

Transaktionsparadigma

Definition: Transaktion
 „ununterbrechbare“ Folge von DML-/DDL-Befehlen

begin transaction --- end transaction

- *begin*: meist implizit mit ersten Datenbankzugriff
- *end*: *commit (work)* oder *abort (rollback)*

Überführung der Datenbank von einem
 logisch konsistenten Zustand in einen neuen
 logisch konsistenten Zustand

→Eigenschaften von TAs: ACID-Prinzip

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

ACID-Prinzip

Atomicity: "Alles-oder-Nichts"-Eigenschaft (Fehlerisolierung)

Consistency: Erhalt der DB-Konsistenz (definierte Integritätsbedingungen)

Isolation: Logischer Einbenutzerbetrieb (alle Aktionen innerhalb einer TA müssen vor parallel ablaufenden TAs verborgen werden)

Durability: Überleben aller Änderungen trotz beliebiger (erwarteter) Fehler garantiert (Persistenz)

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Transaktionsverwaltung

Anforderungen:

Synchronisation (concurrency control): mehrere TAs sollen gleichzeitig (nebenläufig) ablaufen

Logging

Recovery

Commit-Behandlung

Integritätssicherung

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (1)

Im Mehrbenutzerbetrieb kann es zu folgenden Anomalien kommen:

Lost Update

Dirty Read

Non-Repeatable Read

Phantom Reads

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (2)

Lost Update:

Schritt	T1	T2
1	read(A, a1)	
2	$a1 = a1 - 300$	
3		read(A, a2)
4		$a2 = a2 * 1,03$
5		write(A, a2)
6	write(A, a1)	
7	read(B, b1)	
8	$b1 = b1 + 300$	
9	write(B, b1)	

T1 transferiert 300 € von Konto A nach B.

T2 schreibt Konto A 3% Zinsen gut.

Interessante Schritte: 5 und 6.

Änderung von TA 2 unkontrolliert überschrieben und somit verloren.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (3)

Dirty Read

Schritt	T1	T2
1	read(A, a1)	
2	$a1 = a1 - 300$	
3	write(A, a1)	
4		read(A, a2)
5		$a2 = a2 * 1,03$
6		write(A, a2)
7	read(B, b1)	
8	...	
9	abort	

T1 transferiert 300 € von Konto A nach B.

T2 schreibt Konto A 3% Zinsen gut.

Interessante Schritte: 4 und 9.

T1 muss zurückgesetzt werden,
 T2 hat aber in Schritt 5/6 die Zinsen auf "falschem" Wert berechnet.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (4)

Non-Repeatable Read

Schritt	T1	T2
1	select gehalt from pers where pnr = 2	
2		update pers set gehalt = gehalt + 1000 where pnr = 2
3		update pers set gehalt = gehalt + 2000 where pnr = 3
4	select gehalt from pers where pnr = 2	

T1 liest verschiedene Gehälter.
T2 vergibt Gehaltserhöhungen.

T1 liest zweimal ein Gehalt mit zwei unterschiedlichen Ergebnissen.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (4)

Non-Repeatable Read (komplexerer Fall)

Schritt	T1	T2
1	select gehalt into :gehalt from pers where pnr = 2	
2	s = s + gehalt	
3		update pers set gehalt = gehalt + 1000 where pnr = 2
4		update pers set gehalt = gehalt + 2000 where pnr = 3
5	select gehalt into :gehalt from pers where pnr = 3	
6	s = s + gehalt	

T1 addiert verschiedene Gehälter.
T2 vergibt Gehaltserhöhungen.

Wenn man T1 ein weiteres Mal ausführt, bekommt man eine andere Summe s als beim ersten Mal.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Anomalien (5)

Phantom Read

Schritt	T1	T2
1		select sum(KontoStand) from Konten;
2	insert into Konten values (C, 1000)	
3		select sum(KontoStand) from Konten;

T2 fragt zweimal die Summe aller Kontostände ab.
T1 erzeugt eine neues Konto mit 1000 € Guthaben.

T2 berechnet innerhalb derselben TA zwei unterschiedliche Werte.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Synchronisation (1)

Korrektheitskriterium (Ziel):

logischer Einbenutzerbetrieb, d.h. Vermeidung **aller** Mehrbenutzeranomalien

Formales Korrektheitskriterium:

Serialisierbarkeit: Parallele Ausführung einer Menge von Transaktionen ist serialisierbar, wenn es **eine** serielle Ausführung derselben TA-Menge gibt, die den gleichen DB-Zustand und die gleichen Ausgabewerte wie die ursprüngliche Ausführung erzielt.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Synchronisation (2)

Aber: Serialisierbarkeit behindert parallele Ausführung von Transaktionen

→ Inkaufnahme von Anomalien ermöglicht weniger Behinderung von TAs
sehr mit **Vorsicht** zu verwenden!!

Wie Gewährleistung der Serialisierbarkeit?
.. durch *Sperrverfahren*.

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Sperrverfahren (1)

Beispiel: RX-Sperrverfahren (einfach)

zwei Sperrmodi:

Lese- oder Read (R)-Sperrungen
Schreib- oder exclusive (X)-Sperrungen

Kompatibilitätsmatrix:

	keine	R	X
R	+	+	-
X	+	-	-

"+" bedeutet: Sperre wird gewährt

"-" bedeutet: Sperrkonflikt

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Sperrverfahren (2)

bei Sperrkonflikt muss anfordernde TA warten bis unverträgliche Sperre(n) freigegeben werden
Blockierung und Deadlocks möglich
Sperrern u.U. bis zum TA-Ende gehalten
mögliche Optimierungen:
hierarchische Sperrverfahren
reduzierte Konsistenzebene
Mehrversionen-Ansatz

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Konsistenzebenen in SQL

vier Konsistenzebenen (isolation levels)
bestimmt durch die Anomalien, die in Kauf genommen werden
Lost Update wird immer vermieden: Schreibsperrern bis zum TA-Ende.
Default: Serializable

	Dirty Read	Non-Repeatable R.	Phantome
Read Uncommitted	+	+	+
Read Comitted	-	+	+
Repeatable Read	-	-	+
Serializable	-	-	-

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Konsistenzebenen in DB2 (1)

Sperrverfahren in DB2: Hierarchisches Sperrverfahren (komplexer, aber leistungsfähiger als RX)
Konsistenzebenen:
Uncommitted Read (UR): entspricht Read Uncommitted
Cursor Stability (CS): entspricht Read Comitted
Read Stability (RS): entspricht Repeatable Read
Repeatable Read (RR): entspricht Serializable

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Konsistenzebenen in DB2 (2)

Wahl der Konsistenzebene im CLI:
default: CS (!)
Änderung: ohne Datenbankverbindung
`change isolation to {RS | RR | UR | CS}`
Optimierer legt Sperrern fest

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Deadlock, Timeout

Unverträglichkeit eines Sperrwunsches: TA muss warten
Deadlock: in periodischen Abständen (einstellbar) wird nach Deadlocks gesucht (Zyklen-Erkennung), durch Zurücksetzen einzelner TA aufgelöst
Timeout: maximale Wartezeit (einstellbar) nach der eine TA abgebrochen wird

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Sperrern in DB2

Sperrgranulate: Table Spaces, Tables, Rows
Anwartschaftssperrern (Intentionssperrern)
Sperrern auf Tabellen (tables)
Sperrern auf Zeilen (rows)
...

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Hierarchische Sperrverfahren (1)

Kompatibilitätsmatrix reicht nicht aus
Sperrdisziplin erforderlich
zusätzliche Regeln für Anfordern und Freigeben von Sperren

Anfordern:

von der Wurzel zu den Blättern, d.h. von gröberen zu feineren Sperrgranulaten

...

Freigeben

von den Blättern zur Wurzel, d.h. von feineren zu gröberen Sperrgranulaten

bei TA-Ende sind alle Sperren freizugeben

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012

Hierarchische Sperrverfahren (2)

zusätzliche Regeln (cont'd)

Sperr-Konversion:

Sperren können in einen restriktiveren Modus konvertiert werden

Sperr-Eskalation:

Sperren können von kleinerem Granulat auf größeres Granulat ausgeweitet werden

Sperrdauer und Konsistenzebene:

Freigabe nach SQL-Anweisung (kurze Sperren)

Freigabe bei TA-Ende (lange Sperren)

Lange Sperren sind restriktiver

Datenbanksysteme für Hörer anderer Fachrichtungen WS 2011/2012

18.01.2012