Dieses Dokument ist eine Zweitveröffentlichung (Postprint) /

This is a self-archiving document (accepted version):

Rainer Gemulla, Henrike Berthold, Wolfgang Lehner

Hierarchisches gruppenbasiertes Sampling

Erstveröffentlichung in / First published in:

Informatik, Forschung und Entwicklung. 2005, 20 (1), S. 45-56 [Zugriff am: 17.11.2022]. Springer. ISSN 0949-2925.

DOI: https://doi.org/10.1007/s00450-004-0175-3

Diese Version ist verfügbar / This version is available on:

https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa2-822047

SLUB
Wir führen Wissen.

TECHNISCHE UNIVERSITÄT DRESDEN

QUCOSA
Quality Content of Saxony

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

Rainer Gemulla • Henrike Berthold • Wolfgang Lehner

Hierarchisches gruppenbasiertes Sampling

Zusammenfassung In Zeiten wachsender Datenbankgrößen ist es unumgänglich, Anfragen näherungsweise auszuwerten um schnelle Antworten zu erhalten. Dieser Artikel stellt verschiedene Methoden vor, dieses Ziel zu erreichen, und wendet sich anschließend dem Sampling zu, welches mit Hilfe einer Stichprobe schnell zu adäquaten Ergebnissen führt. Enthalten Datenbankanfragen Verbund- oder Gruppierungsoperationen, so sinkt die Genauigkeit vieler Sampling-Verfahren sehr stark; insbesondere werden vor allem kleine Gruppen nicht erkannt. Dieser Artikel befasst sich mit hierarchischen gruppenbasiertem Sampling, welches Sampling, Gruppierung und Verbundoperationen kombiniert.

Schlüsselwörter Datenbanksysteme · Anfrageverarbeitung · Näherung · Stichprobenerhebung

Abstract In times of increasing database sizes it is crucial to process queries approximately in order to obtain answers quickly. This article introduces several methods for achieving this goal and afterwards focuses on sampling, yielding appropriate results by using only a subset of the actual data. If database queries contain join or group-by operations, the accuracy of many sampling methods drops significantly; especially small groups are not recognized. This article is concerned with hierarchical group-based sampling, which combines sampling, grouping and joins.

Keywords Database systems · Query processing · Approximation · Sampling

CR Subject Classification H.2.4 · H.4.2 · H.3.3

R. Gemulla · H. Berthold · W. Lehner(✉)
Technische Universität Dresden, Database Technology Group, 01062 Dresden
Tel.: 0351-463 38257
Fax: 0351-463 38259
E-mail: {gemulla,henrike.berthold,lehner}@inf.tu-dresden.de

1 Einleitung

Aufgrund zunehmend einfacherer und vielfältigerer Möglichkeiten der Datengewinnung sowie stetig steigender Rechen- und Speicherkapazität hat in den letzten Jahren die auf Medien aller Art abgelegte Informationsmenge enorm zugenommen[1]. Datenbanken und insbesondere Data-Warehouse-Systeme können diese Datenflut aufnehmen und persistent ablegen. Je umfangreicher der Datenbestand allerdings ist, um so schwieriger wird es, sinnvolle Informationen aus ihm zu gewinnen.

Diesem Problem widmen sich insbesondere die Forschungsgebiete Knowledge Discovery und Data Mining. Die dort entwickelten Verfahren besitzen eine hohe Komplexität und erfordern mehrere Durchläufe durch die Datenbasis. Aus diesem Grund ist die Antwortzeit, also die Zeit zwischen der Anfrage des Nutzers und der Auslieferung des Ergebnisses, oft sehr hoch. Für Anfragen explorativer Art (z.B. Test von Hypothesen) reicht dagegen eine Schätzung des Ergebnisses mit einer Fehlerbetrachtung aus.

Methoden der *näherungsweisen Anfrageauswertung* erzielen einen Kompromiss zwischen geringer Antwortzeit und hoher Ergebnisqualität. Kernidee ist es, zur Anfragezeit nicht mehr den gesamten Datenbestand zu betrachten, sondern nur einen aus ihm gewonnenen Extrakt. Aufgrund der geringeren Datenmenge werden Ergebnisse so schneller generiert. Sie stellen allerdings nur eine Näherung des exakten Ergebnisses dar, welches nur durch Betrachtung des vollständigen Datenbestandes gewonnen werden kann. Damit solche Abschätzungen sinnvoll verwendbar sind, muss mit ihnen auch eine Fehlerbetrachtung erstellt und an den Nutzer weitergegeben werden.

Eine Möglichkeit, Anfragen auf diese Art auszuwerten, ist die Erhebung einer Stichprobe. Dieser Artikel befasst sich mit der Lösung zweier dabei auftretender Probleme: einerseits werden bei Gruppierungen auf Stichproben kleine Gruppen nicht erkannt, andererseits ist die Selektivität bei

[1] http://www.sims.berkeley.edu/research/projects/how-much-info-2003/index.htm

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

Verbundoperationen zwischen Stichproben sehr gering. Abschnitt 2 beschreibt diese Probleme genauer und gibt einen Überblick über verschiedene Methoden der näherungsweisen Anfrageauswertung und insbesondere der Stichprobenerhebung (engl. *sampling*). Abschnitt 3 präsentiert Techniken, um die bei Gruppierung und Sampling bzw. Verbund und Sampling auftretenden Probleme zu umgehen. In den Abschn. 4 und 5 werden diese Techniken zu *hierarchischen gruppenbasierten Sampling-Verfahren* kombiniert. Abschnitt 6 evaluiert die gewonnenen Erkenntnisse mit Hilfe des TPC-D Benchmarks [20]. Abschließend fasst Abschn. 7 diesen Artikel zusammen und gibt einen Ausblick auf aktuelle und zukünftige Arbeiten.

2 Näherungsweise Anfrageauswertung

Näherungsweise Anfrageauswertung setzt eine Reduktion der Datenbasis voraus. Der New Jersey Data Reduction Report [4] gibt einen kurzen Überblick über die vielfältigen Möglichkeiten der Datenreduktion. Grundlegend werden dabei parametrische und nicht-parametrische Ansätze unterschieden (siehe Abb. 1). Erstere nehmen ein Modell für die Datenbasis an und versuchen anschließend, Parameter dieses Modells zu schätzen, letztere verwenden kein Modell. Im Folgenden stellen wir ausschließlich nicht-parametrische Verfahren vor, darunter Wavelets, Histogramme und Stichproben.

Die *diskrete Wavelet-Transformation* arbeitet auf einer multidimensionalen Datenmatrix und ist nicht verlustbehaftet, reduziert also den Datenbestand nicht. Die durch sie gewonnenen Koeffizienten können anschließend so approximiert werden, dass möglichst wenig Speicherplatz für Regionen geringer Aktivität verwendet wird. Somit werden sowohl dünn besetzte als auch homogene Datenräume sehr gut reduziert. Regionen hoher Aktivität, d.h. Regionen, in denen benachbarte Werte starken Schwankungen unterliegen, wird dagegen mehr Aufmerksamkeit gewidmet. Probleme bereiten ungeordnete Attribute, da die Kompression von der aus der Ordnung resultierenden Korrelation benachbarter Werte profitiert. Außerdem steigt der Aufwand der Wavelet-Transformation exponentiell mit der Anzahl der Dimensionen. Sie ist also für hochdimensionale Datenräume nur bedingt geeignet. Mit Wavelets im Datenbankbereich befassen sich bspw. [5,22].

Ein *Histogramm* [15,19] wird gewonnen, in dem die Datenbasis durch eine Partitionierungsregel in so genannte Buckets aufgeteilt wird. Den einfachsten Fall stellt ein Histogramm über einem numerischen Attribut dar. So kann bspw. ein Attribut X in zwei Buckets aufgeteilt werden: eins mit allen Werten $x \leq 50$ und eins mit den Werten $x > 50$. Für Elemente eines Buckets wird eine gleichmäßige Streuung und gleichmäßige Häufigkeit angenommen. Somit ist es möglich, ein Bucket mit wenigen Charakteristiken zu beschreiben, d.h. der Gesamtzahl der Elemente, dem Minimum und Maximum sowie der Anzahl der unterschiedlichen Werte. Datenbankanfragen können nun mit Hilfe dieser Bucketbeschreibungen näherungsweise beantwortet werden.

Schwieriger wird die Histogrammerstellung bei Betrachtung mehrerer Attribute. Einerseits kann für jedes Attribut ein Histogramm, andererseits auch ein multidimensionales Histogramm für alle Attribute angelegt werden. Im ersten Fall gehen allerdings Abhängigkeiten zwischen den Attributen verloren, im zweiten stellen hochdimensionale Datenräume ein Problem dar. Ein weiterer Nachteil von Histogrammen ist die Beschränkung auf numerische Attribute. Diese kann man umgehen, indem das Histogramm vollständig erzeugt wird, d.h. jedes Bucket nur einen einzigen Wert umfasst. Alternativ kann, sofern möglich, eine numerische Abbildung der Werte verwendet werden.

