UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i	INF212 — Databaseteori
Eksamensdag:	28. mai 2003
Tid for eksamen:	9.00 – 15.00
Oppgavesettet er på 8 sider.	
Vedlegg:	1 - Enkel grammatikk for parsing av spørsmål
Tillatte hjelpemidler:	Kalkulator

Kontroller at oppgavesettet er komplett før
du begynner å besvare spørsmålene.

For hver oppgave er det oppgitt en prosentsats som angir omtrent hvor mye oppgaven utgjør av den totale arbeidsmengden. Endelig vekting av de enkelte punktene vil bli bestemt i samråd med sensor.

Verktøygrossist

Vi skal som underlag for eksamenssettet se på en enkel relasjonsdatabase for en verktøygrossist. Kundene av grossisten er detaljister (butikker eller foretak, gjerne også enkeltpersoner, som videreselger verktøy til privatpersoner og profesjonelle). Detaljistene kan bestille varer fra grossisten via en internettapplikasjon.

Databasen inneholder et kunderegister *Kunde* der det for hver kunde er registrert et entydig kundenummer, kundens navn, eventuelt kjønn hvis det er et enpersonsforetak, adresse og foretaksnummer. Dessuten inneholder databasen et varelagerregister *Vare* der det for hvert vareslag er registrert en tilhørende entydig kode, et navn på varen, hvilken varegruppe varen tilhører, enhetspris samt hvor mange enheter grossisten har på lager.

For å holde rede på bestillingene har databasen ytterligere to registre *Ordre* og *Ordreeier*. Hver ordre har et ordrenummer *ordreID*. I relasjonen *Ordre* er det til hver vare i en ordre angitt hvor mange enheter kunden har bestilt av varen. I *Ordreeier* finner man opplysninger om hvilken kunde ordren gjelder og datoen da bestillingen ble registrert.

I databaseskjemaet er *kandidatnøkler* understreket mens primærnøkler har to understrekninger. Det er fremmednøkler fra *Ordre* til *Vare* og *Ordreeier*, og fra *Ordreeier* til *Kunde*. Hvilke attributter som utgjør fremmednøkler,

(Fortsettes på side 2.)

fremgår ved at det er valgt samme attributtnavn i fremmednøkkel som i den relasjonen fremmednøkkelen refererer.

Kunde(<u>*kundeID*</u>, *navn*, *kjønn*, *adresse*, <u>*foretaksnr*</u>)

Vare(<u>*vareID*</u>, *varenavn*, *varegr*, *enhpris*, *antall*)

Ordre(<u>*ordreID*, *vareID*</u>, *antbest*)

Ordreeier(<u>*ordreID*</u>, *kundeID*, *dato*)

Attributtet *dato* i *Ordreeier* antas å være en standard SQL DATE, dvs. tekst i format '2002-12-31'. *Kunde* kan være en enkeltperson eller en bedrift, dette kan sees fra attributtet *kjønn* i *Kunde* som for personer er 'K' for kvinne og 'M' for mann, mens bedrifter har 'B'. Hvis vedkommende ikke er en bedrift, vil *foretaksnr* være personnummer isteden.

Oppgave 1 FDer (20%)

Betrakt følgende alternativ til relasjonsdatabasen {*Kunde*, *Vare*, *Ordre*, *Ordreeier*}:

Vare(<u>*vareID*</u>, *varenavn*, *varegr*, *enhpris*, *antall*)

Ordre(<u>*ordreID*, *vareID*</u>, *antbest*)

Ordreinfo(<u>*ordreID*</u>, *dato*, *kundeID*, *navn*, *kjønn*, *adresse*, *foretaksnr*)

(i) Hvilke FDer må gjelde i *Ordreinfo* for at vi skal kunne dekomponere *Ordreinfo* tapsfritt til {*Kunde*, *Ordreeier*}?

(ii) Angi hvilke normalformer det er brudd på i *Ordreinfo*, og beskriv for hver hvor bruddet/bruddene er.

(iii) Er det brudd på noen normalformer i noen av relasjonene *Kunde*, *Vare*, *Ordre*, *Ordreeier*? Begrunn ditt svar.

Oppgave 2 SQL (20%)

Besvar følgende spørsmål med SQL.

(i) Finn navn og adresse på alle kunder som har bestilt varer i varegruppen 'TANNLEGEVERKTØY'.

(ii) Lag en liste over navn, adresse og totalt beløp hver kunde har bestilt for i mai 2003. Kunder som ikke har bestilt noen varer i denne måneden, skal ikke være med på listen.

(Fortsettes på side 3.)

(iii) Lag en liste over alle varene i varegruppen 'TANNLEGEVERKTØY'. Hver linje i listen skal inneholde varenavn, enhetspris, det totale antall enheter bestilt av varen i år 2002 og antall forskjellige kunder som bestilte varen i 2002. Listen skal være sortert etter totalt antall bestilt slik at varen med flest bestilte enheter kommer først, og varer som ikke er bestilt av noen, kommer sist.

