

# Transacties en multi-user gebruik

[wim.bertels@ucll.be](mailto:wim.bertels@ucll.be)

Naamsvermelding-NietCommercieel-GelijkDelen 4.0 Unported  
Licentie

# Probleem?



- Single-user vs multi-user
- Wanneer meerdere gebruikers dezelfde data willen lezen/schrijven
  - Conflicten
  - Concurrency control is nodig (gelijktijdigheid)
- Data uit veel tabellen moet geraadpleegd worden

# Transacties

- Def.: verzameling SQL-instructies die door één gebruiker ingevoerd wordt en waarvan de mutaties blijvend moeten zijn of ongedaan moeten worden
- Autocommit:
  - Elke SQL-instructie is een transactie
  - Elke transactie is permanent
- Commit: permanent maken van een transactie
- Rollback: ongedaan maken van een transactie
  - Eenmaal gecommited is er geen rollback mogelijk

# Transacties

- Transactie : vanaf begin tot een commit of rollback
- Laatste : steeds commit of rollback
- Wanneer zinvol ?
  - Als een bepaald gegeven uit meerdere tabellen geschrapt moet worden
  - Als gebruiker zich vergist heeft bij aanpassingen
- Mogelijke uitzonderingen (product beperkingen) : instructies die de catalogus wijzigen

# Hoe

- Impliciete start bv na ROLLBACK of COMMIT
  - Cf autocommit
- Expliciete start
  - `begin; sql code; commit ; -- dialect`
  - `begin; sql code; rollback; -- dialect`
  - `start transaction; sql code; commit ;`
  - `start transaction; sql code; rollback;`

# Savepoints

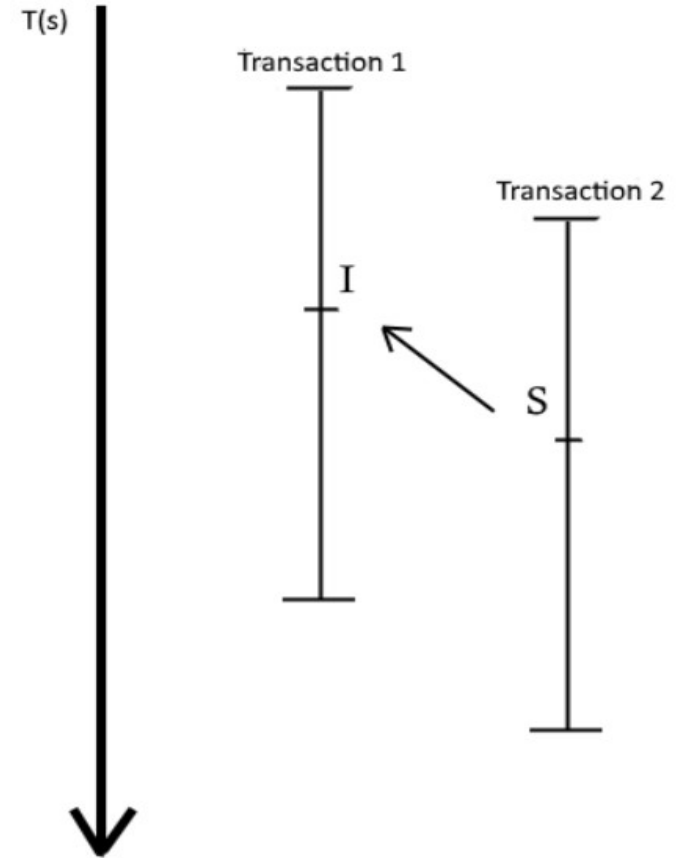
- Savepoints :
  - maken een deel van een actuele transactie ongedaan
  - Tussentijdse momentopnames in een transactie
- Vb.
  - Update ...
  - insert ...
  - savepoint S1
  - insert ...
  - savepoint S2
  - delete ...
  - rollback work to savepoint S2
  - ...

# Problemen multi-user gebruik

- Dirty read (uncommitted read) :
  - een gebruiker leest een gegeven dat nooit gecommit werd
- Nonrepeatable (of nonreproducible) read :
  - Een gebruiker leest voor en na de commit andere gegevens (gegevens worden gewijzigd)
- Phantom read :
  - een gebruiker leest voor en na de commit andere gegevens (er komen nieuwe gegevens)
- Lost update :
  - Een wijziging van één gebruiker wordt overschreven door een andere gebruiker
- Serialization Anomaly:
  - Het resultaat van het succesvol vastleggen van een groep transacties is inconsistent met alle mogelijke volgordes van het één voor één uitvoeren van die transacties.

