

Diplomarbeit Nr. 2178

# Bewertung von Kosten- und Kardinalitätsschätzungen

Thilo Marquardt

<b>Studiengang:</b>	Informatik
<b>Prüfer:</b>	Prof. Dr. Bernhard Mitschang
<b>Betreuer:</b>	Dr. Holger Schwarz
<b>begonnen am:</b>	18. März 2004
<b>beendet am:</b>	17. September 2004
<b>CR-Klassifikation:</b>	H.2.4

Institut für Parallele und  
Verteilte Systeme (IPVS)  
Universität Stuttgart  
Universitätsstraße 38  
D-70569 Stuttgart

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>6</b>
1.1	Inhalt und Ziel der Arbeit . . . . .	6
1.2	Gliederung . . . . .	7
<b>2</b>	<b>Grundlagen</b>	<b>8</b>
2.1	Kostenbasierte Anfrageoptimierung . . . . .	8
2.2	Kosten- und Kardinalitätsschätzungen . . . . .	9
2.3	Statistiken und Histogramme . . . . .	12
<b>3</b>	<b>Einflussparameter</b>	<b>16</b>
3.1	Qualität der Statistiken . . . . .	16
3.2	Struktur der Daten . . . . .	18
3.2.1	Datenverteilung . . . . .	18
3.2.2	Datenabhängigkeit . . . . .	20
3.3	Anfragespezifische Parameter . . . . .	22
3.3.1	Beteiligte Relationen und Sichten . . . . .	22
3.3.2	Logische Operatoren . . . . .	23
3.3.3	Prädikate mit Attributarithmetik . . . . .	25
3.3.4	Spezielle Konzepte in SQL:1999 . . . . .	25
3.4	Ausführungsplanspezifische Parameter . . . . .	26
3.4.1	Struktur des Ausführungsplans . . . . .	26
3.4.2	Planoperatoren und deren Realisierungen . . . . .	28
3.5	Laufzeitparameter . . . . .	30
3.5.1	Verfügbarkeit der Ressourcen . . . . .	30
3.5.2	Laufzeitbindungen von Host-Variablen . . . . .	31
<b>4</b>	<b>Vorhandene Bewertungsansätze</b>	<b>32</b>
4.1	Klassifikation der Bewertungsansätze . . . . .	32
4.2	Statische, qualitative Bewertung . . . . .	33
4.2.1	Statistiken für Zwischenergebnisse . . . . .	34
4.2.2	Bewertung durch Ungenauigkeitspotenzial . . . . .	35
4.3	Statische, quantitative Bewertung . . . . .	36
4.3.1	Maximale Abweichung pro Bucket . . . . .	36
4.3.2	Maximale Abweichung pro Histogramm . . . . .	37
4.4	Dynamische Bewertung . . . . .	38
4.4.1	LEO – LEarning Optimizer von DB2 . . . . .	38

4.4.2	Verzögertes Lernen . . . . .	38
4.4.3	Sofortiges Lernen . . . . .	39
<b>5</b>	<b>Messungen</b>	<b>41</b>
5.1	Beschreibung der Messungen . . . . .	41
5.1.1	Messumgebung . . . . .	41
5.1.2	Definition der Anfragen . . . . .	45
5.1.3	Ablauf der Messungen . . . . .	47
5.2	Messergebnisse . . . . .	49
5.2.1	Überblick . . . . .	49
5.2.2	Aggregation . . . . .	54
5.2.3	Datenabhängigkeiten . . . . .	55
5.2.4	Semantik innerhalb der Anfrage . . . . .	56
5.2.5	Standardwert für Schätzung . . . . .	56
5.2.6	Stringvergleich . . . . .	57
5.2.7	Temporäre Tabellen . . . . .	60
5.2.8	Anti-Verbund . . . . .	62
5.2.9	Überblick über die Abfragesequenzen . . . . .	63
<b>6</b>	<b>Ergebnisse</b>	<b>66</b>
6.1	Interpretation der Messergebnisse . . . . .	66
6.2	Beurteilung der Bewertungsansätze . . . . .	69
6.2.1	Beurteilung der quantitativen Ansätze . . . . .	69
6.2.2	Beurteilung der qualitativen Ansätze . . . . .	70
6.2.3	Erweiterung der qualitativen Bewertung . . . . .	72
6.3	Anwendungsszenarien . . . . .	75
<b>7</b>	<b>Schluss</b>	<b>78</b>
7.1	Zusammenfassung . . . . .	78
7.2	Ausblick . . . . .	79
	<b>Literaturverzeichnis</b>	<b>81</b>
	<b>Anhang A: Anfragen</b>	<b>86</b>

# Abbildungsverzeichnis

2.1	Anfrageverarbeitung . . . . .	8
2.2	Equi-Width-Histogramm . . . . .	14
2.3	Equi-Depth-Histogramm . . . . .	14
3.1	Verwendung von Statistiken bei Ausführungsplänen . . . . .	17
3.2	Statistiken und Qualität der Schätzwerte . . . . .	18
3.3	Equi-Depth-Histogramm mit Datenverteilung . . . . .	20
3.4	Plan mit Ausführungsreihenfolge 1 . . . . .	27
3.5	Plan mit Ausführungsreihenfolge 2 . . . . .	28
3.6	Anfragegraph mit logischen Operatoren . . . . .	29
3.7	Ausführungsplan mit Planoperatoren . . . . .	30
4.1	Klassifikation der Bewertungsansätze . . . . .	33
4.2	Bewertung von Schätzergebnissen in „Statistics on Views“ . . . . .	34
4.3	LEO’s Feedbackschleife . . . . .	39
5.1	Schema des TPC Benchmark H . . . . .	44
5.2	Durchführung der Messungen . . . . .	48
5.3	Beispiel einer Planauswertung . . . . .	49
5.4	Schätzfehler des Optimierers . . . . .	53
5.5	Ausführungsplan zu Anfrage Q <sub>t</sub> 05 . . . . .	55
5.6	Ausführungsplan zu Anfrage Q <sub>e</sub> 12 . . . . .	59
5.7	Ausführungsplan zu Anfrage Q7 . . . . .	61
5.8	Ausführungsplan zu Anfrage Q <sub>e</sub> 25 . . . . .	63
5.9	Ausführungspläne zu den Anfragen der Sequenz Q <sub>o</sub> a1_seq . . . . .	65
6.1	Ausführungsplan zu Anfrage Q <sub>t</sub> 03 . . . . .	70
6.2	Optimale Abbildung von Schätzfehler auf UP . . . . .	75

# Tabellenverzeichnis

3.1	Beispiel für die Parameter einer Datenverteilung . . . . .	19
5.1	Eigenschaften der Anfragen . . . . .	46
5.2	Übersicht der Messergebnisse . . . . .	53
5.3	DB2-Filterfaktoren bei fehlenden Statistiken . . . . .	57
5.4	Kardinalitätsschätzungen des LIKE-Prädikats . . . . .	59
5.5	Messergebnisse der Anfragesequenzen . . . . .	64
6.1	Ungenauigkeitspotenzial und Schätzfehler der Anfragen . . . . .	74
6.2	Mediane der Ungenauigkeitspotenziale . . . . .	74
6.3	Bewertete Kosten $\tilde{K}$ für $s = 3$ . . . . .	77

# Abkürzungsverzeichnis

DBS	<u>D</u> aten <u>b</u> ankmanagementsystem
LEO	<u>L</u> earning <u>O</u> ptimizer for DB2
OLAP	<u>O</u> nline <u>A</u> nalytical <u>P</u> rocessing
ORBIT	<u>O</u> ptimization and Integration of <u>B</u> usiness <u>I</u> ntelligence <u>T</u> echnology
SF	<u>S</u> kalierungsfaktor
TPC	<u>T</u> ransaction <u>P</u> rocessing Performance <u>C</u> ouncil
TPC-H	<u>T</u> PC Benchmark <u>H</u>
TPCH1	Datenbasis <u>1</u> des <u>T</u> PC- <u>H</u> mit einem Datenvolumen von ca. 100 MB
TPCH2	Datenbasis <u>2</u> des <u>T</u> PC- <u>H</u> mit einem Datenvolumen von ca. 1 GB
TPCH3	Datenbasis <u>3</u> des <u>T</u> PC- <u>H</u> mit einem Datenvolumen von ca. 3 GB
TPCH4	Datenbasis <u>4</u> des <u>T</u> PC- <u>H</u> mit einem Datenvolumen von ca. 10 GB
UP	<u>U</u> ngenauigkeitspotenzial

# 1 Einleitung

Anfragen an ein Datenbanksystem werden in deklarativen Anfragesprachen formuliert. Der Benutzer beschreibt in seiner Anfrage, welche Daten er vom System anfordert. Auf welche Weise diese Daten ermittelt werden, bleibt ihm jedoch verborgen. Der Anfrageoptimierer des Datenbanksystems übernimmt diese komplexe Aufgabe. Zu einer gegebenen Anfrage bestimmt er einen möglichst optimalen Ausführungsplan, der exakt die angeforderten Daten zurückliefert. Wesentlicher Einflussfaktor für die Wahl eines optimalen Plans ist die Qualität der Kosten- und Kardinalitätsschätzungen, die für die Bewertung alternativer Pläne innerhalb des Optimierungsprozesses durchgeführt werden müssen.

## 1.1 Inhalt und Ziel der Arbeit

Die vorliegende Diplomarbeit beschäftigt sich mit der Bewertung von Kosten- und Kardinalitätsschätzungen im Rahmen der Anfrageoptimierung. Auf dem Gebiet der Schätzverfahren in Datenbanksystemen wurde in den letzten beiden Jahrzehnten umfassend geforscht. Zum heutigen Zeitpunkt existiert eine Vielzahl an Arbeiten, die unterschiedliche Ansätze für Schätzungen verfolgen. Die Zahl der im praktischen Einsatz befindlichen Verfahren ist jedoch beschränkt. Es haben sich hier die Histogramme als Vertreter der nicht parametrischen Schätzverfahren durchgesetzt. Deshalb stehen sie in dieser Arbeit im Mittelpunkt der Betrachtung.

Datenbanksysteme verwenden Kostenformeln für die einzelnen Operatoren, um die Kosten und die Kardinalität des Ergebnisses einer Anfrage abzuschätzen. Als Basis dafür dienen Statistiken und Histogramme, die Informationen über die Datenbasis bereitstellen. Die Schätzwerte, die ein Datenbanksystem liefert, sind von unterschiedlicher Qualität. Schätzfehler können im Ausführungsplan über mehrere Stufen propagiert werden, was zu Kosten- und Kardinalitätsschätzungen führen kann, die um Größenordnungen von den tatsächlichen Werten abweichen. Verantwortlich dafür sind verschiedene Einflussfaktoren, wie beispielsweise die Komplexität der Anfrage, die Struktur des Ausführungsplans oder Eigenschaften der Daten, die das Schätzergebnis unterschiedlich stark beeinflussen. Ein erstes Ziel dieser Arbeit ist die Beschreibung dieser Parameter. Dabei wird erläutert, welchen Einfluss sie auf das Schätzergebnis haben können.

Darauf aufbauend beschäftigt sich diese Arbeit mit Ansätzen zur Bewertung der Verlässlichkeit von Kosten- und Kardinalitätsschätzungen. Wie kann unter Berücksichtigung der Einflussfaktoren eine Aussage über das Fehlerpotenzial der Schätzwerte gemacht werden? Welche qualitativen und quantitativen Aussagen sind möglich? Antworten auf diese Fragen sind ein weiteres Ziel dieser Arbeit. Ausgangspunkt hierfür sind die wenigen vorhandenen Bewertungsansätze aus verwandten Arbeiten, die Aussagen über Qualität und Genauigkeit von Schätzwerten liefern. Diese theoretischen Betrachtungen werden durch umfangreiche Messungen überprüft. Dazu werden die verschiedenen Einflussfaktoren variiert und die vom Optimierer geschätzten Werte zu Kardinalität und Ausführungszeit ebenso wie die tatsächlichen Werte systematisch

erfasst. Die Interpretation der Messergebnisse dient dazu, den Zusammenhang zwischen Einflussfaktoren und Qualität der Schätzwerte herzustellen. Mit den durch die Messungen gewonnenen Erkenntnissen werden die vorgestellten Bewertungsansätze beurteilt und erweitert. Abschließend werden mögliche Anwendungsszenarien für die Bewertung von Kosten- und Kardinalitätsschätzungen vorgestellt.

## 1.2 Gliederung

Im einführenden **Kapitel 2** soll ein grundlegendes Verständnis für den Prozess der Anfrageoptimierung in Datenbanksystemen und die dafür notwendigen Kosten- und Kardinalitätsschätzungen gewonnen werden.

**Kapitel 3** beschreibt auf einer abstrakten Ebene mögliche Einflussparameter, die sich auf die Qualität der Schätzergebnisse auswirken.

Die vorhandenen Ansätze aus der Literatur zur Bewertung von Kosten- und Kardinalitätsschätzungen werden in **Kapitel 4** nach verschiedenen Kriterien klassifiziert und vorgestellt. Im Mittelpunkt stehen qualitative Aussagen, die bereits zum Übersetzungszeitpunkt möglich sind.

Die Messungen in **Kapitel 5** beschreiben die empirische Herangehensweise an das Thema. Das theoretische Verständnis über den Zusammenhang zwischen Einflussfaktoren und Qualität der Schätzergebnisse wird experimentell überprüft.

**Kapitel 6** enthält die Ergebnisse dieser Arbeit. Die durch die Messungen gewonnenen Erkenntnisse werden dazu verwendet, die theoretischen Überlegungen zu den Einflussparametern aus Kapitel 3 zu bewerten. Außerdem ermöglichen sie eine Beurteilung der Bewertungsansätze aus Kapitel 4. Mögliche Anwendungsgebiete für Verfahren zur Bewertung von Kosten- und Kardinalitätsschätzungen werden vorgestellt.

Abschließend fasst **Kapitel 7** die wichtigsten Punkte dieser Arbeit zusammen und gibt Ansatzpunkte für weitere Forschungsarbeiten.

## 2 Grundlagen

Das Kapitel dient als Hinführung zum Thema. Einleitend werden der Ablauf der Anfrageoptimierung in einem *Datenbanksystem* (DBS) und die dabei beteiligten Komponenten beschrieben. Die Bedeutung der Kosten- und Kardinalitätsschätzungen innerhalb der Optimierung wird herausgestellt. Es wird ein Überblick gegeben über existierende Schätzverfahren, die hier zum Einsatz kommen können. Als wichtigster Vertreter hiervon werden die Histogramme genauer vorgestellt. Während dieser einführenden Betrachtungen werden wichtige Begrifflichkeiten, die im weiteren Verlauf der Arbeit verwendet werden, erklärt.

### 2.1 Kostenbasierte Anfrageoptimierung

Der Ablauf der Anfrageverarbeitung lässt sich in verschiedene Phasen unterteilen [Mit95]. Abbildung 2.1 zeigt, welche Verarbeitungsschritte notwendig sind, um von der deklarativen Anfrage des Benutzers zu dem gewünschten Anfrageergebnis zu gelangen.

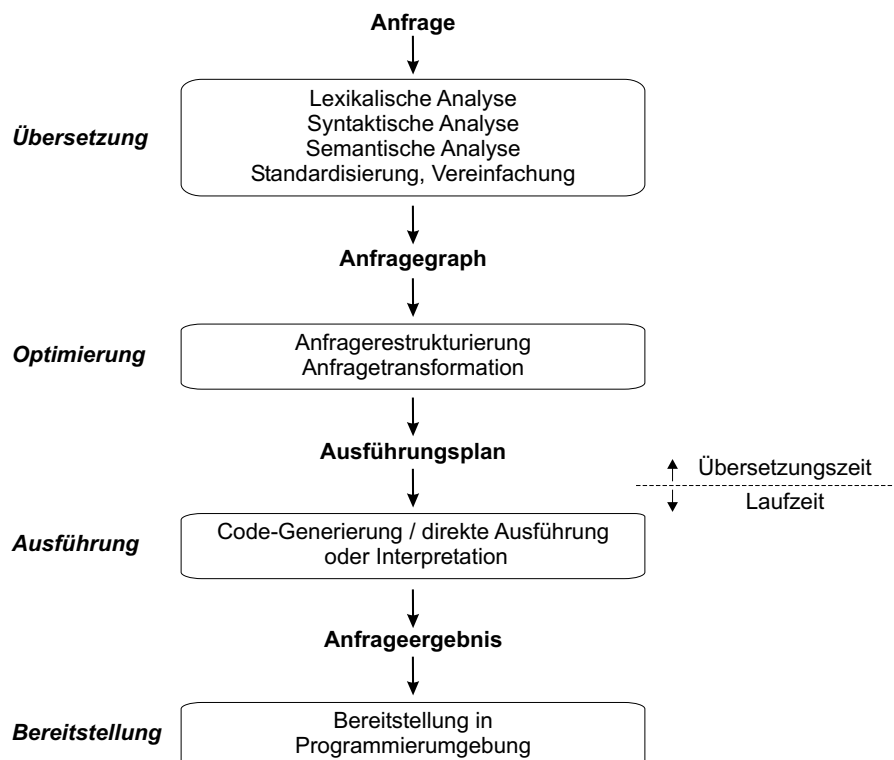


Abbildung 2.1: Anfrageverarbeitung

Die Anfrage wird in einem ersten Schritt in eine Interndarstellung überführt. Verschiedene Darstellungen sind aus der Literatur bekannt [HR01]. Als Darstellungsschema wird hier das Anfragegraphmodell verwendet. Es handelt sich dabei um einen Operatorgraph. Die Knoten des Graphen sind die Operatoren, die Kanten beschreiben den Datenfluss. Man spricht auch von *Objektstrom*. Jeder Operator innerhalb eines Operatorgraphen produziert ausgehend von einem Eingabestrom  $E_i$  einen Ausgabestrom  $A_i$ . Der Anfragegraph ist die zentrale Datenstruktur für den Optimierer.

Die Optimierungskomponente generiert durch Transformation des Anfragegraphen mehrere Ausführungspläne. Bei vielen Datenbanksystemen (z. B. Microsoft SQL-Server, Tandem's NonStopSQL oder IBM's DB2) geschieht dies *Bottom-Up* durch die Generierung von Teilplänen [WG00]. Die logischen Operatoren des Anfragegraphen werden dabei durch physische Operatoren – auch Planoperatoren genannt – ersetzt. Beispielsweise können die beiden logischen Operatoren Projektion und Selektion in einen physischen Operator Access transformiert werden, der durch Attributprojektionslisten und Selektionsprädikatslisten parametrisiert ist. Auf Basis der Planoperatoren werden konkrete Methoden zur Realisierung ausgewählt. Durch die Variation der Operationsreihenfolge und die Wahl verschiedener Implementierungen für einen Planoperator entstehen alternative Pläne, aus denen der kostengünstigste auszuwählen ist. Es können nicht alle möglichen Pläne untersucht werden, da der Suchraum bei komplexen Anfragen sehr groß werden kann (z. B.  $10^{70}$  Ausführungspläne bei einer Anfrage mit 15 Verbunden [SI93]). Es bedarf deshalb einer Suchstrategie, um auf effiziente Weise zu einem möglichst optimalen Plan zu gelangen. Verschiedene Ansätze hierzu finden sich bei Mitschang [Mit95]. Der Optimierer bewertet die generierten Pläne bezüglich der zu erwartenden Ausführungskosten. Es werden dazu Kostenmodelle verwendet, die unterschiedliche Parameter für die Schätzung berücksichtigen. Man spricht deshalb auch von kostenbasierter Anfrageoptimierung und kostenbasierter Planauswahl [GUW02, SH99]. Der gemäß den geschätzten Kosten günstigste Plan ist das Ergebnis der Optimierung.

Die Ausführungskomponente bekommt als Eingabe den optimierten Ausführungsplan. Zur Berechnung des Anfrageergebnisses existieren zwei Ausführungsmöglichkeiten. Der vorliegende Plan wird entweder interpretiert oder es wird eine Code-Generierung durchgeführt, um eine mehrfache Interpretation bei wiederholt auszuführenden Anfragen zu vermeiden. Das Ergebnis der Anfrage wird in der Programmierumgebung bereitgestellt.

## 2.2 Kosten- und Kardinalitätsschätzungen

Elmasri, Härder u. a. unterscheiden folgende Kostenarten, die in die Berechnung einfließen [EN89, HR01]:

- E/A-Kosten (Anzahl physischer Referenzen)
- Berechnungskosten (CPU)
- Kommunikationskosten (Anzahl der übertragenen Daten)
- Speicherkosten (temporäre Belegung des Datenbankpuffers)

Kommunikationskosten kommen hauptsächlich in verteilten Systemen zum Tragen. Ebenso spielen die Speicherkosten bei immer größeren Datenbankpuffern eine untergeordnete Rolle.

In kommerziellen, zentralisierten DBS werden Kostenmodelle als gewichtete Funktion zwischen E/A-Kosten und Berechnungskosten verwendet [HR01]:

$$K = \text{Anzahl der physischen Seitenzugriffe} + W \cdot (\text{Anzahl der Aufrufe des Zugriffssystems})$$

Über den Faktor  $W$  kann das System konfiguriert werden. Ein relativ großes  $W$  wird für Systeme gewählt, bei denen die Ressource CPU beschränkt ist. Rechenintensive Ausführungspläne werden damit benachteiligt. Die Wahl eines kleinen Wertes für  $W$  beeinflusst die Kostenschätzungen derart, dass E/A-intensive Lösungen nach Möglichkeit vermieden werden.

Ausgehend von den Blättern des Ausführungsplans (Relationen) bis zur Wurzel (Anfrageergebnis) werden für jeden Knoten die zu erwartenden Kosten schrittweise bestimmt. Eine präzise Kostenschätzung ist nur dann möglich, wenn die Parameter, die in das Kostenmodell einfließen, genau geschätzt werden können. Bei E/A-Kosten ist die Kardinalität von Bedeutung. Die Kardinalität  $Card$  bezeichnet die Anzahl der Tupel einer Relation, einer Sicht oder eines Zwischenergebnisses innerhalb des Ausführungsplans. Große Objektströme zwischen Knoten resultieren bei der Ausführung in einer großen Anzahl zeitintensiver Sekundärspeicherzugriffe. Garcia-Molina u. a. beschreiben eine Heuristik, die besagt, dass ein Plan gegenüber einem anderen als besser zu erachten ist, wenn die Summe der Größen seiner Zwischenergebnisse kleiner ist als bei einem alternativen Plan [GUW02]. Pläne mit großen Zwischenergebnissen haben viele E/A-Zugriffe zur Folge und führen somit zu einer schlechteren Ausführungszeit als Pläne mit kleineren Zwischenergebnissen. Markl u. a. messen anhand ihrer praktischen Erfahrung mit DB2 den Kardinalitäten einen ähnlich hohen Stellenwert bei. Sie behaupten, dass Kostenschätzungen auf Basis korrekter Kardinalitäten abhängig von den anderen geschätzten Parametern nur um 10–15 % von den tatsächlichen Kosten abweichen können. Fehler in den Kardinalitätsschätzungen jedoch können zu Kostenschätzungen führen, die um Größenordnungen danebenliegen [MLR03].

Direkten Einfluss auf die Kardinalität der Knoten hat die Selektivität eines an den Operator geknüpften Prädikatterms. Die Selektivität  $s(p)$  eines Prädikats  $p$  beschreibt den erwarteten Anteil an Elementen, die diese Bedingung erfüllen [Mit95]. Für die Selektion und den Verbund ist sie wie folgt definiert<sup>1</sup>:

- Selektion mit Bedingung  $p$ :

$$s(p) := \frac{|\sigma_p(R)|}{|R|} = \frac{Card(\sigma_p(R))}{Card(R)}$$

- Verbund von  $R$  mit  $S$ :

$$s(RS) := \frac{|R \bowtie S|}{|R \times S|} = \frac{|R \bowtie S|}{|R| \cdot |S|} = \frac{Card(R \bowtie S)}{Card(R) \cdot Card(S)}$$

Der Einfluss der Berechnungszeit (CPU-Kosten) auf das Schätzergebnis wird in dieser Arbeit vernachlässigt. Das Interesse gilt den E/A-Kosten und somit den Kardinalitäten. In einfachen Kostenmodellen werden nur E/A-Kosten berücksichtigt [HR01]. Die CPU-Kosten ermöglichen

---

<sup>1</sup> $Card(R)$  und  $|R|$  sind äquivalente Darstellungsformen. Härder und Mitschang verwenden  $Card(R)$ , während bei Kemper und Eickler beispielsweise Betragsstriche zum Einsatz kommen [Mit95, HR01, KE01].

eine noch detailliertere Schätzung, wirken sich aber nicht in dem Maße aus, wie das bei E/A-Kosten der Fall ist.

Da der Fokus dieser Arbeit auf der Bewertung solcher Schätzungen liegt und Aussagen über die Qualität der Schätzergebnisse gesucht werden, wird einleitend ein Überblick über existierende Verfahren im Bereich der Kosten- und Kardinalitätsschätzungen gegeben. Man findet in der Literatur unterschiedliche Klassifikationen vorhandener Verfahren. Auch existieren einige vor allem neue Ansätze, die sich nicht eindeutig einer Kategorie zuordnen lassen. Über folgende Einteilung sind sich jedoch die meisten Arbeiten auf diesem Gebiet einig [CR94, PIHS96, HSN97, AGPR99, KE01]:

- *Parametrische Verfahren*

Bei diesen Verfahren wird die Datenverteilung durch eine mathematische Funktion mehrerer Parameter beschrieben. Beispiele hierfür sind Funktionen für Gleichverteilung, Normalverteilung oder Zipfverteilung. Die Schätzung der Parameter dieser Funktionen ist ausschlaggebend für die Genauigkeit der Verfahren. Gute Schätzungen mit diesen Verfahren sind dann möglich, wenn die tatsächliche Datenverteilung durch die verwendete Funktion gut angenähert werden kann. Reale Daten folgen im Allgemeinen nicht einer mathematischen Beschreibung, sie sind vielmehr Veränderungen unterworfen. Die Qualität der Schätzergebnisse kann unvorhersehbar sein. Beispiele für parametrische Verfahren und Verweise auf weitere Arbeiten finden sich in [CR94, SLRD93].

- *Nicht parametrische Verfahren*

Hier wird versucht, die Datenverteilung durch geeignete Datenstrukturen und darauf arbeitende Algorithmen zu beschreiben. Bekannteste Vertreter sind die Histogramme, die sich in aktuellen DBS durchgesetzt haben. Sie werden deshalb im folgenden Kapitel näher beschrieben. Weitere Ansätze dieser Kategorie sind z. B. die Cardinality Maps von Oommen und Thiyagarajah [OT99, OT00] oder Wavelet-Verfahren [MVW98, CGRS01].

- *Sampling-Verfahren*

Es existieren viele Arbeiten auf diesem Gebiet [LN90, HS92, HSN97]. Sampling-Verfahren entnehmen den Daten Stichproben (Samples). Für die Kardinalitätsschätzungen werden die Anfragen mit den Samples ausgeführt. Die Datenverteilung und vorhandene Abhängigkeiten zwischen den Daten haben keinen Einfluss auf die Qualität der Schätzergebnisse. Die Struktur der Daten wird durch die Samples erfasst. Eine hinreichend große Anzahl an Samples ermöglicht eine exakte Abschätzung der Kardinalitäten. Hier zeigt sich aber auch der große Nachteil dieser Verfahren, der einen großflächigen Einsatz in der Praxis bislang verhindert hat [CR94]. Eine große Anzahl erforderlicher Samples führt zu hohen Kosten. Sampling Verfahren kommen deshalb nur in einigen Spezialbereichen zum Einsatz, in denen Kardinalitätsschätzungen weniger häufig durchgeführt werden müssen, z. B. bei aggregierten Anfragen wie COUNT() oder AVG(). Sampling-Verfahren werden außerdem zum Erstellen von Statistiken verwendet (s. Abschnitt 2.3).

In vielen DBS (DB2, Oracle, SQL-Server, Informix, Ingres, Sybase) werden für die Kosten- und Kardinalitätsschätzungen Histogramme verwendet [Ioa03, IP95a]. Diese Verfahren sind anderen in vielerlei Hinsicht überlegen. Auf Grund ihres praktischen Einsatzes bilden sie die Grundlage für die weiteren Untersuchungen und werden deshalb im folgenden Abschnitt genauer vorgestellt.

### 2.3 Statistiken und Histogramme

Um mittels Kostenmodellen eine Abschätzung der zu erwartenden Kosten durchführen zu können, müssen die Parameter einer Kostenfunktion bekannt sein. Dafür werden im Datenbankkatalog statistische Angaben über die Datenbankobjekte geführt. Dazu gehören nach Mitschang [Mit95]:

- *Statistische Kenngrößen pro Segment S*  
Anzahl der Datenseiten in S, Anzahl leerer Seiten in S, Belegungsfaktor
- *Statistische Kenngrößen pro Relation R*  
Anzahl der Tupel in  $R = \text{Card}(R)$ , Anzahl der Seiten mit Tupeln aus R, Blockungsfaktor
- *Statistische Kenngrößen pro Attribut A*  
Anzahl der Attributwerte von A, Histogramm der Werteverteilung von A, minimaler und maximaler Attributwert von A, Wiederholungsfaktor von Attributwerten
- *Statistische Kenngrößen pro Zugriffspfadstruktur ZP*  
Anzahl der Schlüsselwerte in ZP, Baumeigenschaften von ZP

Mannino u. a. ordnen diese Kategorien hierarchisch zu einem komplexen Datenbankprofil an. Ein Relationenprofil impliziert mehrere Attributprofile [MCS88]. Diese wiederum können mehrere Indexprofile besitzen. Unter den statistischen Kenngrößen für Relationen finden sich die schon häufig angesprochenen Kardinalitäten. Bei der Bottom-Up-Verarbeitung eines Ausführungsplans müssen in einem ersten Schritt Abschätzungen auf den Basistabellen durchgeführt werden. Die Anzahl der zu verarbeitenden Tupel wird durch die Information im Datenbankkatalog bereitgestellt. Eine Detaillierungsstufe tiefer werden Informationen zu Attributen einzelner Relationen gehalten. Damit können Selektivitätsabschätzungen für die Prädikate der Planoperatoren durchgeführt werden. „Einfache Statistiken“, wie z. B. die Anzahl der Attributwerte oder der minimale und maximale Attributwert, sind nicht sonderlich aussagekräftig. Vor allem machen sie keine Aussagen über eine Verteilung der Attributwerte. Die sind jedoch für vielerlei Anfragen notwendig (z. B. Selektion über Attribut A, Aggregationen über Attribut A).

Für die Beschreibung der Datenverteilung haben sich die Histogramme durchgesetzt. Hierzu existiert eine Vielzahl an Arbeiten, die sich mit unterschiedlichen Histogrammtypen, deren Genauigkeit, Erstellung und Wartung auseinander setzen. Einen umfassenden Überblick findet man in der Arbeit „The History of Histograms“ von Ioannidis [Ioa03]. Ein Histogramm für ein Attribut A approximiert die Verteilung der Attributwerte. Die *Frequenz* bezeichnet die Anzahl der Attributwerte eines Attributs innerhalb einer Relation. Alle Paare (Attributwert, Frequenz) werden in disjunkte Intervalle, so genannte Buckets, unterteilt [IP95a, PIHS96, Ioa03]. Die

Attributwerte und Frequenzen innerhalb eines Buckets werden approximiert. Die Aufteilung in Buckets kann nach verschiedenen Kriterien geschehen. Poosala und Ioannidis führten folgende Taxonomie ein [PI97]:

*Partitionierungsbedingung(Sortierparameter, Quellparameter)*

- *Partitionierungsbedingung*: Nach dieser mathematischen Bedingung wird die Partitionierung in Buckets vorgenommen. Beispiel: *Equi-Sum*. Die Summe der Quellparameterwerte ist annähernd gleich.
- *Sortierparameter*: Der Sortierparameter lässt sich aus der Datenverteilung (Attributwert, Frequenz) ableiten. Die Buckets sind nach diesem Parameter zusammenhängend angeordnet. Es existieren keine Überlappungen mit benachbarten Buckets. Beispiele: Value (V), Frequency (F), Area (A).
- *Quellparameter*: Der Quellparameter beschreibt eine Eigenschaft der Datenverteilung. Er wird zusammen mit der Partitionierungsbedingung dazu verwendet, eine einheitliche Aufteilung in Buckets vorzunehmen, und stellt den für die Schätzung kritischen Parameter dar. Beispiele: Frequency (F), Area (A), Spread (S).

Je nach Wahl dieser Parameter und der Approximation innerhalb der Buckets entstehen unterschiedliche Histogrammklassen. Die zwei traditionellen Vertreter sind das Equi-Width-Histogramm und das Equi-Depth-Histogramm.

