakova u prezimenu nad kojim je primijenjena TRIM funkcija.

S obzirom da se kod tipa podatka VARCHAR ne dodaju prateće praznine, funkcija CHAR\_LENGTH će za originalni niz znakova i niz iz kojeg su primjenom TRIM funkcije izbačene prateće praznine dati jednak rezultat.

```
SELECT CHAR_LENGTH(prezime)
      , CHAR_LENGTH(TRIM(prezime))
  FROM student;
```

?column?	?column?
7	7
5	5

Ispisuje broj znakova i broj okteta (bajtova) koji čine prezime

```
SELECT CHAR_LENGTH(prezime) AS znakova
      , OCTET_LENGTH(prezime) AS okteta
  FROM student;
```

znakova	okteta
7	9
5	5

Zbog utf8:  
ž-2 oktet  
ć-2 oktet

# Operacije s vremenskim trenutcima i intervalima (prema SQL standardu)

Operand 1	Operator	Operand 2	Tip rezultata
Datetime	-	Datetime	Interval**
Datetime	+ ili -	Interval	Datetime
Interval	+	Datetime	Datetime
Interval	+ ili -	Interval	Interval
Interval	* ili /	Numeric	Interval
Numeric	*	Interval	Interval

Napomena:

Naziv Datetime predstavlja tipove podataka DATE, TIME i TIMESTAMP.

\*\*Posebno za DATE tip podatka u PostgreSQL dobijemo rezultat je tipa INTEGER prepostavljamo zbog kompatibilnosti prema starijim verzijama

# Aritmetički Date/Time Operatori - PostgreSQL

Operacije	Dozvoljeni operatori ( $\alpha$ )	Tip rezultata
DATE $\alpha$ DATE	-	INTEGER
DATE $\alpha$ TIME	+	TIMESTAMP
DATE $\alpha$ TIMETZ	+	TIMESTAMP WITH TIMEZONE
DATE $\alpha$ INT4	+ -	DATE
TIME $\alpha$ DATE	+	TIMESTAMP
TIME $\alpha$ INTERVAL	+ -	TIME
TIMETZ $\alpha$ DATE	+	TIMESTAMP WITH TIMEZONE
TIMETZ $\alpha$ INTERVAL	+ -	TIMETZ
TIMESTAMP $\alpha$ TIMESTAMP	-	INTERVAL
TIMESTAMP $\alpha$ INTERVAL	+ -	TIMESTAMP
INTERVAL $\alpha$ TIME	+	TIME

<http://etutorials.org/SQL/Postgresql/Part+I+General+PostgreSQL+Use/Chapter+2.+Working+with+Data+in+PostgreSQL/DateTime+Values/>

# Temporalni tipovi podataka s vremenskom zonom

- TIME WITH TIME ZONE [(p)] ili TIMETZ [(p)]
- TIMESTAMP WITH TIME ZONE [(p)] ili TIMESTAMPTZ [(p)]

Primjeri: '19:28:50.328304+01:00' '20.03.2019 19:28:50.328304 CET'

U ovom kolegiju se nećemo baviti temporalnim podatcima s vremenskom zonom.

Rezultate funkcija CURRENT\_TIME i CURRENT\_TIMESTAMP, čiji rezultat uključuje vremensku zonu, svest ćemo na odgovarajuće tipove bez vremenske zone:

- CURRENT\_TIME ::TIME(x),
- CURRENT\_TIMESTAMP ::TIMESTAMP(x)

# Funkcije (primjeri) – funkcije s datumom

```
CREATE TABLE nastavnik (
    sifNastavnik      INTEGER
, datumZaposlenOd   DATE
, datumZaposlenDo   DATE);
```

nastavnik

sifNastavnik	datumZaposlenOd	datumZaposlenDo
1	22.01.2000	28.02.2019
2	01.06.2010	25.03.2019

Napomena: pretpostavka je da je sljedeći upit izведен dana 20.03.2019.

Broj dana koji je protekao nakon prestanka zaposlenja nastavnika

```
SELECT sifNastavnik
      , CURRENT_DATE - datumZaposlenDo
    FROM nastavnik;
```

sifNastavnik	?column?
1	20
2	-5

Ispisuje dan, mjesec i godinu zaposlenja nastavnika, dan u tjednu i dan u godini

```
SELECT EXTRACT(DAY      FROM datumZaposlenOd)  d
      , EXTRACT(MONTH    FROM datumZaposlenOd)  m
      , EXTRACT(YEAR    FROM datumZaposlenOd)  g
      , EXTRACT(DOW     FROM datumZaposlenOd)  dow
      , EXTRACT(DOY     FROM datumZaposlenOd)  doy
    FROM nastavnik;
```

d	m	g	dow	doy
22	1	2000	6	22
1	6	2010	2	152

# Funkcije (primjeri) – funkcije s datumom

```
CREATE TABLE nastavnik (
    sifNastavnik      INTEGER
, datumZaposlenOd   DATE
, datumZaposlenDo    DATE);
```

nastavnik

sifNastavnik	datumZaposlenOd	datumZaposlenDo
1	22.01.2000	28.02.2019
2	01.06.2010	25.03.2019

Ispisati datum koji odgovara sljedećem danu nakon prestanka zaposlenja nastavnika

```
SELECT EXTRACT(DAY FROM datumZaposlenDo)+1 || '.' ||
       EXTRACT(MONTH FROM datumZaposlenDo) || '.' ||
       EXTRACT(YEAR FROM datumZaposlenDo)
  FROM nastavnik
```

?column?  
29.02.2019  
26.03.2019



2019. nije prijestupna!

```
SELECT (EXTRACT(DAY FROM datumZaposlenDo)+1 || '.' ||
        EXTRACT(MONTH FROM datumZaposlenDo) || '.' ||
        EXTRACT(YEAR FROM datumZaposlenDo))::DATE
  FROM nastavnik
```

ERROR: date/time field value out of range: "29.2.2019"

Ispravno rješenje:

```
SELECT datumZaposlenDo+1
  FROM nastavnik;
```

?column?  
01.03.2019  
26.03.2019

# Primjeri ispisa datuma

```
SELECT CURRENT_DATE,  
       '18.3.2019',  
       '18.3.2019'::DATE,  
       '18.3.2019'::TIMESTAMP;
```

	current_date date	?column? text	date	timestamp timestamp without time zone
1	18.03.2019	18.3.2019	18.03.2019	18.03.2019 00:00:00

# Trenutno vrijeme – CURRENT\_TIME

---

- Funkcija CURRENT\_TIME vraća vrijeme (dobiveno iz operacijskog sustava) u mikrosekundama, s vremenskom zonom, npr. 20:03:46.286634+01:00
- Primjeri:
  - trenutno vrijeme u mikrosekundama bez vremenske zone:  
CURRENT\_TIME :: TIME(6) 20:03:46.286634
  - trenutno vrijeme u sekundama bez vremenske zone:  
CURRENT\_TIME :: TIME(0) 20:03:46
  - 2 sata i 10 minuta nakon trenutnog vremena  
CURRENT\_TIME + '2 hours 10 min':: INTERVAL 22:13:46.286634+01:00
  - 2 sata i 10 minuta nakon trenutnog vremena – u sekundama, bez vremenske zone:  
CURRENT\_TIME ::TIME(0) + '2 hours 10 min':: INTERVAL 22:13:46

Analogno vrijedi i za funkciju CURRENT\_TIMESTAMP

# Funkcije (primjeri) – funkcije s vremenom i vremenskim trenutkom

```
CREATE TABLE predavanje (
    sifPredmet      INTEGER
    , oznGrupa      VARCHAR(10)
    , predavanjeOd   TIMESTAMP
    , predavanjeDo    TIMESTAMP);
```

predavanje

sifPredmet	oznGrupa	predavanjeOd	predavanjeDo
1	P02	26.03.2019 12:15:00	26.03.2019 14:00:00
2	P05	26.03.2019 16:15:00	26.03.2019 17:00:00

Ispisuje vremenski trenutak (TIMESTAMP) i vrijeme (TIME) u trenutku izvođenja upita, vrijeme koje će proteći od trenutka izvođenja upita do početka predavanja i trajanje predavanja.

```
SELECT CURRENT_TIMESTAMP::TIMESTAMP(0)
       CURRENT_TIME::TIME(0)
       predavanjeOd - CURRENT_TIMESTAMP::TIMESTAMP(0)
       predavanjeDo - predavanjeOd
FROM predavanje
```

sada,  
vrijemeSada,  
doPocetka,  
trajanje

sada	vrijemeSada	doPocetka	trajanje
18.03.2019 18:45:06	18:45:06	7 days 17:29:54	01:45:00
18.03.2019 18:45:06	18:45:06	7 days 21:29:54	00:45:00

# Funkcije (primjeri) – funkcije s datumom

Ispisuje vremenski trenutak star godinu dana

```
SELECT (EXTRACT(DAY FROM CURRENT_DATE) || '.' ||  
        EXTRACT(MONTH FROM CURRENT_DATE) || '.' ||  
        (EXTRACT(YEAR FROM CURRENT_DATE)-1) || ' ' ||  
        CURRENT_TIME) ::TIMESTAMP(0) AS vrijemeStarogodinudana
```

vrijemestarogodinudana
20.03.2018 18:48:56

## Loše rješenje

Napomena: pretpostavka je da je gornji upit izведен dana 20.03.2019.

## Bolje rješenje prethodnog problema

Ispisuje vremenski trenutak star godinu dana

```
SELECT CURRENT_TIMESTAMP::TIMESTAMP(0) - '1 YEAR'::interval  
      AS prijeGodinuDana;
```

prijeGodinuDana
timestamp without time zone
20.03.2018 18:48:56

Ispisuje vremenski trenutak za deset mjeseci i tri tjedna udaljen od sadašnjeg.

```
SELECT CURRENT_TIMESTAMP::TIMESTAMP(0)  
      + '10 MONTHS 3 WEEKS'::interval  
      AS za10MjTriTjedna;
```

za10mjtritjedna
timestamp without time zone
10.02.2020 18:49:27

Napomena: pretpostavka je da su gornji upiti izvedeni dana 20.03.2019.

# Primjeri s intervalima

Interval se unosi kao: **quantity unit [quantity unit...] [direction]**

**quantity** je broj,

**unit**: *microsecond, millisecond, second, minute, hour, day, week, month, year, decade, century, millennium*;

**direction** može biti ago, ili može biti ispušten.

Dani, sati, minute i sekunde mogu se navesti bez jedinica.

```
CREATE TABLE intervali(
    interval1 INTERVAL,
    interval2 INTERVAL YEAR TO MONTH,
    interval3 INTERVAL DAY TO SECOND,
    interval4 INTERVAL HOUR TO SECOND);
INSERT INTO intervali VALUES
('5 years 2 months 1 day', '3 years 5 months', '2 14:24:54', '0:4:54');
SELECT * FROM intervali;
```

	interval1 interval	interval2 interval	interval3 interval	interval4 interval
1	5 years 2 mons 1 day	3 years 5 mons	2 days 14:24:54	00:04:54

# Funkcije (primjeri) – funkcije s vremenskim trenutkom i intervalom

```
CREATE TABLE predavanje (
    sifPredmet      INTEGER
,   oznGrupa        VARCHAR(10)
,   datum           DATE
,   vrijemePoc      TIME
,   trajanje        INTERVAL);
```

predavanje

sifPredmet	oznGrupa	datum	vrijemePoc	trajanje
1	P02	26.03.2019	12:15:00	00:45:00
1	P02	26.03.2019	13:15:00	00:45:00

Ispisuje vrijeme početka (TIME) i vremenski trenutak (TIMESTAMP) kraja predavanja.

```
SELECT vrijemePoc,
       datum + vrijemePoc + trajanje krajPredavanja
  FROM predavanje
```

vrijemePoc	krajPredavanja
12:15:00	26.03.2019 13:00:00
13:15:00	26.03.2019 14:00:00

# Funkcije i NULL vrijednosti

---

- Neka je binarni operator  $\alpha \in \{ +, -, *, /, || \}$ , a X i Y su izrazi
  - ako jedan ili oba operanda X, Y poprimaju NULL vrijednost, tada je rezultat izraza X  $\alpha$  Y također NULL vrijednost
- Neka je unarni operator  $\beta \in \{ +, - \}$ , a X je izraz
  - ako operand X poprima NULL vrijednost, tada je rezultat izraza  $\beta$  X također NULL vrijednost
- Slično vrijedi i za funkcije
  - ako se kao jedan ili više argumenata funkcije zada NULL vrijednost, rezultat funkcije će također biti NULL vrijednost

# Funkcije i NULL vrijednosti (primjer)

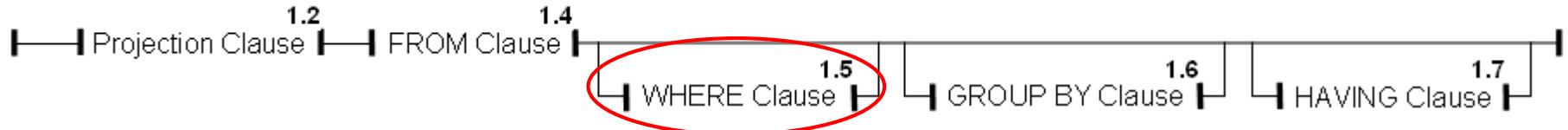
bodovi	mbr	prez	bodLab
	101	Novak	12
	103	Ban	NULL
	107	NULL	21
	109	Kolar	NULL

```
SELECT mbr
      , MOD(bodLab,10) AS ostatak
      , SUBSTRING(prez FROM 1 FOR 2) AS podniz
  FROM bodovi;
```

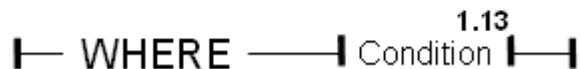
mbr	ostatak	podniz
101	2	No
103	NULL	Ba
107	1	NULL
109	NULL	Ko

# WHERE Clause

## 1.1. SELECT Options



## 1.5. WHERE Clause

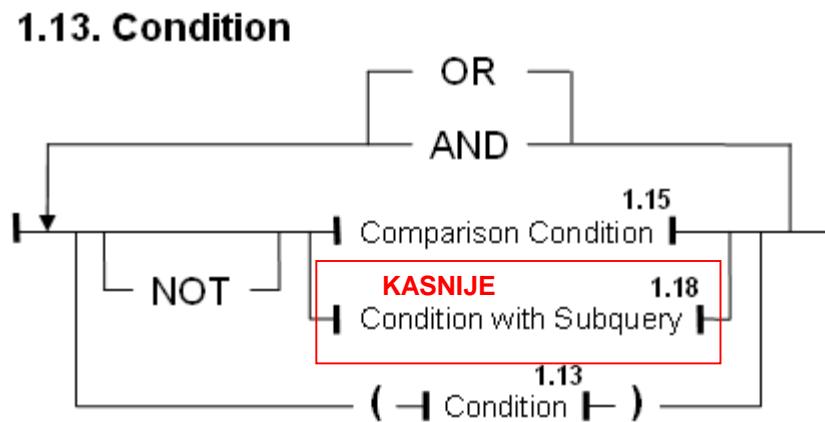


- Vrijednosti svake n-torke iz relacije *table* se uvrštavaju u *Condition* (a to je u stvari predikat). Ako je dobiveni sud istinit (*true*), n-torka se pojavljuje u rezultatu
- Mogući rezultati izračunavanja uvjeta: *true*, *false*, *unknown*

# Condition (ponavljanje)

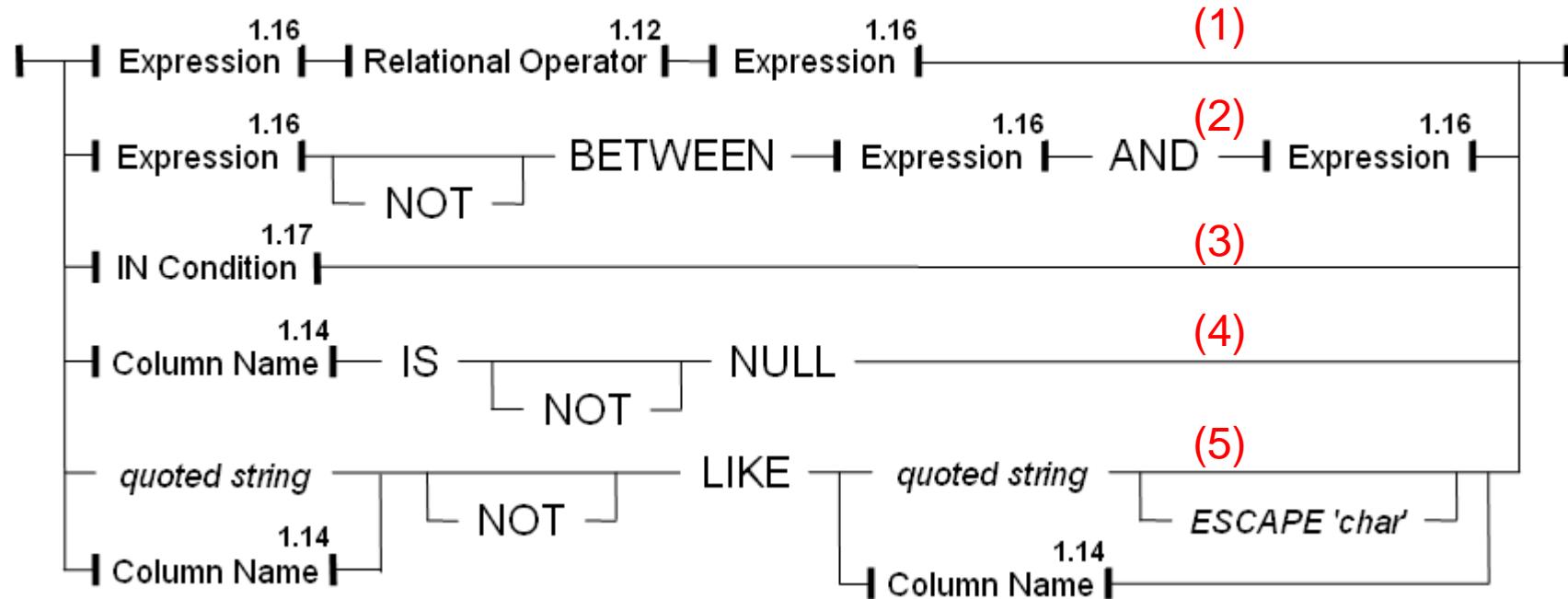
- `SELECT Select List FROM table [WHERE Condition]`
- Uvjet (*Condition*) se sastoji od operanada, operatora i zagrada
  - operandi su:
    - imena atributa iz relacije *table*
    - konstante
  - operatori su:
    - operatori usporedbe: < <= = <> > >=
    - logički operatori: AND OR NOT

- SQL omogućava dodatne oblike za opisivanje uvjeta



# Uvjet usporedbe (Comparison Condition)

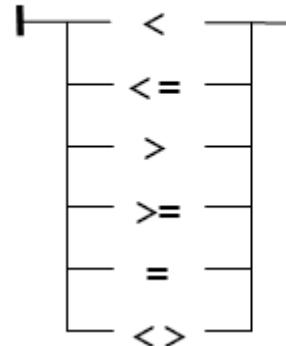
## 1.15. Comparison Condition



# Uvjet usporedbe (Comparison Condition) (1)

1.16 Expression      1.12 Relational Operator      1.16 Expression

## 1.12. Relational Operator

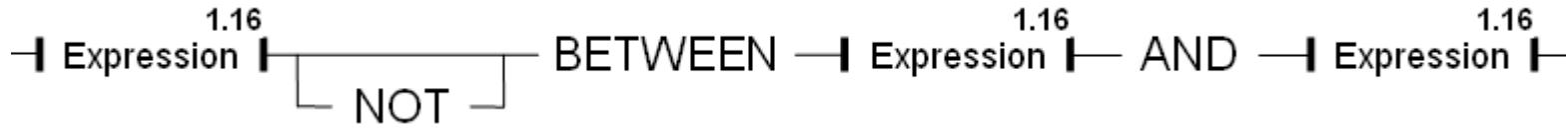


student	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	NULL	21000

```
SELECT * FROM student  
WHERE prez <> 'Kolar';
```

matBr	ime	prez	postBr
102	Ana	Horvat	10000

# Uvjet usporedbe (Comparison Condition) (2)



stanjeSklad	sifArt	minS	maxS	stanje
	1	10	50	50
	2	20	60	30
	3	10	80	5
	4	NULL	10	15
	5	10	20	NULL

```
SELECT * FROM stanjeSklad  
WHERE stanje BETWEEN minS AND maxS;
```

sifArt	minS	maxS	stanje
1	10	50	50
2	20	60	30

```
SELECT * FROM stanjeSklad  
WHERE stanje NOT BETWEEN minS AND maxS;
```

sifArt	minS	maxS	stanje
3	10	80	5
4	NULL	10	15

# Uvjet usporedbe (Comparison Condition) (3)

## 1.17. IN Condition



```
SELECT * FROM student  
WHERE prez IN ('Kolar', 'Horvat');
```

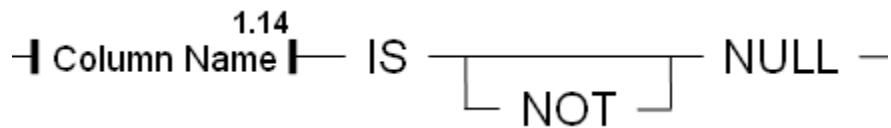
student	matBr	prez
100	Kolar	
102	Horvat	
103	Novak	
105	Horvat	
107	NULL	
109	Ban	

```
SELECT * FROM student  
WHERE prez NOT IN ('Kolar', 'Horvat');
```

matBr	prez
103	Novak
109	Ban

- ako *Expression* ima vrijednost NULL, tada je rezultat logička vrijednost *unknown*, bez obzira na vrijednosti navedene u skupu

# Uvjet usporedbe (Comparison Condition) (4)



student	matBr	prez	postBr
	100	Kolar	52000
	102	Horvat	10000
	105	Novak	NULL
	107	Ban	10000

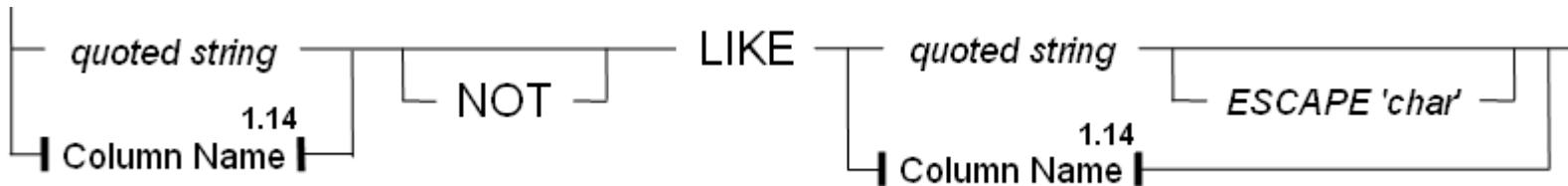
```
SELECT * FROM student  
WHERE postBr IS NULL;
```

```
SELECT * FROM student  
WHERE postBr IS NOT NULL;
```

matBr	prez	postBr
105	Novak	NULL

matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
107	Ban	10000

# Uvjet usporedbe (Comparison Condition) (5)



- služi za ispitivanje zadovoljava li (ili ne zadovoljava) vrijednost atributa ili znakovna konstanta zadani uzorak (*pattern*)
- mogu se koristiti sljedeći *wildcard* znakovi:
  - znak **%** zamjenjuje bilo koju kombinaciju znakova (0 ili više znakova)
  - znak **\_** zamjenjuje točno jedan znak

# Uvjet usporedbe (Comparison Condition) (5)

osoba

matBr	ime
1	Matija
2	Metka
3	Matilda
4	Ratkec
5	Marko
6	Ivan

```
SELECT * FROM osoba  
WHERE ime LIKE 'M%';
```

matBr	ime
1	Matija
2	Metka
3	Matilda
5	Marko

```
SELECT * FROM osoba  
WHERE ime LIKE 'Mat%';
```

matBr	ime
1	Matija
3	Matilda

```
SELECT * FROM osoba  
WHERE ime LIKE 'Ma_k%';
```

matBr	ime
5	Marko

char:

```
SELECT * FROM osoba  
WHERE TRIM(ime)  
LIKE '%tk_';
```

varchar:

```
SELECT * FROM osoba  
WHERE ime LIKE '%tk_';
```

matBr	ime
2	Metka

```
SELECT * FROM osoba  
WHERE ime LIKE '%tk%';
```

matBr	ime
2	Metka
4	Ratkec

# Uvjet usporedbe (Comparison Condition) (5)

tekstovi

rbr	tekst
1	deset %
2	pet % kisika
3	nije pet
4	nije_pet
5	% i _

- znak *char* naveden iza ESCAPE služi za poništavanje specijalnog značenja znakova % ili \_ koji su navedeni neposredno iza znaka *char*

```
SELECT * FROM tekstovi  
WHERE tekst LIKE '#%%'  
ESCAPE '#';
```

rbr	tekst
5	% i _

```
SELECT * FROM tekstovi  
WHERE tekst LIKE '%$%'  
ESCAPE '$';
```

rbr	tekst
1	deset %

```
SELECT * FROM tekstovi  
WHERE tekst LIKE '%$%'  
ESCAPE '$';
```

rbr	tekst
1	deset %
2	pet % kisika
5	% i _

```
SELECT * FROM tekstovi  
WHERE tekst LIKE '%!_pet'  
ESCAPE '!!';
```

rbr	tekst
4	nije_pet

# Uvjet usporedbe (Comparison Condition) (5)

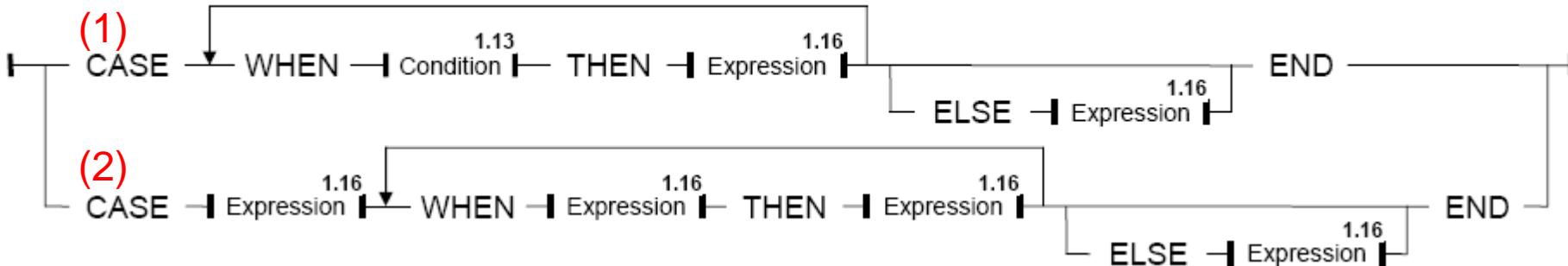
---

- $x \text{ LIKE } 'AB\%'$       *true za svaki x koji započinje s AB*
- $x \text{ LIKE } '%AB'$       *true za svaki x koji završava\* s AB*
- $x \text{ LIKE } '%%AB'$       *true za svaki x koji završava\* s AB*
  
- $x \text{ LIKE } 'AB\%CD'$       *true za svaki x koji započinje s AB i završava\* s CD*
- $x \text{ LIKE } '%AB\%'$       *true za svaki x koji sadrži AB*
- $x \text{ LIKE } '_AB'$       *true za svaki x duljine 3 znaka koji završava\* s AB*
- $x \text{ LIKE } '___AB'$       *true za svaki x duljine 4 znaka koji završava\* s AB*
- $x \text{ LIKE } 'AB__'$       *true za svaki x duljine\* 4 znaka koji započinje s AB*
- $x \text{ LIKE } '_AB\%'$       *true za svaki x koji započinje bilo kojim znakom, nastavlja se sa znakovima AB, te završava s bilo kojim znakovima*

\*ako je x varchar, inače trim(x)

# Uvjetni izraz (*Conditional Expression*)

## Conditional Expression



- 1. oblik izraza je sličan **if • else if • else** naredbi za višestranu selekciju u programskom jeziku C
- 2. oblik izraza je sličan **switch • case • default** naredbi za selekciju u programskom jeziku C
- pri čemu postoji bitna razlika:
  - C naredbama "odlučuje se" koje će se naredbe obaviti
  - SQL uvjetnim izrazom "odlučuje se" koja **vrijednost predstavlja rezultat** uvjetnog izraza

# Uvjetni izraz (*Conditional Expression*) (1)

```
SELECT *  
, CASE  
    WHEN ocjena = 5 THEN 'izvrstan'  
    WHEN ocjena = 4 THEN 'vrlo dobar'  
    WHEN ocjena = 3 THEN 'dobar'  
    WHEN ocjena = 2 THEN 'dovoljan'  
    WHEN ocjena = 1 THEN 'nedovoljan'  
    WHEN ocjena IS NULL THEN 'nepoznato'  
    ELSE 'neispravno'  
END AS opis  
FROM ispit;
```

ispit	matBr	ocjena
	100	5
	102	3
	103	1
	107	NULL
	109	6

matBr	ocjena	opis
100	5	izvrstan
102	3	dobar
103	1	nedovoljan
107	NULL	nepoznato
109	6	neispravno

- ako se više izraza uz WHEN izračuna kao *true*, rezultat izraza je *Expression* naveden uz prvi WHEN čiji se uvjet izračuna kao *true*
- ako se ELSE dio izraza ne navede, a niti jedan uvjet uz WHEN se ne izračuna kao *true*, tada je rezultat izraza NULL vrijednost

# Uvjetni izraz (*Conditional Expression*) (2)

```
SELECT *  
  , CASE ocjena  
    WHEN 5 THEN 'izvrstan'  
    WHEN 4 THEN 'vrlo dobar'  
    WHEN 3 THEN 'dobar'  
    WHEN 2 THEN 'dovoljan'  
    WHEN 1 THEN 'nedovoljan'  
    ELSE 'neispravno'  
  END AS opis  
FROM ispit;
```

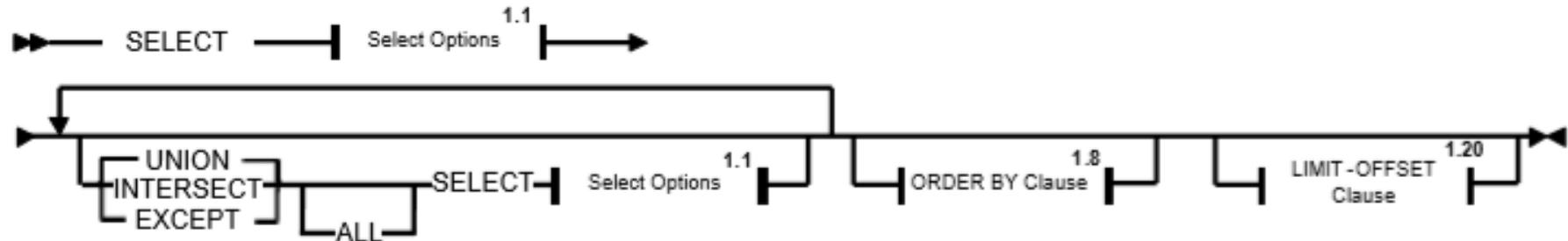
ispit	matBr	ocjena
	100	5
	102	3
	103	1
	107	NULL
	109	6

matBr	ocjena	opis
100	5	izvrstan
102	3	dobar
103	1	nedovoljan
107	NULL	neispravno
109	6	neispravno

- ako više izraza uz WHEN zadovoljava uvjet jednakosti, rezultat izraza je *Expression* naveden uz prvi WHEN koji zadovoljava uvjet
- ako se ELSE dio izraza ne navede, a niti jedan izraz ne zadovoljava uvjet jednakosti, tada je rezultat izraza NULL vrijednost

# Unija, presjek i razlika

## 1. SELECT Statement



- *SELECT Statement* može se graditi od jednog ili više *SELECT dijelova*
- **UNION, INTERSECT, EXCEPT** - uz izbacivanje duplikata (kopija n-torki)
- **UNION ALL, INTERSECT ALL, EXCEPT ALL** - bez izbacivanja duplikata (kopija n-torki)
- imena stupaca (atributa rezultantne relacije) određuju se na temelju imena stupaca iz prvog navedenog *SELECT dijela*
- poredak atributa u različitim *SELECT dijelovima* mora biti jednak
- korespondentni atributi / izrazi moraju odgovarati po tipu podatka i po značenju – unijska kompatibilnost

# Unija (*UNION*)

- polozioMat  $\cup$  polozioProg  $\cup$  polozioDiglog

polozioMat

mbr	imeSt	prezSt
100	Ivan	NULL
102	Ana	Novak
105	Rudi	Kolar
111	Jura	Horvat

polozioProg

mbr	ime	prez
100	Ivan	NULL
103	NULL	Ban
105	Rudi	Kolar

polozioDiglog

mbr	ime	prez
102	Ana	Novak
103	NULL	Ban
105	Rudi	Kolar
111	Jura	Horvat

```
SELECT * FROM polozioMat  
UNION  
SELECT * FROM polozioProg  
UNION  
SELECT * FROM polozioDiglog;
```

mbr	imeSt	prezSt
100	Ivan	NULL
102	Ana	Novak
103	NULL	Ban
105	Rudi	Kolar
111	Jura	Horvat

# Unija – multisetovi (*UNION ALL*)

- rezultat sljedeće naredbe nije relacija!

polozioMat

mbr	ime	prez
100	Ivan	NULL
102	Ana	Novak
105	Rudi	Kolar
111	Jura	Horvat

polozioProg

mbr	imeSt	prezSt
100	Ivan	NULL
103	NULL	Ban
105	Rudi	Kolar

polozioDiglog

mbr	imeSt	prezSt
102	Ana	Novak
103	NULL	Ban
105	Rudi	Kolar
111	Jura	Horvat

```
SELECT * FROM polozioMat
UNION ALL
SELECT * FROM polozioProg
UNION ALL
SELECT * FROM polozioDiglog;
```

mbr	ime	prez
100	Ivan	NULL
102	Ana	Novak
105	Rudi	Kolar
111	Jura	Horvat
100	Ivan	NULL
103	NULL	Ban
105	Rudi	Kolar
102	Ana	Novak
103	NULL	Ban
105	Rudi	Kolar
111	Jura	Horvat

# Unija (*UNION*)

- naredba je ispravna ako su korespondentni atributi istih tipova podataka (INTEGER-INTEGER, CHAR-CHAR, ...), ali odgovornost je korisnika (programera) voditi računa o unijskoj kompatibilnosti
- npr. sljedeća naredba će se obaviti, ali rezultat je besmislen

pecivo

oznaka	naziv
ZE-33	Žemlja s makom
PR-3	Perec sa sezamom

zrakoplov

oznaka	naziv
PR-3	Piper J-3 Cub
B-747	Boeing 747
A-360	Airbus 360

```
SELECT * FROM pecivo  
UNION  
SELECT * FROM zrakoplov;
```

oznaka	naziv
ZE-33	Žemlja s makom
PR-3	Perec sa sezamom
PR-3	Piper J-3 Cub
B-747	Boeing 747
A-360	Airbus 360

Piper J-3 Cub



Perec sa sezamom



# Presjek (INTERSECT)

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

studenti koji su položili i  
Matematiku i Programiranje

**polozioMatem  $\cap$  polozioProgr**

```
SELECT * FROM polozioMatem
INTERSECT
SELECT * FROM polozioProgr;
```

mbr	ime	prez
102	Ana	Novak
107	Jura	Horvat

# Presjek – multisetovi (INTERSECT ALL)

polagaoMatem

mbr	ime	prez
100	Ivan	Kolar
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat
107	Jura	Horvat

polagaoProgr

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat
107	Jura	Horvat
107	Jura	Horvat

Ispisati podatke o studentima onoliko puta koliko puta su polagali i Matematiku i Programiranje

**polagaoMatem  $\cap_{\text{ALL}}$  polagaoProgr**

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
102	Ana	Novak
107	Jura	Horvat
107	Jura	Horvat

```
SELECT * FROM polagaoMatem  
INTERSECT ALL  
SELECT * FROM polagaoProgr;
```

# Razlika (EXCEPT)

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

studenti koji su položili Matematiku,  
ali nisu položili Programiranje

**polozioMatem \ polozioProgr**

```
SELECT * FROM polozioMatem  
EXCEPT  
SELECT * FROM polozioProgr
```

mbr	ime	prez
100	Ivan	Kolar
103	Tea	Ban

# Razlika – multisetovi (EXCEPT ALL)

polagaoMatem

mbr	ime	prez
100	Ivan	Kolar
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat
107	Jura	Horvat

polagaoProgr

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat
107	Jura	Horvat
107	Jura	Horvat

studenti koji su polagali  
Matematiku više puta nego  
što su polagali Programiranje

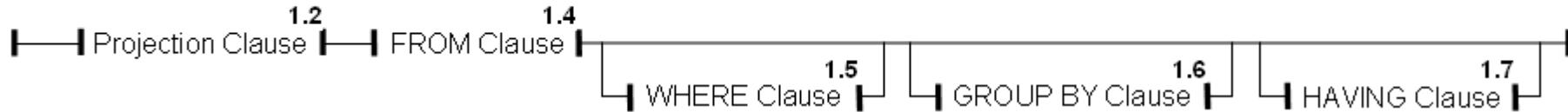
**polagaoMatem \ ALL polagaoProgr**

```
SELECT * FROM polagaoMatem  
EXCEPT ALL  
SELECT * FROM polagaoProgr;
```

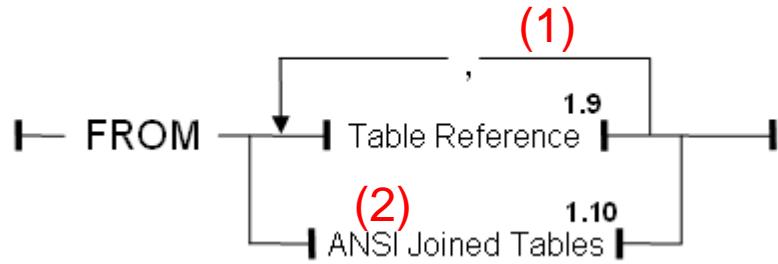
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
102	Ana	Novak
103	Tea	Ban

# FROM Clause

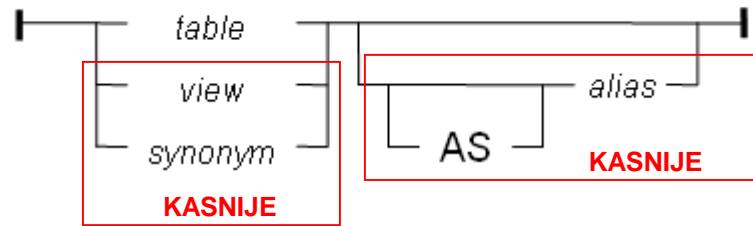
## 1.1. SELECT Options



## 1.4. FROM Clause



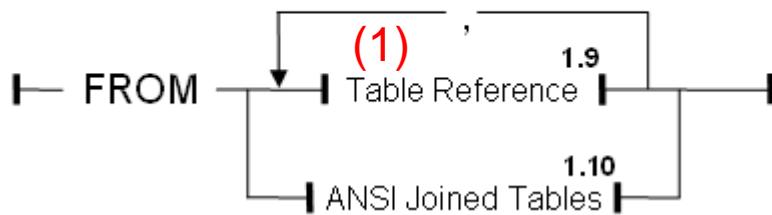
## 1.9. Table Reference



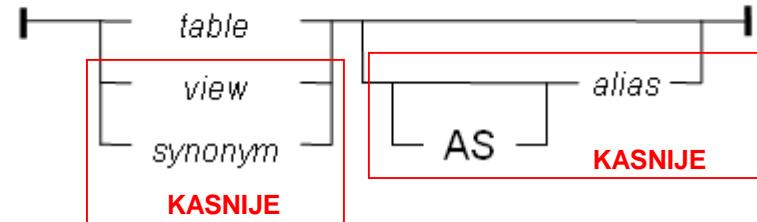
- (1) klasična sintaksa (*classical, comma-delimited*) za spajanje relacija
- (2) ANSI sintaksa za spajanje relacija

# FROM Clause (1)

## 1.4. FROM Clause



## 1.9. Table Reference



- klasična sintaksa (*classical, comma-delimited*) može se koristiti za obavljanje operacija:
  - Kartezijev produkt
  - spajanje uz uvjet i spajanje s izjednačavanjem
  - prirodno spajanje
- uvjeti spajanja se navode u WHERE dijelu SELECT naredbe, zajedno s eventualnim uvjetima selekcije (uvjeti spajanja i uvjeti selekcije se u tom slučaju povezuju logičkim operatorom AND)

# FROM Clause (1)

- Zadane su relacije:  $r (\{ A, B \})$     $s (\{ C, D \})$     $t (\{ D, E \})$

$r \times s$

```
SELECT *
  FROM r, s;
```

$r \bowtie s$   
 $A = C \wedge B \geq D$

```
SELECT *
  FROM r, s
 WHERE A = C
   AND B >= D;
```

$r \bowtie s$   
 $B = C$

```
SELECT *
  FROM r, s
 WHERE B = C;
```

$\sigma_{D>5}(r \bowtie s)$   
 $B = C$

```
SELECT *
  FROM r, s
 WHERE B = C
   AND D > 5;
```

# *FROM Clause (1)*

- $r (\{ A, B \}) \quad s (\{ C, D \}) \quad t (\{ D, E \})$

$(r \times s) \bowtie t$

```
SELECT r.* , s.* , t.E  
FROM r , s , t  
WHERE s.D = t.D;
```

$(r \bowtie s) \bowtie t$

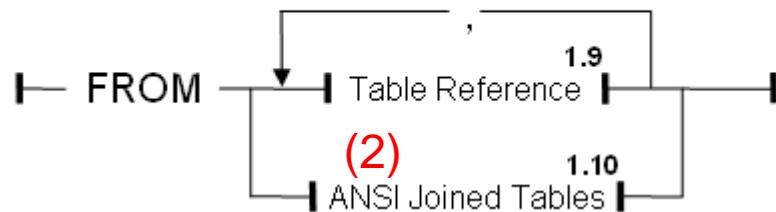
```
SELECT r.* , s.* , t.E  
FROM r , s , t  
WHERE s.D = t.D;
```

$\sigma_{C=100}(s \bowtie t)$

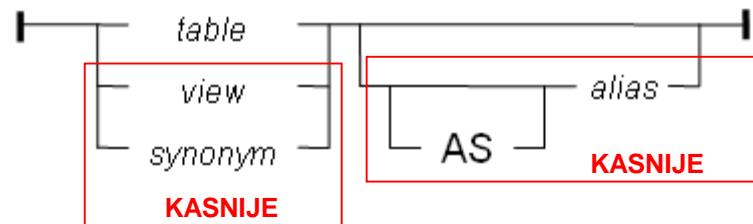
```
SELECT s.* , t.E  
FROM s , t  
WHERE s.D = t.D  
AND C = 100;
```

# FROM Clause (2)

## 1.4. FROM Clause



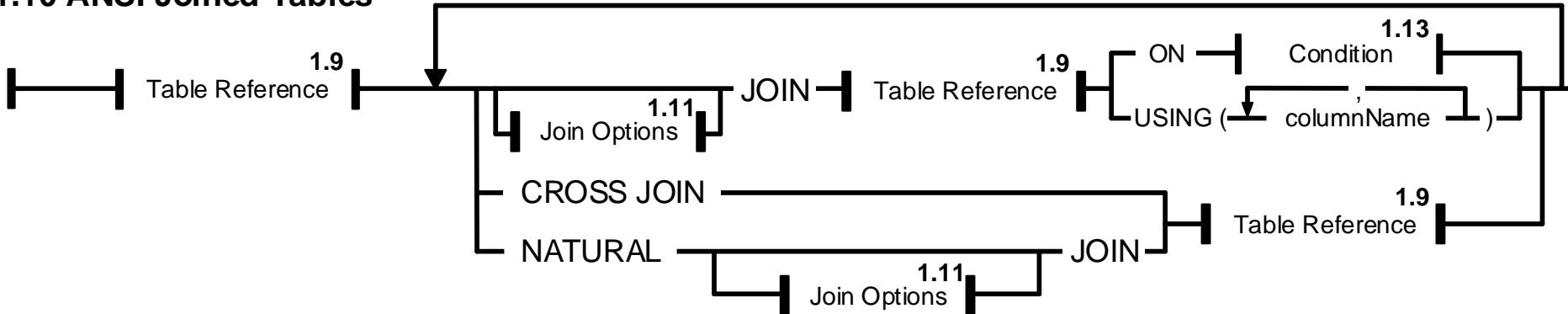
## 1.9. Table Reference



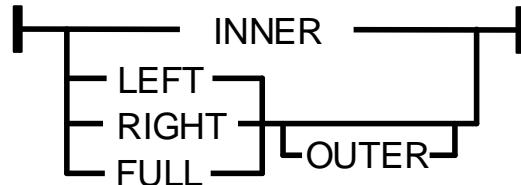
- ANSI sintaksa za spajanje relacija. Može se koristiti za obavljanje operacija:
  - Kartežijev produkt
  - spajanje uz uvjet i spajanje s izjednačavanjem
  - prirodno spajanje
  - **vanjsko spajanje**
    - vanjsko spajanje uz uvjet
    - vanjsko spajanje s izjednačavanjem
    - prirodno vanjsko spajanje

# FROM Clause (2)

## 1.10 ANSI Joined Tables



## 1.11 Join Options



- r CROSS JOIN s
- r NATURAL JOIN s
- r NATURAL LEFT JOIN s
- r NATURAL RIGHT JOIN s
- r NATURAL FULL JOIN s
- r INNER JOIN s ON uvjetSpajanja
- r LEFT OUTER JOIN s ON uvjetSpajanja
- r RIGHT OUTER JOIN s ON uvjetSpajanja
- r FULL OUTER JOIN s ON uvjetSpajanja

## **FROM Clause (2)**

---

- Specifično za PostgreSQL – prirodna spajanja - unutarnja i vanjska:

```
r INNER JOIN s USING (atr1, atr2, ...)  
r LEFT OUTER JOIN s USING (atr1, atr2, ...)  
r RIGHT OUTER JOIN s USING (atr1, atr2, ...)  
r FULL OUTER JOIN s USING (atr1, atr2, ...)
```

# **FROM Clause (2)**

- Rezervirane riječi OUTER i INNER se smiju izostaviti:

`r INNER JOIN s`       $\equiv$     `r JOIN s`

`r LEFT OUTER JOIN s`    ≡    `r LEFT JOIN s`

`r RIGHT OUTER JOIN s` ≡ `r RIGHT JOIN s`

`r FULL OUTER JOIN s`    $\equiv$   `r FULL JOIN s`

# FROM Clause (2)

- Zadane su relacije:  $r (\{ A, B \})$     $s (\{ C, D \})$     $t (\{ D, E \})$

$r \times s$

```
SELECT *
  FROM r CROSS JOIN s;
```

$r \bowtie s$   
 $A = C \wedge B \geq D$

```
SELECT *
  FROM r
  INNER JOIN s
    ON A = C AND B >= D;
```

$r \bowtie s$   
 $B = C$

```
SELECT *
  FROM r
  INNER JOIN s
    ON B = C;
```

$\sigma_{D>5}(r \bowtie s)$

```
SELECT *
  FROM r
  INNER JOIN s
    ON B = C
 WHERE D > 5;
```

## FROM Clause (2)

- $r(\{A, B\}) \quad s(\{C, D\}) \quad t(\{D, E\}) \quad p(\{E, F\})$

$(r \times s) \bowtie t$

```
SELECT r.*, s.*, t.E  
FROM r  
CROSS JOIN s  
INNER JOIN t  
ON s.D = t.D;
```

```
SELECT r.*, s.*, t.E  
FROM r  
CROSS JOIN s  
NATURAL JOIN t;
```

$(s \bowtie t) \bowtie p$

```
SELECT s.*, t.E, p.F  
FROM s  
INNER JOIN t  
ON s.D = t.D  
INNER JOIN p  
ON t.E = p.E;
```

```
SELECT s.*, t.E, p.F  
FROM s  
NATURAL JOIN t  
NATURAL JOIN p;
```

$\sigma_{C=100}(s \bowtie t)$

```
SELECT s.*, t.E  
FROM s  
INNER JOIN t  
ON s.D = t.D  
WHERE C = 100;
```

```
SELECT s.*, t.E  
FROM s  
NATURAL JOIN t  
WHERE C = 100;
```

# FROM Clause (2)

- $r(\{A, B\}) \ s(\{C, D\}) \ t(\{D, E\}) \ p(\{E, F\})$

$(r \times s)^* \triangleright \triangleleft t$

```
SELECT r.* , s.* , t.E  
FROM r  
CROSS JOIN s  
LEFT OUTER JOIN t  
ON s.D = t.D;
```

```
SELECT r.* , s.* , t.E  
FROM r  
CROSS JOIN s  
NATURAL LEFT JOIN t;
```

$(s^* \triangleright \triangleleft^* t) \triangleright \triangleleft^* p$

```
SELECT s.* , t.D D1 , p.*  
FROM s  
FULL OUTER JOIN t  
ON s.D = t.D  
RIGHT OUTER JOIN p  
ON t.E = p.E;
```

```
SELECT s.* , t.D D1 , p.*  
FROM s  
NATURAL FULL JOIN t  
NATURAL RIGHT JOIN p;
```

$\sigma_{C=100}(s^* \triangleright \triangleleft t)$

```
SELECT s.* , t.E  
FROM s  
LEFT OUTER JOIN t  
ON s.D = t.D  
WHERE C = 100;
```

```
SELECT s.* , t.E  
FROM s  
NATURAL LEFT JOIN t  
WHERE C = 100;
```

# FROM Clause (2)

- Ako se obavlja operacija spajanja i selekcija, uvjete spajanja treba navesti u ON dijelu, a uvjete selekcije treba navesti u WHERE dijelu SELECT naredbe
  - iako, u slučaju kad nema vanjskog spajanja, rezultat upita ne ovisi o tome je li uvjet selekcije naveden u ON ili WHERE dijelu naredbe

student	matBr	prez	pbrSt
	101	Kolar	10000
	102	Horvat	21000
	103	Novak	NULL

```
SELECT *
  FROM student
  JOIN mjesto
    ON pbrSt = pbr
 WHERE prez = 'Kolar';
```

$\sigma_{\text{prez} = \text{'Kolar'}}(\text{student} \bowtie \text{mjesto})$   
 $\quad \quad \quad \text{pbrst} = \text{pbr}$

mjesto	pbr	nazMjesto
	10000	Zagreb
	21000	Split

```
SELECT *
  FROM student
  JOIN mjesto
    ON pbrSt = pbr
   AND prez = 'Kolar';
```

student  $\bowtie$  mjesto  
 $\text{pbrst} = \text{pbr} \wedge \text{prez} = \text{'Kolar'}$

- oba upita daju isti rezultat

matBr	prez	pbrSt	pbr	nazMjesto
101	Kolar	10000	10000	Zagreb

# FROM Clause (2)

- Ako se koristi vanjsko spajanje, navođenje uvjeta selekcije u ON dijelu umjesto WHERE dijelu može **bitno** utjecati na rezultat

student	matBr	prez	pbrSt
	101	Kolar	10000
	102	Horvat	21000
	103	Novak	NULL

mjesto	pbr	nazMjesto
	10000	Zagreb
	21000	Split

```
SELECT *
  FROM student
LEFT JOIN mjesto
    ON pbrSt = pbr
 WHERE prez = 'Kolar';
```

$$\sigma_{\text{prez} = \text{'Kolar'}}(\text{student} * \bowtie \text{mjesto})$$

pbrst = pbr

- Tek nakon obavljenog spajanja prema uvjetu navedenom u ON dijelu naredbe, obavlja se selekcija n-torki prema uvjetu navedenom u WHERE dijelu naredbe

matBr	prez	pbrSt	pbr	nazMjesto
101	Kolar	10000	10000	Zagreb

# FROM Clause (2)

- Ovdje je prikazan upit sličan prethodnom, ali u kojem je uvjet selekcije napisan na "pogrešnom" mjestu

student	matBr	prez	pbrSt
	101	Kolar	10000
	102	Horvat	21000
	103	Novak	NULL

mjesto	pbr	nazMjesto
	10000	Zagreb
	21000	Split

```
SELECT *
  FROM student
  LEFT JOIN mjesto
    ON pbrSt = pbr
   AND prez = 'Kolar';
```

student \*▷◁ mjesto  
pbrst = pbr ∧ prez = 'Kolar'

- Ovdje će se pojaviti sve n-torke iz relacije student - uz one n-torke relacije student koje ne zadovoljavaju uvjet spajanja (uočite koji je uvjet spajanja ovdje naveden) dodat će se NULL vrijednosti

matBr	prez	pbrSt	pbr	nazMjesto
101	Kolar	10000	10000	Zagreb
102	Horvat	21000	NULL	NULL
103	Novak	NULL	NULL	NULL

## *FROM Clause (2)*

---

- **Logički promatrano\***, kada se u upitu spajaju više od dvije relacije, redoslijed spajanja je slijeva na desno: spajaju se prve dvije relacije, zatim se dobiveni rezultat spaja s trećom navedenom relacijom, zatim se dobiveni rezultat spaja s četvrtom navedenom relacijom, itd.
- (\*) konačni rezultat će sigurno odgovarati rezultatu koji bi se dobio kada bi se relacije spajale s lijeva na desno. Fizički promatrano, upit će se možda izvesti drugačijim redoslijedom, ali o tome brine dio SUBP-a koji se naziva optimizator upita

# FROM Clause (2)

- ako se ne koristi vanjsko spajanje, redoslijed spajanja je ionako relevantan, jer vrijedi:

$$( r_1 \bowtie r_2 ) \bowtie r_3 \equiv r_1 \bowtie ( r_2 \bowtie r_3 )$$

- ako se koristi vanjsko spajanje, redoslijed spajanja jest važan jer:

$$( r_1 * \bowtie r_2 ) \bowtie r_3 \neq r_1 * \bowtie ( r_2 \bowtie r_3 )$$

student

mBr	prez	pbr
101	Kolar	10000
102	Horvat	21000
103	Novak	NULL

mjesto

pbr	nazMj	sifZup
10000	Zagreb	21
21000	Split	17

zupanija

sifZup	nazZup
21	Grad Zagreb
17	Splitsko-dalmatinska

mBr	prez	pbr	nazMj	sifZup	nazZup
101	Kolar	10000	Zagreb	21	Grad Zagreb
102	Horvat	21000	Split	17	Splitsko-dalmatinska

$$r_1 * \bowtie ( r_2 \bowtie r_3 )$$

$$( r_1 * \bowtie r_2 ) \bowtie r_3$$

mBr	prez	pbr	nazMj	sifZup	nazZup
101	Kolar	10000	Zagreb	21	Grad Zagreb
102	Horvat	21000	Split	17	Splitsko-dalmatinska
103	Novak	NULL	NULL	NULL	NULL

# FROM Clause (2)

stud			mjesto			zupanija	
mbr	prez	pbrSt	pbr	nazMjesto	sifZupMj	sifZup	nazZup
101	Horvat	42000	42000	Varaždin	7	7	Varaždinska
102	Novak	21000	21000	Split	NULL	4	Istarska

( stud \*  $\triangleright\triangleleft$  mjesto )  $\triangleright\triangleleft$  zupanija  
pbrSt=pbr                    sifZupMj=sifZup

```
SELECT stud.* , mjesto.* , zupanija.*  
FROM stud  
LEFT OUTER JOIN mjesto  
ON pbrSt = pbr  
INNER JOIN zupanija  
ON sifZupMj = sifZup;
```

- prvo se spajaju relacije stud i mjesto, a zatim se dobiveni rezultat spaja s relacijom zupanija

mbr	prez	pbrSt	pbr	nazMjesto	sifZupMj	sifZup	nazZup
101	Horvat	42000	42000	Varaždin	7	7	Varaždinska

## FROM Clause (2)

stud \*▷◁ ( mjesto ▷◁ zupanija )  
pbrSt=pbr                                    sifZupMj=sifZup

≡ ( mjesto ▷◁ zupanija ) ▷◁\* stud  
sifZupMj=sifZup                            pbrSt=pbr

da bismo izraz  
relacijske algebре mogli  
napisati u obliku SQL  
naredbe, napisat ћemo  
ga u drugačijem obliku

```
SELECT stud.* , mjesto.* , zupanija.*  
FROM mjesto  
    INNER JOIN zupanija  
        ON sifZupMj = sifZup  
    RIGHT OUTER JOIN stud  
        ON pbrSt = pbr;
```

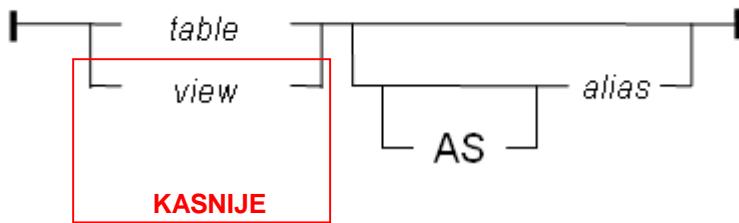
- prvo se spajaju relacije mjesto i zupanija, a zatim se  
dobiveni rezultat spaja s relacijom stud

mbr	prez	pbrSt	pbr	nazMjesto	sifZupMj	sifZup	nazZup
101	Horvat	42000	42000	Varaždin	7	7	Varaždinska
102	Novak	21000	NULL	NULL	NULL	NULL	NULL

# Preimenovanje relacija unutar upita

- relacija se unutar upita može preimenovati u *alias* ime
  - alias* ime je vidljivo samo unutar upita (ne utječe na stvarno ime relacije u bazi podataka)

## 1.9. Table Reference



- rezervirana riječ AS se smije ispuštiti
- na relaciju koja je u upitu dobila *alias* ime, moguće je referencirati se isključivo preko tog istog *alias* imena

```
SELECT nazMjesto, nazZupanija
  FROM mjesto AS town
    , zupanija AS county
 WHERE town.sifZupanija = county.sifZupanija;
```

```
SELECT nazMjesto, nazZupanija
  FROM mjesto AS town
    JOIN zupanija AS county
      ON town.sifZupanija = county.sifZupanija;
```

# Preimenovanje relacija unutar upita

- iako se preimenovanjem relacija može skratiti duljina teksta upita, u praksi se to **ne preporuča** jer upiti postaju manje razumljivi

```
SELECT o.jmbg, prezime, m.pbr, nazMjesto
  FROM osoba AS o
    , mjesto AS m
    , zaposlenje AS z1
    , zupanija AS z2
 WHERE o.jmbg = z1.jmbg
   AND o.pbr = m.pbr
   AND m.sifZup = z2.sifZup
   AND z2.nazZup = 'Varaždinska'
   AND z1.radnoMjesto = 'Dimnjačar'
```

- preimenovanje relacija unutar upita treba se koristiti onda kada se ista relacija pojavljuje u više uloga unutar istog upita

# Paralelno spajanje

student	mbr	prez	pbrRod	pbrStan
	100	Kolar	10000	21000
	102	Novak	21000	10000
	103	Ban	10000	10000

mjesto	pbr	nazMjesto
	10000	Zagreb
	21000	Split

- Kako dobiti sljedeći rezultat:

mbr	prez	pbrRod	pbrStan	nazMjestoR
100	Kolar	10000	21000	Zagreb
102	Novak	21000	10000	Split
103	Ban	10000	10000	Zagreb

- To je lako:

```
SELECT student.* , mjesto.nazMjesto AS nazMjestoR
  FROM student
       , mjesto
 WHERE student.pbrRod = mjesto.pbr;
```

# Paralelno spajanje

student	mbr	prez	pbrRod	pbrStan	mjesto	pbr	nazMjesto
	100	Kolar	10000	21000		10000	Zagreb
	102	Novak	21000	10000		21000	Split
	103	Ban	10000	10000			

- Kako dobiti sljedeći rezultat:

mbr	prez	pbrRod	nazMjestoR	pbrStan	nazMjestoS
100	Kolar	10000	Zagreb	21000	Split
102	Novak	21000	Split	10000	Zagreb
103	Ban	10000	Zagreb	10000	Zagreb

# Paralelno spajanje

student

mbr	prez	pbrRod	pbrStan
100	Kolar	10000	21000
102	Novak	21000	10000
103	Ban	10000	10000

mjesto

pbr	nazMjesto
10000	Zagreb
21000	Split

```
SELECT mbr, prez
      , pbrRod, mjesto.nazMjesto AS nazMjestoR
      , pbrStan, mjesto.nazMjesto AS nazMjestoS
  FROM student
    , mjesto
 WHERE student.pbrRod = mjesto.pbr
   AND student.pbrStan = mjesto.pbr;
```

NEISPRAVNO RJEŠENJE

mbr	prez	pbrRod	nazMjestoR	pbrStan	nazMjestoS
103	Ban	10000	Zagreb	10000	Zagreb

- Upit **nije dobar** jer jednu n-torku iz relacije student pokušavamo spojiti s jednom n-torkom iz relacije mjesto uz sljedeći uvjet spajanja: vrijednost atributa pbrRod, te **istovremeno** i vrijednost atributa pbrStan iz relacije student su jednake vrijednosti atributa pbr iz relacije mjesto

# Paralelno spajanje

- Kad bismo načinili dvije kopije relacije mjesto: mjestoR i mjestoS, sa shemama i sadržajem jednakim relaciji mjesto:



```
SELECT mbr, prez
      , pbrRod, mjestoR.nazMjesto AS nazMjestoR
      , pbrStan, mjestoS.nazMjesto AS nazMjestoS
FROM student, mjestoR, mjestoS
WHERE student.pbrRod = mjestoR.pbr
  AND student.pbrStan = mjestoS.pbr;
```

- konačni rezultat je ispravan, ali radi se o vrlo lošem rješenju!

