Tietokannan hallinta, kevät 2002 **Harjoitus 6 (6.-10.5.) (viimeinen)**

Läsnäolo: 4 tehtävää

Huom. Helatorstaina 9.5. ei ole harjoituksia. Torstain ryhmiin kuuluvat vierailevat muissa ryhmissä. Ylimääräinen ryhmä: pe 10.5. klo 14-16 salissa B453.

1. (**) Oletetaan, että lokiin on kirjoitettu myös lukutoimintoja kuvaavat read-tietueet kuten seuraavassa esimerkissä (lokin sisältö järjestelmän romahtaessa) :

100: [start, T3]
101: [read, T3, C]
102: [write, T3, B, 15, 12]
103: [start, T2]
104: [read, T2, B]
105: [write, T2, B, 12, 18]
106: [start, T1]
107: [read, T1, A]
108: [read, T1, D]
109: [write, T1, D, 20, 25]
110: [read, T2, D]
111: [write, T2, D, 25, 26]
112: [read, T3, A]

a) Mitä tiedetään lokin perusteella muuttujien alkuarvoista?
b) Commit-tietueita ei ole merkitty lokiin, mutta oletetaan, että lopputilanteessa ainakin T3 on kesken eli ei sitoutunut. Voisivatko T1 ja T2 olla sitoutuneita? Millainen elvytys tässä tarvitaan?

2. Tietokannan eheysrajoitteen mukaan tietoalkioilla X ja Y pitää olla sama arvo. Tarkastellaan transaktiojoukon {T1, T2, T3} ajoitusta

T1: read-item(X,v);	a) Ovatko transaktiot oikeellisia?
T1: v:=v+1;	
T1: write-item(X,v);	b) Mikä on kunkin transaktion eristyneisyystaso?
T2: read-item(X,w);	
T2: write-item(Y,w);	c) Mitä lokitietueita syntyy, kun tietokannassa
T2: commit;	on aluksi X = Y = Z = 0?
T3: read-item(Y,p);	
T1: abort;	d) Säilyykö tietokannan eheys?
T3: write-item(Z,p);	
T3: commit.	e) Kuinka ankara 2PL estää esimerkkiajoituksen syntymisen?

3. Oletetaan, että transaktiot T1 ja T2 noudattavat seuraavaa ajoitusta:

```
1   T1: read_item(X);
2         T2: read_item(X);
3   T1: write_item(X);
4   T1: read_item(Y);
5         T2: read_item(X);
6         T2: write_item(X);
7   T1: write-item(Y);
8   T1: commit;
9         T2: commit;
```

a) Mitkä operaatioparit tässä ajoituksessa konfliktoivat? Mikä eristyvyysanomalia kuhunkin konfliktiin liittyy?
b) Anna edelleen kolme esimerkkiä ei-konfliktoivasta operaatioparista ja perustele, miksi kukin niistä ei konfliktoi. Esimerkkien tulee olla erilaisia (kussakin eri peruste).

4. a) Millä ehdolla lukulukon korotus kirjoituslukoksi voi tulla kysymykseen?

b) Kuinka lukon alennus (downgrading) kirjoituslukosta lukulukoksi suoritettaisiin? (Mitä muutoksia operaatioihin read_lock, write_lock, unlock?) Arvioi myös alennuksen käytännön merkitystä.

5. Oletetaan kysely

select * from employee where dno=5;

a) Mitkä tekijät vaikuttavat lukkojen granulaarisuuden valintaan kyselyä toteutettaessa?
b) Eräissä järjestelmissä on mahdollisuus valita lukituksen granulaarisuus dynaamisesti eli vaihtaa granulaarisuutta jopa kesken kyselyn (esim. sivulukituksesta rivilukitukseen tai päinvastoin). Mihin perustuen muutos voitaisiin tehdä? (Vihje: Kyselyn suorituksesta voidaan kerätä suorituksen aikana tilastotietoa.)

6. (**) Tietokannan hallintajärjestelmässä sovelletaan ankaraa kaksivaiheista lukituskäytäntöä. Lukittavat tietoalkiot ovat sivuja. Mitä lukkoja otetaan seuraavan transaktion suorituksessa, kun

a) relaatioon EMPLOYEE ei ole hakemistoa attribuutilla dno,

b) relaatioon EMPLOYEE on hajautushakemisto attribuutilla dno (joko ensisijaisena saantipolkuna tai oheishakemistona)?

```
begin transaction;
update EMPLOYEE set salary = salary + 500
   where dno = 5;
commit transaction.
```

c) Entä lukot päivitysoperaatiossa, joka korottaa kaikkien Jokisten palkkaa, ja käytössä on sukunimen mukainen ISAM-hakemisto?

Muista vastata kurssikyselyyn: http://ilmo.cs.helsinki.fi/kurssit/servlet/Valinta