Upravljanje informacijama i znanjem:

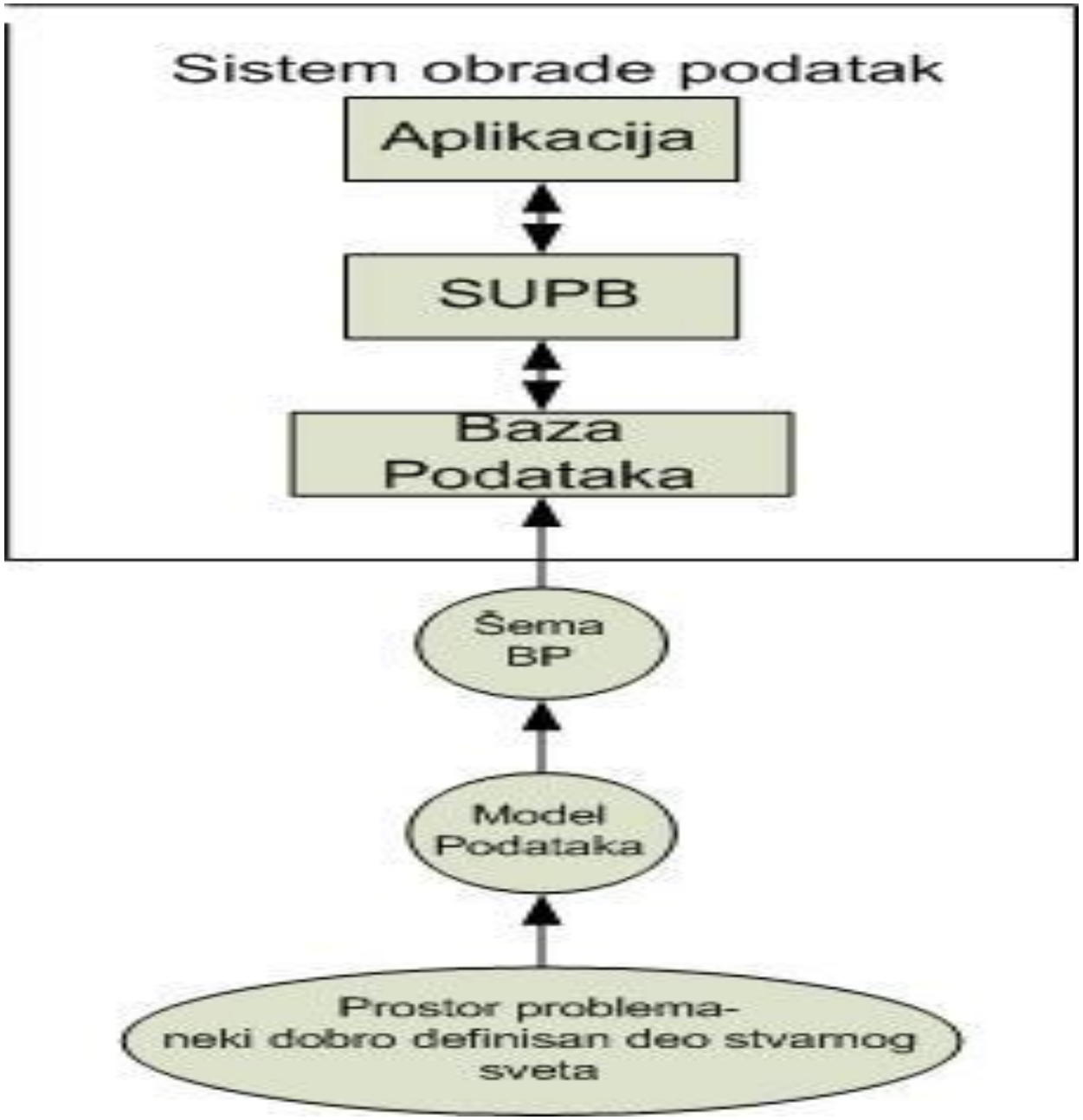
Lekcija 6: Relacioni model podataka (I)

zima 2019/2020

Prof. dr Branimir M. Trenkić

Pregled današnje lekcije

- Modeli baze podataka kod kojih je osnovna struktura podataka – **zapis** (*record*)
  - Hijerarhijski model
  - Mrežni model
  - **Relacioni model**
    - Relaciona algebra
    - Integritet podataka
    - SQL


Sistem obrade podatak
Aplikacija
SUPB
Baza Podataka
Šema BP
Model Podataka
Prostor problema- neki dobro definisan deo stvarnog sveta

Implementacioni modeli podatak

- Nekoliko **generacija DBMS**
- Fundamentalna razlika – razlika u **implementiranom modelu** podataka
- Što se reflektuje na:
  - ***efikasnost pristupa*** podacima i obrade podataka,
  - ***produktivnost*** korisnika,
  - ***funkcionalnost*** sistema i
  - ***podrška*** raznovrsnim ***aplikacijama***

Implementacioni modeli podatak

- Sistemi **datoteka**
- Modeli podataka baziranih na zapisima (***record-based***)
  - Koren vuku iz sistema datoteka
  - Mala fleksibilnost
  - Komplikovana proširenja
    - **Hijerarhijski** model
    - **Mrežni** model
  - **Relacioni model podataka**

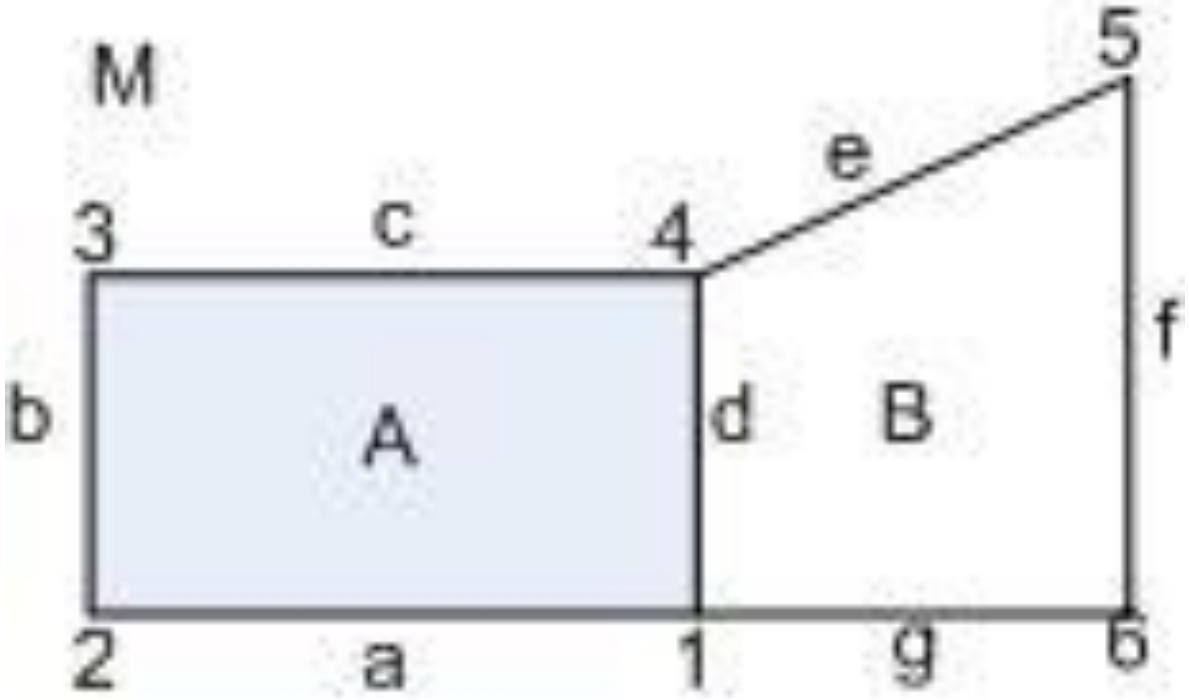
Implementacioni modeli podatak

- **Objektno-orjentisani** (O-O) model podataka
  - **Eliminiše razliku** između **entiteta i** međusobnog **odnosa** – što je bilo uobičajno u E-R modelu i njegovim ekstenzijama
  - Može se primenjivati i na **konceptualnom** nivou dizajna i na **logičkom** nivou

