Sustavi baza podataka

1. Međuispit - **odabrani zadaci**

3. travnja 2009.

- *odgovore na pitanja 1 - 5 napisati na vlastitim listovima papira*
- *netočni odgovori na pitanja 1 - 5 ne donose negativne bodove*

1. Navedite po čemu bitnom se međusobno razlikuju: nepostojana (*volatile*), postojana (*non-volatile*) i stabilna (*stable*) memorija. Navedite jednu vrstu medija kojim se implementira *off-line* stabilna memorija i jednu vrstu medija kojim se implementira *on-line* stabilna memorija. *(2 boda)*

2. Za B^+-stablo reda *n* vrijedi:
   - najmanja popunjenost korijena je 2
   - najmanja popunjenost internog čvora je $\lceil n / 2 \rceil$
   - najmanja popunjenost lista je $\lceil (n - 1) / 2 \rceil \approx \lceil n / 2 \rceil$

   Izvedite izraz kojim se određuje najveća moguća dubina B^+-stabla reda *n* koje sadrži *m* zapisa podataka. Zašto je najveća moguća dubina B^+-stabla važan podatak? *(2 boda)*

3. Definirajte svojstvo izdržljivosti transakcije (*Durability* iz ACID svojstava transakcije) *(2 boda)*

4. Nacrtajte graf transakcije T_1 koja je opisana sljedećim pseudokôdom:

```
begin work;
   read(x, p1);      -- u varijablu p1 učitaj vrijednost elementa baze podataka x
   p2 ← p1 + 8;      -- u varijablu p2 upiši rezultat operacije p1 + 8
   write(v, p2);     -- u element baze podataka v zapiši vrijednost varijable p2
   p3 ← p1 + 1;
   write(x, p3);
   read(y, p4);
   read(z, p5);
   p6 ← p4 + p5;
   write(u, p6);
commit work;
```

   U graf transakcije ucrtati najmanji mogući broj lûkova: isključivo one lûkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije. Nakon toga napišite dva različita (bilo koja) topološka poretka koji proizlaze iz grafa transakcije kojeg ste nacrtali. *(2 boda)*

5. Za SUBP koji koristi *undo* tehniku obnove vrijede sljedeća važna pravila vođenja dnevnika:

   a) zapis dnevnika <T, x, V>, koji je nastao kao posljedica obavljanja operacije *write(x, V)*, mora se upisati u stabilnu memoriju **prije** nego se obavi operacija *output(x)*.

      Objasnite zašto se ne smije upisati **poslije**.

   b) zapis dnevnika <commit T> transakcije T smije se upisati u stabilnu memoriju tek **nakon** što se obave sve operacije *output* koje pripadaju transakciji T

      Objasnite zašto se ne smije upisati **prije**. *(2 boda)*

Sustavi baza podataka - 2. međuispit
15. svibnja 2009.
odabrana pitanja

- *odgovore na pitanja 1 - 5 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. SUBP koristi striktni 2PL protokol, S i X ključeve. Za detekciju potpunih zastoja koristi se graf čekanja (WFG). Sustav ispituje WFG nakon svake operacije te odmah poništava transakciju koja je izazvala potpuni zastoj. SUBP pokušava izvršiti sljedeću povijest:

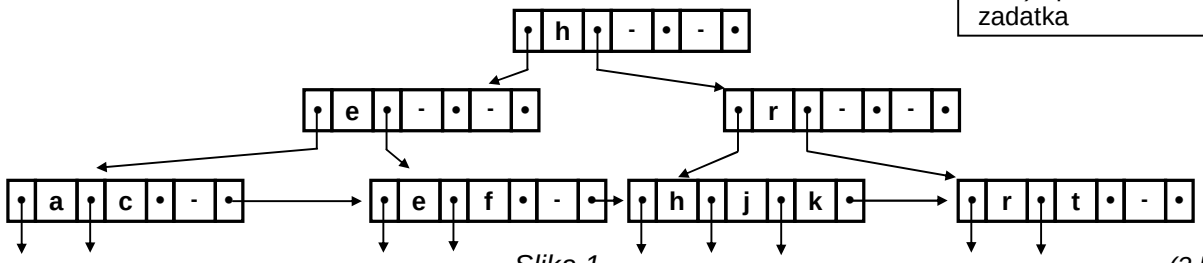
   H: $r_3[v]$, $r_4[z]$, $r_2[x]$, $w_2[y]$, $w_1[x]$, $w_3[y]$, $w_4[v]$, $w_2[z]$, $r_3[z]$, c_3, c_2, c_1, c_4

   a) nacrtati WFG u trenutku kada je nastao potpuni zastoj, te u trenutku neposredno nakon razrješenja potpunog zastoja

   b) napisati povijest H' (također u obliku topološkog poretka) koju će SUBP producirati na temelju povijesti H *(3 boda)*

2. Nacrtati B^+-stablo:
   a) nakon brisanja zapisa s ključem ***e*** iz originalnog B^+-stabla na slici 1.
   b) nakon unosa zapisa s ključem ***p*** u originalno B^+-stablo na slici 1.

   Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka



Slika 1. *(3 boda)*

3. Transakcija T_1 je opisana sljedećim pseudokodom:

```
begin work;
   read(x, p1)
   read(y, p2)
   p3 ← p1 + p2;
   write(x, p3);
commit work;
```

   a) nacrtati graf transakcije T_1: ucrtati isključivo one lúkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije
   b) SUBP koristi striktni 2PL protokol, S i X ključeve. Transakcija T_2 je identična transakciji T_1 (obavlja iste operacije nad istim elementima kao T_1). Nacrtati povijest H koja sadrži T_1 i T_2, koja nije CSR i koja će izazvati ili anomaliju izgubljene izmjene ili anomaliju nekonzistentne analize (odabrati samo jednu od navedenih anomalija i napisati koju ste odabrali!). Nacrtati SG(H). Povijest H napisati u obliku topološkog poretka i označiti ju s H'.
   c) SUBP koristi striktni 2PL protokol, S i X ključeve. Za svaku od ANSI razina izolacija navedite hoće li sustav, ako obje transakcije izvršava uz dotičnu razinu izolacije, producirati povijest koja je identična povijesti H' (tj. neće promijeniti redoslijed operacija u odnosu na onaj koji je naveden u H') *(3 boda)*

4. a) u čemu se razlikuju implementacije temeljnog, striktnog i rigoroznog 2PL protokola?
   b) navedite primjer CSR povijesti koja nije striktna (ST) i objasnite problem koji bi mogao nastati izvršavanjem takve povijesti
   c) zašto se u SQL sustavima za upravljanje bazama podataka koristi rigorozni 2PL protokol?
   d) što je to konzervativni 2PL protokol? *(3 boda)*

5. U sustavu IBM IDS kreirana je relacija osoba (uočiti primarni ključ) i dodatni indeks nad atributom ime. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: relacija → n-torka) i protokol zaključavanja indeksa. Ne koriste se U-ključevi.

sif	ime	prez
100	Ana	Horvat
101	Ivo	Ban
102	Maja	Ban
103	Ivo	Kolar
104	Jure	Novak
105	Ivo	Novak
106	Ana	Turk

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(50)
 , prez   CHAR(50)
 , PRIMARY KEY (sif)
) LOCK MODE ROW;

CREATE INDEX idxIme
   ON osoba (ime);
```

SQL naredbe pristižu u sustav u redoslijedu u skladu s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	T_2	T_3

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
		`BEGIN WORK;`
		`SET LOCK MODE TO WAIT;`
		`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`

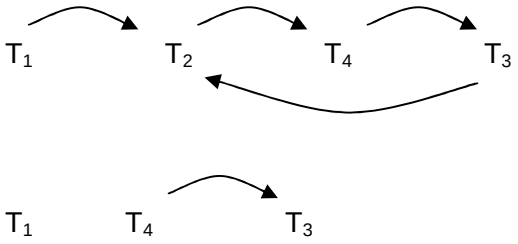
`1. SELECT * FROM osoba WHERE sif = 101;`	`2. SELECT * FROM osoba WHERE sif = 102;`	
`3. UPDATE osoba SET prez='Hil' WHERE sif = 101;`	`4. UPDATE osoba SET prez='Tak' WHERE sif = 102;`	
`5. SELECT * FROM osoba WHERE prez='Novak';`		
		`6. SELECT * FROM osoba WHERE ime = 'Ana';`
	`7. INSERT INTO osoba VALUES(150, 'Ana', 'Ban');`	

Za svaku naredbu (1-7) napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka i da li je zaključavanje uspjelo (s obzirom na već postavljene ključeve). *(3 boda)*

Sustavi baza podataka - 2. međuispit
15. svibnja 2009.

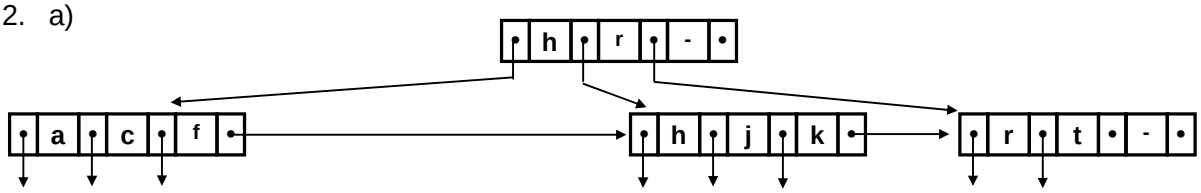
Rješenja odabranih zadataka

1. a)

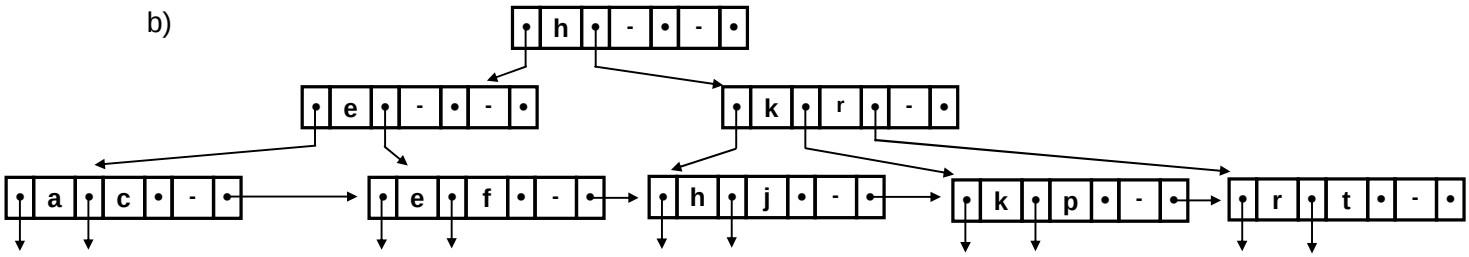


b) H': $r_3[v]$, $r_4[z]$, $r_2[x]$, $w_2[y]$, a_2, $w_1[x]$, $w_3[y]$, $r_3[z]$, c_3, $w_4[v]$, c_1, c_4

2. a)

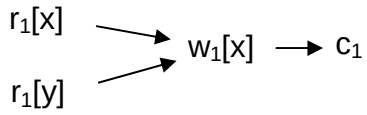


b)



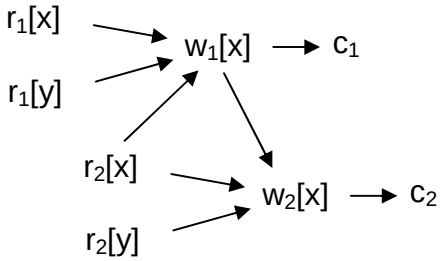
3.

a)

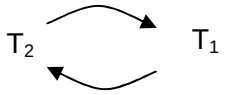


b) Povijest koja bi (ako bi se izvršila) izazvala anomaliju izgubljene izmjene:

H



SG(H)



H': $r_1[x], r_1[y], r_2[x], r_2[y], w_1[x], c_1, w_2[x], c_2$

alternativno H': $r_1[x], r_1[y], r_2[x], r_2[y], w_1[x], w_2[x], c_1, c_2$

c)

za READ UNCOMMITTED: dopušta
za READ COMMITTED: dopušta
za REPEATABLE READ: ne dopušta
za SERIALIZABLE: ne dopušta

alternativno
za READ UNCOMMITTED: ne dopušta
za READ COMMITTED: ne dopušta
za REPEATABLE READ: ne dopušta
za SERIALIZABLE: ne dopušta

4.

a) temeljni: X i S ključevi se mogu otpuštati prije kraja transakcije
striktni: S ključevi se smiju otpustiti čim počne faza sažimanja, a X ključevi tek poslije točke potvrđivanja
rigorozni: S ključevi i X-ključevi se otpuštaju poslije točke potvrđivanja (S može neposredno prije)

b) $r_1[x], w_1[x], w_2[x], a_1, c_2$
poništavanjem T_1 pomoću *undo* gubi se rezultat operacije $w_2[x]$

c) jer bi se inače morala uvesti posebna SQL naredba kojom bi se signaliziralo da transakcija neće tražiti nove ključeve i posebna naredba za otpuštanje S ključeva

d) protokol u kojem se svi ključevi postavljaju na samom početku transakcije u jednom nedjeljivom koraku

5.

1. IS na relaciju osoba - da; S na zapis indeksa sif=101 - da; S na n-torku sif=101 - da
2. IS na relaciju osoba - da; S na zapis indeksa sif=102 - da; S na n-torku sif=102 - da
3. IS $\rightarrow$ IX na relaciju osoba - da; S $\rightarrow$ X na n-torku sif=101 - da
4. IS $\rightarrow$ IX na relaciju osoba - da; S $\rightarrow$ X na n-torku sif=102 - da
5. IX $\rightarrow$ SIX na relaciju osoba - ne - T_1 čeka
6. IS na relaciju osoba - da; S na zapis indeksa ime='Ana' - da; S na n-torke ime='Ana' - da
7. X na zapis indeksa ime='Ana' - ne; - T_2 čeka

Sustavi baza podataka - Završni ispit - Odabrani zadaci
23. lipnja 2009.

- *odgovore na pitanja 1 - 5 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. Zadana je shema baze podataka (potcrtani su primarni ključevi relacija):

knjiga	**autor**	**drzava**
sifknjiga	sifAut	oznDrz
naslov	imePrez	nazDrz
sifAutKnj	oznDrzRod	brojStanM ***(broj stanovnika u milijunima)***

Na raspolaganju su sljedeći statistički podaci iz rječnika podataka (neki su nevažni za zadatak):

```
N(knjiga) = 5 000
N(autor) = 200
N(drzava) = 50
```

```
V(nazDrz, drzava) = 50
V(oznDrzRod, autor) = 25
V(sifAutKnj, knjiga) = 10
V(brojStanM, drzava) = 10
```

```
min(sifAut, autor) = 1
max(sifAut, autor) = 200
min(sifKnjiga, knjiga) = 10 000
max(sifKnjiga, knjiga) = 20 000
```

Izvršava se upit:

```
SELECT *
   FROM knjiga, autor, drzava
   WHERE sifAutKnj = sifAut
     AND oznDrzRod = oznDrz
     AND sifAut = 23
     AND brojStanM <= 50;
```

a) nacrtati inicijalni plan izvršavanja upita (u SELECT naredbi **uočiti operacije Kartezijevog produkta**, a ne spajanja). Redoslijed spajanja relacija u inicijalnom planu mora odgovarati redoslijedu relacija u FROM dijelu SELECT naredbe. U plan izvršavanja ne treba ucrtavati fizičke operatore.

b) nacrtati plan izvršavanja (dovoljno je nacrtati konačni rezultat) nakon provedene heurističke optimizacije. Redoslijed spajanja odrediti na temelju procjene ukupnog broja n-torki u međurezultatima (zanemariti mogući utjecaj veličine n-torke na veličinu međurezultata). U plan izvršavanja ne treba ucrtavati fizičke operatore, ali treba ucrtati procijenjeni broj n-torki međurezultata. *(4 boda)*

2. Transakcija T_1 je opisana pseudokôdom, a transakcija T_2 grafom transakcije:

T_1
```
begin work;
   read(x, p1);
   read(y, p2);
   p3 ← p1 + 7 * p2;
   write(z, p3);
   p4 ← 55;
   write(v, p4);
commit work;
```

T_2 $w_2[x] \longrightarrow w_2[y] \longrightarrow c_2$

a) nacrtati graf transakcije T_1: u graf transakcije T_1 ucrtati isključivo one lúkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije

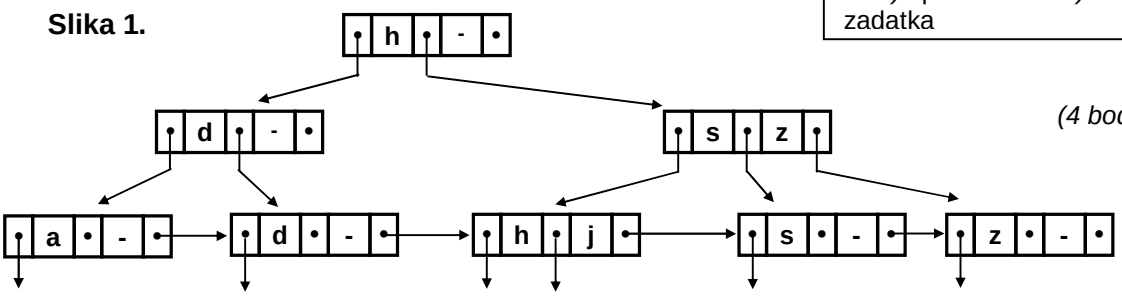
b) SUBP koristi striktni 2PL protokol, S i X ključeve. Nacrtati graf povijesti **H** koja sadrži transakcije T_1 i T_2 i posjeduje sljedeća svojstva: nije CSR, ali bi se uspjela izvršiti uz ANSI SQL razinu izolacije READ UNCOMMITTED, pri čemu bi izazvala anomaliju nekonzistentne analize

c) u obliku topološkog poretka napisati povijest **H'** koju bi SUBP producirao na temelju povijesti **H**, ako bi koristio ANSI SQL razinu izolacije SERIALIZABLE. U **H'** ne treba upisivati operacije postavljanja i otpuštanja ključeva *(4 boda)*

3. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem ***r*** u originalno B^+-stablo na slici 1.
   b) nakon brisanja zapisa s ključem ***d*** iz originalnog B^+-stabla na slici 1.

Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



(4 boda)

4. U sustavu IBM IDS kreirana je relacija osoba (uočiti primarni ključ) i dodatni indeks nad atributom ime. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: relacija → n-torka) i protokol zaključavanja indeksa. Ne koriste se U-ključevi.

sif	ime	prez
100	Ana	Horvat
101	Ivo	Ban
102	Jure	Ban
103	Ivo	Kolar
104	Jure	Novak
105	Ivo	Novak
106	Ana	Turk
107	Edo	Keler

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(50)
 , prez   CHAR(50)
 , PRIMARY KEY (sif)
) LOCK MODE ROW;

CREATE INDEX idxIme
   ON osoba (ime);
```

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	**T_2**

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`

`1. SELECT * FROM osoba WHERE sif = 101;`	`2. SELECT * FROM osoba WHERE ime = 'Jure';`
`3. UPDATE osoba SET prez='Hil' WHERE sif = 101;`	`4. UPDATE osoba SET prez='Tak' WHERE ime = 'Jure';`
`5. DELETE FROM osoba WHERE ime = 'Edo';`	`6. SELECT * FROM osoba WHERE prez = 'Keler';`

Za svaku naredbu (1-6) napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka i da li je zaključavanje uspjelo (s obzirom na već postavljene ključeve). *(4 boda)*

5. U distribuiranom sustavu za upravljanje bazama podataka koristi se distribuirani menadžer zaključavanja (svaki čvor održava vlastiti, lokalni WFG). Transakcija T_1 se inicira u čvoru S_1, a transakcije T_2 i T_3 u čvoru S_2. Neka je:

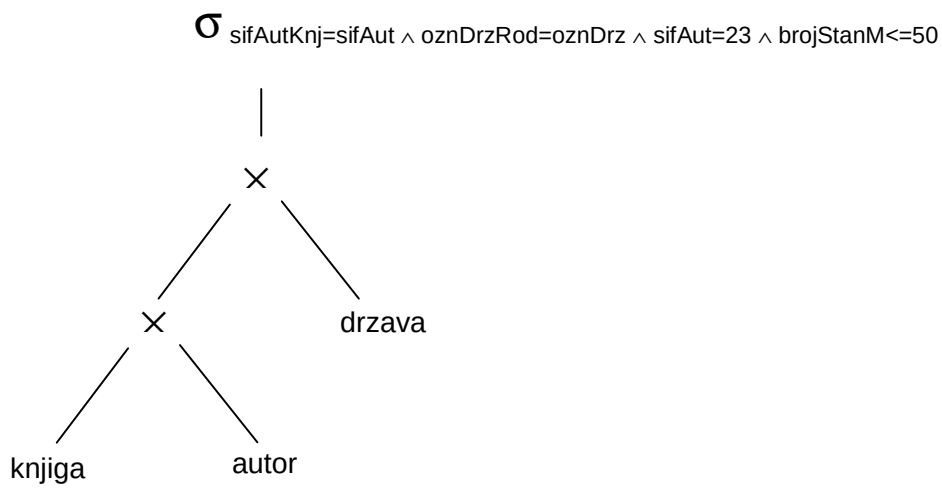
   Shema alokacije S_1: x; S_2: y, z
   Transakcije T_1: $r_1[x]$, $w_1[y]$ T_2: $r_2[y]$, $r_2[z]$, $w_2[x]$ T_3: $w_3[z]$
   Globalna povijest $r_1[x]$, $r_2[y]$, $r_2[z]$, $w_3[z]$, $w_2[x]$, $w_1[y]$

   a) napisati koje se subtransakcije obavljaju u čvorovima S_1 i S_2
   b) nacrtati lokalne grafove čekanja (WFG) u čvorovima S_1 i S_2 u trenutku kada u sustavu nastane potpuni zastoj
   c) nacrtati globalni graf čekanja (WFG) *(4 boda)*

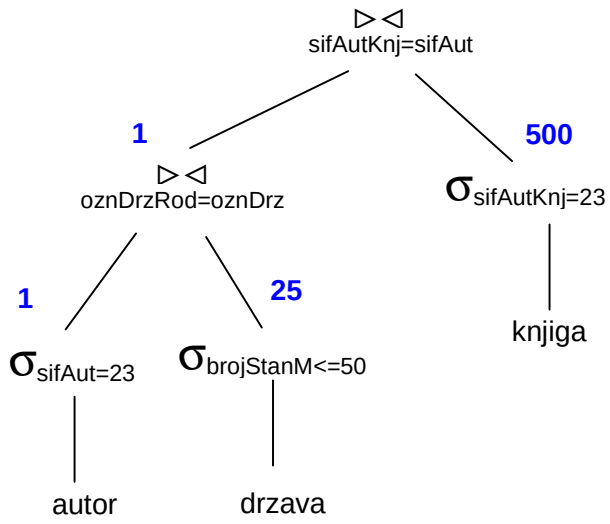
Sustavi baza podataka - Završni ispit
23. lipnja 2009.

Rješenja odabranih zadataka

1. a) Inicijalni plan izvršavanja

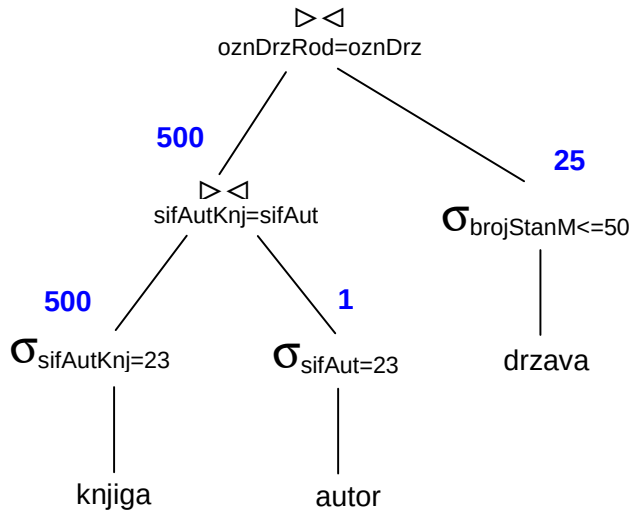


b) Plan izvršavanja nakon heurističke optimizacije



ukupno n-torki u međurezultatima: **527**

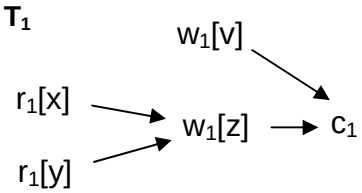
Komentar (**nije dio traženog rješenja**): zašto je odabran redoslijed spajanja prikazan na gornjoj slici?



