# Andre sett obligatoriske oppgaver i INF3100/INF4100 V2007

## Formalia

Studentene skal levere individuell besvarelse. Hvis to studenter ønsker å levere felles besvarelse, må dette søkes om og bli godkjent på forhånd av gruppelærer. Studenter som tar Inf 3190/4190 og har hjemmeeksamen med innlevering fredag 11. mai, kan søke gruppelærer om en ukes utsettelse på innlevering.

Besvarelsen skal sendes med e-post til gruppelærer.

E-posten skal ha følgende subjektfelt:
Subject: Oblig 2 inf3100 (<brukernavn student>)

**Merk:** Besvarelser som leveres etter fristen, vil bli underkjent. Alle spørsmålene må besvares for å få godkjent besvarelsen.

Studenter som har fått godkjent de obligatoriske oppgavene (eventuelt en av dem) og likevel vil trekke seg fra eksamen, er selv ansvarlig for å ta vare på papirkopier av besvarelsene hvor gruppelæreren har kvittert for at oppgaven(e) er godkjent. Dette gjelder bare studenter som trekker seg før 14-dagersfristen.

## Innleveringfrist: Fredag 11. mai kl. 12.00

Det er lurt å begynne tidlig.

## Oppgave 1

Gitt følgende datastruktur for en relasjonsdatabase Primærnøkler er markert med **fete** typer, kandidatnøkler er i *kursiv*, fremmednøkler fremgår av attributtnavnene):

> PERSON (**Fnr**, Etternavn, Fornavn, Adresse)
> EKTESKAP (**Fnr-k** *Fnr-m*, ***Dato***, Etternvn-k, Etternavn-m)
> NAVNESKIFTE (**Fnr**, **Dato**, Etternavn, Fornavn)

Nåværende navn lagres i tabellen PERSON. Følgelig lagres det navnet en person hadde umiddelbart før han/hun skiftet navn, i NAVNESKIFTE. Navnene i EKTESKAP er etternavnene etter vielsen.

Løs følgende to oppgaver både med relasjonsalgebra og SQL:

1. Finn navn og adresse til alle kvinner som i årene 1968-72 på sin bryllupsdag skiftet etternavn til et navn forskjellig fra sin brudgoms etternavn.
2. Finn navn og adresse til alle kvinner som ved en vielse har byttet navn med sin brudgom. (Ektefeller med samme etternavn skal ikke være med.)

# Oppgave 2

Med utgangspunkt i Oppgave 1 skal det lages en OO-datastruktur. Det er opplagt at tabellen PERSON skal erstattes av en klasse Person. Det er også nokså opplagt at tabellen NAVNESKIFTE skal erstattes av en lokal liste i hvert personobjekt. (Vi kunne selvfølgelig ha valgt en mengde eller bag i stedet for en liste, men navneskiftene for en person er jo ordnet.)

For tabellen EKTESKAP er saken ikke så opplagt. Vi kan velge å erstatte den med en egen klasse Ekteskap eller å erstatte den med lokale data i hvert enkelt personobjekt.

## Oppgave 2 a

Skriv en fullstendig ODL-definisjon for hvert av alternativene ovenfor (med hhv. to og en klasse). Ikke glem metodene, og husk at det bare er metodenes signatur som er en del av ODL!

## Oppgave 2 b

Bruk OQL til å besvare begge spørsmålene i Oppgave 1 mot hver av ODL-definisjonene i 2 a.

# Oppgave 3

Anta at vi har et lagringssystem som bruker (modifiserte) Seagate Cheetah X15 disker Noen av spesifikasjonene er som følger:

```
Diskplater:                         8
Spor:                               20000
Antall sektorer per spor:           700 (en ikke-sonet disk)
Bytes per sektor:                   512
Bytes per "gap"/sjekksum/...:       64
Gjennomsnittlig søketid:            3.6 ms
Spor-spor søk:                      0.3 ms
Rotasjonshastighet:                 15.000 RPM
```

Anta videre at lagringssystemet bruker 4 KB blokker, og at vi kan se bort i fra forsinkelser som å prosessere forespørsler, overføring over busser, vente på tur, etc.

## Oppgave 3 a

1. Hva er total kapasitet på disken hvis platene brukes på begge sider?
2. Hva er gjennomsnittlig rotasjonsforsinkelse?
3. Hva er gjennomsnittlig aksesstid for en vilkårlig blokk?

