Informatik-Fachberichte 298

Herausgeber: W. Brauer
im Auftrag der Gesellschaft für Informatik (GI)

Alfons Kemper

Zuverlässigkeit und Leistungsfähigkeit objekt-orientierter Datenbanksysteme

Springer-Verlag
Berlin Heidelberg New York London Paris
Tokyo Hong Kong Barcelona Budapest

Autor
Alfons Kemper
RWTH Aachen, Lehrstuhl für Informatik III
Ahornstraße 55, W-5100 Aachen

CR Subject Classification (1991): H.1-2, D.1.5, D.3.1, D.3.2

ISBN-13: 978-3-540-55120-1 e-ISBN-13: 978-3-642-77266-5
DOI: 10.1007/ 978-3-642-77266-5

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, bei auch nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

© Springer-Verlag Berlin Heidelberg 1992

Satz: Reproduktionsfertige Vorlage vom Autor
33/3140-543210 – Gedruckt auf säurefreiem Papier

Vorwort

Objekt-orientierte Datenbanksysteme werden von vielen Forschern als die "nächste Generation" der Datenbanktechnologie insbesondere in den ingenieurwissenschaftlichen Anwendungsgebieten gesehen. Obwohl sie in ihrer Funktionalität den relationalen Systemen überlegen sind, werden sich Objektbanksysteme aber nur dann durchsetzen können, wenn sie in bezug auf *Zuverlässigkeit* und *Leistungsfähigkeit* den marktgängigen relationalen Systemen zumindest ebenbürtig werden.

Die vorliegende Arbeit leistet einen Beitrag hinsichtlich beider Aspekte:

- *Zuverlässigkeit*
  Es wird ein Typisierungskonzept für die strenge Typisierung der Sprache eines (persistenten) Objektmodells entwickelt, das die vollständige statische Typverifikation ermöglicht, ohne die Flexibilität und Expressivität des Modells einzuschränken.

- *Leistungsfähigkeit*
  Es werden zugriffsunterstützende Maßnahmen vorgestellt, die in streng typisierte Objektmodelle integriert werden können, um die assoziative Suche nach persistenten Objekten (auf dem Hintergrundspeicher) zu optimieren.

In dieser Arbeit wird zunächst das persistente Objektmodell GOM (Generic Object Model) in einer tutoriellen Darstellung eingeführt. Dieses Objektmodell dient uns sowohl als "Forschungsvehikel" für die Diskussion der konzeptuellen Arbeiten als auch als "Implementierungsplattform" für die prototypische Umsetzung der Typisierungs- und Optimierungsmaßnahmen. GOM wurde in Ermangelung eines weithin anerkannten (Standard-)Objektmodells konzipiert. Beim Entwurf wurden die allgemein als essentiell angesehenen objekt-orientierten Basiskonstrukte in einem orthogonalen Modell vereint. Durch die Beschränkung auf diese wenigen Basisstrukturen ist sichergestellt, daß die in dieser Arbeit beschriebenen Ergebnisse auf eine Vielzahl von anderen Objektmodellen anwendbar sind—solange diese auf den Basiskonstrukten aufbauen.

In GOM wird konsequent das Prinzip der *strengen Typisierung* verfolgt, wodurch Anwendungsprogramme zur Übersetzungszeit (also statisch) auf Typkonsistenz überprüft werden können. Wir entwickeln schrittweise ein Typinferenzsystem, das die statische Typkonsistenzverifikation ermöglicht—ohne die Expressivität der Sprache zu beeinträchtigen. Um die Flexibiltät der Sprache insbesondere für die Mengenverarbeitung zu erhöhen, werden polymorphe Operationen eingeführt, deren typkonsistente Anwendbarkeit auch zur Übersetzungszeit verifizierbar ist. Die polymorphen Operationen sind insbesondere für die aus Datenbanksicht wichtigen Selektionsoperatoren (und andere Mengenverarbeitungs-Operationen) von zentraler Bedeutung.

Der zweite Teil der Arbeit befaßt sich mit den zugriffsunterstützenden Maßnahmen, die in einem streng typisierten Objektmodell möglich sind, um den assoziativen Zugriff auf persistente Objekte zu optimieren. Es werden zwei Indexierungsmechanismen ausgearbeitet:

1. *Zugriffsrelationen*
   In Zugriffsrelationen werden häufig traversierte Referenzketten, also über eine beliebige Anzahl von komplexwertigen Attributen gehende Verweisketten zwischen Objekten, vorab materialisiert.