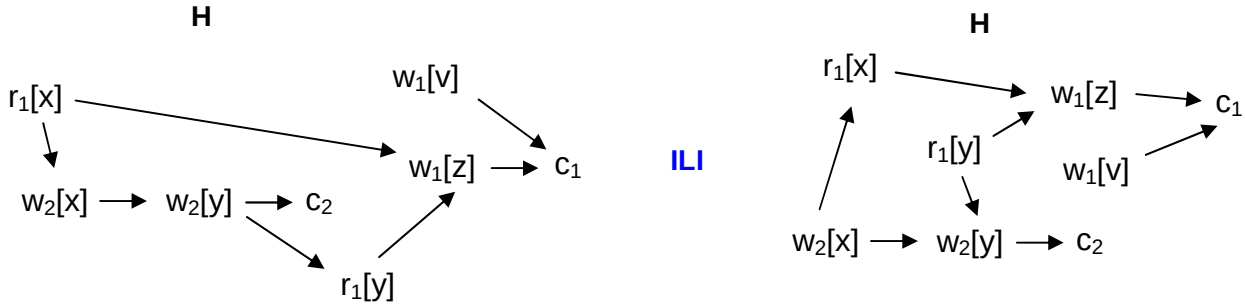
ukupno n-torki u međurezultatima: **1026**

Varijanta u kojoj bi se spajali prvo relacije drzava i knjiga rezultirala bi Kartezijevim produktom (u međurezultatima bi sigurno bilo više od 250 000 n-torki)

2. a)



b) Povijest koja bi (ako bi se izvršila) izazvala anomaliju nekonzistentne analize:



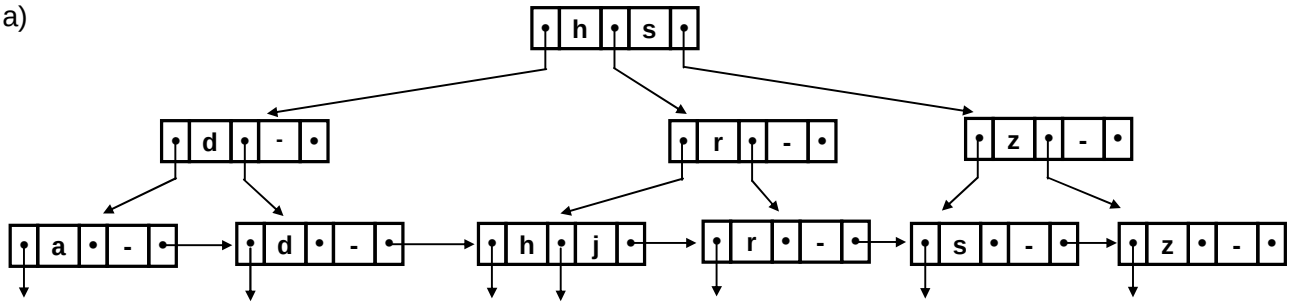
c) Povijest H' koju bi SUBP producirao uz razinu izolacije SERIALIZABLE

H': $r_1[x]$, $r_1[y]$, $w_1[z]$, $w_1[v]$, c_1, $w_2[x]$, $w_2[y]$, c_2

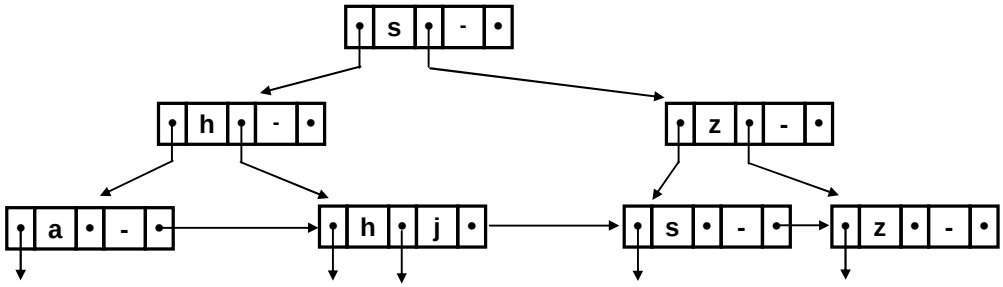
ILI

H': $w_2[x]$, $w_2[y]$, c_2, $r_1[x]$, $r_1[y]$, $w_1[z]$, $w_1[v]$, c_1

3. a)



b)



4. 1. IS na relaciju osoba - da; S na zapis indeksa sif=101 - da; S na n-torku sif=101 - da
2. IS na relaciju osoba - da; S na zapis indeksa ime='Jure' - da; S, S na dvije n-torke ime='Jure' - da
3. IS $\rightarrow$ IX na relaciju osoba - da; S $\rightarrow$ X na n-torku sif=101 - da
4. IS $\rightarrow$ IX na relaciju osoba - da; S $\rightarrow$ X, S $\rightarrow$ X na dvije n-torke ime='Jure' - da
5. IX na relaciji osoba već postoji; X na zapis indeksa ime='Edo' - da; X na n-torku ime='Edo' - da
6. IX $\rightarrow$ SIX na relaciju osoba - ne - T_2 čeka

5. a)

$T_1^1 : r_1[x]$

$T_1^2 : w_1[y]$

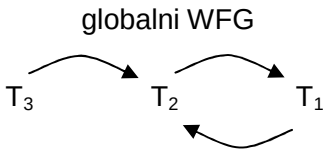
$T_2^1 : w_2[x]$

$T_2^2 : r_2[y], r_2[z]$

$T_3^2 : w_3[z]$

b)

c)



Sustavi baza podataka - 1. međuispit
1 travnja 2010
- odabrani zadaci -

1. Napisati pohranjenu proceduru **`provediIsplatu`** koja prima ulazne parametre: broj računa i iznos. Procedurom se umanjuje **`stanje`** računa za zadani iznos, ali samo ako se time neće prekoračiti dozvoljeno prekoračenje za taj račun. Za provjeru prekoračenja iskoristiti rezultat poziva postojeće funkcije **`dohvatiPrekoracenje`**, koja kao ulazni parametar prima broj računa, a kao rezultat vraća iznos dozvoljenog prekoračenja na računu.

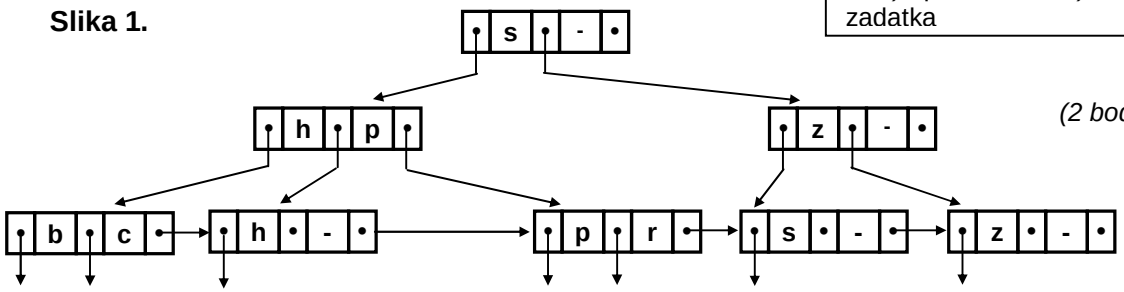
```
CREATE TABLE racun (
  brojRacun INTEGER      NOT NULL
, stanje    DECIMAL(9,2) NOT NULL
, PRIMARY KEY (brojRacun)
);
```

   Ukoliko bi provođenje isplate uzrokovalo nedozvoljeno prekoračenje na računu, u pozivajući program vratiti pogrešku s tekstom **`'Nema dovoljno sredstava na računu'`**. Ukoliko se pri izmjeni podataka u relaciji **`racun`** dogodi jedna od ISAM pogrešaka s brojevima **`-107`** ili **`-113`**, u pozivajući program treba vratiti pogrešku s tekstom 'Zapis je zaključan'. U slučaju pojave bilo koje druge pogreške, u pozivajući program vratiti originalnu pogrešku. *(2 boda)*

2. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **g** u originalno B^+-stablo na slici 1.
   b) nakon brisanja zapisa s ključem **s** iz originalnog B^+-stabla na slici 1.

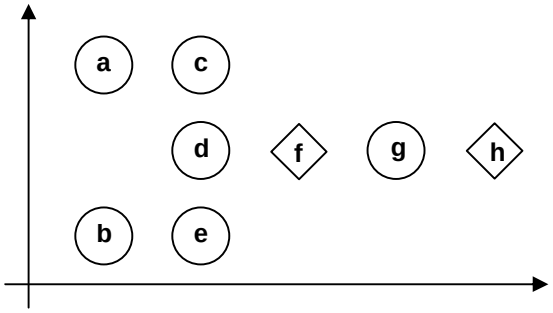
Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



(2 boda)

3. Nacrtati R-stablo **<u>najveće moguće dubine</u>** za prostorne podatke (objekte, odnosno geometrijske likove a, b, c, ...) prikazane na slici 2.



Slika 2.

   Najveći dopušteni broj zapisa (ključ+kazaljka) u listu i internom čvoru R-stabla je 3. Najmanji broj zapisa je 2. Rješenje treba sadržavati ukupno 3 slike. Na prvoj slici nacrtati samo objekte i njihove MBR-ove (označiti ih s R_1, R_2, ...). Na drugoj slici nacrtati MBR-ove sa svih razina (ali označiti samo one koji nisu prikazani na prvoj slici). Na trećoj slici nacrtati R-stablo. *(2 boda)*

4. a) Slikom prikazati postupak sortiranja relacije (datoteke) prikazane na slici 3. metodom vanjskog sortiranja s uparivanjem (*external sort-merge*). Relaciju sortirati prema prvom atributu. Broj raspoloživih blokova glavne memorije M = 3. Broj n-torki (zapisa) u bloku je 2. Označiti faze i napisati što se obavlja u kojoj fazi, te na koji se način u kojoj fazi koriste raspoloživi blokovi glavne memorije (odnosno međuspremnici).

   b) Uz pretpostavku da datoteka ima B(r) blokova, da je broj raspoloživih blokova glavne memorije M, izvesti izraz kojim se određuje broj U/I operacija potrebnih za sortiranje datoteke metodom vanjskog sortiranja s uparivanjem u slučaju kada je zbog $\lceil B(r) / M \rceil \geq M$ sortiranje potrebno provesti u više koraka. *(2 boda)*

   Napomena: a) i b) dio zadatka rješavati kao potpuno nezavisne zadatke

b	1
g	2
k	3
d	4
f	5
n	6
a	7
m	8
c	9
h	10
e	11

Slika 3.

5. Zadane su relacije r(<u>A</u>, B) i s(<u>C</u>, D, E, F). Primarni ključevi su potcrtani. Nema indeksa. Za spajanje se koristi *block-nested-loop-join*. Na raspolaganju su 1002 bloka primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Koristeći priložene podatke, procijeniti ukupni broj **<u>U/I operacija</u>** tijekom izvršavanja operacije, te **<u>broj n-torki</u>** u konačnom rezultatu operacije

$$r \underset{B=D}{\triangleright\!\triangleleft} \sigma_{E>2000 \wedge F=15}(s)$$

   U rješenju prikazati postupak (ne samo konačni rezultat). *(2 boda)*

```
V(B, r) = 1000
V(D, s) = 500
V(E, s) = 20
V(F, s) = 8
```

```
min(B, r) = 0
max(B, r) = 2000
min(D, s) = 0
max(D, s) = 8000
min(E, s) = 0
max(E, s) = 10000
min(F, s) = 0
max(F, s) = 5000
```

```
N(r) = 5000
N(s) = 20000
```

```
B(r) = 5000
B(s) = 8000
```

```
velicina n-torke(r) = 0.5 blokova
velicina n-torke(s) = 0.2 blokova
```

Sustavi baza podataka - 1. međuispit

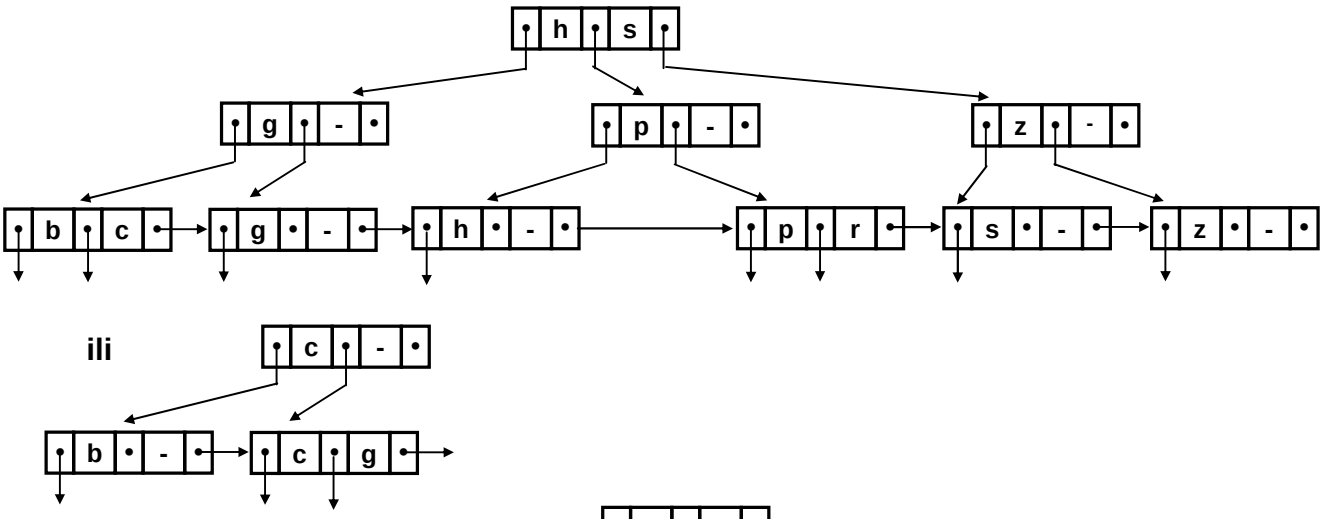
1. travnja 2010.

rješenja odabranih zadataka

1.

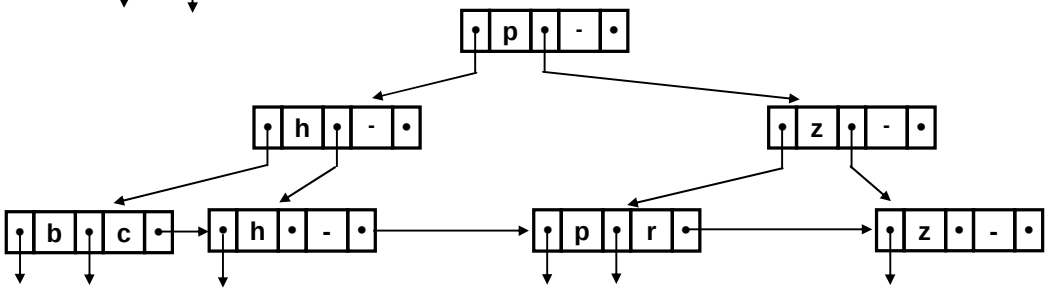
```
CREATE PROCEDURE provediIsplatu (p_brojRacun LIKE racun.brojRacun,
                                 p_iznosTransakcija LIKE racun.stanje)

  DEFINE p_stanjeRacuna LIKE racun.stanje;

  LET p_stanjeRacuna = (SELECT stanje FROM racun WHERE brojRacun = p_brojRacun);
  IF (p_stanjeRacuna - p_iznosTransakcija) < - dohvatiPrekoracenje(p_brojRacun) THEN
     RAISE EXCEPTION -746, 0, ' Nema dovoljno sredstava na računu';
  END IF;
  BEGIN
    ON EXCEPTION IN (-107, -113)
      RAISE EXCEPTION -746, 0, 'Zapis je zaključan';
    END EXCEPTION
    UPDATE racun SET stanje = stanje - p_iznosTransakcija
     WHERE brojRacun = p_brojRacun;
  END

END PROCEDURE;
```

2.a)

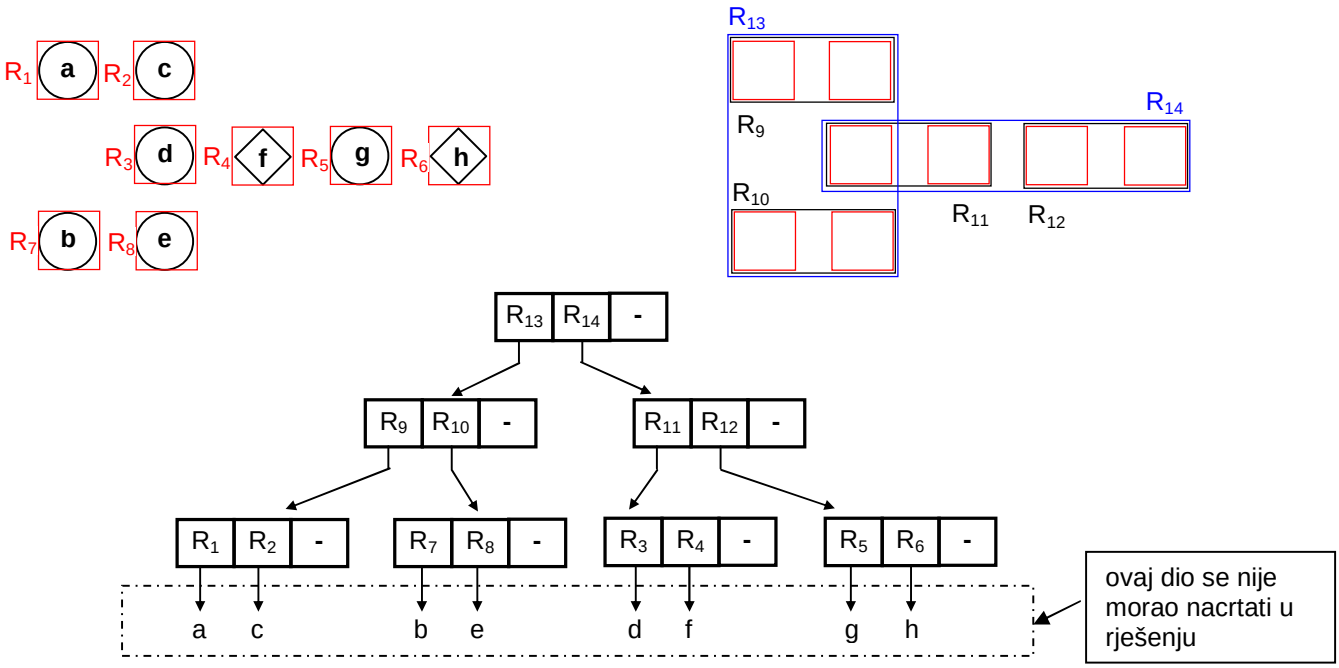


ili

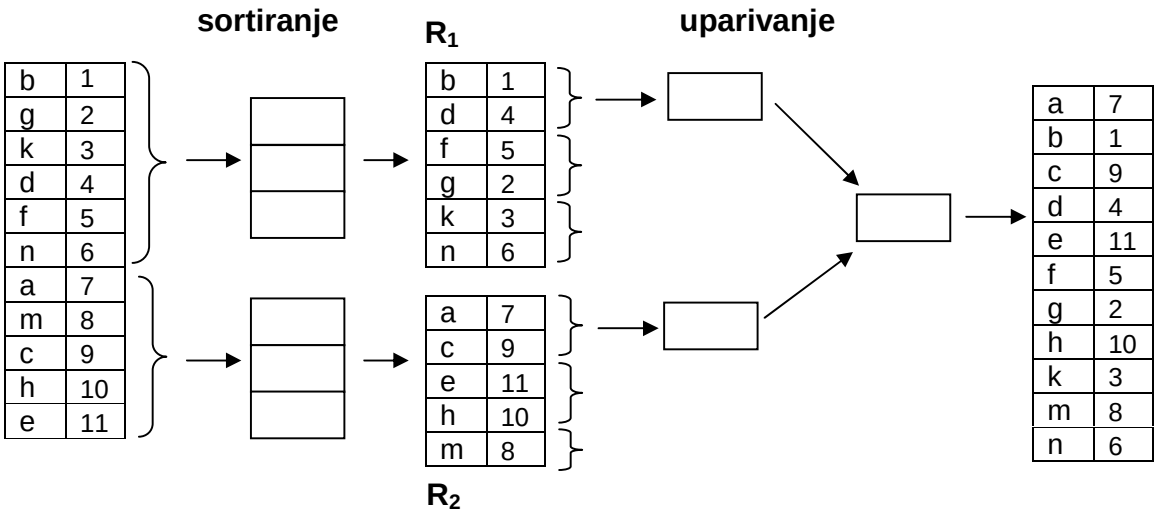
b)



3. (prikazano je jedno od mogućih rješenja)



4.a)



U fazi sortiranja odjednom se čita M blokova, dakle 6 n-torki. Učitane n-torke se sortiraju i zapisuju u segment (*runfile*) R_i. Postupak se ponavlja dok se ne iscrpi cijela ulazna datoteka. Rezultat je N=2 segmenata. Budući da je N < M, uparivanje se može provesti u samo jednom koraku. Jedan blok međuspremnika koristi se za čitanje R_1, jedan za R_2, a treći blok međuspremnika se koristi za pisanje rezultata.

5. Broj n-torki

- $t_1 = \sigma_{E>2000 \wedge F=15}(s)$
- $t_2 = r \underset{B=D}{\triangleright\triangleleft} t_1$
- $f(E \leq 2000, s) = (2000 - \min(E, s)) / (\max(E, s) - \min(E, s)) = 0.2$
- $f(E > 2000, s) = 1 - f(E \leq 2000, s) = 0.8$
- $f(F = 2000, s) = 1 / V(F, s) = 0.125$
- $f(E \leq 2000 \wedge F = 2000, s) = f(E > 2000, s) \cdot f(F = 2000, s) = 0.1$
- $N(t_1) = N(s) \cdot f(E \leq 2000 \wedge F = 2000, s) = 2000$
- $V(D, t_1) = V(D, s)$
- $N(t_2) = N(r) \cdot N(t_1) / \max(V(B, r), V(D, t_1)) = 10000$

Broj U/I operacija

- čitanje relacije s pri obavljanju $\sigma_{E>2000 \wedge F=15}(s) \Rightarrow B(s) = 8000$
- $B(t_1) = N(t_1) \cdot 0.2 = 400$
- zapisivanje međurezultata $t_1 \Rightarrow 400$
- spajanje r i $t_1 \Rightarrow B(t_1) + B(r) = 5400$ (jer t_1 cijela stane u međuspremnike)
- Ukupno $\Rightarrow 8000 + 400 + 5400 = 13800$

Zato jer nije bilo teško propustiti uočiti da t_1 cijela stane u međuspremnike, na ovom međuispitu se priznavalo i sljedeće **pogrešno** rješenje. Rješenje je pogrešno jer se navedeni izraz ne smije primjenjivati kada jedna od ulaznih tablica (relacija) u cijelosti stane u međuspremnike

- spajanje r i $t_1 \Rightarrow B(t_1) / (M-2) \cdot B(r) + B(t_1) = 2400$
- Ukupno $\Rightarrow 8000 + 400 + 2400 = 10800$

Sustavi baza podataka - 2. međuispit
13. svibnja 2010.

- *odgovore na pitanja 1 - 5 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. Relacija **raspored** sadrži studente koji su već raspoređeni za međuispit, dok relacija **dvorana** sadrži broj preostalih slobodnih mjesta u dvoranama. Za studenta koji još nije raspoređen za međuispit, ne postoji n-torka u relaciji **raspored**.

   Napisati pohranjenu proceduru **rasporediStudenta** koja će jmbag studenta i šifru dvorane, zadane kao ulazne parametre procedure, upisati kao novu n-torku u relaciju **raspored** te smanjiti broj preostalih slobodnih mjesta u odgovarajućoj dvorani. Ako se tijekom izvršavanja procedure utvrdi da je student već bio raspoređen (tj. upisan u relaciju **raspored**), u pozivajući program vratiti pogrešku s tekstom: "Student je već raspoređen". U slučaju bilo koje druge pogreške, u pozivajući program treba dojaviti originalnu pogrešku. Nije potrebno provjeravati postoje li zadana dvorana i zadani student.

   Proceduru napisati tako da samostalno upravlja granicama transakcije, tj. započinje i potvrđuje/poništava transakciju.

```
CREATE TABLE raspored
( jmbag        CHAR(10)
, sifDvorana   INTEGER NOT NULL
, PRIMARY KEY (jmbag)
, FOREIGN KEY (sifDvorana)
    REFERENCES dvorana (sifDvorana))
```

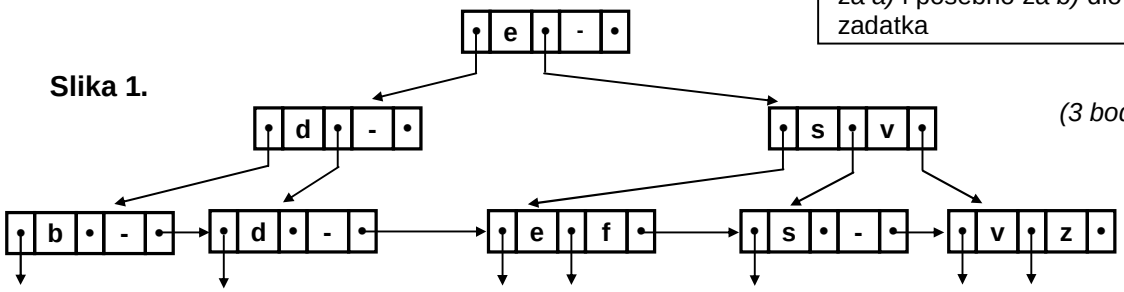
```
CREATE TABLE dvorana
( sifDvorana INTEGER
, slobMjesta SMALLINT NOT NULL
, PRIMARY KEY (sifDvorana))
```

   (3 boda)

2. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **r** u originalno B^+-stablo na slici 1.
   b) nakon brisanja zapisa s ključem **b** iz originalnog B^+-stabla na slici 1.

   Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

   Slika 1.



   (3 boda)

3. Za povijesti H_1, H_2 i H_3 koje su prikazane u obliku topološkog poretka operacija transakcija, odrediti radi li se o RC (*recoverable*) i/ili ACA (*avoids cascading aborts*) i/ili ST (*strict*) povijesti.

   H_1: $r_1[x]$ $w_2[x]$ $r_1[y]$ $w_2[y]$ c_2 c_1
   H_2: $w_2[x]$ $r_1[y]$ $r_1[x]$ c_1 $w_2[y]$ c_2
   H_3: $w_2[x]$ $w_2[y]$ $w_1[y]$ c_2 $r_1[x]$ c_1

   Odgovor se mora napisati u sljedećem obliku (slijedi primjer rješenja, a ne točno rješenje):
   H_1: nije RC, jest ACA, nije ST (***napomena**: nije potrebno objašnjavati zašto)*
   H_2: jest RC, nije ACA, jest ST
   H_3: nije RC, nije ACA, jest ST

   (3 boda)

4. Povijest H je prikazana u obliku topološkog poretka operacija.

   H: $r_2[x]$ $w_1[x]$ $w_2[x]$ $w_2[y]$ c_2 $r_1[y]$ c_1 $w_3[x]$ $r_3[z]$ $w_3[y]$ c_3

   a) Odgovoriti je li povijest H pogled-serijalizabilna (VSR) i objasniti zašto (zašto jest ili zašto nije).
   b) Odgovoriti je li povijest H konflikt-serijalizabilna (CSR) i objasniti zašto (zašto jest ili zašto nije).

   (2 boda)

5. Transakcije T_1 i T_2 su opisane pseudokôdom:

T_1
```
begin work;
   read(x, p1);
   read(y, p2);
   read(z, p3);
   p4 ← p2 + p3;
   write(z, p4);
   display(p1);
commit work;
```

T_2
```
begin work;
   read(x, q1);
   q2 ← q1 + 30;
   write(x, q2);
   write(y, 50);
commit work;
```

a) Nacrtati graf transakcije T_1 i graf transakcije T_2. U grafove transakcija ucrtati najmanji mogući broj lúkova: isključivo one lúkove koji proizlaze iz semantike transakcija ili su nužni da bi se zadovoljila pravila konstrukcije grafova transakcije.

b) U obliku grafa nacrtati povijest H_1 koja sadrži samo transakcije T_1 i T_2. Povijest H_1 mora biti takva da izaziva problem nekonzistentne analize (*inconsistent analysis*). Za povijest H_1 nacrtati serijalizacijski graf $SG(H_1)$. Odgovoriti na temelju čega se može zaključiti da H_1 nije CSR (konflikt-serijalizabilna) povijest.