# Paralelno spajanje

- Ispravno rješenje: za tablicu **mjesto** definiramo dvije različite uloge

student

mbr	prez	pbrRod	pbrStan
100	Kolar	10000	21000
102	Novak	21000	10000
103	Ban	10000	10000

mjesto AS mjestoR

mjesto	
pbr	nazMjesto
10000	Zagreb
21000	Split

mjesto AS mjestoS

mjestoR

pbr	nazMjesto
10000	Zagreb
21000	Split

mjestoS

pbr	nazMjesto
10000	Zagreb
21000	Split

```
SELECT mbr, prez
      , pbrRod, mjestoR.nazMjesto AS nazMjestoR
      , pbrStan, mjestoS.nazMjesto AS nazMjestoS
FROM student
      , mjesto AS mjestoR
      , mjesto AS mjestoS
WHERE student.pbrRod = mjestoR.pbr
    AND student.pbrStan = mjestoS.pbr;
```

- u upitu se ista relacija pojavljuje u dvije različite uloge

# Paralelno spajanje - ANSI

- Ispravno rješenje:

student			
mbr	prez	pbrRod	pbrStan
100	Kolar	10000	21000
102	Novak	21000	10000
103	Ban	10000	10000

mjesto AS mjestoR

mjesto	
pbr	nazMjesto
10000	Zagreb
21000	Split

mjesto AS mjestoS

mjestoR	
pbr	nazMjesto
10000	Zagreb
21000	Split

mjestoS	
pbr	nazMjesto
10000	Zagreb
21000	Split

```
SELECT mbr, prez
      , pbrRod, mjestoR.nazMjesto AS nazMjestoR
      , pbrStan, mjestoS.nazMjesto AS nazMjestoS
FROM student
      JOIN mjesto AS mjestoR
        ON student.pbrRod = mjestoR.pbr
      JOIN mjesto AS mjestoS
        ON student.pbrStan = mjestoS.pbr;
```

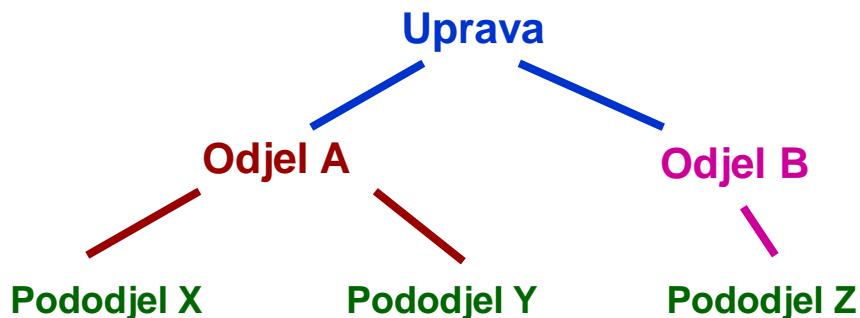
# Refleksivno spajanje

- Pojedine n-torke iz relacije povezane su s drugim n-torkama iz **iste** relacije

orgjed

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

- Uprava nema nadređenu org. jedinicu
- Odjelu A neposredno nadređena jedinica je Uprava
- Odjelu B neposredno nadređena jedinica je Uprava
- Pododjelu X neposredno nadređena jedinica je Odjel A
- itd.



# Refleksivno spajanje

- Kako dobiti sljedeći rezultat

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
1	Uprava	NULL	NULL
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

- problem je sličan i slično se rješava kao u slučaju paralelnog spajanja
- radi se o spajanju relacije same sa sobom
- relacija orgjed treba se u upitu pojaviti dva puta, jednom u ulozi organizacijske jedinice, a jednom u ulozi njezine nadređene organizacijske jedinice

# Refleksivno spajanje

orgjed

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	<b>NULL</b>
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

orgjed (u ulozi nadređene org.jedinice)

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	<b>NULL</b>
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

```
SELECT orgjed.sifOrgjed
      , orgjed.nazOrgjed
      , orgjed.sifNadorgjed
      , nadorgjed.nazOrgjed AS nazNadorgjed
  FROM orgjed, orgjed AS nadOrgjed
 WHERE orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
```

# Refleksivno spajanje

```
SELECT orgjed.sifOrgjed  
      , orgjed.nazOrgjed  
      , orgjed.sifNadorgjed  
      , nadorgjed.nazOrgjed AS nazNadorgjed  
  FROM orgjed, orgjed AS nadOrgjed  
 WHERE orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
```

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

- Nema organizacijske jedinice Uprava? Kako to popraviti?

# Refleksivno spajanje

```
SELECT orgjed.sifOrgjed
      , orgjed.nazOrgjed
      , orgjed.sifNadorgjed
      , nadorgjed.nazOrgjed AS nazNadorgjed
  FROM orgjed
    LEFT OUTER JOIN orgjed AS nadOrgjed
      ON orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
```

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
1	Uprava	NULL	NULL
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

# Refleksivno spajanje

- Kako dobiti sljedeći rezultat
  - uz svaku organizacijsku jedinicu ispisati nazine neposredno podređenih organizacijskih jedinica
  - ako org. jedinica ima više od jedne podređene org. jedinice, u popisu se pojavljuje više puta
  - u popisu se moraju naći i one organizacijske jedinice koje nemaju niti jednu podređenu organizacijsku jedinicu

sifOrgjed	nazOrgjed	nazPodorgjed
1	Uprava	Odjel A
1	Uprava	Odjel B
2	Odjel A	Pododjel X
2	Odjel A	Pododjel Y
3	Odjel B	Pododjel Z
4	Pododjel X	NULL
5	Pododjel Y	NULL
6	Pododjel Z	NULL

# Refleksivno spajanje

orgjed

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	<b>NULL</b>
2	Odjel A	<b>1</b>
3	Odjel B	<b>1</b>
4	Pododjel X	<b>2</b>
5	Pododjel Y	<b>2</b>
6	Pododjel Z	<b>3</b>

orgjed (u ulozi podređene org.jedinice)

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	<b>NULL</b>
2	Odjel A	<b>1</b>
3	Odjel B	<b>1</b>
4	Pododjel X	<b>2</b>
5	Pododjel Y	<b>2</b>
6	Pododjel Z	<b>3</b>

```
SELECT orgjed.sifOrgjed  
      , orgjed.nazOrgjed  
      , podOrgjed.nazOrgjed AS nazPodorgjed  
FROM orgjed  
    LEFT OUTER JOIN orgjed AS podOrgjed  
      ON podOrgjed.sifNadorgjed = orgjed.sifOrgjed;
```

# Preimenovanje relacija unutar upita

- Još jedan primjer u kojem se koristi preimenovanje relacije
- Ispisati podatke o svim osobama čija je plaća manja od plaće osobe sa šifrom 103

osoba103	sifra	ime	prez	placa
	103	Ana	Novak	5000

osoba	sifra	ime	prez	placa
	100	Ana	Novak	6000
	101	Ana	Kolar	5000
	102	Ivan	Kolar	3000
	103	Ana	Novak	5000
	104	Jura	Ban	4000

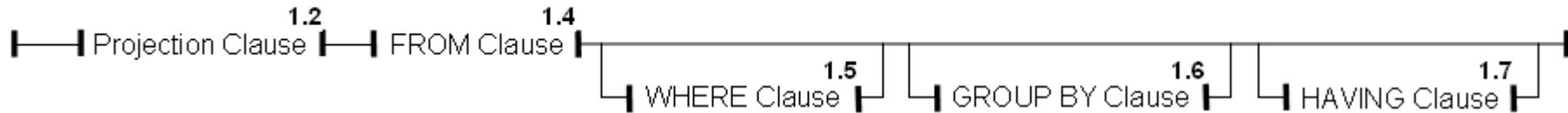
$\text{osoba} \bowtie \text{Posoba103}(s,i,p,\text{placa103}) \ (\sigma_{\text{sifra} = 103}(\text{osoba}))$   
placa < placa103

```
SELECT osoba.*  
FROM osoba  
INNER JOIN osoba AS osoba103  
ON osoba.placa < osoba103.placa  
AND osoba103.sifra = 103;
```

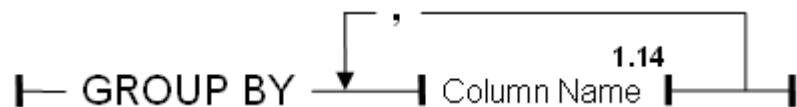
sifra	ime	prez	placa
102	Ivan	Kolar	3000
104	Jura	Ban	4000

# GROUP BY Clause

## 1.1. SELECT Options



## 1.6. GROUP BY Clause



- U GROUP BY dijelu naredbe se navodi jedan ili više **atributa** relacija koje su navedene u FROM dijelu naredbe

# GROUP BY Clause

ispit

matBr	nazPredmet	ocjena
100	Matematika	3
100	Programiranje	2
100	Fizika	5
101	Matematika	2
101	Programiranje	2
101	Fizika	3
102	Matematika	4

```
SELECT nazPredmet AS naziv  
      , AVG(ocjena) AS prosjek  
  FROM ispit  
 GROUP BY nazPredmet;
```

naziv	prosjek
Matematika	3
Programiranje	2
Fizika	4

- PostgreSQL dopušta (ali ne i svi ostali SUBP-ovi) u GROUP BY koristiti izraze ili zamjenska imena atributa (*display\_label*)

```
SELECT nazPredmet AS naziv  
      , AVG(ocjena)  
  FROM ispit  
 GROUP BY naziv;
```

# HAVING Clause

ispit

	matBr	nazPredmet	ocjena
100	Matematika	3	
100	Programiranje	2	
100	Fizika	5	
101	Matematika	2	
101	Programiranje	2	
101	Fizika	3	
102	Matematika	4	

```
SELECT nazPredmet AS naziv  
      , AVG(ocjena) AS prosjek  
  FROM ispit  
GROUP BY nazPredmet;
```

naziv	prosjek
Matematika	3
Programiranje	2
Fizika	4

- Kako u rezultatu prikazati samo one grupe koje zadovoljavaju neki uvjet, npr. kako u rezultatu prikazati samo one predmete za koje je prosjek ocjena veći od 2 ?

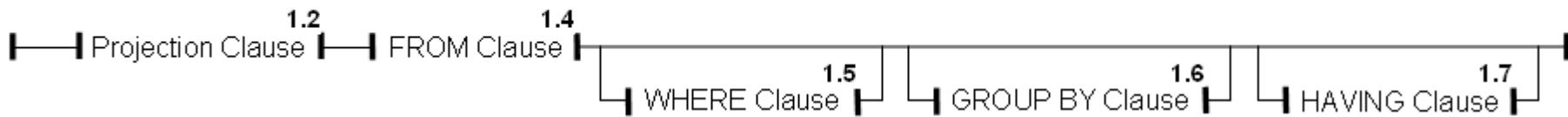
```
SELECT nazPredmet AS naziv  
      , AVG(ocjena) AS prosjek  
  FROM ispit  
GROUP BY nazPredmet  
HAVING AVG(ocjena) > 2;
```

naziv	prosjek
Matematika	3
Fizika	4

# HAVING Clause

- U *Condition* koji se navodi u HAVING dijelu naredbe dopušteno je u izrazima izvan agregatnih funkcija koristiti samo one atribute koji su navedeni u GROUP BY dijelu naredbe

## 1.1. SELECT Options



## 1.7. HAVING Clause



```
SELECT nazPredmet AS naziv
      , AVG(ocjena) AS prosjek
    FROM ispit
   GROUP BY nazPredmet
 HAVING matBr > 104;
```

- U HAVING dijelu SELECT naredbe postavljaju se **uvjeti na grupe** nastale grupiranjem navedenim u GROUP BY dijelu
- za razliku od WHERE dijela u kojem se postavljaju uvjeti na pojedine n-torke

# HAVING Clause

- Primjer: ispisati nazine predmeta i njihove prosječne ocjene, ali samo za one predmete u kojima je najveća ikad dobivena ocjena bila **manja ili jednaka** 4

ispit

matBr	nazPredmet	ocjena
100	Matematika	3
100	Programiranje	2
100	Fizika	5
101	Matematika	2
101	Programiranje	2
101	Fizika	3
102	Matematika	4

```
SELECT nazPredmet AS naziv  
      , AVG(ocjena) AS prosjek  
  FROM ispit  
 GROUP BY nazPredmet  
 HAVING MAX(ocjena) <= 4 ;
```

naziv	prosjek
Matematika	3
Programiranje	2

# HAVING Clause

- U rezultatu se pojavljuju one grupe za koje se navedeni uvjet (*Condition*) izračuna kao logička vrijednost *true*. U rezultatu se ne pojavljuju one grupe za koje se navedeni uvjet izračuna kao logička vrijednost *false* ili *unknown*

ispit	matBr	nazPredmet	ocjena
	100	Matematika	3
	100	Programiranje	2
	100	Fizika	NULL
	101	Matematika	2
	101	Programiranje	2
	101	Fizika	NULL
	102	Matematika	4

```
SELECT nazPredmet AS naziv  
      , AVG(ocjena) AS prosjek  
  FROM ispit  
 GROUP BY nazPredmet  
 HAVING AVG(ocjena) > 2;
```

naziv	prosjek
Matematika	3

# ORDER BY Clause

- Koristi se za sortiranje rezultata upita
- Ispisati podatke o položenim ispitima: poredati ih prema ocjenama, tako da se bliže početku liste nalaze studenti s većim ocjenama. Studente koji imaju međusobno jednake ocjene poredati prema prezimenima, tako da se "manja" prezimena ispisuju prije "većih" prezimena (tj. po abecedi)

```
SELECT *
  FROM poloziliProg
 ORDER BY ocjena DESC
        , prez ASC;
```

poloziliProg

matBr	prez	ocjena
100	Horvat	3
107	Novak	3
102	Horvat	5
101	Kolar	5
103	Kolar	2
104	Horvat	3

matBr	prez	ocjena
102	Horvat	5
101	Kolar	5
104	Horvat	3
100	Horvat	3
107	Novak	3
103	Kolar	2

DESC

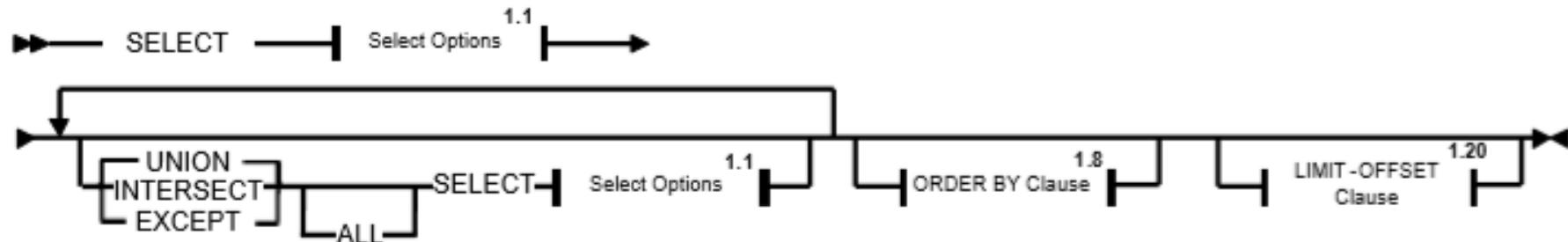
silazno (*descending*)

ASC

uzlazno (*ascending*)

# ***ORDER BY Clause***

## **1. SELECT Statement**



## **1.8 ORDER BY Clause**



- Ako se smjer sortiranja ne navede, podrazumijeva se uzlazni (ASC) smjer sortiranja

# ***ORDER BY Clause***

---

- U ORDER BY dijelu naredbe mogu se koristiti i izrazi koji nisu navedeni u listi za selekciju
- U jednoj SELECT naredbi može se pojaviti samo jedan ORDER BY dio naredbe
  - ako se u SELECT naredbi koristi UNION, INTERSECT ili EXCEPT (skupovske ili multisetovske operacije), ORDER BY se nalazi iza posljednjeg SELECT dijela naredbe
- SQL standard zahtijeva da se NULL vrijednosti pri sortiranju smatraju ili uvijek manjim ili uvijek većim od svih drugih vrijednosti
  - PostgreSQL NULL vrijednosti pri sortiranju uvijek tretira kao da je veća od svih ostalih vrijednosti

# ORDER BY Clause

bodoviMat

mbr	prez	bodLab	bodMI
101	Novak	20	30
103	Horvat	NULL	20
107	Ban	10	80

bodoviProg

mbr	prez	bodLab	bodMI
102	Kolar	12	NULL
104	Novak	30	0

- ispisati podatke o bodovima na lab. vježbama i međuispitu, te ukupnom broju bodova svih studenata, poredati po ukupnom broju bodova: studenti s manjim ukupnim brojem bodova nalaze se bliže početku liste

```
SELECT *, bodLab + bodMI AS ukupno
  FROM bodoviMat
UNION
SELECT *, bodLab + bodMI AS ukupno
  FROM bodoviProg
 ORDER BY ukupno;
```

mbr	prez	bodLab	bodMI	ukupno
104	Novak	30	0	30
101	Novak	20	30	50
107	Ban	10	80	90
102	Kolar	12	NULL	NULL
103	Horvat	NULL	20	NULL

# "Redoslijed obavljanja" dijelova SELECT naredbe

- 
- ```
graph TD; A[1. FROM  
2. WHERE  
3. GROUP BY  
4. HAVING  
5. DISTINCT] --- B[6. UNION  
7. ORDER BY]; B --> C[rezultat]
```
1. FROM
  2. WHERE
  3. GROUP BY
  4. HAVING
  5. DISTINCT
  6. UNION
  7. ORDER BY

rezultat

- **Logički promatrano**, tj. konačni rezultat će sigurno odgovarati rezultatu koji bi se dobio kada bi se operacije obavljale navedenim redoslijedom. Fizički promatrano, upit će se možda izvesti drugačijim redoslijedom.

## Baze podataka

# Predavanja

## 5. SQL – 2. dio



# **Podupiti**

# Podupiti (Subqueries)

| vozilo | sifVoz | nosivost |
|--------|--------|----------|
|        | 101    | 2500     |
|        | 102    | 2000     |
|        | 103    | 800      |
|        | 104    | 1000     |

| teret | sifTeret | tezina |
|-------|----------|--------|
|       | 1001     | 1800   |
|       | 1002     | 1200   |
|       | 1003     | 1000   |

- Ispisati podatke o vozilima čija je nosivost veća od težine najtežeg tereta
- Pogrešan način: prvo obaviti upit kojim se određuje težina najtežeg tereta

```
SELECT MAX(tezina) FROM teret;
```

- Zapamtiti dobiveni rezultat (1800), te napisati novi upit:

```
SELECT *
  FROM vozilo
 WHERE nosivost > 1800;
```

| sifVoz | nosivost |
|--------|----------|
| 101    | 2500     |
| 102    | 2000     |

# Podupiti (Subqueries)

- Ispravan način:

```
SELECT *  
FROM vozilo  
WHERE nosivost > (SELECT MAX(tezina)  
                     FROM teret);
```

podupit

| teret    |        |
|----------|--------|
| sifTeret | tezina |
| 1001     | 1800   |
| 1002     | 1200   |
| 1003     | 1000   |

| vozilo |          |
|--------|----------|
| sifVoz | nosivost |
| 101    | 2500     |
| 102    | 2000     |
| 103    | 800      |
| 104    | 1000     |

1800

2500 > (SELECT MAX ...) → true  
2000 > (SELECT MAX ...) → true  
800 > (SELECT MAX ...) → false  
1000 > (SELECT MAX ...) → false

| sifVoz | nosivost |
|--------|----------|
| 101    | 2500     |
| 102    | 2000     |

- U navedenom primjeru je rezultat podupita jednak za svaku n-torku iz relacije vozilo, stoga je (fizički promatrano) rezultat podupita dovoljno izračunati samo jednom tijekom obavljanja upita

# Podupiti (Subqueries)

---

- podupit je upit koji je ugrađen u neki drugi upit
  - upit u kojeg je podupit ugrađen naziva se vanjski upit (*outer query*)
  - osim izraza **podupit** (*subquery*), u literaturi se također koristi i izraz **ugniježđeni upit** (*nested query*)
- podupit se u vanjski upit može ugraditi
  - u uvjet (*Condition*) u WHERE dijelu vanjskog upita
  - u uvjet (*Condition*) u HAVING dijelu vanjskog upita
  - u listu za selekciju (*SELECT List*) vanjskog upita
- podupit može sadržavati sve do sada spomenute dijelove SELECT naredbe osim ORDER BY dijela naredbe
- u vanjski upit se može ugraditi više podupita, u svaki od podupita se može ugraditi više podupita, itd.

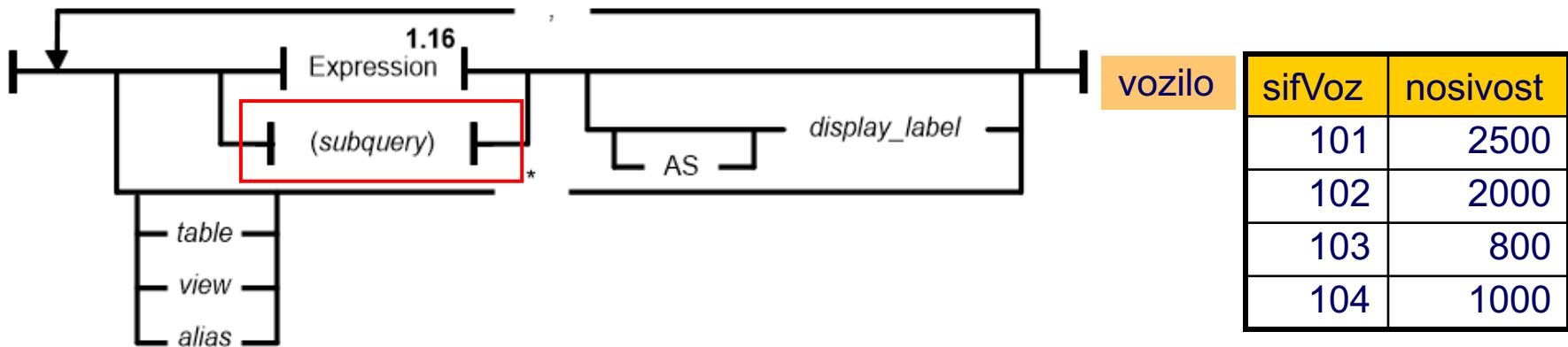
# Skalarni podupit (*Scalar subquery*)

---

- za početak, razmatrat će se najjednostavniji oblik podupita: podupit čiji je rezultat **jedna jednostavna** vrijednost (skalar)
  - npr. podatak tipa: cijeli broj, niz znakova, datum, itd.
- može se reći: rezultat skalarnog podupita je "relacija" stupnja jedan i kardinalnosti jedan
  - vrijednost atributa n-torke dotične "relacije" se u vanjskom upitu koristi kao skalarna vrijednost

# Podupiti u listi za selekciju

## 1.3 SELECT List



- Ispisati podatke o svim vozilima. Uz svako vozilo ispisati podatak o najvećoj težini tereta

```
SELECT *
      , (SELECT MAX(tezina)
          FROM teret) AS maxTezina
   FROM vozilo;
```

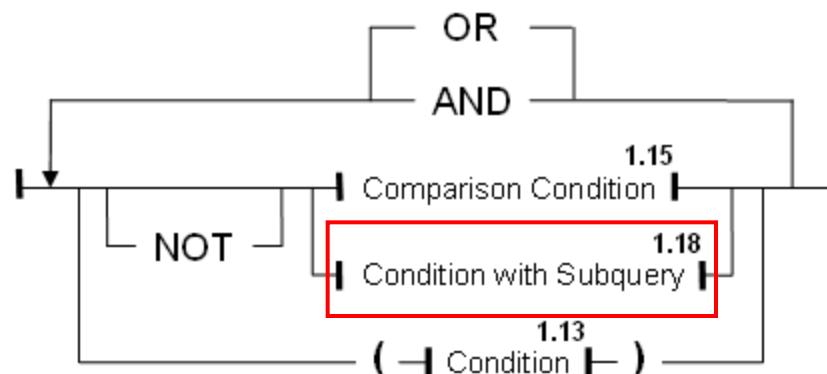
| sifVoz | nosivost | maxTezina |
|--------|----------|-----------|
| 101    | 2500     | 1800      |
| 102    | 2000     | 1800      |
| 103    | 800      | 1800      |
| 104    | 1000     | 1800      |

# Podupiti u WHERE dijelu naredbe

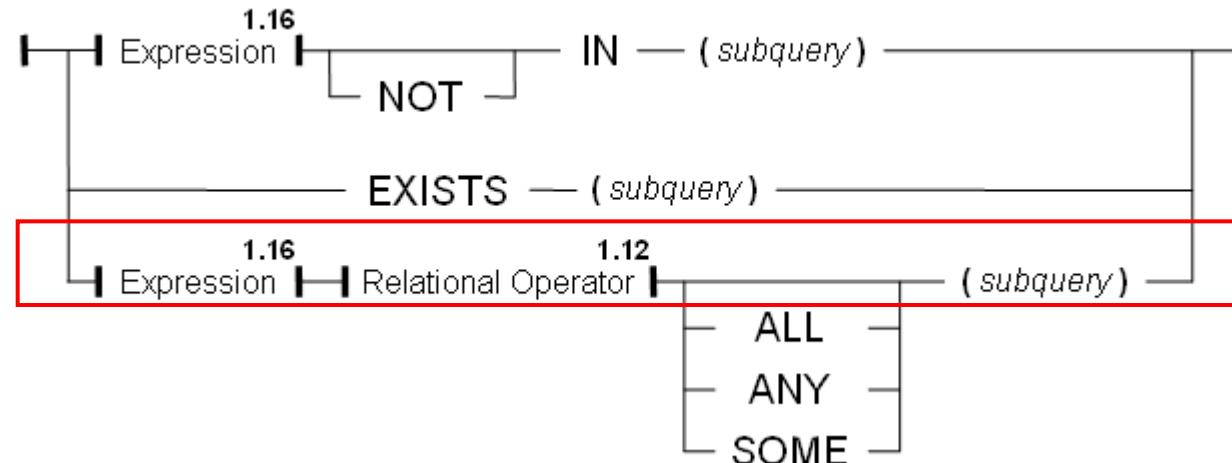
## 1.5. WHERE Clause



## 1.13. Condition



## 1.18. Condition with Subquery



- Ako se u WHERE ili HAVING dijelu naredbe koristi **ovdje označeni oblik uvjeta**, dopušteno je koristiti **isključivo skalarne podupite**

# Podupiti u WHERE dijelu naredbe

| vozilo | sifVoz | nosivost |
|--------|--------|----------|
|        | 101    | 2500     |
|        | 102    | 2000     |
|        | 103    | 800      |
|        | 104    | 1000     |

| teret | sifTeret | tezina |
|-------|----------|--------|
|       | 1001     | 1800   |
|       | 1002     | 1200   |
|       | 1003     | 1000   |

- Ispisati podatke o vozilima čija je nosivost veća od težine najtežeg tereta

```
SELECT *
  FROM vozilo
 WHERE nosivost > (SELECT MAX(tezina)
                      FROM teret);
```

| sifVoz | nosivost |
|--------|----------|
| 101    | 2500     |
| 102    | 2000     |

# Podupiti u WHERE dijelu naredbe

- Ispisati podatke o studentima koji stanuju u mjestu Ludbreg

| stud | mbr | prez   | pbrSt |
|------|-----|--------|-------|
|      | 100 | Horvat | 42230 |
|      | 101 | Kolar  | 21000 |
|      | 102 | Novak  | 42230 |

| mjesto | pbr   | nazMjesto |
|--------|-------|-----------|
|        | 42000 | Varaždin  |
|        | 42230 | Ludbreg   |
|        | 21000 | Split     |

```
SELECT * FROM stud  
WHERE pbrSt = (SELECT pbr FROM mjesto  
                WHERE nazMjesto = 'Ludbreg');
```

42230

- Često se problem može riješiti bez podupita. U konkretnom slučaju, bolje rješenje glasi:

```
SELECT stud.*  
FROM stud  
      JOIN mjesto  
        ON stud.pbrSt = mjesto.pbr  
WHERE nazMjesto = 'Ludbreg';
```

# Podupiti u WHERE dijelu naredbe

| stud | mbr | prez   | pbrSt |
|------|-----|--------|-------|
|      | 100 | Horvat | 42230 |
|      | 101 | Kolar  | 21000 |
|      | 102 | Novak  | 42230 |

| mjesto | pbr   | nazMjesto |
|--------|-------|-----------|
|        | 42000 | Varaždin  |
|        | 42230 | Ludbreg   |
|        | 21000 | Split     |

- Ukoliko podupit čiji bi rezultat trebao biti skalar vrati više od jedne n-torce ili više nego jedan atribut, sustav će dojaviti pogrešku

```
SELECT * FROM stud  
WHERE pbrSt = (SELECT pbr FROM mjesto  
                WHERE nazMjesto LIKE '%r%');
```

Pogreška

- Ukoliko podupit čiji bi rezultat trebao biti skalar ne vrati niti jednu n-torku, dobivena skalarna vrijednost će biti NULL vrijednost

```
SELECT * FROM stud  
WHERE pbrSt = (SELECT pbr FROM mjesto  
                WHERE nazMjesto = 'Grad Split');
```

NULL

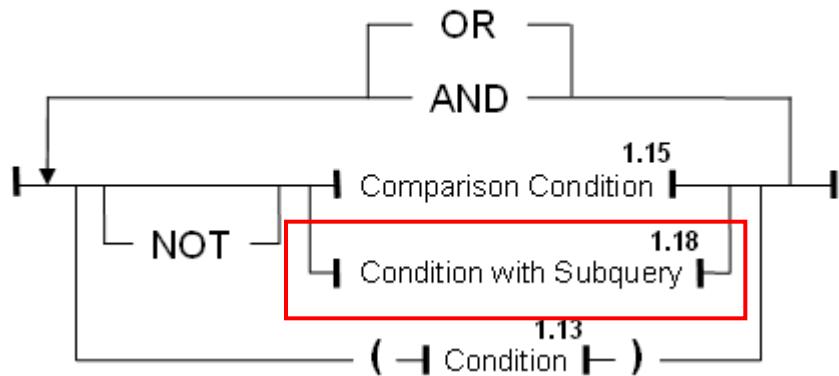
# Podupiti u HAVING dijelu naredbe

## 1.7. HAVING Clause



- Podupiti u HAVING dijelu naredbe koriste se na jednak način kao u WHERE dijelu naredbe

## 1.13. Condition



# Podupiti u HAVING dijelu naredbe

- Primjer:

dvorana

| oznDv | kapacitet |
|-------|-----------|
| D1    | 150       |
| D2    | 200       |
| A201  | 80        |

raspored

| predmet     | oznGr | brojSt |
|-------------|-------|--------|
| Matematika  | M1    | 200    |
| Matematika  | M2    | 50     |
| Matematika  | M3    | 50     |
| Fizika      | F1    | 250    |
| Fizika      | F2    | 150    |
| Elektronika | E1    | 50     |
| Elektronika | E2    | 50     |
| Elektronika | E3    | 100    |
| Elektronika | E4    | 150    |

- Ispisati nazine predmeta za koje se u "D dvoranama" predavanja mogu održati istovremeno za sve grupe. Uz svaki takav predmet ispisati ukupni broj studenata na predmetu

```
SELECT predmet
      , SUM(brojSt) AS ukupnoStud
    FROM raspored
   GROUP BY predmet
  HAVING SUM(brojSt) <=
        (SELECT SUM(kapacitet)
         FROM dvorana
        WHERE oznDv LIKE 'D%');
```

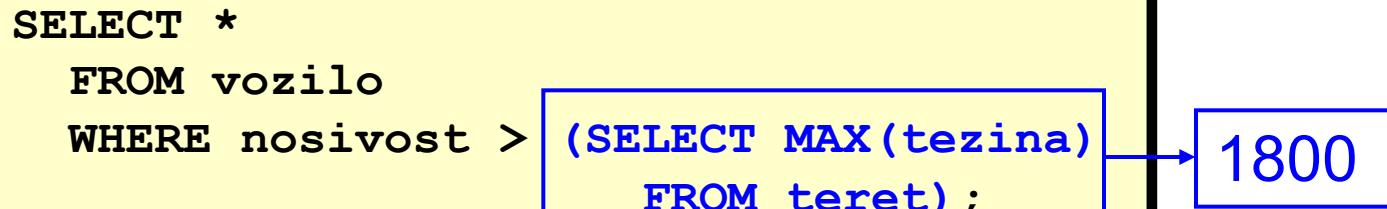
| predmet     | ukupnoStud |
|-------------|------------|
| Matematika  | 300        |
| Elektronika | 350        |

350

# Korelirani podupit (*Correlated subquery*)

- Ako se u podupitu koriste atributi iz vanjskog upita, za podupit i vanjski upit se kaže da su korelirani (*correlated*)
- Za podupit koji je koreliran s vanjskim upitom koristi se naziv korelirani podupit (*correlated subquery*)
- Najčešće se korelirani podupit mora (fizički) izvršiti po jedanput za svaku n-torku iz vanjskog upita
- Sljedeći podupit nije korelirani podupit: u podupitu se ne koriste atributi vanjskog upita. Rezultat podupita ne ovisi o vrijednostima n-torki iz vanjskog podupita, stoga se taj podupit (fizički) treba izvršiti samo jednom tijekom jednog obavljanja vanjskog upita

```
SELECT *
  FROM vozilo
 WHERE nosivost > (SELECT MAX(tezina)
                        FROM teret);
```



# Korelirani podupit (Correlated subquery)

- ispisati podatke o strojevima koji su ukupno korišteni više od dopuštenog broja radnih sati

stroj

| oznStr | dopBrSati |
|--------|-----------|
| S1     | 1000      |
| S2     | 1500      |
| S3     | 500       |

radStroja

| oznStr | godina | brSatiRada |
|--------|--------|------------|
| S1     | 2016   | 700        |
| S1     | 2017   | 100        |
| S1     | 2018   | 300        |
| S2     | 2016   | 700        |
| S2     | 2017   | 500        |
| S3     | 2019   | 600        |

```
SELECT oznStr, dopBrSati
  FROM stroj
 WHERE dopBrSati <
    (SELECT SUM(brSatiRada)
      FROM radStroja
     WHERE oznStr = stroj.oznStr);
```

| oznStr | dopBrSati |
|--------|-----------|
| S1     | 1000      |
| S3     | 500       |

- korelirani podupit: rezultat podupita ovisi o vrijednostima atributa vanjskog upita - za svaku n-torku vanjskog upita dobiva se drugačiji rezultat podupita

# Korelirani podupit (Correlated subquery)

```
SELECT oznStr, dopBrSati  
  FROM stroj  
 WHERE dopBrSati <  
       (SELECT SUM(brSatiRada)  
        FROM radStroja  
       WHERE radStroja.oznStr = stroj.oznStr);
```

| radStroja |        |            |
|-----------|--------|------------|
| oznStr    | godina | brSatiRada |
| S1        | 2016   | 700        |
| S1        | 2017   | 100        |
| S1        | 2018   | 300        |
| S2        | 2016   | 700        |
| S2        | 2017   | 500        |
| S3        | 2019   | 600        |

- upit se (logički promatrano) obavlja na sljedeći način:
  - vanjski upit uzima jednu n-torku iz relacije stroj. Na temelju sadržaja te n-torke i sadržaja relacije radStroja, u podupitu se izračunava suma sati rada dotičnog stroja. Ukoliko je uvjet usporedbe zadovoljen, testirana n-torka se pojavljuje u rezultatu
  - postupak se ponavlja za svaku n-torku relacije stroj

| oznStr | dopBrSati |
|--------|-----------|
| S1     | 1000      |
| S2     | 1500      |
| S3     | 500       |

1000 < (SELECT SUM ... WHERE oznStr = 'S1'  $\Rightarrow$  1100)  $\rightarrow$  true  
1500 < (SELECT SUM ... WHERE oznStr = 'S2'  $\Rightarrow$  1200)  $\rightarrow$  false  
500 < (SELECT SUM ... WHERE oznStr = 'S3'  $\Rightarrow$  600)  $\rightarrow$  true

| oznStr | dopBrSati |
|--------|-----------|
| S1     | 1000      |
| S3     | 500       |

# Korelirani podupit (*Correlated subquery*)

- Primjer: korelirani podupit u listi za selekciju
- uz svaki stroj koji je korišten više od dopuštenog broja sati, ispisati broj sati korištenja stroja

```
SELECT oznStr
      , dopBrSati
      , (SELECT SUM(brSatiRada)
          FROM radStroja
          WHERE radStroja.oznStr = stroj.oznStr) AS koristenSati
    FROM stroj
   WHERE dopBrSati <
        (SELECT SUM(brSatiRada)
          FROM radStroja
          WHERE radStroja.oznStr = stroj.oznStr) ;
```

| oznStr | dopBrSati | koristenSati |
|--------|-----------|--------------|
| S1     | 1000      | 1100         |
| S3     | 500       | 600          |

# Korelirani podupit (*Correlated subquery*)

- Rješenje istog problema bez korištenja podupita:

```
SELECT stroj.oznStr
      , dopBrSati
      , SUM(brSatiRada) AS koristenSati
  FROM stroj
    JOIN radStroja
      ON stroj.oznStr = radStroja.oznStr
 GROUP BY stroj.oznStr, dopBrSati
 HAVING dopBrSati < SUM(brSatiRada) ;
```

| oznStr | dopBrSati | koristenSati |
|--------|-----------|--------------|
| S1     | 1000      | 1100         |
| S3     | 500       | 600          |

# Korelirani podupit (Correlated subquery)

- Primjer: korelirani podupit u HAVING dijelu naredbe

ispit

| mbr | predmet     | akGod | ocj |
|-----|-------------|-------|-----|
| 100 | Matematika  | 2018  | 2   |
| 101 | Matematika  | 2018  | 3   |
| 102 | Matematika  | 2018  | 4   |
| 100 | Fizika      | 2018  | 2   |
| 101 | Fizika      | 2018  | 5   |
| 100 | Elektronika | 2018  | 3   |
| 101 | Elektronika | 2018  | 3   |
| 110 | Matematika  | 2019  | 3   |
| 111 | Matematika  | 2019  | 5   |
| 110 | Fizika      | 2019  | 3   |
| 111 | Fizika      | 2019  | 3   |
| 112 | Fizika      | 2019  | 3   |
| 113 | Fizika      | 2019  | 2   |
| 111 | Elektronika | 2019  | 5   |
| 112 | Elektronika | 2019  | 4   |

- ispisati predmete čija je prosječna ocjena za 2019. godinu veća od prosječne ocjene tog istog predmeta za 2018. godinu

```
SELECT predmet
  FROM ispit AS ispit2019
 WHERE akGod = 2019
 GROUP BY predmet
 HAVING AVG(ocj) >
        (SELECT AVG(ocj)
          FROM ispit
         WHERE predmet = ispit2019.predmet
           AND ispit.akGod = 2018);
```

| predmet     |
|-------------|
| Matematika  |
| Elektronika |

# Korištenje atributa vanjskog upita u podupitu

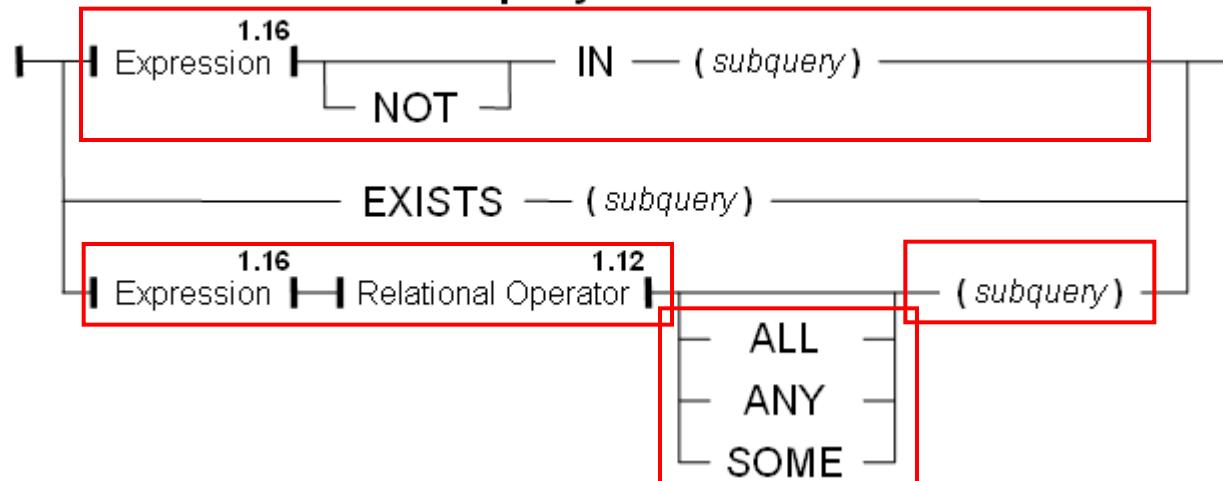
---

- u podupitu se mogu koristiti atributi iz vanjskog upita (obratno ne vrijedi)
- ukoliko se imena atributa (relacija) vanjskog upita podudaraju s imenima atributa (relacija) podupita:
  - ime atributa (relacije) navedeno u podupitu se odnosi na ime atributa (relacije) iz podupita
  - ime atributa (relacije) navedeno u vanjskom upitu se odnosi na ime atributa (relacije) vanjskog upita
- ukoliko je potrebno razriješiti dvosmislenost (npr. ista relacija se koristi u FROM dijelu vanjskog upita i FROM dijelu podupita, a u podupitu se koriste atributi relacije iz vanjskog upita), dovoljno je preimenovati relaciju u vanjskom upitu ili u podupitu
  - prethodni primjer ilustrira takav slučaj

# Jednostupčani podupit (*Single-column subquery*)

- Rezultat jednostupčanog podupita je relacija stupnja jedan, s (moguće) više n-torki
  - također je dopušteno da jednostupčani podupit vratи jednu ili niti jednu n-torku
- jednostupčani podupiti se koriste u WHERE dijelu ili HAVING dijelu vanjskog upita
- jednostupčani podupiti se ne koriste u listi za selekciju

## 1.18. Condition with Subquery



# Jednostupčani podupit (*Single-column subquery*)

- Ispisati podatke o studentima čije je prezime različito od svih prezimena nastavnika

| stud | mbr   | ime    | prez |
|------|-------|--------|------|
| 100  | Ivan  | Horvat |      |
| 101  | Ana   | Kolar  |      |
| 102  | Marko | Novak  |      |

| nastavnik | jmbg  | prez   |
|-----------|-------|--------|
|           | 12345 | Kolar  |
|           | 23456 | Ban    |
|           | 34567 | Pernar |

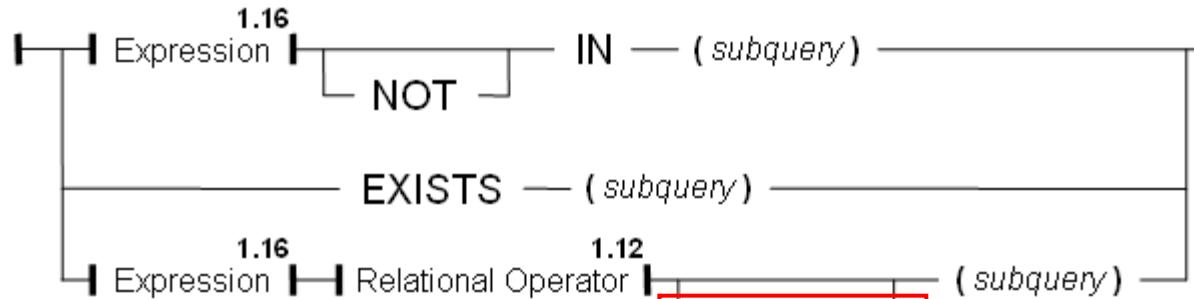
```
SELECT *
  FROM stud
 WHERE prez <> (SELECT prez
                      FROM nastavnik);
```

|        |
|--------|
| Kolar  |
| Ban    |
| Pernar |

- Ovako napisan podupit **nije ispravan** - sustav će dojaviti pogrešku jer pomoću relacijskog operatora `<>` pokušavamo usporediti skalarnu vrijednost i rezultat podupita koji sadrži tri vrijednosti
- Potrebno je koristiti oblike usporedbe s IN, ALL, ANY, SOME

# Jednostupčani podupit (*Single-column subquery*)

## 1.18. Condition with Subquery



|        |
|--------|
| Kolar  |
| Ban    |
| Pernar |

```
SELECT *
  FROM stud
 WHERE prez <> ALL (SELECT prez
                        FROM nastavnik);
```

| mbr | ime   | prez   |
|-----|-------|--------|
| 100 | Ivan  | Horvat |
| 102 | Marko | Novak  |

- uvjet selekcije će biti zadovoljen za one n-torce iz relacije stud čija je vrijednost atributa prez različita od vrijednosti svih članova (multi)skupa dobivenog obavljanjem podupita

# Jednostupčani podupit (*Single-column subquery*)

---

- izraz { < | <= | = | <> | > | >= } **ALL** (podupit)
  - true ako je izraz { < | <= | = | <> | > | >= } od svih vrijednosti dobivenih podupitom
- izraz { < | <= | = | <> | > | >= } **SOME** (podupit)
  - true ako je izraz { < | <= | = | <> | > | >= } od barem jedne vrijednosti dobivene podupitom
- **ANY** je sinonim za **SOME**

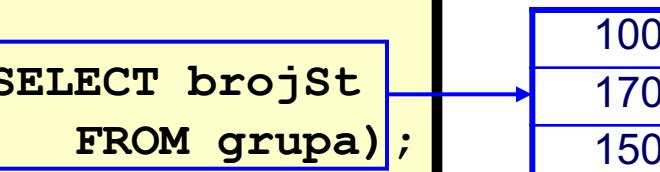
# Jednostupčani podupit (*Single-column subquery*)

- Ispisati podatke o dvoranama čiji je kapacitet veći od broja studenata u barem jednoj od grupa

| dvorana | oznDv | kapacitet |
|---------|-------|-----------|
|         | D1    | 150       |
|         | D2    | 120       |
|         | A201  | 80        |

| grupa | oznGr | brojSt |
|-------|-------|--------|
|       | M1    | 100    |
|       | F1    | 170    |
|       | E4    | 150    |

```
SELECT *
  FROM dvorana
 WHERE kapacitet > SOME (SELECT brojSt
                           FROM grupa);
```

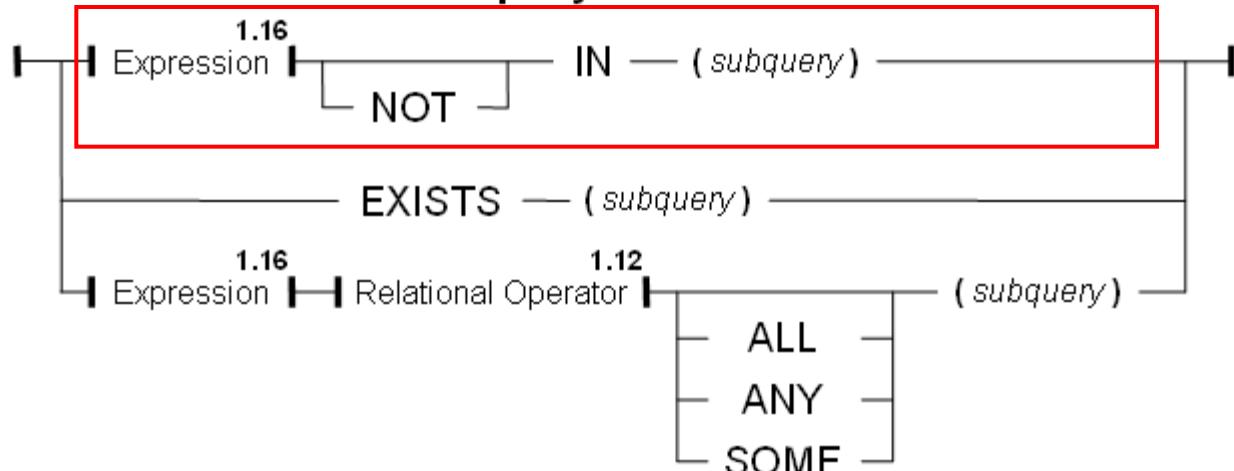


| oznDv | kapacitet |
|-------|-----------|
| D1    | 150       |
| D2    | 120       |

- za vježbu riješiti bez podupita (spajanje, selekcija, projekcija)

# Jednostupčani podupit (*Single-column subquery*)

## 1.18. Condition with Subquery



- izraz **IN** (podupit)
  - *true* ako se u (multi)skupu vrijednosti dobivenih podupitom nalazi barem jedan element jednak vrijednosti izraza
  - ekvivalentno sa: izraz = SOME (podupit)
- izraz **NOT IN** (podupit)
  - *true* ako se u (multi)skupu vrijednosti dobivenih podupitom ne nalazi niti jedan element jednak vrijednosti izraza
  - ekvivalentno sa: izraz <> ALL (podupit)

# Jednostupčani podupit (*Single-column subquery*)

- Ispisati podatke o studentima koji su bilo koji predmet položili tijekom akademske godine 2018.

| stud |        |
|------|--------|
| mbr  | prez   |
| 100  | Horvat |
| 101  | Kolar  |
| 102  | Novak  |
| 103  | Ban    |

| ispit |             |       |        |
|-------|-------------|-------|--------|
| mbr   | predmet     | akGod | ocjena |
| 100   | Matematika  | 2019  | 1      |
| 100   | Elektronika | 2018  | 2      |
| 100   | Fizika      | 2018  | 3      |
| 102   | Elektronika | 2018  | 2      |
| 102   | Fizika      | 2019  | 5      |
| 103   | Elektronika | 2018  | NULL   |
| 103   | Matematika  | 2020  | 4      |

```
SELECT *
  FROM stud
 WHERE mbr IN
      (SELECT mbr
        FROM ispit
       WHERE akGod = 2018
         AND ocjena > 1);
```

|     |
|-----|
| 100 |
| 100 |
| 102 |

| mbr | prez   |
|-----|--------|
| 100 | Horvat |
| 102 | Novak  |

- za vježbu riješiti bez podupita (spajanje, selekcija, projekcija)

# Jednostupčani podupit (*Single-column subquery*)

- Ispisati podatke o studentima koji nisu položili niti jedan predmet tijekom ak. godine 2018.

| stud |        |
|------|--------|
| mbr  | prez   |
| 100  | Horvat |
| 101  | Kolar  |
| 102  | Novak  |
| 103  | Ban    |

| ispit |             |       |        |
|-------|-------------|-------|--------|
| mbr   | predmet     | akGod | ocjena |
| 100   | Matematika  | 2019  | 1      |
| 100   | Elektronika | 2018  | 2      |
| 100   | Fizika      | 2018  | 3      |
| 102   | Elektronika | 2018  | 2      |
| 102   | Fizika      | 2019  | 5      |
| 103   | Elektronika | 2018  | NULL   |
| 103   | Matematika  | 2020  | 4      |

```
SELECT *
  FROM stud
 WHERE mbr NOT IN
    (SELECT mbr
      FROM ispit
     WHERE akGod = 2018
       AND ocjena > 1);
```

|     |
|-----|
| 100 |
| 100 |
| 102 |

| mbr | prez   |
|-----|--------|
| 101 | Horvat |
| 103 | Ban    |

- može li se riješiti bez podupita?

