Løsningsforslag eksamen INF3100 2016

Katalog(motortype, kompnr, kompnavn, delnr, delnavn)
Vedlikehold(Lnr, motornr, motortype, kompnr, delnr, dato)

I

1a motortype, kompnr → kompnavn

1b I oppgaven er oppgitt Lnr → motornr, kompnr, dato
motornr → motortype

Kandidatnøkkel: Alle må inneholde Lnr, delnr siden disse ikke forekommer i noen høyreside.

$(Lnr, delnr)^+$ = Lnr, delnr, motornr, kompnr, dato, motortype

$(Lnr, delnr)^+$ omfatter alle attributtene, så (Lnr, delnr) er (eneste) kandidatnøkkel.

Lnr → motornr : Venstre side ikke supernøkkel, så ikke BCNF.
Høyre side ikke nøkkelattributt, så ikke 3NF.
Venstre side inneholdt i kun, så ikke 2NF.
Bryter 2NF, er på 1NF
Dermed bryter også Vedlikehold 2NF og er bare på 1NF.

1c Hvis vi bruker MVD'en motortype, kompnr →→ delnr på eksempelinstansen, ser vi at vi må ha med tuppelet

(414, 16221, S2, K-118, 2015-11-10)

i Vedlikehold. Da oppfyller instansen regelen (2) hva gjelder Lnr 403 og 414, for da er alle delene til komponenten (S2,1) med i Vedlikehold. Men det samme er ikke tilfellet for løpenummer 415, for her mangler fortsatt komponenten (S2,2) delen S-310 (som er med i Katalog).

Så nei, det er ikke tilstrekkelig å ta med denne MVD'en for å implementere regel (2).

1d Bruker chase:

	S	T	U	V	W	X	Y	Z
TWYZ	s_1^S	t	u_1	v_1^V	w	x_1	y	z
VXYZ	$s_2^{S_1S}$	t_2	u_2^u	v	w_2^W	x	y	z
STUVX	s	t	u	v	w_3^W	x	y_3	z_3z
STWXY	s	t	u_4^u	v_4^V	w	x	y	$z_4^{z_3}z$

$(XV \to U, YZ \to S, UW \to S, SZ \to W, ST \to VZ)$

Siden siste rad er uten indekser, er dekomposisjonen tapsfri.

II (I løsningsforslagene har jeg antatt at det for hver motortype er slik at et delnummer forekommer i bare én av komponentene. Jeg har ikke tenkt gjennom hvilke endringer det ville medføre under om dette ikke var tilfellet.)

2a Deler med flere navn:

```
select k.delnr, k.motortype
from Katalog k
where k.delnr in (select k2.delnr
                  from Katalog k2
                  group by k2.delnr
                  having count(distinct k2.delnavn) > 1);
```

Eller bruk with-select:

```
with flerenavn as (
    select delnr
    from Katalog
    group by delnr
    having count(distinct delnavn) > 1)
select delnr, motortype
from Katalog
where delnr in (flerenavn);
```

Eller:

```
select distinct
  k1.delnr, k1.motortype
from Katalog k1, Katalog k2
where k1.delnr = k2.delnr
    and
    k1.delnavn <> k2.delnavn;
```

26 Deler i alle motorer:

```
select distinct k.delnr
from Katalog k
where not exists(
    (select motortype from Katalog)
    except
    (select k2.motortype
     from Katalog k2
     where k2.delnr = k.delnr)
);
```

Hvis k.delnr benyttes i alle motortyper, blir denne tom.

{select motortype from Katalog} alle motortyper

de motortypene som benytter k.delnr (fra ytre select)

Eller:

```
with totant as (
    select count(distinct motortype) as ant
    from Katalog)
select delnr
from Katalog
group by delnr
having count(motortype) in
        (select * from totant);
```

III

Alle S2-motorer der S-110 er undersøkt mer enn 10 ganger:

$$\sigma_{ant>10}(\gamma_{motornr,\ count(delnr)\rightarrow ant}(\sigma_{motortype = S2 \text{ and } delnr = S\text{-}110}(Vedlikehold)))$$

plukker ut de aktuelle tuplene

grupperer på motornr og teller antall forekomster

henter ut de hvor antallet er større enn 10

IV

Organisasjon(ansattnr, navn, stilling, leder)

