



OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG  
FAKULTÄT INFORMATIK  
INSTITUT FÜR TECHNISCHE INFORMATIONSSYSTEME

## Diplomarbeit

### Ermittlung extensionaler Überlappungen beim Entwurf föderierter Datenbanken

eingereicht am: 17. November 1997  
von: Anette Klaue, geb. Opp  
geboren am 10. Februar 1965  
in Brandenburg an der Havel

Betreuer: Prof. Dr. habil. Gunter Saake, Universität Magdeburg, ITI  
Dipl.-Inf. Ingo Schmitt, Universität Magdeburg, ITI

**Klaue, Anette:**

Ermittlung extensionaler Überlappungen  
beim Entwurf föderierter Datenbanken

Diplomarbeit

42 Seiten, 7 Abbildungen, 4 Tabellen

Otto-von-Guericke-Universität Magdeburg 1997

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>3</b>
1.1	Begriff des föderierten Datenbanksystems . . . . .	4
1.2	Architektur föderierter Datenbanksysteme . . . . .	6
1.3	Entwurf föderierter Datenbanksysteme . . . . .	7
<b>2</b>	<b>Definition wichtiger Begriffe</b>	<b>8</b>
2.1	Objektdatenbankmanagementsysteme . . . . .	8
2.2	Extensionen und extensionale Überlappungen . . . . .	9
2.3	Basisextensionen . . . . .	10
2.4	Spezialisierungsbeziehungen . . . . .	12
2.5	Schema einer Beispieldatenbank . . . . .	13
2.6	Totale und exklusive Spezialisierung . . . . .	14
2.7	Rollen . . . . .	15
2.8	Mehrfachspezialisierung . . . . .	15
<b>3</b>	<b>Ziel und Voraussetzungen der Arbeit</b>	<b>17</b>
3.1	Ziel der Arbeit . . . . .	17
3.2	Voraussetzungen für die Untersuchung . . . . .	18
3.3	Vorgehen bei der Untersuchung . . . . .	19
<b>4</b>	<b>Ermittlung der Basisextensionen für eine Komponentendatenbank</b>	<b>20</b>
4.1	Ableitung der Basisextensionen . . . . .	20
4.2	Eindeutigkeit des Verfahrens . . . . .	23
4.3	Angeben der Basisextensionen . . . . .	23
4.4	Zusammenfassung . . . . .	24
<b>5</b>	<b>Verallgemeinerung des Verfahrens</b>	<b>26</b>
5.1	Mehrere Oberklassen . . . . .	26
5.2	Rollenbeziehungen . . . . .	27
5.3	Zusammensetzen der Tabellen . . . . .	31
5.4	Spezialfälle für die Überlappung . . . . .	31
5.5	Mehrfachspezialisierung . . . . .	31
<b>6</b>	<b>Zusammenfassung</b>	<b>33</b>
6.1	Bestimmung der Basisextensionen der Komponentendatenbanken . . . . .	33
6.2	Bestimmung der Basisextensionen der föderierten Datenbank . . . . .	35
6.3	Eigenschaften des Verfahrens . . . . .	35

# Vorwort

In der vorliegenden Arbeit wird ein Verfahren zum Auffinden der Basisextensionen einer föderierten Datenbank vorgestellt. Damit soll der Prozeß der Schemenintegration beim Entwurf eines föderierten Datenbanksystems unterstützt werden. Die Basisextensionen dienen als Ausgangspunkt für die Erzeugung der Klassen des föderierten Schemas.

Es wird gezeigt, daß die Bestimmung der Basisextensionen nur eindeutig ist, wenn außer den Informationen über Klassenüberlappungen weitere Informationen zur Verfügung stehen. Das Verfahren ist deshalb im allgemeinen nicht vollständig automatisierbar.

An dieser Stelle möchte ich Herrn Prof. Dr. habil. Gunter Saake und Herrn Dipl.-Inf. Ingo Schmitt für die sehr gute Betreuung der Diplomarbeit danken.

# Kapitel 1

## Einleitung

Die Verwaltung von großen Datenbeständen mit Hilfe von Datenbanksystemen ist ein wichtiges Anwendungsgebiet der EDV. Klassische Einsatzgebiete von Datenbanksystemen in Unternehmen sind Buchhaltung, Material- und Personalverwaltung. Ein neueres Anwendungsgebiet ist z. B. die Verwaltung komplexer Objekte für CAx-Anwendungen.

Es entstand eine große Anzahl heterogener Datenbanksysteme, die jeweils für die speziellen Anwendungsgebiete entwickelt wurden und in denen die Daten so abgelegt sind, daß die Anwendungsprogramme effizient auf sie zugreifen können.

Diese Datenbanksysteme können sich sowohl in der Art der physischen Speicherung der Daten als auch in den verwendeten Datenmodellen, der Transaktionsverwaltung und der Integritätssicherung unterscheiden. Ein Datenbanksystem kann auf dem relationalen Datenmodell und ein anderes auf einem objektorientierten Datenmodell basieren. Aber auch bei einem einheitlichen Datenmodell gibt es Unterschiede zwischen den Schemata der Datenbanken.

Die so entstandene Heterogenität von Systemen wird in [SCC<sup>+</sup>97] unterschieden nach

- der Heterogenität der Hardware,
- der Heterogenität der zugrunde liegenden Datenmodelle,
- der Heterogenität der Datenhaltungs- bzw. Datenbankmanagementsysteme,
- der Heterogenität der Schemata und
- der Heterogenität der Daten.

Trotz dieser Heterogenität können die unterschiedlichen Datenbanken Daten über denselben Realweltausschnitt enthalten. So können Daten über in einem Unternehmen verarbeitete Materialien sowohl in einer Datenbank der Lagerverwaltung, als auch in einer Datenbank der Buchungsabteilung gespeichert sein, wobei hier sicher jeweils unterschiedliche Eigenschaften wie z. B. Lagerort oder Preis als Attribute angegeben sind.

Die Verwaltung solcher Daten in verschiedenen Einzeldatenbanken hat viele Nachteile. Durch die Redundanzen der Datenspeicherung besteht die Gefahr von Inkonsistenz. Es ist nicht gesichert, daß eine Änderung, die in einer der Datenbanken durchgeführt wurde, auch in allen anderen erfolgt. Wegen der Heterogenität der Schemata ist es schwer möglich, Anfragen über Daten mehrerer Datenbanken zu ermöglichen. Auch ist der Austausch von Daten zwischen den verschiedenen Datenbanken nur über definierte Schnittstellen möglich. Neue Anwendungen benötigen oft eine einheitliche Datenbankschnittstelle zu den verteilt

verwalteten Daten. Über diese Schnittstelle soll globalen Anwendungen der Zugriff auf Daten in verschiedenen Datenbanken möglich sein, ohne daß die bisherigen Anwendungen in ihrer Funktion beeinträchtigt werden.

Das erneute Erfassen aller Daten in einer gemeinsamen Datenbank ist aus verschiedenen Gründen nicht sinnvoll. Während die einzelnen Datenbanken auf unterschiedlichen Datenmodellen basieren, muß für die gemeinsame Datenbank ein einziges Datenmodell und ein einheitliches Schema gefunden werden und die Schemata der Einzeldatenbanken an dieses angepaßt werden. Die Anwendungen, die für die ursprünglichen Datenmodelle und Schemata geschrieben wurden, könnten dann größtenteils nicht mehr eingesetzt werden. In einem Unternehmen müßte die Umstellung auf die neue Datenverwaltung während des laufenden Betriebs erfolgen. Bei dieser kann es zu Fehlern bei der Übertragung der Daten kommen. Außerdem ist es nicht immer gewünscht oder sinnvoll, alle gespeicherten Daten allen Anwendern zur Verfügung zu stellen.

Eine andere Möglichkeit für einen gemeinsamen Zugriff auf die Daten in unterschiedlichen Datenbanken ist die Integration der Einzelsysteme in ein föderiertes Datenbanksystem. Solche *föderierten Datenbanksysteme* sollen die verschiedenen Formen der Heterogenität beherrschbar machen und gleichzeitig die lokale Autonomie der Systeme wahren.

Wir werden den Begriff des föderierten Datenbanksystems im nächsten Abschnitt genauer beschreiben.

## 1.1 Begriff des föderierten Datenbanksystems

Bei der *Föderierung von Datenbanksystemen* erfolgt die Integration der Daten nicht auf der Datenebene, sondern auf der Schemaebene. Das bedeutet, daß die Daten weiterhin von den lokalen Systemen verwaltet werden und für globale Anwendungen ein integriertes Schema einer globalen Datenbank simuliert wird.

Nach [SCC+97] besteht ein *föderiertes Datenbanksystem* aus einem *föderierten Datenbankmanagementsystem* und aus einem oder mehreren *Komponentendatenbanksystemen*. Das föderierte Datenbankmanagementsystem wird als *Föderierungsdienst* oder *Föderierungsschicht* bezeichnet. Logisch und physisch verteilte heterogene Datenbanksysteme werden bei weitgehender Erhaltung der lokalen Autonomie der Komponentendatenbanksysteme föderiert.

Die Komponentendatenbanksysteme können dabei hinsichtlich der Laufzeitumgebung, des Datenmodells und der Datenbankschemata heterogen sein. Auf den lokalen Schnittstellen der Komponentendatenbanken arbeiten lokale Applikationen, die auf deren spezielle Gegebenheiten zugeschnitten sind. Sie werden von der Föderierung normalerweise nicht beeinflusst. Das föderierte Datenbankmanagementsystem, der Föderierungsdienst, operiert ebenfalls auf den lokalen Schnittstellen. Es verhält sich wie eine lokale Anwendung.

Ein *föderiertes Datenbanksystem* ist also ein System zur Realisierung eines integrierten Zugriffs auf Daten, die in unterschiedlichen heterogenen Datenbanken gespeichert sind. Der Zugriff auf die Daten erfolgt dabei durch den Föderierungsdienst. Zu den Aufgaben des Föderierungsdienstes gehören:

- die Überwindung der Heterogenität,
- die Transaktionsverwaltung,
- die Konsistenzsicherung und
- die Bereitstellung einer Abfragesprache.

Über externe Schnittstellen können globale Anwendungen die Dienste des föderierten Datenbanksystems nutzen.

Die Verteilung der Daten auf die Komponentendatenbanken ist für die Anwendungen transparent. Für diese ist nicht wesentlich, aus welcher der Komponentendatenbanken die Daten stammen, mit denen sie arbeiten. Welche Daten der Föderation zur Verfügung gestellt werden, entscheiden die Administratoren der Komponentendatenbanken.

Eine Datenbank kann auch als Komponente an mehreren Föderierungen beteiligt sein. Während der Lebenszeit der Föderation können weitere Datenbanken hinzukommen oder auch Datenbanken aus der Föderation herausgenommen werden.

Für die föderierte Datenbank wird ein neues Schema, das *föderierte Schema*, entwickelt, ohne daß die Schemata der Komponentendatenbanken davon beeinflusst werden. Neue Anwendungen können dann auch direkt für das Schema der föderierten Datenbank entwickelt werden.

Unter einem *föderierten Datenbanksystem* verstehen wir also eine Menge von heterogenen Datenbanksystemen, den *Komponentendatenbanksystemen*, auf die ein globaler Zugriff mit Hilfe eines föderierten Datenbankmanagementsystems, dem *Föderierungsdienst*, möglich ist. Die Komponentendatenbanksysteme bleiben autonom, d. h. die lokalen Anwendungen werden von der Föderierung (normalerweise) nicht beeinflusst und können auch ausgeführt werden, während die Datenbank an der Föderierung teilnimmt. Der Datenaustausch zwischen den Komponentendatenbanken und den globalen Anwendungen auf der föderierten Datenbank erfolgt über den Föderierungsdienst, der wie eine lokale Anwendung auf den Komponentendatenbanken arbeitet.

