Einsatz und Realisierung von Datenbanksystemen Zentralübung

Maximilian E. Schüle

Garching, 27. Juli 2017

Klausur

Hauptklausur
Dienstag, 08.08.2017, 16 Uhr bis 18 Uhr

A – Gunt	MI HS1
Gunz – Lutj	Interims 1
Lutz – Pfa	MW 2001 (Gallerie)
Pfi – Z	MW 2001 (unten)

Wiederholungsklausur
Mittwoch, 11.10.2017, 10:30 Uhr bis 12:30 Uhr
MW2001

Durchführung
90 Minuten
Keine Hilfsmittel erlaubt

Überblick Prüfungsstoff

Disclaimer

Werde heute hauptsächlich besprechen welche Themen besonders wichtig sind.

Falls etwas auf den Folien nicht erwähnt ist, aber in den Übungen / Vorlesung besprochen wurde, kann daraus nicht geschlossen werden, dass es nicht in der Prüfung drankommt. Auf den Folien dargestellte Übungen können Fehler enthalten.

Blatt 01 - Recovery

ACID

Force / Steal

Write Ahead Logging
1) Schreiben der Log-Einträge vor Commit
2) Vor Auslagerung einer Seite: Schreiben aller zugehörigen Log-Einträge

Wiederanlaufphasen
1. Analyse der Winner und Loser
2. Redo aller Transaktion
3. Undo der Losertransaktionen

Log-Einträge und Kompensationslogeinträge (CLR)

Physische/Logische Protokollierung

Blatt 01 – Logisch zu Physisch

A=900, C=1000, B=2000

Schritt	T_1	T_2	Log [LSN,TA,PageID,Redo,Undo,PrevLSN]
1.	**BOT**		$[\#1, T_1, \textbf{BOT}, 0]$
2.	$r(A, a_1)$		
3.		**BOT**	$[\#2, T_2, \textbf{BOT}, 0]$
4.		$r(C, c_2)$	
5.	$a_1 := a_1 - 50$		
6.	$w(A, a_1)$		$[\#3, T_1, P_A,$ A=850, A=900 $, \#1]$
7.		$c_2 := c_2 + 100$	
8.		$w(C, c_2)$	$[\#4, T_2, P_C,$ C=1100, C=1000 $, \#2]$
9.	$r(B, b_1)$		
10.	$b_1 := b_1 + 50$		
11.	$w(B, b_1)$		$[\#5, T_1, P_B,$ B=2050,B=2000 $, \#3]$
12.	**commit**		$[\#6, T_1, \textbf{commit}, \#5]$
13.		$r(A, a_2)$	
14.		$a_2 := a_2 - 100$	
15.		$w(A, a_2)$	$[\#7, T_2, P_A,$ A=750, A=850 $, \#4]$
16.		**commit**	$[\#8, T_2, \textbf{commit}, \#7]$

Blatt 01 – Recovery

Winner | Loser

Schritt	T_1	T_2	Log [LSN,TA,PageID,Redo,Undo,PrevLSN]
1.	**BOT**		$[\#1, T_1, \textbf{BOT}, 0]$
2.	$r(A, a_1)$		
3.		**BOT**	$[\#2, T_2, \textbf{BOT}, 0]$
4.		$r(C, c_2)$	
5.	$a_1 := a_1 - 50$		
6.	$w(A, a_1)$		$[\#3, T_1, P_A, A{-}{=}50, A{+}{=}50, \#1]$
7.		$c_2 := c_2 + 100$	
8.		$w(C, c_2)$	$[\#4, T_2, P_C, C{+}{=}100, C{-}{=}100, \#2]$
9.	$r(B, b_1)$		
10.	$b_1 := b_1 + 50$		
11.	$w(B, b_1)$		$[\#5, T_1, P_B, B{+}{=}50, B{-}{=}50, \#3]$
12.	**commit**		$[\#6, T_1, \textbf{commit}, \#5]$
13.		$r(A, a_2)$	
14.		$a_2 := a_2 - 100$	$<\#4', T_2, P_C, C{-}{=}100, \#4, \#2>$
15.		$w(A, a_2)$	$<\#2', T_2, -, -, \#4', 0>$
16.		**commit**	

<LSN, TA, Page-ID, Redo, PrevLSN, NextLSN>

Blatt 01 - Seitenersetzungsstrategien

Notwendige Wiederanlaufphasen:

	Force	No Force
Steal	Kein Redo Undo	Redo Undo
No Steal	Kein Redo Kein Undo	Redo Kein Undo