# NULL vrijednosti i jednostupčani podupiti

---

- izraz  $\text{relOp}$  **ALL** (podupit)
  - ako je podupitom dobiven skup vrijednosti  $\{ x_1, x_2, \dots, x_n \}$ , efektivno se uvjet izračunava na sljedeći način:
    - izraz  $\text{relOp } x_1 \text{ AND } \text{izraz relOp } x_2 \text{ AND } \dots \text{ AND } \text{izraz relOp } x_n$
- izraz  $\text{relOp}$  **SOME** (podupit)
  - ako je podupitom dobiven skup vrijednosti  $\{ x_1, x_2, \dots, x_n \}$ , efektivno se uvjet izračunava na sljedeći način:
    - izraz  $\text{relOp } x_1 \text{ OR } \text{izraz relOp } x_2 \text{ OR } \dots \text{ OR } \text{izraz relOp } x_n$
- izraz **IN** (podupit)
  - ekvivalentno sa: izraz = **SOME** (podupit)
- izraz **NOT IN** (podupit)
  - ekvivalentno sa: izraz  $\text{relOp } <> \text{ALL}$  (podupit)

# Primjeri: podupiti i NULL vrijednosti

- naročitu pažnju pri korištenju podupita čiji rezultat može sadržavati NULL vrijednosti treba obratiti na uvjete selekcije oblika:

```
WHERE expression relationalOperator ALL (subquery)
```

```
WHERE expression NOT IN (subquery)
```

ukoliko se u rezultatu ovakvih podupita nalazi makar jedna NULL vrijednost, rezultat izračunavanja uvjeta selekcije nikad neće biti *true*

# Primjeri: podupiti i NULL vrijednosti

| uplata | rbrUpl | iznosUpl |
|--------|--------|----------|
|        | 1      | 50       |
|        | 2      | NULL     |
|        | 3      | 200      |
|        | 4      | 120      |

| isplata | rbrIspl | iznosIspl |
|---------|---------|-----------|
|         | 1       | 80        |
|         | 2       | 100       |
|         | 3       | NULL      |
|         | 4       | 150       |

```
SELECT * FROM uplata
WHERE iznosUpl >
    SOME (SELECT iznosIspl FROM isplata);
```

|      |
|------|
| 80   |
| 100  |
| NULL |
| 150  |

| rbrUpl | iznosUpl |
|--------|----------|
| 1      | 50       |
| 2      | NULL     |
| 3      | 200      |
| 4      | 120      |

50 > 80 OR 50 > 100 OR 50 > NULL OR 50 > 150 → *unknown*  
NULL > 80 OR NULL > 100 OR NULL > NULL OR NULL > 150 → *unknown*  
200 > 80 OR 200 > 100 OR 200 > NULL OR 200 > 150 → *true*  
120 > 80 OR 120 > 100 OR 120 > NULL OR 120 > 150 → *true*

| rbrUpl | iznosUpl |
|--------|----------|
| 3      | 200      |
| 4      | 120      |

# Primjeri: poduputi i NULL vrijednosti

| uplata | rbrUpl | iznosUpl |
|--------|--------|----------|
|        | 1      | 50       |
|        | 2      | NULL     |
|        | 3      | 200      |
|        | 4      | 120      |

| isplata | rbrIspl | iznosIspl |
|---------|---------|-----------|
|         | 1       | 80        |
|         | 2       | 100       |
|         | 3       | NULL      |
|         | 4       | 150       |

```
SELECT * FROM uplata
WHERE iznosUpl >
ALL (SELECT iznosIspl FROM isplata);
```

|      |
|------|
| 80   |
| 100  |
| NULL |
| 150  |

| rbrUpl | iznosUpl |
|--------|----------|
| 1      | 50       |
| 2      | NULL     |
| 3      | 200      |
| 4      | 120      |

50 > 80 AND 50 > 100 AND 50 > NULL AND 50 > 150 → false  
NULL > 80 AND NULL > 100 AND NULL > NULL AND NULL > 150 → unknown  
200 > 80 AND 200 > 100 AND 200 > NULL AND 200 > 150 → unknown  
120 > 80 AND 120 > 100 AND 120 > NULL AND 120 > 150 → false

| rbrUpl | iznosUpl |
|--------|----------|
|        |          |

# Primjeri: podupiti i NULL vrijednosti

| uplata | rbrUpl | iznosUpl |
|--------|--------|----------|
|        | 1      | 50       |
|        | 2      | NULL     |
|        | 3      | 100      |
|        | 4      | 80       |

| isplata | rbrIspl | iznosIspl |
|---------|---------|-----------|
|         | 1       | 80        |
|         | 2       | 100       |
|         | 3       | NULL      |
|         | 4       | 150       |

```
SELECT * FROM uplata  
WHERE iznosUpl IN  
(SELECT iznosIspl FROM isplata);
```

|      |
|------|
| 80   |
| 100  |
| NULL |
| 150  |

| rbrUpl | iznosUpl |
|--------|----------|
| 1      | 50       |
| 2      | NULL     |
| 3      | 100      |
| 4      | 80       |

50 = 80 OR 50 = 100 OR 50 = NULL OR 50 = 150 → *unknown*  
NULL = 80 OR NULL = 100 OR NULL = NULL OR NULL = 150 → *unknown*  
100 = 80 OR 100 = 100 OR 100 = NULL OR 100 = 150 → *true*  
80 = 80 OR 80 = 100 OR 80 = NULL OR 80 = 150 → *true*

| rbrUpl | iznosUpl |
|--------|----------|
| 3      | 100      |
| 4      | 80       |

# Primjeri: podupiti i NULL vrijednosti

| uplata | rbrUpl | iznosUpl |
|--------|--------|----------|
|        | 1      | 50       |
|        | 2      | NULL     |
|        | 3      | 100      |
|        | 4      | 80       |

| isplata | rbrIspl | iznosIspl |
|---------|---------|-----------|
|         | 1       | 80        |
|         | 2       | 100       |
|         | 3       | NULL      |
|         | 4       | 150       |

```
SELECT * FROM uplata
WHERE iznosUpl NOT IN
    (SELECT iznosIspl FROM isplata);
```

|      |
|------|
| 80   |
| 100  |
| NULL |
| 150  |

| rbrUpl | iznosUpl |
|--------|----------|
| 1      | 50       |
| 2      | NULL     |
| 3      | 100      |
| 4      | 80       |

$50 \neq 80 \text{ AND } 50 \neq 100 \text{ AND } 50 \neq \text{NULL} \text{ AND } 50 \neq 150 \rightarrow \text{unknown}$

$\text{NULL} \neq 80 \text{ AND } \text{NULL} \neq 100 \text{ AND } \text{NULL} \neq \text{NULL} \text{ AND } \text{NULL} \neq 150 \rightarrow \text{unknown}$

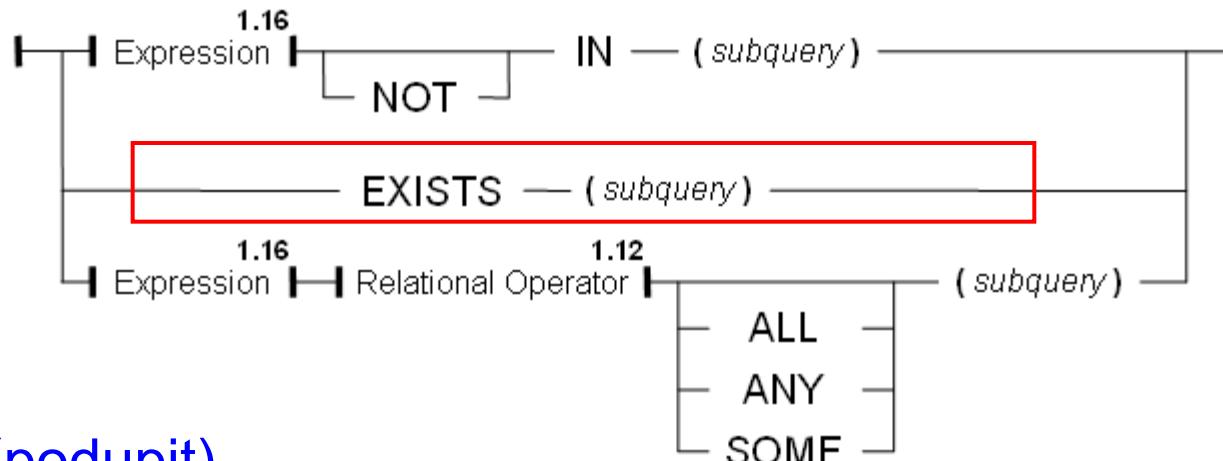
$100 \neq 80 \text{ AND } 100 \neq 100 \text{ AND } 100 \neq \text{NULL} \text{ AND } 100 \neq 150 \rightarrow \text{false}$

$80 \neq 80 \text{ AND } 80 \neq 100 \text{ AND } 80 \neq \text{NULL} \text{ AND } 80 \neq 150 \rightarrow \text{false}$

| rbrUpl | iznosUpl |
|--------|----------|
|        |          |

# Operator EXISTS

## 1.18. Condition with Subquery



- **EXISTS (podupit)**
  - *true* ako rezultat podupita sadrži barem jednu n-torku (bilo kakvu). Pri tome nije važno koliko u dobivenoj n-torci ili n-torkama ima atributa (podupit ne mora biti jednostupčan) niti koje su vrijednosti njihovih atributa
- **NOT EXISTS (podupit)**
  - *true* ako rezultat podupita ne sadrži niti jednu n-torku
- na rezultat vanjskog upita ne utječe eventualna pojava NULL vrijednosti u rezultatu podupita

# Operator EXISTS

- Ispisati podatke o studentima koji u akademskoj godini u kojoj su upisali studij nisu položili niti jedan ispit

| stud | mbr | prez   | akGodUpis |
|------|-----|--------|-----------|
|      | 100 | Horvat | 2018      |
|      | 101 | Kolar  | 2018      |
|      | 102 | Novak  | 2019      |
|      | 103 | Ban    | 2018      |

| ispit | mbr | predmet     | akGod | ocjena |
|-------|-----|-------------|-------|--------|
|       | 100 | Matematika  | 2018  | 1      |
|       | 100 | Fizika      | 2019  | 2      |
|       | 100 | Elektronika | 2020  | 4      |
|       | 100 | Matematika  | 2019  | 3      |
|       | 101 | Matematika  | 2018  | 2      |
|       | 101 | Fizika      | 2018  | 5      |
|       | 102 | Matematika  | 2019  | 4      |

```
SELECT *
  FROM stud
 WHERE NOT EXISTS
    (SELECT * FROM ispit
     WHERE ispit.mbr = stud.mbr
       AND akGod = akGodUpis
       AND ocjena > 1 );
```

ovdje se npr. moglo napisati samo mbr: dobio bi se jednak rezultat

| mbr | prez   | akGodUpis |
|-----|--------|-----------|
| 100 | Horvat | 2018      |
| 103 | Ban    | 2018      |

# Podupiti u HAVING dijelu naredbe

- Svi prikazani oblici podupita mogu se također koristiti i u HAVING dijelu naredbe
- Primjer: ispisati naziv(e) predmeta s najvećim prosjekom

```
SELECT predmet
  FROM ispit
 GROUP BY predmet
 HAVING AVG(ocjena) >= ALL
       (SELECT AVG(ocjena)
  FROM ispit
 GROUP BY predmet);
```

| ispit |             |        |
|-------|-------------|--------|
| mbr   | predmet     | ocjena |
| 100   | Matematika  | 4      |
| 101   | Matematika  | 5      |
| 102   | Matematika  | 3      |
| 100   | Fizika      | 3      |
| 101   | Fizika      | 4      |
| 101   | Elektronika | 5      |
| 102   | Elektronika | 3      |

|     |
|-----|
| 4.0 |
| 3.5 |
| 4.0 |

|             |
|-------------|
| predmet     |
| Matematika  |
| Elektronika |

# Nepotrebno korištenje podupita

- Ispisati podatke o studentima i nazivima mjesta u kojima stanuju. U rezultatu trebaju biti i studenti čije je mjesto stanovanja nepoznato
- **Vrlo loše rješenje:**

```
SELECT student.*  
      , (SELECT nazMjesto FROM mjesto  
           WHERE pbr = pbrStan) AS nazMjesto  
   FROM student;
```

- **Ispravno rješenje:**

```
SELECT student.* , nazMjesto  
   FROM student  
LEFT OUTER JOIN mjesto  
     ON pbrStan = pbr;
```

- Zašto LEFT OUTER JOIN?

# Kada moramo koristiti podupite u SELECT listi

- U situacijama kada u SELECT listu treba uključiti izraz čiji su uvjeti ili način grupiranja različiti od uvjeta u vanjskom upitu.
- Primjer (baza studAdmin): za svaki predmet ispisati šifru, ukupan broj studenata koji su ga do sada položili te ukupan broj upisa predmeta:

```
SELECT sifPredmet,
       (SELECT COUNT(*)
        FROM upisanpredmet up
        WHERE up.sifPredmet = upisanpredmet.sifPredmet
          AND ocjena > 1) položili,
       COUNT(*) upisan
      FROM upisanpredmet
 GROUP BY sifPredmet;
```

# SQL naredbe za izmjenu sadržaja relacije

**INSERT  
DELETE  
UPDATE**

# INSERT

```
CREATE TABLE mjesto (
    pbr          INTEGER NOT NULL
    , nazMjesto  VARCHAR(30) NOT NULL
    , sifZup    SMALLINT
);
```



```
INSERT INTO mjesto
VALUES (42000, 'Varaždin', 7);
```

```
INSERT INTO mjesto
(pbr, sifZup, nazMjesto)
VALUES (52100, 4, 'Pula');
```

```
INSERT INTO mjesto
(pbr, nazMjesto)
VALUES (42230, 'Ludbreg');
```

```
INSERT INTO mjesto
VALUES (10000, 'Zagreb', NULL);
```

| mjesto |           |        |
|--------|-----------|--------|
| pbr    | nazMjesto | sifZup |
| 42000  | Varaždin  | 7      |
| 52100  | Pula      | 4      |
| 42230  | Ludbreg   | NULL   |
| 10000  | Zagreb    | NULL   |

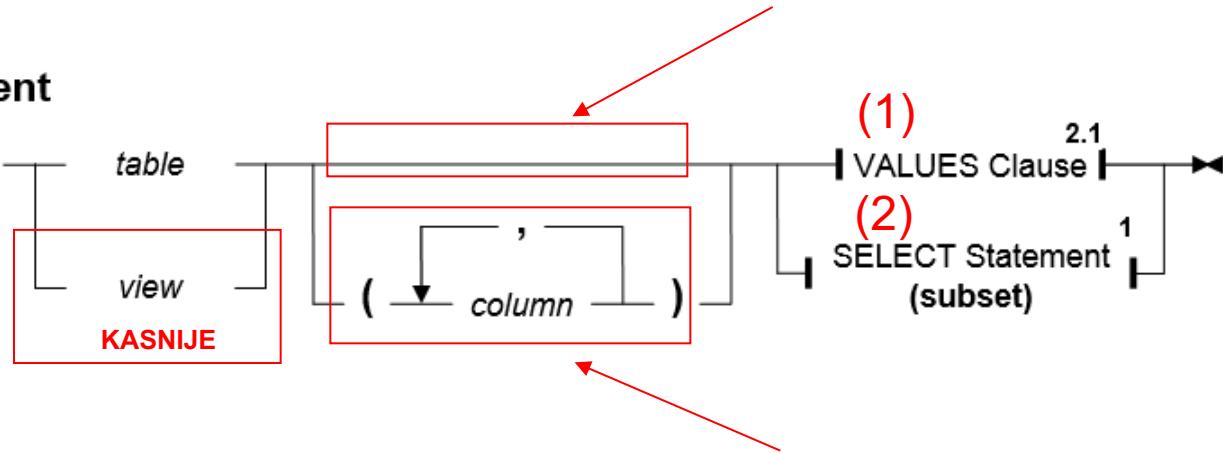
# INSERT

- INSERT naredba se koristi za unos jedne n-torke (1) ili skupa n-torki (2) u relaciju *table*

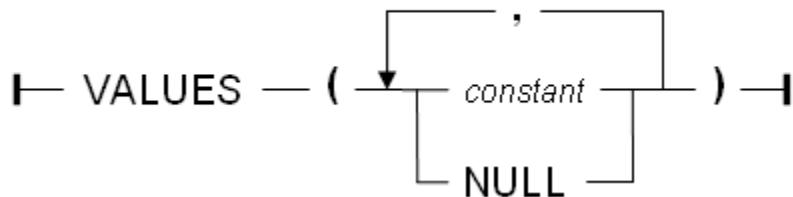
bez navedene liste atributa

## 2. INSERT Statement

► INSERT INTO



## 2.1. VALUES Clause



# INSERT (1), bez navedene liste atributa – antiprimjer!

- U relaciju se upisuje jedna n-torka pri čemu **vrijednosti svih** atributa n-torke **moraju biti navedene redoslijedom** kojim su atributi navedeni u CREATE TABLE naredbi kojom je relacija definirana

```
CREATE TABLE stud (
    mbr      INTEGER
,  ime      VARCHAR(30)
,  prez     VARCHAR(50)
,  stipend  VARCHAR(1) DEFAULT 'N'
,  pbrStan  INTEGER DEFAULT 10000
);
```

stud

|     |     |      |         |         |
|-----|-----|------|---------|---------|
| mbr | ime | prez | stipend | pbrStan |
|-----|-----|------|---------|---------|

```
INSERT INTO stud VALUES (
    100
,  'Ivan'
,  'Horvat'
,  'N'
,  NULL
);
```

stud

|     |      |        |         |         |
|-----|------|--------|---------|---------|
| mbr | ime  | prez   | stipend | pbrStan |
| 100 | Ivan | Horvat | N       | NULL    |

Znači: ako se prilikom unosa nove n-torke ne navede vrijednost za atribut stipend, sustav će kao vrijednost tog atributa postaviti vrijednost N

# INSERT (1), bez navedene liste atributa

---

- Opisani oblik INSERT naredbe ima neke nedostatke:
  - u slučaju kada se u relaciju upisuje n-torka čiji relativno veliki broj atributa treba postaviti na NULL vrijednost ili prepostavljenu vrijednost (*default value*)
    - ipak se moraju navesti vrijednosti **svih** atributa
  - u slučaju kada se relacijska shema promijeni (npr. promijeni se "redoslijed" atributa)
    - budući da vrijednosti atributa moraju biti navedene redoslijedom atributa u CREATE TABLE naredbi, INSERT naredbe koje su napisane prije promjene relacijske sheme više neće biti ispravne
- zbog navedenih nedostataka, preporuča se korištenje oblika INSERT naredbe s navedenom listom atributa

# INSERT (1), uz navedenu listu atributa

- U prvom dijelu naredbe navode se imena atributa (i njihov redoslijed) čije će vrijednosti (u odgovarajućem redoslijedu) biti navedene u drugom dijelu naredbe
  - atributi čije vrijednosti nisu navedene u INSERT naredbi, postavljaju se na prepostavljenu (*default*) vrijednost (ukoliko je takva definirana u CREATE TABLE naredbi) ili na NULL vrijednost

```
INSERT INTO stud (
    prez
    , mbr
    , pbrStan)
VALUES (
    'Kolar'
    , 101
    , 10000);
```

| stud |      |        |         |         |
|------|------|--------|---------|---------|
| mbr  | ime  | prez   | stipend | pbrStan |
| 100  | Ivan | Horvat | N       | NULL    |

→

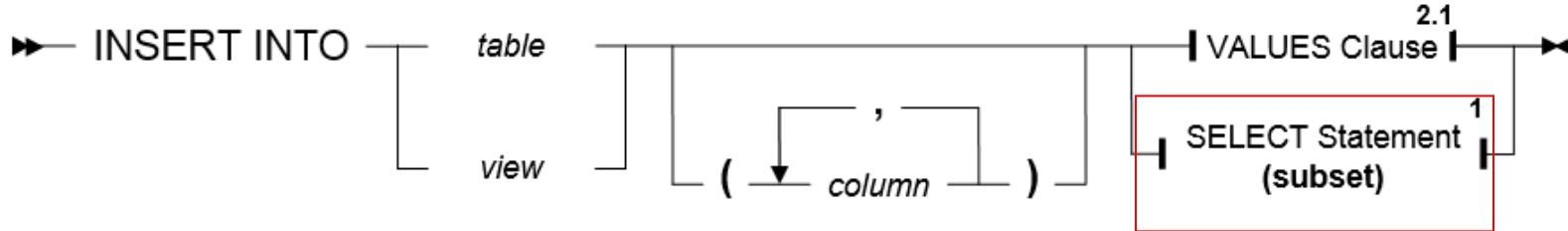
| stud |      |        |         |         |
|------|------|--------|---------|---------|
| mbr  | ime  | prez   | stipend | pbrStan |
| 100  | Ivan | Horvat | N       | NULL    |
| 101  | NULL | Kolar  | N       | 10000   |

ime?

stipend?

# INSERT (2)

## 2. INSERT Statement



- u relaciju `table` upisuju se n-torke dobivene dijelom INSERT naredbe koji je sličan SELECT naredbi - u sintaksnom dijagramu taj je dio naredbe označen sa `SELECT Statement (subset)`

# INSERT (2), bez navedene liste atributa

- u relaciju polozioFiz upisati podatke o studentima koji su položili predmet Fizika

| stud |      |        |       |
|------|------|--------|-------|
| mbr  | ime  | prez   | pbrSt |
| 102  | Ana  | Novak  | 10000 |
| 105  | Rudi | Kolar  | 21000 |
| 107  | Jura | Horvat | 41000 |
| 109  | Tea  | Ban    | 51000 |

| ispit |             |        |
|-------|-------------|--------|
| mbr   | predmet     | ocjena |
| 102   | Elektronika | 1      |
| 102   | Matematika  | 3      |
| 105   | Fizika      | 3      |
| 105   | Matematika  | 4      |
| 107   | Fizika      | 1      |
| 109   | Fizika      | 5      |

| polozioFiz |       |        |
|------------|-------|--------|
| mbr        | imeSt | prezSt |

```
INSERT INTO polozioFiz
SELECT stud.mbr, ime, prez
FROM stud, ispit
WHERE stud.mbr = ispit.mbr
AND predmet = 'Fizika'
AND ocjena > 1;
```

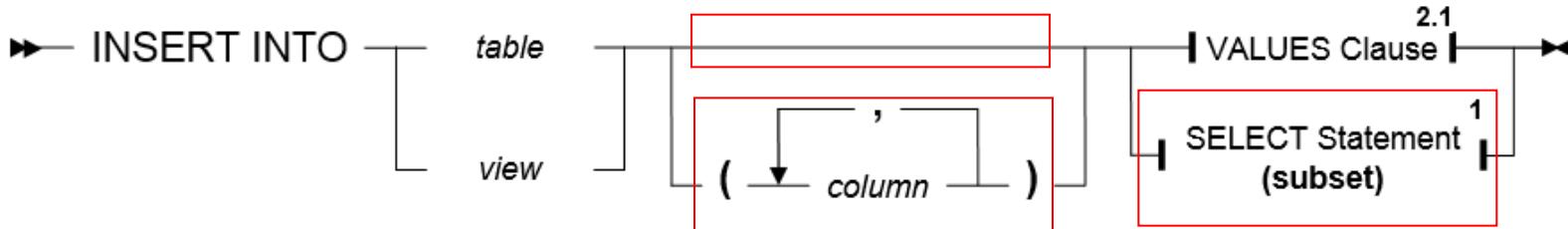
imena atributa u SELECT listi  
ne moraju odgovarati imenima  
atributa u relaciji polozioFiz



| polozioFiz |       |        |
|------------|-------|--------|
| mbr        | imeSt | prezSt |
| 105        | Rudi  | Kolar  |
| 109        | Tea   | Ban    |

# INSERT (2)

## 2. INSERT Statement



- jednako kao kod oblika INSERT naredbe za unos jedne n-torke u relaciju
  - bez navedene liste imena atributa, broj i redoslijed atributa u n-torkama dobivenim obavljanjem SELECT dijela naredbe mora odgovarati broju i redoslijedu atributa u CREATE TABLE naredbi (priказано у prethodnom primjeru)
  - uz navedenu listu imena atributa, atributi čije vrijednosti nisu navedene u INSERT naredbi, postavljaju se na prepostavljenu (*default*) vrijednost (ukoliko je takva definirana u CREATE TABLE naredbi) ili na NULL vrijednost

# INSERT i SERIAL

```
CREATE TABLE predmet (
    sifPred  SERIAL
, nazPred  VARCHAR(100)
, ECTSBody NUMERIC(4,1));
```

```
INSERT INTO predmet (nazPred, ECTSBody)
VALUES ('Baze podataka', 6.0);
```

```
INSERT INTO predmet (nazPred, ECTSBody)
VALUES ('Programiranje', 7.0);
INSERT INTO predmet (nazPred, ECTSBody)
VALUES ('Badminton', 1.5);
```

| predmet |         |          |
|---------|---------|----------|
| sifPred | nazPred | ECTSBody |
|         |         |          |

| sifPred | nazPred       | ECTSBody |
|---------|---------------|----------|
| 1       | Baze podataka | 6.0      |

- SERIAL je autoinkrementalni INTEGER
  - raspon je: [1, 2147483647]
- uvijek se generira prvi sljedeći broj (interni generator o tome vodi računa)

# DELETE

- brisanje n-torki iz relacije

mjesto

| pbr   | nazMjesto | sifZup |
|-------|-----------|--------|
| 42230 | Ludbreg   | 7      |
| 42000 | VARAŽDIN  | 7      |
| 52100 | Pula      | 4      |

```
DELETE FROM mjesto  
WHERE sifZup = 7;
```

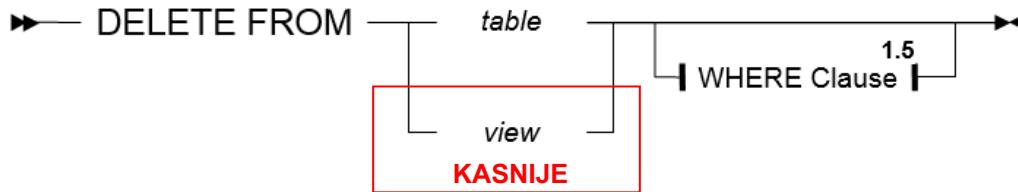


mjesto

| pbr   | nazMjesto | sifZup |
|-------|-----------|--------|
| 52100 | Pula      | 4      |

# DELETE

## 4. DELETE Statement



## 1.5. WHERE Clause



- DELETE naredba briše one n-torke relacije *table* za koje se uvjet naveden u WHERE dijelu naredbe izračuna kao *true*
  - ako se WHERE dio naredbe ne navede, iz relacije *table* se brišu **sve** n-torke
- u WHERE dijelu naredbe mogu se koristiti svi oblici uvjeta (*Condition*) koji se koriste u WHERE dijelu SELECT naredbe
- neki sustavi (npr. IBM Informix) ne dopuštaju korištenje relacije *table* u koreliranom podupitu, a neki (npr. PostgreSQL, SQL Server, Oracle) dopuštaju.

```
DELETE FROM mjesto
WHERE pbr IN (SELECT pbr FROM mjesto m1
               WHERE m1.nazMjesto = mjesto.nazMjesto);
```

# DELETE

- iz relacije polozioFiz obrisati n-torce onih studenata koji (sudeći prema podacima iz relacije ispit) nisu položili predmet Fizika

stud

| mbr | ime  | prez   | pbrSt |
|-----|------|--------|-------|
| 102 | Ana  | Novak  | 10000 |
| 105 | Rudi | Kolar  | 21000 |
| 107 | Jura | Horvat | 41000 |
| 109 | Tea  | Ban    | 51000 |

ispit

| mbr | predmet     | ocjena |
|-----|-------------|--------|
| 102 | Elektronika | 1      |
| 102 | Matematika  | 3      |
| 105 | Fizika      | 3      |
| 105 | Matematika  | 4      |
| 107 | Fizika      | 1      |
| 109 | Fizika      | 5      |

polozioFiz

| mbr | imeSt | prezSt |
|-----|-------|--------|
| 102 | Ana   | Novak  |
| 105 | Rudi  | Kolar  |
| 107 | Jura  | Horvat |
| 109 | Tea   | Ban    |
| 111 | Ivan  | Polak  |

```
DELETE FROM polozioFiz
WHERE mbr NOT IN
(SELECT mbr
FROM ispit
WHERE predmet = 'Fizika'
AND ocjena > 1);
```



polozioFiz

| mbr | imeSt | prezSt |
|-----|-------|--------|
| 105 | Rudi  | Kolar  |
| 109 | Tea   | Ban    |

# INSERT, DELETE i SERIAL

```
CREATE TABLE predmet (
    sifPred  SERIAL
, nazPred  VARCHAR(100)
, ECTSBody NUMERIC(4,1));
```

predmet

| sifPred | nazPred       | ECTSBody |
|---------|---------------|----------|
| 1       | Baze podataka | 6.0      |
| 2       | Programiranje | 7.0      |
| 3       | Badminton     | 1.5      |

```
DELETE FROM predmet
WHERE sifPred = 3;
INSERT INTO predmet (nazPred, ECTSBody)
VALUES ('Badminton', 1.5);
```

predmet

| sifPred | nazPred       | ECTSBody |
|---------|---------------|----------|
| 1       | Baze podataka | 6.0      |
| 2       | Programiranje | 7.0      |
| 4       | Badminton     | 1.5      |

- Eventualni slobodni brojevi (identifikatori, šifre) nastali brisanjem se ne koriste ponovno

# UPDATE

- Svim studentima, čija je ocjena manja od 5, ocjenu uvećati za 1, a broj bodova postaviti na NULL

```
UPDATE ispit
SET ocjena = ocjena + 1
, brBod = NULL
WHERE ocjena < 5;
```

ili

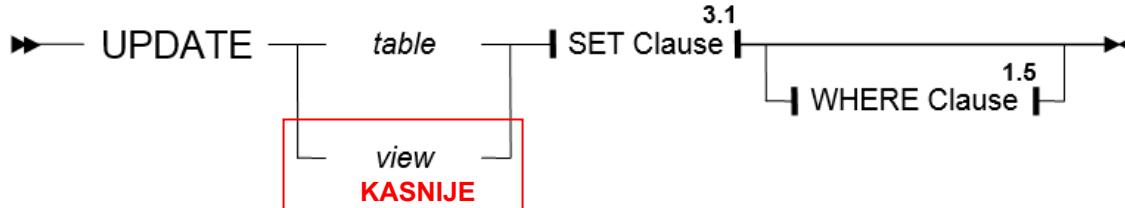
```
UPDATE ispit
SET (ocjena, brBod) = (ocjena + 1, NULL)
WHERE ocjena < 5;
```

| ispit |       |        |
|-------|-------|--------|
| mbrSt | brBod | ocjena |
| 1001  | 200   | 5      |
| 1002  | 150   | 4      |
| 1003  | 130   | 3      |
| 1004  | 110   | 2      |

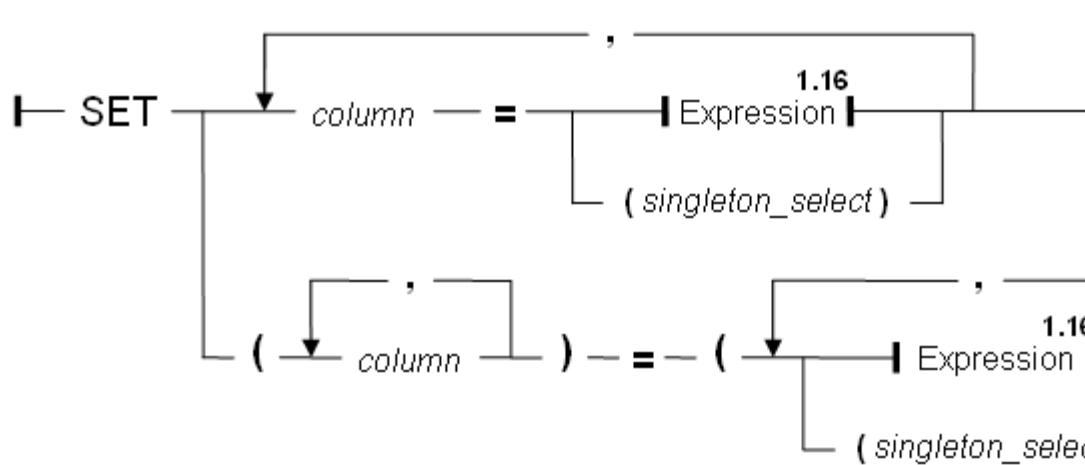
| ispit |       |        |
|-------|-------|--------|
| mbrSt | brBod | ocjena |
| 1001  | 200   | 5      |
| 1002  | NULL  | 5      |
| 1003  | NULL  | 4      |
| 1004  | NULL  | 3      |

# UPDATE

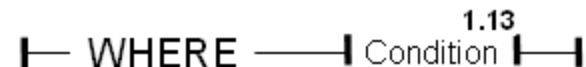
## 3. UPDATE Statement



### 3.1. SET Clause



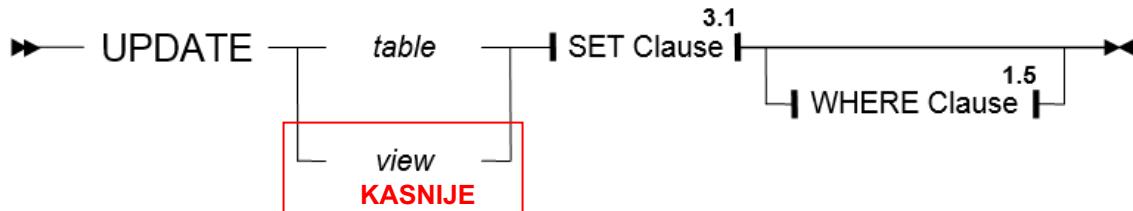
### 1.5. WHERE Clause



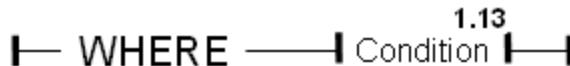
- UPDATE naredba mijenja vrijednosti atributa postojećih n-torki relacije *table*. U WHERE dijelu naredbe opisuje se koje n-torke će biti promijenjene; u SET dijelu naredbe opisuje se koji će atributi biti postavljeni na koje vrijednosti

# UPDATE

## 3. UPDATE Statement



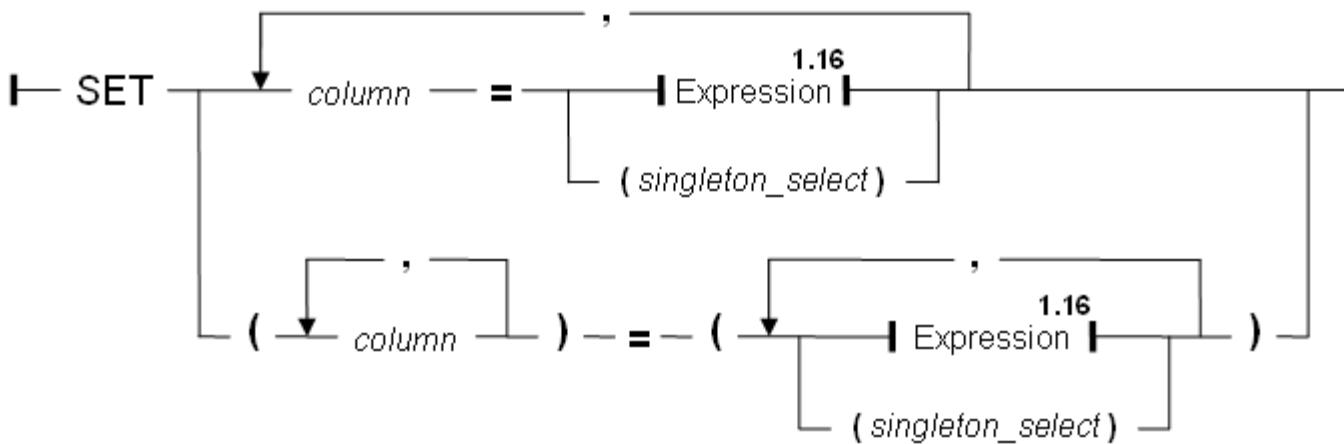
### 1.5. WHERE Clause



- UPDATE naredba mijenja vrijednosti atributa onih n-torki relacije *table* za koje se uvjet naveden u WHERE dijelu naredbe izračuna kao *true*
  - ako se WHERE dio naredbe ne navede, mijenjaju se vrijednosti atributa u svim n-torkama relacije *table*
  - u WHERE dijelu naredbe mogu se koristiti svi oblici uvjeta (*Condition*) koji se koriste u WHERE dijelu SELECT naredbe,
  - neki sustavi (npr. IBM Informix) ne dopuštaju korištenje relacije *table* u koreliranom podupitu, a neki (npr. SQL Server, Oracle) dopuštaju.

# UPDATE

## 3.1. SET Clause



- SET dio naredbe određuje nove vrijednosti atributa. Nova vrijednost atributa može biti definirana kao:
  - *Expression* - konstante, NULL, atributi iz relacije *table*, binarni i unarni operatori, funkcije, uvjetni izraz, zagrade
  - *singleton\_select* - jednako kao skalarni podupit (korelirani ili nekorelirani)
    - relaciju *table* nije dopušteno koristiti u FROM dijelu

# UPDATE

- Bodove na međuispitu uvećati za 10 bodova onim studentima koji time ne bi stekli ukupno (bodLab+bodMI) više od 100 bodova

| bodovi | mbr    | prez | bodLab | bodMI |
|--------|--------|------|--------|-------|
| 101    | Novak  |      | 30     | 55    |
| 102    | Polak  |      | NULL   | 20    |
| 103    | Kolar  |      | 20     | 10    |
| 104    | Ban    |      | 10     | 80    |
| 105    | Horvat |      | 50     | 49    |
| 106    | Seljan |      | 10     | NULL  |

```
UPDATE bodovi  
SET bodMI = bodMI + 10  
WHERE bodLab + bodMI + 10 <= 100;
```



| bodovi | mbr    | prez | bodLab | bodMI |
|--------|--------|------|--------|-------|
| 101    | Novak  |      | 30     | 65    |
| 102    | Polak  |      | NULL   | 20    |
| 103    | Kolar  |      | 20     | 20    |
| 104    | Ban    |      | 10     | 90    |
| 105    | Horvat |      | 50     | 49    |
| 106    | Seljan |      | 10     | NULL  |

# UPDATE

- Bodove na međuispitu uvećati za 2 boda studentima koji time ne bi stekli više od 50 bodova za međuispit, bodove za laboratorij povećati za 3 boda onim studentima koji time ne bi stekli ukupno više od 40 bodova za laboratorij

```
UPDATE bodovi SET  
    bodMI = CASE  
        WHEN bodMI + 2 <= 50  
            THEN bodMI + 2  
        ELSE bodMI  
    END  
, bodLab = CASE  
    WHEN bodLab + 3 <= 40  
        THEN bodLab + 3  
    ELSE bodLab  
END ;
```

bodovi

| mbr | prez   | bodLab | bodMI |
|-----|--------|--------|-------|
| 101 | Novak  | 30     | 55    |
| 102 | Polak  | NULL   | 20    |
| 103 | Kolar  | 20     | 10    |
| 104 | Ban    | 10     | 80    |
| 105 | Horvat | 50     | 49    |
| 106 | Seljan | 10     | NULL  |

bodovi

| mbr | prez   | bodLab | bodMI |
|-----|--------|--------|-------|
| 101 | Novak  | 33     | 55    |
| 102 | Polak  | NULL   | 22    |
| 103 | Kolar  | 23     | 12    |
| 104 | Ban    | 13     | 80    |
| 105 | Horvat | 50     | 49    |
| 106 | Seljan | 13     | NULL  |

# UPDATE

- U sintaksnom dijagramu za *SET Clause* može se vidjeti da postoje dva slična, jednako vrijedna oblika:
  - SET atribut1=vrijednost1, atribut2=vrijednost2, ...
  - SET (atribut1, atribut2, ...)=(vrijednost1, vrijednost2, ...)
- Prethodna UPDATE naredba se mogla napisati na sljedeći način:

```
UPDATE bodovi SET
    (bodMI, bodLab) =
    (CASE
        WHEN bodMI + 2 <= 50
            THEN bodMI + 2
        ELSE bodMI
    END
    , CASE
        WHEN bodLab + 3 <= 40
            THEN bodLab + 3
        ELSE bodLab
    END
) ;
```

# UPDATE

- U relaciji mjesto zamijeniti stare poštanske brojeve i nazine (samo onim mjestima za koje postoje opisani novi poštanski brojevi i nazivi)

| mjesto |           |
|--------|-----------|
| pbr    | nazMjesto |
| 41000  | ZAGREB    |
| 51400  | PAZIN     |
| 52000  | PULA      |
| 54000  | OSIJEK    |

| konverzija |         |           |
|------------|---------|-----------|
| stariPbr   | noviPbr | noviNaziv |
| 51400      | 52000   | Pazin     |
| 52000      | 52100   | Pula      |
| 54000      | 31000   | Osijek    |

```
UPDATE mjesto
SET (pbr, nazMjesto) = (
    (SELECT noviPbr
     FROM konverzija
     WHERE konverzija.stariPbr = mjesto.pbr
    )
    , (SELECT noviNaziv
       FROM konverzija
       WHERE konverzija.stariPbr = mjesto.pbr
      )
)
WHERE pbr IN (SELECT stariPbr
              FROM konverzija);
```

| mjesto |           |
|--------|-----------|
| pbr    | nazMjesto |
| 41000  | ZAGREB    |
| 52000  | Pazin     |
| 52100  | Pula      |
| 31000  | Osijek    |

# UPDATE

- Nastavnicima koji su na ispitima iz BP podijelili (prosječno) najveće ocjene, povećati plaću za 20000

```
UPDATE nast SET placa = placa + 20000
WHERE sifNast IN (
    SELECT sifNast
    FROM ispitBP
    GROUP BY sifNast
    HAVING AVG(ocjena) >= ALL
        (SELECT AVG(ocjena)
        FROM ispitBP
        GROUP BY sifNast)
);
```

| nast    |        |       |
|---------|--------|-------|
| sifNast | prez   | placa |
| 101     | Novak  | 52000 |
| 102     | Kolar  | 55000 |
| 103     | Horvat | 48000 |
| 104     | Ban    | 57000 |

| ispitBP |        |         |
|---------|--------|---------|
| mbrSt   | ocjena | sifNast |
| 1001    | 5      | 101     |
| 1002    | 4      | 101     |
| 1003    | 3      | 101     |
| 1004    | 2      | 102     |
| 1005    | 4      | 102     |
| 1006    | 5      | 103     |
| 1007    | 5      | 103     |
| 1008    | 2      | 103     |
| 1009    | 3      | 104     |

| nast    |        |       |
|---------|--------|-------|
| sifNast | prez   | placa |
| 101     | Novak  | 72000 |
| 102     | Kolar  | 55000 |
| 103     | Horvat | 68000 |
| 104     | Ban    | 57000 |

## Baze podataka

# Predavanja

## 6. Oblikovanje sheme relacijske baze podataka (1. dio)

Ožujak, 2020.



# Oblikovanje sheme baze podataka

---

- cilj: oblikovati shemu baze podataka s dobrim svojstvima
- karakteristike loše koncipirane sheme baze podataka:
  - redundancija (čije su posljedice):
    - anomalija unosa
    - anomalija izmjene
    - anomalija brisanja
  - pojava lažnih n-torki

# Primjer loše koncipirane sheme baze podataka

- Prodavaonice šalju svoje narudžbe proizvođaču:

Konzum-7

Ilica 20

10 000 Zagreb

Kraš

Ravnice bb

10 000 Zagreb

**Narudžba**

br. 13/25

**datum: 1.5.2018**

Molimo isporučite nam **1200** komada proizvoda **Napolitanke** (šifra **129**) i **2000** komada proizvoda **Petit beurre** (šifra **139**)

Spar-28

Bolska 7

**21 000 Split**

Kraš

Ravnice bb

10 000 Zagreb

**Narudžba**

br. 43-21

**datum: 7.2.2018**

Molimo isporučite nam **1200** komada proizvoda **Napolitanke** (šifra **129**) i **1800** komada proizvoda **Domaćica** (šifra **221**)

Konzum-7

Ilica 20

**10 000 Zagreb**

Kraš

Ravnice bb

10 000 Zagreb

**Narudžba**

br. 41/56

**datum: 4.2.2019**

Molimo isporučite nam **1100** komada proizvoda **Napolitanke** (šifra **129**)

- proizvođač želi pohraniti podatke o narudžbama u svoju bazu podataka.  
Svi podaci se pohranjuju u relaciju **narudzbaArtikla**

**narudzbaArtikla**

**nazProd**

**pbr**

**nazMjesto**

**adresa**

**brNar**

**datNar**

**sifArtikl**

**nazArtikl**

**kolicina**

# Neracionalno korištenje prostora za pohranu

- Sadržaj relacije nakon unosa podataka iz prispjelih narudžbi:

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- na više mjesta se ponavlja isti (redundantan) podatak:
  - Konzum-7 je prodavaonica u Zagrebu
  - adresa prodavaonice Konzum-7 je Ilica 20
  - naziv artikla sa šifrom 129 je Napolitanke
  - naziv mjesta s poštanskim brojem 10000 je Zagreb
  - datum narudžbe s brojem 13/25 je 1.5.2018
  - itd.

# Anomalija unosa

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- ne mogu se unijeti podaci o artiklima koje nitko nije naručio
- ne mogu se unijeti podaci o prodavaonicama koje ništa nisu naručile
- ...
- svaki put kad se unosi novi podatak o narudžbi nekog artikla, mora se ponovno upisivati i naziv i mjesto i adresa prodavaonice koja taj artikl naručuje
  - pri tome treba paziti da se podaci za istu prodavaonicu uvijek jednako unesu da bi se zadržala konzistentnost podataka

# Anomalija izmjena

- ako neka prodavaonica promijeni adresu, promjenu adrese potrebno je obaviti na više mesta da bi se zadržala konzistentnost podataka

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- npr. prodavaonica Konzum-7 se preseli jedan kućni broj dalje od centra

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 22 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 22 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 22 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

# Anomalija brisanja

- brisanjem svih narudžbi za neki artikl gube se podaci o artiklu

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- npr. ako se obriše posljednja n-torka o narudžbama artikla Domaćica, podatke o tom artiklu više nećemo imati u bazi podataka

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

# Pokušaj (neuspješni) popravka sheme baze podataka

- Podaci o narudžbama će se pohranjivati u dvije relacije

$\text{narudzba} = \pi_{\text{nazProd}, \text{pbr}, \text{nazMjesto}, \text{adresa}, \text{brNar}, \text{datNar}, \text{sifArtikl}}(\text{narudzbaArtikla})$

$\text{artikl} = \pi_{\text{sifArtikl}, \text{nazArtikl}, \text{kolicina}}(\text{narudzbaArtikla})$

| narudzba | nazProd | pbr    | nazMjesto | adresa | brNar    | datNar | sifArtikl |
|----------|---------|--------|-----------|--------|----------|--------|-----------|
| Konzum-7 | 10000   | Zagreb | Ilica 20  | 13/25  | 1.5.2018 | 129    |           |
| Konzum-7 | 10000   | Zagreb | Ilica 20  | 13/25  | 1.5.2018 | 139    |           |
| Spar-28  | 21000   | Split  | Bolska 7  | 43-21  | 7.2.2018 | 129    |           |
| Spar-28  | 21000   | Split  | Bolska 7  | 43-21  | 7.2.2018 | 221    |           |
| Konzum-7 | 10000   | Zagreb | Ilica 20  | 41/56  | 4.2.2019 | 129    |           |

| artikl | sifArtikl | nazArtikl    | kolicina |
|--------|-----------|--------------|----------|
|        | 129       | Napolitanke  | 1200     |
|        | 139       | Petit beurre | 2000     |
|        | 221       | Domaćica     | 1800     |
|        | 129       | Napolitanke  | 1100     |

- Ovakva shema baze podataka uzrokovat će pojavu lažnih (*spurious*) n-torki
- dolazi do gubitka informacije!

# Pojava lažnih n-torki

- Obavljanjem operacije narudzba  $\bowtie$  artikl dobije se **više n-torki** nego ih je bilo u relaciji narudzbaArtikla (neke n-torke u rezultatu su "lažne" - označene su zvjezdicom)

narudzbaArtikla<sub>2</sub>

$\text{narudzbaArtikla}_2 = \text{narudzba} \bowtie \text{artikl} \neq \text{narudzbaArtikla}$

|   | nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|---|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| * | Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| * | Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1100     |
| * | Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| * | Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| * | Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1100     |
| * | Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| * | Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |
| * | Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1200     |

- Što bi se dogodilo ako se na temelju relacija narudzba i artikl pokuša izračunati ukupni broj naručenih proizvoda Napolitanke

```
SELECT SUM(kolicina)
  FROM narudzba, artikl
 WHERE narudzba.sifArtikl = artikl.sifArtikl
   AND nazArtikl = 'Napolitanke' ;
```

# Ispravna shema baze podataka

mjesto

| pbr   | nazMjesto |
|-------|-----------|
| 10000 | Zagreb    |
| 21000 | Split     |

prodavaonica

| nazProd  | pbr   | adresa   |
|----------|-------|----------|
| Konzum-7 | 10000 | Ilica 20 |
| Spar-28  | 21000 | Bolska 7 |

artikl

| sifArtikl | nazArtikl    |
|-----------|--------------|
| 129       | Napolitanke  |
| 139       | Petit beurre |
| 221       | Domaćica     |

narudzba

| brNar | nazProd  | datNar   |
|-------|----------|----------|
| 13/25 | Konzum-7 | 1.5.2018 |
| 43-21 | Spar-28  | 7.2.2018 |
| 41/56 | Konzum-7 | 4.2.2019 |

stavkaNarudzbe

| brNar | sifArtikl | kolicina |
|-------|-----------|----------|
| 13/25 | 129       | 1200     |
| 13/25 | 139       | 2000     |
| 43-21 | 129       | 1200     |
| 43-21 | 221       | 1800     |
| 41/56 | 129       | 1100     |

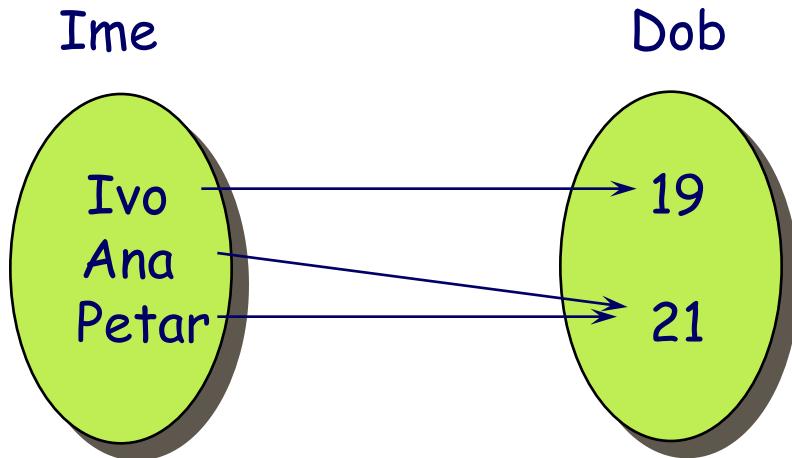
- Za vježbu provjerite
  - postoji li redundancija u ovoj bazi podataka ?
  - je li moguća pojava lažnih n-torki ?

# Kako odrediti zamjenu za loše koncipiranu relacijsku shemu?

---

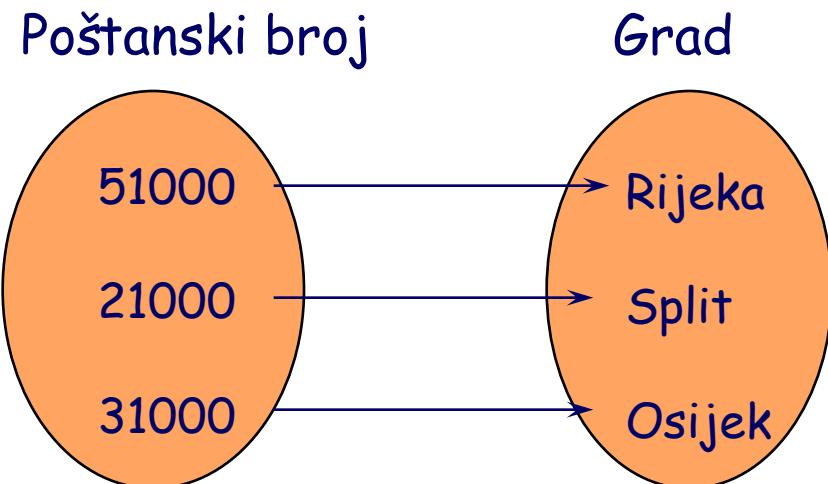
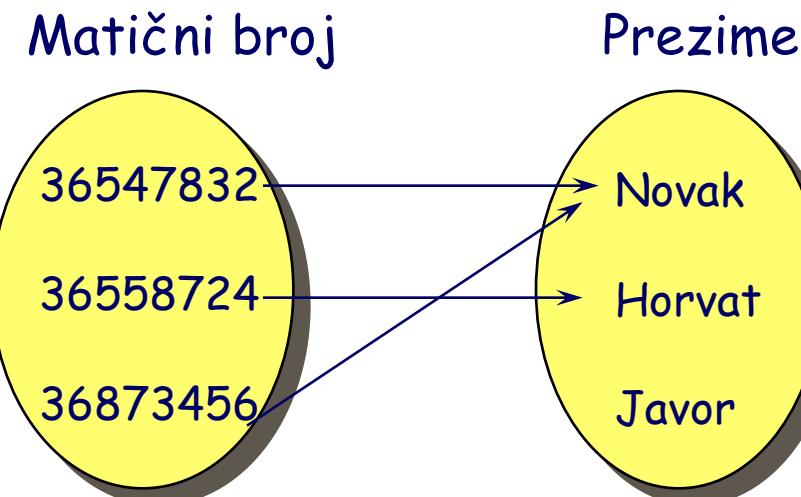
- proučavanjem značenja podataka (semantike)
  - proučavanjem zavisnosti među podacima
  - uvođenjem ograničenja koja su ovisna o semantici podataka
- 
- najvažnije su **FUNKCIJSKE ZAVISNOSTI**

# Funkcija



Preslikavanje kod kojeg vrijedi:

svakom članu skupa **Ime** pridružen je jedan i samo jedan član skupa **Dob**



# Ponavljanje: X-vrijednost n-torke

- Neka je  $X \subseteq R$ . n-torka t reducirana na skup atributa X naziva se X-vrijednost n-torke t i označava s  $t(X)$
- Primjer:

$$t = \{ \text{matBr:102, prez:Novak, ime: Marko} \}$$

$$X = \{ \text{matBr, prez} \} \quad X \subseteq R$$

$$t(X) = t(\{ \text{matBr, prez} \}) = \{ \text{matBr:102, prez:Novak} \}$$

| osoba | matBr | prez  | ime   |
|-------|-------|-------|-------|
|       | 101   | Kolar | Josip |
| t     | 102   | Novak | Marko |

t(X)

# Funkcijske zavisnosti - definicija

---

- Neka je  $r$  relacija sa shemom  $R$  i neka su  $X$  i  $Y$  skupovi atributa,  $X \subseteq R$ ,  $Y \subseteq R$

**Funkcijska zavisnost  $X \rightarrow Y$  vrijedi na shemi  $R$  ukoliko**

u svim dopuštenim stanjima relacije  $r(R)$  svaki par n-torki  $t_1$  i  $t_2$  koje imaju jednake  $X$ -vrijednosti, također imaju jednake  $Y$ -vrijednosti, odnosno:

$$t_1(X) = t_2(X) \Rightarrow t_1(Y) = t_2(Y)$$

Kratica za funkciju zavisnost je FZ.

# Funkcijske zavisnosti - primjer

- relacija osoba(OSOBA)

| osoba | matBr | prezime | ime   | postBr | grad   |
|-------|-------|---------|-------|--------|--------|
|       | 11234 | Novak   | Josip | 21000  | Split  |
|       | 12345 | Horvat  | Ivan  | 10000  | Zagreb |
|       | 22211 | Kolar   | Ante  | 21000  | Split  |
|       | 33345 | Ban     | Tomo  | 31000  | Osijek |
|       | 23456 | Kolar   | Ana   | 31000  | Osijek |

- Funkcijska zavisnost **postBr** → **grad** vrijedi na shemi OSOBA jer svaki par n-torki koje imaju jednake vrijednosti atributa **postBr** također imaju jednake vrijednosti atributa **grad** (i to vrijedi ne samo za trenutačno stanje relacije, nego za sva dopuštena stanja relacije)
- Vrijedi li funkcija zavisnost **prezime** → **postBr**?