Sampling-Verfahren

Der Hauptaugenmerk dieses Artikels liegt auf Sampling-Verfahren, also Methoden, die nur auf einer Stichprobe (engl. *sample*) des gesamten Datenbestandes operieren. Das Sampling ist vor allem für Aggregationsanfragen wie Summe oder Durchschnitt geeignet. Im Wesentlichen wird zwischen *Online Sampling* [7,11,13,14,18,21] und *Offline Sampling* [1, 2,3,6,9,10,16,17] unterschieden (siehe Abb. 2). Beim Online Sampling wird die Stichprobe (ψ) erst zur Zeit der Anfrage durch den Nutzer erhoben und anschließend zur Auswertung benutzt. Das Offline Sampling dagegen berechnet die Stichprobe schon vor der eigentlichen Anfrage. Durch dieses Vorgehen ist es möglich, mehr Aufwand in die Erstellung der Stichprobe zu investieren, um die Genauigkeit der aus ihr gewonnenen Ergebnisse zu erhöhen. Allerdings muss die Stichprobe bei Änderungen der Datenbasis aktuali-

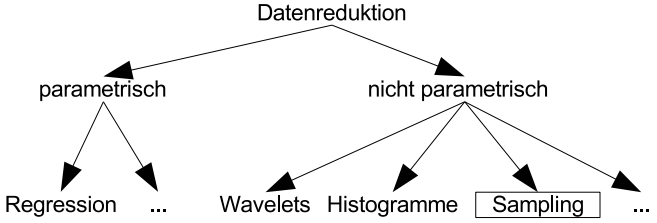


Abb. 1 Überblick über Datenreduktionsverfahren

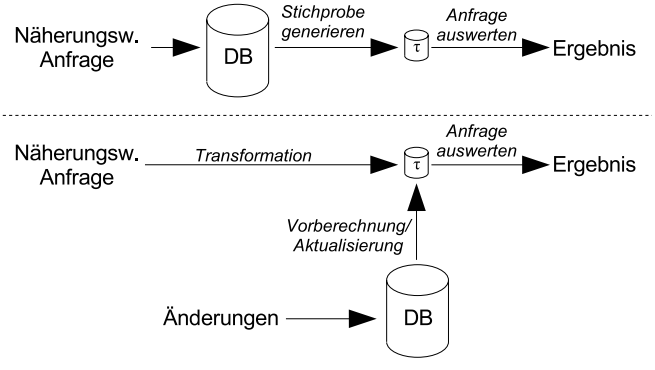


Abb. 2 (*oben*) Online Sampling (*unten*) Offline Sampling

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

siert werden. Somit eignet sich das Online Sampling eher für kurzlebige, das Offline Sampling eher für langlebige Daten.

Die *Online Aggregation* [14] ist ein Vertreter der ersten dieser beiden Klassen. Kernidee ist es, dem Nutzer iterativ verfeinerte näherungsweise Ergebnisse für Aggregationsanfragen zu liefern. Dazu wird der betrachtete Datenbestand schrittweise vergrößert. Probleme erzeugen blockierende Operationen wie etwa Gruppierung und Verbund, da sie im Allgemeinen erst den gesamten Datenbestand lesen (z.B. aufgrund von Sortierungen), bevor sie Ergebnisse liefern. Dieses Problem kann mit Hilfe von besonderen Verbundoperationen (*ripple join* [12]) und Indizes auf Gruppierungsattributen (*index striding* [14]) teilweise umgangen werden. Dem Nutzer werden neben den aktuellen Aggregaten auch deren erwartete Fehler präsentiert. Er ist somit in der Lage, die Bearbeitung abzubrechen, wenn ihm die erreichte Genauigkeit ausreicht.

Offline Sampling

Beim Offline Sampling werden Stichproben im Voraus berechnet und in der Datenbasis materialisiert. Die meisten Sampling-Verfahren dieser Klasse widmen sich je einem speziellen Problem, welches bei der Stichprobenerhebung bzw. -verwendung auftreten kann. Im Folgenden werden verschiedene dieser Probleme und jeweilige Lösungsansätze kurz umrissen.

Oft ist es notwendig, eine fest vorgegebene Anzahl Tupel zufällig aus einer Datenbasis unbekannten Ausmaßes zu gewinnen. Das *Reservoir Sampling* [21,17] löst dieses Problem durch Anpassung einer temporären Stichprobe (Reservoir) mit jedem eingehenden Tupel der Datenbasis. Dabei wird garantiert, dass am Ende der Verarbeitung die Aufenthaltswahrscheinlichkeit jedes Tupels im Sample identisch ist. Viele der im Folgenden vorgestellten Verfahren nutzen das Reservoir Sampling, um Samplegrößen im Voraus festlegen zu können. Die von Ganti et al. entwickelten *ICICLES* [9] versuchen, das Sample in Abhängigkeit der am Datenbanksystem anliegenden Last zu erstellen und aktuell zu halten. Dazu wird die Idee des Reservoirs zweckentfremdet: die Ergebnistupel jeder (exakten) Anfrage an das System werden genauso wie Tupel der Basisdaten interpretiert und in das Reservoir aufgenommen. Somit treten Tupel, auf die häufig zugegriffen wird, mit erhöhter Wahrscheinlichkeit im Sample auf. Potentiell können dadurch Duplikate in der Stichprobe auftreten, ihre Behandlung erfordert zusätzlichen Aufwand [10]. Ein wesentlicher Nachteil dieses Vorgehens besteht darin, dass auch selten benutzte Tupel von Interesse sein können, aber sehr wahrscheinlich nicht im Sample vorkommen.

Das von Chaudhuri et al. entwickelte *Outlier Indexing* [6] versucht in den Daten vorhandene Verzerrungen (d.h. nichtgleichverteilte Wertverteilungen) zu erkennen und aufzulösen. Das Verfahren beschränkt sich auf ein Aggregationsattribut. Die Datenbasis wird in zwei Partitionen unterteilt: eine mit gleichverteilten Daten und eine mit so genannten Ausreißern, d.h. Tupeln, deren Wert stark vom Durchschnitt abweicht und somit zu einer hohen Streuung führt. Aus der ersten der beiden Partitionen wird eine Stichprobe gewonnen, die zweite komplett indiziert. Anfragen an das System werden auf beiden Teilen ausgeführt und die Ergebnisse miteinander verschmolzen. Das Outlier Indexing funktioniert dann gut, wenn die Daten im Wesentlichen gleichverteilt sind und nur einige Ausreißer auftreten. Die Behandlung von mehr als einem Aggregationsattribut ist allerdings noch offen. Besitzt man Wissen über die Daten, können die durch Datenverzerrungen entstehenden Fehler auch direkt ausgeglichen werden [16].

Gruppierungsoperationen partitionieren die Datenbasis nach gegebenen Attributen, den Gruppierungsattributen, und berechnen für jede der entstehenden Partitionen (Gruppen) Aggregationswerte wie Anzahl, Summe oder Durchschnitt. Führt man eine Gruppierung auf einer Stichprobe aus, so tritt es oft auf, dass Gruppen, die nur verhältnismäßig wenige Tupel enthalten, keine Tupel in der Stichprobe besitzen und somit nicht im näherungsweisen Ergebnis vorkommen. Das *Senate Sampling* [1] und das *Small Group Sampling* [3] versuchen dieses Problem zu lösen. Die Kernideen dieser Verfahren werden in Abschn. 3 vorgestellt und sind Ausgangspunkt für die in diesem Artikel vorgeschlagenen Erweiterungen.

Chaudhuri et al. zeigen in [8], dass Sampling und Verbundoperationen erhebliche Probleme bereiten. Insbesondere ist der Verbund zweier Samples nicht identisch mit dem Sample des Verbundes der beteiligten Relationen (siehe Abb. 3). Eine Lösung dieses Problems für den Fall des Fremdschlüsselverbundes liefern Acharya et al. mittels *Join Synopsen* [2]. Alle der bisher vorgestellten Sampling-Verfahren arbeiten nur auf einer einzigen Relation. Sollen mehrere Relationen und deren Beziehungen betrachtet werden, so müssen diese Verfahren entsprechend erweitert werden. Für die vorgestellten gruppenbasierten Sampling-Verfahren werden dabei auftretende Probleme und Lösungsvorschläge in diesem Artikel explizit erörtert (siehe Abbildung 4). Die hier angewendeten Methoden können in angepasster Form auch für andere Verfahren der Stichprobenerhebung verwendet werden.

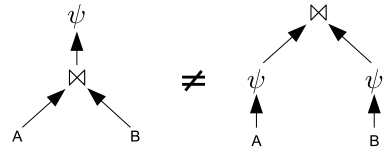


Abb. 3 Kombination von Sampling (ψ) und Verbund ($\bowtie$)

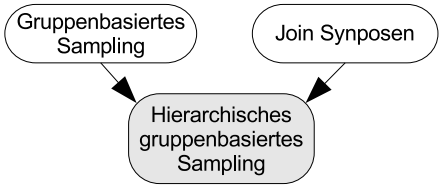


Abb. 4 Kombination der Sampling-Verfahren

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

3 Sampling, Verbund und Gruppierung

Um das Problem der kleinen Gruppen zu umgehen, verwenden die beiden Sampling-Verfahren Senate Sampling [1] und Small Group Sampling [3] sehr unterschiedliche Ansätze. Das Senate Sampling garantiert, dass Tupel jeder Gruppe in der Stichprobe auftreten, setzt aber voraus, dass alle potentiellen Gruppierungsattribute schon bei der Erstellung des Samples angegeben werden. Das Small Group Sampling dagegen ist generisch, kann aber nicht gewährleisten, dass alle Gruppen in der Stichprobe repräsentiert werden.