- *Equi-Width-Histogramm, Equi-Sum(V,S)*: Angelehnt an die obige Taxonomie ist das Equi-Width-Histogramm wie folgt definiert: Partitionierungsbedingung = Equi-Sum, Quellparameter = Breite (Spread), Sortierparameter = Attributwert (Value). Alle Werteintervalle sind somit gleich breit. Der Histogramminhalt ist nach aufsteigenden Attributwerten sortiert. Abbildung 2.2 zeigt ein solches Histogramm. Jeder Bucket erstreckt sich hier über einen Wertebereich von 10 (1-10, 11-20, ...).
- *Equi-Depth-Histogramm, Equi-Sum(V,F)*: Eine Verbesserung gegenüber dem Equi-Width-Histogramm ist das Equi-Depth-Histogramm, auch Equi-Height-Histogramm genannt (s. Abbildung 2.3). Alle Buckets enthalten hier gleich viele Tupel. Der Flächeninhalt der Buckets ist gleich groß. Die Schätzfehler dieser Histogramme sind sowohl im schlechtesten wie auch im durchschnittlichen Fall bei gemischten Anfragen geringer als bei Equi-Width-Histogrammen [PC84]. In aktuellen DBS kommt dieser Histogrammtyp zum Einsatz.

Die eigentliche Datenverteilung, bestehend aus Paaren von Attributwert und Frequenz (s. Abschnitt 3.2.1), wird durch die Buckets approximiert. Bei der Approximation der Frequenz wird für jeden Attributwert innerhalb eines Buckets die durchschnittliche Frequenz aller Attributwerte dieses Buckets zu Grunde gelegt. In Abbildung 2.2 beispielsweise ist das für alle Attributwerte zwischen 21 und 30 eine Frequenz von 160. Für die Approximation der Attributwerte existieren zwei Möglichkeiten: Für jeden Bucket werden der minimale und der maximale Attributwert gespeichert. Unter der Annahme, dass die Attributwerte den Bereich stetig ausfüllen, wird jeder Wert als existent angenommen. Ein anderer Ansatz speichert zusätzlich die Anzahl

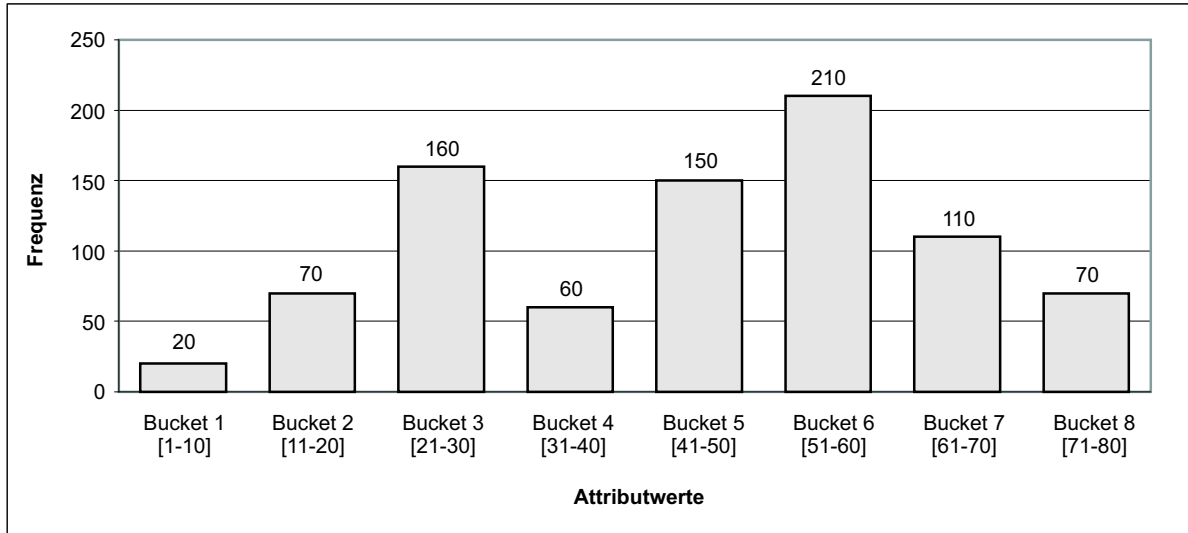


Abbildung 2.2: *Equi-Width-Histogramm*

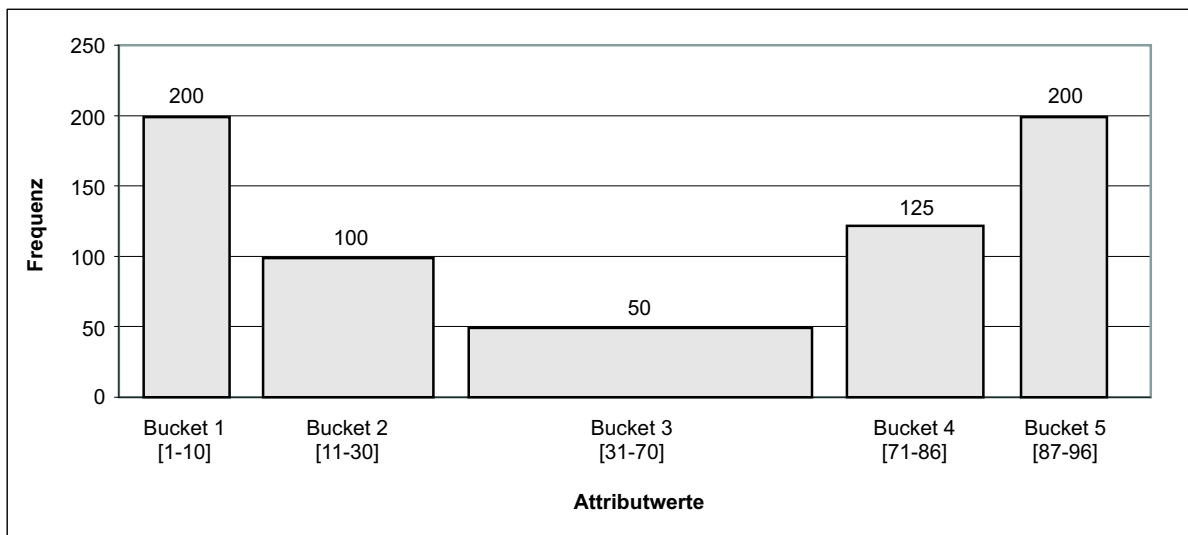


Abbildung 2.3: *Equi-Depth-Histogramm*

der unterschiedlichen Werte pro Bucket. Die Attributwerte werden dann in gleichen Abständen in dem Wertintervall platziert [Ioa03, PIHS96].

Eine neue Histogrammklasse entstand durch die Verwendung der Frequenz als Sortierparameter, an Stelle des Attributwerts. Diese Histogramme gruppieren gleiche Frequenzen in gleiche Buckets. Sie wurden anfangs als *Serielle Histogramme* bezeichnet [IC93]. Der Begriff wurde jedoch verallgemeinert und steht heute für alle Histogramme, die in nicht überlappende Buckets unterteilt sind, unabhängig von der Wahl des Sortierparameters. Ein Vertreter dieser Klasse sind *V-optimale(F,F)* Histogramme. Sie sind Histogrammen mit dem Attributwert als Sortierparameter in Schätzgenauigkeit überlegen [IP95b]. Ein gravierender Nachteil ist jedoch, dass für die Speicherung weitaus mehr Platz benötigt wird, da für jeden Bucket die darin enthaltenen Tupel gespeichert werden müssen. Der nötige Speicheraufwand hat eine vollständige Umsetzung in DBS bislang verhindert. Lediglich Mischformen wurden implementiert, indem z. B. die Werte mit den höchsten Frequenzen in separaten Buckets gehalten werden. Neuere histogrammbasierte Ansätze sind die Cardinality Maps von Oommen und Thiyagaraiah [OT99, OT00] oder Wavelets [MVW98, CGRS01]. Für detailliertere Informationen sowie weitere Histogrammklassen und -unterklassen sei an dieser Stelle auf die umfangreiche Literatur verwiesen [Ioa03].

Die hier beschriebenen eindimensionalen Histogramme beschreiben die Datenverteilung eines einzelnen Attributs. Datenabhängigkeiten zwischen mehreren Attributen innerhalb einer Relation und auch über mehrere Relationen hinaus können damit nicht berücksichtigt werden. Um in solchen Fällen zu guten Schätzwerten zu gelangen, existieren mehrdimensionale Ansätze [WKW94, PI97, LKC99].

Eindimensionale Histogramme liefern für Selektivitätsabschätzungen ihres Attributs gute Schätzwerte und verbrauchen wenig Speicherplatz im Datenbankkatalog. Für die Erstellung werden Sampling-Strategien angewendet. Es muss somit nicht die gesamte Datenbasis betrachtet werden. Die Wartung geschieht asynchron zum laufenden Betrieb und ist immer dann nötig, wenn die im Histogramm gespeicherten Werte von der tatsächlichen Datenverteilung abweichen. Auf Grund dieser Vorteile haben sich Histogramme als Statistik zur Approximation der Datenverteilung in DBS gegenüber anderen Verfahren durchgesetzt.

## 3 Einflussparameter

Es existieren verschiedene Einflussparameter, die sich unterschiedlich stark auf die Schätzergebnisse auswirken. Je nach verwendetem Schätzverfahren variiert dieser Einfluss. Deshalb sollen hier die möglichen Einflussparameter auf einer abstrakten Ebene vorgestellt werden. Eine quantitative Analyse ist nur auf Basis eines bestimmten Schätzverfahrens möglich. Bei Chu, Halpern und Seshadri findet sich eine Klassifizierung der Einflussfaktoren nach Datenparameter, Anfrageparameter und Laufzeitparameter [CHS99]. Die Anfrageparameter werden nachfolgend nochmals unterteilt: In Parameter, die direkt aus der SQL-Anfrage entnommen werden können, und in Parameter, die einen konkreten Ausführungsplan voraussetzen. Diese Unterteilung ist angelehnt an die Stufen des Optimierungsprozesses, bei dem ausgehend von der SQL-Anfrage Ausführungspläne generiert werden. Es entstehen dabei neue Freiheitsgrade und somit weitere Einflussfaktoren für die Kostenschätzung. Zusätzlich ist die Liste um die Qualität der Statistiken erweitert, da Statistiken als elementare Grundlage für die Kostenschätzungen dienen. Sie haben direkten Einfluss auf die Qualität der Schätzergebnisse. Es ergibt sich folgende Aufteilung:

- Qualität der Statistiken
- Parameter, die Daten beschreiben
- Parameter, welche die SQL-Anfrage betreffen
- Parameter, die spezifisch für einen Ausführungsplan sind
- Laufzeitparameter

In den nachfolgenden Abschnitten werden die hier aufgeführten Einflussparameter genauer betrachtet und deren Auswirkung auf die Qualität der Schätzergebnisse untersucht.

### 3.1 Qualität der Statistiken

Abhängig von der Anfrage greift der Optimierer auf Statistiken des DBS zu, um möglichst genaue Schätzungen für die Knoten des Ausführungsplans zu liefern. Zwei Kriterien lassen sich unterscheiden:

- Welche Informationen stehen für die Schätzungen zur Verfügung?
- Wie gut ist deren Qualität?

Bei dem ersten Punkt geht es darum, ob überhaupt Statistiken existieren. Abbildung 3.1 zeigt zwei Ausführungspläne, für die der Optimierer die anfallenden Kosten schätzen soll. Betrachten wir Plan 1 und nehmen an, es existieren Statistiken für das Attribut 2 in Relation R2, nicht

jedoch für das Attribut 3 in Relation R3. Für die Selektion auf R2 kann dann auf diese Statistiken zurückgegriffen werden. Für die Selektion auf R3 ist das nicht möglich, was ungenauere Schätzwerte zur Folge hat.

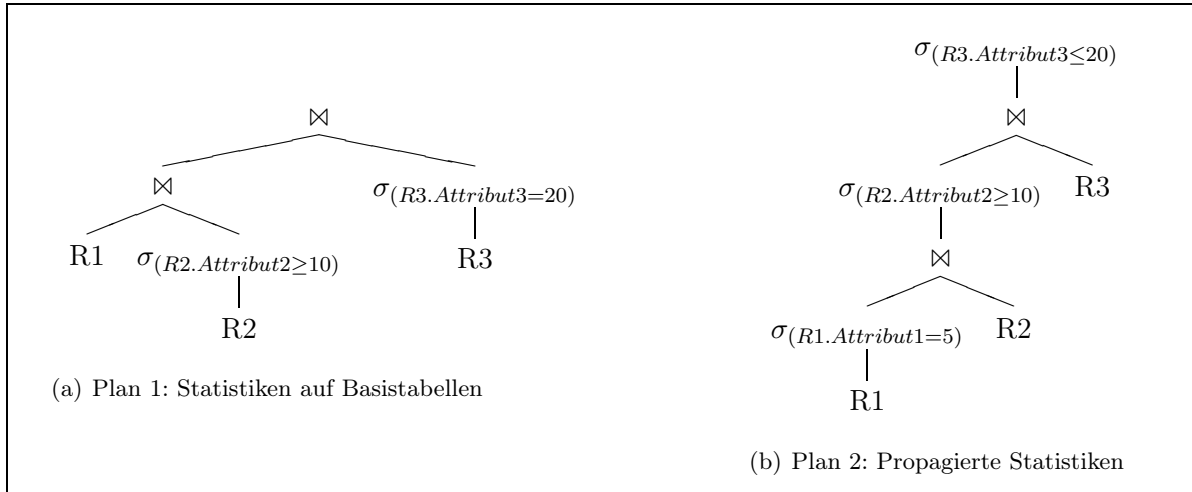
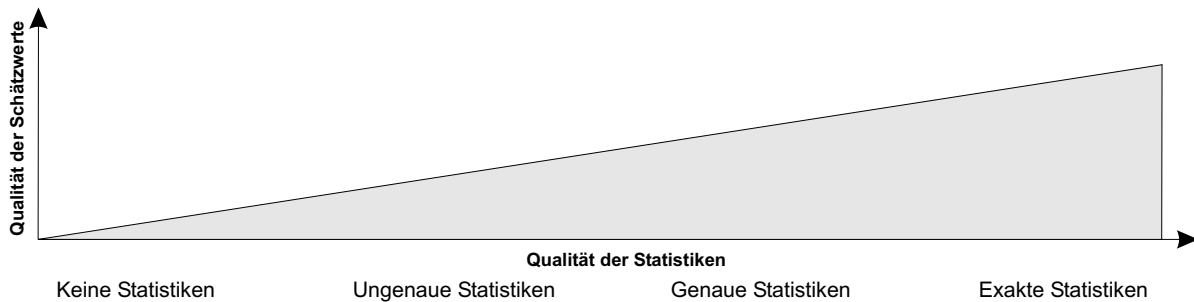


Abbildung 3.1: Verwendung von Statistiken bei Ausführungsplänen

Der zweite Aspekt betrifft die eigentliche Qualität der Statistiken. Wenn Statistiken verfügbar sind, wie hier für das Attribut 2 in R2, so ist die Qualität des geschätzten Zwischenergebnisses abhängig von der Qualität der Statistiken. Je nach Genauigkeit der gespeicherten Werte differiert die geschätzte Kardinalität von der tatsächlichen Kardinalität. Abbildung 3.2 zeigt schematisch diese Beziehung. Die gewählten Begriffe „ungenauere Statistiken“ und „genaue Statistiken“ sind unscharf. Was sind „genaue Statistiken“? Gemeint ist hier, wie exakt die Statistiken den tatsächlichen Datenbankzustand beschreiben. Ein Maß für die Genauigkeit ist zur Darstellung des Zusammenhangs auch nicht nötig. Qualitativ hochwertigere Statistiken liefern genauere Schätzwerte. Oommen und Rueda haben gezeigt, dass bei der Verwendung von Statistiken, welche anderen in Genauigkeit überlegen sind, die Wahrscheinlichkeit größer ist, dass ein kostengünstigerer Ausführungsplan gewählt wird, als bei der Verwendung von ungenaueren Statistiken [OR01].

Es ist aber durchaus möglich, dass Schätzungen mit Statistiken schlechter ausfallen, als wenn gar keine verfügbar sind. Das ist dann der Fall, wenn Statistiken stark von dem tatsächlichen Datenbankzustand, den sie beschreiben, abweichen. Bei fehlenden Statistiken verwendet das DBS für die Schätzung Standard-Filterfaktoren [IBM04]. Die Standardwerte liefern gute Schätzergebnisse, wenn die Annahmen des Optimierers über die Struktur der Daten (Datenelemente sind gleichverteilt und unabhängig voneinander) korrekt sind (s. Abschnitt 3.2). Treffen diese Annahmen nicht zu, so können die Standard-Filterfaktoren zu schlechten Schätzergebnissen führen.

Ebenso ausschlaggebend für die Qualität der Schätzwerte ist die Stelle im Ausführungsplan, an der auf die Statistiken zugegriffen wird. Es ist ein Unterschied, ob Statistiken direkt auf eine Relation angewendet werden können oder ob sie durch den Ausführungsplan propagiert

Abbildung 3.2: *Statistiken und Qualität der Schätzwerte*

werden [GJWW03]. Der Ausführungsplan 2 in Abbildung 3.1 zeigt ein solches Beispiel. Nehmen wir an, zu jedem Attribut existieren Statistiken über die Datenverteilung. Dann kann damit bei den Kardinalitätsschätzungen der Zwischenergebnisse die Selektivität der verwendeten Attribute bestimmt werden. Die Schätzwerte der Selektion auf R1 werden im Allgemeinen die höchste Qualität aufweisen, da die Statistiken direkt auf die unmodifizierten Basistabellen angewendet werden können. Die darauf folgenden Selektionen setzen auf den Zwischenergebnissen auf. Die Basisstatistiken liefern hier schlechtere Schätzwerte auf Grund von möglichen Datenabhängigkeiten in den Relationen R1 und R2 (s. Abschnitt 3.2). Noch ungenauer ist die Schätzung der dritten Selektion auf Relation R3.

## 3.2 Struktur der Daten

Wenn keine Statistiken über die Daten vorhanden sind, legt der Anfrageoptimierer für seine Schätzungen die zwei folgenden Annahmen zu Grunde [Chr84]:

- Alle Datenelemente und alle Attributwerte sind gleichverteilt
- Die Suchprädikate in Anfragen sind unabhängig

Beide Annahmen sind im allgemeinen Fall jedoch falsch. Treffen sie nicht zu, so werden die geschätzten Kardinalitäten von den tatsächlichen Werten abweichen. Dieser Zusammenhang wird in den beiden folgenden Abschnitten genauer betrachtet.

### 3.2.1 Datenverteilung

Bei der Datenverteilung lassen sich zwei Gruppen unterscheiden. Die *eindimensionale Datenverteilung* ist über einem einzelnen Attribut definiert und beschreibt die Verteilung der Werte dieses Attributs. Sie besteht aus einer Menge von Paaren der Form (Attributwert, Frequenz). Analog dazu beschreibt die *mehrdimensionale Datenverteilung* die Verteilung von Wertekombinationen mehrere Attribute. In den meisten Fällen wird der Begriff der Datenverteilung im Sinne einer eindimensionalen Datenverteilung verwendet. Die folgenden mathematischen Definitionen sind Poosala und König entnommen [IP99, Koe01]:

- Die Attribute einer Relation  $R$  werden mit  $A_1 \dots A_i$  bezeichnet

- Die Werte eines Attributs  $A_j$  sind definiert über einer Domäne  $D_{A_j}$  möglicher Attributwerte
- Die Menge  $V_{A_j} \in D_{A_j}$ ,  $V_{A_j} = \{v_1, \dots, v_n\}$  mit  $v_k < v_j$  für  $k < j$  ist die Menge der Attributwerte von  $A_j$ , die tatsächlich in der Relation  $R$  vorkommen
- Die *Spannweite*  $s_k$  von  $v_k$  ist definiert als  $s_k = v_{k+1} - v_k$  mit  $s_n = 1$
- Die *Frequenz*  $f_k$  von  $v_k$  steht für die Anzahl der Tupel in  $R$  mit  $A_j = v_k$
- Die *Fläche*  $a_k$  von  $v_k$  ist definiert als  $a_k = f_k \cdot s_k$
- Die *Datenverteilung* von Attribut  $A_j$  ist eine Menge von Paaren bestehend aus Attributwert und Frequenz:  $T_{A_j} = \{(v_1, f_1), (v_2, f_2), \dots, (v_n, f_n)\}$
- Die *mehrdimensionale Datenverteilung* von  $d$  Attributen  $A_{j_1}, \dots, A_{j_d}$  ist eine Menge von Paaren bestehend aus Attributwertkombination und Frequenz:  
 $T_{A_{j_1}, \dots, A_{j_d}} = \{(v_1, f_1), (v_2, f_2), \dots, (v_n, f_n)\}$  mit  $v_t \in V_{A_{j_1}} \times \dots \times V_{A_{j_d}}$

Tabelle 3.1 zeigt ein Beispiel für eine Datenverteilung (Attributwert, Frequenz) und mögliche Werte für die oben definierten Parameter Spannweite und Fläche. Die Relation besitzt genau fünf verschiedene Attributwerte (10, 60, 70, 90, 100). Der Attributwert 10 existiert 100-mal, die Spannweite zum nächsten Attributwerte 60 beträgt 50. Daraus ergibt sich eine Fläche von 5000.

Parameter	Datenverteilung				
Attributwert $v_k$	10	60	70	90	100
Frequenz $f_k$	100	120	10	80	2000
Spannweite $s_k$	50	10	20	10	1
Fläche $a_k$	5000	1200	200	800	2000

Tabelle 3.1: Beispiel für die Parameter einer Datenverteilung

Parametrische Schätzverfahren (s. Abschnitt 2.2) versuchen die Datenverteilung möglichst genau mathematisch zu beschreiben. Sie sind jedoch unzureichend, da reale Daten nicht einer solchen Verteilung folgen. Die Daten sind darüber hinaus strukturellen Änderungen unterworfen, was ihre Beschreibung zusätzlich erschwert.

Bei Histogrammen (s. Abschnitt 2.3) wird die Datenverteilung durch Buckets approximiert. Innerhalb jedes Buckets wird eine Gleichverteilung der Attributwerte zwischen minimalem und maximalem Attributwert angenommen. Somit muss nur eine geringe Datenmenge im Datenbankkatalog zur Repräsentation für die Datenverteilung gespeichert werden. Ein vollständiges Abbild der Verteilung wäre aus Effizienz- und Speicherplatzgründen auch nicht möglich. In der Realität liegt im Allgemeinen jedoch keine Gleichverteilung der Daten vor, sondern es existiert eine *Datenungleichverteilung (Data-Skew)*, z. B. einige wenige Werte mit hohen bzw. tiefen Frequenzen. Das kann bei Verwendung nicht parametrischer Schätzverfahren zu ungenauen Kosten- und Kardinalitätsschätzungen führen. Ein Beispiel hierfür sind die im praktischen Einsatz befindlichen Equi-Depth-Histogramme. In ihrer grundlegenden Form sehen sie für extreme Frequenzwerte in der Datenverteilung keine gesonderten Buckets vor.

Abbildung 3.3 zeigt das gleiche Histogramm wie Abbildung 2.3 auf Seite 14. Zusätzlich ist die Kurve einer möglichen Datenverteilung eingezeichnet, welche die vorliegende Bucketeinteilung zur Folge hat.

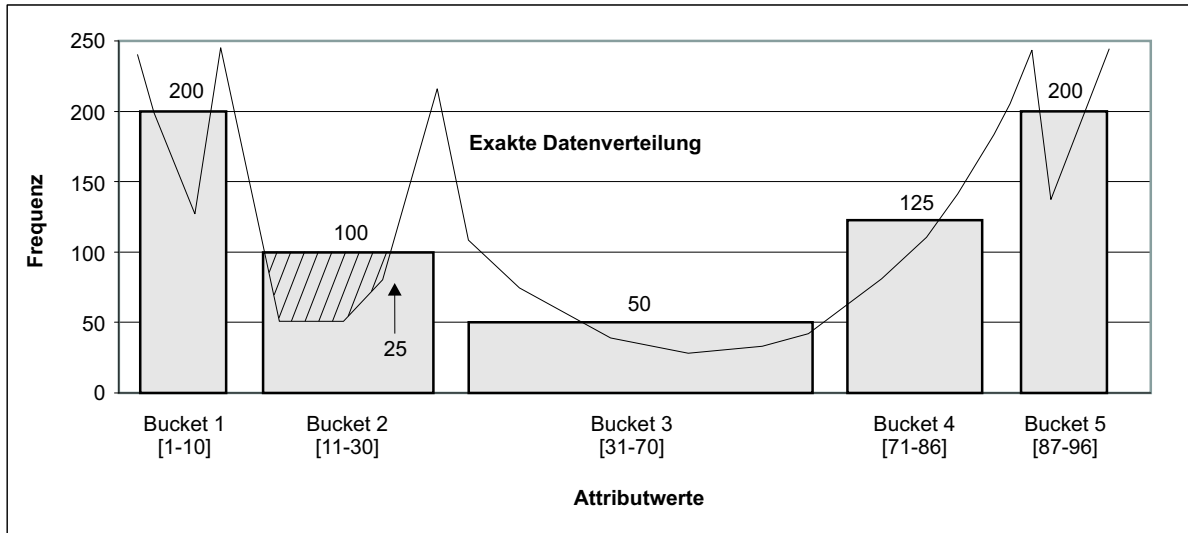


Abbildung 3.3: *Equi-Depth-Histogramm mit Datenverteilung*

Wir nehmen an, dass zu jedem Attributwert in  $R$  mindestens ein Tupel existiert, d. h., die gesamte Attributdomäne wird lückenlos ausgeschöpft. In jedem Bucket befinden sich 2000 Tupel, d. h., die Fläche aller Buckets ist gleich:  $a_k = 2000$ . Die reale Datenverteilung wird durch das Histogramm in Abbildung 3.3 nur unzureichend approximiert. Eine Anfrage mit der Selektionsbedingung  $1 \leq \text{Attributwert} \leq 25$  verdeutlicht das. Die Anzahl der qualifizierten Tupel aus dem Wertebereich von Bucket 1 kann mit  $10 \cdot 200 = 2000$  exakt geschätzt werden, da der Bereich durch die Anfrage vollständig überdeckt wird. Anders sieht es mit Bucket 2 aus, der nur teilweise überdeckt wird. Hier wird für alle Attributwerte von 11 bis 25 die durchschnittliche Frequenz von 100 angenommen. Es qualifizieren sich somit  $15 \cdot 100 = 1500$  Tupel. Die Verwendung der approximierten Frequenzwerte führt hier zu Schätzfehlern, denn die tatsächliche Anzahl der Tupel, die einen Wert von 11 bis 25 annehmen, liegt weit darunter. Der Schätzfehler ist schraffiert dargestellt.

Je nach Stärke der Datenungleichverteilung schwanken die unter der Annahme der Gleichverteilung erzielten Schätzergebnisse. Als Literatureinstieg in erste Arbeiten auf diesem Gebiet sei auf Mannino u. a. verwiesen [MCS88].

### 3.2.2 Datenabhängigkeit

Für die Kosten- und Kardinalitätsschätzungen setzt der Optimierer die Unabhängigkeit der Daten voraus. Abhängigkeiten verschiedener Attributwerte beeinflussen somit die Qualität der Schätzergebnisse. Dazu gehören sowohl Abhängigkeiten innerhalb einer Relation, als auch Abhängigkeiten über mehr als eine Relation hinaus. Zwei Beispiele sollen diesen Zusammenhang verdeutlichen.

Angenommen, in einer Tabelle werden die personenspezifischen Daten Name, Alter, Größe (in cm), Gewicht (in kg), Postleitzahl und Wohnort gespeichert. Folgende Anfrage wird an das DBS gestellt:

```
(Q1) SELECT  *
      FROM    Personen
      WHERE   Alter > 20 AND Groesse > 160 AND Gewicht > 65
```

Die Selektivität einer Anfrage ergibt sich aus dem Produkt der Selektivitäten für die einzelnen Prädikate [Mit95]:  $s(p_1 \wedge p_2 \wedge p_3) = s(p_1) \cdot s(p_2) \cdot s(p_3)$ . Nehmen wir folgende Verteilung an: 70 % aller Personen sind älter als 20 Jahre, 30 % sind größer als 160 cm und 50 % wiegen mehr als 65 Kilogramm. Die Annahme der Datenunabhängigkeit führt zu folgender Selektivitätsabschätzung:

$$s(\text{Alter} > 20) \cdot s(\text{Groesse} > 160) \cdot s(\text{Gewicht} > 65) = 0,7 \cdot 0,3 \cdot 0,5 = 0,105 \approx 10\%$$

Es ist aber offensichtlich, dass die Selektivität auf Grund existierender Abhängigkeiten zwischen allen drei Attributen viel zu gering geschätzt ist und die obigen Eigenschaften auf mehr als 10 % aller Personen zutreffen. Für eine quantitative Angabe des Fehlers müssten die Datenverteilungen exakt spezifiziert sein. Für das obige Beispiel ist dieser Aufwand sehr groß, weshalb eine quantitative Analyse anhand des folgenden einfachen Beispiels durchgeführt wird.

```
(Q2) SELECT  *
      FROM    Personen
      WHERE   Postleitzahl = 70173 AND Ort = 'Stuttgart'
```

Die Attribute *Postleitzahl* ( $P$ ) und *Ort* ( $O$ ) der Relation *Personen* sind funktional abhängig<sup>1</sup>:

$$P \rightarrow O := \forall u, v \in \text{Personen} : (u.P = v.P) \Rightarrow (u.O = v.O)$$

Die Postleitzahl legt den zugehörigen Ort eindeutig fest. Die Anfrage Q2 enthält eine Selektion über die beiden funktional abhängigen Attribute. Angenommen, es existieren 20 000 verschiedene Postleitzahlen, dann beträgt der reale Selektivitätsfaktor

$$s_r(P = p) = \frac{1}{20\,000} = 0,5 \cdot 10^{-4}$$

Die Selektivitätsabschätzung des Optimierers führt unter der Annahme der Datenunabhängigkeit zu folgendem Ergebnis:

$$s_g(P = 70173 \wedge O = \text{'Stuttgart'}) = 0,5 \cdot 10^{-4} \cdot 0,5 \cdot 10^{-4} = 0,25 \cdot 10^{-8}$$

Durch die funktionale Abhängigkeit des Orts von der Postleitzahl liegt die geschätzte Selektivität  $s_g$  um Größenordnungen neben dem realen Wert  $s_r$  mit

$$s_r(P = p \wedge O = o) = 0,5 \cdot 10^{-4}$$

---

<sup>1</sup>Definition *funktionale Abhängigkeit* von Kemper und Eickler [KE01]

Daraus resultieren folgende Werte für die Kardinalitäten des Anfrageergebnisses:

$$\begin{aligned}Card_g(Q2) &= s_g \cdot Card(Personen) = 0,25 \cdot 10^{-8} \cdot 20\,000 = 0,5 \cdot 10^{-4} \\Card_r(Q2) &= s_r \cdot Card(Personen) = 0,5 \cdot 10^{-4} \cdot 20\,000 = 1\end{aligned}$$

Der Schätzwert weicht somit um den Faktor  $F = \frac{s_g}{s_r} = \frac{1}{20\,000}$  von der tatsächlichen Kardinalität ab. Der Fehler vergrößert sich, sobald noch weitere funktional abhängige Attribute hinzukommen.

Man könnte jetzt entgegenhalten, dass funktionale Abhängigkeiten abgesehen vom Primärschlüssel einer Relation selten vorkommen und beim Datenbankentwurf vermieden werden sollten. Doch bereits schwächere Abhängigkeiten zwischen Attributwerten führen zu Schätzfehlern. Selbst wenn der Wert eines Attributs P den Wert von Attribut O nicht exakt bestimmt, sondern auf 100 Möglichkeiten einschränkt, würde eine Selektion über Attribut P und über den Wertebereich der zugehörigen 100 Tupel in O noch zu einem Fehlerfaktor von  $F = \frac{1}{200}$  führen:

$$\begin{aligned}s_g(P = p \wedge 1 < O \leq 100) &= 0,5 \cdot 10^{-4} \cdot \frac{100}{20\,000} = 0,25 \cdot 10^{-6} \\ \Rightarrow F = \frac{s_g}{s_r} &= \frac{0,25 \cdot 10^{-6}}{0,5 \cdot 10^{-4}} = \frac{1}{200}\end{aligned}$$

Ausschlaggebend für die Größe des Schätzfehlers sind sowohl der Grad der Datenabhängigkeit als auch die Anzahl der bei der Kostenschätzung beteiligten abhängigen Attribute. Die Abhängigkeiten können auch über mehrere Relationen hinausgehen.

## 3.3 Anfragespezifische Parameter

In diesen Bereich fallen Einflussfaktoren, die bereits aus der SQL-Anfrage abgeleitet werden können. Verschiedene Parameter beeinflussen die Komplexität der Anfrage:

- Beteiligte Relationen und Sichten
- Logische Operatoren
- Prädikate mit Attributarithmetik
- Funktionen, Stored Procedures, Trigger, Rekursionen

Diese anfragespezifischen Parameter sind nicht unabhängig voneinander. Sind z.B. mehrere Relationen beteiligt, so ergibt sich als logischer Operator der Verbundoperator, über den die Relationen verknüpft sind. Bei der nachfolgenden Betrachtung der einzelnen Parameter sollte man sich dieser Abhängigkeiten bewusst sein.

