Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Prof. H.-J. Schek *A. Feßler, U. Röhm, H. Schuldt, R. Weber*

Informationssysteme IS-G SS 1997

Übung 5 Abgabe: 13.5.1997

Gegeben sei das aus früheren Übungen bekannte Schema einer Universitätsdatenbank:

Professor (P_Name, Abt_Nr, Gebäude, Raum, Tel)
Abteilung(Abt_Nr, A_Name, Vorsteher)
Gebäude (Gebäude, Hausmeister, Adresse)
Student (ETH_Nr, S_Name, Semester, Abt_Nr)
Prüfung (ETH_Nr, Fach, Prüfer, Note)

Aufgabe 1: Allgemeine Integritätsbedingungen

Spezifizieren Sie die folgenden (nicht notwendigerweise mit der realen Welt übereinstimmenden) Integritätsbedingungen mit Hilfe von Assertion– oder Trigger-Deklarationen:

a) Studenten dürfen die Prüfung im Fach Informationssysteme erst ab dem 5. Semester ablegen.

b) Studenten, die bereits im 15. Semester sind, müssen mindestens eine Prüfung abgelegt und bestanden haben.

c) Professoren sollen sich ihr Büro nicht mit Kollegen teilen müssen. Allerdings kann der Fall auftreten, dass zwei Professoren ihre Räume miteinander tauschen.

d) Die ETH-Nr eines Studenten darf nie verändert werden.

e) Die Note eines Studenten darf nur erhöht, nicht aber erniedrigt werden.

f) Studenten der Abteilung IIIc dürfen nur an die Abteilung IIIb wechseln, wenn sie in Elektrotechnik mindestens die Note 4.5 erreicht haben.

Aufgabe 2: Deferrability

a) In einer Datenbank mit 100 Tabellen sind dreitausend Assertions (alle mit der Deferrability `not deferrable`) spezifiziert, die laut Definition jederzeit gültig sein müssen. Wie kann das Datenbanksystem vermeiden, dass nach jeder Operation *alle* diese Assertions auf ihre Gültigkeit überprüft werden müssen? Welche Informationen sind hierzu nötig ?

b) Wann ist es sinnvoll, dass Assertions bzw. Constraints die Deferrability `deferrable initially deferred` besitzen? Geben Sie hierzu ein Beispiel an.

c) Gegeben sei die aus der Vorlesung bekannte Beispieldatenbank:

Kunden	(KNr, Name, Stadt, Saldo, Rabatt)
Produkte	(PNr, Bez, Gewicht, Preis, Lagerort, Vorrat)
Bestellungen	(BestNr, Monat, Tag, KNr, PNr, Menge, Summe, Status)

sowie die Assertion `Max_Wert`:

```
CREATE ASSERTION Max_Wert CHECK (
  NOT EXISTS (
     SELECT *
     FROM   Produkte
     WHERE  Preis*Vorrat > 500000))
DEFERRABLE INITIALLY IMMEDIATE
```

Gegeben ist der folgende Ausschnitt einer Transaktion, in der u.a. ein neues Produkt sowie eine Bestellung dieses Produktes eingegeben wird, und die Bearbeitung der Lieferung erfolgt.

```
BEGIN OF TRANSACTION
.....
.....
.....
INSERT INTO Produkte VALUES (8, 'Server', 7.500, 80000, 'Zug', 10);
.....
INSERT INTO Bestellungen VALUES (17, 4, 29, 5, 8, 5, 400000, 'neu');
.....
UPDATE Produkte SET Vorrat = Vorrat-5 WHERE PNr=8;
.....
UPDATE Bestellungen SET Status = 'geliefert' WHERE BestNr=17;
.....
.....
.....
END OF TRANSACTION
```

Die Assertion `Max_Wert` soll so früh wie möglich innerhalb der Transaktion überprüft werden. Wie muss man obige Transaktion hierzu erweitern ?
Welche Nachteile kann ein Test der Assertion am Ende der Transaktion mit sich führen?

Aufgabe 3: Fremdschlüsselbedingungen

a) Für die Relation *Student* ist die folgende Fremdschlüsselbedingung definiert:

```
CREATE TABLE Student (
...
FOREIGN KEY Abt_Nr REFERENCES Abteilung(Abt_Nr),
...
)
```

„Simulieren“ Sie diese Fremdschlüsselbedingung

i. als „gewöhnlichen“ Table Constraint in der Relation *Student*
ii. als allgemeine Assertion (Zusatzfrage: Wie sähe diese Assertion im Relationenkalkül aus?)
iii. als Trigger

b) Für die Relation Prüfung seien die folgenden Fremdschlüsselbedingungen spezifiziert:

```
CREATE TABLE Prüfung (
...
FOREIGN KEY ETH_Nr REFERENCES Student (ETH_Nr)
   ON DELETE CASCADE
FOREIGN KEY Prüfer REFERENCES Professor (P_Name)
   ON DELETE SET NULL )
```

„Simulieren“ Sie mit Hilfe geeigneter Trigger-Spezifikationen den Effekt dieser Fremdschlüsselbedingungen.