Eksamen Inf3100 2013, løsningsforslag

Romutstyr(rom, utstyr)

Romreservasjon(rom, tid, aktivitet, utstyr, ansvarlig)

I

(i) Et rom kan bare være reservert for én aktivitet av gangen:

rom, tid → aktivitet

En aktivitet skal på et gitt tidspunkt bare ha én ansvarlig:

aktivitet, tid → ansvarlig

(ii) Kandidatnøkler i Romreservasjon: Siden rom, tid og utstyr ikke forekommer i noen høyreside i FDene i (i), må alle disse være med i enhver kandidatnøkkel.

$(rom, tid, utstyr)^* = rom, tid, utstyr, aktivitet, ansvarlig$

Eneste kandidatnøkkel er (rom, tid, utstyr).

rom, tid → aktivitet : Venstre side er ikke en supernøkkel. (Brudd på BCNF)
Høyre side er ikke et nøkkelattributt. (Brudd på 3NF)
Venstre side er ekte inneholdt i kandidatnøkkelen. (Brudd på 2NF)
Oppfyller 1NF, bryter 2NF.

aktivitet, tid → ansvarlig: Venstre side er ikke en supernøkkel. (Brudd på BCNF)
Høyre side er ikke et nøkkelattributt. (Brudd på 3NF)
Venstre side er ikke en delmengde av kandidatnøkkelen. (Oppfyller 2NF)
Oppfyller 2NF, bryter 3NF.

Totalt bryter følgelig Romreservasjon 2NF, men oppfyller 1NF.

(Det er tilstrekkelig å fastslå at rom, tid → aktivitet bryter 2NF for å fastslå at Romreservasjon er på 1NF, men bryter 2NF.)

I

(iii) En aktivitets krav til utstyr er uavhengig av når, hvor og hvem:

aktivitet ⇒ utstyr

(og dermed også aktivitet ⇒ rom, tid, ansvarlig).

(iv)

	A	B	C	D	E	F	G	H
ABCF	a	b	c	~~d_1~~ d	~~e_1~~ e	f	~~g_1~~ g	~~h_1~~ h
ABDG	a	b	~~c_2~~	d	~~e_2~~ ~~e_1~~ e	~~f_2~~ f	g	~~h_2~~
BCDEG	~~a_3~~ a	b	c	d	e	~~f_3~~ f	g	~~h_3~~ h
CDEFH	~~a_4~~ ~~a_3~~ a	~~b_4~~	c	d	e	f	~~g_4~~ g	h

Bruker chasealgoritmen med FDene (AB → E, AE → G, CF → H, DE → AF, GH → D, H → A). Resultatet blir som vist over. Siden minst én av radene i resultatet er uten indekser, er dekomposisjonen tapsfri.

II

(i) Antall romreservasjoner pr. aktivitet:

NB!

```
create view AntallRes as
    select aktivitet, count(distinct (rom, tid)) as antres
    from Romreservasjon
    group by aktivitet;
```

De aktivitetene som har flest romreservasjoner:

```
select *
from AntallRes a
where a.antres >= all ( select antres from AntallRes);
```

(ii) Om Romreservasjon oppfyller aktivitet, tid → rom:

```
select distinct r1.aktivitet, r1.tid, r1.rom
from Romreservasjon r1, Romreservasjon r2
where r1.aktivitet = r2.aktivitet and r1.tid = r2.tid and
      r1.rom <> r2.rom;
```

II

(iii) Aktiviteter der et reservert rom ikke inneholder det nødvendige utstyret:

```
create view ManglendeUtstyr as
    select distinct r.aktivitet
    from Romreservasjon r
    where r.utstyr not in (select u.utstyr
                           from Romutstyr u
                           where u.rom = r.rom);
```

Rom som har noe av utstyret som kreves, og som derfor er potensielle kandidater:

```
create view Utstyrsoverlapp as
    select m.aktivitet, u.rom, count(distinct u.utstyr) as antTreff
    from Romreservasjon r, Romutstyr u
    where r.aktivitet in (select * from ManglendeUtstyr) and r.utstyr = u.utstyr
    group by m.aktivitet, u.rom;
```

Hvor mange forskjellige utstyrsenheter hver av aktivitetene faktisk trenger:

```
create view UtstyrsAntall as
    select aktivitet, count(distinct utstyr) as antutstyr
    from Romreservasjon r,
    where r.aktivitet in (select * from ManglendeUtstyr)
    group by r.aktivitet;
```

De aktivitetene der erstatningsrom finns:

```
create view Erstatningsrom as
    select a.aktivitet, p.rom
    from UtstyrsAntall a, Utstyrsoverlapp p
    where a.aktivitet = p.aktivitet and a.antutstyr = p.antTreff;
```

Svaret blir da:

```
select m.aktivitet, e.rom
from ManglendeUtstyr m
    left outer join
    Erstatningsrom e
    on m.aktivitet = e.aktivitet;
```

III

$$\delta\left(\Pi_{r1.aktivitet, r1.tid, r1.rom}\left(\rho_{r1}(\text{Romreservasjon}) \bowtie \rho_{r2}(\text{Romreservasjon})\right)\right)$$