### 3.3.1 Beteiligte Relationen und Sichten

Einen wesentlichen Einfluss auf die Komplexität der Anfrage haben die beteiligten Relationen. Entscheidend ist hier, wie umfangreich die zur Verfügung stehenden Statistiken für diese Relationen sind und ob sie zur Kostenschätzung der Anfrage verwendet werden können (s. Abschnitt 3.1). Basisstatistiken wie die Kardinalität einer Relation stehen dem Optimierer für jede

Relation im Datenbankkatalog zur Verfügung. Das ist bei detaillierteren Statistiken wie z. B. der Datenverteilung einzelner Attributwerte nicht mehr gewährleistet.

Die Zahl der beteiligten Relationen erhöht die Komplexität der Anfrage. Die Kostenschätzung einzelner Relationen kann im Gegensatz zur Schätzung mehrerer Relationen noch relativ genau durchgeführt werden. Mit der Anzahl der Relationen steigt die Zahl der durchzuführenden Schätzungen und damit die Zahl potenzieller Fehlerstellen.

Ein ähnliches Problem ergibt sich bei der Verwendung von Sichten, wenn sie nicht auf Relationen abgebildet werden können, für die Statistiken verfügbar sind. Das gilt insbesondere für Sichten, die über mehrere Relationen erstellt wurden. Genaue Schätzungen sind auf Grund fehlender Statistiken dann nicht möglich. Galindo-Legaria u. a. haben in ihrer Arbeit zur Verbesserung der Schätzgenauigkeit den Ansatz vorgestellt, Statistiken für Sichten bereitzustellen und ihre Vorschläge auch in Microsoft's SQL-Server implementiert [GJWW03].

#### 3.3.2 Logische Operatoren

Einer SQL-Anfrage lässt sich bereits entnehmen, welche logischen Operatoren verwendet werden. In der relationalen Algebra existieren sechs Operatoren: Selektion, Projektion, Vereinigung, Mengendifferenz, Kartesisches Produkt und Umbenennung [KE01]. Diese sechs Operatoren sind alle in SQL verfügbar, inklusive einiger Erweiterungen für tupelbezogene Operationen wie z. B. die Gruppierung oder die Sortierung. Der Verbundoperator lässt sich durch Anwendung des kartesischen Produkts und Filterung der dabei entstehenden Tupelmengen aus den Operatoren der relationalen Algebra ableiten. Anhand der Anzahl der Eingaberelationen unterscheidet man *unäre Operatoren* und *binäre Operatoren*. Die wichtigsten, in SQL verfügbaren Operatoren, welche die Kardinalitäten der Zwischen- und Endergebnisse beeinflussen, sind:

- Unäre Operatoren:
  - Selektionsoperator
  - Projektionsoperator
  - Gruppierungsoperator
- Binäre Operatoren:
  - Verbundoperator
  - Mengentheoretische Operatoren (Vereinigung, Schnitt, Differenz)

Bei den Schätzungen aller Operatoren spielen die verfügbaren Statistiken und deren Qualität (s. Abschnitt 3.1) eine entscheidende Rolle. Die Kardinalitäten für unäre Operatoren lassen sich im Allgemeinen genauer abschätzen als die Kardinalitäten für binäre Operatoren, die auf mindestens zwei Eingabeströmen arbeiten. Nachfolgend werden die einzelnen Operatoren unter dem Gesichtspunkt der Kardinalitätsschätzung genauer betrachtet.

#### Unäre Operatoren

Die unären Operatoren erwarten als Eingabe genau einen Objektstrom. Es muss somit nur eine Eingabe für die Schätzung berücksichtigt werden. Die Zahl der Fehlerquellen ist dadurch geringer als bei den binären Operatoren.

Selektionen über ein Attribut einer Basisrelation bereiten kaum Probleme, wenn die Datenverteilung in Form von Histogrammen vorliegt. Sind keine ausreichenden Statistiken verfügbar, so verwendet der Optimierer für seine Abschätzungen Standardwerte. Für ein Vergleichsprädikat der Form  $Attribut < Wert$  wird beispielsweise der Selektivitätsfaktor  $s = \frac{1}{3}$  angenommen [IBM04]. Solche Schätzergebnisse sind von geringer Qualität. Ebenso problematisch sind Selektionen auf mehreren Attributen. Eventuelle Abhängigkeiten der Attributwerte wirken sich negativ auf die Qualität der Selektivitätsschätzung aus (s. Abschnitt 3.2.2). Multidimensionale Histogramme können dazu verwendet werden, die Datenverteilung mehrerer abhängiger Attribute zu speichern [PI97]. In aktuellen DBS sind multidimensionale Histogramme allerdings erst ansatzweise umgesetzt, so dass die Selektivitäten unter der Annahme der Datenunabhängigkeit für jedes Attribut getrennt geschätzt werden müssen.

Die beiden Operatoren Projektion und Gruppierung können gemeinsam betrachtet werden. Die Kardinalität des eingehenden und des ausgehenden Objektstroms eines Projektionsoperators sind identisch, es sei denn, die Projektion enthält eine Eliminierung von Duplikaten. In diesem Fall ist der Projektionsoperator ein Spezialfall des Gruppierungsoperators. Ebenso wie bei der Selektion sind bei der Gruppierung immer dann genaue Schätzungen zu erwarten, wenn der Optimierer für die Selektivitätsschätzung Statistiken des Datenbankkatalogs verwenden kann (Anzahl unterschiedlicher Attributwerte, Datenverteilung von Attributwerten). Bei der Gruppierung nach mehreren Attributen ergibt sich erneut das Problem der Datenabhängigkeit. Ein weiterer Punkt beeinflusst die Schätzergebnisse. Im Gegensatz zur Selektion, die im Rahmen der Anfragerestrukturierung nach Möglichkeit bis zu den Blättern des Anfragegraphen verschoben wird, findet die Gruppierung und Duplikateliminierung meistens später statt. Die Verwendung von Basisstatistiken oder Standardwerten an solchen Zwischenknoten im Ausführungsplan führt zu ungenauen Schätzwerten.

### Binäre Operatoren

In diesem Abschnitt wird der Verbundoperator als Vertreter der binären Operatoren genauer betrachtet. Im Gegensatz zu den mengentheoretischen Operatoren gehört er zu den Basisoperatoren und findet in vielen SQL-Anfragen Verwendung. Mehrwegeverbunde gehören ebenfalls zu dieser Gruppe, da sie sich in mehrere binäre Verbunde zerlegen lassen.

Der binäre Verbund erwartet zwei Objektströme ( $E_1, E_2$ ) als Eingabe. Durch ein Verbundprädikat der Form  $P := [Va_1 = Va_2]$  wird ein Ausgabestrom  $A$  erzeugt:  $E_1 \bowtie_P E_2 \Rightarrow A$ . Statistiken, die eine exakte Angabe der Kardinalität ermöglichen, stehen im DBS nicht zur Verfügung. Die einzigen Statistiken, welche für die Abschätzungen verwendet werden können, sind die Kardinalitäten der Eingaberelationen. Das führt dazu, dass die Selektivität des Verbundattributs für jede Relation getrennt geschätzt werden muss. Die Kardinalitätsschätzung ergibt sich wie folgt:

$$Card(R \bowtie S) = s_R(p) \cdot Card(R) \cdot s_S(p) \cdot Card(S) \quad \text{mit } s_X(p) := \frac{Card(\sigma_p(X))}{Card(X)}$$

Qualifizieren sich nur wenige Tupel für das Verbundprädikat, so kann die Verbundselektivität und damit die geschätzte Kardinalität stark von der tatsächlichen Kardinalität abweichen, da sich der Fehler direkt multipliziert. Bei Mehrwegeverbunden kommt hinzu, dass Schätzfehler der

Kardinalitäten von Zwischenergebnissen propagiert werden. Ioannidis und Christodoulakis haben gezeigt, dass Kardinalitätsfehler exponentiell mit der Anzahl der Verbunde wachsen [IC91]. Da die an das DBS gestellten Anfragen immer komplexer werden, ist eine exakte Schätzung besonders wichtig. Diese Einflussgröße korreliert mit der Anzahl der beteiligten Relationen (s. Abschnitt 3.3.1). Die Anzahl der durch Verbundoperatoren verketteten Relationen erhöht die Komplexität der Anfrage und vermindert die Qualität der vom Optimierer geschätzten Kardinalitäten.

#### 3.3.3 Prädikate mit Attributarithmetik

Es wurde bereits beschrieben, dass Kostenschätzungen von Selektionsprädikaten mit Hilfe von Histogrammen durchgeführt werden. Bei einfachen Wertevergleichen liefern die im Histogramm abgelegten Informationen über die Datenverteilung dafür gute Schätzwerte. Bestehen die Prädikate jedoch aus mehreren Attributen, die in einem Ausdruck arithmetisch miteinander verknüpft sind, so gestaltet sich die Auswertung schwieriger und es können leicht große Fehler entstehen.

```
(Q3) SELECT *
      FROM   lineitem
      WHERE  l_extendedprice * (1-l_discount) > 50 000
```

Die Anfrage Q3 zeigt ein solches Beispiel aus dem *TPC Benchmark H* (TPC-H), bei dem der diskontierte Preis durch einfache Arithmetik über zwei Attribute der gleichen Tabelle berechnet wird [TPCH04]. Die eindimensionalen Histogramme liefern Kardinalitäten für die zugehörigen Attribute, lassen jedoch keine genaue Schätzung über das Produkt zu, für welches die Attribute abhängig voneinander betrachtet werden müssen. Ebenso problematisch sind Berechnungen mit skalaren Operatoren wie z. B. Moduloarithmetik und Operationen auf Strings wie die Konkatination oder die Extraktion von Teilstrings. Für derartige Operationen lassen sich die im Datenbankkatalog vorhandenen Statistiken nicht verwenden. Die reine Datenverteilung gibt keinen Aufschluss darüber, wie viele Tupel sich für solche Prädikate qualifizieren. Der Optimierer muss dann auf Standardwerte zurückgreifen, die Schätzfehler zur Folge haben können.

#### 3.3.4 Spezielle Konzepte in SQL:1999

Konzepte wie Trigger, benutzerdefinierte Funktionen, Objektorientierung, Stored Procedures, strukturierte Datentypen oder Rekursion, die alle im SQL:1999-Standard vorgesehen sind, tragen zu einer erhöhten Anfragekomplexität bei [MS02]. Der Optimierer muss solche Besonderheiten differenziert handhaben. Manche dieser Konzepte können unter Umständen auf Basisoperatoren abgebildet werden. Beispielsweise lassen sich sowohl Trigger als auch Stored Procedures als Folgen von „normalen SQL-Anweisungen“ darstellen. Der Fokus dieser Arbeit liegt jedoch nicht auf obigen Besonderheiten, sondern auf Anfragen mit Operatoren wie Selektion, Projektion, Verbund oder Gruppierung, da sie die Mehrzahl aller Anfragen repräsentieren.

## 3.4 Ausführungsplanspezifische Parameter

Während der Optimierung werden zu einem Anfragegraphen verschiedene Ausführungspläne generiert und bewertet. Auf dieser Ebene existieren weitere Einflussfaktoren, die der reinen SQL-Anfrage noch nicht entnommen werden können, sehr wohl aber Auswirkungen auf die erzielten Schätzergebnisse haben. Auf zwei Parameter wird im Folgenden genauer eingegangen: die Struktur des Ausführungsplans sowie die verwendeten Planoperatoren und deren Realisierungen.

### 3.4.1 Struktur des Ausführungsplans

Die Kosten eines Ausführungsplans sind abhängig von der Anzahl und der Größe der Zwischenergebnisse. Die Verarbeitung großer Objektströme erfordert einen erhöhten E/A- und CPU-Aufwand. Es ist deshalb von Vorteil, Anzahl und Größe der Zwischenergebnisse möglichst klein zu halten. Ausschlaggebend dafür ist die Struktur des Ausführungsplans.

#### Anzahl der Knoten

Die Anzahl der Knoten im Ausführungsplan korreliert mit der Komplexität der Anfrage. Eine Anfrage mit einem Mehrwegeverbund über  $n$  Relationen z. B. führt zu einem Plan mit mindestens  $(n - 1)$  Knoten und erfordert  $(n - 1)$  Abschätzungen für die zugehörigen Verbundselektivitäten. Ein Plan mit einer hohen Anzahl an Knoten erfordert mehr Kostenschätzungen als ein Plan mit weniger Knoten. Es gibt mehr Stellen, an denen Schätzfehler auftreten können. Diese werden über nachfolgende Knoten propagiert und können sich durch darauf aufbauende Schätzungen vergrößern.

Optimierungsmaßnahmen im Rahmen der Anfragetransformation, welche die Zahl der Knoten reduzieren, wirken sich dabei nur bedingt auf die Schätzungen aus, wie die folgenden zwei Beispiele zeigen:

1. Wenn die Zahl der Knoten durch Zusammenfassen mehrerer Operatoren zu neuen logischen Operatoren verringert wird, z. B. indem zwei aufeinander folgende Selektionen zu einer Selektion zusammengefasst werden, so muss nur noch eine einzelne Schätzung durchgeführt werden. Diese ist dafür schwieriger, denn sie enthält beide Selektivitätsprädikate. Die Schätzergebnisse unterscheiden sich in der Summe der Kardinalitäten, da bei einer zweistufigen Schätzung insgesamt mehr Tupel verarbeitet werden müssen. Die Qualität des resultierenden Schätzergebnisses ist jedoch gleich.
2. Eine besonders effektive Reduzierung der Knotenzahl wird durch die Identifikation gemeinsamer Teilbäume erreicht. Das Zwischenergebnis wird einmal für den Teilbaum berechnet. Ebenso wird die Kostenschätzung nur einmal durchgeführt. Sie würde aber auch bei mehrmaliger Anwendung auf den gleichen Teilbaum jedes Mal zum selben Ergebnis führen. Auch hier reduziert die Optimierungsmaßnahme zwar die Knotenanzahl und dadurch die Summe der Kardinalitäten, beeinflusst aber nicht die Qualität der Schätzung.

### Größe der Zwischenergebnisse

Neben der Knotenzahl ist die Größe der Zwischenergebnisse ausschlaggebend für die entstehenden Kosten. Die Größe wird beeinflusst durch die Reihenfolge der verwendeten Operatoren. Eine erste Optimierung wird bereits bei der Anfragerestrukturierung durchgeführt, indem z. B. Selektionen und Projektionen so früh wie möglich ausgeführt werden. Es verbleiben allerdings noch Freiheitsgrade bei der Wahl der Ausführungsreihenfolge logischer Operatoren. Ein Beispiel hierfür ist die Verbundreihenfolge. Die Variation der Reihenfolgen führt zu unterschiedlichen Kardinalitäten wie nachfolgendes Beispiel zeigt:

```
(Q4) SELECT *
      FROM Ort O, Niederlassung N, Mitarbeiter M
      WHERE N.Standort = O.Name AND
           M.NiederlassungId = N.Id AND
           O.Einwohner > 250 000 AND
           M.Gehalt > 40 000
```

Angenommen, die Daten haben folgende Eigenschaften:

- $Card(M) = 2000 \cdot n$ ,  $Card(N) = n$  und  $Card(O) = 400 \cdot n$
- $s(O.Einwohner > 250\,000) = 0,1$  und  $s(M.Gehalt > 40\,000) = 0,5$

Die Abbildungen 3.4 und 3.5 zeigen zwei mögliche Ausführungspläne mit unterschiedlichen Verbundreihenfolgen. In eckigen Klammern stehen die zugehörigen Kardinalitäten. Die Kardinalitätsschätzung für das Ergebnis ist in beiden Plänen gleich ( $1000 \cdot n$ ). Für die inneren Knoten werden jedoch unterschiedliche Werte geschätzt. In Plan 1 wird das relativ große Zwischenergebnis nach der Selektion auf M von  $2000 \cdot n$  durch den gesamten Plan propagiert. Es muss ein größerer Objektstrom bearbeitet werden als bei Plan 2, was zu höheren Ausführungskosten führt.

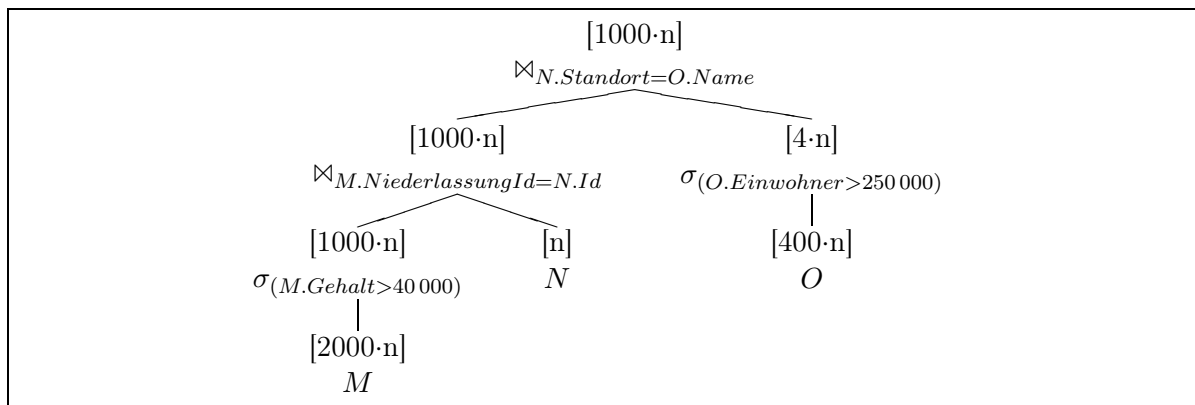


Abbildung 3.4: Plan mit Ausführungsreihenfolge 1

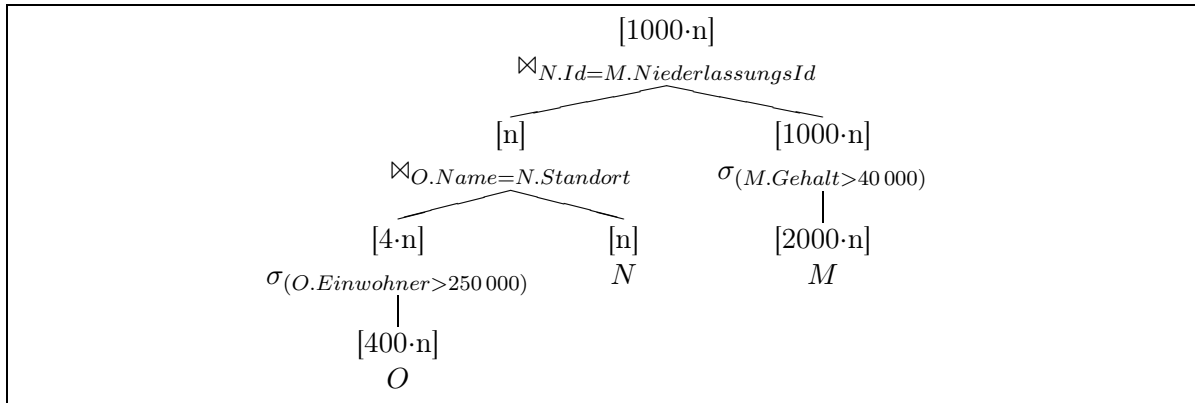


Abbildung 3.5: Plan mit Ausführungsreihenfolge 2

Der Unterschied der Ausführungskosten beider Pläne lässt sich quantitativ beschreiben. Dabei wird angenommen, dass die Ausführungskosten proportional zur Summe aller Zwischenergebnisse im Plan sind. Die Annahme ist darin begründet, dass mit der Anzahl der zu verarbeitenden Tupel auch beide Parameter des Kostenmodells (E/A-Zugriff und CPU-Zeit) steigen. Die Summe aller Zwischenergebnisse innerhalb des Plan  $x$  sei mit  $\sum Card(Plan_x)$  bezeichnet. Das Verhältnis der Kosten beider Pläne ergibt sich wie folgt:

$$\begin{aligned} \sum Card(Plan_1) &= 3004 \cdot n \\ \sum Card(Plan_2) &= 2005 \cdot n \\ \Rightarrow Kosten(Plan_1) &\approx 1,5 \cdot Kosten(Plan_2) \end{aligned}$$

Der Unterschied der beiden Kostenschätzungen zeigt, dass sich die Reihenfolge der Operatoren auf die Qualität der Schätzergebnisse auswirken kann. Angenommen, bei der Kostenschätzung für den von einem Knoten  $X$  produzierten Objektstrom tritt ein Schätzfehler auf. Befindet sich der Knoten in der Nähe der Blätter des Graphen, so verwendet jeder Vaterknoten für die folgenden Kostenschätzungen den fehlerhaften Wert. Der Fehler kann sich durch nachfolgende Schätzungen weiter vergrößern. Ist die Struktur des Ausführungsplans derart, dass Knoten  $X$  nahe der Wurzel ist, dann ist das Fehlerpotenzial geringer.

Das Problem der optimalen Verbundreihenfolge ist NP-vollständig und kann somit zum Optimierungszeitpunkt nicht effizient gelöst werden [IK84]. Verschiedene Verfahren zur effizienten Bestimmung wurden vorgeschlagen. Einen Überblick findet man in der Arbeit von Steinbrunn u. a. [SMK97].

### 3.4.2 Planoperatoren und deren Realisierungen

Bei der Anfragetransformation werden die logischen Operatoren des Anfragegraphen durch physische Operatoren – so genannte Planoperatoren – ersetzt. Planoperatoren kann man als kleinste Verarbeitungszelle innerhalb des Ausführungsplans betrachten. Sie sind Verbraucher

und Erzeuger von Objektströmen. Ein Planoperator kann mehrere logische Operatoren zusammenfassen. Durch eine Parametrisierung wird eine konkrete Methode zur Ausführung der Operation festgelegt, die vom Optimierer mit einem Kostenmodell bei der Planauswahl bewertet werden muss [Mit95]. Anhand der Anfrage Q5 wird dieser Transformationsschritt erläutert.

```
(Q5) SELECT  Name, Beruf
      FROM    ABT A, PERS P
      WHERE   A.Anr = P.Anr AND A.Ort = 'Stuttgart'
```

Einen zugehörigen Anfragegraphen mit logischen Operatoren zeigt Abbildung 3.6. In Abbildung 3.7 sieht man, wie der logische Selektions- und Projektionsoperator zu einem physischen Operator ACCESS zusammengefasst wird, der über Projektions- und Selektionsprädikatslisten parametrisiert ist. Ebenso wird aus dem Verbundoperator und der nachfolgenden Projektion ein einzelner Planoperator JOIN mit dem Verbundprädikat als Parameter. Für einen Operator existieren mehrere Methoden zur Realisierung. Beispielweise kann für den JOIN-Operator ein Nested-Loop-Verbund, Misch-Verbund oder ein Hash-Verbund verwendet werden. Der ACCESS-Operator kann als Tabellenzugriff oder Indexzugriff ausgeführt werden.

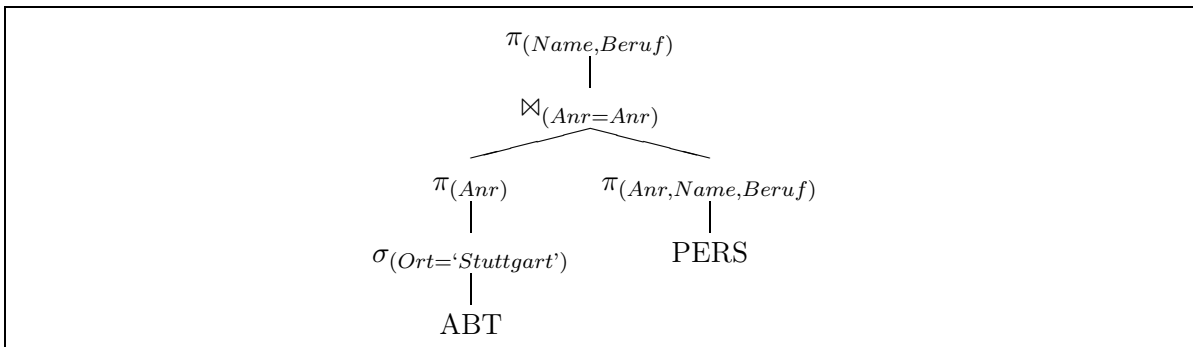


Abbildung 3.6: Anfragegraph mit logischen Operatoren

Die Aufwände für die verschiedenen Methoden eines Operators sind unterschiedlich. Betrachten wir den Verbund<sup>2</sup> von zwei Relationen R und S mit  $\text{Card}(R)=\text{Card}(S)=N$ :

- Nested-Loop-Verbund:  $O(N^2)$
- Misch-Verbund:  $O(N \cdot \log N)$
- Hash-Verbund:  $O(N)$

Für den Aufwand sind die existierenden Zugriffspfadstrukturen von Interesse. So kann der Aufwand des Nested-Loop-Verbunds bei Verwendung einer Indexstruktur auf  $O(N \cdot \log N)$  gesenkt werden. Ist für nachfolgende Operatoren beispielsweise eine Sortierung nach dem Verbundattribut notwendig, so wird die Wahl auf den Misch-Verbund fallen, da der Ausgabestrom nach der Mischphase sortiert vorliegt. Sind bereits die Eingabeströme sortiert, so reduziert sich die

<sup>2</sup>vgl. Härder und Rahm [HR01] für eine detaillierte Beschreibung der verschiedenen Verbundmethoden

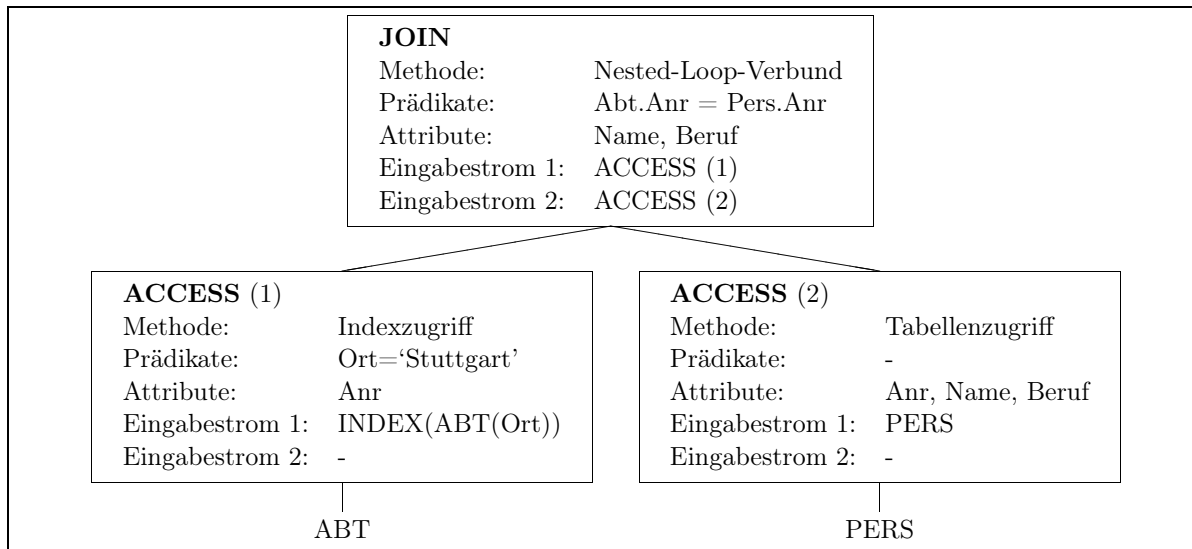


Abbildung 3.7: Ausführungsplan mit Planoperatoren

Komplexität des Misch-Verbundes auf  $O(N)$ . Für den Hash-Verbund spielt die Größe des verfügbaren Hauptspeichers eine Rolle. Bei großer Kardinalität des Eingabestroms ist der Aufwand linear abhängig von der Anzahl der Partitionen  $p$ :  $O(p \cdot N)$ .

Es wird deutlich, dass die unterschiedlichsten Faktoren die Wahl der Implementierungsmethode beeinflussen und sich somit auch auf die Kostenschätzung auswirken. Die geschätzten CPU- und E/A-Kosten sind abhängig von der Komplexität des verwendeten Planoperators. Unberührt davon bleiben jedoch die geschätzten Kardinalitäten. Die Größe des Ausgabestroms eines logischen Operators ist unabhängig von der gewählten Implementierung.

## 3.5 Laufzeitparameter

In Abschnitt 2.3 wurde gezeigt, welche Rolle Statistiken im Datenbankkatalog bei der Optimierung spielen. Histogramme mit Informationen über die Datenverteilung verbessern die Kostenschätzung. Sie repräsentieren aber stets den Datenzustand des Zeitpunkts, an dem die Statistiken erstellt wurden. Daten sind jedoch Änderungen unterworfen, so dass sie von den anfangs korrekten Statistiken im Laufe der Zeit abweichen können. Zur Laufzeit liegt somit nur ein Abbild der Vergangenheit vor. Neben dem tatsächlichen Datenzustand existieren noch weitere Laufzeiteigenschaften, die zum Übersetzungszeitpunkt unbekannt sind und deshalb nicht oder nur unzureichend genau berücksichtigt werden können. Hierzu gehören die Verfügbarkeit der Ressourcen sowie die Laufzeitbindungen von Host-Variablen.

### 3.5.1 Verfügbarkeit der Ressourcen

Die aktuelle Systemlast beeinflusst in erster Linie die Ausführungsdauer der Anfrage. Bei einem zentralisierten DBS kann die Last nicht auf andere Rechnerknoten verteilt werden. Wenn

beispielsweise mehrere Benutzer parallel auf die Datenbank zugreifen, steigt die Systemlast. Ressourcen wie Hauptspeicher, CPU-Zeit oder E/A-Zugriffe sind aber beschränkt. Diese Faktoren wirken sich nicht nur auf die Ausführungsdauer der Anfrage aus, sondern beeinflussen auch die kostenbasierte Planauswahl. In Abschnitt 2.2 wurde das Kostenmodell als gewichtete Funktion zwischen E/A-Kosten und CPU-Kosten beschrieben [HR01]. Bei der Kostenschätzung für einen Ausführungsplan zum Übersetzungszeitpunkt kennt der Optimierer die tatsächliche Verfügbarkeit der Ressourcen zur späteren Laufzeit nicht. Es ist beispielsweise anzunehmen, dass ein Plan, der einen relativ geringen E/A-Aufwand erfordert, dafür aber sehr rechenintensiv ist, einem anderen Plan mit umgekehrten Eigenschaften vorgezogen wird, wenn die kritische Ressource des Systems der Zugriff auf Sekundärspeicher ist. Die vom Optimierer gewählte Lösung ist aber nur unter der Voraussetzung, dass der Sekundärspeicherzugriff den Engpass darstellt, eine gute Lösung. Ist die CPU-Last des Systems zum Ausführungszeitpunkt unerwartet hoch, so wäre die gewählte Lösung nicht mehr optimal. Das Gleiche gilt, falls der Systemengpass beispielsweise durch schnellere Speichermedien beseitigt wird. In diesem Fall müsste anhand eines neuen Kostenmodells eine Neuoptimierung aller Anfragen durchgeführt werden.

#### 3.5.2 Laufzeitbindungen von Host-Variablen

Eine weitere Unbekannte für den Optimierer sind Host-Variablen, die erst zur Laufzeit gebunden werden. Host-Variablen sind Variablen einer Wirtssprache wie beispielsweise COBOL, C, C++ oder Java, die in einer SQL-Anfrage referenziert werden. Das ist z. B. der Fall wenn ein Attributwert mit einer Benutzervariablen verglichen wird, deren Wert erst zur Laufzeit festgelegt wird. Die Planauswahl zum Übersetzungszeitpunkt wird zu keiner Lösung führen, die für alle möglichen Variablenwerte zufrieden stellend ist. Die Qualität einer solchen statischen Kostenschätzung ist abhängig von dem Wert, den die Benutzervariablen zum Zeitpunkt der Ausführung annehmen. Erst dann liegt die tatsächliche Selektivität des Prädikats fest. Statische Optimierungsansätze stoßen hier an ihre Grenzen, insbesondere dann, wenn die Selektivitäten für verschiedene Werte stark variieren. Cole und Graefe haben ein dynamisches Verfahren vorgeschlagen, bei dem zum Übersetzungszeitpunkt eine Vorauswahl möglicher Pläne getroffen wird und erst zur Laufzeit der endgültige Plan abhängig von der Selektivität bestimmt wird [CG94].

## 4 Vorhandene Bewertungsansätze

Die Bewertung der Schätzverfahren zielt darauf ab, eine Aussage darüber zu machen, wie genau die geschätzten Werte für einen konkreten Ausführungsplan sind. Zwei Punkte müssen dabei unterschieden werden. Das ist zum einen die Bewertungsaussage an sich. Wie lässt sich der Fehler möglichst zuverlässig erkennen und beschreiben? Zum anderen die Frage nach der Verwendung dieser Information: Wie wird das gewonnene Wissen über die Qualität der Kostenschätzung angewandt? Wie kann davon profitiert werden? Nachfolgend sollen die Qualitätsaussagen von Schätzwerten im Vordergrund stehen, während die Verwendung der Information nur kurz beschrieben wird. Mögliche Anwendungsszenarien werden am Schluss der Arbeit in Kapitel 6 vorgestellt.