# Funkcijske zavisnosti

- **Funkcijske zavisnosti proizlaze iz značenja podataka (semantike), a ne iz trenutačnog stanja relacije!**
- Primjer: relacija osoba(OSOBA)

| osoba | matBr | prezime | ime   | pbr |
|-------|-------|---------|-------|-----|
| 11234 | Kolar | Ante    | 21000 |     |
| 22211 | Kolar | Ante    | 31000 |     |
| 33345 | Ban   | Tomo    | 10000 |     |

- promatranjem samo trenutačnog stanja relacije mogli bismo (**pogrešno!**) zaključiti da vrijedi FZ **prezime → ime**
- **međutim**, poznavanjem značenja podataka u relaciji možemo zaključiti da je u gore prikazanu relaciju dopušteno unijeti n-torku **<76555, Kolar, Zrinka, 51000>**
- **⇒ FZ prezime → ime ne vrijedi na shemi OSOBA**

# Priroda funkcijskih zavisnosti

---

- Postojanje funkcijске zavisnosti ne može se dokazati na temelju postojećih podataka u relaciji.
- Analizom postojećih podataka u relaciji moguće je tek **prepostaviti** da bi funkcijска zavisnost mogla vrijediti.
- Dokaz za postojanje FZ treba tražiti u **značenju** pojedinih atributa.

# Priroda funkcijskih zavisnosti

$$R = \{ A, B, C \}$$

r(R)

| A | B        | C |
|---|----------|---|
| a | $\alpha$ | 1 |
| b | $\gamma$ | 1 |
| b | $\alpha$ | 1 |
| c | $\alpha$ | 3 |
| a | $\alpha$ | 1 |

r(R)

| A | B        | C |
|---|----------|---|
| a | $\alpha$ | 1 |
| b | $\gamma$ | 1 |
| b | $\alpha$ | 1 |
| c | $\alpha$ | 3 |
| a | $\alpha$ | 1 |

Vrijedi li FZ  $AB \rightarrow C$  na shemi R?

- moguće je da vrijedi, ali to ne možemo sa sigurnošću tvrditi
- bez poznavanja značenja atributa A, B i C, ne možemo zaključiti koje funkcijске zavisnosti zaista vrijede na shemi R

Vrijedi li FZ  $BC \rightarrow A$  na shemi R?

**Sa sigurnošću možemo tvrditi: NE**

# Priroda funkcijskih zavisnosti

---

- Ako u relacijskoj shemi R vrijedi FZ  $X \rightarrow Y$ , relacija  $r(R)$  ne može sadržavati dvije n-torce koje imaju jednake X-vrijednosti i različite Y-vrijednosti
- Primjer: ako u relacijskoj shemi  
 $R = \{ \text{matBr}, \text{ prezime}, \text{ grad}, \text{ telefon} \}$   
vrijedi FZ  $\text{matBr} \rightarrow \text{ prezime}$  tada relacija  $r(R)$  ne smije sadržavati dvije n-torce s istim matičnim brojem i različitim prezimenom

# Priroda funkcijskih zavisnosti - primjer

| ispit | mbr | sifPred | datIspit  | sifNast | ocjena |
|-------|-----|---------|-----------|---------|--------|
|       | 101 | 10      | 30.1.2018 | 1003    | 1      |
|       | 101 | 10      | 15.1.2019 | 1002    | 4      |
|       | 102 | 10      | 30.1.2018 | 1001    | 3      |
|       | 102 | 11      | 15.1.2018 | 1002    | 5      |

- studenta  $mbr$  je na ispitu iz predmeta  $sifPred$  na datum  $datIspit$  nastavnik  $sifNast$  ocijenio ocjenom  $ocjena$
- vrijedi li FZ  $mbr \ sifNast \rightarrow ocjena$ 
  - ne, jer bi to značilo da nastavnik  $x$  studentu  $y$  uvijek mora dati istu ocjenu
- vrijedi li FZ  $mbr \ sifPred \rightarrow ocjena$ 
  - ne, jer bi to značilo da student  $x$  iz predmeta  $y$  mora dobiti uvijek istu ocjenu
- vrijedi li FZ  $mbr \ datIspit \rightarrow ocjena$ 
  - ne, jer bi to značilo da student  $x$  na datum  $y$  uvijek mora dobiti sve jednake ocjene
- vrijedi li FZ  $mbr \ sifPred \ sifNast \rightarrow ocjena$  NE (Zašto?)
- vrijedi li FZ  $mbr \ sifPred \ datIspit \rightarrow ocjena$  DA (Zašto?)

# Funkcijske zavisnosti - SQL primjer

- pomoću SELECT naredbe ispitati bi li u relaciji ispit eventualno mogla vrijediti FZ  
 $mbr \ sifNast \rightarrow ocjena \ datIspit$

| ispit | mbr | sifPred | datIspit  | sifNast | ocjena |
|-------|-----|---------|-----------|---------|--------|
|       | 101 | 10      | 30.1.2018 | 1003    | 1      |
|       | 101 | 10      | 15.1.2019 | 1002    | 4      |
|       | 102 | 10      | 30.1.2018 | 1001    | 3      |
|       | 102 | 11      | 15.1.2018 | 1002    | 5      |

- ispituju se svi parovi n-torki  $t_1, t_2$  koje imaju jednake X-vrijednosti (u primjeru  $X = \{ mbr, sifNast \}$ )
- ako postoji par n-torki  $t_1$  i  $t_2$  koje imaju iste X-vrijednosti, a različite Y-vrijednosti (u primjeru  $Y = \{ ocjena, datIspit \}$ ), tada FZ sigurno ne vrijedi

```
SELECT *  
  FROM ispit AS t1, ispit AS t2  
 WHERE t1.mbr = t2.mbr  
   AND t1.sifNast = t2.sifNast  
   AND (t1.ocjena <> t2.ocjena  
        OR t1.datIspit <> t2.datIspit);
```

n-torke  $t_1$  i  $t_2$   
koje imaju  
jednake  
X-vrijednosti ... }  
... a različite  
Y-vrijednosti }

- ako takve n-torke ne postoje, onda FZ **možda** vrijedi

# Armstrongovi aksiomi

---

- Projektant sheme baze podataka specificira FZ koje su mu semantički očite, no obično vrijede i brojne druge FZ koje mogu biti izvedene iz početnih FZ. Korištenjem Armstrongovih aksioma izvode se nove FZ.

## ARMSTRONGOVI AKSIOMI

Neka je  $R$  relacijska shema, neka su  $X, Y, Z$  skupovi atributa i neka vrijedi:

$$X \subseteq R, Y \subseteq R, Z \subseteq R$$

### A-1 REFLEKSIVNOST

- Ako je  $Y \subseteq X$ , tada vrijedi  $X \rightarrow Y$

### A-2 UVEĆANJE

- Ako u shemi  $R$  vrijedi  $X \rightarrow Y$ , tada vrijedi i  $XZ \rightarrow Y$

### A-3 TRANZITIVNOST

- Ako u shemi  $R$  vrijedi  $X \rightarrow Y$  i  $Y \rightarrow Z$ , tada vrijedi i  $X \rightarrow Z$

# Armstrongovi aksiomi

## A-1 REFLEKSIVNOST

- **Ako je  $Y \subseteq X$ , tada vrijedi  $X \rightarrow Y$** 
  - uvijek vrijedi  $X \rightarrow X$

PRIMJER:

osoba(OSOBA)

| matBr | prezime | ime   | postBr | grad   |
|-------|---------|-------|--------|--------|
| 11234 | Novak   | Josip | 21000  | Split  |
| 12345 | Horvat  | Ivan  | 10000  | Zagreb |
| 23456 | Kolar   | Ana   | 31000  | Osijek |
| 34567 | Novak   | Josip | 31000  | Osijek |

$$X = \{ \text{prezime, ime} \} \quad Y = \{ \text{prezime} \}$$

$Y \subseteq X \Rightarrow$  u relaciji *osoba* vrijedi i FZ **prezime ime  $\rightarrow$  prezime**

$X \subseteq X \Rightarrow$  u relaciji *osoba* vrijedi i FZ **prezime ime  $\rightarrow$  prezime ime**

# Armstrongovi aksiomi

## A-2 UVEĆANJE

- **Ako u shemi R vrijedi  $X \rightarrow Y$ , tada vrijedi i  $XZ \rightarrow Y$** 
  - možemo uvećati lijevu stranu funkcijске zavisnosti

PRIMJER:

osoba(OSOBA)

| matBr | prezime | ime   | postBr | grad   |
|-------|---------|-------|--------|--------|
| 11234 | Novak   | Josip | 21000  | Split  |
| 12345 | Horvat  | Ivan  | 10000  | Zagreb |
| 23456 | Kolar   | Ana   | 31000  | Osijek |
| 34567 | Novak   | Josip | 31000  | Osijek |

U relaciji *osoba* vrijedi FZ **matBr  $\rightarrow$  ime**

- ⇒ u relaciji *osoba* vrijedi i FZ **matBr prezime  $\rightarrow$  ime**
- ⇒ u relaciji *osoba* vrijedi i FZ **matBr prezime grad  $\rightarrow$  ime**

# Armstrongovi aksiomi

## A-3 TRANZITIVNOST

- **Ako u shemi R vrijedi  $X \rightarrow Y$  i  $Y \rightarrow Z$ , tada vrijedi i  $X \rightarrow Z$** 
  - $X \rightarrow Z$  je tranzitivna zavisnost

PRIMJER:

osoba(OSOBA)

|       | matBr  | prezime | ime   | postBr | grad |
|-------|--------|---------|-------|--------|------|
| 11234 | Novak  | Josip   | 21000 | Split  |      |
| 12345 | Horvat | Ivan    | 10000 | Zagreb |      |
| 23456 | Kolar  | Ana     | 31000 | Osijek |      |
| 34567 | Novak  | Josip   | 31000 | Osijek |      |

U relaciji *osoba* vrijede FZ **matBr  $\rightarrow$  postBr** i **postBr  $\rightarrow$  grad**

⇒ u relaciji *osoba* vrijedi i FZ **matBr  $\rightarrow$  grad**

# Pravila koja proizlaze iz Armstrongovih aksioma

---

Neka je  $R$  relacijska shema, neka su  $X, Y, Z, V$  skupovi atributa i neka vrijedi:  
 $X \subseteq R, Y \subseteq R, Z \subseteq R, V \subseteq R$

## P-1 PRAVILO UNIJE (pravilo o aditivnosti)

- Ako u shemi  $R$  vrijedi  $X \rightarrow Y$  i  $X \rightarrow Z$ , tada vrijedi i  $X \rightarrow YZ$

## P-2 PRAVILO DEKOMPOZICIJE (pravilo o projektivnosti)

- Ako u shemi  $R$  vrijedi  $X \rightarrow YZ$ , tada vrijedi i  $X \rightarrow Y$

## P-3 PRAVILO O PSEUDOTRANZITIVNOSTI

- Ako u shemi  $R$  vrijedi  $X \rightarrow Y$  i  $Y \rightarrow Z$ , tada vrijedi i  $X \rightarrow Z$

# Pravila koja proizlaze iz Armstrongovih aksioma

## P-1 PRAVILO UNIJE (pravilo o aditivnosti)

- Ako u shemi R vrijedi  $X \rightarrow Y$  i  $X \rightarrow Z$ , tada vrijedi i  $X \rightarrow YZ$

PRIMJER:

| osoba(OSOBA) | matBr | prezime | ime   | postBr | grad   |
|--------------|-------|---------|-------|--------|--------|
|              | 11234 | Novak   | Josip | 21000  | Split  |
|              | 12345 | Horvat  | Ivan  | 10000  | Zagreb |
|              | 23456 | Kolar   | Ana   | 31000  | Osijek |
|              | 34567 | Novak   | Josip | 31000  | Osijek |

U relaciji *osoba* vrijede FZ **matBr → ime** i **matBr → prezime**

⇒ u relaciji *osoba* vrijedi i FZ **matBr → ime prezime**

# Pravila koja proizlaze iz Armstrongovih aksioma

## P-2 PRAVILO DEKOMPOZICIJE (pravilo o projektivnosti)

- Ako u shemi R vrijedi  $X \rightarrow YZ$ , tada vrijedi i  $X \rightarrow Y$

PRIMJER:

osoba(OSOBA)

| matBr | prezime | ime   | postBr | grad   |
|-------|---------|-------|--------|--------|
| 11234 | Novak   | Josip | 21000  | Split  |
| 12345 | Horvat  | Ivan  | 10000  | Zagreb |
| 23456 | Kolar   | Ana   | 31000  | Osijek |
| 34567 | Novak   | Josip | 31000  | Osijek |

U relaciji osoba vrijedi FZ **matBr → ime prezime**

⇒ u relaciji osoba vrijedi i FZ **matBr → ime**

⇒ u relaciji osoba vrijedi i FZ **matBr → prezime**

# Pravila koja proizlaze iz Armstrongovih aksioma

## P-3 PRAVILO PSEUDOTRANZITIVNOSTI

- Ako u shemi R vrijedi  $X \rightarrow Y$  i  $Y \rightarrow Z$ , tada vrijedi i  $X \rightarrow Z$

PRIMJER:

zaposlenje(ZAPOSLENJE)

| matbr | strSprema | funkcija | zaposlOd | zaposlDo   | placa |
|-------|-----------|----------|----------|------------|-------|
| 101   | VSS       | direktor | 1.1.2006 | 31.12.2007 | 10000 |
| 101   | VSS       | tajnik   | 1.1.2008 | 31.12.2008 | 8000  |
| 102   | VŠS       | direktor | 1.1.2009 | 31.12.2009 | 9000  |
| 102   | VŠS       | tajnik   | 1.1.2006 | 31.12.2007 | 7000  |
| 103   | VSS       | direktor | 1.1.2010 | 31.12.2010 | 10000 |
| 101   | VSS       | direktor | 1.1.2011 | 31.12.2011 | 10000 |

U relaciji zaposlenje vrijede FZ

**matbr → strSprema i funkcija strSprema → placa**

⇒ u relaciji zaposlenje vrijedi i FZ **matbr funkcija → placa**

# Primjer korištenja aksioma i pravila

---

Uz pretpostavku da na relacijskoj shemi  $R = \{ A, B, C, D, E \}$  vrijedi skup funkcijskih zavisnosti  $F = \{ A \rightarrow BD, B \rightarrow C, D \rightarrow E \}$ , dokazati da vrijedi FZ  $\text{AE} \rightarrow AC$ .

Dokaz:

- $A \rightarrow BD$  (P2: dekompozicija)  $\Rightarrow A \rightarrow B$
- $A \rightarrow B \wedge B \rightarrow C$  (A3: tranzitivnost)  $\Rightarrow A \rightarrow C$
- (A1: refleksivnost)  $\Rightarrow A \rightarrow A$
- $A \rightarrow A \wedge A \rightarrow C$  (P1: unija)  $\Rightarrow A \rightarrow AC$
- $A \rightarrow AC$  (A2: uvećanje)  $\Rightarrow AE \rightarrow AC$

# Pravilo o akumulaciji

---

Sljedeće dodatno pravilo omogućuje "algoritamski" pristup rješavanju sličnih zadataka

## PRAVILO O AKUMULACIJI

- **Ako u shemi R vrijedi**
  - $X \rightarrow VZ$  i  $Z \rightarrow W$ , tada vrijedi i  $X \rightarrow VZW$

# Primjer korištenja pravila o akumulaciji

Uz pretpostavku da na relacijskoj shemi  $R = \{ A, B, C, D, E \}$  vrijedi skup funkcijskih zavisnosti  $F = \{ A \rightarrow BD, B \rightarrow C, D \rightarrow E \}$ , dokazati da vrijedi FZ  $AE \rightarrow AC$ .

Označimo lijevu stranu FZ s  $X$  ( $X=AE$ ), a desnu stranu FZ s  $Y$  ( $Y=AC$ ).

Dokaz (primjenom A-1, pravila o akumulaciji i P-2):

1. korak:  $X \rightarrow X$

- (A1: refleksivnost)  $\Rightarrow AE \rightarrow AE$

u sljedećim koracima pomoći pravila akumulacije "uvećavati desnu stranu FZ" sve dok desna strana ne sadrži  $Y$

2. { ▪  $AE \rightarrow AE \wedge A \rightarrow BD$  (akumulacija)  $\Rightarrow AE \rightarrow AE BD$

3. { ▪  $AE \rightarrow AE BD \wedge B \rightarrow C$  (akumulacija)  $\Rightarrow AE \rightarrow AE BDC$

4. { u zadnjem koraku, kad (i ako) desna strana FZ sadrži  $Y$   
▪  $AE \rightarrow AE BDC$  (P2: dekompozicija)  $\Rightarrow AE \rightarrow AC$

# Primjer korištenja pravila o akumulaciji (za vježbu 1)

---

$R = \{ L, M, N, P, Q, R \}$ ,  $F = \{ Q \rightarrow R, M \rightarrow PQ, PQL \rightarrow N \}$   
dokazati da vrijedi FZ  $\text{MLR} \rightarrow \text{QN}$ .

- (A1: refleksivnost)  $\Rightarrow \text{MLR} \rightarrow \text{MLR}$
- $\text{MLR} \rightarrow \text{MLR} \wedge M \rightarrow PQ$  (akumulacija)  $\Rightarrow \text{MLR} \rightarrow \text{MLRPQ}$
- $\text{MLR} \rightarrow \text{MLRPQ} \wedge PQL \rightarrow N$  (akumulacija)  $\Rightarrow \text{MLR} \rightarrow \text{MLRPQN}$
- $\text{MLR} \rightarrow \text{MLRPQN}$  (P2: dekompozicija)  $\Rightarrow \text{MLR} \rightarrow \text{QN}$

# Primjer korištenja pravila o akumulaciji (za vježbu 2)

---

$R = \{ L, M, N, P, Q, R \}$ ,  $F = \{ Q \rightarrow R, M \rightarrow PQ, PQL \rightarrow N \}$   
dokazati da vrijedi FZ  $MQ \rightarrow LN$ .

- (A1: refleksivnost)  $\Rightarrow MQ \rightarrow MQ$
- $MQ \rightarrow MQ \wedge Q \rightarrow R$  (akumulacija)  $\Rightarrow MQ \rightarrow MQR$
- $MQ \rightarrow MQR \wedge M \rightarrow PQ$  (akumulacija)  $\Rightarrow MQ \rightarrow MQRP$
- ne postoji FZ kojom bi se moglo nastaviti "uvećavati desnu stranu"
- $\Rightarrow MQ \rightarrow LN$  ne vrijedi

# Ključ entiteta, ključ relacije

---

- entitet je bilo što, što ima suštinu ili bit i posjeduje značajke s pomoću kojih se može razlučiti od svoje okoline
- ključ entiteta sadrži one atribute koji omogućuju da se pojedini entiteti mogu razlučiti od okoline
- relacijom se opisuje skup entiteta

Ključ relacije je skup atributa koji nedvosmisleno određuje n-torce relacije.

- Ključ relacije ima svojstvo da funkcijски određuje atribute u preostalom dijelu relacije

# Ključ relacije

---

- ključ relacijske sheme  $R$  je skup atributa  $K$ ,  $K \subseteq R$ , koji ima sljedeća svojstva:
  1.  $K \rightarrow (R \setminus K)$  (također vrijedi i  $K \rightarrow R$ )
    - ključ funkcijски određuje attribute u preostalom dijelu relacijske sheme
  2. ne postoji  $K' \subset K$  za kojeg vrijedi  $K' \rightarrow R$ 
    - ključ je minimalan skup atributa koji funkcijски određuje attribute u preostalom dijelu relacijske sheme

# Ključ relacije - primjer

| osoba | matBr | prezime | ime   | postBr | grad   |
|-------|-------|---------|-------|--------|--------|
|       | 11234 | Novak   | Josip | 21000  | Split  |
|       | 12345 | Horvat  | Ivan  | 10000  | Zagreb |
|       | 23456 | Kolar   | Ana   | 31000  | Osijek |
|       | 34567 | Novak   | Josip | 10000  | Zagreb |

Ključ:  $K_{OSOBA} = \{ \text{matBr} \}$

$\text{matBr} \rightarrow \text{prezime}$

$\text{matBr} \rightarrow \text{ime}$

$\text{matBr} \rightarrow \text{postBr}$

$\text{matBr} \rightarrow \text{grad}$

Za  $K = \{ \text{matBr}, \text{prezime} \}$  također vrijedi

$K \rightarrow \{ \text{ime}, \text{postBr}, \text{grad} \},$

ali  $K$  **nije** ključ jer postoji  $K' = \{ \text{matBr} \}$ ,  $K' \subset K$ , za kojeg vrijedi

$K' \rightarrow \{ \text{prezime}, \text{ime}, \text{postBr}, \text{grad} \}$

# Ključevi relacije

- mogući ključevi (*candidate key*)
- primarni ključ (*primary key*) odabire se jedan od mogućih ključeva
- alternativni ključevi (*alternate key*) ostali mogući ključevi

PRIMJER:

djelatnik

| matBr | prezime | ime   | OIB         |
|-------|---------|-------|-------------|
| 11234 | Novak   | Josip | 15707332975 |
| 12345 | Horvat  | Ivan  | 69435151530 |
| 23456 | Kolar   | Ana   | 59351332978 |
| 34567 | Novak   | Josip | 42794313596 |

- mogući ključevi:
  - { matBr }
  - { OIB }
- primarni ključ: { matBr }
- alternativni ključ: { OIB }

# Struktura relacije

- Relacijska shema sastoji se od:
  - atributa koji su dio ključa (ključni atributi, ključni dio relacije)
  - atributa iz zavisnog dijela relacije (neklijučni atributi, neključni dio relacije)

PRIMJER:

| djelatnik | matBr | prezime | ime   | OIB         |
|-----------|-------|---------|-------|-------------|
|           | 11234 | Novak   | Josip | 15707332975 |
|           | 12345 | Horvat  | Ivan  | 69435151530 |
|           | 23456 | Kolar   | Ana   | 59351332978 |
|           | 34567 | Novak   | Josip | 42794313596 |

- primarni ključ: { matBr }
- alternativni ključ: { OIB }

- ključni atributi, ključni dio relacije:
  - matBr
  - OIB
- neključni atributi, neključni dio relacije:
  - prezime
  - ime

# Zadatak:

- Odrediti moguće ključeve, primarni ključ, alternativne ključeve, ključni dio relacije, neključni dio relacije
  - uzeti u obzir da klub tijekom istog dana može igrati najviše jednu utakmicu

| utakmicaPrvenstva | domaci    | gosti     | datum      | rezultat |
|-------------------|-----------|-----------|------------|----------|
|                   | Arsenal   | Liverpool | 12.06.2018 | 2:1      |
|                   | Arsenal   | Liverpool | 08.03.2019 | 2:1      |
|                   | Newcastle | Everton   | 08.03.2019 | 3:3      |
|                   | Everton   | Liverpool | 22.03.2019 | 4:0      |
|                   | Liverpool | Everton   | 05.04.2019 | 5:2      |

# Baze podataka

## Predavanja

### 7. Oblikovanje sheme relacijske baze podataka (2. dio)

Travanj, 2020.





# Normalizacija

# Postupci normalizacije

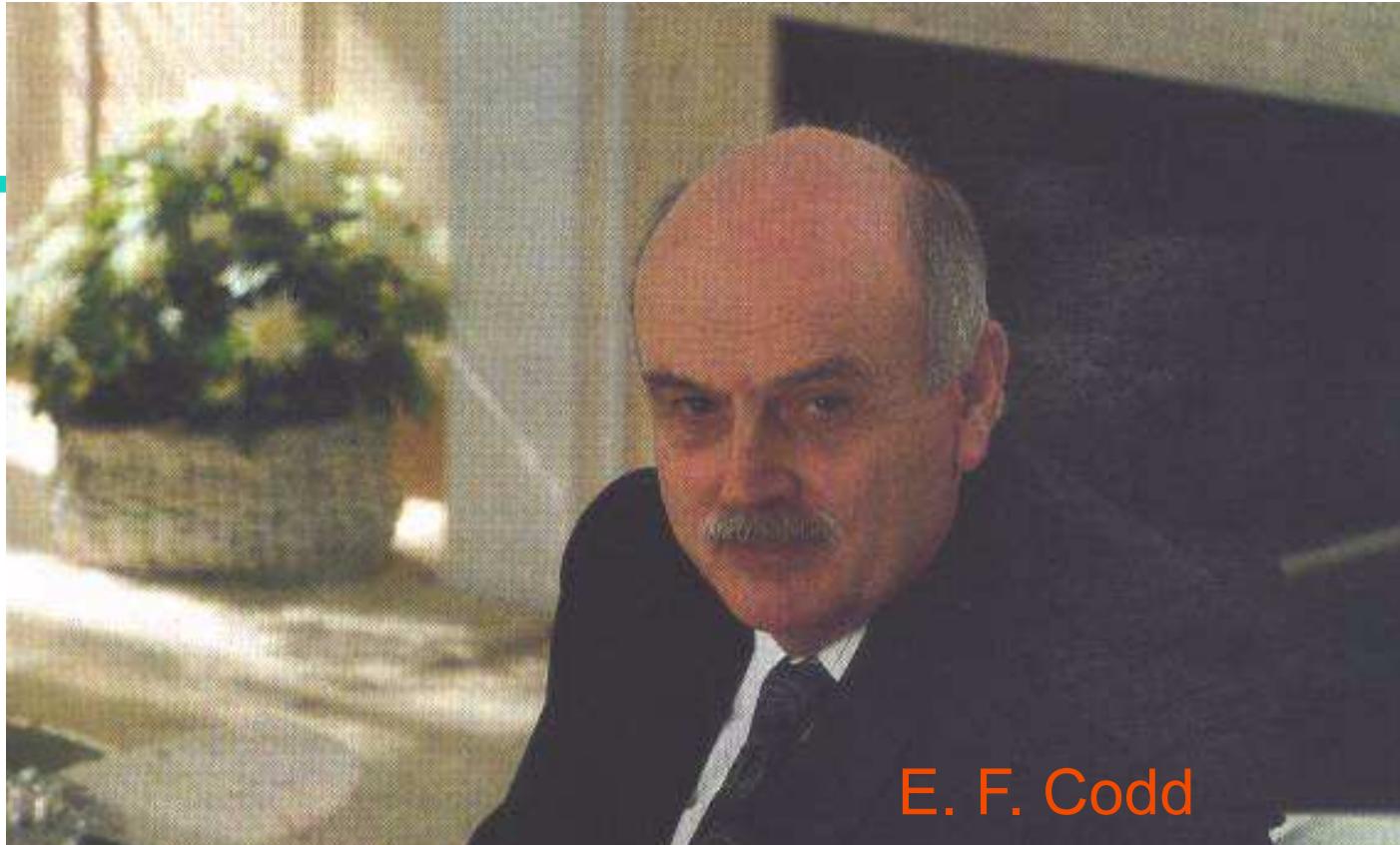
---

- Uočena znanja o međusobnim **funkcijskim zavisnostima** atributa relacije koriste se u postupcima normalizacije.
- **Cilj:**
  - **ukloniti redundanciju**
    - anomalije unosa, izmjene i brisanja
  - **spriječiti pojavu lažnih n-torki**
- Postupci normalizacije omogućavaju da se postupno, **točno definiranom metodom**, odredi dobra zamjena za loše koncipiranu relacijsku shemu

**E. F. Codd:**

**“Normalized data base structure: A brief tutorial”**

*Proc. ACM SIGFIDET Workshop on Data Description, Access and Control, 1971*



E. F. Codd

"I called it normalization because then-President Nixon was talking a lot about normalizing relations with China. I figured that if he could normalize relations, so could I."

('A "FIRESIDE" CHAT', DBMS, Dec. 1993)

# Normalne forme

---

- Prva normalna forma - 1 NF
- Druga normalna forma - 2 NF
- Treća normalna forma - 3 NF
- Boyce-Coddova normalna forma - BCNF

Temelje se na FUNKCIJSKIM ZAVISNOSTIMA

---

- Četvrta normalna forma - 4NF  
Temelji se na VIŠEZNAČNIM ZAVISNOSTIMA
- Projekcijsko-spojna normalna forma - PJNF  
Temelji se na SPOJNIM ZAVISNOSTIMA

# Postupci normalizacije

---

- Dekompozicija
  - početne relacije (relacijske sheme) se dekomponiraju na temelju uočenih funkcijskih zavisnosti
- Sinteza
  - zadan je skup atributa i nad njima skup funkcijskih zavisnosti iz kojih se sintetiziraju relacijske sheme koje zadovoljavaju 3NF

# Dekompozicija relacijske sheme (relacije)

---

- Dekompozicijom (razlaganjem) relacijska shema  $R$  zamjenjuje se shemama  $R_1, R_2, \dots, R_n$ ,  $R_i \subseteq R$ , pri čemu vrijedi  $R = R_1 R_2 \dots R_n$
- Dekompozicijom se relacija  $r(R)$  zamjenjuje relacijama  $r_1(R_1), r_2(R_2), \dots, r_n(R_n)$ , pri čemu je  $r_i(R_i) = \pi_{R_i}(r)$ , za  $i = 1, \dots, n$
- Relacija  $r(R)$  se dekomponira na relacije  $r_1(R_1), r_2(R_2), \dots, r_n(R_n)$  **bez gubitaka informacija (lossless decomposition)** ako vrijedi:

$$r_1(R_1) \bowtie r_2(R_2) \bowtie \dots \bowtie r_n(R_n) = r(R)$$

odnosno

$$\pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \dots \pi_{R_n}(r) = r(R)$$

# Dekompozicija relacije - primjer

- Zadana je relacija:

| r(R) |    |    |    |
|------|----|----|----|
| A    | B  | C  | D  |
| a1   | b1 | c1 | d1 |
| a2   | b2 | c1 | d1 |

- Relaciju  $r(R)$  dekomponirati na relacije

- $r_1(R_1), R_1 = \{ A, C \}$
- $r_2(R_2), R_2 = \{ B, C \}$
- $r_3(R_3), R_3 = \{ C, D \}$

| r <sub>1</sub> (R <sub>1</sub> ) |    | r <sub>2</sub> (R <sub>2</sub> ) |    | r <sub>3</sub> (R <sub>3</sub> ) |    |
|----------------------------------|----|----------------------------------|----|----------------------------------|----|
| A                                | C  | B                                | C  | C                                | D  |
| a1                               | c1 | b1                               | c1 | c1                               | d1 |
| a2                               | c1 | b2                               | c1 |                                  |    |

- Je li dekompozicija obavljena bez gubitaka informacija?

$r_1(R_1) \bowtie r_2(R_2) \bowtie r_3(R_3)$

| A  | B  | C  | D  |
|----|----|----|----|
| a1 | b1 | c1 | d1 |
| a1 | b2 | c1 | d1 |
| a2 | b1 | c1 | d1 |
| a2 | b2 | c1 | d1 |

$\Rightarrow NE$

# Razlaganje relacije bez gubitaka na dvije projekcije

- Relacija se bez gubitaka razlaže na svoje dvije projekcije ako:
  - projekcije imaju zajedničke attribute
  - zajednički atributi su ključ u barem jednoj od projekcija

PRIMJER:

| osoba | matBr  | prez  | ime   | postBr | nazMj |
|-------|--------|-------|-------|--------|-------|
| 11234 | Novak  | Josip | 21000 | Split  |       |
| 12345 | Horvat | Ivan  | 10000 | Zagreb |       |
| 23456 | Kolar  | Ana   | 31000 | Osijek |       |
| 34567 | Novak  | Josip | 10000 | Zagreb |       |

$$osoba_1 = \pi_{\text{matBr}, \text{prez}, \text{ime}, \text{postBr}} (\text{osoba})$$

$$mjesto = \pi_{\text{postBr}, \text{nazMj}} (\text{osoba})$$

- Hoće li se relacija osoba dekomponirati bez gubitaka informacija na relacije  $osoba_1$  i mjesto? Odnosno, vrijedi li:

$$osoba \equiv osoba_1 \triangleright\triangleleft mjesto$$

# Primjer razlaganja relacije na dvije projekcije

osoba<sub>1</sub>

| matBr | prez   | ime   | postBr |
|-------|--------|-------|--------|
| 11234 | Novak  | Josip | 21000  |
| 12345 | Horvat | Ivan  | 10000  |
| 23456 | Kolar  | Ana   | 31000  |
| 34567 | Novak  | Josip | 10000  |

$$\text{OSOBA}_1 = \{ \text{matBr}, \text{prez}, \text{ime}, \text{postBr} \}$$

$$K_{\text{OSOBA}_1} = \{ \text{matBr} \}$$

mjesto

| postBr | nazMj  |
|--------|--------|
| 21000  | Split  |
| 10000  | Zagreb |
| 31000  | Osijek |

$$\text{MJESTO} = \{ \text{postBr}, \text{nazMj} \}$$

$$K_{\text{MJESTO}} = \{ \text{postBr} \}$$

$$\text{OSOBA}_1 \cap \text{MJESTO} = \{ \text{postBr} \}$$

$$\Rightarrow \text{osoba} \equiv \text{osoba}_1 \bowtie \text{mjesto}$$

osoba

| matBr | prez   | ime   | postBr | nazMj  |
|-------|--------|-------|--------|--------|
| 11234 | Novak  | Josip | 21000  | Split  |
| 12345 | Horvat | Ivan  | 10000  | Zagreb |
| 23456 | Kolar  | Ana   | 31000  | Osijek |
| 34567 | Novak  | Josip | 10000  | Zagreb |

# Prva normalna forma (1NF)

---

- Definicija:

Relacijska shema je u **1NF** ako:

- domene atributa sadrže samo jednostavne (nedjeljive) vrijednosti
  - vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa
  - neključni atributi relacije funkcijски ovise o ključu relacije
- 
- Shema baze podataka  $R = \{ R_1, R_2, \dots, R_n \}$  je u 1NF ako je svaka relacijska shema  $R_1, R_2, \dots, R_n$  u 1NF

# Prva normalna forma - primjer

---

- Poduzeće evidentira podatke o radnicima
  - $\text{RADNIK} = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{datRod}, \text{sifOdjel} \}$

| radnik (RADNIK) | matBr  | prezime | ime        | datRod | sifOdjel |
|-----------------|--------|---------|------------|--------|----------|
| 1111            | Novak  | Ivan    | 28.12.1970 | 50     |          |
| 1121            | Kolar  | Iva     | 16.10.1965 | 30     |          |
| 1133            | Horvat | Krešo   | 19.03.1978 | 50     |          |

$$K_{\text{RADNIK}} = \{ \text{matBr} \}$$

- domene svih atributa sadrže jednostavne (nedjeljive) vrijednosti
- vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa
- neklučni atributi relacije funkcijски ovise o ključu relacije

⇒ Relacijska shema RADNIK je u 1NF

# Prva normalna forma - primjer

---

- Poduzeće evidentira podatke o radnicima i njihovoј djeci - korisnicima zdravstvenog osiguranja.
  - $\text{RADNIK}_1 = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{imenaDjece} \}$

$K_{\text{RADNIK}_1} = \{ \text{matBr} \}$

| radnik <sub>1</sub> (RADNIK <sub>1</sub> ) | matBr | prezime | ime   | imenaDjece       |
|--------------------------------------------|-------|---------|-------|------------------|
|                                            | 1111  | Novak   | Ivan  | Jasna, Vedran    |
|                                            | 1121  | Kolar   | Iva   | Ivan             |
|                                            | 1133  | Horvat  | Krešo | Petar, Ana, Ivan |

- domena atributa *imenaDjece* ne sadrži jednostavne (nedjeljive vrijednosti)

⇒ Relacijska shema RADNIK<sub>1</sub> nije u 1NF

# Prva normalna forma - primjer

- Poduzeće evidentira podatke o radnicima i njihovoј djeci - korisnicima zdravstvenog osiguranja.
  - $\text{RADNIK}_2 = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{imeDj}, \text{datRodDj} \}$

| $\text{radnik}_2(\text{RADNIK}_2)$         | $\text{matBr}$ | $\text{prezime}$ | $\text{ime}$ | $\text{imeDj}$       | $\text{datRodDj}$                      |
|--------------------------------------------|----------------|------------------|--------------|----------------------|----------------------------------------|
| $K_{\text{RADNIK}_2} = \{ \text{matBr} \}$ | 1111           | Novak            | Ivan         | Jasna<br>Vedran      | 21.01.1995<br>13.12.1997               |
|                                            | 1121           | Kolar            | Iva          | Ivan                 | 23.03.2000                             |
|                                            | 1133           | Horvat           | Krešo        | Petar<br>Ana<br>Ivan | 22.02.1998<br>19.09.2000<br>05.11.2002 |

- domene sadrže jednostavne vrijednosti, ali vrijednost atributa  $\text{imeDj}$  nije uvijek samo jedna vrijednost iz domene tog atributa (isto vrijedi i za atribut  $\text{datRodDj}$ )

⇒ Relacijska shema  $\text{RADNIK}_2$  nije u 1NF

# Normalizacija na 1NF - izdvajanjem atributa u posebnu relaciju

- u posebnu relaciju izdvaja se skup atributa koji se ponavlja s jednakom kratnošću, zajedno s ključem originalne relacije

$$RADNIK_2 = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{imeDj}, \text{datRodDj} \} \quad K_{RADNIK_2} = \{ \text{matBr} \}$$

| radnik <sub>3</sub> (RADNIK <sub>3</sub> ) | matBr | prezime | ime   |
|--------------------------------------------|-------|---------|-------|
|                                            | 1111  | Novak   | Ivan  |
|                                            | 1121  | Kolar   | Iva   |
|                                            | 1133  | Horvat  | Krešo |

$$RADNIK_3 = \{ \text{matBr}, \text{prezime}, \text{ime} \}$$

$$K_{RADNIK_3} = \{ \text{matBr} \}$$

| dijete (DIJETE) | matBr | imeDj  | datRodDj   |
|-----------------|-------|--------|------------|
|                 | 1111  | Jasna  | 21.01.1995 |
|                 | 1111  | Vedran | 13.12.1997 |
|                 | 1121  | Ivan.  | 23.03.2000 |
|                 | 1133  | Petar  | 22.02.1998 |
|                 | 1133  | Ana    | 19.09.2000 |
|                 | 1133  | Ivan   | 05.11.2002 |

$$DIJETE = \{ \text{matBr}, \text{imeDj}, \text{datRodDj} \}$$

$$K_{DIJETE} = \{ \text{matBr}, \text{imeDj} \}$$

- operacija je izvedena bez gubitaka informacija - relacijske sheme imaju zajedničke attribute (matBr), zajednički attribute su ključ u RADNIK<sub>3</sub>

# Normalizacija na 1NF - promjenom ključa

---

$RADNIK_2 = \{ matBr, prezime, ime, imeDj, datRodDj \}$      $K_{RADNIK_2} = \{ matBr \}$

$RADNIK_4 = \{ matBr, prezime, ime, imeDj, datRodDj \}$

$K_{RADNIK_4} = \{ matBr, imeDj \}$

| radnik <sub>4</sub> (RADNIK <sub>4</sub> ) | matBr  | prezime | ime    | imeDj      | datRodDj |
|--------------------------------------------|--------|---------|--------|------------|----------|
| 1111                                       | Novak  | Ivan    | Jasna  | 21.01.1995 |          |
| 1111                                       | Novak  | Ivan    | Vedran | 13.12.1997 |          |
| 1121                                       | Kolar  | Iva     | Ivan   | 23.03.2000 |          |
| 1133                                       | Horvat | Krešo   | Petar  | 22.02.1998 |          |
| 1133                                       | Horvat | Krešo   | Ana    | 19.09.2000 |          |
| 1133                                       | Horvat | Krešo   | Ivan   | 05.11.2002 |          |

# Druga normalna forma (2NF)

---

- Definicija:

Relacijska shema  $R$  je u **2NF** ako je u **1NF** i ako je

- svaki atribut iz zavisnog dijela potpuno funkcijски ovisan o svakom ključu relacije

- Shema baze podataka  $R = \{ R_1, R_2, \dots, R_n \}$  je u **2NF** ako je svaka relacijska shema  $R_1, R_2, \dots, R_n$  u **2NF**

# Potpuna funkcionalna zavisnost

---

- Skup atributa  $Y$  **potpuno je funkcionalno ovisan** o skupu atributa  $X$  relacijske sheme  $R$  ako:
  - $Y$  funkcionalno ovisi o  $X$
  - i
  - ne postoji pravi podskup od  $X$  koji funkcionalno određuje  $Y$

PRIMJER:

Zadan je skup FZ  $F = \{ ABC \rightarrow DE, E \rightarrow F \}$ .

Je li  $\{ D, E \}$  potpuno funkcionalno ovisan o  $\{ A, B, C \}$  ?

Da, jer ne postoji skup  $Z \subset \{ A, B, C \}$  takav da  $Z \rightarrow \{ D, E \}$

# Nepotpuna funkcionalna zavisnost

---

Zadana je relacijska shema  $R$  i skupovi atributa  $X$  i  $Y$  iz  $R$ , tj.  $X \subseteq R$ ,  $Y \subseteq R$ . Neka u  $R$  vrijedi FZ  $X \rightarrow Y$ .

FZ  $X \rightarrow Y$  je **nepotpuna** ako postoji skup atributa  $Z$  koji je pravi podskup od  $X$ , za koji vrijedi  $Z \rightarrow Y$

odnosno

FZ  $X \rightarrow Y$  je **nepotpuna** ako  $(\exists Z) (Z \subset X) : Z \rightarrow Y$

PRIMJER:

Zadan je skup FZ  $F = \{ ABC \rightarrow D, BC \rightarrow E, E \rightarrow D \}$ .

Je li  $\{ D \}$  potpuno funkcionalski ovisan o  $\{ A, B, C \}$  ?

Ne, jer postoji skup  $\{ B, C \} \subset \{ A, B, C \}$  takav da  $\{ B, C \} \rightarrow \{ D \}$

## Druga normalna forma - primjer

$\text{RADNIK}_4 = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{imeDj}, \text{datRodDj} \}$        $K_{\text{RADNIK}_4} = \{ \text{matBr}, \text{imeDj} \}$

| radnik <sub>4</sub> ( $\text{RADNIK}_4$ ) | matBr  | prezime | ime    | imeDj      | datRodDj |
|-------------------------------------------|--------|---------|--------|------------|----------|
| 1111                                      | Novak  | Ivan    | Jasna  | 21.01.1995 |          |
| 1111                                      | Novak  | Ivan    | Vedran | 13.12.1997 |          |
| 1121                                      | Kolar  | Iva     | Ivan.  | 23.03.2000 |          |
| 1133                                      | Horvat | Krešo   | Petar  | 22.02.1998 |          |
| 1133                                      | Horvat | Krešo   | Ana    | 19.09.2000 |          |
| 1133                                      | Horvat | Krešo   | Ivan   | 05.11.2002 |          |

- relacijska shema  $\text{RADNIK}_4$  zadovoljava 1NF
- **postoji FZ:**  $\text{matBr} \rightarrow \text{prezime } \text{ime}$
- $\text{matBr } \text{imeDj} \rightarrow \text{prezime } \text{ime}$  je nepotpuna FZ!

⇒ Relacijska shema  $\text{RADNIK}_4$  nije u 2NF

# Normalizacija na 2NF

- Normalizacijom na 2NF nastaju:
  - relacijska shema koja sadrži skup atributa koji su bili nepotpuno funkcijски ovisni o ključu i dio ključa o kojem su potpuno funkcijски ovisni
  - relacijska shema koja sadrži ključ originalne relacije i skup atributa koji su potpuno funkcijski ovisni o ključu

$RADNIK_5 = \{ \text{matBr}, \text{ prezime}, \text{ ime} \}$

$K_{RADNIK_5} = \{ \text{matBr} \}$

| radnik <sub>5</sub> (RADNIK <sub>5</sub> ) |         |       |
|--------------------------------------------|---------|-------|
| matBr                                      | prezime | ime   |
| 1111                                       | Novak   | Ivan  |
| 1121                                       | Kolar   | Iva   |
| 1133                                       | Horvat  | Krešo |

$DIJETE = \{ \text{matBr}, \text{ imeDj}, \text{ datRodDj} \}$

$K_{DIJETE} = \{ \text{matBr}, \text{ imeDj} \}$

| dijete (DIJETE) |        |            |
|-----------------|--------|------------|
| matBr           | imeDj  | datRodDj   |
| 1111            | Jasna  | 21.01.1995 |
| 1111            | Vedran | 13.12.1997 |
| 1121            | Ivan.  | 23.03.2000 |
| 1133            | Petar  | 22.02.1998 |
| 1133            | Ana    | 19.09.2000 |
| 1133            | Ivan   | 05.11.2002 |

# Normalizacija na 2NF

---

Neka su  $X, Y, Z, V$  atributi ili skupovi atributa. Zadana je relacijska shema  $R = XYZV$  i na njoj skup funkcijskih zavisnosti  $F = \{ XY \rightarrow ZV, X \rightarrow Z \}$ .

Ključ relacije  $K_R = XY$ . R je u 1NF. **Zadovoljava li R 2NF?**

- funkcijска zavisnost  $XY \rightarrow Z$  je nepotpuna  
**R ne zadovoljava 2NF**

Normalizacijom na 2NF shema R se zamjenjuje shemama:

$$R_1 = XZ$$

$$R_2 = XYV$$

$$K_{R_1} = X$$

$$K_{R_2} = XY$$

Relacija  $r(R)$  se normalizacijom na 2NF zamjenjuje projekcijama:

$$r_1 = \pi_{XZ}(r) \quad r_2 = \pi_{XYV}(r)$$

- operacija je izvedena bez gubitaka informacija - relacijske sheme imaju zajedničke attribute (X), zajednički atributi su ključ u  $R_1$ .

# Treća normalna forma (3NF)

---

- Definicija:

Relacijska shema je u 3NF ako je u 1NF i ako:

- niti jedan atribut iz zavisnog dijela nije tranzitivno funkcijski ovisan o bilo kojem ključu relacije
- Shema baze podataka  $R = \{ R_1, R_2, \dots, R_n \}$  je u 3NF ako je svaka relacijska shema  $R_1, R_2, \dots, R_n$  u 3NF

# Tranzitivna funkcionalna zavisnost

---

Zadano je:

- relacijska shema  $R$ ,
- skupovi atributa  $X \subseteq R$ ,  $Y \subseteq R$ ,  $Z \subseteq R$

Skup atributa  $Z$  je tranzitivno ovisan o  $X$  ako vrijedi:

- $X \rightarrow Y$ ,  $Y \not\rightarrow X$  i  $Y \rightarrow Z$

# Tranzitivna funkcionalna zavisnost - primjer

---

Zadana je relacijska shema  $R = \{ A, B, C, D, E, F, G \}$  i skup FZ  
 $F = \{ AB \rightarrow CD, D \rightarrow EF, CD \rightarrow ABG \}$

**Jesu li EFG tranzitivno funkcijски ovisni o AB?**

EF je tranzitivno funkcijски ovisan o AB, jer

- $AB \rightarrow D, D \not\rightarrow AB, D \rightarrow EF$

G nije tranzitivno funkcijски ovisan o AB, jer iako

- $AB \rightarrow CD, CD \rightarrow G$
- nije zadovoljen uvjet  $CD \not\rightarrow AB$

# Treća normalna forma - primjer

$\text{OSOBA} = \{ \text{matBr}, \text{prez}, \text{ime}, \text{postBr}, \text{nazMjesto} \}$     $K_{\text{OSOBA}} = \{ \text{matBr} \}$

| osoba (OSOBA) | matBr  | prez  | ime   | postBr | nazMjesto |
|---------------|--------|-------|-------|--------|-----------|
| 1111          | Novak  | Ivan  | 10000 | Zagreb |           |
| 1121          | Kolar  | Iva   | 31000 | Osijek |           |
| 1133          | Horvat | Krešo | 10000 | Zagreb |           |

- Relacijska shema OSOBA zadovoljava 1NF.
  - vrijedi FZ:  $\text{matBr} \rightarrow \text{postBr}$
  - vrijedi FZ:  $\text{postBr} \rightarrow \text{nazMjesto}$
  - ne vrijedi FZ:  $\text{postBr} \rightarrow \text{matBr}$
- $\text{matBr} \rightarrow \text{nazMjesto}$  je tranzitivna zavisnost !
- Relacijska shema OSOBA ne zadovoljava 3NF.

# Normalizacija na 3NF

$\text{OSOBA} = \{ \text{matBr}, \text{prez}, \text{ime}, \text{postBr}, \text{nazMjesto} \}$     $K_{\text{OSOBA}} = \{ \text{matBr} \}$

Normalizacijom na 3NF nastaju:

- relacijska shema koja sadrži skup atributa relacijske sheme OSOBA koji su tranzitivno ovisni o ključu (*nazMjesto*) te srednji skup atributa uočene tranzitivne zavisnosti (*postBr*)
- relacijska shema koja sadrži ključ relacijske sheme OSOBA (*matBr*) i neključne attribute relacijske sheme OSOBA koji nisu tranzitivno ovisni o ključu

$\text{MJESTO} = \{ \text{postBr}, \text{nazMjesto} \}$

$K_{\text{MJESTO}} = \{ \text{postBr} \}$

| mjesto (MJESTO) |           |
|-----------------|-----------|
| postBr          | nazMjesto |
| 10000           | Zagreb    |
| 31000           | Osijek    |

$\text{OSOBA}_1 = \{ \text{matBr}, \text{prezime}, \text{ime}, \text{postBr} \}$

$K_{\text{OSOBA}_1} = \{ \text{matBr} \}$

| osoba <sub>1</sub> (OSOBA <sub>1</sub> ) |         |       |        |
|------------------------------------------|---------|-------|--------|
| matBr                                    | prezime | ime   | postBr |
| 1111                                     | Novak   | Ivan  | 10000  |
| 1121                                     | Kolar   | Iva   | 31000  |
| 1133                                     | Horvat  | Krešo | 10000  |

# Normalizacija na 3NF

---

Neka su  $X, Y, Z, V$  atributi ili skupovi atributa. Zadana je relacijska shema  $R = XYZV$  i na njoj skup funkcijskih zavisnosti  $F = \{ X \rightarrow YZV, Z \rightarrow V \}$ .

Ključ relacije  $K_R = X$ . R je u 1NF. **Zadovoljava li R 3NF?**

- funkcijска зависност  $X \rightarrow V$  је транзитивна  
 $R$  не задоволjava 3NF

Normalizацијом на 3NF shema R se zamjenjuje shemama:

$$R_1 = XYZ$$

$$K_{R1} = X$$

$$R_2 = ZV$$

$$K_{R2} = Z$$

Relacija  $r(R)$  se normalizацијом на 3NF zamjenjuje пројекцијама:

$$r_1 = \pi_{XYZ}(r) \quad r_2 = \pi_{ZV}(r)$$

- операција је изведена без губитака информација - relacijske sheme имају zajedničке атрибуте ( $Z$ ), zajedničки атрибути су ključ у  $R_2$ .

## Treća normalna forma - komentar

---

Normalizacija na 2NF nije nužni preuvjet za provođenje normalizacije na 3NF jer se nepotpune FZ mogu promatrati kao tranzitivne FZ.

**Primjer:** zadana je shema  $R = XYZV$  i na njoj skup funkcijskih zavisnosti  $F = \{ XY \rightarrow ZV, X \rightarrow Z \}$ . Ključ relacije  $K_R = XY$ . R je u 1NF, ali nije u 2NF jer postoji nepotpuna FZ  $XY \rightarrow Z$ . Međutim, postoji i tranzitivna funkcionska zavisnost  $XY \rightarrow Z$  ( $XY \rightarrow X \wedge X \rightarrow Z$ ).

Normalizacijom na 3NF shema R se zamjenjuje shemama:

$$R_1 = XZ$$

$$K_{R1} = X$$

$$R_2 = XYV$$

$$K_{R2} = XY$$

$R_1$  i  $R_2$  su u 2NF i 3NF

Preporuka: normalizaciju ipak obavljati postupno

$1NF \Rightarrow 2NF \Rightarrow 3NF$

# Normalizacija na 3NF - primjer

---

$\text{OSOBA}_2 = \{ \text{matBr}, \text{prez}, \text{ime}, \text{OIB} \}$

| osoba2 (OSOBA2) | matBr | prez   | ime   | OIB         |
|-----------------|-------|--------|-------|-------------|
|                 | 1111  | Novak  | Ivan  | 69435151530 |
|                 | 1121  | Kolar  | Iva   | 59351332978 |
|                 | 1133  | Horvat | Krešo | 42794313596 |

- postoji FZ:  $\text{matBr} \rightarrow \text{prez} \text{ } \text{ime} \text{ } \text{OIB}$
- postoje FZ:  $\text{OIB} \rightarrow \text{prez} \text{ } \text{ime}$  i  $\text{OIB} \rightarrow \text{matBr}$
  
- $\text{matBr}$  i  $\text{OIB}$  su mogući ključevi
- Relacijska shema  $\text{OSOBA}_2$  zadovoljava 3NF.

$$K1_{\text{OSOBA}2} = \{ \text{matBr} \}$$

$$K2_{\text{OSOBA}2} = \{ \text{OIB} \}$$

## Normalizacija na 3NF - dodatna razmatranja

---

Neka su  $X, Y, Z, V$  atributi ili skupovi atributa. Zadana je relacijska shema  $R = XYZV$  i na njoj skup funkcijskih zavisnosti

$F = \{ X \rightarrow YZV, Z \rightarrow V, Z \rightarrow X \}$ . R je u 1NF. Neka je ključ  $K_R = X$ .

- vrijedi  $X \rightarrow Z$  i  $Z \rightarrow V$ , ali  $X \rightarrow V$  nije tranzitivna FZ jer vrijedi i  $Z \rightarrow X$
- Zbog  $X \rightarrow Z$  i  $Z \rightarrow X$  funkcijsku zavisnost  $X \rightarrow V$  nije potrebno ukloniti jer u tom slučaju nema redundancije.
- $Z$  je također mogući ključ u R

$$K_{R1} = X$$

$$K_{R2} = Z$$

$X$  i  $Z$  su mogući ključevi.

- Relacijska shema R zadovoljava 3NF.

