Sočasni dostop do podatkovne baze

Vsaka transakcija naj bi zadoščala štirim osnovnim lastnostim:

- Atomarnost: transakcija predstavlja atomaren sklop operacij. Ali se izvede vse ali nič. Atomarnost mora zagotavljati SUPB.
- Konsistentnost: transakcija je sklop operacij, ki podatkovno bazo privede iz enega konsistentnega stanja v drugo. Zagotavljanje konsistentnosti je naloga SUPB (zagotavlja, da omejitve nad podatki niso kršene…) in programerjev (preprečuje vsebinske neskladnosti).
- Izolacija: transakcije se izvajajo neodvisno ena od druge → delni rezultati transakcije ne smejo biti vidni drugim transakcijam. Za izolacijo skrbi SUPB.
- Trajnost: učinek potrjene transakcije je trajen – če želimo njen učinek razveljaviti, moramo to narediti z novo transakcijo, ki z obratnimi operacijami podatkovno bazo privede v prvotno stanje. Zagotavljanje trajnosti je naloga SUPB.

Zaklepanje

Pomen deljenega in ekskluzivnega zaklepanja:

- Če ima transakcija deljeno zaklepanje nad neko podatkovno enoto, lahko enoto prebere, ne sme pa vanjo pisati.
- Če ima transakcija ekskluzivno zaklepanje nad neko podatkovno enoto, lahko enoto prebere in vanjo piše.
- Deljeno zaklepanje nad neko podatkovno enoto ima lahko več transakcij, ekskluzivno pa samo ena.

Postopek zaklepanja:

- Če transakcija želi dostopati do neke podatkovne enote, mora pridobiti deljeno (samo za branje) ali ekskluzivno zaklepanje (za branje in pisanje).
- Če enota ni že zaklenjena, se transakciji zaklepanje odobri.
- Če je enota že zaklenjena:
  - če hoče transakcija podatek deljeno zakleniti in če je obstoječe zaklepanje deljeno, se zaklepanje odobri, če pa hoče podatek ekskluzivno zakleniti, mora počakati (če je podatek že prej deljeno zaklenjen SAMO s strani iste transakcije, se lahko zaklepanje nadgradi).
  - če je obstoječe zaklepanje ekskluzivno, mora transakcija počakati, da se sprosti.
- Ko transakcija enkrat pridobi zaklepanje, le-to velja, dokler ga ne sprosti. To se lahko zgodi eksplicitno ali implicitno (ob prekinitvi ali potrditvi transakcije).

Nekateri sistemi omogočajo prehajanje iz deljenega v ekskluzivno zaklepanje in obratno.

Sama uporaba zaklepanja ne zagotavlja serializacije urnika, upoštevati je potrebno še dodatna pravila (protokol), kje v transakciji so lahko postavljena zaklepanja in kje se sprostijo.

2 protokola:

- **PXC** (Protocol eXclusive Commit): temelji na ekskluzivnem zaklepanju podatkov,
- **PSC** (Protocol Shared Commit): temelji na ekskluzivnem in deljenem zaklepanju.

- Pravila, ki jih uvaja PXC protokol:
  - transakcija, ki želi podatek ažurirati, ga mora najprej ekskluzivno zakleniti,
  - če zahteva po zaklepanju ne more biti takoj odobrena, preide transakcija v stanje čakanja, njeno izvajanje se nadaljuje po odobriti zaklepanja,
  - vsa zaklepanja se smejo sprostiti šele po zaključku transakcije (uspešnem ali neuspešnem),
  - Transakcija, ki želi le prebrati podatek in ji ni mar za sočasno ažuriranje tega podatka s strani kake druge transakcije, ga sme prebrati ne glede na to, ali je podatek zaklenjen ali ne.
- Dodatno vsebuje protokol PSC še naslednje pravilo:
  - Transakcija, ki želi ekskluzivno zakleniti podatek, mora imeti pred tem odobreno njegovo deljeno zaklepanje.

Opis protokolov za preprečevanje mrtve zanke (Mohorič):

Vsaki izmed transakcij pripiše SUPB ob njenem pričetku časovno oznako - njen startni čas. Na ta način je možno za poljubni dve transakciji ugotoviti katera je "starejša" in katera "mlajša". Starejša transakcija je tista, ki se že dlje časa izvaja in je zato njen startni čas manjši.