Die Bewertung der Schätzqualität ist abhängig von dem verwendeten Schätzverfahren, durch das die Werte entstanden sind. Von Interesse sind in dieser Arbeit gängige Methoden aus aktuellen DBS, also die Kostenschätzung auf Basis von Statistiken und Histogrammen im Datenbankkatalog. Nach einleitenden Betrachtungen über Klassifikationsmöglichkeiten von Bewertungsansätzen werden bekannte Ansätze aus der Literatur vorgestellt. Eine Beurteilung dieser Ansätze findet auf Basis der durchgeführten Messungen und Messergebnisse in Kapitel 6 statt.

### 4.1 Klassifikation der Bewertungsansätze

Bewertungsverfahren lassen sich auf unterschiedliche Weise klassifizieren. Eine Möglichkeit ist der Zeitpunkt, zu dem die Bewertung der Schätzergebnisse stattfindet: statisch zum Übersetzungszeitpunkt oder dynamisch zur Laufzeit. Auch hybride Ansätze existieren.

- *Statische Ansätze*: Welche Möglichkeiten existieren bei der statischen Optimierung zum Übersetzungszeitpunkt, die Qualität der geschätzten Werte eines Ausführungsplans zu bestimmen?
- *Dynamische Ansätze*: Zur Laufzeit ergeben sich weitere Möglichkeiten, da hier Informationen verwendet werden können, die zum Übersetzungszeitpunkt nicht verfügbar sind. Bei teilweiser Ausführung der Anfrage sind beispielsweise die Kardinalitäten von Zwischenergebnissen bekannt, die dann mit den geschätzten Werten verglichen werden können.

Der Großteil der Optimierung in DBS geschieht zum Übersetzungszeitpunkt. Können hier bereits Aussagen über die Qualität der Schätzergebnisse getroffen werden, so lässt sich der Optimierungsprozess weiter verbessern, ohne dass es Auswirkungen auf das Laufzeitverhalten hat. Der Fokus dieser Arbeit liegt deshalb auf den statischen Ansätzen. Ein weiterer Grund ist, dass im nachfolgenden Kapitel versucht wird, die vorgestellten Ansätze durch Messungen zu belegen bzw. zu neuen Bewertungsmöglichkeiten auf Basis der erzielten Messergebnisse zu gelangen. Die kostenbasierte Planauswahl des dabei verwendeten DBS (DB2 von IBM) geschieht ebenfalls zum Übersetzungszeitpunkt.

Dennoch werden nachfolgend auch kurz dynamische Ansätze beschrieben, da diese ebenfalls von der Bewertung einer statischen Kostenschätzung profitieren können. Die Suche nach einem optimalen Ausführungsplan zur Laufzeit kann dieselben Bewertungsverfahren der statischen Planauswahl anwenden, da die verfügbaren Informationen, die zur Bewertung der Schätzung zum Übersetzungszeitpunkt herangezogen werden, eine echte Teilmenge der vorhandenen Laufzeitinformationen sind.

Eine andere Klassifikation unterscheidet nach der Bewertung der Kostenschätzung. Es sind qualitative und quantitative Aussagen möglich.

- *Qualitative Bewertung*: Von Interesse sind Angaben über das Fehlerpotenzial einer Schätzung. Das sind z. B. Aussagen der Form gut/schlecht oder genau/ungenau. Pläne mit hohem Fehlerpotenzial können damit bereits aussortiert werden, ohne exakte mathematische Angaben über die Größe des Fehlers machen zu müssen.
- *Quantitative Bewertung*: Die quantitative Bewertung liefert exakte Aussagen über die Schätzergebnisse, z. B. obere Schranken für den Schätzfehler.

In der Literatur findet man nur wenig qualitative Bewertungsansätze. Die vorhandenen quantitativen Aussagen beziehen sich auf spezielle Schätzverfahren, für deren Fehler mathematische Obergrenzen angegeben werden. In dieser Klasse sind nur die histogrammbasierten Ansätze von Interesse. Im Folgenden werden die existierenden Verfahren vorgestellt. Eine Einteilung nach der obigen Klassifikation mit Vertretern der Klassen aus der Literatur zeigt Abbildung 4.1.

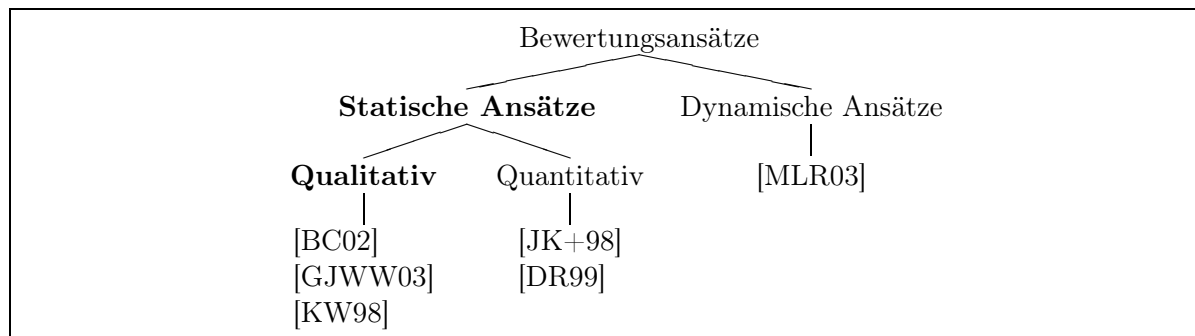


Abbildung 4.1: Klassifikation der Bewertungsansätze

Der Fokus dieser Arbeit liegt auf den statischen, qualitativen Bewertungsansätzen. Existierende Verfahren werden im folgenden Abschnitt vorgestellt. Die anschließenden Messungen (s. Kapitel 5) zielen ebenfalls auf Aussagen über diese Klasse ab.

## 4.2 Statische, qualitative Bewertung

Statische Ansätze zur Bewertung von Kostenschätzungen sind besonders wertvoll, da bereits zum Übersetzungszeitpunkt die Informationen für weitere Optimierungen genutzt werden können. Die beiden folgenden Abschnitte zeigen zwei Möglichkeiten für eine qualitative Abschätzung aus Arbeiten auf dem Gebiet der Anfrageoptimierung.

### 4.2.1 Statistiken für Zwischenergebnisse

Eine Möglichkeit, den Prozess der Anfrageoptimierung zu verbessern, besteht darin, dem Optimierer für die Schätzungen weitere Informationen bereitzustellen. Ein Ansatz sind Statistiken für Zwischenergebnisse, wie sie in der Arbeit von Bruno und Chaudhuri beschrieben werden [BC02]. Im Gegensatz zur Verwendung von propagierten Basisstatistiken liefern diese für Kostenschätzungen von Zwischenknoten genauere Werte.

Der gleiche Ansatz wird in „Statistics on Views“ von Galindo-Legaria u. a. verfolgt [GJWW03]. Die Autoren zielen darauf ab, Teile der Anfrage auf Sichten abzubilden, für die zuvor Statistiken erstellt wurden. Auch dabei wird die Verwendung ungenauer Basisstatistiken vermieden. Der Optimierer wird um eine Transformationsregel erweitert, welche die Abbildung der Anfragen auf Sichten übernimmt.

Die Transformationsregel enthält eine Bewertungskomponente, mit der entschieden wird, wie zuverlässig die Schätzwerte der einzelnen Operatoren sind. Für die geschätzten Zwischenergebnisse werden zwei Bewertungen unterschieden: *genaue Kardinalität* und *ungenau Kardinalität*. Ein Maß für die Genauigkeit ist nicht gegeben, weshalb man von einer qualitativen Bewertung sprechen kann.

Die Bewertung beruht darauf, die Kardinalitäten aller Basistabellen und Sichten mit *genau* zu kennzeichnen, alle anderen Elemente mit *ungenau*. Durch Transformation wird ein Plan gesucht mit möglichst vielen *genauen* Knoten, d. h. ein Plan mit hoher Schätzgenauigkeit. Abbildung 4.2 zeigt ein Beispiel mit zwei alternativen Ausführungsplänen. Die mit einem Stern versehenen Knoten im Plan sind die Stellen, für die eine genaue Schätzung der Kardinalitäten möglich ist. In Plan 1 sind das die Basistabellen. Für diese existieren Statistiken im Datenbankkatalog mit exakten Kardinalitäten. Für den darauf aufsetzenden Verbundoperator ist keine genaue Schätzung mehr möglich, im Gegensatz zu der Verwendung von Sichten im Ausführungsplan 2. Die Qualität der Schätzergebnisse dieses Plans ist höher, da Teile der Anfrage auf eine materialisierte Sicht abgebildet werden können, für die Statistiken existieren. Die geschätzte Kardinalität der Sicht wird deshalb als genau angenommen. Weitere Transformationen des Plans sind denkbar und können die Verwendung weiterer Sichten ermöglichen, wodurch die Schätzqualität nochmals erhöht wird.

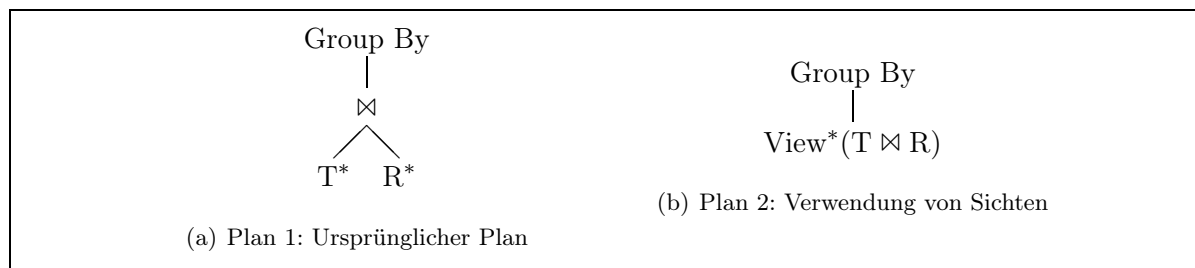


Abbildung 4.2: Bewertung von Schätzergebnissen in „Statistics on Views“

*Verwendung der Genauigkeitsinformation:* Die Aussagen über die Qualität der Schätzungen werden hier direkt für die statische Optimierung verwendet, um Pläne mit möglichst genauen Schätzwerten auszuwählen.

### 4.2.2 Bewertung durch Ungenauigkeitspotenzial

Die Arbeit „Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans“ von Kabra und DeWitt zeigt Strategien zur dynamischen Reoptimierung einer Anfrage [KW98]. Hierzu werden während der Ausführung Statistiken gesammelt, um die Suboptimalität eines Plans zu erkennen und darauf zu reagieren. Diese Statistiken enthalten z. B. reale Kardinalitäten sowie Informationen über die Datenverteilung von Zwischenergebnissen. Es ist zur Laufzeit zu aufwändig, für jeden Knoten diese Statistiken zu sammeln. Deshalb stellt sich die Frage, an welchen Stellen solche Informationen nützlich sind, d. h. wo die Wahrscheinlichkeit für Schätzfehler am größten ist.

Ein Algorithmus platziert im Ausführungsplan so genannte Sammelpunkte, an denen zur Laufzeit Statistiken erhoben werden. Die Auswahl dieser kritischen Stellen basiert auf einer qualitativen Bewertung. Es wird ein *Ungenauigkeitspotenzial* (UP) für die Qualität der Schätzergebnisse definiert. Drei Ausprägungen werden dabei unterschieden: *niedrig*, *mittel* und *hoch*. Die Bewertung verläuft ebenso wie die Kostenschätzung Bottom-Up von den Blattknoten zur Wurzel. Folgende Bewertungsregeln werden dabei verwendet:

- *UP = niedrig*:
  - Serielles Histogramm auf einem Attribut einer Basistabelle.
  - Bestimmung der Anzahl eindeutiger Werte eines Attributs einer Basistabelle. Es existieren Informationen darüber im Datenbankkatalog.
- *UP = mittel*:
  - Equi-Width- und Equi-Depth-Histogramm auf einem Attribut einer Basistabelle.
- *UP = hoch*:
  - Attribut einer Basistabelle ohne Histogramm.
  - Bestimmung der Anzahl eindeutiger Werte eines Attributs von Zwischenergebnissen.
  - Selektionsprädikat mit benutzerdefinierten Methoden.
  - Verbundoperation (außer Gleichverbund).
- Abhängigkeiten zwischen  $UP(Input)$  und  $UP(Output)$ :
  - Selektionsprädikat mit einem Attribut:  $UP(Output) = UP(Input)$ .
  - Selektionsprädikat mit mehreren Attributen einer Relation sowie ein Gleichverbund ohne Schlüsselattribute erhöhen das UP um eine Stufe:  
*niedrig* → *mittel*, *mittel* → *hoch*, *hoch* → *hoch*.
  - Gleichverbund mit Schlüsselattributen:  
 $UP(Output) = \max\{UP(Input_1), \dots, UP(Input_n)\}$
  - $UP(Output)$  eines Aggregationsoperators ist gleich dem UP des Operators, der das Wissen über die Anzahl eindeutiger Werte nach der zu gruppierenden Spalte liefert. Das führt bei der Anwendung auf einer Basistabelle zu  $UP = niedrig$ , in allen anderen Fällen zu  $UP = hoch$ .

Es fällt auf, dass das UP nur bei zwei Operationen mit *niedrig* eingestuft wird. Das geschieht bei Operationen auf Basistabellen unter Verwendung zuverlässiger Statistiken. In ihrer Arbeit haben Kabra und DeWitt für ihre dynamische Reoptimierung eine Performancesteigerung bei komplexen Anfragen von 10 %–30 % gegenüber der normalen Ausführung nachgewiesen. Diese Zahlen sind nicht als Maß für den Bewertungsalgorithmus zu sehen, der lediglich über die Platzierung der Sammelpunkte entscheidet. Eine Aussage über den Beitrag der Bewertungskomponente zur gesamten Performancesteigerung wird von Karba und DeWitt nicht gemacht.

*Verwendung der Genauigkeitsinformation:* Die Aussagen über die Qualität der Schätzungen werden für die dynamische Reoptimierung genutzt. Auf Basis der Bewertungen werden die Stellen im Ausführungsplan identifiziert, an denen eine Überprüfung der Schätzwerte stattfinden soll.

### 4.3 Statische, quantitative Bewertung

Wie im vorherigen Abschnitt beschrieben, können qualitative Bewertungsaussagen im Sinne von *genau* und *ungenau* auf verschiedene Schätzungen angewandt werden. Anhand von Regeln lassen sich Aussagen darüber machen, wie die Qualität der Schätzergebnisse durch den Plan propagiert wird. Bei einer quantitativen Bewertung geht es dagegen um eine mathematische Beschreibung des Schätzfehlers. Dazu gehören z. B. Fehlerschranken für den schlechtesten Fall. Eine mögliche Fragestellung könnte lauten: Wie groß ist der maximale Schätzfehler, der bei der Kostenschätzung einer Selektion auf Attribut A mit der Datenverteilung X unter der Annahme der Datenunabhängigkeit auftreten kann? Für eine Antwort benötigt man das Wissen über die Attributverteilung – eine Information, die für eine qualitative Analyse nicht nötig war. Daraus lässt sich schließen, dass eine quantitative Bewertung an eine konkrete Schätzung gebunden ist.

Da ein Großteil der Kostenschätzung auf der Verwendung von Histogrammen beruht, werden in den folgenden beiden Abschnitten zwei Möglichkeiten vorgestellt, wie sich der dabei entstehende Schätzfehler mathematisch beschreiben lässt.

#### 4.3.1 Maximale Abweichung pro Bucket

In ihrer Arbeit „Optimal Histograms with Quality Guarantees“ haben Jagadish u. a. die Kostenschätzung basierend auf Histogrammen untersucht [JK+98]. Sie stellen Algorithmen vor, die bei vorgegebenem Speicherplatz zu einer optimalen Bucket-Einteilung für das Histogramm gelangen. Darüber hinaus wird die Speicherung der Datenverteilung in Form von Histogrammen um eine Information erweitert, die es ermöglicht, Aussagen über die Qualität der damit durchgeführten Selektivitätsabschätzungen zu machen.

Die Qualität einer Selektivitätsabschätzung ist definiert als Obergrenze für den absoluten Fehler. Gegeben sei Relation  $R$ , die Frequenz des Wertes  $k$  von Attribut  $a$  aus  $R$  mit  $f_a[k]$  und die Einteilung des Histogramms in die Buckets  $b_1, b_2, \dots, b_n$ . Für ein Prädikat der Form  $a = k$ , dessen Größe mit der durchschnittlichen Frequenz  $f_i$  des Buckets  $b_i$  mit dem Wert  $k$  approximiert wird, ergibt sich der absolute Fehler  $e_k$  zu

$$e_k = |f_a[k] - f_i|$$

und der maximale Fehler in Bucket  $b_i$  zu

$$E_i = \max\{e_k | k \in b_i\}$$

Speichert man im Histogramm für jeden Bucket  $b_i$  zusätzlich den Wert  $E_i$ , so kann für eine Selektivitätsabschätzung dieser maximale Fehler als Garantie für die Qualität des Schätzwerts geliefert werden.

Diese Fehlerbeschreibung lässt sich auch auf Bereichsprädikate der Form  $a \leq k$  anwenden. Zusätzlich zu den obigen Annahmen sei der kleinste und der größte Attributwert in  $b_i$  gegeben mit  $m$  und  $M$ . Es ergibt sich daraus die mit dem Histogramm geschätzte Kardinalität zu

$$Card_{hist} = \sum_{j=1}^{i-1} (|b_j| \cdot f_j) + (k - m + 1) \cdot f_i$$

Für die ersten  $i - 1$  Buckets entsteht kein Schätzfehler, da der Bereich durch die Anfrage vollständig abgedeckt wird. Der Schätzfehler in Bucket  $b_i$  kann mit der Formel für das Gleichheitsprädikat nach oben hin abgeschätzt werden, indem für jeden Wert in  $b_i$  der maximale Fehler  $E_i$  angenommen wird:

$$e_{b_i} = \min(k - m + 1, M - k + 1) \cdot E_i$$

Die vorgestellten Abschätzungen erfordern die Speicherung eines Fehlerwertes pro Bucket, was zu einem erhöhten Speicherbedarf führt. Es muss folglich abgewogen werden, in welcher Relation der benötigte Aufwand (Speicherplatz) zum erbrachten Nutzen (Schätzgenauigkeit) steht. Ist ausreichend Speicherplatz vorhanden, so kann die Speicherung weiterer Statistiken wie beispielsweise der durchschnittliche Fehler pro Bucket zu einer noch genaueren Beschreibung des Fehlers führen.

*Verwendung der Genauigkeitsinformation:* Die Fehlerschranke ist eine Erweiterung des Histogramms und wird bei Jagadish u. a. dazu verwendet, die Größe des Anfrageergebnisses genauer zu schätzen.

### 4.3.2 Maximale Abweichung pro Histogramm

Auf Grund des hohen Platzverbrauchs bei der Speicherung eines Fehlerwertes für jeden Bucket schlagen Donjerkovic und Ramakrishnan die Speicherung eines einzelnen Wertes pro Histogramm für die Fehlerbeschreibung vor [DR99]. Die Abschätzung einer Bereichsanfrage für Attribut  $a$  und Attributwert  $k$  der Form  $a \leq k$  lässt sich dann folgendermaßen beschreiben. Sei  $P_{real}(k)$  die kumulierte Wahrscheinlichkeitsverteilung der realen Daten und  $P_{hist}$  die der histogrammbasierten Daten. Die maximale Abweichung  $\epsilon$  beträgt dann

$$\epsilon = \max_{-\infty < x < \infty} |P_{real}(x) - P_{hist}(x)|$$

Dieser Wert wird für jedes Histogramm gespeichert. Er kann durch Sampling bei der Histogrammerstellung hinreichend genau berechnet werden. Mit  $\epsilon$  kann eine exakte Obergrenze für den Fehler im schlechtesten Fall angegeben werden.

*Verwendung der Genauigkeitsinformation:* Ebenso wie bei Jagadish verwenden Donjerkovic und Ramakrishnan die Genauigkeitsinformation für eine exaktere Schätzung der Kardinalität einer Anfrage.

## 4.4 Dynamische Bewertung

Bei der statischen Optimierung wird der Ausführungsplan für eine Anfrage zum Übersetzungszeitpunkt bestimmt. Dieses Konzept wird bei der dynamischen Optimierung erweitert. Zum Übersetzungszeitpunkt wird wie auch im statischen Fall eine Anfrageoptimierung durchgeführt. Das Ergebnis dieses ersten Schritts sind ein oder mehrere alternative Ausführungspläne. Zur Laufzeit findet dann eine Reoptimierung des gesamten Ausführungsplans oder von Teilen davon statt.

Es besteht dabei die Möglichkeit, Teile der Anfrage bereits auszuführen, um durchgeführte Kostenschätzungen überprüfen zu können. Abweichungen werden somit erkannt und alternative Pläne können verwendet werden. Hier ist abzuwägen, inwiefern es sich lohnt, einen bereits teilweise ausgeführten Plan zu verwerfen und auf die Fehler zu reagieren. Ein weiterer Vorteil der dynamischen Optimierung ist, dass bei der Bewertung von Kostenschätzungen Laufzeitparameter berücksichtigt werden können, die zum Übersetzungszeitpunkt nicht verfügbar sind (s. Abschnitt 3.5).

Das Gebiet der dynamischen Anfrageoptimierung und der Anfragereoptimierung ist groß. Da der Schwerpunkt der Arbeit auf den statischen Ansätzen liegen soll, wird hier beispielhaft nur ein Vertreter herausgesucht, der DB2 LEarning Optimizer von IBM.

### 4.4.1 LEO – LEarning Optimizer von DB2

Der *LEarning Optimizer von DB2* (LEO) unterscheidet sich von einem herkömmlichen Optimierer dadurch, dass während der Ausführung der Anfrage versucht wird, Schätzfehler zu erkennen und daraus zu lernen [MLR03, SLMK01].

Zum Übersetzungszeitpunkt wird ein optimaler Ausführungsplan ermittelt. Der Optimierer vergleicht zur Laufzeit die geschätzten Kardinalitäten für diesen Plan mit den tatsächlichen Kardinalitäten. Der Unterschied dieser beiden Werte dient als Maß zur Bewertung der Qualität der Kostenschätzung. Bei auftretenden Abweichungen kann der LEO aus seinen Fehlern lernen. Hierzu gibt es zwei Möglichkeiten:

- *Verzögertes Lernen:* Es werden Korrekturwerte für die Statistiken berechnet, die zur Verbesserung zukünftiger Kostenschätzungen eingesetzt werden.
- *Sofortiges Lernen:* Noch während der Ausführung wird die Suboptimalität des Ausführungsplans erkannt und Teile davon reoptimiert. Bereits berechnete Zwischenergebnisse können dafür verwendet werden.

Während das verzögerte Lernen statische und dynamische Elemente enthält, kann das sofortige Lernen vollständig der dynamischen Optimierung zugeordnet werden. Die beiden Teile werden nachfolgend beschrieben.

### 4.4.2 Verzögertes Lernen

Das verzögerte Lernen beruht auf der Annahme, dass zukünftige Anfragen den vorhergehenden Anfragen ähnlich sind, d. h., sie stimmen in mindestens einem Prädikat überein. Eine zentrale Feedbackschleife ermöglicht diesen Lernprozess (s. Abbildung 4.3). Sie besteht aus vier Komponenten: Monitor, Analyse, Feedback und Verwertung. Die vier Teile können unabhängig

voneinander arbeiten, was eine flexible Konfiguration des Systems ermöglicht. Die Lernfunktion kann dadurch auch ganz ausgeschaltet werden.

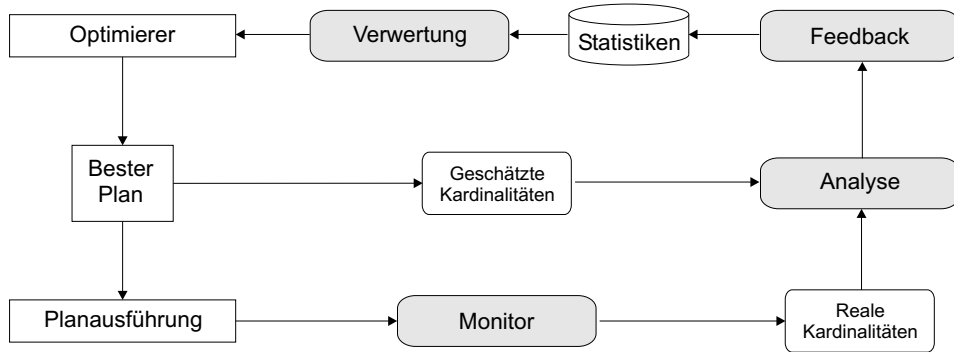


Abbildung 4.3: LEO's Feedbackschleife

Die Monitorkomponente überwacht die Ausführung der Anfrage und ermittelt die realen Kardinalitäten der Zwischenergebnisse im Ausführungsplan. Sie ist die einzige Komponente, die zur Laufzeit aktiv ist und diese beeinflusst. Der Mehraufwand für die Monitorkomponente beträgt weniger als 4% der Ausführungszeit. In der Analysephase werden die realen Kardinalitäten mit den geschätzten Werten im Plan verglichen. Schätzfehler werden in dieser Phase erkannt. Der Feedbackprozess berechnet anschließend Korrekturwerte, die zusätzlich zu den existierenden Statistiken im Datenbankkatalog gespeichert werden. Die alten Statistiken werden dabei nicht überschrieben. Die Verwertungskomponente nutzt diese Korrekturwerte, um die Optimierung zu verbessern, indem die fehlerhaften Schätzwerte korrigiert werden.

*Verwendung der Genauigkeitsinformation:* Die Informationen über die Qualität der Schätzungen aus der Analysephase werden für die Verbesserung vorhandener Statistiken verwendet, mit dem Ziel, zukünftig genauere Schätzwerte zu erhalten.

#### 4.4.3 Sofortiges Lernen

Bei großen Differenzen zwischen den geschätzten und den realen Kardinalitäten von Zwischenergebnissen ist der vorliegende Ausführungsplan vermutlich stark suboptimal. Ein aktueller Forschungsschwerpunkt des LEO-Projekts ist die Suche nach dynamischen Reoptimierungsmöglichkeiten, d. h. nach Möglichkeiten, den aktuellen Ausführungsplan zur Laufzeit zu verändern. Zu dem Zeitpunkt, an dem die Abweichungen der Kardinalitäten erkannt werden, sind bereits Teile der Anfrage ausgeführt. Es macht deshalb wenig Sinn, den gesamten Ausführungsplan zu verwerfen, da dadurch bereits berechnete Zwischenergebnisse verloren gehen. Um das zu verhindern, muss versucht werden, die vorliegenden Zwischenergebnisse in die Reoptimierung mit einzubeziehen. Die aktuellen Überlegungen von Markl u. a. beschäftigen sich in diesem Zusammenhang mit der Verwendung materialisierter Sichten [MLR03]. Bereits berechnete Zwischenergebnisse können als materialisierte Sichten definiert und für die Reoptimierung der Anfrage verwendet werden.

Noch ungelöst ist das Problem, wann genau die sofortige Reoptimierung durchgeführt werden soll. Dabei geht es um eine Abwägung zwischen der Zeit, welche für die Reoptimierung benötigt

wird, und dem erzielten Laufzeitgewinn durch einen verbesserten Ausführungsplan. Je mehr Teile der Anfrage bereits ausgeführt werden, desto kleiner wird der erzielbare Laufzeitgewinn einer Reoptimierung. Die Lösung dieses Problems ist vergleichbar mit einer Bewertungskomponente, welche die bereits vorliegenden Ergebnisse mit ihren Schätzwerten beurteilt und damit eine Reoptimierungsentscheidung trifft.

*Verwendung der Genauigkeitsinformation:* Die Wissen über Schätzfehler wird noch während der Ausführung für eine Reoptimierung des Ausführungsplans verwendet.

## 5 Messungen

Der zweite Teil dieser Arbeit beschreibt die empirische Herangehensweise an das Thema. Es werden Messungen mit dem Datenbanksystem DB2 von IBM durchgeführt. Für verschiedene Anfragen werden die vom Optimierer gewählten Ausführungspläne mit den darin enthaltenen Schätzwerten sowie den zugehörigen realen Werten für die Kardinalitäten von Zwischen- und Endergebnissen systematisch erfasst.

Es ist hierbei von Interesse, welcher Zusammenhang zwischen den Einflussfaktoren und den Abweichungen der geschätzten von den realen Werten besteht. Die Messergebnisse sollen dazu dienen, das Verständnis, das aus dem theoretischen Teil dieser Arbeit gewonnen wurde, zu überprüfen. Die Ergebnisse können dabei mit den theoretischen Überlegungen übereinstimmen oder mit diesen in Konflikt stehen. Darüber hinaus sollen nach Möglichkeit neue oder detailliertere Erkenntnisse über den Zusammenhang von Einflussfaktoren und Schätzergebnissen gewonnen werden.

### 5.1 Beschreibung der Messungen

Einleitend werden die durchzuführenden Messungen beschrieben. Grundlage für alle Messungen ist die dabei verwendete Messumgebung. Es werden Anfragen spezifiziert, welche in dieser Umgebung ausgeführt werden. Beim Ausführen der Anfragen werden die gewünschten Messwerte erfasst. Für jede Anfrage werden folgende Größen bestimmt:

- Der vom Optimierer gewählte Ausführungsplan. Dieser enthält:
  - Geschätzte Ausführungszeit
  - Geschätzte Kardinalitäten für Zwischen- und Endergebnisse
- Das Anfrageergebnis mit folgenden Werten:
  - Reale Ausführungszeit
  - Reale Kardinalitäten für Zwischen- und Endergebnisse

#### 5.1.1 Messumgebung

Dieser Abschnitt beschreibt die Umgebung, in welcher die Messungen durchgeführt werden. Dazu gehören Hardware und Software sowie eine Datenbasis, auf der die Anfragen operieren.

#### Hardware und Software

Für die Messungen wird ein Sun-Enterprise-4500/5500-Rechner mit dem Betriebssystem Solaris 8 verwendet. Der Rechner besitzt zwölf Prozessoren des Typs Sun UltraSPARC-II mit

einer Taktfrequenz von jeweils 400 MHz. Die Hauptspeichergröße beträgt 12 GB. Als virtueller Speicher stehen weitere 10,5 GB zur Verfügung.

Als DBS für die Messungen wird DB2 Universal Database in der Version v8.1.0.36 verwendet. Für die Durchführung der Anfragen und die Ermittlung der Messwerte kommen das DB2-Werkzeug *db2batch* sowie die DB2-Explain-Werkzeuge *db2exfmt* und *Visual Explain* zum Einsatz.

**db2batch:** Mit diesem Kommandozeilen-Werkzeug kann ein Benchmark erstellt und ausgewertet werden. Dazu liest *db2batch* die SQL-Anfragen entweder von der Standardeingabe oder aus einer Datei, führt diese aus und liefert die Ergebnismenge. Der Ausführungsprozess wird überwacht. Dabei können unterschiedliche Informationen gesammelt werden. Folgende Ausführungsvariante wurde für die Messungen verwendet:

```
db2batch -d tpch -f sql_input -r query_output -o e 2 r 100 p 1 -v on -s on
```

Beschreibung der Parameter:

- -d *db\_name*: Name der Datenbank.
- -f *file2name*: Input-Datei mit einer oder mehreren SQL-Anfrage(n).
- -r *outfile*: Datei für Anfrageergebnis.
- -o e *explain\_mode*: Modus für Explain-Informationen. Explain Modus 2: Anfrage wird ausgeführt und Explain-Informationen werden gesammelt. Diese werden in Explain-Tabellen abgelegt.
- -o r *rows\_out*: Anzahl der Zeilen der Ergebnismenge, die ausgegeben werden.
- -o p *perf\_detail*: Legt den Detaillierungsgrad für die zu sammelnden Informationen fest.
- -v *on/off*: Gibt an, ob Fehlermeldungen ausgegeben werden.
- -s *on/off*: Legt fest, ob eine Zusammenfassung über die ausgeführten Anfragen geschrieben wird. Diese enthält die Ausführungsdauer sowie Teile des Anfrageergebnisses.

**Explain-Werkzeuge *db2exfmt* und *Visual Explain*:** Explain-Werkzeuge werden dazu verwendet Explain-Informationen zu sammeln, darzustellen und auszuwerten. Dazu gehören folgende Daten des vom Optimierer gewählten Ausführungsplans einer SQL-Anfrage:

- Interne Repräsentation des Ausführungsplans mit verwendeten Tabellen, Indizes und physischen Operatoren.
- Entscheidungskriterien des Optimierers wie Selektivitätsabschätzungen und Kosten- und Kardinalitätsschätzungen für jeden Operator.