Oppgave 3 Implementasjon (40%)

I denne oppgaven skal du vise din forståelse av hvordan et databasesystem implementeres.

Vi skal bruke følgende spørring for å finne navnene og adressene til kvinnelige kunder som har bestilt varer i varegruppen 'FYRVERKERI' i fjor, dvs. i '2002':

```
select  Kunde.navn, Kunde.adresse
from    Kunde, Ordreeier, Ordre, Vare
where   Kunde.kundeID = Ordreeier.kundeID      and
        Ordreeier.ordreID = Ordre.ordreID      and
        Ordreeier.dato like '2002'             and
        Ordre.vareID = Vare.vareID             and
        Vare.varegr = 'FYRVERKERI'             and
        Kunde.kjønn = 'K'
```

Databasen har "clustered" indekser på primærnøklene. I tillegg er det indekser på attributtet *dato* i *Ordreeier* og på attributtet *foretaksnr* i *Kunde*.

Kommentar til regneoppgavene: Det er tillatt å bruke kalkulator. Det viktigste er imidlertid å vise din forståelse av problemet og dets løsning, og ikke et nøyaktig svar! Det betyr at et riktig formulert uttrykk med feil numerisk svar vurderes høyere enn et riktig numerisk svar uten forklaring/formulering!

3a Parsing

(i) Bruk den enkle grammatikken i vedlegg 1 til å lage et parse-tre for spørringen ovenfor.

(ii) Hvilke(n) hovedoppgave(r) har preprosessoren?

(Fortsettes på side 4.)

3b Logisk spørreplan

Konverter parse-treet i oppgave 3a ovenfor til en logisk spørreplan i relasjonsalgebra (tegn uttrykkstreet). NB! Denne oppgaven skal løses uten optimering. Optimering hører til neste oppgave!

3c Optimering

(i) Hvilke regler benyttes ofte (gir oftest stor ytelsesgevinst) for optimering av logiske spørreplaner?

(ii) Optimer den logiske spørreplanen i deloppgave 3b ovenfor (tegn det nye uttrykkstreet).

3d Eksekvering

Hvis datamengden er så stor at vi ikke kan holde hele datasettet i minnet samtidig, baserer vi eksekveringen av (fullrelasjons) operatorene på to grunnprinsipper.

(i) Hvilke er disse, og hvordan fungerer de i korte trekk?

(ii) I hvilke tilfeller vil du bruke de forskjellige metodene? Begrunn svaret.

3e Datalagring

Anta at vi har en disk med følgende spesifikasjoner for lagring av våre data:

Diskplater:	10 (med 2 overflater hver)
Spor:	10.000 pr. overflate
Antall sektorer pr. spor:	1000 (en ikke-sonet disk)
Byte pr. sektor:	512
Byte pr. "gap":	64
Gjennomsnittlig søketid:	5 ms.
Spor-spor søk:	0,5 ms
Rotasjonshastighet:	10.000 RPM

Anta også at hver relasjon er lagret "clustered" på disken. Databasen inneholder følgende antall tupler:

Antall *Kunde*-tupler:	100.000
Antall *Vare*-tupler:	10.000
Antall *Ordre*-tupler:	10.000.000
Antall *Ordreeier*-tupler:	1.000.000

I tillegg gjelder følgende informasjon om diverse størrelser:

(Fortsettes på side 5.)

- Hver blokk har en "header" (hode) på 20 byte.
- Hver "record" (post) har et hode på 10 byte.
- Hvert attributt har følgende størrelse i antall bytes:

Kunde		*Vare*		*Ordre*		*Ordreeier*	
kundeID:	16	*vareID*:	16	*ordreID*:	16	*ordreID*:	16
navn:	40	*varenavn*:	40	*vareID*:	16	*kundeID*:	16
kjønn:	1	*varegr*:	10	*antbest*:	10	*dato*:	10
adresse:	40	*enhpris*:	10				
foretaksnr:	11	*antall*:	10				

(i) Hva er diskens utnyttbare kapasitet? Ikke glem at hver plate har to overflater!

(ii) Hvilke faktorer inngår i å aksessere en blokk på disken, og hva er gjennomsnittlig aksesstid for en vilkårlig 4 Kbyte blokk?

(iii) Hvor stor plass trenger disse relasjonene på disken i tilfellet "unspanned" lagring (dvs. hvis ingen enkelt post er delt over flere blokker)?

(iv) Hvor lang tid tar det å lese hele relasjonen *Ordre* uavbrutt hvis vi antar vilkårlig plassering av data i diskblokker på disken?

(v) Hvordan kan man optimere en slik lesing av en relasjon ved å forandre blokkplassering, og hva blir den nye tiden for å lese hele *Ordre*? NB: Skriv ned eventuelle antakelser du gjør!

