Nicklas M. Hamang

# NICKLASH
# Obligatorisk oppgave 2
# INF3100
# V2013

Oppg1
Vareplassering (Reolnr, hyllenr, vare, antall, avdeling)
Vareplassering (A, B, C, D, E)

FD:
Reolnr -> avdeling, A->E
Vare -> avdeling, C->E
Reolnr, hyllenr -> vare, antall, avdeling, AB->CDE

i)
A->E 1NF: E er en delmengde av en av kandidat nøklene så den er ikke 2NF
C->E 2NF: C er ikke en super nøkkel så den er ikke 3NF.
AB->CDE 3NF:

Ii)
(A, B, C, D, E)
$A^+ = AE$
$R_1 = AE$, $R_2 = ABCD$.
Er ikke FD bevarende.

iii)
Innkjøp (vare, dato, antall, enhetspris, bestiller, avdeling)
Innkjøp (A, B, C, D, E, F, G)

FD'er:
A->F
E->F
A->D

iv)
Tabell som vedlegg side 8.
R(A, B, C, D, E, F, G, H)
Dekomposisjonen er tapsfri hvis FD'ene ikke skaper falske tupler.
Er ikke tapsfri ettersom den skaper falske tupler eller FD-bevarende.

Oppg 2
i)
```
SELECT v.vare, COUNT(*)
FROM vareplassering v
GROUP BY v.vare
HAVING COUNT(*) =
                    (SELECT COUNT(*)
                    FROM vareplassering
                    GROUP BY vare
                    ORDER BY COUNT(*) DESC
                    LIMIT 1);
```

ii)
```
SELECT SUM(i.enhetspris *i.antall)/sum(i.antall) ass gjennomsnitt
From innkjøp i
Where extract (year from i.dato) = 2011 AND
i.vare =’bananer’;
```

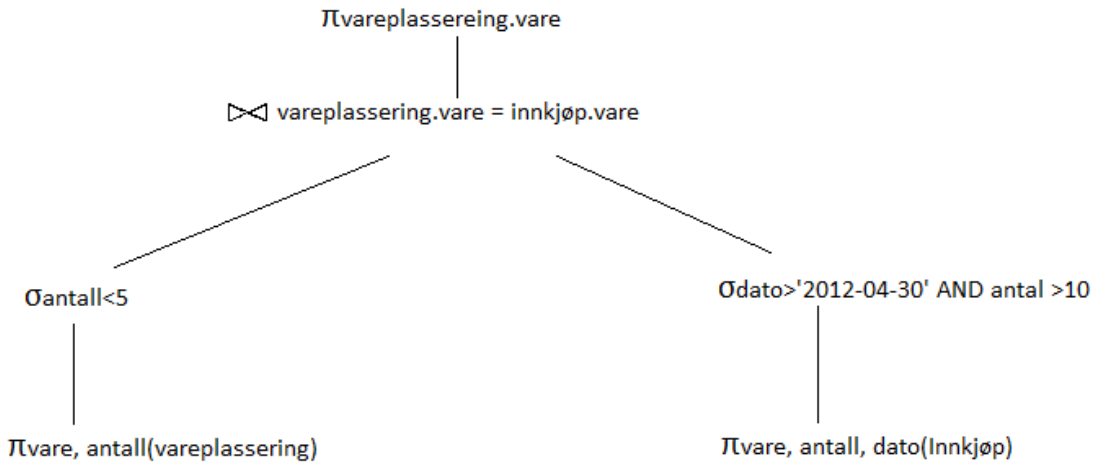
iii)
```
CREATE TEMP TABLE sum AS
SELECT v.reolnr, sum(i.enhetspris) AS sum
FROM vareplassering v, innkjøp i
WHERE v.vare = i.vare AND
v.avdelign = i.avdeling AND
v.antall = i.antall
GRUOP BY teolnr;

SELECT v.reolnr, MAX(s.sum)
FROM sum s, vareplassering v;
```
iv)
```
SELECT v.reolnr, COUNT(i.bestiller) AS antallbestill
FROM vareplassering v, innkjøp i
WHERE v.antall = i.antall AND
v.avdeling = i.avdeling AND
v.vare = i.vare AND
antallbestill = 1
ORDER BY v.reolnr;
```

v)
//Mangler løsning

Oppg 3 i)

$$\pi_{innkjøp.avdeling,vareplassering.avdeling,vareplassering.vare}\left(\sigma_{innkjøp.avdeling=vareplassering.avdeling}\left(innkjøp \bowtie_{(\text{innkjøp.vare=vareplassering.vare})} vareplassering\right)\right)$$

ii)