Diese Daten werden auf Anforderung in so genannten Explain-Tabellen gespeichert. Dafür existieren zwei Möglichkeiten. Explain-Informationen können explizit über eine in SQL formulierte Explain-Anfrage angefordert werden (*EXPLAIN query\_statement*), oder sie werden implizit

bei der Ausführung einer oder mehrerer Anfragen durch das Werkzeug `db2batch` gesammelt (s. oben: Parameter `-o e`).

Mit den zugehörigen Explain-Werkzeugen können die Informationen extrahiert und für den Benutzer lesbar dargestellt werden. Visual Explain ist die grafische Version des Kommandozeilen-Werkzeugs `db2exfmt` und unterscheidet sich nur dadurch, dass Explain-Informationen ausschließlich für einzelne Anfragen ausgewertet werden können. Für die Messungen wurde das Werkzeug somit nur für Ad-hoc-Auswertungen verwendet. In allen anderen Fällen, und somit auch für die Dokumentation der Messergebnisse, kam `db2exfmt` zum Einsatz. Es liefert eine formatierte Textausgabe der Explain-Informationen. Das Programm wurde wie folgt aufgerufen:

```
db2exfmt -d tpch -w -1 -n % -s % -# 0 -g -o explain_output
```

Beschreibung der Parameter:

- `-d db_name`: Name der Datenbank.
- `-w timestamp`: Zeitstempel, der zu analysierenden Explain-Anfrage. Die Zahl `-1` ermittelt den aktuellsten Eintrag aus den Explain-Tabellen.
- `-n name`: Name der Explain-Anforderung.
- `-s schema`: Schema der Explain-Anforderung.
- `-# sectnbr`: Sektionsnummer. Die Zahl `0` fordert alle Sektionen an.
- `-g`: Gibt den Ausführungsplan mit Explain-Informationen aus.
- `-o outfile`: Datei für die Ausgabe der Explain-Informationen.

Beim Zugriff auf die Explain-Tabellen führt das DBS für die Abfrage von *timestamp*, *name*, und *schema* eine Selektion mit dem LIKE-Prädikat durch. Die Verwendung des Zeichens „%“ würde eigentlich mehrere Einträge selektieren. Da der Zeitstempel jedoch eindeutig ist, liefert der obige Programmaufruf von `db2exfmt` die Explain-Informationen zu genau einer Anfrage.

### Datenbasis

Für die Messungen wird ein Datenbankschema des *Transaction Processing Performance Council* (TPC) zu Grunde gelegt: das Schema des *TPC Benchmark H* (TPC-H). Der TPC-H bewertet die Leistungsfähigkeit von Decision-Support-Systemen. Er besteht aus 22 Ad-hoc-Anfragen sowie zwei Refresh-Funktionen zur Generierung neuer Datensätze und wird für Systeme mit den folgenden charakteristischen Eigenschaften verwendet [TPCH04]:

- Verwendung von Anfragen mit hoher Komplexität
- Verwendung großer Datenvolumina
- Systeme liefern Antworten auf kritische, betriebswirtschaftliche Fragestellungen

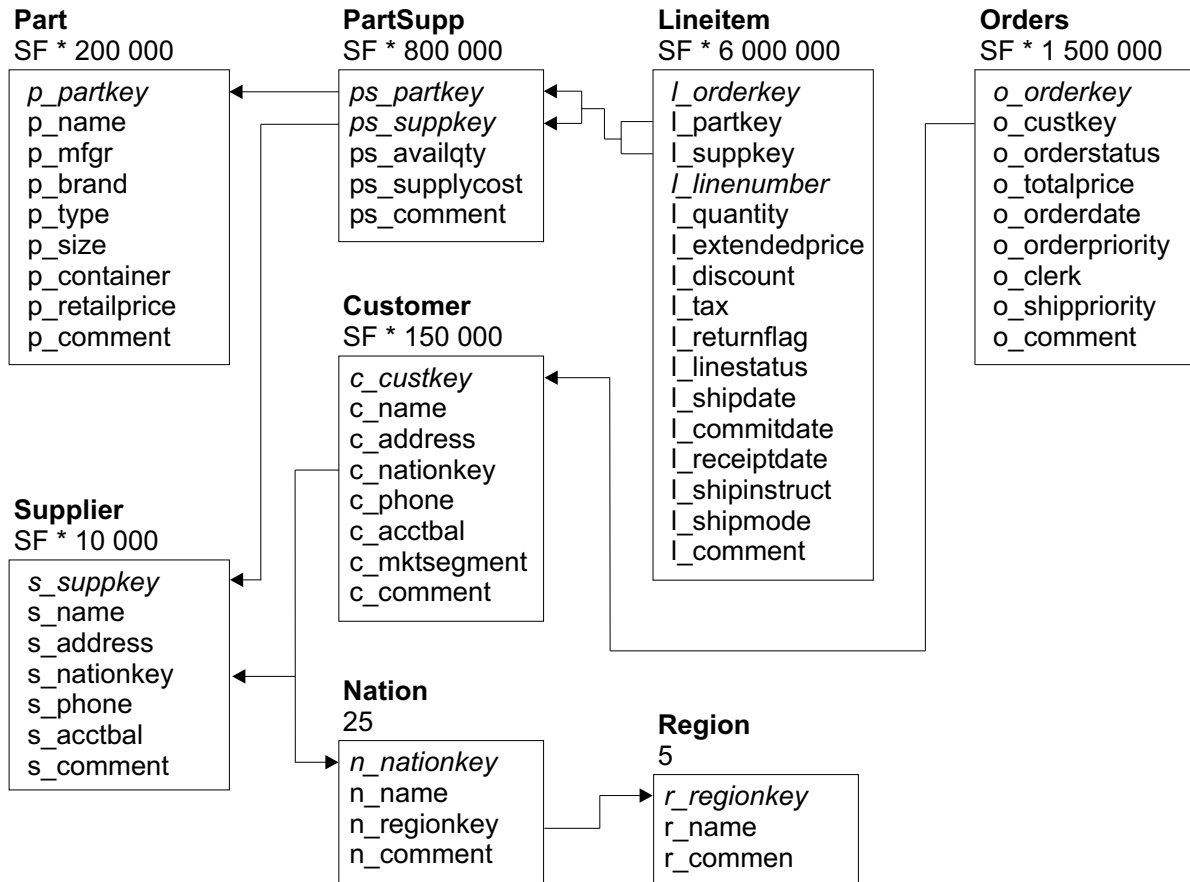


Abbildung 5.1: Schema des TPC Benchmark H

Auf Grund dieser Eigenschaften und der weiten Verbreitung des Benchmarks kann das Datenbankschema des TPC-H als repräsentativer Vertreter für reale Decision-Support-Systeme und Data-Warehouse-Anwendungen bezeichnet werden. Abbildung 5.1 zeigt das TPC-H-Schema mit den Tabellen und Beziehungen.

Im Rahmen einer Studienarbeit der Abteilung Anwendersoftware wurde eine Anwendung aus dem Bereich des *Online Analytical Processing* (OLAP) optimiert [Wag00]. Dabei wurde obiges TPC-H-Schema in eine typische Snowflake-Struktur überführt und um einige Dimensionstabellen erweitert. Das Projekt *Optimization and IntegRation of Business Intelligence Technology* (ORBIT) der Abteilung Anwendersoftware beschäftigt sich u. a. mit der Optimierung von Abfragesequenzen, die von OLAP-Anwendungen generiert werden [KSRM03]. Dabei wird dieses modifizierte Schema verwendet. Um auch aktuelle Anfragen dieses Projekts berücksichtigen zu können, wird für die Messungen ebenfalls das erweiterte Schema zu Grunde gelegt.

Mit einem *Skalierungsfaktor* (SF) können Datenmengen unterschiedlicher Größe erzeugt werden. Für die Messungen standen vier verschiedene Datenbasen zur Verfügung:

- $SF = 0, 1$ : Datenbasis TPCH1 mit einem Datenvolumen von ca. 100 MB
- $SF = 1$ : Datenbasis TPCH2 mit einem Datenvolumen von ca. 1 GB
- $SF = 3$ : Datenbasis TPCH3 mit einem Datenvolumen von ca. 3 GB
- $SF = 10$ : Datenbasis TPCH4 mit einem Datenvolumen von ca. 10 GB

Für die Messungen wurde die Datenbasis TPCH4 verwendet. Bei Anfragen, die auf Grund ihrer Komplexität auch nach mehreren Tagen mit den TPCH4-Daten noch kein Ergebnis lieferten, wurde auf kleinere Datenvolumina ausgewichen.

### 5.1.2 Definition der Anfragen

Um zu einem aussagekräftigen Ergebnis zu gelangen, werden Anfragen unterschiedlicher Komplexität an das DBS gestellt. Mit der Selektion, dem Verbund und der Gruppierung kommen drei Operatoren zum Einsatz, welche sich auf die Kardinalitäten von Zwischen- und Endergebnissen auswirken. Der Umfang der Messungen erstreckt sich über 53 Anfragen. Sie lassen sich in drei Gruppen einteilen ( $Q_t$ ,  $Q_e$ ,  $Q_o$ ), die nachfolgend erläutert werden.

Einen Überblick aller Anfragen zeigt die Tabelle 5.1. Für Aussagen über die Komplexität der Anfragen werden die folgenden vier Eigenschaften herangezogen:

- $T$ : Anzahl der referenzierten Tabellen.
- $P$ : Anzahl der verwendeten Prädikate. Ein Prädikat der Form  $\text{Attribut IN } ('x_1', \dots, 'x_n')$  wird dabei  $n$ -mal gezählt, das BETWEEN-Prädikat wird doppelt gezählt.
- $S$ : Die Schachtelungstiefe bezeichnet die Anzahl der ineinander geschachtelten SELECT-Anweisungen. Eine einzelne SELECT-Anweisung hat den Wert  $S = 0$ .
- $G$ : Gibt an, ob die Anfrage mindestens einen Gruppierungsoperator besitzt:  $G \in \{\text{Ja, Nein}\}$ .

Man sollte beachten, dass diese Angaben nur eine grobe Beschreibung der Anfragekomplexität ermöglichen, da z. B. ein einfaches Vergleichsprädikat gleich gezählt wird wie ein EXISTS- oder LIKE-Prädikat, deren Kardinalitätsschätzungen für den Optimierer schwieriger sind.

### Anfragen des TPC-H

Für die Messungen wurde das Datenbankschema des TPC Benchmark H gewählt (s. Abbildung 5.1). Das ermöglicht die Verwendung der zugehörigen Anfragen. Sie sind repräsentativ für Fragestellungen von Decision-Support-Systemen. Die Anfragen werden zur Leistungsmessung solcher Systeme verwendet, d. h., es kommen Anfragen mit unterschiedlichen Eigenschaften zum Einsatz. Die Verwendung einer heterogenen Menge von Anfragen ist dadurch gewährleistet. Insgesamt wurden Messungen mit 22 Anfragen aus dem TPC-H durchgeführt:  $Q_t01$ – $Q_t22$ . Die Anfragen finden sich auf den Internetseiten des TPC [TPCH04].

## 5.1 Beschreibung der Messungen

---

Anfrage	<i>T</i>	<i>P</i>	<i>S</i>	<i>G</i>
Q <sub>t</sub> 01	1	2	0	Ja
Q <sub>t</sub> 02	9	13	1	Nein
Q <sub>t</sub> 03	3	5	0	Ja
Q <sub>t</sub> 04	2	5	1	Ja
Q <sub>t</sub> 05	6	9	0	Ja
Q <sub>t</sub> 06	1	5	0	Nein
Q <sub>t</sub> 07	6	11	1	Ja
Q <sub>t</sub> 08	8	12	1	Ja
Q <sub>t</sub> 09	6	7	1	Ja
Q <sub>t</sub> 10	4	6	0	Ja
Q <sub>t</sub> 11	6	7	1	Ja
Q <sub>t</sub> 12	2	11	0	Ja
Q <sub>t</sub> 13	2	2	1	Ja
Q <sub>t</sub> 14	2	3	0	Nein
Q <sub>t</sub> 15	3	2	1	Nein
Q <sub>t</sub> 16	3	13	1	Ja
Q <sub>t</sub> 17 <sup>2</sup>	3	5	1	Nein
Q <sub>t</sub> 18	4	3	1	Ja
Q <sub>t</sub> 19	2	39	0	Nein
Q <sub>t</sub> 20 <sup>2</sup>	5	10	2	Nein
Q <sub>t</sub> 21	6	13	1	Ja
Q <sub>t</sub> 22	3	18	2	Ja
Q <sub>o</sub> a1	7	19	1	Ja
Q <sub>o</sub> b1	7	8	0	Ja
Q <sub>o</sub> c1 <sup>2</sup>	23	45	3	Ja
Q <sub>o</sub> c2	9	19	2	Ja
Q <sub>o</sub> c3 <sup>3</sup>	11	36	3	Ja

Anfrage	<i>T</i>	<i>P</i>	<i>S</i>	<i>G</i>
Q <sub>e</sub> 01	10	10	0	Nein
Q <sub>e</sub> 02	3	3	0	Ja
Q <sub>e</sub> 03	2	2	0	Nein
Q <sub>e</sub> 04	2	2	1	Ja
Q <sub>e</sub> 05	10	11	0	Nein
Q <sub>e</sub> 06	1	1	0	Nein
Q <sub>e</sub> 07	4	5	0	Nein
Q <sub>e</sub> 08	1	0	0	Ja
Q <sub>e</sub> 09	4	4	0	Ja
Q <sub>e</sub> 10	2	4	1	Nein
Q <sub>e</sub> 11	3	5	0	Nein
Q <sub>e</sub> 12	3	5	0	Nein
Q <sub>e</sub> 13	1	1	0	Nein
Q <sub>e</sub> 14	1	1	0	Nein
Q <sub>e</sub> 15	2	1	1	Nein
Q <sub>e</sub> 16	1	3	0	Ja
Q <sub>e</sub> 17	2	5	1	Nein
Q <sub>e</sub> 18	1	1	0	Ja
Q <sub>e</sub> 19	3	3	0	Ja
Q <sub>e</sub> 20	2	2	1	Nein
Q <sub>e</sub> 21	1	1	0	Nein
Q <sub>e</sub> 22	1	1	0	Ja
Q <sub>e</sub> 23	2	2	0	Nein
Q <sub>e</sub> 24	2	2	0	Nein
Q <sub>e</sub> 25	2	2	0	Nein
Q <sub>e</sub> 26 <sup>2</sup>	2	1	1	Nein

*T* = Anzahl der Tabellen; *P* = Anzahl der Prädikate; *S* = Schachtelungstiefe; *G* = Gruppierung  
<sup>2</sup> = Datenbasis TPCH2; <sup>3</sup> = Datenbasis TPCH3

Tabelle 5.1: *Eigenschaften der Anfragen*

### Eigene Anfragen

Der zweite große Anfrageblock besteht aus Anfragen, die speziell für diese Arbeit zusammengestellt wurden. Sie wurden mit dem Wissen über die potenziellen Einflussfaktoren des theoretischen Teils ausgewählt und dienen insbesondere dazu, die Erkenntnisse aus dem theoretischen Teil der Arbeit zu überprüfen. Für die eigenen Anfragen wird folgende Bezeichnung verwendet: Q<sub>e</sub>01–Q<sub>e</sub>26. Die Anfragen sind im Anhang A aufgelistet.

### Anfragen aus dem Projekt ORBIT

Bei der dritten Gruppe von Anfragen handelt es sich um OLAP-Anfragen aus dem ORBIT-Projekt, die bereits in Vorarbeiten für Messungen verwendet wurden. Das Projekt beschäftigt sich u. a. mit der Optimierung von Abfragesequenzen in OLAP-Anwendungen [KSRM03]. Solche Abfragesequenzen lassen sich in einer einzigen Anfrage formulieren. Die einzelnen Anfragen weisen dann eine hohe Komplexität auf. Sie referenzieren viele Tabellen, enthalten viele Prädikate und besitzen im Vergleich zu den anderen Anfragegruppen eine relativ hohe Schachtelungstiefe (s. Tabelle 5.1). Es wurden sowohl Messungen mit den einzelnen Anfragen als auch mit den zugehörigen Abfragesequenzen durchgeführt. Folgende Anfragen wurden untersucht (s. Anhang A): Q<sub>o</sub>a1, Q<sub>o</sub>a1\_seq, Q<sub>o</sub>b1, Q<sub>o</sub>c1, Q<sub>o</sub>c1\_seq, Q<sub>o</sub>c2, Q<sub>o</sub>c2\_seq, Q<sub>o</sub>c3, Q<sub>o</sub>c3\_seq.

### 5.1.3 Ablauf der Messungen

Dieser Abschnitt beschreibt den Ablauf der Messungen. Dazu gehören die Durchführung mit den DB2-Werkzeugen sowie die anschließende Auswertung der Resultate.

#### Durchführung

Die eigentliche Durchführung geschieht mit den oben beschriebenen DB2-Programmen `db2batch` und `db2exfmt`. Es existiert für jede SQL-Anfrage eine Datei, die als Input für die DB2-Programme dient. Für jede Anfrage werden die beiden Programme wie folgt aufgerufen (für die Erklärung der Parameter s. Abschnitt 5.1.1):

```
db2batch -d tpch -f sql_input -r query_output -o e 2  
db2exfmt -d tpch -w -1 -n % -s % -# 0 -g -o explain_output
```

Die zwei Schritte der Messung sind in Abbildung 5.2 dargestellt. Der Aufruf von `db2batch` sorgt für die Ausführung einer Anfrage und schreibt sowohl das Resultat als auch die gemessene Ausführungszeit in eine Datei. Parallel dazu werden die Explain-Tabellen mit den geschätzten Werten für die Anfrage geschrieben. Der direkt anschließende Aufruf von `db2exfmt` extrahiert die Explain-Informationen und schreibt sie in eine zweite Datei.

Auf diese Weise können die Messungen für alle einzelnen Anfragen durchgeführt werden. Lediglich für die Abfragesequenzen ist eine andere Vorgehensweise erforderlich. Während `db2batch` mehrere Anfragen mit einem Aufruf bearbeiten kann, liest der obige Aufruf von `db2exfmt` immer die aktuellsten Explain-Informationen aus. Bei einer Abfragesequenz werden jedoch für jede einzelne Teilanfrage Explain-Informationen benötigt. Sie werden deshalb mit separaten Explain-Anfragen gesammelt. Das ermöglicht eine Kennzeichnung und eine anschließende Zuordnung zur ausgeführten Teilanfrage.

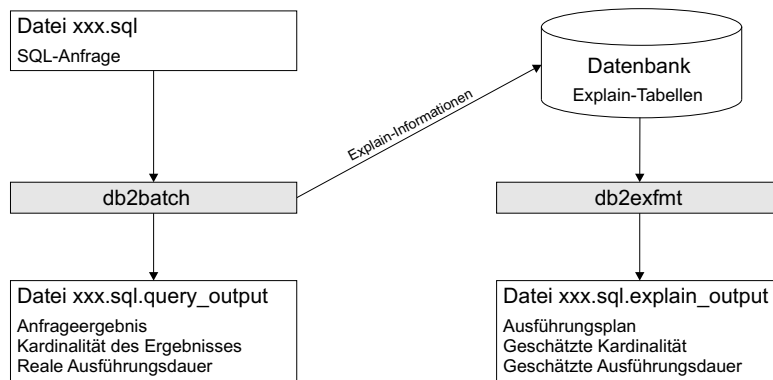


Abbildung 5.2: Durchführung der Messungen

### Auswertung

Nach der Durchführung der Messungen existieren zu jeder Anfrage zwei Dateien mit den Werten der geschätzten und der realen Ausführungsdauer sowie den Kardinalitäten der Zwischen- und Endergebnisse (s. Abbildung 5.2). Die Ausführungszeit spielt dabei nur eine untergeordnete Rolle. Der Fokus der Messungen liegt auf den Kardinalitäten. Das hat mehrere Gründe. Erstens ist die Ausführungsdauer einer Anfrage abhängig von den Kardinalitäten. Je mehr Tupel verarbeitet werden müssen, desto mehr Zeit wird benötigt. Der Hauptgrund ist, dass sich die Werte der geschätzten und der tatsächlichen Ausführungszeit nicht sinnvoll vergleichen lassen. Die von db2batch ermittelten, realen Ausführungszeiten werden in *Sekunden* angegeben. Der Optimierer benutzt für seine Planauswahl jedoch die Einheit *Timeron* für die Schätzung der Ausführungszeit. In den Explain-Informationen sind keine Sekundenangaben verfügbar. Eine gegenseitige Abbildung dieser Einheiten ist nicht möglich. Dies führt dazu, dass die Ausführungszeit für den Vergleich der geschätzten und realen Werte ausscheidet.

Auch bei den Kardinalitäten gibt es Vergleichsschwierigkeiten, die sich allerdings als lösbar herausstellen. In den Explain-Tabellen werden für jeden Zwischenknoten die geschätzten Kardinalitäten abgelegt. Sie lassen sich mit dem Programm db2exfmt extrahieren. Die Ausführung der Anfrage mit db2batch liefert allerdings nur Informationen über die Kardinalität des Endergebnisses. Um an einem Zwischenknoten im Ausführungsplan einen Vergleich der Kardinalitäten anstellen zu können, muss die reale Kardinalität bestimmt werden. Dafür werden Ad-hoc-Anfragen spezifiziert, deren Plan mit dem zu untersuchenden Teilplan übereinstimmt. Die Kardinalität des Endergebnisses der Ad-hoc-Anfrage entspricht dann der realen Kardinalität für den zu untersuchenden Zwischenknoten. Die Formulierung von Teilanfragen für Zwischenknoten gelingt nicht immer, da besonders bei komplexen Anfragen der Ausführungsplan vielen Optimierungsschritten unterliegt. Die Optimierung einer Teilanfrage führt dann nicht zwangsläufig zu einem kongruenten Teilplan.

Die Vorgehensweise bei der Auswertung ist wie folgt: Zuerst wird die Kardinalität des Endergebnisses mit der geschätzten Kardinalität verglichen. Weichen die Werte ab, so wird in einem zweiten Schritt versucht, die Stellen im Ausführungsplan zu finden, welche für die fehlerhafte Schätzung verantwortlich sind. Dafür werden die Kardinalitäten der Zwischenknoten wie oben beschrieben verglichen.

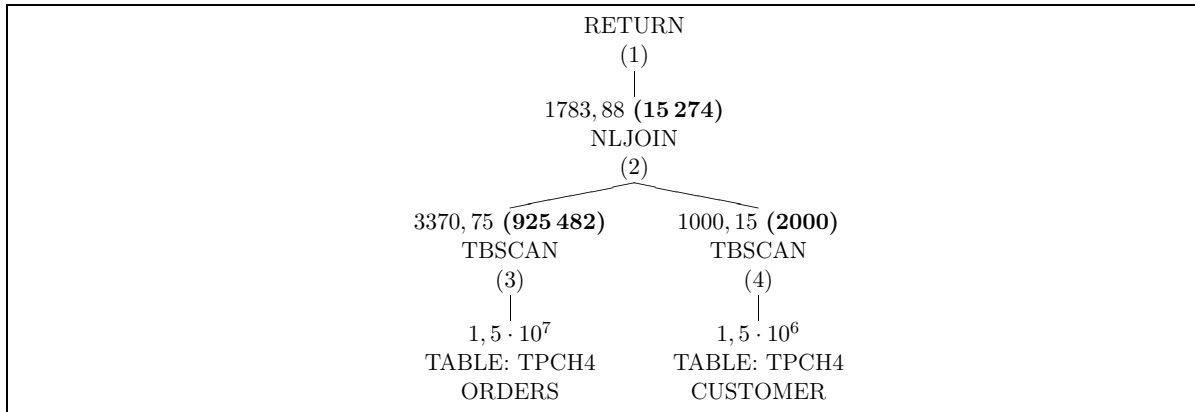


Abbildung 5.3: *Beispiel einer Planauswertung*

Abbildung 5.3 zeigt einen ausgewerteten Ausführungsplan. Das Format entspricht im Wesentlichen der Ausgabe von db2exfmt. Die Knoten sind nummeriert. Oberhalb jedes Operators steht die vom Optimierer geschätzte Kardinalität. In Klammern sind die realen Kardinalitäten hinzugefügt, welche über die Ad-hoc-Anfragen ermittelt werden. Das Programm db2exfmt liefert zu den einzelnen Knoten noch detailliertere Informationen wie z. B. kumulierte Kosten und Schätzwerte für Selektivitätsprädikate. Um zu einer kompakten Darstellung zu gelangen, wird bei den nachfolgenden Auswertungen der Messergebnisse lediglich die Übersichtsinformation in Form des Ausführungsplans als Dokumentation aufgenommen. Die detaillierteren Informationen zu den Knoten werden nur so weit in die Erläuterungen mit einbezogen, wie es für das Verständnis notwendig ist. Ebenso werden die Ad-hoc-Anfragen zur Ermittlung der realen Kardinalitäten nicht aufgeführt.

## 5.2 Messergebnisse

Die Darstellung der Messergebnisse steht im Mittelpunkt dieses Abschnitts. Das geschieht auf zweierlei Ebenen. Als Erstes wird ein Überblick über die Resultate gegeben. Dabei werden die Messergebnisse nach den aufgetretenen Problemen eingeteilt. Im Anschluss daran werden Vertreter dieser Problemgruppen detaillierter dargestellt. Dazu gehört eine Analyse der Ausführungspläne, bei der die Ursachen für aufgetretene Schätzfehler erläutert werden. Die Schätzwerte des Optimierers im Ausführungsplan sind mit unterschiedlicher Genauigkeit gegeben. Die Anzahl der Nachkommastellen variiert zwischen den Werten 5 und 9. Bei den Plananalysen ab Abschnitt 5.2.2 wird durchgängig mit fünf Nachkommastellen gerechnet und auf die Verwendung des Zeichens „≈“ verzichtet.

### 5.2.1 Überblick

Insgesamt wurden über 50 einzelne Anfragen untersucht ( $Q_t01$ – $Q_t22$ ,  $Q_e01$ – $Q_e26$ ,  $Q_oa1$ ,  $Q_ob1$ ,  $Q_oc1$ ,  $Q_oc2$ ,  $Q_oc3$ ). Dazu kommen für die letzte Gruppe der Anfragen aus dem Projekt ORBIT ( $Q_o$ ) noch die zugehörigen Anfragesequenzen.

In der Tabelle 5.2 sind die Messergebnisse aller Anfragen mit der geschätzten Kardinalität  $Card_g(E)$  und der realen Kardinalität des Anfrageergebnisses  $Card_r(E)$  aufgelistet. Als Maß für die Abweichung wird folgender Fehlerfaktor  $F$  definiert:

$$F := \begin{cases} 1 & \text{für } Card_g(E) = Card_r(E) \\ \frac{Card_g(E)}{Card_r(E)} & \text{für } Card_g(E) > Card_r(E) \wedge Card_r(E) \neq 0 \\ -\frac{Card_r(E)}{Card_g(E)} & \text{für } Card_g(E) < Card_r(E) \wedge Card_g(E) \neq 0 \\ \infty & \text{für } Card_g(E) > 0 \wedge Card_r(E) = 0 \\ -\infty & \text{für } Card_r(E) > 0 \wedge Card_g(E) = 0 \end{cases}$$

Der Fehlerfaktor  $F$  bewegt sich in den Intervallen  $]-\infty; -1[$  und  $[1; \infty[$ . Die Werte in der Tabelle sind auf eine Nachkommastelle gerundet. Ein Wert  $F \geq 1,0$  bedeutet, dass die Kardinalität des Ergebnisses zu hoch geschätzt wurde. Ein Faktor  $F \leq -1,0$  signalisiert eine zu geringe Schätzung, während  $F \equiv 1$  für die exakte Schätzung der Kardinalität steht. Zusätzlich zu den Größen  $Card_g(E)$ ,  $Card_r(E)$  und  $F$  ist die tatsächliche Ausführungsdauer der Anfrage  $T_r$  angegeben. Die Tabelleneinträge sind nach den Anfragegruppen  $Q_x$  und dem absoluten Fehlerfaktor  $|F|$  sortiert. Für alle Anfragen mit einem Wert  $F \geq 1,5$  wurde eine Fehleranalyse durchgeführt. In der letzten Spalte stehen die Ursachen für die Schätzfehler. Hier ist zu beachten, dass nicht immer ein einziger Faktor für die Abweichungen verantwortlich gemacht werden kann. Es wurden gleiche und ähnliche Ursachen gruppiert und unter einem Oberbegriff zusammengefasst. Die Tabelle enthält die Abkürzungen der identifizierten Fehlergruppen:

- **Datenabhängigkeit (Dat)**: Abhängigkeiten der Daten führen zu Schätzfehlern.
- **Standardwert (Sta)**: Auf Grund fehlender Informationen muss bei der Abschätzung auf Standardwerte zurückgegriffen werden.
- **Gruppierung (Gru)**: Die Kardinalität des Gruppierungsoperators wird falsch geschätzt.
- **LIKE-Prädikat (LP)**: Der Selektivitätsfaktor des LIKE-Prädikats wird falsch geschätzt.
- **Anti-Verbund (AV)**: Ein Anti-Verbund ist Ursache des Schätzfehlers (äußerer Verbund mit Vergleich von Null-Werten, NOT-EXISTS-Prädikat).
- **Semantik (Sem)**: Die Anfrage enthält Semantik, die im DBS nicht abgebildet ist.
- **String-Operationen (SO)**: Operationen auf Strings führen zu Schätzfehlern.
- **Temporäre Tabellen (Tem)**: Der Optimierer speichert Zwischenergebnisse in temporären Tabellen.
- **Vergleichsprädikat mit Wert einer Aggregatfunktion (VmWeA)**: Ein Prädikat, das den Wert einer Aggregatfunktion enthält, kann nicht korrekt abgeschätzt werden.

Die Analyse der Messergebnisse ergab, dass sich die Ergebnisse der TPC-H-Anfragen ( $Q_t$ ) von denen der selbst spezifizierten Anfragen ( $Q_e$ ) nicht wesentlich unterscheiden. Beide Anfragegruppen lieferten ein breites Ergebnisspektrum. Sowohl Pläne mit exakten Schätzergebnissen ( $F \equiv 1$ ) als auch Pläne mit großen Abweichungen wurden identifiziert. Ebenso variierten die

Kardinalitäten des Anfrageergebnisses sowie die Ausführungszeiten in beiden Gruppen. Auch die Ursachen für fehlerhafte Schätzungen waren ähnlich. Das ist u. a. darauf zurückzuführen, dass in beiden Gruppen Anfragen unterschiedlicher Komplexität vertreten sind. Die Grafik in Abbildung 5.4 zeigt die gemessenen Schätzfehler und verdeutlicht die ungleichmäßige Verteilung des Fehlers über die verschiedenen Anfragen. Eine Aussage über Gemeinsamkeiten mit der Gruppe der Abteilungsanfragen ( $Q_o$ ) ist nicht möglich, da diese Anfragegruppe mit insgesamt fünf Anfragen dafür zu klein ist. Der Zusammenhang der Messergebnisse der einzelnen Anfragen mit den Ergebnissen der entsprechenden Anfragesequenzen wird in einem gesonderten Abschnitt beschrieben (s. 5.2.9).

In den nachfolgenden Abschnitten werden die einzelnen Fehlergruppen genauer analysiert. Dabei werden Vertreter aus den Gruppen herausgegriffen und es wird anhand des Ausführungsplans erläutert, wie die Schätzfehler zu Stande kommen. Die Problemgruppen LP und SO sind unter dem Abschnitt Stringvergleich zusammengefasst, die Gruppen Gru und VmWeA unter dem Abschnitt Aggregation. Ansonsten entsprechen die Namen der nachfolgenden Abschnitte denen der obigen Fehlergruppen.