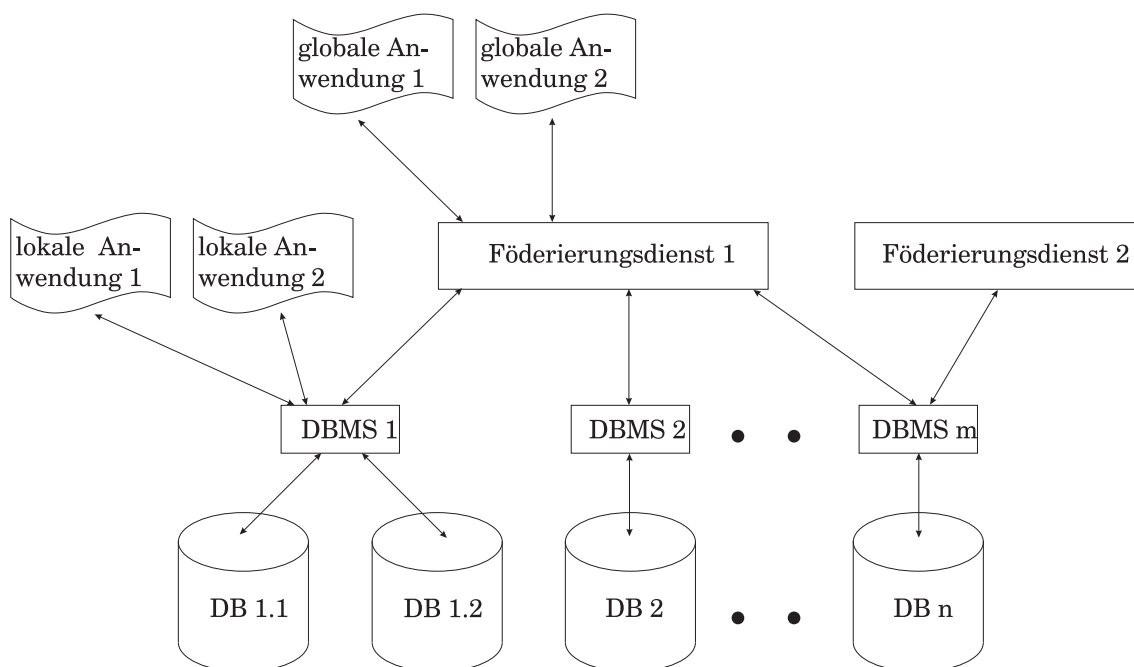


Abbildung 1.1: Föderierte Datenbanken

Der prinzipielle Aufbau eines föderierten Datenbanksystems ist in Abbildung 1.1 dargestellt. Auf die Datenbankmanagementsysteme 1 bis  $m$  greifen sowohl lokale Anwendungen als auch

der Föderierungsdienst zu. Zusätzlich können globale Anwendungen direkt auf der föderierten Datenbank laufen.

Im nächsten Abschnitt werden wir eine mögliche Architektur für föderierte Datenbanken angeben.

## 1.2 Architektur föderierter Datenbanksysteme

Die Architektur von Datenbanken wird in [TK78] durch die ANSI/SPARC–Drei-Ebenen-Schemen-Architektur mit *internen Schema*, *konzeptuellen Schema* und *externen Schemata* beschrieben. Dieses reicht für die Beschreibung der Architektur föderierter Datenbanksysteme aber nicht aus. Eine mögliche Erweiterung ist die Beschreibung durch eine 5-Ebenen-Architektur nach [SL90]. Diese soll durch die Abbildung 1.2 veranschaulicht werden.

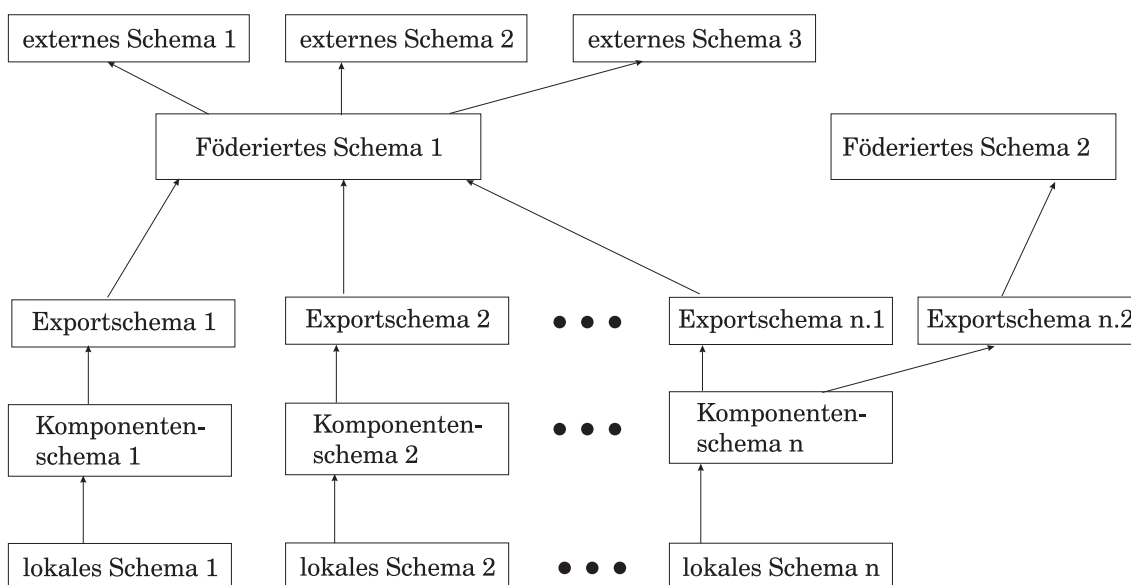


Abbildung 1.2: 5-Ebenen-Architektur

Hierbei entsprechen die *lokalen Schemata* den konzeptuellen Schemata der Komponentendatenbanken. Diese Schemata können auf unterschiedlichen Datenmodellen basieren. Die lokalen Schemata werden zunächst in *Komponentenschemata* transformiert. Man erhält dadurch für jede Komponentendatenbank eine Darstellung in einem einheitlichen Datenmodell. Da im allgemeinen nicht alle Daten der Komponentendatenbanken auch in der föderierten Datenbank enthalten sein sollen, werden die Daten noch einmal in den *Exportschemata* dargestellt. Diese enthalten den Teil der Daten der Komponentenschemata, der der föderierten Datenbank zur Verfügung gestellt werden soll.

Aus den Exportschemata wird dann das Schema der föderierten Datenbank entwickelt. Dieses *föderierte Schema* ist also die Integration der Exportschemata der Komponentendatenbanken zu einem einheitlichem Schema. Die Überführung der Schemata der Komponentendatenbanken in das föderierte Schema heißt *Schemenintegration*.

Aus dem föderierten Schema können nun noch unterschiedliche Sichten für die verschiedenen Anwendergruppen abgeleitet werden, die auch hier *externe Schemata* heißen.

## 1.3 Entwurf föderierter Datenbanksysteme

Bevor man föderierte Datenbanksysteme einsetzen kann, muß ein Entwurf der föderierten Datenbank durchgeführt werden. Ein Teil dieses sehr komplexen Vorganges ist, wie schon im letzten Abschnitt erwähnt, die *Schemenintegration*. Innerhalb dieses Schrittes muß die Heterogenität der zu integrierenden Schemata überwunden werden. Die Datenmodellheterogenität wird durch die Transformation der zu integrierenden Schemata in ein einheitliches Datenmodell beseitigt. Es gibt verschiedene Wege, das föderierte Schema zu erzeugen.

In [SEHT96] wird die Transformation der zu integrierenden Schemata in das Datenmodell GIM (*Generic Integration Model*) vorgeschlagen. Eine wichtige Rolle dabei spielt der *extensionale Konflikt*. Damit ist gemeint, daß Informationen über dasselbe Realweltobjekt in unterschiedlichen Klassen unterschiedlicher Datenbanken gespeichert sein können.

Zur Auflösung dieses Konfliktes müssen diese sich überlappenden Klassen erkannt und die Mengen von Objekten, die gleichzeitig in verschiedenen Klassen liegen, beschrieben werden. Dadurch werden Redundanzen in der föderierten Datenbank vermieden.

Mit dieser Arbeit soll der Prozeß der Schemenintegration unterstützt werden, indem ein Verfahren zum Auffinden der extensionalen Überlappungen von Objektklassen angegeben wird. Nach der Einführung weiterer Begriffe in den folgenden Kapiteln werden wir die Zielstellung der Arbeit in Kapitel 3 genauer beschreiben.

# Kapitel 2

## Definition wichtiger Begriffe

Im ersten Kapitel haben wir den Begriff des föderierten Datenbanksystems eingeführt. Die Schemata der Komponentendatenbanken einer föderierten Datenbank können auf unterschiedlichen Datenmodellen beruhen. In dieser Arbeit gehen wir jedoch davon aus, daß die Komponentendatenbanken der zu betrachtenden föderierten Datenbanken auf objektorientierten Datenmodellen basieren. Wir werden jetzt wichtige objektorientierte Begriffe definieren.

### 2.1 Objektdatenbankmanagementsysteme

Für den Begriff des Objektes gibt es keine allgemein akzeptierte Definition. Nach [SST97] verstehen wir unter einem *Objekt* eine Modellierungseinheit, die einen eingekapselten Zustand hat sowie Operationen bereitstellt. Objekte können miteinander kooperieren, indem sie Botschaften austauschen. Sie

- haben einen veränderbaren Zustand,
- besitzen eine zustandsunabhängige Identität und
- werden explizit bei der Modellierung eines bestimmten Problembereichs entworfen.

Unter einem *Objektdatenbankmanagementsystem* verstehen wir nach [SST97] ein Datenbankmanagementsystem, das

- auf einem Objektdatenbankmodell basiert,
- eine objektorientierte Datenbankprogrammierungsumgebung anbietet
- (zumindest konzeptuell) erweiterbar ist, und
- darüber hinaus die folgende Datenbankfunktionalität unterstützt:
  1. Persistenz
  2. Anfragesprache
  3. Transaktionen
  4. Synchronisation im Mehrbenutzerbetrieb
  5. Datensicherung

6. Integritätssicherung
7. Sichtdefinition
8. Zugriffsschutz
9. Schemaevolution

Im Gegensatz zu relationalen Datenbankmanagementsystemen, deren Datenmodell eindeutig beschrieben ist, gibt es für Objektdatenbanken kein einheitliches, sondern viele verschiedene, sehr unterschiedliche Objektdatenbankmodelle. Auf Unterschiede zwischen den Datenmodellen, die für diese Arbeit wesentlich sind, werden wir an den entsprechenden Stellen eingehen. Grundlegende objektorientiert Konzepte liegen aber allen diesen Datenbankmodellen zugrunde.

Ein objektorientiertes System muß eine sich dynamisch entwickelnde Anzahl von Objekten verwalten. Objekte mit gleichen Merkmalen und gleichem Verhalten werden zu einer *Objektklasse* oder kurz *Klasse* zusammengefaßt.

## 2.2 Extensionen und extensionale Überlappungen

Als nächstes definieren wir den Begriff der *Extension einer Objektklasse*. Die Extension einer Objektklasse umfaßt nach [SST97] die Menge aller Objekte, die in dieser Klasse liegen. Diese Definition bezieht sich nur auf Objekte im aktuellen Datenbankzustand. In dieser Arbeit ist es aber auch wichtig, die Objekte zu behandeln, die erst während der Arbeit mit einer Datenbank eingefügt werden. Deshalb werden wir den Begriff wie in [SS97] erweitern.

Unter der *Extension einer Objektklasse* verstehen wir die Menge aller möglichen Objekte dieser Klasse, also auch die Objekte, die erst während der Arbeit mit der Datenbank in die Klasse eingefügt werden können.

Deshalb werden wir unter einem *Objekt in einer (förderierten) Datenbank* auch stets ein mögliches Objekt verstehen, also ein Objekt, das bereits gespeichert ist oder das in der Zukunft gespeichert werden könnte.

Die Aussage, daß ein Objekt in einer bestimmten Objektklasse liegt, bezieht sich also nicht nur auf den aktuellen, sondern auch auf alle künftigen Datenbankzustände.

Enthält die Extension einer Klasse auch Objekte, die in den Extensionen anderer Klassen liegen, dann überlappen sich diese Klassen extensional. Unter der *extensionalen Überlappung* von Objektklassen verstehen wir also den Teil der Extension dieser Klassen, der Objekte enthält, die dieselben Realweltobjekte abbilden. Da Klassenextensionen Objektmengen sind, können wir den Begriff der extensionalen Überlappung mit Hilfe von Mengenbeziehungen beschreiben.

Es *überlappen* sich  $n$  Klassen  $Klasse_1, \dots, Klasse_n$  *extensional*, wenn der Durchschnitt der Extensionen dieser Klassen  $K_1, \dots, K_n$  nicht leer ist. Diese Schnittmenge  $K_1 \cap \dots \cap K_n \neq \emptyset$  ist dann die *extensionale Überlappung* dieser Klassen.

Extensionale Überlappungen können zwischen den Klassen einer Datenbank, aber auch zwischen Klassen unterschiedlicher Komponentendatenbanken einer förderierten Datenbank auftreten.

## 2.3 Basisextensionen

Die Basisextensionen bilden den Ausgangspunkt für die Bildung der Klassen des föderierten Schemas im Datenmodell GIM [SEHT96].

*Basisextensionen* sind Mengen von Objekten einer (föderierten) Datenbank, die in jeweils denselben Klassen liegen. Basisextensionen können durch eine Relation beschrieben werden. Objekte stehen dann in Relation zueinander, wenn sie jeweils in denselben Klassen liegen.

Diese Relation ist reflexiv, transitiv und symmetrisch, also eine Äquivalenzrelation. Die Äquivalenzklassen dieser Relation bilden die Basisextensionen. Das bedeutet, daß jedes Objekt der Datenbank in genau einer der Basisextensionen liegt. Basisextensionen sind deshalb auch nie leer und die Vereinigung aller Basisextensionen ist genau die Menge aller Objekte der Datenbank.