Ko transakcija TA zahteva zaklepanje podatka, ki ga je že zaklenila transakcija TB, in se njeni zahtevi zaradi nekompatibilnosti zaklepanj ne da pri priči ugoditi, se zgodi naslednje:

- po protokolu **Čakaj ali izdihni (Wait-Die)**:
  če je transakcija TA starejša od TB, preide TA v stanje čakanja na odobritev, če je mlajša, pa se njeno izvajanje prekine, transakcija se razveljavi in posreduje transakcijskemu programu v ponovno izvajanje (z istim časovnim žigom, s čimer se zagotovi, da sčasoma postane starejša (da se je ne bi venomer prekinjalo));
- po protokolu **Rani ali čakaj (Wound-Wait)**:
  če je transakcija TA starejša od TB, se prekine, razveljavi in vrne v ponovno izvajanje transakcija TB (po njeni razveljavitvi se transakciji TA odobri zaklepanje podatka), če je TA mlajša, pa preide v stanje čakanja.

Protokol Čakaj ali izdihni (Wait-Die) zagotavlja, da v primeru nekompatibilnosti zahtev po zaklepanju podatkov starejše transakcije čakajo na zaključek mlajših in s tem na sprostitev z njihove strani zaklenjenih podatkov. Tako se ne more nikoli pripetiti, da bi T0 čakala na T1, T1 na T2, ... in Tn na T0.

Po protokolu Rani ali čakaj (Wound-Wait) je čakanje na sprostitev podatkov tudi urejeno po starosti, vendar tako, da mlajše transakcije čakajo na zaključek starejših.

Po obeh protokolih so vedno mlajše transakcije tiste, katerih izvajanje se prekine, če obstaja potencialna nevarnost za nastop mrtve zanke. Da bi se ne dogajalo, da bi bila vedno ena in ista transakcija prekinjena in vrnjena v ponovno izvajanje, je potrebno poskrbeti, da vsaka transakcija prej ali slej postane starejša. *To se lahko doseže tako, da se pri ponovnem izvajanju transakcije ohrani njena prejšnja časovna oznaka*, kar pa pomeni, da mora pri dodeljevanju časovne oznake SUPB ločevati med transakcijami, ki se izvajajo prvič, in transakcijami, ki so bile razveljavljene in vrnjene v ponovno izvajanje zaradi pravil protokola.

Konstrukcija čakalnega grafa:

- WFG je usmerjen graf G = (N, E), kjer N vozlišča, E povezave.
- Postopek risanja WFG:
  - Kreiraj vozlišče za vsako transakcijo
  - Kreiraj direktno povezavo Ti → Tj, če Ti čaka na zaklepanje podatkovne enote, ki je zaklenjena s strani Tj.
- Pojav mrtve zanke označuje cikel v grafu.
- SUPB periodično gradi graf in preverja obstoj mrtve zanke.
- Ko je mrtva zanka detektirana, je potrebno eno ali več transakcij prekiniti.
- Pomembno:
  - Izbira transakcije za prekinitev: možni kriteriji: ‘starost’ transakcije, število sprememb, ki jih je transakcija naredila, število sprememb, ki jih transakcija še mora opraviti.
  - Koliko transakcije preklicati: namesto preklica cele transakcije včasih mrtvo zanko moč rešiti s preklicom le dela transakcije.
  - Izogibanje stalno istim žrtvam: potrebno preprečiti, da ni vedno izbrana ista transakcija. Podobno živi zanki (live lock)

Naloga 1:

Imamo račune R1=200 €, R2=300 € in R3=400 €. Ukazi transakcij T1 in T2 se izvedejo po naslednjem razporedu:

T1	T2
PoiščiPreberi (R1, a) a:=a-10 Ažuriraj (R1, a)	
	PoiščiPreberi (R2, c) c:=c-20 Ažuriraj (R2, c)
PoiščiPreberi (R2, b) b:=b+10 Ažuriraj (R2, b) Pomni	
	poiščiPreberi (R3, d) Pozabi

Vsota na računih R1, R2 in R3 mora po izvedbi transakcij biti enaka kot pred izvedbo T1 in T2. Če ni, opiši kaj je narobe in napiši urnik izvajanja, z uporabo protokola PXC.

Rešitev naloge 1:

- Če se T1 in T2 izvedeta v takem urniku kot je podan, bo vsota na računih na koncu naslednja:
  R1=190
  R2=300
  R3=400
  SUM= 890

  Pride do branja nepotrjenega podatka (branje neobstoječega podatka).

