Tentamen Databases

19 april 2011

17:00 - 20:00, Educatorium-Gamma

- Scheur de antwoordvellen doormidden.
- Maak elke vraag op een ander vel.
- Vermeld op elk vel je naam en studentnummer. Indien één van deze zaken ontbreekt, wordt het vel niet nagekeken.
- Toon bij het inleveren je collegekaart.
- Schrijf en formuleer duidelijk.
- Bij elke vraag wordt verwacht dat je laat zien hoe je aan het antwoord komt (tenzij anders wordt vermeld).
- Je mag een A4 met aantekeningen raadplegen.
- Het tentamen duurt 3 uur en bestaat uit 6 opgaven.
- Gebruikte afko's:
  2PL = Two-phase locking
  2PC = Two-phase commit
  BCNF = Boyce-Codd normaalvorm
  COORD = Coordinator
  CTP = Cooperative termination protocol
  DP = Dependency preserving
  FD = functional dependency
  RA = Relationele algebra
  SQL = SQL (Structured Query Language)
- Vergeet niet de vakevaluatie in te vullen. Zie de education pagina.
- **Succes!**

Puntentelling: (totaal = 104 punten; 100 punten geeft een 10)
1: 20 punten
2: 7x2 + 4 = 18 punten
3: 15 punten
4: 20 punten
5: 10 + 5 = 15 punten
6: 4x4 = 16 punten

1

1 Algemeen

Geef van de volgende beweringen aan of zij correct zijn of niet. Een simpel JA of NEE volstaat. Er hoeft geen toelichting gegeven te worden.

1. Dat elke FD $X \rightarrow Y$ uit de oorspronkelijke FD-set F in één van de relatieschema's past, is een noodzakelijke voorwaarde voor de DP-eigenschap.
2. Als een schema in 3NF is, is dit schema ook in BCNF.
3. Is de volgende herschrijfregel geldig?
   $X \rightarrow Z \Rightarrow X \rightarrow Y, Y \rightarrow Z$
4. Is de volgende herschrijfregel geldig?
   $X \rightarrow Z, X \rightarrow Y \Rightarrow Y \rightarrow Z$
5. Voor elk relatieschema is een verliesvrije DP BCNF decompositie mogelijk.
6. Je mag nooit database-data naar disk schrijven voordat commitment heeft plaatsgevonden.
7. Het recovery-mechanisme op basis van UNDO+REDO (gecombineerd) geeft de IO-manager de meeste vrijheid.
8. Het 2PC-protocol kan geblokkeerd raken als één van de participants nog niet gestemd heeft.
9. Elke seriële schedule wordt geaccepteerd door een 2PL-scheduler.
10. De hash table is een geschikte indexstructuur voor range-queries.

2 Functional dependencies

We willen een database opzetten ten behoeve van genealogisch onderzoek. Elke persoon in deze database wordt geïdentificeerd met behulp van een unieke identifier: `pid`.
In de tabel `Persoonsbeschrijving` leggen we een aantal eigenschappen van elke persoon vast; `pid` is de primary key.

```
Persoonsbeschrijving
(pid, achternaam, voorletters, voorvoegsel, gebdatum, sterfdatum, geslacht)
```

De volgende tabel (die nog niet genormaliseerd is) bevat familierelaties van de persoon aangeduid met `pid`. De attributen `vader, moeder` en `partner` nemen waarden aan van relevante pid's. De attributen `van` en `tot` geven aan wanneer het partnerschap tussen `pid` en `partner` gold.
Ga uit van algemeen geldende veronderstellingen ten aanzien van (biologische) familierelaties.

```
Familie (pid, vader, moeder, partner, van, tot)
```

(i) Geef voor de volgende FD's aan of deze wel of niet gelden. Toelichting is niet nodig.

1. pid → vader, moeder
2. vader, moeder → pid
3. pid → partner
4. pid → partner, van, tot
5. partner, van, tot → pid
6. vader → moeder
7. partner, van → moeder

(ii) Waarom voegen we geen attributen `zoon` en `dochter` toe aan de relatie?

