
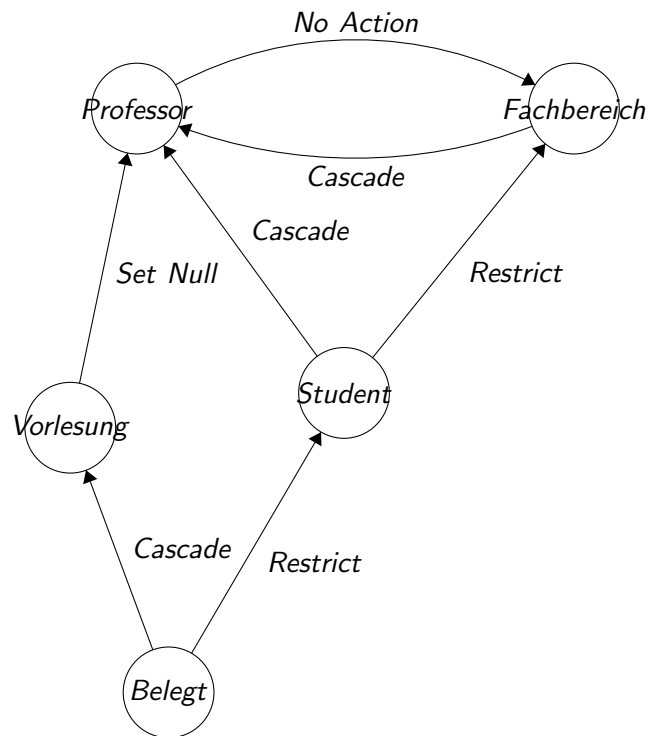


| | | | | |
|---|-------------------|-----------------------------|--------|---------------|
|  | Lehrveranstaltung | Grundlagen von Datenbanken | | WS 2014/15 |
| | Aufgabenzettel | 4 | | |
| | STiNE-Gruppe 19 | Meimerstorf, Jochens, Töter | | |
| | Ausgabe | Mi. 15.11.2017 | Abgabe | Fr. 12.1.2018 |

1 Referentielle Aktion

1.1




1.2

Falls man einen Professor löscht, und dann als erstes, der Student gelöscht wird, läuft man je nach Reihenfolge entweder in das Restrict beim Belegt oder wenn man als erstes den Fachbereich löscht, in das Restrict beim Studenten, bzw. wenn der Student vorher schon gelöscht wurde, würde hierbei nicht einmal das Restrict. Abschließend also kein sicheres Schema.

1.3

```
ALTER TABLE 'Student'
DROP FOREIGN KEY 'fkFachbereich';
```

```
ALTER TABLE 'table1'
ADD CONSTRAINT 'fkFachbereich' FOREIGN KEY ('Fachbereich') REFERENCES
'Fachbereich' ON DELETE CASCADE;
```

| | | | | |
|---|-------------------|--|--------|----------------------|
|  | Lehrveranstaltung | Grundlagen von Datenbanken WS 2014/15 | | |
| | Aufgabenzettel | 4 | | |
| | STiNE-Gruppe 19 | Meimerstorf, Jochens, Töter | | |
| | Ausgabe | Mi. 15.11.2017 | Abgabe | Fr. 12.1.2018 |

1.4

Ja

2 Änderbarkeit von Sichten

2.1

```
Create View GesamtwerkBöll
AS SELECT b.ISBN,b.TITEL
FROM Buch b , Schriftsteller autor
WHERE b.Autor = Autor.Snr
AND autor.Nachname = 'Böll'
AND autor.Vorname = 'Heinrich'
```

Operationen nicht zulässig, zwei Tabellen.

```
Create View AlteBritischeAutoren
AS SELECT Snr,Vorname,Nachname
FROM Schriftsteller
WHERE Nationalität = 'britisch'
AND Geburtsjahr < 1960
```


Operationen zulässig SNR ist dabei.

```
Create View BücherHeyne
AS SELECT Titel
FROM Buch
WHERE VERLAG = "Heyne"
```

Nicht zulässig, schlüssel fehlt.

