AG Datenbanken und Informationssysteme

Wintersemester 2006 / 2007

Prof. Dr.-Ing. Dr. h. c. Theo Härder
Fachbereich Informatik
Technische Universität Kaiserslautern

http://wwwdvs.informatik.uni-kl.de

7. Übungsblatt

Für die Übung am Donnerstag, **14. Dezember 2006**,
von 15:30 bis 17:00 Uhr in 13/222.

Aufgabe 1: JDBC und Rekursion

Gegeben sei eine DB mit folgenden Relationen:

```
TEIL (TNR, TNAME, GEWICHT)
TEILSTRUKTUR (OTNR,UTNR, MENGE)
```

In der `TEIL`-Relation befinden sich Teileinformationen wie Teilennummer, -name und -gewicht. Ein Teil kann zum einen aus mehreren Unterteilen bestehen, die jeweils wiederum Unterteile haben können. Zum anderen kann ein Teil ein Unterteil von mehreren Oberteilen sein. Weiterhin sei angenommen, dass ein Teil sich selbst sowohl direkt als auch indirekt nicht enthalten kann. Nur für Teile, die keine Unterteile besitzen, ist ein Gewicht ungleich 0 eingetragen. Die `TEILSTRUKTUR`-Relation beschreibt die Beziehung zwischen einem Oberteil und Unterteil mit der zugehörigen Mengenanzahl des Unterteils, die in den Oberteil eingeht. Basierend auf den beiden gegebenen Relationen schreiben Sie unter der Anwendung von JDBC ein rekursives Java-Programm, das den für den DB-Zugriff benötigten Benutzernamen und -passwort sowie eine Teilennummer als Eingabeparameter erhält und das Gesamtgewicht des eingegebenen Teils berechnet.

Aufgabe 2: JDBC und Transaktionen

Eine Flug-DB enthalte eine Relation `SITZPLATZ`, in der Sitzplatzreservierungen von Flügen gespeichert werden und deren Schema durch folgende SQL-Anweisung erzeugt wurde:

```
CREATE TABLE SITZPLATZ(
    FLUGNR       VARCHAR(6),
    DATUM        DATE,
    REIHENNR     SMALLINT,
    PLATZNR      VARCHAR(1),
    KUNDENNAME   VARCHAR(20),
    PRIMARY KEY (FLUGNR, DATUM, REIHENNR, PLATZNR));
```

Es wird angenommen, dass Tupel, die Flugsitzplätze repräsentieren, in der Relation `SITZPLATZ` bereits vorliegen. Hat ein Kunde einen Sitzplatz für einem Flug an einem bestimmten Tag reserviert, so steht sein Name im Attribut `KUNDENNAME` des entsprechenden Tupels. Freie Sitzplätze erkennt man an Tupeln, für die ein Nullwert im Attribut `KUNDENNAME` eingetragen ist.

Ein Reservierungsvorgang für einen Kunden besteht aus den folgenden drei Phasen:

(1) Anzeigen von freien Sitzplätzen bzgl. eines Fluges und Datums

(2) Auswahl eines Sitzplatzes

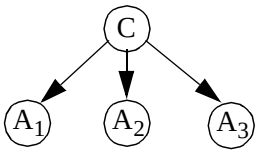
(3) Belegen des gewählten Sitzplatzes und bestätigen

Schreiben Sie mit Hilfe von JDBC ein Java-Programm, das den oben beschriebenen Reservierungsvorgang im Mehrbenutzerbetrieb realisiert, und setzen Sie dabei das Transaktionskonzept ein. Es soll gewährleistet sein, dass kein Sitzplatz zweimal vergeben wird. Wenn ein Kunde eine Bestätigung für einen Sitzplatz bekommt, dann erhält er diesen garantiert.

Achten Sie auch darauf, dass lange Wartezeiten bei der Durchführung eines Reservierungsvorgangs möglichst vermieden werden. Neben dem Benutzernamen und -passwort soll Ihr Programm die Flugnummer, das Flugdatum sowie den Kundennamen als Eingabeparameter erhalten. Die Sitzplatzwahl soll nach dem Anzeigen der freien Plätze eingelesen werden. Wenn der Platz gebucht ist, erhält der Kunde die Bestätigung.

Aufgabe 3: Verteiltes 2-PC-Protokoll

Betrachten Sie das vollständige Zweiphasen-Commit-Protokoll in der folgenden Umgebung mit einem Koordinator und drei Agenten, die vom Koordinator jeweils in der Reihenfolge A_1, A_2 und A_3 kontaktiert werden:



Welche und insgesamt wieviele Nachrichten werden gesendet und welche Log-Daten werden von den Komponenten geschrieben, wenn der Koordinator den Auftrag für ein Commit erhält und

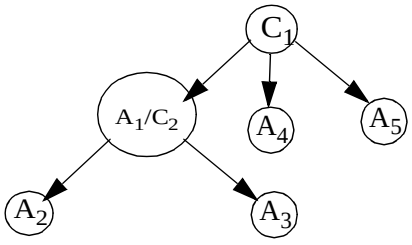
a) alle drei Agenten ohne Zwischenfall am Commit teilnehmen?

b) der Agent A3 auf das empfangene PREPARE aufgrund eines internen Fehlers mit FAILED antwortet?

c) der Agent A2 auf das empfangene PREPARE ein READY sendet, aber sofort danach abstürzt? Was passiert beim Wiederanlauf des Agenten?

Aufgabe 4: Optimierungen für das 2-Phasen-Commit-Protokoll

In dieser Aufgabe betrachten wir verschiedene Optimierungen für das 2-PC-Protokoll. Gegeben sei folgende Aufrufstruktur zwischen Koordinatoren und Agenten, die in einer hierarchischen Struktur angeordnet sind. Die Agenten werden von C_1 in der Reihenfolge A_1/C_2,A_4, A_5 und von C_2 in der Reihenfolge A_2, A_3 angesprochen.:



Wie viele Nachrichten und Log-Ausgaben werden im Erfolgsfall versendet bzw. geschrieben, wenn:

a) das vollständige hierarchische 2-PC-Protokoll verwendet wird?
b) die spezielle Optimierung für Leser verwendet wird?
c) A_i nach jedem Aufruf in den Prepared-Zustand wechselt?
d) A_i erst beim letzten Aufruf in den Prepared Zustand wechselt?
e) die ACK-Nachricht expizit weggelassen wird?
f) das "spartanische Protokoll" zum Einsatz kommt?

Auf was ist bei den entsprechenden Realisierungsformen zu achten?

Aufgabe 5: Schachtelung von Transaktionen

Können Transaktionen, die bekanntlich ACID-Eigenschaften besitzen, geschachtelt werden? Begründen Sie Ihre Anwort.