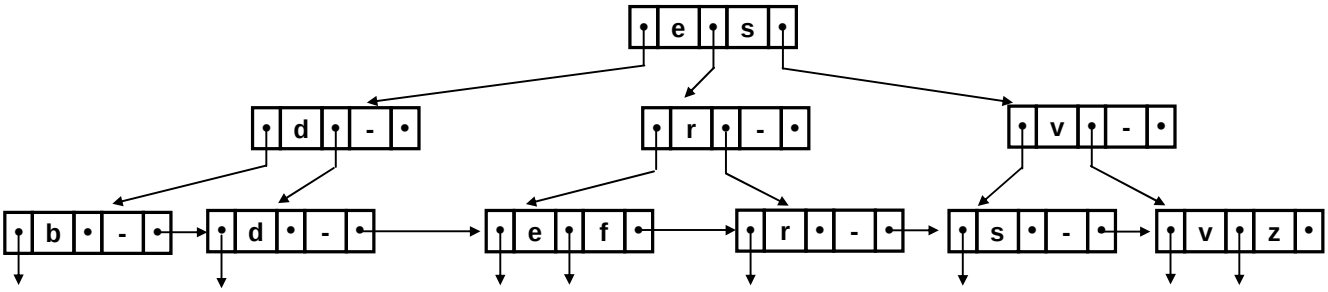
c) U obliku topološkog poretka operacija iz transakcija T_1 i T_2 prikazati povijest H_2 koja sadrži samo transakcije T_1 i T_2 i za koju vrijedi: u slučaju pogreške transakcije T_2 (ili eksplicitnog zahtjeva korisnika za poništavanjem T_2) pojaviti će se problem prljavog čitanja (*dirty read*). *(4 boda)*

Sustavi baza podataka - 2. međuispit
13. svibnja 2010.

Rješenja odabranih zadataka

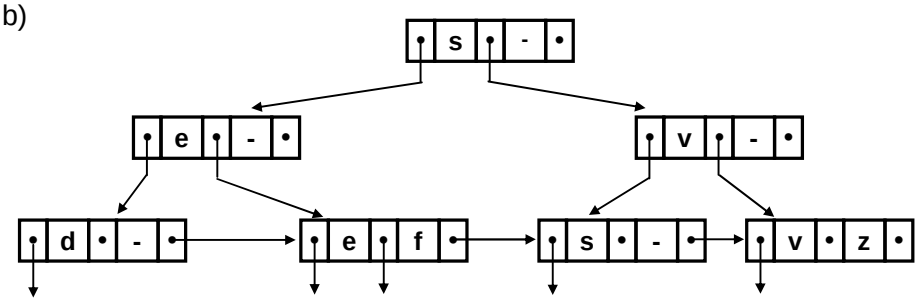
1.
```
CREATE PROCEDURE rasporediStudenta (pMbrStud    LIKE raspored.mbrStud
                                  , pSifDvorana LIKE dvorana.sifDvorana)
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);

   ON EXCEPTION SET sqle, isame, errdata
      ROLLBACK WORK;
      RAISE EXCEPTION sqle, isame, errdata;
   END EXCEPTION

   BEGIN WORK;

   IF EXISTS (SELECT * FROM raspored WHERE mbrStud = pMbrStud) THEN
      RAISE EXCEPTION -746, 0, 'Student je već raspoređen';
    END IF

   UPDATE dvorana SET brojMjesta = brojMjesta-1
     WHERE sifDvorana = pSifDvorana;

   INSERT INTO raspored VALUES (pMbrStud, pSifDvorana);

   COMMIT WORK;
END PROCEDURE;
```

2. a)



U alternativnom rješenju, u listu su zajedno ključevi **f** i **r**.

b)



3. H_1: jest RC, jest ACA, jest ST
   H_2: nije RC, nije ACA, nije ST
   H_3: jest RC, jest ACA, nije ST

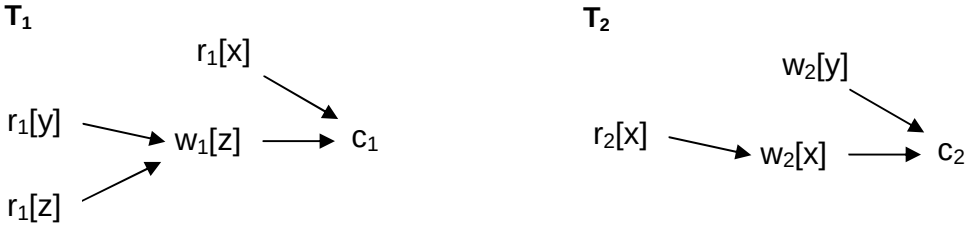
4. a) H je VSR jer je H pogled-ekvivalentna serijskoj povijesti H_s

   H_s: $r_2[x]$ $w_2[x]$ $w_2[y]$ c_2 $w_1[x]$ $r_1[y]$ c_1 $w_3[x]$ $r_3[z]$ $w_3[y]$ c_3

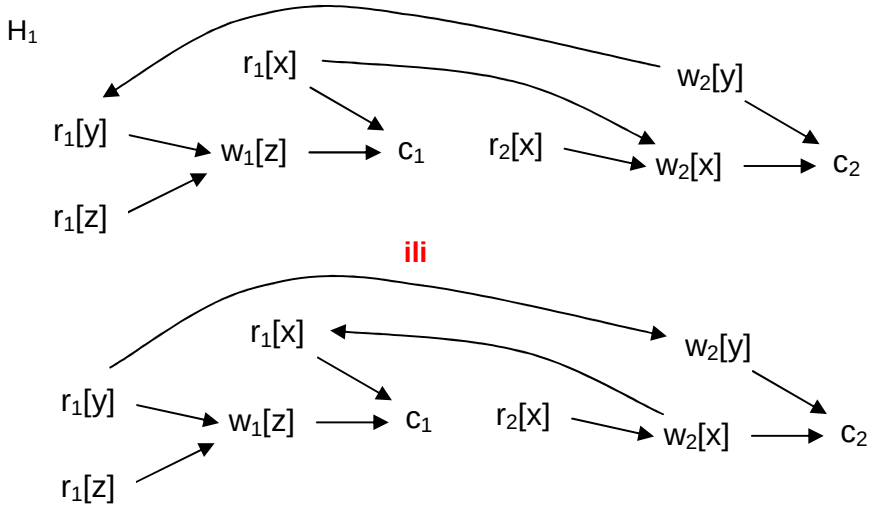
   $H \equiv_V H_S$ jer:
   - po pitanju čitanja inicijalnih vrijednosti
     - za obje povijesti vrijedi: T_2 čita inicijalni x
     - za obje povijesti vrijedi: T_3 čita inicijalni z
   - po pitanju čitanja vrijednosti koje je zapisala neka druga transakcija
     - za obje povijesti vrijedi: T_1 čita y kojeg je zapisala T_2
   - po pitanju završne operacija pisanja
     - za obje povijesti vrijedi: T_3 zapisuje završnu vrijednost za x
     - za obje povijesti vrijedi: T_3 zapisuje završnu vrijednost za y

   b) U H postoje konfliktne operacije: $r_2[x] < w_1[x]$ i $w_1[x] < w_2[x]$. To je dovoljno za zaključak da u SG(H) postoji petlja između T_1 i T_2, stoga H ne može biti CSR.

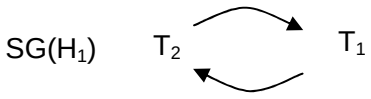
5. a)



   b) Povijest koja bi (ako bi se izvršila) izazvala problem nekonzistentne analize:



   Povijest H_1 nije CSR jer u $SG(H_1)$ postoji petlja



   c) H_2: $r_2[x]$ $w_2[x]$ $w_2[y]$ $r_1[x]$ a_2 $r_1[y]$ $r_1[z]$ $w_1[z]$ c_1

Sustavi baza podataka - Završni ispit - odabrani zadaci i rješenja
29 lipnja 2010

1. Napisati pohranjenu proceduru **`provediIsplatu`** koja prima ulazne parametre: identifikator poslovne transakcije, broj računa i iznos. Procedurom se umanjuje **`stanje`** računa za zadani iznos, a u relaciju **`isplata`** unosi odgovarajući zapis kojim se ta isplata evidentira.

```
CREATE TABLE racun (
  brojRacun INTEGER       NOT NULL
, stanje    DECIMAL(9,2)  NOT NULL
, PRIMARY KEY (brojRacun)
);
```

```
CREATE TABLE isplata (
  idTransakcije INTEGER   NOT NULL
, brojRacun     INTEGER   NOT NULL
, iznos    DECIMAL(9,2)   NOT NULL
, PRIMARY KEY (idTransakcije)
);
```

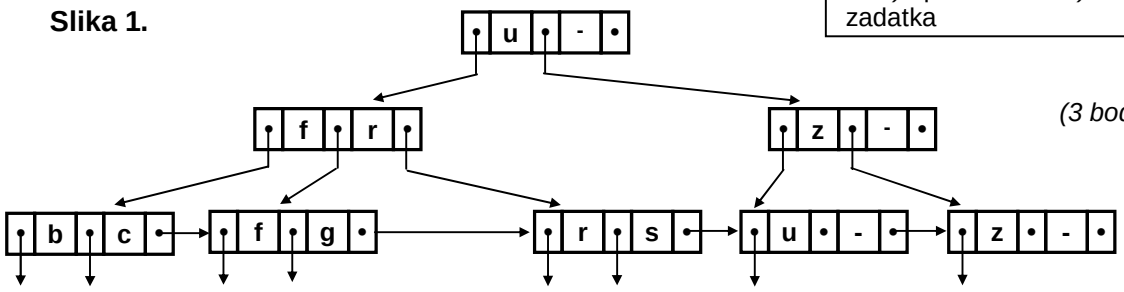
Nije potrebno provjeravati postoji li račun za kojeg se traži isplata. Isplata se ne smije obaviti ako bi stanje računa zbog isplate postalo negativno. U takvom slučaju u pozivajući program vratiti pogrešku s tekstom **`'Nema dovoljno na računu'`**. Ako se pri izmjeni stanja računa dogodi jedna od pogrešaka s brojevima **`-107`** ili **`-113`**, u pozivajući program treba vratiti pogrešku s tekstom **`'Zapis u relaciji račun je zaključan'`**. U slučaju pojave bilo koje druge pogreške, u pozivajući program vratiti originalnu pogrešku.

Proceduru napisati tako da samostalno upravlja granicama transakcije, tj. započinje i potvrđuje/poništava transakciju. U proceduri pomoću prikladne SQL naredbe postaviti razinu izolacije kojom će se osigurati serijalizabilno izvršavanje transakcije. *(4 boda)*

2. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **e** u originalno B^+-stablo na slici 1.
   b) nakon brisanja zapisa s ključem **u** iz originalnog B^+-stabla na slici 1.

Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



(3 boda)

3. Transakcije T_1 i T_2 su opisane istim pseudokôdom:

```
begin work;
   read(x, p1);
   read(y, p2);
   read(z, p3);
   p4 ← p1 - p2;
   write(x, p4);
commit work;
```

   a) nacrtati graf transakcije T_1: u graf transakcije T_1 ucrtati isključivo one lúkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije
   b) nacrtati graf povijesti **H** koja sadrži transakcije T_1 i T_2 i koja nije CSR jer (kad bi se uspjela izvršiti) izaziva anomaliju izgubljene izmjene (*lost update*). Nacrtati pripadni serijalizacijski graf
   c) navesti jedan od mogućih topoloških poredaka koji odgovara povijesti **H** iz b) dijela zadatka, a kojeg bi sustav uspio izvršiti uz ANSI SQL izolaciju COMMITTED READ. Navesti zašto taj isti niz operacija sustav ne bi mogao izvršiti uz razinu izolacije REPEATABLE READ. *(4 boda)*

4. U sustavu IBM IDS kreirana je relacija osoba (uočiti primarni ključ) i dodatni indeks nad atributom ime. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: baza podataka → relacija → n-torka) i protokol zaključavanja indeksa. Ne koriste se U-ključevi.

sif	ime	prez
100	Ana	Horvat
101	Ivo	Ban
102	Jure	Ban
103	Ivo	Kolar
104	Jure	Novak
105	Ivo	Novak
106	Ana	Turk
107	Pero	Keler

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(20)
 , prez   CHAR(20)
 , PRIMARY KEY (sif)
) LOCK MODE ROW;

CREATE INDEX idxIme
   ON osoba (ime);
```

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva, inače čeka, a njene daljnje operacije se ne izvršavaju).

T_1	**T_2**

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`

`1. SELECT * FROM osoba WHERE sif = 103;`	`2. SELECT * FROM osoba WHERE ime = 'Pero';`
`3. UPDATE osoba SET prez='Tak' WHERE sif = 103;`	`4. DELETE FROM osoba WHERE ime = 'Pero';`
`5. SELECT * FROM osoba WHERE ime = 'Pero';`	`6. SELECT * FROM osoba;`

a) za svaku naredbu (1-6) napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka, je li zaključavanje uspjelo (s obzirom na već postavljene ključeve) i je li se pri pokušaju zaključavanja dogodio potpuni zastoj
b) ako bi se naredba za kreiranje relacije osoba promijenila tako da umjesto ROW piše PAGE, bi li to povećalo ili smanjilo konkurentnost izvršavanja transakcija T_1 i T_2? Navesti kako bi to utjecalo na izvršavanje naredbi iz a) dijela zadatka *(5 bodova)*

5. U distribuiranom sustavu za upravljanje bazama podataka koristi se distribuirani menadžer zaključavanja (svaki čvor održava vlastiti, lokalni WFG). Transakcija T_1 se inicira u čvoru S_2, a transakcije T_2, T_3 i T_4 u čvoru S_3. Neka je:

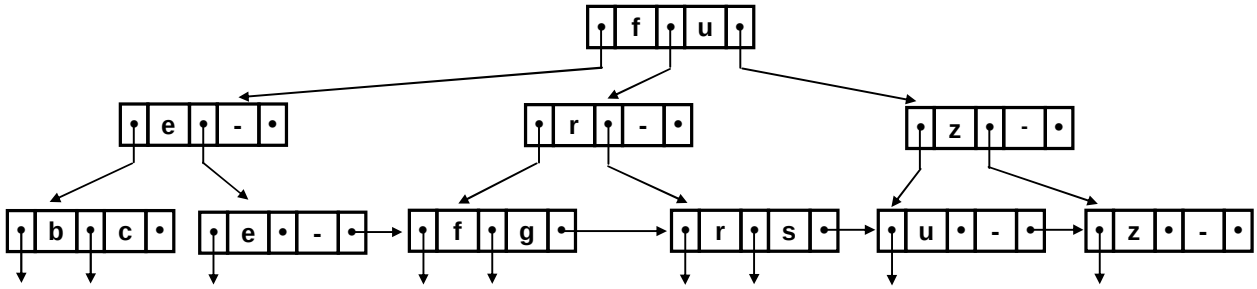
Shema alokacije	S_1: u, v; S_2: x; S_3: y, z
Transakcije	T_1: $r_1[x]$, $r_1[y]$ T_2: $r_2[u]$, $r_2[v]$, $w_2[z]$ T_3: $w_3[y]$, $w_3[v]$ T_4: $w_4[z]$, $w_4[u]$
Globalna povijest	$r_1[x]$, $r_2[u]$, $r_2[v]$, $w_3[y]$, $r_1[y]$, c_1, $w_3[v]$, c_3, $w_4[z]$, $w_2[z]$, c_2, $w_4[u]$, c_4

a) napisati koje se subtransakcije obavljaju u kojim od čvorova S_1, S_2 i S_3
b) nacrtati lokalne grafove čekanja (WFG) u čvorovima S_1, S_2 i S_3 u trenutku kada u sustavu nastane potpuni zastoj
c) nacrtati globalni graf čekanja u trenutku kada u sustavu nastane potpuni zastoj
d) koji se mehanizam u današnjim sustavima najčešće primjenjuje za detektiranje globalnog potpunog zastoja? *(4 boda)*

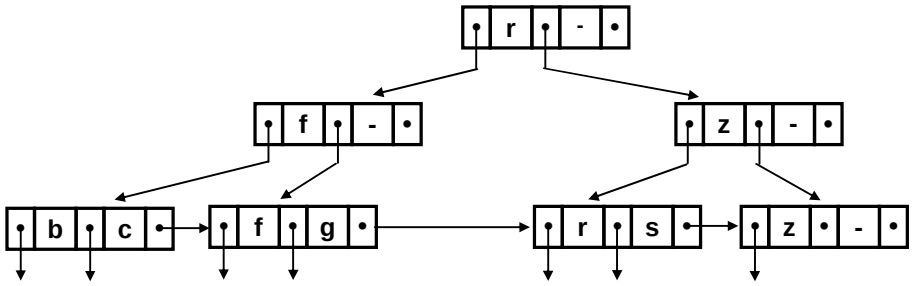
RJEŠENJA

1.
```
CREATE PROCEDURE provediIsplatu (p_idTransakcije LIKE isplata.idTransakcije
                                , p_brojRacun LIKE racun.brojRacun
                                , p_iznos LIKE isplata.iznos)
  DEFINE p_stanjeRacuna LIKE racun.stanje;
  DEFINE sqle, isame INTEGER;
  DEFINE errdata CHAR(80);

  ON EXCEPTION SET sqle, isame, errdata
     ROLLBACK WORK;
     RAISE EXCEPTION sqle, isame, errdata;
  END EXCEPTION

  SET ISOLATION TO REPEATABLE READ;
  BEGIN WORK;
  LET p_stanjeRacuna = (SELECT stanje FROM racun WHERE brojRacun = p_brojRacun);
  IF (p_stanjeRacuna - p_iznos) < 0 THEN
     RAISE EXCEPTION -746, 0, ' Nema dovoljno na računu';
  END IF;
  BEGIN
     ON EXCEPTION IN (-107, -113)
        RAISE EXCEPTION -746, 0, 'Zapis u relaciji račun je zaključan';
     END EXCEPTION
     UPDATE racun SET stanje = stanje - p_iznos
        WHERE brojRacun = p_brojRacun;
  END
  INSERT INTO isplata VALUES (p_idTransakcije, p_brojRacun, p_iznos);
  COMMIT WORK;
END PROCEDURE;
```

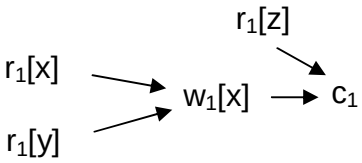
2. a)



b)

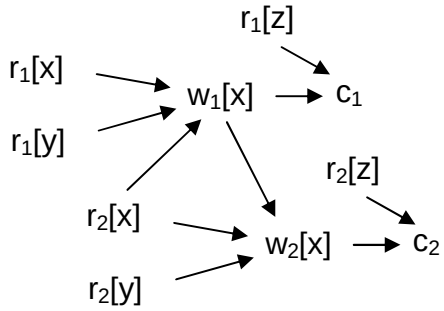


3. a)

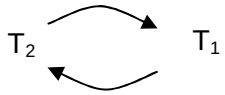


b)

H



SG(H)



c) $r_1[z]$, $r_1[x]$, $r_1[y]$, $r_2[x]$, $r_2[y]$, $w_1[x]$, c_1, $w_2[x]$, $r_2[z]$, c_2

uz REPEATABLE READ, $w_1[x]$ se ne bi mogla izvršiti zbog S-ključa kojeg je postavila operacija $r_2[x]$

4. a)

1. IS na b.p. - da; IS na relaciju osoba - da; S na zapis indeksa sif=103 - da; S na n-torku sif=103 - da
2. IS na b.p. - da; IS na relaciju osoba - da; S na zapis indeksa ime='Pero' - da; S na n-torku ime='Pero' - da
3. IS $\rightarrow$ IX na b.p. - da; IS $\rightarrow$ IX na relaciju osoba - da; S na zapisu indeksa već postoji; S $\rightarrow$ X na n-torku - da
4. IS $\rightarrow$ IX na b.p. - da; IS $\rightarrow$ IX na relaciju osoba - da; S $\rightarrow$ X na zapis indeksa ime='Pero' - da; S $\rightarrow$ X na n-torku - da
5. treba IS, IX na b.p. već postoji; treba IS, IX na relaciji osoba već postoji -da; pokušaj S na zapis indeksa ime='Pero' - ne - T_1 čeka
6. treba IS, IX na b.p. već postoji; IX $\rightarrow$ SIX na relaciju osoba - ne - T_2 čeka - potpuni zastoj

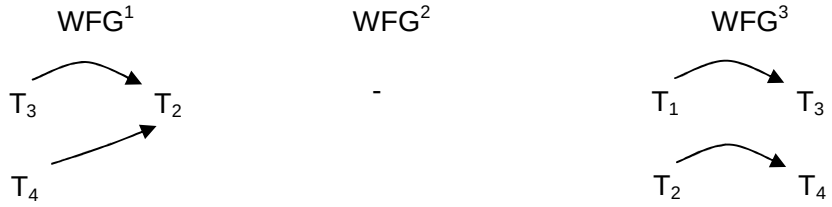
b) Konkurentnost bi se (u općem slučaju) smanjila, jer je povećana granulacija podataka pri zaključavanju. Naredba 1. bi S-ključem umjesto n-torke zaključala stranicu, a time najvjerojatnije sve n-torke koje se nalaze u (ovako maloj) relaciji. Naredba 2. bi također S-ključem umjesto n-torke zaključala stranicu, a time najvjerojatnije sve n-torke koje se nalaze u relaciji. Naredba 3. ne bi mogla postaviti X-ključ na stranicu. T_1 čeka. Naredba 4. ne bi mogla postaviti X-ključ na stranicu.T_2 čeka. Potpuni zastoj se dogodio prije nego u a) dijelu zadatka.

5. a) S_1: $T_2^1 : r_2^1[u]\ r_2^1[v]$ $T_3^1 : w_3^1[v]$ $T_4^1 : w_4^1[u]$

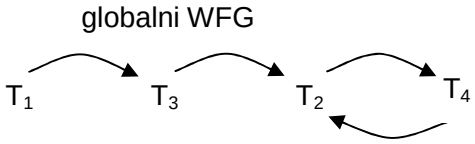
S_2: $T_1^2 : r_1^2[x]$

S_3: $T_1^3 : r_1^3[y]$ $T_2^3 : w_2^3[z]$ $T_3^3 : w_3^3[y]$ $T_4^3 : w_4^3[z]$

b)



c)



d) ograničenjem vremena čekanja na odobravanje zaključavanja (timeout)

Sustavi baza podataka - 1. međuispit
29. ožujka 2011.

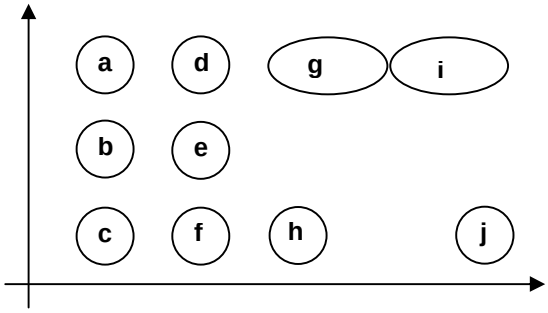
1. **a)** Nacrtati B^+-stablo **reda 4** tako da **ukupni broj čvorova** B-stabla bude **maksimalan**. Stablo treba sadržavati sljedeće vrijednosti ključeva: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

   b) Nacrtati B^+-stablo koje nastane brisanjem zapisa s ključem 3 iz stabla koje je nacrtano u a) dijelu zadatka

Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka. Blokove s podacima nije potrebno crtati

(2 boda)

2. Nacrtati R-stablo za prostorne podatke (objekte, odnosno geometrijske likove a, b, c, ...) koji su prikazani na slici. Stablo nacrtati tako da **ukupni broj čvorova** R-stabla bude **minimalan**.



Najveći dopušteni broj zapisa (ključ+kazaljka) u listu i internom čvoru R-stabla je 3, a najmanji broj zapisa je 2. Rješenje treba sadržavati ukupno 3 slike. Na prvoj slici nacrtati samo objekte i njihove MBR-ove (označiti ih s R_1, R_2, ...). Na drugoj slici nacrtati MBR-ove sa svih razina (ali označiti samo one koji nisu prikazani na prvoj slici). Na trećoj slici nacrtati R-stablo i blokove s podacima na koje pokazuju kazaljke iz listova. *(2 boda)*

3. Relacije *uplataA* i *uplataB* imaju jednake sheme. Napisati pohranjenu proceduru **`prebaciUplate`** koja prima ulazni parametar *potrebanIznos*. Procedurom se n-torke iz relacije *uplataA*, redom prema rednom broju uplate, prebacuju u relaciju *uplataB*. Procedura prebacuje onoliko n-torki koliko je potrebno da bi se prebacio ukupan iznos uplata zadan parametrom *potrebanIznos*. Ne treba provjeravati ima li u relaciji *uplataA* dovoljno n-torki potrebnih za prebacivanje. Kao rezultat, procedura vraća broj prebačenih n-torki. Npr. za relacije iz primjera, ako se procedura pozove s parametrom 200.00, procedura će prebaciti n-torke s rednim brojevima 5, 6, 7 (izbrisati te n-torke iz *uplataA* i upisati ih u *uplataB*) te u pozivajući program vratiti cijeli broj 3.