- Uporaba protokola PXC (ekskluzivno zaklepanje podatkov):

T1	T2
E-PoiščiPreberi (R1, a) a:=a-10 Ažuriraj (R1, a)	

	E-PoiščiPreberi (R2, c) c:=c-20 Ažuriraj (R2, c)
E-PoiščiPreberi (R2, b) *(čakanje na odobritev)*	
	E-PoiščiPreberi (R3, d) *Pozabi*
(zaseženje odobreno, ukaz se izvede) b:=b+10 Ažuriraj (R2, b) *Pomni*	

Po izvedbi transakcij T1 in T2 je vsota na računih naslednja:
R1=190
R2=310
R3=400
R1 + R2 + R3 = 900 → OK

Naloga 2:

V podatkovni bazi imamo podatke o naslednjih računih: R1=100, R2=200 in R3=300 €. Transakciji T0 prenese vrednost 10 € iz računa R1 na R2, sočasno pa se izvaja transakcija T1, ki iz R2 na R3 prenese vrednost 20€.

1. Kakšna bi bila vsota na računih R1, R2 in R3, če se transakciji izvedeta zaporedno, najprej T0 in nato T1?
2. Kakšna je vrednost na računih, če se operacije transakcij izvedejo v podanem vrstnem redu?
3. Kako se izvedejo podane operacije z uporabo deljenega zaklepanja - PSC? Komentirajte, narišite čakalni graf!
4. Kako bi se navedeni transakciji izvedli z uporabo protokola "Čakaj ali izdihni", pri čemer uporabljamo ekskluzivno zaklepanje podatkov?

T0	T1
PoiščiPreberi (R1, a) a:=a-10 Ažuriraj (R1, a) PoiščiPreberi (R2, b) b:=b+10	
	PoiščiPreberi (R2, c) c:=c-20 Ažuriraj (R2, c)
Ažuriraj (R2, b) *Pomni*	
	PoiščiPreberi (R3, d) d:=d+20 Ažuriraj (R3, d) *Pomni*

Rešitev naloge:

1. Vsota na računih bi znašala 600 €. Prikaži potek.
2. Vsota po podanem vrstnem redu bi znašala 620 €. Prikaži potek.
3. Uporaba deljenega zaklepanja, nariši čakalni graf.

T0	T1
D-PoiščiPreberi (R1, a) a:=a-10 E-Ažuriraj (R1, a) D-PoiščiPreberi (R2, b) b:=b+10	
	D-PoiščiPreberi (R2, c) c:=c-20 E-Ažuriraj (R2, c) *(čakanje na odobritev)*
E-Ažuriraj (R2, b) *(čakanje na odobritev)* ***MRTVA ZANKA!!!*** *Pomni*	
	PoiščiPreberi (R3, d) d:=d+20 Ažuriraj (R3, d) *Pomni*

4. Potek operacij z uporabo protokola "Čakaj ali izdihni":

Ko transakcija TA zahteva zaklepanje podatka, ki ga je že zaklenila transakcija TB, in se njeni zahtevi zaradi nekompatibilnosti zaklepanj ne da pri priči ugoditi, se zgodi naslednje:

* po protokolu **Čakaj ali izdihni (Wait-Die)**:
  če je transakcija TA starejša od TB, preide TA v stanje čakanja na odobritev, če je mlajša, pa se njeno izvajanje prekine, transakcija se razveljavi in posreduje transakcijskemu programu v ponovno izvajanje;

T0	T1
E-PoiščiPreberi (R1, a) a:=a-10 Ažuriraj (R1, a) E-PoiščiPreberi (R2, b) b:=b+10	
	E-PoiščiPreberi (R2, c) *(prekinitev izvajanja)*
Ažuriraj (R2, b) *Pomni*	
	PoiščiPreberi (R2, c) c:=c-20 Ažuriraj (R2, c) PoiščiPreberi (R3, d)

	d:=d+20 Ažuriraj (R3, d) *Pomni*

Naloga 3:

Izdelajte čakalni graf (WFG - Wait For Graph) za naslednji scenarij izvajanja transakcij in utemeljite, ali se zaradi sočasnega izvajanja pojavi mrtva zanka.

Transakcija	Podatkovni elementi, ki jih ima transakcija zaklenjene	Podatkovni elementi, na katere transakcija čaka
T1	x2	x3
T2	x3, x5	x2, x4, x6
T3	x4	x6, x2
T4	x6	/

Naloga 4:

Izdelajte čakalni graf (WFG - Wait For Graph) za naslednji scenarij izvajanja transakcij in utemeljite, ali se zaradi sočasnega izvajanja pojavi mrtva zanka.