5.2 Messergebnisse

Anfrage	$Card_q(E)$	$Card_r(E)$	$F$	$T_r$	Ursache
Q <sub>t</sub> 04	5,0	5	$\equiv 1$	19 m 29 s	–
Q <sub>t</sub> 12	2,0	2	$\equiv 1$	9 m 43 s	–
Q <sub>t</sub> 18	636,1	624	1,0	8 m 47 s	–
Q <sub>t</sub> 16	26 248,3	27 840	–1,1	35 s	–
Q <sub>t</sub> 06	1 233 360,0	1 139 264	1,1	8 m 14 s	–
Q <sub>t</sub> 14	866 774,0	749 223	1,2	10 m 43 s	–
Q <sub>t</sub> 09	225,0	175	1,3	14 m 50 s	–
Q <sub>t</sub> 10	551 591,0	381 105	1,4	18 m 36 s	–
Q <sub>t</sub> 01	6,0	4	1,5	7 m 36 s	–
Q <sub>t</sub> 17	1 448,1	587	2,5	5 m 49 s	VmWeA
Q <sub>t</sub> 08	9,0	2	4,5	11 m 21 s	Gru
Q <sub>t</sub> 05	25,0	5	5,0	20 m 26 s	Gru
Q <sub>t</sub> 19	13,2	171	–12,9	8 m 30 s	Dat
Q <sub>t</sub> 03	3 024 830,0	114 003	26,5	10 m 42 s	Dat
Q <sub>t</sub> 13	5 395,7	46	117,3	37 s	Gru
Q <sub>t</sub> 07	5 625,0	4	1 406,3	10 m 32 s	Gru
Q <sub>t</sub> 15	4 000,0	1	4 000,0	8 m 31 s	VmWeA, Sta
Q <sub>t</sub> 02	0,1	4 667	–60 091,6	1 m 39 s	VmWeA, Dat
Q <sub>t</sub> 20	0,0	204	–66 064,3	16 m 51 s	VmWeA, LP
Q <sub>t</sub> 22	0,0	7	–482 376,0	2 m 55 s	AV, SO
Q <sub>t</sub> 21	0,0	4 009	–8,0 · 10 <sup>12</sup>	31 m 45 s	AV, Gru
Q <sub>t</sub> 11	106 667,0	0	$\infty$	1 m 30 s	VmWeA, Sta

Q <sub>e</sub> 16	154,0	154	$\equiv 1$	7 m 54 s	–
Q <sub>e</sub> 24	1 000,0	1 000	$\equiv 1$	16 m 00 s	–
Q <sub>e</sub> 08	50,0	50	$\equiv 1$	8 m 23 s	–
Q <sub>e</sub> 01	59 986 100,0	59 986 052	1,0	24 m 42 s	–
Q <sub>e</sub> 05	2 399 440,0	2 398 579	1,0	10 m 15 s	–
Q <sub>e</sub> 20	1 500 000,0	1 481 218	1,0	153 m 41 s	–
Q <sub>e</sub> 07	1 436 360,0	1 443 201	–1,0	1 m 15 s	–
Q <sub>e</sub> 09	23 298,8	25 083	–1,1	1 m 20 s	–
Q <sub>e</sub> 15	19 995 400,0	22 796 456	–1,1	86 m 25 s	–
Q <sub>e</sub> 14	16 256 200,0	15 000 562	1,1	27 m 01 s	–
Q <sub>e</sub> 10	210 917,0	282 538	–1,3	17 s	–
Q <sub>e</sub> 04	7 500 000,0	10 319 855	–1,4	17 m 04 s	–
Q <sub>e</sub> 13	1 935 030,0	1 145 588	1,7	8 m 26 s	Sem
Q <sub>e</sub> 03	29 982 100,0	59 986 052	–2,0	23 m 50 s	Dat
Q <sub>e</sub> 18	5 000 000,0	2 140 173	2,3	2 m 55 s	VmWeA
Q <sub>e</sub> 02	8,3	25	–3,0	1 s	VmWeA, Sta
Q <sub>e</sub> 06	1 415 580,0	75 190	18,8	2 s	LP
Q <sub>e</sub> 12	1 783,9	433 274	–242,9	112 m 18 s	LP

## 5.2 Messergebnisse

Q <sub>e</sub> 17	30 025 900,0	49 831	602,6	16 m 21 s	Dat
Q <sub>e</sub> 22	1,9	81 624,05	-42 427,9	19 s	Dat
Q <sub>e</sub> 25	5,3	38 489 225	-7 294 737,8	158 m 13 s	AV
Q <sub>e</sub> 11	0,0	0	∞	16 m 25 s	LP
Q <sub>e</sub> 19	2 666 670,0	0	∞	1 m 16 s	VmWeA, Sta
Q <sub>e</sub> 21	19 995 400,0	0	∞	8 m 08 s	Sem
Q <sub>e</sub> 23	19 995 400,0	0	∞	10 m 55 s	Dat, Sta
Q <sub>e</sub> 26	399 960,0	0	∞	6 s	VmWeA

Q <sub>o</sub> b1	10,3	9	1,1	6 m 24 s	-
Q <sub>o</sub> c2	13 583,5	29 764	-2,2	36 m 12 s	VmWeA, Sta
Q <sub>o</sub> a1	166 031,0	27	6 149,3	61 m 42 s	VmWeA, Sta
Q <sub>o</sub> c3	$6,4 \cdot 10^{11}$	6 436	98 703 697,9	18 m 44 s	Tem, Sta
Q <sub>o</sub> c1	$3,8 \cdot 10^{14}$	2 933	$1,3 \cdot 10^{11}$	5 m 36 s	Tem, Sta

Tabelle 5.2: Übersicht der Messergebnisse

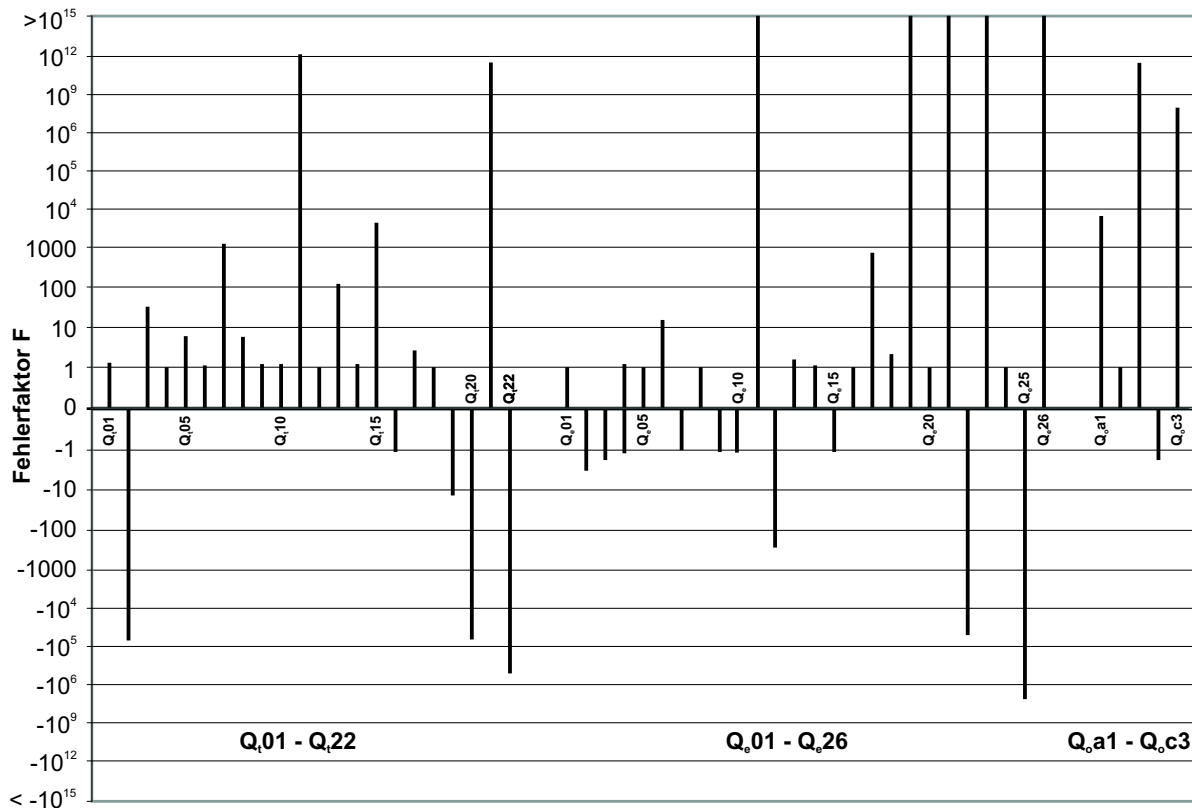


Abbildung 5.4: Schätzfehler des Optimierers

### 5.2.2 Aggregation

Die größte Fehlergruppe (insgesamt 16 Anfragen) sind die Anfragen, welche eine Aggregation der Daten enthalten. Es werden zwei Ursachentypen unterschieden. Auf der einen Seite die reine Gruppierung der Daten. Für fünf Anfragen ist es dem Optimierer nicht möglich, die ausgehende Kardinalität eines Gruppierungsoperators korrekt zu schätzen. Auf der anderen Seite stellen Aggregatfunktionen eine Ursache für Schätzfehler dar. Bei elf Anfragen werden Vergleichsprädikate ungenau abgeschätzt, was auf die fehlerhafte Abschätzung des Werts einer Aggregatfunktion zurückzuführen ist. Diese beiden Probleme werden nachfolgend genauer erläutert.

```
(Qt05) SELECT      n_name, SUM(l_extendedprice * (1 - l_discount)) AS revenue
FROM              tpch4.customer, tpch4.orders, tpch4.lineitem,
                   tpch4.supplier, tpch4.nation, tpch4.region
WHERE             c_custkey = o_custkey AND l_orderkey = o_orderkey AND
                   l_suppkey = s_suppkey AND c_nationkey = s_nationkey AND
                   s_nationkey = n_nationkey AND
                   n_regionkey = r_regionkey AND r_name = 'ASIA' AND
                   o_orderdate >= date('1994-01-01') AND
                   o_orderdate < date('1994-01-01') + 1 YEAR
GROUP BY         n_name
ORDER BY         revenue DESC
```

Die Anfrage  $Q_{t05}$  berechnet zu jedem Lieferanten aus der Region Asien den erzielten Jahreserlös. Ein Teil des zugehörigen Ausführungsplans ist in Abbildung 5.5 zu sehen. In Knoten 8 stimmt die geschätzte Kardinalität (76 767,7) noch mit der realen Kardinalität (72 985) überein. Die anschließende Gruppierung nach den Nationen wird bereits im Rahmen der Sortierung in Knoten 7 durchgeführt. Der Optimierer schätzt für den ausgehenden Objektstrom 25 Tupel, was der Kardinalität der Tabelle *nation* entspricht. Die vorausgegangene Selektion mit dem Prädikat  $r\_name = 'ASIA'$  bleibt unberücksichtigt. Tatsächlich existieren aber nur fünf Nationen aus der Region Asien. Daraus resultiert der Fehlerfaktor von  $F = 5$ .

Die Anfrage  $Q_{e26}$  zeigt ein Beispiel für Probleme bei Abschätzungen von Vergleichsprädikaten mit Werten von Aggregatfunktionen (MIN, MAX, SUM, COUNT, AVG). Die Ergebnismenge der Anfrage ist leer, da die verfügbare Anzahl eines Teils (*ps\_availqty*) niemals die Summe aller jemals bestellten Teile (15 334 802) übersteigt.

```
(Qe26) SELECT      ps_suppkey
FROM              tpch2.partsupp
WHERE             ps_availqty > (
                   SELECT SUM(l_quantity)
                   FROM tpch2.lineitem)
```

Bei der Schätzung des Prädikats  $ps\_availqty > SUM(l\_quantity)$  wird nicht berücksichtigt, dass es sich bei dem Vergleichswert um den Rückgabewert einer Aggregatfunktion handelt. Die Selektivität wird auf 0,49995 geschätzt, als handle es sich um einen normalen Attributwert. Die Aggregatfunktionen MIN, MAX und COUNT führen zum gleichen Schätzwert wie die

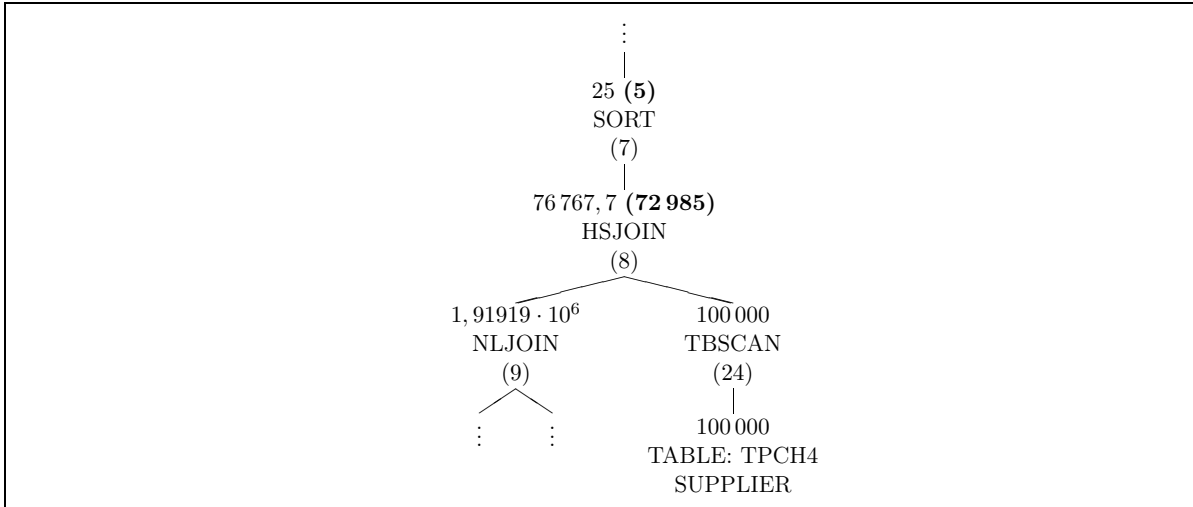


Abbildung 5.5: Ausführungsplan zu Anfrage  $Q_{t05}$

Funktion SUM. Dieses Verhalten führt zwangsläufig in mindestens einem der Fälle zu Schätzfehlern, da sich die Werte der Funktionen SUM und MIN je nach Größe der Datenmenge stark unterscheiden.

$$s(ps\_availqty > AGG(l\_quantity)) = 0,499\,95 \quad AGG \in \{SUM, MIN, MAX, COUNT\}$$

Die Messungen zeigten, dass die Verwendung von Rückgabewerten von Aggregatfunktionen in Vergleichsprädikaten häufig zu fehlerhaften Schätzungen führt. Im Verhältnis zu anderen Ursachen wurden dabei relativ große Fehlerfaktoren gemessen.

### 5.2.3 Datenabhängigkeiten

Ein Problem, das bei sieben Anfragen identifiziert wurde, sind Abhängigkeiten von Attributwerten. Dabei lassen sich drei Gruppen unterscheiden: Abhängigkeiten von Werten desselben Attributs ( $Q_{t02}$ ,  $Q_{e22}$ ), Abhängigkeiten zwischen Attributwerten einer Relation ( $Q_{e17}$ ) und Abhängigkeiten zwischen Attributwerten verschiedener Relationen ( $Q_{t03}$ ,  $Q_{t19}$ ,  $Q_{e03}$ ,  $Q_{e17}$ ,  $Q_{e23}$ ).

Im Datenbankschema existieren verschiedene Attribute des Typs Datum:

- $o\_orderdate$ : Bestelldatum
- $l\_receiptdate$ : tatsächliches Eingangsdatum beim Empfänger
- $l\_commitdate$ : vereinbartes Lieferdatum
- $l\_shipdate$ : Auslieferdatum

Die Attributwerte sind voneinander abhängig. Beispielsweise liegt das tatsächliche Empfangsdatum einer Bestellung  $l\_receiptdate$  nie vor dem Bestelldatum  $o\_orderdate$ . Der Optimierer besitzt kein Wissen über diese Abhängigkeiten. Folgende Anfrage verdeutlicht das:

```
(Q6) SELECT  *
      FROM    tpch1.orders, tpch1.lineitem
      WHERE   o_orderkey = l_orderkey AND o_orderdate <= o_shipdate
```

Das Ergebnis der Anfrage Q6 enthält 600 572 Tupel. Das Prädikat  $o\_orderdate > o\_shipdate$  dagegen liefert 0 Tupel. Für beide Versionen schätzt der Optimierer eine Kardinalität von 300 398. Das führt zu dem Fehlerfaktor  $F = \infty$ . Bei den Messungen konnten für Anfragen, die mehrere dieser abhängigen Datumfelder enthalten, verschiedene Fehlerfaktoren gemessen werden.

### 5.2.4 Semantik innerhalb der Anfrage

Anfragen mit semantischen Informationen, die im Datenbankschema nicht abgebildet sind, stellen den Optimierer vor Schwierigkeiten. Zwei Beispiele dafür sind die Anfragen  $Q_{e13}$  und  $Q_{e21}$ , die solche Informationen enthalten.

```
(Qe13) SELECT  *
        FROM    tpch4.lineitem
        WHERE   DAY(l_commitdate) = 31
```

Bei  $Q_{e13}$  wird der Tag eines Datums auf den Wert 31 geprüft, bei  $Q_{e21}$  der Monat auf den Wert 13. Das DBS besitzt keine Integritätsbedingung für die Beschränkung des Wertebereichs von Teilen des Datums. Ein Tag kann nur die Werte 1 bis 31 annehmen, ein Monat die Werte 1 bis 12. Hinzu kommt, dass bei hinreichend vielen gleichverteilten Daten der Tag 31 seltener vorkommt als Tage mit einem Wert kleiner 31, da nur sieben Monate im Jahr 31 Tage haben. Auch dieses Wissen besitzt der Optimierer nicht, wie die Auswertung der Anfrage  $Q_{e13}$  zeigt:

$$s_g(\text{DAY}(l\_commitdate) = 31) = \frac{1}{31} = 0,032\,26$$

$$s_r(\text{DAY}(l\_commitdate) = 31) = \frac{\text{Card}(\text{Ergebnis } Q_{e13})}{\text{Card}(\text{lineitem})} = \frac{1\,145\,588}{59\,986\,052} = 0,019\,10$$

$$F = \frac{s_g}{s_r} = \frac{0,032\,26}{0,019\,10} = 1,7$$

Der Optimierer vertraut darauf, dass die Anfrage semantisch korrekt ist und der Vergleichswert des Prädikats zwischen 1 und 31 liegt. Alle Werte werden als gleich wahrscheinlich angenommen. Der Selektivitätsfaktor  $s_g(\text{DAY}(l\_commitdate) = t)$  hat für alle Tage  $t \in \mathbb{N}$  den Wert  $\frac{1}{31}$ . Dadurch entsteht der Schätzfehler  $F = 1,7$ .

Der Fehler vergrößert sich sprunghaft, wenn für den Vergleich ein Wert größer als 31 verwendet wird. An der Selektivitätsschätzung des Optimierers ändert sich dadurch nichts, jedoch fällt die Kardinalität des Ergebnisses auf null.

### 5.2.5 Standardwert für Schätzung

Wenn keine Statistiken verfügbar sind, so verwendet das DBS Standard-Filterfaktoren für die Schätzungen von Prädikaten. Die Tabelle 5.3 zeigt die Standardwerte von DB2 [IBM04]. Bei den gemessenen Anfragen sind der Standard-Filterfaktor 0,04 für das Prädikat  $\text{Spalte} = \text{Literal}$

und der Faktor  $0, \bar{3}$  für das Prädikat *Spalte Op Literal*,  $Op \in \{<, \leq, >, \geq\}$  aufgetreten. Die Messungen zeigten, dass die Verwendung von Standardwerten fast immer zu Schätzfehlern führt. Das ist darauf zurückzuführen, dass die Struktur der Daten an den zu schätzenden Stellen nicht den Annahmen des Optimierers entspricht (Datenelemente sind gleichverteilt, Datenelemente sind unabhängig voneinander).

Prädikat	Standard-Filterfaktor
Spalte = Literal	1/25
Spalte <i>Op</i> Literal $Op \in \{<, \leq, >, \geq\}$	1/3
Spalte IS NULL	1/25
Spalte IN (Literalliste)	(Anzahl der Literale)/25
Spalte LIKE Literal	1/10
Spalte BETWEEN Literal_1 AND Literal_2	1/10

Tabelle 5.3: *DB2-Filterfaktoren bei fehlenden Statistiken*

Diese Problemgruppe ist eng verknüpft mit anderen Fehlerursachen. In den folgenden Zusammenhängen wurden Schätzfehler bei der Verwendung von Standard-Filterfaktoren identifiziert:

- Die fehlerhafte Abschätzung von Prädikaten mit einem Standard-Filterfaktor ist auf Datenabhängigkeiten zurückzuführen (s. Abschnitt 5.2.3).
- Ein Prädikat mit dem Wert einer Aggregatfunktion wird mit einem Standardwert abgeschätzt (s. Abschnitt 5.2.2).
- Die Speicherung von Zwischenergebnissen in temporären Tabellen veranlasst den Optimierer, bei nachfolgenden Schätzungen auf Standard-Filterfaktoren zurückzugreifen (s. Abschnitt 5.2.7).

Unabhängig von dem Zusammenhang, in dem diese Standard-Filterfaktoren verwendet wurden, traten Abweichungen zwischen geschätzter und realer Kardinalität auf. Der Einsatz von Standardwerten ist ein Zeichen dafür, dass nicht ausreichend Informationen zur Verfügung stehen, um genauere Filterfaktoren anzuwenden. Ein ausführliches Beispiel für die Verwendung von Standardwerten im Zusammenhang mit temporären Tabellen ist in Abschnitt 5.2.7 gegeben.

### 5.2.6 Stringvergleich

Mehrere Anfragen enthalten Prädikate, die auf Strings operieren. Bei insgesamt zehn Messungen wird ein Stringvergleich mit dem LIKE-Prädikat unter Verwendung des Platzhalters „%“ durchgeführt. Eine Anfrage enthält einen Vergleich von Teilstrings mit dem Prädikat „=“. Die Kardinalitätsschätzungen für die dabei beteiligten Objektströme stellen sich als fehleranfällig heraus.

Als Beispiel für die Schätzfehler des Optimierers bei Stringvergleichen dient die Anfrage  $Q_e12$ . Sie enthält eine Selektion mit dem LIKE-Prädikat auf dem String-Attribut *o.o\_comment*. Es werden alle Einträge selektiert, bei denen der Attributwert auf den Buchstaben „a“ endet.

```
(Qe12) SELECT *
FROM tpch4.orders o, tpch4.lineitem l, tpch4.customer c
WHERE l.l_shipmode = 'SHIP' AND
o.o_comment LIKE '%a' AND
o.o_custkey = c.c_custkey AND
c.c_acctbal > 1000.1
```

Der zugehörige Ausführungsplan ist in Abbildung 5.6 dargestellt. Der Optimierer schätzt die Kardinalität des Anfrageergebnisses (Knoten 1) auf 1784 Tupel. Mit diesem Wert liegt er um den Faktor 242,9 unter dem tatsächlichen Ergebnis von 433 274 Tupel. Die Untersuchung der Zwischenknoten ergab, dass der fehlerhafte Wert in Knoten 7 entsteht und bis zur Wurzel propagiert wird. Die Selektivität des LIKE-Prädikats wird an dieser Stelle unterschätzt. Die reale Selektivität  $s_r$  sowie der Fehlerfaktor für den Knoten 7 ( $F_{k=7}$ ) ergeben sich wie folgt:

$$s_g(o\_comment \text{ LIKE } '%a') = 0,000\,23$$

$$s_r(o\_comment \text{ LIKE } '%a') = \frac{Card_r(Q_{e12_{k=7}})}{Card_r(orders)} = \frac{925\,482}{15\,000\,000} = 0,061\,70$$

$$F_{k=7} = -\frac{s_r}{s_g} = -\frac{0,061\,70}{0,000\,23} \approx -274,2$$

Dem Optimierer ist es an dieser Stelle nicht möglich, einen exakten Schätzwert zu bestimmen. Das ist darauf zurückzuführen, dass die im Datenbankkatalog gehaltenen Statistiken des Attributs `o_comment` für die Abschätzung unbrauchbar sind. In den Katalogtabellen werden unter anderem folgende Informationen für das Attribut gehalten:

- Anzahl unterschiedlicher Werte: 5 631 965
- Kardinalitäten der zehn häufigsten Attributwerte
- Equi-Depth-Histogramm mit Datenverteilung der Attributwerte.

In der Tabelle SYSTAT.COLDIST stehen Einträge der Form:

- Anzahl der Werte kleiner als „accounts according to the furios“ = 7500
- Anzahl der Werte kleiner als „blithely final platelets wake abo“ = 787 500

Diese Informationen ermöglichen keine Aussage über Attributwerte, die auf den Buchstaben „a“ enden, da die Histogrammwerte alphabetisch sortiert sind. Laut DB2-Dokumentation wird bei fehlenden Statistiken auf Interpolationsformeln und Standardwerte für die Schätzung zurückgegriffen. Es konnte jedoch nicht nachvollzogen werden, wie der Wert des Selektivitätsfaktors  $s_g(o\_comment \text{ LIKE } '%a')$  zu Stande kommt.

Eine Abschätzung der Form „Selektiere alle Tupel, die mit dem Buchstaben ‚a‘ beginnen“, kann dagegen genauer durchgeführt werden, da hierfür auf die obige Datenverteilung zurückgegriffen werden kann. Die Tabelle 5.4 zeigt die realen sowie die geschätzten Kardinalitäten für die Prädikate `o_comment LIKE '%x'` und `o_comment LIKE 'x%'`, wobei das „x“ stellvertretend für die Buchstaben „a“ bis „d“ steht.

## 5.2 Messergebnisse

Prädikat	$Card_g$	$Card_r$
LIKE '%a'	925 482	3 371
LIKE '%b'	203 307	3 371
LIKE '%c'	607 331	3 371
LIKE '%d'	322 930	3 371

Prädikat	$Card_g$	$Card_r$
LIKE 'a%'	25 515	280 856
LIKE 'b%'	938 760	1 770 915
LIKE 'c%'	1 112 585	1 682 382
LIKE 'd%'	189 117	386 951

Tabelle 5.4: Kardinalitätsschätzungen des LIKE-Prädikats

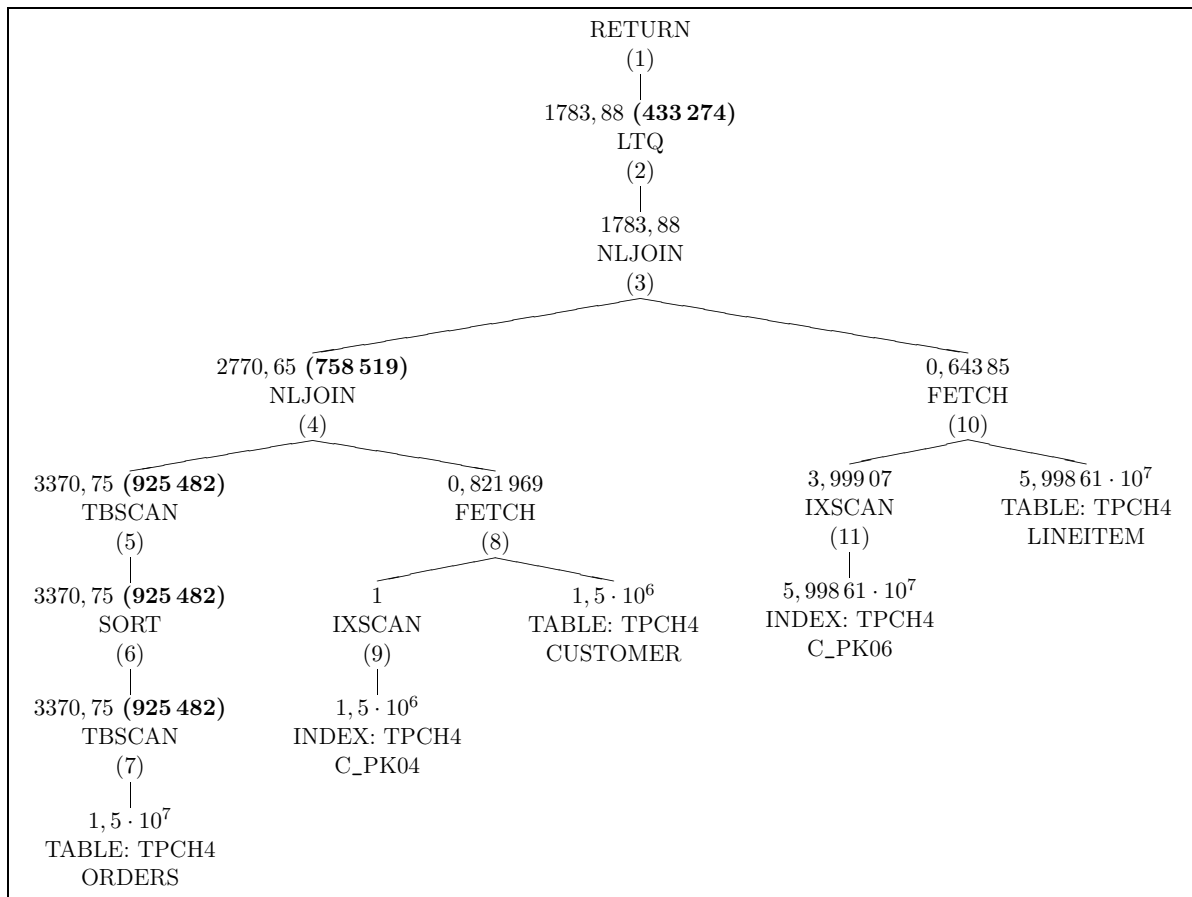


Abbildung 5.6: Ausführungsplan zu Anfrage  $Q_{e12}$

Die fehlerhaft geschätzte Kardinalität in Knoten 7 wird durch die Schätzungen der nachfolgenden Knoten propagiert. Dabei gibt es keine weiteren Abweichungen, welche die obige Größenordnung des Fehlerfaktors beeinflussen. Kleine Schätzfehler (z. B. in Knoten 10: 0,64385 statt 0,57143) verändern den Fehlerfaktor geringfügig, so dass sich für die gesamte Anfrage der Faktor  $F = -243$  ergibt.

### 5.2.7 Temporäre Tabellen

Bei den beiden Anfragen  $Q_{oc1}$  und  $Q_{oc3}$  wurden sehr große Fehlerfaktoren gemessen. Diese Anfragen aus dem Projekt ORBIT sind von hoher Komplexität, was sich auch in den zugehörigen Ausführungsplänen widerspiegelt: Die Pläne bestehen aus 77 bzw. 43 Knoten. Der Optimierer kann bei den Ausführungsplänen gemeinsame Teilbäume identifizieren, so dass deren Zwischenergebnis nur einmal berechnet und in temporären Tabellen abgelegt wird. Das führt jedoch dazu, dass nachfolgende Schätzungen auf Standard-Filterfaktoren zurückgreifen (s. Standardwerte in Abschnitt 5.2.5). Insbesondere bei Verbundoperationen entstanden dadurch große Schätzfehler.