$r1.aktivitet = r2.aktivitet$
and $r1.tid = r2.tid$
and $r1.rom <> r2.rom$

IV

Strict 2PL:
- Når en lås er blitt frigitt, kan ikke ytterligere låser tas.
- Alle skrivelåser skal holdes til det er klart om transaksjonen kan committe eller må aborteres

(i)

Frigivelse av låsen på a, $u_1(a)$, kan foretas når som helst i dette området

T_1: $sl_1(a); r_1(a); xl_1(b); u_1(a); r_1(b); w_1(b); u_1(b)$

T_2: $xl_2(b); r_2(b); xl_2(c); r_2(c); w_2(c); w_2(b); u_2(b); u_2(c)$

Strict 2PL krever at skrivelåser ikke frigis før etter at T_2 er ferdig, så her har vi ikke noe valg på når $u_2(c)$ skal skje (bortsett fra at vi kan ha ... $w_2(b); u_2(c); u_2(b)$).

T_3: $xl_3(a); r_3(a); sl_3(b); r_3(b); xl_3(c); u_3(b); r_3(c); w_3(a); w_3(c); u_3(a); u_3(c)$

Tilsvarende T_1 kan $u_3(b)$ gjøres når som helst etter at siste lås er tatt.

IV

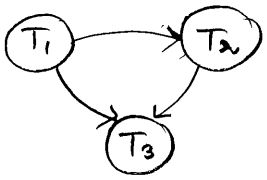
(ii)

T_1	T_2	T_3
$sl_1(a)$		
$r_1(a)$		
$xl_1(b)$		
$u_1(a)$		
$r_1(b)$		
	$xl_2(b)$- vent	
		$xl_3(a)$
		$r_3(a)$
		$sl_3(b)$- vent
$w_1(b)$		
$u_1(b)$		
	får låsen	
	$r_2(b)$	
	$xl_2(c)$	
	$r_2(c)$	
	$w_2(c)$	
	$w_2(b)$	
	$u_2(b)$	
	$u_2(c)$	
		får låsen
		$r_3(b)$
		$xl_3(c)$
		$u_3(b)$
		$r_3(c)$
		$w_3(a)$
		$w_3(c)$
		$u_3(a)$
		$u_3(c)$

IV

(iii) 2PL er <u>alltid</u> konfliktserialiserbar!

(Alternativt kan man tegne presedensgrafen:



Siden grafen er sykelfri, er planen under pkt (ii) konfliktserialiserbar.)

(iv)

T_1 : $r_1(a); r_1(b); l_1(b); w_1(b); u_1(b)$

T_2 : $r_2(b); r_2(c); l_2(c); w_2(c); l_2(b); w_2(b); u_2(c); u_2(b)$

T_3 : $r_3(a); r_3(b); r_3(c); l_3(a); w_3(a); l_3(c); w_3(c); u_3(a); u_3(c)$

(Noen besvarelser vil kanskje ha med disse, men de er ikke nødvendige)

IV

(v)

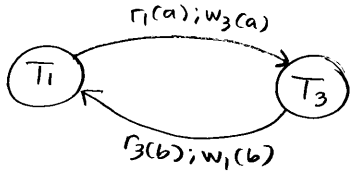
T_1	T_2	T_3	Commit(a)	Commit(b)	Commit(c)
$r_1(a)$					
$r_1(b)$					
	$r_2(b)$				
	$r_2(c)$				
		$r_3(a)$			
		$r_3(b)$			
		$r_3(c)$			
$L_1(b)$					
$W_1(b)$					
c_1				T_1	
$u_1(b)$					
	$L_2(c)$				
	$W_2(c)$				
	$L_2(b)$ – avslått; må rulles tilbake fordi T_1 er samtidig og de har felles skrivemengde {b}				
	a_2				
	$u_2(c)$				
		$L_3(a)$			
		$W_3(a)$			
		$L_3(c)$ – OK; T_2 er samtidig og de har felles skrivemengde {c}, men T_2 er avbrutt			
		$W_3(c)$			
		c_3	T_3		T_3
		$u_3(a)$			
		$u_3(c)$			

IV

(vi) Den resulterende eksekveringsplanen når T_2 er fjernet, er essensielt

$r_1(a); r_1(b); r_3(a); r_3(b); r_3(c); w_1(b); w_3(a); w_3(c)$

Presedensgrafen er slik:



Eksekveringsplanen er ikke konfliktserialiserbar siden grafen har en sykel.

V

2PC: Brukes i distribuerte systemer for å committe distribuerte transaksjoner. En av nodene tar rollen som koordinator.

Fase 1: Koordinator sender en melding til alle deltakerne og spør om de kan committe.
Hver av deltakerne avgjør om de kan committe eller må rulles tilbake, og sender melding til koordinator om dette.

Fase 2: Når koordinator har mottatt svar fra alle, eventuelt etter at en timeoutperiode er nådd:

Hvis alle svarte at de kan committe, sender koordinator melding til samtlige om at de nå skal committe.

I alle andre situasjoner sender koordinator melding til samtlige om at de må rulles tilbake.