Schematische Ausnahmebehandlung in relationalen Datenbanken

Daniel Schaller (daniel.schaller@mail.inf.tu-dresden.de)
Technische Universität Dresden, Fakultät Informatik
Arbeitsgruppe Datenbanken

1 Einleitung

Dieses Papier diskutiert das Problem der *Schematischen Ausnahmebehandlung in relationalen Datenbanken*, einer alternativen Form von Ausnahmen zu den bereits bekannten Ausnahmen auf Instanzebene (z.B. Verletzung von Check-Constraints). Diese schematische Form von Ausnahmen wird in diesem Beitrag anhand eines konkreten Beispiels, dem Ausschnitt aus einer Prüfungsordnung (Abbildung 1), vorgestellt. Die zugrundeliegende Studienarbeit behandelt dieses Thema ausführlich.

Im vorliegenden Szenario (Abbildung 1) ist die Zulassungsbedingung ((M1 $\vee$ M2) $\wedge$ (M3 $\vee$ M4) $\rightarrow$ Zulassung) für die Mathematik Abschlussprüfung dargestellt. Ein Student erfüllt dieses Kriterium, wenn er die Scheinklausuren *Mathematik 1* oder *Mathematik 2* und *Mathematik 3* oder *Mathematik 4* erfolgreich absolviert hat. In der graphischen Notation wird dies durch die gewählte Knotenbildung und die Anzahl der benötigten, erfüllten Unterknoten ausgedrückt. Damit beispielsweise die Bedingung (M1 $\vee$ M2) dargestellt werden kann, wird ein Zwischenknoten *Mathematik, 1. Studienjahr* eingeführt, welcher mindestens einen erfüllten Unterknoten benötigt, damit er selbst als erfüllt gilt.

Interessant ist, dass es im zeitlichen Verlauf zu Veränderungen der Prüfungsordnung kommen kann: entweder allgemeiner Art, d.h. die Änderung ist für alle Studenten von Bedeutung oder aber auch nur für eine Teilmenge (aufgrund von Ausnahmeanträgen beim Prüfungsausschuss, etc.) von Studenten. Diese Abweichungen werden im Folgenden *schematische Ausnahmen* genannt.

2 Lösungsansätze

Ein erster naiver Ansatz besteht darin, das komplexe Objekt, hier die komplette Prüfungsordnung, quasi zu kopieren und somit eine „private“ Kopie für eine Ausnahme zu erstellen. Dies ist auf den ersten Blick recht einfach, birgt aber eine Vielzahl von Problemen wie z.B. hohe Datenredundanz und damit einhergehende Änderungsanomalien, da nur Teile der Prüfungsordnung im Endeffekt Ausnahme von der betreffenden sind. Im Fall von Aktualisierungen muss enormer Aufwand betrieben werden, um alle

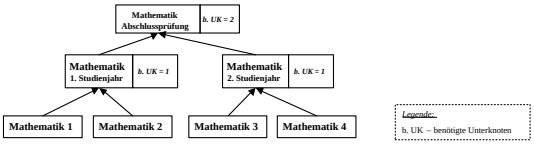

Abbildung 1: Graphische Darstellung der Zulassungskriterien zur Mathematik Abschlussprüfung

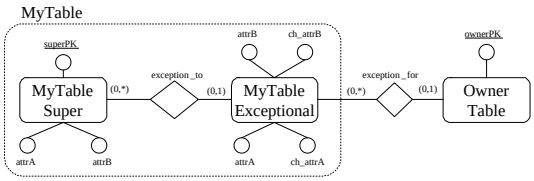

Abbildung 2: ER-Diagramm: Super Relation, Exceptional Relation, Owner Relation

„privaten“ (und eben zum großen Teil identischen) Kopien zu aktualisieren. Zusätzlich obliegt es der Anwendung (!) für die Konsistenzsicherung zu sorgen.