Transakcija	Podatkovni elementi, ki jih ima transakcija zaklenjene	Podatkovni elementi, na katere transakcija čaka
T1	x2	x1, x3
T2	x3, x10	x7, x8
T3	x8	x4, x5
T4	x7	x1
T5	x1, x5	x3
T6	x4, x9	x6
T7	x6	x5

Naloga 5:

Za zagotovitev konsistentnosti uporablja SUPB protokol PXC, za preprečevanje mrtvih zank pa protokol “rani ali čakaj”.

X, Y in Z so naslovi zapisov v podatkovni bazi.

1. Opišite, kje pride do težav in napišite vrstni red izvajanja transakcij T0 in T1 najprej z uporabo samo protokola PXC in nato še
2. z uporabo protokola "rani ali čakaj“.
3. Razvijte tudi čakalni graf (wait-for graph - WFG).

T0	T1
PoiščiPreberi (X, x) x:=x-20 Ažuriraj (X, x) PoiščiPreberi (Y, y) y:=y+20	
	PoiščiPreberi (Z, z) z:=z+10 PoiščiPreberi(Y, y) y:=y-10 Ažuriraj(Y, y)
Ažuriraj (Y, y) PoiščiPreberi(Z, z) z:=z+20 Ažuriraj (Z, z) *Pomni*	
	z:=z+10 PoiščiPreberi (X, x) x:=x+10 Ažuriraj (X, x) Ažuriraj (Z, z) *Pomni*

* po protokolu **Rani ali čakaj (Wound-Wait)**:
če je transakcija TA starejša od TB, se prekine, razveljavi in vrne v ponovno izvajanje transakcija TB (po njeni razveljavitvi se transakciji TA odobri zaklepanje podatka), če je TA mlajša, pa preide v stanje čakanja.

<u>Rešitev naloge:</u>

1. Izgubljeno ažuriranje

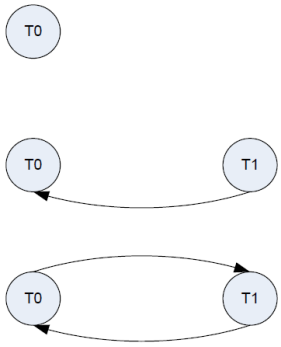
T0	T1
E-PoiščiPreberi (X, x) x:=x-20 Ažuriraj (X, x) E-PoiščiPreberi (Y, y) y:=y+20	
	E-PoiščiPreberi (Z, z) z:=z+10 E-PoiščiPreberi(Y, y) (čakanje na odobritev)

Ažuriraj (Y, y) E-PoiščiPreberi(Z, z) (čakanje na odobritev) MRTVA ZANKA	

2. Z uporabo protokola "rani ali čakaj" in PXC:

T0	T1
E-PoiščiPreberi (X, x) x:=x-20 Ažuriraj (X, x) E-PoiščiPreberi (Y, y) y:=y+20	
	E-PoiščiPreberi (Z, z) z:=z+10 E-PoiščiPreberi(Y, y) (čakanje na odobritev)
Ažuriraj (Y, y) E-PoiščiPreberi(Z, z) (prekinitev T1) (odobritev zaklepa) z:=z+20 Ažuriraj (Z, z) *Pomni*	
	E-PoiščiPreberi (Z, z) z:=z+10 E-PoiščiPreberi(Y, y) y:=y-10 Ažuriraj(Y, y) z:=z+10 E-PoiščiPreberi (X, x) x:=x+10 Ažuriraj (X, x) Ažuriraj (Z, z) *Pomni*

3. Čakalni graf:



Naloga 6:

Zapišite razpored po katerem se izvedejo transakcije T1, T2 in T3, če bi se sicer njihovi ukazi izvajali izmenoma (najprej ukaz T1, nato ukaz T2, nato ukaz T3 itd.). Transakcijo T1 sproži klic programa Sum(X, Y, Z), transakcijo T2 klic Sum(Y, Z, X), transakcijo T3 pa klic Sum(Z, X, Y), kjer so X, Y in Z naslovi zapisov v PB. Do česa pride v primeru takega razporeda? Prikažite še izvajanje z uporabo samo PXC in PSC, narišite ustrezne čakalne grafe. Nato izvedite transakcije še z uporabo protokola "Čakaj ali izdihni" in PXC.

```
Program Sum(A, B, C)
    read (A, a)
    read (B, b)
    read (C, c)
    m:=a+b+c
    write (A, m)
    Commit
```