Die Anfrage Q7 ist angelehnt an die Anfrage  $Q_{oc3}$  und verdeutlicht, wie es zu der Verwendung von Standard-Filterfaktoren kommt. Sie enthält zwei identische Teilanfragen, die über das Prädikat  $a1.custkey = a2.custkey$  miteinander verknüpft sind. Den Ausführungsplan zeigt Abbildung 5.7.

```
(Q7) SELECT a1.custkey, a1.turnover, a2.turnover
FROM (
    SELECT      a1.custkey, a2.orderyearkey, SUM(a1.endprice)
    FROM      tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
    WHERE     a2.orderdate = a1.orderdate AND
              a2.orderyearkey IN (1992, 1993)
    GROUP BY a1.custkey, a2.orderyearkey
) AS a1(custkey, year, turnover),
(
    SELECT      a1.custkey, a2.orderyearkey, SUM(a1.endprice)
    FROM      tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
    WHERE     a2.orderdate = a1.orderdate AND
              a2.orderyearkey IN (1992, 1993)
    GROUP BY a1.custkey, a2.orderyearkey
) AS a2(custkey, year, turnover)
WHERE a1.custkey = a2.custkey AND a1.year = 1992 AND a2.year = 1993
```

Das Zwischenergebnis wird in einer temporären Tabelle abgelegt (Knoten 4). Ab dieser Stelle verliert der Optimierer jegliches Wissen über die darin enthaltenen Daten und verwendet für die nachfolgenden Schätzungen ausschließlich Standard-Filterfaktoren. Die Selektivitätsfaktoren

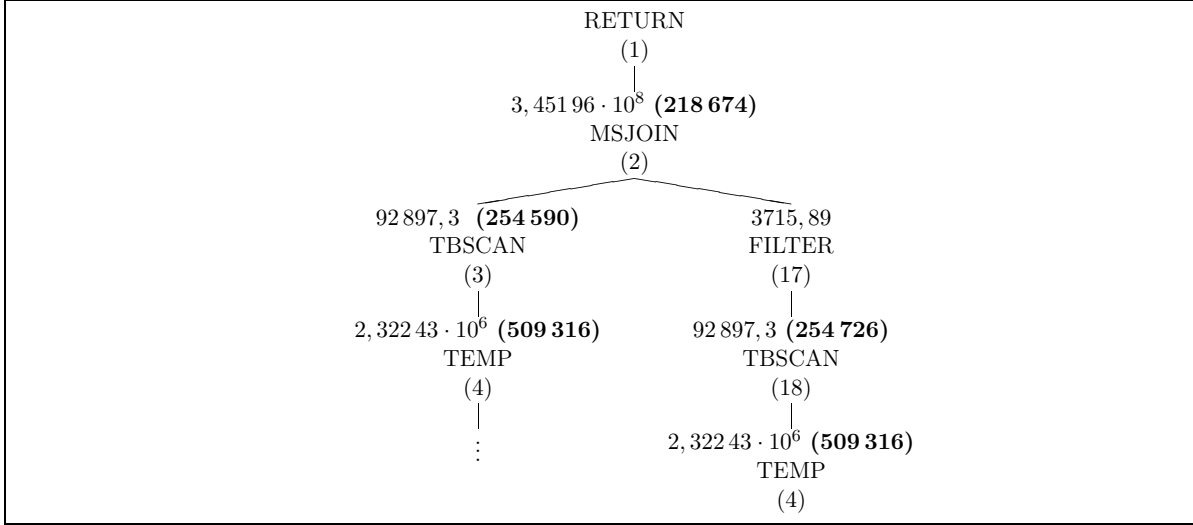


Abbildung 5.7: Ausführungsplan zu Anfrage Q7

der Knoten 3 und 18 werden deswegen fehlerhaft geschätzt:

$$\begin{aligned}
 s_g(a1.year = 1992) &= s_g(a2.year = 1993) = s_g(a1.custkey = a2.custkey) = \frac{1}{25} \\
 s_r(a1.year = 1992) &= \frac{Card_r(Q7_{k=3})}{Card_r(temp)} = \frac{254\,590}{509\,316} = 0,499\,87 \\
 s_r(a1.year = 1993) &= \frac{Card_r(Q7_{k=18})}{Card_r(temp)} = \frac{254\,726}{509\,316} = 0,500\,13 \\
 F_{k=3} &= -\frac{s_r(a1.year = 1992)}{s_g(a1.year = 1992)} = -12,5 \quad F_{k=18} = -\frac{s_r(a2.year = 1993)}{s_g(a2.year = 1993)} = -12,5
 \end{aligned}$$

Um zu ermitteln, welcher Fehlerfaktor beim Verbund in Knoten 2 entsteht, müssen die realen Kardinalitäten als Eingabe der Schätzungen zu Grunde gelegt werden.

$$\begin{aligned}
 Card_g(Q7_{k=2}) &= s_g(a1.custkey = a2.custkey) \cdot Card_r(Q7_{k=3}) \cdot Card_r(Q7_{k=18}) \\
 &= \frac{1}{25} \cdot 254\,590 \cdot 254\,726 \approx 2,594 \cdot 10^9 \\
 Card_r(Q7_{k=2}) &= 218\,674 \\
 F_{k=2} &= \frac{Card_g(Q7_{k=2})}{Card_r(Q7_{k=2})} = 11\,863,4
 \end{aligned}$$

Das Beispiel zeigt, wie es durch die Verwendung von Standard-Filterfaktoren zu großen Schätzfehlern kommt. Die Kardinalitäten der Selektionen  $a1.year = 1992$  und  $a2.year = 1993$  werden hier unterschätzt, während die Kardinalität des anschließenden Verbunds viel zu hoch geschätzt wird. Das führt für die gesamte Anfrage zu einem Fehlerfaktor von

$$F = \frac{Card_g(\text{Ergebnis } Q7)}{Card_r(\text{Ergebnis } Q7)} = \frac{3,45196 \cdot 10^8}{218\,674} = 1\,578,6$$

Dass dieser Fehler auf die Verwendung temporärer Tabellen zurückzuführen ist, zeigt die Anfrage Q8. Sie liefert die gleiche Ergebnismenge wie Q7, unterscheidet sich jedoch in ihrer Struktur und dem zugehörigen Ausführungsplan. Der Plan enthält keine temporären Tabellen und es werden keine Standard-Filterfaktoren verwendet. Die Kardinalität des Ergebnisses wird dadurch mit 331 776 wesentlich genauer geschätzt. Der Fehlerfaktor beträgt nur  $F = 1,5$ .

```
(Q8) SELECT a1.custkey, a1.turnover1992, a2.turnover1993
FROM (
    SELECT      a1.custkey, a2.orderyearkey, SUM(a1.endprice)
    FROM      tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
    WHERE     a2.orderdate = a1.orderdate AND
              a2.orderyearkey IN (1992)
    GROUP BY a1.custkey, a2.orderyearkey
) AS a1(custkey, year, turnover1992),
(
    SELECT      a1.custkey, a2.orderyearkey, SUM(a1.endprice)
    FROM      tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
    WHERE     a2.orderdate = a1.orderdate AND
              a2.orderyearkey IN (1993)
    GROUP BY a1.custkey, a2.orderyearkey
) AS a2(custkey, year, turnover1993)
WHERE a1.custkey = a2.custkey
```

### 5.2.8 Anti-Verbund

Ein Anti-Verbund ist ein Verbund mit inverser Logik. Es werden die Tupel der linken Seite des Verbundprädikats zurückgegeben, welche auf der rechten Seite keinen Verbundpartner finden [LG03]. Der Anti-Verbund lässt sich in SQL auf unterschiedliche Weise ausdrücken: durch das NOT-EXISTS-Prädikat, das NOT-IN-Prädikat, den OUTER-JOIN mit Vergleich auf Null-Werte sowie mit dem Mengenoperator EXCEPT.

Die Anfragen  $Q_t21$  und  $Q_t22$  enthalten das Prädikat NOT-EXISTS, die Anfrage  $Q_{e25}$  ein OUTER-JOIN-Konstrukt. In allen drei Fällen lagen die Kardinalitätsschätzungen für den Anti-Verbund um mehr als den Faktor  $F = 100\,000$  neben dem realen Kardinalitätswert.

```
(Qe25) SELECT *
FROM      tpch4.lineitem l LEFT JOIN tpch4.orders o
           on l.l_comment = o.o_comment
WHERE     o.o_comment IS NULL
```

Abbildung 5.8 zeigt den Ausführungsplan zu Anfrage  $Q_{e25}$ . Der Anti-Verbund wird durch den äußeren Verbund in Knoten 3 mit der anschließenden Filterung in Knoten 4 realisiert. Die Abweichung der Kardinalitäten in Knoten 4 ist auf das Verbundprädikat  $l.l\_comment = o.o\_comment$  zurückzuführen. Der wesentlich größere Fehler entsteht bei der anschließenden Filterung und dem Vergleich mit dem Null-Wert. Der Optimierer geht davon aus, dass es sich bei dem Vergleichswert NULL um einen eindeutigen Wert des Attributs  $l.l\_comment$  der

Tabelle *orders* handelt. Der vorausgegangene Verbund, der eine Vielzahl solcher Null-Werte erzeugt, bleibt dabei unberücksichtigt. Das hat folgenden Schätzfehler zur Folge:

$$s_g(o.o\_comment \text{ IS NULL}) = \frac{1}{Card_g(orders)} = \frac{1}{15\,000\,000} = 6,666\,67 \cdot 10^{-8}$$

$$s_r(o.o\_comment \text{ IS NULL}) = \frac{Card_r(Q_e25_{k=3})}{Card_r(Q_e25_{k=4})} = \frac{38\,489\,225}{1,858\,45 \cdot 10^9} = 0,020\,71$$

$$F_{k=3} = -\frac{s_r}{s_g} = -\frac{0,020\,71}{6,666\,67 \cdot 10^{-8}} = -310\,649,8$$

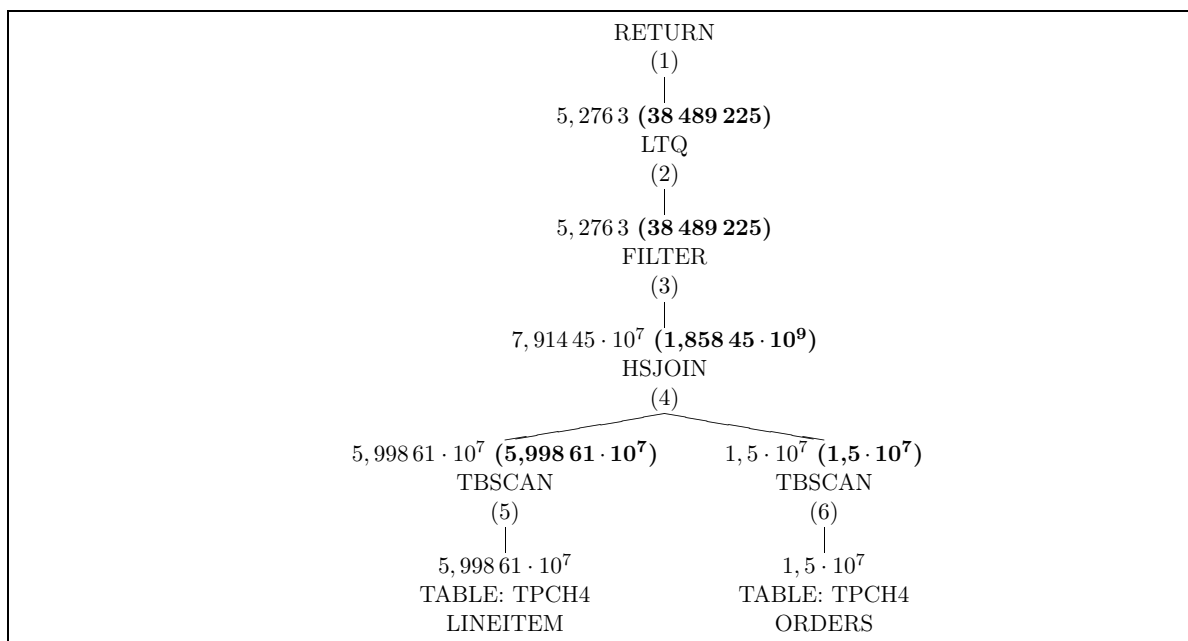


Abbildung 5.8: Ausführungsplan zu Anfrage  $Q_e25$

Auch die Schätzfehler der beiden Anfragen  $Q_t21$  und  $Q_t22$  mit dem NOT-EXISTS-Prädikat liegen in dieser Größenordnung. Im Vergleich zu den Schätzfehlern anderer Problemgruppen sind die Fehler für Abschätzungen des Anti-Verbunds relativ groß.

### 5.2.9 Überblick über die Anfragesequenzen

Die zugehörigen Anfragesequenzen zu den einzelnen Anfragen aus dem Projekt ORBIT ( $Q_o$ ) wurden ebenfalls untersucht. Die Messergebnisse stehen in Tabelle 5.5. Ein Vergleich der Ausführungszeiten zeigt, dass in drei von vier Fällen die Anfragesequenz schneller ausgeführt wurde als die einzelne Anfrage.

Bei den Kardinalitätsschätzungen der Sequenzen kann der Optimierer Informationen über zuvor berechnete Zwischenergebnisse verwenden. Es war jedoch nicht möglich, eine sinnvolle Auswertung der Messergebnisse vorzunehmen, da die Schätzwerte der Ausführungspläne widersprüchlich waren.

5.2 Messergebnisse

Anfrage	$Card_g(E)$	$Card_r(E)$	$F$	$T_r$
Q <sub>oa1_seq1</sub>	1 395 920, 0	1 229 098	1, 1	23 m 01 s
Q <sub>oa1_seq2</sub>	1 395 920, 0	1 229 473	1, 1	21 m 13 s
Q <sub>oa1_seq3</sub>	4, 7	378 463	80 964, 5	7 m 25 s
Q <sub>oa1_seq4</sub>	8, 7	27	3, 1	1 s
				51 m 40 s
Q <sub>oc1_seq1</sub> <sup>2</sup>	86 577	104 448	1, 2	21 s
Q <sub>oc1_seq2</sub> <sup>2</sup>	86 477	104 448	1, 2	10 s
Q <sub>oc1_seq3</sub> <sup>2</sup>	86 608	104 448	1, 2	10 s
Q <sub>oc1_seq4</sub> <sup>2</sup>	49, 2	3 207	65, 2	1 s
Q <sub>oc1_seq5</sub> <sup>2</sup>	25, 0	3 207	128, 3	2 m 16 s
Q <sub>oc1_seq6</sub> <sup>2</sup>	25, 0	3 207	128, 3	6 s
Q <sub>oc1_seq7</sub> <sup>2</sup>	141 024 000	3 207	43 973, 8	3 s
Q <sub>oc1_seq8</sub> <sup>2</sup>	12, 7	2 933, 0	231, 6	0 s
				3 m 7 s
Q <sub>oc2_seq1</sub>	1 100 260	868 074	1, 3	10 m 33 s
Q <sub>oc2_seq2</sub>	1 100 260	867 087	1, 3	8 m 46 s
Q <sub>oc2_seq3</sub>	1 100 260	866 336	1, 3	8 m 46 s
Q <sub>oc2_seq4</sub>	49, 2	32 492	660, 1	10 s
Q <sub>oc2_seq5</sub>	1 100 260	994 846	1, 1	9 m 30 s
Q <sub>oc2_seq6</sub>	207, 3	32 492	156, 7	33 s
Q <sub>oc2_seq7</sub>	12, 7	29 764	2 349, 8	2 s
				38 m 20 s
Q <sub>oc3_seq1</sub> <sup>3</sup>	331 776, 0	206 074	1, 6	05 m 49 s
Q <sub>oc3_seq2</sub> <sup>3</sup>	0, 3	6 985	21 558, 6	2 s
Q <sub>oc3_seq3</sub> <sup>3</sup>	0, 4	6 985	17 827, 0	2 s
Q <sub>oc3_seq4</sub> <sup>3</sup>	331 776, 0	297 833	1, 1	02 m 39 s
Q <sub>oc3_seq5</sub> <sup>3</sup>	199	6 985	35, 1	18 s
Q <sub>oc3_seq6</sub> <sup>3</sup>	12, 7	6 436	508, 1	0 s
				8 m 50 s

<sup>2</sup> = Datenbasis TPCH2; <sup>3</sup> = Datenbasis TPCH3

Tabelle 5.5: Messergebnisse der Anfragesequenzen

Abbildung 5.9 zeigt Ausschnitte zweier Ausführungspläne zu Teilen der Anfragesequenz  $Q_{o}a1\_seq$ . In einem ersten Schritt ( $Q_{o}a1\_seq\_1$ ) werden Daten in der temporären Tabelle *TEMPTABLEA010* zwischengespeichert. Die Kardinalität des INSERT-Operators wird mit 1 395 920 recht genau geschätzt. Der Wert steht im Widerspruch zu der Kardinalität von 94, welche für die temporäre Tabelle angenommen wird. Dieser Wert müsste mit der Kardinalität des INSERT-Operators übereinstimmen. Die nachfolgende Schätzung der Anfrage  $Q_{o}a1\_seq\_3$  verwendet den Wert  $Card_g(TEMPTABLEA010) = 94$ , obwohl die Tabelle 1 229 098 Tupel enthält.

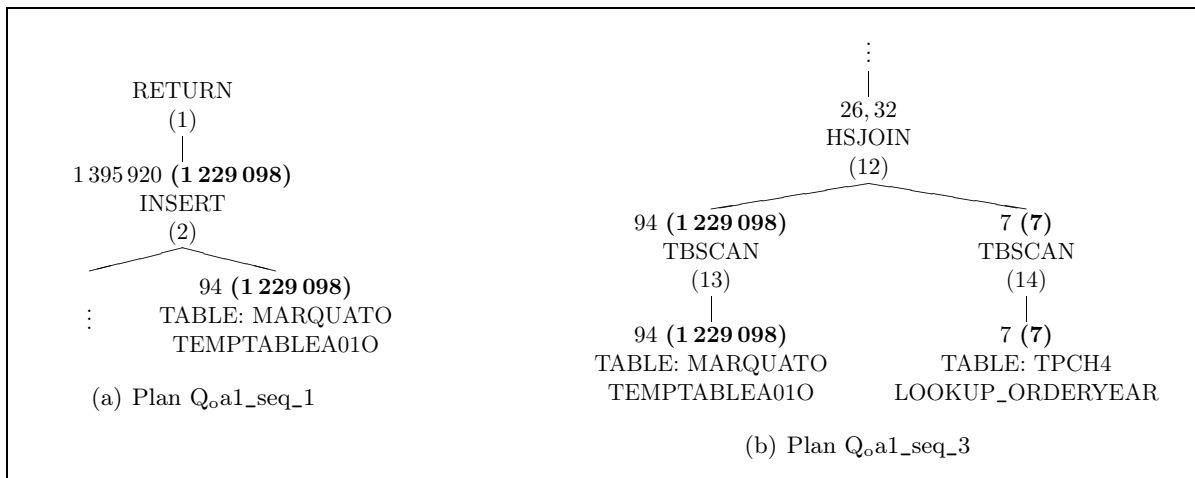


Abbildung 5.9: Ausführungspläne zu den Anfragen der Sequenz  $Q_{o}a1\_seq$

Diese Unstimmigkeit konnte nicht aufgelöst werden. Eine Aussage über die Qualität der Schätzergebnisse der Anfragesequenzen ist somit ebenso wenig möglich wie ein Vergleich mit den Schätzwerten der einzelnen Anfragen.

## 6 Ergebnisse

Die Ergebnisse dieser Arbeit werden nachfolgend beschrieben. Dabei wird der Bezug zu den vorangegangenen Kapiteln hergestellt. Die durchgeführten Messungen ermöglichen sowohl eine Beurteilung der theoretischen Überlegungen zu den verschiedenen Einflussparametern aus Kapitel 3 als auch der vorhandenen Ansätze zur Bewertung von Kosten- und Kardinalitätsschätzungen aus Kapitel 4. Mögliche Anwendungsszenarien für Bewertungsverfahren bilden den Schluss dieses Kapitels.

### 6.1 Interpretation der Messergebnisse

Der Zusammenhang zwischen den Einflussparametern und der Qualität der Schätzergebnisse wurde sowohl theoretisch in Kapitel 3 als auch experimentell in Kapitel 5 untersucht. Die Interpretation der Messergebnisse soll die Gemeinsamkeiten und Unterschiede der beiden Betrachtungsweisen aufzeigen. Der Umfang der Messungen ist mit 53 untersuchten Anfragen nicht groß genug, um bei den Ergebnissen von fundierten Aussagen sprechen zu können. Da die Anfragen jedoch speziell für die Überprüfung der theoretischen Erkenntnisse ausgewählt wurden, sind die Ergebnisse für eine Beurteilung der Einflussparameter aussagekräftig genug.

Das aus Kapitel 3 gewonnene Verständnis für den Zusammenhang von Einflussparametern und Qualität der Schätzergebnisse wurde durch die Messungen weitgehend bestätigt. Die einzelnen Einflussfaktoren und deren Auswirkungen werden nachfolgend durch die Interpretation der Messergebnisse beurteilt.

1. Die Messungen zeigten, dass die *Qualität der Statistiken* (s. Abschnitt 3.1) wesentlich für die Genauigkeit der geschätzten Kardinalitäten verantwortlich ist. Für die meisten Attribute existierten Equi-Depth-Histogramme mit der Datenverteilung der Attributwerte in der Tabelle SYSTAT.COLDDIST des Datenbankkatalogs. Selektionen mit Attributen von Basistabellen (Blattknoten des Ausführungsplans) konnten mit den Histogramminformationen genau abgeschätzt werden. Traten hier Abweichungen zwischen geschätzten und realen Werten auf, so hatten diese nur vernachlässigbar kleine Auswirkungen auf den gesamten Fehlerfaktor der Anfrage. Die Qualität der Schätzergebnisse verschlechterte sich, sobald die Abschätzungen an Zwischenknoten durchgeführt wurden, für welche die Basisstatistiken aus dem Datenbankkatalog nicht mehr exakt mit den vorliegenden Daten übereinstimmten. Zu großen Schätzfehlern führte die Verwendung von Standard-Filterfaktoren, wenn keine Statistiken für die Kardinalitätsschätzung verfügbar waren (s. Abschnitt 5.2.5).

Die Messergebnisse decken sich mit den theoretischen Überlegungen, die in Abbildung 3.2 auf Seite 18 dargestellt sind. Die Qualität der Schätzwerte steigt abhängig von den Statistiken: keine Statistiken (Schätzungen mit Standard-Filterfaktoren), ungenaue Statistiken (Schätzungen an Zwischenknoten), genaue Statistiken (Schätzungen an Blattknoten).

2. Die Messungen gaben Aufschluss über den Einfluss der **Struktur der Daten** (s. Abschnitt 3.2) auf die Schätzqualität. Insbesondere die Annahme des Optimierers, dass alle Daten unabhängig voneinander sind, führte in einigen Fällen zu großen Schätzfehlern. Das Datenbankschema der Messungen enthält Abhängigkeiten zwischen verschiedenen Datumfeldern. Wenn solche Abhängigkeiten nicht über Integritätsbedingungen im DBS abgebildet sind, so kann über die Qualität der Schätzung keine Aussage gemacht werden. Datenabhängigkeiten sind ein bekanntes Problem in der Optimierung. Mehrdimensionale Histogramme könnten hier Abhilfe schaffen. Im verwendete DBS war diese Form der Statistiken jedoch nicht verfügbar.

Auch die zweite Annahme des Optimierers, dass alle Attributwerte gleichverteilt sind, stellte sich als problematisch heraus. Knoten, an denen die Verwendung von Histogramm-Informationen nicht möglich war und für die somit eine Gleichverteilung der Attributwerte angenommen werden musste, erwiesen sich als fehleranfällig. Das ist darauf zurückzuführen, dass diesen Knoten bereits Operationen vorausgingen, welche die ursprüngliche Gleichverteilung der Daten negativ beeinflusst haben. Diese Problematik trat insbesondere im Zusammenhang mit der Verwendung temporärer Tabellen und Standard-Filterfaktoren auf (s. Abschnitt 5.2.7), aber auch bei Gruppierungen, bei denen alle Werte als existent angenommen wurden, obwohl vorausgegangene Operationen bereits Daten gefiltert hatten (s. Abschnitt 5.2.2, Anfrage Q<sub>t05</sub>).

3. Der Einfluss der **anfragespezifischen Parameter** (s. Abschnitt 3.3) erwies sich nur teilweise als vorhersehbar. Als Maß für die Komplexität der Anfrage wurden für die Messungen folgende Eigenschaften herangezogen: die Anzahl der Tabellen, die Anzahl der Prädikate, die Schachtelungstiefe sowie die Verwendung von Gruppierungsoperatoren (s. Tabelle 5.1 auf Seite 46). Diese anfragespezifischen Parameter geben keinen Aufschluss über die Qualität der Schätzwerte. Die Messergebnisse enthalten sowohl Anfragen mit relativ vielen Tabellen und Prädikaten, deren Kardinalität exakt geschätzt wurde (z. B. Q<sub>t09</sub> und Q<sub>e01</sub>) als auch Anfragen mit einer geringen Anzahl an Tabellen und Prädikaten, bei denen Schätzfehler entstanden sind (z. B. Q<sub>t15</sub> und Q<sub>e19</sub>). Ebenso existieren Anfragen mit und ohne Gruppierungsoperatoren, deren Schätzfehler unterschiedlich stark ausgeprägt sind.

Die Unvorhersehbarkeit ist auf die verwendete Granularität der Anfragekomplexität zurückzuführen. Diese ist zu grob, um alle Eigenschaften einer Anfrage abzubilden. Neben der Anzahl der Prädikate ist eine weitere Aufgliederung der verwendeten Prädikate sinnvoll. Prädikate mit Stringvergleichen (LIKE-Prädikat) konnten mehrmals als Fehlerursache identifiziert werden. Ebenso führte die Verwendung eines Anti-Verbunds (NOT-EXISTS-Prädikat, OUTER JOIN mit Vergleichswert NULL) stets zu Schätzfehlern. Diese Prädikate dürfen nicht mit einem Selektionsprädikat gleichgestellt werden, für das meist genaue Schätzungen möglich sind. Darüber hinaus ist die Gruppierung differenzierter zu betrachten. Nach welchem Attribut werden die Daten gruppiert? Existieren für dieses Attribut Statistiken? Auf welcher Schachtelungstiefe findet die Gruppierung statt? Das sind Faktoren, welche die Schätzungen beeinflussten, die jedoch bei der Aussage „Gruppierung Ja oder Nein“ nicht berücksichtigt werden. Eine feingranularere Beschreibung der Anfragekomplexität ist notwendig, um den Einfluss der anfragespezifischen Parameter

differenzierter darstellen zu können. Auf Grund der dadurch entstehenden Kombinationsmöglichkeiten der Anfrageeigenschaften erfordert das jedoch eine größere Anzahl an Anfragen für die Messungen, als in dieser Arbeit untersucht wurden.

Die Messergebnisse lassen einen direkten Zusammenhang der Schachtelungstiefe zur Schätzqualität vermuten. Alle Anfragen mit einer Schachtelungstiefe  $S \geq 2$  haben einen Fehlerfaktor von  $|F| > 2$ . Für die beiden einzigen Anfragen mit einer Schachtelungstiefe von  $S = 3$  (Q<sub>oc</sub>1 und Q<sub>oc</sub>3) wurden Werte von  $F > 10^6$  gemessen. Wie groß der Anteil der Schachtelungstiefe an diesen Abweichungen ist, lässt sich nicht bestimmen. Die großen Fehler für  $S = 3$  sind z. B. auf die Verwendung temporärer Tabellen im Ausführungsplan zurückzuführen.

Letztendlich sind nur einzelne Aussagen über die anfragespezifischen Parameter möglich. Übereinstimmend mit den Überlegungen aus dem theoretischen Teil sind die negativen Auswirkungen auf die Schätzgenauigkeit durch die Verwendung von Attributarithmetik (z. B. Stringoperationen) sowie bei Selektionen abhängiger Attributwerte. Darüber hinaus wurden weitere Einflussgrößen identifiziert, welche ebenfalls zu Schätzwerten führten, die von den tatsächlichen Kardinalitäten abweichen. Im Einzelnen waren das: Anfragen mit einem Anti-Verbund, Anfragen mit einer Schachtelungstiefe von  $S \geq 2$  und Anfragen mit Vergleichsprädikaten, die Werte von Aggregatfunktionen enthalten.

4. Als *ausführungsplanspezifische Parameter* sind im ersten Teil der Arbeit die Struktur des Ausführungsplans und die Planoperatoren angeführt (s. Abschnitt 3.4). Die Planoperatoren haben nur Auswirkungen auf die Laufzeit, nicht aber auf die Kardinalitäten des Ausführungsplans. Die Ausführungszeiten konnten auf Grund der unterschiedlichen Einheiten des DBS (Timerons bei den Ausführungsplänen und Sekunden bei db2batch) nicht verglichen werden. Aussagen über den Einfluss der verwendeten Planoperatoren und deren Realisierung auf die Qualität der Kostenschätzung sind deshalb nicht möglich.

Bei den Ausführungsplänen erwies sich die Verwendung temporärer Tabellen als Fehlerquelle. Das stellt einen Widerspruch zu den Behauptungen aus Abschnitt 3.4.1 dar, wonach sich die Reduzierung der Knotenzahl im Rahmen der Anfragetransformation nicht auf die Qualität der Schätzergebnisse auswirkt. Die Identifizierung gemeinsamer Teilbäume und der Einsatz temporärer Tabellen führten bei den Messungen jedoch zur Verwendung von Standard-Filterfaktoren und somit zu Abweichungen in der Größenordnung von  $F = 100\,000$ . Dieses Verhalten wurde bei zwei Anfragen gemessen (Q<sub>oc</sub>1 und Q<sub>oc</sub>3). Die Verwendung von Standard-Filterfaktoren im Ausführungsplan hatte stets schlechte Schätzwerte zur Folge.

5. *Laufzeitparameter* wurden bei den Messungen nicht berücksichtigt. Es stand ausschließlich die statische Optimierung im Mittelpunkt. Eine Bewertung der theoretischen Überlegungen aus Abschnitt 3.5 ist deshalb nicht möglich.

Ein Großteil der theoretischen Überlegungen wird durch die Messungen bestätigt. Darüber hinaus enthält das Ergebnis weitere Einflussparameter, deren Bedeutung im ersten Teil der Arbeit vernachlässigt wurde. Insbesondere bei der Verwendung des Anti-Verbunds, bei der Verwendung von Rückgabewerten von Aggregatfunktionen in Vergleichsprädikaten und beim Einsatz temporärer Tabellen wurden verhältnismäßig große Schätzfehler nachgewiesen.

Für Aussagen über den Zusammenhang verschiedenener Einflussparameter auf die Schätzqualität ist ein größeres und breiteres Anfragespektrum notwendig, d. h., es bedarf einer größeren Anfragenmenge und einer feingranulareren Beschreibung der Anfragekomplexität. Diese Parameter bieten Ansatzpunkte für weitere Messreihen.

## 6.2 Beurteilung der Bewertungsansätze

Die in Kapitel 4 vorgestellten statischen Bewertungsansätze werden nachfolgend beurteilt. Darüber hinaus werden eigene Erweiterungen auf Basis der Messergebnisse vorgeschlagen.

### 6.2.1 Beurteilung der quantitativen Ansätze

Die quantitativen Bewertungsansätze aus Abschnitt 4.3 liefern obere Fehlerschranken für Kardinalitätsschätzungen mit Histogrammen, indem für jeden Bucket oder für jedes Histogramm die maximale Abweichung gespeichert wird. Bei dem verwendeten Datenbanksystem DB2 werden solche maximalen Abweichungen nicht im Datenbankkatalog gehalten. Allerdings gab es bei keiner Anfrage Probleme bei Kardinalitätsschätzungen mit Histogrammen, wenn diese auf Basistabellen angewandt wurden. Die dabei entstandenen Abweichungen bewegen sich in Größenordnungen, die den Fehlerfaktor der gesamten Anfrage nicht beeinflussen.

```
(Qt03) SELECT      l_orderkey, SUM(l_extendedprice * (1 - l_discount)) AS revenue,
                   o_orderdate, o_shippriority
FROM              tpch4.customer, tpch4.orders, tpch4.lineitem
WHERE             c_custkey = o_custkey AND l_orderkey = o_orderkey AND
                   c_mktsegment = 'BUILDING' AND
                   o_orderdate < date('1995-03-15') AND
                   l_shipdate > date('1995-03-15')
GROUP BY         l_orderkey, o_orderdate, o_shippriority
ORDER BY         revenue DESC, o_orderdate
```

Die Anfrage Q<sub>t</sub>03 enthält drei Selektionen, deren Abschätzungen mit Histogrammen durchgeführt werden und die genaue Schätzwerte liefern. Der Ausführungsplan in Abbildung 6.1 zeigt, dass die Abweichungen in den Knoten 9, 11 und 12 gegenüber dem gesamten Fehler der Anfrage vernachlässigbar klein sind.

$$|F_{k=11}| = 1,01 \leq |F_{k=9}| = 1,02 \leq |F_{k=12}| = 1,03 \ll |F_{Q_t03}| = 26,5$$

Der gesamte Schätzfehler  $F_{Q_t03} = 26,5$  wird im Wesentlichen durch die fehlerhafte Schätzung in Knoten 8 und der nachfolgenden Gruppierung verursacht.

Eine quantitative Bewertung der Kosten- und Kardinalitätsschätzung auf Histogrammebene, wie sie in Abschnitt 4.3 beschrieben ist, erwies sich bei den gemessenen Anfragen als nutzlos, da die auftretenden Fehler die Größenordnungen nachfolgender Schätzungen nicht beeinflussten. Darüber hinaus sind die Bewertungsverfahren nur auf Selektivitätsschätzungen von Basistabellen anwendbar. An Zwischenknoten im Ausführungsplan liefert diese Fehlerbeschreibung falsche

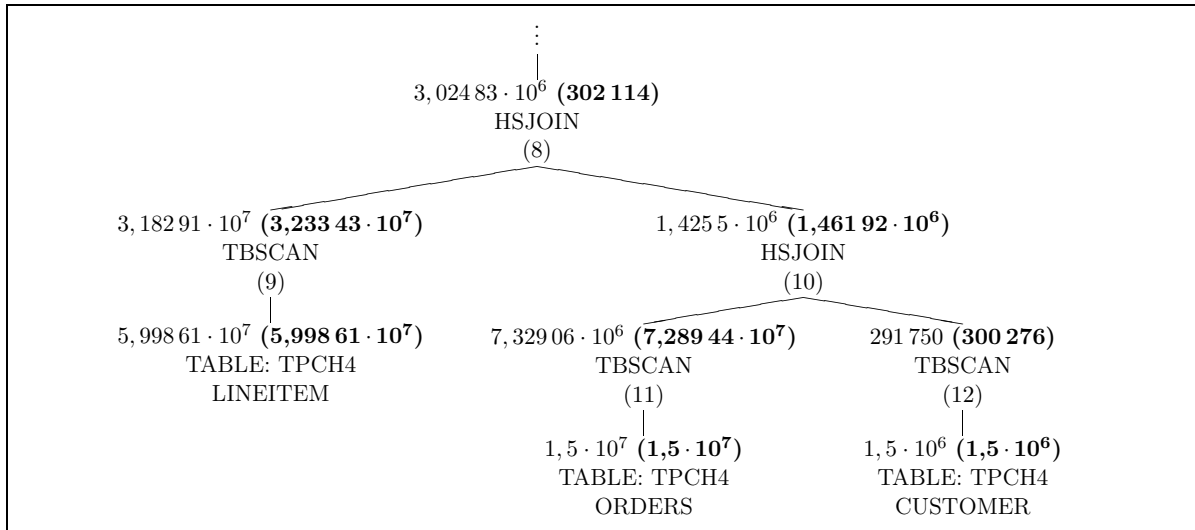


Abbildung 6.1: Ausführungsplan zu Anfrage  $Q_t03$

Werte, da die Datenverteilung des eingehenden Objektstroms nicht mit der ursprünglichen Datenverteilung zum Zeitpunkt der Histogrammerstellung übereinstimmt. Die vorgestellten quantitativen Bewertungsansätze sind ungeeignet, um Qualitätsaussagen über den Schätzfehler der gesamten Anfrage zu machen. Sie sind stets an bestimmte Schätzverfahren gebunden (hier die Schätzung mit Histogrammen) und ermöglichen keine ganzheitliche Betrachtungsweise.

