

BAZE PODATAKA

EKONERG

o virtualne machine

o root, bpadmuh, horvat, novak, kolar
+ shutdown

→ gazi se s "shutdown -h now"

CREATE DATABASE bla;

DROP DATABASE bla;

• CREATE TABLE mjesto
 pli INTEGER
 , nazMjesta CHAR(30)
 , sitwp SMALLINT
);

• DROP TABLE mjesto;

INSERT INTO mjesto
 VALUES (42000, 'Varazdin', 7);

↳ Unete je

podatke

SELECT * FROM mjesto
 WHERE sitwp = 7;

↳ dobivat podatke

ze where

UPDATE mjesto

SET nazMjesta = 'VARAZDIN'

WHERE pli = 42000; → sada pli je još novi

DELETE FROM mjesto

WHERE sitwp = 7; → brisati se se sitwp = 7

→ komentari: /* */ i -- u reka

EKONERG

2

2

listo u selekciju

SELECT * FROM table

Ovo znači sve iz tablice

np: SELECT naziv, mesto, plat
FROM mesto;

projekcija! JT

SELECT DISTINCT tenor, grad
FROM narudz;

→ izdvajanje vertikalnog podskupa
uz eliminaciju duplicita

tenor grad = JT(narudz, grad)

selekcija! OR

SELECT * FROM student

WHEREime = 'Ana'

OR poni>1000;

ime = 'Ana' V poni > 1000 (student)

kartažni produkt! *

SELECT *

[it]

SELECT student.* , product.*

FROM student, product;

FROM student, product

[it]

SELECT *

FROM student CROSS JOIN product;

primenjivanje atributa *

SELECT sifra_pojave AS sifraz

, naziv AS naziv

FROM napomene;

} Svi je se ispunili
nijeć AS!

* D J - RIGHT OUTER JOIN

* D L - LEFT OUTER JOIN

* D D - FULL OUTER JOIN

EKONERG

1)

Spajanje na vrst. $\Delta \Delta = \sigma(r \times s)$

SELECT *

FROM linje, zrakoplov
WHERE dolet >= udaljenost;

SELECT *

FROM linje JOIN zrakoplov
ON dolet >= udaljenost;

Spajanje na vrst i selekcije $\sigma(linje \Delta zrakoplov)$

SELECT *

FROM linje, zrakoplov
WHERE dolet >= udaljenost
AND udaljenost > 4000;

SELECT *

FROM linje
JOIN zrakoplov
ON dolet >= udaljenost
WHERE udaljenost > 4000;

Spajanje na vrst i projektije $\Pi(linje \Delta zrakoplov)$

SELECT DISTINCT tip

FROM linje, zrakoplov
WHERE dolet >= udaljenost;

SELECT DISTINCT tip

FROM linje
JOIN zrakoplov
ON dolet >= udaljenost;

Spajanje s izrednovarivanjem

SELECT *

FROM mjesto, zupanija
WHERE sifzap = sifzup;

SELECT *

FROM mjesto
JOIN zupanija
ON sifzap = sifzup;

Prirodno spajanje - izbrage se istovremeni atributi

SELECT mjesto.*; zupanija.* FROM

FROM mjesto, zupanija

WHERE mjesto.sifzap = zupanija.sifzap;

SELECT mjesto.*; zupanija.* FROM

FROM mjesto

JOIN zupanija

ON mjesto.sifzap = zupanija.sifzap;

Agregacija

COUNT - određuje broj prava (broj i duplike tako ih možeš)

SUM - sume vrijednosti

AVG - aritmetičke sredine

MIN

MAX

npr agregacija

`SELECT AVG(ocena) AS projOcj AS rezultat`

`FROM ispit;`

tablica u kojoj je

rezultat

EKONERG

grupiranje

`SELECT nazPred`

, alkod

, AVG(ocjena) AS projek

→ svih atributov koji se nalaze u listi
u selekciji a koji nisu argumenti
agregatnih funkcija, moraju! biti

`FROM ispit`

navedeni u GROUP BY olych

`GROUP BY nazPred, alkod;`

navedbi

ispisivanje podataka

`SELECT mjesto.* FROM mjesto, zupanja`

`WHERE mjesto.sifra = zupanja.sifra`

`AND nazime = 'Vukovarsko-srijemska'`

`ORDER BY nazMjesta;`

Opcionalni pojmovi

⇒ koriste se samo jednostavni navodnici!

⇒ da bi se pliš možio koristiti DECIMAL(m,n) tačan je zapis (bez
greški kao pri floatu u (-n))

DATE → pisanje se kao integer (mogu se datući matu i broj, soluzioni
zbrojaj ih dodavati / odjavljivati gledajući konstante)

⇒ `SELECT FIRST 2 * { ! } FROM student` → ne može se kreira tablica kao "prije drugi"
obitanci do porudžbe n-tika nije definiran

EKONERG

Moguce kombinacije u matem operatori:

npr SELECT mbr,ime || prez
FROM bodovalj;

mbr	(expression)
100	Ana Novak

npr SELECT mbr || '-' || ime
FROM bodovalj;

(expression)
100 - Ana

Funkcije

ABS (num-expression)

→ apsolutna vrijednost

MOD (dividend, divisor) -

→ vraca ostatak i pri radovanju se prima samo celi broj
dvo arguments

ROUND (expression, [rounding-factor])

→ zaokrujuje vrijednost, ako se ne navede, po def. 0

SUBSTRING (source_string FROM start-position FOR length)

→ vraca podniz

UPPER() i **LOWER()** → isto kao i u en

TRIM (source_expression)

→ izbacuje praznine s početka i krajem rednog niza

CHAR_LENGTH()
4cmos7

→ vraca broj znakova + PRATECE PRAZNINE!

OCTET_LENGTH()

→ broj byteova - PRAZNINE

EKONERG

USER

vraca login korisnika trenutno prijavljenog za id se bava

TODAY

vraca daninji datum (unike je osa)

MDY (month, day, year)

vraca date iz tri INTEGER varijable koje predstavljaju dan, mesec i godinu

DAY (date_exp)

vraca redni broj dana za datum

MONTH (date_exp)

redni broj meseca

YEAR (date_exp)

redni broj godine

WEEKDAY (date_exp)

vraca redni broj dana u tednu \Rightarrow 0=nedelja
1=ponedjeljak

\rightarrow ako se jedan ili više argumenta ne zade kao NULL vrijednost, rezultat je da tablica biti NULL vrijednost!

EKONERG

Uvjeti uporedbe

operatori uporedbe < <= = <> > >=

logički operatori AND OR NOT

◦ (NOT) BETWEEN X AND Y

◦ (NOT) IN (rješenje) ne podupit

◦ IS (NOT) NULL

◦ (NOT) LIKE % → razjeljuje bio bilo koliko znakova (0 or more)

- → razjeljuje točku i znak!

↳ četir znak naveden za ESCAPE znak za ponistavljanje

Spec. razdvojeno znakove %, _, #, ! - koji su navedeni između četiri znaka (mogu biti #, #, !, !)

↳ npr. SELECT * FROM tekstovi

WHERE tekst LIKE '%. \$ %'

ESCAPE #;

uvjetni riječi

CASE → poput switch u c-u

npr. SELECT *,

CASE

WHEN x THEN y

⇒ ako riječ nije true za myda
WHEN, NULL je rez

ELSE

END AS ...

⇒ ako je za više true, gleda se

FROM ispr;

Ono ne primi WHEN koji

zadovoljava

Unija

UNION ⇒ bilo koliko uz izbacivanje duplikata

UNION ALL ⇒ -||- BEZ izbacivanja duplikata

npr. SELECT * FROM polozio_mat

UNION

SELECT * FROM polozio_Rlog;

From naredba

→ kod spajanja VJEĆE SPOMJENJA navesti u ON djelu, a
VJEĆE SELJEĆE u WHERE djelu

paralelno spajanje

→ spajano relaciju samo se dobiva! (pri se 84. sljede 4. pret)

```
SELECT mbr_prez, prezime, mjesto_r.naziv_r AS nazivest_r,
       prezime, mjesto_s.naziv_s AS nazivest_s
  FROM student,
       mesto AS mjesto_r, mjesto AS mjesto_s
```

GROUP BY

- podan ili više atributa relacije navedeni u FROM djelu
- nije dozvoljeno koristiti ključne teme za grupiranje i mrežu

HAVING

- samo oni koji zadovoljavaju upit
- bitno je da su atributi navedeni u GROUP BY djelu
- ne pojeduljiva se grupa je False li niti vrijednost

ORDER BY

- sortiraju rezultante upite
- DESC - silans, ASC - ulans
default

- može se referisati na zajedničko ime

PODUPITI

EKONERG

→ Upit untuk update

mp: SELECT *

FROM vorile

WHERE normost > (SELECT MAX(terima)

FROM trete);

→ Note di update u:

WHERE ? u unit
HAVING

SELECT → u listu za selekcijs

→ note se dalamai se dijelove SELECT - hasilnya ocm ORDER BY
mp:

→ u SELECT

SELECT *, (SELECT MAX(terima) FROM trete)
AS Maxterma
FROM vorile;

Korelirani podupit

Select omust, dopBrsati
FROM rHoj

WHERE dopBrsati < (SELECT SUM(misahirada)

FROM radRok

WHERE omust = stigi omust);

-u HAVING : Dijem

SELECT predmet, SUM(bilstd) AS ukupostud
FROM rekreasi

GROUP BY predmet

HAVING SUM(bilstd) < (SELECT sum(kapacitet) FROM dwara

WHERE omdu like 'D%');

polnostu počini podupit

↳ U WHERE i HAVING dupl. "vauzig". upite

hyp

SELECT *

FROM stud ALL

WHERE prez <|> (SELECT prez FROM nastavnik);

ALL, ANY, SOME

(SOME = ANY)

→ faktot IN (podupit) ≡ SOME

NOT IN <|> ALL ⇒ NULL ih sjećava! ≡ vrijek false

→ operator (NOT) EXISTS → ne uđeće se ujedno sa pojmom NULL vrijednosti
 prenik
 vrednost

INSERT

INSERT INTO mjesto (ime, prez, ...) VALUES ('ime', 'prez', ...)