Dieser Abschnitt beschreibt die Kernidee beider Sampling-Verfahren und stellt anschließend eine Möglichkeit vor, um Sampling und Verbundoperationen miteinander zu kombinieren.

3.1 Senate Sampling

Das von Acharya et al. entwickelte Senate Sampling [1] setzt folgende zwei Punkte voraus:

1. Alle Attribute, nach denen potentiell gruppiert werden kann, müssen vor der Erhebung der Stichprobe bekannt sein.
2. Die Anzahl der entstehenden, nicht-leeren Gruppen bei Gruppierung nach allen angegebenen Gruppierungsattributen muss kleiner sein als die Größe der Stichprobe.

Beide Anforderungen ergeben sich aus der Tatsache, dass das Senate Sampling jeder Gruppe die gleiche Anzahl Tupel in der Stichprobe reserviert. Die zweite der genannten Voraussetzungen garantiert, dass für jede Gruppe mindestens ein Tupel zur Verfügung steht.

Das eigentliche Sampling erfolgt in einem Durchlauf durch die Relation. Dazu wird für jede neu auftretende Gruppe ein eigenes Reservoir [21] angelegt. Sind bisher m Gruppen aufgetreten und soll die Stichprobe n Tupel enthalten, so wird jedes dieser Reservoirs auf $s_g = n/m$ Tupel beschränkt. Abschließend werden alle Reservoirs in eine einzige Relation ausgeschrieben.

Anfragen mit Gruppierungen können nun mit Hilfe des Senate Samples beantwortet werden. Das Verfahren birgt jedoch einige Schwierigkeiten. Zum einen wird es oft so sein, dass durch Angabe aller potentiellen Gruppierungsattribute die Anzahl der entstehenden Gruppen viel zu hoch ist und somit keine sinnvolle Stichprobe erzeugt werden kann. Alternativ können natürlich mehrere Stichproben für je eine Teilmenge der Gruppierungsattribute verwendet werden. Weiterhin ist im Allgemeinen die Anzahl s_g der pro Gruppe reservierten Tupel nicht ganzzahlig. Rundet man ab, so wird die Zielgröße n der Stichprobe nicht erreicht, rundet man auf wird sie überschritten. Einzige Lösung ist es, einige Gruppen auf- und einige abzurunden. Dazu gibt es verschiedene Vorgehensweisen, bspw. kann man den größten Gruppen mehr Speicherplatz spendieren. Nichtzuletzt können Gruppen weit weniger Tupel beinhalten, als ihnen Platz in der Stichprobe zugedacht ist. Dadurch bleibt wiederum ein Teil des Speicherplatzes ungenutzt.

Weil jeder Gruppe in der Stichprobe gleichviel Speicherplatz zugesichert wird, erhöht sich die Genauigkeit kleinerer Gruppen, denn sie erhalten durch dieses Vorgehen mehr Tupel. Mit der selben Begründung werden große Gruppen schlechter repräsentiert. Diesen Nachteil versucht das Basic Congress Sampling [1] zu beheben, in dem es den pro Gruppe reservierten Speicherplatz zusätzlich von der Anzahl der Tupel in der Gruppe abhängig macht. Noch einen Schritt weiter geht das Congressional Sampling [1], welches jede mögliche Gruppierung nach einer Teilmenge der Gruppierungsattribute ebenso bei der Ermittlung des Speicherplatzes in der Stichprobe berücksichtigt.

3.2 Small Group Sampling

Das Small Group Sampling [3] wurde von Babcock et al entwickelt. Es ist ein Spezialfall der so genannten *dynamischen Stichprobenwahl* (engl. dynamic sample selection), bei der mehrere Stichproben erzeugt und zur Anfragezeit eine adäquate Teilmenge zur Abarbeitung ausgewählt wird. Kernidee des Small Group Sampling ist es, pro Attribut der Basisrelation eine so genannte Kleine-Gruppen-Tabelle (engl. *small group table*, SGT) anzulegen, die alle diejenigen Tupel aufnimmt, deren Ausprägung in diesem Attribut selten ist. Jedes dieser seltenen Attributwerte führt zu einer kleinen Gruppe, falls dieses Attribut sich unter den Gruppierungsattributen befindet. Zusätzlich wird eine zufällige Stichprobe der Basisrelation erzeugt. Dieses *Basissample* beinhaltet i.A. Tupel aller großen Gruppen.

Vom Anwender sind zur automatischen Erhebung des Small Group Samples die drei Parameter r, τ und t anzugeben. Die Basissamplingrate r bestimmt die Größe n des Basissamples in Abhängigkeit der Anzahl N der Tupel der Basisrelation. Es gilt $n = N * r$.

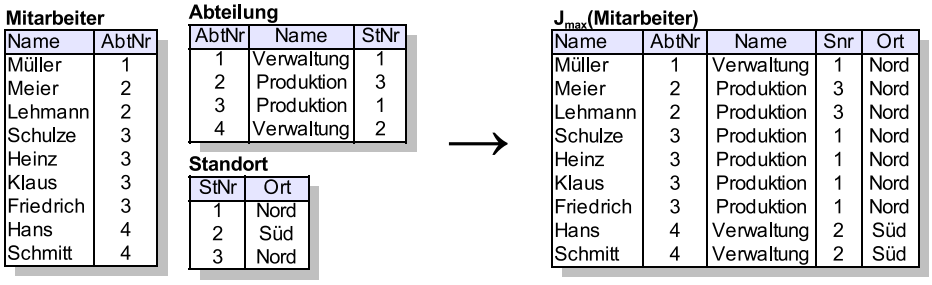
Besitzt ein Attribut mehr als τ verschiedene Ausprägungen, so wird für dieses Attribut keine SGT erzeugt. Dies tritt insbesondere bei aus nur einem Attribut bestehenden Primärschlüsseln auf. τ wird benutzt, um automatisch zu bestimmen, nach welchen Spalten potentiell gruppiert wird und nach welchen nicht.

Der Parameter t (engl. *small group fraction*) bestimmt schließlich die maximale Größe einer SGT ($n_{sgt} = N * t$). Bei der Erstellung der SGT werden nur die am seltensten auftretenden Attributausprägungen in die SGT aufgenommen. Somit zieht der Parameter t implizit die Grenze zwischen selten und häufig auftretenden Ausprägungen.

Die eigentliche Erhebung der Stichprobe erfolgt in zwei Phasen mit je einem Durchlauf durch die Relation. Zuerst werden Histogramme für alle Attribute erzeugt. Mit deren Hilfe kann pro Attribut entschieden werden, welche Ausprägungen selten sind und welche nicht. In der zweiten Phase werden mit diesem Wissen das Basissample und die SGTs erzeugt. Dabei wird zu jedem Tupel die Information mit abgespeichert, in welche der Stichprobentabellen es aufgenommen wurde. Zur Anfragezeit wird dann das Basissample und die SGTs der jeweiligen Gruppierungsattribute zur Beant-

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3



Mitarbeiter

Name	AbtNr
Müller	1
Meier	2
Lehmann	2
Schulze	3
Heinz	3
Klaus	3
Friedrich	3
Hans	4
Schmitt	4

Abteilung

AbtNr	Name	StNr
1	Verwaltung	1
2	Produktion	3
3	Produktion	1
4	Verwaltung	2

Standort

StNr	Ort
1	Nord
2	Süd
3	Nord

J_{max}(Mitarbeiter)

Name	AbtNr	Name	Snr	Ort
Müller	1	Verwaltung	1	Nord
Meier	2	Produktion	3	Nord
Lehmann	2	Produktion	3	Nord
Schulze	3	Produktion	1	Nord
Heinz	3	Produktion	1	Nord
Klaus	3	Produktion	1	Nord
Friedrich	3	Produktion	1	Nord
Hans	4	Verwaltung	2	Süd
Schmitt	4	Verwaltung	2	Süd

Abb. 5 Beispielszenario und maximaler Fremdschlüsselverbund

wortung verwendet. Tupel aller Gruppen mit mindestens einer seltenen Ausprägung werden durch dieses Verfahren zu 100% abgespeichert. Somit kann ihr Aggregat exakt ermittelt werden. Alle anderen Gruppen werden durch das Basissample bedient und somit näherungsweise betrachtet.

Probleme bereiten dem Small Group Sampling Abhängigkeiten zwischen Attributen. Insbesondere werden seltene Kombinationen häufiger Attributausprägungen nicht erkannt und die dadurch entstehenden Gruppen fallen durch das Raster. Im Gegensatz zum Senate Sampling müssen die Gruppierungsattribute nicht im Voraus bekannt sein, dafür ist die Parametrierung schwierig und erfordert Kenntnisse der Daten.

3.3 Join Synopsen

Eine Möglichkeit der Kombination von Sampling und Verbundoperationen sind die von Acharya et al. entwickelten *Join Synopsen* [2]. Sie basieren auf Fremdschlüsselverbünden. Seien bspw. R und S zwei Relationen und besitze R einen Fremdschlüssel auf S. Im Folgenden wird S auch als Nachfolger von R bezeichnet. Der Verbund $R \bowtie S$ über die Fremdschlüsselbeziehung heißt dann Fremdschlüsselverbund. Zwischen der Relation R und dem Fremdschlüsselverbund $R \bowtie S$ besteht eine 1:1-Beziehung: der Fremdschlüsselverbund erweitert die Relation R praktisch nur um die Attribute von S.