Brez vsega:

	T1	**T2**	**T3**
t0	Sum(X, Y, Z)		
t1		Sum(Y, Z, X)	
t2			Sum(Z, X, Y)
t3	PP (X, a)		
t4		PP (Y, a)	
t5			PP (Z, a)
t6	PP (Y, b)		
t7		PP (Z, b)	
t8			PP (X, b)
t9	PP (Z, c)		
t10		PP (X, c)	
t11			PP (Y, c)
t12	m:=a+b+c		
t13		m:=a+b+c	
t14			m:=a+b+c
t13	A (X, m)		
t14		A (Y, m)	
t15			A (Z, m)
t16	Commit		
t17		Commit	
t18			Commit

PXC:

	T1	**T2**	**T3**
t0	Sum(X, Y, Z)		
t1		Sum(Y, Z, X)	
t2			Sum(Z, X, Y)
t3	E-PP (X, a)		
t4		E-PP (Y, a)	
t5			E-PP (Z, a)
t6	E-PP (Y, b) (čakanje na odobritev)		
t7		E-PP (Z, b) (čakanje na odobritev)	
t8			E-PP (X, b) (čakanje na odobritev)

Nastopi mrtva zanka.

PSC:

	T1	**T2**	**T3**
t0	Sum(X, Y, Z)		
t1		Sum(Y, Z, X)	
t2			Sum(Z, X, Y)
t3	D-PP (X, a)		
t4		D-PP (Y, a)	
t5			D-PP (Z, a)
t6	D-PP (Y, b)		
t7		D-PP (Z, b)	
t8			D-PP (X, b)
t9	D-PP (Z, c)		
t10		D-PP (X, c)	
t11			D-PP (Y, c)
t12	m:=a+b+c		
t13		m:=a+b+c	
t14			m:=a+b+c
t13	E-A (X, m) (čakanje na odobritev)		
t14		E-A (Y, m) (čakanje na odobritev)	
t15			E-A (Z, m) (čakanje na odobritev)

Nastopi mrtva zanka.

"Čakaj ali izdihni" in PXC:

	T1	T2	T3
t0	Sum(X, Y, Z)		
t1		Sum(Y, Z, X)	
t2			Sum(Z, X, Y)
t3	E-PP (X, a)		
t4		E-PP (Y, a)	
t5			E-PP (Z, a)
t6	E-PP (Y, b) (čakanje T2 na odobritev)		
t7		E-PP (Z, b) (čakanje T3 na odobritev)	
t8			E-PP (X, b) (prekinitev izvajanja)
t9		(zaklep podatka Z odobren) E-PP (X, c) (prekinitev izvajanja)	
t10	(zaklep podatka Y odobren) E-PP (Z, c)		
t11	m:=a+b+c		
t12	A (X, m)		
t13	Pomni		
t14		Sum(Y, Z, X) E-PP (Y, a) E-PP (Z, b) E-PP (X, c) m:=a+b+c A (Y, m) Pomni	
t15			Sum(Z, X, Y) E-PP (Z, a) E-PP (X, b) E-PP (Y, c) m:=a+b+c A (Z, m) Pomni

Obnavljanje podatkov po nesrečah

Naloga 1:

Za zaščito PB se uporablja *obnavljanje s senčnimi stranmi*. Stanje tekoče tabele strani pred pričetkom transakcije je prikazano na sliki. V okviru transakcije se izvedejo po vrsti naslednji ukazi:

read(A,a),
read(D,d),
write(G,g),
read(C,c),
write(E,e),
write(I,i).

Kakšna je vsebina tekoče tabele strani, če se transakcija zaključi s Pozabi, in kakšna, če se zaključi s Pomni? V kakšnem zaporedju se strani berejo z diska oziroma se izpisujejo nanj, če je v datotečnem vmesniku prostora le za dve strani hkrati, in se pri zasedenem vmesniku prepiše vedno najdalj časa neuporabljena stran?

Tekoča tabela strani

s1
s2
s3

Podatkovna baza

s1	A	B	C
s2	D	E	F
s3	G	H	I
s4			
s5			
s6			

Rešitev naloge 1:

s1	A	B	C
s2	D	E	F
s3	G	H	I
s4	G	H	I
s5	D	E	F
s6	G	H	I

Tekoča tabela strani:

s1
~~s2~~, s5
~~s3~~, ~~s4~~, s6

Datotečni vmesnik:

DV1	~~s1~~, ~~s3~~, s2
DV2	~~s2~~, ~~s1~~, s4