Tentti: Transaktionhallinta tietokantajärjestelmissä **T-106.5221**
Exam: Transaction Management in Databases P 10.04.2015 (two pages)

Kirjoitathan ystävällisesti jokaiseen palauttamaasi paperin ylälaitaan selvästi 'T-106.5221, 10.04.2015', koko nimesi, opiskelijanumerosi, koulutusohjelmasi sekä montako paperia palautit yhteensä. Laskin on sallittu mutta sitä ei tarvita. Kysymyksiä on yhteensä viisi ja jokaisen kysymyksen maksimipistemäärä on kuusi.
Please write on top of every answer sheet clearly 'T-106.5221, 10.04.2015', your full name, your student number, your study program, and the total nr of sheets that you returned. Calculators are allowed but are not needed. There are a total of five questions, each of which is worth six points.

1. Määritä lyhyesti mutta selkeästi seuraavat käsitteet a-d. *Explain briefly but clearly the following terms a-d.*

   (a) *Transaction.* (b) *SQL isolation level 'Repeatable Read'.*

   (c) *Physical logging.* (d) *Conditional lock.*

   (e) *Contrast the 'Wait-Die' and 'Wound-Wait' policies from an older transaction's point of view.*

2. (a) Onko seuraava ajoitus S_1 konflikti-sarjallistuva? Entä onko S_1 ei-sarjallinen? Voiko ajoitus S_1 sisältää mitään eristyvyysanomalioita? Perustele vastauksesi lyhyesti.
   Is the following schedule S_1 conflict serializable? Is S_1 nonserial? Can schedule S_1 contain any isolation anomalies? Justify your answers briefly.
   S_1: $B_3R_3[z]B_2W_2[x]W_2[y]B_1R_1[x]R_3[x]R_2[z]\mathbf{C_2}R_3[y]\mathbf{C_3}W_1[x]\mathbf{C_1}$

   (b) Olkoon ajoitus S_2 sekä tietokannan alkutila D kuten alla. Käytetään avainvälilukitusta. Mitä lukkoja ajoituksen S_2 transaktiot varaavat ja milloin lukot vapautetaan avainvälilukituskäytännössä? Esitä vastauksesi taulukkomuodossa ja mainitse jos jotain tarvittavaa lukkoa ei voida myöntää.
   Consider schedule S_2 with its database state at onset D as shown below. We apply the key-range locking protocol. What locks are acquired by the transactions in schedule S_2 and when are those locks released under key-range locking? Show your results in a tabular format and mention what, if any, is the lock that cannot be granted.
   S_2: $B_2D_2[5]B_1R_1[x_1, > 3]\mathbf{C_2}B_3R_3[x_2, > 4]I_1[5, 2]\mathbf{C_1}\mathbf{C_3}$
   Tietokannan alkutila *Database initial state* $D = \{(4, 2); (5, 1); (6, 0)\}$.

   (c) Mikä eristyvyysanomalia (likainen kirjoitus, likainen luku, toistokelvoton luku) tai haamuilmiö ajoituksessa S_2 mahdollisesti esiintyy? Perustele lyhyesti.
   Does the schedule S_2 contain any isolation anomaly (dirty write, dirty read, unrepeatable read) or a phantom phenomenon? Justify briefly.

3. Olkoon alla olevat neljä puskurinhallintakäytäntöä P1, P2, P3 ja P4. Vastaathan tämän pohjalta kohtaan (a) valitsemalla jokaiselle seuraavalle kuvaukselle (D1-D4) jonkin *yhden* käytännöistä P1-P4, joka parhaiten kuvaa sitä. Kaikkia käytäntöjä (P1-P4) ei tarvitse käyttää.

   $P1$ =Force & Steal, $P2$ =No-Force & Steal, $P3$ =Force & No-Steal, $P4$ =No-Force & No-Steal.

   Assume the above four buffer management policies P1, P2, P3 and P4 as described above. Based on this, please answer (a). For each of the following descriptions (D1-D4), indicate which *one* policy P1-P4 best describes it. Not all policies (P1-P4) need to be used.