VALUES ('ime', 'prez', ...) ab neči redom kojim su te

→ MOJE DE korisni i

INSERT INTO polozioFiz

SELECT stud.mbr, ime, prez

FROM stud, ispit

WHERE stud.mbr = ispit.mbr

AND pedat = 'Fiz'

AND ocjena > 1;

Ukoliko ne ovaj način dodaje, tada navedi se

atributke, tako ne navedeno ne bi se postavljao detaljnija vrijednost - ako je nene ovde NULL

EKONERG

DELETE

```
DELETE FROM mbito  
WHERE sifzap = 7;
```

11
DELETE FROM polzorofiz
WHERE mbit NOT IN
(SELECT mbit FROM ispit
WHERE predmet = 'Fiz'
AND ocena > 1);

UPDATE

```
UPDATE ispit
```

SET ocena = ocena + 1, blbod = NULL
WHERE ocena < 5;

```
UPDATE ispit
```

SET (ocena, blbod) = (ocena + 1, NULL)
WHERE ocena < 5;

Ako se ne navede, nije je ce vrijednost svih kolonika u tablici!

```
UPDATE bodovi SET
```

bodovi = 'Cafe'

WHEN Bodovi + 2 = 50

THEN Bodovi - 2

ELSE Bodovi

END;

Oblikovanje skupne relacijske baze podataka

izbjegavane:
-redundancija (anomalijski ulaz/pisaj i riga)
-pojam raznih n-torki

↳ funkcja to su **FUNKCIJSKE ZAVISNOSTI**

kada jedan atribut jedinstveno određuje drugi atribut
ili skupinu atributa \Rightarrow postoji između njih veza

Armstrongovi akcioni

A1 REFLEKCIJNOST Ako je $X \subseteq Y$ vijedi $X \rightarrow Y$

A2 UVEĆANJE Ako je $X \rightarrow Y$, tada je i $XZ \rightarrow Y$

A3 TRANZAKTIVNOST Ako je $X \rightarrow Y$; $Y \rightarrow Z$ tada je i $X \rightarrow Z$

\Rightarrow pravilo koja proizlaze iz aktioma

P₁ pravilo UNIJE

ako $X \rightarrow Y$ i $X \rightarrow Z$, tada i $X \rightarrow YZ$

P₂ pravilo DEKOMPOZICIJE

ako vrijedi $X \rightarrow YZ$ onda i $X \rightarrow Y$

P₃ pravilo o PSEUDOTRANZITIVNOSTI

ako vrijedi $X \rightarrow Y$ i $Y \rightarrow Z$ tada je $X \rightarrow Z$

\oplus pravilo o AKUMULACIJI

ako vrijedi $X \rightarrow VZ$ i $Z \rightarrow W$ vrijedi $X \rightarrow VZW$

ključ entiteta

→ sadrži one atributke koji omogućuju da se pojedini entitet mogu razlikovati od ostalih

ključ relacije

→ grupa atributke koji nedvosmisleno određuje n-ticu relacije

↳ funkcionalno određuju atributke u preostaloj dijelu relacije

↳ minimalan

NORMALIZACIJA

→ dekompozacija i sinteza dolazi do normalnih formi
 → relacija se bez gubitaka razlaže na ^{najveće} podne propkrelje

◦ propkrelje imaju zajedničke atributke

◦ zajednički atributi su ključ u barem jednoj od propkrelje

1NF

- domen atribut sadrži samo jedinstvene (nedjeljive) vrijednosti
- svaki atribut je samo jedna vrijednost iz domena tog klijeca
- neključni atributi relacije funkcijeski ovise o ključu relacije

EKONERG

II **2NF** → ako je u INF i ako je svaki atribut iz zavisnog
djelatnog ovisa o svakom ključu relacije
potpuno funkcionalno

→ svaki atribut Y potpuno je funkcionalni ovis o skupu atributa X
ako vrijedi:

- Y funkcijalni ovis o X
- ne postoji plavi podskup od X koji funkcijalno određuje Y

$$\text{np., } F = \{ABC \rightarrow DE, E \rightarrow F\}$$

$EDF \}$ potpuno ovise o $\{ABC\}$ da je već poznati $Z \in \{ABC\}$
tako da $Z \rightarrow ED \}$

III **3NF** → ako je u INF i ako svaki jedan atribut iz zavisnog djelatnog
trenutnog ovisa o bilo kojem ključu relacije

→ svaki atribut je transitični o X ako vrijedi
 $X \rightarrow Y$, $Y \not\rightarrow X$ i $Y \rightarrow Z$

N **BOYCE-CODD NF** → ako je u INF i ako svaki jedan atribut nije
trenutno funkcijalni o bilo kojem ključu relacije
→ stoga 3NF

FIZIČKA ORGANIZACIJA PODATAKA

EKONERG

→ ne užice ne rezultate operacija s podacima, ali imo velik utjecaj na učinkovitost sustava te upravljanje bazu podataka

organizacija men = velike brojne pristupe podacima; slape

→ kapacitet redoslijed i neposredna memorija

organizacija men → h-i operacije spore

→ metode pristupa podacima: · neposredna (heap) dodatak

· b-stablo

· sorted file, hash, index-segregated file

B-STABLA

strukture = dugim putevima od kojeg do drugog

b-stabla = najveći dugi putevi od kojeg do liste

b+ stabla = najveći puti koji mora biti manji

→ balansirano stablo = dubina jednaka za svaki list u stablu

⇒ B⁺ stablo reda n

• interni list = najveći n karaktera

• najmanji $\lceil \frac{n}{2} \rceil$ karaktera

put p kroz listu u koraku, broj preprodih vrijednosti je p-1

olist = najveći n-1 vrijednosti i priznatih karaktera

• najmanji $\lceil \frac{(n-1)}{2} \rceil$ i karaktera

→ za dohvrat jednog čvora potrebna je jedna UII operacija

INDEXI

CREATE INDEX oznaka_poz ON oznaka (paz);

↑

formira se struktura b-stabla

EKONERG

`CREATE [UNIQUE] INDEX ime_indexa ON tablica (atribut);`

osigurava se jedinstvenost vrijednosti nekogog atributa

`DROP INDEX ime_indexa;` => uklanjanje indeksa

→ indeksi se stvaraju za primarne i alternativne ključeve referencijske
strukture, te za atribuke prema kojima se često obavlja
sličanje ili koji se često koriste za postavljanje vrste selekcije

note se još dodati

ASC i.e. DESC

→ sloren indeksi

`CREATE INDEX ime_indexa ON tablica (atribut1, atribut2);`

↳ može se koristiti za upite koji u vrtku koriste razne atribute i atribute
ili gde je sami atributi, no tada gde je sami atributi ne u iste
se koristi ovaj indeks

INTEGRITET BAZE PODATAKA

↳ bavist se integritetom podataka sačuvanjem u bazi

↳ može biti manuelno ili automatski pogreškom korisnika kod unosa/izmjene
ili pogreškom aplikacijskog programera/sustava

rijecnicke baze podataka → može se pohranjivati integritetna ograničenja

↳ pravila postavlju nezadovoljstva za nakogn korišćenja sustava

integritetna ograničenja

I. ENTITETSKI INTEGRITET

II. INTEGRITET KLJUČA

III. DOMENSKI INTEGRITET

IV. OGRANIČENJA NULL VREDNOSTI

V. REFERENCIJSKI INTEGRITET

VI. OPĆA INTEGRITETSKA OBGRANIČENJA

I. ENTITETSKI INTEGRITET

→ biti jedan atribut primarnog ključa ne smije biti NULL

II. INTEGRITET ključa

→ u relaciji ne smiju postojati dve n-torce s jednakim vrijednostima ključa (vrijedi za sve moguće ključeve)

III. DOMENSKI INTEGRITET

→ atribut mora popuniti samo 1 vrijednost iz domene atributa

IV. OGRANIČENJE NULL VRIJEDNOSTI

→ o. određene atribute se mora definisati ograničenje prema kojem vrijednost atributa ne smije popuniti NULL vrijednost

V. REFERENCIJSKI INTEGRITET (STRANI KLJUČ)

→ n-torce je jedna relacija koja se poziva (referencira) na drugu relaciju i može povratiti referenciju samo na pravoj n-torce (primarni klj.) u kojoj relaciji

→ biti mora jedna relacija uz primarni ključ i u istom ključu (atribut koji je ključ u nekoj drugoj relaciji) te se ponovo upozri u tom referencirati redosredno!