```
CREATE TABLE uplataA (
  rbrUpl INTEGER      NOT NULL
, iznos  DECIMAL(8,2) NOT NULL
, PRIMARY KEY (rbrUpl)
);
```

uplataA

rbrUpl	iznos
5	100.00
6	50.00
7	120.00
8	100.00
...	

uplataB

rbrUpl	iznos
1	100.00
2	500.00
3	20.00
4	150.00
...	

Ako se pri obavljanju SQL operacije INSERT u relaciji uplataB dogodi bilo koja vrsta pogreške, procedura u pozivajući program vraća pogrešku s tekstom **`'Pogreška unosa u relaciju uplataB'`**. U slučaju pojave bilo koje druge pogreške na bilo kojem drugom mjestu u proceduri, procedura u pozivajući program mora proslijediti originalnu pogrešku. *(3 boda)*

4. Zadane su relacije r(A, B) i s(C, D, E). Primarni ključevi relacija su potcrtani. Nad relacijama nije izgrađen niti jedan indeks. Za operaciju Kartezijevog produkta koristi se fizički operator *block nested-loop join*. Za obavljanje fizičkih operacija na raspolaganju je 300 blokova primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Operacije se izvršavaju redoslijedom kako je navedeno u izrazu relacijske algebre, što znači da ne treba provoditi heurističku optimizaciju. Koristeći priložene podatke, procijeniti broj **U/I operacija** tijekom izvršavanja upita, te **broj n-torki** u konačnom rezultatu. U rješenju prikazati postupak (ne samo konačni rezultat).

$\sigma_{B\neq 2}(r \times \sigma_{D=10 \vee E=15}(s))$

```
V(B, r) = 8
V(D, s) = 5
V(E, s) = 4
```

```
N(r) =  1000
N(s) = 20000
```

```
B(r) =   250
B(s) = 10000
```

```
veličina n-torke(r) = 0.1 blokova
veličina n-torke(s) = 0.2 blokova
```

(3 boda)

5. Zadana je shema baze podataka (potcrtani su primarni ključevi relacija):

```
knjiga
   sifKnj
   sifAutorKnj
   brStrKnj
   format
```

```
clanak
   sifCl
   sifAutorCl
   brStrCl
```

Na raspolaganju su sljedeći podaci iz rječnika podataka:

```
N(knjiga) = 5 000
N(clanak) = 6 000
```

```
V(format, knjiga) = 50
V(sifAutorKnj, knjiga) = 500
V(sifAutorCl, clanak) = 200
```

Izvršava se upit:

```
SELECT DISTINCT sifKnj, sifCl
   FROM knjiga, clanak
   WHERE sifAutorKnj = sifAutorCl
     AND brStrCl > 5
     AND format LIKE '%F'
```

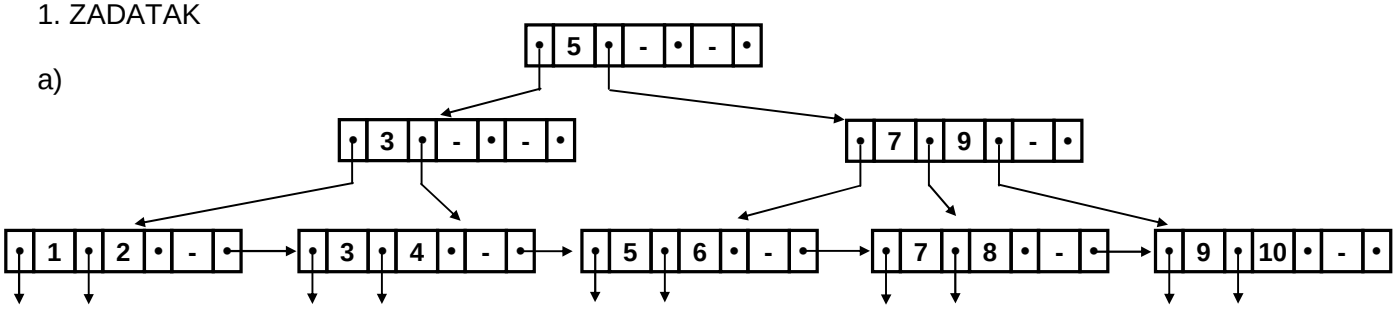
a) nacrtati inicijalni plan izvršavanja upita. U SQL upitu uočiti operaciju **Kartezijevog produkta** (a ne spajanja), te DISTINCT.

b) provesti heurističku optimizaciju. Između inicijalnog plana i završnog plana izvršavanja treba nacrtati barem dva stabla izvršavanja koja prikazuju neku od međufaza optimizacije. U završnom stablu izvršavanja procijeniti broj n-torki u međurezultatima uz navođenje izraza ("formule") prema kojem je procjena obavljena. Fizičke operatore nije potrebno ucrtavati u plan. *(3 boda)*

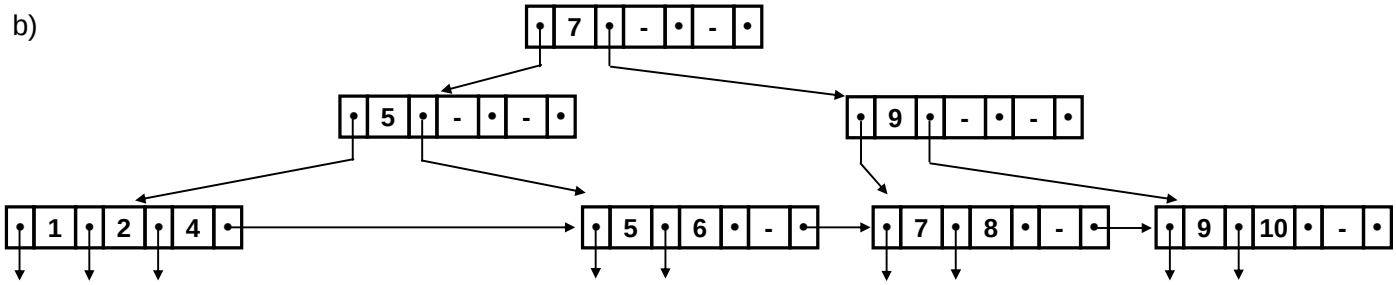
Rješenja odabranih zadataka

1. ZADATAK

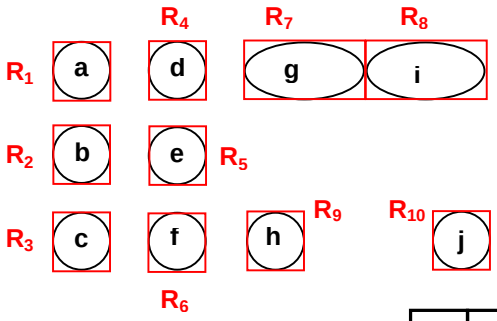
a)

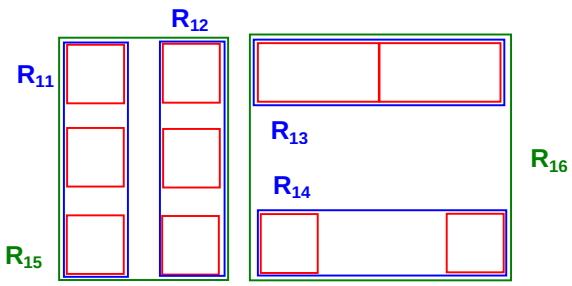


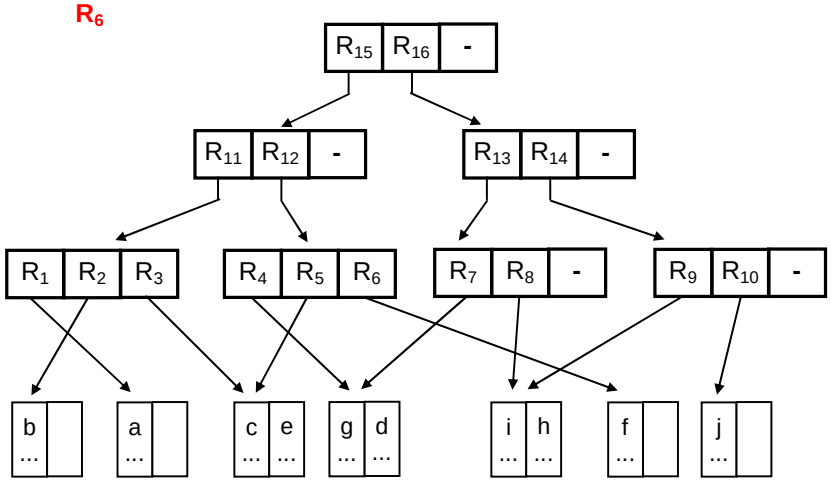
b)



2. ZADATAK







3. ZADATAK

```
CREATE PROCEDURE prebaciUplate(potrebanIznos LIKE uplataA.iznos) RETURNING INTEGER
   DEFINE p_rbrUpl LIKE uplataA.rbrUpl;
   DEFINE p_iznos, prebacenoIznos LIKE uplataA.iznos;
   DEFINE brojPrebacenih INTEGER;
   LET prebacenoIznos = 0.0;
   LET brojPrebacenih = 0;
   FOREACH SELECT * INTO p_rbrUpl, p_iznos
             FROM uplataA
             ORDER BY rbrUpl
      BEGIN
         ON EXCEPTION
             RAISE EXCEPTION -746, 0, 'Pogreška unosa u relaciji uplataB';
         END EXCEPTION
         INSERT INTO uplataB VALUES (p_rbrUpl, p_iznos);
      END
      DELETE FROM uplataA WHERE rbrUpl = p_rbrUpl;
      LET prebacenoIznos = prebacenoIznos + p_iznos;
      LET brojPrebacenih = brojPrebacenih + 1;
      IF prebacenoIznos >= potrebanIznos THEN
         EXIT FOREACH;
      END IF
   END FOREACH
   RETURN brojPrebacenih;
END PROCEDURE;
```

4. ZADATAK

Broj n-torki

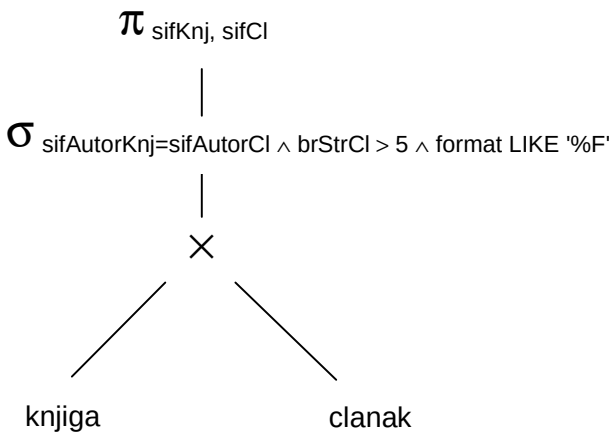
- selektivnost f(D=10, s) = (N(s) / V(D, s)) / N(s) = 0.2
- selektivnost f(E=15, s) = (N(s) / V(E, s)) / N(s) = 0.25
- $t_1 = \sigma_{D=10 \vee E=15}(s)$
- $N(t_1) = N(s) \cdot (1 - (1 - f(D=10, s)) \cdot (1 - f(E=15, s)) = 8\,000$
- $t_2 = r \times t_1$
- $N(t_2) = N(r) \cdot N(t_1) = 8\,000\,000$
- $f(B=2, t_2) = (N(t_2) / V(B, t_2)) / N(t_2) = 0.125$, jer je $V(B, t_2) = V(B, r)$
- $N(\sigma_{B=2}(t_2)) = N(t_2) \cdot f(B=2, t_2) = 1\,000\,000$
- $N(\sigma_{B\neq2}(t_2)) = N(t_2) - N(\sigma_{B=2}(t_2)) = 7\,000\,000$

Broj U/I operacija

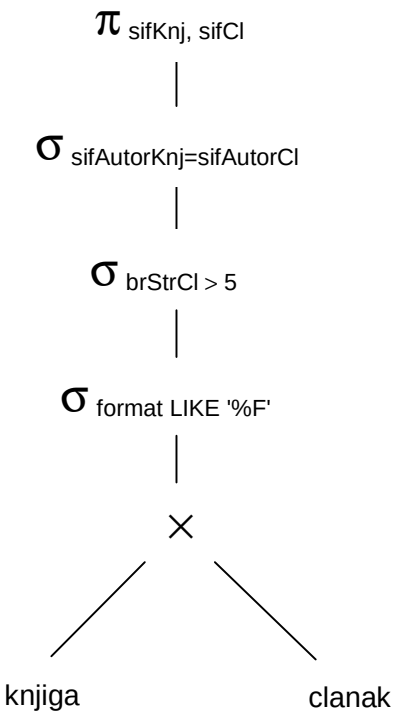
- čitanje relacije s pri obavljanju $\sigma_{D=10 \vee E=15}(s) \Rightarrow B(s) =$ **10 000**
- veličina međurezultata $B(t_1) = N(t_1) \cdot 0.2 = 1\,600$
- zapisivanje međurezultata $t_1 \Rightarrow$ **1 600**
- čitanje r i t_1 pri obavljanju $r \times t_1 \Rightarrow B(t_1) + B(r) =$ **1 850** (jer r cijela stane u međuspremnike)
- veličina međurezultata $B(t_2) = N(t_2) \cdot (0.2 + 0.1) = 2\,400\,000$
- zapisivanje međurezultata $t_2 \Rightarrow$ **2 400 000**
- čitanje t_2 radi izračunavanja $\sigma_{C=2}(t_2) \Rightarrow$ **2 400 000**
- Ukupno $\Rightarrow$ 10 000 + 1 600 + 1 850 + 2 400 000 + 2 400 000 = 4 813 450

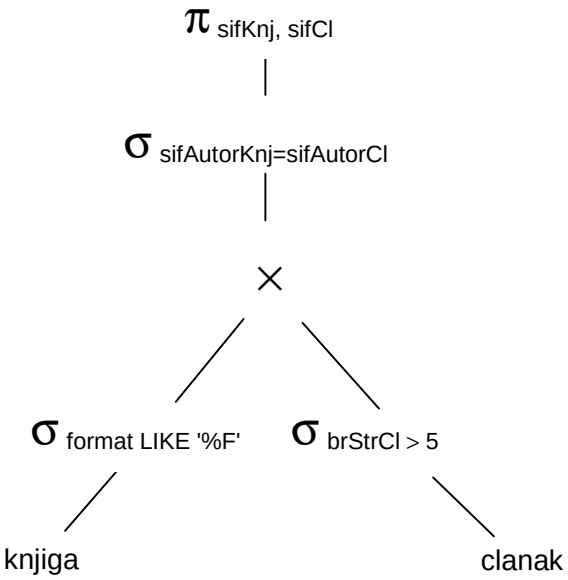
5. ZADATAK

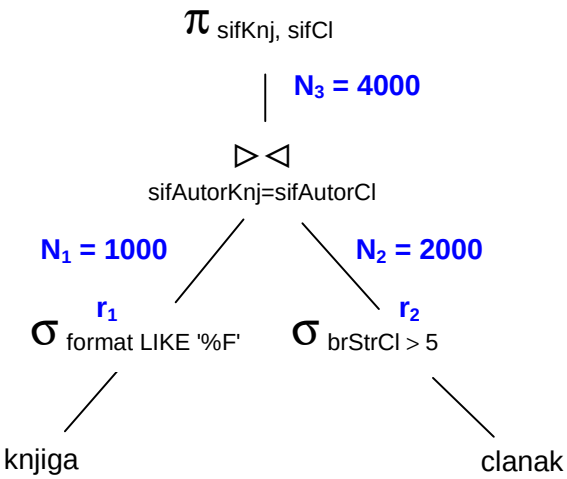
a) Inicijalni plan izvršavanja



b)







N_1 = N(knjiga) / 5

N_2 = N(clanak) / 3

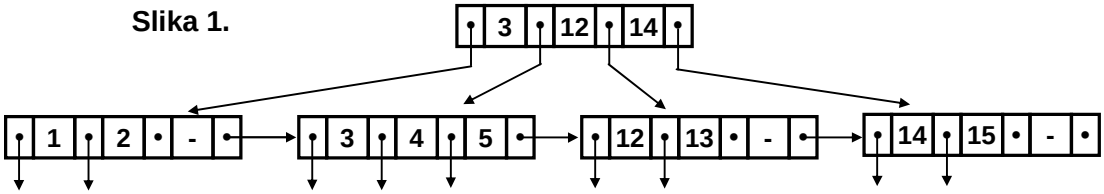
$N_3 = N(r_1) \cdot N(r_2)$ / max(V(knjiga, sifAutorKnj), V(clanak, sifAutorCl))

Sustavi baza podataka - 2. međuispit
10. svibnja 2011.

- *odgovore na pitanja 1 - 6 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

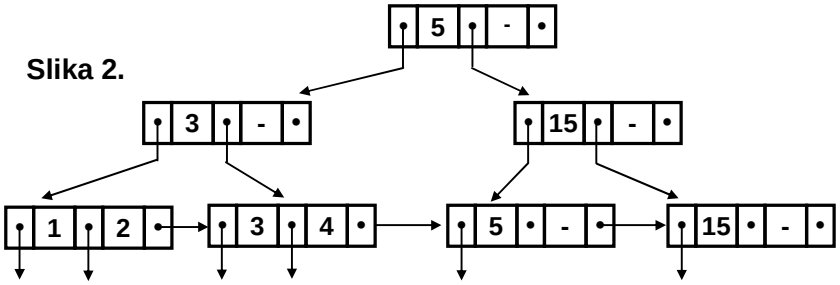
1. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **11** u B^+-stablo **na slici 1.**
   b) nakon brisanja zapisa s ključem **5** iz B^+-stabla **na slici 2.**

Slika 1.



Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 2.



(2.2 boda)

2. Naredbe kojima su kreirane relacije **racun** i **stavka** prikazane su na slikama.

```
CREATE TABLE racun (
  brRacun     INTEGER
, datRacun    DATE NOT NULL
, PRIMARY KEY (brRacun));
```

```
CREATE TABLE stavka (
  brRacun    INTEGER
, rbrStavka  INTEGER
, iznos      DECIMAL(8,2) NOT NULL
, PRIMARY KEY (brRacun, rbrStavka)
, FOREIGN KEY (brRacun)
     REFERENCES racun (brRacun));
```

Napisati pohranjenu proceduru **ubaciRacStavke** koja će kao ulazne argumente prihvatiti broj i datum računa. Procedura ubacuje odgovarajuću n-torku u relaciju **racun** i stotinu stavki (stotinu n-torki) tog računa u relaciju **stavka**. Za redne brojeve stavki upisati brojeve od 1 do 100, a kao iznos na svakoj stavki upisati 1000.00. Ne smije se obaviti upis podataka o računu, a da pri tome ne bude upisano i pripadnih stotinu stavki računa.

Ako se tijekom obavljanja INSERT naredbe za relaciju **racun** (to se ne odnosi na relaciju **stavka**!) dogodi bilo koja vrsta pogreške, u pozivajući program vratiti pogrešku s tekstom: "Unos u racun nije uspio". U slučaju bilo koje druge pogreške, u pozivajući program treba dojaviti originalnu pogrešku. Nikakve dodatne provjere nije potrebno provoditi.

Proceduru napisati tako da samostalno upravlja granicama transakcije, tj. započinje i potvrđuje/poništava transakciju. *(2.2 boda)*

3. a) definirati pojam *korektna transakcija*.
   b) definirati pojam *svojstvo izolacije* (u ovom pitanju se *izolacija* odnosi na jedno od ACID svojstava transakcije). *(2.2 boda)*

4. U tablici su prikazane operacije i redoslijed izvršavanja operacija transakcija T_1 i T_2.

   a) nacrtati graf transakcije T_1 i graf transakcije T_2. U grafove transakcija ucrtati najmanji mogući broj lúkova: isključivo one lúkove koji proizlaze iz semantike transakcija ili su nužni da bi se zadovoljila pravila konstrukcije grafova transakcija.

   b) nacrtati povijest H (u obliku grafa) koja obuhvaća transakcije T_1 i T_2 i koja odgovara redoslijedu izvršavanja prikazanom u tablici. U graf ucrtati najmanji mogući broj lúkova: isključivo one lúkove koji proizlaze iz semantike transakcija ili su nužni da bi se zadovoljila pravila konstrukcije grafa povijesti.

   c) nacrtajte serijalizacijski graf za povijest H iz b) dijela zadatka.

   (3 boda)

T_1	**T_2**
`read(x, p)`	
`p ← p + 100`	
`write(x, p)`	
	`read(x, p)`
	`p ← p * 2`
	`write(x, p)`
`read(y, p)`	
`read(z, r)`	
`write(z, p)`	
`p ← p + r`	
`write(y, p)`	
	`read(y, p)`
	`p ← p * 2`
	`write(y, p)`
`commit`	
	`commit`

5. Povijest H je prikazana u obliku topološkog poretka operacija.

   H: $r_3[y]$, $w_1[y]$, $w_3[y]$, $w_3[z]$, $r_2[z]$, c_3, $w_1[x]$, c_1, $w_2[y]$, c_2

   a) Odgovoriti je li povijest H konflikt-serijalizabilna (CSR) i objasniti zašto (zašto jest ili zašto nije).

   b) Odgovoriti je li povijest H pogled-serijalizabilna (VSR) i objasniti zašto (zašto jest ili zašto nije).

   (3 boda)

6. Za povijest H koja je prikazana u obliku topološkog poretka operacija transakcija, odrediti radi li se o RC (*recoverable*) i/ili ACA (*avoids cascading aborts*) i/ili ST (*strict*) povijesti. Objasniti odgovor.

   H: $w_2[x]$ $r_1[y]$ $r_1[x]$ $w_2[z]$ c_2 c_1

   Odgovor se mora napisati u sljedećem obliku (slijedi primjer rješenja, a ne točno rješenje):

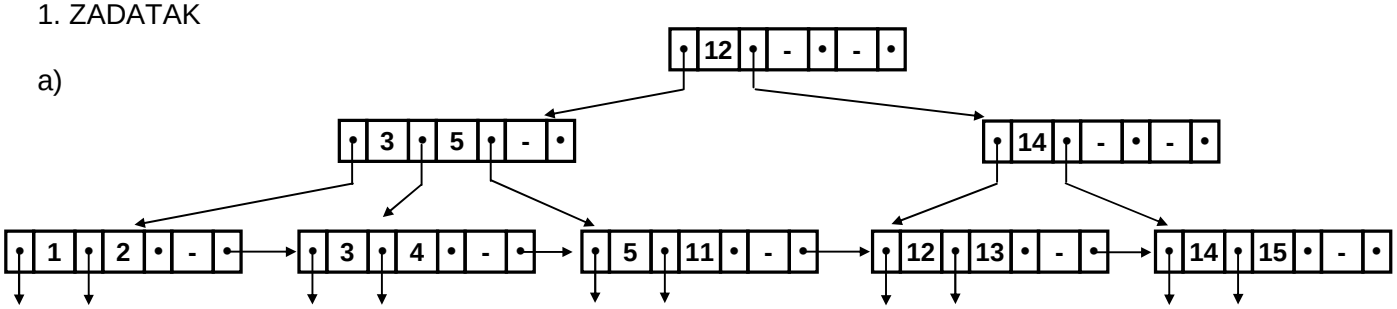
   nije RC **zato jer** ...; nije ACA **zato jer** ...; jest ST **zato jer** ... *(2 boda)*

Sustavi baza podataka - 2. međuispit
10. svibnja 2011.

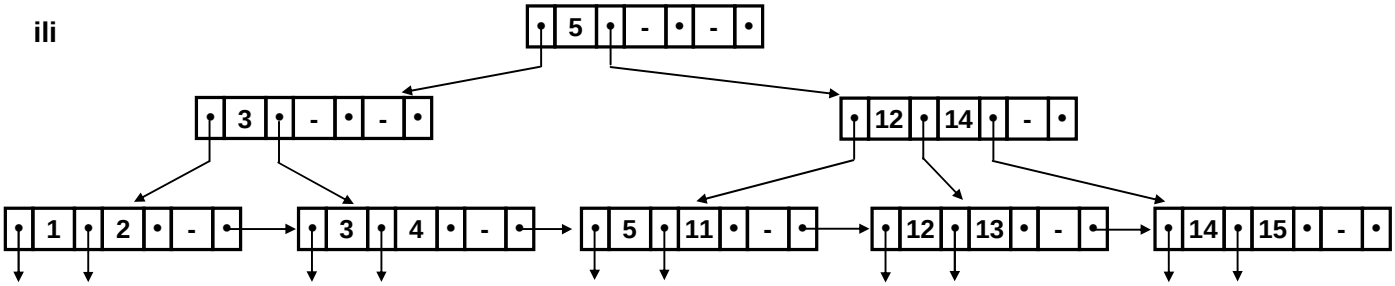
Rješenja odabranih zadataka

1. ZADATAK

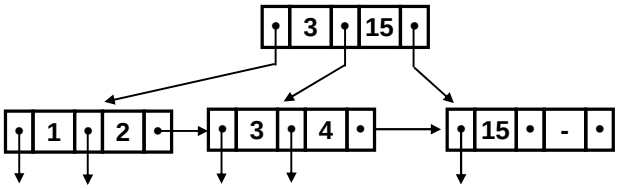
a)



ili



b)



2. ZADATAK

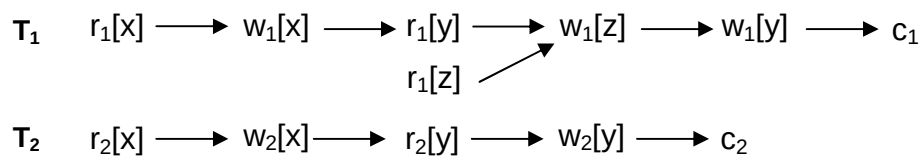
```
CREATE PROCEDURE ubaciRacStavke (p_brRacun LIKE racun.brRacun, p_datRacun LIKE racun.datRacun)
   DEFINE rbr INTEGER;
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);

   ON EXCEPTION SET sqle, isame, errdata
      ROLLBACK WORK;
      RAISE EXCEPTION sqle, isame, errdata;
   END EXCEPTION