Sind in einer (föderierten) Datenbank die Extensionen der Objektklassen disjunkt, so bildet jede Extension auch eine Basisextension. Überlappen sich einige der Klassen extensional, so gibt es weitere Basisextensionen, deren Anzahl von der Zahl der Überlappungen und der Anzahl der an den Überlappungen beteiligten Klassen abhängt.

Die Abbildung 2.1 veranschaulicht noch einmal die Begriffe *extensionale Überlappung* und *Basisextension* für eine Datenbank mit drei Objektklassen  $Klasse_1$ ,  $Klasse_2$  und  $Klasse_3$ . Dabei sind  $K_1$ ,  $K_2$  und  $K_3$  die Extensionen dieser Klassen.

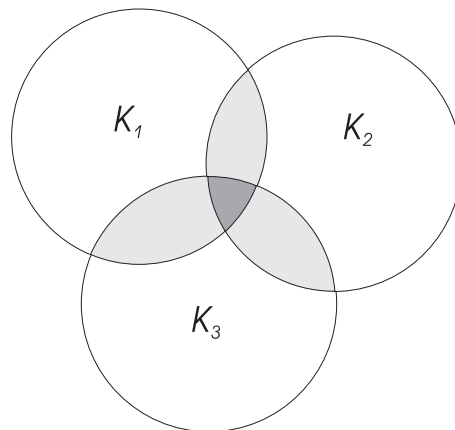


Abbildung 2.1: Überlappung dreier Klassen

Durch die Überlappung der drei Klassen entstehen vier extensionale Überlappungen und zwar zwischen den folgenden Klassen:

1.  $Klasse_1$  und  $Klasse_2$ ,
2.  $Klasse_1$  und  $Klasse_3$ ,
3.  $Klasse_2$  und  $Klasse_3$  sowie
4.  $Klasse_1$ ,  $Klasse_2$  und  $Klasse_3$ .

Diese Überlappungen bilden sieben Basisextensionen, die die Objekte enthalten, die jeweils in folgenden Klassenextensionen liegen:

1. nur in  $K_1$ ,
2. nur in  $K_2$ ,
3. nur in  $K_3$ ,
4. in  $K_1$  und  $K_2$  und nicht in  $K_3$ ,
5. in  $K_1$  und  $K_3$  und nicht in  $K_2$ ,
6. in  $K_2$  und  $K_3$  und nicht in  $K_1$ ,
7. in  $K_1, K_2$  und  $K_3$ .

Um die Basisextensionen darzustellen wird in [SS97] eine Tabelle wie 2.1 benutzt. Die Spalten der Tabelle entsprechen hier den Basisextensionen. Markierungen geben an, aus welchen Extensionen die Objekte der jeweiligen Basisextension stammen. Eine Markierung in einer Zeile bedeutet, daß die Basisextension Objekte aus der entsprechenden Klasse enthält. Ist eine Klasse nicht markiert, so enthält die Basisextension keine Objekte dieser Klasse. Die Tabelle 2.1 zeigt die Darstellung der Basisextensionen für das Beispiel auf Seite 10.

Klasse	Basisextension						
	1	2	3	4	5	6	7
$K_1$	x			x	x		x
$K_2$		x		x		x	x
$K_3$			x		x	x	x

Tabelle 2.1: Basisextensionen für drei Klassen

Eine weitere Möglichkeit die Basisextensionen zu beschreiben, sind *Schnittmengen der beteiligten Klassenextensionen*. Eine Basisextension ist dann der Schnitt aller Klassenextensionen, aus denen sie Objekte enthält. Außerdem muß gefordert werden, daß diese Schnittmenge die Komplemente der Extensionen der restlichen Klassen enthält, um auszuschließen, daß verschiedene Basisextensionen gemeinsame Objekte haben. Die Basisextensionen aus dem Beispiel werden durch die folgenden Mengendurchschnitte beschrieben:

1.  $K_1 \cap \overline{K_2} \cap \overline{K_3}$
2.  $\overline{K_1} \cap K_2 \cap \overline{K_3}$
3.  $\overline{K_1} \cap \overline{K_2} \cap K_3$
4.  $K_1 \cap K_2 \cap \overline{K_3}$
5.  $K_1 \cap \overline{K_2} \cap K_3$
6.  $\overline{K_1} \cap K_2 \cap K_3$

7.  $K_1 \cap K_2 \cap K_3$ 

Wir werden diese Beschreibung der Basisextensionen durch Schnittmengen in dieser Arbeit benutzen, da sich so die Beziehungen zwischen den Klassenextensionen einfach darstellen lassen.

Auch in der Arbeit [Bus91] wird bei der Behandlung eines formalen Ansatz zur Beschreibung des Schemenentwurfes diese Darstellung für die Objektmengen benutzt. Die Schnittmengen werden dort als Klauseln bezeichnet. Wir werden die Schnittmengen ebenfalls als *Klauseln* und die nichtleeren Schnittmengen als *Basisklauseln* bezeichnen.

Da in den Klauseln alle Mengen und ihre Komplemente in beliebigen Kombinationen auftreten können, aber jede nur einmal, entweder als Menge oder Komplement, gibt es für  $n$  Klassen insgesamt  $2^n$  mögliche Klauseln. Nur die nichtleeren Klauseln beschreiben aber auch eine Basisextension.

Diese Basisklauseln lassen sich auf die Spalten der Tabelle 2.1 abbilden, wenn für jede Klasse, die nicht als Komplement in der Klausel auftritt, eine Markierung gesetzt wird. Wenn also die Basisklauseln bekannt sind, dann läßt sich aus diesen sofort die Tabelle der Basisextensionen ableiten.

Für das Beispiel wird durch jede mögliche Kombination von Klassenextensionen und Komplementen – außer  $\overline{K_1} \cap \overline{K_2} \cap \overline{K_3}$  – auch eine Basisextension beschrieben, da sich die Klassen vollständig überlappen. *Vollständige Überlappung* bedeutet hier, daß jede Extension an allen möglichen Überlappungen der anderen teilnimmt. Das wird aber in einer Datenbank normalerweise nicht der Fall sein. Die Zahl der Basisextensionen wird dort im allgemeinen wesentlich geringer sein, als die Zahl der möglichen Klauseln, da sie von der Art und Anzahl der extensionalen Überlappungen abhängt.

## 2.4 Spezialisierungsbeziehungen

Wir werden in diesem Abschnitt erklären, was wir unter *Spezialisierung* von Klassen verstehen. Die Erklärung des Begriffs entnehmen wir [SST97].

Mit Hilfe des Konzepts der *Spezialisierung zwischen Klassen* lassen sich Spezialisierungsbeziehungen zwischen verschiedenen Begriffen ausdrücken, die durch Klassen repräsentiert werden. Die Spezialisierung erlaubt eine Strukturierung von Klassen. Faßt man *Spezialisierung als Prozeß* auf, werden die Extensionen der spezialisierten Klassen aus existierenden Klassenextensionen abgeleitet.

Stehen Klassen zueinander in einer Spezialisierungsbeziehung hat das Auswirkungen auf die Extensionen dieser Klassen. Für uns ist die Aussage wichtig, daß die Extensionen der spezialisierten Unterklassen *Teilmengen* der Extensionen der zu spezialisierenden Oberklassen sind. Spezialisierungsbeziehungen sind transitiv, reflexiv und antisymmetrisch. Diese Beziehungen gelten dann auch für die Teilmengenbeziehungen der Extensionen. Die Spezialisierungsbeziehungen können in einer Spezialisierungshierarchie dargestellt werden, an deren Spitze die allgemeinsten Klassen stehen. Alle Extensionen der durch Spezialisierung aus diesen allgemeinsten Klassen abgeleiteten Klassen sind dann auch Teilmengen von deren Extensionen.

Die Spezialisierungsbeziehungen und damit auch die Teilmengenbeziehungen einer bestimmten Datenbank können aus dem Schema entnommen werden.

Wird eine Klasse durch Spezialisierung in Unterklassen zerlegt, unterscheidet man zwischen der *flachen* und der *tiefen* Extension dieser Klasse. Der Begriff der tiefen Extension

entspricht der Definition der Extension in Abschnitt 2.2. Zur flachen Extension gehören alle Objekte, die nicht in einer der Unterklassen liegen. Hat eine Klasse keine Unterklassen, so sind flache und tiefe Extension gleich.

In einigen Datenmodelle für objektorientierte Datenbanken werden alle Objektklassen aus einer einzigen allgemeinen Oberklasse spezialisiert. In anderen kann es mehrere solche Oberklassen geben. Im ersten Fall sind die Extensionen aller Klassen der Datenbank Teilmengen der Extension der allgemeinen Oberklasse.

Nach der Einführung einer Beispieldatenbank im nächsten Abschnitt werden wir weitere wichtige Begriffe im Zusammenhang mit Spezialisierungsbeziehungen definieren.

## 2.5 Schema einer Beispieldatenbank

Wir vereinbaren, im folgenden in der Schreibweise nicht mehr zwischen den Bezeichnungen für die Klassen und den für die Extensionen zu unterscheiden, da die Bedeutung aus dem Zusammenhang hervorgeht. Es bezeichnet also z. B.  $K_i$  gleichzeitig die Klasse selbst und ihre Extension.

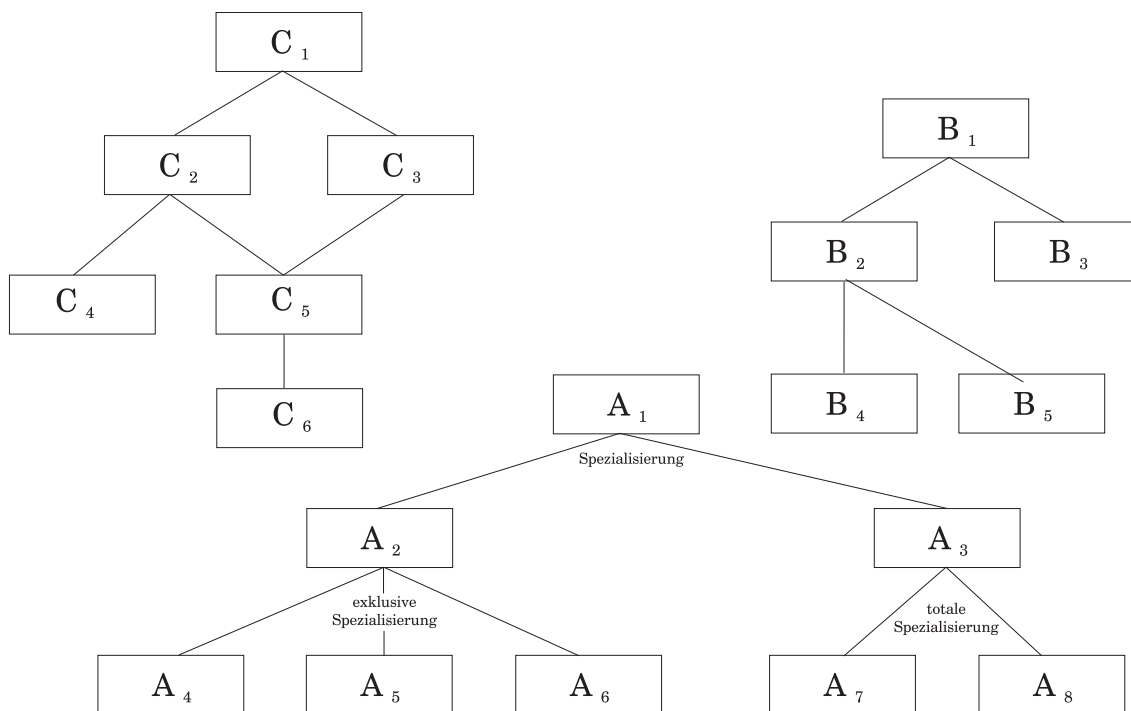


Abbildung 2.2: Schema der Beispieldatenbank

In diesem Abschnitt werden wir das Schema einer Beispieldatenbank angeben. An dieser werden wir in diesem und den nächsten Kapiteln die Definitionen und Überlegungen veranschaulichen. Das Schema der Beispieldatenbank ist in Abbildung 2.2 dargestellt.

Die Beispieldatenbank hat drei allgemeinste Klassen, die jeweils keine weiteren Oberklassen mehr haben. Alle anderen Klassen sind aus einer dieser Klassen spezialisiert worden.

Für die Überlegungen müssen wir oft nur einen Teil des Schemas betrachten. Da wir die von den drei allgemeinsten Klassen  $A_1$ ,  $B_1$  und  $C_1$  abgeleiteten Klassen mit  $A_i$ ,  $B_i$  bzw.  $C_i$  bezeichnen, werden wir dann auch die drei Teile des Schemas mit  $A$ ,  $B$  bzw.  $C$  bezeichnen.

