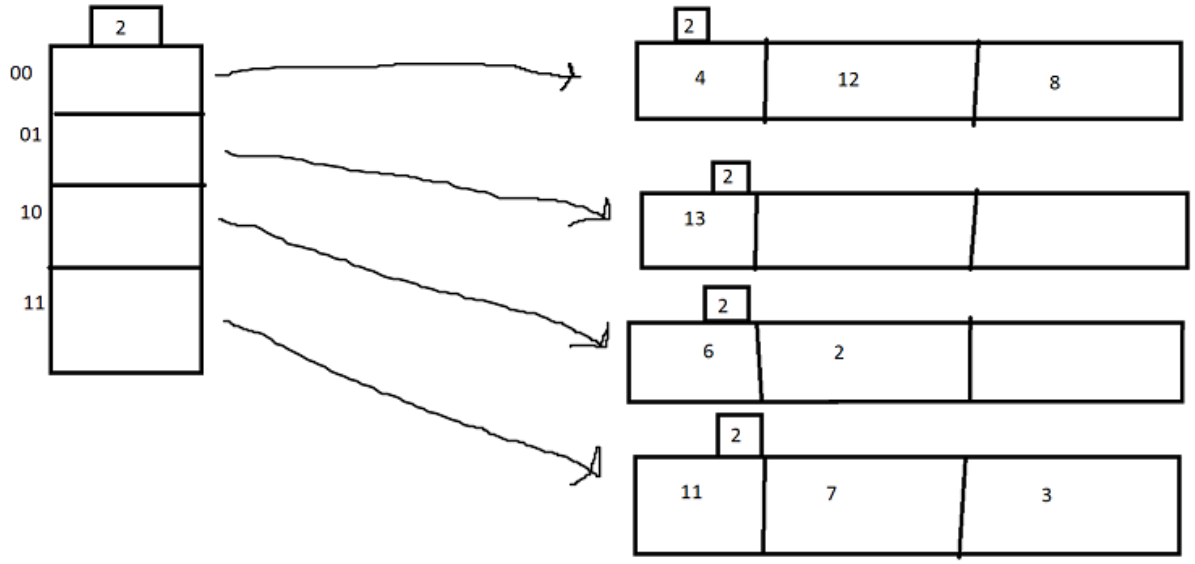
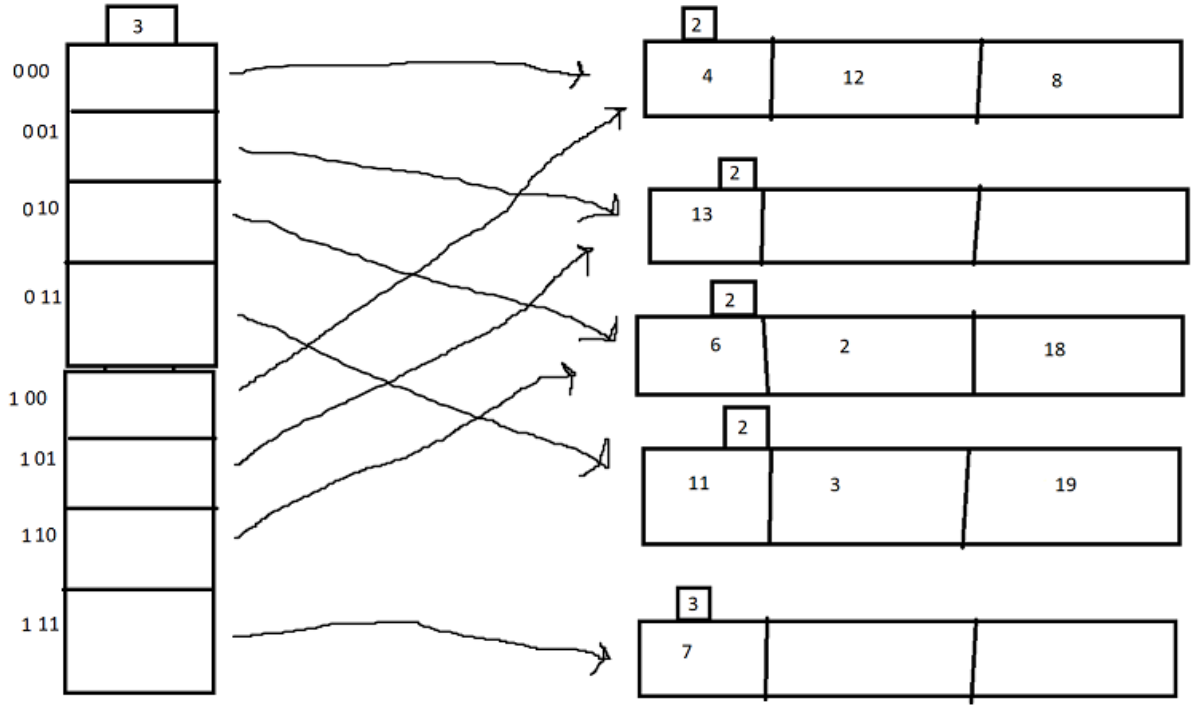
TDT4145 Datamodellering og databasesystemer - Øving 4