2. *Funktionenmaterialisierung*
   Typspezifische Operationen, die häufig in Suchprädikaten Verwendung finden, werden vorberechnet und in Strukturen abgespeichert, die den effizienten Zugriff ermöglichen.

Beide Optimierungsmaßnahmen basieren darauf, daß die Datenbankkomponenten—Attribute, Mengenelemente und persistente Variablen—streng typisiert sind. Nur in diesem Fall können die Indexstrukturen mit vertretbarem Aufwand aktuell gehalten werden.

Um die Indexstrukturen bei der Anfragebearbeitung ausnutzen zu können, wird ein regelbasierter Anfrageoptimierer vorgestellt. Durch die Regelbasierung ist der von uns konzipierte Anfrageoptimierer äußerst modular aufgebaut, wodurch er sehr leicht erweiterbar bzw. modifizierbar ist.

Dadurch können wir mit vertretbarem Aufwand Modifikationen der im Optimierer eingebauten Suchheuristiken durchführen und auch in modularer Form weitere (neu zu entwickelnde) Indexierungsmechanismen in die Anfragebearbeitung einbringen.

Für die quantitative Analyse der Zugriffsrelationen wird ein analytisches Kostenmodell entwickelt, das im wesentlichen zwei Aufgaben erfüllen kann: erstens dient es als Grundlage für den physikalischen Objektbankentwurf, um die günstigste Zugriffsrelationen-Konfiguration zu ermitteln. Zum zweiten kann der Anfrageoptimierer auf der Basis des Kostenmodells die Bearbeitungskosten alternativer Anfragebearbeitungspläne ermitteln. Zur quantitativen Bewertung der Funktionenmaterialisierung werden Auswertungen verschiedener repräsentativer Benchmark-Anwendungen vorgenommen.

Eine wissenschaftliche Arbeit entsteht nicht im luftleeren Raum, sondern ist vielmehr eine Leistung, an der viele Menschen Anteil haben. Nachfolgend möchte ich diejenigen hervorheben, die direkt an der Entstehung dieser Arbeit beteiligt waren.

Dieses Buch ist meine Habilitation, die ich an der Universität Kalrsruhe, am Institut für Programmstrukturen und Datenorganisation angefertigt habe. Herrn Prof. Dr. P. Lockemann gebührt mein ganz besonderer Dank für die langjährige Unterstützung meiner Arbeiten an seinem Lehrstuhl. Herrn Prof. Dr. P. Dadam möchte ich an dieser Stelle für die Übernahme des Korreferats dieser Habilitationsschrift danken.

Die hier vorgestellten Arbeiten sind im Rahmen des GOM-Projekts entstanden, an dem viele andere Wissenschaftler und Studenten beteiligt waren. Insbesondere möchte ich hier meinen Kollegen und Freund Dr. Guido Moerkotte hervorheben, der fast von der "Stunde Null" an alle Höhen und Tiefen des GOM-Projekts miterlebt hat. Er hat an der Entwicklung aller in diesem Buch beschriebenen Konzepte maßgeblichen Anteil. Weiterhin haben meine Kollegen Christoph Kilger, Hans-Dirk Walter und Andreas Zachmann, die an der Entwicklung von GOM beteiligt sind, zu dieser Ausarbeitung beigetragen.

Ein Projekt dieser Größenordnung kann in einer Universität natürlich nicht ohne die tatkräftige Mitarbeit zahlreicher Studenten durchgeführt werden, denen ich hier für ihren enthusiastischen Einsatz danken möchte. Nachfolgend sind die in den letzten zwei Jahren im Rahmen des Projekts von uns betreuten Studenten und deren Arbeitsgebiet aufgeführt: Monika Altmann (Einsatz des GOM-Systems im Maschinenbau), Eckard Appel (autonome Objekte), Axel Armbruster (GOM-Einsatz im CAD-Bereich), Uwe Degel (Lineares Hashverfahren mit Separatoren), Thomas Demmler (Schema-Browser), Wolfgang Häfelinger (Type-Checker), Andreas Horder (Leistungsanalyse), Donald Kossmann (Objekt-Manager), Kai Leberer (Update-Funktionen des Zugriffsrelationen-Managers), Angela Lopes de Lima (Erweiterbares Hashverfahren), Helmuth Ott (Realisierung des Kostenmodells), Uwe Oetken (Realisierung des ASR-Managers), Apostolos Papapostolou (Join-Algorithmen für den ASR-Manager), Klaus Peithner (Anfrageübersetzer), Dorothée Rühl (Synchronisation autonomer Objekte), Alexandre Saad (GOM Schema-Verwaltung), Bertil Sobottke (Objekt-Manager), Heiner Spies (Anfrage-Optimierer), Michael Steinbrunn (Funktionenmaterialisierung), Axel Tetzner (Dokumentation), Edmund Wanner (Synchronisationsverfahren) und Rüdiger Waurig (Anfrage-Optimierer).