2.2

- i) zulässig
Nur in SciFiKlassiker Fischer können neue auftreten.
- ii) zulässig In SciFiKlassiker Fischer und in SciFiKlassiker können neue auftreten.

| | | | | | |
|---|-------------------|------------------------------------|--------|----------------------|------------|
|  | Lehrveranstaltung | Grundlagen von Datenbanken | | | WS 2014/15 |
| | Aufgabenzettel | 4 | | | |
| | STiNE-Gruppe 19 | Meimerstorf, Jochens, Töter | | | |
| | Ausgabe | Mi. 15.11.2017 | Abgabe | Fr. 12.1.2018 | |

- iii)
Fehler durch Cascaded Vererbte Check Option bei Klassiker.
- iv)
Kein Fehler, da einfügen direkt in Klassiker den Check nicht auslöst.
Werden in allen sichtbar.

3 Serialisierbarkeit, Anomalien

3.1

- S_1 Endwert: A:30 B:130
- S_2 Endwert: A:10 B:120
- S_3 Endwert: A:10 B:200
- S_4 Endwert: A:10 B:120
- S_5 Endwert: A:10 B:20
- S_6 Endwert: A:10 B:110

3.2


Abhängigkeiten hier allgemein gehalten, welche entstehen können und nicht für jede einzelne Schedule. Da zu faul. (Genaue Abhängigkeiten bei c).

Bei den Transaktionen besteht die Abhängigkeit dabei, dass sie beide auf den Werten a/b agieren. Das heißt wenn die eine Transaktion einen Wert eingelesen hat, bevor die andere Transaktion ihren aktuellen Wert zurückgeschrieben hat, kommt es zu einer Anomalie, da die erste Transaktionen einen dreckigen Wert liest und mit dem weiterrechnet.

Also muss die w-Operation einer Transaktion immer vor der nächsten Lese Operation auf den Wert ausgeführt werden.

3.3

- S_1 Seriell (hintereinander weggearbeitet)
- S_2 Seriell (hintereinander weggearbeitet)
- S_3 nicht serialisierbar (r_1 liest S_1 bevor r_2 schreibt bzw. r_2 liest bevor w_1 schreibt)
- S_4 serialisierbar (die reads der anderen Transaktion treten immer erst auf nachdem die andere geschrieben hat)

| | | | | | |
|---|-------------------|------------------------------------|--------|----------------------|------------|
|  | Lehrveranstaltung | Grundlagen von Datenbanken | | | WS 2014/15 |
| | Aufgabenzettel | 4 | | | |
| | STiNE-Gruppe 19 | Meimerstorf, Jochens, Töter | | | |
| | Ausgabe | Mi. 15.11.2017 | Abgabe | Fr. 12.1.2018 | |

- S_5 nicht serialisierbar (r_1 lieSSt bevor r_2 schreibt bzw. r_2 liest bevor w_1 schreibt)
- S_6 nicht serialisierbar (r_1 lieSSt bevor r_2 schreibt bzw. r_2 liest bevor w_1 schreibt und außerdem liest r_1 auch A bevor w_2 schreibt.)

4 2PL-Synchronisation mit R/X-Sperre

| | T_1 | T_2 | T_3 | Bemerkung |
|----|-----------|-----------|-----------|-----------------|
| 0 | | | | |
| 1 | | | lock(x,X) | |
| 2 | | | write(x) | |
| 3 | | | lock(y,R) | |
| 4 | lock(x,R) | | read(y) | |
| 5 | read(x) | lock(y,R) | | |
| 6 | | read(y) | lock(y,X) | T3 wartet T2 |
| 7 | lock(x,X) | | | |
| 8 | write(x) | lock(z,R) | | |
| 9 | unlock(x) | read(z) | | |
| 10 | commit | unlock(y) | | Nachricht an T3 |
| 11 | | unlock(z) | write(y) | |
| 12 | | commit | lock(x,R) | |
| 13 | | | read(x) | |
| 14 | | | unlock(y) | |
| 15 | | | unlock(x) | |
| 16 | | | commit | |
| 17 | | | | |

Bei dem Blatt bin ich mir sehr unsicher... Vermutlich zurecht xD