Implementacija DBMS

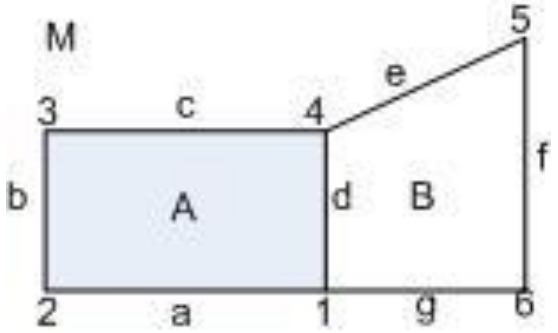
- U prve dve generacije svrstavaju se tzv. **hijerarhijski** i **mrežni** (CODASYL) DBMS
- III generacija - **Relacioni** sistem za upravljanje BP (RDBMS).
- Proširenje relacionog modela konceptima objektnog modela
  - **Objektno-relacioni** sistem za upravljanje BP (OR DBMS)
  - **Objektno orijentisani** sistem za upravljanje BP (OO DBMS)

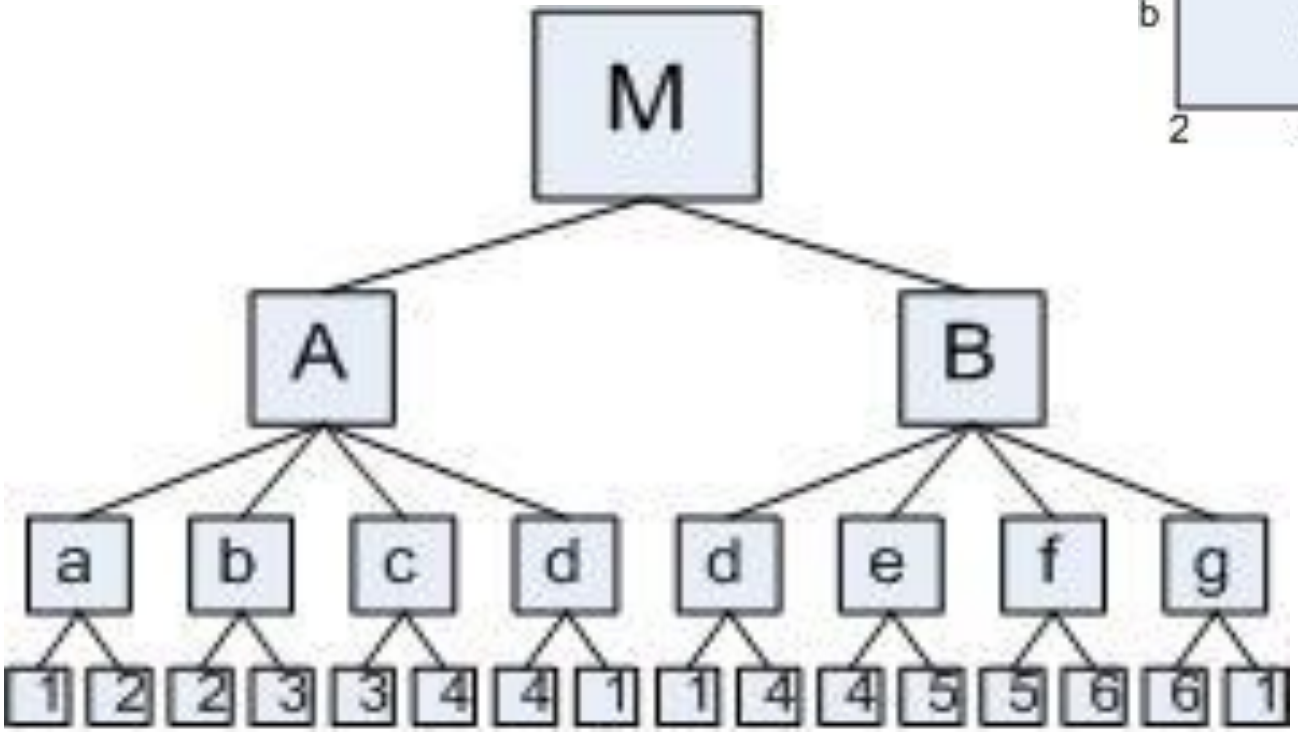
Primer iz geometrije: G. slika M sa dva četvorougla A i B



Hijerarhijski model podataka

Podaci organizovani u hijerarhijsku strukturu:





Hijerarhijski model podataka

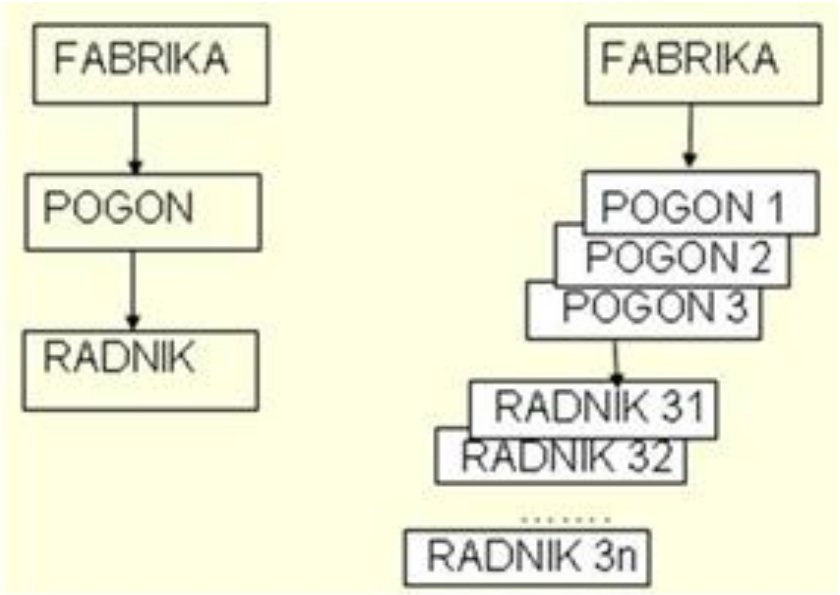
- Podaci organizovani u vidu ***hijerarhijske strukture*** - ***invertovanog stabla***
- Model:

1. Struktura podataka
   - ***Zapis*** (***segment***)
   - Na vrhu hijerarhije – ***koren segment***
2. Međusobni odnos entiteta
   - ***Link*** (***pointer***)
   - Svaki entitet (segment) ima samo ***jednog “roditelja”*** ali može imati ***više “dece”***

Hijerarhijski model podataka

- ***Pristupni putevi*** su tačno ***određeni***
- ***Pristup*** svakom ***segmentu*** izuzev korenskom segmentu vrši se ***preko segmenta "roditelja"***

Drugi primer:



Hijerarhijski model podataka

- Dobra karakteristika:
- Relativno ***jednostavan postupak pretraživanja*** baze
- Nedostatak:
- Redundantnost podataka
- Može jednostavno implementirati samo **1:N odnos** između entiteta
- ***Anomalije*** u operaciji ***ažuriranja*** – pri dodavanju i brisanju podataka