   (a) (D1) *This policy gives maximum flexibility for buffer management.*

   (D2) *This policy does not prevent dirty data from reaching the database disk but is guaranteed to speed up the REDO phase in recovery.*

   (D3) *A page updated by transaction T may be evicted from the buffer frame before T commits, but following commit, all the updated pages of the transaction T will be on the database disk.*

   (D4) *This policy does not require flushing a modified page to the database disk at commit, but prevents the flushing of a dirty page before commit.*

(b) Olkoon tietokantajärjestelmä $Syst_1$, joka perustuu Force & Steal käytäntöön WALin kanssa. Kun $Syst_1$ romahtaa, miksi ARIESin elvytyksessä tarvitaan silti REDO-vaihetta? Kommentoi $Syst_1$:n suorituskykyä normaalitoiminnan aikana. Epäonnistuisiko elvytys jos *ei* käytettäisikään WALia?
*Consider a DBMS, called $Syst_1$, that is based on the Force & Steal policy using WAL. Following a crash in $Syst_1$, why do we still need the REDO-phase when recovering with ARIES? Comment on the performance of $Syst_1$ during normal operation. Would the recovery fail if WAL were *not* used?*

4. Lokin sisältö häiriön sattuessa on kuten alla. Levyversion PageLSN sivulle p on 106. (Oletetaan, että kaikki peruutusvaiheen käänteisoperaatiot voidaan suorittaa fyysisesti.)
The contents of the log saved on disk in a system crash are as shown below. The PageLSN value of the disk version for page p is 106. (It is assumed all the operations in the UNDO-phase can be performed physically.)

		105:	<T_1, B>
		106:	<T_1, I, p, x_1, 6, 105>
		107:	<T_2, B>
101:	<begin-checkpoint>	108:	<T_2, I, p, x_2, 13, 107>
102:	<transaction-table, {}>	109:	<T_3, B>
103:	<page-table, {}>	110:	<T_3, I, p, x_3, 15, 109>
104:	<end-checkpoint>	111:	<T_3, I, p, x_4, 11, 110>
		112:	<T_3, A>
		113:	<T_3, I^{-1}, p, x_4, 110>
		114:	<T_1, C>

(a) Mitä saadaan ARIES analyysivaiheen tuloksena? *What do we get as a result of the ARIES analysis phase?*

(b) Suorita ARIESin TOISTO-vaihe, esitä vastauksesi taulukkomuodossa.
Perform the ARIES REDO-phase, show your answer in a tabular format.

(c) Suorita ARIESin PERUUTA-vaihe. *Perform the ARIES UNDO-phase.*

5. Käytössämme on monirakeisuuslukinta tietokannassa b, jossa on relaatio r. Transaktio T_1 haluaa lukea kyseisestä relaatiosta r erään monikon t_1, kun taas toinen transaktio T_2 haluaa samasta relaatiosta r samanaikaisesti lukea joitakin monikkoja ja päivittää monikkoa $t_2 \neq t_1$.
Multi-granular locking is to be used in a database b consisting of a relation r. Transaction T_1 needs to read from relation r a certain tuple t_1 while another transaction T_2 needs to simultaneously read some tuples and update another tuple $t_2 \neq t_1$ from the same relation r.

(a) Mikä lukko T_1:n tulisi asettaa tietokantaan b? Entä relaatioon r? Entä monikkoon t_1?
What lock should T_1 place on database b? On relation r? On tuple t_1?

(b) Jos transaktio T_2 asettaisi SIX-lukon relaatioon r, olisiko tällä mitään haittoja transaktio T_1:n lukuoperaation kannalta relaatiossa r? Perustele lyhyesti.
If transaction T_2 were to place a SIX-lock on relation r, would this have any disadvantages for the read operation of transaction T_1 in relation r? Justify briefly.

(c) Kun käytetään päivityslukkoa (U-lukkoa) ja T_2 tarvitsee U-lukon monikkoon t_2, mikä lukko tulisi asettaa relaatioon r? Entä tietokantaan d? Selitä lyhyesti ja selkeästi miten U-lukko eroaa S-lukosta ja miten korottamalla U-lukko X-lukoksi voidaan estää lukkiumat.
When using Update-Mode locks (U-locks) and T_2 needs to acquire a U-lock on tuple t_2, what lock should transaction T_2 obtain on relation r and what lock on database d? Explain briefly and clearly how a U-lock differs from an S-lock and how a U-lock can be used to avoid deadlocks through an upgrade to an X-lock.