# 1. primjer normalizacije

---

- Zadana je relacijska shema:

ISPIT = { matBr, prez, ime, sifPred, nazPred, datlsp, ocj, sifNas, prezNas }

i trenutna vrijednost relacije **ispit**(ISPIT):

| ispit (ISPIT) |       |       |         |         |          |     |        |         |
|---------------|-------|-------|---------|---------|----------|-----|--------|---------|
| matBr         | prez  | ime   | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
| 1111          | Novak | Ivan  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111          | Novak | Ivan  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111          | Novak | Ivan  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111          | Novak | Ivan  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234          | Kolar | Petar | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

- funkcijske zavisnosti odrediti na temelju značenja podataka
- odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijски ovise o ključu)
- postupno normalizirati relacijsku shemu ISPIT na 2NF i 3NF

# 1. primjer normalizacije - 1NF

| ispit (ISPIT) |       |       |         |         |          |     |        |         |
|---------------|-------|-------|---------|---------|----------|-----|--------|---------|
| matBr         | prez  | ime   | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
| 1111          | Novak | Ivan  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111          | Novak | Ivan  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111          | Novak | Ivan  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111          | Novak | Ivan  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234          | Kolar | Petar | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

- Određivanje ključa: ako se (**pogrešno**) prepostavi da je  $K = \{ \text{matBr} \}$   
Bi li tada postojali **neklučni atributi** koje **ključ funkcijski ne određuje**?
- $\text{matBr} \rightarrow \text{prez } \text{ime}$   
**međutim:**
- $\text{matBr} \not\rightarrow \text{sifPred}$        $\text{matBr} \not\rightarrow \text{nazPred}$   
 $\text{matBr} \not\rightarrow \text{datlsp}$        $\text{matBr} \not\rightarrow \text{ocj}$   
 $\text{matBr} \not\rightarrow \text{sifNas}$        $\text{matBr} \not\rightarrow \text{prezNas}$

# 1. primjer normalizacije - 1NF

ispit (ISPIT)

| matBr | prez  | ime   | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
|-------|-------|-------|---------|---------|----------|-----|--------|---------|
| 1111  | Novak | Ivan  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | Novak | Ivan  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | Novak | Ivan  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | Novak | Ivan  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | Kolar | Petar | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

- Ako se pretpostavi  $K = \{ \text{matBr}, \text{sifPred}, \text{datlsp} \}$   
Bi li tada postojali **neklučni atributi koje ključ funkcijski ne određuje?**
- $\text{matBr } \text{sifPred } \text{datlsp} \rightarrow \text{prez } \text{ime } \text{nazPred } \text{ocj } \text{sifNas } \text{prezNas}$   
postoji li skup  $X \subset \{ \text{matBr}, \text{sifPred}, \text{datlsp} \}$  za kojeg vrijedi  $X \rightarrow R$  ?  
 $\Rightarrow \text{NE} \Rightarrow \{ \text{matBr}, \text{sifPred}, \text{datlsp} \}$  je mogući ključ
- $K_{\text{ISPIT}} = \{ \text{matBr}, \text{sifPred}, \text{datlsp} \}$
- **zadovoljen je uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu**

# 1. primjer normalizacije - 2NF

ispit (ISPIT)

| matBr | prez  | ime   | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
|-------|-------|-------|---------|---------|----------|-----|--------|---------|
| 1111  | Novak | Ivan  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | Novak | Ivan  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | Novak | Ivan  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | Novak | Ivan  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | Kolar | Petar | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

- Postoje li neključni atributi koji ovise o dijelu ključa?
  - $\text{matBr} \rightarrow \text{prez, ime}$

$\text{student} = \pi_{\text{matBr, prez, ime}}(\text{ispit})$

$\text{ispit}_1 = \pi_{\text{matBr, sifPred, nazPred, datlsp, ocj, sifNas, prezNas}}(\text{ispit})$

$K_{\text{STUDENT}} = \{ \text{matBr} \}$

| student (STUDENT) |       |       |
|-------------------|-------|-------|
| matBr             | prez  | ime   |
| 1111              | Novak | Ivan  |
| 1234              | Kolar | Petar |

2NF, 3NF: O.K.

ispit<sub>1</sub> (ISPIT<sub>1</sub>)

$K_{\text{ISPIT}_1} = \{ \text{matBr, sifPred, datlsp} \}$

| matBr | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
|-------|---------|---------|----------|-----|--------|---------|
| 1111  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

# 1. primjer normalizacije - 2NF (nastavak)

ispit<sub>1</sub> (ISPIT<sub>1</sub>)

- Postoje li neključni atributi koji ovise o dijelu ključa?

| matBr | sifPred | nazPred | datlsp   | ocj | sifNas | prezNas |
|-------|---------|---------|----------|-----|--------|---------|
| 1111  | 1001    | Mat-1   | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | 1001    | Mat-1   | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | 1003    | Fiz-1   | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | 1002    | Mat-2   | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | 1001    | Mat-1   | 29.01.19 | 3   | 2222   | Brnetić |

- sifPred → nazPred

$$\text{predmet} = \pi_{\text{sifPred}, \text{nazPred}}(\text{ispit}_1)$$

$$\text{ispit}_2 = \pi_{\text{matBr}, \text{sifPred}, \text{datlsp}, \text{ocj}, \text{sifNas}, \text{prezNas}}(\text{ispit}_1)$$

K<sub>PREDMET</sub> = { sifPred }

| predmet (PREDMET) |         |
|-------------------|---------|
| sifPred           | nazPred |
| 1001              | Mat-1   |
| 1003              | Fiz-1   |
| 1002              | Mat-2   |

2NF, 3NF: O.K.

ispit<sub>2</sub> (ISPIT<sub>2</sub>)

K<sub>ISPIT<sub>2</sub></sub> = { matBr, sifPred, datlsp }

| matBr | sifPred | datlsp   | ocj | sifNas | prezNas |
|-------|---------|----------|-----|--------|---------|
| 1111  | 1001    | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | 1001    | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | 1003    | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | 1002    | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | 1001    | 29.01.19 | 3   | 2222   | Brnetić |

2NF: O.K.

# 1. primjer normalizacije - 3NF

ispit<sub>2</sub> (ISPIT<sub>2</sub>)

| matBr | sifPred | datlsp   | ocj | sifNas | prezNas |
|-------|---------|----------|-----|--------|---------|
| 1111  | 1001    | 29.01.19 | 1   | 1111   | Pašić   |
| 1111  | 1001    | 05.02.19 | 3   | 1111   | Pašić   |
| 1111  | 1003    | 28.06.18 | 2   | 3333   | Horvat  |
| 1111  | 1002    | 27.06.18 | 4   | 2222   | Brnetić |
| 1234  | 1001    | 29.01.19 | 3   | 2222   | Brnetić |

- Postoje li neključni atributi koji tranzitivno ovise o ključu?
- matBr sifPred datlsp → sifNas      sifNas → prezNas

nastavnik =  $\pi_{\text{sifNas}, \text{prezNas}}(\text{ispit}_2)$

ispit<sub>3</sub> =  $\pi_{\text{matBr}, \text{sifPred}, \text{datlsp}, \text{ocj}, \text{sifNas}}(\text{ispit}_2)$

K<sub>NASTAVNIK</sub> = { sifNas }

| nastavnik (NASTAVNIK) |         |
|-----------------------|---------|
| sifNas                | prezNas |
| 1111                  | Pašić   |
| 3333                  | Horvat  |
| 2222                  | Brnetić |

3NF: O.K.

ispit<sub>3</sub> (ISPIT<sub>3</sub>)

K<sub>ISPIT<sub>3</sub></sub> = { matBr, sifPred, datlsp }

| matBr | sifPred | datlsp   | ocj | sifNas |
|-------|---------|----------|-----|--------|
| 1111  | 1001    | 29.01.19 | 1   | 1111   |
| 1111  | 1001    | 05.02.19 | 3   | 1111   |
| 1111  | 1003    | 28.06.18 | 2   | 3333   |
| 1111  | 1002    | 27.06.18 | 4   | 2222   |
| 1234  | 1001    | 29.01.19 | 3   | 2222   |

3NF: O.K.

# 1. primjer normalizacije - 3NF

| student (STUDENT) |       |       |
|-------------------|-------|-------|
| matBr             | prez  | ime   |
| 1111              | Novak | Ivan  |
| 1234              | Kolar | Petar |

$$K_{\text{STUDENT}} = \{ \text{matBr} \}$$

| predmet (PREDMET) |         |
|-------------------|---------|
| sifPred           | nazPred |
| 1001              | Mat-1   |
| 1003              | Fiz-1   |
| 1002              | Mat-2   |

$$K_{\text{PREDMET}} = \{ \text{sifPred} \}$$

| nastavnik (NASTAVNIK) |         |
|-----------------------|---------|
| sifNas                | prezNas |
| 1111                  | Pašić   |
| 3333                  | Horvat  |
| 2222                  | Brnetić |

$$K_{\text{NASTAVNIK}} = \{ \text{sifNas} \}$$

| ispit <sub>3</sub> (ISPIT <sub>3</sub> ) |         |          |     |        |
|------------------------------------------|---------|----------|-----|--------|
| matBr                                    | sifPred | datlsp   | ocj | sifNas |
| 1111                                     | 1001    | 29.01.19 | 1   | 1111   |
| 1111                                     | 1001    | 05.02.19 | 3   | 1111   |
| 1111                                     | 1003    | 28.06.18 | 2   | 3333   |
| 1111                                     | 1002    | 27.06.18 | 4   | 2222   |
| 1234                                     | 1001    | 29.01.19 | 3   | 2222   |

$$K_{\text{ISPIT}_3} = \{ \text{matBr}, \text{sifPred}, \text{datlsp} \}$$

- Shema baze podataka STUSLU:  
$$\text{STUSLU} = \{ \text{STUDENT}, \text{PREDMET}, \text{NASTAVNIK}, \text{ISPIT}_3 \}$$
- Shema baze podataka STUSLU zadovoljava 3NF

## 2. primjer normalizacije

---

Zadana je relacijska shema  $R = ABCDEFGH$  i na njoj skup funkcijskih zavisnosti

$$F = \{ ABC \rightarrow DEFGH, A \rightarrow D, BC \rightarrow FGH, FG \rightarrow H \}.$$

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

## 2. primjer normalizacije

---

$R = ABCDEFGH$

$F = \{ ABC \rightarrow DEFGH, A \rightarrow D, BC \rightarrow FGH, FG \rightarrow H \}$

- **Odrediti primarni ključ relacije**

Vrijedi li  $ABC \rightarrow DEFGH$  ?

**DA**

postoji li skup  $X \subset ABC$  za kojeg vrijedi  $X \rightarrow R$  ?

**NE**

⇒  $ABC$  je mogući ključ i može se odabratи kao primarni ključ sheme  $R$ .

$R = ABCDEFGH$

$K_R = ABC$

$R$  je u 1NF

## 2. primjer normalizacije - 2NF

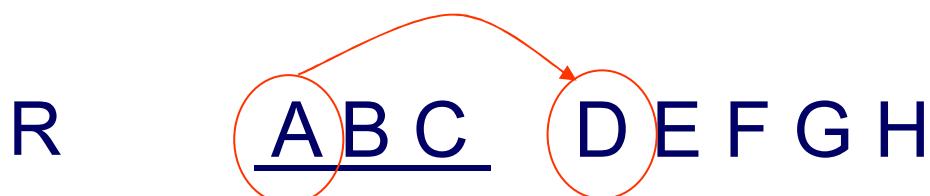
---

$$R = ABCDEFGH \quad K_R = ABC$$

$$F = \{ ABC \rightarrow DEFGH, A \rightarrow D, BC \rightarrow FGH, FG \rightarrow H \}$$

- **Normalizacija na 2NF**

Svi atributi iz zavisnog dijela moraju biti **potpuno** funkcijski ovisni o ključu.



- $ABC \rightarrow D$  je nepotpuna FZ, jer vrijedi  $A \rightarrow D$        $R$  nije u 2NF

Normalizacijom na 2NF se  $R$  zamjenjuje shemama:

$$R_1 = AD$$

$$K_{R1} = A$$

$R_1$  je u 2NF

$$R_2 = ABCEFGH$$

$$K_{R2} = ABC$$

$R_2$  nije u 2NF

## 2. primjer normalizacije - 2NF (nastavak)

---

$$R_1 = AD$$

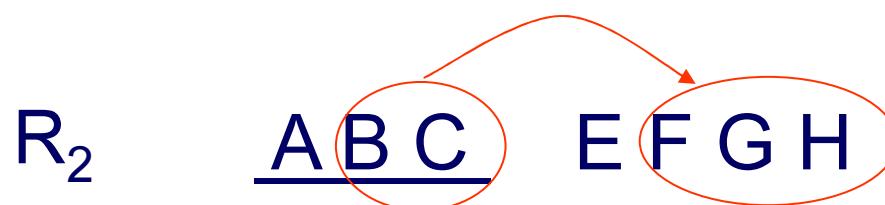
$$K_{R_1} = A$$

$$R_2 = ABC E F G H$$

$$K_{R_2} = ABC$$

$$F = \{ ABC \rightarrow DEFGH, A \rightarrow D, BC \rightarrow FGH, FG \rightarrow H \}$$

Svi atributi iz zavisnog dijela moraju biti **potpuno** funkcijski ovisni o ključu.



- $ABC \rightarrow FGH$  je nepotpuna FZ, jer vrijedi  $BC \rightarrow FGH$      $R_2$  nije u 2NF

Normalizacijom na 2NF se  $R_2$  zamjenjuje shemama:

$$R_{21} = BCFGH$$

$$K_{R_{21}} = BC$$

$R_{21}$  je u 2NF

$$R_{22} = ABCE$$

$$K_{R_{22}} = ABC$$

$R_{22}$  je u 2NF

## 2. primjer normalizacije - 3NF

$$R_1 = AD$$

$$K_{R_1} = A$$

$R_1$  je u 3NF

$$R_{21} = BCFGH$$

$$K_{R_{21}} = BC$$

$R_{21}$  nije u 3NF

$$R_{22} = ABCE$$

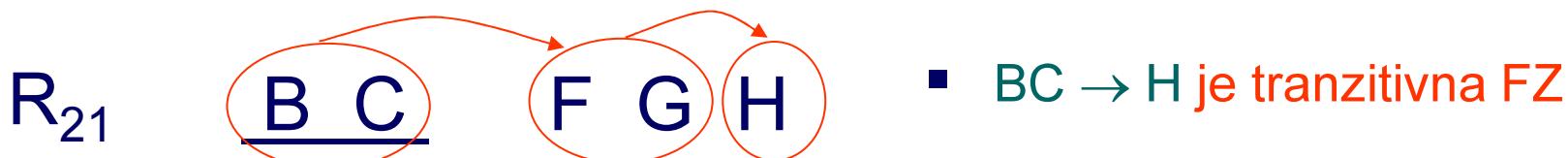
$$K_{R_{22}} = ABC$$

$R_{22}$  je u 3NF

$$F = \{ ABC \rightarrow DEFGH, A \rightarrow D, BC \rightarrow FGH, FG \rightarrow H \}$$

- **Normalizacija na 3NF**

Niti jedan atribut iz zavisnog dijela ne smije biti tranzitivno ovisan o ključu.



▪  $BC \rightarrow H$  je tranzitivna FZ

Normalizacijom na 3NF se  $R_{21}$  zamjenjuje shemama:

$$R_{211} = BCFG$$

$$K_{R_{211}} = BC$$

$R_{211}$  je u 3NF

$$R_{212} = FGH$$

$$K_{R_{212}} = FG$$

$R_{212}$  je u 3NF

**Shema baze podataka u 3NF sastoji se od relacijskih shema:**

$R_1, R_{22}, R_{211}$  i  $R_{212}$

## Baze podataka

# Predavanja

## 8. Oblikovanje sheme relacijske baze podataka (3. dio - primjeri)

# Travanj, 2020.



# Zadatak 1

- Prodavaonice šalju svoje narudžbe proizvođaču:

Konzum-7

Ilica 20

10 000 Zagreb

Kraš

Ravnice bb  
10 000 Zagreb

**Narudžba**

br. 13/25

**datum: 1.5.2018**

Molimo isporučite nam **1200** komada proizvoda **Napolitanke** (šifra **129**) i **2000** komada proizvoda **Petit beurre** (šifra **139**)

Spar-28

Bolska 7  
**21 000 Split**

Kraš

Ravnice bb  
10 000 Zagreb

**Narudžba**

br. 43-21

**datum: 7.2.2018**

Molimo isporučite nam **1200** komada proizvoda **Napolitanke** (šifra **129**) i **1800** komada proizvoda **Domaćica** (šifra **221**)

Konzum-7

Ilica 20

10 000 Zagreb

Kraš

Ravnice bb  
10 000 Zagreb

**Narudžba**

br. 41/56

**datum: 4.2.2019**

Molimo isporučite nam **1100** komada proizvoda **Napolitanke** (šifra **129**)

- proizvođač želi pohraniti podatke o narudžbama u svoju bazu podataka.  
Svi podaci se pohranjuju u relaciju **narudzbaArtikla**

**narudzbaArtikla**

**nazProd**

**pbr**

**nazMjesto**

**adresa**

**brNar**

**datNar**

**sifArtikl**

**nazArtikl**

**kolicina**

# Zadatak 1

- Sadržaj relacije nakon unosa podataka iz prispjelih narudžbi:

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- Normalizirajte relaciju narudzbaArtikla na 1NF, 2NF, 3NF ako vrijedi da je svaki broj narudžbe jedinstven (ne može se desiti da brojevi narudžbi prispjelih iz različitih prodavaonica budu jednaki)

$\text{brNar} \rightarrow \text{nazProd}$

# Zadatak 1

**NARUDZBAARTIKLA = { nazProd, pbr, nazMjesto, adresa, brNar, datNar, sifArtikl, nazArtikl, kolicina}**

- trenutna vrijednost relacije **narudzbaArtikla (NARUDZBAARTIKLA)** :

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- odrediti funkcione zavisnosti na temelju značenja podataka
- odrediti primarni ključ relacije (tako da bude zadovoljen uvjet **1NF** prema kojem neključni atributi funkcionalno ovise o ključu)
- postupno normalizirati relacijsku shemu NARUDZBAARTIKLA na **2NF** i **3NF**

# Zadatak 1 - 1NF

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- Određivanje ključa: bi li brNar bio dobar odabir za ključ?
- Postoje li neključni atributi koji ne ovise o broju narudžbe (brNar)?
- $\text{brNar} \rightarrow \text{nazProd}$   $\text{brNar} \rightarrow \text{pbr}$   $\text{brNar} \rightarrow \text{nazMjesto}$   $\text{brNar} \rightarrow \text{adresa}$   $\text{brNar} \rightarrow \text{datNar}$   
međutim:
- $\text{brNar} \not\rightarrow \text{sifArtikl}$        $\text{brNar} \not\rightarrow \text{nazArtikl}$        $\text{brNar} \not\rightarrow \text{kolicina}$
- O kojim atributima funkcijски ovisi atribut nazArtikl?       $\text{sifArtikl} \rightarrow \text{nazArtikl}$
- O kojim atributima funkcijски ovisi atribut kolicina?       $\text{brNar sifArtikl} \rightarrow \text{kolicina}$ 
  - **KLJUČ?**       $\text{sifArtikl} \not\rightarrow \text{kolicina}$

# Zadatak 1 - 1NF

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- Pretpostavimo  $K = \{ \text{brNar}, \text{sifArtikl} \}$   
Provjerite postoje li neključni atributi koje ključ funkcijски ne određuje.
- $\text{brNar } \text{sifArtikl} \rightarrow \text{nazProd } \text{pbr } \text{nazMjesto } \text{adresa } \text{datNar } \text{nazArtikl } \text{kolicina}$   
postoji li skup  $X \subset \{ \text{brNar}, \text{sifArtikl} \}$  za kojeg vrijedi  $X \rightarrow R$  ?  
 $\Rightarrow \text{NE} \Rightarrow \{ \text{brNar}, \text{sifArtikl} \}$  je mogući ključ

$$K_{\text{NARUDZBAARTIKLA}} = \{ \text{brNar}, \text{sifArtikl} \}$$

- zadovoljen je uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu

# Zadatak 1 - 2NF

narudzbaArtikla

| nazProd  | pbr   | nazMjesto | adresa   | brNar | datNar   | sifArtikl | nazArtikl    | kolicina |
|----------|-------|-----------|----------|-------|----------|-----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 129       | Napolitanke  | 1200     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25 | 1.5.2018 | 139       | Petit beurre | 2000     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 129       | Napolitanke  | 1200     |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21 | 7.2.2018 | 221       | Domaćica     | 1800     |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56 | 4.2.2019 | 129       | Napolitanke  | 1100     |

- Postoje li neključni atributi koji ovise o dijelu ključa?

- vrijedi:  $\text{brNar} \rightarrow \text{nazProd}$   $\text{pbr}$   $\text{nazMjesto}$   $\text{adresa}$   $\text{datNar}$   
 $\Rightarrow$  Na koje relacije treba razložiti relaciju narudzbaArtikla?  
Koji su ključevi novonastalih relacija?

$\text{narudzba} = \pi_{\text{nazProd}, \text{pbr}, \text{nazMjesto}, \text{adresa}, \text{brNar}, \text{datNar}}(\text{narudzbaArtikla})$

$K_{\text{NARUDZBA}} = \{ \text{brNar} \}$

$\text{stavkaNarudzbe} = \pi_{\text{brNar}, \text{sifArtikl}, \text{nazArtikl}, \text{kolicina}}(\text{narudzbaArtikla})$

$K_{\text{STAVKANARUDZBE}} = \{ \text{brNar}, \text{sifArtikl} \}$

# Zadatak 1 - 2NF

brNar → nazProd pbr nazMjesto adresa datNar

narudzba

| nazProd  | pbr   | nazMjesto | adresa   | <u>brNar</u> | datNar   |
|----------|-------|-----------|----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25        | 1.5.2018 |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21        | 7.2.2018 |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56        | 4.2.2019 |

- narudzba ima jednostavan ključ  
⇒ **2NF OK**

stavkaNarudzbe

| <u>brNar</u> | <u>sifArtikl</u> | nazArtikl    | kolicina |
|--------------|------------------|--------------|----------|
| 13/25        | 129              | Napolitanke  | 1200     |
| 13/25        | 139              | Petit beurre | 2000     |
| 43-21        | 129              | Napolitanke  | 1200     |
| 43-21        | 221              | Domaćica     | 1800     |
| 41/56        | 129              | Napolitanke  | 1100     |

Jesu li relacije narudzba i stavkaNarudzbe u 2NF?

# Zadatak 1 - 2NF

- Je li stavkaNarudzbe u 2NF?  
(postoje li neključni atributi koji ovise o dijelu ključa?)

- Vrijedi:  $sifArtikl \rightarrow nazArtikl$
- ⇒ Na koje relacije treba razložiti relaciju stavkaNarudzbe?
- Koji su ključevi novonastalih relacija?

stavkaNarudzbe

| brNar | sifArtikl | nazArtikl    | kolicina |
|-------|-----------|--------------|----------|
| 13/25 | 129       | Napolitanke  | 1200     |
| 13/25 | 139       | Petit beurre | 2000     |
| 43-21 | 129       | Napolitanke  | 1200     |
| 43-21 | 221       | Domaćica     | 1800     |
| 41/56 | 129       | Napolitanke  | 1100     |

$$artikl = \pi_{sifArtikl, nazArtikl}(stavkaNarudzbe)$$

$$K_{ARTIKL} = \{ sifArtikl \}$$

$$stavkaNarudzbe_1 = \pi_{brNar, sifArtikl, kolicina}(stavkaNarudzbe)$$

$$K_{STAVKANARUDZBE1} = \{ brNar, sifArtikl \}$$

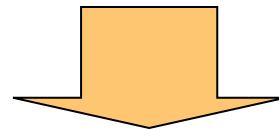
# Zadatak 1 - 2NF

stavkaNarudzbe

| <u>brNar</u> | <u>sifArtikl</u> | <u>nazArtikl</u> | <u>kolicina</u> |
|--------------|------------------|------------------|-----------------|
| 13/25        | 129              | Napolitanke      | 1200            |
| 13/25        | 139              | Petit beurre     | 2000            |
| 43-21        | 129              | Napolitanke      | 1200            |
| 43-21        | 221              | Domaćica         | 1800            |
| 41/56        | 129              | Napolitanke      | 1100            |

sifArtikl → nazArtikl

Jesu li relacije artikl i  
stavkaNarudzbe<sub>1</sub> u 2NF?



artikl

| <u>sifArtikl</u> | <u>nazArtikl</u> |
|------------------|------------------|
| 129              | Napolitanke      |
| 139              | Petit beurre     |
| 221              | Domaćica         |

2NF O.K.

stavkaNarudzbe<sub>1</sub>

| <u>brNar</u> | <u>sifArtikl</u> | <u>kolicina</u> |
|--------------|------------------|-----------------|
| 13/25        | 129              | 1200            |
| 13/25        | 139              | 2000            |
| 43-21        | 129              | 1200            |
| 43-21        | 221              | 1800            |
| 41/56        | 129              | 1100            |

2NF O.K.

# Zadatak 1 - 3NF

- Postoje li neključni atributi koji tranzitivno ovise o ključu?

narudzba

| nazProd  | pbr   | nazMjesto | adresa   | <u>brNar</u> | datNar   |
|----------|-------|-----------|----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25        | 1.5.2018 |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21        | 7.2.2018 |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56        | 4.2.2019 |

Je li  
relacija  
narudzba  
u 3NF?

stavkaNarudzbe<sub>1</sub>

| <u>brNar</u> | <u>sifArtikl</u> | kolicina |
|--------------|------------------|----------|
| 13/25        | 129              | 1200     |
| 13/25        | 139              | 2000     |
| 43-21        | 129              | 1200     |
| 43-21        | 221              | 1800     |
| 41/56        | 129              | 1100     |

artikl

| <u>sifArtikl</u> | nazArtikl    |
|------------------|--------------|
| 129              | Napolitanke  |
| 139              | Petit beurre |
| 221              | Domaćica     |

3NF O.K.

3NF O.K.

# Zadatak 1 - 3NF

- Postoje li u relaciji narudzba neključni atributi koji tranzitivno ovise o ključu?

| narudzba |       |           |          |              |          |
|----------|-------|-----------|----------|--------------|----------|
| nazProd  | pbr   | nazMjesto | adresa   | <u>brNar</u> | datNar   |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25        | 1.5.2018 |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21        | 7.2.2018 |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56        | 4.2.2019 |

- Vrijedi:
$$\text{brNar} \rightarrow \text{nazProd}$$
$$\text{nazProd} \rightarrow \text{pbr}$$
$$\text{nazProd} \rightarrow \text{nazMjesto}$$
$$\text{nazProd} \rightarrow \text{adresa}$$
$$\text{nazProd} \not\rightarrow \text{brNar}$$

⇒ Na koje relacije treba razložiti relaciju narudzba?

Koji su ključevi novonastalih relacija?

prodavaonica =  $\pi_{\text{nazProd}, \text{pbr}, \text{nazMjesto}, \text{adresa}}(\text{narudzba})$

$K_{\text{PRODAVAONICA}} = \{ \text{nazProd} \}$

$\text{narudzba}_1 = \pi_{\text{brNar}, \text{nazProd}, \text{datNar}}(\text{narudzba})$

$K_{\text{NARUDZBA}_1} = \{ \text{brNar} \}$

# Zadatak 1 - 3NF

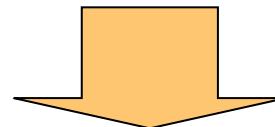
narudzba

| nazProd  | pbr   | nazMjesto | adresa   | <u>brNar</u> | datNar   |
|----------|-------|-----------|----------|--------------|----------|
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 13/25        | 1.5.2018 |
| Spar-28  | 21000 | Split     | Bolska 7 | 43-21        | 7.2.2018 |
| Konzum-7 | 10000 | Zagreb    | Ilica 20 | 41/56        | 4.2.2019 |

$\text{brNar} \rightarrow \text{nazProd}$

$\text{nazProd} \not\rightarrow \text{brNar}$

$\text{nazProd} \rightarrow \text{pbr}$   $\text{nazProd} \rightarrow \text{nazMjesto}$   $\text{nazProd} \rightarrow \text{adresa}$



Jesu li relacije prodavaonica i  
narudzba<sub>1</sub> u 3NF?

prodavaonica

| <u>nazProd</u> | pbr   | nazMjesto | adresa   |
|----------------|-------|-----------|----------|
| Konzum-7       | 10000 | Zagreb    | Ilica 20 |
| Spar-28        | 21000 | Split     | Bolska 7 |

3NF?

narudzba<sub>1</sub>

| <u>brNar</u> | nazProd  | datNar   |
|--------------|----------|----------|
| 13/25        | Konzum-7 | 1.5.2018 |
| 43-21        | Spar-28  | 7.2.2018 |
| 41/56        | Konzum-7 | 4.2.2019 |

3NF: O.K.

# Zadatak 1 - 3NF

- Postoje li neključni atributi koji tranzitivno ovise o ključu u relaciji prodavaonica?

| prodavaonica   |       |           |          |
|----------------|-------|-----------|----------|
| <u>nazProd</u> | pbr   | nazMjesto | adresa   |
| Konzum-7       | 10000 | Zagreb    | Ilica 20 |
| Spar-28        | 21000 | Split     | Bolska 7 |

- $\text{nazProd} \rightarrow \text{pbr}$        $\text{pbr} \rightarrow \text{nazMjesto}$
- $\text{pbr} \not\rightarrow \text{nazProd}$

⇒ Na koje relacije treba razložiti relaciju prodavaonica?  
Koji su ključevi novonastalih relacija?

$$\text{mjesto} = \pi_{\text{pbr}, \text{nazMjesto}}(\text{prodavaonica})$$

$$K_{MJESTO} = \{ \text{pbr} \}$$

$$\text{prodavaonica}_1 = \pi_{\text{nazProd}, \text{pbr}, \text{adresa}}(\text{prodavaonica})$$

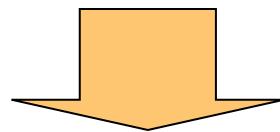
$$K_{PRODAVAONICA1} = \{ \text{nazProd} \}$$

# Zadatak 1 - 3NF

prodavaonica

| <u>nazProd</u> | pbr   | nazMjesto | adresa   |
|----------------|-------|-----------|----------|
| Konzum-7       | 10000 | Zagreb    | Ilica 20 |
| Spar-28        | 21000 | Split     | Bolska 7 |

- $\text{nazProd} \rightarrow \text{pbr}$        $\text{pbr} \rightarrow \text{nazMjesto}$
- $\text{pbr} \not\rightarrow \text{nazProd}$



Jesu li relacije mjesto i  
prodavaonica<sub>1</sub> u 3NF?

mjesto

| <u>pbr</u> | nazMjesto |
|------------|-----------|
| 10000      | Zagreb    |
| 21000      | Split     |

3NF: O.K.

prodavaonica<sub>1</sub>

| <u>nazProd</u> | pbr   | adresa   |
|----------------|-------|----------|
| Konzum-7       | 10000 | Ilica 20 |
| Spar-28        | 21000 | Bolska 7 |

3NF: O.K.

# Zadatak 1 - 3NF

mjesto

| pbr   | nazMjesto |
|-------|-----------|
| 10000 | Zagreb    |
| 21000 | Split     |

prodavaonica<sub>1</sub>

| nazProd  | pbr   | adresa   |
|----------|-------|----------|
| Konzum-7 | 10000 | Ilica 20 |
| Spar-28  | 21000 | Bolska 7 |

artikl

| sifArtikl | nazArtikl    |
|-----------|--------------|
| 129       | Napolitanke  |
| 139       | Petit beurre |
| 221       | Domaćica     |

narudzba<sub>1</sub>

| brNar | nazProd  | datNar   |
|-------|----------|----------|
| 13/25 | Konzum-7 | 1.5.2018 |
| 43-21 | Spar-28  | 7.2.2018 |
| 41/56 | Konzum-7 | 4.2.2019 |

stavkaNarudzbe<sub>1</sub>

| brNar | sifArtikl | kolicina |
|-------|-----------|----------|
| 13/25 | 129       | 1200     |
| 13/25 | 139       | 2000     |
| 43-21 | 129       | 1200     |
| 43-21 | 221       | 1800     |
| 41/56 | 129       | 1100     |

- Shema baze podataka u 3NF sastoji se od relacijskih shema:  
mjesto, prodavaonica<sub>1</sub>, artikl, narudzba<sub>1</sub>, stavkaNarudzbe<sub>1</sub>

## Zadatak 2

---

Zadana je relacijska shema  $R = ABCDEF$  i na njoj skup funkcijskih zavisnosti:

$$F = \{ AB \rightarrow CD, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \} .$$

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacijske sheme (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

## Zadatak 2 – 1NF

---

$R = ABCDEF$

$F = \{ AB \rightarrow CD, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \}$

- Odrediti primarni ključ relacije.

$AB \rightarrow CD$

$AB \rightarrow EF$

$\Rightarrow AB \rightarrow CDEF$  (P-1: unija)

postoji li skup  $X \subset AB$  za kojeg vrijedi  $X \rightarrow R$  ?

NE

$\Rightarrow R = ABCDEF$

$K_R = AB$

R je u 1NF

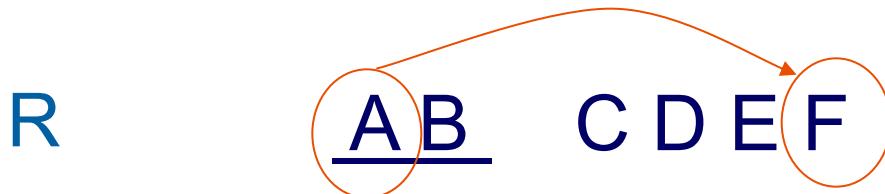
## Zadatak 2 - 2NF

$R = ABCDEF$

$K_R = AB$

$F = \{ AB \rightarrow CD, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \}$

Postoje li atributi iz zavisnog dijela koji nisu potpuno funkcijски ovisni o ključu?



- $AB \rightarrow F$  je nepotpuna FZ, jer vrijedi  $A \rightarrow F$   $R$  nije u 2NF

Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu  $R$ .  
Odredite ključeve.

$R_1 = AF$

$K_{R1} = A$

$R_1$  je u 2NF

$R_2 = ABCDE$

$K_{R2} = AB$

$R_2$  je u 2NF

## Zadatak 2 - 3NF

$$R_1 = AF$$

$$K_{R1} = A$$

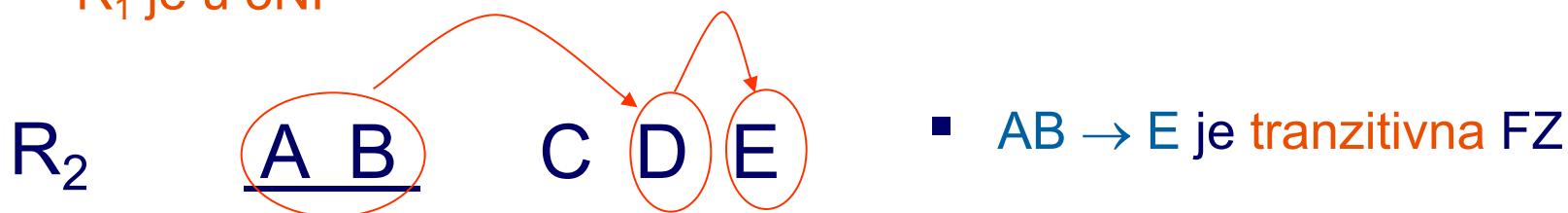
$$R_2 = ABCDE$$

$$K_{R2} = AB$$

$$F = \{ AB \rightarrow CD, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \}$$

- Jesu li  $R_1$  i  $R_2$  u 3NF?
- Postoje li u  $R_1$  i  $R_2$  neključni atributi koji tranzitivno ovise o ključu?

$R_1$  je u 3NF



Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu  $R_2$ .  
Odredite ključeve.

$$R_{21} = DE$$

$$K_{R_{21}} = D$$

$$R_{22} = ABCD$$

$$K_{R_{22}} = AB$$

## Zadatak 2 - 3NF

- Jesu li  $R_{21}$  i  $R_{22}$  u **3NF**?

$$F = \{ AB \rightarrow CD, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \}$$

$$R_1 = AF$$

$$K_{R1} = A$$

$R_1$  je u 3NF

$$R_{21} = DE$$

$$K_{R_{21}} = D$$

$R_{21}$  je u 3NF

$$R_{22} = ABCD$$

$$K_{R_{22}} = AB$$

$R_{22}$  je u 3NF

Shema baze podataka u 3NF sastoji se od relacijskih shema:

$$R_1, R_{21}, R_{22}$$

# Zadatak 3

U biblioteci se evidentiraju **posudbe (primjeraka) knjiga**.

Relacijska shema **POSUDBAPRIMJ** sastoji se od sljedećih atributa:

sifCln - šifra člana

prezCln - prezime člana

imeCln - ime člana

pbr - poštanski broj mjesta stanovanja člana

nazMj - naziv mjesta stanovanja člana

adrCln - adresa člana

invBrPrim – inventarski broj primjerka

datPos - datum posudbe

datVr – datum vraćanja (datum kad je primjerak vraćen)

sifKnj - šifra knjige

nazKnj - naziv knjige

sifIzd - šifra izdavača

nazIzd - naziv izdavača

Vrijede sljedeća pravila:

- jedna knjiga može imati više primjeraka
- jedan član istoga dana može posuditi više primjeraka
- jedan član isti primjerak može posuditi više puta, ali ne istog dana
- jedna knjiga ima jednog izdavača

# Zadatak 3

| posudbaPrimj |         |        |       |        |         | posud                                                                                                                    |
|--------------|---------|--------|-------|--------|---------|--------------------------------------------------------------------------------------------------------------------------|
| sifCln       | prezCln | imeCln | pbr   | nazMj  | adrCln  | invBrPrim, datPos, datVr, sifKnj, nazKnj, siflzd, nazlzd                                                                 |
| 123          | Novak   | Jasna  | 10000 | Zagreb | Unska 6 | <11234,3.1.2007,,567,Kiklop,12,AGM><br><21345,3.1.2007,2.2.2007,351,Geto,12,AGM><br><19435,29.1.2007,, 459,Bajke,15,VBZ> |
| 124          | Horvat  | Krešo  | 10020 | Zagreb | Siget 8 | <19435,2.1.2007,9.1.2007,459,Bajke,15,VBZ><br><23414, 2.1.2007,,398,Svila, 15, VBZ>                                      |
| 234          | Grgić   | Ana    | 10000 | Zagreb | Krčka 1 | <21345,3.2.2007,,351,Geto,12,AGM>                                                                                        |

- $K_{PosudbaPrimj} = \{sifCln\}$   
 $sifCln \rightarrow prezCln \text{ } imeCln \text{ } pbr \text{ } nazMj \text{ } adrCln \text{ } posud$
- Vrijednosti atributa *posud* su n-torke koje sadrže vrijednosti atributa:  
*invBrPrim, datPos, datVr, sifKnj, nazKnj, siflzd, nazlzd*
- Normalizirajte relacijsku shemu POSUDBAPRIMJ na 1NF (izdvajanjem atributa u novu relaciju), 2NF i 3NF

# Zadatak 3 – 1NF

clan

| sifCln | prezCln | imeCln | pbr   | nazMj  | adrCln  |
|--------|---------|--------|-------|--------|---------|
| 123    | Novak   | Jasna  | 10000 | Zagreb | Unska 6 |
| 124    | Horvat  | Krešo  | 10020 | Zagreb | Siget 8 |
| 234    | Grđić   | Ana    | 10000 | Zagreb | Krčka 1 |

CLAN = { sifCln, prezCln, imeCln, pbr, nazMj, adrCln }   K<sub>CLAN</sub> = { sifCln }

POSUDBA={ sifCln, invBrPrim, datPos, datVr, sifKnj, nazKnj, siflzd, nazlzd }

Odredite ključ za relacijsku shemu POSUDBA tako da ona zadovoljava 1NF.

posudba

K<sub>POSUDBA</sub> = { sifCln, invBrPrim, datPos }

| sifCln | invBrPrim | datPos    | datVr    | sifKnj | nazKnj | siflzd | nazlzd |
|--------|-----------|-----------|----------|--------|--------|--------|--------|
| 123    | 11234     | 3.1.2007  | NULL     | 567    | Kiklop | 12     | AGM    |
| 123    | 21345     | 3.1.2007  | 2.2.2007 | 351    | Geto   | 12     | AGM    |
| 123    | 19435     | 29.1.2007 | NULL     | 459    | Bajke  | 15     | VBZ    |
| 124    | 19435     | 2.1.2007  | 9.1.2007 | 459    | Bajke  | 15     | VBZ    |
| 124    | 23414     | 2.1.2007  | NULL     | 398    | Svila  | 15     | VBZ    |
| 234    | 21345     | 3.2.2007  | NULL     | 351    | Geto   | 12     | AGM    |

# Zadatak 3 – 2NF

clan

| sifCln | prezCln | imeCln | pbr   | nazMj  | adrCln  |
|--------|---------|--------|-------|--------|---------|
| 123    | Novak   | Jasna  | 10000 | Zagreb | Unska 6 |
| 124    | Horvat  | Krešo  | 10020 | Zagreb | Siget 8 |
| 234    | Grgić   | Ana    | 10000 | Zagreb | Krčka 1 |

CLAN zadovoljava 2NF  
- ZAŠTO?

Zadovoljava li POSUDBA 2NF?

Postoje li neključni atributi koji ovise o dijelu ključa?

Normalizirajte relacijsku shemu POSUDBA na 2NF.

posudba

| sifCln | invBrPrim | datPos    | datVr    | sifKnj | nazKnj | sifLzd | nazLzd |
|--------|-----------|-----------|----------|--------|--------|--------|--------|
| 123    | 11234     | 3.1.2007  | NULL     | 567    | Kiklop | 12     | AGM    |
| 123    | 21345     | 3.1.2007  | 2.2.2007 | 351    | Geto   | 12     | AGM    |
| 123    | 19435     | 29.1.2007 | NULL     | 459    | Bajke  | 15     | VBZ    |
| 124    | 19435     | 2.1.2007  | 9.1.2007 | 459    | Bajke  | 15     | VBZ    |
| 124    | 23414     | 2.1.2007  | NULL     | 398    | Svila  | 15     | VBZ    |
| 234    | 21345     | 3.2.2007  | NULL     | 351    | Geto   | 12     | AGM    |

# Zadatak 3 – 2NF

posudba<sub>1</sub>

| <u>sifCln</u> | <u>invBrPrim</u> | <u>datPos</u> | <u>datVr</u> |
|---------------|------------------|---------------|--------------|
| 123           | 11234            | 3.1.2007      | NULL         |
| 123           | 21345            | 3.1.2007      | 2.2.2007     |
| 123           | 19435            | 29.1.2007     | NULL         |
| 124           | 19435            | 2.1.2007      | 9.1.2007     |
| 124           | 23414            | 2.1.2007      | NULL         |
| 234           | 21345            | 3.2.2007      | NULL         |

2NF?  
OK

primjerak

| <u>invBrPrim</u> | <u>sifKnj</u> | <u>nazKnj</u> | <u>sifLzd</u> | <u>nazLzd</u> |
|------------------|---------------|---------------|---------------|---------------|
| 11234            | 567           | Kiklop        | 12            | AGM           |
| 21345            | 351           | Geto          | 12            | AGM           |
| 19435            | 459           | Bajke         | 15            | VBZ           |
| 23414            | 398           | Svila         | 15            | VBZ           |

2NF?  
OK

# Zadatak 3 – 3NF

| clan | sifCln | prezCln | imeCln | pbr   | nazMj  | adrCln  |
|------|--------|---------|--------|-------|--------|---------|
|      | 123    | Novak   | Jasna  | 10000 | Zagreb | Unska 6 |
|      | 124    | Horvat  | Krešo  | 10020 | Zagreb | Siget 8 |
|      | 234    | Grgić   | Ana    | 10000 | Zagreb | Krčka 1 |

Zadovoljava li CLAN 3NF?

Postoje li neključni atributi koji tranzitivno ovise o ključu?

Normalizirajte relacijsku shemu CLAN na 3NF.

clan<sub>1</sub>

| sifCln | prezCln | imeCln | pbr   | adrCln  |
|--------|---------|--------|-------|---------|
| 123    | Novak   | Jasna  | 10000 | Unska 6 |
| 124    | Horvat  | Krešo  | 10020 | Siget 8 |
| 234    | Grgić   | Ana    | 10000 | Krčka 1 |

3NF?  
OK

mjesto

| pbr   | nazMj  |
|-------|--------|
| 10000 | Zagreb |
| 10020 | Zagreb |

3NF?  
OK

# Zadatak 3 – 3NF

posudba<sub>1</sub>

| sifCln | invBrPrim | datPos    | datVr    |
|--------|-----------|-----------|----------|
| 123    | 11234     | 3.1.2007  | NULL     |
| 123    | 21345     | 3.1.2007  | 2.2.2007 |
| 123    | 19435     | 29.1.2007 | NULL     |
| 124    | 19435     | 2.1.2007  | 9.1.2007 |
| 124    | 23414     | 2.1.2007  | NULL     |
| 234    | 21345     | 3.2.2007  | NULL     |

primjerak



| invBrPrim | sifKnj | nazKnj | sifLzd | nazLzd |
|-----------|--------|--------|--------|--------|
| 11234     | 567    | Kiklop | 12     | AGM    |
| 21345     | 351    | Geto   | 12     | AGM    |
| 19435     | 459    | Bajke  | 15     | VBZ    |
| 23414     | 398    | Svila  | 15     | VBZ    |

POSUDBA<sub>1</sub> zadovoljava 3NF  
- ZAŠTO?

Zadovoljava li PRIMJERAK 3NF?

Postoje li neključni atributi koji tranzitivno ovise o ključu?

Normalizirajte relacijsku shemu PRIMJERAK na 3NF.

# Zadatak 3 – 3NF

primjerak<sub>1</sub>

| <u>invBrPrim</u> | sifKnj |
|------------------|--------|
| 11234            | 567    |
| 21345            | 351    |
| 19435            | 459    |
| 23414            | 398    |

3NF?  
OK

knjiga

| sifKnj | nazKnj | siflzd | nazlzd |
|--------|--------|--------|--------|
| 567    | Kiklop | 12     | AGM    |
| 351    | Geto   | 12     | AGM    |
| 459    | Bajke  | 15     | VBZ    |
| 398    | Svila  | 15     | VBZ    |



Zadovoljava li KNJIGA 3NF?

Postoje li u relacijskoj shemi KNJIGA atributi u zavisnom dijelu koji su tranzitivno ovisni o ključu?

Normalizirajte relacijsku shemu KNJIGA na 3NF.

knjiga<sub>1</sub>

| <u>sifKnj</u> | nazKnj | siflzd |
|---------------|--------|--------|
| 567           | Kiklop | 12     |
| 351           | Geto   | 12     |
| 459           | Bajke  | 15     |
| 398           | Svila  | 15     |

3NF?  
OK

izdavac

| <u>siflzd</u> | nazlzd |
|---------------|--------|
| 12            | AGM    |
| 15            | VBZ    |

3NF?  
OK

# Zadatak 3 – Shema baze podataka u 3NF

$CLAN_1 = \{ sifCln, prezCln, imeCln, pbr, adrCln \}$

$K_{CLAN1} = \{ sifCln \}$

$MJESTO = \{ pbr, nazMj \}$

$K_{MJESTO} = \{ pbr \}$

$PRIMJERAK_1 = \{ invBrPrim, sifKnj \}$

$K_{PRIMJERAK1} = \{ invBrPrim \}$

$KNJIGA_1 = \{ sifKnj, nazKnj, siflzd \}$

$K_{KNJIGA1} = \{ sifKnj \}$

$IZDAVAC = \{ siflzd, nazlzd \}$

$K_{IZDAVAC} = \{ siflzd \}$

$POSUDBA_1 = \{ sifCln, invBrPrim, datPos, datVr \}$

$K_{POSUDBA} = \{ sifCln, invBrPrim, datPos \}$

## Zadatak 4

---

Zadana je relacijska shema  $R = ABCDEF$  i na njoj skup funkcijskih zavisnosti:

$$F = \{ AB \rightarrow CDE, B \rightarrow EF, F \rightarrow B \} .$$

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacijske sheme (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

## Zadatak 4 – 1NF

---

$R = ABCDEF$

$F = \{ AB \rightarrow CDE, B \rightarrow EF, F \rightarrow B \}$

- **Odrediti primarni ključ relacije.**

$AB \rightarrow CDE$

$B \rightarrow EF \Rightarrow AB \rightarrow EF$  (A-2: uvećanje)

$\Rightarrow AB \rightarrow CDEF$  (P-1: unija)

postoji li skup  $X \subset AB$  za kojeg vrijedi  $X \rightarrow R$  ?

NE

$\Rightarrow R = ABCDEF$

$K_R = AB$

R je u 1NF

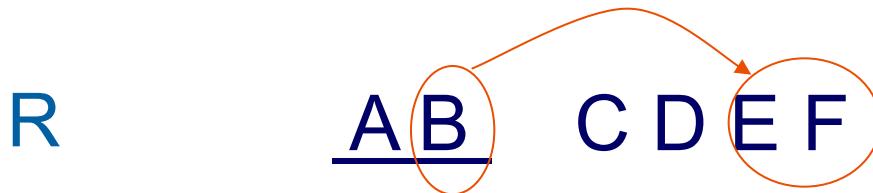
## Zadatak 4 - 2NF

$$R = ABCDEF$$

$$K_R = AB$$

$$F = \{ AB \rightarrow CD, B \rightarrow EF, F \rightarrow B \}$$

Postoje li atributi iz zavisnog dijela koji nisu potpuno funkcijски ovisni o ključu?



- $AB \rightarrow EF$  je nepotpuna FZ, jer vrijedi  $B \rightarrow EF$        $R$  nije u 2NF

Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu  $R$ .

Odredite ključeve.

$$R_1 = BEF$$

$$K_{R1} = B$$

$R_1$  je u 2NF

$$R_2 = ABCD$$

$$K_{R2} = AB$$

$R_2$  je u 2NF

## Zadatak 4 - 3NF

$$R_1 = BEF$$

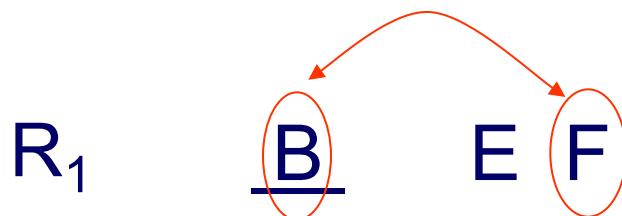
$$K_{R1} = B$$

$$R_2 = ABCD$$

$$K_{R2} = AB$$

$$F = \{ AB \rightarrow CD, B \rightarrow EF, F \rightarrow B \}$$

- Jesu li  $R_1$  i  $R_2$  u 3NF?
- Postoje li u  $R_1$  i  $R_2$  neključni atributi koji tranzitivno ovise o ključu?



- Nema tranzitivnih funkcijskih ovisnosti.
- $B \rightarrow F$  i  $F \rightarrow B$

$\Rightarrow F$  je također mogući ključ u  $R_1$

$$K1_{R1} = B$$

$$K2_{R1} = F$$

$R_1$  i  $R_2$  su u 3NF

# Zadatak 5

---

Zadane su relacijske sheme UREDJAJ i KVAR:

$$\text{UREDJAJ} = \{ \text{mbrUr}, \text{oznVrUr}, \text{nazVrUr}, \text{oznPr}, \text{nazPr} \}$$

$$K_{\text{UREDJAJ}} = \{ \text{mbrUr} \}$$

$$\text{KVAR} = \{ \text{mbrUr}, \text{datKv}, \text{oznVrKv}, \text{opVrKv}, \text{napKv} \}$$

$$K_{\text{KVAR}} = \{ \text{mbrUr}, \text{datKv}, \text{oznVrKv} \}$$

i vrijedi da se za jedan uređaj istog dana može evidentirati više različitih vrsta kvarova.

- mbrUr – matični broj uređaja
  - oznVrUr – oznaka vrste uređaja
  - nazVrUr – naziv vrste uređaja
  - oznPr - oznaka proizvođača
  - nazPr – naziv proizvođača
  - datKv – datum kvara
  - oznVrKv – oznaka vrste kvara
  - opisVrKv – opis vrste kvara
  - napKv – napomena uz kvar (napomena uz konkretni kvar na određenom uređaju određenog datuma)
- Relacijske sheme UREDJAJ i KVAR su u 1NF (provjerite!). Normalizirati te relacijske sheme na 2NF i 3NF.

# Zadatak 5 – 2NF

2NF?    UREDJAJ = { mbrUr, oznVrUr, nazVrUr, oznPr, nazPr }  
           $K_{UREDJAJ} = \{ mbrUr \}$

UREDJAJ zadovoljava 2NF (zašto?).

KVAR = { mbrUr, datKv, oznVrKv, opVrKv, napKv }  
           $K_{KVAR} = \{ mbrUr, datKv, oznVrKv \}$

Postoje li neključni atributi koji ne ovise o čitavom ključu nego samo o dijelu ključa?  
Normalizirajte relacijsku shemu KVAR na 2NF.

VRSTAKVARA = { oznVrKv, opVrKv }  
           $K_{VRSTAKVARA} = \{ oznVrKv \}$

$KVAR_1 = \{ mbrUr, datKv, oznVrKv, napKv \}$   
           $K_{KVAR1} = \{ mbrUr, datKv, oznVrKv \}$

# Zadatak 5 – 3NF

Postoje li neključni atributi koji tranzitivno ovise o ključu?

$\text{UREDJAJ} = \{ \text{mbrUr}, \text{oznVrUr}, \text{nazVrUr}, \text{oznPr}, \text{nazPr} \}$

$K_{UREDJAJ} = \{ \text{mbrUr} \}$



Normalizirajte relacijsku shemu UREDJAJ na 3NF.

$\text{VRSTAUREDJ} = \{ \text{oznVrUr}, \text{nazVrUr} \}$

$K_{VRSTAUREDJ} = \{ \text{oznVrUr} \}$

$\text{PROIZVODJAC} = \{ \text{oznPr}, \text{nazPr} \}$

$K_{PROIZVODJAC} = \{ \text{oznPr} \}$

$\text{UREDJAJ}_1 = \{ \text{mbrUr}, \text{oznVrUr}, \text{oznPr} \}$

$K_{UREDJAJ1} = \{ \text{mbrUr} \}$

**3NF?**

**OK**

# Zadatak 5 – 3NF

---

$VRSTAKVARA = \{ oznVrKv, opVrKv \}$

3NF OK

$K_{VRSTAKVARA} = \{ oznVrKv \}$

$KVAR1 = \{ mbrUr, datKv, oznVrKv, napKv \}$

3NF OK

$K_{KVAR1} = \{ mbrUr, datKv, oznVrKv \}$

Shema baze podataka u 3NF sastoji se od relacijskih shema:  
 $VRSTAUREDJ$ ,  $PROIZVODJAC$ ,  $UREDJAJ_1$ ,  $VRSTAKVARA$ ,  $KVAR_1$

# Zadatak 6

Zadane su relacijske sheme LINIJA i PROMET:

$$\text{LINIJA} = \{ \text{lin}, \text{sifOdr}, \text{nazOdr}, \text{vrijPol}, \text{trVoz} \} \quad K_{\text{LINIJA}} = \{ \text{lin} \}$$

$$\text{PROMET} = \{ \text{lin}, \text{sifPrij}, \text{nazPrij}, \text{sifAut}, \text{tipAut}, \text{datPol}, \text{brSjed}, \text{brKart} \}$$
$$K_{\text{PROMET}} = \{ \text{lin}, \text{sifPrij}, \text{sifAut}, \text{datPol} \}$$

- lin – broj linije na kojoj se odvija promet
- sifPrij – šifra prijevoznika (poduzeća)
- nazPrij – naziv prijevoznika
- sifAut - šifra autobusa – određuje je prijevoznik
- tipAut – tip autobusa
- brSjed – broj sjedala
- sifOdr – šifra mjesta - odredišta
- nazOdr – naziv mjesta - odredišta
- datPol – datum polaska
- vrijPol – vrijeme polaska
- trVoz – trajanje vožnje
- brKart – broj prodanih karata

Relacijske sheme PROMET i LINIJA su u 1NF (provjeriti!)

# Zadatak 6

---

$LINIJA = \{ \text{lin}, \text{sifOdr}, \text{nazOdr}, \text{vrijPol}, \text{trVoz} \}$

$K_{LINIJA} = \{ \text{lin} \}$

$PROMET = \{ \text{lin}, \text{sifPrij}, \text{nazPrij}, \text{sifAut}, \text{tipAut}, \text{datPol}, \text{brSjed}, \text{brKart} \}$

$K_{PROMET} = \{ \text{lin}, \text{sifPrij}, \text{sifAut}, \text{datPol} \}$

Normalizirati navedene relacijske sheme na 2NF i 3NF ako vrijedi:

- linija određuje odredište, vrijeme polaska i trajanje vožnje
- istog dana na istoj liniji može prometovati više autobusa (istog ili različitih prijevoznika)
- šifru autobraza određuje prijevoznik – mogu postojati različiti autobusi različitih prijevoznika koji imaju istu šifru
- autobusi istog tipa imaju jednak broj sjedala

# Zadatak 6 – 2NF

2NF? LINIJA = { lin, sifOdr, nazOdr, vrijPol, trVoz } 2NF OK

PROMET = { lin, sifPrij, nazPrij, sifAut, tipAut, datPol, brSjed, brKart }

- šifru autobusa određuje prijevoznik – mogu postojati različiti autobusi različitih prijevoznika koji imaju istu šifru

sifPrij sifAut → tipAut brSjed

Normalizirajte relacijsku shemu PROMET na 2NF.

PRIJEVOZNIK = { sifPrij, nazPrij }

K<sub>PRIJEVOZNIK</sub> = { sifPrij }

AUTOBUS = { sifPrij, sifAut, tipAut, brSjed }

K<sub>AUTOBUS</sub> = { sifPrij, sifAut }

PROMET<sub>1</sub> = { lin, sifPrij, sifAut, datPol, brKart }

K<sub>PROMET1</sub> = { lin, sifPrij, sifAut, datPol }

# Zadatak 6 – 3NF

---

$LINIJA = \{ \text{lin}, \text{sifOdr}, \text{nazOdr}, \text{vrijPol}, \text{trVoz} \}$        $K_{LINIJA} = \{ \text{lin} \}$     3NF?

Normalizirajte relacijsku shemu LINIJA na 3NF.

$ODREDISTE = \{ \text{sifOdr}, \text{nazOdr} \}$        $K_{ODREDISTE} = \{ \text{sifOdr} \}$

$LINIJA_1 = \{ \text{lin}, \text{sifOdr}, \text{vrijPol}, \text{trVoz} \}$        $K_{LINIJA1} = \{ \text{lin} \}$

$PRIJEVOZNIK = \{ \text{sifPrij}, \text{nazPrij} \}$        $K_{PRIJEVOZNIK} = \{ \text{sifPrij} \}$     3NF OK

$PROMET_1 = \{ \text{lin}, \text{sifPrij}, \text{sifAut}, \text{datPol}, \text{brKart} \}$

$K_{PROMET1} = \{ \text{lin}, \text{sifPrij}, \text{sifAut}, \text{datPol} \}$

3NF?