Explizite Unterstützung schematischer Ausnahmen

Dieser Ansatz der expliziten Unterstützung schematischer Ausnahmen basiert auf dem Grundsatz der minimalen Änderung. Bezogen auf das laufende Beispiel werden nur für bestimmte Teile der Prüfungsordnung Ausnahmen angelegt und explizit vom System transparent vor dem Benutzer verwaltet. Dies hat Vorteile: zum einen lassen sich diese Ausnahmen später leicht ermitteln und zum anderen, dass durch entsprechende Systemunterstützung die Komplexität der Ausnahmebehandlung vor dem Benutzer sich verbergen lässt. In der hier zugrundeliegenden Arbeit wird der Weg über einen Middleware Ansatz gewählt, welcher die in Abbildung 2 dargestellte Struktur für jede Relation, für welche Ausnahmen definiert werden, umsetzt. Für den Nutzer wird hierfür eine minimale Erweiterung von SQL ($\mathrm{SQL}-\mathrm{E}^{\mathrm{T}}$) bereitgestellt, um eine Transparenz der internen Strukturen der Middleware für den Anwender zu gewährleisten.

Um die in Abbildung 1 dargestellte Struktur mittels $\mathrm{SQL}-\mathrm{E}^{\mathrm{T}}$und der Middleware in ein DBMS abzubilden, muss zunächst die entsprechende Relation `POrdnung` angelegt werden.

```
CREATE TABLE POrdnung WITH PRIVATE EXCEPTION REFERENCES Student(MatrNr)
(
  poID            DECIMAL          NOT NULL,
  Bezeichnung     VARCHAR(255)     NOT NULL,
  parent          DECIMAL          NULL EXCEPTIONAL,
  bUK             INTEGER          NULL EXCEPTIONAL,
  CONSTRAINT PK_POrdnung PRIMARY KEY (poID),
  CONSTRAINT FK_POrdnung FOREIGN KEY (parent) REFERENCES POrdnung (poID)
)
```

Die oben dargestellt Anweisung hat zur Folge, dass die $\mathrm{SQL}-\mathrm{E}^{\mathrm{T}}$-Middleware im DBMS die folgenden zwei Relationen anlegt:

POrdnungSuper(poID,Bezeichnung, parent → POrdnungSuper(poID),bUK)) **und**
POrdnungExceptional(ID, parent, ch_parent, bUK, ch_bUK, superPK → POrdnungSuper(poID), ownerPK → Student(MatrNr))

Dabei bestimmt der Benutzer mit *WITH PRIVATE EXCEPTION*, dass die Relation mit schematischen Ausnahmen verwaltet werden soll und mit Angabe von *EXCEPTIONAL* über welchen Attributen Ausnahmen zugelassen werden sollen. Die Attribute `ch_XXX` dienen als Tag-Attribute, um anzuzeigen, ob über dem Attribut `XXX` eine gültige Ausnahme vorliegt.

Nach dem Eintragen der Beziehungen zur Beschreibung der Zulassungskriterien, weist die Relation `POrdnungSuper` nun folgende Ausprägung auf:

POrdnungSuper			
poID	**Bezeichnung**	**parent**	**bUK**
1	Mathematik Abschlussprüfung	NULL	2
2	Mathematik 1. Studienjahr	1	1
3	Mathematik 2. Studienjahr	1	1
4	Mathematik 1	2	NULL
5	Mathematik 2	2	NULL
6	Mathematik 3	3	NULL
7	Mathematik 4	3	NULL

3 Operationen

Prinzipiell werden alle Operationen der DML unterstützt. Interessant ist jedoch die Betrachtung der wichtigsten DML-Operation von $\mathrm{SQL}-\mathrm{E}^{\mathrm{T}}$, dem Update. Hierbei sind prinzipiell drei Fälle zu unterscheiden: a) Ein allgemein gültiges Tupel wird so verändert, dass eine Ausnahme angelegt wird; b) Eine bestehende Ausnahme wird so verändert, dass sie gegenüber dem zugehörigen allgemein gültigen Tupel

keine Ausnahme mehr darstellt; c) Ein allgemeines Tupel wird so verändert, dass existierende Ausnahmen bezüglich dieser Veränderung keine Ausnahme mehr sind. Das bedeutet im Kontext dieser Ausnahmen hat sich der allgemein gültige Zustand dem der Ausnahmen angenähert und die Ausnahme ist zur Normalität geworden. Zur Illustration werden im Folgenden nur die beiden Fälle a) und c) betrachtet.