Wir wollen noch bemerken, daß die Abbildung 2.2 auch als Beispiel für mehrere Datenbanken dienen kann, indem man die Teile  $A$ ,  $B$  und  $C$  als Schemata verschiedener Datenbanken auffaßt.

## 2.6 Totale und exklusive Spezialisierung

Zwei weitere wichtige Begriffe, die wir in der Arbeit benötigen werden, sind die *totale* und die *exklusive* Spezialisierung.

Eine Spezialisierung einer Oberklasse in Unterklassen ist *total*, wenn die Oberklasse so in Unterklassen zerlegt wird, daß alle Objekte der Oberklasse auch in mindestens einer der Unterklassen liegen. Anders ausgedrückt bedeutet totale Spezialisierung, daß die flache Extension der Oberklasse leer ist.

Eine Spezialisierung einer Klasse in Unterklassen ist *exklusiv*, wenn sich die Unterklassen nicht überlappen, d. h. jedes Objekt der Oberklasse liegt entweder in der flachen Extension oder in genau einer der Unterklassen. Die Unterklassen haben also keine gemeinsamen Objekte.

Eine Spezialisierungsbeziehung kann auch total und exklusiv oder weder total noch exklusiv sein. Im Gegensatz zu [SST97] werden wir auch dann von totaler und exklusiver Spezialisierung sprechen, wenn es nur eine spezialisierte Unterklasse gibt, um diesen Fall nicht extra behandeln zu müssen. Eine solche Spezialisierung ist dann immer exklusiv und normalerweise nicht total.

Wir werden den Teil  $A$  des Schemas der Beispieldatenbank nutzen, um die Begriffe noch einmal zu veranschaulichen. In der Beispieldatenbank ist die Spezialisierung der Klasse  $A_2$  in die Klassen  $A_3$ ,  $A_4$  und  $A_5$  exklusiv, aber nicht total, die Spezialisierung der Klasse  $A_3$  in  $A_7$  und  $A_8$  total, aber nicht exklusiv. In den Abbildungen 2.3 und 2.4 sind diese Beziehungen dargestellt.

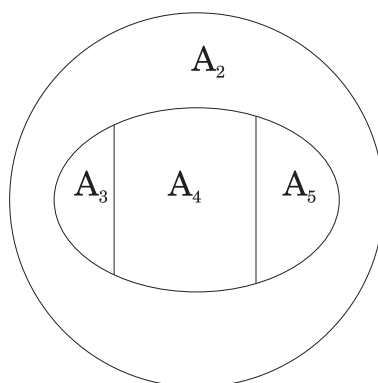
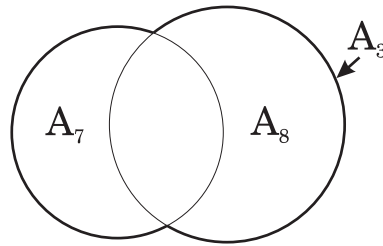


Abbildung 2.3: Exklusive Spezialisierung von  $A_2$  in  $A_3$ ,  $A_4$  und  $A_5$

Nichtexklusive Spezialisierungen sind nicht in allen objektorientierten Datenmodellen erlaubt. Sie können nur auftreten, wenn das Datenmodell Rollen zuläßt, ein Objekt also gleichzeitig in mehr als einer Klasse liegen darf, oder wenn Mehrfachspezialisierung zugelassen ist. Wir werden diese beiden Konzepte in den nächsten Abschnitten näher beschreiben.

Abbildung 2.4: Totale Spezialisierung von  $A_3$  in  $A_7$  und  $A_8$ 

Mit den Begriffen totale und exklusive Spezialisierung können wir auch die anderen Spezialisierungen der Beispieldatenbank beschreiben. Der Teil  $A$  besteht aus acht Klassen. Alle Klassen werden von einer gemeinsamen Oberklasse  $A_1$  abgeleitet. Diese Oberklasse  $A_1$  wird in die Unterklassen  $A_2$  und  $A_3$  spezialisiert. Diese Spezialisierung ist weder total noch exklusiv. Die Spezialisierung von  $A_2$  in  $A_4$ ,  $A_5$  und  $A_6$  ist exklusiv, aber nicht total und die Spezialisierung von  $A_3$  in  $A_7$  und  $A_8$  nicht exklusiv, aber total. Für den Teil  $B$  nehmen wir an, daß alle Spezialisierungen total und exklusiv sind. Im Teil  $C$  tritt Mehrfachspezialisierung auf. Wir werden diesen Teil des Schemas im übernächsten Abschnitt nach der Vorstellung diese Konzeptes genauer beschreiben.

## 2.7 Rollen

Läßt das Datenmodell einer objektorientierten Datenbank *Rollen* zu, können Objekte gleichzeitig in den Extensionen verschiedener Klassen liegen, auch wenn diese nicht in einer Spezialisierungsbeziehung stehen. Läßt das Datenmodell nichtexklusive Spezialisierung zu, dann sind Rollen zwischen Klassen, die durch nichtexklusiven Spezialisierung aus einer gemeinsamen Oberklasse hervorgegangen sind, erlaubt. Nach [SST97] sind in den meisten Datenmodellen nur Rollen zwischen Klassen, die aus einer gemeinsamen Oberklasse spezialisiert sind, zugelassen. Wir werden aber auch den Fall untersuchen, daß Objekte Rollen zwischen beliebigen Klassen annehmen können.

Ist von einer Menge von Objektklassen bekannt, daß ein Objekt in ihnen Rollen annimmt, so überlappen sich diese Klassen extensional.

Ob Rollen zugelassen sind, kann man den Informationen über das Datenmodell entnehmen. Läßt das Datenmodell einer Komponentendatenbank Rollen zu, muß vom Administrator der Datenbank erfragt werden, ob und zwischen welchen Klassen der Komponentendatenbank Rollen auftreten. Manche Objektmodelle verlangen eine Angabe der Klassen zwischen denen Rollen zugelassen sind.

Für unser Beispiel wollen wir annehmen, daß neben den Rollenbeziehungen, die auf Grund der nichtexklusiven Spezialisierung erlaubt sind, Objekte auch Rollen in den Klassen  $A_3$  und  $B_2$  haben können.

## 2.8 Mehrfachspezialisierung

Ein Objekt kann auch dann gleichzeitig in verschiedenen Klassen liegen, wenn eine Klasse mehrere direkte Oberklassen hat. Diese Art der Spezialisierung wird als *Mehrfachspezialisierung* bezeichnet. Bei Mehrfachspezialisierungen enthält die spezialisierte Klasse genau

die Objekte, die in allen zu spezialisierenden Klassen liegen. Das Konzept der Mehrfachspezialisierung kann als Ersatz für die Rollenbeziehung genutzt werden, die nicht in allen Objektdatenbankmodellen zugelassen ist, indem gemeinsame Objekte mehrerer Klassen in einer gemeinsamen Unterklasse erfasst werden.

In der Beispieldatenbank tritt Mehrfachspezialisierung im Teil  $C$  auf. Dort hat die Klasse  $C_5$  als direkte Oberklasse sowohl die Klasse  $C_2$  als auch der Klasse  $C_3$ . Die Klassen  $C_2$  und  $C_3$  überlappen sich extensional. Es liegen aber alle gemeinsamen Objekte in der Klasse  $C_5$ . Die flachen Extensionen der Oberklassen sind damit disjunkt. Es kann damit auch kein Objekt der Überlappung  $C_2 \cap C_3 = C_5$  in der Klasse  $C_4$  liegen.

Die Spezialisierung von  $C_6$  aus  $C_5$  ist ein Beispiel für die Spezialisierung nur einer Klasse. Diese Spezialisierungen werden wir als exklusiv und nicht total auffassen.

# Kapitel 3

## Ziel und Voraussetzungen der Arbeit

Wie bereits in Kapitel 1 angedeutet, ist die Erstellung einer föderierten Datenbank ein aufwendiger Prozeß. Ein Teil ist die Schemenintegration, die Überführung der Schemata der Komponentendatenbanken in das Schema der föderierten Datenbank. Das Ziel der Schemenintegration ist nach [SCC<sup>+</sup>97] die Überwindung der Heterogenität zwischen den auf verschiedenen Datenmodellen basierenden Schemata der Komponentendatenbanken. Dabei sind die Konflikte zwischen den verschiedenen Schemata aufzulösen. Ziel ist es ein minimales föderiertes Schema zu erzeugen, um Redundanzen zu vermeiden.

### 3.1 Ziel der Arbeit

Mit dieser Arbeit soll der Prozeß der Schemenintegration unterstützt werden. Um auf ein minimales föderiertes Schema zu kommen, ist es notwendig, alle extensionalen Überlappungen zwischen den Klassen der zu föderierenden Datenbanken und die dabei entstehenden Basisextensionen zu kennen.

Es soll untersucht werden, welche Informationen hilfreich sind, um die Überlappungen zu erkennen und wie man diese Informationen finden und verarbeiten kann. Dabei sollen möglichst viele Schritte automatisiert werden.

Die Überlegungen müssen so allgemein sein, daß sie auch Änderungen des Datenbankzustandes während der Arbeit mit der föderierten Datenbank und den Komponentendatenbanken durch das Einfügen, Ändern und Löschen von Objekten berücksichtigen. Das bedeutet, daß nicht nur die Überlappungen betrachtet werden müssen, die zu einem bestimmten Zeitpunkt in den Komponentendatenbanken bzw. der föderierten Datenbank vorhanden sind, sondern auch solche, die möglicherweise während der Lebenszeit der Datenbanken auftreten können. Aus diesem Grund haben wir im Abschnitt 2.1 die Extension einer Objektklasse als Menge aller mögliche Objekte definiert.

Am Ende soll die Angabe eines Verfahrens zum Erkennen aller Basisextensionen und der Aufstellung der entsprechenden Tabelle unter Einbeziehung aller Informationen stehen. Aus diesen Basisextensionen werden dann die Klassen des Schemas der föderierten Datenbank erzeugt.

Mit den Begriffen aus dem letzten Kapitel können wir auch sagen, daß das Ziel unserer Untersuchungen das *Erkennen aller Basisklauseln* ist.

## 3.2 Voraussetzungen für die Untersuchung

Um die extensionalen Überlappungen und die Basisextensionen zu finden, müssen wir Informationen über die Komponentendatenbanken und die gesamte föderierte Datenbank haben. Wir stellen in diesem Abschnitt dar, welche Informationen uns für die Überlegungen zur Verfügung stehen.

Das sind einmal Aussagen über die Komponentendatenbanken:

- Aus den *Schemata* lassen sich Aussagen über Spezialisierungsbeziehungen und damit über die Teilmengenbeziehungen der Extensionen entnehmen. Diese Information ist wichtig, da sich Ober- und Unterklassen in jedem Fall extensional überlappen. Ebenso wichtig sind auch Aussagen über die Art der Spezialisierung (total und/oder exklusiv).
- Weitere Aussagen können dem *Datenmodell* entnommen werden. Lässt es keine Rollen oder Mehrfachspezialisierungen zu, können Überlappungen innerhalb der Komponentendatenbank nur zwischen Klassen auftreten, die in einer Spezialisierungsbeziehung stehen. Weiterhin kann man dem Datenmodell entnehmen, ob alle Klassen von einer gemeinsamen Oberklasse abgeleitet werden.
- Die *Objektidentifikatoren* sind innerhalb jeder Komponentendatenbank eindeutig. An ihnen kann man erkennen, ob ein Objekt in mehreren verschiedenen Klassen liegt.
- Um Aussagen zu erhalten, die nicht direkt aus dem Schema zu entnehmen sind, z. B. über die Art der Spezialisierung, müssen die *Datenbankadministratoren* befragt werden.

Weiterhin haben wir folgende Informationen über die föderierte Datenbank:

- Die *SAME-Beziehung* der föderierten Datenbank macht Aussagen darüber, welche Objekte der föderierten Datenbank dasselbe Realweltobjekt beschreiben. Ist die SAME-Beziehung bekannt, dann können mit ihrer Hilfe die Überlappungen im aktuellen Datenbankzustand ermittelt werden. Sie spielt eine ähnliche Rolle, wie die Objektidentifikatoren innerhalb der Komponentendatenbanken.
- Weiterhin müssen gerade hier die *Datenbankadministratoren* der Komponentendatenbanken befragt werden, um Aussagen über jene Überlappungen zu bekommen, die erst im Laufe der Arbeit mit der Datenbank auftreten können.

Mit Hilfe der Eindeutigkeit der *Objektidentifikatoren* bzw. der *SAME-Beziehung* kann man zwar alle Überlappungen im aktuellen Datenbankzustand finden. Es ist aber ohne zusätzliche Information nicht möglich, die Überlappungen zu bestimmen, die während der Arbeit mit der föderierten Datenbank auftreten können. Deshalb werden wir vor allem die Informationen aus dem Schema nutzen, um die Basisextensionen zu bestimmen. Trotzdem ist auch die Untersuchung des aktuellen Zustandes sinnvoll, denn hier gefundene Überlappungen werden sicher auch in Zukunft vorhanden sein werden.