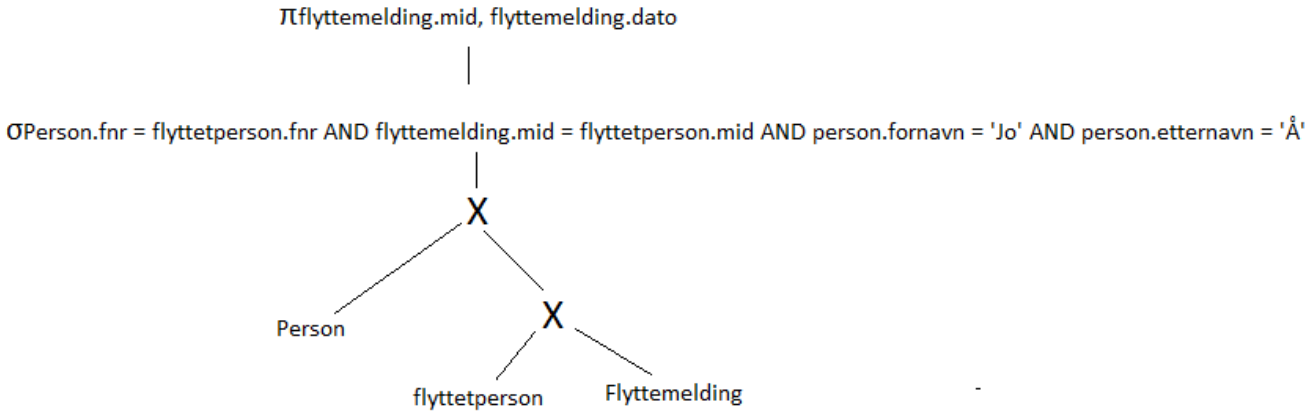



Oppg 4
i)
vedlegg side 9.

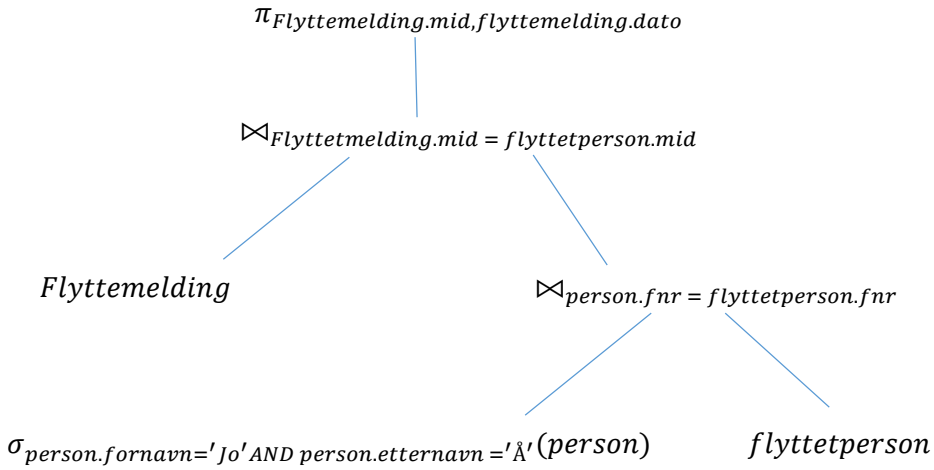
ii)
Vi kan se at det blir vranglås, ettersom T1 blir ventende på et trinn som er låst i T2 og vice versa ved bruken av eksklusive låser.

oppg 5a)
Vedlegg side 10.

oppg 5b)
i)



ii)




Oppg 5c)

I)

$Plater * 2(Spor(sektor * bytes))$

$10 * 2\big(10000(1000 * 1024)\big) = 204.8 * 10^9 bytes = 190.7\ GB$

ii)

Faktorene som trengs for å finne aksesstid er:

- Gjennomsnitt søketid.
- Gjennomsnitt rotasjonsforsinkelse.
- Overføringstid.
- Andre faktorer som for eksempel CPU-tid og diskkontroll.

Men de siste faktorene er så små at de ikke brukes i sammenligning med de andre faktorene.