OK

# Zadatak 6 – 3NF

AUTOBUS = { sifPrij, sifAut, tipAut, brSjed }

3NF?

$K_{AUTOBUS}$  = { sifPrij, sifAut }

- autobusi istog tipa imaju jednak broj sjedala

Normalizirajte relacijsku shemu AUTOBUS na 3NF.

TIPAUTOB = { tipAut, brSjed }

$K_{TIPAUTOB}$  = { tipAut }

AUTOBUS<sub>1</sub> = { sifPrij, sifAut, tipAut }

$K_{AUTOBUS_1}$  = { sifPrij, sifAut }

3NF OK

Shema baze podataka u 3NF sastoji se od relacijskih shema:

ODREDISTE, LINIJA<sub>1</sub>, PRIJEVOZNIK,  
PROMET<sub>1</sub>, TIPAUTOB, AUTOBUS<sub>1</sub>

# Zadatak 7

---

- Zadana je relacijska shema FILMOVI sa sljedećim atributima:
  - sifFilm – šifra filma
  - nazFilm – naziv filma
  - sifDistrib – šifra distributera
  - nazDistrib – naziv distributera
  - sifFun – šifra funkcije
  - nazFun – naziv funkcije u filmu (glumac, režiser, scenarist, ...)
  - sifOsoba – šifra osobe
  - prezOsoba – prezime osobe
  - imeOsoba – ime osobe
- Normalizirati relacijsku shemu FILMOVI na 1NF, 2NF i 3NF ako vrijede pravila:
  - jedan film ima jednog distributera,
  - jedna osoba može u jednom filmu imati različite uloge (npr. glumac i režiser),
  - u jednom filmu više osoba može imati istu ulogu (npr. glumac)

# Zadatak 7 – rješenje - 1/4

---

FILMOVI = sifFilm, nazFilm, sifDistrib, nazDistrib, sifFun, nazFun, sifOsoba, prezOsoba, imeOsoba

**1NF – odrediti ključ**

FILMOVI = sifFilm, nazFilm, sifDistrib, nazDistrib, sifFun, nazFun, sifOsoba, prezOsoba, imeOsoba

Napomena: atributi koji čine ključ su potcrtani, odnosno, vrijedi da je:

$K_{FILMOVI} = \{ sifFilm, sifFun, sifOsoba \}$

**1NF – OK**

To znači da u relacijskoj shemi FILMOVI vrijedi funkcija zavisnost (FZ):

$sifFilm, sifFun, sifOsoba \rightarrow nazFilm, sifDistrib, nazDistrib, nazFun, prezOsoba, imeOsoba$

# Zadatak 7 – rješenje - 2/4

FILMOVI = sifFilm, nazFilm, sifDistrib, nazDistrib, sifFun, nazFun, sifOsoba, prezOsoba, imeOsoba

**2NF – odrediti nepotpune FZ i u skladu s tim obaviti dekompozicije**

- **FZ: sifFilm, sifFun, sifOsoba → nazFilm, sifDistrib, nazDistrib**  
**JE NEPOTPUNA**, jer vrijedi i  $\text{sifFilm} \rightarrow \text{nazFilm, sifDistrib, nazDistrib}$  (1)  
Dekompozicija zbog(1):  
FILMOVI1 = sifFilm, nazFilm, sifDistrib, nazDistrib      **2NF - OK**  
FILMOVI2 = sifFilm, sifFun, nazFun, sifOsoba, prezOsoba, imeOsoba **2NF ?**
- **FZ: sifFilm, sifFun, sifOsoba → nazFun**  
**JE NEPOTPUNA**, jer vrijedi i  $\text{sifFun} \rightarrow \text{nazFun}$  (2)  
Dekompozicija zbog (2):  
FUNKCIJA = sifFun, nazFun **2NF - OK**  
FILMOVI2 = sifFilm, sifFun, sifOsoba, prezOsoba, imeOsoba **2NF ?**
- **FZ: sifFilm, sifFun, sifOsoba → prezOsoba, imeOsoba**  
**JE NEPOTPUNA**, jer vrijedi i  $\text{sifOsoba} \rightarrow \text{prezOsoba, imeOsoba}$  (3)  
Dekompozicija zbog (3):  
OSOBA = sifOsoba, prezOsoba, imeOsoba      **2NF - OK**  
FUN\_OSOBA\_FILM = sifFilm, sifFun, sifOsoba      **2NF- OK**

## Zadatak 7 – rješenje - 3/4

## **DISKUSIJA – 2NF**

FILMOVI2 = sifFilm, sifFun, sifOsoba, prezOsoba, imeOsoba

Normalizacijom relacijske sheme FILMOVI2 zbog toga što je

FZ: sifFilm, sifFun, sifOsoba → prezOsoba, imeOsoba **JE NEOTPUNA**, jer vrijedi i  
sifOsoba → prezOsoba, imeOsoba

## Nastaju sheme:

OSOBA = sifOsoba, prezOsoba, imeOsoba - nastala na temelju FZ (3)

**FUN\_OSOBA\_FILM** = sifFilm, sifFun, sifOsoba

- u ovoj shemi ostaje ključ originalne relacije (FILMOVI2) i neključni atributi koji su preostali
  - u ovom slučaju nema atributa koji su preostali
  - ako bismo izbacili/zaboravili ovu relacijsku shemu – **dekompozicija ne bi bila obavljena bez gubitaka informacija!**

Neka su filmovi, filmovi1, funkcija i osoba, relacije definirane na pripadnim relacijskim shemama:

filmovi (FILMOVI); filmovi1 (FILMOVI1); funkcija(FUNKCIJA); osoba (OSOBA)

Tada:

filmovi1 (FILMOVI1) ▷◁ funkcija(FUNKCIJA) ▷◁ osoba (OSOBA) ≠ filmovi (FILMOVI)

**Rezultat operacije : filmovi1 (FILMOVI1) ▷◁ funkcija(FUNKCIJA) ▷◁ osoba (OSOBA)**  
je jednak njihovom Kartezijevom produktu! **(ZAŠTO?)**

# Zadatak 7 – rješenje - 4/4

Normalizacijom relacijske sheme FILMOVI na 2NF nastale su relacijske sheme:

|                                                                   |                 |                 |
|-------------------------------------------------------------------|-----------------|-----------------|
| FILMOVI1 = <u>sifFilm</u> , nazFilm, sifDistrib, nazDistrib       | <b>2NF – OK</b> |                 |
| FUNKCIJA = <u>sifFun</u> , nazFun                                 | <b>2NF – OK</b> | <b>3NF - OK</b> |
| OSOBA = <u>sifOsoba</u> , prezOsoba, imeOsoba                     | <b>2NF – OK</b> | <b>3NF - OK</b> |
| FUN_OSOBA_FILM = <u>sifFilm</u> , <u>sifFun</u> , <u>sifOsoba</u> | <b>2NF - OK</b> | <b>3NF – OK</b> |

Relacijske sheme: FUNKCIJA, OSOBA i FUN\_OSOBA\_FILM zadovoljavaju i 3NF. (**ZAŠTO?**)

Relacijska shema:

FILMOVI1 = sifFilm, nazFilm, sifDistrib, nazDistrib  
ne zadovoljava 3NF zbog toga što postoji FZ: sifDistrib → nazDistrib

Atribut nazDistrib je tranzitivno ovisan o ključu (sifFilm):

$\text{sifFilm} \rightarrow \text{sifDistrib}$                     NE POSTOJI:  $\text{sifDistrib} \rightarrow \text{sifFilm}$   
 $\text{sifDistrib} \rightarrow \text{nazDistrib}$

Normalizacijom na 3NF nastaju relacijske sheme:

DISTRIB = sifDistrib, nazDistrib  
FILM = sifFilm, nazFilm, sifDistrib

Shema baze podataka sastoji se od shema: **FILM, DISTRIB, FUNKCIJA, OSOBA, FUN\_OSOBA\_FILM**

# Baze podataka

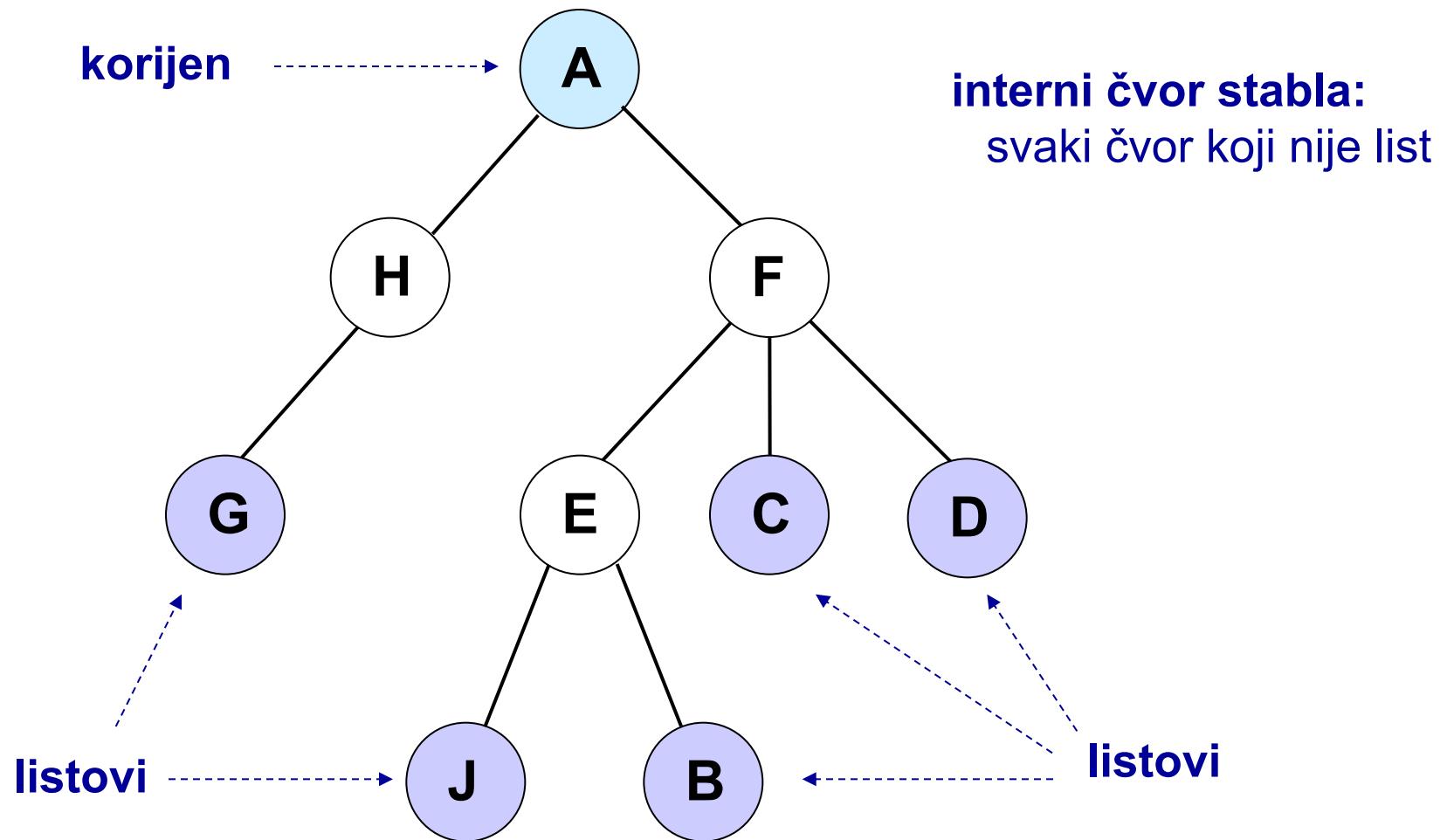
# Dopunski materijali (za one koji žele znati više)

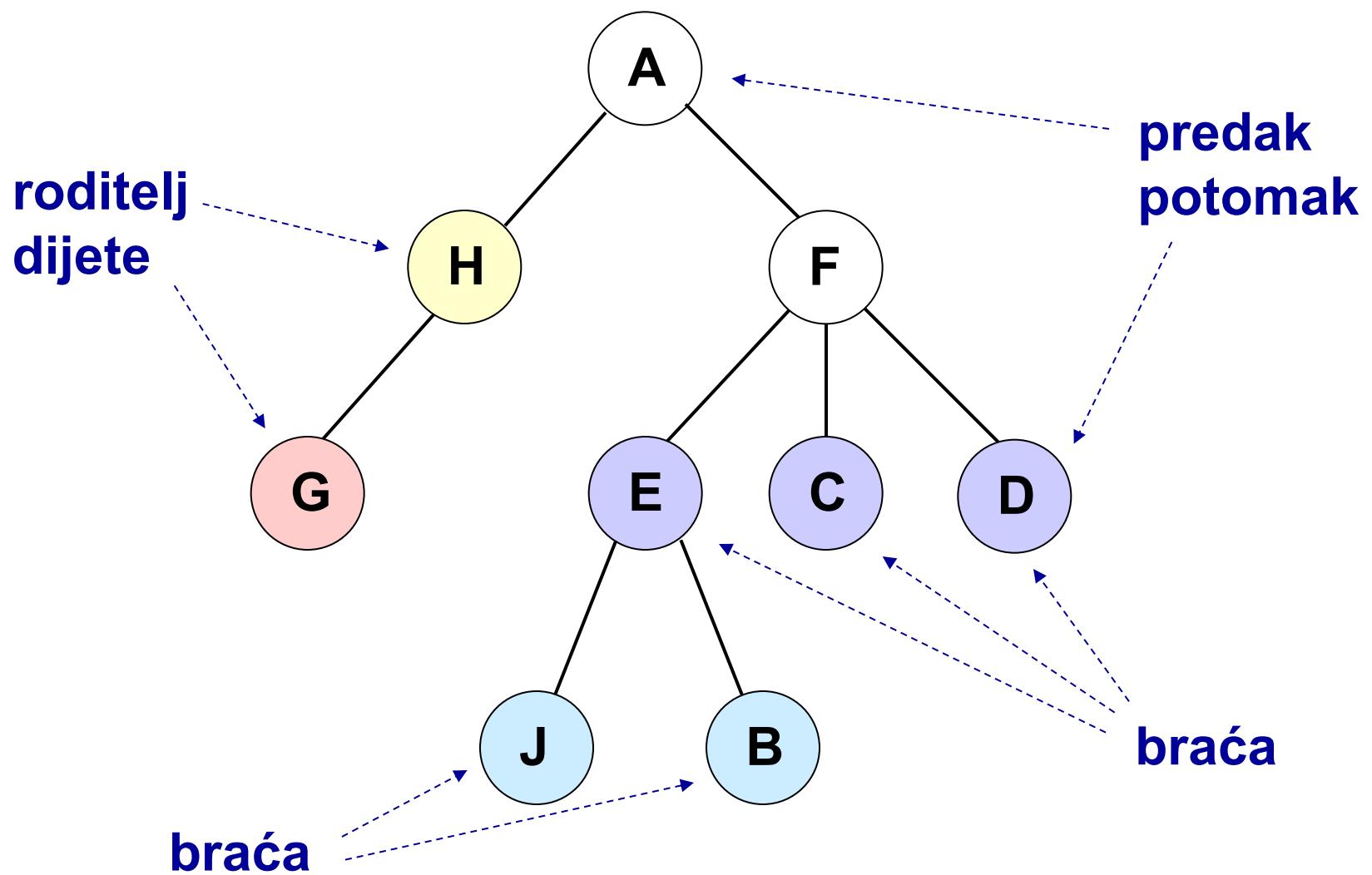
# B - stabla

# Travanj, 2020.



# 1. Stablo kao struktura podataka



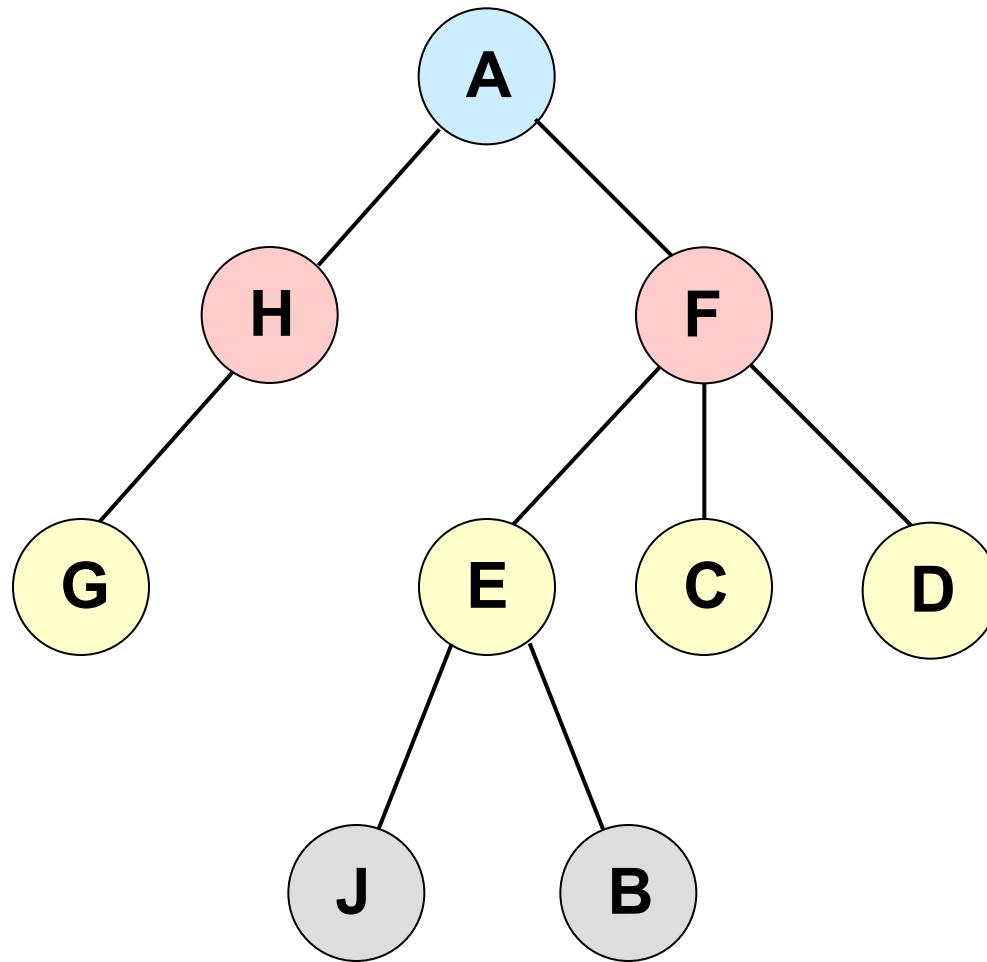


razina 0

razina 1

razina 2

razina 3



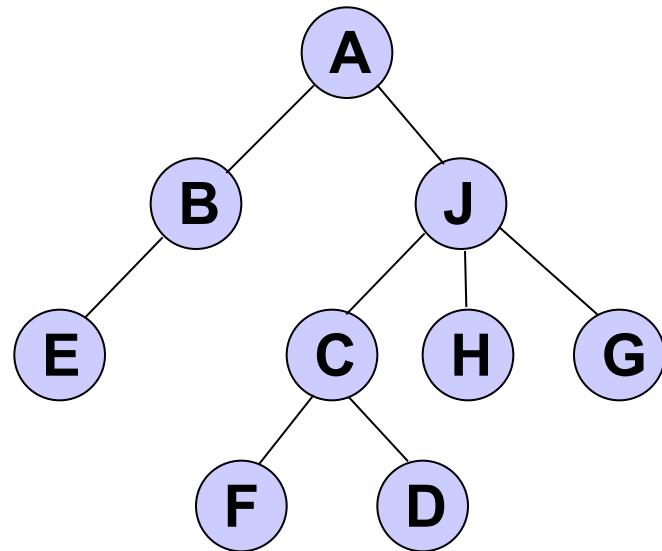
**razina čvora (level):** duljina puta od korijena do čvora

**dubina stabla (depth):** najveća duljina puta od korijena do lista

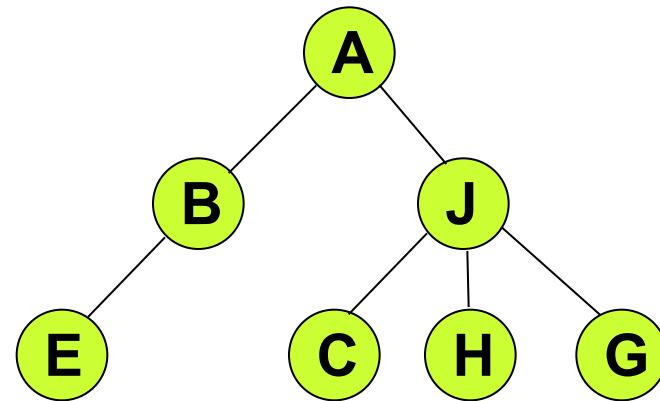
**red stabla (order):** najveći broj djece koje čvor može imati

Stablo je **balansirano (balanced)** ukoliko je duljina puta od korijena do lista jednaka za svaki list u stablu

stablo nije balansirano



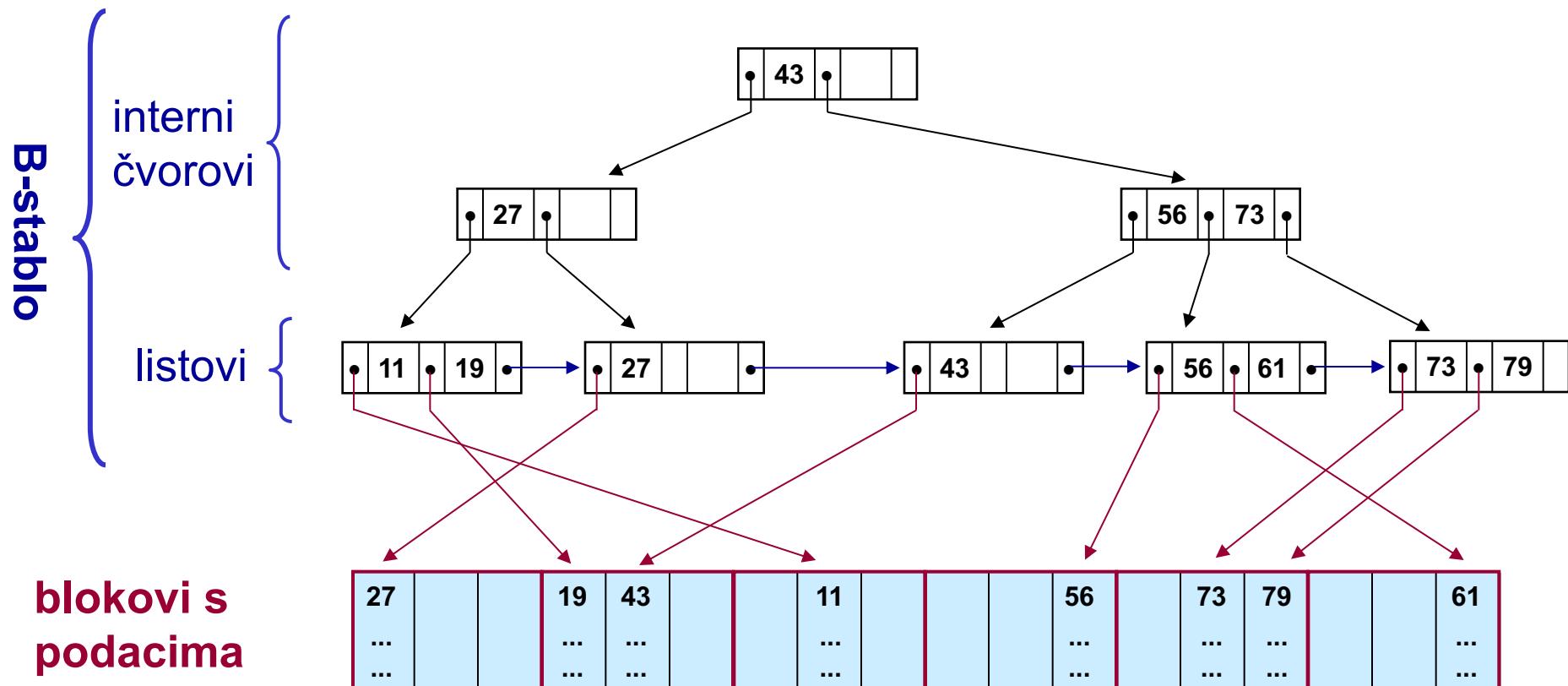
stablo je balansirano



Oznaka **B** u B-stablo znači "**balansirano!**"!

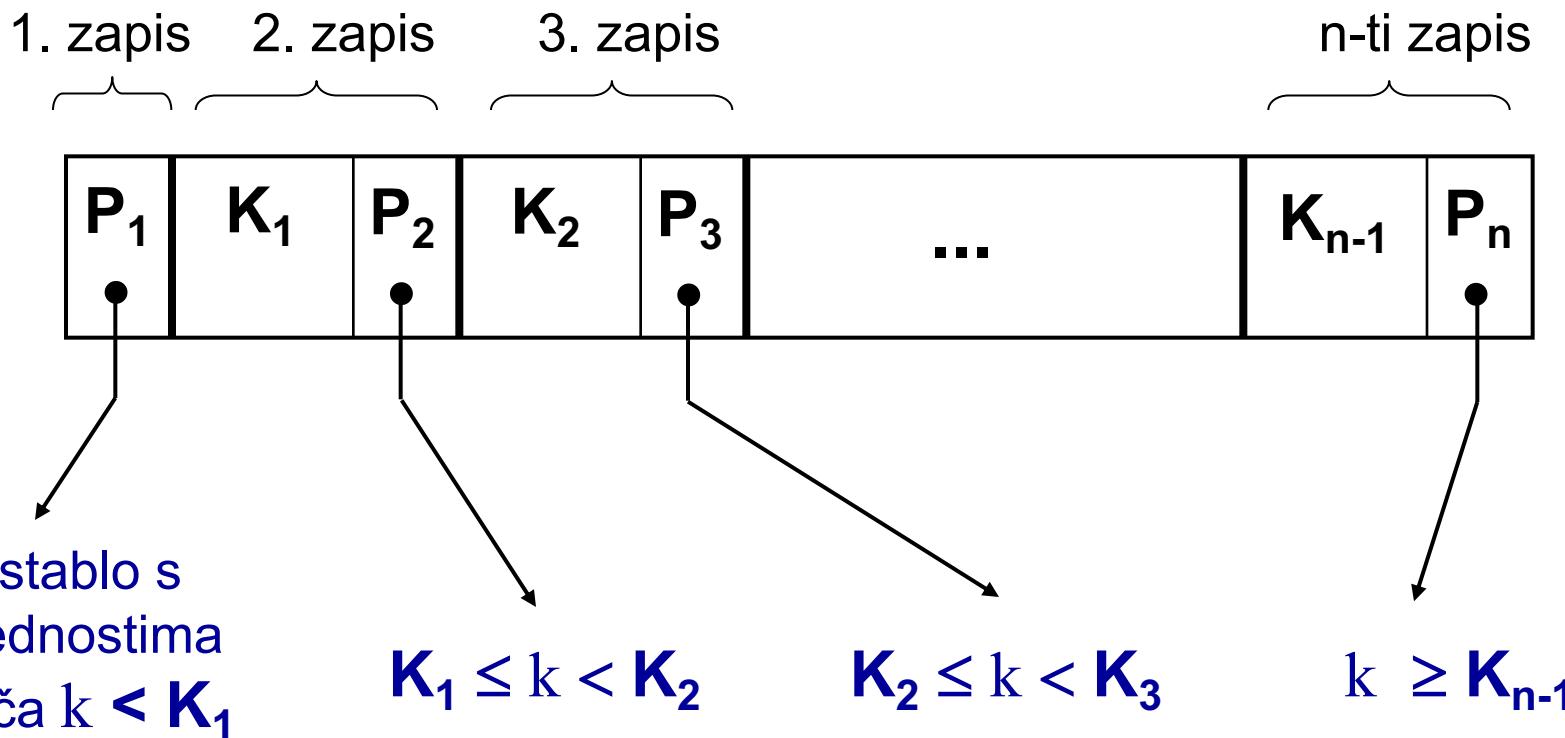
## 2. Struktura B-stabla

Opisujemo varijantu B-stabla koja se naziva **B<sup>+</sup>-stablo**. Opisi ostalih varijanti B-stabala (B\*-stablo, B-stablo, ...) mogu se pronaći u literaturi.



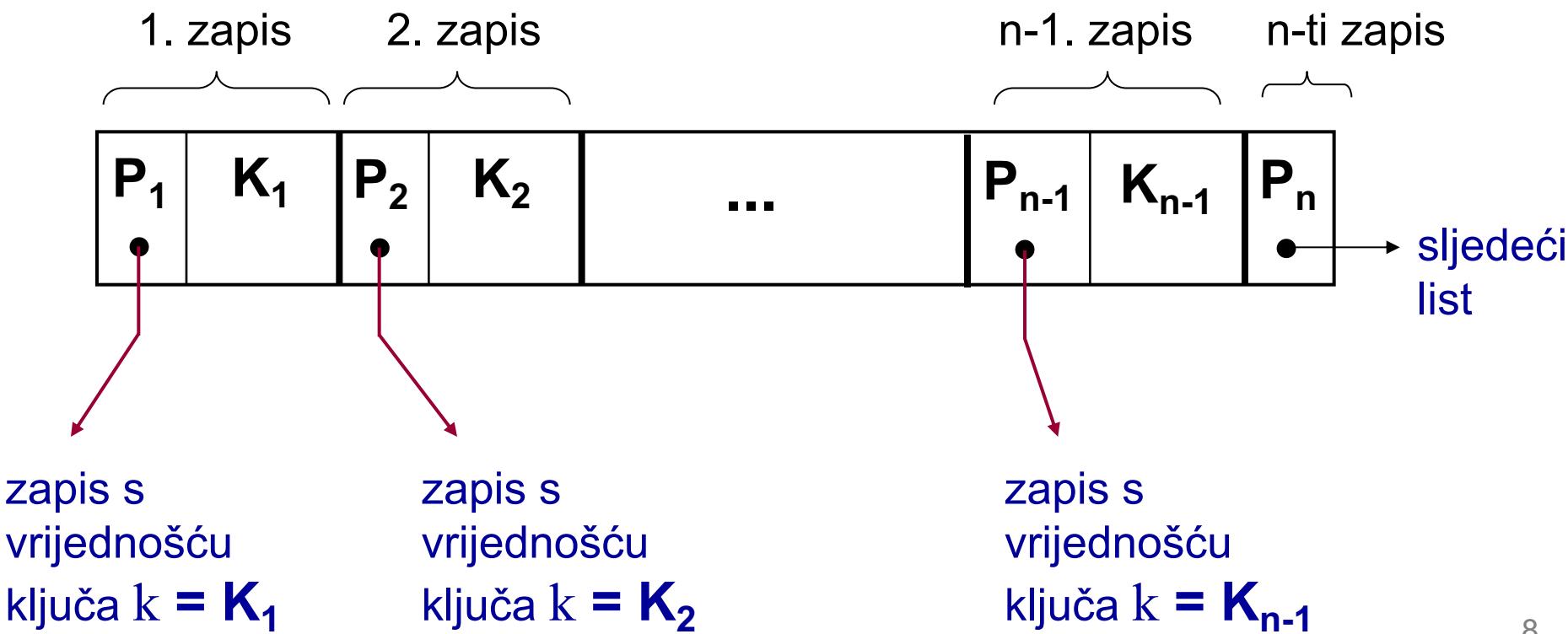
## Struktura internog čvora

- U B<sup>+</sup>-stablu reda  $n$ , interni čvor sadrži
  - najviše  $n$  kazaljki
  - najmanje  $\lceil n/2 \rceil$  kazaljki  $\rightarrow \lceil a \rceil$  je najmanji cijeli broj  $\geq a$ 
    - ovo ograničenje ne vrijedi za korijen ( $2 \dots n$  kazaljki)
- uz  $p$  kazaljki u čvoru, broj pripadnih vrijednosti  $K_i$  u čvoru je  $p-1$

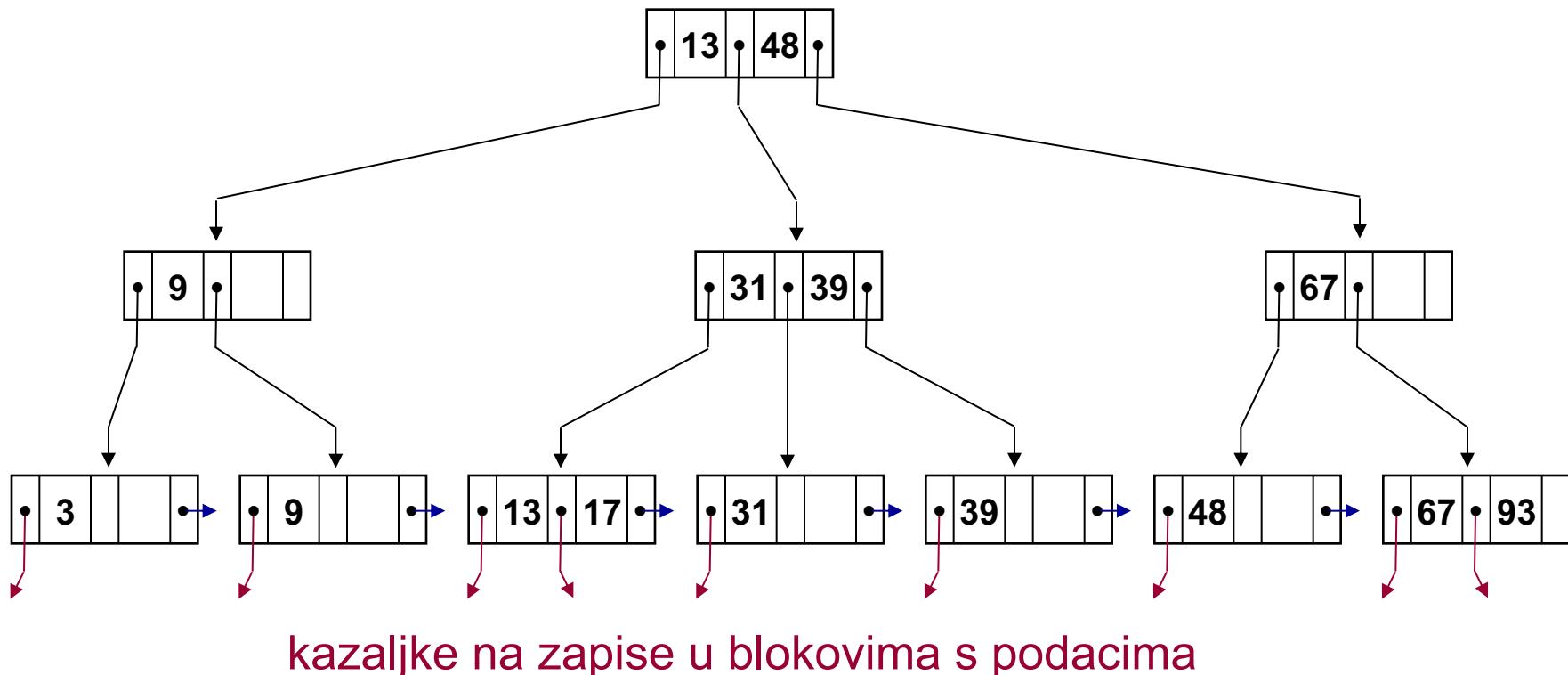


# Struktura lista

- U B<sup>+</sup>-stablu reda  $n$ , list sadrži
  - najviše  $n-1$  vrijednosti  $K_i$  i pripadnih kazaljki na zapise
  - najmanje  $\lceil (n-1)/2 \rceil$  vrijednosti  $K_i$  i pripadnih kazaljki na zapise
  - svi listovi, osim krajnje desnog, sadrže kazaljku na sljedeći list
    - omogućuju se upiti tipa od-do



# Primjer B<sup>+</sup>-stabla reda 3



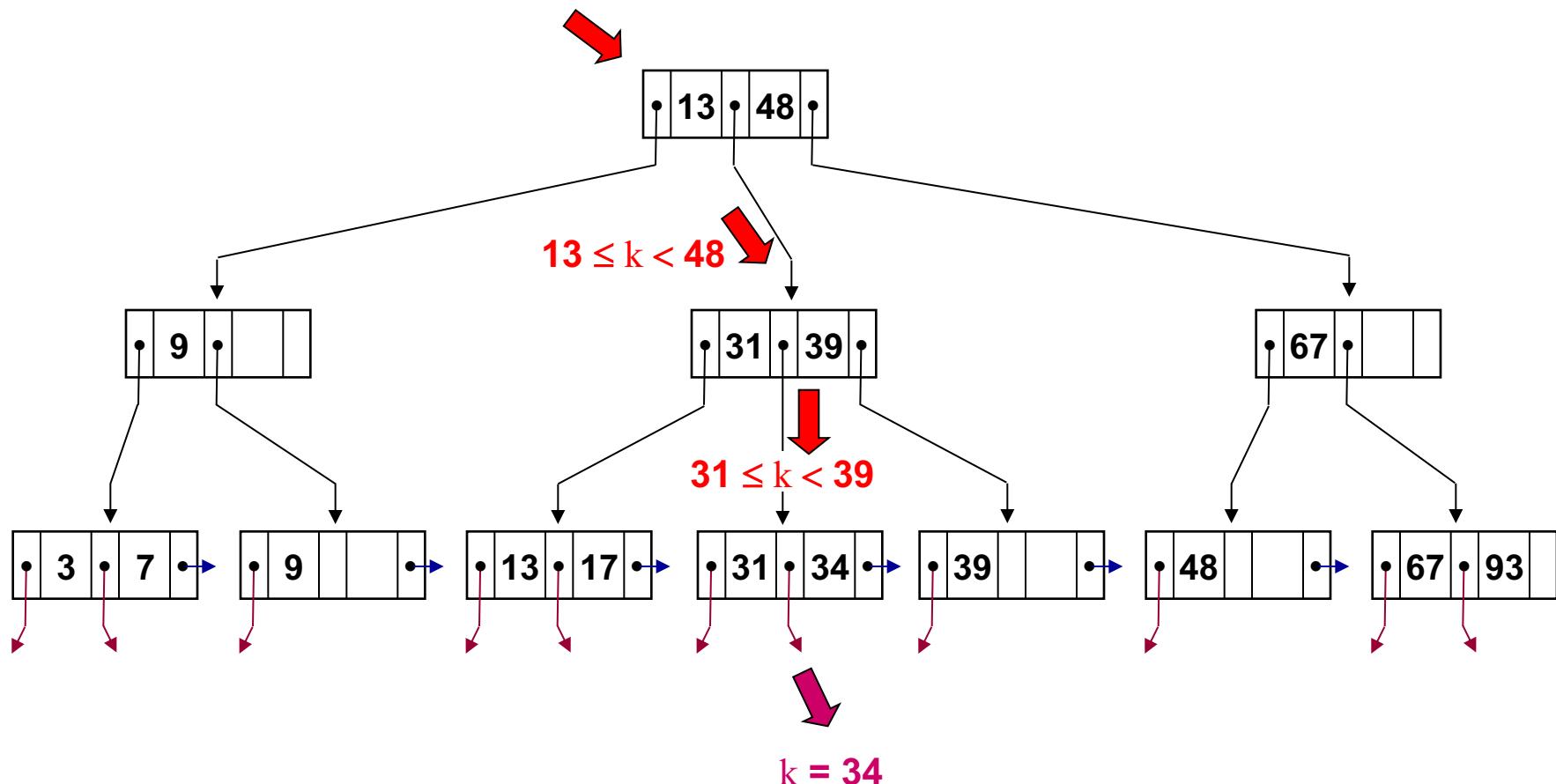
Odabire se red stabla čija primjena rezultira veličinom čvora koja odgovara veličini fizičkog bloka. Ovisno o veličini ključa, red stabla je uglavnom veličine 10 - 200.

### **3. Algoritmi za B-stablo**

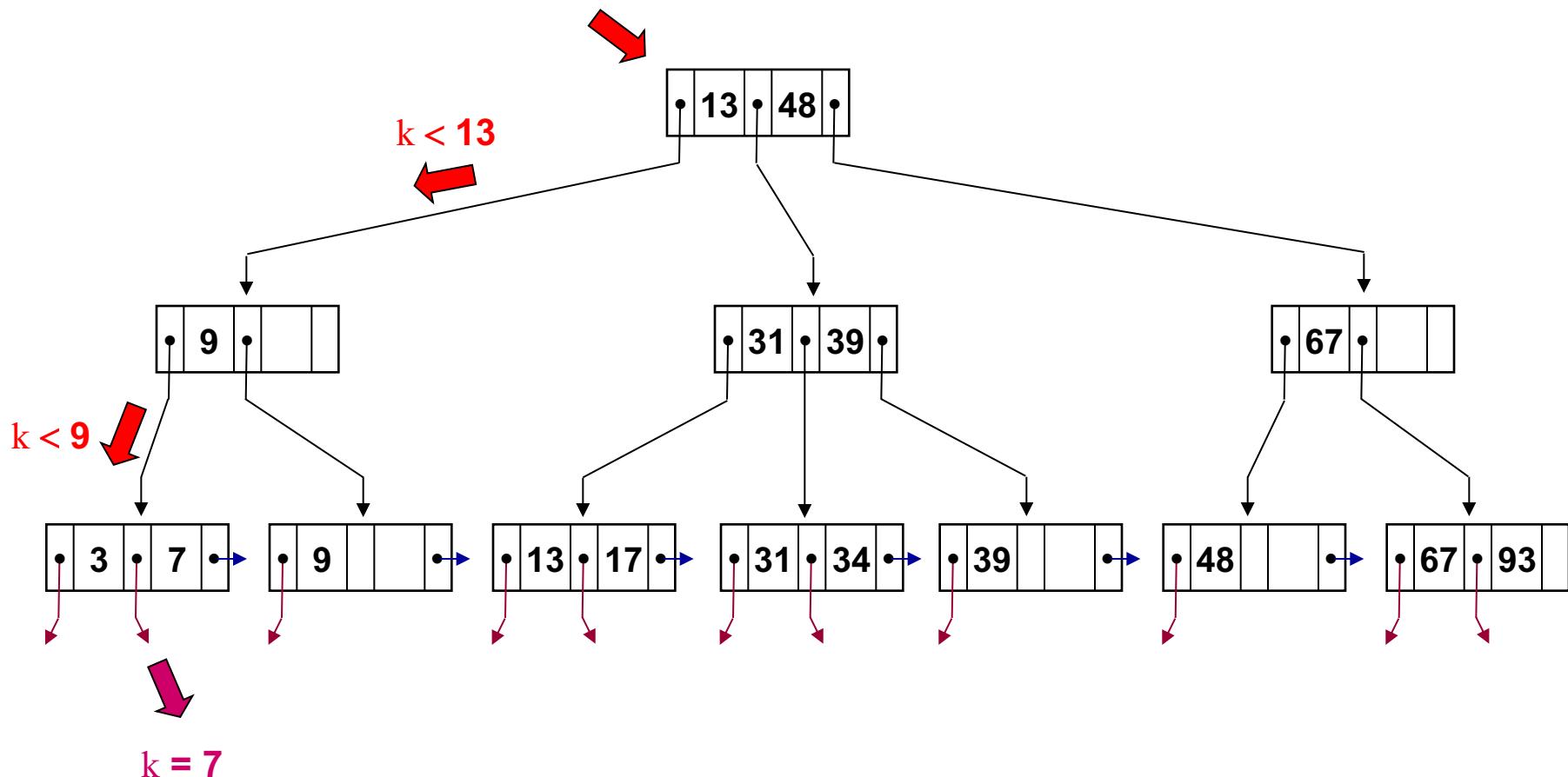
## Algoritam za pretragu B<sup>+</sup>-stabla

- algoritam za traženje zapisa s ključem vrijednosti k je rekurzivan
  - cilj je u svakom koraku rekurzije (pretraga i-te razine) pronaći čvor na nižoj, (i+1)-voj razini, koji će voditi prema listu u kojem se nalazi ključ čija je vrijednost k
- traženje zapisa započinje od korijena (0-te razine)
- u čvoru i-te razine potrebno je pronaći najveću vrijednost ključa koja je manja ili jednaka traženoj vrijednosti k
  - za prvu kazaljku internog čvora nije navedena vrijednost ključa, pa ona "pokriva" sve vrijednosti ključeva manje od prve vrijednosti ključa ( $K_1$ ) navedene u čvoru
- nakon pronalaženja odgovarajuće vrijednosti ključa, slijedi se pripadna kazaljka i time se obavlja pozicioniranje na (i+1)-vu razinu
- postupak se ponavlja rekursivno sve dok se ne dođe do lista. U njemu se mora nalaziti, ukoliko postoji, ključ čija je vrijednost k, te pripadna kazaljka prema traženom zapisu u datoteci

traži se zapis s ključem 34



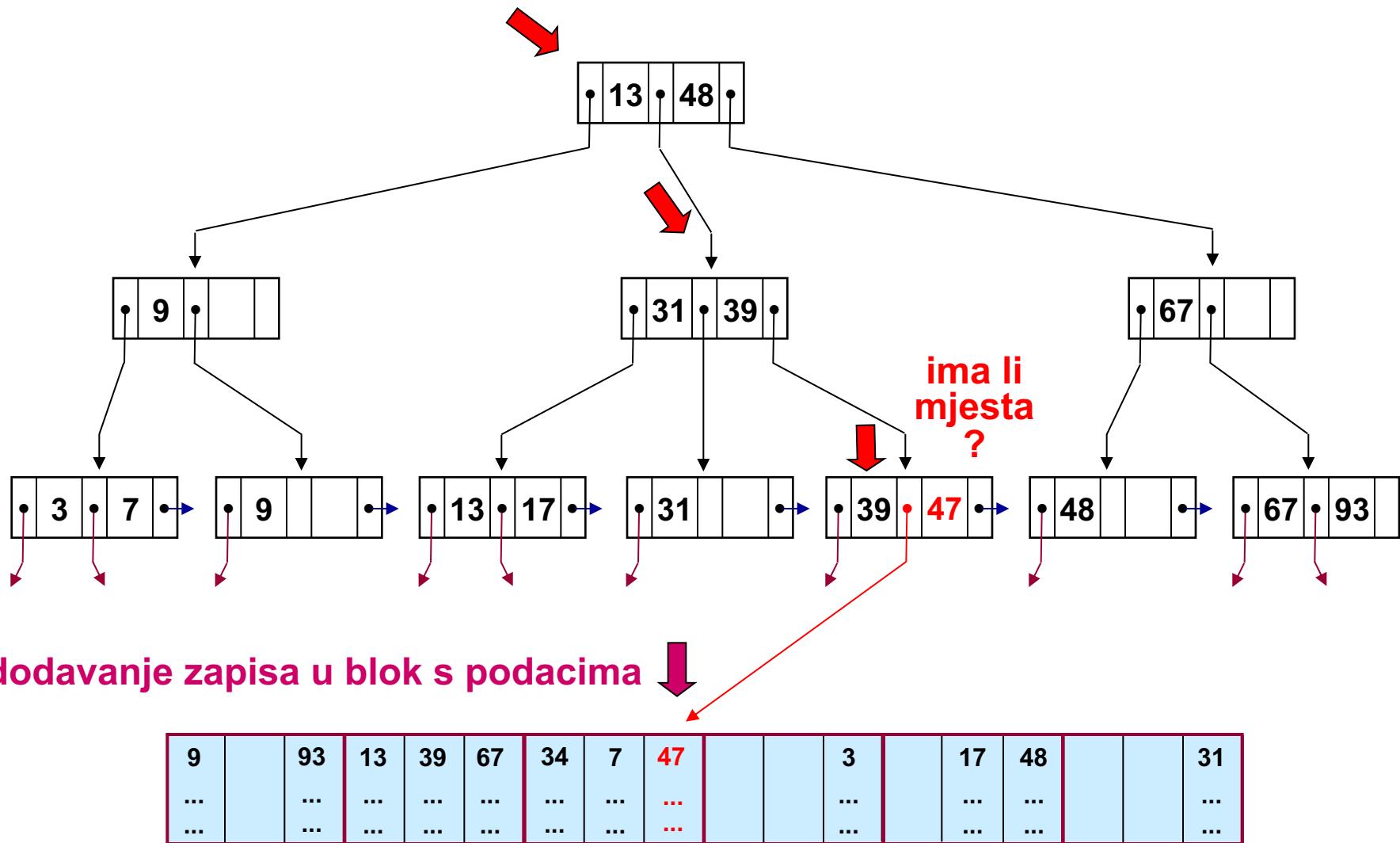
traži se zapis s ključem 7



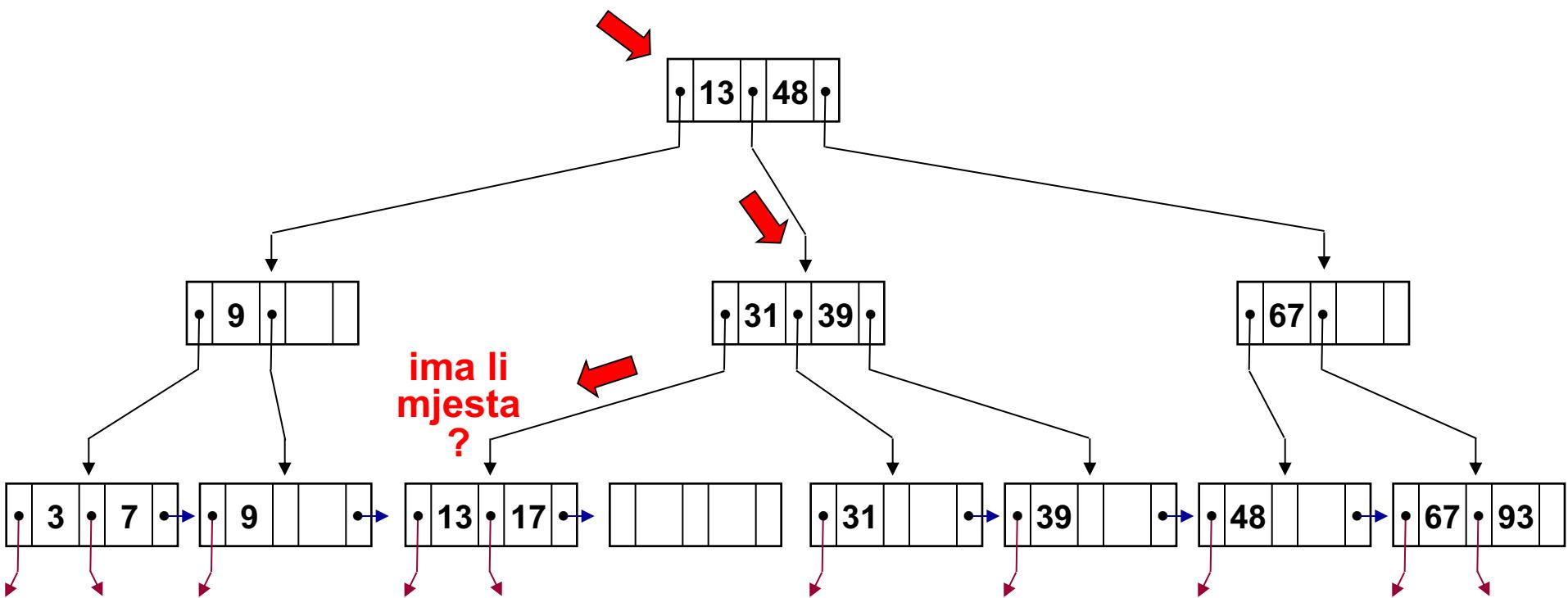
## Algoritam za dodavanje zapisa u B<sup>+</sup>-stablu

- zapis (podaci) se upisuje u jedan od slobodnih blokova s podacima
- obavlja se algoritam za pronalaženje lista L u koji pripada vrijednost ključa k
- ako čvor L nije popunjen, u čvor se dodaje zapis s ključem k i kazaljkom na pripadni zapis u datoteci, uz očuvanje poretku vrijednosti ključeva
  - nova vrijednost nikad nije prva u čvoru, osim u slučaju krajnjeg lijevog čvora
- ako je čvor L popunjen
  - stvara se novi čvor i zapisi među njima se podijele, pri čemu svakom od čvorova pripadne polovica zapisa
  - budući da je dodan novi čvor, u nadređeni čvor potrebno je dodati zapis s kazaljkom i najmanjom vrijednošću ključa u novom čvoru
  - za dodavanje novog zapisa u nadređeni čvor koristi se ista procedura kao za dodavanje zapisa u čvor na nižoj razini
  - postupak je rekurzivan i mora se obaviti za svaku nadređenu razinu (sve dok se ne dođe do korijena ili se na nekoj od razina nađe dovoljno mesta za upis vrijednosti ključa i kazaljke, bez dodavanja novih čvorova).
  - ako se dođe do korijena, može se desiti da u korijenu nema mesta za novi zapis. Tada se dodaje novi čvor i formira se novi korijen na višoj razini

## dodavanje zapisa s ključem 47

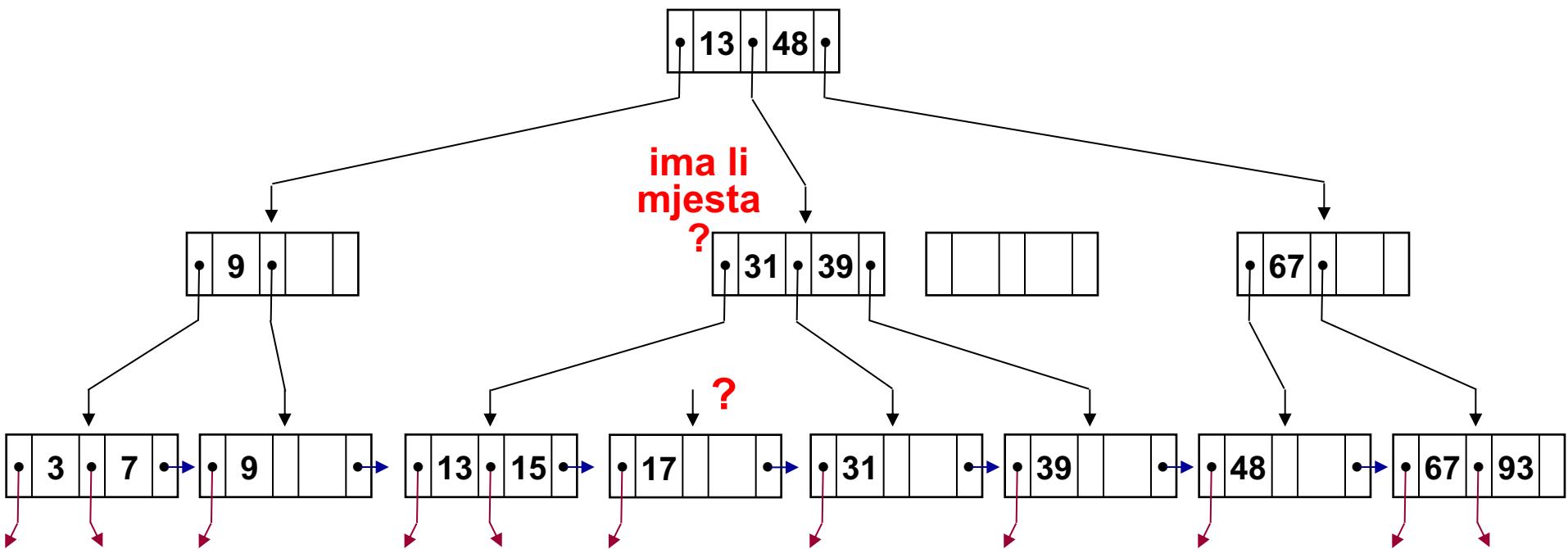


## dodavanje zapisa s ključem 15



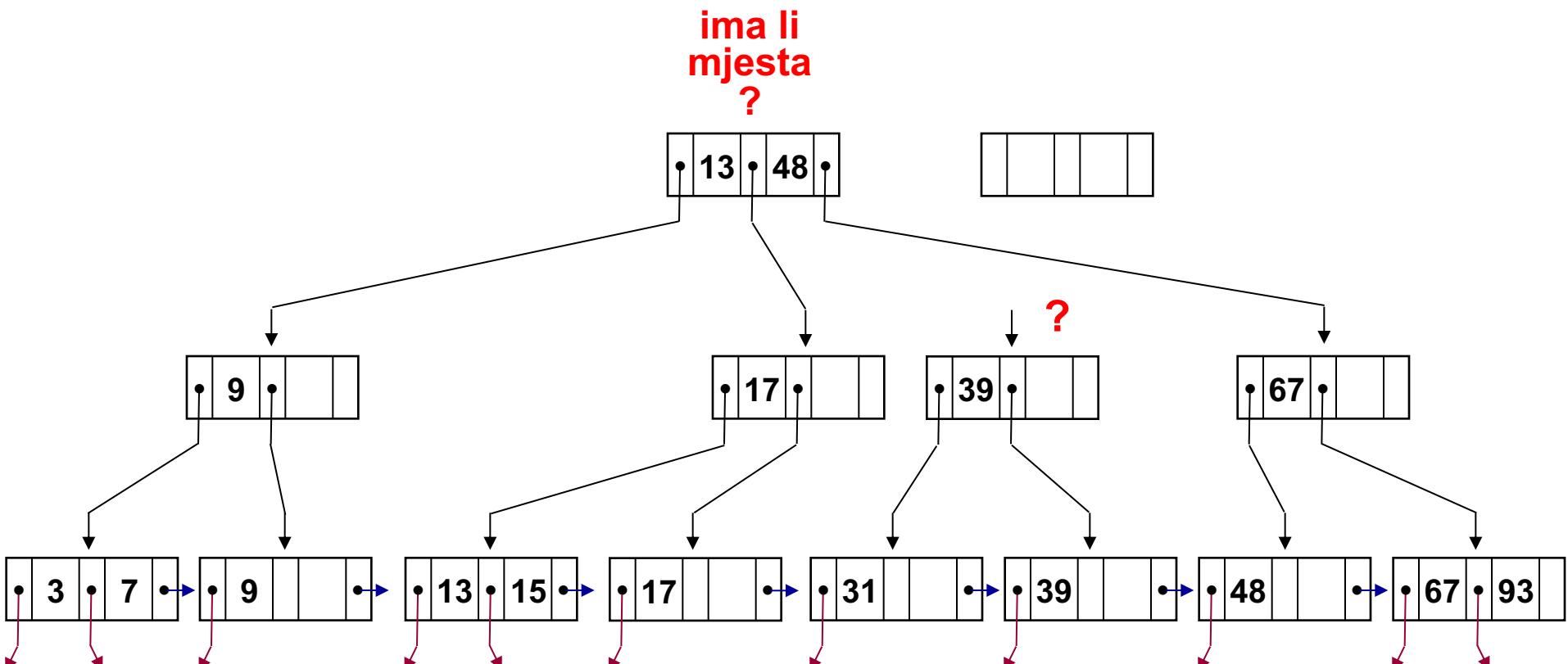
dodati novi čvor i podijeliti zapise  
13, 15, 17 među čvorovima