→ Morate i na sebe sebe referencirati

IMPLEMENTACIJA U SQL-U

PRIMARY KEY

CREATE TABLE repit (

mbn INTEGER,

sifprod INTEGER

datkip DATE

:

PRIMARY KEY (mbn,sifprod,datkip)

);

→ sv. BP omogućava

entitetski integritet

i integritet ključa

EKONERG

UNIQUE → osigurava integritet ključeva (Nije dozvoljeno da se u tabeli pojavljuje isti vrednost)

CREATE TABLE nastavnik (

tv.vrednost)

sifnost INTEGER,
 jmbgNast CHAR(13),
 prezNast CHAR(40),
 PRIMARY KEY (jmbgNast) } niste dozvoljeno
 , UNIQUE (jmbgNast) } da krajnji, a ako je samo 1, potreba
);

CHECK → osigurava domensku integritet

→ niste dozvoljeno upisati nevažeće vrijednosti

CREATE TABLE ispit (

, log SMALLINT CHECK (log BETWEEN 1 AND 5)
 , sifraNast INTEGER
);

→ niste dozvoljeno upisati vrijednosti atributa u intervalu

CREATE TABLE radnik (

datrod DATE
 , datzap DATE
 , CHECK (datzap - datrod >= 16 * 365 AND datzap - datrod <= 65 * 365)

NOT NULL

→ nisu dozvoljene prazne vrednosti atributa pri definiciji

FOREIGN KEY (ključ) REFERENCES tablica (kak se naziva)

→ niste dozvoljeno da uvek iste vrednosti u različitim tablicama

→ niste dozvoljeno da uvek u tablici b' uvek u tablici povezani atributi pri definiciji

→ niste dozvoljeno da uvek u tablici b' uvek u tablici povezani atributi pri definiciji

kad
je zapisan

ON DELETE SET NULL

ON DELETE SET DEFAULT → gde su potrebni parametri da se uvek u tablici povezani atributi uvek u tablici b' uvek u tablici povezani atributi pri definiciji

ON DELETE CASCADE → kada uvek u tablici povezani atributi uvek u tablici b' uvek u tablici povezani atributi pri definiciji

Izmjenov primjerog ključa • ON UPDATE NO ACTION

SET NULL

SET DEFAULT

CASCADE

→ korovjej ograničenja CONSTRAINT (ne ograničuje)

• primjer uklanjanje ograničenja

ALTER TABLE imenica DROP CONSTRAINT (ne ograničuje)

→ neću surazno podrazum

• ON DELETE CASCADE

• ON DELETE NO ACTION

• ON UPDATE NO ACTION

ALTER TABLE tablica ADD CONSTRAINT - PRIMARY KEY

- CHECK

- FOREIGN KEY ... REFERENCES

PRIJEMNE I VIRTUALNE RELACIJE

• vrste relacija:

• TEMELJNA RELACIJA → shema i sadržaj trajno pohranjen u bazi podataka

• PRIJEMNA RELACIJA → shema i sadržaj pohranjeni privremeno

• VIRTUALNA RELACIJA → shema i sadržaj definirani izazom

(relacijske algebre) koji su operandi temeljni ili virtualne relacije

→ u praksi se shema i sadržaj opisuju u obliku SQL upita

• ovaj je trenutakom stavljen temeljnih relacija!

TEMELJNA → shema i sadržaj Postojanje

SQL-sessiju → jedan korisnik obavlja SQL upite putem jedne veze

EKONERG

CREATE TEMP TABLE → privremena relacija

- ↳ vidljiva isključivo unutar SQL-sessija u kojoj je i stvorena
- ↳ uklanjaju se sa DROP TABLE ili gorenjem SQL-sessijom

↳ privremena tablica se **neće** automatski promjeniti ako se promjeni podaci u temeljnoj relaciji na koju je vezana!

CREATE VIEW → podaci su "up to date", radenju se tek u trenutku obavljaju upite

- ↳ može imati sve iz SELECT naredbe osim ORDER BY i FIRST n

DROP VIEW imenje

- ↳ obaveštjujući naredbu se pohranjuje kao definicije virtualne relacije
- ↳ mogu u bojisti smatrati gdje i u temeljnoj relaciji
- ↳ definicije virtualne relacije je highly pohranjene u bazi podataka
- ↳ virtualne funkcije su u SQL-sessiji
- tako se drugačije u naredbi, kada nazive atribute je temeljne relacije
- tako se koriste izraz, nazivi atributa se mogu eksplizivno navedi!

CREATE VIEW projek (sifred, proslog) AS

SELECT sifred, Avg(oz)

FROM polozaj_izpt

GROUP BY sifred;

Implementacija virtualnih relacija

- ↳ implementacija upita
- brige za materijalizirane virtualne relacije

→ mogu se koristiti u naredbama INSERT, UPDATE, DELETE

- ↳ no ande se podaci mijenjaju u temeljnim relacijama

↳ problem migracijom n-tacki

WITH CHECK OPTION - ne dozvoljava menjanje u n-tacki preko virtualne relacije ako n-tacke nisu dozvoljene operacije nije ne bi propustio virtualog reda putem kojeg je izvršena transakcija

EKONERG

20

Izvještive virtuelne relacije

• ako nastoji je samo jedna temeljna relacija i neuna!

◦ neuna DISTINCT

◦ neuna relacija u SELECT

◦ roštanjeni atributi ne mogu imati NOT NULL ograničenje (č. naredba može da bude vr)

◦ neuna spajanje (u UNION)

◦ neuna HAVING i GROUP BY

Izvještive

CREATE VIEW poloziliwiste AS

SELECT * FROM std

WHERE NOT EXISTS

(SELECT * FROM ispt

WHERE ispt.mbi = std.mbr

AND ogj > 1)

WITH CHECK OPTION;

NE Izvještive

CREATE VIEW poloziliwmo AS

SELECT DISTINCT mabi

FROM ispt

WHERE ogj > 1;

Implementacija stvarnih relacija u virtualnim relacijama

ožo različite kategorije konstrukcija definicija se razlikuje upravljač (stvarne relacije)

PORTRAJENE PROCEDURE I OKIDAČI

◦ portrajen procedure (funkcije) - pottopljeni begin je polneyen u njima su podatke i rezultat se u kontekstu SUBP

→ procedure → track rec

→ funkcije → track rec

CREATE FUNCTION brojznam (niz CHAR(25))

RETURNING SMALLINT AS broj

DEFINE brojac ;i SMALLINT

LET brojac = 0

FOR i=1 TO CHAR_LENGTH(niz)

IF SUBSTRING (niz FROM i FOR 1) BETWEEN '0' AND '9' THEN LET brojac=brojac+1

END IF;

END FOR;

RETURN brojac;

END FUNCTION;

EKONERG

GRANT EXECUTE ON imf-funkcije TO komej PUBLIC, che-koncove;

6) now se npr. EXECUTE FUNCTION imf-funkcije ('');

DONOLE

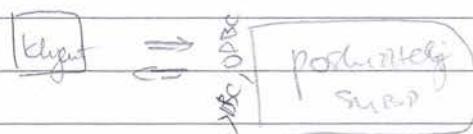
GRANT EXECUTE ON {procedure | function}
TO {PUBLIC | userlist | rolelist}
[WITH GRANT OPTION]

REVOKE EXECUTE ON {procedure | function}
FROM {PUBLIC | userlist | rolelist}
[CASCADE | RESTRICT]

Izimljuje

RAISE EXCEPTION → signalizira neku pogrešku koju su se mogu prepoznati
npr.: RAISE EXCEPTION -> "0, "test_greska"

→ predstavlja proceduru o proštrici neoguščene SQL-a
o različita podatkovna mesta funkcije
o onoguščene uporabljajo klijent-podatkovni arhitekturam
overe produtivnost i manje neuguščene pogreške



ECA on event → uvoz, izvoz li brišanje podataka
if condition then action

CREATE TRIGGER insbla

INSERT ON update iplate

REFERENCING NEW AS novi_update iplate

FUR ELEFT ROW

WHEN (now_update > 0 or < 0)

(UPDATE ... ←)

EKONERG**TRIGGER**

omogće je specifičan kada se okreće obravljaju

opo jednom za svaku n-toru koja je aktivirala sledeća

FOR EACH ROW

osam jedn, način operacije koja je aktivirala trigger

APFTER INSERT / UPDATE / DELETE

osam jedn, pje - - -

BEFORE INSERT / UPDATE / DELETE

◦ DROP TRIGGER im;

prvina implementacije integrativnih organizacija

!!!◦ pridruži reda korisnika

◦ sustavi obaveštavanja

INSTEAD OF skidači → mehanizma koji daju virtualne relacije pozicijama

strukture novih tablica

CREATE PROCEDURE (FUNCTION) Ime_procedure (eventualni arg.)

definisi varijable → DEFINE Ime CHAR(20);

vezedbe; → it! DEFINE Ime LIKE student.ime%;

korakbe;

END PROCEDURE (FUNCTION);

LET imc = 'Jora' → pridruži vrijednost atributu.

LET sum_og = (SELECT sum(ogcaj) FROM ogcaj);

IF uslov THEN

vezedbe

ELIF uslov THEN

vezedbe

ELSE

vezedbe

END IF;

WHILE uslov

vezedbe

break EXIT WHILE;

CONTINUE WHILE;

END WHILE;

For i=m TO n STEP k

vezedbe

EXIT FOR;

CONTINUE FOR;

END FOR;

impl DEFINE V-linedash like student.mined

SELECT the final ...

INTO V-inequid

from student

where $m_{\text{final}} = 12345$;

CREATE PROCEDURE ime (imetArg tip, imetArg tip, ..)

* —— it live (line big like nekaj, nekaj, ...)

CREDIT PROCEDURE

RETURNING tip AS line , (CHTR(25) AS lines)

postagi i RETURN hr RETURNING

point: EXECUTE PROCEDURE (FUNCTION) inc ();

OPTIMIRANJE UPITA

- administrator is responsible for **UPDATE STATISTICS**

- every tree - natural products 12.12.2023 relational algebra

• Metodika i redoslijed montaže optičkih
odeljaka: plan - montaža - operacija

170

The Mycotic = the

~~the way she~~

α pale blue

Ogdenia
glauca

Filmsko 2.vj

Gneiss

→ ophirange species suitable for breeding at marsh plant
species.

algebraiske transformacije:

- primodus spajanje ◦ komutativnost i asocijativnost
- $X_1 \cup X_2 \cap X_3 \rightarrow X_2 \cap X_1 \cup X_3$
- \Rightarrow spajanje (spajanje u uvjet) nije uvjet asocijativnost!

planite

• selekcija

$$\text{VANDE } \delta_C(\text{AND } \delta_{C_1}(r)) = \delta_{C_1}(\delta_C(r)) = \delta_{C_2}(\delta_{C_1}(r))$$

$$\text{kor } \delta_C(\text{OR } \delta_{C_1}(r)) = \delta_{C_1}(r) \vee \delta_{C_2}(r)$$

• ako je uvjet je primjenjuju samo na r
 $\delta_C(r \Delta S) = \delta'_C(r) \Delta S$

Vrijedi i za kartezijev produkt i spajanje uz uvjet

Heurističke pravila → prioritare iz istraživača

1. posiskavajuće selekcije → u tom lanacu faw

2. kombinirajuće operacije selekcije: Kartezijev produkt

• Šime zavojne selekcije

)

\Rightarrow

$\Delta \Delta$

Šime zavojne selekcije

\nearrow

\searrow

fikske vrij. gumeni

\nearrow

\searrow

fikske vrij. gumeni

→ vario je veličine rezultatata!

Govim o: broj n-torki u rezultatih
 veličini n-torce u rezultatih

EKONERG

25

~~prosjek veličine rezultata spajanja~~ → neka je $t = r \Delta s$

1. $X \cap Y = \emptyset - N(t) = N(r) * N(s)$

2. $X \cap Y$ je klijenč od r $N(t) = N(s)$

3. $X \cap Y$ nije pravim stup, nije klijenč od r ili s $A = X \cap Y$

• svaka n-torka iz r se spaja s $N(s) / V(A, s)$

$N(t) = N(r) \cdot N(s) / V(A, s)$

• ako $V(A, s) \neq V(A, r)$

$N(t) = N(r) \times N(s) / \max(V(A, r), V(A, s))$

→ najveći dio broška razrađuje upite odnosno na korak stružnjika u I operaciju

→ takođe odabira redoslijeda spajanja operanade je veličina rezultata

Čitanje n-torka

• table-scan → sljedimo čitaju podatke iz relacije

↳ kada nema 'upotrebljivog' indeksa prema kojim bi se prakticiralo

• index-scan

• key-only index scan → tako su svi podaci koji se odnose do jednog indeksa

• index scan → tako svi potrebni podaci u indeksu, no ne ce pristupati podatkovnim tablicama

Metode spajanja

• nested loop join (spajače ligajte tablju pojedino)

• relacije koje se spajaju se zovu outer i inner table

↳ iz vanjske se da svake n-torce, a iz svake inner table n-torce

↳ se n-torce iz unutarvanje relacije koji zadovoljavaju ugov spajanje

↳ više načina približavanja

• za svaku n-torku iz vanjske

relacije subi mreže po jednoj

pretravati svaku unutarvanju

relaciju

• sljedeci

• indeksi

• anti-index path - optimizator se napravi indeksi

• hash join (raspoređeno spajanje)

↳ faza raspodjeljivanja

↳ faza ispitivanja

EKONERG

- možemo dormati koji plan je SUBP upotrijebio za svaku izdužnu naredbu ako napisemo: **SET EXPLAIN ON**, a gatno se **SET EXPLAIN OFF**
 → upisuju se u datoteku i spremna u tom mrežki folder

način prikaza relacije: **SEQUENTIAL SCAN**

INDEX PATH

AUTO INDEX PATH

DYNAMIC HASH JOIN ⇒ tako uje ovo, radi se o spajanju ugovorenim petljama

ER MODEL BAZE PODATAKA

(Entity-Relationship model)

entitet - bilo što, što ima suštinu li bit, ima jasnoću kao članica ili ideja

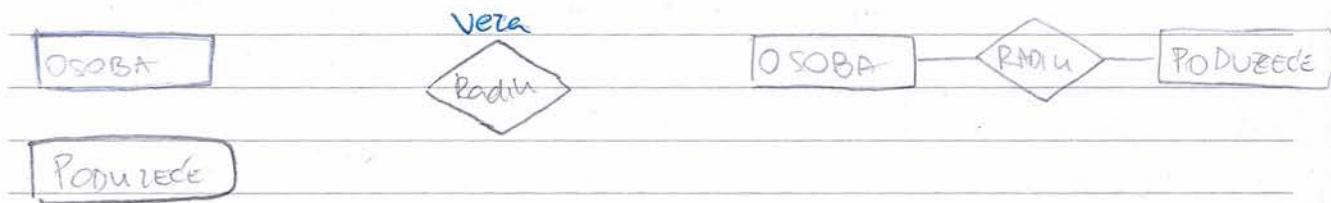
skup entiteta - slični entiteti se grupiraju u skupove

skup veza

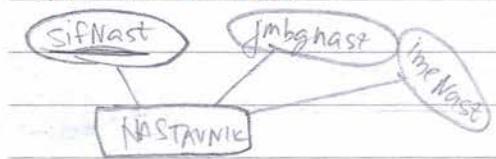
Vrlo - funkcije kojim skup entiteta obavlja u skupu veza

atribut - funkcije koje presljekava iz skupa veza ili skupa entiteta u skup vrijednosti

Entiteti



Atributi entiteta

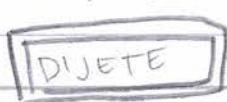


Vlastiti atribut entiteta - atribut koji opisuje

izvješće o entitetu koja se pripisuje, npr. imenu
svojim entitetom, a mukao u vezi s
drugim entitetima

Regularni entiteti - može postojati samo za sebe

Slabi entiteti - ne postoji "kako" ne postoji i neki drugi entitet

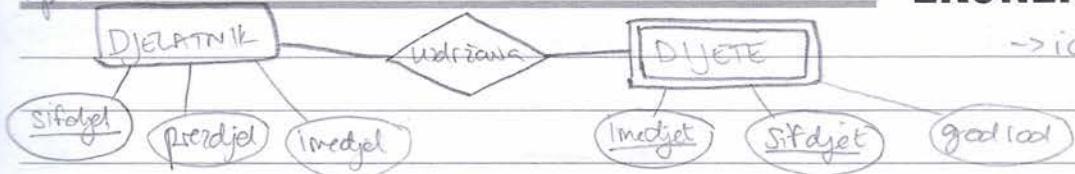


↳ kod određivanja identifikatora im nisu dovoljni vlastiti atributi
 ↳ za identifikaciju se koriste i ključni atributi entiteta ustanovite

EKONERG

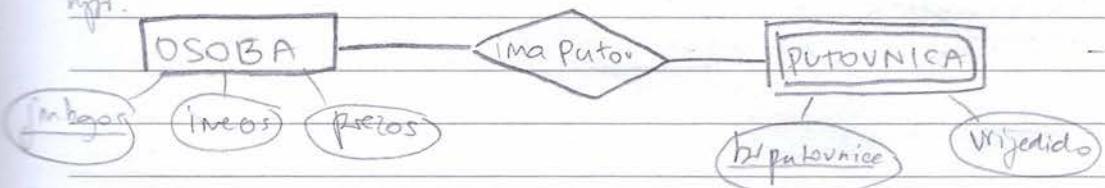
27

npr.



→ identifikacijski
slab entitet

npr.



→ registracijsko
slab entitet

stupanj veze: broj entiteta kojih povezuje datchna veza

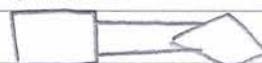
→ može biti unarni (refleksivni), binarni

binarni (refleksivni) - jedan entitet koji u vezi ima dvije različite uloge

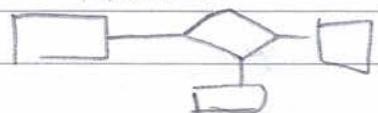
binarni



binarni



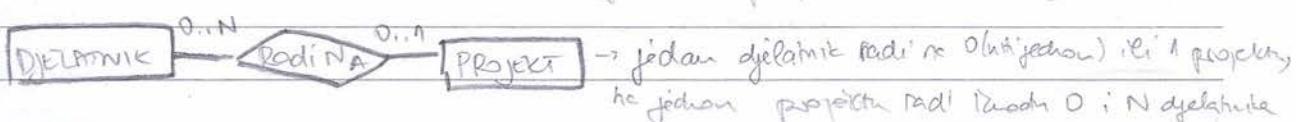
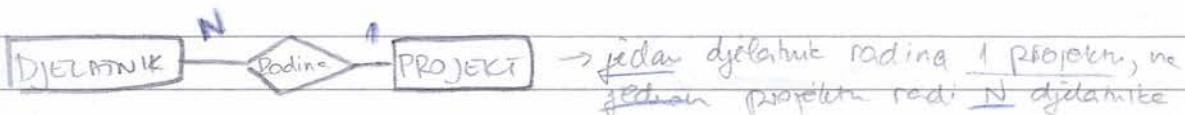
ternarni



spojnost veze → ograničuje preslikavanje pojedinačnih entiteta kojih veza povezuje

→ vrijednost spojnosti: 1, više (N)

korakne 1, N ili nekonvencionalne: npr. 0..1, 1..N, 1..2 itd



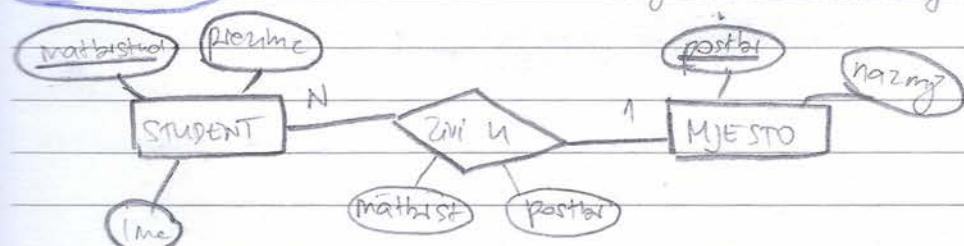
preslikavanje

→ međusobni odnos entiteta u vezi

obod binarnih mogućnosti: 1:1, 1:N, N:1 i N:N

Atributi veze

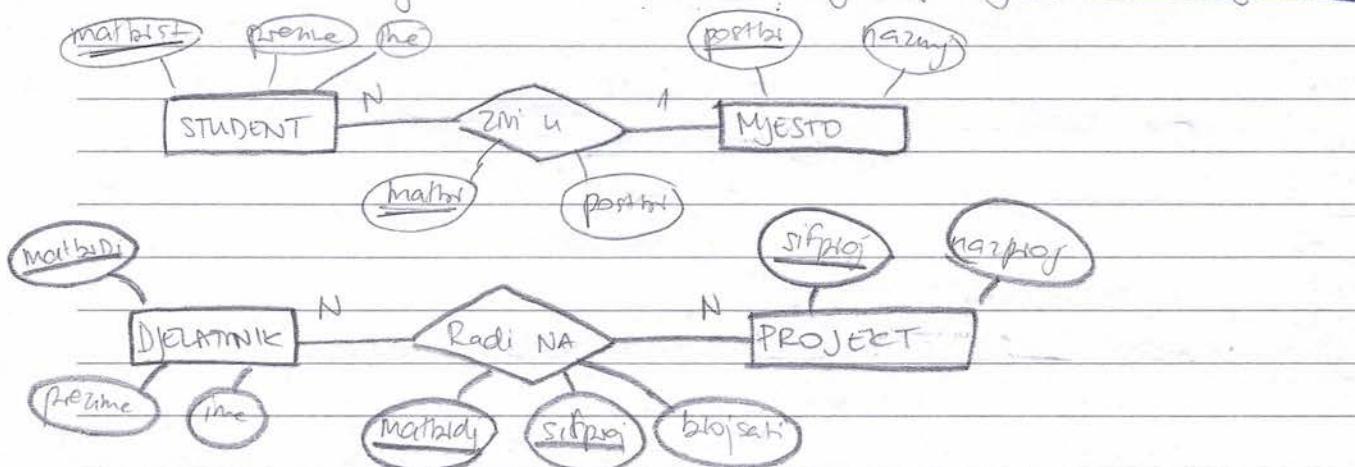
→ shema veze sadrži kognitivne entitete koje povezuju te vlastite atribute



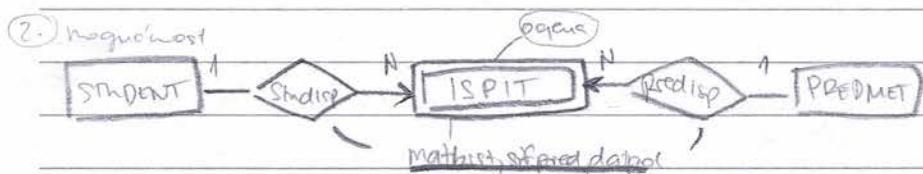
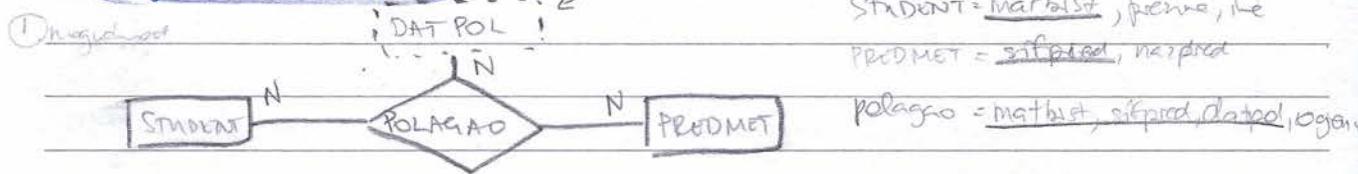
EKONERG

Ključni VERA → definirani pomoću ključnih entiteta koje povezuju i njihovih spojnosti.

def1 → za svaku vrijednost svih entiteta, imaju ipastoji točko 1 vrijednost Ek



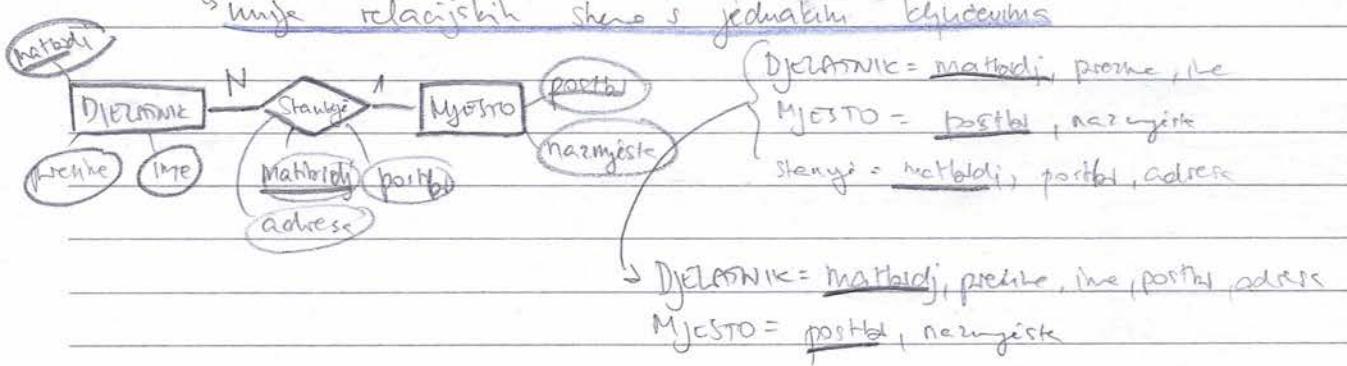
→ degenerirani entitet!



VERA 1:N → preslikavanje u relacijski model

↳ relacijske sheme opisuju entitete (vere postaju entiteti) *

↳ mnoge relacijske sheme s jednatom ključevim

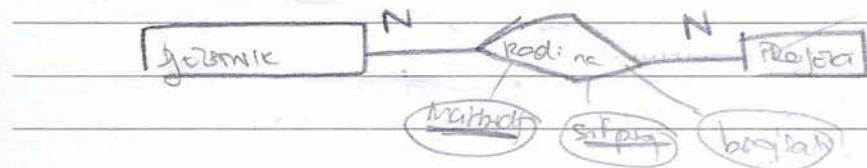
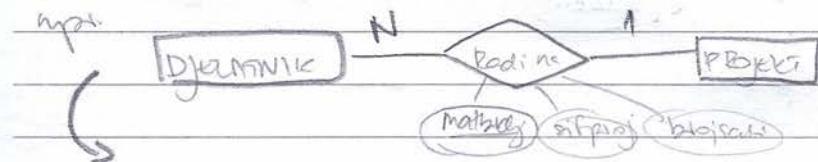


EKONERG

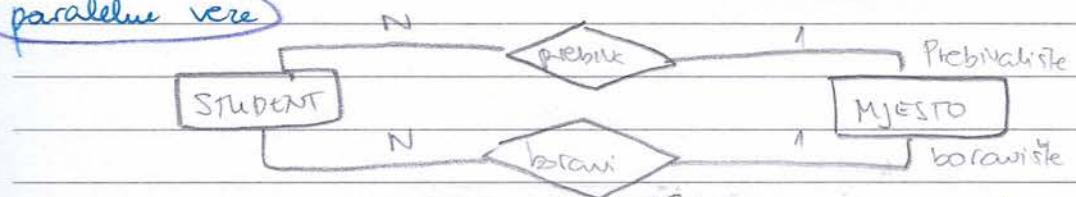
VEZA N:N → relacijske stvari opisuju entitete (veza povezuje entitete) *

→ Vazno je ispravno odrediti vlastite attribute entiteta i veze!

Vlastiti atribut entitete joj atribut koji opisuje značaj o entitetu kjer se napisan je isključivo samom entitetu, a nikako vezi s drugim entitetima npr.



parallelne veze



STUDENT = matematik, fizika, chem, program, plinko + prava integrante

MJESTO = plink, narav

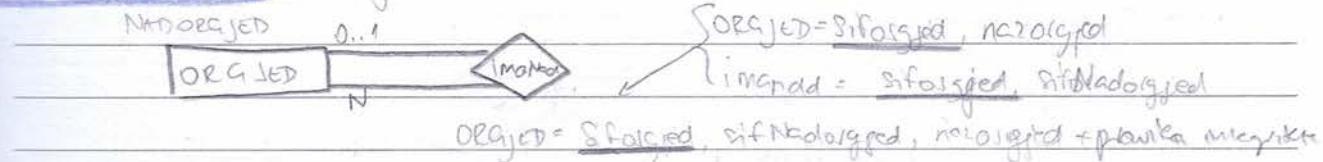
PREBIVALISTE = matematik, fizika

boravi = matematik, plinko

STUDENT = matematik, fizika, chem, program, plinko + prava integrante

MJESTO = plink, narav

Homogeni stablo ⇒ rješavanje refleksivnih veza



→ ŠIFRA ORGANIZARJSKE JEDINICE NE SMJE BITI GOVOREĆA!

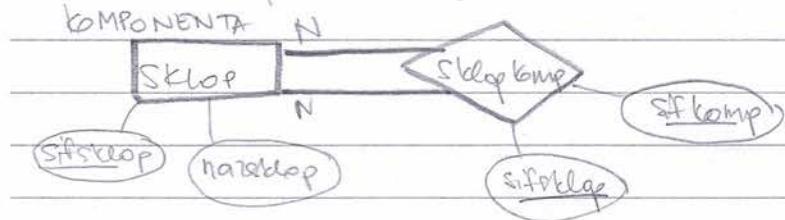
Obliskovanje ER modela:

- ✗ definira entitete
- ✗ definira veze
- ✗ definira attribute entitete
- ✗ definira attribute veze

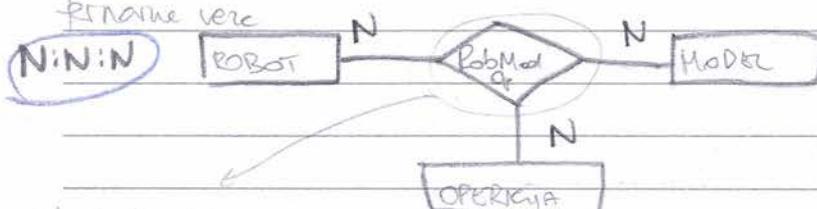
EKONERG

Homogeni mreža

↳ kada je izgrađen svih veća moguće svezki na jednu refleksivnu, samo i preinovljajuću atribute.



kratke verzije

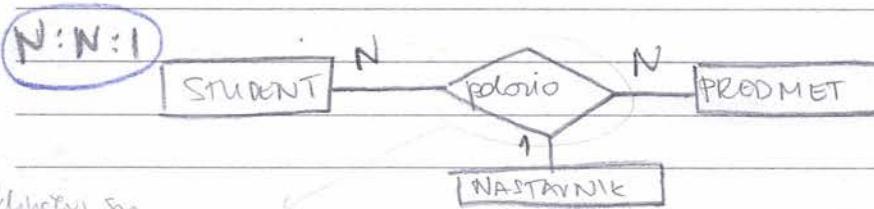


→ N može se računati koliko

bihatom veze bez gubitka
informacija

pri prebacivanju

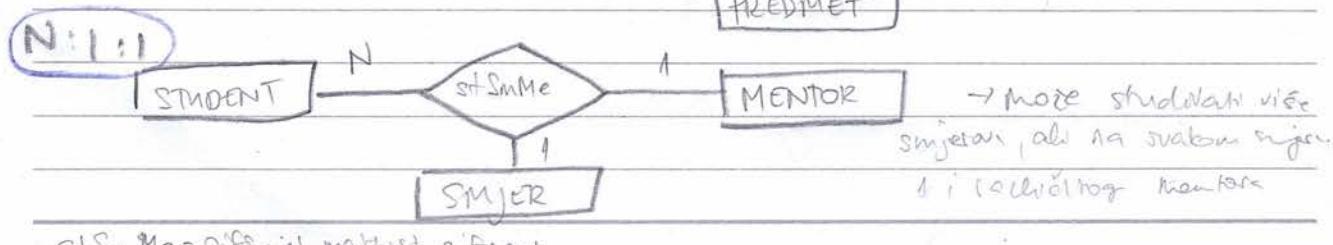
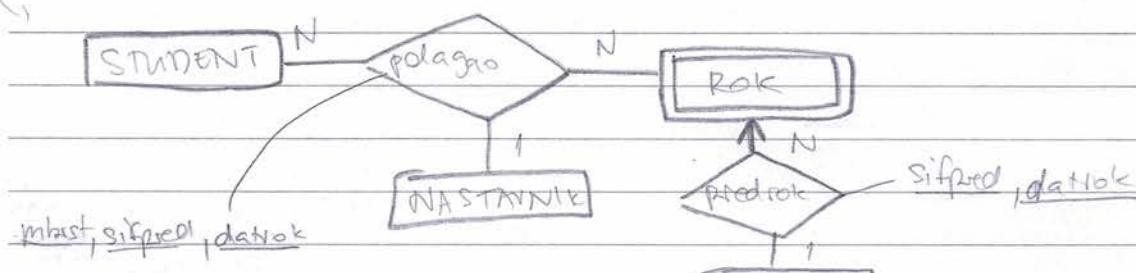
u relacijski model, ostaje kao rezervni entitet!



ključevi su

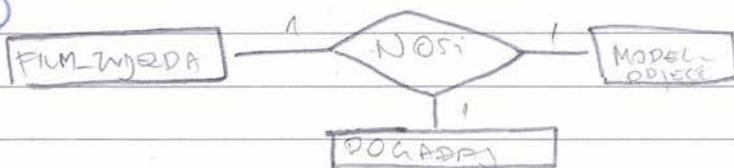
(mathist, sifpred, znanstvenik itd) po kojima bih smo 1 po 1 predmetu i studentu

treba se dodati ključ datpol obuzrom da se isti predmet može više puta polagati



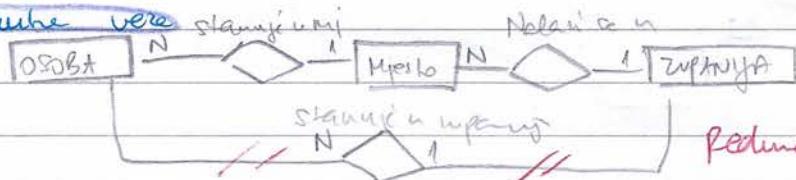
EKONERG

1:1:1

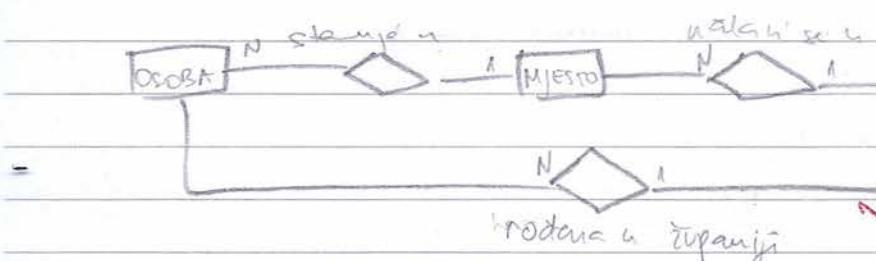


Nosi = sifrig, sfrog, sfmodel

Redundantne veze stampajući

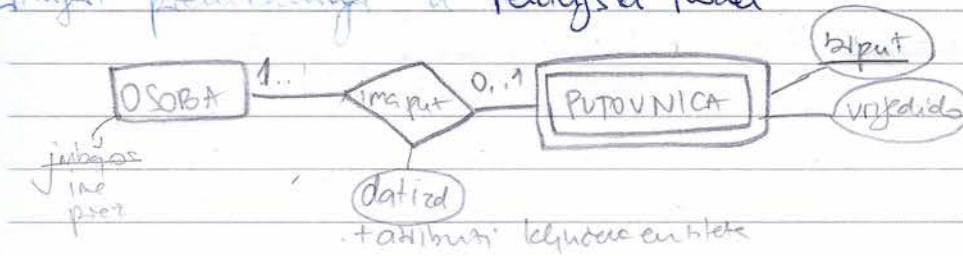


Redundantne veze



nije redundantna veza

primjeri pozitivnoga u relacijski model



```

CREATE TABLE osoba {
  jmbgOs   PRIMARY KEY,
  ime,
  prez,
  datLid
}
  
```

CREATE TABLE putovnica {

izput PRIMARY KEY

,UMjedn. id

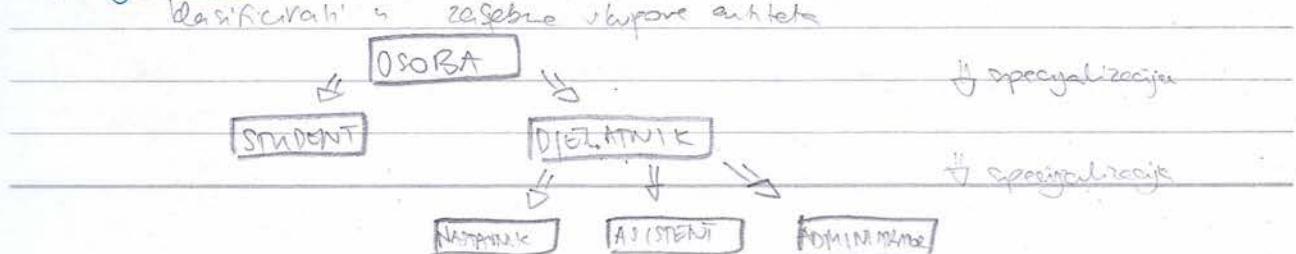
datLid

,jmbgOs NOT NULL

,UNIQUE key (jmbgOs)

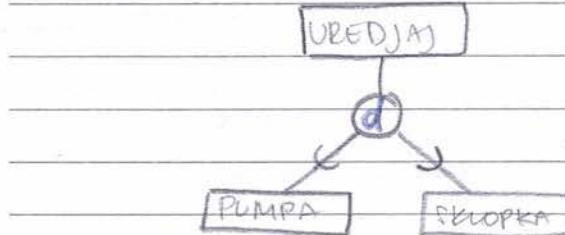
Foreign key (jmbgOs) REFERENCES osoba (jmbgOs);

Specijalizacija → kada se objekt jednog tipa može funkcijama klasifikacije razsebiti u viševe entitete



EKONERG

generalizacija → obimni: sin je od specijalizacije



→ ekskluzivna generalizacija / specijalizacija

→ uređaj može biti ili pumpa ili sklopka

→ specijalizacije nemaju vlastite ključeve

↳ pa one prelaze u jednu tablicu uređaj

→ specijalizacije imaju vlastite ključeve

↳ onda ona gornje specijalizacije i njih ključ shvača kao alternativu

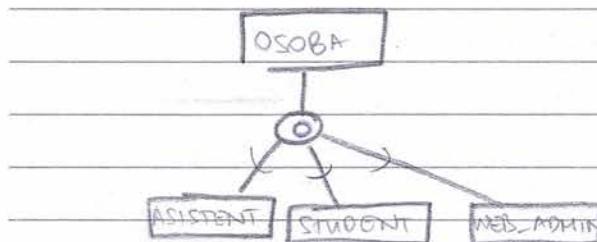
ekskluziva generalizacija / specijalizacija → isključujuće

ekskluzivne



→ tako da jedna osoba može imati više uloga

↳ u krovima se upisuju 0



EKONERG

TRANSAKCIJE I OBNOVA BAZE PODATAKA U SLUČAJU RAZRUŠENJA

SUBP održava od korisnika detalje fizičke pohrane podataka

- omogućuje definiciju i rukovanje s podacima

- obavlja optimizaciju upita

- obavlja funkciju zaštite podataka!

- integritet podataka

- pristup podacima - autorizacija, sigurnost

- osigurava postupak za upravljanje transakcija

◦ obnova u slučaju pogreske ili uništenja baze podataka

- kontrola paralelografskog pristupa

Transakcija

oglednice radnje nad bazu podataka

- početak : BEGIN WORK

- kraj :

- COMMIT WORK - uspješan završetak - potvrđivanje transakcije

- ROLLBACK WORK - neuspješan završetak - poništavanje transakcije i svih izmjena koje je obavila

ako granice nisu eksplizite - definite na redbanku BEGIN/COMMIT/ROLLBACK

tada se granice transakcije određuju implicito - svaka SQL naredba se smatra transakcijom za sebe

Stanje transakcije:

- active → tijekom rada

- partially committed - nakon što je obavljena njena posljednja operacija

- failed - nakon što se ustanovi da nije moguće nastaviti njeno normalno ruciranje

- aborted - nakon što su poništeni njene efekti i baza podataka vracana u stanje prije tv

- committed - uspješno završena

Commit point

- trenutak u kome sve mijene koje je transakcija napravila postaju fiksne

↳ tu se otpuštaju svi ključevi

↳ potvrđena izmjena nikad ne može biti poništava - sustav garantira da će

izmjena biti najuočljivije u bazi podataka, čak i ako bazu

ostane neposredno nakon njezinih potvrđivanja

EKONERG

Efiktive transakcije: A (nedjeljivost, atomičnost.) → mora se obavljati cijeli reči ne uopće

C (konsistentnost) → BP rezultat iz jedne transakcije se ne mijenja u drugu

I (izolacija) → kada se paralelno obavljaju transakcije, učinak mera bi bio da su se sve obavljaju jedna posle druge

D (izdržljivost) → ako je transakcija obavljena npr. posao, ujmani efekti ne smiju biti izgubljeni ako se dogodi kvar sustava

A (Nedjeljivost)

→ korisnik mora biti siguran da je zadani obavljen potpuno i samo jednom

→ tako je dobro da kući tijekom izvođenja transakcije, pouštavajući se u njenu efekti (ako nije bila potvrđena)

D (izdržljivost)

→ bez obzira u kojem trenutku način potvrđivanja dogodio kvar, sustav mora osigurati da su ujmani efekti trajno potvrđeni

II. Obnova baze podataka

→ dovesti bazu podataka u najnovije stanje te logički se ponovno zvući da je radiće ispravno

! Preduzdanja → svaki se podatak mora moći rekonstruirati iz nekoliko drugih informacija. Redundantno - pohrana u nekoj dajdoj u sustavu.

• mirroring - zrcaljenje podataka

• backup - sigurnosne kopije

• logical log - dnevnički izvješće koji slavi za: oponistavanje transakcija

◦ ponovo obavljanje transakcija

Opceniti opis postupka koji omogućuje obnovu

I. periodički kopirajuće sadržaje baze podataka na arhivski medij

II. svake izmjene u bazi se evidentira u logičkom dnevniku izmjena.

◦ stara vrijednost zapise, novu vrijednost

◦ dnevnik, unutar

!!! - izmjena se prvo zapisuje u dnevnik, a tek onda provodi

EKONERG

DATABASE

→ ab je baza potpuno uništena: očitava se najstarija arhivska kopija (ROLLFORWARD)
 o bristeci dnevnik izmjena pokazuje obavljene izmjene koji su se dogodile
 u meduvremenu nakon izrade archive

→ baza nije uništena - sadržaj je nepovredan - program je prekinut tijekom obavljanja
 niza logički povezanih izmjena - potrebno je BP vrati u ispravno stanje
 o pomoći podatku Sadržanju u dnevniku izmjene, poništavaju se sve
 izmjene koji su načinile nezavrsene transakcije

Tipovi pogrešaka:

- transaction failure - posljedica neplaniranog prekida transakcije

- ↳ ponosni dnevnik izmjene poništavaju se efekti transakcije

- system failure - bp je fizički uništenu

- ↳ transakcije koje su se obavljale u trenutku prelaska se na bazu ponosnog
 pokretanje poništavaju

- media failure - bp je fizički uništenu

- ↳ BP se obnavlja ponosni arhivska kopija i pripadajući dnevnik izmjena

→ po završetku sjednice SUBP automatski poništava sve nepovredane transakcije

Nachini zapisivanja spremnika (buffer):

- spremnik dnevnika

- spremnik baze podataka

- sadržaj spremnika se upisuje u dnevnik (BP)

- ↳ kada je spremnik popunjen

- ↳ kada SUBP izda nalog

→ sadržaj spremnika mora biti zapisan u dnevnik na disku prije nego
 što završi procedura potvrđivanja transakcije (Write-ahead log rule)

→ u određenim intervalima (obično svakih 5 minuta) određuju se kontrolne
 točke (checkpoint)

EKONERG

aktivnosti u kontroloj točki

- I. pohrana sadržaja spremnika dnevnika (log buffer) u datotečni dnevnik
- II. zapisivanje zapisa kontrole točke u dat. dnevnika
- III. zapisivanje adresa zapisa kontrole točke iz datoteče dnevnika u restart file
- IV. pohrana sadržaja spremnika BP (database buffer) u bazu podataka

Zapis kontrole točke sadrži: ~~listu svih aktivnih transakcija~~

~~za svaku transakciju - adresu najkrajnjeg zapisa u datoteci dnevnika~~

process obnove

- ↳ stvara se lista za ponistavanje i lista za ponovo obavljanje
- SUBP ne može prihvatići niti jedan zahtev dok se ne izvrši process obnove!

mehanizmi obnove ~~zrealjenje~~

- on-line backup
- incremental backup

KONTROLA ISTODOBNOG PRISTUPA

↳ rezultat transakcije ne smije ovisiti o tome odvijaju li se istodobno i neke druge transakcije!

serializabilan → redoslijed izvršavanja nije serijski ali je učinkovit izvršavanje jednak učinku serijskog, ⇒ BP konzistentnost nije načinac

karakteristični problemi istodobnog čitanja

- P1 pčljavo čitanje → podaci izmjenjeni nepotvrđenim transakcijom
- P2 neporavno čitanje
- P3 sablazne n-torce
- P4 izgubljena izmjena

→ iste transakcije obavljajući istog upita mora dobiti uvjet isti rezultat (osim ako sama nije promijenile podatke čije čitanje ponavlja)

EKONERG

- kontrola istodobnog pisanja
- protokol za snimanje na zaključavanju
- protokol koristi se vremenski okvare
- protokol za snimanje na validaciju
- protokol temeljen na grafovima

- locking manager (dio SPP-a) zaključava zapise i proštuje u skrivenim kada postoji više zahteva za zaključavanjem istog podatka
 - transakcije moraju zaključati podatke i potom spriječiti drugim transakcijama da im pisanje pripada nego ih one otključaju

Vrste zaključavanja

- kљuc za pisanje / izmjenu WRITE LOCK, EXCLUSIVE LOCK

↳ kada T_1 je kљuc, niti jedna druga ne može zaključati dok one ne otključuju

↳ naredbe : INSERT, UPDATE, DELETE

- kљuc za čitanje READ LOCK, SHARED LOCK

↳ kada kљuc druga transakcija može zaključati za čitanje

↳ niti jedna ne može zaključati za pisanje

protokol dvostrukog zaključavanja

1. pije obavljajuće operacije nad objektom transakcije mora za taj objekt zahtijeti kљuc

2. nakon otpuštanja kљuča transakcije ne smije više zahtijevati nikakav kљuc

granulacija podataka

- mreža
- fizička stranica
- relacija
- baza podataka

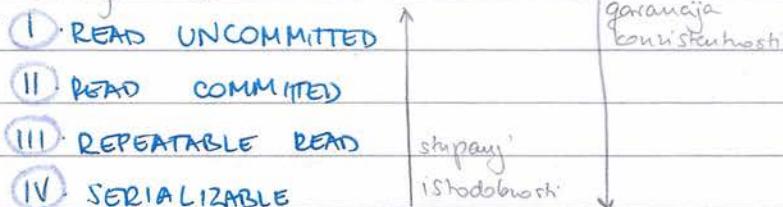
finije

grublje

→ rannu izolaciju je razvijeno da bi se transakcije omogućile balansiranje između konzistentnosti i istovremenosti

EKONERG

razina izolacije SQL-92:



I. READ UNCOMMITTED: podaci se čitaju bez zaključavanja i bez pobjave guli mogućan:
 → mogu se pojaviti n-torce koje nikada nisu potvrđene u bazi podataka

II. READ COMMITTED: čitaju se isključivo potvrđene n-torce

→ pobjavljuje se još treći vrsta podatkov zaključan za pišanje

→ pobjavljuje se stavljajući "konačnog" ključ za čitanje

→ ključ je postavljen samo za vrijeme čitanja podataka!

III. REPEATABLE READ: osigurava ponovljivo čitanje podataka unutar transakcije

→ podatak se zaključava i ostaje zaključan krajem za čitanje do kraja transakcije

→ dalje problem s sačuvanim n-torcama!

IV. SERIALIZABLE: čitaju se zaključavaju krajem za čitanje i ostaje zaključan

do kraja transakcije

→ sprečava bareme probleme!

SQL-92 -	IBM	INFORMIX
	SET ISOLATION TO	SET TRANSACTION ISOLATION LEVEL
I	DIRTY READ	READ UNCOMMITTED
II	COMMITTED READ	READ COMMITTED
III		
IV	REPEATABLE READ	SERIALIZABLE

↑
 traje do kraja sjednice ili
 do nove SET ISOLATION TO
 naredbe

↓
 nije moguće mijenjati unutar iste
 transakcije

SIGURNOST BAZE PODATAKA

- moraju biti definisane pravila koji korisnik ne smije narušiti
- pravila se primenjuju u rješenje podataka
- SUBP nadgleda rad korisnika - osigurava poštovanje pravila
- zaštite na više razina: ona razini SUBP
 - na razini OS-a
 - na razini računalne mreže
 - fizička zaštita
 - zaštita na razini korisnika
- dve modela kontrole pristupa podacima
 - mandatna kontrola pristupa
 - diskrečijska kontrola pristupa
- → dowe su opisane trigom <korisnik, objekt, vrsta operacije>

Korisnici u SQL-u

- korisnik s određenom identifikacijskom ovakom (USER ID)
- obični korisnik (PUBLIC)

Objekti - relacija

- atribut
- virtualna relacija
- baza podataka

Vlasnik objekta

- korisnik koji je stvorio objekt
- vlasnik baze je onaj koji je je stvorio
- vlasnik relacije je vlasnik DB i onaj koji ju je stvorio
- vlasnik objekta dobiva dowe za obavljanje

SVIH niste operacije nad objektom

- dodjeljivanje dowe drugim koliko taj objekat
- uništavanje objekata

u IBM Informixu

- CONNECT - povezni korisnik
- RESOURCE - CONNECT + kreiranje novih relacija
- DBA → korisnik koji je kreirao bazu podataka

EKONERG

Vrste dozvole : · SELECT

· UPDATE

· INSERT

· DELETE

· REFERENCES → koriste se relacije pri def. stranog ključa

· INDEX → kreiranje indexa nad relacijom

· ALTER → izmjenju strukturu relacije i def. integritetnih ogranicenja

· ALL PRIVILEGES

→ naredbe za dodjeljivanje / uklanjanje dozvola : GRANT → ON ... , TO ... , (WITH GRANT OPTION)

REVOKE → ON ... FROM ... (CASCADE RESTRICT)

↓
po defaultu

npr

bpadmin

CREATE DATABASE studbase;

GRANT RESOURCE TO horvat;

GRANT CONNECT TO novak;

horvat

CREATE TABLE zupanija (...)

GRANT SELECT, INSERT, UPDATE

ON zupanija TO novak;

novak

SELECT * FROM zupanija;

INSERT INTO zupanija ...

UPDATE zupanija ...

→ WITH GRANT OPTION → prenosive dozvole

→ tako se ukloni jekao osoba iz baze, uklone se svi user koji su je imali dodjeljeno.

→ RESTRICT → uklanja dozvole do djeli dalje, no to može biti tako
kao da nije već dodijelio dalje dozvole!

npr

CREATE VIEW ispitfinka AS

SELECT * FROM ispit

WHERE kategorija = 'Fizika'

WITH CHECK OPTION;

→ propisava je osoba
odaberite TO ujegoran
je ujegoran u pravilima.
Slobodno

EKONERG

Upotreba Synonima

CREATE PRIVATE SYNONYM userid.imeview^{synonym} FOR imewieworiginal;

↳ može ih shorit samo korisnik i DBA dovoljen

DROP SYNONYM userid.imesynonima;

Dodjelyvanje uloga - rješave problem davanja dozvole velikom broju korisnika

CREATE ROLE uloga;

GRANT SELECT, INSERT, UPDATE, ... ON imetable TO uloga;

GRANT uloga TO userID;

REVOKE uloga FROM userID;

GRANT DELETE ON imetable TO uloga;

Korisnik dozvole dobivenih putem uloga

↳ nakon uspostavljanja sjednice korisnik može dodjeliti dozvole:

I. sve dozvole kao i PUBLIC

II. sve dozvole koje su dodjeljene izmimo njemu

III. sve dozvole nad objektima kojima je vlasnik

IV. dozvole na rangu BP (ako je DBA)

↳ ako namerava koliko dozvole dodjeljuje nekoj ulogu može obaviti naredbom

SET ROLE uloga;

→ korisnik može biti dodjeljena više od jedne uloge, ali u jednom trenutku može koristiti samo jednu od njih!

→ ako ne želi koristiti ništa jednu ulogu SET ROLE NONE

Pričinje rade konflikte

o Audit trail → u datoteci prihvati nad konflikte

↳ ponovo TRIGGER-a

EKONERG

DISTRIBUIRANE I NoSQL BAZE PODATAKA

Distribuirana baza podataka: skup logički povezanih baza podataka raznijestavljenih u različitim okvirima računalne mreže (LAN, MAN, WAN)

DSUBP → programski sustav koji upravlja distribuiranom bazom podataka na takav način da je distribuiranost sustava transparentna prema korisnicima

→ Svaki okvir može direktno ili indirektno komunicirati sa svakim drugim

→ Okvirni distribuiranih DSUBP-a ne dijele zajedničke fizičke komponente (disk, memorija...)

→ Okvirni posrednici određuju stupanj lokalne autonomije

- lokalne aplikacije

- globalne aplikacije

→ baza je distribuirana ako podržava bazu jedne globalne aplikacije

→ funkcionalnosti i tehnike DSUBP, koje su rezultat mnogobrojnih provedenih istraživanja, nisu u cijelosti implementirane niti u jednom (!!!) danas raspoloživom komercijalnom sustavu

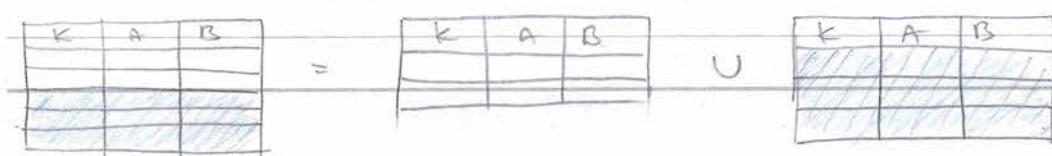
→ oblikovanje distribucije = fragmentacija + lokacija

Shema fragmentacije: o podjeli BP u disjunktni skup fragmenata koji obuhvaćaju sve podatke u bazi podataka uz zadovoljeće pravila da se BP može rekonstruirati iz tih fragmenata

- relacije mogu biti razdjeljene u fragmente horizontalno ili verticalno (ili oba)

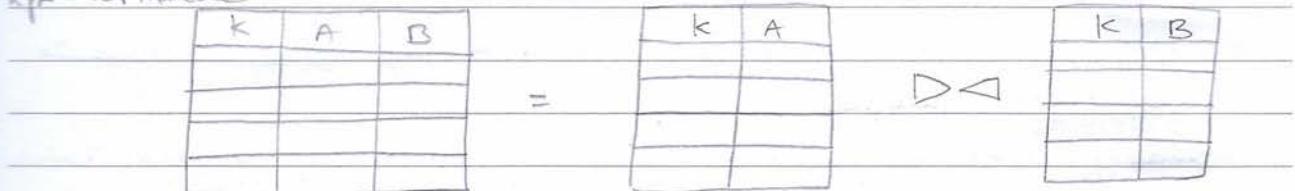
Shema lokacije: shema kojom se opisuju koji je fragment pripadač kojem okviru

npr. horizontal:



EKONERG

upr. vertikalne



upr. horizontalne

$$\begin{array}{c}
 r = r_1 \cup r_2 \\
 \swarrow \quad \searrow \\
 r_1 = r_{11} \Delta r_{12} \quad r_2 = r_{21} \Delta r_{22} \\
 \swarrow \quad \searrow \quad \swarrow \quad \searrow \\
 r_{11} \quad r_{12} \quad r_{21} \quad r_{22}
 \end{array}$$

Alocacija → svaki fragment mora biti alociran u barem jednom čvoru!

o particionirana (ili nereplikirana) BP

↳ svaki od fragmenata alociran je u barem jednom čvoru

o potpuno replicirana BP

↳ svaki od fragmenata alociran je u svim čvorovima

o parcialno replicirana BP

↳ BP nije niti particionirana niti potpuno replicirana (svaki od fragmenata može biti alociran u jednom, više ili svim čvorovima)

Transparentnost podataka

- o transparentnost fragmentacije } Ovisno o transakciji, konačna verzija je u ostalih baza i takođe o tome
- o transparentnost lokacije } su ostalo baze i takođe o tome
- o transparentnost replicacije } daje informacije o BP

Transakcije u DSUBP-u

↳ u svakom čvoru se radi o posebni, potpuno funkcionalan SUBP

↳ globalne transakcije je skup subtransakcija koje koristimo različne SUBP-ovi u više čvorova i pri tome prenose distribuirani našim podatkovima i ujedno u drugo konzistentno stanje

↳ DSUBP provodi protokol asfemog potvrđivanja

⇒ i u sve subtransakcije globalne transakcije obavljene ili nijedna!