Søketid er allerede gitt som 3.7ms. Rotasjonsforsinkelse er da $\left(\frac{1}{15000RPM}\right) 0{,}5 * 1000ms = 2ms$

Hvert spor har 1000 sektorer som vil si

$sektorer\ per\ spor * \big((bytes\ per\ sektor) + (bytes\ per\ gap)\big)$ gir antall bytes per spor.

$1000 * (1024 + 64) = 1.088 * 10^6$. Med dette kan vi finne antall bytes per millisekund

Som da er $\frac{bytes\ per\ spor}{ms\ per\ rotasjon} \Leftrightarrow \frac{1.088*10^6}{4ms} = 272000\ bytes/ms.$

Hver sektor er på 1024 bytes ⇔ 1KB det vil da si at det trenges 8 sektorer og 7 gap ettersom ((antall gap = (antall sektorer) - 1) dette leder til $\big((8 * 1024) + (7 * 64)\big) = 8640\ bytes$

Da er lesetid $\frac{bytes}{\frac{bytes}{ms}} = ms\ \frac{8640}{272000} = 0{,}031$ms

Hvis vi da bruker aksesstid = søketid + rotasjonsforsinkelse + lesetid ⇔ $\underline{3{,}7ms + 2ms + 0{,}031ms = 5{,}731ms.}$

iii)
Hodet til hver blokk er 20 *bytes* og hver post har et hode på 10 *bytes*.
Som da betyr: $\frac{1024-20}{10+4+64+64} = 7{,}07$.
Det vil si det er plass til 7 personer per blokk, det skal være 5.000.000 personer i databasen
$\frac{5 * 10^6}{7} = 714285{,}714$
Det tar 714285,714 sektorer for denne relasjonen, på 1024 *bytes* per sektor blir det 731428571,734 bytes eller 697,54MB
Med samme fremgangsmetode på flyttemelding gir det 2668,2MB

iv)
For å lese relasjonen trenger vi aksesstid og antall sektorer relasjonen tar. Dette ble funnet i tidligere deloppgaver. Aksesstid er 5,731ms, antall sektorer er 714285,714. $Sketorer * aksesstid$.

$$5{,}731ms * 714285{,}714 = 4093571{,}42ms = 4093{,}57s$$

v)
Hvor mange pekere og søkenøkler som får plass i hver blokk er viktig for denne oppgaven. Så om vi da har *n*-antall søkenøkler så har vi *n*+1 pekere. Siden etternavn er på 64*bytes* og jeg antar en blokk er på 1024*bytes* gir det:
$64.\mathrm{n} + 8(n+1) \leq 1024$ Med dette finner vi *n*=14.
$B^+$ treet sin høyde kan finnes ved å snu denne likningen $(\mathrm{n} * 75\%)^h = 5 * 10^6$ som gi 6.56 nivåer men jeg velger å runde opp til 7. så det vil ta 7 diskaksesser pluss diskaksessen som inneholder tuppelet. Hvis vi antar at diskaksessen leser 1024*bytes.*

$$8(5{,}731ms * 2ms + \left(\frac{1024bytes}{272000\frac{Bytes}{ms}}\right) = 45{,}6ms$$

