Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

EKSAMENSOPPGAVE I FAG TDT4145 – DATAMODELLERING OG DATABASESYSTEMER

Faglig kontakt under eksamen: Svein Erik Bratsberg

Tlf.: 73550382

Eksamensdato:	**26. mai 2009**
Eksamenstid:	**09.00-13.00**
Tillatte hjelpemiddel:	**D: Ingen trykte eller håndskrevne hjelpemiddel tillatt. Bestemt, enkel kalkulator tillatt.**

Språkform: Bokmål

Sensurdato: 16. juni 2009

Oppgave 1 – Datamodellering – 20 %

Vi ønsker å designe en database som lagrer informasjon om filmer, skuespillere og utgivelsesselskaper. Filmer kan opprinnelig være laget for kino eller TV. Senere kan filmene bli utgitt på video. Filmer finnes i forskjellige kategorier, slik som krim, drama, fantasi, SciFi, grøsser, romanse, familie og barn. En film kan ha flere kategorier, for eksempel SciFi og barn. En film har en eller flere regissører. En film har tittel, regissører, manusforfattere, lengde, utgivelsesår og lanseringsdato. Vi antar det ikke finnes to filmer som har samme tittel i løpet av et utgivelsesår.

Skuespillere har navngitte roller i filmer. Vi ønsker å vite hvilke roller skuespillere har i de forskjellige filmene. Vi ønsker også å vite navn, fødeår og fødeland for skuespillere, regissører og manusforfattere. Manusforfattere og regissører kan også være skuespillere i filmer de har laget. For selskaper vet vi både adresse og land.

Lag en ER -modell for en database som dekker kravene ovenfor. Husk å angi alle kardinalitetsrestriksjoner og nøkler. Databasen skal blant annet kunne brukes for å finne følgende (du trenger ikke å skrive disse spørringene):

- Navnet på alle rollene en gitt skuespiller har.
- Hvilke filmer som en gitt skuespiller opptrer i.
- Hvilke filmselskap som lager flest grøssere.
- Hvilke skuespillere er det som har hatt roller for filmer vist av et TV-selskap.

Forklar kort eventuelle forutsetninger du finner det nødvendig å gjøre.

Oppgave 2 – Relasjonsalgebra og SQL – 25 %

I denne oppgaven skal du ta utgangspunkt i en database som inneholder informasjon om øvingsgodkjenning fra alle fag ved NTNU våren 2009. Anta følgende relasjonsskjema (primærnøkler er understreket, attributter med samme navn som en annen tabell sin primærnøkkel er fremmednøkler):

```
Student(studnr,studnavn,epost)
Fag(fagnr,fagnavn)
Øving(fagnr,studnr,øvingsnr,godkjent)

godkjent har verdiene 'ja' eller 'nei'.
```

For de følgende spørringene skal du gi svaret i *relasjonsalgebra*. Er du usikker på hvordan symbolene for operatorene ser ut kan du skrive dem med ord, f.eks. "JOIN".

a) Finn navn på fag hvor student 'Ole Brum' har levert øvinger.

b) Finn navn på fag hvor 'Ole Brum' ikke har levert øvinger.

For hver av spørringene nedenfor skal du gi svaret i *SQL*.

c) Finn `epost` for alle studenter som har levert øvinger i faget med fagnr `'TDT4145'`.
d) Definer et view `Svarteliste(studnr,studnavn)` som er definert som alle studenter som har levert øvinger i faget med fagnr `'TDT4145'` og som ikke har 5 godkjente øvinger i faget.
e) Lag en liste av `studnavn`, antall godkjente øvinger (uansett fag), sortert etter antall godkjent øvinger, hvor de med høyest antall kommer først.
f) Finn fagnavn på faget/fagene som har flest ikke-godkjente øvinger.

Oppgave 3 – Lagring og indekser – 25 %

Gitt at det er definert en tabell på følgende måte i SQL:

```
CREATE TABLE Student (
     studnr INT NOT NULL PRIMARY KEY,
     studnavn VARCHAR(50),
     epost CHAR(8)
);
```

a) Forklar en måte (postformat) å lagre *studentposter* (records) slik at den utnytter plassen i blokker best mulig. Beskriv også hvordan vil du lagre studentpostene i en datablokk (blokkformat) for å utnytte plassen best mulig i blokka.
b) Anta at hver datablokk (page) i databasen har plass til 120 studentposter i gjennomsnitt. Det er totalt 24000 studentposter lagret i denne databasen. Anta at en B-tre-indekspost for denne tabellen er 12 byte, fyllgraden i B-treet er 67% og 80% for hashindeksen. Hver data- og indeksblokk i databasen er 8192 byte. Hvor mange blokker vil denne tabellen utgjøre i databasen ved de følgende lagringsmåter for tabellen. Begrunn svarene:
    i. Heapfil
    ii. Clustered B-tre-indeks
    iii. Clustered hash-indeks
    iv. Unclustered B-tre-indeks med heapfil
c) Av de måtene som er nevnt ovenfor, bestem hvilke/hvilken som er de/den mest hensiktsmessige måten å lagre tabellen gitt at de følgende SQL-setningene er dominerende. **NB**: Gi ett (muligens forskjellig) svar per deloppgave under. Begrunn svarene:

```
 i.   SELECT * FROM Student where studnr=10234;
ii.   INSERT INTO STUDENT VALUES (10234, 'Jens Jensen',
      'jensjens');
iii.  SELECT studnr FROM Student ORDER BY studnr;
 iv.  SELECT studnr, studnavn FROM Student ORDER BY studnr;
```

Oppgave 4 – Transaksjoner – 15 %

a) Gitt de følgende historiene

(1) $r_1(X);r_2(Z);r_3(X);r_1(Z);r_2(Y);r_3(Y);w_1(X);w_2(Z);w_3(Y);w_2(Y)$
(2) $r_1(X);r_2(Z);r_1(Z);r_3(X);r_3(Y);w_1(X);w_3(Y);r_2(Y);w_2(Z);w_2(Y)$
(3) $r_1(X);w_2(Z);r_1(Z);r_3(Z);r_3(Y);w_2(X);w_3(Y);r_2(Y);w_1(Z);w_2(Y)$

Tegn presedensgrafene for historiene og avgjør hvilke som er konfliktserialiserbare.
Hvis en historie er konfliktserialiserbar, skriv ned den konfliktekvivalente serielle historien.

b) Forklar hvordan ARIES minimaliserer antall datablokker som må leses inn under REDO-recovery.

Oppgave 5 – Normalisering – 15%

Gitt følgende tabell som registrerer en fast ukeplan for forelesninger ved NTNU:

`Forelesning(auditorium,bygning,faglærer,antallSittepl,ukedag,tid,fagnr)`

a) Foreslå hvilke funksjonelle avhengigheter som gjelder for denne tabellen.

b) Finn alle kandidatnøkler for tabellen

c) Vis hvordan tabellen bør normaliseres slik at alle de resulterende tabellene oppfyller kravene til Boyce-Codd normalform (BCNF).