   BEGIN WORK;
   BEGIN
      ON EXCEPTION
         RAISE EXCEPTION -746, 0, 'Unos u racun nije uspio';
      END EXCEPTION
      INSERT INTO racun VALUES (p_brRacun, p_datRacun);
   END
   FOR rbr = 1 TO 100
      INSERT INTO stavka VALUES (p_brRacun, rbr, 1000.00);
   END FOR
   COMMIT WORK;
END PROCEDURE;
```

4. ZADATAK

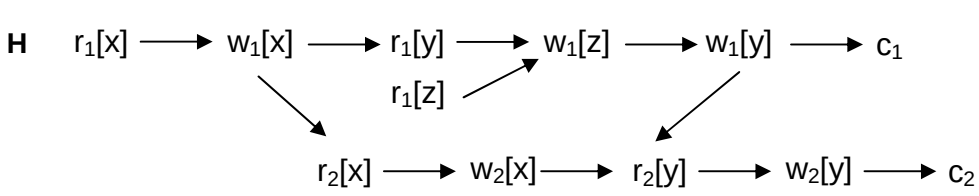
Ovisno o tome je li se u obzir uzela činjenica da se ista varijabla *p* koristi više puta (time se, strogo gledajući, znatno ograničava dopušteni redoslijed operacija), moguće su dvije varijante rješenja. Obje varijante rješenja priznate su kao jednako vrijedne.

1. varijanta

a)



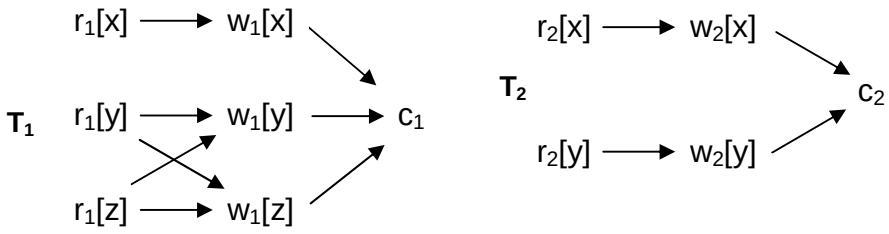
b)



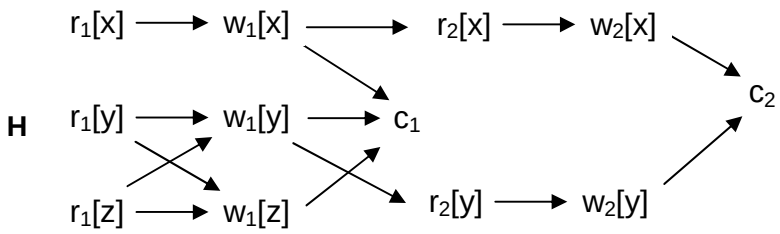
2. varijanta

(ovo rješenje više bi odgovaralo zadatku u kojem bi transakcije bile opisane kao u priloženoj tablici)

a)



b)



T₁	**T₂**
`read(x, p)`	
`p ← p + 100`	
`write(x, p)`	
	`read(x, p)`
	`p ← p * 2`
	`write(x, p)`
`read(y, r)`	
`read(z, s)`	
`write(z, r)`	
`q ← r + s`	
`write(y, q)`	
	`read(y, r)`
	`r ← r * 2`
	`write(y, r)`
`commit`	
	`commit`

U obje varijante, c) dio zadatka ima jednako rješenje

c) SG(H) $T_1 \longrightarrow T_2$

5. ZADATAK

a) U H postoje konfliktne operacije: $r_3[y] < w_1[y]$ i $w_1[y] < w_3[y]$. To je dovoljno za zaključak da u SG(H) postoji petlja između T_1 i T_3, stoga H ne može biti CSR.

b) H je VSR jer je H pogled-ekvivalentna serijskoj povijesti H_s

H: $r_3[y]$, $w_1[y]$, $w_3[y]$, $w_3[z]$, $r_2[z]$, c_3, $w_1[x]$, c_1, $w_2[y]$, c_2

H_s: $r_3[y]$, $w_3[y]$, $w_3[z]$, c_3, $w_1[y]$, $w_1[x]$, c_1, $r_2[z]$, $w_2[y]$, c_2

$H \equiv_V H_S$ jer:

- po pitanju čitanja inicijalnih vrijednosti
  - za obje povijesti vrijedi: T_3 čita inicijalni y
- po pitanju čitanja vrijednosti koje je zapisala neka druga transakcija
  - za obje povijesti vrijedi: T_2 čita z kojeg je zapisala T_3
- po pitanju završne operacija pisanja
  - za obje povijesti vrijedi: T_1 zapisuje završnu vrijednost za x
  - za obje povijesti vrijedi: T_2 zapisuje završnu vrijednost za y
  - za obje povijesti vrijedi: T_3 zapisuje završnu vrijednost za z

6. ZADATAK

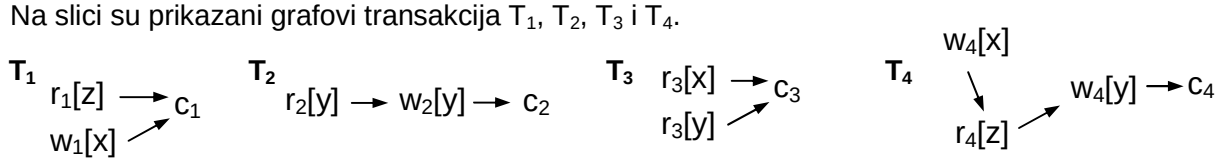
H jest RC zato jer iako T_1 čita x iz T_2, $c_2 < c_1$; nije ACA zato jer T_1 čita x iz T_2 i pri tome je $r_1[x] < c_2$; nije ST, jer ako nije ACA, tada nije niti ST

Sustavi baza podataka - Završni ispit
21. lipnja 2011.

Odabrani zadaci

- *odgovore na pitanja 1 - 5 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. Na slici su prikazani grafovi transakcija T_1, T_2, T_3 i T_4.



   a) među ponuđenima odabrati dva grafa, te za odabrane grafove transakcija nacrtati graf povijesti čije bi izvršavanje (kad bi se dopustilo) izazvalo anomaliju nekonzistentne analize
   b) među ponuđenima odabrati dva grafa, te za odabrane grafove transakcija nacrtati graf povijesti čije bi izvršavanje (kad bi se dopustilo) izazvalo anomaliju izgubljene izmjene
   c) među ponuđenima odabrati dvije transakcije, te napisati niz operacija odabranih transakcija čije bi izvršavanje uz upotrebu 2PL protokola izazvalo potpuni zastoj

   Napomena: zadaci a), b), c) se rješavaju nezavisno *(4 boda)*

2. U bazi podataka BP nalaze se relacije R_1, R_2, R_3, itd. U relaciji R_1 nalaze se n-torke $t_{1.1}$, $t_{1.2}$, itd, u relaciji R_2 se nalaze n-torke $t_{2.1}$, $t_{2.2}$, itd. SUBP koristi rigorozni **2PL protokol i MGL protokol** uz hijerarhiju objekata: baza podataka → relacija → n-torka. Ne koriste se U-ključevi. Modalitet zaključavanja je postavljen tako da transakcije, ako ne dobiju ključ, čekaju na zaključavanje. U sustav pristižu operacije transakcija T_1, T_2 i T_3, redom kako su navedene:

   $r_1[t_{1.1}]$, $w_2[t_{2.2}]$, r_3[sve n-torke R_3], $w_3[t_{2.9}]$, r_2[sve n-torke R_1], $w_2[t_{3.6}]$, $w_3[t_{3.3}]$

   Objašnjenje oznaka:
   $r_1[t_{2.3}]$ - T_1 čita n-torku $t_{2.3}$
   r_1[sve n-torke R_2] - T_1 čita sve n-torke relacije R_2
   $w_1[t_{2.3}]$ - T_1 piše u n-torku $t_{2.3}$

   Za svaku operaciju navesti koji su ključevi postavljeni (ili odbijeni) na kojem elementu baze podataka. Rješenje mora po formi biti slično priloženom primjeru rješenja.

   (4 boda)

   Primjer rješenja:
   $r_1[t_{1.1}]$: dobije X na $t_{1.1}$, dobije S na BP
   $w_2[t_{2.2}]$: odbijen S na BP, T_2 čeka
   r_3[sve n-torke R_3]: ima S na BP, dobije S na svaku n-torku u R_3
   $w_3[t_{2.9}]$: promovira S→X na BP, dobije X na $t_{2.3}$
   r_2[sve n-torke R_1]: T_2 i dalje čeka
   itd.

3. U distribuiranom sustavu za upravljanje bazama podataka koristi se distribuirani menadžer zaključavanja (svaki čvor održava vlastiti, lokalni WFG). Transakcija T_1 se inicira u čvoru S_1, T_2 u čvoru S_2, T_3 u čvoru S_4, T_4 u čvoru S_3. Neka je:
   Shema alokacije S_1: x, y; S_2: z; S_3: u; S_4: v
   Globalna povijest $r_4[y]$, $w_3[u]$, $r_3[v]$, $w_4[u]$, $w_1[z]$, $w_2[x]$, $w_3[y]$, $w_2[v]$, $w_1[x]$, $r_2[z]$, c_1, c_2, c_3, c_4

   a) napisati koje se subtransakcije obavljaju u kojim od čvorova S_1, S_2, S_3 i S_4
   b) nacrtati lokalne grafove čekanja (WFG) u čvorovima S_1, S_2, S_3 i S_4 u trenutku kada u sustavu nastane potpuni zastoj
   c) nacrtati globalni graf čekanja u trenutku kada u sustavu nastane potpuni zastoj *(3 boda)*

4. U sustavu IBM IDS kreirana je relacija osoba (uočiti primarni ključ) i dodatni indeks nad atributom ime. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: relacija → n-torka) i protokol zaključavanja indeksa. Ne koriste se U-ključevi.

sif	ime	prez
100	Ana	Horvat
101	Ivo	Ban
102	Jure	Ban
103	Ivo	Kolar
104	Jure	Novak
105	Ivo	Novak
106	Ana	Turk
107	Edo	Keler

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(50)
 , prez   CHAR(50)
 , PRIMARY KEY (sif)
) LOCK MODE ROW;

CREATE INDEX idxIme
   ON osoba (ime);
```

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	**T_2**

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;`

`1. SELECT * FROM osoba` `WHERE ime = 'Ana';`	`2. UPDATE osoba SET prez='Tak'` `WHERE ime = 'Jure';`
`3. UPDATE osoba SET prez='Hil'` `WHERE sif = 106;`	`4. DELETE FROM osoba` `WHERE ime = 'Ana';`

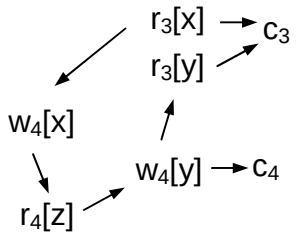
Za svaku naredbu (1-4) napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka i je li zaključavanje uspjelo (s obzirom na već postavljene ključeve). *(3 boda)*

5. Sustav s potpuno repliciranom bazom podataka obuhvaća čvorove S_1, S_2, ..., S_5. Odrediti parametre za *quorum consensus* protokol uz koje će taj protokol djelovati kao:
   a) protokol zaključavanja uz pomoć primarne kopije koja se uvijek nalazi u čvoru S_1
   b) pristrani protokol (*biased protocol*)

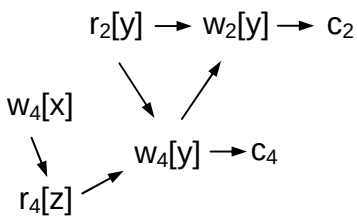
   Napomena: zadaci a), b) se rješavaju nezavisno *(3 boda)*

Rješenja odabranih zadataka

1. **a)**



b)



c) $r_3[y], w_4[x], r_3[x], r_4[z], w_4[y]$

ili

$w_4[x], r_3[y], r_3[x], r_4[z], w_4[y]$

2. $r_1[t_{1.1}]$: dobije IS na BP, dobije IS na R_1, dobije S na $t_{1.1}$

$w_2[t_{2.2}]$: dobije IX na BP, dobije IX na R_2, dobije X na $t_{2.2}$

r_3[sve n-torke R_3]: dobije IS na BP, dobije S na R_3

$w_3[t_{2.9}]$: promovira IS→IX na BP, dobije IX na R_2, dobije X na $t_{2.9}$

r_2[sve n-torke R_1]: ima IX na BP, dobije S na R_1

$w_2[t_{3.6}]$: ima IX na BP, odbijen IX na R_3, T_2 čeka

$w_3[t_{3.3}]$: ima IX na BP, promovira S→SIX na R_3, dobije X na $t_{3.3}$

3. a)

S_1: $T_1 : w_1[x]$ $T_2 : w_2[x]$ $T_3 : w_3[y]$ $T_4 : r_4[y]$

S_2: $T_1 : w_1[z]$ $T_2 : r_2[z]$

S_3: $T_3 : w_3[u]$ $T_4 : w_4[u]$

S_4: $T_2 : r_3[v]$ $T_3 : w_2[v]$

U potpuno korektnom rješenju subtransakcija i njihove operacije bi bile označene indeksom čvora u kojima se izvršavaju, npr.

T_2^1

U slučajevima u kojima ne može doći do zabune (poput ovog), notacija se može pojednostaviti.

U dijelovima b) i c), ovaj zadatak nije bio posve precizno zadan, te je interpretiran na različite načine. Zato su se priznavala sva rješenja koja su se zadržala u okviru mogućih interpretacija zadatka.

A) rješenje uz strogu interpretaciju dijela rečenice "u trenutku kada u sustavu nastane potpuni zastoj". Ovdje se podrazumijeva da se operacije, koje pristižu nakon trenutka u kojem je (bilo gdje u sustavu) nastao potpuni zastoj, uopće ne razmatraju. Potpuni zastoj nastat će već pokušajem obavljanja operacije $w_3[y]$. Stanje lokalnih i globalnog WFG u tom je trenutku sljedeće (primijetiti i one čvorove koji nisu povezani lukovima):

b)

WFG^1: T_2; $T_3 \longrightarrow T_4$

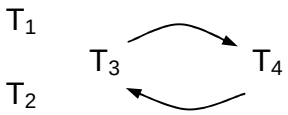
WFG^2: T_1

WFG^3: $T_4 \longrightarrow T_3$

WFG^4: T_3

c)

Globalni WFG



B) rješenje uz interpretaciju dijela rečenice "u trenutku kada u sustavu nastane potpuni zastoj" na taj način da se trenutak nastanka potpunog zastoja smatra trenutak u kojem je potpuni zastoj (sa zakašnjenjem) detektiran. Zakašnjenje bi se npr. svakako dogodilo ako bi se za detekciju potpunih zastoja koristio mehanizam *deadlock-timeout*. Ovdje se smatra da su se operacije transakcija obavljale sve dok sve četiri transakcije nisu ušle u stanje čekanja na zaključavanje, a potpuni zastoj je detektiran tek nakon isteka perioda *deadlock-timeout*.

Zadnja operacija koja se u tom slučaju pokušala obaviti je operacija $w_1[x]$, Nakon pokušaja obavljanje te operacije sve četiri transakcije su u stanju čekanja (operacija $r_2[z]$ se neće pokušati obaviti jer je T_2 već od prije u stanju čekanja). Stanje lokalnih i globalnog WFG u tom je trenutku sljedeće (primijetiti i one čvorove koji nisu povezani lukovima):

b)

WFG^1: $T_1 \longrightarrow T_2$; $T_3 \longrightarrow T_4$

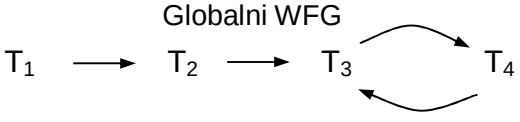
WFG^2: T_1

WFG^3: $T_4 \longrightarrow T_3$

WFG^4: $T_2 \longrightarrow T_3$

c)

Globalni WFG



4.

1. IS na relaciju osoba - da; S na zapis indeksa ime='Ana' - da; S na n-torke sif=100 i 106 - da
2. IX na relaciju osoba - da; S na zapis indeksa ime='Jure' - da; X na n-torke sif=102 i 104 - da
3. IS $\rightarrow$ IX na relaciju osoba - da; S na zapis indeksa sif=106; S $\rightarrow$X na n-torku sif=106 - da
4. IX na relaciju osoba već postoji; X na zapis indeksa ime='Ana' - odbijen

5. **a)** težinski faktor čvora $w_1 = 5$, težinski faktori ostalih čvorova: $w_i = 1$, za $2 \leq i \leq 5$
za svaki element x replicirane baze podataka: $Q_r = 5$, $Q_w = 5$

b) težinski faktori svih čvorova: $w_i = 1$, $1 \leq i \leq 5$
za svaki element x replicirane baze podataka: $Q_r = 1$, $Q_w = 5$

Ovdje prikazano rješenje, naravno, nije jedino moguće.

Sustavi baza podataka - Međuispit
26. travnja 2012.

1. a) Nacrtati B^+-stablo **reda 3** tako da **ukupni broj čvorova** B-stabla bude **minimalan**. Stablo treba sadržavati sljedeće vrijednosti ključeva: 2, 3, 4, 5, 7, 8, 10, 11, 12, 13.

   b) Nacrtati B^+-stablo nakon dodavanja zapisa s ključem 6 i zapisa s ključem 9 u stablo nacrtano u **a)** dijelu zadatka.

   Dovoljno je nacrtati samo konačna rješenja za a) i za b) dio zadatka. Blokove s podacima nije potrebno crtati

   (3 boda)

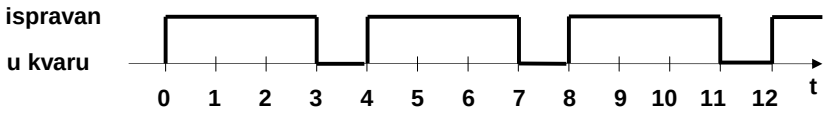
2. Za obnovu nakon pogreške sustava SUBP koristi *redo*/*undo* tehniku i kontrolne točke (*checkpoint*).
   a) operacija *flush log* kojom se zapis dnevnika <T, x, V_{old}, V_{new}> upisuje u stabilnu memoriju mora biti obavljena prije operacije *output*(x). Objasniti zašto se **ne smije** obaviti poslije.
   b) tijekom obnove nakon pogreške sustava moraju se **poništavati** i neke operacije koje su u dnevnik upisane prije posljednje kontrolne točke.
      - Na operacije kojih transakcija se odnosi ta tvrdnja?
      - Zašto se te operacije moraju poništavati?
      - Zašto se, slično tome, ne moraju **ponovno obavljati** operacije koje su u dnevnik upisane prije posljednje kontrolne točke (nego samo one koje su u dnevnik upisane poslije kontrolne točke)?

      (3 boda)

3. Graf na slici 1. ilustrira ponašanje jednog SUBP-a tijekom vremena (sustav se promatra u kontekstu obnove nakon pogreške).
   a) izračunati raspoloživost (*availability*) prikazanog sustava
   b) nacrtati sličan graf koji prikazuje ponašanje SUBP-a **jednake** raspoloživosti, ali **manje** pouzdanosti (*reliability*) od sustava na slici 1. *Rješenje nije jednoznačno, nacrtati bilo koje od mogućih rješenja.*
   c) što je to *resilience*? Na koje ili koja od svojstava *availability*, *reliability*, *resilience*, utječu karakteristike sustava za obnovu?

   (4 boda)

Slika 1.



4. Napisati pohranjenu proceduru **`razdvojiRacune`** koja će svaki račun čiji broj ne završava slovom A ili B zamijeniti s dva nova računa. Brojevi novih računa dobiju se ulančavanjem prethodne vrijednosti broja računa sa slovom A odnosno B, a novi iznosi računa kao polovice prethodnih iznosa računa. Na slici je prikazan sadržaj relacije `racun` prije i nakon uspješnog izvršavanja procedure.

```
CREATE TABLE racun (
  brRac CHAR(10)      NOT NULL
, iznos  DECIMAL(8,2) NOT NULL
, PRIMARY KEY (brRac)
);
```

racun (prije)

brRac	iznos
1	100.00
4A	500.00
32	300.00
...	

racun (poslije)

brRac	iznos
1A	50.00
1B	50.00
4A	500.00
32A	150.00
32B	150.00
...	

Ako je iznos na nekom od računa kojeg treba razdvojiti negativan, u pozivajući program dojaviti pogrešku s tekstom **`'Neispravni podaci'`**. Ako se pri obavljanju bilo koje INSERT operacije dogodi bilo koja vrsta pogreške, procedura u pozivajući program vraća pogrešku s tekstom **`'Pogreška unosa'`**. U slučaju pojave bilo koje druge pogreške na bilo kojem drugom mjestu u proceduri, procedura u pozivajući program mora proslijediti originalnu pogrešku.

Procedura mora osigurati da računi koji zadovoljavaju kriterij za zamjenu budu zamijenjeni svi ili niti jedan.

(4 boda)

5. Obavlja se operacija `SELECT * FROM r ORDER BY a`. Postoji indeks za atribut a u relaciji r. Navedite koji bi faktori mogli dovesti do odluke optimizatora da se sortiranje izvrši:
   a) u glavnoj memoriji
   b) vanjskim sortiranjem s uparivanjem (*external sort-merge*)
   c) korištenjem indeksa *(2 boda)*

6. Zadane su relacije r(A, B), s(C, D), t(E, F). Izvršava se upit koji je opisan sljedećim izrazom relacijske algebre:

$$\sigma_{B=C \wedge D<7 \wedge A \neq F} ((r \times s) \underset{D=E}{\rhd\lhd} t)$$

   a) nacrtati stablo upita (*query tree*) za prikazani izraz relacijske algebre
   b) nacrtati stablo upita i odgovarajući izraz relacijske algebre nakon provedene heurističke optimizacije
   (3 boda)

7. Zadane su relacije r(<u>A</u>, B) i s(<u>C</u>, D, E, F). Primarni ključevi relacija su potcrtani. Nad relacijama nije izgrađen niti jedan indeks. Za operaciju spajanja uz uvjet koristi se fizički operator *block nested-loop join*. Za obavljanje fizičkih operacija na raspolaganju je 100 blokova primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Operacije se izvršavaju redoslijedom kako je navedeno u izrazu relacijske algebre, što znači da ne treba provoditi heurističku optimizaciju. Koristeći priložene podatke, procijeniti broj **<u>U/I operacija</u>** tijekom izvršavanja upita, te **<u>broj n-torki</u>** u konačnom rezultatu. U rješenju prikazati postupak (ne samo konačni rezultat).

$$\sigma_{D \neq 2 \vee E \leq 1000} (s) \underset{A=F}{\rhd\lhd} r$$

```
V(B, r) = 50
V(D, s) = 2
V(E, s) = 10
V(F, s) = 5
```

```
N(r) =  5000
N(s) =  2000
```

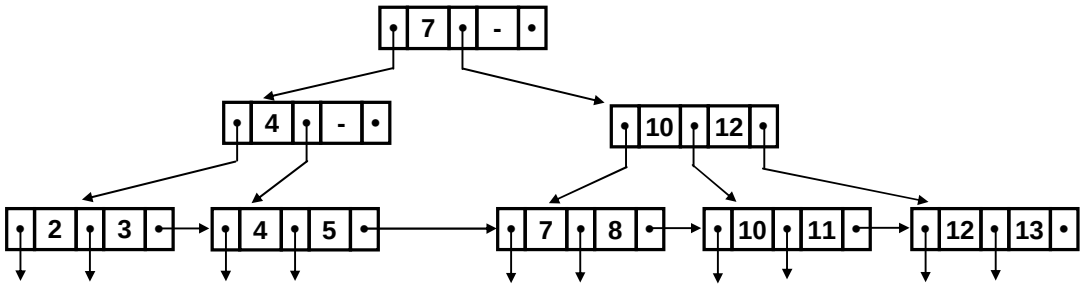
```
B(r) =  5000
B(s) =  1000
```

```
veličina n-torke(r) = 0.5 blokova
veličina n-torke(s) = 0.1 blokova
```

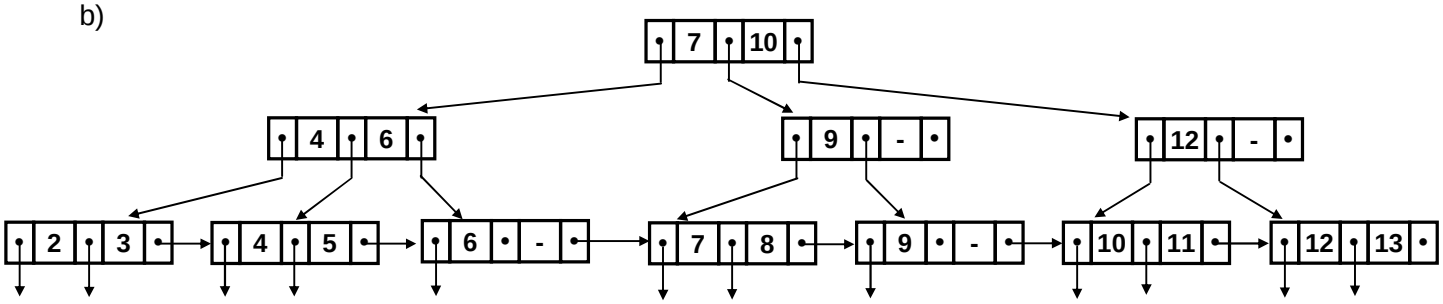
(4 boda)

Rješenja odabranih zadataka:

1. a)

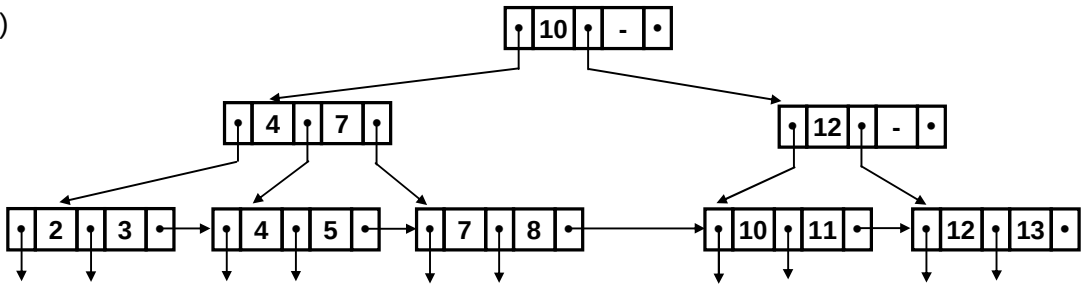


b)

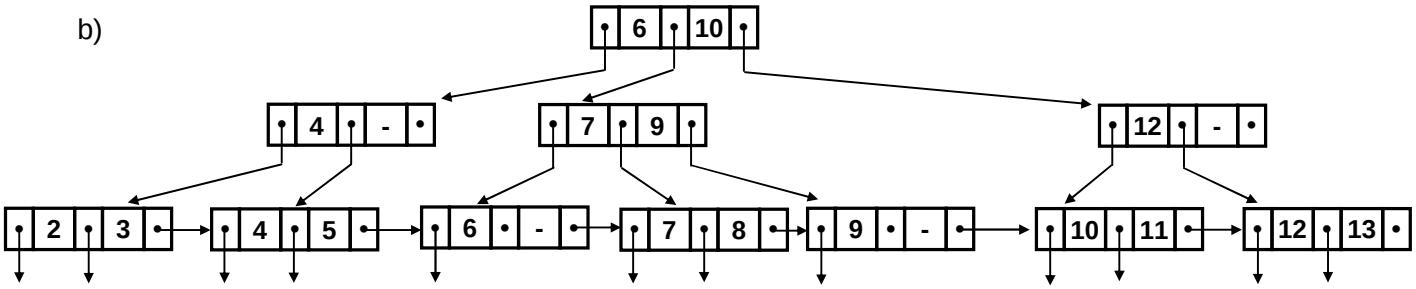


ili

a)



b)



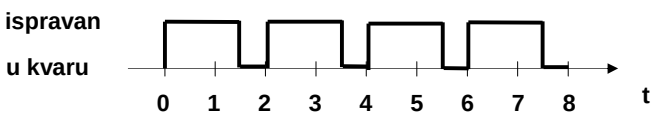
2. b) Tvrdnja se odnosi na transakcije koje su bile u tijeku u trenutku posljednje kontrolne točke, a nisu potvrđene (dosegle točku potvrđivanja) do trenutka kvara.

Te se operacije moraju poništiti jer su rezultati njihovih *write* operacija tijekom kontrolne točke upisani u postojanu memoriju operacijom *output*.

Transakcije čije se operacije moraju ponovo obaviti su dosegle točku potvrđivanja prije kvara. Njihove operacije, koje su u dnevnik upisane prije kontrolne točke, ne moraju se ponovo obaviti jer su rezultati tih operacija sigurno tijekom kontrolne točke operacijom *output* upisani u postojanu memoriju

3. a) 0.75 ili 75%

b)



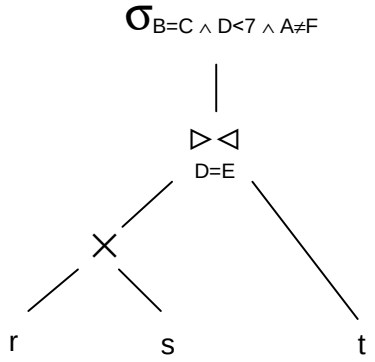
c) sposobnost brzog oporavka od različitih vrsta pogrešaka. Sustav za obnovu može utjecati na svojstva *avalilability* (raspoloživost) i *resilience* (sposobnost brzog oporavka nakon pogreške). *Reliability* nije predmet razmatranja u sustavima za obnovu sustava za upravljanje bazama podataka.