Hijerarhijski model podataka

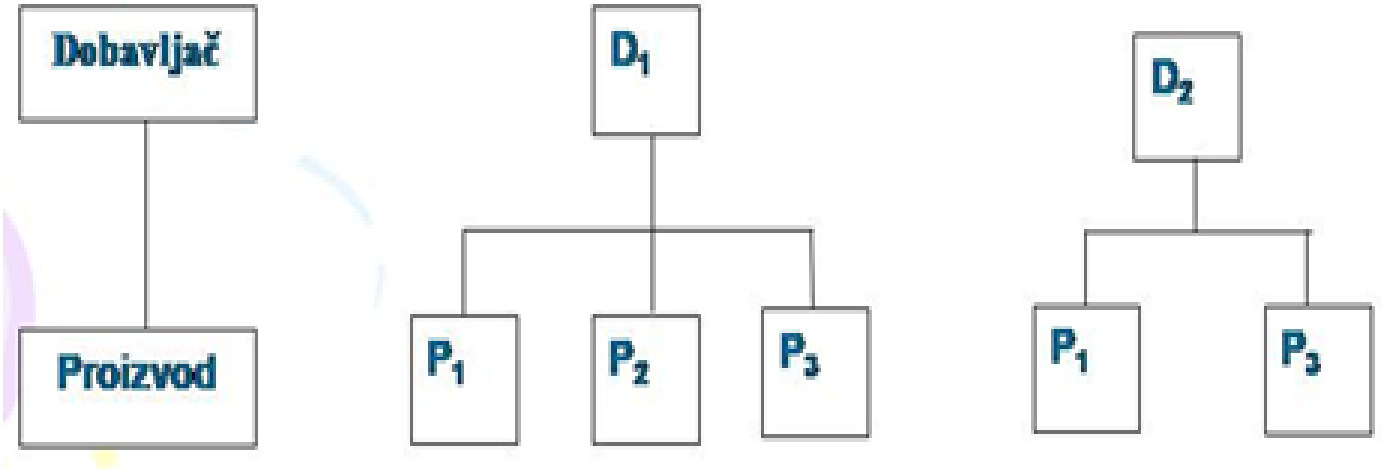
Primer:

A. Jedan ***dobavljač*** može da nabavlja jedan ili više ***proizvoda***

B. Jedan ***proizvod*** može da nabavlja jedan ili više ***dobavljača***

- ***U hijerarhijskom modelu*** može se predstaviti ***odnos*** da ***dobavljač*** bude ***na višem*** hijerarhijskom ***nivou*** od ***proizvoda***

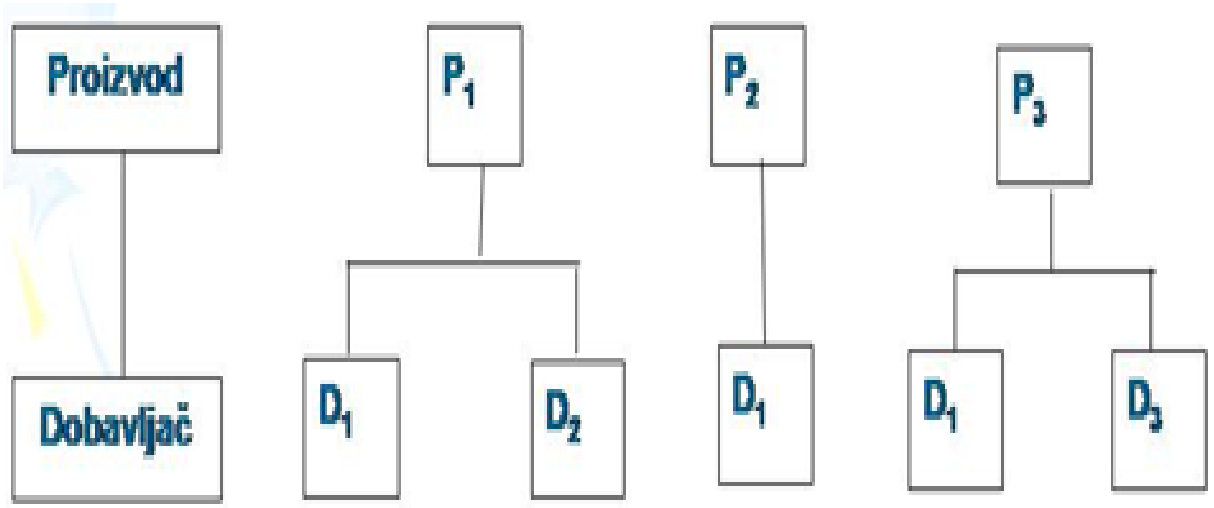
Hijerarhijski model podataka



- U tom slučaju, ***jednostavno*** je dobiti ***odgovor na pitanje*** –
  Koje proizvode nabavlja dati dobavljač?

Hijerarhijski model podataka

- Ako je potrebno ***odgovoriti na pitanje*** – Koji dobavljači nabavljaju dati proizvod?
  - ***Simetrični zahtev***
- Potrebna je drugačija struktura modela:



Hijerarhijski model podataka

- ***Osnovni zahtev za modele podataka*** u odnosu na programe za izveštavanje:

  Simitrični zahtevi za izveštavanjem rezultuju upitima ili programima ***iste (simetrične) složenosti***

- ***Razlika*** u složenosti programa koji realizuju simetrične upite pouzdan je ***indikator anomalija u ažuriranju***

Hijerarhijski model podataka

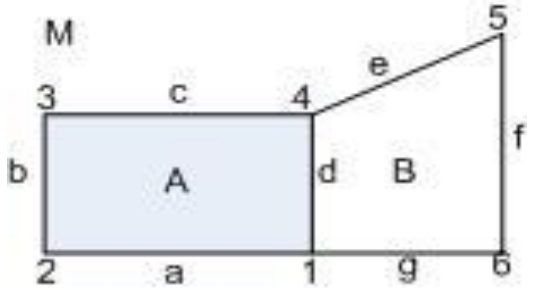
- ***Model podataka je pogodan*** za realizaciju programa za izveštavanje ukoliko je ***struktura upita saglasna sa strukturom modela podataka***
- Ukoliko model nije pogodan – ***postoje anomalije u primenama***