Natürlich kann ein Fremdschlüsselverbund weit mehr als zwei Relationen zusammenfassen. Der *maximale Fremdschlüsselverbund* $J_{max}(R)$ verbindet eine *Quellrelation* R mit allen durch Fremdschlüsselbeziehungen erreichbaren Relationen. Abbildung 5 demonstriert dies am Beispiel: die Quellrelation *Mitarbeiter* besitzt einen Fremdschlüssel (*AbtNr*) auf die Relation *Abteilung*, welche wiederum die Relation *Standort* referenziert (mit *StNr*). Es gilt somit:

$$J_{max}(Mitarb.) = Mitarbeiter \bowtie Abteilung \bowtie Standort$$

Das Ergebnis dieses Verbundes ist ebenfalls in Abb. 5 dargestellt. Prinzipiell kann allerdings nur dann ein maximaler Fremdschlüsselverbund angegeben werden, wenn das Datenbankschema zyklenfrei ist, oder genauer, wenn man ausgehend von der Quellrelation keinen Zyklus erreicht. Im Rahmen dieses Artikels setzen wir Zyklenfreiheit voraus und gehen sogar noch einen Schritt weiter: jede Relation darf nur einmal im Fremdschlüsselverbund auftreten. Diese Restriktion wird nur zum einfacheren Verständnis des Folgenden getroffen, ist aber keine Voraussetzung für Fremdschlüsselverbund oder hierarchische Sampling-Verfahren.

Die Join Synopse einer Relation R entspricht einer Stichprobe des *maximalen* Fremdschlüsselverbundes $J_{max}(R)$. Aufgrund der 1:1-Beziehung der Tupel in R und $J_{max}(R)$ kann allerdings das Sampling vor den Verbund gezogen werden, d.h. zuerst ermittelt man eine Stichprobe U_R von R und bildet anschließend den maximalen Fremdschlüsselverbund $J_{max}(U_R)$. Für die Relation *Mitarbeiter* ergibt sich:

$$J_{max}(U_{Mitarb.}) = U_{Mitarb.} \bowtie Abteilung \bowtie Standort$$

Die so entstehende Join Synopse kann für alle Anfragen verwendet werden, die die Relation R sowie von ihr über Fremdschlüsselbeziehungen referenzierte Relationen verwenden. Insbesondere lassen sich Gruppierungen auf den Attributen aller beteiligten Relationen berechnen. Die in den folgenden beiden Kapiteln vorgestellten Sampling-Verfahren kombinieren diese Join Synopsen mit gruppenbasiertem Sampling.

4 Hierarchisches Senate Sampling

Die naheliegende Idee, einfach ein Senate Sample S_R der Relation R zu berechnen und dessen maximalen Fremdschlüsselverbund $J_{max}(S_R)$ als Stichprobe in der Datenbank abzulegen, ist leider nicht sinnvoll. Durch diese Beschränkung des Senate Samples auf die Relation R können Gruppierungsattribute auch nur Attribute von R sein. Sinnvoll ist aber eine Stichprobe, die beliebige Gruppierungsattribute zulässt. Naiv kann dies erreicht werden, in dem das Senate Sample ausgehend von $J_{max}(R)$ berechnet wird. So kann bspw. mittels des maximalen Fremdschlüsselverbunds $J_{max}(Mitarbeiter)$ aus Abb. 5 problemlos ein Senate Sample mit beliebigen Gruppierungsattributen berechnet werden. Allerdings ist dieses Vorgehen sehr aufwändig. In diesem Abschnitt stellen wir eine Technik vor, die es erlaubt, Gruppierungsattribute auf allen beteiligten Relationen zu definieren ohne vorher den großen Verbund $J_{max}(R)$ zu berechnen. Dazu werden in einem ersten Schritt Informationen über die in den beteiligten Relationen auftretenden Gruppen gesammelt und in einem zweiten Schritt zur Berechnung des hierarchische Senate Samples verwendet. Wir werden im Folgenden das Szenario aus Abb. 5 als Beispiel verwenden. Die Gruppierungsattribute seien *Standort.Ort* und *Abteilung.Name*.

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

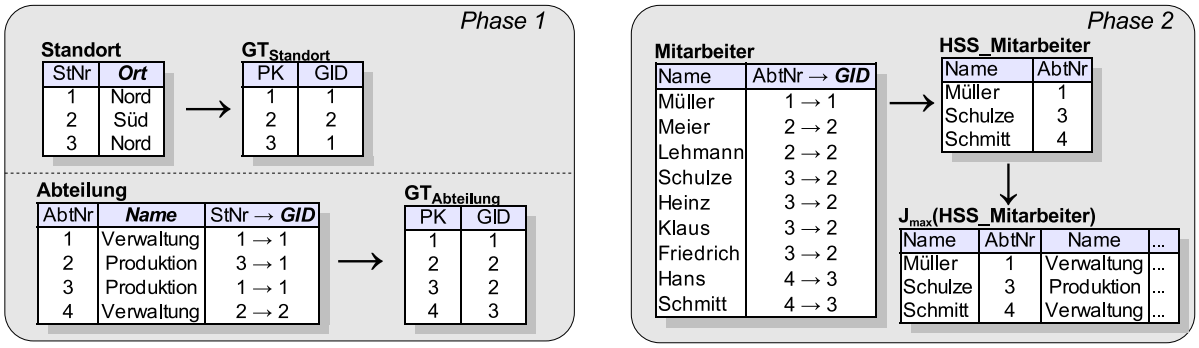


Abb. 6 Berechnung des hierarchischen Senate Samples

4.1 Gruppentabellen

Das Senate Sampling macht es erforderlich, zu jedem Tupel der Quellrelation zu bestimmen, in welche Gruppe es gehört. Im Fall einer einzelnen Relation kann dieses Wissen direkt aus den Ausprägungen der Gruppierungsattribute gewonnen werden. Sind allerdings mehrere Relationen beteiligt, ist dies nicht mehr möglich. Somit müssen vor dem Sampling der Quellrelation zusätzliche Informationen aus der Datenbasis gewonnen werden.

Die *Gruppentabelle* GT_R einer Relation R ordnet jedem Tupel aus R seine Gruppe unter gegebenen Gruppierungsattributen zu. Jedes Tupel wird dabei durch seinen Primärschlüssel, jede Gruppe durch einen eindeutigen *Gruppenidentifikator* (GID) repräsentiert. Ziel ist es, anhand der Fremdschlüssel der Quellrelation und der GT der jeweils referenzierten Relation die Gruppe des Tupels bestimmen zu können. Die realen Ausprägungen der Gruppierungsattribute sind dabei nicht von Interesse, es ist ausreichend, wenn je zwei Tupel einer Gruppe denselben GID und je ein Tupel zweier verschiedener Gruppen auch einen verschiedenen GID besitzen.

Die Gruppentabelle muss nicht für jede beteiligte Relation berechnet werden. Eine Relation R heißt *gruppierungsrelevant*, wenn entweder auf ihr Gruppierungsattribute definiert worden sind und/oder sie einen Fremdschlüssel auf eine gruppierungsrelevante Relation besitzt. Im Beispielszenario sind alle Relationen gruppierungsrelevant: *Standort* und *Abteilung* besitzen Gruppierungsattribute, *Mitarbeiter* einen Fremdschlüssel auf die gruppierungsrelevante Relation *Abteilung*.

Ziel der ersten Phase des hierarchischen Senate Samplings ist es, die GT aller direkt von der Quellrelation referenzierten gruppierungsrelevanten Relationen zu berechnen. Dazu beginnen wir mit den Relationen, welche keinen Fremdschlüssel auf gruppierungsrelevante Relationen besitzen (Blätter) und arbeiten uns rückwärts entlang der Fremdschlüsselbeziehungen zur Quellrelation vor (Bottom-Up).

Algorithmus 1 beschreibt die Erstellung der Gruppentabelle einer Relation R mit k Fremdschlüsselbeziehungen auf gruppierungsrelevante Relationen. Ist R ein Blatt ($k = 0$), so erfolgt die Bestimmung der Gruppe eines Tupels durch einfaches Zusammenfassen der Ausprägungen der auf dieser

Alg. 1 Erstellung der Gruppentabelle

```
 1: in R                                          // Eingaberelation
 2: in T                                          // Gruppierungsattribute in R
 3: in pk, fk_1, ..., fk_k                        // Primär- u. Fremdschlüssel von R
 4: in GT_1, ..., GT_k                            // Gruppentab. der ref. Relationen
 5: out GT_R                                      // Ergebnis: Gruppentabelle von R
 6:
 7: create H                                      // Hashtabelle erzeugen
 8: for all t ∈ R do                              // für jedes Tupel t
 9:    group := proj_T(t)                         // Gruppe bestimmen
10:    if k > 0 then                              // kein Blatt?
11:       // GID der Nachf. als zusätzliche Attr. verwenden
12:       group += (GT_1[fk_1(t)], ..., GT_k[fk_k(t)])
13:    end if
14:    gid := H[group]                            // Gruppen-ID ermitteln
15:    if gid = empty then                        // neue Gruppe?
16:       gid := newid(group)                     // Gruppen-ID vergeben
17:       H[group] := gid                         // Hashtabelle aktualisieren
18:    end if
19:    GT_R[pk(t)] := gid                         // Gruppentabelle aktualisieren
20: end for
21: delete H, GT_1, ..., GT_k                     // nicht mehr benötigt
```

Relation definierten Gruppierungsattribute T (Zeile 9). Zur Ermittlung des GID ist eine temporäre Datenstruktur (hier Hashtabelle H) notwendig, welche die Zuordnung zwischen Gruppe und GID festhält (Zeilen 14–18). Nach der Berechnung der GT kann sie sofort wieder gelöscht werden. Für jedes Tupel t wird abschließend ein Eintrag bestehend aus seinem Primärschlüssel $pk(t)$ und seinem GID in die GT aufgenommen (Zeile 19).