# Dirty read

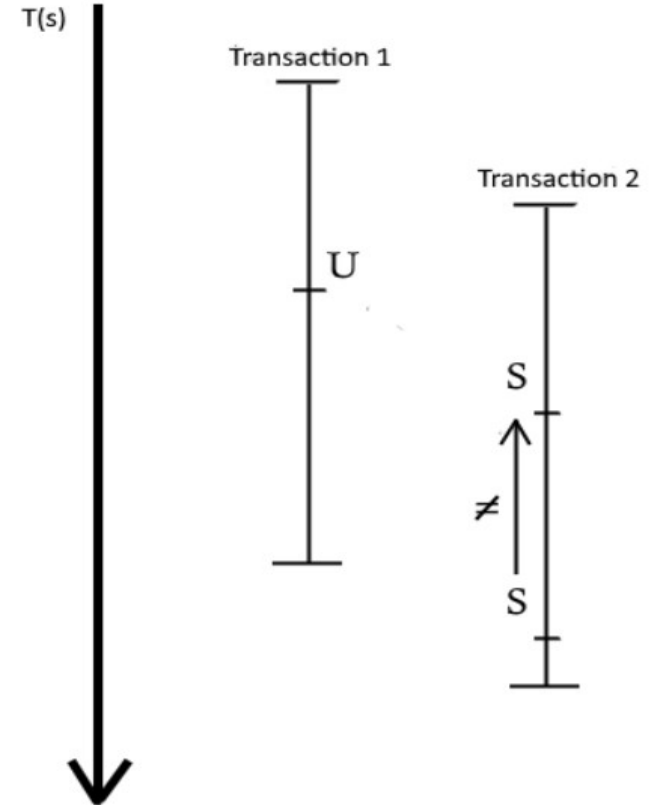
- Een transactie leest data gecreëerd door een gelijktijdige transactie die nog niet gecommit is





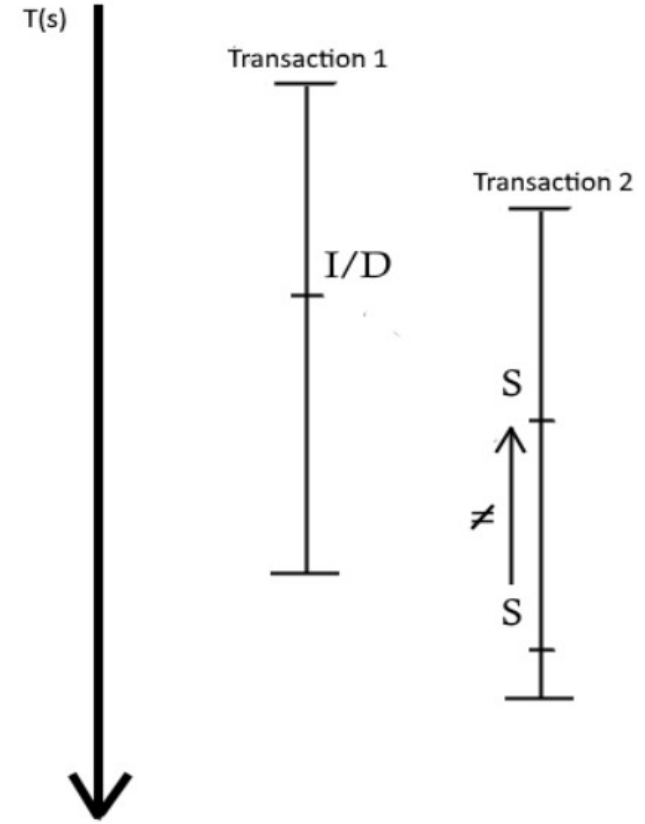
# Nonrepeatable read

- Een transactie leest data die eerder gelezen is en vindt dat de data aangepast is door een transactie die gecommit is sinds de eerste lees operatie.



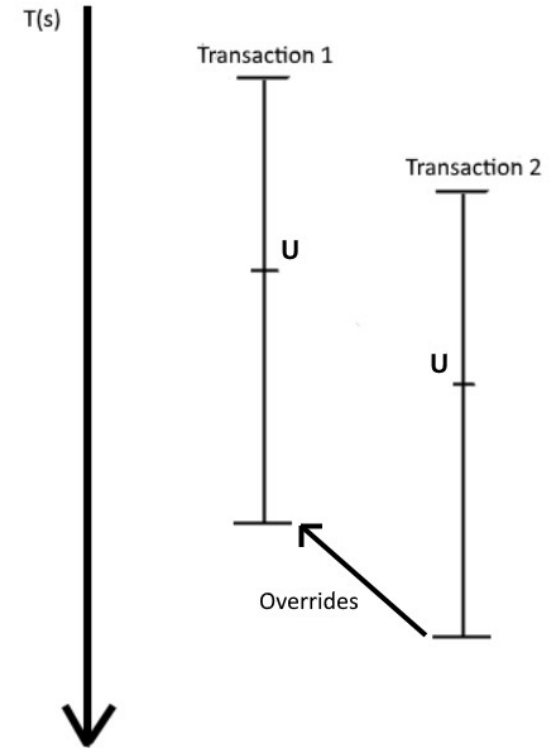
# Phantom read

- Een transactie herleest data die eerder gelezen is en vindt dat er data bijgekomen of verwijderd is door een andere transactie die gecommit is sinds de eerste lees operatie.



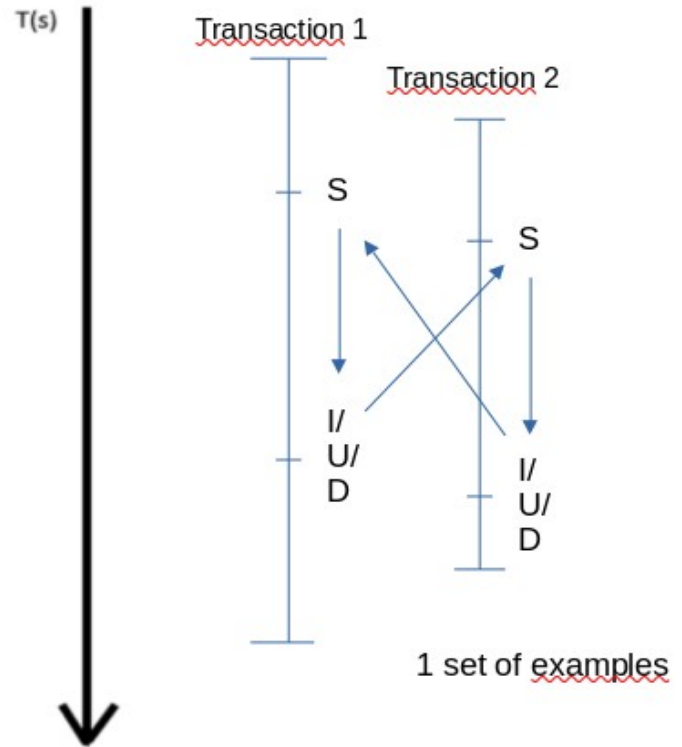
# Lost update

- Enkel de veranderingen van de laatste commit van gelijktijdige transacties die dezelfde rijen updaten zullen behouden worden.



# Serialization Anomaly

- Een andere (interne) volgorde van de overlappende transacties zorgt voor een ander resultaat.



# Oplossing

- Oplossing :
  - Transacties serieel verwerken!
- Oplossing indien honderden gebruikers tegelijk willen werken
  - Transacties parallel verwerken!

# LOCK TABLE ..

- Locking :
  - de rij waar één gebruiker mee werkt wordt gelocked voor de andere gebruikers
  - als transactie afgelopen is, wordt de blokkade opgeheven
- Locking gebeurt in de buffer (eg RAM)
- Verschillende opties voor granulariteit en rechten (bv SHARE vs EXCLUSIVE)

# Deadlocks

- Cf operatings systems
- Deadlock :
  - indien twee of meerdere gebruikers op elkaar wachten
- Oplossing :
  - indien deadlock aanwezig, dan wordt één transactie afgebroken

# Transacties: ISOLATION LEVEL

- Isolation level : mate van isolatie van gebruikers
- Niveaus:
  - Serializable : maximaal gescheiden
  - Repeatable read :
    - lezen : share blokkades (stopt bij einde transactie)
    - muteren : exclusive blokkades
  - Read committed (cursor stability):
    - lezen : share blokkades (stopt bij einde select)
    - muteren : exclusive blokkades
  - Read uncommitted (dirty read):
    - lezen : share blokkades (stopt bij einde select)
    - muteren : exclusive blokkades (stopt bij einde mutatie)



# Gevolgen

- Vermijd lang durende transacties
- Serializable :
  - concurrency is het laagst
  - Snelheid laagst
- Read Uncommitted:
  - concurrency is hoog, moeten weinig op elkaar wachten
  - Kunnen gegevens lezen die enkele momenten later niet meer bestaan

Vb.

- Set transaction isolation level serializable

# Usefull links for PostgreSQL

- <http://www.postgresql.org/docs/current/static/mvcc.html>
- <http://www.postgresql.org/docs/current/static/transaction-iso.html>
- <http://www.postgresql.org/docs/current/static/sql-start-transaction.html>