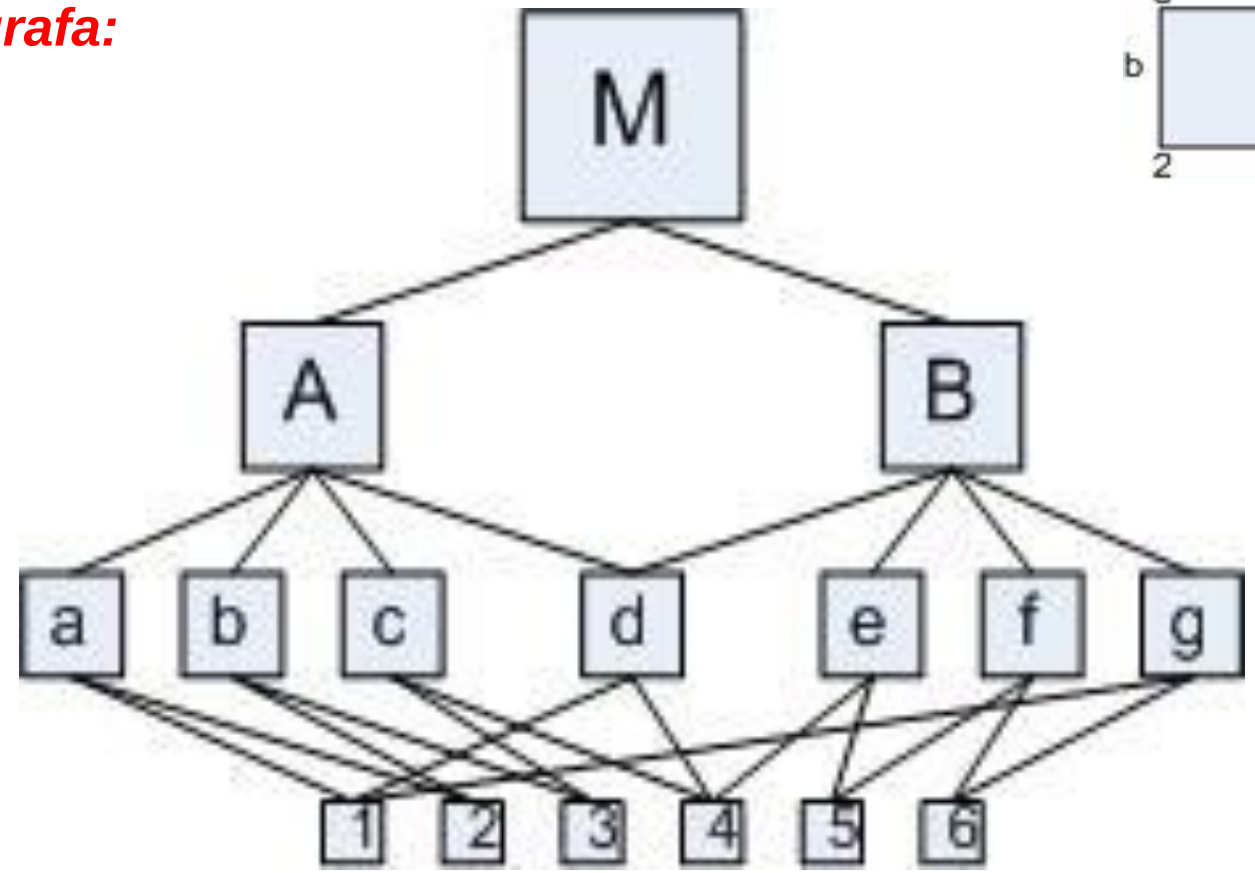
Anomalije hijerarhijskog modela podataka

- Anomalija u dodavanju novih zapisa
  - Ne može se dodati podatak o nekom proizvodu ako ne postoji dobavljač koji ga nabavlja
- Anomalija u izbacivanju – brisanju zapisa
  - Ako se obriše podatak o dobavljaču koji je jedini nabavljao neki prozvod, gubi se i podatak o tom prozvodu
- Anomalija u izmeni sadržaja
  - Ako je potrebno promeniti naziv nekog proizvoda, onda je to potrebno uraditi ne samo na jednom mestu, već na onoliko mesta koliko ima različitih dobavljača koji nabavljaju taj proizvod

Mrežni model podataka

Podaci organizovani u obliku orientisanog grafa:





Mrežni model podataka

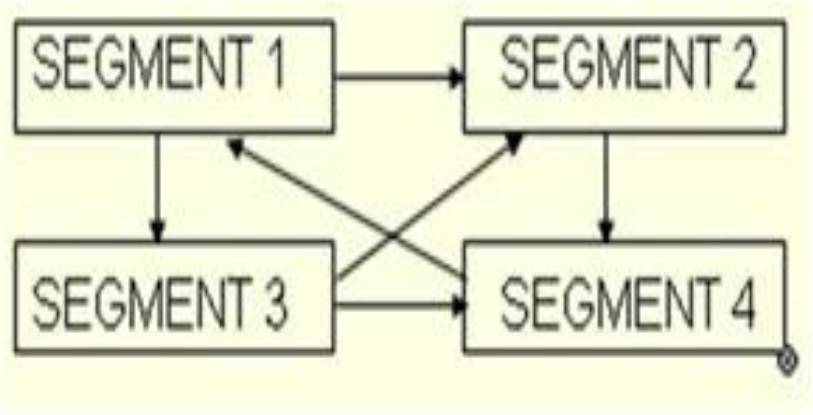
- ***1970 CO****nference on* ***DA****ta* ***SY****stem* ***L****anguages* – osnovana radna grupa za baze podataka
- Izveštaj CODASYL ***1975***. – ***modifikacija hijerarhijskog modela***
- Dve osnovne konstrukcije:
  - ***Zapisi***
  - ***Veze***

Mrežni model podataka

Mrežni model se predstavlja orjentisanim grafovima opšteg tipa

- Čvorovi predstavljaju klase entiteta
- Linije grafa prestavljaju odgovarajuće veze
- Set je takav tip veze kod koga jednom pojavljivanju jednog segmenta odgovara jedno ili više pojavljivanja drugog segmenta

Mrežni model podataka



- Jedan objekat može biti i nadređeni i podređeni
- Segment od koga polazi usmerena linija - "vlasnik" (Owner) seta
- Segment na koji pokazuje strelica je " član" (Member) seta

Mrežni model podataka

- Omogućava predstavljanje proizvoljnih vrsta veza između logičkih zapisa, pa i veze tipa više prema više
- Mrežni model podataka je skup međusobno povezanih logičkih zapisa u kojima preslikavanja između ključeva zapisa

  ((Km, Kn), (Kn, Km)) mogu biti bilo kog tipa

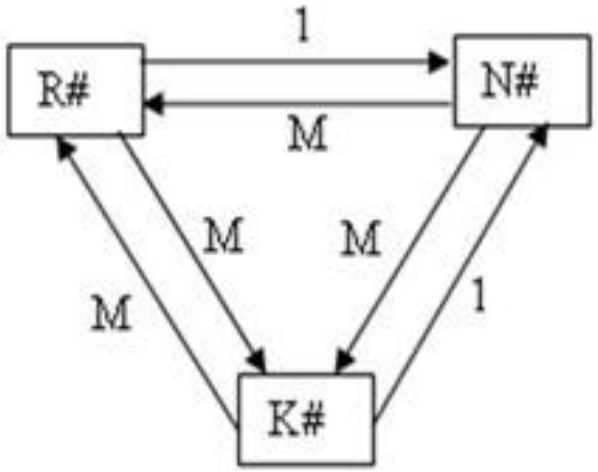
  Km, Kn ključevi bilo koja dva logička zapisa

Mrežni model podataka

Primer:

R# - Šifra radnika
N# - Šifra nastavnika
K# - Šifra kursa

Semantika modela:
Jedan nastavnik vodi sve kurseve



Direktno preslikavanje	Inverzno preslikavanje	Kardinalnost
((R#, K#)	(K#, R#))	(M : M)
((R#, N#)	(N#, R#))	(1 : M)
((N#, K#)	(K#, N#))	(M : 1)

Mrežni model podataka

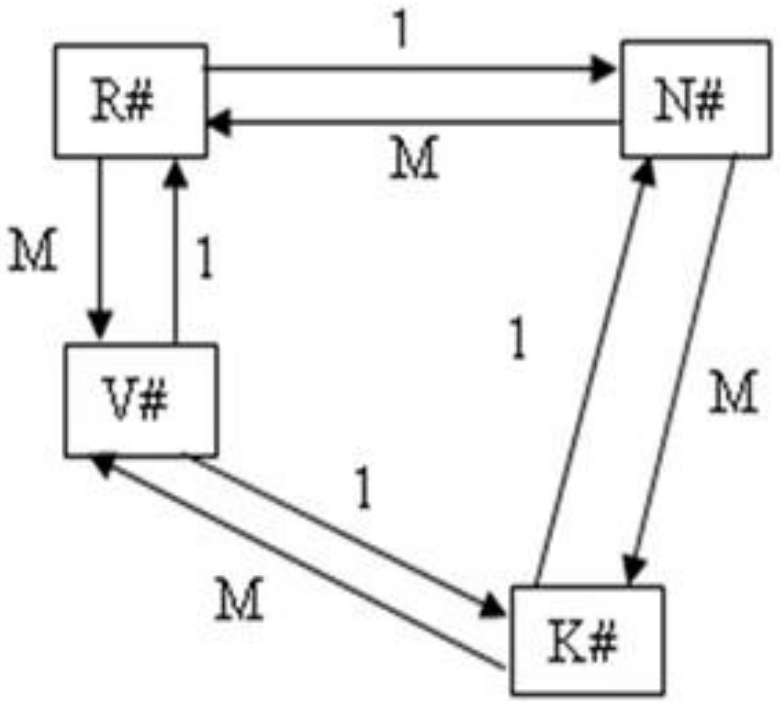
- Preslikavanje tipa (M : M) između dva zapisa realizuje se u mrežnom modelu preko dva preslikavanja tipa (M : 1) uz pomoć dodatnog zapisa, tzv. zapisa veze