### 3.3 Vorgehen bei der Untersuchung

Innerhalb der Komponentendatenbanken treten viele Überlappungen auf Grund von Spezialisierungsbeziehungen auf. Informationen darüber können direkt aus den Schemata der Komponentendatenbanken entnommen werden. Wir werden zuerst die Basisextensionen der Komponentendatenbanken auffinden. Dabei werden wir insbesondere zeigen, welchen Einfluß die Überlappung von Klassen auf weitere Überlappungen der jeweiligen Ober- und Unterklassen hat.

Wir werden sehen, daß wir die Methoden, die wir bei der Untersuchungen der Komponentendatenbanken benutzt haben, verallgemeinern und auch für das Finden von Überlappungen innerhalb der föderierten Datenbank anwenden können. Das betrifft insbesondere die Betrachtung des Einflusses von Rollen.

Sind die Tabellen der Basisextensionen für die Komponentendatenbanken bekannt, so kann die Tabelle der föderierten Datenbank daraus, unter Berücksichtigung weiterer Informationen, zusammengesetzt werden. Deshalb wird die Untersuchung der Komponentendatenbanken einen großen Teil der Arbeit ausmachen.

Wir werden ein Verfahren zur Bestimmung der Basisextensionen einer Komponentendatenbank angeben. Anschließend werden wir dieses Verfahren verallgemeinern und auf die föderierte Datenbank anwenden. Zum Schluß werden wir angeben, unter welchen Bedingungen mit diesem Verfahren die Basisextensionen eindeutig bestimmt werden können.

# Kapitel 4

## Ermittlung der Basisextensionen für eine Komponentendatenbank

In diesem Kapitel werden wir zeigen, wie die Basisextensionen innerhalb einer Komponentendatenbank ermittelt werden können. Dabei stehen uns zunächst nur die Informationen aus dem Schema zur Verfügung. Das sind Informationen über die Anordnung der Objektklassen in der Klassenhierarchie, Informationen über Ober- und Unterklassen sowie Informationen darüber, ob die Spezialisierungen total und/oder exklusiv sind. Falls einige dieser Aussagen nicht direkt dem Schema zu entnehmen sind, so muß an dieser Stelle schon auf den Datenbankadministrator und sein Wissen über die Datenbank zurückgegriffen werden.

Wie in Kapitel 2 dargestellt, werden wir die Basisextensionen durch *Schnittmengen der beteiligten Extensionen* beschreiben und für diese den Begriff *Klausel* verwenden. Die nicht-leeren Klauseln beschreiben genau die Basisextensionen. Wir müssen also die nicht-leeren Klauseln – die Basisklauseln – bestimmen. Aus diesen Basisklauseln kann dann die Tabelle der Basisextensionen abgeleitet werden. Wir werden aber sehen, daß nur mit den Aussagen die wir aus der Betrachtung des Schemas erhalten, die Basisextensionen im allgemeinen nicht eindeutig bestimmt werden können. Wir werden dann überlegen, welche weiteren Informationen benötigt werden, um Eindeutigkeit zu erreichen und wie diese Informationen gewonnen werden können.

Als Beispiel nutzen wir in diesem Abschnitt das Schema der Datenbank 2.2 auf Seite 13. Da wir in diesem Kapitel nur den Einfluß von Spezialisierungsbeziehungen behandeln werden, betrachten zunächst nur den Teil *A* mit den Klassen  $A_1$  bis  $A_8$ . Erst im nächsten Kapitel werden wir das Verfahren verallgemeinern und auch den Einfluß von Rollen und Mehrfachvererbung behandeln.

### 4.1 Ableitung der Basisextensionen

In dem zu betrachtenden Teil der Beispieldatenbank sind alle Klassen aus der gemeinsamen Oberklasse  $A_1$  spezialisiert worden. Deshalb wissen wir, daß alle Objekte in der Extension der Klasse  $A_1$  liegen. Diese bildet aber keine Basisextension. Da  $A_1$  Unterklassen hat, liegen einige Objekte aus  $A_1$  gleichzeitig noch in weiteren Extensionen.

Um die dadurch entstehenden Basisextensionen zu bestimmen, zerlegen wir als erstes die Extension  $A_1$  in Teilmengen. Diese Teilmengen müssen so gewählt werden, daß sie nicht-leer und paarweise disjunkt sind und die Vereinigung wieder  $A_1$  ergibt. Da die Spezialisierung von

$A_1$  nichttotal und nicht exklusiv ist, erhalten wir vier solche Teilmengen, die im folgenden aufgeführt sind.

- Die flache Extension von  $A_1$  ist nicht leer, da die Spezialisierung nicht total ist.

$$A_1 \supset A_1 \cap \overline{A_2} \cap \overline{A_3},$$

- Die folgenden beiden Mengen sind nicht leer, da die Extensionen der Unterklassen stets Teilmengen der Extension der Oberklassen sind.

$$A_1 \supset A_1 \cap A_2 \cap \overline{A_3}$$

$$A_1 \supset A_1 \cap \overline{A_2} \cap A_3$$

- Die folgende Menge ist nicht leer, da sich die Unterklassen wegen der nichtexklusiven Spezialisierung überlappen.

$$A_1 \supset A_1 \cap A_2 \cap A_3$$

Da  $A_2$  und  $A_3$  ebenfalls Unterklassen haben, sind damit aber immer noch nicht die Basisextensionen gefunden. Die Teilmengen müssen weiter zerlegt werden. Dazu schreiben wir auch für die Extensionen von  $A_2$  und  $A_3$  die Teilmengen auf.

Für  $A_2$  erhalten wir vier Teilmengen, da die Spezialisierung exklusiv ist. (Bei nichtexklusiver Spezialisierung würden auch noch alle möglichen Überlappungen der Unterklassen dazukommen.) Die Spezialisierung von  $A_3$  ist total aber nicht exklusiv. Deshalb gibt es nur drei Teilmengen.

$$1. A_2 \supset A_2 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6}$$

$$2. A_2 \supset A_2 \cap A_4 \cap \overline{A_5} \cap \overline{A_6}$$

$$3. A_2 \supset A_2 \cap \overline{A_4} \cap A_5 \cap \overline{A_6}$$

$$4. A_2 \supset A_2 \cap \overline{A_4} \cap \overline{A_5} \cap A_6$$

$$5. A_3 \supset A_3 \cap A_7 \cap \overline{A_8}$$

$$6. A_3 \supset A_3 \cap \overline{A_7} \cap A_8$$

$$7. A_3 \supset A_3 \cap A_7 \cap A_8$$

Wegen der Transitivität der Teilmengenbeziehung lassen sich in die Teilmengen von  $A_1$  für  $A_2$  und  $A_3$  die entsprechenden Mengen aus den Beziehungen 1 bis 7 einsetzen. Wir erhalten dann eine weitere Zerlegung von  $A_1$ . Durch die Ersetzung entstehen die folgenden Teilmengen:

1. Beim Einsetzen der Mengen aus 1., 2., 3. und 4. für  $A_2$  in  $A_1 \cap A_2 \cap \overline{A_3}$ .

$$(a) A_1 \supset A_1 \cap A_2 \cap \overline{A_3} \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6}) \cap \overline{A_3},$$

$$(b) A_1 \supset A_1 \cap A_2 \cap \overline{A_3} \supset A_1 \cap (A_2 \cap A_4 \cap \overline{A_5} \cap \overline{A_6}) \cap \overline{A_3},$$

$$(c) A_1 \supset A_1 \cap A_2 \cap \overline{A_3} \supset A_1 \cap (A_2 \cap \overline{A_4} \cap A_5 \cap \overline{A_6}) \cap \overline{A_3},$$

$$(d) A_1 \supset A_1 \cap A_2 \cap \overline{A_3} \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap A_6) \cap \overline{A_3}$$

2. Beim Einsetzen der Mengen aus 5, 6 und 7 für  $A_3$  in  $A_1 \cap \overline{A_2} \cap A_3$ .

$$(a) A_1 \supset A_1 \cap \overline{A_2} \cap A_3 \supset A_1 \cap \overline{A_2} \cap (A_3 \cap A_7 \cap \overline{A_8})$$

$$(b) A_1 \supset A_1 \cap \overline{A_2} \cap A_3 \supset A_1 \cap \overline{A_2} \cap (A_3 \cap \overline{A_7} \cap A_8)$$

$$(c) A_1 \supset A_1 \cap \overline{A_2} \cap A_3 \supset A_1 \cap \overline{A_2} \cap (A_3 \cap A_7 \cap A_8)$$

3. In die Beziehung  $A_1 \cap A_2 \cap A_3$  können sowohl für  $A_2$  als auch für  $A_3$  die entsprechenden Mengen eingesetzt werden. Dabei ist zunächst jede Kombination möglich. Auf die Frage, ob die dabei entstehenden Mengen wirklich nicht leer sind, gehen wir im nächsten Abschnitt ein.

$$(a) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap \overline{A_8})$$

$$(b) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap \overline{A_7} \cap A_8)$$

$$(c) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap A_8)$$

$$(d) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap A_4 \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap \overline{A_8})$$

$$(e) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap A_4 \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap \overline{A_7} \cap A_8)$$

$$(f) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap A_4 \cap \overline{A_5} \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap A_8)$$

$$(g) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap A_5 \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap \overline{A_8})$$

$$(h) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap A_5 \cap \overline{A_6}) \cap (A_3 \cap \overline{A_7} \cap A_8)$$

$$(i) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap A_5 \cap \overline{A_6}) \cap (A_3 \cap A_7 \cap A_8)$$

$$(j) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap A_6) \cap (A_3 \cap A_7 \cap \overline{A_8})$$

$$(k) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap A_6) \cap (A_3 \cap \overline{A_7} \cap A_8)$$

$$(l) A_1 \supset A_1 \cap A_2 \cap A_3 \supset A_1 \cap (A_2 \cap \overline{A_4} \cap \overline{A_5} \cap A_6) \cap (A_3 \cap A_7 \cap A_8)$$

4. In die Beziehung  $A_1 \supset A_1 \cap \overline{A_2} \cap \overline{A_3}$  kann nichts eingesetzt werden, da nur die Komplemente der Extensionen von  $A_2$  und  $A_3$  auftreten.

Damit sind alle möglichen Ersetzungen erfolgt, da man für die Klassen  $A_4, A_5, A_6, A_7$  und  $A_8$  nur die Extensionen selbst als Teilmengen angeben kann. Bei einer Ersetzung erhält man dann keine neuen Mengen mehr. Die so entstandenen 20 Teilmengen von  $A_1$  sind paarweise disjunkt und die Vereinigung ergibt wieder die Extension  $A_1$ .

Um aus diesen Klauseln zu bilden, müssen wir noch die Komplemente der jeweils nicht angegebenen Klassen ergänzen. Allerdings haben wir auf diesem Weg auch Klauseln erzeugt, die nicht mit Sicherheit Basisextensionen beschreiben, d. h. durch zusätzliche Informationen könnten wir erfahren, daß einige der Klauseln doch leer sind. Das betrifft die Klauseln im Punkt 3, da hier zwar wegen der nichtexklusiven Spezialisierung eine Überlappung von  $A_2$  und  $A_3$  entsteht, es aber nicht sicher ist, ob Objekte aus der Überlappung in allen Teilmengen von  $A_2$  und  $A_3$  liegen.

Bevor wir die Tabelle der Basisextensionen aus diesen Klauseln ableiten, werden wir deshalb im nächsten Abschnitt den Einfluß weiterer Informationen untersuchen.

## 4.2 Eindeutigkeit des Verfahrens

Wie im letzten Abschnitt erwähnt, muß das Verfahren der Bestimmung von Klauseln durch Mengenersetzung nicht in jedem Fall auf einen Basisextension führen. Wir betrachten noch einmal die Teilmengen von  $A_1$  auf Seite 21.

Die erste Menge beschreibt die flache Extension von  $A_1$ , die auf Grund der nichttotalen Spezialisierung nicht leer ist. Daran kann sich auch durch zusätzliche Informationen über die Verteilung der Objekte nichts ändern. Wir könnten höchstens erfahren, daß die Spezialisierung doch nicht total ist, dann würde aber das Schema der Datenbank selbst geändert. Die flache Extension ist auch an keiner Überlappung der Unterklassen von  $A_1$  beteiligt, so daß weitere Teilmengen nicht beeinflusst werden, selbst wenn die flache Extension leer wird.

Die beiden folgenden Mengen beschreiben die Extensionen von  $A_2$  bzw.  $A_3$ . Diese könnten nur leer werden, wenn die Extensionen keine Objekte enthielten. Das wird aber sicher nie geschehen. Auch weitere Teilmengen, die durch Ersetzung von Extensionen der Unterklassen entstehen, sind nie leer.