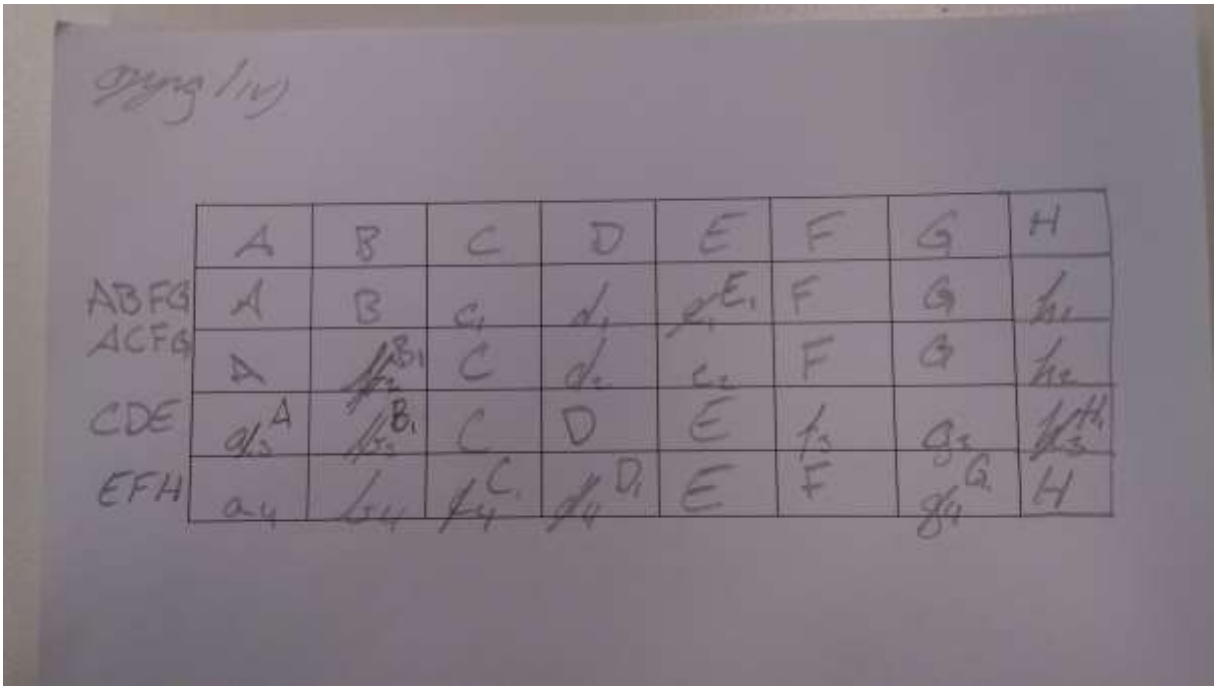
Oppg 6
a)
i) Vedlegg side 11.
ii) Det leder nok til en deadlock når $T_1$ prøver å få tilgang til et element som er i bruk på $T_2$

b) hvis vi bruker «snapshot» på S vil da $T_3$ abortert når den kommer til punktet for å oppdatere B ettersom $T_2$ Allerede har commitet den. Dette vil da også skje på $T_2$ med C.

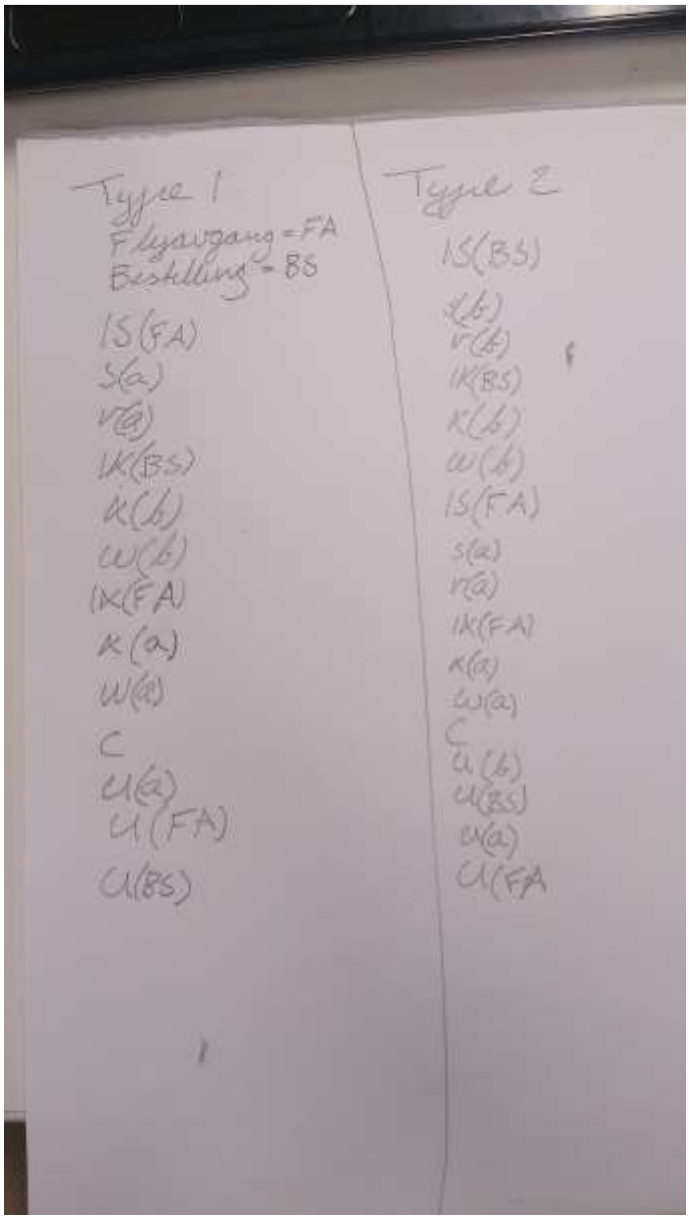
c)
i)
<$T_2$, Start>
<$T_2$ B, 5, 15>
<$T_2$ C, 0, 5>