Abbildung 6 links zeigt die erste Phase des hierarchischen Senate Samplings des Beispielszenarios aus Abb. 5. Die einzige Blattrelation ist *Standort*. Sie besitzt nur ein Gruppierungsattribut (*Ort*), ihre Gruppentabelle $GT_{Standort}$ ist im oberen Teil dargestellt.

Um die GT einer Relation mit Fremdschlüsseln zu berechnen ($k > 0$), müssen die GT aller von ihr referenzierten gruppierungsrelevanten Relationen schon berechnet worden sein. In unserem Beispiel wird also $GT_{Standort}$ benötigt, um $GT_{Abteilung}$ berechnen zu können. Prinzipiell ist das Vorgehen dazu identisch wie bei Blättern, allerdings wird der Wert jedes Fremdschlüssels $fk_i(t)$ in der entsprechenden Gruppentabelle GT_i nachgeschlagen und als zusätzliches Gruppierungsattribut verwendet (Zeile 12). Durch dieses Vorgehen werden

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

auch die in den Nachfolgerrelationen entstandenen Gruppen berücksichtigt.

Abbildung 6 zeigt unten links die GT der Relation *Abteilung*. Dazu wurden das Attribut *Name* und der Gruppenidentifikator aus der Gruppentabelle $GT_{Standort}$ als Gruppierungsattribute verwendet. Nach der Berechnung wird $GT_{Standort}$ nicht mehr benötigt und kann aus dem Hauptspeicher entfernt werden. Die erste Phase ist an dieser Stelle bereits abgeschlossen, da für die Quellrelation keine GT berechnet werden muss.

4.2 Sampling

Das eigentliche Sampling der Quellrelation ist nahezu identisch mit dem Vorgehen des normalen Senate Sampling. Einziger Unterschied ist genau wie bei der Erzeugung einer GT für eine Relation mit gruppierungsrelevanten Nachfolgern, dass Fremdschlüssel in der entsprechenden GT der referenzierten Relation nachgeschlagen werden und der so ermittelte GID ein zusätzliches Gruppierungsattribut darstellt.

In Abb. 6 rechts ist ein mögliches Ergebnis für das Beispielszenario und eine Samplegröße von $n = 3$ Tupeln dargestellt. Aus jeder der Gruppen (*Verwaltung*, *Nord*), (*Produktion, Nord*) und (*Verwaltung*, *Süd*) wurde je ein Tupel in die Stichprobe aufgenommen. Abschließend muss noch der maximale Fremdschlüsselverbund dieser Stichprobe berechnet werden.

Die beschriebene Vorgehensweise kann in identischer Form auch für das Basic Congress Sampling verwendet werden. Probleme treten erst beim Congressional Sampling auf, denn es benötigt die Ausprägungen aller Gruppierungsattribute. Die dazu notwendige Erweiterung der Gruppentabellen erhöht deren Speicherbedarf. Der Rechenaufwand für das Congressional Sampling steigt außerdem exponentiell: für $|T|$ Gruppierungsattribute müssen $2^{|T|}$ mögliche Teilgruppierungen betrachtet werden. Aus diesem Grund ist Congressional Sampling mit vielen Gruppierungsattributen nicht effizient umsetzbar.

5 Hierarchisches Small Group Sampling

Wie auch beim Senate Sampling kann das hierarchische Small Group Sample einer Relation R naiv mittels deren maximalem Fremdschlüsselverbund $J_{max}(R)$ berechnet werden. Durch ein hierarchisches Vorgehen kann aber auch in diesem Fall sehr viel Aufwand eingespart werden. Die Kernprobleme des Small Group Samplings, Ermittlung der seltenen Ausprägungen und Bestimmung aller Tupel mit diesen Ausprägungen, müssen ohne Berechnung des maximalen Fremdschlüsselverbundes gelöst werden. Das hierarchische Small Group Sampling verläuft dazu dreiphasig: Phase 1 berechnet Histogramme über Relationengrenzen hinweg, Phase 2 bestimmt die Tupel mit seltenen Ausprägungen. In der dritten Phase werden abschließend das Basissample und die SGTs erstellt.

5.1 Berechnung der Histogramme

Wie in Abschn. 3.2 erläutert, erfolgt die Bestimmung der seltenen Ausprägungen durch Erstellung von Histogrammen für jedes Attribut. Ihre Berechnung gestaltet sich im Falle einer einzigen Relation R sehr einfach: in einem Durchlauf durch R wird jede Ausprägung ihrer Attribute gezählt. Bei Betrachtung mehrerer Relationen ist es jedoch nicht möglich, die Histogramme jeder Relation getrennt zu berechnen, da dadurch der Einfluss der Fremdschlüssel verloren geht. Stattdessen müssen die Referenzen auf jedes Tupel bei der Histogrammberechnung einbezogen werden.

Die *Referenztabelle* $RT_{R \Rightarrow S}$ beinhaltet zu jedem Tupel aus S die Anzahl der Tupel aus R, die es referenzieren. Dazu enthält sie Primärschlüssel und Anzahl der Referenzierungen jedes Tupels aus S. Nicht-referenzierte Tupel müssen nicht in der Referenztabelle auftreten.

Die Berechnung der Histogramme und der RT werden im Folgenden mit Hilfe des Beispielszenarios aus Abb. 5 erläutert. Ausgehend von der Quellrelation *Mitarbeiter* werden in Richtung der Fremdschlüsselbeziehungen (Top-Down) die Histogramme jeder Relation berechnet. Simultan wird die Referenztabelle auf alle in Fremdschlüsselbeziehungen stehenden Relationen erzeugt, d.h. bei Abarbeitung der Relation *Mitarbeiter* (M) die Referenztabelle $RT_{M \Rightarrow A}$ auf die Relation *Abteilung* (A).

Die Referenztabelle $RT_{M \Rightarrow A}$ enthält wie schon erwähnt die Anzahl der Referenzen von M auf jedes Tupel in A. Diese Anzahl entspricht der Häufigkeit des Auftretens jedes Fremdschlüssels auf A, die gesuchte Referenztabelle ist somit identisch mit dem Histogramm der Fremdschlüsselattribute. Abbildung 7 links zeigt das Histogramm des Attributes *Name* der Relation *Mitarbeiter* sowie die Referenztabelle $RT_{M \Rightarrow A}$. In unserem Beispiel besteht der Fremdschlüssel nur aus einem Attribut (*AbtNr*). Dadurch ist die RT identisch mit dem Histogramm des Attributs *AbtNr*.

Während der Histogrammberechnung werden alle Histogramme mit mehr als τ Einträgen (vgl. Abschn. 3.2) sofort gelöscht. Im Beispiel ist $\tau = 2$ und somit werden beide Histogramme verworfen. Außerdem werden alle Histogramme, die nach Abarbeitung einer Relation übrig bleiben, auf die seltenen Ausprägungen gekürzt (vgl. Abschn. 3.2). Dieser Fall tritt hier im ersten Schritt allerdings nicht auf.