Oppgave 1

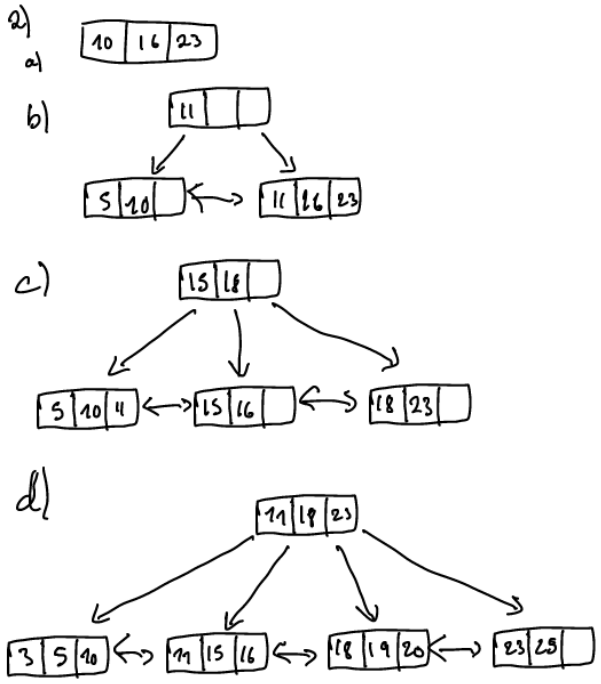
Steg 1:
Hashfunksjon: k % 4



Steg 2:
Hashfunksjon: k % 8



Oppgave 2



Oppgave 3

a)

1 blokk = 2048 byte totalt
1 person = 1 post
10 000 personer
10 000 / 8 = 1250 blokker
1250*3/2 = 1875 blokker

b) nivå 1: ((1875/2048 byte)*⅔)/(4+4 byte)) = 12 blokker
nivå 2: Roten består kun av èn blokk

c)

1) SELECT * FROM Person where PersonID = 195454;
3 blokker aksesseres

2) SELECT * FROM Person;
2 + 1875 blokker = 1877 blokker aksesseres

3) SELECT * FROM Person ORDER BY PersonID ASC;
2 + 1875 blokker = 1877 blokker aksesseres

4) SELECT FirstName, LastName FROM Person WHERE PersonID < 100000;

1875 * 0.05 + 2 = 93,75 + 2 = 96 blokker
96 blokker aksesseres

Oppgave 4

1) SELECT * From Person;
   1250 blokker aksesseres
2) 1250 blokker aksesseres fordi man må scanne hele heapfilen.
3) **3** Blokker i b+-treet + **antall personer med LastName** = “søkerud” i headfilen
4) Går rett til indeksen siden treet har LastName som søkenøkkel: 2 + 300 = 302 blokker aksesseres
5) (1+1) + (3+1) = 6 Hash read and write + B+tree read and write

Oppgave 5

25 * (32 + 12800) = 320 800 blokker leses totalt i løpet av joinen
Forklaring:

- Vi tar utgangspunkt i *Student*-tabellen siden den er lagret med færrest blokker.
- Buffer har plass til 34 blokker, hvor to av de blir satt av til *Eksamensregistrering*, så vi står igjen med 32 blokker til *Student*.
- For hver loop kan bufferen bare ta 32 blokker => vi trenger 800 / 32 = 25 runder.
- Det settes av 12800 blokker til *Eksamensregistrering*.

Oppgave 6

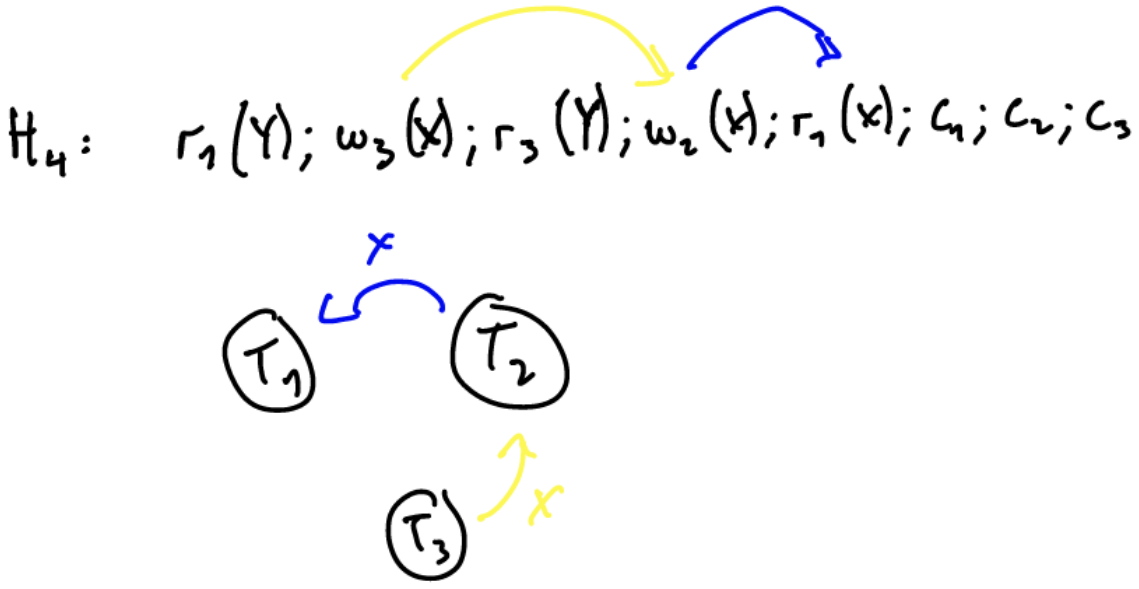
a) Det er hovedsakelig to årsaker til hvorfor vi ønsker transaksjoner:
   1. Flerbrukerkontroll - støtter deling og samtidig aksess av data
   2. Logging og recovery - støtter sikker, pålitelig, atomisk aksess til store mengder data.