Bei den ersten drei Mengen kann eine zusätzliche Information über die Verteilung der Objekte also nicht dazu führen, daß die entstandenen Klauseln leer werden,

Anders ist das bei der vierten Menge. Sie enthält die extensionale Überlappung der Unterklassen auf Grund nichtexklusiver Spezialisierung. Bei Klauseln, die durch Einsetzen in die Menge  $A_1 \cap A_2 \cap A_3$  entstanden sind, können zusätzliche Informationen dazu führen, daß diese Klauseln leer werden. Das kann geschehen, wenn die Objekte des Überlappungsgebietes  $A_2 \cap A_3$ , das durch nichtexklusive Spezialisierung entsteht, nicht auf alle weiteren Unterklassen verteilt sind.

Die Informationen über diese Verteilung kann man aber normalerweise nicht aus dem Schema der Datenbank entnehmen. In Kapitel 3 haben wir aber noch andere Informationsquellen erwähnt. Diese können wir jetzt nutzen, um die Verteilung der Objekte der Überlappung zu erfahren. Die Informationen können direkt vom Datenbankadministrator erfragt werden. Es können aber auch zuerst durch eine Betrachtung des aktuellen Datenbankzustandes mit Hilfe der Objektidentifikatoren die derzeitigen Überlappungen erkannt werden. Anschließend müssen nur die Überlappungen, für die so keine Entscheidung getroffen werden kann, dem Administrator vorgelegt werden.

Um ein schnelles Anwachsen der Klauselmenge zu vermeiden, sollten die Informationen über die Verteilung der Objekte des Überlappungsgebiets vor der Behandlung weiterer Unterklassen gesammelt werden.

## 4.3 Angeben der Basisextensionen

Wenn wir Informationen über die Verteilung der Objekte aus der Überlappung bei nichtexklusiver Spezialisierung haben, können einige der Klauseln aus Punkt 3 leer werden. Nehmen wir an, daß wir erfahren, daß in der Beispieldatenbank die Objekte der Überlappung von  $A_2$  und  $A_3$  nur in den Unterklassen  $A_4$ ,  $A_7$  sowie in der flachen Extension von  $A_2$  liegen. Dann werden alle Klauseln leer, die den Schnitt  $A_2 \cap A_3$  und die Extensionen  $A_5$ ,  $A_6$  oder  $A_8$  enthalten. Es bleiben nur die zwei Basisklauseln 3a und 3d übrig. Durch zusätzliche Informationen über die Verteilung der Objekte kann die Menge der Klauseln also in diesem Fall erheblich reduziert werden.

Zum Schluß dieses Abschnittes wollen wir die erkannten Basisklauseln noch einmal auflisten und die daraus abgeleitete Tabelle der Basisextensionen angeben.

1.  $A_1 \cap A_2 \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8}$
2.  $A_1 \cap A_2 \cap \overline{A_3} \cap A_4 \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8}$
3.  $A_1 \cap A_2 \cap \overline{A_3} \cap \overline{A_4} \cap A_5 \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8}$
4.  $A_1 \cap A_2 \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap A_6 \cap \overline{A_7} \cap \overline{A_8}$
5.  $A_1 \cap \overline{A_2} \cap A_3 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap A_7 \cap \overline{A_8}$
6.  $A_1 \cap \overline{A_2} \cap A_3 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap A_8$
7.  $A_1 \cap \overline{A_2} \cap A_3 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap A_7 \cap A_8$
8.  $A_1 \cap A_2 \cap A_3 \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap A_7 \cap \overline{A_8}$
9.  $A_1 \cap A_2 \cap A_3 \cap A_4 \cap \overline{A_5} \cap \overline{A_6} \cap A_7 \cap \overline{A_8}$
10.  $A_1 \cap \overline{A_2} \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8}$

Die Tabelle 4.1 der Basisextensionen entsteht, indem für jede Klasse, die nicht als Komplement in der Klausel steht eine Markierung gesetzt wird. Sie hat dann 10 Spalten. Damit haben wir alle Basisextensionen in der Beispieldatenbank gefunden.

Klasse	Basisextension									
	1	2	3	4	5	6	7	8	9	10
$A_1$	x	x	x	x	x	x	x	x	x	x
$A_2$	x	x	x	x				x	x	
$A_3$					x	x	x	x	x	
$A_4$		x							x	
$A_5$			x							
$A_6$				x						
$A_7$					x		x	x	x	
$A_8$						x	x			x

Tabelle 4.1: Basisextensionen der Beispieldatenbank  $A$

## 4.4 Zusammenfassung

Mit den Ausführungen im letzten Abschnitt können wir behaupten, daß das vorgestellte Verfahren zur Ermittlung der Basisextensionen nur dann eindeutig zur Bestimmung der Basisextensionen führt, wenn keine nichtexklusiven Spezialisierungen zugelassen sind. Entstehen Überlappungen durch nichtexklusive Spezialisierungen, so muß mit Hilfe weiterer Informationen ermittelt werden, wie die Objekte aus der Überlappung verteilt sind.

Zusätzlich müssen wir noch fordern, daß keine der Objektklassen eine leere Extension haben darf. Da die Extension aber als Menge aller möglichen Objekte definiert ist und es

sicher keine während der gesamten Lebenszeit der Datenbank leere Klasse geben wird, ist diese Forderung normalerweise immer erfüllt.

Die Anzahl der entstehenden Basisextensionen ist abhängig von der Art der Spezialisierung. Sind keine nichtexklusiven Spezialisierungen zugelassen, so setzt sich die Anzahl der Basisextensionen aus der Anzahl der Klassen, die keine weiteren Unterklassen mehr haben, und der Anzahl der Klassen, die nichttotal spezialisiert sind, zusammen.

Sind nichtexklusive Spezialisierungen zugelassen, so entstehen jeweils  $2^n - 1$  mögliche Überlappungsgebiete zwischen  $n$  Unterklassen. Es muß aber durch zusätzliche Informationen geklärt werden, ob alle diese Überlappungen wirklich nicht leer sind. Diese Entscheidung sollte getroffen werden, bevor weitere Unterklassen betrachtet werden, damit es nicht zu einem exponentiellen Anwachsen der Klauselmenge kommt. Anschließend muß die Frage geklärt werden, in welchen Unterklassen der spezialisierten Klassen Objekte der Überlappung liegen.

# Kapitel 5

## Verallgemeinerung des Verfahrens

Im letzten Kapitel haben wir gezeigt, wie die Basisextensionen für eine Datenbank ermittelt werden können, in der es nur eine allgemeine Klasse ohne weitere Oberklassen gibt. Da manche Datenmodelle für objektorientierte Datenbanken aber mehrere solcher Klassen zulassen, betrachten wir in der Beispieldatenbank 2.2 nun die Teile  $A$  und  $B$  gemeinsam.

In diesem Kapitel werden wir zunächst zeigen, wie die Tabelle der Basisextensionen unter der Voraussetzung, daß es keine Überlappungen zwischen Klassen der beiden Teile gibt, erstellt werden kann. Anschließend untersuchen wir, welchen Einfluß es auf die Basisextensionen hat, wenn Rollen zugelassen werden.

Die in diesem Kapitel vorgestellten Vorgehensweisen gelten gleichermaßen für die Untersuchung der extensionalen Überlappungen zwischen Klassen unterschiedlicher Datenbanken. Wenn die beiden Teile nicht zu einer einzigen Datenbank, sondern zu verschiedenen Komponentendatenbanken gehörten, so wären dieselben Überlegungen nötig, um die Basisextensionen zu finden.

### 5.1 Mehrere Oberklassen

Zunächst setzen wir voraus, daß es keine Überlappungen zwischen Klassen der verschiedenen Teile  $A$  und  $B$  gibt. Anschließend untersuchen wir, welchen Einfluß die Rollenbeziehung zwischen  $A_3$  und  $B_2$  hat.

Um die Tabelle der Basisextensionen für beide Teile zu ermitteln, können wir auf den bereits behandelten Fall zurückgreifen, indem wir eine zusätzliche Klasse konstruieren, deren Extension alle Objekte aus  $A_1$  und  $B_1$  enthält. Die Spezialisierung dieser Klasse in  $A_1$  und  $B_1$  wäre dann total und, wenn wir zunächst davon ausgehen, daß Rollen nicht zugelassen sind, auch exklusiv. Diese zusätzliche Klassenextension  $A_1 \cup B_1$  hat dann zwei Teilmengen:

1.  $A_1 \cup B_1 \supset A_1 \cap \overline{B_1}$  und
2.  $A_1 \cup B_1 \supset \overline{A_1} \cap B_1$ .

Die Ersetzung von  $A_1$  und  $B_1$  durch weitere Teilmengen und die Erstellung der Tabelle der Basisextensionen läuft dann genauso ab, wie im letzten Kapitel beschrieben. Der wesentliche Unterschied ist, daß im Gegensatz zur Zerlegung einer echten Oberklasse die gedachte Klasse  $A_1 \cup B_2$  nicht selbst in den Klauseln auftritt.

Wir werden die Klauseln hier nicht mehr im einzelnen angeben, sondern das Vorgehen nur andeuten. Die Ersetzung von  $A_1$  führt z. B. auf

$$A_1 \cup B_1 \supset A_1 \cap \overline{B_1} \supset A_1 \cap \overline{A_2} \cap \overline{A_3} \cap \overline{B_1}$$

und nach Ergänzung der Komplemente auf die Klausel

$$A_1 \cap \overline{A_2} \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8} \cap \overline{B_1} \cap \overline{B_2} \cap \overline{B_3} \cap \overline{B_4} \cap \overline{B_5}$$

Auf diese Art entstehen aus der Teilmenge  $A_1 \cap \overline{B_1}$  dieselben Klauseln, die wir schon im letzten Kapitel gefunden haben, denn das Ergänzen der Komplemente der Extensionen  $B_i$  ändert nichts an der durch die Klausel beschriebenen Menge.

Für  $B$  erhält man durch Einsetzen der entsprechenden Teilmengen in  $\overline{A_1} \cap B_1$  und Ergänzen der Komplemente die folgenden drei Klauseln:

1.  $\overline{A_1} \cap \overline{A_2} \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8} \cap B_1 \cap B_2 \cap \overline{B_3} \cap B_4 \cap \overline{B_5}$
2.  $\overline{A_1} \cap \overline{A_2} \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8} \cap B_1 \cap B_2 \cap \overline{B_3} \cap \overline{B_4} \cap B_5$
3.  $\overline{A_1} \cap \overline{A_2} \cap \overline{A_3} \cap \overline{A_4} \cap \overline{A_5} \cap \overline{A_6} \cap \overline{A_7} \cap \overline{A_8} \cap B_1 \cap \overline{B_2} \cap B_3 \cap \overline{B_4} \cap \overline{B_5}$

Nichtexklusive Spezialisierung tritt in diesem Teil des Schemas nicht auf, so daß die Basis-klauseln für  $B$  eindeutig bestimmt sind.

Damit kann die Tabelle der Basisextensionen erstellt werden. Sie ist in 5.1 dargestellt. Kennt man allerdings die Tabellen der beiden Teile  $A$  und  $B$ , so kann man die Tabelle der Basisextensionen auch gleich aus diesen zusammensetzen. Es entspricht der linke obere Teil der Tabelle genau der Tabelle 4.1 der Basisextensionen von  $A$  und der rechte untere Teil der Tabelle der Basisextensionen, die man für  $B$  erhalten würde.

Die Anzahl der Basisextensionen für die gesamte Datenbank ist die Summe der Anzahl der Basisextensionen für jeden Teil. Für die Beispieldatenbank erhalten wir damit 13 Basis-extensionen.

## 5.2 Rollenbeziehungen

Da wir in 2.5 für die Beispieldatenbank Rollen zwischen den Klassen  $A_3$  und  $B_2$  zugelassen haben, überlappen sich die Klassen  $A_3$  und  $B_2$  extensional. Deshalb sind in der Tabelle 5.1 noch nicht alle Basisextensionen erfaßt. Auch dieser Fall läßt sich durch das Einfügen einer zusätzlichen gedachten Klasse auf das Vorgehen im Kapitel 4 zurückführen.

Da wir die Klauseln für  $A$  und  $B$  schon kennen und nur die zusätzlich entstehenden Klauseln finden wollen, ist es nicht sinnvoll diese Klasse wieder als gemeinsame Oberklasse von  $A_1$  und  $B_1$  einzufügen.

Da alle Spezialisierungen in  $B$  exklusiv sind, können Objekte der Überlappung nicht in  $B_3$  liegen. Die Spezialisierung der Oberklasse von  $A_3$  ist aber nicht exklusiv. Deshalb ist es auch möglich, daß alle weiteren Unterklassen von  $A_1$  Objekte aus dem Überlappungsgebiet enthalten. Um dieses zu berücksichtigen, wählen wir die gedachte Klasse  $A_1 \cup B_2$  als Ausgangspunkt der Überlegungen. Die Spezialisierung aus dieser angenommenen Oberklasse ist total aber nicht exklusiv. Diese Überlegungen veranschaulicht die Abbildung 5.1. Es gibt hier zwei Klassen ohne weitere Oberklassen, nämlich  $B_1$  und die gedachte Klasse  $A_1 \cup B_2$ .