4.

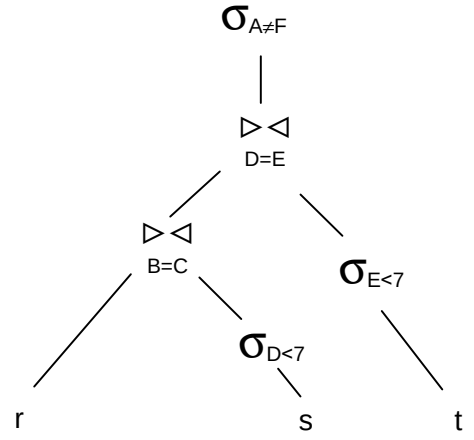
```
CREATE PROCEDURE razdvojiRacune ()
   DEFINE p_brRac LIKE racun.brRac;
   DEFINE p_iznos LIKE racun.iznos;
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);

   ON EXCEPTION SET sqle, isame, errdata
      ROLLBACK WORK;
      RAISE EXCEPTION sqle, isame, errdata;
   END EXCEPTION

   BEGIN WORK;
   FOREACH SELECT * INTO p_brRac, p_iznos
             FROM racun WHERE brRac NOT LIKE '%A' AND brRac NOT LIKE '%B'
      IF p_iznos < 0 THEN
         RAISE EXCEPTION -746, 0, 'Neispravni podaci';
      END IF
      BEGIN
        ON EXCEPTION
           RAISE EXCEPTION -746, 0, 'Pogreška unosa';
        END EXCEPTION
        INSERT INTO racun VALUES(TRIM(p_brRac) || 'A', p_iznos/2);
        INSERT INTO racun VALUES(TRIM(p_brRac) || 'B', p_iznos/2);
      END
      DELETE FROM racun WHERE brRac = p_brrac;
   END FOREACH
   COMMIT WORK;
END PROCEDURE;
```

5. a) proces ima na raspolaganju dovoljno primarne memorije za smještaj cijele relacije
b) proces nema na rapolaganju dovoljno primarne memorije za smještaj cijele relacije
c) visok stupanj grupiranja indeksa (*clustered index*)

6. a)



b)



$$\sigma_{A\neq F}\left(\left(\sigma_{D<7}(s) \underset{B=C}{\rhd\lhd} r\right) \underset{D=E}{\rhd\lhd} \sigma_{E<7}(t)\right)$$

7. Broj n-torki

- selektivnost $f(D\neq 2, s) = N(\sigma_{D\neq 2}, s) / N(s) = (N(s) - N(\sigma_{D=2}, s)) / N(s) =$
  $= (N(s) - N(s) / V(D, s)) / N(s) = 0.5$
- selektivnost $f(E\leq 1000, s) = N(\sigma_{E\leq 1000}, s) / N(s) = N(s) / 3 / N(s) = 0.3333$
- $t_1 = \sigma_{D\neq 2 \vee E\leq 1000}(s)$
- $N(t_1) = N(s) \cdot (1 - (1 - f(D\neq 2, s)) \cdot (1 - f(E\leq 1000, s))) = 2\,000 \cdot 0.6666 = 1333$
- $t_2 = t_1 \rhd\lhd_{A=F} r$
- $N(t_2) = N(t_1)$ jer je A primarni ključ u r
- $N(t_2) = 1333$

Broj U/I operacija

- čitanje relacije s pri obavljanju $\sigma_{D\neq 2 \vee E\leq 1000}(s) \Rightarrow B(s) =$ **1 000**
- veličina međurezultata $B(t_1) = N(t_1) \cdot 0.1 = 134$
- zapisivanje međurezultata $t_1 \Rightarrow$ **134**
- obavljanje $t_1 \rhd\lhd_{A=F} r$: niti jedna relacija ne stane cijela u međuspremnike)
- $\Rightarrow B(t_1)/(M-2) \cdot B(r) + B(t_1) = 134/98 \cdot 5000 + 134 = 2 \cdot 5000 + 134 =$ **10 134**
- objašnjenje: treba dva puta čitati cijeli r: jednom za uparivanje s prvih 98 blokova relacije t_1, a drugi puta radi uparivanja s preostalih 36 blokova relacije t_1. Na kraju pribrojiti trošak čitanja same relacije t_1: u prvom koraku 98 blokova, u drugom 36 blokova, ukupno $B(t_1)$
- ukupno (bez zapisivanja konačnog rezultata) $\Rightarrow$ 1 000 + 134 + 10 134 = 11 268

Prezime, ime

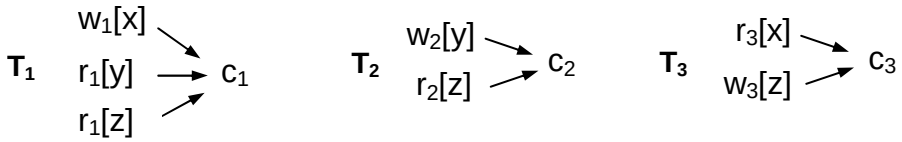
Sustavi baza podataka - Završni ispit
21. lipnja 2012.

- *odgovore na pitanja 1 - 6 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. Na slici su prikazani grafovi transakcija T_1 i T_2.

   a) Nacrtati graf povijesti H_1 koja obuhvaća T_1 i T_2 i čije bi izvršavanje (kad bi se dopustilo) izazvalo nekonzistentnu analizu.
   b) Što je to prljavo čitanje? Ako je u H_1 moguća pojava prljavog čitanja, navesti koje ga točno operacije i zbog čega mogu uzrokovati?
   c) U općem slučaju, koje od svojstava povijesti (RC, ACA, ST) sprečava prljavo čitanje? Objasniti zašto. *(6 bodova)*

2. Na slici su prikazani grafovi transakcija T_1, T_2 i T_3.



   a) U obliku niza operacija (ne u obliku grafa) napisati povijest H_2 koja će dovesti do potpunog zastoja u kojem će sudjelovati sve tri transakcije.
   b) Nacrtati WFG graf u trenutku nastanka potpunog zastoja.
   c) Što će sustav poduzeti u cilju razrješavanja potpunog zastoja i kako WFG izgleda u trenutku neposredno nakon razrješenja potpunog zastoja?
   d) Ispitati je li povijest H_2 konflikt-serijalizabilna. *(5 bodova)*

3. Što to znači da je 2PC protokol *blokirajući protokol*? U kojem se trenutku i zašto dešava blokiranje 2PC protokola? *(3 boda)*

4. Potpuno replicirani sustav obuhvaća čvorove S_1 i S_2. Shema baze podataka sadrži shemu samo jedne relacije. Shema i sadržaj relacije u trenutku t_0 su prikazani na slikama.

```
CREATE TABLE zgrada (
   sif     INTEGER
 , visina  INTEGER
 , PRIMARY KEY (sif));
```

   U trenutku t_0 baza podataka je konzistentna: sve fizičke kopije logičkih elemenata su međusobno jednake.

sif	visina
10	22
11	50
12	22

   a) Uz pretpostavku korištenja asinkrone dvosmjerne replikacije, napisati jedan primjer niza akcija koje će korisnici i sustav obaviti nakon trenutka t_0, a koji će dovesti do nekonzistentnosti (*system delusion*), odnosno konflikta kojeg će morati razriješiti čovjek.
   b) Objasniti kako bi primjena sinkrone (*eager*) dvosmjerne replikacije sigurno spriječila neželjeni slijed događaja iz a) dijela zadatka. *(3 boda)*

5. Objasniti zašto je u tablici kompatibilnosti ključeva za MGL protokol u presjeku stupca **IX** i retka **IS** oznaka *true*, te zašto je u presjeku stupca **SIX** i retka **IX** oznaka *false*. *(3 boda)*

6. U bazi podataka kreirana je relacija *osoba* i indeks *idx*. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: *baza podataka* → *relacija* → *n-torka*) i protokol zaključavanja indeksa, ali samo ako su za korištenje protokola zaključavanja indeksa ostvareni potrebni preduvjeti. Ne koriste se U-ključevi.

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(20)
 , prez   CHAR(20)
) LOCK MODE ROW;

CREATE UNIQUE INDEX idx
  ON osoba (sif);
```

sif	ime	prez
100	Ana	Horvat
103	Ivo	Ban
106	Jure	Ban
109	Ivo	Kolar
112	Jure	Novak
115	Ivo	Novak
118	Ana	Turk
121	Edo	Keler

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	**T_2**

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`

`1. SELECT * FROM osoba WHERE sif=100`	`2. SELECT * FROM osoba WHERE sif=100`
`3. SELECT * FROM osoba WHERE sif=118`	`4. UPDATE osoba SET prez='Li' WHERE sif=112`
`5. UPDATE osoba SET prez='Ho' WHERE sif=118`	`6. DELETE FROM osoba WHERE sif=106`
`7. INSERT INTO osoba VALUES(106, 'Jo', 'Wu')`	`8. SELECT * FROM osoba WHERE sif=106`
`9. SELECT * FROM osoba`	`10. SELECT * FROM osoba`

a) Za svaku naredbu (1-10) koja se uspije obaviti prije nego transakcija uđe u stanje čekanja na postavljanje ključa, napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka. Ako zaključavanje nije uspjelo navesti i razlog zašto. Rješenje treba biti u obliku prikazanom u priloženom primjeru rješenja.

Primjer oblika rješenja:
1. T_1 postavlja S na n-torke 100, 102, 102, postavlja X na n-torke 105, 107
2. T_2 postavlja IS na n-torke 103, 104, X na osoba
3. T_1 **promovira** S u IX na n-torci 102
4. T_2 ne uspijeva postaviti S na bazu podataka jer je T_2 već prije s X zaključala relaciju, T_2 **čeka**
5. T_1 uspješno promovira S u X nad bazom podataka

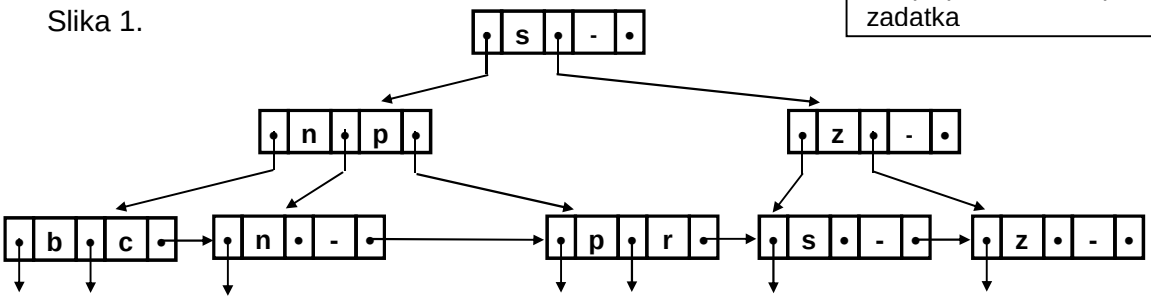
b) Ponovo riješiti zadatak pod a), ali ovog puta uz pretpostavku da nad relacijom osoba nije kreiran indeks `idx`. *(6 bodova)*

Sustavi baza podataka - Međuispit
30. travnja 2013.

1. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **m** u B^+-stablo na slici 1.

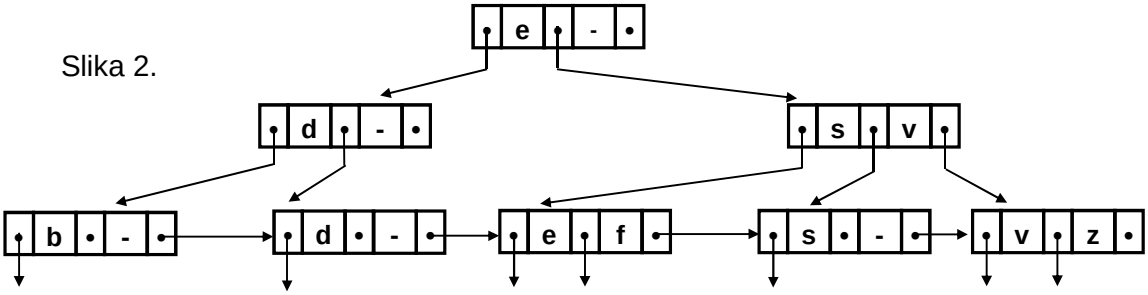
Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



   b) nakon brisanja zapisa s ključem **d** iz B^+-stabla na slici 2.

Slika 2.



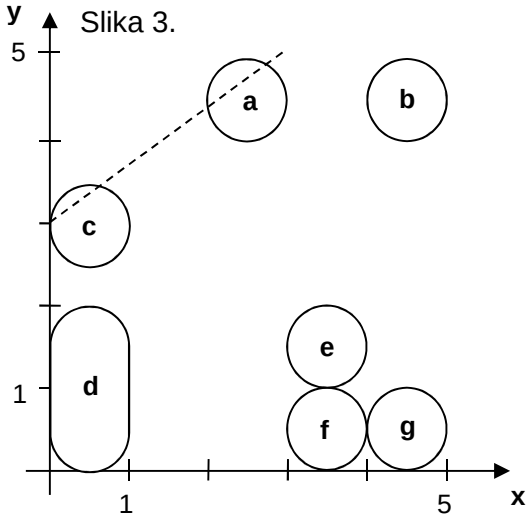
(4 boda)

2. a) Nacrtati R-stablo za prostorne podatke (objekte, odnosno geometrijske likove a-g) prikazane na slici 3.

   Najveći dopušteni broj zapisa (ključ+kazaljka) u listu i internom čvoru R-stabla je 3, a najmanji broj zapisa je 2. Rješenje a) dijela zadatka treba sadržavati dvije slike. Na prvoj slici nacrtati sve minimalne granične okvire (MBR). Na drugoj slici nacrtati R-stablo.

   b) Objasniti postupak kojim se pomoću R-stabla iz a) dijela zadatka pronalaze svi objekti koji su u presjeku s linijom koja se proteže od točke s koordinatama (x=0, y=3) do točke s koordinatama (x=3, y=5) *(4 boda)*

Slika 3.



3. Usporediti magnetski disk (HDD) i *flash* memoriju (SSD) prema svojstvima koja su značajna za njihovo korištenje u sustavima za upravljanje bazama podataka. *(3 boda)*

4. Relacije *lijeva*, *desna* i *ulazna* imaju jednake sheme. Njihove sheme sadrže samo atribut *rbr* tipa INTEGER, te je za taj atribut u sve tri relacije kreiran primarni ključ. Relacije *lijeva* i *desna* su prazne, a relacija *ulazna* je popunjena nepoznatim brojem n-torki.

   Napisati pohranjenu proceduru **`razvrstaj`** koja će sve n-torke iz relacije *ulazna* razvrstati u relacije *lijeva* i *desna* na sljedeći način: poredane prema vrijednosti *rbr*, od manjih prema većim, prva n-torka prebacuje se u *lijeva*, druga n-torka u *desna*, treća n-torka u *lijeva*, itd.

   Primjer: neka se u relaciji *ulazna* nalaze n-torke s vrijednostima atributa *rbr* 7, 4, 1, 2 i 5. Nakon uspješnog izvršavanja procedure, relacija *lijeva* treba sadržavati n-torke 1, 4 i 7, relacija *desna* n-torke 2 i 5, dok relacija *ulazna* treba biti prazna.

   Proceduru napisati tako da se njenim izvršavanjem pouzdano razvrstaju ili sve n-torke ili niti jedna. Ako se pri obavljanju INSERT naredbe u relaciju *lijeva* dogodi bilo koja vrsta pogreške, procedura u pozivajući program vraća pogrešku s tekstom **`'Pogreška unos lijeva'`**. U slučaju pojave bilo koje druge pogreške na bilo kojem mjestu u proceduri, procedura u pozivajući program mora proslijediti originalnu pogrešku. *(5 bodova)*

5. Zadane su relacije r(A, B, C, D), s(E, F) i t(A, B, C, D, E, F). Primarni ključevi relacija su potcrtani. Relacije nisu unaprijed sortirane. Nad relacijama nije izgrađen niti jedan indeks. Za operaciju spajanja s izjednačavanjem koristi se fizički operator *block nested-loop join*.

```
V(B, r) =   2
V(C, r) =   5
V(D, r) = 200
V(F, s) = 100
```

   Za obavljanje fizičkih operacija na raspolaganju je 1002 blokova primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Operacije se izvršavaju redoslijedom kako je navedeno u izrazu relacijske algebre, što znači da ne treba provoditi heurističku optimizaciju. Koristeći priložene podatke, procijeniti **broj n-torki** i **broj U/I operacija** tijekom izvršavanja upita. U rješenju prikazati postupak (ne samo konačni rezultat).

```
N(r) = 20 000
N(s) =  1 000
N(t) = 50 000
```

```
B(r) = 10 000
B(s) =    500
B(t) =  5 000
```

$$\sigma_{B=15 \vee C<>10}(r) \underset{D=F}{\rhd\lhd} s \cup^{B} t$$

```
veličina n-torke(r) = 0.3 blokova
veličina n-torke(s) = 0.2 blokova
veličina n-torke(t) = 0.6 blokova
```

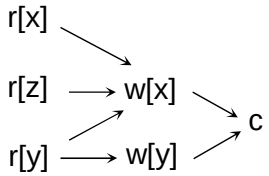
(5 bodova)

6. Obavlja se prirodno spajanje relacija r i s koje su pohranjene u B(r), odnosno B(s) blokova.
   a) Objasniti na koji se način obavlja fizička operacija particioniranog spajanja pomoću raspršenog adresiranja (*partitioned hash join*).
   b) Navesti izraz (formulu) kojom se procjenjuje trošak te operacije. Objasniti na koji je način taj izraz izveden. *(4 boda)*

7. SUBP periodički (npr. svakih 10 minuta) određuje kontrolnu točku. Pri tome obavlja pet koraka: u prvom koraku SUBP prestaje obavljati operacije koje zahtijevaju pisanje u međuspremnike podataka ili dnevnika, dok u petom koraku nastavlja s uobičajenim radom. Navesti što se u okviru kontrolne točke obavlja u drugom, trećem i četvrtom koraku. Zatim objasniti zašto nije dopušteno zamijeniti redoslijed drugog i trećeg koraka, te zašto nije dopušteno zamijeniti redoslijed trećeg i četvrtog koraka. *(4 boda)*

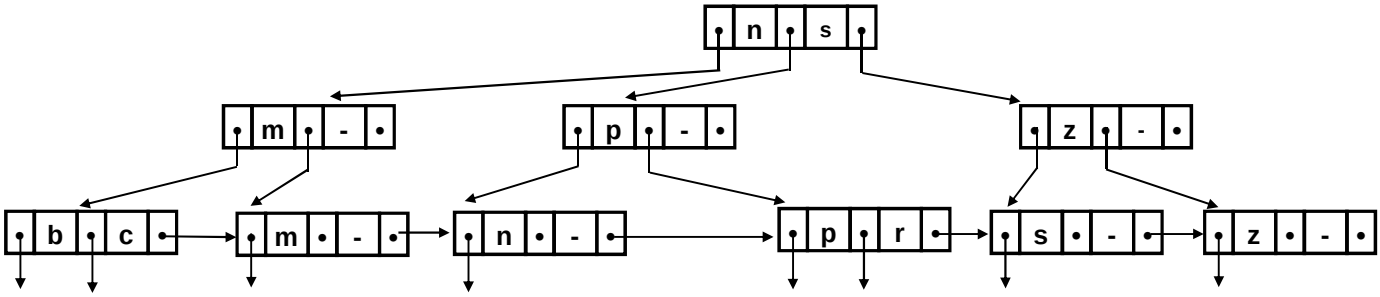
8. Na slici 4. grafom je prikazan model transakcije T. Navesti koje od ucrtanih lúkova graf mora sadržavati prema definiciji modela transakcije. Objasniti koje je značenje preostalih lúkova u grafu. *(4 boda)*

Slika 4.

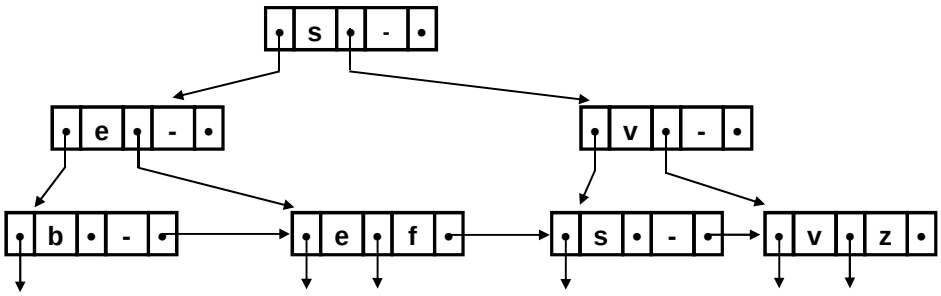


Rješenja odabranih zadataka:

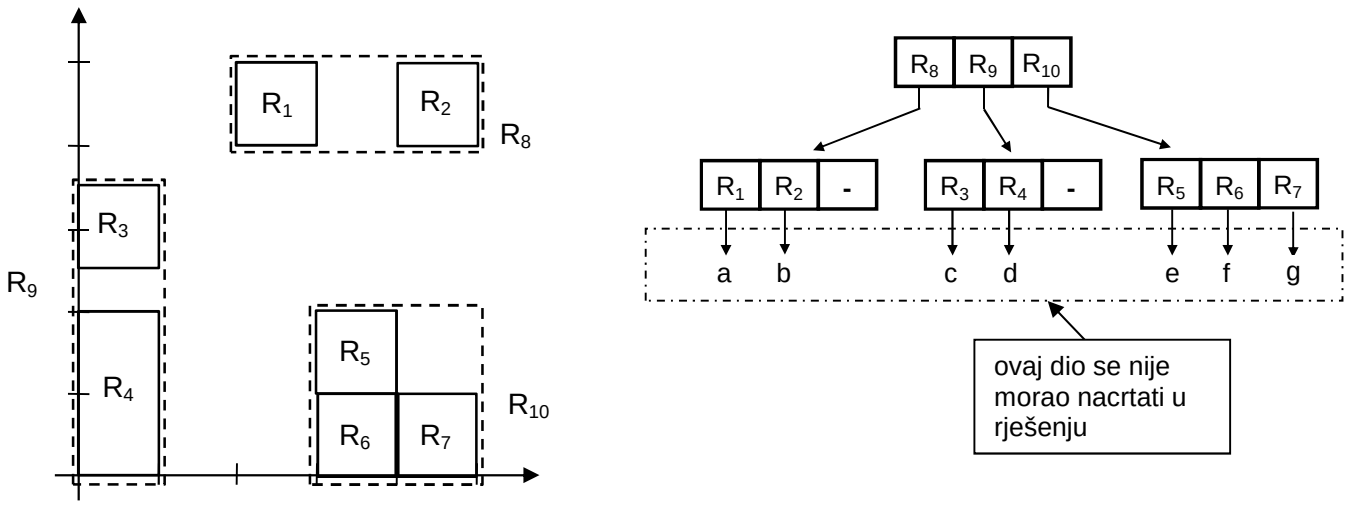
1. a)



b)



2.a)



b) Odrediti minimalni granični okvir (MBR) za zadanu liniju (npr. MBR_x). U korijenu stabla pronaći sve MBR-ove koji se sijeku s MBR_x. To su R_8 i R_9. Zatim slijediti pokazivač uz R_8 i utvrditi koji je od MBR-ova R_1 i R_2 u presjeku s MBR_x. To je R_1. Utvrditi je li objekt *a* u presjeku sa zadanom linijom (ako jest, dodati ga u rješenje). Zatim slijediti pokazivač uz R_9 i utvrditi koji je od MBR-ova R_3 i R_4 u presjeku s MBR_x. To je R_3. Utvrditi je li objekt *c* u presjeku sa zadanom linijom (ako jest, dodati ga u rješenje).

3. Vidjeti predavanja. Usporediti vrijeme pristupa (*access time*), brzinu prijenosa (*transfer speed*), cijenu, otpornost na vanjske utjecaje, utrošak energije, ograničenje broja operacija pisanja.

4.