b) ACID - egenskaper ved en transaksjon:
   - A - atomiske: enten kjører de fullstendig, eller så kjører de ikke.
   - C - konsistensbevaring: overholder konsistenskrav (primary key, references, check, osv)
   - I - isolering: som er isolert fra hverandre. Merker ikke at noen kjører samtidig.
   - D - durabilitet: er permanente, dvs. mistes ikke etter en commit.

c) Gjenopprettbar: Hver transaksjon committer etter at transaksjoner de har lest fra har committet.
   ACA: En transaksjon kan kun lese verdier som har blitt committet.
   Strict: En transaksjon kan verken lese eller skrive ikke-committede verdier.
   - H1 - Gjenopprettbar
   - H2 - ACA
   - H3 - Ikke gjenopprettbar

d) Man kan avgjøre om to operasjoner i en historie er i konflikt ved å endre på rekkefølgen deres, dersom dette fører til endringer i databasen vet man at operasjonene er i konflikt med hverandre.

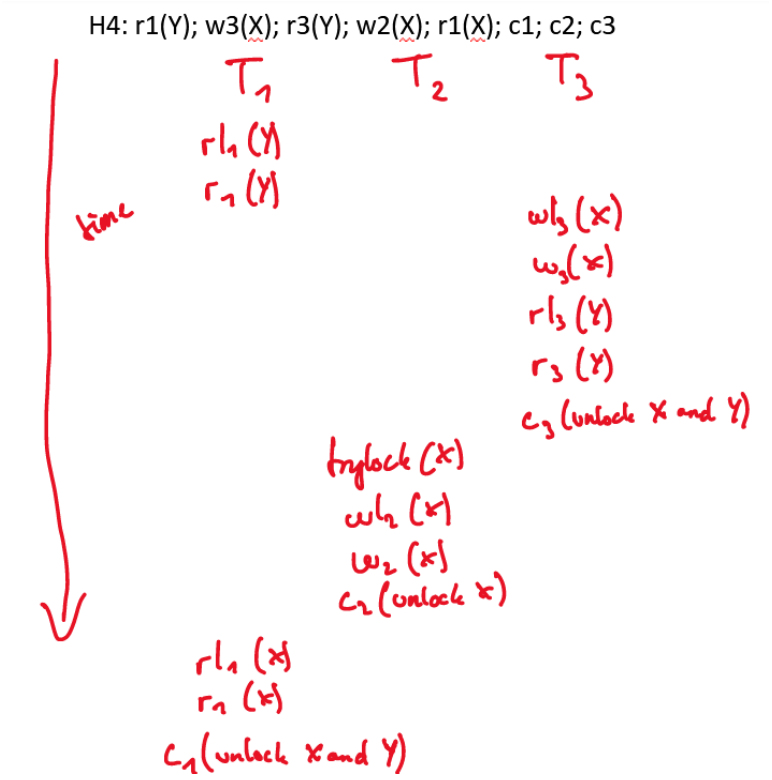
e) Historien H4 er konfliktserialiserbar, da det er ingen sykler.



f) En vranglås oppstår hvis to ulike transaksjoner venter på hverandre på grunn av låser som blokkerer. En vranglås må løses ved at en av transaksjonene må begynne på nytt.

g) Under har vi brukt rigorous tofaselåsing (rigorous 2PL) på H4.

H4: r1(Y); w3(X); r3(Y); w2(X); r1(X); c1; c2; c3



Oppgave 7

a) T1 og T2 er transaksjonene som er vinnere, mens T3 er taperen. I loggen ser man at kun skjer to commits fra end_checkpoint til kræsj (LSN = 174).

b) Transaksjonstabell

Transaction_id	Last_LSN	Status
T1	173	Commit
T2	170	Commit
T3	174	In progress

Dirty Page Table (DPT)

Page_id	Recovery_LSN
A	168
B	169