Steal: Undo-Phase notwendig
No Force: Redo-Phase notwendig

Kombination für Hauptspeicherdatenbanken:
No Force, No Steal: Redo-Phase, keine Undo-Phase notwendig

Blatt 02 - Transaktionsverwaltung

Welche Probleme?
Historienklassen und Zuordnung (RC, ACA, ST, SR, seriell)
http://home.in.tum.de/~becher/DBtransactions/

- SR: Serialisierbarkeitsgraph $SG(H)$ azyklisch
- RC: Wenn T_i liest T_j, dann $c_j <_H c_i$
- ACA: Wenn T_i liest T_j bezüglich Datum A, dann $c_j <_H r_i(A)$
- ST: Wenn $w_j(A) <_H o_i(A)$, dann $c_j <_H o_i(A)$ oder $a_j <_H o_i(A)$

Aufspannen des **Serialisierbarkeitsgraphen** und des **Wartegraphen**
Deadlocks (Gründe/Erkennung/Vermeidung)

2PL (2-Phase-Locking) / Timestamps / optimistische Synchronisation

Nicht wichtig:

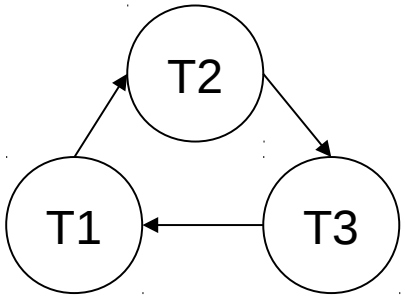
- Isolation Level
- Synchronisation von Bäumen

Blatt 02 - Transaktionsverwaltung

w_1(x), r_2(y), w_3(y), w_2(x), w_3(z), c_3, w_1(z), c_2 , c_1

Serialisierbarkeit
Aufspannen des Serialisierbarkeitsgraphen bezüglich der Konfliktoperationen (read vor write, write vor read, write vor write)

SG



nicht SR	(SG zyklisch)
RC	
ACA	
nicht ST	w_1(x) < w_2(x) < c_1

Blatt 02 - Transaktionsverwaltung

$$H = w_1(x), w_1(y), r_2(x), r_2(y)$$

a) Fügen Sie `commits` in H so ein, dass die Historie RC aber nicht ACA erfüllt:

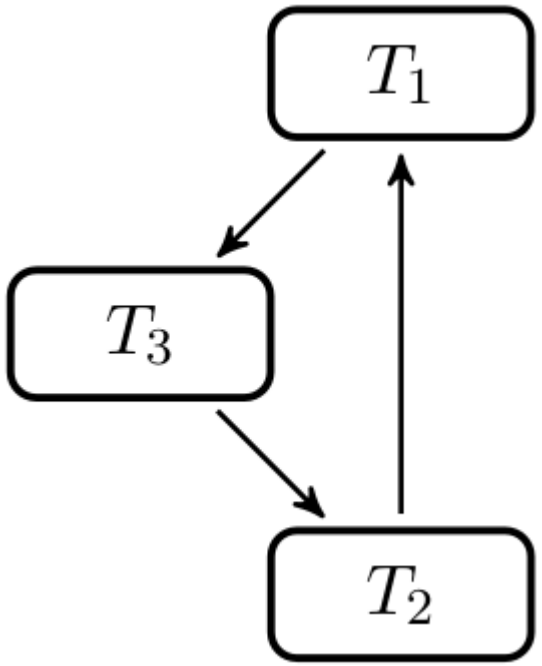
$$w_1(x), w_1(y), r_2(x), r_2(y), c_1, c_2$$

b) Fügen Sie `commits` in das ursprüngliche H so ein, dass die Historie ACA erfüllt.