Oppgave 4 Transaksjoner (20%)

Kundene bestiller varer over internett gjennom en "handlevognsapplikasjon". Typisk vil kundene først gå gjennom en fase hvor de undersøker hvilke varer som fins, og deretter en fase hvor de velger ut hvilke varer, og hvor mange enheter av hver vare, som skal bestilles. Deretter ber de handlevognsapplikasjonen om å gjennomføre og bekrefte bestillingen.

Vi skal konsentrere oss om det som skjer med relasjonen *Vare*. For hver bestilt vare skal verdien i attributtet *antall* telles ned med det bestilte antall enheter. For å gjennomføre en bestilling på vegne av en kunde, utfører applikasjonen en transaksjon som leser tuplene til de varene som skal bestilles, sjekker at det er nok enheter på lager av hver vare, og for de varene der det er nok enheter på lager, teller ned i *antall*. Dersom det ikke er nok enheter på lageret, foretas ingen bestilling av denne varen.

Hvis vi lar $r_i(A)$ og $w_i(A)$ betegne at en transaksjon T_i henholdsvis leser og skriver et tuppel A i *Vare*, så kan derfor to mulige transaksjoner i en gitt

(Fortsettes på side 6.)

databasetilstand se slik ut:

$$T_1 : \; r_1(A); r_1(B); w_1(A); w_1(B)$$
$$T_2 : \; r_2(A); r_2(B); w_2(B)$$

(T_1 bestiller enheter av to varer, representert ved henholdsvis A og B; T_2 bestiller bare B fordi det viser seg å ikke være nok enheter av A.)

Betrakt følgende eksekveringsplan ("schedule") S av T_1 og T_2:

$$S : r_1(A); r_2(A); r_1(B); w_1(A); w_1(B); r_2(B); w_2(B)$$

4a Serialiserbarhet

(i) Vis at S ikke er konfliktserialiserbar.

(ii) Begrunn at S heller ikke generelt er serialiserbar.

(iii) Finn en antakelse om A og B som gjør S serialiserbar. Er det rimelig å anta at S er serialiserbar i handlevognapplikasjonen? Begrunn.

4b Samtidighetskontroll

Anta at vi har eksklusive låser på hvert tuppel i *Vare*. Låsene skal brukes på vanlig måte, ved at hver lese- og skriveaksjon skal ha en forutgående låseaksjon og en etterfølgende opplåsningsaksjon. Dessuten skal hver transaksjon benytte tofaselåsing (2PL).

La aksjonen $l_i(Y)$ bety at T_i tar låsen på Y og $u_i(Y)$ at T_i frigir låsen på Y.

(i) Legg inn aksjoner av formen $l_i(Y)$ og $u_i(Y)$ i hver av T_1 og T_2 slik at de oppfyller reglene for bruk av låsene under 2PL, og samtidig frigjør låser så snart som mulig.

(ii) Beskriv hva som skjer hvis vi prøver å utføre aksjonene i de resulterende transaksjonene slik at lese/skriveaksjonene utføres mest mulig i samsvar med rekkefølgen angitt av S.

Anta så at vi på hvert tuppel i *Vare* har to låser – en delt (S-lås, shared lock) og en eksklusiv (X-lås), der en S-lås kan oppgraderes til (byttes ut med) en X-lås ved behov. Låsene skal forøvrig brukes som vanlig er for S/X-låser og i henhold til 2PL.

La aksjonene $ls_i(Y)$ og $lx_i(Y)$ bety at T_i tar henholdsvis S-låsen og X-låsen på Y, og $u_i(Y)$ at T_i frigir alle låser på Y.

(Fortsettes på side 7.)

(iii) Legg inn aksjoner av formen $ls_i(Y)$, $lx_i(Y)$ og $u_i(Y)$ i hver av T_1 og T_2 slik at de oppfyller reglene for bruk av låsene under 2PL, og slik at transaksjonene ikke benytter X-låser mer enn strengt nødvendig (dvs. de benytter oppgradering der dette er mulig). Låser skal frigjøres så snart som mulig.

(iv) Beskriv hva som skjer hvis vi prøver å utføre aksjonene i de resulterende transaksjonene slik at lese/skriveaksjonene utføres mest mulig i samsvar med rekkefølgen angitt av S.

Naci Akkök *Ellen Munthe-Kaas* *Ragnar Normann*

(Fortsettes på side 8.)

Vedlegg 1 - Grammatikk for parsing av spørsmål

```
<query> ::= <SFW>

<SFW> ::= SELECT <selList> FROM <fromList> WHERE <condition>

<selList> ::= <attribute>
<selList> ::= <attribute>, <selList>

<fromList> ::= <relation>
<fromList> ::= <relation, <fromList>

<condition> ::= <condition> AND <condition>
<condition> ::= <attribute> = <attribute>
<condition> ::= <attribute> = <pattern>
```

Elementære syntaktiske kategorier som `<attribute>`, `<relation>` og `<pattern>` har ingen regler, men oversettes henholdsvis med navnet på attributtet, navnet på relasjonen eller en streng i anførselstegn.