- Zapis veze uvek ima složeni ključ sastavljen od ključeva zapisa koji su u vezi (M : M)

((Km, Kn), (Kn, Km))	M : M
((Kn, V), (V, Kn))	M : 1
((Km, V), (V, Km))	M : 1

Mrežni model podataka

- Složeni model sa prethodne slike pretvara se u prost mrežni model
- V# = K# + R#

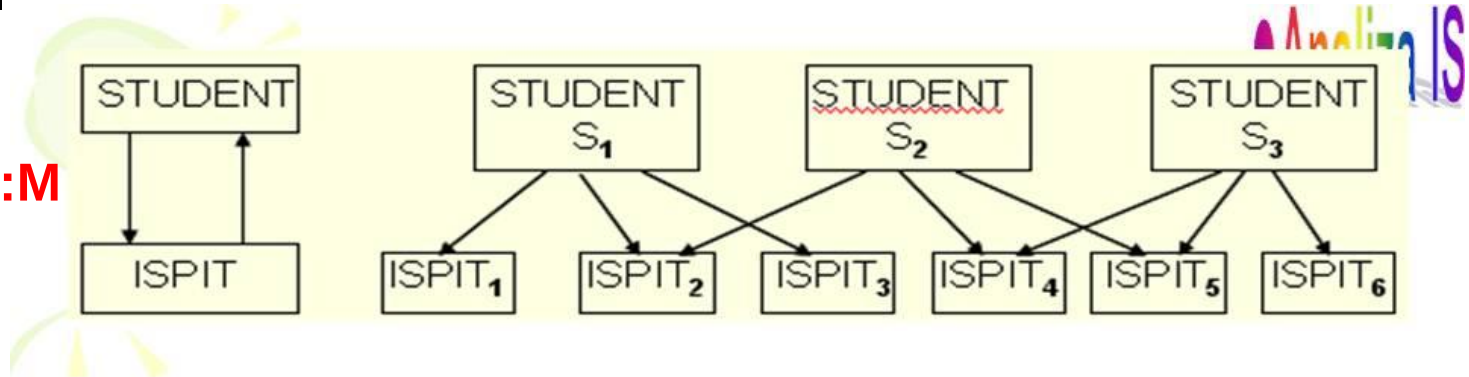


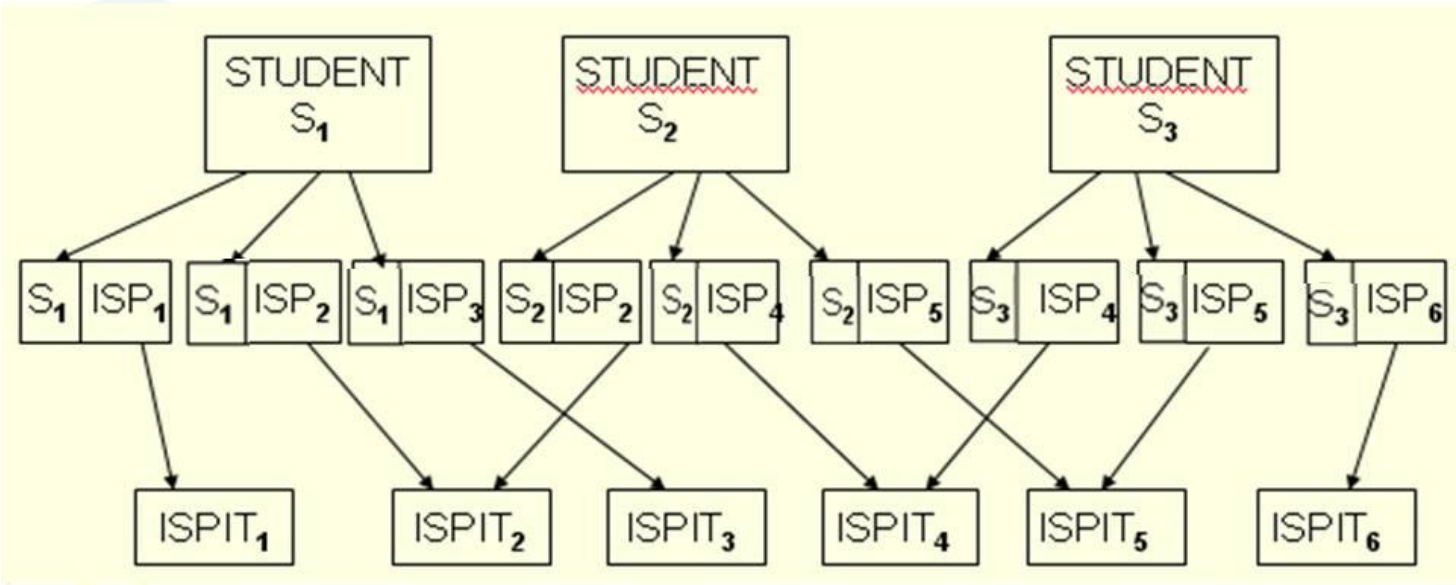
((Km, Kn), (Kn, Km))	M : M
((Kn, V), (V, Kn))	M : 1
((Km, V), (V, Km))	M : 1

64

Mrežni model podataka

M:M

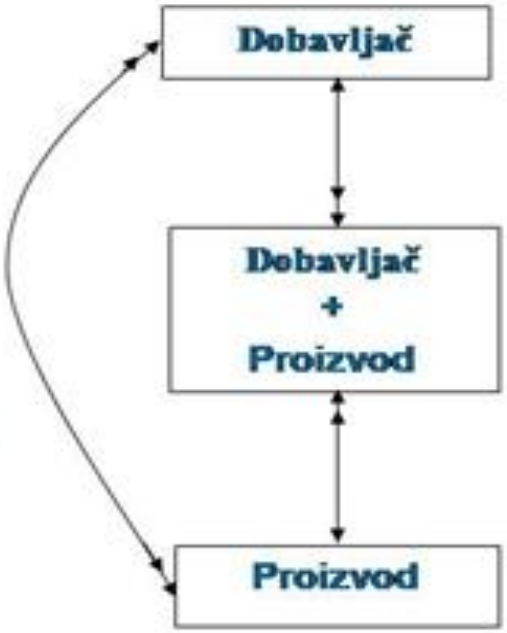




Mrežni model podataka

- M : M
- Složeni ključ

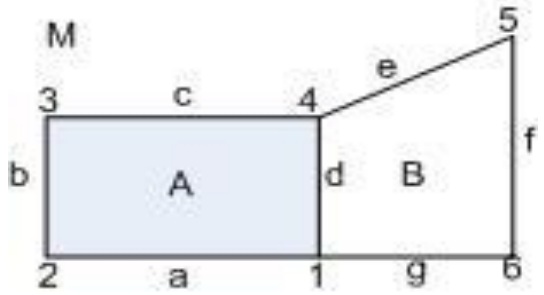
- Dve strelice ukazuju na preslikavanje tipa M u datom smeru
- Jedna strelica znači preslikavanje tipa 1



