TDT4145 Øving 5 Sander Lindberg

Oppgave 1

a) Her ville en heapfil vært best. Det er bare å returnere alt med en gang. Med et Clustered B+ tre måtte vi først nå gått igjennom 3 blokker, deretter returnert.

b) Ville brukt nr 4). Selvom ikke alle "bøttene" du aksesserer nødvendigvis har postene du leter etter, er de i nærheten. Grunnen til at jeg ville brukt 4) ovenfor 2) er at 4) ~~har en~~ gjennomsnittlig aksesserer 1.25 blokker, og B+ treet vil være mindre effektivt.

c) Her fungerer 3) bra. Postene er allerede sortert på (EksamenNR, StudentNr), så vi kan bare hente ut postene kallenes i treet.

4) Heapfil igjen. Den er ikke sortert på noen måte, så det er bare å legge rett inn.

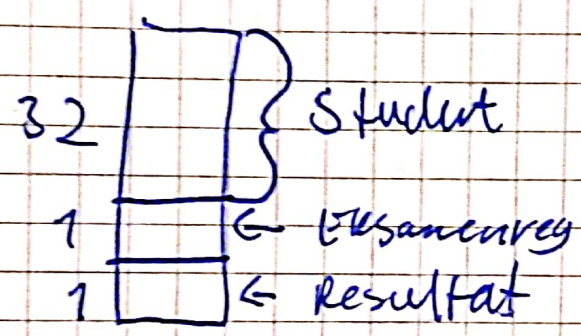
Oppgave 2

~~Blokker~~ Poster

Student: 47 000 ~~blokker~~ 800 blokker.

Eksamensreg: 600 000 Poster 12800 blokker.

Bruker 32 bufferblokker til student,
1 til Eksamensreg og en til resultat:



~~Total I/O = 2·(32+12800) = 25 600~~

Total I/O = 25·(32+12800) = 320 800

Kunne gjort det andre veien med
Total I/O = 400·(32+800) = 332 800
som er mindre effektivt.

Oppgave 3

a) Transaksjoner støtte deling og samtidig aksess av data, og støtter sikker, pålitelig, atomiske aksess til store mengder data. (Recovery).

b) ACID:

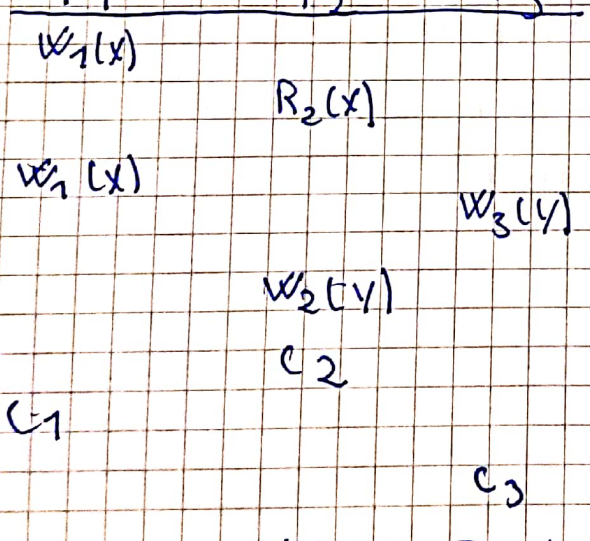
A: Atomisk transaksjon. Enten kjører de fullstendig, eller ikke i det hele tatt.

C: Consistency. Overholder konsistenskrav som PK, referanser, sjekker osv....

I: Isolation. Transaksjonene er isolert fra hverandre. De merker ikke at noen kjører samtidig

D: Durability. Transaksjonene er permanente.

c) H_1:



H_1 ikke gjennoprettbar. T_2 leser x etter T_1 har skrevet x. og Comitter før T_1.

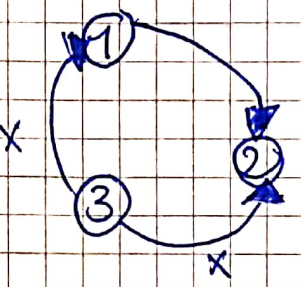
H_2 er ~~ACA, T_1 comitter ikke før~~ T_2 og T_3 skriver til y, men ingen leser fra y

H_3 er gjennoprettbar fordi at T_1 comitter etter T_3 som den leser fra.

d) To operasjoner er i konflikt ved 3 tilfeller:

1. $W_1(x)$ og $R_2(x)$
2. $W_1(x)$ og $W_2(x)$
3. $R_1(x)$ og $W_2(x)$.

e)



Det er ingen sykler, Historien er konflikt serialiserbar.

f) Det vil si at to eller flere transaksjoner venter gjensidig på hverandres låser.
F.eks:

T_1	T_2
$Rl_1(x)$	$Rl_2(x)$
$Wl_1(x)$	$Wl_2(x)$

g)

T_1	T_2	T_3
		$Wl_3(x)$
		$W_3(x)$
Try-lock(x)		
	Try-lock(x)	
		$Rl_3(y)$
		C_3; $Ul_3(x,y)$
$Rl_1(x)$	$Rl_2(x)$	
$R_1(x)$	$R_2(x)$	
C_1; $Ul_1(x)$	Try-lock(x)	
	~~$Wl_2(x)$~~ $Ul_2(x)$	
	~~$R_2(x)$~~ $Wl_2(x)$	
	~~$W_2(x)$~~ $W_2(x)$	
	~~C_2; $Ul_2(x)$~~ C_2; $Ul_2(x)$	

tid

Oppgave 4)

a) T_1 og T_3 ruller ikke comitte, så disse får UNDO, T_2 ruller comitte og får REDO.

b) Transaksjonstabell:

TransID	lastLSN	Status/tilstand
T_1	151	In Progress
T_2	150	Comitt
T_3	152	In Progress.

Dirty page table:

PageID	RecoveryLSN
A	148
B	149