Das Vorgehen bei der Zerlegung von  $B_1$  ist schon im letzten Abschnitt beschrieben worden, und die dabei entstehenden Basisklauseln sind auf Seite 27 angegeben.

Klasse	Basisextension												
	A										B		
	1	2	3	4	5	6	7	8	9	10	11	12	13
$A_1$	x	x	x	x	x	x	x	x	x	x			
$A_2$	x	x	x	x				x	x				
$A_3$					x	x	x	x	x				
$A_4$		x							x				
$A_5$			x										
$A_6$				x									
$A_7$					x		x	x	x				
$A_8$						x	x						
$B_1$											x	x	x
$B_2$											x	x	
$B_3$													x
$B_4$											x		
$B_5$												x	

Tabelle 5.1: Basisextensionen für  $A$  und  $B$

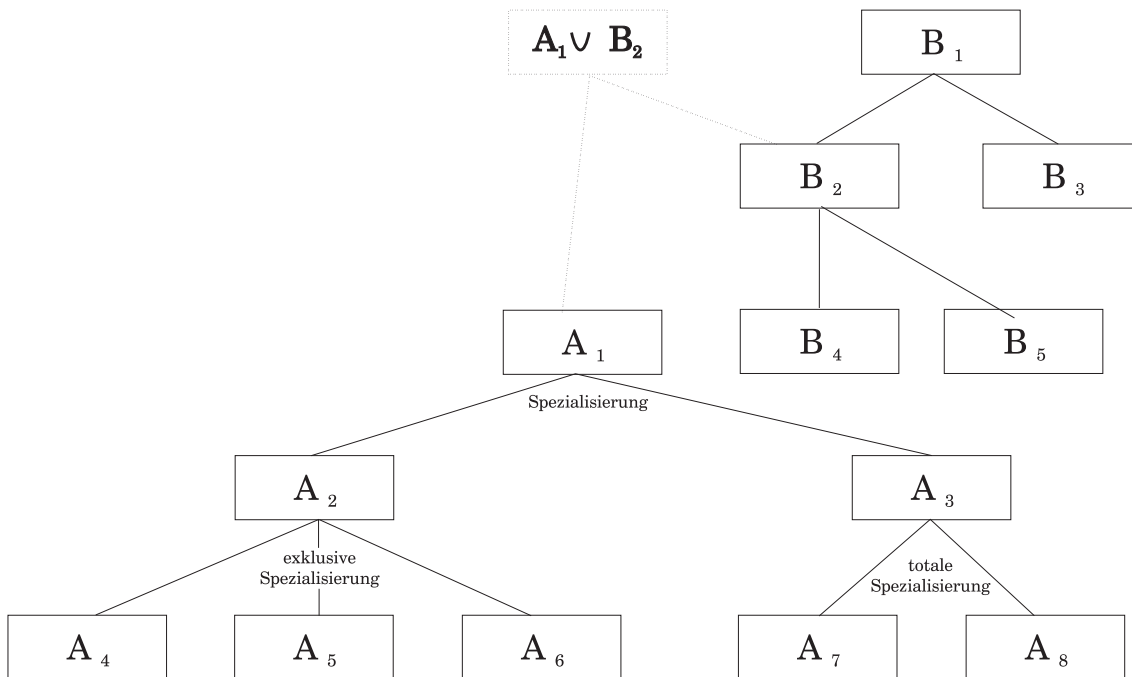


Abbildung 5.1: Überlappung von Klassen zweier Hierarchien

Interessant sind hier aber die aus der Zerlegung der gedachten Klasse  $A_1 \cup B_2$  entstehenden Klauseln. Wir schreiben zunächst alle Teilmengen auf. Für  $B_2$  können wir hier auch  $B_1 \cap B_2$  schreiben, da  $B_1$  Oberklasse von  $B_2$  ist.

1.  $A_1 \cup B_2 \supset A_1 \cap \overline{B_2}$
2.  $A_1 \cup B_2 \supset \overline{A_1} \cap B_1 \cap B_2$
3.  $A_1 \cup B_2 \supset A_1 \cap (B_1 \cap B_2)$

Hier ist nur die letzte Menge von Interesse, da die beiden anderen Teilmengen schon zerlegt worden und die Basisextensionen in der Tabelle 5.1 angegeben sind.

Da die Teilmengen von  $A_1$  genau die in Tabelle 4.1 gefundenen Basisextensionen sind und es für  $B_2$  nur die beiden Teilmengen

1.  $B_1 \cap B_2 \supset B_1 \cap B_2 \cap B_4$  und
2.  $B_1 \cap B_2 \supset B_1 \cap B_2 \cap B_5$

gibt, entstehen durch die Kombination der 10 Teilmengen von  $A_1$  und der zwei Teilmengen von  $B_2$  20 möglicherweise nichtleere Teilmengen von  $A_1 \cap B_1 \cap B_2$ . Um die Klauseln zu vervollständigen, müssen jeweils noch die Komplemente der restlichen Extensionen ergänzt werden.

Es ist hier wieder die Frage zu stellen, wie die Objekte der Überlappung  $A_1 \cap B_2$  auf die Unterklassen verteilt sind. Geht man von einer gleichmäßigen Verteilung aus, so kann die Menge der möglicherweise nichtleeren Klauseln schnell wachsen. Deshalb sollten Klassen, die keine Objekte der Überlappung enthalten, möglichst ausgeschlossen werden.

Zusätzliche Informationen über die Verteilung der Objekte aus der Überlappung kann man wieder durch die Untersuchung der Objektidentifikatoren und die Befragung des Administrators erhalten.

Für das Beispiel nehmen wir an, daß wir erfahren haben, daß sich Objekte der Überlappung nicht in der Klasse  $B_5$  und, trotz der nichtexklusiven Spezialisierung, auch nicht in  $A_2$  befinden können. Damit müssen bei der Kombination der Klauseln nur 5., 6., 7. und 10. auf Seite 24 und 1. auf Seite 29 berücksichtigt werden. Statt 20 möglicher zusätzlicher Klauseln erhalten wir nur vier, von denen wir jetzt sicher wissen, daß sie nicht leer sind.

Für die Teile  $A$  und  $B$  des Schemas der Beispieldatenbank gibt es also insgesamt 17 Basisextensionen, das sind

1. 10 aus Teil  $A$  und
2. 3 aus Teil  $B$ , beides angegeben in Tabelle 5.1, sowie
3. 4 aus der Überlappung beider Teile unter Einbeziehung zusätzlicher Information.

Die gesamte Tabelle ist noch einmal in 5.2 dargestellt.

Klasse	Basisextension																
	A										B			A $\cap$ B			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
$A_1$	x	x	x	x	x	x	x	x	x	x				x	x	x	x
$A_2$	x	x	x	x													
$A_3$					x	x	x	x	x				x	x	x		
$A_4$		x							x								
$A_5$			x														
$A_6$				x													
$A_7$					x		x	x	x				x			x	
$A_8$						x	x							x	x		
$B_1$										x	x	x	x	x	x	x	
$B_2$										x	x		x	x	x	x	
$B_3$												x					
$B_4$										x			x	x	x	x	
$B_5$											x						

Tabelle 5.2: Basisextensionen bei Rollenbeziehungen

## 5.3 Zusammensetzen der Tabellen

Wenn die Tabellen der Basisextensionen der beiden Teile und die zusätzlichen Informationen über die Verteilung der Objekte bekannt sind, so kann auch die Tabelle 5.2 aus diesen Einzeltabellen zusammengesetzt werden.

Im Beispiel muß man dazu nur die Basisextensionen in der Tabelle 5.1 für  $A$ , die nicht die Extension  $A_2$  enthalten, und die Basisextensionen für  $B$ , die nicht  $B_5$  enthalten, miteinander kombinieren.

Damit ergibt sich im Fall, daß die Tabellen der Basisextensionen der beteiligten Hierarchien bereits ermittelt sind, die Möglichkeit, die durch Rollen entstehenden Basisklauseln direkt aus diesen Tabellen abzuleiten.

## 5.4 Spezialfälle für die Überlappung

Im letzten Abschnitt haben wir die Überlappung  $A_3 \cap B_2$  behandelt.

Wir sind bei den Überlegungen davon ausgegangen, daß die Überlappung partiell ist, d. h. es gilt  $A_3 \cap B_2 \neq \emptyset$ , aber nicht  $A_3 \subseteq B_2$  oder  $B_2 \subseteq A_3$ . Den Fall, daß die Überlappung auf eine Teilmengenbeziehung oder Gleichheit der Klassenextensionen führt, wollen wir jetzt behandeln.

Nehmen wir an, daß  $A_3 \subseteq B_2$  gilt. Dann ist  $A_3 \cup B_2 = B_2$ . Da wir wegen der nichtexklusiven Spezialisierung alle weiteren Unterklassen von Klasse  $A_1$  betrachten müssen, ändert diese Beziehung nichts am Ergebnis.

Interessanter ist der Fall  $B_2 \subseteq A_3$  mit  $B_2 \cup A_3 = A_3$ . Anstatt eine neue Klasse  $A_1 \cup B_2$  zu schaffen, kann dann auch  $B_2$  als eine weitere Unterklasse von  $A_3$  aufgefaßt werden. Alle Klauseln, die  $B_2$  enthalten, müssen dann auch  $A_3$  (und  $A_1$ ) enthalten. In der Tabelle 5.2 müßten die Basisextensionen 11 und 12 entsprechend geändert werden, so daß sie auch  $A_1$  und  $A_3$  enthalten.

Im allgemeinen Fall führt eine gefundene Teilmengenbeziehung oder Gleichheit zwischen Klassenextensionen zu einer Veränderung der Menge der bereits erkannten Basisextensionen.

## 5.5 Mehrfachspezialisierung

Mehrfachspezialisierung bedeutet, wie in 2.8 beschrieben, daß eine Klasse mehrere direkte Oberklassen hat. Für die Extensionen im Teil  $C$  der Beispieldatenbank, in dem Mehrfachspezialisierung auftritt, erhalten wir die folgenden Teilmengen:

1.  $C_1 \supset C_1 \cap C_2 \cap \overline{C_3}$
2.  $C_1 \supset C_1 \cap \overline{C_2} \cap C_3$
3.  $C_1 \supset C_1 \cap C_2 \cap C_3 = C_1 \cap C_2 \cap C_3 \cap C_5$
4.  $C_2 \supset C_2 \cap C_4 \cap \overline{C_5}$
5.  $C_2 \supset C_2 \cap \overline{C_4} \cap C_5$
6.  $C_3 \supset C_3 \cap \overline{C_5}$
7.  $C_3 \supset C_3 \cap C_5$

$$8. C_5 \supset C_5 \cap \overline{C_6}$$

$$9. C_5 \supset C_5 \cap C_6$$

Die Extensionen  $C_2$  und  $C_3$  überlappen sich zwar, alle gemeinsamen Objekte liegen aber in der Klasse  $C_5$ . Bei der Ersetzung in der Teilmenge 3 müssen deshalb nicht alle möglichen Kombinationen von  $C_2$  und  $C_3$  berücksichtigt werden, sondern es darf nur in die Menge  $C_1 \cap C_2 \cap C_3 \cap C_5$  eingesetzt werden. Aus dieser Ersetzung entstehen die Klauseln:

$$1. C_1 \cap C_2 \cap C_3 \cap \overline{C_4} \cap C_5 \cap \overline{C_6} \quad \text{und}$$

$$2. C_1 \cap C_2 \cap C_3 \cap \overline{C_4} \cap C_5 \cap C_6.$$

Alle anderen Ersetzungen laufen wie in Kapitel 4 beschrieben.

Durch das Erkennen von Mehrfachspezialisierung erhält man also eine zusätzliche Information über die extensionalen Überlappungen. Da hier keine nichtexklusiven Spezialisierungen zugelassen sind, sind die Basisextensionen trotz der Überlappung der Klassen  $C_2$  und  $C_3$  eindeutig bestimmbar.

# Kapitel 6

## Zusammenfassung

In Kapitel 4 haben wir ein Verfahren für die Bestimmung von Basisextensionen innerhalb einer Komponentendatenbank beschrieben. Dieses haben wir in Kapitel 5 so erweitert, daß es auch für die Ermittlung der Basisextensionen einer föderierten Datenbank angewendet werden kann.

Klassen unterschiedlicher Datenbanken können sich überlappen, wenn sie Datenbankobjekte enthalten, die die gleichen Realweltobjekte beschreiben. Diese Überlappungen können erkannt werden, indem die SAME-Beziehung der föderierten Datenbank untersucht wird oder die Administratoren der Komponentendatenbanken befragt werden.

Aufgefundene Überlappungen von Klassen können weitere Überlappungen von Unterklassen oder Oberklassen der betreffenden Klassen zur Folge haben.

### 6.1 Bestimmung der Basisextensionen der Komponentendatenbanken

Wir werden hier das Vorgehen zur Erstellung der Tabelle der Basisextensionen für eine Komponentendatenbank noch einmal zusammenfassen.

In Kapitel 4 sind wir zunächst davon ausgegangen, daß es nur *eine allgemeine Klasse* gibt, von der alle anderen Klassen des Schemas spezialisiert werden. Dann sind alle Basisextensionen Teilmengen der Extension dieser Klasse.

Um die Basisextensionen zu bestimmen, haben wir alle Klassenextensionen unter Ausnutzung der *Informationen über die Spezialisierungsbeziehungen* in disjunkte Teilmengen zerlegt. Diese Teilmengen haben wir schrittweise in die Extension der allgemeinen Klasse eingesetzt und diese dadurch immer weiter zerlegt. Nachdem alle möglichen Ersetzungen durchgeführt waren, erhielten wir eine Menge von disjunkten Teilmengen dieser Extension, die wir Klauseln genannt haben. Wir haben festgestellt, daß, wenn alle Spezialisierungen in der Datenbank exklusiv sind, diese Klauseln genau die Basisextensionen der Datenbank bilden. Für die Eindeutigkeit des Verfahrens gilt dann folgendes:

Ist nur exklusive Spezialisierung zugelassen, so sind die Basisextensionen einer Komponentendatenbank eindeutig aus den Spezialisierungsbeziehungen ableitbar.

Tritt jedoch *nichtexklusive Spezialisierung* auf, so überlappen sich zwei oder mehr Klassen, die nicht in einer direkten Spezialisierungsbeziehung stehen. Die Objekte des entstehenden

Überlappungsgebietes müssen aber nicht in allen weiteren Unterklassen der spezialisierten Klassen liegen. Deshalb können genauere *Informationen über die Verteilung der Objekte* dazu führen, daß einige der Klauseln als leer erkannt werden.

Ein anderes Problem bei der Betrachtung nichtexklusiver Spezialisierung von mehr als zwei Klassen ist, daß sich die Unterklassen zwar überlappen, diese Überlappung aber nicht vollständig sein muß. Es gilt deshalb:

Ist nichtexklusive Spezialisierung zugelassen, so können nur mit der Schema-information die Basisextensionen *nicht eindeutig bestimmt* werden. Es müssen zusätzlich zu diesen auch Informationen über die Verteilung der Objekte aus dem Überlappungsgebiet und, wenn in mehr als zwei Klassen spezialisiert wird, auch Informationen über die Art der Überlappung vorhanden sein.

Diese Informationen kann man erhalten, wenn man den Datenbankadministrator befragt. Eine andere Informationsquelle ist die Untersuchung der Objektidentifikatoren.

Werden diese Informationen gleich bei der Bestimmung der Teilmengen der Klassenextensionen genutzt, so treten bei der Ersetzung keinen leeren Mengen auf. Leere Klauseln werden gar nicht erst erzeugt. Eine andere Möglichkeit ist, zuerst alle Klauseln abzuleiten und anschließend die leeren Klauseln zu verwerfen.

Welcher Weg gewählt werden sollte, hängt sicher von der Anzahl der Klassen im Schema ab. Bei sehr vielen Klassen kann die Zahl der Klauseln schnell wachsen. Besonders wichtig ist eine frühe Entscheidung bei nichtexklusiven Spezialisierungen in mehr als zwei Unterklassen, da die Anzahl der möglichen Überlappungen hier exponentiell mit der Anzahl der Unterklassen wächst.

In Kapitel 5 haben wir das Verfahren auf Datenbanken mit *mehr als einer allgemeinen Klasse* verallgemeinert. Wir haben hier eine Klasse konstruiert, die alle Objekte der Datenbank enthält. Diese bildet dann eine gedachte Oberklasse aller Klassen und dient als Ausgangspunkt für die Ersetzungen.

Es ist hier aber auch möglich, zuerst die Tabellen für die einzelnen Teile des Schemas zu erstellen und diese anschließend zu einer Gesamttabelle zusammensetzen.

Eine besondere Bedeutung hat die *Behandlung von Rollen zwischen beliebigen Klassen*. Dadurch entstehen Basisextensionen, die durch das bisher beschriebene Verfahren noch nicht bestimmt worden sind. Dieser Fall kann aber durch das Einfügen einer gedachten Klasse auf die Behandlung von nichtexklusiver Spezialisierung zurückgeführt werden. Wo diese gedachte Klasse eingefügt wird, hängt von der Art der Spezialisierungsbeziehungen weiterer Oberklassen ab. Aus der gedachten Klasse können dann die Klauseln, die durch die Überlappung entstehen, hergeleitet werden. Auch hier gibt es das Problem, daß die Objekte des Überlappungsgebietes eventuell nicht in allen Unterklassen liegen. Deshalb müssen ebenfalls zusätzliche Information über die Verteilung der Objekte herangezogen werden. Überlappen sich mehr als zwei Klassen, so muß auch wieder die Art der Überlappung betrachtet werden.

Eine andere Möglichkeit, mehrere allgemeine Klassen zu behandeln, ist die Erstellung der Tabellen für jede dieser Klassen und das anschließende Zusammensetzen zu einer Gesamttabelle unter Beachtung der Informationen über die Überlappungen. Für die Eindeutigkeit des Verfahrens gilt:

Wenn *Rollen zwischen beliebigen Klassen* zugelassen sind, so können die Basisextensionen nur eindeutig bestimmt werden, wenn zusätzlich zur den Informationen, die man direkt aus dem Schema erhalten kann, auch Informationen über die Verteilung der Objekte bekannt sind.

Die Quellen dieser Informationen sind dieselben, wie bei der nichtexklusiven Spezialisierung.

## 6.2 Bestimmung der Basisextensionen der föderierten Datenbank

Das im letzten Abschnitt beschriebene Verfahren kann auch zu Bestimmung der Basisextensionen einer föderierten Datenbank benutzt werden. Nachdem durch die Untersuchung der SAME-Beziehung und der Befragung der Administratoren Informationen darüber gewonnen worden sind, welche Klassen unterschiedlicher Datenbanken sich überlappen, kann diese Überlappung wie eine Rollenbeziehung behandelt werden.

Es gilt auch hier, daß die Basisextensionen nur dann eindeutig bestimmbar sind, wenn außer den Aussagen über die Klassenüberlappungen auch Informationen über die Verteilung der Objekte aus diesen Überlappungen zur Verfügung stehen.

Diese zusätzlichen Informationen kann man aus der Betrachtung der SAME-Beziehung der föderierten Datenbank und von den Administratoren der Komponentendatenbanken erhalten.

## 6.3 Eigenschaften des Verfahrens

Wir wollen noch einmal auf die Frage eingehen, ob mit dem beschriebenen Verfahren, die Basisextensionen in jedem Fall eindeutig bestimmt werden können. Diese Frage kann nur in Abhängigkeit davon beantwortet werden, welche Informationen zur Verfügung stehen.

Wenn wir sowohl vollständige Informationen aus dem Schema, als auch für jede Überlappung von Objektklassen vollständige Angaben darüber haben, auf welche Art sich die Klassen überlappen und wie die Objekte aus der Überlappung auf weitere Unterklassen verteilt sind, dann können die Basisextensionen eindeutig bestimmt werden.

Während die Informationen aus dem Schema normalerweise vollständig zur Verfügung stehen, ist die Information über die Art der Überlappung und die Verteilung der Objekte nur in Zusammenarbeit mit den Administratoren zu erfahren.

Da hier alle künftigen Datenbankzustände berücksichtigt werden müssen, kann es auch möglich sein, daß diese Informationen nicht vollständig beschafft werden können. Dann ist es nur möglich festzustellen, welche der Klauseln mit Sicherheit Basisextensionen beschreiben, welche mit Sicherheit keine Basisextensionen beschreiben und als drittes die Klauseln anzugeben, für die eine Entscheidung nicht getroffen werden kann. Die Vollständigkeit des Verfahrens hängt also davon ab, ob die Information über die Art der Überlappung und die weitere Verteilung der Objekte des Überlappungsgebietes bekannt ist.

Da diese Informationen normalerweise nur ermittelt werden können, indem die Datenbankadministratoren befragt werden, ist das Verfahren nicht vollständig automatisierbar. Die Ableitung der Klauseln durch Teilmengenersetzung kann aber automatisch durchgeführt werden.

# Abbildungsverzeichnis

1.1	Föderierte Datenbanken . . . . .	5
1.2	5-Ebenen-Architektur . . . . .	6
2.1	Überlappung dreier Klassen . . . . .	10
2.2	Schema der Beispieldatenbank . . . . .	13
2.3	Exklusive Spezialisierung . . . . .	14
2.4	Totale Spezialisierung . . . . .	15
5.1	Überlappung von Klassen zweier Hierarchien . . . . .	29

# Tabellenverzeichnis

2.1	Basisextensionen für drei Klassen . . . . .	11
4.1	Basisextensionen der Beispieldatenbank $A$ . . . . .	24
5.1	Basisextensionen für $A$ und $B$ . . . . .	28
5.2	Basisextensionen bei Rollenbeziehungen . . . . .	30

# Literaturverzeichnis

- [Bus91] Busse, R.: Einordnung von Anfrageergebnissen der Objektalgebra in einen Klassenverband und ihre Implementierung. Diplomarbeit, Technische Universität Clausthal, 1991.
- [Dup94] Dupont, Y.: Resolving Fragmentation Conflicts in Schema Integration. In Loucopoulos, P. (Hrsg.): *Entity-Relationship Approach — ER'94, Proc. of the 13th Int. Conf. on the Entity-Relationship Approach, Manchester, UK*, Lecture Notes in Computer Science, Band 881, S. 513–532. Springer-Verlag, Berlin, Dezember 1994.
- [HS95] Heuer, A.; Saake, G.: *Datenbanken — Konzepte und Sprachen*. International Thomson Publishing, Bonn, 1995.
- [SCC<sup>+</sup>97] Saake, G.; Christiansen, A.; Conrad, S.; Höding, M.; Schmitt, I.; Türker, C.: Föderierung heterogener Datenbanksysteme und lokaler Datenhaltungskomponenten zur systemübergreifenden Integritätssicherung — Kurzvorstellung des Projekts **SIGMA**<sub>FDB</sub>. In Dittrich, K. R.; Geppert, A. (Hrsg.): *Datenbanksysteme in Büro, Technik und Wissenschaft, BTW'97, GI-Fachtagung, Ulm, März 1997*, Informatik aktuell, S. 322–331. Springer-Verlag, Berlin, 1997.
- [SEHT96] Schmitt, I.; Ebert, A.; Höding, M.; Türker, C.: **SIGMA**<sub>Bench</sub> – Ein Werkzeug zum Entwurf föderierter Datenbanken. In Hasselbring, W. (Hrsg.): *Kurzfassungen zum 2. Workshops “Föderierte Datenbanken”, Dortmund, 12.-13. Dezember 1996*, SWT Memo Nr. 90, S. 19–26. Fachbereich Informatik, Universität Dortmund, 1996.
- [SL90] Sheth, A. P.; Larson, J. A.: Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, Band 22, Nr. 3, S. 183–236, September 1990.
- [SS97] Schmitt, I.; Saake, G.: Merging Inheritance Hierarchies for Schema Integration based on Concept Lattices. Preprint Nr. 2, Fakultät für Informatik, Universität Magdeburg, 1997.
- [SST97] Saake, G.; Schmitt, I.; Türker, C.: *Objektdatenbanken — Konzepte, Sprachen, Architekturen*. International Thomson Publishing, Bonn, 1997.
- [TK78] Tsichritzis, D. C.; Klug, A.: The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Database Management Systems. *Information Systems*, Band 3, Nr. 3, S. 173–191, 1978.

# Selbständigkeitserklärung

Ich versichere, daß ich diese Diplomarbeit selbständig verfaßt und keine anderen als die angegebenen Hilfsmittel verwendet habe.

Magdeburg, 17. November 1997

# Thesen

1. Ein föderiertes Datenbanksystem ist ein System zur Realisierung eines integrierten Zugriffs auf Daten in heterogenen Datenbanken unter Beibehaltung der lokalen Autonomie.
2. Mit dieser Arbeit wird der Prozeß der Schemenintegration beim Entwurf eines föderierten Datenbanksystems unterstützt, indem ein Verfahren zum Auffinden der Basisextensionen einer föderierten Datenbank entwickelt wird.
3. Um die Basisextensionen zu bestimmen, müssen die extensionale Überlappungen von Objektklassen ermittelt werden.
4. Die Basisextensionen können nur dann eindeutig bestimmt werden, wenn außer den Aussagen über extensionale Überlappungen weitere Informationen zur Verfügung stehen.
5. Das Verfahren im allgemeinen nicht vollständig automatisierbar.