Osobine mrežnog modela podataka

- Mrežne baze podataka
  - sastoje se iz velikog broja rekorda (zapisa)
  - koji sadrže malu količinu informacija
  - sadrže veliki broj pokazivača prema drugim skupovima zapisa
- Složen za implementaciju i loše utiče na performanse
- Način realizacije preslikavanja M : M - izbegavaju se uglavnom svi problemi koji su se javljali kod hijerarhijskog modela
- Saglasnost strukture upita i strukture modela se gotovo uvek može postići

Relacioni model podataka



G. Slika

M	A	B

Četvorougao

A	a	b	c	d
B	d	e	f	g

Duž

a	1	2
b	2	3
c	3	4
d	4	1
e	4	5
f	5	6
g	6	1

Tačka

1	x1	y1
2	x2	y2
3	x3	y3
4	x4	y4
5	x5	y5
6	x6	y6

Relacioni model baze podataka

- ***Teorijski dobro definisan*** model rada i upravljanja skupom podataka
- Oslanja se na **tri** posebna **koncepta**
  1. Struktura podataka
  2. Integritet podataka
  3. Rukovanje podacima – jezik upita (SQL)

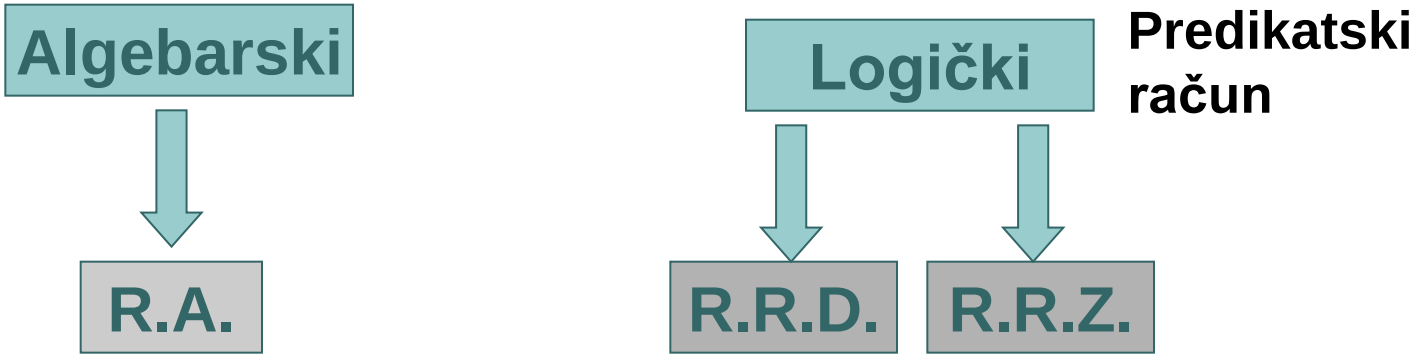
Relacioni model podataka – komponente i pristup

RMP = Strukture podataka + Dodeljeni operatori

Strukture podataka:

- ***Domeni***
- ***Atributi***
- ***n-torke (zapisi)***
- ***Relacije(Tabele)***

Dve familije operatora:



Relacioni model podataka – komponente i pristup

- **Algebarski pristup** vidi tabele kao **skupove**, i skup operatora rade sa njima kao **algebra**
- **Logički pristup** podrazumeva relacioni račun koji se zasniva na **predikatskom računu**
  - Relacioni račun domena (RRD)
  - Relacioni račun zapisa (RRZ)

Struktura podataka - tabele

- **Osnovno pravilo**: **podaci** se organizuju isključivo **u obliku tabela**
- **Tabela**, **tip entiteta** – formalni naziv **relacija**
- **Pravila** koja se odnose na tabele:
  - Sastoji se od **konačnog** **broja kolona** (polja, atributa, obeležja)
  - Proizvoljnog broja **redova** (zapisi, slogovi, n-torke)

Struktura podataka - tabele

- ***Relacija*** (***Tabela***) – skup informacija koje se odnose na datu vrstu objekta (tip entiteta) – tj. ***skup zapisa sa istim atributima***
- Kolone – **atributi**; Redovi – **zapisi**
- Svaka kolona u tabeli sadrži određeni tip podataka
  - Prilikom **definisanja tabele**, korisnik zadaje **tip podataka** za svaku kolonu

Struktura podataka - formalnije

- **Šema zapisa**:
- Šema zapisa, τ, je skup uređenih parova oblika:

$$\tau = \{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$$

gde je:

- $\{A_1, A_2, ..., A_n\}$ – skup naziva **atributa**
- $\{D_1, D_2, ..., D_n\}$ – **domeni** pridruženi odgovarajućim atributima

Struktura podataka - formalnije

- **Zapis**:
- **Zapis *t*, šeme zapisa** τ, gde je

$$\tau = \{(A_1,D_1), (A_2,D_2), \ldots, (A_n,D_n)\}$$

je skup parova oblika:

$$t = \{(A_1,v_1), (A_2,v_2), \ldots, (A_n,v_n)\}$$

šema zapisa

takvih da $\forall i\ v_i \in D_i$

zapis

Struktura podataka - formalnije

- ***Domeni***:
- Domen – dozvoljen skup vrednosti
- Problem: Šta će se desiti ako ne znamo vrednost koju zapis ima za neki svoj atribut?
- Rešenje u programskim jezicima: -1, “empty”, “”, 0, “No address”, “--”,....
- Rešenje u RMP – ***NULL*** vrednost (?)

Struktura podataka - formalnije

- **Operatori** nad zapisima:
- Dat je zapis: $t = \{(A_1,v_1), (A_2,v_2), \ldots, (A_n,v_n)\}$
- ***GET***
  - $v_i = GET(t,A_i)$
- ***SET***
  - $SET(t,A_i,w_i) = \{(A_1,v_1),\ldots,(A_i,w_i), \ldots, (A_n,v_n)\}$

Struktura podataka - formalnije

- **Operatori** nad zapisima:
- Usvojena označavanja:

GET(t, A_i):	$t.A_i$	$t(A_i)$
SET(t, A_i, w_i):	$t.A_i \leftarrow w_i$	$t(A_i) \leftarrow w_i$

Struktura podataka - formalnije

- **Šema relacije**:
- Relacija je skup zapisa sa istom šemom
- Šema relacije = šema zapisa koji čine datu relaciju
- Oznaka:

$$R((A_1,D_1), (A_2,D_2), ..., (A_n,D_n))$$

- Definiše relaciju ***R*** sa šemom

$$\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$$

Struktura podataka - formalnije

- Karakteristike relacije:
- ***Stepen relacije*** – **broj atributa** u njenoj šemi
- ***Kardinalnost relacije*** – **broj zapisa** koji čine relaciju
- ***Kompatibilnost*** – dve relacije ***R*** i ***S*** su kompatibilne ako su im **šeme identične**

RA – Relaciona Algebra

- ***RA*** – ***Skup*** **unarnih** i **binarnih** ***operatora*** definisanih nad relacijama
- ***Zatvoreni operatori*** – rezultat primene bilo kog RA operatora nad jednom ili dve tebele – je opet relacija (***tabela***)

RA – Relaciona Algebra

Skupovni operatori:
- ***Unija***
- ***Razlika***
- ***Presek***
- ***Dekartov proizvod***

Relacioni operatori:
- ***Selekcija***
- ***Projekcija***
- ***Deljenje***
- ***Spajanje***

Specijalni operator: ***Rename***

RA (*Rename* Operator)

$R((A_i, B_i), \ldots, (A_j, B_j))$ ili $R[B_1, \ldots, B_n]$

Neka je ***R* relacija** sa šemom

$$\{(A_1,D_1), (A_2,D_2), \ldots, (A_n,D_n)\}$$

Promena imena atributa u ***R***, A_i, ..., A_j u B_i,..., B_j, rezultuje u relaciju koja sadrži sve zapise iz relacije ***R***, ali sa odgovarajućom **promenom imena atributa**