Unter Verwendung der Referenztabelle $RT_{M \Rightarrow A}$ kann die Relation *Abteilung* betrachtet werden. Die Histogramm- und Referenztabellenberechnung verläuft nahezu identisch wie bei der Quellrelation. Allerdings wird nun jedes Tupel mit der Anzahl der Referenzen auf sich gewichtet (siehe Abb. 7, Phase 1, Mitte). Dadurch sind die entstehenden Histogramme identisch mit denen des maximalen Fremdschlüsselverbundes (vgl. Abb. 5). Für die Relation *Abteilung* bleibt nur das Histogramm des Attributes *Name* übrig ($\tau = 2$). Dieses wird auf die seltenen Ausprägungen gekürzt; im Beispiel sei $t = \frac{1}{3}$, d.h. jede SGT kann bis zu 3 Tupel aufnehmen. Somit sind alle Tupel mit der Ausprägung "Verwaltung" selten, die häufige Ausprägung "Produktion" wird aus dem Histogramm gestrichen. Die alte Referenztabelle $RT_{M \Rightarrow A}$ wird

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

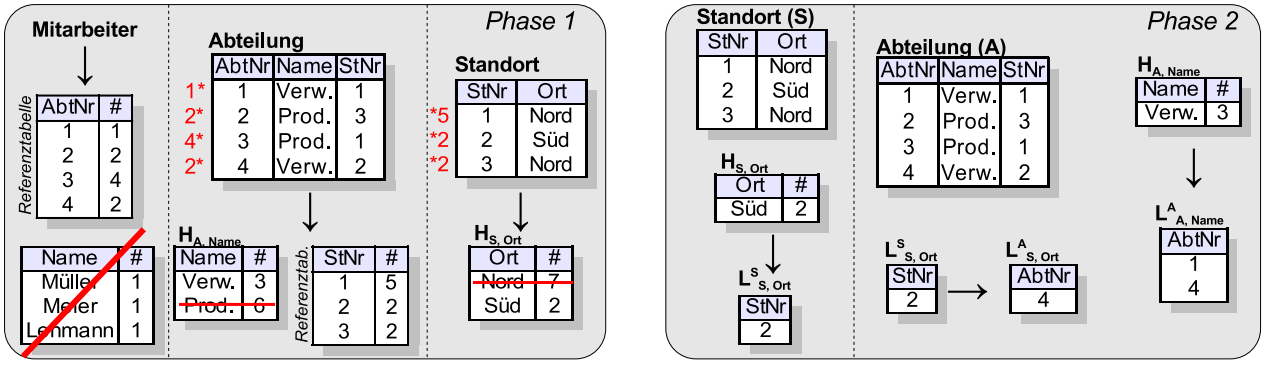


Abb. 7 Phase 1 und 2 der Berechnung des hierarchischen Small Group Samples

nicht mehr benötigt und somit aus dem Hauptspeicher gelöscht. Die neu angelegte $RT_{M \Rightarrow S}$ ermöglicht die Berechnung der Histogramme der Relation *Standort* (S), welche ebenfalls in Abb. 7 dargestellt sind.

Algorithmus 2 stellt die komplette erste Phase noch einmal übersichtlich dar. Er beschreibt die Ermittlung der Histogramme $H_1, \ldots, H_c$ einer Relation R, wenn die Referenztabelle $RT_{W \Rightarrow R}$ der Quellrelation W auf R schon berechnet worden ist. Außerdem werden die Referenztabellen auf die von R referenzierten Relationen $N_1, \ldots, N_k$ berechnet.

Alg. 2 Referenztabellen und Histogramme

```
 1: in R                                          // Eingaberelation
 2: in RT_{W⇒R}                  // Referenztab. der Quellrelation W auf R
 3: in pk, fk_1, ..., fk_k                 // Primär- u. Fremdschlüssel von R
 4: out H_1, ..., H_c                                // Histogramme von R
 5: out RT_{W⇒N_1}, ..., RT_{W⇒N_k}                  // RT auf Nachfolger
 6:
 7: for all t = (t_1, t_2, ..., t_c) ∈ R do                // für jedes Tupel t
 8:   // Anzahl der Referenzen ermitteln
 9:   if R = W then                                        // Quellrelation?
10:     ref := 1                                          // nicht skalieren
11:   else                          // sonst ist Referenztabelle gegeben
12:     ref := RT_{W⇒R}[pk(t)]                             // skalieren
13:   end if
14:
15:   // Referenztabellen aller Nachfolger aktualisieren
16:   for i := 1, 2, ..., k do
17:     RT_{W⇒N_i}[fk_i(t)] += ref                        // Zähler erhöhen
18:   end for
19:
20:   // Histogramme aktualisieren
21:   for i := 1, 2, ..., c do
22:     H_i[t_i] += ref                                   // Zähler erhöhen
23:     if |H_i| ≥ τ then                      // zu viele Ausprägungen?
24:       delete H_i                           // dann Histogramm löschen
25:     end if
26:   end for
27: end for
28:
29: cuthists()                                        // Histogramme kürzen
30: delete RT_{W⇒R}                                   // nicht mehr benötigt
```

5.2 Schlüsselmengen

Nach der Ermittlung und Kürzung der Histogramme ist bereits bekannt, zu welchen Attributen SGTs erzeugt werden müssen. In der zweiten Phase des hierarchischen Small Group Samplings werden Informationen gesammelt, die es ermöglichen, zu jedem Tupel der Quellrelation zu bestimmen, in welche dieser SGTs es aufgenommen werden muss. Dazu werden die gekürzten Histogramme Schritt für Schritt in so genannte *Schlüsselmengen* konvertiert, welche die Primärschlüssel aller Tupel mit selten auftretenden Ausprägungen enthalten.

Die Berechnung der Schlüsselmengen erfolgt Bottom-Up, d.h. es wird mit den Relationen begonnen, welche keine Referenzen besitzen. Die Quellrelation wird dabei ausgenommen. Nur Attribute, die nach der ersten Phase noch ein Histogramm besitzen, es also nicht gelöscht wurde, werden im Folgenden betrachtet. Das Histogramm $H_{R,i}$ des Attributes i der Relation R enthält nur noch selten vorkommende Attributausprägungen. In einem Durchlauf durch R wird es in eine Schlüsselmenge $L^R_{R,i}$ konvertiert. Dazu werden die Primärschlüssel aller Tupel, deren Ausprägung des Attributs i im Histogramm vorkommt, in die Schlüsselmenge aufgenommen. Im Beispielszenario werden SGTs für das Attribut *Ort* der Relation *Standort* (S) und das Attribut *Name* der Relation *Abteilung* erstellt. Abbildung 7 rechts zeigt das Ergebnis der Konvertierung des Histogramms $H_{S,Ort}$ in die Schlüsselmenge $L^S_{S,Ort}$. Das einzige Tupel mit einer seltenen Ausprägung ist (2, *Süd*), die Schlüsselmenge enthält somit nur den Primärschlüssel 2.

Für eine Relation mit Fremdschlüsseln gestaltet sich das Verfahren etwas komplizierter. Zusätzlich zur Berechnung der Schlüsselmengen der Attribute dieser Relation müssen die Schlüsselmengen referenzierter Relationen "geliftet" werden. Die gerade berechnete Schlüsselmenge $L^S_{S,Ort}$ enthält Primärschlüssel der Relation *Standort* und wird auf Primärschlüssel der Relation *Abteilung* konvertiert. Wir bezeichnen die dabei entstehende Menge mit $L^A_{S,Ort}$. Sie wird erstellt, indem für jedes Tupel der Fremdschlüssel auf *Standort* in der alten Schlüsselmenge nachgeschlagen wird, und, falls er darin vorkommt, der Primärschlüssel des Tupels in die neue Schlüsselmenge aufgenommen wird. In Abb. 7 rechts wird deutlich, dass nur das Tupel (4, *Verw.*, 2) einen Eintrag in $L^S_{S,Ort}$ besitzt. Somit wird nur der Primärschlüssel 4 dieses

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

Tupels in die neue Schlüsselmenge $L^{A}_{S,Ort}$ aufgenommen. Simultan dazu wird das Histogramm $H_{A,Name}$ in eine Schlüsselmenge umgewandelt.

Algorithmus 3 beschreibt die Erstellung der Schlüsselmengen einer Relation R mit Referenzen auf die Relationen $N_1, \ldots, N_k$. In den Zeilen 7–12 werden die zu berechnenden Schlüsselmengen angelegt, in den Zeilen 15–26 mit den entsprechenden Primärschlüsseln gefüllt. Nach Beendigung der Abarbeitung der Relation R können ihre Histogramme und die Schlüsselmengen mit Primärschlüsseln der Nachfolgerrelationen entfernt werden.

5.3 Sampling

Das Sampling der Quellrelation erfolgt genau wie beim normalen Small Group Sampling. Für jedes Tupel kann anhand der Schlüsselmengen entschieden werden, ob es in eine Small Group Table aufgenommen werden muss oder nicht. Tritt der Fremdschlüssel $fk_j(t)$ auf die Relation N_j in der Schlüsselmenge $L^{N_j}_{X,i}$ auf, so muss das aktuelle Tupel t in die SGT des Attributs i der Relation X aufgenommen werden. Im Beispielszenario werden bspw. nur Tupel mit der Abteilungsnummer 1 oder 4 in die SGT des Attributes *Abteilung.Name* aufgenommen (siehe Abb. 8). Abschließend müssen das Basissample und die erstellten SGTs mit den Relationen *Abteilung* und *Ort* verbunden werden[2].