```
CREATE PROCEDURE razvrstaj()
   DEFINE p_rbr LIKE ulazna.rbr;
   DEFINE lijevaDesna INTEGER;
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);

   ON EXCEPTION SET sqle, isame, errdata
      ROLLBACK WORK;
      RAISE EXCEPTION sqle, isame, errdata;
   END EXCEPTION

   LET lijevaDesna = -1;
   BEGIN WORK;
   FOREACH SELECT rbr INTO p_rbr FROM ulazna ORDER BY rbr
      IF lijevaDesna = -1 THEN
         BEGIN
            ON EXCEPTION
               RAISE EXCEPTION -746, 0, 'Pogreška unos lijeva';
            END EXCEPTION
            INSERT INTO lijeva VALUES(p_rbr);
         END
      ELSE
         INSERT INTO desna VALUES(p_rbr);
      END IF
      LET lijevaDesna = - lijevaDesna;
   END FOREACH
   DELETE FROM ulazna;
   COMMIT WORK;
END PROCEDURE;
```

5. Broj n-torki

- selektivnost $f(B=15, r) = N(\sigma_{B=15}, r) / N(r) = (N(r) / V(B, r)) / N(r) = 0.5$
- selektivnost $f(C\neq10, r) = N(\sigma_{C\neq10}, r) / N(r) = (N(r) - N(\sigma_{C=10}, r)) / N(r) =$
  $= (N(r) - N(r) / V(C, r)) / N(r) = 0.8$
- $t_1 = \sigma_{B=15 \vee C\neq10}(r)$
- $N(t_1) = N(r) \cdot (1 - (1 - f(B=15, r)) \cdot (1 - f(C\neq10, r))) = 20\,000 \cdot 0.9 = 18\,000$
- $t_2 = t_1 \triangleright_{D=F} \triangleleft s$
- $N(t_2) = N(t_1) \cdot N(s) / \max(V(D, r), V(F, s)) = 18\,000 \cdot 1\,000 / 200 = 90\,000$
- $t_3 = t_2 \cup^B t$
- $N(t_3) = N(t_2) + N(t) = 140\,000$

Broj U/I operacija

- čitanje relacije r pri obavljanju $\sigma_{B=15 \vee C<>10}(r) \Rightarrow B(r) =$ **10 000**
- veličina međurezultata $B(t_1) = N(t_1) \cdot 0.3 = 5\,400$
- zapisivanje međurezultata $t_1 \Rightarrow$ **5 400**
- obavljanje $t_1 \triangleright_{D=F} \triangleleft s$: relacija s cijela stane u međuspremnike
- $\Rightarrow B(t_1) + B(s) = 5\,400 + 500 =$ **5 900**
- zapisivanje međurezultata $t_2 \Rightarrow N(t_2) \cdot (0.3 + 0.2) =$ **45 000**
- čitanje relacija t_2 i t pri obavljanju $t_3 = t_2 \cup^B t \Rightarrow B(t_2) + B(t) = 45\,000 + 5\,000 =$ **50 000**
- ukupno (bez zapisivanja konačnog rezultata)
- $\Rightarrow 10\,000 + 5\,400 + 5\,900 + 45\,000 + 50\,000$

8. Uputa za rješavanje: prema definiciji modela transakcije, međusobno poredane moraju biti operacije čitanja i pisanja **nad istim elementom**, te **sve** operacije čitanja i pisanja moraju prethoditi operaciji *abort* ili *commit*. Lúkovi koji osiguravaju navedeno, u graf moraju biti ucrtani zbog definicije modela transakcije, neovisno o semantici transakcije.

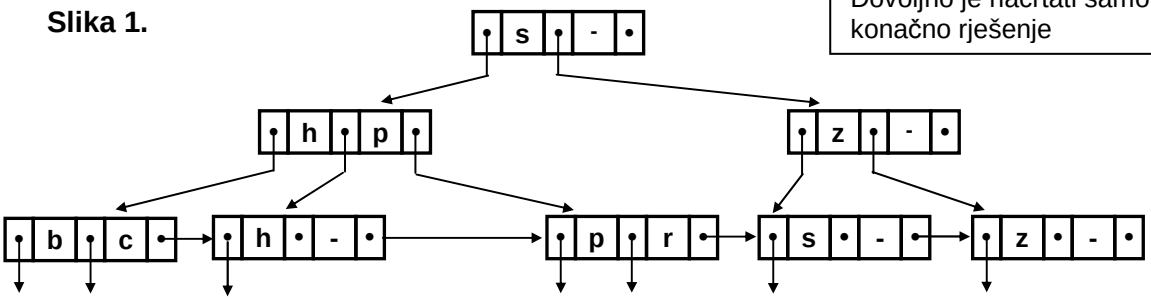
Prezime, ime

Sustavi baza podataka - Završni ispit - Odabrani zadaci
27 lipnja 2013

- *odgovore na pitanja 1 - 7 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. Nacrtati B^+-stablo nakon unosa zapisa s ključem **g** u B^+-stablo na slici 1.

Slika 1.



Dovoljno je nacrtati samo konačno rješenje

(3 boda)

2. Spajaju se relacije r i s koje su pohranjene u B(r), odnosno B(s) blokova.
   a) Objasniti na koji se način obavlja fizička operacija blokovsko spajanje s ugniježđenim petljama (*block nested-loop join*)
   b) Navesti izraze (formule) kojima se procjenjuje trošak te operacije, ovisno o raspoloživoj količini primarne memorije. Objasniti na koji su način ti izrazi izvedeni. *(6 bodova)*

3. Povijest H_1 je prikazana u obliku topološkog poretka operacija transakcija T_1, T_2 i T_3:

   H_1 $r_1[y]$ $w_1[y]$ $r_2[y]$ $w_3[x]$ $w_3[y]$ c_1 $w_3[z]$ c_3 $r_2[z]$ c_2

   a) Povijest H_1 prikazati u obliku grafa.
   b) Objasniti na koji se način utvrđuje je li povijest iz a) dijela zadatka konflikt-serijalizabilna.
   c) Koji se oblik karakterističnog problema istodobnog pristupa pojavljuje u povijesti H_1? Objasniti zašto se u povijesti H_1 pojavljuje taj problem.
   d) Uz pretpostavku da se povijest H_1 izvršava u sustavu koji koristi rigorozni 2PL protokol, u obliku niza operacija prikazati povijest koja će biti producirana (izvršena) u takvom sustavu.
   e) Uz pretpostavku da se povijest H_1 izvršava u sustavu koji koristi rigorozni 2PL protokol, ali u kojem je razina izolacije svih transakcija postavljena na READ UNCOMMITTED, u obliku niza operacija prikazati povijest koja će biti producirana (izvršena) u takvom sustavu. *(9 bodova)*

4. Povijest H_2 je prikazana u obliku topološkog poretka operacija transakcija T_1, T_2 i T_3:

   H_2 $r_1[x]$ $r_1[z]$ $r_2[x]$ $r_1[y]$ $w_2[x]$ $w_3[z]$ $w_1[y]$ $w_1[x]$ $r_2[y]$ c_1 c_2 c_3

   Sustav koristi 2PL protokol. Navesti u kojem točno trenutku izvršavanjem povijesti H_2 dolazi do potpunog zastoja te opisati kako će se potpuni zastoj razriješiti ako se povijest H_2 izvršava u sustavu koji koristi graf čekanja (WFG graf). *(5 bodova)*

5. U bazi podataka kreirana je relacija *osoba* i indeks *idx*. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: *baza podataka* → *relacija* → *n-torka*) i protokol zaključavanja indeksa, ali samo ako su za korištenje protokola zaključavanja indeksa ostvareni potrebni preduvjeti. Ne koriste se U-ključevi.

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(20)
 , prez   CHAR(20)
) LOCK MODE ROW;

CREATE UNIQUE INDEX idx
  ON osoba (sif);
```

sif	ime	prez
100	Ana	Horvat
103	Ivo	Ban
106	Jure	Ban
109	Ivo	Kolar
112	Jure	Novak
115	Ivo	Novak
118	Ana	Turk
121	Edo	Keler
125	Lino	Hertz

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	**T_2**
`BEGIN WORK; SET LOCK MODE TO WAIT; SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`	`BEGIN WORK; SET LOCK MODE TO WAIT; SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`
`1. SELECT * FROM osoba WHERE sif=100`	`2. UPDATE osoba SET prez='Zu' WHERE sif=106`
`3. UPDATE osoba SET prez='Ho' WHERE sif=100`	`4. SELECT * FROM osoba WHERE sif=112`
`5. SELECT * FROM osoba WHERE sif=100`	`6. DELETE FROM osoba WHERE sif=118`
`7. INSERT INTO osoba VALUES(118, 'Jo', 'Wu')`	`8. SELECT * FROM osoba WHERE ime='Lino'`

Za svaku naredbu (1-8) koja se uspije obaviti prije nego transakcija uđe u stanje čekanja na postavljanje ključa, napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka. Ako zaključavanje nije uspjelo navesti i razlog zašto. Rješenje treba biti u obliku prikazanom u priloženom primjeru rješenja.

(6 bodova)

Primjer oblika rješenja:

1. T_1 postavlja S na n-torke 100, 102, 102, postavlja X na n-torke 105, 107
2. T_2 postavlja IS na n-torke 103, 104, X na osoba
3. T_1 **promovira** S u IX na n-torci 102
4. T_2 ne uspijeva postaviti S na bazu podataka jer je T_2 već prije s X zaključala relaciju, T_2 **čeka**
5. T_1 uspješno promovira S u X nad bazom podataka

6. Objasniti zašto je u tablici kompatibilnosti ključeva za MGL protokol u presjeku stupca **IS** i retka **IX** oznaka *true*, te zašto je u presjeku stupca **IX** i retka **S** oznaka *false*. *(4 boda)*

7. Sustav s potpuno repliciranom bazom podataka obuhvaća čvorove S_1, S_2, ..., S_5. Odrediti parametre za *quorum consensus* protokol uz koje će taj protokol djelovati kao:

   a) protokol zaključavanja uz pomoć primarne kopije koja se uvijek nalazi u čvoru S_5

   b) većinski protokol (*majority protocol*) *(5 bodova)*

Sustavi baza podataka - Međuispit - Odabrani zadaci
23. travnja 2014.

1. Relacija *stipendija* sadrži podatke o rezultatima natječaja za stipendije. Za svakog studenta unaprijed je definiran iznos stipendije (*predvIznos*), a hoće li mu stipendija zaista biti odobrena ovisi o njegovom rangu i ukupnom raspoloživom iznosu sredstava.

```
CREATE TABLE stipendija (
   mbr          SMALLINT PRIMARY KEY,
   rang         SMALLINT NOT NULL UNIQUE,
   predvIznos   DECIMAL(8, 2) NOT NULL,
   odobrenaStip CHAR(1) DEFAULT 'N');
```

stipendija

mbr	rang	predvIznos	odobrenaStip
561	2	1000.00	N
256	4	1320.00	N
498	1	2000.00	N
578	3	1000.00	N

Stipendije se odobravaju redoslijedom određenim rangom studenta. Odobravanje stipendija se prekida kada se ukupni raspoloživi iznos sredstava potroši u cijelosti ili preostali iznos nije dovoljan da bi podmirio predviđeni iznos stipendije studenta koji je na redu za dodjelu stipendije.

Napisati pohranjenu proceduru **odobriStip** koja prima ulazni parametar *raspolozivIznos*. Procedura odobrava stipendije studentima kojima stipendija još nije odobrena tako što im vrijednost atributa *odobrenaStip* postavlja na vrijednost 'D'. Procedura kao rezultat treba vratiti nedodijeljeni iznos.

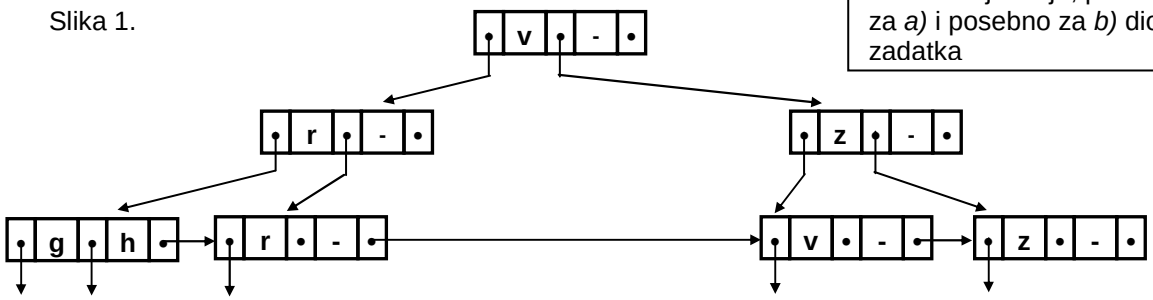
Proceduru napisati tako da se njenim izvršavanjem stipendija odobri svim studentima kojima treba biti odobrena ili niti jednom. Ako se pri obavljanju UPDATE naredbe u relaciji *stipendija* dogodi bilo koja vrsta pogreške, procedura u pozivajući program vraća pogrešku s tekstom **'Pogreška pri izmjeni**.' U slučaju pojave bilo koje druge pogreške na bilo kojem drugom mjestu u proceduri, procedura u pozivajući program mora proslijediti originalnu pogrešku. Nije dopuštena upotreba pomoćnih relacija niti SAVEPOINT mehanizma.

Primjer: za sadržaj relacije kao na slici, pozivom procedure s ulaznim parametrom 3500.00 dodijelit će se stipendija studentima s rangom 1 i 2, a nedodijeljeni iznos sredstava iznosit će 500.00. Ponovnim pozivom procedure s ulaznim parametrom 1200.00, stipendija će biti dodijeljena studentu s rangom 3, a nedodijeljeni iznos sredstava će iznositi 200.00. *(4 boda)*

2. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **a** u B^+-stablo na slici 1.

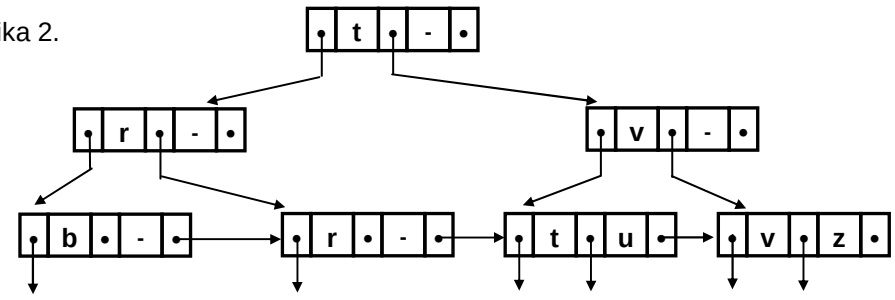
Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



   b) nakon brisanja zapisa s ključem **r** iz B^+-stabla na slici 2.

Slika 2.



(3 boda)

3. Za relaciju r(A, B) kreiran je samo indeks za atribut A. Za svaku od sljedećih operacija objasniti hoće li se za njeno izvršavanje koristiti indeks, pod kojim uvjetima i zašto.

   a) **`SELECT * FROM r WHERE A = 100 AND B = 200`**

   b) **`SELECT * FROM r WHERE A = 100 OR B = 200`** *(3 boda)*

4. Slikom prikazati postupak sortiranja relacije (datoteke) prikazane na slici 3. metodom vanjskog sortiranja s uparivanjem (*external sort-merge*). Relaciju sortirati prema prvom atributu. Broj raspoloživih blokova glavne memorije M = 3. Broj n-torki (zapisa) u bloku je 3. Označiti faze i objasniti što se obavlja u kojoj fazi, te na koji se način u kojoj fazi koriste raspoloživi blokovi glavne memorije (odnosno međuspremnici). *(4 boda)*

2	a
7	b
9	c
4	d
6	e
11	f
1	g
10	h
13	i
8	j
12	k
5	l
3	m

Slika 3.

5. Zadane su relacije r(<u>A</u>, B, C) i s(<u>D</u>, E, F).

   Primarni ključevi su potcrtani. Nema indeksa. Na raspolaganju je 102 blokova primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Kartezijev produkt se obavlja metodom *block nested-loop join*. Operacije se izvršavaju redoslijedom kako je navedeno u izrazu relacijske algebre, što znači da ne treba provoditi heurističku optimizaciju. Koristeći priložene podatke, procijeniti ukupni broj **<u>U/I operacija</u>** te broj blokova i broj n-torki u konačnom rezultatu. <u>U rješenju prikazati postupak (ne samo konačni rezultat).</u>

$$\sigma_{C=100 \wedge B>200}(r) \times \sigma_{E \text{ LIKE } 'R\%'}(s)$$

```
V(B, r) =  200
V(E, s) =   20
V(F, s) = 5000
```

```
N(r) = 200000
N(s) =  10000
```

```
B(r) = 150000
B(s) =   5000
```

```
min(B, r) =    100
max(B, r) =    500
min(C, r) =      0
max(C, r) = 400000
min(D, s) =      0
max(D, s) =  50000
```

```
veličina n-torke(r) = 0.5 blokova
veličina n-torke(s) = 0.2 blokova
```

(*5 bodova)*

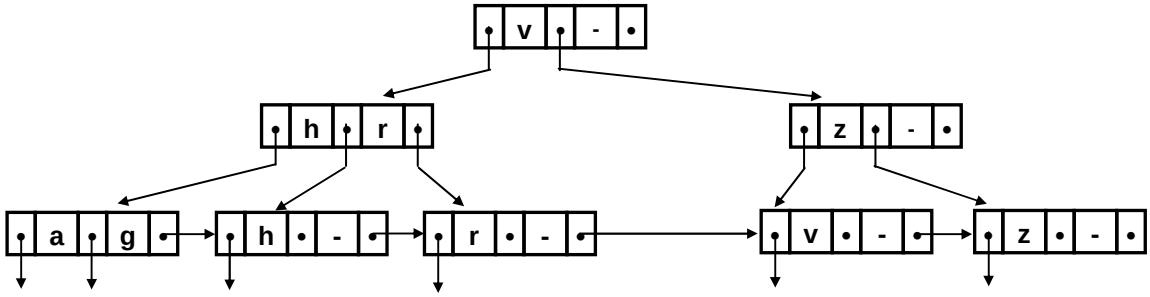
6. a) Napisati definiciju striktnog parcijalnog poretka.
   b) Napisati definiciju modela transakcije. *(4 boda)*

Rješenja odabranih zadataka:

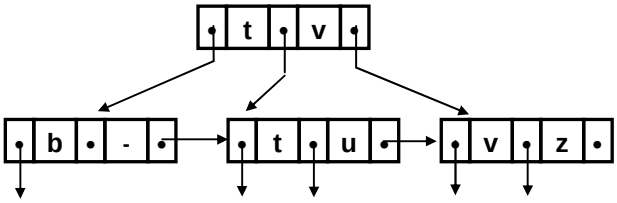
1.

```
CREATE FUNCTION odobriStip (p_iznos DECIMAl (10, 2))
   RETURNING DECIMAl (10, 2) AS nedodijeljeno;
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);
   DEFINE p_rang LIKE stipendija.rang;
   DEFINE p_predvIznos LIKE stipendija.predvIznos;

   ON EXCEPTION SET sqle, isame, errdata
      ROLLBACK WORK;
      RAISE EXCEPTION sqle, isame, errdata;
   END EXCEPTION

   BEGIN WORK;
   FOREACH SELECT rang, predvIznos
             INTO p_rang, p_predvIznos
             FROM stipendija
            WHERE odobrenaStip = 'N'
           ORDER BY rang
      IF (p_predvIznos > p_iznos) THEN
          EXIT FOREACH;
      END IF
      LET p_iznos = p_iznos - p_predvIznos;
      BEGIN
         ON EXCEPTION
            RAISE EXCEPTION -746, 0, 'Pogreška pri izmjeni.';
         END EXCEPTION
         UPDATE stipendija
            SET odobrenaStip = 'D'
          WHERE rang = p_rang;
      END
   END FOREACH
   COMMIT WORK;
   RETURN p_iznos;
END FUNCTION;
```

2. a)



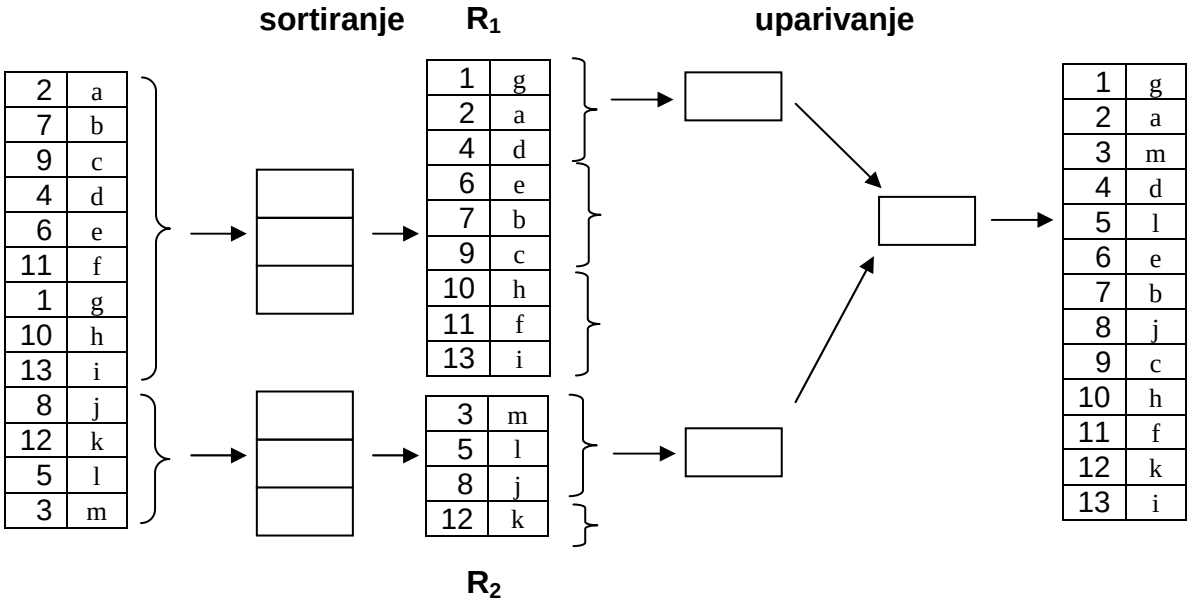
b)



3. a) *komentar odgovora*: indeks će se koristiti pod određenim uvjetima. Važno je navesti kako broj n-torki koje zadovoljavaju uvjet A=100 ne smije biti prevelik (relativno u odnosu na ukupan broj n-torki). Stupanj grupiranja (*cluster factor*) može imati utjecaj, ali odgovor u kojem bi bio naveden samo stupanj grupiranja je neispravan jer se isključivo na temelju faktora grupiranja ne može ocijeniti isplativost korištenja indeksa. Ako se indeks koristi, operacijom *index-scan* čitaju se n-torke koje zadovoljavaju uvjet A=100, a nad onima koje taj uvjet zadovoljavaju primjenjuje se uvjet B=200 (čime se ne povećava broj UI operacija).

b) *komentar odgovora*: indeks se sigurno neće koristiti. Uzaludno je korištenjem indeksa dohvatiti n-torke koje zadovoljavaju uvjet A=100, jer će se ionako za dohvat n-torki koje zadovoljavaju uvjet B=200 morati pročitati svi blokovi relacije. Korištenje indeksa u ovom slučaju predstavljalo bi samo dodatni trošak na ionako neizbježan trošak čitanja svih blokova relacije.

4.



U fazi sortiranja odjednom se čita M blokova (9 n-torki) i sortiraju, zapisuju u segment (*runfile*) R_i. Postupak se ponavlja dok se ne iscrpi cijela ulazna datoteka. Rezultat je N=2 segmenata. Budući da je N < M, uparivanje se može provesti u samo jednom koraku. Jedan blok međuspremnika koristi se za čitanje R_1, jedan za R_2, a treći blok međuspremnika se koristi za pisanje rezultata.

5. Broj n-torki

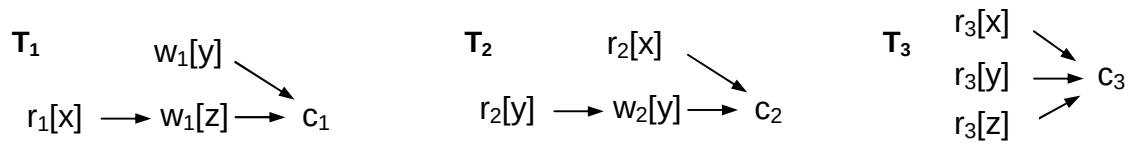
- $f(C=100, r) = N(\sigma_{C=100}(r)) / N(r) = N(r) / 10 \ / N(r) = 0.1$
- $f(B \leq 200, r) = N(\sigma_{B \leq 200}(r)) / N(r) = N(r) \cdot (200 - 100) / (500 - 100) / N(r) = 0.25$
- $f(B > 200, r) = 1 - f(B \leq 200, r) = 0.75$
- $t_1 = \sigma_{C=100 \wedge B>200}(r)$
- $N(t_1) = N(r) \cdot f(C=100, r) \cdot f(B>200, r) = 200\,000 \cdot 0.75 \cdot 0.1 = 15\,000$
- $f(E \text{ LIKE 'R\%'}, s) = N(\sigma_{E \text{ LIKE 'R\%'}}, s) / N(s) = N(s) / 5 \ / N(s) = 0.2$
- $t_2 = \sigma_{E \text{ LIKE 'R\%'}}(s)$
- $N(t_2) = N(s) \cdot f(E \text{ LIKE 'R\%'}, r) = 2\,000$
- $t_3 = t_1 \times t_2$
- $N(t_3) = N(t_1) \cdot N(t_2) = 30\,000\,000$

Broj U/I operacija

- veličina konačnog rezultata $N(t_3) \cdot (0.5 + 0.2) = 30\,000\,000 \cdot 0.7 = 21\,000\,000$
- čitanje relacije r pri obavljanju $\sigma_{C=100 \wedge B>200}(r) \Rightarrow B(r) =$ **150 000**
- veličina međurezultata $B(t_1) = N(t_1) \cdot 0.5 = 7\,500$
- zapisivanje međurezultata $t_1 \Rightarrow$ **7 500**
- čitanje relacije s pri obavljanju $\sigma_{E \text{ LIKE 'R\%'}}(s) \Rightarrow B(s) =$ **5 000**
- veličina međurezultata $B(t_2) = N(t_2) \cdot 0.2 = 400$
- zapisivanje međurezultata $t_2 \Rightarrow$ **400**
- obavljanje $t_1 \times t_2$: niti jedna relacija ne stane cijela u međuspremnike
- $\Rightarrow B(t_2)/(M-2) \cdot B(t_1) + B(t_2) = 4 \cdot 7\,500 + 400 =$ **30 400**
- ukupno (bez zapisivanja konačnog rezultata) $\Rightarrow$ 150 000 + 7 500 + 5 000 + 400 + 30400 = 193 300

Sustavi baza podataka - Završni ispit - Odabrani zadaci
17 lipnja 2014

1. Na slici su prikazani grafovi transakcija T_1, T_2 i T_3.



   a) Nacrtati graf povijesti H_1 koja obuhvaća dvije od tri prikazane transakcije i čije bi izvršavanje (kad bi se dopustilo) izazvalo anomaliju nekonzistentne analize.
   b) Nacrtati graf povijesti H_2 koja obuhvaća dvije od tri prikazane transakcije i čije bi izvršavanje (kad bi se dopustilo) izazvalo anomaliju izgubljene izmjene.
   c) Za svaku od povijesti H_1 i H_2 navesti koju minimalnu ANSI SQL razinu izolacije sustav treba koristiti kako bi se spriječilo neserijalizabilno izvršavanje povijesti.
   d) Za dvije od tri prikazane transakcije vrijedi da je konflikt-serijalizabilna (CSR) **svaka** povijest izvršavanja tih transakcija. Navesti o kojim se transakcijama radi i objasniti zašto za taj par transakcija vrijedi ova tvrdnja. *(6 bodova)*

2. U bazi podataka kreirana je relacija *osoba* i indeks *idx*. U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: *baza podataka* → *relacija* → *n-torka*) i protokol zaključavanja indeksa, ali samo ako su za korištenje protokola zaključavanja indeksa ostvareni potrebni preduvjeti. Ne koriste se U-ključevi.

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(20)
 , prez   CHAR(20)
) LOCK MODE ROW;

CREATE UNIQUE INDEX idx
  ON osoba (sif);
```

sif	ime	prez
100	Ana	Horvat
103	Ivo	Ban
106	Jure	Ban
109	Ivo	Kolar
112	Jure	Novak
115	Ivo	Novak
118	Ana	Turk
121	Edo	Keler
125	Lino	Hertz

Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1 `BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`
	T_2 `BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`

`1. DELETE FROM osoba WHERE sif=118`	`2. SELECT * FROM osoba WHERE sif=100`
`3. SELECT * FROM osoba WHERE sif=100`	`4. UPDATE osoba SET prez='Li' WHERE sif=125`
`5. INSERT INTO osoba VALUES(118, 'Jo', 'Wu')`	`6. SELECT * FROM osoba WHERE sif=125`
`7. SELECT * FROM osoba WHERE prez='Wu'`	`8. SELECT * FROM osoba WHERE sif=118`

Za svaku naredbu (1-8) koja se pokuša obaviti prije nego transakcija uđe u stanje čekanja na postavljanje ključa, napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka. Ako zaključavanje nije uspjelo navesti i razlog zašto. Rješenje treba biti u obliku prikazanom u priloženom primjeru rješenja.

(5 bodova)

Primjer oblika rješenja:
1. T_1 postavlja S na n-torke 100, 102, 102, postavlja X na n-torke 105, 107
2. T_2 postavlja IS na n-torke 103, 104, X na osoba
3. T_1 **promovira** S u IX na n-torci 102
4. T_2 ne uspijeva postaviti S na bazu podataka jer je T_2 već prije s X zaključala relaciju, T_2 **čeka**
5. T_1 uspješno promovira S u X nad bazom podataka

3. Objasniti zašto je u tablici kompatibilnosti ključeva za MGL protokol u presjeku stupca **X** i retka **IS** oznaka *false*, te zašto je u presjeku stupca **IX** i retka **IX** oznaka *true*. *(4 boda)*

4. Sustav koji koristi 2PL protokol istodobno izvršava transakcije T_1 i T_2 čiji su grafovi prikazani na slici.

   a) U obliku topološkog poretka operacija transakcija prikazati povijest H koja će izazvati potpuni zastoj u sustavu koji ne koristi ključeve za izmjenu (U-ključeve). Operacije zaključavanja i otključavanja u povijesti H nije potrebno eksplicitno navoditi.
   b) Objasniti na koji će način sustav koji, pored S-ključeva i X-ključeva, koristi i ključeve za izmjenu (U-ključeve) spriječiti pojavu potpunog zastoja tijekom izvršavanju povijesti H. *(5 bodova)*

5. Objasniti po čemu se razlikuju temeljni, striktni i rigorozni 2PL protokoli. *(4 boda)*

6. Particionirani sustav obuhvaća čvorove S_1 i S_2. Korisnici sustava koji pristupaju čvoru S_1 u najvećem broju slučajeva (ali ne isključivo) koriste podatke o osobama čiji je prihod manji ili jednak 50 000, a korisnici sustava koji pristupaju čvoru S_2 u najvećem broju slučajeva (ali ne isključivo) koriste podatke o osobama čiji je prihod veći od 50 000.

```
CREATE TABLE osoba (
   sif     INTEGER
 , imeprez CHAR(50)
 , prihod  INTEGER
 , PRIMARY KEY (sif));
```

   a) Opisati za ovaj slučaj prikladne sheme fragmentacije i alokacije. Shemu fragmentacije treba opisati u obliku izraza relacijske algebre.
   b) Što je transparentnost fragmentacije?
   c) Objasniti kako bi se u ovom slučaju pomoću SQL objekata mogla osigurati transparentnost lokacije i transparentnost fragmentacije? *(4 boda)*

Sustavi baza podataka - Međuispit - Odabrani zadaci
27. travnja 2015.

- *odgovore na pitanja 1 - 6 napisati na vlastitim listovima papira. Netočni odgovori na ova pitanja ne donose negativne bodove.*

1. U bazi podataka su kreirane relacije *racun* i *stavka*. Relacije su prazne.

```
CREATE TABLE racun (
  brRac     INTEGER,
  PRIMARY KEY (brRac));

CREATE TABLE stavka (
  brRac     INTEGER,
  rbrStavka INTEGER,
  PRIMARY KEY
   (brRac, rbrStavka),
  FOREIGN KEY (brRac)
    REFERENCES racun (brRac));
```

   Napisati pohranjenu proceduru **generiraj** kojom će se u relacije upisati podaci za testiranje. Procedura u relaciju *racun* treba pokušati upisati ukupno 100 n-torki s vrijednostima atributa *brRac* od 1 do 100, a za svaku takvu n-torku u relaciji *racun* upisati točno 10 pripadnih n-torki u relaciju *stavka*, s vrijednostima za atribut *rbrStavka* od 1 do 10.

   Proceduru napisati tako da bude osigurano sljedeće:
   ako je u relaciju *racun* upisana n-torka za neki račun, tada u relaciju *stavka* moraju biti upisane sve n-torke koje pripadaju tom računu (to ne znači da u relacije mora biti upisano ili svih 100+1000 n-torki ili niti jedna - vidjeti primjer).

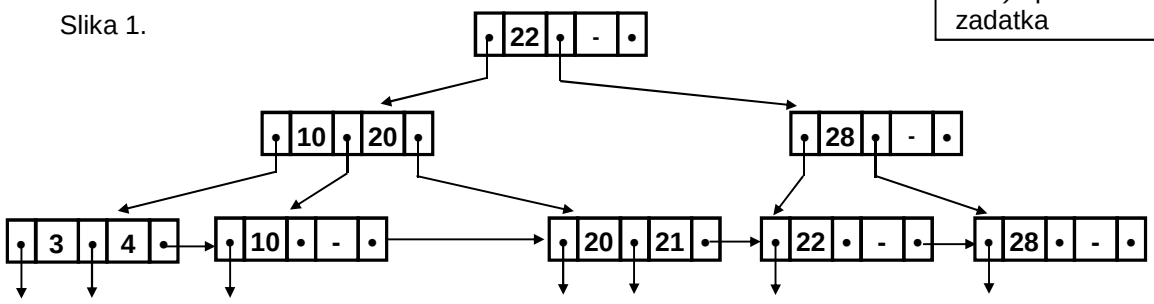
   Ako se pri obavljanju operacije unosa u relaciju *stavka* dogodi bilo koja vrsta pogreške, procedura u pozivajući program vraća pogrešku s tekstom **'Pogreška zbog stavke**'. U slučaju pojave bilo koje druge pogreške na bilo kojem drugom mjestu u proceduri, procedura u pozivajući program mora proslijediti originalnu pogrešku. Nije dopuštena upotreba pomoćnih relacija, okidača (*trigger*) niti SAVEPOINT mehanizma.

   Primjer: prikazana su tri od ukupno stotinu i jednog mogućeg ishoda poziva procedure
   - obje relacije ostale su prazne
   - u relaciji *racun* je 75 n-torki (brojevi računa 1-75), a u relaciji *stavka* 750 n-torki (za svaki broj računa od 1-75 točno po 10 stavki s rednim brojevima od 1-10). **Uočite:** nije dopušteno da se nakon završetka procedure u relaciji *racun* nalazi 75 n-torki, a u relaciji *stavka* 749 n-torki.
   - u relaciji *racun* je 100 n-torki, a u relaciji *stavka* 1000 n-torki (4 boda)

2. Nacrtati B^+-stablo:
   a) nakon unosa zapisa s ključem **9** u B^+-stablo na slici 1.

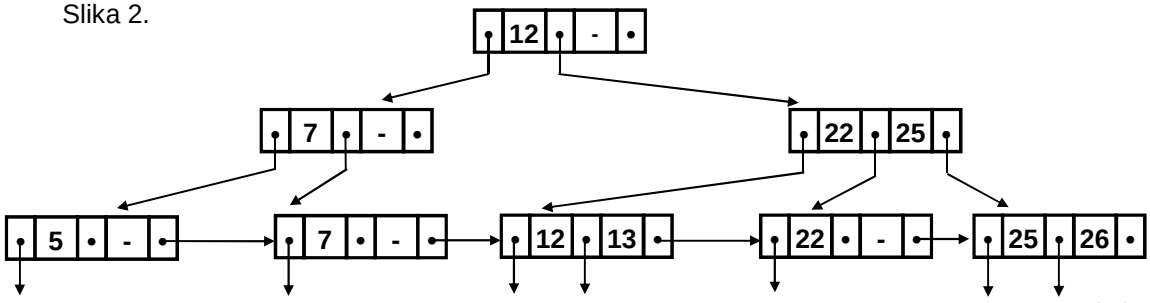
Dovoljno je nacrtati samo konačna rješenja, posebno za *a)* i posebno za *b)* dio zadatka

Slika 1.



   b) nakon brisanja zapisa s ključem **7** iz B^+-stabla na slici 2.

Slika 2.



(4 boda)

3. a) Objasniti što je *cluster index* i što je stupanj grupiranja (*cluster factor*)
   b) Objasniti kada i zašto optimizator koristi informaciju o stupnju grupiranja *(4 boda)*

4. a) Definirati svojstva transakcije *atomarnost* i *izdržljivost*
   b) Navesti i objasniti tri vrste pogrešaka u sustavu za upravljanje bazama podataka (SUBP). Pojedinačno, za svaku vrstu pogreške navesti koje mehanizme SUBP koristi da bi osigurao svojstvo *atomarnosti*, a koje mehanizme koristi da bi osigurao svojstvo *izdržljivosti*

   (4 boda)

5. Zadane su relacije r(A, B, C, D) i s(E, F).

   Primarni ključevi su potcrtani. Nema indeksa. Na raspolaganju je 1002 blokova primarne memorije. Svi međurezultati se materijaliziraju (zapisuju u sekundarnu memoriju). Za operaciju spajanja uz uvjet koristi se spajanje ugniježđenim petljama (*nested-loop join*). Operacije se izvršavaju redoslijedom kako je navedeno u izrazu relacijske algebre, što znači da ne treba provoditi heurističku optimizaciju. Izvršava se sljedeća operacija relacijske algebre:

$$\sigma_{B \leq 1000 \vee C=20}(r) \underset{D = F}{\triangleright\!\triangleleft} s$$

```
V(B, r) =  200
V(C, r) =    5
V(D, r) = 1000
V(F, s) =  100
```

```
N(r) =  5000
N(s) =  1500
```

```
B(r) =  2000
B(s) =  1200
```

```
min(B, r) =   500
max(B, r) =  2500
min(C, r) =     0
max(C, r) =   100
min(D, r) =   200
max(D, r) =   400
min(F, s) =    50
max(F, s) =   250
```

```
veličina n-torke(r) = 0.2 blokova
veličina n-torke(s) = 0.5 blokova
```

   Koristeći priložene podatke iz rječnika podataka, za zadanu operaciju relacijske algebre:

   a) prvo procijeniti broj n-torki u međurezultatima i konačnom rezultatu

   b) zatim procijeniti ukupni broj **U/I operacija** tijekom izvršavanja operacije relacijske algebre i broj blokova u konačnom rezultatu.

   U rješenju prikazati postupak (ne samo konačni rezultat). (*4 boda)*

6. Transakcija T je opisana sljedećim pseudokôdom:

```
begin work;
   read(x, p1);
   read(y, p2);
   read(z, p3);
   p4 ← p1 + p2 + p3;
   write(y, p4);
   write(z, 3.14); /* zapisuje se konstantna vrijednost */
commit work;
```

   a) nacrtati graf transakcije. U graf ucrtati isključivo one lúkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije
   b) svaki lûk u grafu označiti proizvoljno odabranim rednim brojem, te referencirajući se na te brojeve, pojedinačno za svaki lûk navesti zašto je ucrtan u graf transakcije

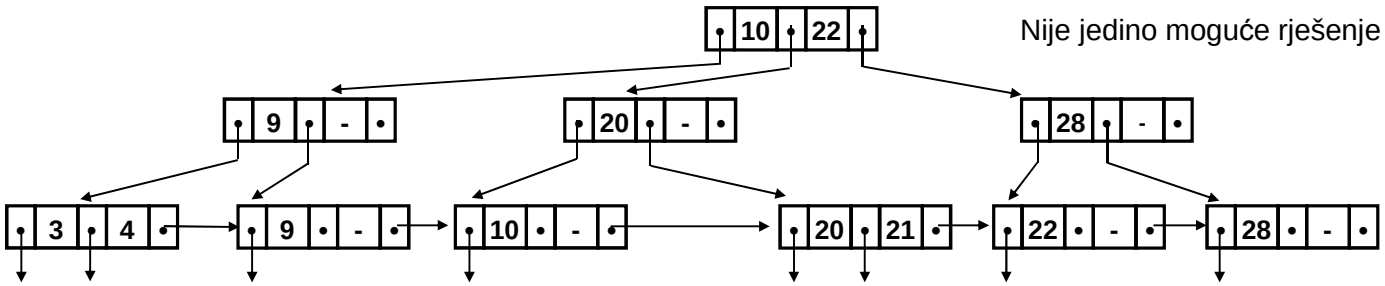
   (3 boda)

Rješenja:

1.

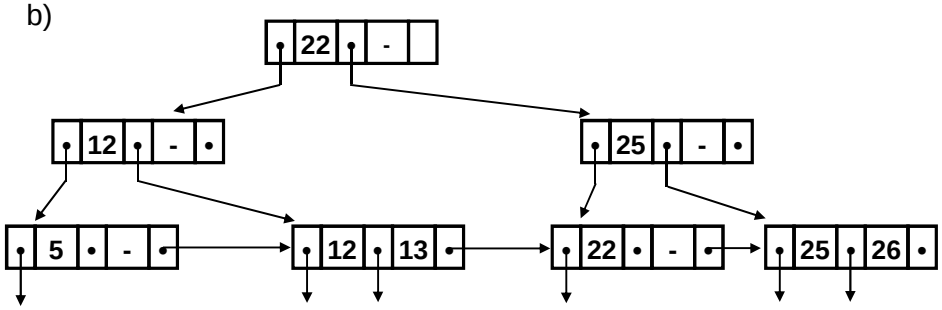
```
CREATE PROCEDURE generiraj ()
   DEFINE pBrRac, pRbrStavka INTEGER;
   DEFINE sqle, isame INTEGER;
   DEFINE errdata CHAR(80);
   FOR pBrRac = 1 TO 100
      ON EXCEPTION SET sqle, isame, errdata
         ROLLBACK WORK;
         RAISE EXCEPTION sqle, isame, errdata;
      END EXCEPTION
      BEGIN WORK;
      INSERT INTO racun VALUES (pBrRac);
      FOR pRbrStavka = 1 TO 10
         ON EXCEPTION
            RAISE EXCEPTION -746, 0, 'Pogreška pri unosu stavke.';
         END EXCEPTION
         INSERT INTO stavka VALUES (pBrRac, pRbrStavka);
      END FOR
      COMMIT WORK;
   END FOR
END PROCEDURE;
```

2. a)



Nije jedino moguće rješenje

b)



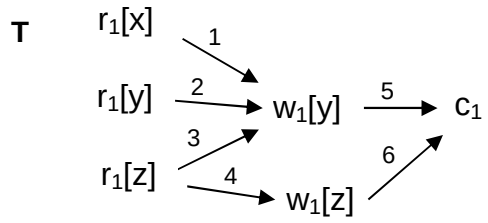
5. a) Broj n-torki

- $f(B \leq 1000, r) = N(\sigma_{B \leq 1000}(r)) / N(r) =$
  $= N(r) \cdot (v - min(B, r)) / (max(B, r) - min(B, r)) / N(r) =$
  $= N(r) \cdot (1000 - 500) / (2500 - 500) / N(r) = 0.25$
- $f(C=20, r) = N(\sigma_{C=20}(r)) / N(r) = N(r) / V(C, r) / N(r) = 1 / 5 = 0.2$
- $f(B \leq 1000 \vee C=20, r) = 1 - (1- f(B \leq 1000, r) \cdot (1- f(C=20, r)) =$
  $= 1 - 0.75 \cdot 0.8 = 1 - 0.6 = 0.4$
- $t_1 = \sigma_{B \leq 1000 \vee C=20} (r)$
- $N(t_1) = N(r) \cdot f(B \leq 1000 \vee C=20, r) = 5\,000 \cdot 0.4 = 2\,000$
- $t_2 = t_1 \rhd\lhd_{D=F} s$
- $N(t_2) = N(t_1) \cdot N(s) / max(V(D, r), V(F, s)) = 2\,000 \cdot 1\,500 / 1000 = 3\,000$

b) Broj U/I operacija

- veličina konačnog rezultata $B(t_2) = N(t_2) \cdot (0.2 + 0.5) = 3\,000 \cdot 0.7 = 2\,100$
- čitanje relacije r pri obavljanju $\sigma_{B \leq 1000 \vee C=20}(r) \Rightarrow B(r) =$ **2 000**
- veličina međurezultata $B(t_1) = N(t_1) \cdot 0.2 = 2\,000 \cdot 0.2 = 400$
- zapisivanje međurezultata $t_1 \Rightarrow$ **400**
- obavljanje $t_1 \rhd\lhd_{D=F} s$
  u ovom slučaju, bez obzira koristi li se *nested-loop* ili *block nested-loop*, zato jer t_1 cijela stane u međuspremnik $\Rightarrow B(t_1) + B(s) = 400 + 1\,200 =$ **1 600**
- ukupno (bez zapisivanja konačnog rezultata) $\Rightarrow 2\,000 + 400 + 1\,600 =$
  $= 4\,000$

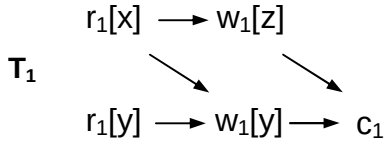
6. a)



b) 1 - y se izračunava na temelju x
2 - y se izračunava na temelju y; r i w operacije nad istim elementom moraju biti poredane
3 - y se izračunava na temelju z
4 - r i w operacije nad istim elementom moraju biti poredane
5 - sve operacije moraju biti poredane u odnosu na operaciju *commit/abort*
6 - sve operacije moraju biti poredane u odnosu na operaciju *commit/abort*

Sustavi baza podataka - Završni ispit
23 lipnja 2015

1. Na slici je prikazan graf transakcije T_1. U sustavu se istodobno izvršava još samo transakcija T_2 koja je identična transakciji T_1.



   a) nacrtati **graf** povijesti H koja obuhvaća transakcije T_1 i T_2 i čije bi izvršavanje (kad bi se dopustilo) izazvalo anomaliju izgubljene izmjene.
   b) uz koje ANSI SQL razine izolacije bi sustav dopustio izvršavanje nepromijenjene povijesti H? Objasnite zašto. *(5 bodova)*

2. U bazi podataka kreirana je relacija *osoba* i indeks *idx1*.

   U relaciju su upisane samo n-torke prikazane na slici (sadržaj relacije je važan). SUBP koristi MGL protokol (uz hijerarhiju objekata: *baza podataka* → *relacija* → *n-torka*) i protokol zaključavanja indeksa, ali samo ako su za korištenje protokola zaključavanja indeksa ostvareni potrebni preduvjeti. Ne koriste se U-ključevi.

```
CREATE TABLE osoba (
   sif    INTEGER
 , ime    CHAR(20)
 , prez   CHAR(20)
) LOCK MODE ROW;

CREATE UNIQUE INDEX idx1
  ON osoba (sif);
```

sif	ime	prez
100	Ana	Horvat
103	Ivo	Ban
106	Jure	Ban
109	Ivo	Kolar
112	Jure	Novak
115	Ivo	Novak
118	Ana	Turk
121	Edo	Keler
125	Lino	Hertz

   Redoslijed kojim SQL naredbe pristižu u sustav u skladu je s rednim brojevima naredbi. U sustav pristigla SQL naredba obavlja se odmah i u cijelosti (ako nije bilo zapreka za postavljanje potrebnih ključeva).

T_1	**T_2**

`BEGIN WORK;`
`SET LOCK MODE TO WAIT;`
`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`
	`BEGIN WORK;`
	`SET LOCK MODE TO WAIT;`
	`SET TRANSACTION ISOLATION LEVEL SERIALIZABLE`

`1. SELECT * FROM osoba WHERE sif=100`	`2. SELECT * FROM osoba WHERE sif=115`
`3. UPDATE osoba SET prez='Li' WHERE sif=100`	`4. DELETE FROM osoba WHERE sif=115`
`5. UPDATE osoba SET prez='Wu'` `WHERE prez='Ban'`	`6. SELECT * FROM osoba WHERE ime='Ivo'`

   a) za svaku naredbu (1-6) koja se pokuša obaviti prije nego transakcija uđe u stanje čekanja na postavljanje ključa, napisati kojom vrstom ključa se pokušava zaključati koji element baze podataka. Ako zaključavanje nije uspjelo navesti i razlog zašto. Rješenje treba biti u obliku prikazanom u priloženom primjeru rješenja.

Primjer oblika rješenja:
1. T_1 postavlja S na n-torke 100, 102, 103, postavlja X na n-torke 105, 107
2. T_2 postavlja IS na n-torke 103, 104, X na osoba
3. T_1 **promovira** S u IX na n-torci 102
4. T_2 ne uspijeva postaviti S na bazu podataka jer je T_2 već prije s X zaključala relaciju, T_2 **čeka**
5. T_1 uspješno promovira S u X nad bazom podataka

   b) ponoviti zadatak pod a), ali uz pretpostavku da nad relacijom osoba nije kreiran niti jedan indeks. *(5 bodova)*

3. Objasniti zašto menadžer zaključavanja (*lock manager*) zahtjeve transakcija za zaključavanjem istog elementa baze podataka mora odobravati u skladu s redoslijedom kojim ti zahtjevi pristižu u sustav (*first come - first served*). *(3 boda)*

4. Sustav koji koristi 2PL protokol istodobno izvršava transakcije T_1, T_2 i T_3. Transakcije su prikazane u obliku topoloških poredaka:

   T_1: $r_1[x]$, $r_1[y]$, $w_1[x]$, c_1

   T_2: $w_2[y]$, $w_2[x]$, c_2

   T_3: $r_3[x]$, $w_3[x]$, c_3

   a) prikazati povijest u obliku topološkog poretka koja obuhvaća T_1, T_2 i T_3 u kojoj kao posljednje tri operacije moraju biti navedene operacije c_1, c_2, c_3. Pri tome, povijest treba biti takva da bi pokušaj izvršavanja u sustavu koji ne koristi U-ključeve izazvao potpuni zastoj, a pokušaj izvršavanja (te iste povijesti) u sustavu koji koristi U-ključeve ne bi izazvao potpuni zastoj.
   b) nacrtati WFG graf u trenutku kada se izvršavanjem povijesti iz a) dijela zadatka dogodi potpuni zastoj.
   c) u obliku topološkog poretka prikazati povijest koju će na temelju ulazne povijesti iz a) dijela zadatka producirati sustav koji koristi U-ključeve i rigorozni 2PL protokol. U topološkom poretku također prikazati i operacije zaključavanja i otključavanja. *(5 bodova)*

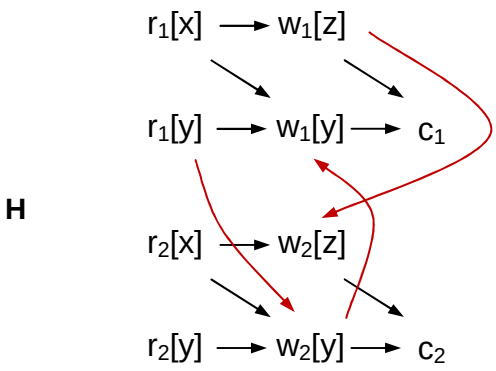
5. Objasniti zašto protokol vremenskih oznaka (*timestamp-ordering*) nikad ne izaziva potpuni zastoj. *(3 boda)*

6. Što znači da je neki protokol *nezavisan s obzirom na mogućnost obnove*? Objasniti u kojem koraku izvršavanja, zašto i s kojom posljedicom 2PC protokol pokazuje kako ne posjeduje to svojstvo. *(3 boda)*

7. a) objasniti prednosti i nedostatke korištenja protokola zaključavanja uz pomoć primarne kopije (*primary-copy locking*) u sustavu s repliciranom bazom podataka.
   b) distribuirani sustav s potpuno repliciranom bazom podataka obuhvaća čvorove S_1, S_2, ..., S_{10}. Odrediti parametre za *quorum consensus* protokol uz koje će taj protokol djelovati kao protokol zaključavanja uz pomoć primarne kopije koja se uvijek nalazi u čvoru S_{10}. *(4 boda)*

RJEŠENJA ODABRANIH ZADATAKA

1. a)



orijentacija luka $w_1[z]$ - $w_2[z]$ u ovom slučaju ne utječe na ispravnost rješenja, ali luk mora biti ucrtan

b) Isolation Level Read Uncommitted i Isolation Level Read Committed (+ objasniti zašto)

2. a)

1. IS na b.p. - da; IS na relaciju osoba - da; S na zapis indeksa sif=100 - da; S na n-torku sif=100 - da
2. IS na b.p. - da; IS na relaciju osoba - da; S na zapis indeksa sif=115 - da; S na n-torku sif=115 - da
3. IS→IX na b.p. - da; IS→IX na relaciju osoba - da; **S** na zapisu indeksa sif=100 već postoji; S→X na n-torku sif=100 - da
4. IS→IX na b.p. - da; IS→IX na relaciju osoba - da; S→X na zapis indeksa sif=115 - da; S→X na n-torku sif=115 - da
5. IX na b.p. već postoji - da; IX→SIX (jer treba pročitati sve n-torke) na relaciju osoba odbijen zbog IX kojeg drži T_2, T_1 čeka
6. treba IS, IX na b.p. već postoji - da; treba S na relaciju osoba, S→SIX ne uspijeva zbog IX kojeg drži T_1, T_2 čeka, potpuni zastoj

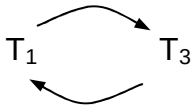
b)

1. IS na b.p. - da; S na relaciju osoba - da;
2. IS na b.p. - da; S na relaciju osoba - da;
3. IS→IX na b.p. - da; S→SIX na relaciju osoba - ne uspijeva zbog S kojeg drži T_2, T_1 čeka
4. IS→IX na b.p. - da; S→SIX na relaciju osoba - ne uspijeva zbog S kojeg drži T_2, T_1 čeka, potpuni zastoj

4. a) $r_1[x]$, $r_3[x]$, $r_1[y]$, $w_1[x]$, $w_3[x]$, $w_2[y]$, $w_2[x]$, c_1, c_2, c_3

ovo rješenje nije jedino moguće, ali u svakom ispravnom rješenju u potpuni zastoj trebaju biti uključene T_1 i T_3.

b)



c) $uL_1[x]$, $r_1[x]$, $uL_3[x]$ odbijen, $sL_1[y]$, $r_1[y]$, $xL_1[x]$, $w_1[x]$, $xL_2[y]$ odbijen, c_1, $u_1[x]$, $u_1[y]$, $uL_3[x]$, $r_3[x]$, $xL_3[x]$, $w_3[x]$, $xL_2[y]$, $w_2[y]$, $xL_2[x]$ odbijen, c_3, $u_3[x]$, $xL_2[x]$, $w_2[x]$, c_2, $u_2[y]$, $u_2[x]$

7. b) težinski faktor čvora $w_{10} = 10$, težinski faktori ostalih čvorova: $w_i = 1$, za $1 \leq i \leq 9$
za svaki element x replicirane baze podataka: $Q_r = 10$, $Q_w = 10$