RA (*Rename* Operator)

$R((A_i, B_i), \ldots, (A_j, B_j))$ ili $R[B_1, \ldots, B_n]$

$$R((A_i, B_i), \ldots, (A_j, B_j)) =$$

$$\{\{(A_1, v_1), \ldots, (B_i, v_i), \ldots, (B_j, v_j), \ldots, (A_n, v_n)\} \mid$$

$$\{(A_1, v_1), \ldots, (A_i, v_i), \ldots, (A_j, v_j), \ldots, (A_n, v_n)\} \in R\}$$

Šema rezultujuće relacije je:

$$\{(A_1, D_1), \ldots, (B_i, D_i), \ldots, (B_j, D_j), \ldots, (A_n, D_n)\}$$

RA (*Rename* Operator)

Napomena: *rename* operator se primenjuje samo na relacije a **NE** na šeme relacije

Primer: Neka je relacija Mostovi predstavljena sledećom tabelom:

Mostovi	
pkod	rkod
44	r2
46	r2
45	r3
28	r1
16	r1

RA (*Rename* Operator)

Primer:

Mostovi((pkod, kodPuta),(rkod, kodReke)) =
Ili,
Mostovi[kodPuta, kodReke] =

Mostovi	
kodPuta	kodReke
44	r2
46	r2
45	r3
28	r1
16	r1

RA (Unija)

$R \cup S$

R i ***S*** moraju imati **iste** šeme

Neka su ***R*** i ***S*** dve kompatibilen relacije sa šemom $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$

Unija ***R*** i ***S*** je relacija sa **istom šemom** kao i ***R*** i ***S***, formirana od **svih zapisa** koji pripadaju ***R***, ili ***S*** ili obema relacijama

$$R \cup S = \{ t \mid t \in R \vee t \in S \}$$

RA (Razlika)

$R - S$

R i ***S*** moraju imati **iste** šeme

Neka su ***R*** i ***S*** dve kompatibilne relacije sa šemom $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$

Razlika između ***R*** i ***S*** je relacija sa **istom šemom** kao i ***R*** i ***S***, formirana od **svih zapisa** koji pripadaju ***R*** i ne pripadaju ***S***

$$R - S = \{ t \mid t \in R \wedge t \notin S \}$$

RA (Presek)

$R \cap S$

R i ***S*** moraju imati **iste** šeme

Neka su ***R*** i ***S*** dve kompatibilne relacije sa šemom $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$

Presek ***R*** i ***S*** je relacija sa **istom šemom** kao i ***R*** i ***S***, formirana od ***svih zapisa*** koji pripadaju istovremeno i ***R*** i ***S***

$$R \cap S = \{ t \mid t \in R \wedge t \in S \}$$

RA (Dekartov proizvod)

$$R \times S$$

R i ***S*** **ne mogu imati ista** imena atributa

Neka su ***R*** i ***S*** dve relacije sa šemama

$\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$ i $\{(B_1,E_1), (B_2,E_2), ..., (B_m,E_m)\}$

Dekartov proizvod ***R*** i ***S*** je relacija čija je šema **unija šema *R* i *S***, formirana od **zapisa** koji mogu biti dobijeni **svim kombinacijama** jednog iz ***R*** i jednog iz ***S***

RA (Dekartov proizvod)

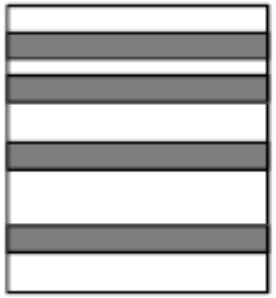
$$R \times S = \{ \{(A_1, v_1), \ldots, (A_n, v_n), (B_1, w_1), \ldots, (B_m, w_m)\} \mid \{(A_1, v_1), \ldots, (A_n, v_n)\} \in R \text{ and } \{(B_1, w_1), \ldots, (B_m, w_m)\} \in S\}$$

Šema rezultujuće relacije dobijena sa ***R*** **?** ***S***:

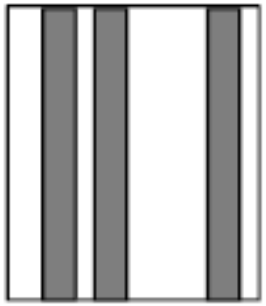
$\{(A_1,D_1), (A_2,D_2), \ldots, (A_n,D_n), (B_1,E_1), (B_2,E_2), \ldots, (B_m,E_m)\}$

RA (Relacioni Operatori)

Selekcija:



Projekcija:



Deljenje:

a	x
a	y
a	z
b	x
c	y

÷

x
y

=

a

Spajanje:

a1	b1
a2	b1
a3	b2

⋈

b1	c1
b2	c2
b3	c3

=

a1	b1	c1
a2	b1	c1
a3	b2	c2

RA (Projekcija)

$$\pi_{Ai, Aj, \ldots, Ak}(R)$$

Neka je ***R*** relacija sa šemom $\{(A_1,D_1), (A_2,D_2), \ldots, (A_n,D_n)\}$ i neka je $\{A_i, A_j, \ldots, A_k\}$ podskup imena atributa iz ***R*** sa ***m*** elemenata

Projekcija R na $\{A_i, A_j, \ldots, A_k\}$ je relacija koja je **definisana** na sledeći način:

$$\pi_{A_i, A_j, \ldots, A_k}(R) = \{ ((A_i, v_i), (A_j, v_j), \ldots, (A_k, v_k)) \mid \exists t \in R : ((A_i, v_i), (A_j, v_j), \ldots, (A_k, v_k)) \subseteq t \}$$

RA (Selekcija)

$\sigma_F(R)$

Neka je ***R*** relacija sa šemom $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n)\}$. Selekcija u ***R*** u odnosu na uslov ***F***, označavamo sa $\sigma_F(R)$, je relacija sa **istom šemom** kao i ***R***, koja je formirana sa **svim zapisima** iz ***R*** za koje **važi uslov F**

$$\sigma_F(R) = \{t \mid t \in R \wedge F(t) \text{ ima vrednost tacno}(T)\}$$

RA (Selekcija)

- **Kog je oblika uslov F**?
- Izrazi poređenja:
  - Null(A_i)
  - $A_i \; \alpha \; A_j$
  - $A_i \; \alpha$ a

gde je α **operator poređenja** (>,<, ≥, ≤, ≠, =)

A_i i A_j su **imena atributa**

a je **vrednost** iz domena atributa A_i

Uslov = izrazi poređenja + (i) + $(\vee, \wedge, \neg)$

RA (Selekcija)

- **Kako se vrši evaluacija uslova F**?
- NULL vrednost ❶ proširenje, ***tro-vrednosna logika***
  - ***{T, F, undefined}***
- Ako je ***F(t)*** u obliku:
  - $A_i \ \alpha \ A_j$ tada se ***F(t)*** računa kao *undefined* ako bar jedan od atributa A_i ili A_j ima vrednost NULL u ***t***; **inače** vrednost se dobija kao rezultat upoređivanja $t(A_i) \ \alpha \ t(A_j)$

RA (Selekcija)

- **Kako se vrši evaluacija uslova F**?
- Ako je ***F(t)*** u obliku:
  - $A_i \; \alpha \; a$ tada se ***F(t)*** računa kao *undefined* ako atributa A_i ima vrednost NULL u ***t***; **inače** vrednost se dobija kao rezultat upoređivanja $t(A_i) \; \alpha \; a$
  - $\text{Null}(A_i)$ tada se ***F(t)*** računa kao ***T*** ako atributa A_i ima vrednost NULL u *t*; **inače** vrednost je ***F***.