3 Schemaontwerp

We hebben een schema $R(ABCDEFGH)$ en een set FDs
$F = \{A \rightarrow C, AB \rightarrow DG, C \rightarrow EF, EF \rightarrow BD\}$.
Geef een verliesvrije, DP 3NF-decompositie van R. Laat zien welke methode u gebruikt.

4 Queries

Een uitgeverij van een tijdschrift over console games houdt een database bij van de games die in het tijdschrift gerecenseerd worden. Een recensie betreft een spel (uniek identificeerbaar op basis van de naam) voor een bepaald platform (Xbox360, Playstation3, Wii) en is opgenomen in een uitgave van het tijdschrift (uniek identificeerbaar op basis van het jaar en de maand). Een spel krijgt bij een recensie een bepaalde beoordeling. Een spel kan meerdere keren gerecenseerd worden - al dan niet in dezelfde uitgave van het tijdschrift -, mits het telkens een ander platform betreft. Van elk spel wordt naast de naam het genre en de uitgever van het spel geregistreerd. Van elke uitgave van het tijdschrift wordt het jaar en de maand van uitgave bijgehouden, alsmede het aantal bladzijden en de oplage. Het databaseschema van de database is als volgt (primary keys zijn in cursief):

Uitgave (*jaar*, *maand*, bladzijden, oplage)
Recensie (*naam*, *platform*, jaar, maand, beoordeling)
Spel (*naam*, genre, uitgever)

Gegeven zijn de volgende queries:

Q1: Geef het genre en de uitgever van de spellen die minstens eenmaal gerecenseerd zijn.
Q2: Geef de oplage van de uitgave in 2010 waarin de spellen de laagste gemiddelde beoordeling krijgen van alle uitgaven in dat jaar.
Q3: Geef het genre en de uitgever van de spellen die nooit gerecenseerd zijn.

Hieronder volgen expressies in de RA of in SQL. Geef voor elke query aan welke expressie(s) met de query corresponderen. De relatie tussen queries en expressies is many-to-many en optioneel. Ga *niet* uit van de specifieke beperkingen van SQLite.

E1: $\pi_{genre,uitgever}((\pi_{naam}(Recensie) \cap \pi_{naam}(Spel)) \bowtie Spel)$

E2: $\pi_{genre,uitgever}((\pi_{naam}(Spel) - \pi_{naam}(Recensie)) \bowtie Spel)$

E3: $\pi_{genre,uitgever}((\pi_{naam}(Recensie) \cup \pi_{naam}(Spel)) \bowtie Spel)$

E4: $\pi_{genre,uitgever}((\pi_{naam}(Recensie) \div \pi_{naam}(Spel)) \bowtie Spel)$

E5: $\pi_{genre,uitgever}((\pi_{naam}(Recensie) - \pi_{naam}(Spel)) \bowtie Spel)$

E6: $\pi_{genre,uitgever}(Spel \bowtie Recensie)$

E7:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) >= ALL (
    SELECT AVG(Recensie.beoordeling)
    FROM Recensie, Uitgave
    WHERE Recensie.jaar = Uitgave.jaar
    AND Recensie.jaar = 2010
    GROUP BY Recensie.jaar)
```

E8:

```
SELECT Spel.genre, Spel.uitgever
FROM Spel
WHERE Spel.naam IN (
    SELECT Recensie.naam
    FROM Recensie)
```

E9:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.maand = Uitgave.maand
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) <= ALL (
    SELECT AVG(Recensie.beoordeling)
    FROM Recensie, Uitgave
    WHERE Recensie.jaar = Uitgave.jaar
    AND Recensie.maand = Uitgave.maand
    AND Recensie.jaar = 2010
    GROUP BY Recensie.jaar, Recensie.maand)
```

E10:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.maand = Uitgave.maand
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) < ALL (
    SELECT AVG(Recensie.beoordeling)
    FROM Recensie, Uitgave
    WHERE Recensie.jaar = Uitgave.jaar
    AND Recensie.maand = Uitgave.maand
    AND Recensie.jaar = 2010
    GROUP BY Recensie.jaar, Recensie.maand)
```

E11:

```
SELECT Spel.genre, Spel.uitgever
FROM Spel
WHERE Spel.naam NOT IN (
    SELECT Spel.naam
    FROM Spel, Recensie
    WHERE Spel.naam = Recensie.naam)
```

E12:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.maand = Uitgave.maand
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) >= SOME (
    SELECT AVG(Recensie.beoordeling)
    FROM Recensie, Uitgave
    WHERE Recensie.jaar = Uitgave.jaar
    AND Recensie.maand = Uitgave.maand
    AND Recensie.jaar = 2010
    GROUP BY Recensie.jaar, Recensie.maand)
```

E13:

```
SELECT Spel.genre, Spel.uitgever
FROM Spel, Recensie
WHERE Spel.naam = Recensie.naam
```

E14:

```
SELECT Spel.genre, Spel.uitgever
FROM Spel
WHERE Spel.naam NOT IN (
    SELECT Recensie.naam
    FROM Recensie
    WHERE Recensie.naam = Spel.naam)
```

E15:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.maand = Uitgave.maand
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) <= ALL
```

E16:

```
SELECT Uitgave.oplage
FROM Recensie, Uitgave
WHERE Recensie.jaar = Uitgave.jaar
AND Recensie.maand = Uitgave.maand
AND Recensie.jaar = 2010
GROUP BY Recensie.jaar, Recensie.maand, Uitgave.oplage
HAVING AVG(Recensie.beoordeling) <= ALL (
    SELECT AVG(Recensie.beoordeling)
    FROM Recensie, Uitgave
    WHERE Recensie.jaar = Uitgave.jaar
    AND Recensie.maand = Uitgave.maand
    GROUP BY Recensie.jaar, Recensie.maand)
```

E17:

```
SELECT Spel.genre, Spel.uitgever
FROM Spel
WHERE NOT EXISTS (
    SELECT Recensie.naam
    FROM Recensie
    WHERE Recensie.naam = Spel.naam)
```

5 Concurrency

Hieronder zijn twee schedules gegeven.

(i) Stel voor elk van de schedules de complete precedentiegraaf op. Geef aan of deze schedules serializeerbaar zijn of niet. Licht toe. Geef zo mogelijk de equivalente seriële schedules.

(ii) Geef eveneens aan of de schedules getolereerd worden door een 2PL-scheduler. Geef hierbij een korte toelichting.

S1

T1	T2	T3	T4	T5
			r(z)	
w(z)				
	w(x)			
		r(z)		
r(x)				
	r(y)			
				r(x)
r(z)				
			w(y)	
				w(y)

S2

T1	T2	T3	T4	T5
			w(z)	
r(z)				
	r(x)			
		w(z)		
w(x)				
	w(y)			
				w(x)
w(z)				
			r(y)	
				r(y)

6 Query processing

We gaan uit van twee relatieschema's R en S die één attribuut A gemeenschappelijk hebben. We definiëren een algebraïsche operator $\ltimes$ (semi-join) als volgt:

> $R \ltimes S$ bevat de tuples r in R waarvoor geldt dat er een tuple s in S bestaat met $r.A = s.A$.

Met andere woorden, de semi-join verwijdert uit R de tupels die de natural join met S niet "overleven".
Je hoeft bij deze opgave geen formele bewijzen te leveren.

(i) Hoe distribueert de selectie over de semi-join? (Gevalsonderscheid kan nodig zijn.)

(ii) Hoe distribueert de projectie over de semi-join?

(iii) Beargumenteer waarom de volgende algebraïsche equivalentie geldig is:

$$R \bowtie S \equiv (R \ltimes (\pi_A S)) \bowtie S$$

(iv) Stel we hebben een gedistribueerde omgeving met R op site 1 en S op site 2. Het resultaat van $R \bowtie S$ wordt gevraagd op site 2. Beschrijf hoe de semi-join een rol kan vervullen bij de processing van deze gedistribueerde query.

Einde