Übungsblatt 6 – Concurrency Control

Aufgabe 1

Gegeben sind die Transaktionen T1 und T2, sowie die Objekte A und B. Eine Leseoperation wird mit r bezeichnet, eine Schreiboperation mit w. Gegeben sind die folgenden Schedules:

S1: ($r_1(B)$, $r_1(A)$, $r_2(A)$, $w_2(A)$, $w_1(B)$, $w_1(A)$)
S2: ($r_2(A)$, $r_2(B)$, $r_1(B)$, $r_1(A)$, $w_1(A)$, $w_1(B)$)
S3: ($r_1(B)$, $r_1(A)$, $r_2(A)$, $r_2(B)$, $w_1(A)$, $w_1(B)$)
S4: ($r_2(A)$, $r_1(B)$, $w_1(B)$, $w_1(A)$, $r_2(B)$)

a) Geben Sie für jeden dieser Schedules den Serialisierbarkeitsgraphen an.

b) Welche dieser Schedules sind seriell oder serialisierbar?

c) Geben Sie bei allen serialisierbaren Schedules den äquivalenten seriellen Schedule an.

d) In welchen Schedules tritt ein Lost-Update-Problem auf? Geben Sie gegebenenfalls an, welcher Änderungsbefehl auf welchem Objekt verloren geht.

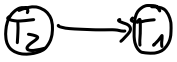
e) Zeigen Sie bei allen nicht-seriellen Schedules, wie ein sperrbasiertes Verfahren die einzelnen Befehle synchronisieren würde. Geben Sie an, falls ein Deadlock auftreten würde.

a) S1: 

→ Zyklus ↯

S2: 

→ serialisierbar

S3: 

→ serialisierbar

S4: 

→ Zyklus ↯

b) S1: nicht serialisierbar
S2: seriell (T2, T1)
S3: serialisierbar
S4: nicht serialisierbar

c) $S_2 \triangleq (T_2, T_1)$
$S_3 \triangleq (T_2, T_1)$

Aufgabe 2

Das Problem der inkonsistenten Analyse soll in Oracle rekonstruiert werden. Dazu können Sie den folgenden, vereinfachten SQL-Code übernehmen. Für die zweite Transaktion (Überweisung) benötigen Sie einen zweiten Account, hier dbsysXY.

```
CREATE TABLE girokonto (
  name VARCHAR2(20) primary key,
  kontostand INT,
  land VARCHAR2(20)
  );

GRANT INSERT, SELECT, UPDATE ON girokonto TO dbsysXY;

INSERT INTO girokonto VALUES ('A', 1000, 'D');
INSERT INTO girokonto VALUES ('B', 1000, 'D');
INSERT INTO girokonto VALUES ('C', 1000, 'D');
INSERT INTO girokonto VALUES ('D', 1000, 'D');
INSERT INTO girokonto VALUES ('E', 1000, 'D');
INSERT INTO girokonto VALUES ('F', 1000, 'CH');
INSERT INTO girokonto VALUES ('G', 1000, 'CH');
INSERT INTO girokonto VALUES ('H', 1000, 'CH');
INSERT INTO girokonto VALUES ('I', 1000, 'CH');
INSERT INTO girokonto VALUES ('J', 1000, 'CH');
COMMIT;
```

Es soll nun die Summe aller Kontostände berechnet werden. Um die Gleichzeitigkeit mit einer anderen Transaktion zu erzwingen werden zuerst die Kontostände aller deutschen Girokonten und danach aller schweizer Konten addiert.

```
SELECT SUM(kontostand) FROM girokonto
WHERE land = 'D';

SELECT SUM(kontostand) FROM girokonto
WHERE land = 'CH';
COMMIT;
```

Zwischen der Berechnung der beiden Teilsummen kann man nun eine Überweisung von 500 Euro durch eine andere Transaktion durchführen.

```
UPDATE eck.girokonto SET kontostand = kontostand - 500 WHERE
   name = 'A';
UPDATE eck.girokonto SET kontostand = kontostand + 500 WHERE
   name = 'F';
COMMIT;
```

Aufgaben:

a) Testen Sie, ob das Problem der inkonsistenten Analyse eintritt.

b) Ändern Sie das Isolation Level so, dass die Summenberechnung korrekt erfolgt.

c) Vergleichen Sie das Verhalten von Oracle mit der in der Vorlesung vorgestellten klassischen Sperrverwaltung.