Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

KONTINUASJONSEKSAMENSOPPGAVE I FAG TDT4145 – DATAMODELLERING OG DATABASESYSTEMER

Faglig kontakt under eksamen: Svein Erik Bratsberg

Tlf.: 99539963

Eksamensdato:	**10. august 2009**
Eksamenstid:	**09.00-13.00**
Tillatte hjelpemiddel:	**D: Ingen trykte eller håndskrevne hjelpemiddel tillatt. Bestemt, enkel kalkulator tillatt.**

Språkform: Bokmål

Sensurdato: 31. august 2009

Oppgave 1 – Datamodellering – 20 %

I denne oppgaven skal vi designe en database for å registrere artikler, forfattere, institusjoner og siteringer. En artikkel har en tittel og er skrevet av en eller flere forfattere. I artikkelen har hver forfatter en tilknytning til en institusjon, som kan være et universitet, en skole, et forskningsinstitutt eller en bedrift. Artikkelen er publisert i en konferansebok, en bok, en journal eller som en rapport. En artikkel vil ha start- og sluttsidenummer. Konferansebøker, bøker og journaler er utgitt av forlag, mens rapporter er tilknyttet en institusjon. Alle publiseringer har et årstall og en måned. En forfatter kan skifte tilknytning, slik at to artikler av samme forfatter kan være fra forskjellige institusjoner. En artikkel inneholder siteringer til andre artikler. Databasen skal inneholde en oversikt over disse siteringene.

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):

- Alle artikler en gitt forfatter har skrevet
- Hvilke artikler har sitert en gitt artikkel
- Hvilke artikler er gitt ut av en bestemt journal.
- Hvor mange selvsiteringer finnes for en gitt forfatter, dvs. hvor mange av sine egne artikler har denne forfatteren sitert.

Forklar kort eventuelle forutsetninger du finner det nødvendig å gjøre.

Oppgave 2 – Relasjonsalgebra og SQL – 20 %

I denne oppgaven skal du ta utgangspunkt i en database som inneholder informasjon om filmer og skuespillere. Anta følgende relasjonsskjema (primærnøkler er understreket, attributter med samme navn som en annen tabell sin primærnøkkel er fremmednøkler):

```
Film(filmtittel,filmår,lengde,filmkategori)
SpillerI(navn,fødselsår,filmtittel,filmår,rollenavn)
Skuespiller(navn,fødselsår,fødeland)
```

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 fødeland for alle skuespillere som spiller i filmen med tittel ’Cinema paradiso’.

b) Finn filmtitler hvor skuespilleren med navn ’Jim Carrey’ og fødselsår 1962 ikke spiller.

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

c) Lag en sortert liste av filmtittel, filmår som er sortert på stigende filmår.

d) Finn rollenavn i alle filmer med filmkategori 'komedie'.

e) Lag en liste av filmkategori, summen av lengde for alle filmer av en gitt filmkategori. Listen skal være sortert etter synkende sum av lengde.

f) Finn navn og fødselsår for de(n) skuespilleren(e) som har spilt i flest filmer. NB: Det kan være flere skuespillere som oppfyller kriteriet.

Oppgave 3 – Lagring og indekser – 20%

Se på følgende tabell:

Ansatt(*ansattid:* integer, lønn*:* integer, *alder:* real, *avdnr:* integer)

Det er en clustered index på *ansattid* og en unclustered index på *alder*.

a) (4%) Hvordan vil du bruke indeksene til å implementere restriksjonen at *ansattid* er nøkkel?
b) (4%) Gi et eksempel på en oppdatering som er blitt raskere pga. indeksene.
c) (4%) Gi et eksempel på en oppdatering som er blitt tregere pga. indeksene.
d) (8%) Hvor mange blokker vil denne tabellen og indeksene bruke hvis tabellen er lagret som en clustered hashindeks på *ansattid* og indeksen på *alder* som et unclustered b+-tre. Anta at integer og real begge er 4 byte og at alle blokker er 8192 byte stor. Tabellen inneholder 20000 ansatte jevnt fordelt i alderen 21 til 70 år. En referanse til en blokk er 4 byte. Hashfila har 80% fyllgrad, mens B+-treet har 67% fyllgrad.

Oppgave 5 – Logging, recovery og transaksjoner – 20%

a) Forklar sammenhengen mellom "**Write-Ahead Logging**" og **NO-FORCE** buffer-metoden.
b) Gitt følgende historier:
   S_1: $r_3(Z)$; $r_3(Y)$, $w_2(Y)$; $r_2(Z)$; $w_2(X)$; $w_1(X)$; c_2; $r_1(X)$; c_1; $r_3(X)$; c_3;
   S_2: $r_3(Z)$; $w_2(X)$; $w_2(Y)$; $r_1(X)$; $r_3(X)$; $r_2(Z)$, c_2; $r_3(Y)$, c_3; $w_1(X)$; c_1;
   S_3: $r_2(Z)$; $w_2(X)$; $w_2(Y)$, c_2; $w_1(X)$; $r_1(X)$; c_1; $r_3(X)$; $r_3(Z)$; $r_3(Y)$; c_3;
   Avgjør recovery-egenskapene (strict, cascadeless (ACA), recoverable, ingen av delene) til historiene over.
c) I en **end_checkpoint**-loggpost lagres "**Dirty Page Table**" og "**Transaction table**". Forklar hvordan de to strukturene brukes under "node crash recovery".
d) ACID beskriver fire egenskaper ved transaksjoner. Knytt hver av de følgende begrepene til en eller flere av de fire egenskapene:
   a. serialiserbarhet
   b. gjenopprettbarhet
   c. ACA - Avoiding Cascading Aborts
   d. restriksjoner

e. strict historie
f. rollback
g. loggpost

Oppgave 6 – Normalisering – 20%

Gitt følgende tabell for å registrere resultater fra eksamener våren 2009:

Navn	Snr	Emnenr	Eksamensdato	Kandidatnr	Karakter
Jon Bang	1	TDT4145	26.05.2009	10001	A
Jon Bang	1	TDT4140	27.05.2009	10002	B
Per Dal	2	TDT4145	26.05.2009	10002	C
Eva Nord	3	TDT4220	26.05.2009	10001	C
Gry Sand	4	TDT4145	26.05.2009	10003	A
Gry Sand	4	TDT4140	27.05.2009	10001	D

a) Hvilke funksjonelle avhengigheter (eng: functional dependencies) gjelder for tabellen beskrevet over? Gjør rede for de forutsetningene du eventuelt bygger på.

b) Tabellen har en rekke uheldige egenskaper (anomalier). Gjør rede for disse og gi eksempler på problemer som kan oppstå.

c) Hva er den høyeste normalformen som oppfylles av tabellen? Svaret må begrunnes.

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