



**Übungen zur Vorlesung
"Datenbanksysteme II"**
Wintersemester 2008/2009
Manuel Mayr (manuel.mayr@uni-tuebingen.de)

13. Übungsblatt

Ausgabe: 29. Januar 2009 · Besprechung: 5. Februar 2009

Aufgabe 1: 2 Phase Locking

(5 Punkte)

Beachten Sie den folgenden *Schedule*:

\mathcal{T}_1	\mathcal{T}_2	\mathcal{T}_3
		$r(X)$
$r(X)$	$r(y)$	
$w(Y)$	$r(X)$	
		$w(x)$
c	c	c

Ist es möglich, dass der *Schedule* von *Scheduler* erzeugt wurde, der das 2PL-Protokoll befolgt? Wenn ja, zeigen Sie dies, indem Sie *lock/unlock* Operationen einfügen, die den 2PL Regeln entsprechen. Wenn nein, erklären Sie warum.

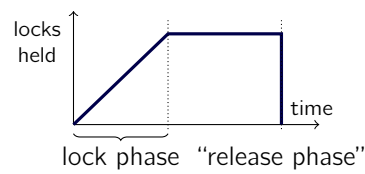
Aufgabe 2: Strict 2 Phase Locking

(10 Punkte)

In der Vorlesung wurden sowohl, das 2PL-, als auch das strikte 2PL-Protokoll behandelt, wobei das letztere von vielen Datenbankimplementationen benutzt wird. Im Folgenden wird noch einmal kurz auf die Funktionsweise des strikten 2PL-Protokolls eingegangen:

- Bei einem lesenden Zugriff wird der Transaktion ein *read lock* auf das Datenbank-Objekt gewährt.
- Bei einem schreibenden Zugriff wird der Transaktion, wird ein *write lock* auf das Datenbank-Objekt gewährt. Wenn bereits ein *read lock* auf dem Objekt existiert, dann wird nach Möglichkeit der *read lock* zu einem *write lock* aufgewertet.
- Wenn kein *lock* gewährt werden kann, dann wird die Transaktion so lange unterbrochen, bis er gewährt werden kann.

- Am Ende der Transaktion sendet der Klient ein COMMIT zum Server, welcher sogleich alle *locks* freigibt.



Strict 2PL

Folgende *Schedules* werden auf der Datenbank nacheinander ausgeführt.

1. $S_1 = \langle r_3(Z), r_2(X), r_2(Y), w_2(X), c_2, w_3(Z), r_1(X), w_1(Y), c_1, r_3(X), c_3 \rangle$
2. $S_2 = \langle r_1(X), r_3(W), r_2(Y), w_2(X), w_3(Y), w_1(W), r_3(Z), c_3, w_1(Z), c_1, r_2(W), c_2 \rangle$
3. $S_3 = \langle r_1(Y), r_3(X), w_3(X), r_2(X), w_1(Y), c_1, r_3(Y), w_2(Y), c_2, r_3(Z), c_3 \rangle$

Auf Basis der angegebenen *Schedules* führen Sie die folgenden Aufgaben aus:

- Schreiben Sie die *Schedules* auf, welche durch das **strict 2PL** aus den angegebenen Original-*Schedules* resultieren. Benutzen Sie dafür folgende Notation:
 - $rl_i(X)$ steht für einen *read lock* auf dem Objekt X in Transaktion i .
 - $wl_i(X)$ steht für einen *write lock* auf dem Objekt X in Transaktion i .
 - $ul_i(X)$ steht für die Freigabe aller *locks* auf dem Objekt X , die von der Transaktion i gehalten werden.
- Der zweite Schedule führt zu einem typischen Problem in nicht konservativen 2PL-Protokollen. Was verursacht das Problem und welche Techniken gibt es, solche Problem zu lösen?