## dodavanje zapisa s ključem 15



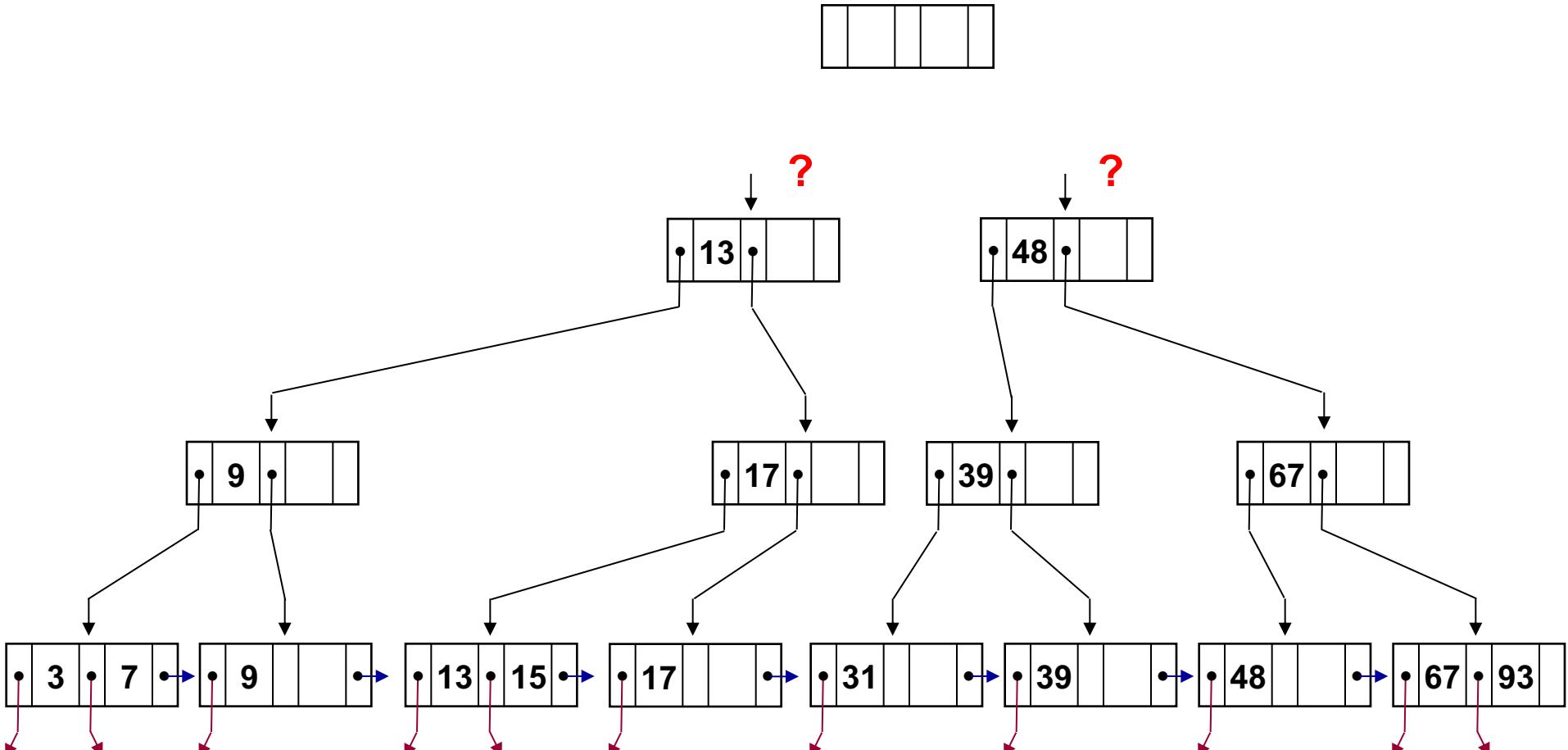
dodati novi čvor i podijeliti zapise  
13, 17, 31, 39 među čvorovima

## dodavanje zapisa s ključem 15



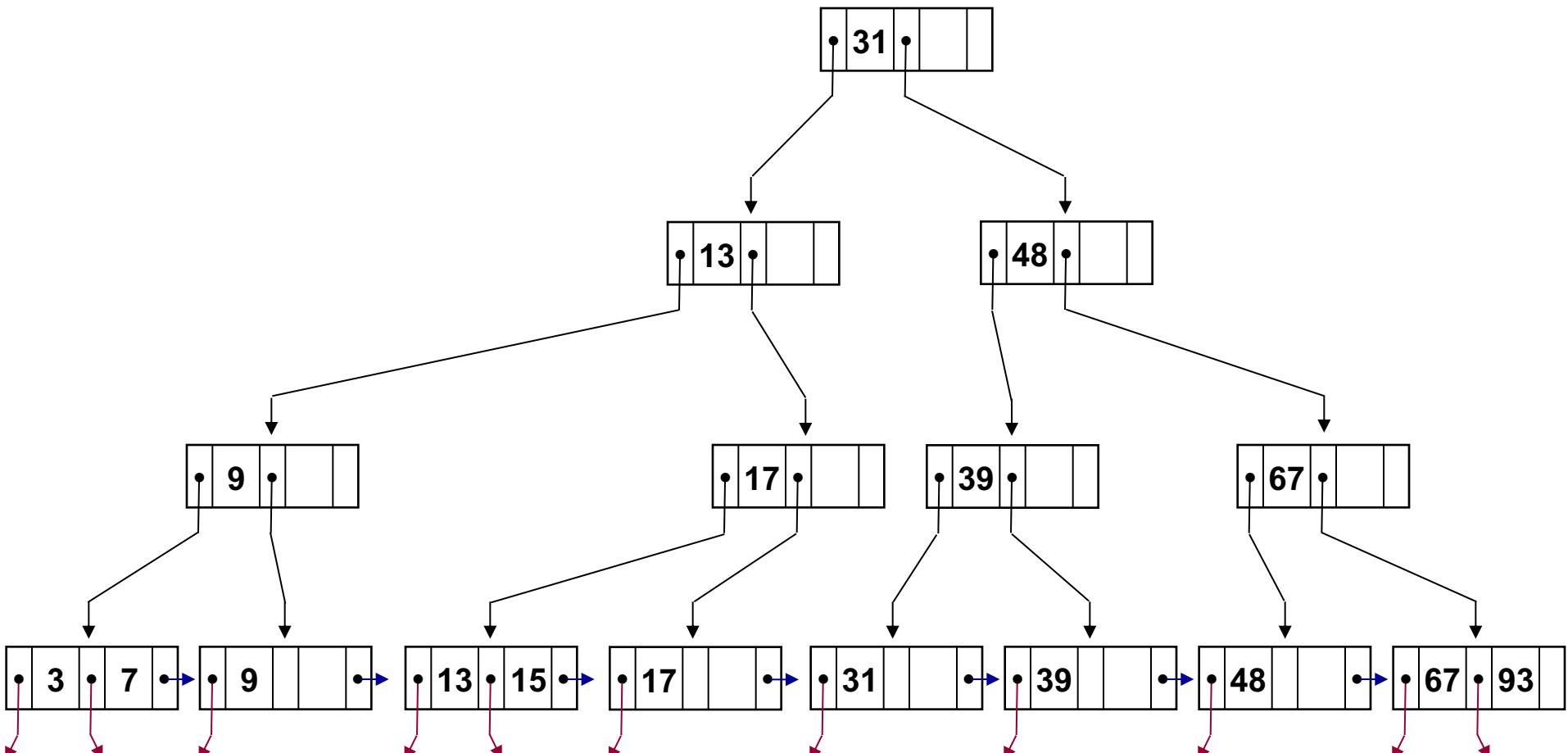
dodati novi čvor i podijeliti zapise  
3, 13, 31, 48 među čvorovima

## dodavanje zapisa s ključem 15



dodati novi korijen i upisati zapise 3, 31

## dodavanje zapisa s ključem 15

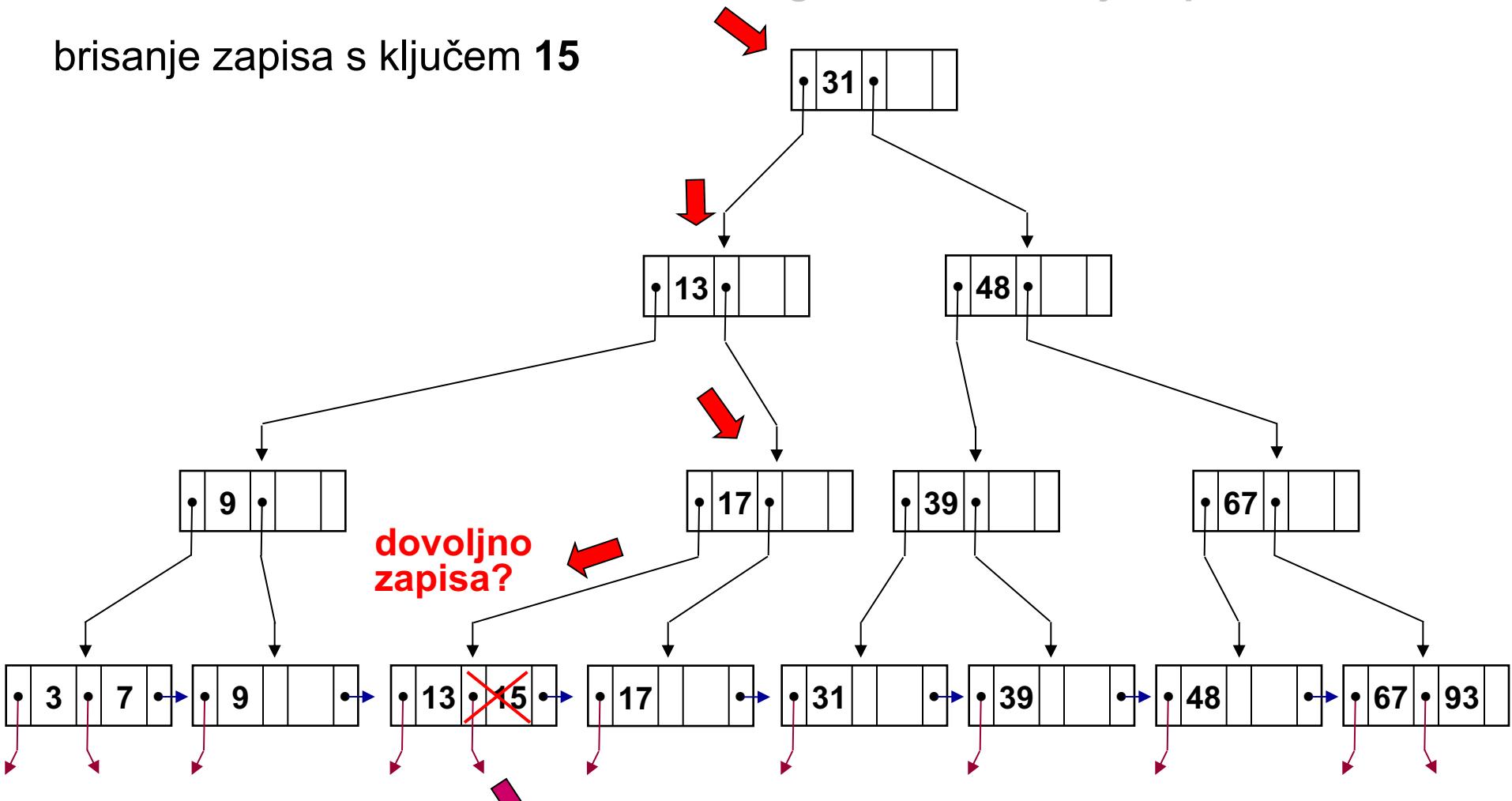


rezultat je balansirano stablo veće dubine

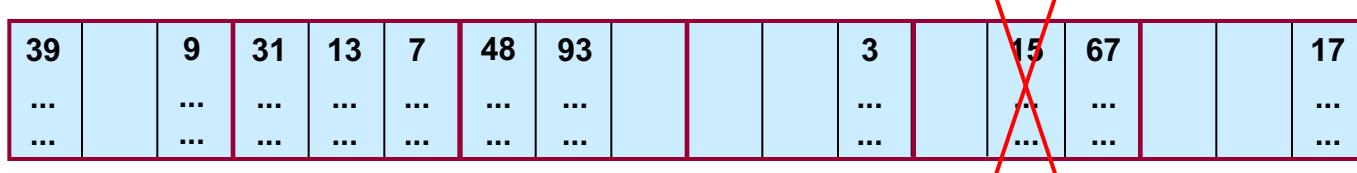
## Algoritam za brisanje zapisa iz B<sup>+</sup>-stabla

- obavlja se algoritam za pronalaženje lista L u kojem se nalazi ključ k
- zapis se briše iz bloka s podacima, a vrijednost ključa i kazaljka iz čvora L
- ako čvor L sadrži dovoljan broj zapisa
  - ukoliko je potrebno (onda kada se iz lista obriše prvi zapis, osim u krajnje lijevom listu), mijenja se vrijednost ključa u nekom od predaka
- ako čvor L nakon brisanja nema dovoljan broj zapisa, traži se čvor-brat  $L_X$  koji se nalazi neposredno s lijeva ili s desna čvoru L i ima više od dovoljnog broja zapisa. Ako se takav ne pronađe, traži se bilo koji čvor-brat  $L_X$  koji se nalazi neposredno s lijeva ili s desna čvoru L
  - ako odabrani brat  $L_X$  ima više od dovoljnog broja zapisa, tada se zapisi podijele između čvorova L i  $L_X$ , uz zadržavanje poretku zapisa. U pretcima čvorova L i  $L_X$  obavljaju se potrebne izmjene vrijednosti ključeva
  - inače (odabrani brat  $L_X$  ima upravo dovoljan broj zapisa), čvorovi L i  $L_X$  se spajaju u jedan čvor. U nadređenom čvoru briše se zapis za L i eventualno mijenja vrijednost ključa u nekom od predaka. Brisanje zapisa u nadređenom čvoru svodi se na rekurzivno izvođenje procedure za brisanje. Ako se putem prema korijenu dođe u situaciju da treba spojiti jedina dva čvora-djeteta korijena, tada se oni spajaju, postaju novi korijen stabla, a stari se korijen briše

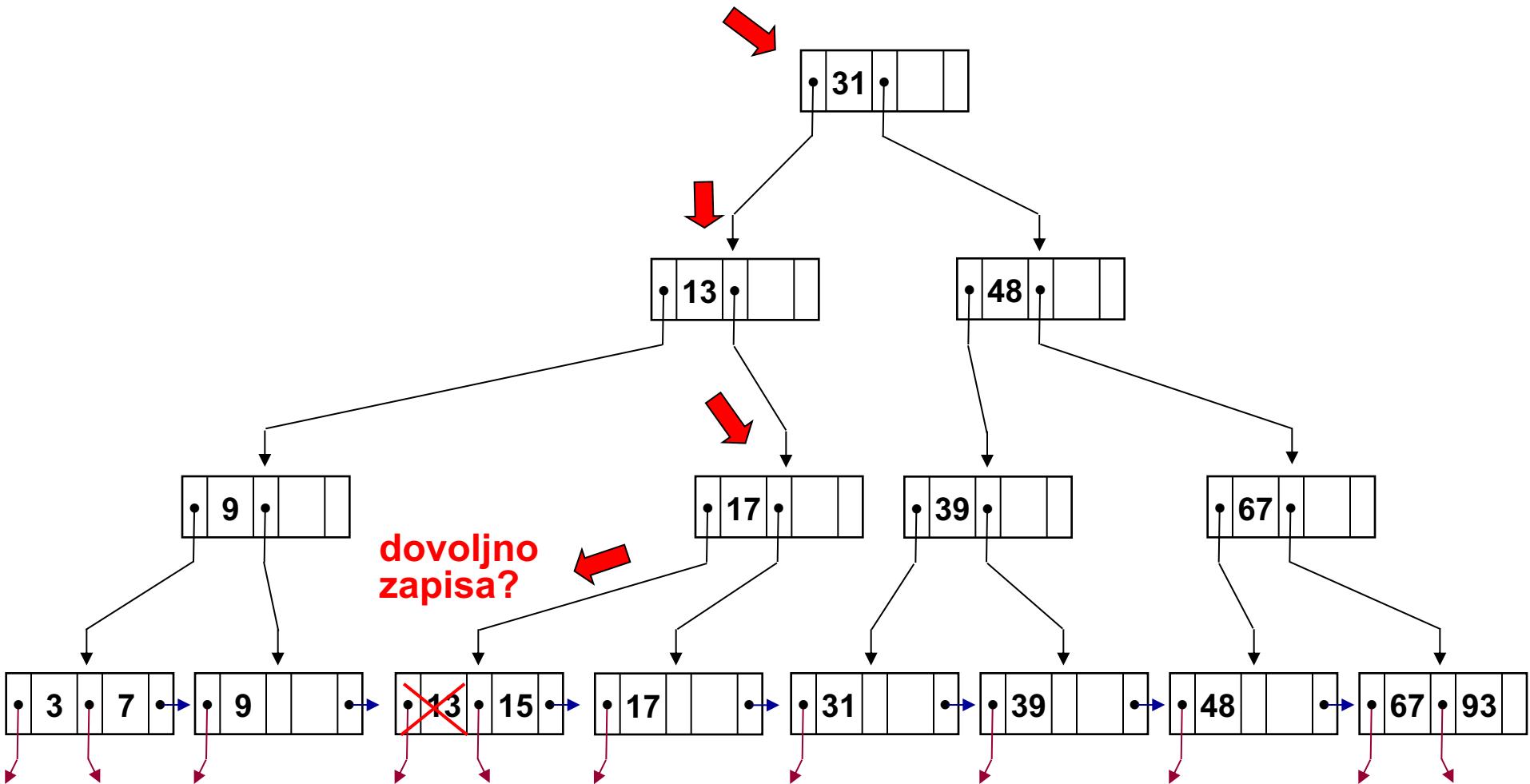
brisanje zapisa s ključem 15



brisanje zapisa iz bloka s podacima



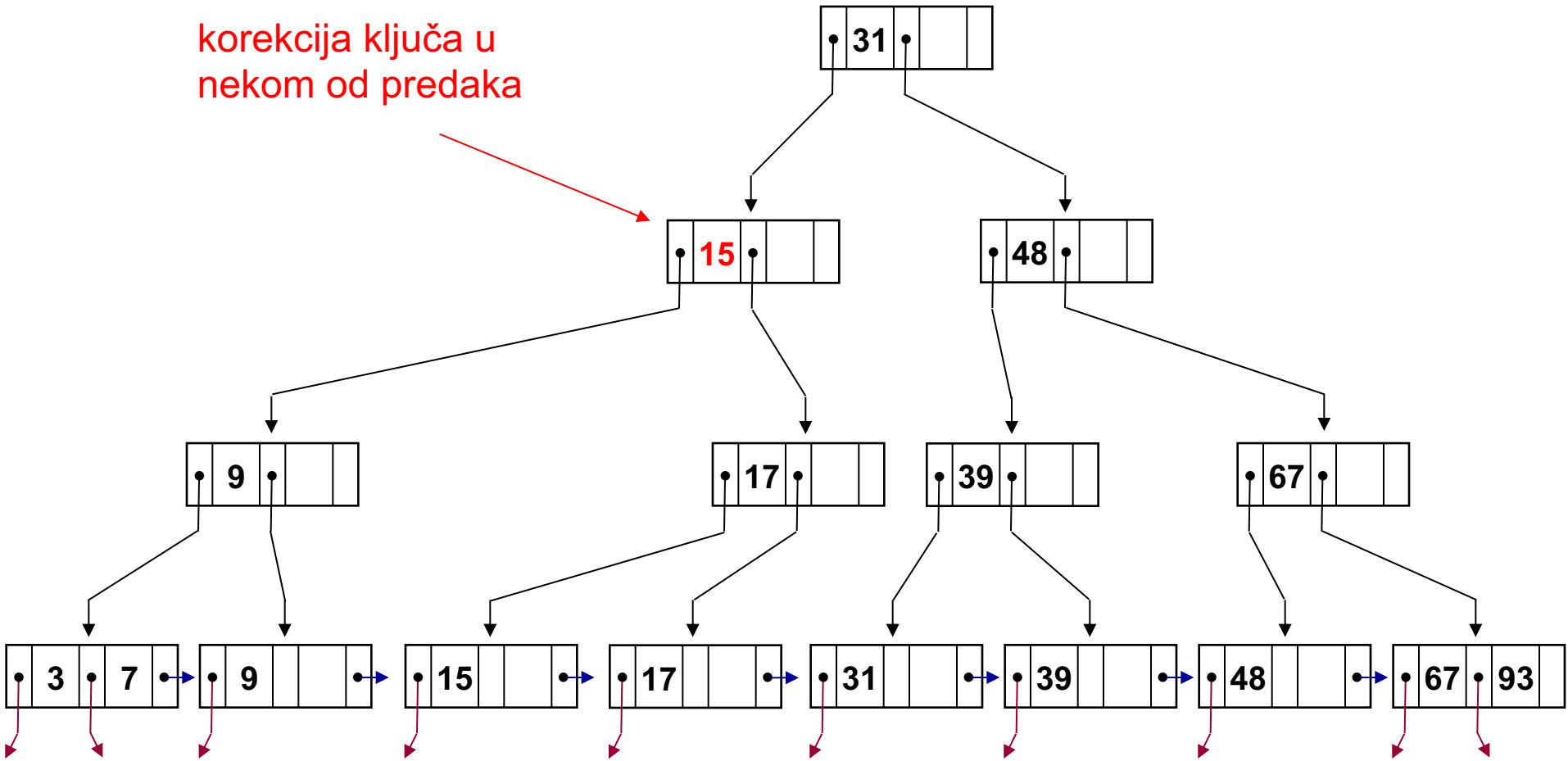
## brisanje zapisa s ključem 13



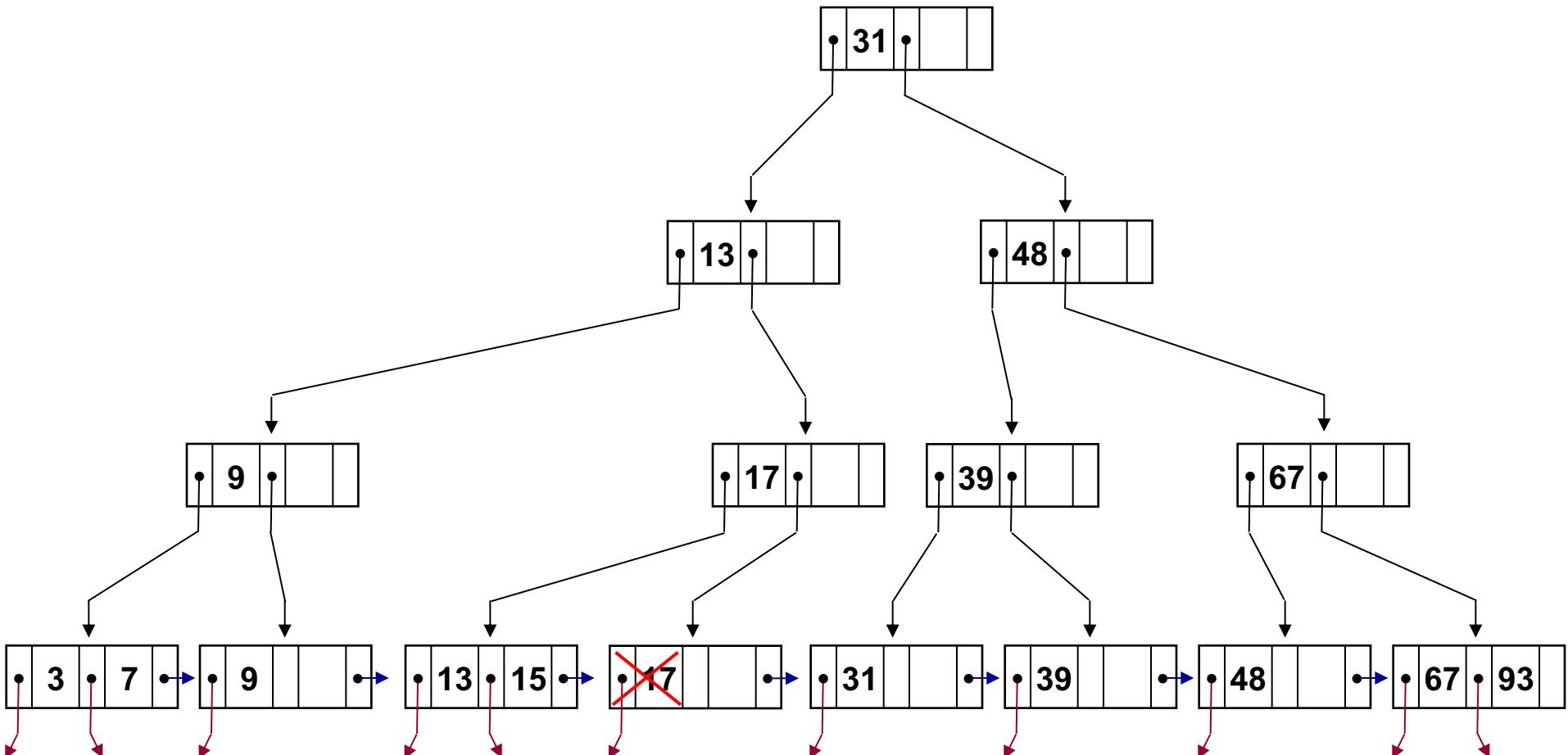
obaviti korekciju ključa u nekom od predaka jer je obrisan  
prvi zapis lista

## brisanje zapisa s ključem 13

korekcija ključa u  
nekom od predaka



## brisanje zapisa s ključem 17

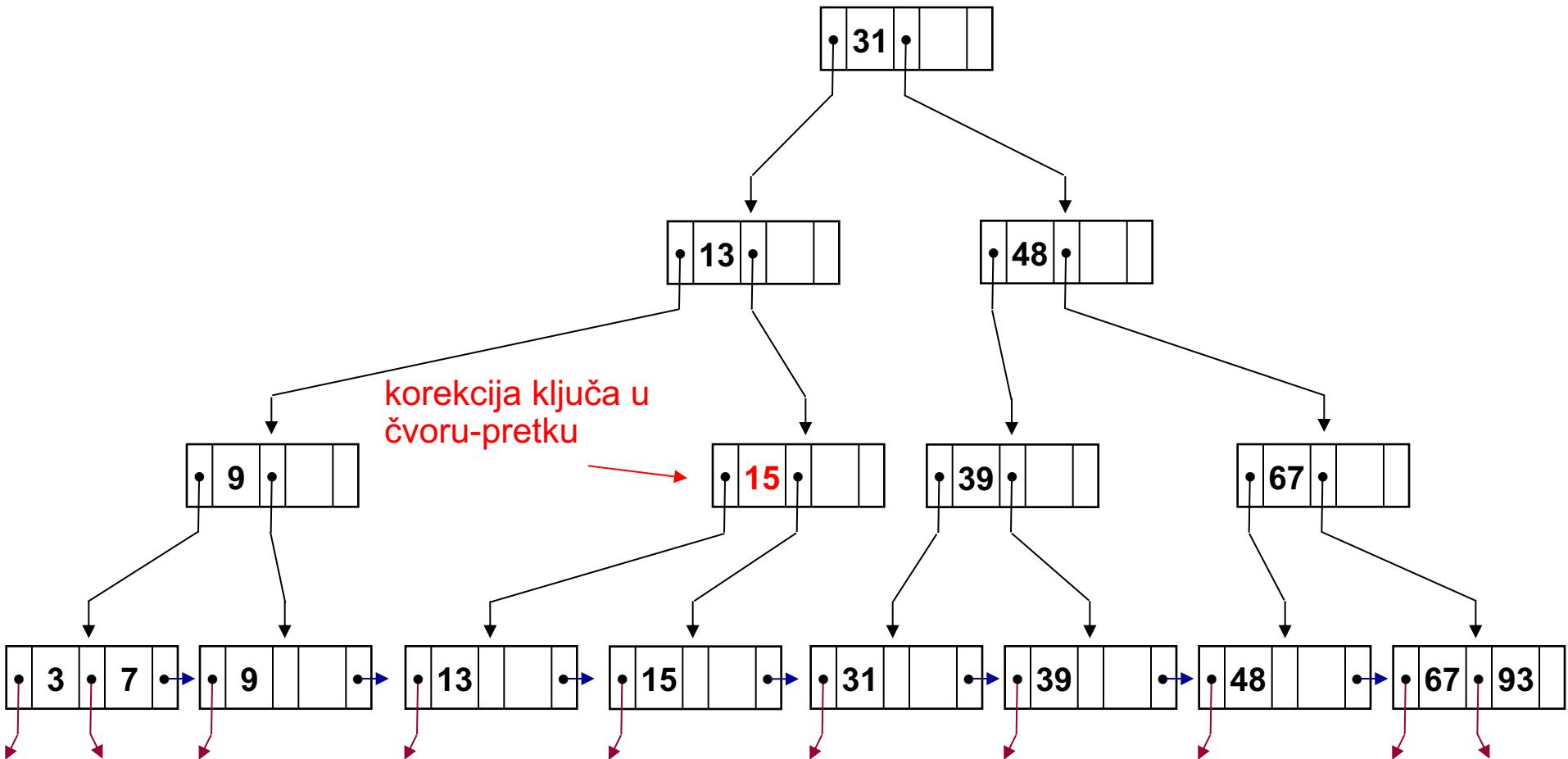


ima li brat  
L<sub>x</sub> više  
nego treba?

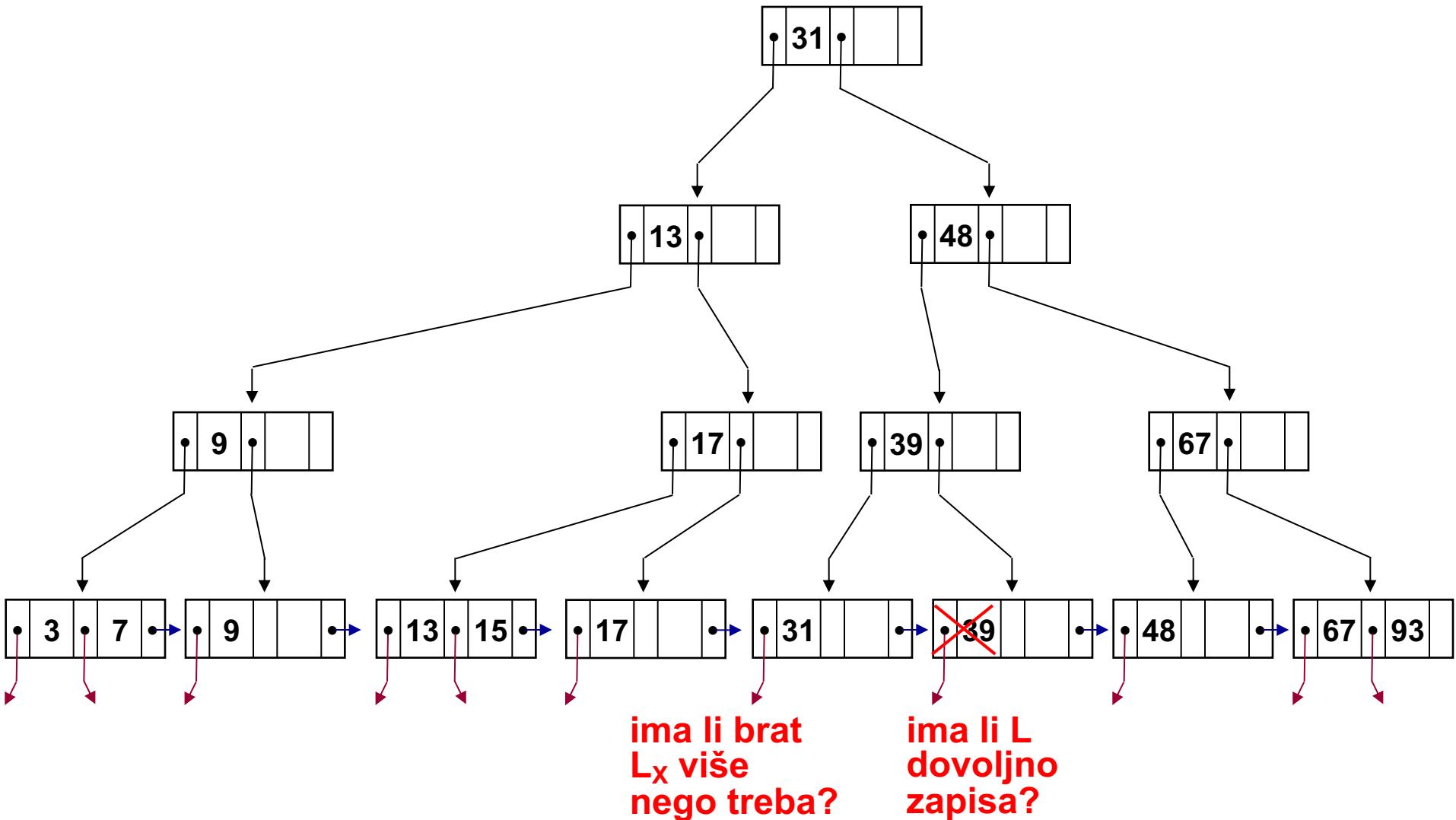
ima li L  
dovoljno  
zapis?

podijeliti zapise s bratom L<sub>x</sub> i obaviti potrebne  
korekcije ključeva u pretcima

## brisanje zapisa s ključem 17

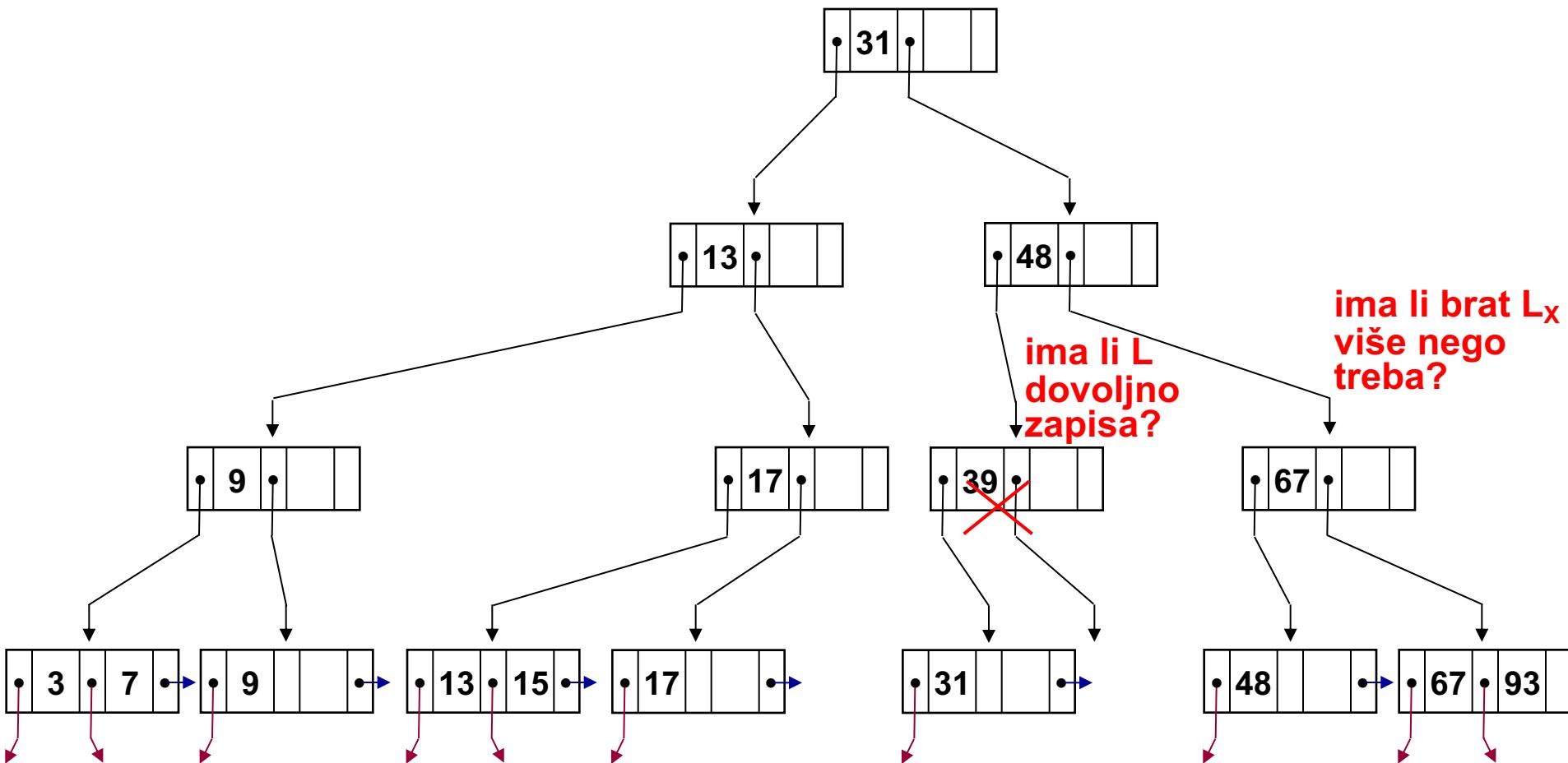


## brisanje zapisa s ključem 39



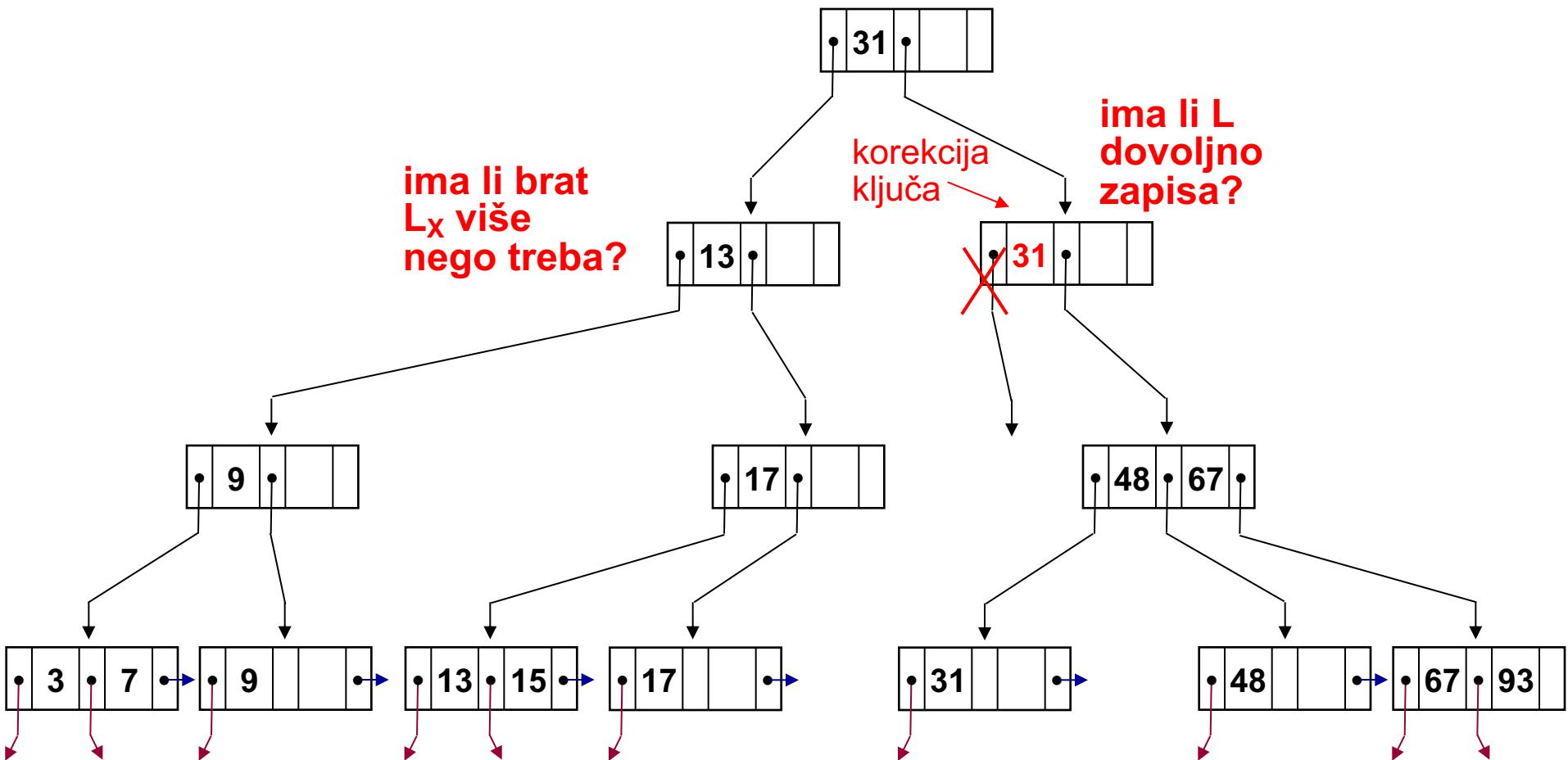
spojiti  $L$  i  $L_x$ . Obrisati kazaljku na  $L$  iz nadređenog čvora i  
(eventualno) promijeniti vrijednost ključa u nekom od predaka.

## brisanje zapisa s ključem 39



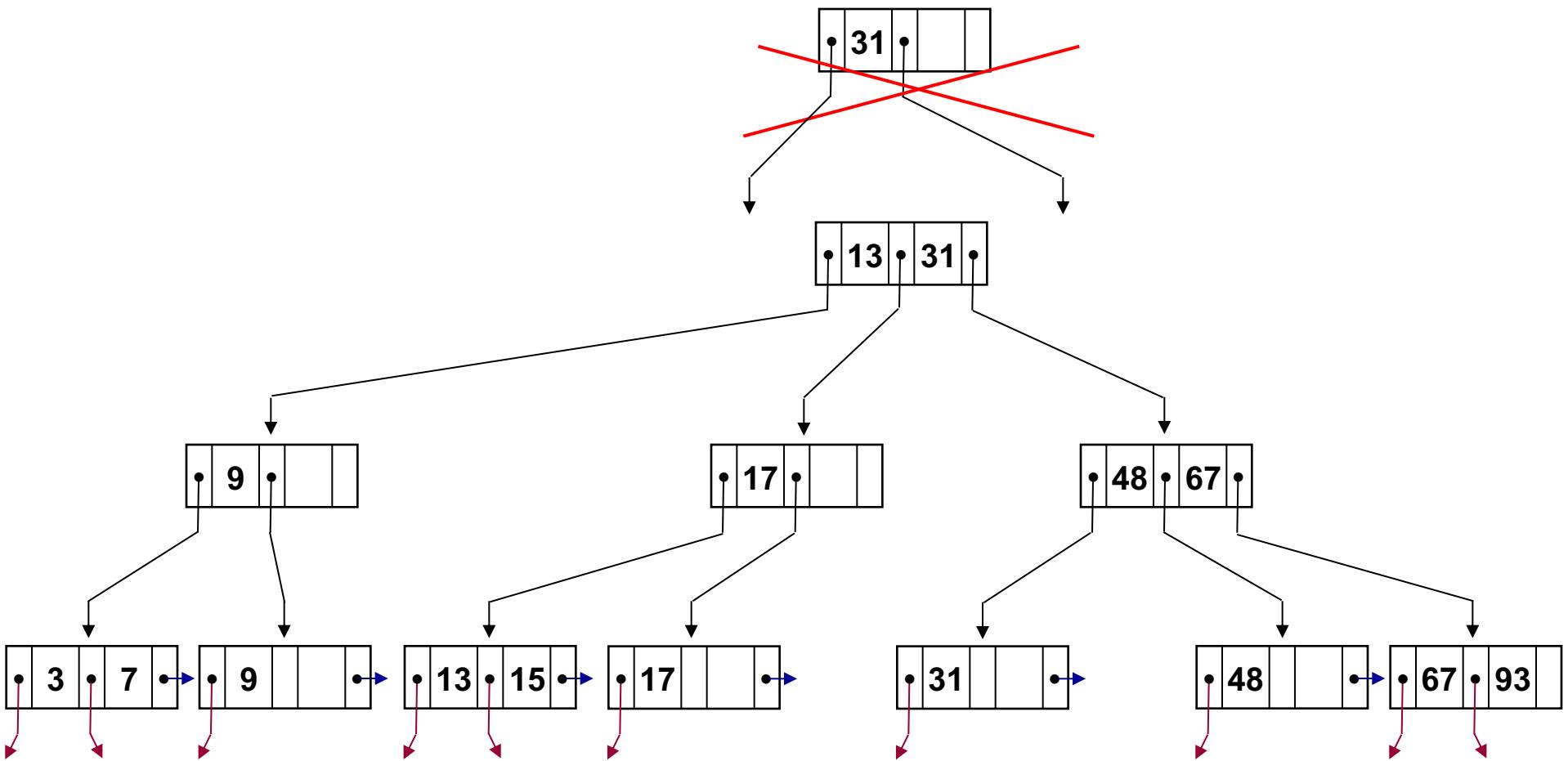
spojiti L i L<sub>x</sub>. Obrisati kazaljku na L iz nadređenog čvora i  
(eventualno) promijeniti vrijednost ključa u nekom od predaka.

## brisanje zapisa s ključem 39



spojiti L i L<sub>x</sub>. Spajanjem L i L<sub>x</sub> nastaje novi korijen.  
Obrisati stari korijen.

## brisanje zapisa s ključem 39



rezultat je balansirano stablo manje dubine

## 4. Učinkovitost operacija pretrage u B-stablu

- broj I/O operacija u stablu pri traženju zapisa ovisi o broju razina u stablu
- pretpostavka: stablo reda  $n$  sadrži kazaljke na  $m$  zapisa podataka
- stablo će imati najveći broj razina ako su čvorovi najmanje popunjeni
  - najmanja popunjenošć korijena je **2**
  - najmanja popunjenošć internog čvora:  $\lceil n / 2 \rceil$
  - najmanja popunjenošć lista:  $\lceil (n - 1) / 2 \rceil \approx \lceil n / 2 \rceil$ , za dovoljno veliki  $n$
- u korijenu (1. razina) ima 1 čvor i najmanje **2** kazaljke
- na 2. razini ima najmanje 2 čvora i zato najmanje  $2 \cdot \lceil n / 2 \rceil$  kazaljki
- u čvorovima 3. razine ima najmanje  $2 \cdot \lceil n / 2 \rceil \cdot \lceil n / 2 \rceil$  kazaljki
- u čvorovima  $i$ -te razine ima najmanje  $2 \cdot \lceil n / 2 \rceil^{i-1}$  kazaljki
- za broj zapisa u podatkovnim blokovima stabla koje ima **d** razina vrijedi:
  - $m \geq 2 \cdot \lceil n / 2 \rceil^{d-1}$
- iz toga slijedi
  - $d \leq \log_{\lceil n/2 \rceil} (m / 2) + 1$

$$d \leq \log_{\lceil n/2 \rceil} (m / 2) + 1$$

**Primjer:** za  $m = 1\ 000\ 000$ ,  $n = 70$ , ukupni broj razina (uključujući i razinu korijena) u najgorem slučaju je 4.

1.  točno 1 čvor, najmanje 2 kazaljke
2.  najmanje 2 čvora, najmanje 70 kazaljki
3.  najmanje 70 čvorova, najmanje 2 450 kazaljki
4.  najmanje 2 450 čvorova, najmanje 85 750 kazaljki
5.  najmanje 85 750 čvorova, najmanje 3 001 250 kazaljki

B<sup>+</sup>-stablo koje bi imalo ukupno 5 razina, moralo bi imati **najmanje** 3 001 250 kazaljki na zapise.

## Baze podataka

# Predavanja

## 9. Fizička organizacija podataka

# Travanj, 2020.



# UVOD - Fizička organizacija

---

- Pojam fizičke organizacije podataka odnosi se na:
  - strukture podataka primijenjene pri pohrani podataka u sekundarnoj memoriji
  - metode pristupa (*access methods*): postupci koji se primjenjuju pri obavljanju operacija nad podacima
- Fizička organizacija podataka ne utječe na rezultate operacija s podacima, ali ima vrlo veliki utjecaj na učinkovitost sustava za upravljanje bazama podataka
  - važna zadaća sustava za upravljanje bazama podataka: obavljati operacije nad velikim količinama podataka **na učinkovit način**
- SUBP skriva od korisnika detalje fizičke organizacije podataka jer za većinu korisnika sustava nisu značajni

# Pohrana baze podataka u sekundarnoj memoriji

---

- Glavna memorija (radni spremnik, *main memory*)
  - velike brzina pristupa podacima (10-100 ns), relativno skupa, kapacitet  $\sim$  GB
  - neprikladna za pohranu baze podataka jer:
    - ima kapacitet nedovoljan za pohranu baze podataka (previsoka cijena za pohranu velikih količina podataka)
    - nepostojana (*volatile*) memorija: sadržaj memorije se gubi pri gubitku napajanja ili pri pogrešci sustava

# Pohrana baze podataka u sekundarnoj memoriji

---

- karakteristike medija za pohranu podataka uvjetuju da se većina današnjih baza podataka pohranjuje u **sekundarnoj memoriji** (uobičajeno: magnetski diskovi)
  - podaci se između sekundarne i primarne memorije prenose u blokovima (tipično 512 B, 1 kB, 2kB, 4kB)
- ⇒ **dominiraju troškovi U/I (ulazno/izlaznih) operacija:** vrijeme potrebno za obavljanje U/I operacije radi prijenosa bloka podataka između sekundarne i primarne memorije znatno je veće od vremena koje će biti utrošeno za obavljanje operacija nad podacima u primarnoj memoriji

# Važniji ciljevi fizičke organizacije

---

- minimizirati broj U/I operacija pri pohrani i dohvatu podataka,  
minimizirati utrošak prostora za pohranu
  - u koji fizički blok pohraniti logički zapis odnosno n-torku
  - koje je dodatne informacije potrebno pohraniti da bi se omogućio učinkovit pristup podacima
- omogućiti različite metode pristupa koje se koriste za pronalaženje fizičke pozicije zapisa (ili fizičke pozicije bloka u kojem se taj zapis nalazi) na temelju vrijednosti ključa pretrage
  - ključ pretrage (*search key*) ne mora nužno biti primarni ili alternativni ključ. Ključ pretrage može biti bilo koji atribut ili skup atributa relacije ("sekundarni ključ").
  - primjenjivost pojedinih metoda pristupa podacima ovisi o primijenjenim strukturama podataka

# Strukture podataka i metode pristupa podacima

---

- Primjena različitih struktura podataka omogućava različite metode pristupa podacima
- Ne postoji "najbolja" metoda fizičke organizacije, ali dvije se u današnjim sustavima za upravljanje bazama podataka koriste najčešće
  - **Neporedana (*heap*) datoteka**
  - Poredana datoteka (*sorted file*)
  - Raspršena datoteka (*hash file*)
  - Indeksno-slijedna organizacija (*index-sequential file*)
  - **B-stablo (*B-tree*)**

# Neporedana (*heap*) datoteka

- zapis se upisuje na bilo koje slobodno mjesto u datoteci
- pristup podacima (dohvat podatka sa zadanim vrijednošću ključa pretrage) moguć je isključivo linearnim pretraživanjem



|    |       |
|----|-------|
| 15 | Petra |
| 1  | Marko |
| 47 | Ivana |
| 2  | Janko |
| 5  | Ana   |
|    | ...   |

- koristi se za relacije s malim brojem n-torki ili u relacijama čiji se podaci uvijek obrađuju slijedno

# Neporedana (*heap*) datoteka

---

- Dohvat zapisa prema ključu pretrage
  - prema primarnom ili alternativnom ključu
    - u prosjeku je potrebno obaviti  $n/2$  U/I operacija (n predstavlja broj fizičkih blokova u kojima su pohranjeni logički zapisi odnosno n-torke)
    - još gore: u slučaju kada traženi zapis ne postoji, sustav će morati obaviti n U/I operacija
  - prema ostalim ključevima pretrage ili prema zadanim granicama intervala
    - potrebno je obaviti prijenos n fizičkih blokova

# B-Stabla

Literatura:

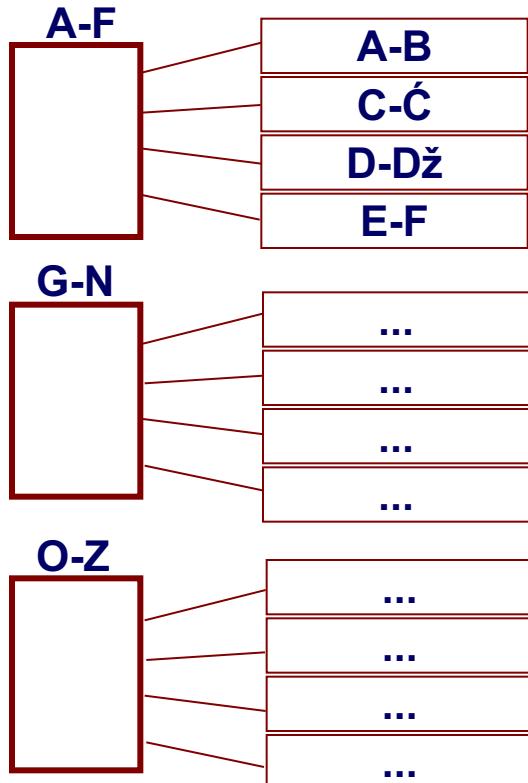
1. **Silberschatz, Korth, Sudarshan:** Database System Concepts
2. **Garcia-Molina, Ullman, Widom:** Database Systems -The Complete Book

Ovdje će biti opisana varijanta B-stabla koja se naziva **B<sup>+</sup>- stablo**. Opisi ostalih varijanti B-stabala (B\*-stablo, B-stablo, ...) mogu se pronaći u literaturi.

# B-stabla

- Ideja se temelji na izgradnji indeksnog kazala na više razina: slično kao kod kataloga u papirnatom obliku u knjižnicima (danас se rijetko koriste)

ormarići      ladice



kartice

|                                      |           |
|--------------------------------------|-----------|
| Albert, V.: Elektronička tehnika     | polica 11 |
| Alexander, C.K.: Electric circuits   | polica 79 |
| Arends, R.I.: Learning to teach      | polica 14 |
| Avčin, F.: Osnove metrologije        | polica 56 |
| Battin, R.H.: Astronautical guidance | polica 79 |
| ...                                  | ...       |

- Kartice su poredane abecedno prema prezimenu i imenu autora, ladice i ormarići su poredani prema abecedi. Knjige na policama **ne moraju** biti poredane po abecednom redoslijedu autora
- Prikazani katalog se može koristiti za efikasno pronalaženje knjiga prema autoru, ali sličan katalog se može izgraditi i za brzo pronalaženje knjiga prema naslovu ili prema nekim drugim pojmovima.