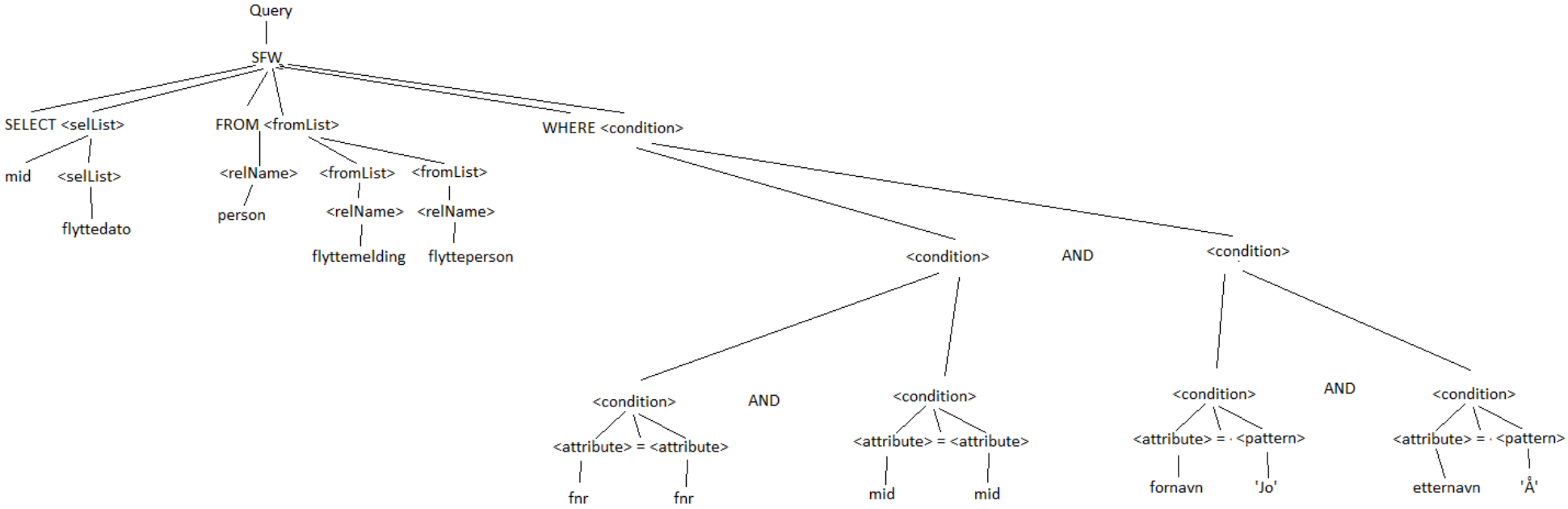
ii)
undo skriver til primærminnet før den skriver til disk. Alle logglinjer som gjelder for X skrives til disk før X endres og alt må skrives til disk før commit. Write blir skrevet før operasjonen utføres.

I redo skrives en linje med den nye verdien til X i loggen for hver operasjon. Loggen skrives til disk under commit.



| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| ABFG | A | B | | | | F | G | |
| ACFG | A | | C | | | F | G | |
| CDE | | | C | D | E | | | |
| EFH | | | | | E | F | | H |


Type 1
Flyavgang = FA
Beställning = BS
IS(FA)
S(a)
V(a)
IK(BS)
K(b)
W(b)
IK(FA)
K(a)
W(a)
C
U(a)
U(FA)
U(BS)
Type 2
IS(BS)
K(b)
V(b)
IK(BS)
K(b)
W(b)
IS(FA)
S(a)
V(a)
IK(FA)
K(a)
W(a)
C
U(b)
U(BS)
U(a)
U(FA


Query
SFW
SELECT <selList>
FROM <fromList>
WHERE <condition>
mid
<selList>
flyttedato
<relName>
person
<fromList>
<relName>
flyttemelding
<fromList>
<relName>
flytteperson
<condition>
AND
<condition>
<condition>
AND
<condition>
<attribute> = <attribute>
fnr
fnr
AND
<condition>
<attribute> = <attribute>
mid
mid
<condition>
<attribute> = <pattern>
fornavn
'Jo'
AND
<condition>
<attribute> = <pattern>
etternavn
'Å'


T1
T2
T3