På lagringssystemet over lager vi en kundedatabase. Hver blokk har en «header» på 12 B, og hver post («record») har en «header» på 20 B. Vi har 1.000.000 kunder, hvor hver post har følgende felter (antall byte):

```
navn    (30)
fnr     (9)
adresse (30)
tlf     (8)
email   (30)
```

## Oppgave 3 b

1. Hva er blokkfaktoren?
2. Hvor lang tid tar det å lese hele tabellen (uavbrutt) hvis vi antar «unspanned» lagring og vilkårlig plassering av blokkene på disken?
3. I steden for vilkårlig plassering prøver vi med kontinuerlig plassering på disken - vi fyller opp et spor først, hvis fullt, så tar vi neste spor i samme sylinder, hvis igjen fullt, neste sylinder. Anta at tabellen begynner i første sektor i første spor i en sylinder. Hvor lang tid tar det nå å lese hele tabellen (uavbrutt) hvis vi fortsatt antar «unspanned» lagring og at det å bytte lesehode ikke tar noe tid (vi kan begynne å lese direkte)?

Anta igjen at blokkene er vilkårlig plassert på disken, og at tiden det tar å prosessere en blokk i minnet er tilnærmet null.

## Oppgave 3 c

1. Hva er gjennomsnittlig tid for å aksessere en vilkårlig post uten indekser?
2. Vi legger på en tett («dense») indeks med binært søk. Unik søkenøkkel er fnr, og pekeren tar 8 B. Hvor mye plass tar indeksen hvis hver blokk er 80 % full i gjennomsnitt?
3. Hva er gjennomsnittlig tid (forutsett et maksimalt antall oppslag i indeksen) for å aksessere en vilkårlig post med en slik indeks?
4. Vi bytter så ut indeksen vår med et $B^+$-tre. Hvor mye plass tar $B^+$-treet hvis hver node bruker en egen diskblokk som er 80 % full i gjennomsnitt?
5. Hva er tiden det nå tar å aksessere en vilkårlig post?

# Oppgave 4

Betrakt følgende eksekveringsplan (schedule) der vi bruker binære låser:

| | **Trans T1** | **Trans T2** | **Trans T3** |
|---|---|---|---|
| 1 | Lock(X) | | |
| 2 | Read(X) | | |
| 3 | | Lock(Y) | |
| 4 | | Read(Y) | |
| 5 | X=f(X) | | |
| 6 | | | Lock(Z) |
| 7 | | | Read(Z) |
| 8 | | Y=g(Y) | |
| 9 | | Write(Y) | |
| 10 | | Unlock(Y) | |
| 11 | | | Z=h(Z) |
| 12 | Write(X) | | |
| 13 | Unlock(X) | | |
| 14 | | | Write(Z) |
| 15 | | | Unlock(Z) |
| 16 | | | Lock(Y) |
| 17 | | | Read(Y) |
| 18 | | Lock(X) | |
| 19 | | Read(X) | |
| 20 | Lock(Z) | | |
| 21 | Read(Z) | | |
| 22 | | X=g(X) | |
| 23 | | | Y=h(Y) |
| 24 | Z=f(Z) | | |
| 25 | | Write(X) | |
| 26 | | Unlock(X) | |
| 27 | | | Write(Y) |
| 28 | | | Unlock(Y) |
| 29 | Write(Z) | | |
| 30 | Unlock(Z) | | |

## Oppgave 4 a

Tegn presedensgrafen (serialiserbarhetsgrafen) for denne planen, og avgjør om planen er konfliktserialiserbar.

### Oppgave 4 b

Begrunn at planen ikke tilfredsstiller 2-faselåsingsprotokollen (2PL).

### Oppgave 4 c

Lag en plan for de tre transaksjonene som tilfredsstiller 2PL, tegn presedensgrafen for denne nye planen, og vis at den er konfliktserialiserbar.

# Oppgave 5

Løs oppgavene 2, 3 og 5 fra Eksamen i INF3100 gitt V2004 (link til eksamenssettet finnes på fagets semesterside).

# Oppgave 6

Løs Eksamen i INF3100 gitt V2006 (link til eksamenssettet finnes på fagets semesterside).

### Slutt på obligatorisk oppgavesett 2

# UNIVERSITETET I OSLO

## Det matematisk-naturvitenskapelige fakultet

| | |
|---|---|
| Eksamen i : | INF3100/INF4100 — Databasesystemer |
| Eksamensdag : | Tirsdag 8. juni 2004 |
| Tid for eksamen : | 09.00 - 12.00 |
| Oppgavesettet er på : | 5 sider |
| Vedlegg : | Ingen |
| Tillatte hjelpemidler : | Kalkulator |

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

**Det er seks oppgaver**

**Alle har samme vekt, så beregn ca 30 minutter på hver**

## Oppgave 1 SQL og relasjonsalgebra

Gitt følgende datastruktur for en relasjonsdatabase (Primærnøkler er markert med **fete** typer, kandidatnøkler er i *kursiv*, fremmednøkler fremgår av attributt-navnene):

PERSON (**Fnr**, Etternavn, Fornavn, Adresse)
EKTESKAP (**Fnr-k** *Fnr-m*, ***Dato***, Etternavn-k, Etternavn-m)
NAVNESKIFTE (**Fnr**, **Dato**, Etternavn, Fornavn)

Nåværende navn lagres i tabellen PERSON. Følgelig lagres det navnet en person hadde umiddelbart før han/hun skiftet navn, i NAVNESKIFTE. Navnene i EKTESKAP er etternavnene etter vielsen. Dato er en standard SQL `DATE`, dvs. tekst i format `'2004-06-08'`, der rekkefølgen er år, måned og dag.