Fall a) Für einen Studenten (MatNr. 4711) wird beschlossen, dass dieser nur noch folgende Zulassungskriterien erfüllen muss: (M3 $\wedge$ M4) $\rightarrow$ Zulassung, d.h. er benötigt kein M1 und M2 mehr, hingegen aber M3 und M4. Daraus ergeben sich folgende Anweisungen:

```
UPDATE POrdnung SET bUK=0 WHERE poID = 2 ON OWNER (MatrNr) VALUES (4711)
UPDATE POrdnung SET bUK=2 WHERE poID = 3 ON OWNER (MatrNr) VALUES (4711)
```

An den Ausprägungen der Relation `POrdnungSuper` findet durch diese Anweisungen keine Änderung statt. Dafür werden in der Relation `POrdnungExceptional` durch die Middleware zwei Tupel eingefügt, sodass `POrdnungExceptional` nun folgende Ausprägung wie in Tabelle 1a) aufweist.

POrdnungExceptional						
ID	**parent**	**ch_parent**	**bUK**	**ch_bUK**	**superPK**	**ownerPK**
1	NULL	FALSE	0	TRUE	2	4711
2	NULL	FALSE	2	TRUE	3	4711

POrdnungSuper			
poID	**Bezeichnung**	**parent**	**bUK**
...			
3	Mathematik 2. Studienjahr	1	2

Tabelle 1: Ausprägungen von a) POrdnungExceptional und b) POrdnungSuper

Fall c) Tritt der Fall ein, dass für alle Studenten M3 **und** M4 verpflichtend sind, also es gilt: (M1 $\vee$ M2) $\wedge$ (M3 $\wedge$ M4) $\rightarrow$ Zulassung, so resultiert dies in folgender Benutzeranweisung und einer Ausprägung in Tabelle 1b):

```
UPDATE POrdnung SET bUK = 2 WHERE poID = 3 WITH EXCEPTION RELEASE
```

Durch die Vorgabe `WITH EXCEPTION RELEASE` wird dem System mitgeteilt, eventuell existierende Ausnahmen aufzulösen, sodass sie gegenüber dem zugehörigen allgemeinen Tupel keine Ausnahme mehr darstellen. Mit dieser Aussage ergibt sich eine Ausprägung für `POrdnungExceptional`, die der von Tabelle 1a) ähnlich ist, jedoch ohne das Tupel mit der `ID=1`.

Um nun Anfragen an diese Relation ohne Kenntnis der internen Umsetzung realisieren zu können, bildet die $\mathrm{SQL} - \mathrm{E}^{\mathrm{T}}$ Middelware dem Benutzer eine Sicht an. Diese Sicht beinhaltet sowohl alle Tupel aus `POrdnungSuper` als auch die zugehörigen Ausnahmen, wobei diese bereits mit ihrem zugehörigen Tupel aus `POrdnungSuper` korrekt präsentiert werden.

```
CREATE VIEW POrdnung AS (
   SELECT s.poID, s.Bezeichnung,
    CASE WHEN ch_parent = 1 THEN e.parent ELSE s.parent END AS parent,
    CASE WHEN ch_bUK = 1 THEN e.bUK ELSE s.bUK END AS bUK,
    e.id AS exceptionalPK,
    e.ownerPK AS exceptionalOwnerPK, e.superPK AS exceptionalSuperPK
   FROM
     POrdnungSuper s INNER JOIN POrdnungExceptional e ON s.poID = e.superPK
   UNION
    SELECT
     poID, Bezeichnung, parent, bUK,
     NULL AS 'exceptionalPK', NULL AS 'exceptionalOwnerPK', NULL AS 'exceptionalSuperPK'
   FROM POrdnungSuper
)
```

4 Zusammenfassung

Schematische Ausnahmen werden in diesem Beitrag am Beispiel eingeführt und es wird eine für den Benutzer transparente Verwaltung durch eine Middleware-Schicht vorgestellt. Bei Aktualisierungsvorgängen wird angegeben, inwieweit bestehende Ausnahmen aufgehoben werden oder bestehen bleiben.