Alg. 3 Berechnung der Schlüsselmenge

```
 1: in R                                          // Eingaberelation
 2: in pk, fk_1, ..., fk_k                        // Primär- u. Fremdschlüssel von R
 3: in L^{N_1}_{*,*}, ..., L^{N_k}_{*,*}          // Schlüsselm. der SGTs der Nachf.
 4: in H_1, ..., H_c                              // Histogramme von R
 5: out L^R_{*,*}                                 // Schlüsselm. der SGTs von R und Nachf.
 6:
 7: for all L^N_{X,i} ∈ {Schlüsselmengen der Nachfolger} do
 8:    create L^R_{X,i}                           // neue Schlüsselmenge erzeugen
 9: end for
10: for all H_{R,i} do                            // für jedes (vorhandene) Histogramm
11:    create L^R_{R,i}                           // Schlüsselmenge erzeugen
12: end for
13:
14: // für jedes Tupel der Eingaberelation
15: for all t = (t_1, t_2, ..., t_c) ∈ R do
16:    for all H_{R,i} do                         // für jedes Histogramm
17:       if t_i ∈ H_{R,i} then                   // falls Ausprägung selten
18:          L^R_{R,i} := L^R_{R,i} ∪ {pk(t)}     // P.schlüssel aufnehmen
19:       end if
20:    end for
21:    for all L^{N_j}_{X,i} ∈ {Schlüsselmengen der Nachfolger} do
22:       if fk_j(t) ∈ L^{N_j}_{X,i} then         // F.schl. in alter S.menge
23:          L^R_{X,i} := L^R_{X,i} ∪ {pk(t)}     // P.schlüssel aufnehmen
24:       end if
25:    end for
26: end for
27:
28: // nicht mehr benötigte Datenstrukturen löschen
29: delete H_{R,*}                                // Histogramme
30: delete L^{N_1}_{*,*}, ..., L^{N_k}_{*,*}     // Schlüsselmengen d. Nachfolger
```

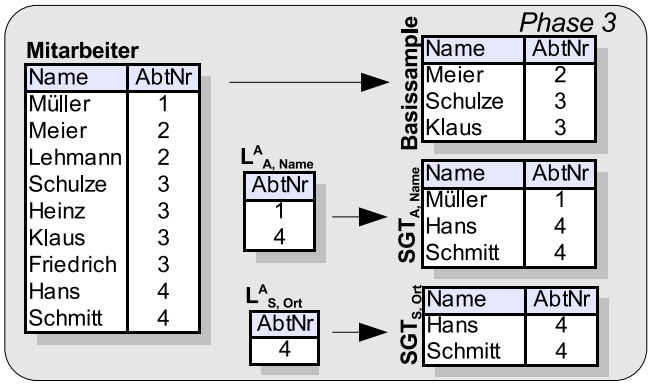


Abb. 8 Phase 3 des hierarchischen Small Group Samplings

6 Evaluation

Die vorgestellten hierarchischen Sampling-Verfahren wurden prototypisch implementiert und mit dem naiven Vorgehen verglichen. Die Datenbank (IBM DB2 UDB v8.1) wurde dabei durch eine Java-Middleware angesprochen, in welcher die Sampling-Verfahren integriert sind. Das naive Verfahren wurde unter Verwendung einer Sicht realisiert, die hierarchischen Verfahren wie in den vorangegangenen Abschnitten beschrieben. Testsystem war ein AMD Athlon™ XP 3000+ mit 2 GB Hauptspeicher. Alle Tests wurden mit dem TPC-D Benchmark [20] und künstlich verzerrten Daten durchgeführt. Die Umfang der in der Benchmark bearbeiteten Daten wird durch einen Skalierungsfaktor ausgedrückt. Die Relationen *Nation* und *Region* wurden ausgeschlossen, da sie nur sehr wenige Tupel enthalten und nicht skalieren. Die Verzerrung der Daten wurde durch eine Zipfverteilung mit Zipffaktor z simuliert. Der Zipffaktor $z = 1$ entspricht einer Gleichverteilung der Daten. Je höher z gewählt wird, desto höher sind die entstehenden Datenverzerrungen.

Für die Messungen des hierarchischen Senate Samplings wurde nach den Attributen *Customer.Nationkey* und *Part.Type* gruppiert. Die Samplingrate betrug 5%, der Zipffaktor wurde auf 1.5 festgesetzt. Abbildung 9 vergleicht die für die Erstellung des Samples benötigte Zeit sowie den Hauptspeicherbedarf. Das hierarchische Verfahren benötigt für große Datenmengen ca. 40% der Zeit des naiven Ansatzes. Der Hauptspeicherbedarf ist bei beiden Vorgehensweisen annähernd identisch, obwohl das hierarchische Verfahren zusätzliche Datenstrukturen benötigt. Der Grund dafür liegt in der unterschiedlichen Belegung der temporären Reservoirs im Hauptspeicher: einerseits bestehen sie nur aus Tupeln der Quellrelation, andererseits werden diese Tupel

[2] Das ist hier nicht dargestellt. Da ein Tupel in mehreren Stichprobentabellen auftreten kann, ist es sinnvoll, den Verbund schon vor dem Ausschreiben zu berechnen, um unnötigen Aufwand einzusparen.

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

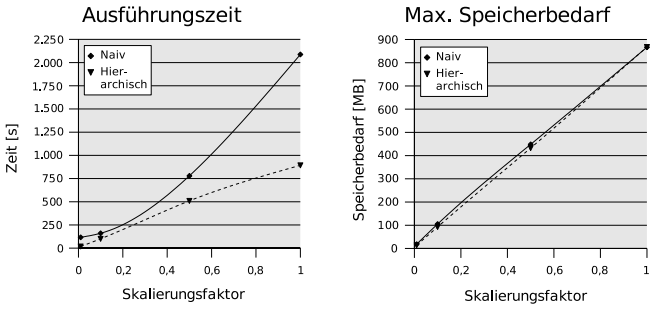


Abb. 9 Erstellungsaufwand in Abh. der Datenbankgröße

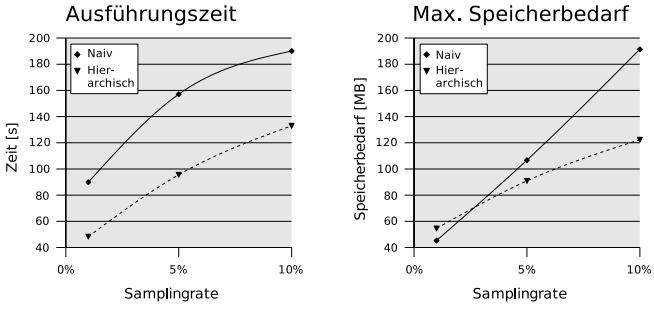


Abb. 10 Erstellungsaufwand in Abh. der Samplingrate

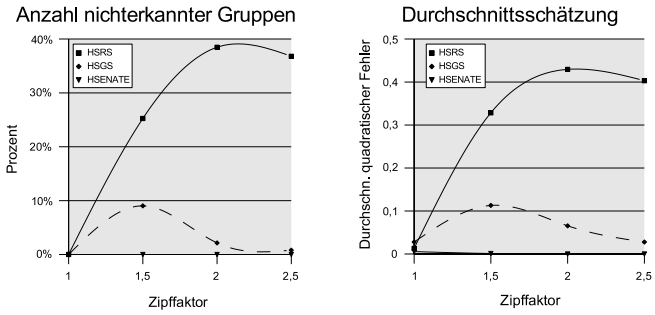


Abb. 11 Genauigkeit in Abhängigkeit der Datenverzerrung

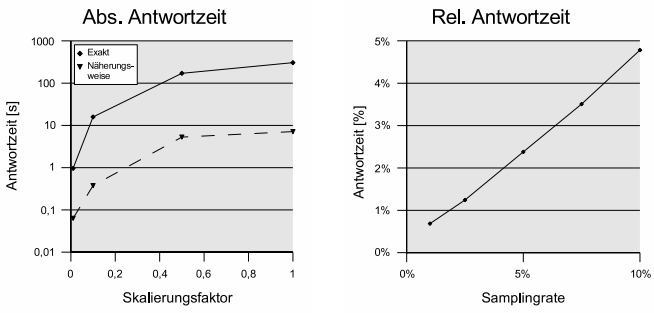


Abb. 12 Geschwindigkeitsgewinn durch näherungsweise Anfragebeantwortung

vorher schon mit allen referenzierten Relationen verbunden und somit deutlich größer.

Abbildung 10 verdeutlicht den Einfluss der Samplingrate auf die Ausführungszeit und den Speicherbedarf für die Erstellung des hierarchischen Senate Samples. Dabei wurde der Skalierungsfaktor 0.1 verwendet. Das hierarchische Verfahren beschleunigt das naive Verfahren um einen konstanten absoluten Anteil, d.h. er ist weitgehend unabhängig von der Samplegröße. Allerdings benötigt das naive Verfahren deutlich mehr Hauptspeicher für zunehmende Samplingraten.

Dieselben Messungen wurden auch für das hierarchische Small Group Sampling durchgeführt. Dabei entstanden tendenziell dieselben Ergebnisse. Einziger Unterschied war, dass das Small Group Sampling deutlich weniger Hauptspeicher, dafür aber deutlich mehr Rechenzeit benötigt. Aus Platzgründen sind diese Messergebnisse hier nicht dargestellt.

Wie in Abb. 11 (links) sichtbar wird, verringert gruppenbasiertes Sampling den Anteil der nichterkannten Gruppen in erheblichem Maße. Für die Messungen wurde nach *Customer.Nationkey* und *Supplier.Nationkey* gruppiert, der Skalierungsfaktor auf 0.1 festgesetzt und die Metriken aus [3] verwendet. Bei einer Gleichverteilung der Daten (Zipffaktor 1) sind alle drei Sampling-Verfahren gleichwertig. Mit steigendem Zipffaktor verlieren einfache Join Synopsen (HSRS) nahezu alle kleinen Gruppen, während das hierarchische Small Group Sampling (HSGS) nur wenige verpasst. Je höher die Datenverzerrung, desto weniger mittelgroße Gruppen gibt es und desto besser bewährt sich das Small Group Sampling. Das hierarchische Senate Sampling (HSENATE) beinhaltet schließlich alle auftretenden Gruppen. Es zieht Nutzen daraus, dass es die Gruppierungsattribute im Voraus kennt. Die Genauigkeit der verschiedenen Verfahren (s. Abb. 11 rechts) lässt sich ähnlich bewerten. Es wurde dabei pro Gruppe der durchschnittliche Umsatz abzüglich des Rabattes berechnet und dessen durchschnittlicher quadratischer Fehler dargestellt.