Seit Juli 1990 werden die Arbeiten im GOM-Projekt von der DFG im Rahmen des Sonderforschungsbereichs 346 "Rechnerintegrierte Konstruktion und Fertigung von Bauteilen" innerhalb des Einzelprojekts A1 "Kooperation in verteilten Objektbanken" (Leitung: Prof. Dr. P. Lockemann und A. Kemper) gefördert.

Alfons Kemper
Aachen, November 1991

Inhalt

1 Einleitung **1**

I Das objekt-orientierte Datenmodell GOM 5

2 Die Grundlagen des Objektmodells GOM **7**
2.1 Sorten und Werte 7
2.2 Objekttypen 7
2.3 Tupelstrukturierte Typen 10
2.4 Operationen 11
2.5 Kollektions-Typen 17
2.6 Objekte, Variablen und Werte 17
2.7 Persistenz 20
2.8 Vererbung und Subtypisierung 21
2.9 Verfeinerung von Operationen und dynamisches Binden 24
2.10 Realisierung von GOM 26
2.11 Bibliographie 27

3 Strenge Typisierung **29**
3.1 GOM^{--}: ein "abgespecktes" Objektmodell 30
3.2 Statische Typisierung 34
3.3 Substituierbarkeit von Untertyp-Instanzen 36
3.4 Verfeinerung von Operationen 38
3.5 "Fallen" für strenge Typisierung 41
3.6 Polymorphismus 42
3.7 Verifikation der Typkonsistenz polymorpher Operationen 45
3.8 Ausdehnung der Typisierungskonzepte auf GOM 47
3.9 Virtuelle Typen 49
3.10 Generische Typen 50
3.11 Bibliographie 52

4 Assoziativer Objektzugriff in GOM **53**
4.1 Selektion von Objekten in GOMpl 53
4.2 GOMql: Eine deklarative Anfragesprache für GOM 56
4.3 Bibliographie 60

II Optimierungskonzepte für Objektbanken 61

5 Zugriffsrelationen **63**
5.1 Grundlegende Definitionen 63
5.2 Speicherstrukturen für Zugriffsrelationen 70
5.3 Überlappung von Zugriffsrelationen 76

5.4 Generierung und Fortschreibung von ASRs 78
5.5 Realisierung des Zugriffsrelationen-Managers 86
5.6 Bibliographie 86

6 Kostenmodell und Auswertungen 89
6.1 Grundlagen 90
6.2 Kardinalität der Zugriffsrelationen 92
6.3 Speicherkosten für Zugriffsrelationen 93
6.4 Einige Beispielauswertungen 94
6.5 Anfragebearbeitung 95
6.6 Fortschreibung der Zugriffsrelationen 104
6.7 Interpretation der Ergebnisse 115
6.8 Bibliographie 116

7 Der regelbasierte Optimierer 117
7.1 Überblick über die Architektur des GOM-Anfrageauswerters 117
7.2 Die Term-sprache 119
7.3 Anwendbarkeit der Zugriffsrelationen 122
7.4 Terminologie und Notation 122
7.5 Transformationsregeln 124
7.6 Der Regelinterpretierer und die Suchheuristik 139
7.7 Auswertung optimierter Terme 146
7.8 Bibliographie 148

8 Funktionenmaterialisierung 151
8.1 Beispiel-Schema 151
8.2 Grundlagen 154
8.3 Materialisierungsrelationen 155
8.4 Zugriff auf Materialisierungsrelationen 156
8.5 Dynamische Aspekte der Materialisierung 158
8.6 Optimierung der GMR-Fortschreibung 163
8.7 Beschränkte GMRs 172
8.8 Realisierung der Funktionenmaterialisierung 176
8.9 Bibliographie 176
8.10 Anhang: Pfadextraktion 178

9 Quantitative Bewertung der Materialisierung 181
9.1 Der Benchmark "Cuboid" 181
9.2 Der Benchmark "Company" 185
9.3 Zusammenfassung der Resultate 189
9.4 Bibliographie 190

10 Zusammenfassung und Ausblick 191
10.1 Zusammenfassung und Stand der Implementierung 191
10.2 Ausblick: zukünftige Arbeiten im Bereich Optimierung 191