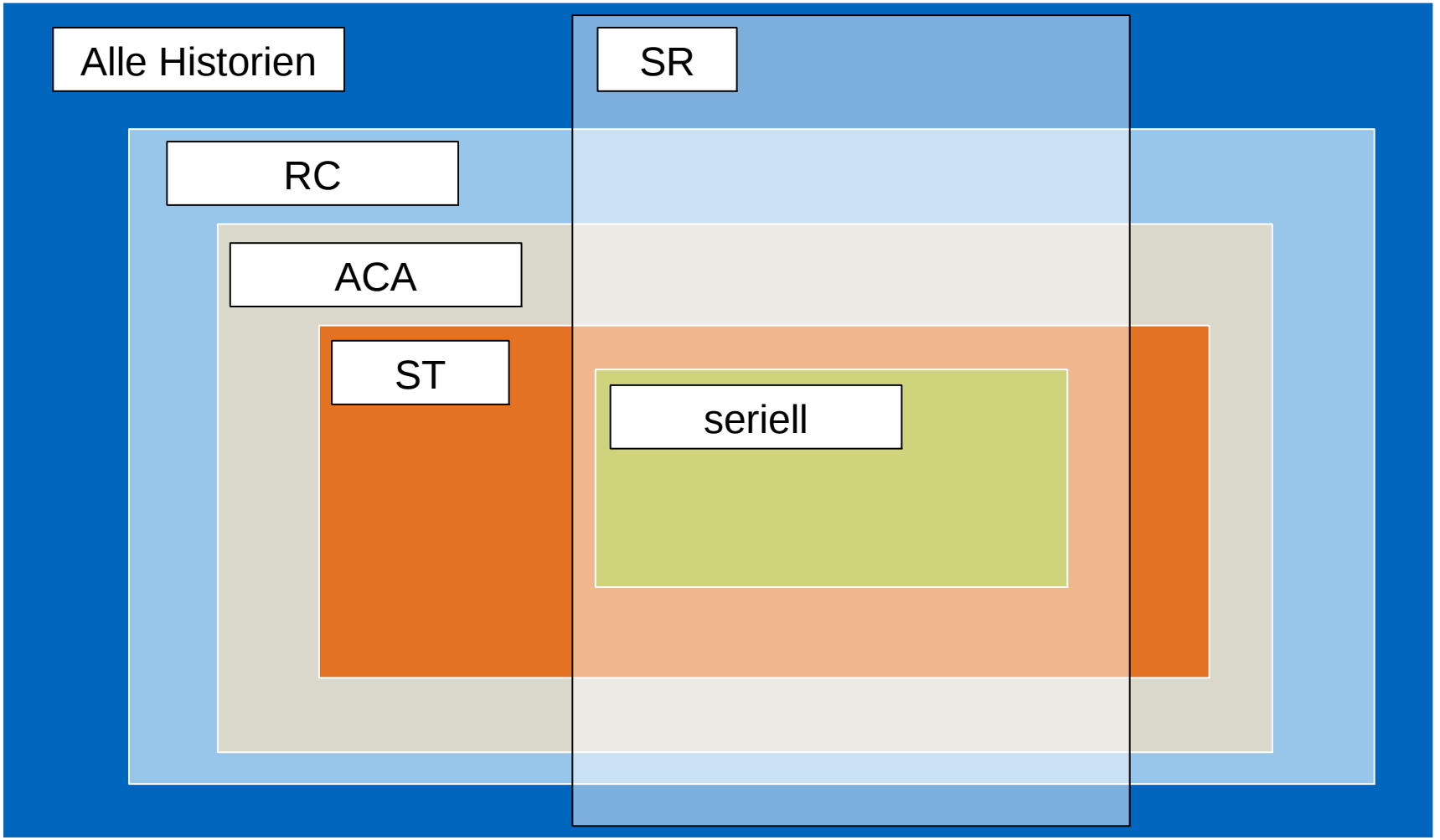
$$w_1(x), w_1(y), c_1, r_2(x), r_2(y), c_2$$

Blatt 02 - Transaktionsverwaltung

Schritt	T_1	T_2	T_3	Bemerkung
1.	BOT			
2.		BOT		
3.			BOT	
4.	lockX(A)			
5.		lockX(B)		
6.			lockX(C)	
7.	write(A)			
8.		write(B)		
9.			write(C)	
10.	lockS(C)			Will C lesen.
11.		lockS(A)		Will A lesen.
12.			lockS(B)	Will B lesen.



Blatt 02 - Transaktionsverwaltung



Blatt 03 - Sicherheitsaspekte

Angriffsarten

K-Anonymität

RSA (Welche Schlüssel?, Verschlüsseln, Entschlüsseln, Signieren)

SQL Injection
http://db.in.tum.de/~schuele/sql_verzeichnis.html

TLS-Protokoll

Keinen Handshake auswendig hinzeichnen können

Blatt 04 - Datalog

Datalog Theorie (Wann ist Programm sicher? Stratifizierbar?)

Datalog Programme! Wichtig!
Definition neuer Regeln
\=, not(...), +
Einfache Regeln zu SQL übersetzen und zurück
Domänenkalkül zu Datalog übersetzen
Rekursion
Keine Aggregationsfunktionen!