Løs følgende oppgave både med relasjonsalgebra og SQL:

> Finn navn og adresse til alle kvinner som ved en vielse i 2003 har byttet etternavn med sin brudgom. Ektefeller med samme etternavn skal ikke være med.

# Oppgave 2 Normalisering

En liten ungdomsklubb tilbyr medlemmene sine (alle har både fasttelefon og mobiltelefon) en rekke kurs. For å holde oversikt over hvem som deltar på hva, har de laget en tabell (i Excel) med følgende 8 kolonner:

Medlemsnr, Navn, Adresse, Fasttelefon, Mobiltelefon, Kurskode, Kursnavn og Instruktør. Følgende funksjonelle avhengigheter gjelder:

Medlemsnr bestemmer Navn, Adresse, Fasttelefon og Mobiltelefon
Mobiltelefon bestemmer Medlemsnr, Navn, Adresse og Fasttelefon
Adresse og Fasttelefon bestemmer hverandre
Kurskode bestemmer Kursnavn og Instruktør

**Oppgave 2 a** Hvilke kandidatnøkler og hvilken normalform har tabellen?

**Oppgave 2 b** Normaliser tabellen til BCNF

# Oppgave 3 Parsing, spørreplan og optimering

I denne oppgaven skal du vise din forståelse av hvordan en spørring parses, en logisk spørreplan lages, og hvordan spørreplanen kan optimeres. Vi skal bruke følgende relasjoner i denne oppgaven:

KUNDE(**kundeID**, kjønn, fornavn, etternavn, personnummer)
KONTO(**kontoNR**, kontotype, rentesats, opprettelsesdato)
KONTOEIERSKAP(**eierskapsID**, kundeID, eierskap, kontoNR)

Merk at primærnøkler er angitt med **dobbelunderstrekede og fete** typer. Andre kandidatnøkler er kun understreket.

Relasjonen KONTO inneholder bankkonti av flere typer og datoene kontoene ble opprettet. Det kan være flere kunder tilknyttet en konto. Dette er angitt i KONTOEIERSKAP der feltet eierskap enten er 'E' som betyr at vedkommende er eier av kontoen, eller 'D' som betyr at vedkommende disponerer den. Eieren er den som har opprettet kontoen. Attributtet opprettelsesdato i KONTO er en standard SQL DATE, dvs. tekst i format '2004-06-08', der rekkefølgen er år, måned og dag. Kjønn i KUNDE kan

være `'K'` eller `'M'`. Attributtet `kontotype` i `KONTO` er `'B'` for bruks-konto, `'S'` for sparekonto og `'L'` for lånekonto.

Vi skal bruke følgende spørring for å finne fornavn, etternavn og personnummer til kvinnelige kunder som har opprettet en sparekonto i 2003:

Oppgave 3 fortsetter på neste side

```
SELECT   KUNDE.fornavn, KUNDE.etternavn,
         KUNDE.personnummer
FROM     KUNDE, KONTO, KONTOEIERSKAP
WHERE    KUNDE.kundeID = KONTOEIERSKAP.kundeID AND
         KONTO.kontoNR = KONTOEIERSKAP.kontoNR AND
         KONTO.opprettelsesdato LIKE '2003%'   AND
         KONTO.kontotype = 'S'                 AND
         KUNDE.kjønn = 'K'
     AND
         KONTOEIERSKAP.eierskap = 'E'
```

Databasen har «clustered» indekser på primærnøklene. I tillegg er det indekser på attributtet `opprettelsesdato` i `KONTO` og på attributtene `kundeID` og `kontoNR` i `KONTOEIERSKAP`.

**Oppgave 3 a Parsing**

**i)** Bruk den enkle grammatikken på side 5 til å lage et parse-tre for spørringen ovenfor.

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

**Oppgave 3 b Logisk spørreplan**

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

**Oppgave 3 c Optimering**

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

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

# Oppgave 4 Indekser

**Oppgave 4 a**

Tegn og forklar hva tette (dense), sparsomme (sparse) og multi-nivå (multi-level) indekser er.

**Oppgave 4 b**

Forklar kort fordeler/ulemper med disse tre typene indekser, i hvilke tilfeller de anbefales brukt, og hvorfor.