Abbildung 12 (links) stellt den Geschwindigkeitszuwachs durch näherungsweise Anfragebeantwortung mit einer Stichprobengröße von 5% dar (logarithmische Skaleneinteilung). Die Antwortzeit ist für Join Synopsen und die hierarchischen gruppenbasierten Verfahren identisch und beträgt im Mittel etwa 2.5% der benötigten Zeit zur exakten Beantwortung der Anfrage. Wie in Abb. 12 (rechts) ersichtlich, besitzt die Samplingrate einen linearen Einfluss auf die Antwortzeit einer näherungsweisen Anfrage relativ zur der einer exakten Anfrage. Es wurde dabei der Skalierungsfaktor 0.1 verwendet.

7 Ausblick

Mit den vorgestellten Lösungsansätzen lassen sich gruppenbasierte Stichproben effizient über mehrere Relationen berechnen. Mit ihnen können alle Anfragen beantwortet werden, in denen die Quellrelation und durch Fremdschlüsselbeziehungen referenzierte Relationen verbunden werden. Dazu ist es nicht notwendig, auf die Originaldaten zuzugreifen, die gesamte benötigte Information befindet sich in der Stichprobe. Anfragen, die Relationen allerdings auf andere Art und Weise miteinander verbinden, erfordern ein angepasstes Vorgehen. Beispielsweise könnte eine Stichprobe für einen beliebigen Gleichheitsverbund erstellt werden. Dazu müssen

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

die hier vorgestellten Verfahren allerdings entsprechend erweitert werden, da es sich dann nicht mehr um N:1-, sondern um M:N-Beziehungen handeln kann. Ob dies sinnvoll möglich ist oder nicht wurde bisher noch nicht untersucht.

Daten in einer Datenbank sind typischerweise nicht starr, sondern ändern sich ständig. Die Aktualisierung der Stichprobe ist eine wichtige Voraussetzung für die Praktikabilität der vorgestellten Lösungen. Die Autoren untersuchten verschiedene Möglichkeiten, die gruppenbasierten hierarchischen Stichproben zu aktualisieren, ohne sie komplett neu zu erstellen. Je genauer die Stichprobe den aktuellen Datenbestand widerspiegelt, desto mehr Aufwand ist für ihre Wartung notwendig. Dieses interessante Gebiet wurde aus Platzgründen nicht mit in diesen Artikel aufgenommen.

Die beiden vorgestellten Verfahren Senate Sampling und Small Group Sampling lassen sich parametrieren. So kann z.B. die Größe der Stichprobe festgelegt werden. Die optimale Parameterwahl im Fall einer Relation wurde in [1,3] betrachtet. Für den Fall mehrerer Relationen ist sie allerdings noch offen, kann aber möglicherweise direkt übernommen werden, da der maximale Fremdschlüsselverbund auch eine Relation darstellt. Das trifft ebenso für die Fehlerbetrachtung zu, welche dem Nutzer die Qualität des Ergebnisses beschreibt.

Nichtzuletzt können die hier vorgestellten Ansätze auch für andere Sampling-Verfahren verwendet werden. So können bspw. mit Hilfe der Referenztabellen die Ausreißer des Outlier Indexing bestimmt werden. Die Betrachtung weiterer Sampling-Verfahren im Fall mehrerer Relationen und eine mögliche Verallgemeinerung der gefundenen Resultate muss noch näher untersucht werden.

Literatur

1. Acharya S, Gibbons PB, Poosala V (2000) Congressional Samples for Approximate Answering of Group-By Queries. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp 487–498
2. Acharya S, Gibbons PB, Poosala V, Ramaswamy S (1999) Join synopses for approximate query answering. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, pp 275–286
3. Babcock B, Chaudhuri S, Das G (2003) Dynamic sample selection for approximate query processing. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp 539–550
4. Barbará D, DuMouchel W, Faloutsos C, Haas PJ, Hellerstein JM, Ioannidis YE, Jagadish HV, Johnson T, Ng RT, Poosala V, Ross KA, Sevcik KC (1997) The New Jersey Data Reduction Report. IEEE Data Eng Bull 20(4):3–45
5. Chakrabarti K, Garofalakis MN, Rastogi R, Shim K (2000) Approximate Query Processing Using Wavelets. In: Proceedings of 26th International Conference on Very Large Data Bases, VLDB 2000, pp 111–122
6. Chaudhuri S, Das G, Datar M, Motwani R, Narasayya VR (2001) Overcoming Limitations of Sampling for Aggregation Queries. In: 17th International Conference on Data Engineering (ICDE' 01), pp 534–544, April
7. Chaudhuri S, Das G, Srivastava U (2004) Effective use of block-level sampling in statistics estimation. In: Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM Press, pp 287–298
8. Chaudhuri S, Motwani R, Narasayya V (1999) On Random Sampling over Joins. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, pp 263–274
9. Ganti V, Lee M-L, Ramakrishnan R (2000) ICICLES: Self-Tuning Samples for Approximate Query Answering. In: The VLDB Journal, pp 176–187
10. Gibbons PB, Matias Y (1998) New sampling-based summary statistics for improving approximate query answers. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data. ACM Press, pp 331–342
11. Gryz J, Guo J, Liu L, Zuzarte C (2004) Query sampling in DB2 Universal Database. In: Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM Press, pp 839–843
12. Haas PJ, Hellerstein JM (1999) Ripple Joins for Online Aggregation. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, pp 287–298
13. Haas PJ, König C (2004) A bi-level Bernoulli scheme for database sampling. In: Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM Press, pp 275–286
14. Hellerstein JM, Haas PJ, Wang HJ (1997) Online Aggregation. In: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, pp 171–182
15. Ioannidis YE, Poosala V (1995) Histogram-Based Solutions to Diverse Database Estimation Problems. Data Engineering Bulletin 18(3):10–18
16. Jermaine C (2003) Robust Estimation With Sampling and Approximate Pre-Aggregation. In: Proceedings of 29th International Conference on Very Large Data Bases, VLDB 2003, September 9–12, 2003, Berlin, Germany, Los Altos, CA 94022, USA, 2003. Morgan Kaufmann Publishers, pp 886–897
17. Jermaine C, Pol A, Arumugam S (2004) Online maintenance of very large random samples. In: Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM Press, pp 299–310
18. Olken F (1993) Random Sampling from Databases. Thesis LBL-32883, Information and Computing Sciences Division, Lawrence Berkeley National Laboratory, Mailstop 50B-3238, 1 Cyclotron Road, Berkeley, California 94720, USA
19. Poosala V, Ganti V, Ioannidis YE (1999) Approximate Query Answering using Histograms. IEEE Data Eng Bull 22(4):5–14
20. Transaction Processing Performace Council (1998) TPC-D Benchmark Version 2.1, February. `http://www.tpc.org`
21. Vitter JS (1985) Random Sampling with a Reservoir. ACM Transactions on Mathematical Software 11(1):37–57, March
22. Vitter JS, Wang M (1999) Approximate computation of multidimensional aggregates of sparse data using wavelets. In: Proceedings of the 1999 ACM SIGMOD international conference on Management of data. ACM Press, pp 193–204

Rainer Gemulla hat von 1999 bis 2004 Informatik an der TU Dresden studiert. Er ist wissenschaftlicher Mitarbeiter am Lehrstuhl für Datenbanken unter der Leitung von Prof. Lehner und beschäftigt sich mit Stichprobenerhebungen in Datenbanksystemen.

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "Informatik, Forschung und Entwicklung" 20 (1), S. 45-56. ISSN: 0949-2925
https://doi.org/10.1007/s00450-004-0175-3

Henrike Berthold studierte an der TU Dresden Informatik. Danach war sie DFG-Stipendiatin und wiss. Mitarbeiterin in der Arbeitsgruppe von Prof. Meyer-Wegener an dieser Universität. Im Jahr 2002 promovierte sie auf dem Gebiet der multimedialen Datenbanken. Seit 2004 beschäftigt sie sich in der Datenbankgruppe von Prof. Lehner an der TU Dresden mit der näherungsweisen Auswertung von Datenbankanfragen.

Prof. Dr.-Ing. Wolfgang Lehner (geb. 1969) studierte Informatik an der Universität Erlangen-Nürnberg. Von 1995 bis 1998 war er dort als wiss. Mitarbeiter am Lehrstuhl für Datenbanksysteme beschäftigt und promovierte 1998. Nach seinem Aufenthalt am IBM Almaden Research Center, San Jose (CA), USA, arbeitete er als wiss. Assistent und schloss diese Tätigkeit mit der Habilitation ab. Seit Oktober 2002 ist er Inhaber des Lehrstuhls für Datenbanken an der Technischen Universität Dresden. Sein Forschungsschwerpunkt liegt in der Entwicklung von Architekturen von Datenbank- und Informationssystemen zur Unterstützung datenintensiver Anwendungen und Prozesse.

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden