NTNU
Kunnskap for en bedre verden

Institutt for datateknologi og informatikk

Løsning for eksamensoppgave i TDT4145 Datamodellering og databasesystemer

Faglig kontakt under eksamen:

Roger Midtstraum: 995 72 420

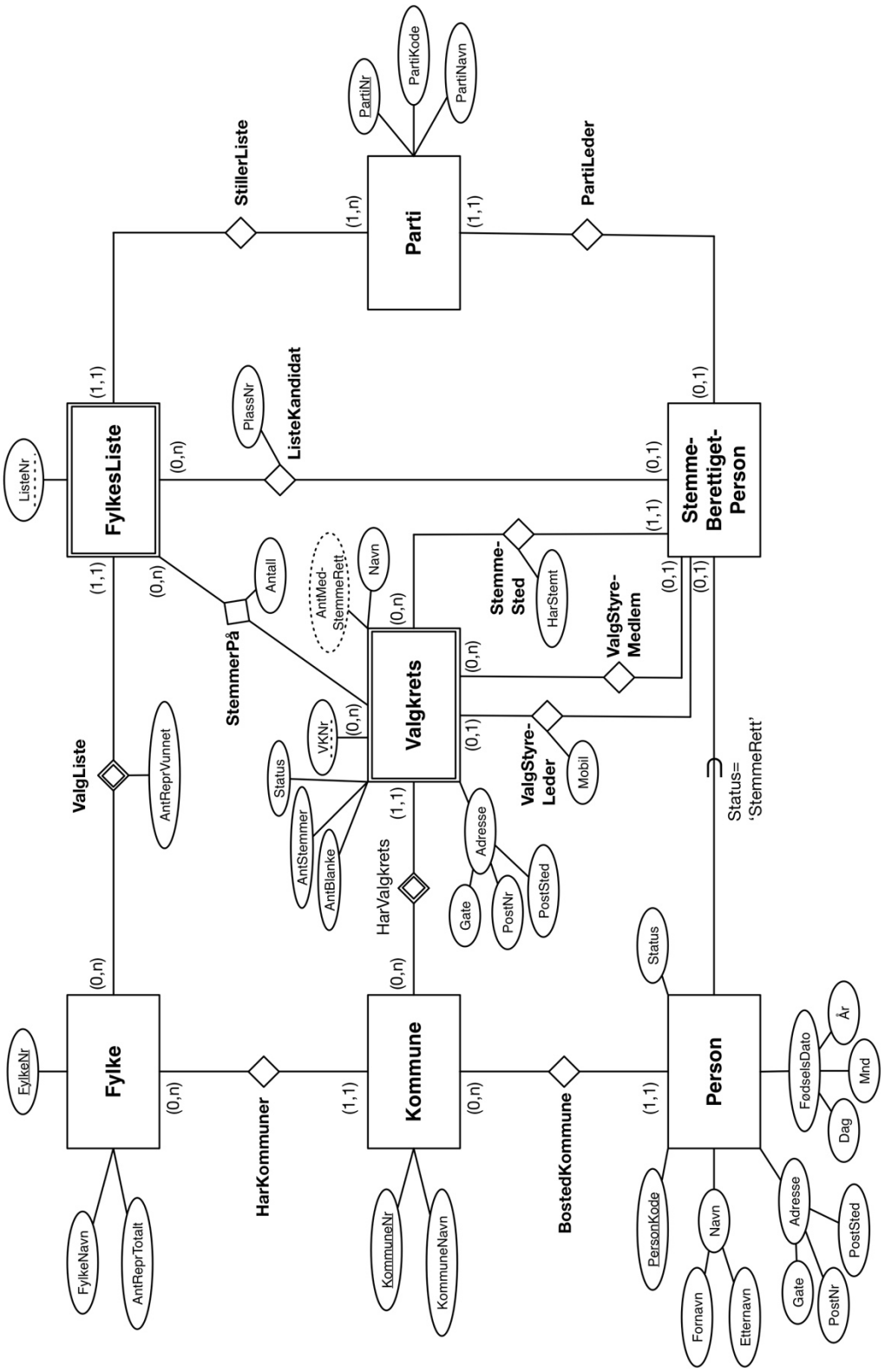
Eksamensdato: 7. juni 2017

Eksamenstid (fra-til): 09:00-13:00

Hjelpemiddelkode/Tillatte hjelpemidler:

D – Ingen trykte eller håndskrevne hjelpemidler tillatt. Bestemt, enkel kalkulator tillatt.

Oppgave 1 – Datamodeller (20 %)



Vi har forutsatt at en person ikke kan være partileder i flere partier eller ha tillitsverv i mer enn en valgkrets. Vi har brukt noen sammensatte attributt og et utledet attributt, det skal ikke trekkes om man ikke har brukt disse virkemidlene, men løst det på en annen, akseptabel måte.

Oppgave 2 – Relasjonsdatabaser, relasjonsalgebra og SQL (24 %)

I løsningene har vi konsekvent brukt naturlig join, der det er mulig, for å forene tabeller. Det skal ikke trekkes for (riktig) bruk av equijoin / inner join. Formulering av relasjonsalgebra som tekst er likeverdig.

a) Forutsetter at PersonID er unik for personer og at Name ikke er det, tilsvarende for Fruits og at Weight representerer samlet innhøsting av en frukttype for en person. Det gir oss følgende:

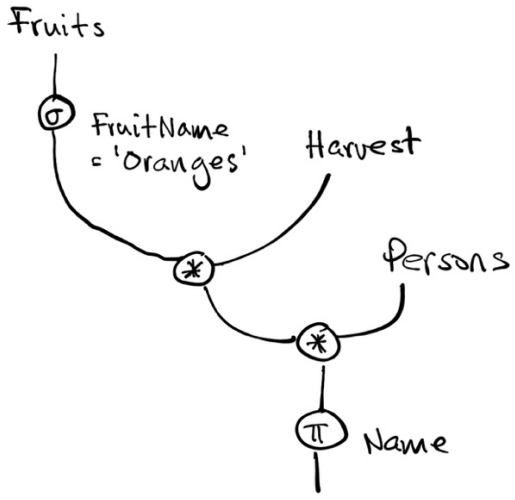
   Nøkler: PersonID i Person, FruitType i Fruits, og PersonID og FruitType i Harvest.

   Fremmednøkler: PersonID i Harvest er en fremmednøkkel mot Person-tabellen. FruitType i Harves er en fremmednøkkel mot Fruits-tabellen.

b) Spørringen vil gi følgende tabellforekomst som resultat:

PersonID	Name	KilosHarvested
1	Kari	400
2	Ola	200
3	Per	100

c) Relasjonsalgebra:



d) SQL:

```
select FruitName
from Persons natural join Harvest natural join Fruits
where Name = 'Kari'
```

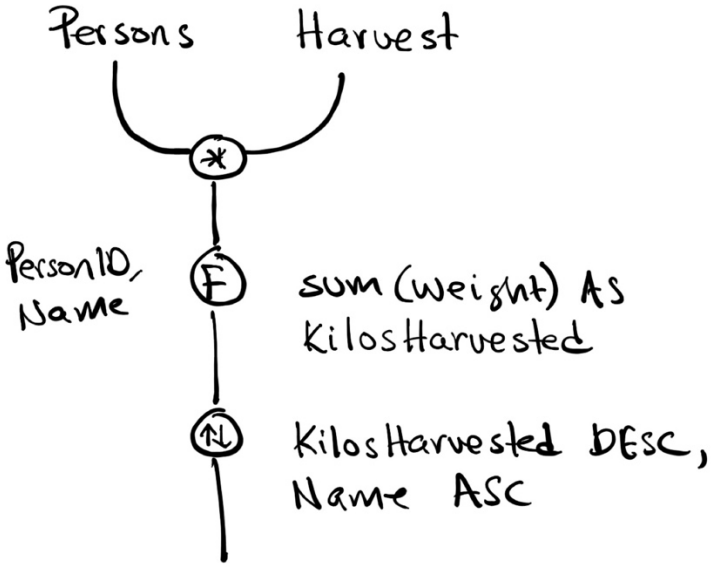
e) SQL:

```
select PersonID, Name
from Persons
where PersonID NOT IN (select PersonID from Harvest)
```

f) SQL:

```
select FruitType, FruitName, sum(weight) as TotalWeight
from Fruits natural join Harvest
group by FruitType, FruitName
having TotalWeight > 100
```

g) Relasjonsalgebra:



Oppgave 3 – Normaliseringsteori (16 %)

a) En *funksjonell avhengighet* er en restriksjon på sammenhengen mellom to mengder attributter. Hvis en funksjonell avhengighet, X→Y, gjelder for en tabell, vil alle rader med samme verdier for attributtene i X måtte ha samme verdier for attributtene i Y.

b) Legg merke til at det er en slurvefeil i F: CustomerNo skulle vært CustNo.

Innsettingsproblemer: Vi kan ikke registrere kunde- eller pizzainformasjon uten å ha en pizzaordre.

Oppdateringsproblemer: Tabellen vil ha redundant lagring av kunde- eller pizzainformasjon som krever konsistent oppdatering av mange rader dersom denne informasjonen endres.

Slettingsproblemer: Kunde- eller pizzainformasjon kan forsvinne fra databasen ved sletting av en pizzaordre.

Løsningen er å bli kvitt potensialet for redundant lagring av informasjon, ved å splitte i tre tabeller: PizzaCustomers(CustNo, CustName), PizzaTypes(PizzaID, PizzaName, PizzaPrice) og PizzaOrders(OrderNo, PizzaID, CustNo, NumberOrdered). I PizzaOrders vil CustNo være fremmednøkkel mot PizzaCustomers og PizzaID vil være fremmednøkkel mot PizzaTypes.

c) R må dekomponeres i R_1(B, C) og R_2(A, C, D). Der vil vi ha $F_1 = \{C \rightarrow B\}$ og $F_2 = \Phi$. Avsjekk av kravene:

- *Attributtbevaring*: Alle attributtene i R er med i enten R_1 eller R_2
- *Bevaring av funksjonelle avhengigheter*: $F_1 \cup F_2 = F$
- *Tapsløst-join:* $R_1 \cap R_2 = C$. Oppfylt siden C er (super-)nøkkel i R_1
- *BCNF:* R_2 har ingen (ikke-trivielle) funksjonelle avhengigheter og er derfor nødvendigvis på BCNF. I R_1 har vi $C \rightarrow B$. Siden C er (super-)nøkkel i R_1 vil BCNF være oppfylt. Man kan evt. gjøre en antagelse om at R oppfyller 1NF, men det er ikke nødvendig for full uttelling.

d) Når vi har Person ->> Sport og dermed også Person ->> Food, må sammenhengene Person-Sport og Person-Food være uavhengige. Under er vist en gyldig tabellforekomst der radene som manglet, er markert med grønn bakgrunnsfarge:

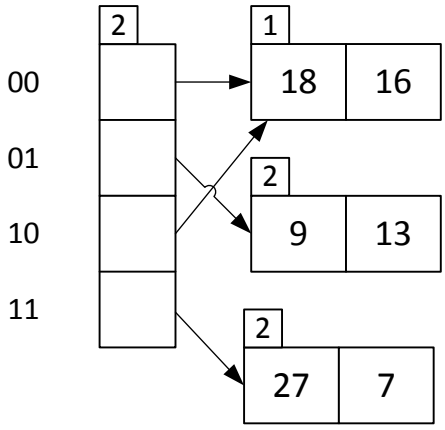
Interests

Person	Sport	Food
Kari	Athletics	Icecream
Ola	Tennis	Chocolate
Kari	Soccer	Biscuits
Kari	Soccer	Icecream
Kari	Athletics	Biscuits

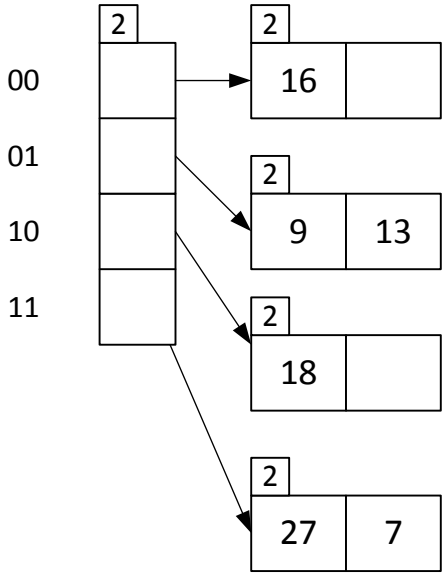
Oppgave 4 – Extendible hashing (5 %)

27 % 4 = 11 (binært)
18 % 4 = 10
9 % 4 = 01
7 % 4 = 11
16 % 4 = 00
13 % 4 = 01

Hvis du bruker løsningen fra pensumnotatet (hasher fra minst signifikante bit), vil strukturen bli slik:



Hvis du bruker løsningen fra læreboka (hasher fra mest signifikante bit), vil strukturen bli slik:



Oppgave 5 – Lagring og queries (16 %)

Vi har en database som lagrer brukervurdering av filmer. Dette er lagret i en tabell:

Movie (<u>movieId</u>, title, directorId, prodYear, nVotes, avgRating)

Hver film vil ha en post i denne databasen. Vi har lagret informasjon om 200 000 filmer.
Hver post (record) i Movie-tabellen er 100 byte. Hver blokk i databasen er 4096 byte (4 KB).

a) *40 poster per blokk. **5000** blokker til sammen.*

b) Antar B+-treet har fyllingsgrad 0,67 (pensumnotatet). Læreboka opererer med 0,69 som fyllingsgrad.
*Level = 0 (løvnivå): 5000 * 1,5 = **7500** blokker på nivå 0.*
*Level = 1: 7500 poster. Hver blokk kan ha (4096 / 8) * (2/3) poster = 341 poster. **22** blokker til sammen.*
*Level = 2: **1** blokk. Får plass til alle 22 postene her.*

c) *Ved å anta a): Gjennomsnittlig må halve heapfila leses: **2500** blokker.*
*Ved å anta b): Leser rett nedover B+-treet: **3** blokker.*

d) *16 000 blokker joines med 500 blokker. Lar Agency (den minste tabellen) ha 50 blokker i buffer og Actor ha 1 blokk og resultatet 1 blokk. Får da 10 loops * (50 + 16 000) = **160 500** blokker lest. Ved å lese den største først får du 320*(50 + 500) = 176 000.*

Oppgave 6 – Transaksjoner: Problemer og låsing (11 %)

Vi har en historie:

H1: r2(A); w1(A); w1(B); w3(B); w2(B); c2; c1; c3

a) (5 %)
Recoverable: ingen read av uncommitede verdier
ACA: ingen read av uncommitede verdier
Ikke strict: fordi det er skriving over en uncommited verdi.

b) (6 %)

T1	**T2**	**T3**
	rl2(A)	
	r2(A)	
må vente på A		
		wl3(B)
		w3(B)
	må vente på B	
		C3; unlock(B)
	wl2(B)	
	w2(B)	
	C2; unlock (A,B)	
wl1(A)		
w1(A)		
wl1(B)		
w1(B)		
C1; unlock (A,B)		

Oppgave 7 – Transaksjoner: Recovery – ARIES (8 %)

a) (5 %) *For hver loggpost i loggen beskriv om det skjer REDO eller ikke og hvorfor/hvorfor ikke?*
*103: Nei, fordi recLSN > Loggpost.LSN (*egentlig ikke nødvendig å si da REDO starter på eldste RecLSN, dvs. 106*).*
106: Nei, fordi PageLSN=Loggpost.LSN
108: Nei, fordi recLSN > Loggpost.LSN
110: Ja, fordi PageLSN < Loggpost.LSN

b) (3 %) Hva er verdiene til A, B, C og D etter at hele recovery (også UNDO) er ferdig? Begrunn svaret ditt.
A = 30, B = 11, C = 40, D = 25. Kun T2 må rulles tilbake, og da får D den forrige verdien, dvs. 25.