Üben!!!
http://datalog.db.in.tum.de/

Blatt 04 - Datalog

EDB: E(a,b). E(b,d). E(d,e). E(d,f). E(f,g).
IDB:
R(X) :- E(a,X).
R(X) :- R(Z), E(Z,X).

Sicher? Stratifzierbar?

$$\{[t] \mid \exists\, v,s,g([v,t,s,g] \in \text{Vorlesungen} \wedge \exists\, v2([v,v2] \in \text{voraussetzen} \wedge \exists\, s2,g2([v2,\text{'Wissenschaftstheorie'},s2,g2] \in \text{Vorlesungen})))\}$$

Es sind die Titel der direkten Voraussetzungen für die *Vorlesung* Wissenschaftstheorie.

```
vorWi(Titel) :- vorlesungen(V,Titel,_,_),voraussetzen(V,V2),
                vorlesungen(V2,'wissenschaftstheorie',_,_).
```

```
indirekt(A,C,S) :- direkt(A,C,_),S=0.
indirekt(A,C,S) :- indirekt(A,B,R),direkt(B,C,_),S=R+1.
indirekt(garching_forschungszentrum,C,S).
```

Blatt 05 – Verteilte Datenbanken

Fragmentierung
– Vertikal / Horizontal
– Korrektheit
– Rekonstruktion
– Join Auswertung

	vertikal	horizontal
fragmentieren	Projektion	Selektion
vereinigen	Join	Union

$$\begin{aligned}
\text{ProfVerw} &:= \Pi_{\text{PersNr, Name, Gehalt, Steuerklasse}}(\text{Professoren}) \\
\text{Profs} &:= \Pi_{\text{PersNr, Name, Rang, Raum, Fakultät}}(\text{Professoren})
\end{aligned}$$

$$\begin{aligned}
\text{TheolProfs} &:= \sigma_{\text{Fakultät} = \text{'Theologie'}}(\text{Profs}) \\
\text{PhysikProfs} &:= \sigma_{\text{Fakultät} = \text{'Physik'}}(\text{Profs}) \\
\text{PhiloProfs} &:= \sigma_{\text{Fakultät} = \text{'Philosophie'}}(\text{Profs})
\end{aligned}$$

Blatt 06 – Verteilte Datenbanken

Synchronisation: 2PC, Quorum-Consensus
Quorum-Consensus

Lesequorum $Q_r(A)$

Schreibquorum $Q_w(A)$

1. $Q_w(A) + Q_w(A) > W(A)$
2. $Q_r(A) + Q_w(A) > W(A)$

Chord
Finden von Schlüsseln
Fingertabellen ausfüllen

Bloom Filter

NoSQL-Datenbanken
CAP: Konsistenz, Verfügbarkeit, Partitionstoleranz

Blatt 7 – Betriebliche Anwendungen

Cube Operator

Bezug zu Group By

Fensterfunktionen (Windowfunctions)

alles

```
SELECT *, avg(gehalt) OVER (PARTITION BY rang ORDER BY gehalt DESC
RANGE BETWEEN 500 PRECEDING AND 500 FOLLOWING) FROM Professoren;
```

Decision Support

gute SQL-Kenntnisse, Stern-Schema, Anfragen darauf

Blatt 8 – Data Mining/Betriebliche Anwendungen

Assoziationsregeln mit Apriori
Frequentitemsets bestimmen, Konfidenz von Assoziationsregeln ableiten

Top-K Anfragen
Threshold/NRA
Skyline (in SQL mit und ohne „skyline“ operator)

Weitere Themen
Clustering (k-Means)
Entscheidungsbäume

Blatt 9/10 – Hauptspeicherdatenbanken

Row vs. Column Store
Bei welchen Anfragen was effizienter?
Implementierung

Speicherhierarchie
NUMA-Architektur
Non-Uniform Memory Access

Indexstruktur: ART
Finden von Schlüsseln, Einfügen von Schlüsseln

MVCC – Multi-Version Concurrency Control
Historie gegeben, entscheiden, ob diese unter MVCC möglich ist

Blatt 11 - XML

Grundkenntnisse XML Schema

XPath
Achsen

Xquery
FLOWR
Grundlegende Aggregatsfunktionen wie count()

Blatt 12 – Big Data

TF-IDF

Xquery
FLOWR
Grundlegende Aggregatsfunktionen wie count()

PageRank
Für gegebenen Graph berechnen können

HITS
Für gegebenen Graph berechnen können

http://db.in.tum.de/teaching/ss17/impldb/

Maximilian E. Schüle
schuele@in.tum.de
02.11.060

Viel Erfolg!