RA (Selekcija)

- **Tro-vrednosna logika:**

G	H	F = G ∧ H	F = G ∨ H
false	false	false	false
undefined	false	false	undefined
true	false	false	true
false	undefined	false	undefined
undefined	undefined	undefined	undefined
true	undefined	undefined	true
false	true	false	true
undefined	true	undefined	true
true	true	true	true

G	F = ¬ G
false	true
undefined	undefined
true	false

RA (Selekcija)

- **Primeri**:

Neka je ***R***

Lični_id	Ime	Adresa
20450120	Petar Petrović	Palih boraca 20
12904569	Miloš Milošević	Kočina 16
?	Janko Janković	Takovska 3
12345678	Zoran Radović	Slavija 29

Operacije:

$$\sigma_{\neg(\text{Ime} = \text{"Petar Petrovic"}) \wedge (\text{Licni\_id} > 12500500)} (R) =$$

$$\sigma_{\neg(\text{Licni\_id} > 12500500)} (R) =$$

RA (Teta (θ) Spajanje)

$$R \bowtie_F S$$

- Dekartov proizvod sa primenjenim **uslovom**

Studenti

stud#	Ime	kurs
100	Marko	EL
200	Ana	CS
300	Jovan	CS

Kursevi

kurs#	Naziv
EL	Elektronika
CS	C/S sistemi

Studenti $\bowtie_{stud\#=200}$ Kursevi

stud#	Ime	kurs	kurs#	Naziv
200	Ana	CS	EL	Elektronika
200	Ana	CS	CS	C/S sistemi

RA (Unutrašnje Spajanje)

$$R \bowtie_{R.strani\_k=S.primarni\_k} S$$

- Teta spajanje gde je F **poklapanje** (=) **primarnih i spoljnih ključeva**

Studenti

stud#	Ime	kurs
100	Marko	EL
200	Ana	CS
300	Jovan	CS

Kursevi

kurs#	Naziv
EL	Elektronika
CS	C/S sistemi

Studenti $\bowtie_{kurs=kurs\#}$ Kursevi

stud#	Ime	kurs	kurs#	Naziv
100	Marko	EL	EL	Elektronika
200	Ana	CS	CS	C/S sistemi
300	Jovan	CS	CS	C/S sistemi

RA (Ekvi-Spajanje)

$$R \triangleright\!\triangleleft S$$

Neka su ***R*** i ***S*** relacije sa šemama $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n),(B_1,E_1), (B_2,E_2), .., (B_m,E_m)\}$ i $\{(B_1,E_1), (B_2,E_2), .., (B_m,E_m), (C_1,F_1), (C_2,F_2), .., (C_p,F_p)\}$, tako da su atributi $B_1, .., B_m$ **zajednički u obe šeme**

RA (Ekvi-Spajanje)

$$R \triangleright\!\triangleleft S$$

Spajanje R i S je relacija koja sadrži sve zapise koji mogu biti formirani kombinovanjem zapisa iz ***R*** sa drugim iz ***S*** tako da oni imaju **iste vrednosti** za sve zajedničke atribute

$$R \triangleright\!\triangleleft S = \{\{(A_1, v_1), \ldots, (A_n, v_n), (B_1, w_1), \ldots, (B_m, w_m), (C_1, y_1), \ldots, (C_p, y_p)\} \mid \{(A_1, v_1), \ldots, (A_n, v_n), (B_1, w_1), \ldots, (B_m, w_m)\} \in R \wedge \{(B_1, w_1), \ldots, (B_m, w_m), (C_1, y_1), \ldots, (C_p, y_p)\} \in S\}$$

RA (Ekvi-Spajanje)

$$R \triangleright\triangleleft S$$

- Unutrašnje spajanje proizvodi **redundantne** podatke (kurs i kurs#). Da bi odstranili duplikate:

$$\pi_{\text{stud\#,Ime,kurs,Naziv}}(\text{Studenti} \bowtie_{\text{kurs=kurs\#}} \text{Kursevi})$$

- Ovaj rezultat se naziva – **prirodno spajanje** ili samo **spajanje** (*join*)

RA (Prirodno Spajanje)

$$R \triangleright\triangleleft S$$

- Formalno:

$$R \bowtie S = \pi_{atrib\text{-}lista}(\sigma_{join\text{-}uslov}(R \times S))$$

gde je

- ***atrib.-lista*** = ***atributi(R)*** ⇘ ***atributi(S)*** (duplikati se eliminišu)
- join-uslov je u formi:

$$R.A_1 = S.A_1 \text{ i } \ldots \text{ i } R.A_n = S.A_n$$

$\{A_1,\ldots,A_n\}$ = atributi(R) ⇘ atributi(S)

RA (Spajanje)

$$R \triangleright\!\triangleleft S$$

Studenti

stud#	Ime	kurs
100	Marko	EL
200	Ana	CS
300	Jovan	CS

Kursevi

kurs#	Naziv
EL	Elektronika
CS	C/S sistemi

stud#	Ime	kurs	Naziv
100	Marko	EL	Elektronika
200	Ana	CS	C/S sistemi
300	Jovan	CS	C/S sistemi

RA (Deljenje)

$$R \div S$$

Neka su ***R*** i ***S*** relacije sa šemama $\{(A_1,D_1), (A_2,D_2), ..., (A_n,D_n),(B_1,E_1), (B_2,E_2), .., (B_m,E_m)\}$ i $\{(B_1,E_1), (B_2,E_2), .., (B_m,E_m)\}$

Deljenje ***R*** sa ***S*** je relacija definisana kao:

$$R \div S = \{ \{(A_1, v_1),\ldots, (A_n, v_n)\} \mid \forall s \in S\,(s = \{(B_1, w_1),\ldots,(B_m, w_m)\} \rightarrow \exists\, t \in R \text{ and } t = \{(A_1, v_1),\ldots,(A_n, v_n), (B_1, w_1),\ldots,(B_m, w_m)\}) \}$$

RA (Deljenje)

$$R \div S$$

Šema rezultujuće relacije je

$$\{(A_1,D_1), (A_2,D_2), \ldots, (A_n,D_n)\}$$

Operator deljenja ***nije ni komutativan ni asocijativan***

- Ova operacija se može **izvesti** pomoću drugih operacija

Zadatak

Studenti

stud#	Ime	adresa	kurs
100	Marko	Beograd	EL
200	Ana	Valjevo	CS
300	Jovan	Beograd	CS

Kursevi

kurs#	Naziv
EL	Elektronika
CS	C/S sistemi

Kreirati ***izraz relacione algebre*** koji prikazuje ***imena studenata iz Beograda*** i ***nazive kurseva*** koje oni pohađaju.

Zadatak - Rešenje

Izrazi relacione algebre:

$$R1 = \text{Studenti} \bowtie_{\text{kurs=kurs\#}} (\text{Kursevi})$$

$$R2 = \sigma_{\text{adresa="Beograd"}}(R1)$$

$$R3 = \pi_{\text{Ime, Naziv}}(R2)$$

Ime	Naziv
Marko	Elektronika
Jovan	C/S sistemi

Zadatak

Problem: Naći imena studenata koji mogu da izađu na ispit iz predmeta Klijent-server sistemi.
Na ispit mogu da izađu samo oni studenti koji su položili sve predviđene kolokvijume iz tog predmeta.

Položeni Kolokvijumi(R)	
Imena_st	**Kolokvijum**
Marko	CSS1
Marko	CSS2
Marko	Kompajleri1
Goran	CSS1
Goran	Kompajleri1
Sara	CSS1
Sara	CSS2

CSS Kolokvijumi (S)
Kolokvijum
CSS1
CSS2

Zadatak - nastavak

Rešiti problem tako da se dobije najednostavniji mogući izraz relacione algebre (sa što manje operatora).