# Stablo kao struktura podataka

razina 0



korijen

razina 1



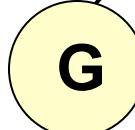
H



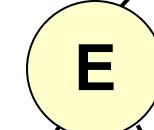
F

**interni čvor stabla:**  
svaki čvor koji nije list

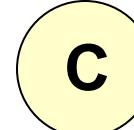
razina 2



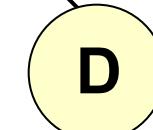
G



E



C



D

razina 3

listovi



J



B

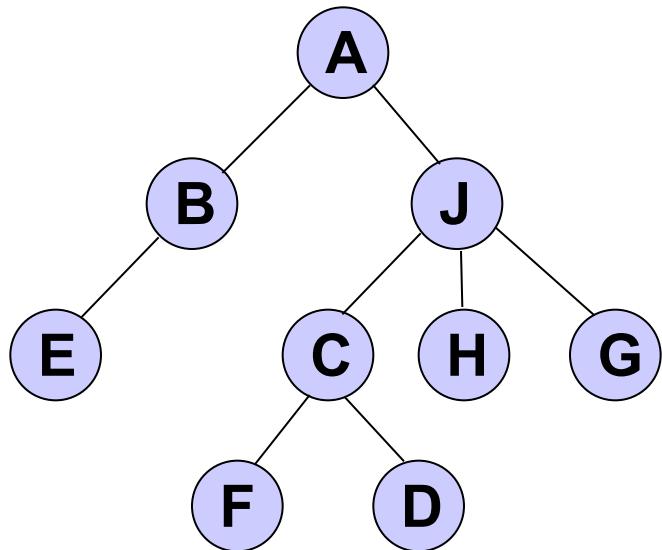
listovi

- **razina čvora (level):** duljina puta od korijena do čvora
- **dubina stabla (depth):** najveća duljina puta od korijena do lista
- **red stabla (order):** najveći broj djece koje čvor može imati

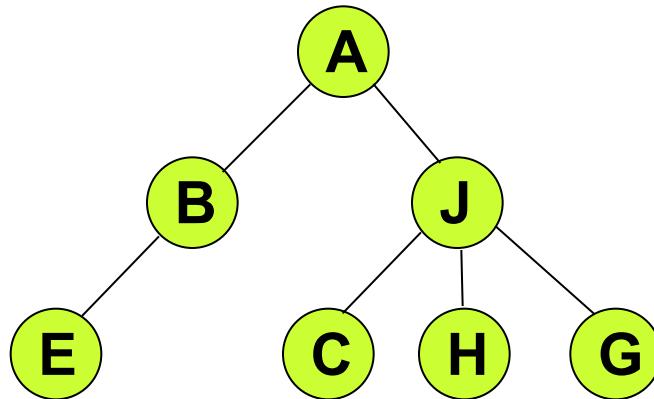
# Stablo kao struktura podataka

Stablo je **balansirano** (*balanced*) ukoliko je duljina puta od korijena do lista jednaka za svaki list u stablu

stablo nije balansirano

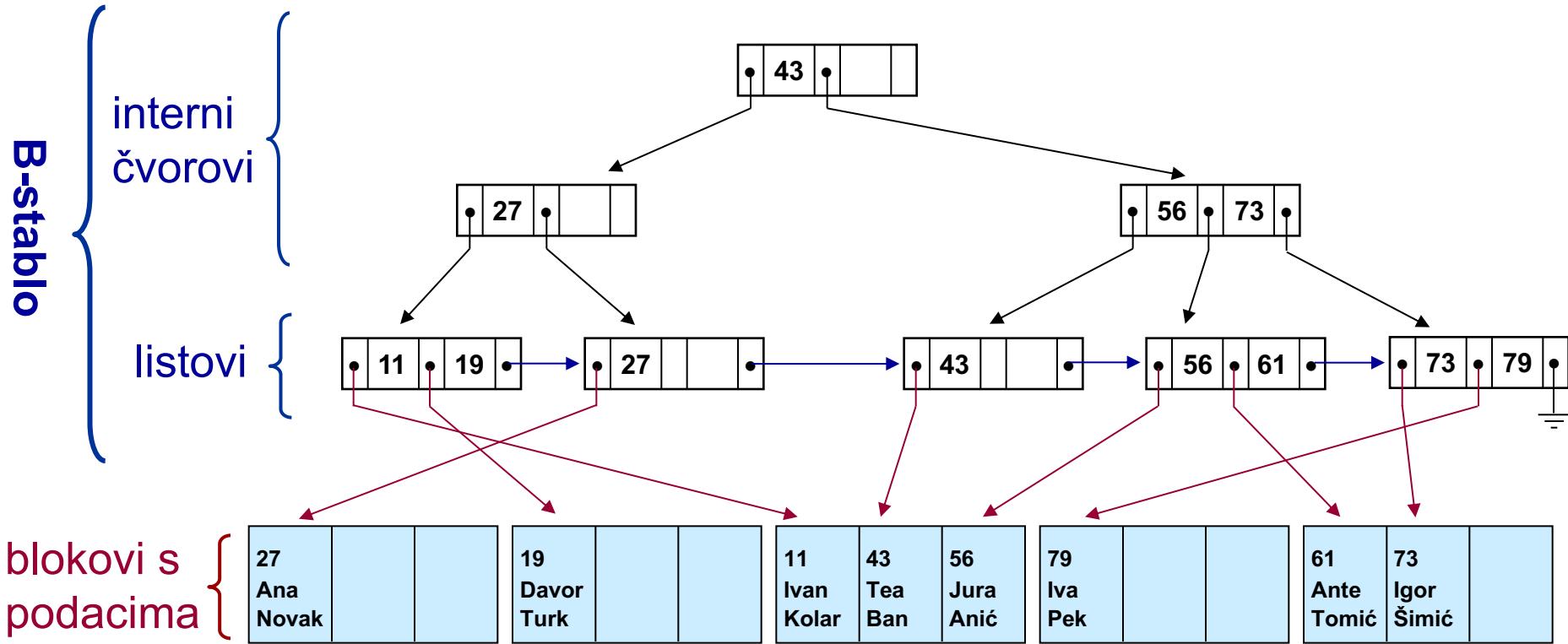


stablo je balansirano



Oznaka **B** u B-stablo znači "**balansirano**"!

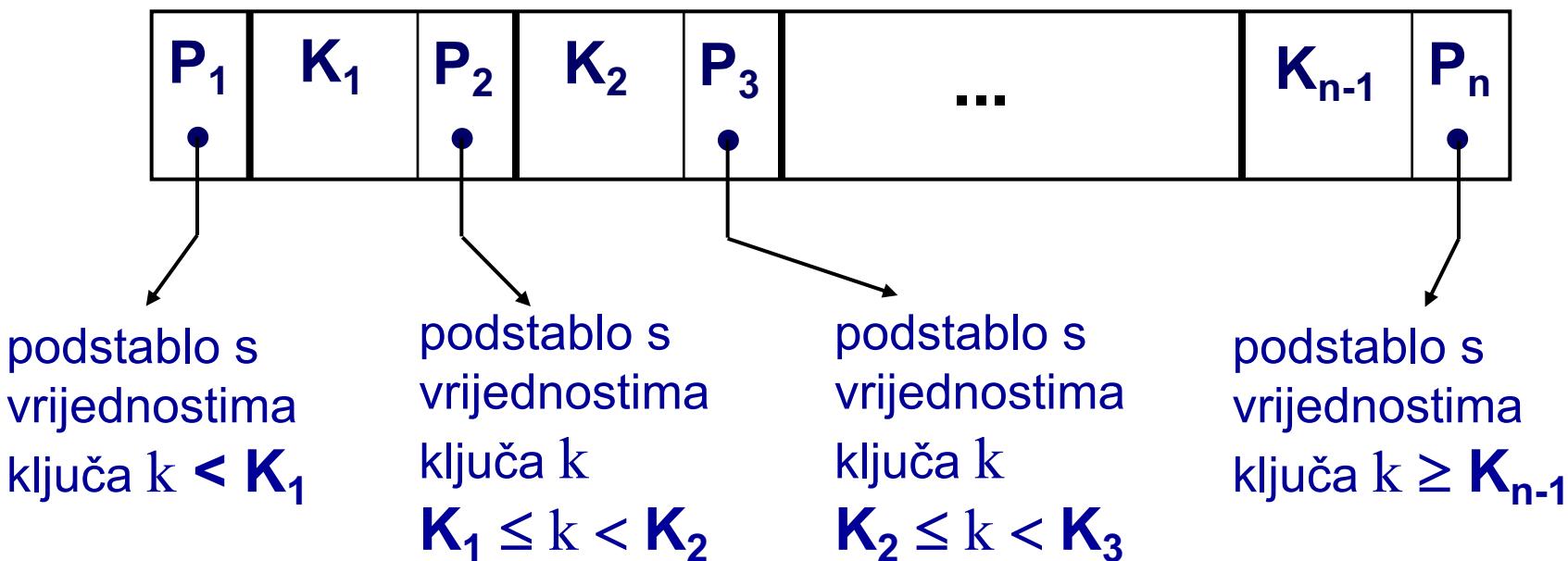
# Struktura B<sup>+</sup>-stabla



- Shema: mbr, ime, prezime; B<sup>+</sup>-stablo je izgrađeno za atribut mbr
- Moguće metode pristupa podacima (za bilo koji ključ pretrage):
  - linearnim pretraživanjem (kao kod neporedane datoteke)
  - ako je ključ pretrage mbr, može se koristiti B-stablo

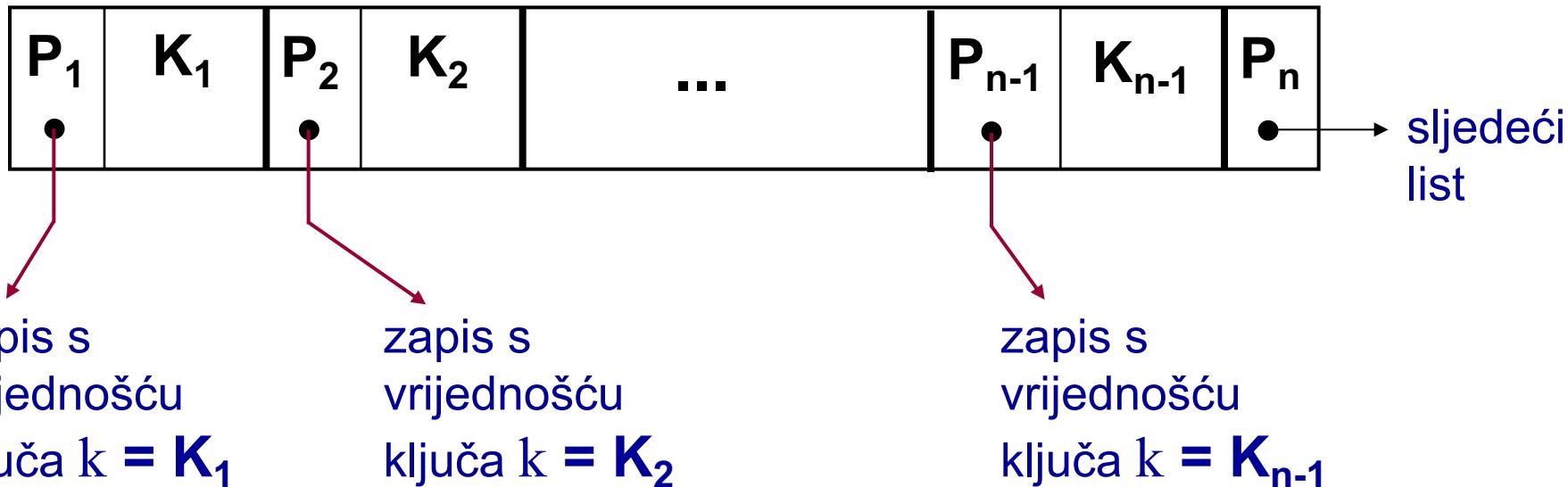
# Struktura internog čvora B<sup>+</sup>-stabla

- U B<sup>+</sup>-stablu reda  $n$ , **interni čvor** sadrži:
  - najviše  $n$  kazaljki
  - najmanje  $\lceil n/2 \rceil$  kazaljki  $\rightarrow \lceil a \rceil$  je najmanji cijeli broj  $\geq a$ 
    - ovo ograničenje ne vrijedi za korijen (najmanji broj kazaljki je 2)
  - uz  $p$  kazaljki u čvoru, broj pripadnih vrijednosti  $K_i$  u čvoru je  $p-1$ 
    - $K_i$  je vrijednost ključa



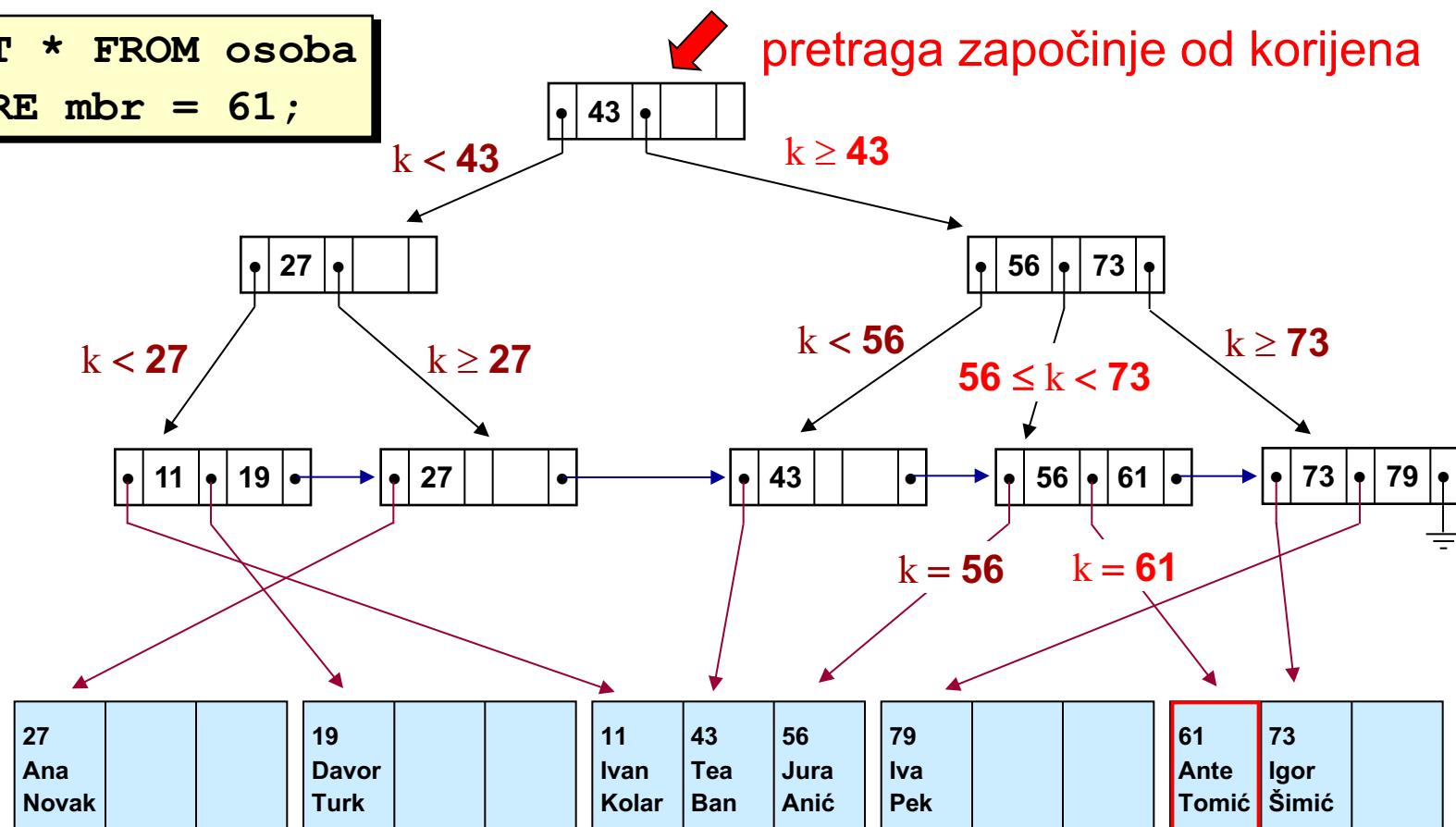
# Struktura lista B<sup>+</sup>-stabla

- U B<sup>+</sup>-stablu reda  $n$ , **list** sadrži:
  - najviše  $n-1$  vrijednosti  $K_i$  i pripadnih kazaljki na zapise
  - najmanje  $\lceil(n-1)/2\rceil$  vrijednosti  $K_i$  i pripadnih kazaljki na zapise
  - svi listovi sadrže kazaljku na sljedeći list
    - omogućava upite tipa od-do (prema zadanim granicama intervala)



# Algoritam za pronalaženje zapisa putem B<sup>+</sup>-stabla

```
SELECT * FROM osoba  
WHERE mbr = 61;
```

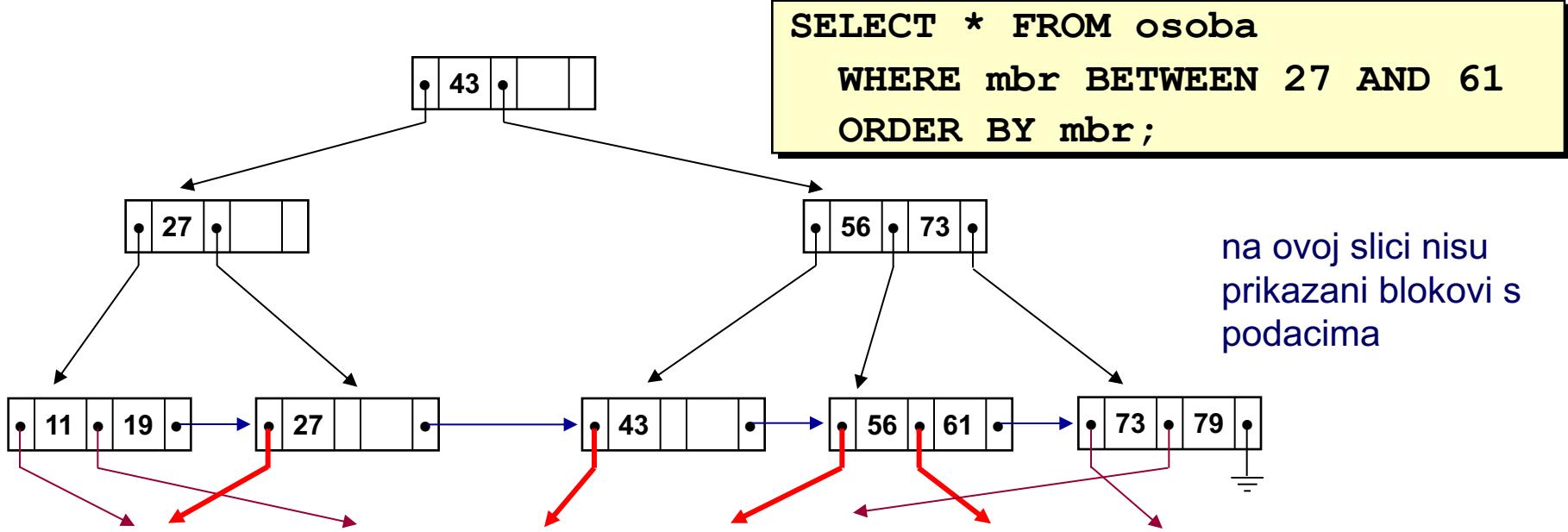


- slijediti odgovarajuću kazaljku do sljedeće razine
- postupak se ponavlja dok se ne dođe do lista u kojem će se naći kazaljka na zapis u bloku s podacima

# Algoritam za pronalaženje zapisa putem B<sup>+</sup>-stabla

- algoritam za traženje zapisa s ključem vrijednosti  $k$  je rekurzivan
  - cilj je u svakom koraku rekurzije (pretraga i-te razine) pronaći čvor na nižoj,  $(i+1)$ -voj razini, koji će voditi prema listu u kojem se nalazi ključ čija je vrijednost  $k$
- traženje zapisa započinje od korijena (0-te razine)
- u čvoru i-te razine potrebno je pronaći najveću vrijednost ključa koja je manja ili jednaka traženoj vrijednosti  $k$ 
  - za prvu kazaljku internog čvora nije navedena vrijednost ključa, pa ona "pokriva" sve vrijednosti ključeva manje od prve vrijednosti ključa ( $K_1$ ) navedene u čvoru
- nakon pronalaženja odgovarajuće vrijednosti ključa, slijedi se pripadna kazaljka i time se obavlja pozicioniranje na  $(i+1)$ -vu razinu
- postupak se ponavlja rekurzivno sve dok se ne dođe do lista. U njemu se mora nalaziti, ukoliko postoji, ključ čija je vrijednost  $k$ , te pripadna kazaljka prema traženom zapisu ( $n$ -torki)

# Dohvat podataka iz intervala, sortiranje



- u listu pronaći kazaljku na zapis s ključem **27**
- redom dohvaćati kazaljke i pripadne zapise dok se ne dođe do kazaljke na zapis s ključem **61**
  - taj postupak omogućavaju kazaljke među listovima
- dobiveni su svi traženi zapisi, pri tome su poredani prema mbr
- Ako se obavlja **SELECT \* ... ORDER BY mbr DESC**
  - pronađene zapise jednostavno ispisati obrnutim redoslijedom

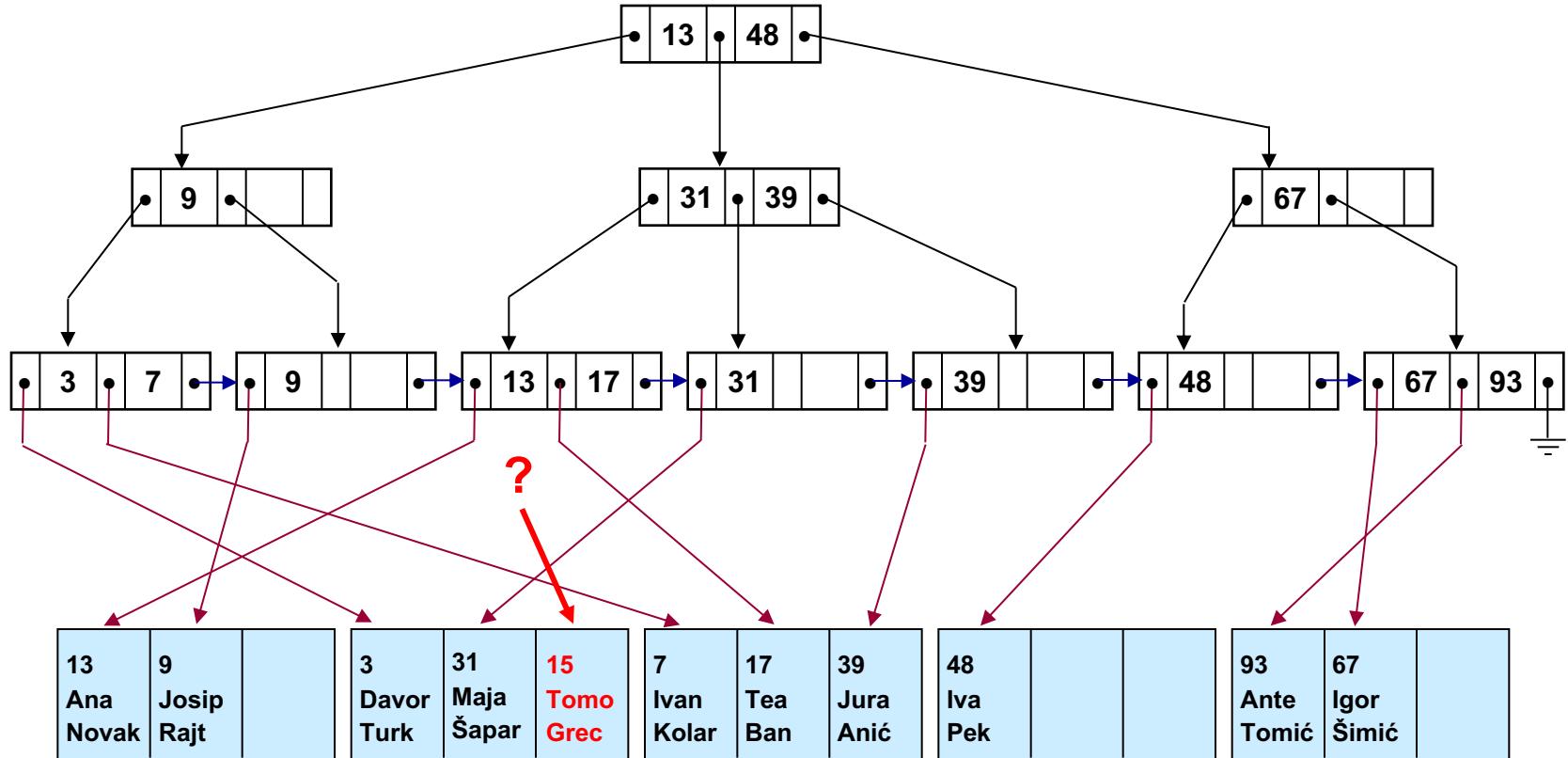
# Dodavanje i brisanje zapisa

- nakon dodavanja ili brisanja zapisa u bloku s podacima, mijenja se i sadržaj  $B^+$ - stabla
  - koriste se algoritmi za dodavanje i brisanje zapisa u  $B^+$  - stablu
  - algoritmi osiguravaju ispravnu popunjenošću internih čvorova i listova  $B^+$ - stabla
  - pri tome se može dogoditi da stablo promijeni dubinu
- operacija izmjene
  - ukoliko se ne mijenjaju vrijednosti atributa za koje je izgrađeno  $B^+$ -stablo, u  $B^+$ -stablu nisu potrebne izmjene
  - ukoliko se mijenjaju vrijednosti atributa za koje je izgrađeno  $B^+$ -stablo, u  $B^+$ -stablu se obavlja algoritam za brisanje zapisa i algoritam za dodavanje zapisa
- **Važno dobro svojstvo algoritama B-stabla:** dubina stabla se automatski prilagođava broju zapisa - čvorovi stabla (osim korijena) su uvijek barem 50% popunjeni

# Primjer dodavanja zapisa

- dodavanje zapisa u  $B^+$ -stablu:

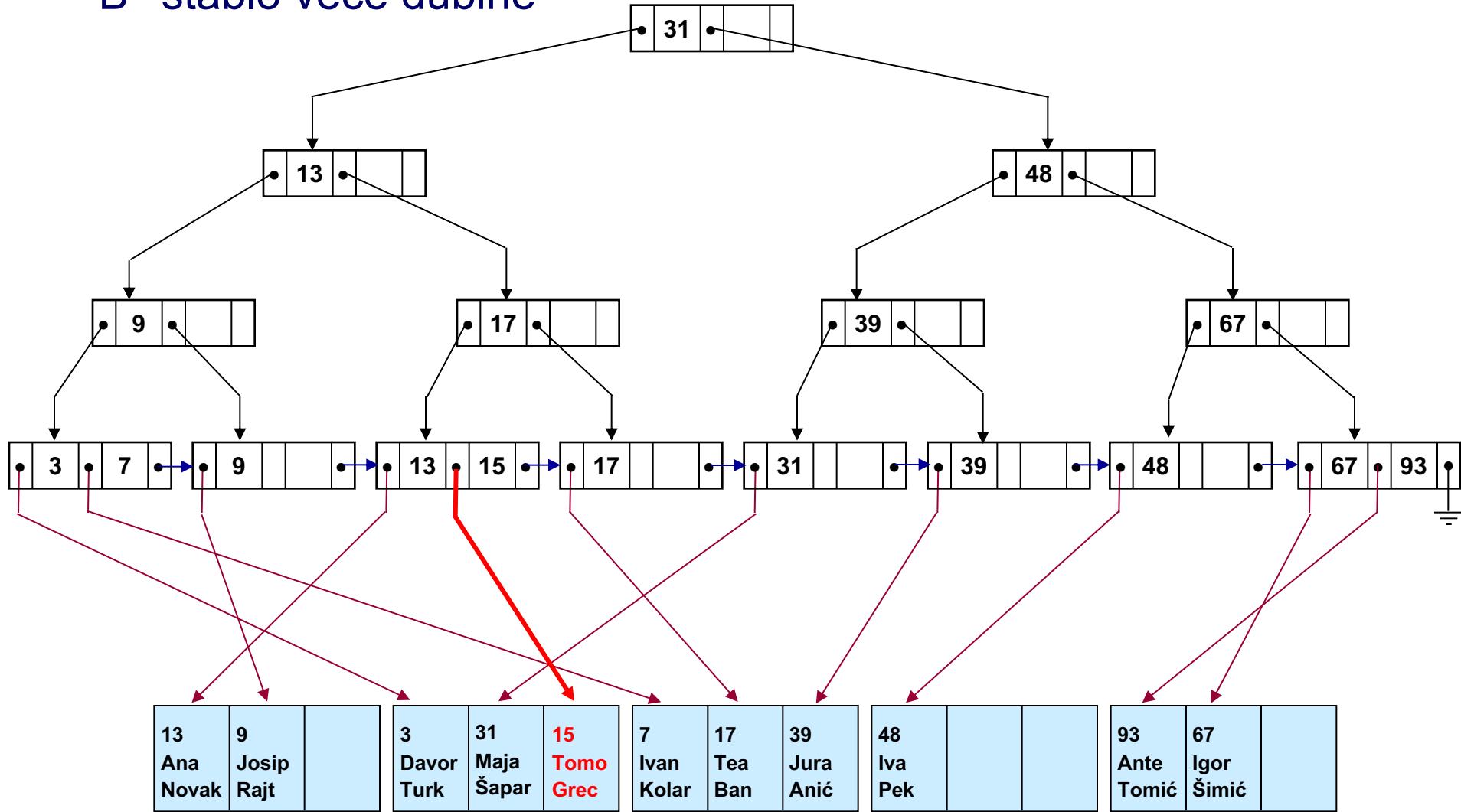
```
INSERT INTO osoba  
VALUES (15, 'Tomo', 'Grec');
```



- rezultat dodavanja zapisa u  $B^+$ -stablu je na sljedećoj slici:

# Rezultat dodavanja zapisa

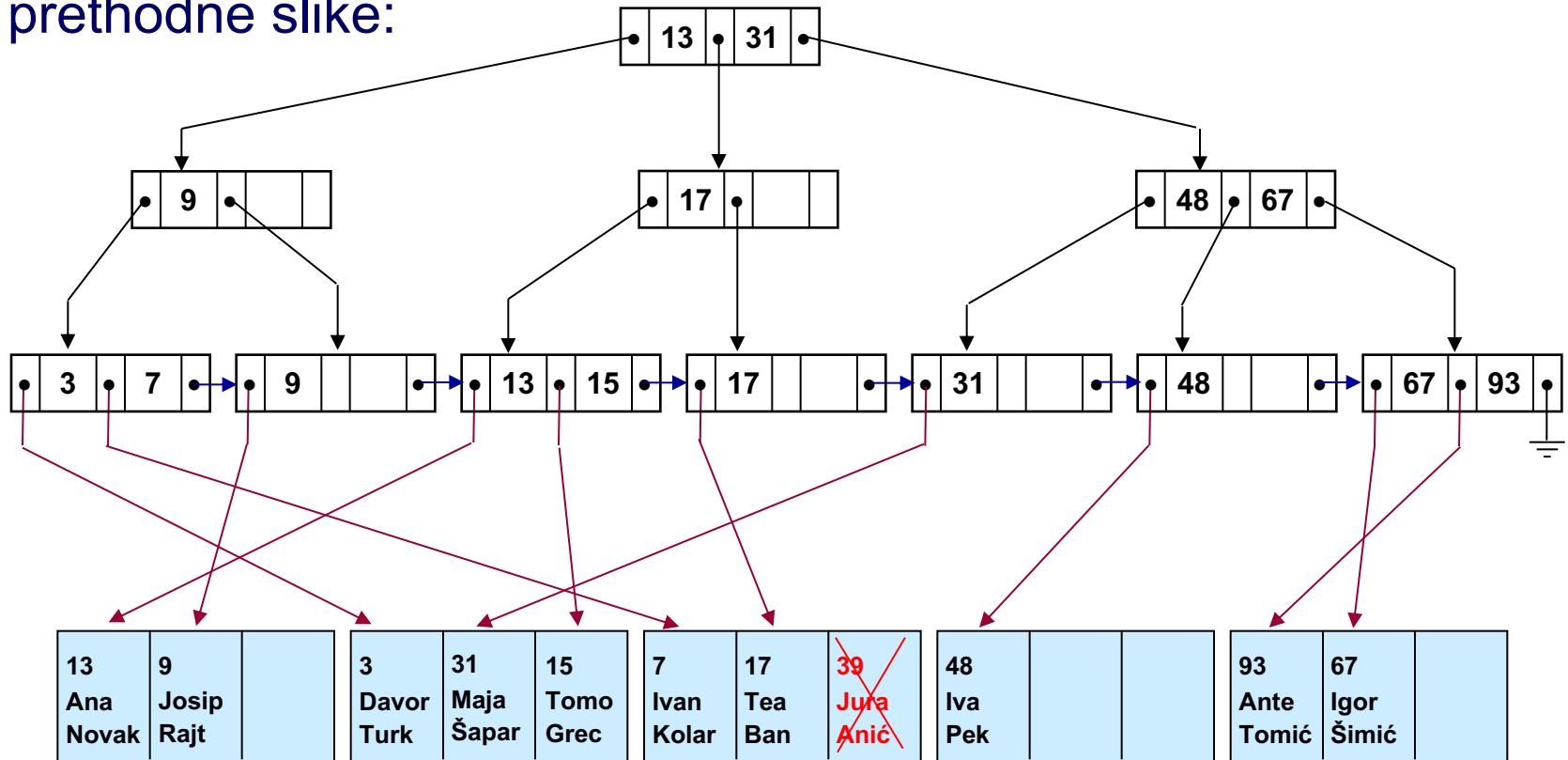
- B<sup>+</sup>-stablo veće dubine



# Primjer brisanja zapisa

- obavljanjem operacije nad  $B^+$ -stablom s prethodne slike:

```
DELETE FROM osoba WHERE mbr = 39;
```



- $B^+$ -stablo manje dubine

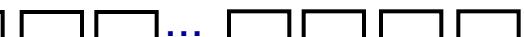
# Učinkovitost operacije pretrage u B<sup>+</sup>-stablu

---

- pretpostavka: stablo reda **n** sadrži kazaljke na **m** zapisa podataka
- **n** se odabire tako da se sadržaj čvora može smjestiti u jedan fizički blok
  - ⇒ za dohvat **jednog** čvora potrebna je **jedna** U/I operacija
- broj U/I operacija u stablu pri traženju zapisa ovisi o broju razina u stablu jer se pri dohvatu zapisa mora obaviti po jedna U/I operacija za svaki čvor B-stabla na putu od korijena do lista
- B-stablo ima najveći broj razina onda kada su čvorovi najmanje popunjeni
  - ⇒ moguće je odrediti koliko će U/I operacija biti potrebno obaviti u najlošijem slučaju

# Učinkovitost operacije pretrage u B<sup>+</sup>-stablu

- **Primjer:** za broj n-torki  $m = 1\ 000\ 000$ , za **red** stabla  $n = 70$ , ukupni broj razina (uključujući i razinu korijena) u najlošijem slučaju je 4:

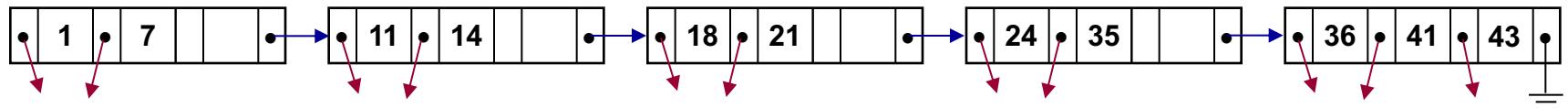
1.  točno 1 čvor, najmanje 2 kazaljke
2.  najmanje 2 čvora, najmanje 70 kazaljki
3.  najmanje 70 čvorova, najmanje 2 450 kazaljki
4.  najmanje 2 450 čvorova, najmanje 85 750 kazaljki
5.  najmanje 85 750 čvorova, najmanje 3 001 250 kazaljki

- B<sup>+</sup>-stablo koje bi imalo ukupno 5 razina, moralo bi imati **najmanje** 3 001 250 kazaljki na zapise. To znači da B-stablo reda 70 čije kazaljke u listovima pokazuju na 1 000 000 n-torki može imati najviše 4 razine.

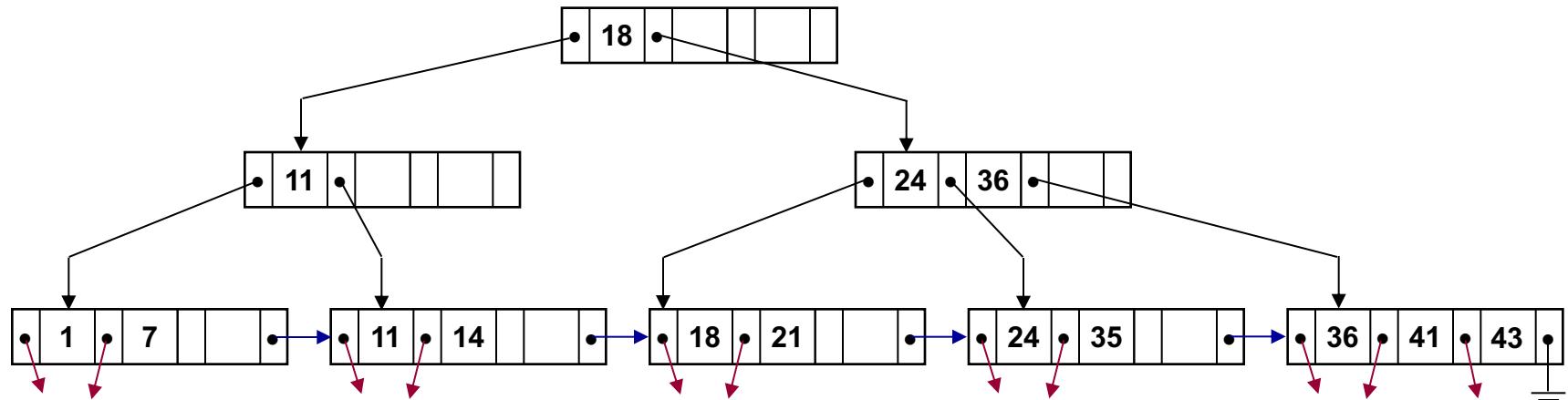
⇒ Za dohvat zapisa prema vrijednosti ključa potrebno je najviše 5 U/I operacija (4 U/I operacije za dohvat lista u kojem se nalazi kazaljka na zapis + 1 U/I operacija za dohvat bloka s podacima)

# Zadatak 1.

- Relacija *stud* (*mbr*, *prez*, *ime*) sadrži n-torce sa sljedećim vrijednostima atributa *mbr*: 1, 7, 11, 14, 18, 21, 24, 35, 36, 41, 43. Nacrtati B<sup>+</sup>-stablo reda 4 za atribut *mbr* tako da popunjeno stabla bude minimalna.
- min. broj kazaljki na zapise (n-torce) u jednom listu je  $\lceil (4 - 1) / 2 \rceil = 2$

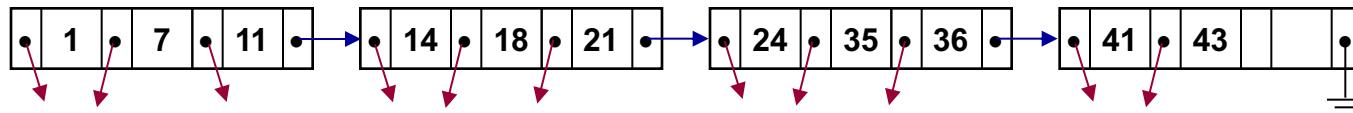


- min. broj kazaljki u jednom internom čvoru (osim korijena) je  $\lceil 4 / 2 \rceil = 2$

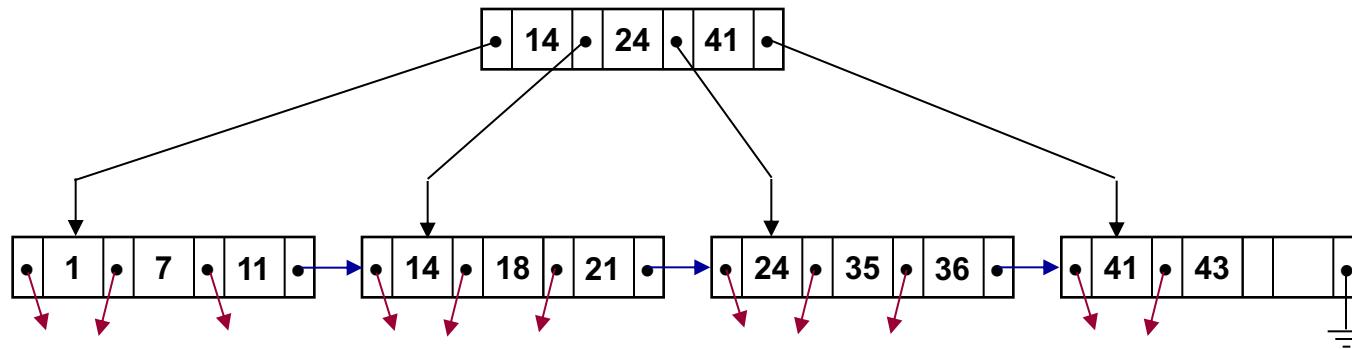


## Zadatak 2.

- Relacija *stud* (*mbr*, *prez*, *ime*) sadrži n-torke sa sljedećim vrijednostima atributa *mbr*: 1, 7, 11, 14, 18, 21, 24, 35, 36, 41, 43. Nacrtati B<sup>+</sup>-stablo reda 4 za atribut *mbr* tako da popunjenočvorova u stablu bude maksimalna.
- maksimalni broj kazaljki na zapise (n-torke) u jednom listu je  $4 - 1 = 3$



- maksimalni broj kazaljki u jednom internom čvoru je 4



## Zadatak 3.

---

- Koliko n-torki sadrži relacija ako je nad njom izgrađeno  $B^+$ -stablo reda 101, s ukupno 5 razina, s minimalno dopuštenom popunjenošću **svih** čvorova
  - min. broj kazaljki u jednom listu je  $\lceil (101 - 1) / 2 \rceil = 50$
  - min. broj kazaljki u jednom internom čvoru (osim korijena) je  $\lceil 101 / 2 \rceil = 51$
  - min. broj kazaljki u korijenu je 2
  - relacija sadrži  $2 \cdot 51 \cdot 51 \cdot 51 \cdot 50 \approx 1.33 \cdot 10^7$  n-torki
- **ZAKLJUČAK:** ako je B-stablo reda 101, do svake n-torce u relaciji koja sadrži  $\approx 1.33 \cdot 10^7$  n-torki može se pristupiti, u najlošijem slučaju, korištenjem tek 6 U/I operacija (5 U/I za dohvati lista, 1 U/I za dohvati fizičkog bloka u kojem se nalazi n-torka)

## Zadatak 4.

---

- Koliko n-torki sadrži relacija ako je nad njom izgrađeno  $B^+$ -stablo reda 101, s **ukupno 5 razina**, s maksimalno popunjениm **svim** čvorovima
- max. broj kazaljki u jednom listu je  $101 - 1 = 100$
- max. broj kazaljki u internom čvoru je **101**
- relacija sadrži  $101 \cdot 101 \cdot 101 \cdot 101 \cdot 100 \approx 1.04 \cdot 10^{10}$  n-torki
- **ZAKLJUČAK:** ako je B-stablo reda 101, u najboljem slučaju, korištenjem tek 6 U/I operacija može se dohvatiti blok s n-torkom koja se nalazi u relaciji koja sadrži čak  $\approx 1.04 \cdot 10^{10}$  n-torki

# SQL: Indeksi

## 7. CREATE INDEX Statement

```
CREATE [UNIQUE] INDEX index ON table (column [ASC|DESC])
```

- Obavljanjem naredbe za kreiranje indeksa nad relacijom, nad blokovima s podacima relacije formira se struktura B-stabla

1

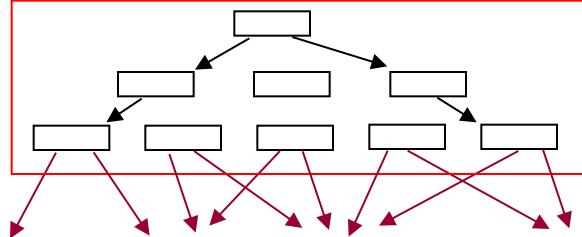
```
CREATE TABLE osoba (
    mbr INTEGER,
    ime NCHAR(20),
    prez NCHAR(20));

INSERT INTO ...;
```

2

```
CREATE INDEX osoba_pres
ON osoba (pres);
```

B-stablo za osoba.pres



The table contains eight data rows:

|   |            |    |         |    |           |    |         |  |  |    |            |    |            |  |
|---|------------|----|---------|----|-----------|----|---------|--|--|----|------------|----|------------|--|
| 7 | Ivan Kolar | 17 | Tea Ban | 39 | Jura Anić | 48 | Iva Pek |  |  | 93 | Ante Tomić | 67 | Igor Šimić |  |
|---|------------|----|---------|----|-----------|----|---------|--|--|----|------------|----|------------|--|

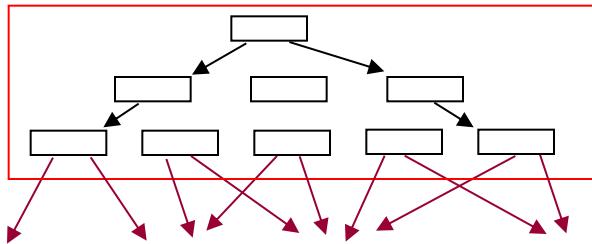
# SQL: Indeksi

- Kreiranjem indeksa uz navođenje rezervirane riječi UNIQUE osigurava se jedinstvenost vrijednosti navedenog atributa

```
CREATE UNIQUE INDEX osoba_mbr  
ON osoba (mbr);
```



B-stablo za osoba.mbr



|                    |                  |                    |                  |  |                     |                     |  |
|--------------------|------------------|--------------------|------------------|--|---------------------|---------------------|--|
| 7<br>Ivan<br>Kolar | 17<br>Tea<br>Ban | 39<br>Jura<br>Anić | 48<br>Iva<br>Pek |  | 93<br>Ante<br>Tomić | 67<br>Igor<br>Šimić |  |
|--------------------|------------------|--------------------|------------------|--|---------------------|---------------------|--|

- ukoliko se indeks pokuša kreirati nad relacijom u kojoj već postoji duplikat vrijednosti atributa mbr, sustav će odbiti kreirati indeks i dojaviti pogrešku
- pokuša li se nakon kreiranja ovog indeksa unijeti n-torka s vrijednošću atributa mbr koja već postoji u nekoj n-torci, sustav će odbiti operaciju i dojaviti pogrešku

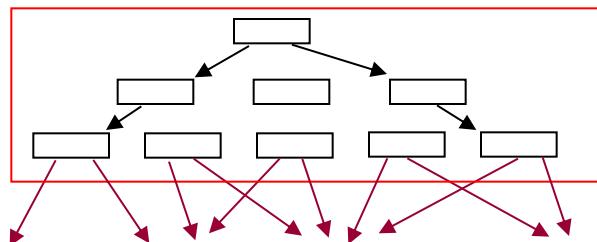
- Uništavanje indeksa - primjer:

```
DROP INDEX osoba_mbr;
```

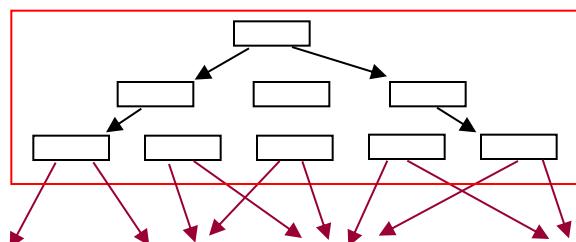
# Više indeksa nad istom relacijom

- nad istom relacijom može se izgraditi više indeksa

B-stablo za osoba.mbr



B-stablo za osoba.prez



|                     |                  |                    |                  |  |
|---------------------|------------------|--------------------|------------------|--|
| 11<br>Ivan<br>Kolar | 43<br>Tea<br>Ban | 56<br>Jura<br>Anić | 79<br>Iva<br>Pek |  |
|---------------------|------------------|--------------------|------------------|--|

|                     |                     |  |  |  |
|---------------------|---------------------|--|--|--|
| 61<br>Ante<br>Tomić | 73<br>Igor<br>Šimić |  |  |  |
|---------------------|---------------------|--|--|--|

- B-stabla (indeksi) prikazani u primjeru omogućuju efikasno obavljanje upita s uvjetima ( $=$ ,  $>$ ,  $\geq$ ,  $<$ ,  $\leq$ , BETWEEN) i efikasno sortiranje (ASC, DESC)
  - za atribut mbr
  - za atribut prez
- prema uvjetima koji sadrže atribut ime, podacima se može pristupati jedino linearnom pretragom svih blokova

# Više indeksa nad istom relacijom

---

- Ako indeksi omogućuju efikasan pristup do n-torki, znači li to da bi indekse trebalo kreirati za svaki atribut u relaciji?

**NE !!!**

**Zašto?**

- indeksi zauzimaju prostor
- operacija unosa ili brisanja n-torke
  - uvijek rezultira promjenama (manjim ili većim) B-stabla
  - npr. ako je nad relacijom izgrađeno 10 različitih indeksa, unosom jedne n-torke u blokove s podacima morat će se unijeti zapisi i u 10 različitih indeksa
- operacija izmjene n-torke
  - izmjena vrijednosti atributa A jedne n-torke rezultirat će brisanjem i dodavanjem zapisa u svim B-stablima za indekse u kojima se koristi atribut A

# Za koje atribute treba kreirati indeks?

---

- za atribute koji se često koriste za postavljanje uvjeta selekcije (zašto?)
- za atribute prema kojima se obavlja spajanje relacija (zašto?)
  - primarni i alternativni ključevi relacije
  - strani ključevi
- za atribute prema kojima se često obavlja sortiranje ili grupiranje (zašto?)

# Za koje atribute treba kreirati indeks?

- **Primjer:**

```
CREATE TABLE stud (
    mbr  INTEGER
,  ime  NCHAR(20)
,  prez NCHAR(20));
```

- Često se postavljaju upiti oblika:

```
SELECT * FROM stud
WHERE mbr = 12345;
```

```
SELECT * FROM stud
WHERE prez > 'Kolar'
ORDER BY prez;
```

⇒ Kreirati indeks za mbr i indeks za prez

- Što se nakon kreiranja navedenih indeksa dešava pri obavljanju:

```
UPDATE stud SET prez = UPPER(prez)
WHERE ime = 'Ivan';
```

- n-torce se pronalaze linearnom pretragom (loše), a zbog izmjene vrijednosti atributa prez mora se izmijeniti sadržaj B-stabla za indeks nad atributom prez (loše)

```
UPDATE stud SET ime = UPPER(ime)
WHERE prez = 'Horvat';
```

- n-torce se pronalaze pomoću B-stabla (dobro), nad atributom ime nije izgrađen indeks, stoga ne postoji B-stablo čiji se sadržaj mora izmijeniti (dobro)

# Za koje atribute ne treba kreirati indeks?

---

- ako vrijednosti atributa imaju relativno mali broj različitih vrijednosti
  - npr. atribut spolOsobe s dozvoljenim vrijednostima M, Ž
- ako relaciji predstoji velik broj upisa, izmjena ili brisanja n-torki. Preporuča se u takvim slučajevima postojeće indekse izbrisati, te ih ponovo izgraditi tek nakon obavljenih promjena nad podacima
- ako relacija sadrži relativno mali broj n-torki (sve n-torke su pohranjene u nekoliko blokova). U takvim slučajevima B-stablo ne pridonosi efikasnosti pretrage
  - npr. relacija zupanija

# Složeni indeksi

```
CREATE TABLE stud (
    mbr INTEGER
, ime NCHAR(20)
, prez NCHAR(20));
```

```
CREATE INDEX stud_ime ON stud (ime);
CREATE INDEX stud_pres ON stud (pres);
```

- efikasno se obavljaju upiti oblika:

```
SELECT * FROM stud
WHERE prez = 'Kolar';
```

```
SELECT * FROM stud
WHERE ime = 'Ivan';
```

- upit oblika:

```
SELECT * FROM stud
WHERE prez = 'Kolar'
AND ime = 'Ivan';
```

- pomoću indeksa nad atributom prez dohvatić će se n-torce studenata čije je prezime 'Horvat', ali će se u dobivenom skupu n-torki linearnom pretragom morati pronaći one čije je ime 'Ivan' (ili indeksom po imenu, a onda linearno po prezimenu)

# Složeni indeksi

- Prethodni upit se efikasnije obavlja ako se umjesto posebnih indeksa za atribute ime i prezime, kreira složeni indeks:

```
CREATE INDEX stud_pres_ime ON stud (pres, ime);
```

- problem: ovaj indeks se koristi za upite oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar'  
AND ime = 'Ivan';
```

=

```
SELECT * FROM stud  
WHERE ime = 'Ivan'  
AND prez = 'Kolar';
```

- također i za upite oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar';
```

- ali se ne može koristiti za upite oblika:

```
SELECT * FROM stud  
WHERE ime = 'Ivan';
```

# Složeni indeksi

- Kako upotreba složenih indeksa utječe na sortiranje:

```
CREATE INDEX stud_pres_ime1 ON stud (pres, ime);
```

- indeks osoba\_pres\_ime1 se efikasno koristi za sortiranje oblika:

```
SELECT * FROM stud  
ORDER BY prez, ime;
```

```
SELECT * FROM stud  
ORDER BY prez DESC, ime DESC;
```

- ali ne i za:

```
SELECT * FROM stud  
ORDER BY prez DESC, ime;
```

- ako se kreira indeks:

```
CREATE INDEX stud_pres_ime2 ON stud (pres DESC, ime);
```

- indeks osoba\_pres\_ime2 se efikasno koristi za sortiranje oblika:

```
SELECT * FROM stud  
ORDER BY prez DESC, ime;
```

```
SELECT * FROM stud  
ORDER BY prez, ime DESC;
```

# Složeni indeksi

- Ako je nad relacijom r (ABCD) kreiran složeni indeks r\_abc1

```
CREATE INDEX r_abc1 ON r (A, B, C);
```

tada sljedeće indekse nije potrebno kreirati:

```
CREATE INDEX r_abc2 ON r (A DESC, B DESC, C DESC);
CREATE INDEX r_a1 ON r (A);
CREATE INDEX r_a2 ON r (A DESC);
CREATE INDEX r_ab1 ON r (A, B);
CREATE INDEX r_ab2 ON r (A DESC, B DESC);
```

- Indekse r\_abc2, r\_a1, r\_a2, r\_ab1 i r\_ab2 ne treba kreirati jer SUBP može koristiti indeks r\_abc1 u svim slučajevima u kojima bi se koristili indeksi r\_abc2, r\_a1, r\_a2, r\_ab1 i r\_ab2.

## Zadatak 5. zadana je relacija stud (mbr ime prez pbrStan)

---

- Za relaciju stud kreirati najmanji mogući broj indeksa koji će omogućiti efikasno obavljanje (pomoću B<sup>+</sup>-stabla) svih navedenih upita:

1. SELECT \* FROM stud ORDER BY ime DESC, prez;
  2. SELECT \* FROM stud ORDER BY ime DESC, prez DESC;
  3. SELECT \* FROM stud ORDER BY ime, prez, pbrStan;
  4. SELECT \* FROM stud WHERE prez = 'Novak' AND ime = 'Ivo';
  5. SELECT \* FROM stud WHERE pbrStan > 51000 ORDER BY pbrStan DESC;
- 

1. (ime DESC, prez)
2. (ime DESC, prez DESC)
3. (ime, prez, pbrStan) - ali sada više nije potreban indeks pod 2.
4. može se koristiti indeks pod 3.
5. (pbrStan)

**Konačno rješenje** - kreirati indekse za: (ime DESC, prez)

(ime, prez, pbrStan)

(pbrStan)

SQL naredbe za  
kreiranje indeksa  
napisati za vježbu!