Idé 1: Lager en tabell på formen Overordnede(ansattnr, navn) med alle overordnede (inklusive 205 selv, det er det enkleste). I det iterative (rekursive) steget legges det til overordnede til de som er i denne tabellen.

```
with recursive Overordnede(ansattnr,navn) as (
    basis { select ansattnr, navn
            from Organisasjon
            where ansattnr = 205
    union   -- må ha union for å ikke få flerforekomster
    rekursjon; vil stoppe når o1.leder = NULL
          { select o2.ansattnr, o2.navn
            from Organisasjon o1, Organisasjon o2, Overordnede p
            where p.ansattnr = o1.ansattnr and
                  o1.leder = o2.ansattnr
    )
select * from Overordnede where ansattnr <> 205;
```

Idé 2: Bruker en array til å holde resultatmengden. Arrayen er på formen <anr1, navn1, anr2, navn2, ...> med alle overordnede listet opp.

```
with recursive Overordnede(første, siste, liste) as
    basis { select ansattnr, leder, array[] as liste   -- lager en tom liste
            from Organisasjon
            where ansattnr = 205
    union
    rekursjon; vil stoppe når siste = NULL
          { select p.første, o.leder as siste, (p.liste || o.ansattnr || o.navn) as liste
            from Overordnede p, Organisasjon o
            where p.siste = o.ansattnr
    )
select liste
from Overordnede
where siste is NULL;
```

(Trenger jo strengt tatt ikke attributtet 'første' i Overordnede, men jeg synes det er lettere å lese hva spørringen gjør når den er tatt med.)

Eksempel som viser hvordan beregningen går:

Organisasjon

ansattnr	navn	...	leder
203	A		208
204	B		208
205	C		208
206	D		209
207	E		209
208	F		211
209	G		211
210	H		211
211	I		212
212	J		NULL

Løsning 1:

Overordnede

ansattnr	navn	
205	C	} basis
208	F	} iterasjon 1
211	I	} iterasjon 2
212	J	} iterasjon 3

Løsning 2:

Overordnede

første	siste	liste	
205	208	[]	} basis
205	211	[208, F]	} iterasjon 1
205	212	[208, F, 211, I]	} iterasjon 2
205	NULL	[208, F, 211, I, 212, J]	} iterasjon 3

V

5a

T_1 : $L_1(b); W_1(b); U_1(b)$

T_2 : $r_2(a); r_2(b); L_2(a); W_2(a); U_2(a)$

T_3 : $r_3(a); r_3(b);$

T_4 : $r_4(b); L_4(b); W_4(b); r_4(a); L_4(a); W_4(a); U_4(b); U_4(a)$

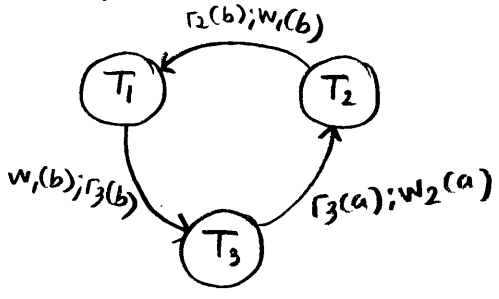
5b

	T_1	T_2	T_3	T_4
start T_2 →		$r_2(a)$		
		$r_2(b)$		
start T_4 →				$r_4(b)$
start T_1 →	$L_1(b)$			
	$W_1(b)$			
	C_1			
slutt T_1 →	$U_1(b)$			
start T_3 →			$r_3(a)$	
			$r_3(b)$	
slutt T_3 →			C_3	
		$L_2(a)$		
		$W_2(a)$		
		C_2		
slutt T_2 →		$U_2(a)$		
slutt T_4 →				$L_4(b)$ – avslått; T_4 og T_1 er samtidige og med overlappende skrivemengder a_4

5c Den resulterende planen (uten T_4) :

$r_2(a); r_2(b); L_1(b); W_1(b); U_1(b); r_3(a); r_3(b); L_2(a); W_2(a); U_2(a)$

Presedensgraf:



Siden presedensgrafen har en sykel, er planen ikke konfliktserialiserbar.

VI

6a

Fase 1: A sender <prepare T> til B og C.

B svarer <aborted T>, C svarer <prepared T>.

Fase 2: A sender <abort T> til B og C.

6b

B og C blir enige om hvem av dem som blir ny koordinator. Hvis det blir B, kan den sende <abort T> til C. Hvis det blir C, spør den B om status; når C får vite at T må aborteres, kan protokollen avsluttes.

6c

Hvis vi er i en situasjon der alle har svart <prepared T> i første fase, og koordinator går ned før den får sendt <commit T> eller <abort T>, så kan ikke de andre nodene vite hva resultatet er, og protokollen vil "henge" (den nye lederen vil ikke kunne fatte en avgjørelse).

Andre protokoller for distribuert commit:

Trefasecommit (3PC)

Paxos commit