Slutt på oppgave 4

Oppgave 5 og 6 står på neste side

# Oppgave 5 Logging

### Oppgave 5 a

Beskriv de to hovedtypene logger: optimistiske (Undo) og pessimistiske (Redo)

### Oppgave 5 b

Fortell hva et sjekkpunkt er. Legg hovedvekt på loggbruken.

# Oppgave 6 Transaksjonshåndtering

De to serialiserbarhetsbegrepene som er undervist i dette kurset, er basert på henholdsvis konflikt- og view-ekvivalens av eksekveringsplaner.

### Oppgave 6 a

Definer konfliktekvivalens og view-ekvivalens.

### Oppgave 6 b

Hva er det flest av, konflikt- eller view-serialiserbare eksekveringsplaner?

Hvilken tilleggsbetingelse må vi ha for at konflikt- og view-ekvivalens skal bli det samme?

Bevis at denne tilleggsbetingelsen sikrer at konflikt- og view-ekvivalens blir det samme.

Slutt på oppgavesettet

Naci Akkøk og Ragnar Normann

## Vedlegg til oppgave 3 — Grammatikk for parsing av spørsmål

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

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

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

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

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

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

# UNIVERSITETET I OSLO

## Det matematisk-naturvitenskapelige fakultet

| | |
|---|---|
| Eksamen i | INF3100/INF4100 — Databasesystemer |
| Eksamensdag: | 9. juni 2006 |
| Tid for eksamen: | 14.30 – 17.30 |
| Oppgavesettet er på 3 sider. | |
| Vedlegg: | Ingen |
| Tillatte hjelpemidler: | Kalkulator og ordbok |

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

*Les oppgavene nøye, og lykke til!*

## Oppgave 1 Relasjonsdatabasespråk (60%)

I denne oppgaven skal du bruke følgende fire relasjoner:

```
Film(filmID,title,productionyear)
Person(personID,gender,firstname,surname)
Participation(personID,filmID,participationname)
ParticipationRange(participationname)
```

Her er filmID og personID primærnøkler i henholdsvis Film og Person. I Participation er alle attributtene med i primærnøkkelen. Nullverdier er ikke tillatt i noe attributt, og 'F' og 'M' (for 'Female' og 'Male') er de eneste lovlige verdiene i gender. Fremmednøkler er gitt ved at de har samme navn som den tilsvarende primærnøkkelen. Relasjonen ParticipationRange har bare fem rader: 'cast', 'director', 'music', 'producer' og 'writing credits'.

### 1a (10%)

Bruk SQL til å definere de to tabellene Person og Participation. Ikke glem integritetsreglene!

### 1b (10%)

Uttrykk i relasjonsalgebra at participationname i Participation er fremmednøkkel til ParticipationRange.

*(Fortsettes på side 2.)*

### 1c (10%)

Gå ut fra at ingen film har mer enn en produsent (producer), og bruk relasjonsalgebra til å finne tittelen på alle filmer som har kvinnelig produsent og minst to regissører (director).

### 1d (15%)

Bruk SQL til å finne de filmene Charlie Chaplin har regissert. Resultattabellen skal ha fem attributter: Filmens tittel, produksjonsåret, antall kvinnelige skuespillere, antall mannlige skuespillere og det totale antall skuespillere. Sorter tabellen på det siste attributtet med det største antallet først.

*Hint*: Det er lov å bruke view.

### 1e (15%)

Lag ett SQL-spørsmål som finner ut hvilket år det ble produsert flest filmer, og hvor mange filmer det ble produsert dette året.

For å få full uttelling, skal denne oppgaven løses uten bruk av view.

## Oppgave 2 Strategi for disklagring (20%)

### 2a (5%)

Forklar kort hovedprinsippene for RAID 5 strategien og fortell hva som skiller RAID 5 fra RAID 4.

### 2b (15%)

Gi en grundigere beskrivelse av RAID 5 ved å beskrive hva som må gjøres av beregninger og skriving når en datablokk skal skrives til disk, og hva som må gjøres når en disk krasjer og må byttes ut.

*(Fortsettes på side 3.)*

# Oppgave 3 Strikt 2PL (20%)

Vi minner om at strikt 2PL (strikt tofaselåsing) skiller seg fra vanlig 2PL ved at i strikt 2PL frigjør en transaksjon ingen låser før den har gjort commit (eller abort).

## 3a (10%)

Vi skal utføre følgende to transaksjoner:

$$t_1 = r_1(x)r_1(z)w_1(z) \quad \text{og} \quad t_2 = r_2(x)r_2(z)w_2(x)r_2(y)w_2(y)$$

Lag en konfliktserialiserbar eksekveringsplan som ikke kan genereres av en strikt 2PL-scheduler for å utføre $t_1$ og $t_2$.

## 3b (10%)

Finnes det en ikke-seriell eksekvering av $t_1$ og $t_2$ som kan genereres av en strikt 2PL-scheduler? Begrunn svaret.