TAPAHTUMIEN HALLINTA

JOUNI HUOTARI

(7.3.2012)

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JYVÄSKYLÄ UNIVERSITY OF APPLIED SCIENCES

TEHTÄVIÄ/KYSYMYKSIÄ

- Määrittele
  - tapahtuma (transaction)
  - tapahtumien hallinta
- Mitä ovat tapahtuman ACID-ominaisuudet?
- Mitä ovat seuraavat lukot:
  - Shared lock
  - Exclusive lock
  - Deadlock
- Mikä ero on optimistisella ja pessimistisellä lukituksella?
- Miksi tapahtumien hallinta on tärkeää?
- Mitä ongelmia voi esiintyä jos useita tapahtumia suoritetaan rinnakkain?
- Miten ongelmat voitaisiin ratkaista?

MENETELMÄ

- Tee määritykset ensin yksin
- Vertaa määrityksiäsi vierustoverisi kanssa
- Miettikää yhdessä (pienryhmissä) vastaukset miksi-, mitä- ja miten-kysymyksiin (Case: pankki tai lennonvarausjärjestelmä)
- Kootaan vastaukset yhteisesti

MÄÄRITELMÄT

- Tapahtuma (transaction) on käyttäjän antama komento tai ohjelma (tai sen osa), joka hakee tai muuttaa tietoa tietokannasta
- Tapahtumien hallinta (tai käsittely) tarkoittaa relaatiotietokannan kykyä hallita tapahtumia
  - jos jokin menee vikaan tapahtuman aikana, ohjelman käskyt perutetaan (rollback); muutoin ohjelman käskyt suoritetaan (commit)
  - esimerkiksi pankkitapahtuman käsittely: pitää tarkistaa sekä maksutili että saajatili

TYYPILLISIÄ ONGELMIA

- Menetetyn päivityksen ongelma (lost update problem): tapahtuman suorittama päivitys jää toteutumatta, koska toinen tapahtuma korvaa sen (ts. kirjoittaa päälle)
- Tilapäisen päivityksen ongelma (temporary update problem, uncommitted dependency problem, "dirty read"): tapahtuman päivitys perustuu tietoon, joka muuttuu toisen tapahtuman keskeytyksen seurauksena
- Virheellisen yhteenvedon ongelma (incorrect summary problem): tapahtuma hakee tietoja useasta paikasta samalla kun toinen tapahtuma päivittää tietoja

ESIMERKKEJÄ ONGELMISTA

- Menetetyn päivityksen ongelma

Time	T_1	T_2	bal_x
t_1		begin_transaction	100
t_2	begin_transaction	read(bal_x)	100
t_3	read(bal_x)	$bal_x = bal_x + 100$	100
t_4	$bal_x = bal_x - 10$	write(bal_x)	200
t_5	write(bal_x)	commit	90
t_6	commit		90

- Tilapäisen päivityksen ongelma

Time	T_3	T_4	bal_x
t_1		begin_transaction	100
t_2		read(bal_x)	100
t_3		$bal_x = bal_x + 100$	100
t_4	begin_transaction	write(bal_x)	200
t_5	read(bal_x)	⋮	200
t_6	$bal_x = bal_x - 10$	rollback	100
t_7	write(bal_x)		190
t_8	commit		190

TAPAHTUMILLA PITÄÄ OLLA ACID-OMINAISUUDET

- **A**tomicity: tapahtuma suoritetaan joko kokonaisuudessaan tai ei lainkaan
- **C**onsistency: tapahtuma siirtää tietokannan eheästä tilasta toiseen eli tapahtuman on säilytettävä kannan eheys
- **I**solation: tapahtuman kantaan tekemät muutokset eivät saa näkyä muille tapahtumille ennen kuin se on lopettanut suorituksensa menestyksellisesti (committed; tapahtuma on eristetty muista tapahtumista)
- **D**urability: sitoutuneen tapahtuman kantaan tekemät muutokset ovat pysyviä, eivätkä ne saa missään tapauksessa kadota

LÄHTEET JA LINKIT

- Thomas Connolly & Begg, C. (2002), Database Systems – a Practical Approach to Design, Implementation, and Management, 3. painos, Addison-Wesley
- Mauri Leppänen (1999), Tietokannan hallintajärjestelmät, luentomoniste, Jyväskylän yliopisto
- Ryan K. Stephens & al. (2001), SQL Trainer Kit, Edita
- WWW, etsi esim. hakusanalla "Transaction processing" tai "Transaction management" + liitä tarkennus, esim. "lost update problem"
- Jaakko Rantanen (2006), Samanaikaisuuden hallinta ohjelmoinnin näkökulmasta:
  http://homes.jamk.fi/~huojo/opetus/IIO30200/samanaik_hallinta.pdf

ESIMERKKI LUKITUKSEN KÄSITTELYSTÄ

- Case: Microsoft Access, joka käyttää tiedostopohjaista lukitusta (.ldb) ja jonka tauluihin voidaan määrittää myös rivikohtainen lukitus
- Avaa yhteisellä levyllä oleva mdb-tiedosto
- Katso, mitä ko. hakemistoon ilmestyy
- Tutki ldb-tiedoston sisältö
- Tee muutos samaan tauluun kaverin kanssa, esim. FIRMA-tauluun
- Mitä tapahtuu, kun tallennat rivin?
- Miten avaat Access-tietokannan Exclusive-tilaan?