### 6.2.2 Beurteilung der qualitativen Ansätze

Die Verwendung von Statistiken für Zwischenergebnisse und die Fehlerbeschreibung durch ein Ungenauigkeitspotenzial UP sind in Abschnitt 4.2 als qualitative Bewertungsansätze beschrieben. Diese Bewertungsverfahren eignen sich im Vergleich zu den quantitativen Verfahren für allgemeinere Aussagen über die Qualität der Schätzwerte einer Anfrage, da sie nicht an spezielle Schätzverfahren gebunden sind.

#### Statistiken für Zwischenergebnisse

Während für die Schätzwerte an den Blattknoten des Ausführungsplans eine hohe Genauigkeit nachgewiesen wurde, traten Fehler vor allem an Zwischenknoten auf. Abschnitt 4.2.1 beschreibt, wie mit Statistiken auf Zwischenergebnissen und Sichten die Qualität der Schätzwerte verbessert werden kann. Der Einfluss solcher Zwischenstatistiken auf die Schätzgenauigkeit konnte durch die Messungen nicht ermittelt werden, da im verwendeten DBS derartige Statistiken nicht implementiert waren. Aber auch ohne experimentellen Nachweis ist das Optimierungspotenzial offensichtlich: An allen Knoten, deren Kardinalitäten fehlerhaft geschätzt werden, kann die Genauigkeit durch den Einsatz von Statistiken für diese Knoten verbessert werden. Je größer die Zahl der Statistiken, die in einem konkreten Plan für die Kostenschätzung verwendet werden, desto höher ist die Qualität der Schätzergebnisse. Theoretisch kann der Fehlerfaktor

dadurch bis auf den Wert  $F = 1$  gesenkt werden, indem für alle inneren Knoten Statistiken im DBS bereitgestellt werden. Hier gilt es jedoch, den entstehenden Aufwand gegenüber dem erzielten Nutzen abzuwägen. Bei einer großen Menge unterschiedlicher Anfragen, die alle zu unterschiedlichen Ausführungsplänen führen, ist dieses Vorgehen nicht praktikabel.

### Bewertung durch Ungenauigkeitspotenzial

Die Bewertung der Schätzqualität durch das Ungenauigkeitspotenzial UP von Kabra und DeWitt ist der einzige Ansatz, der eine Bewertung des gesamten Ausführungsplans vornimmt [KW98]. Für die geschätzte Kardinalität jedes Knotens und somit auch für die Kardinalität des Anfrageergebnisses kann ein  $UP \in \{\text{niedrig}, \text{mittel}, \text{hoch}\}$  berechnet werden.

Wendet man die Regeln auf die Wurzeln der Ausführungspläne aus den Messungen an, so zeigt sich, dass nur wenige Wurzelknoten mit dem Wert *niedrig* oder *mittel* bewertet werden (s. Tabelle 6.1 auf Seite 74). Insgesamt 46 der 53 Anfragen wird ein hohes UP zugewiesen. Das liegt u. a. daran, dass für Schätzungen mit Equi-Depth-Histogrammen, wie sie in DB2 zum Einsatz kommen, bereits ein UP von *mittel* angesetzt wird. Außerdem werden Anfragen, die Gruppierungen enthalten, mit *hoch* bewertet. Für die meisten Anfragen bleiben lediglich zwei Ausprägungen für die Bewertung der Schätzqualität übrig (*mittel* und *hoch*). Man sollte bedenken, dass Kabra und DeWitt das UP verwenden, um die Stellen im Plan zu identifizieren, an denen eine Überprüfung der Schätzwerte zur Laufzeit nötig ist. Das Ziel sind weniger Bewertungsaussagen über den Wurzelknoten des Ausführungsplans einer Anfrage.

Die Regeln, die das Verhältnis von  $UP(\text{Input})$  zu  $UP(\text{Output})$  beschreiben, decken sich mit den Beobachtungen der Messungen. Das UP eines Anti-Verbunds wird ebenso mit *hoch* eingestuft wie die Bestimmung der Anzahl eindeutiger Attributwerte von Zwischenergebnissen. Auch der Einsatz benutzerdefinierter Methoden wird mit *hoch* angesetzt. Verwendet man hier die Definition der „SQL-invoked routines“ aus dem SQL-Standard, so gehören dazu auch die Methoden DAYS, MOD und SUBSTR. Für diese Methoden konnten Schätzfehler nachgewiesen werden. Das UP für Kardinalitätsschätzungen mit Histogrammen wird mit dem Wert  $UP = \text{mittel}$  dagegen geringer bewertet. Gute Schätzwerte für Histogramme konnten durch die Messungen ebenso bestätigt werden wie die beiden folgenden Regeln für das Propagieren des UP durch den Ausführungsplan: Das UP bleibt unverändert bei Selektionen mit einem Attribut sowie bei Gleichverbunden mit Schlüsselattributen und erhöht sich um eine Stufe bei Selektionsprädikaten mit mehreren Attributen einer Relation sowie bei Gleichverbunden ohne Schlüsselattribute. Die Erhöhung des UP bei Prädikaten mit mehreren Attributen wird von Kabra und DeWitt mit Datenabhängigkeiten begründet, die durch eindimensionale Histogramme nicht abgebildet werden können. Probleme bei Datenabhängigkeiten konnten experimentell nachgewiesen werden, beispielsweise bei Anfrage Q<sub>t</sub>19. Das Modell berücksichtigt jedoch keine relationenübergreifenden Datenabhängigkeiten, die ebenfalls als Ursache für Schätzfehler identifiziert wurden. Auch die Bewertung des Gleichverbunds abhängig von den Verbundattributen wurde durch die Messergebnisse bestätigt. Kardinalitätsschätzungen mit Schlüsselattributen lieferten bessere Schätzergebnisse als Schätzungen mit Nicht-Schlüsselattributen. Der Optimierer kann dazu sein Wissen über die referenzielle Integrität anwenden. Bei Nicht-Schlüsselattributen wie beispielsweise bei dem Verbundprädikat  $l\_comment = o\_comment$  in Anfrage Q<sub>e</sub>25 entstanden größere Schätzfehler ( $F = -22, 7$ ).

Das UP von Kabra und DeWitt enthält ein verwendbares Regelwerk zur Bewertung von Kosten- und Kardinalitätsschätzungen. Mit den Ungenauigkeitsstufen *niedrig*, *mittel* und *hoch* ist eine Abstufung möglich, ohne den Schätzfehler quantitativ beschreiben zu müssen. Durch die Regeln, welche das Verhältnis von  $UP(Input)$  zu  $UP(Output)$  beschreiben, kann das UP durch den Plan propagiert werden. Dabei steht man vor dem Problem, dass bereits bei Plänen mit einer geringen Knotenzahl der Schätzung des Anfrageergebnisses der Wert *hoch* zugewiesen wird. Die Pläne der OLAP-Anfragen ( $Q_o$ ) verdeutlichen das. Auf Grund der hohen Komplexität erreicht das UP bei allen OLAP-Anfragen bis zum Wurzelknoten den Wert  $UP = hoch$ . Die Schätzqualitäten sind jedoch verschieden ( $F_{Q_{ob1}} = 1, 1$ ;  $F_{Q_{oc2}} = -2, 2$ ;  $F_{Q_{oa1}} = 6149, 3$ ;  $F_{Q_{oc3}} \approx 10^8$ ;  $F_{Q_{oc1}} \approx 10^{11}$ ). Hier steht man vor einem Dilemma. Auf der einen Seite sind qualitative Aussagen, wie sie durch das UP gegeben sind, erwünscht. Andererseits führt das dazu, dass Einflussparameter, welche die Schätzqualität unterschiedlich stark beeinflussen, gleich bewertet werden. Differenzierte Aussagen sind kaum möglich.

### 6.2.3 Erweiterung der qualitativen Bewertung

Der Ansatz von Kabra und DeWitt, die Qualität der Kosten- und Kardinalitätsschätzungen durch ein UP zu bewerten, wird nachfolgend erweitert. Dabei sollen die Unzulänglichkeiten des Ansatzes korrigiert und die Erkenntnisse aus den Messungen bezüglich Schätzqualität berücksichtigt werden.

- *Erweiterung der Ausprägungen des UP*

Die bisherige Anzahl an Ausprägungen ermöglicht keine differenzierte Bewertung der Schätzqualität. Die Untersuchungen in Abschnitt 6.2.2 haben gezeigt, dass bis auf eine Ausnahme nur die beiden Werte *mittel* und *hoch* auftreten. Es empfiehlt sich hier, mindestens eine weitere Stufe einzuziehen. Denkbar sind fünf Stufen:  $UP \in \{1, 2, 3, 4, 5\}$ , wobei  $UP = 1$  dem Wert *niedrig* und  $UP = 5$  dem Wert *hoch* entspricht.

- *Korrektur des UP für das Equi-Depth-Histogramm*

Die Kardinalitätsschätzungen mit den in DB2 verwendeten Histogrammen lieferten im Vergleich zu anderen Abschätzungen genaue Ergebnisse. Für Abschätzungen mit Equi-Depth-Histogrammen kann  $UP = 1$  angenommen werden.

- *UP für Probleme aus den Messungen*

Für folgende, bei den Messungen aufgetretene Probleme wird das UP auf den maximalen Wert  $UP = 5$  gesetzt, da in diesen Fällen die größten Schätzfehler gemessen wurden:

- Anti-Verbund (s. Abschnitt 5.2.8)
- Vergleichsprädikate mit Werten von Aggregatfunktionen (s. Abschnitt 5.2.2)
- LIKE-Prädikat, für dessen Schätzung keine Histogramme verwendet werden können (s. Abschnitt 5.2.6)
- Schätzungen, die mit Standard-Filterfaktoren durchgeführt werden (s. Abschnitt 5.2.5)

Für Selektionsprädikate mit mehreren Attributen konnten Schätzfehler nachgewiesen werden, die auf Datenabhängigkeiten zurückzuführen waren. Die Schätzfehler hatten allerdings nicht die Größenordnung wie beispielsweise Fehler bei Anti-Verbunden. Um diesem

Qualitätsunterschied gerecht zu werden, wird für Schätzungen von Selektionsprädikaten mit mehreren Attributen einer Relation an Blattknoten ein  $UP = 3$  gesetzt. Derartige Schätzungen an Zwischenknoten erhöhen das UP um eine Stufe.

- *Anzahl unterschiedlicher Attributwerte*

Schätzungen der Anzahl unterschiedlicher Attributwerte von Basistabellen werden bei Kabra mit *niedrig* bewertet, in allen anderen Fällen mit *hoch*. Das führt dazu, dass das UP für Ausführungspläne mit einem Gruppierungsoperator in den meisten Fällen ebenfalls mit *hoch* eingestuft werden, da Gruppierungen selten nach Blattknoten durchgeführt werden. Es wird deshalb vorgeschlagen, für Operationen, welche die Anzahl unterschiedlicher Werte an inneren Knoten bestimmen, das UP um zwei Stufen zu erhöhen. Dadurch werden diese Operationen als gewichtige Fehlerquelle berücksichtigt, es wird aber gleichzeitig verhindert, dass sofort der maximale Ungenauigkeitswert angesetzt wird.

- *Kartesisches Produkt*

Bei Kabra wird keine Aussage über das UP eines kartesischen Produkts gemacht. Alle Nicht-Gleichverbunde werden mit *hoch* bewertet. Die Kardinalität für das kartesische Produkt ist gleich dem Produkt der Kardinalitäten der eingehenden Objektströme. Diese Schätzung ist ebenso genau möglich, wie die Schätzung eines Gleichverbunds mit Schlüsselattributen, d. h.  $UP = \text{niedrig}$ .

- *UP für erweiterte Statistiken*

Existiert im DBS die Möglichkeit, Statistiken für Sichten oder Zwischenergebnisse bereitzustellen, so kann auch dafür ein UP spezifiziert werden. Der qualitative Bewertungsansatz aus Abschnitt 4.2.1, der Schätzungen mit Zwischenstatistiken mit *gut* und Schätzungen ohne Statistiken mit *schlecht* bewertet, lässt sich in das UP-Modell integrieren. Für Schätzungen, die Statistiken für Zwischenergebnisse oder Sichten verwenden, kann  $UP = 1$  angenommen werden.

Unterstützt das DBS mehrdimensionale Histogramme, so kann dadurch die Schätzung von Selektionsprädikaten mit mehreren abhängigen Attributen verbessert werden. Für Blattknoten kann  $UP = 1$  gesetzt werden.

Die Schätzergebnisse der Anfragen wurden mit diesem modifizierten Regelwerk bewertet. Im Gegensatz zu dem UP nach Kabra und DeWitt weist es eine breitere Streuung auf (s. Tabelle 6.1): Es wurden nur 31 statt bisher 46 Anfragen mit dem maximalen UP bewertet. Tabelle 6.2 zeigt die Mediane für die verschiedenen Bereiche der Ungenauigkeitspotenziale. Die Bereiche  $UP = 2$  und  $UP = 4$  enthalten nur drei bzw. zwei Werte, weshalb die zugehörigen Mediane nicht sonderlich aussagekräftig sind. Die Ergebnisse zeigen aber, dass ein Wert  $UP < 5$  ein relativ guter Indikator für genaue Schätzergebnisse ist. Der maximale Fehlerfaktor in diesem Bereich beträgt  $|F| = 1406,3$ ; der Median liegt bei  $\tilde{x} = 1,15$ . Die Anfragen mit den größten Schätzfehlern, welche diesen Maximalwert aus dem Bereich  $UP < 5$  um mehrere Größenordnungen übersteigen, werden alle mit  $UP = 5$  bewertet.  $UP = 5$  bedeutet jedoch nicht zwangsläufig eine geringe Schätzqualität. Es werden einige Pläne mit dem maximalen UP bewertet, deren Schätzwerte eine hohe Genauigkeit aufweisen, jedoch bedeutend weniger als mit dem Ansatz von Kabra.

6.2 Beurteilung der Bewertungsansätze

Anfrage	UP <sub>m</sub>	UP <sub>K</sub>	F
Q <sub>t</sub> 02	5	hoch	60 091, 6
Q <sub>t</sub> 09	5	hoch	1, 3
Q <sub>t</sub> 11	5	hoch	∞
Q <sub>t</sub> 13	5	hoch	117, 3
Q <sub>t</sub> 15	5	hoch	4 000, 0
Q <sub>t</sub> 16	5	hoch	1, 1
Q <sub>t</sub> 17	5	hoch	2, 5
Q <sub>t</sub> 18	5	hoch	1, 0
Q <sub>t</sub> 20	5	hoch	66 064, 3
Q <sub>t</sub> 21	5	hoch	8, 0 · 10 <sup>12</sup>
Q <sub>t</sub> 22	5	hoch	482 376, 0
Q <sub>e</sub> 02	5	hoch	3, 0
Q <sub>e</sub> 04	5	hoch	1, 4
Q <sub>e</sub> 06	5	hoch	18, 8
Q <sub>e</sub> 11	5	hoch	∞
Q <sub>e</sub> 12	5	hoch	242, 9
Q <sub>e</sub> 13	5	hoch	1, 7
Q <sub>e</sub> 15	5	hoch	1, 1
Q <sub>e</sub> 18	5	hoch	2, 3
Q <sub>e</sub> 19	5	hoch	∞
Q <sub>e</sub> 20	5	hoch	1, 0
Q <sub>e</sub> 21	5	hoch	∞
Q <sub>e</sub> 22	5	hoch	42 427, 9
Q <sub>e</sub> 23	5	hoch	∞
Q <sub>e</sub> 24	5	hoch	≡ 1
Q <sub>e</sub> 25	5	hoch	7 294 737, 8
Q <sub>e</sub> 26	5	hoch	∞

Anfrage	UP <sub>m</sub>	UP <sub>K</sub>	F
Q <sub>o</sub> a1	5	hoch	6 149, 3
Q <sub>o</sub> c1	5	hoch	1, 3 · 10 <sup>11</sup>
Q <sub>o</sub> c2	5	hoch	2, 2
Q <sub>o</sub> c3	5	hoch	98 703 697, 9
Q <sub>t</sub> 04	4	hoch	≡ 1
Q <sub>t</sub> 12	4	hoch	≡ 1
Q <sub>t</sub> 01	3	hoch	1, 5
Q <sub>t</sub> 03	3	hoch	26, 5
Q <sub>t</sub> 05	3	hoch	5, 0
Q <sub>t</sub> 07	3	hoch	1 406, 3
Q <sub>t</sub> 08	3	hoch	4, 5
Q <sub>t</sub> 10	3	hoch	1, 4
Q <sub>t</sub> 19	3	hoch	12, 9
Q <sub>e</sub> 09	3	hoch	1, 1
Q <sub>e</sub> 10	3	hoch	1, 3
Q <sub>e</sub> 16	3	hoch	≡ 1
Q <sub>o</sub> b1	3	hoch	1, 1
Q <sub>t</sub> 06	2	mittel	1, 1
Q <sub>e</sub> 03	2	hoch	2, 0
Q <sub>e</sub> 17	2	hoch	602, 6
Q <sub>t</sub> 14	1	mittel	1, 2
Q <sub>e</sub> 01	1	niedrig	1, 0
Q <sub>e</sub> 05	1	niedrig	1, 0
Q <sub>e</sub> 07	1	mittel	1, 0
Q <sub>e</sub> 08	1	niedrig	≡ 1
Q <sub>e</sub> 14	1	mittel	1, 1

UP<sub>m</sub> = Modifiziertes UP, UP<sub>m</sub> ∈ {1, 2, 3, 4, 5}; UP<sub>K</sub> = UP nach Kabra und DeWitt, UP<sub>K</sub> ∈ {niedrig, mittel, hoch}

Tabelle 6.1: Ungenauigkeitspotenzial und Schätzfehler der Anfragen

UP	Median
UP = 1	1,0
UP = 2	2,0
UP = 3	1,5
UP = 4	1,0
UP < 5	1,2
UP = 5	4000,0

UP	Median
UP = niedrig	1,0
UP = mittel	1,1
UP < hoch	1,0
UP = hoch	9,0

Tabelle 6.2: Mediane der Ungenauigkeitspotenziale

Es wurde gezeigt, dass sich der UP-Ansatz, der ursprünglich nur für die Bewertung innerer Knoten entwickelt wurde, derart verfeinern lässt, um eine Aussage für den Wurzelknoten eines Ausführungsplans und damit für eine Gesamtanfrage zu erhalten. Die vorgestellten Erweiterungen tragen den Erkenntnissen aus den Messungen Rechnung und ermöglichen eine detailliertere Beschreibung des Schätzfehlers. Man sollte sich bewusst sein, dass das Bewertungsverfahren u. a. aus den Messungen von ca. 50 Anfragen mit dem Datenbanksystem DB2 entstanden ist. Zur Verifikation der vorgestellten Bewertung der Schätzqualität durch das fünfstufige UP und für Erweiterungen des Verfahrens sind weitere umfangreiche Messungen, auch mit anderen DBS, notwendig. Der Fokus sollte dabei auf den Zusammenhang von UP und Schätzfehler gelegt werden, d. h., die verschiedenen Stufen des UP sollten möglichst gut mit den Größen der gemessenen Schätzfehler korrelieren. Abbildung 6.2 zeigt diesen Zusammenhang. Jedem

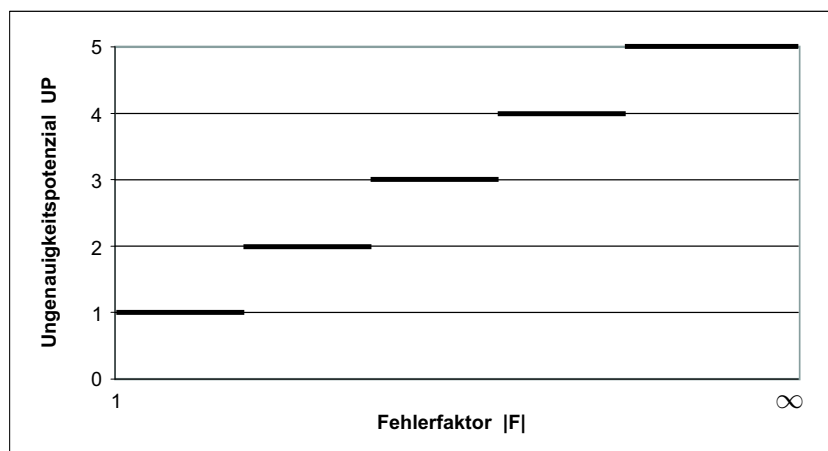


Abbildung 6.2: *Optimale Abbildung von Schätzfehler auf UP*

Schätzfehler wird ein UP zugewiesen:  $|F| \rightarrow UP$ . Eine optimale Abbildung ist dann erreicht, wenn die Funktion  $UP(|F|)$  monoton wachsend ist. Von diesem Idealfall ist das vorgestellte fünfstufige UP noch entfernt, es ist jedoch eine Verbesserung gegenüber dem UP von Kabra und DeWitt. Es stellt sich auch die Frage, ob eine weitere Abstufung des UP zu noch differenzierteren Qualitätsaussagen führt, bzw. ab wann die Monotonie der Funktion  $UP(|F|)$  nicht mehr gewährleistet werden kann.

### 6.3 Anwendungsszenarien

Es wurden einige Ansätze zur Bewertung von Kosten- und Kardinalitätsschätzungen vorgestellt. Die Genauigkeitsinformation wird für unterschiedliche Zwecke verwendet. Bei den quantitativen Bewertungsmethoden ermöglichen die Fehlerschranken Aussagen über die maximalen Abweichungen von geschätzter und realer Kardinalität des Anfrageergebnisses. Sie sind allerdings an spezielle Schätzverfahren gebunden. Die qualitativen Ansätze erlauben dagegen eine Beurteilung beliebiger Knoten des Ausführungsplans und somit auch eine Beurteilung der Schätzqualität für das Anfrageergebnis. Das geschieht entweder zum Übersetzungszeitpunkt (statische Bewertungsansätze) oder zur Laufzeit (dynamische Bewertungsansätze).

Die Qualitätsinformationen werden bei den vorgestellten Ansätzen unterschiedlich verwendet (s. Abschnitt 4.2), dienen jedoch in allen Anwendungsgebieten dazu, die Qualität der Schätzwerte in den Ausführungsplänen zu verbessern.

Die vorgestellten Bewertungsmöglichkeiten leisten jeweils in einem bestimmten Teilgebiet einen Beitrag zur Verbesserung der Schätzqualität. Deshalb ist auch von Bewertungsansätzen die Rede und nicht von Bewertungsverfahren. Der umfassendste Ansatz ist die Bewertung des Ausführungsplans durch das UP von Kabra und DeWitt. Im vorigen Abschnitt sind, basierend auf den theoretischen Überlegungen aus Kapitel 3 und den Messergebnissen aus Kapitel 5, Erweiterungen für diesen Ansatz vorgestellt worden. Einen Schritt weiter geht die nachfolgende Beschreibung eines Anwendungsszenarios: Die Bewertung durch das UP kann in die kostenbasierte Planauswahl integriert werden und zum Übersetzungszeitpunkt zu einer Verbesserung des Optimierungsprozesses führen.

Bisher wurde für die Planauswahl ein Kostenmodell als gewichtete Funktion zwischen E/A-Kosten und Berechnungskosten verwendet:

$$K = \text{Anzahl der physischen Seitenzugriffe} + W \cdot (\text{Anzahl der Aufrufe des Zugriffssystems})$$

Der kostengünstigste Plan wird für die Ausführung der Anfrage verwendet. Die Qualität der Kostenschätzung bleibt dabei unberücksichtigt. Folgendes Szenario ist denkbar: Bei einem Plan 1 weichen die geschätzten Ausführungskosten von den realen Kosten ab. Ein alternativer Plan 2 wird teurer geschätzt, besitzt aber eine höhere Schätzgenauigkeit. Der Optimierer wählt den Plan 1, obwohl die realen Ausführungskosten von Plan 2 kleiner sind.

Um zusätzlich zu den Kosten die Schätzqualität zu berücksichtigen, wird die Erweiterung des Kostenmodells um eine Bewertungskomponente vorgeschlagen. Die Bewertung verwendet das in Abschnitt 6.2.3 beschriebene fünfstufige UP. Folgende Funktion  $f$  skaliert den Wert des  $UP \in \{1, 2, 3, 4, 5\}$  abhängig von dem Skalierungsfaktor  $s \geq 1$  auf einen Wert  $f(UP) \in [1; s]$ :

$$f(UP) = \frac{s-1}{4} \cdot UP + \frac{5-s}{4} \quad s \geq 1$$

Multipliziert man das skalierte UP zur bisherigen Kostenfunktion, so erhält man eine modifizierte Kostenfunktion  $\tilde{K}$ , welche die Qualität der Schätzwerte berücksichtigt:

$$\tilde{K} = K \cdot \left( \frac{s-1}{4} \cdot UP + \frac{5-s}{4} \right)$$

Eine Parametrisierung ist über den Skalierungsfaktor  $s$  möglich. Dieser erlaubt eine Gewichtung der beteiligten Komponenten (geschätzte Kosten, Qualität der Kostenschätzung). Für einen Wert  $s = 1$  erhält man die bisherige Kostenfunktion:  $\tilde{K} = K$ . Der Einfluss des Faktors wird an einem Beispiel erläutert.

Tabelle 6.3 zeigt die Werte der modifizierten Kostenfunktion  $\tilde{K}$  für die Gewichtung mit  $s = 3$ . Angenommen, für einen Plan werden die Kosten mit der bisherigen Kostenfunktion auf  $K = k$  geschätzt, die Qualität der Schätzwerte wird mit  $UP = 3$  bewertet. Das ergibt für die modifizierten Kosten den Wert  $\tilde{K} = 2 \cdot k$  (dunkelgrau hinterlegt). Aus der Tabelle lässt sich ablesen, wie dieser Plan gegenüber anderen Plänen eingeschätzt wird. Er wird allen Plänen mit einem Wert  $\tilde{K} > 2 \cdot k$  vorgezogen (hellgrau hinterlegt) und unterliegt den Plänen mit Werten  $\tilde{K} < 2 \cdot k$ .

### 6.3 Anwendungsszenarien

	$K = 0,5 \cdot k$	$K = 0,75 \cdot k$	$K = k$	$K = 1,5 \cdot k$	$K = 2 \cdot k$
$UP = 1$	$0,500 \cdot k$	$0,750 \cdot k$	$1,000 \cdot k$	$1,500 \cdot k$	$2,000 \cdot k$
$UP = 2$	$0,750 \cdot k$	$1,125 \cdot k$	$1,500 \cdot k$	$2,250 \cdot k$	$3,000 \cdot k$
$UP = 3$	$1,000 \cdot k$	$1,500 \cdot k$	$2,000 \cdot k$	$3,000 \cdot k$	$4,000 \cdot k$
$UP = 4$	$1,250 \cdot k$	$1,875 \cdot k$	$2,500 \cdot k$	$3,750 \cdot k$	$5,000 \cdot k$
$UP = 5$	$1,500 \cdot k$	$2,250 \cdot k$	$3,000 \cdot k$	$4,500 \cdot k$	$6,000 \cdot k$

Tabelle 6.3: Bewertete Kosten  $\tilde{K}$  für  $s = 3$

Die Berücksichtigung des UP in der bewerteten Kostenfunktion  $\tilde{K}$  führt beispielsweise dazu, dass der Plan mit den geschätzten Ausführungskosten  $K = 1,5 \cdot k$  und einer Schätzqualität von  $UP = 1$  als besser eingestuft wird, als der Plan mit geringeren Kosten  $K = k$  und  $UP = 3$ .

Der Mehraufwand für die Berechnung der Kostenfunktion  $\tilde{K}$  im Vergleich zur Funktion  $K$  ist vernachlässigbar. Die Bewertung durch das UP kann parallel zur bisherigen Kosten- und Kardinalitätsschätzung für den Ausführungsplan durchgeführt werden. Durch die Parametrisierung kann der Einfluss der Bewertungskomponente variiert werden. Der Wert  $s = 1$  schaltet die Bewertungskomponente ganz aus. Der erzielte Nutzen dieser neuen Kostenfunktion  $\tilde{K}$  hängt wesentlich von der Qualität der Bewertungskomponente ab, d. h. davon, wie genau die Qualität der Schätzwerte durch das UP beschrieben wird. Gelingt die Abbildung von Schätzfehlern auf das UP gut, so kann die Optimierung von dem Wissen über die Schätzqualität profitieren. Ein mögliches Einsatzgebiet sind Anwendungen, die eine besonders exakte Schätzung der Ausführungsdauer voraussetzen. Die Funktion  $\tilde{K}$  ermöglicht es, nicht zwangsläufig den am kostengünstigsten geschätzten Plan auszuwählen, sondern einen Plan, dessen Schätzwerte gewissen Qualitätskriterien erfüllen (parametrisierbar über  $s$ ).

# 7 Schluss

Abschließend folgt eine Zusammenfassung der wesentlichen Punkte dieser Arbeit und der erzielten Ergebnisse. Darüber hinaus wird ein Ausblick für weitere Forschungsaktivitäten gegeben, welche dieser Arbeit folgen könnten.

## 7.1 Zusammenfassung

Ausgehend von der Anfrageverarbeitung in DBS wurde die kostenbasierte Anfrageoptimierung beschrieben und die Bedeutung der Kosten- und Kardinalitätsschätzungen für die Wahl eines möglichst optimalen Ausführungsplans herausgestellt. Der Optimierer verwendet für die Kostenschätzungen ein Kostenmodell als gewichtete Funktion zwischen E/A-Kosten und Berechnungskosten. Wesentlicher Einflussfaktor für diese Kostenfunktion sind die Kardinalitäten der Knoten im Ausführungsplan. Das DBS verwendet Statistiken aus dem Datenbankkatalog, um die Kardinalitäten möglichst genau abzuschätzen. Zu den am weitesten verbreiteten Statistiken gehören die Equi-Depth-Histogramme als Vertreter der nicht parametrischen Schätzverfahren.

Den einleitenden Betrachtungen folgte eine Beschreibung der wichtigsten Parameter, welche die Kosten- und Kardinalitätsschätzungen des Optimierers beeinflussen. Dazu gehören u. a. die Qualität der Statistiken im Datenbankkatalog und die Struktur der Daten. Die Annahmen des Optimierers über die Struktur der Daten (Datenunabhängigkeit und Datengleichverteilung) können zu Schätzfehlern führen. Weitere Einflussgrößen sind Parameter, die der SQL-Anfrage entnommen werden können wie z. B. die verwendeten Relationen, logischen Operatoren und Prädikate. Als ausführungsplanspezifische Parameter wurden die Struktur des Ausführungsplans und die verwendeten Planoperatoren identifiziert. Eine weitere Gruppe bilden die Laufzeitparameter, die im Gegensatz zu den anderen Einflussfaktoren erst zum Ausführungszeitpunkt der Anfrage bestimmt werden können.

Die Einflussparameter wirken sich unterschiedlich stark auf die Schätzungen des Optimierers aus. Zentraler Punkt dieser Arbeit war die Fragestellung nach Bewertungsmöglichkeiten für die Kosten- und Kardinalitätsschätzungen. Die wenigen Ansätze aus vorhergehenden Forschungsarbeiten wurden vorgestellt. Von besonderem Interesse sind die Bewertungsansätze, die bereits zum Übersetzungszeitpunkt eine Aussage über die Qualität der Schätzwerte ermöglichen. Es wurden zwei solche Ansätze identifiziert. Zum einen die Verwendung von Statistiken für Zwischenergebnisse, die eine exakte Kardinalitätsschätzung an inneren Knoten des Ausführungsplans ermöglichen. Die zweite und umfassendere Möglichkeit ist die Bewertung durch das Ungenauigkeitspotenzial UP von Kabra und DeWitt. Ein Regelwerk ermöglicht es, jedem Knoten ein UP zuzuweisen, d. h., eine Bewertung der Schätzqualität vorzunehmen.

Den theoretischen Überlegungen folgten Messungen mit dem Datenbanksystem DB2. Es wurden Anfragen spezifiziert, die verschiedenen Einflussparameter variiert sowie die geschätzten und die realen Werte für Kardinalität und Ausführungsdauer der Anfragen erfasst. Schätzfehler

unterschiedlicher Größenordnungen wurden dabei gemessen. Bei der anschließenden Plananalyse wurden mehrere Ursachen für die gemessenen Schätzfehler identifiziert. Die größten Abweichungen gab es bei Schätzungen von Anti-Verbunden, Gruppierungen, Vergleichsprädikaten mit Werten von Aggregatfunktionen, LIKE-Prädikaten, bei der Verwendung temporärer Tabellen sowie bei Schätzungen mit Standard-Filterfaktoren. Exakte Werte lieferten die Abschätzungen mit den Equi-Depth-Histogrammen, sofern sie auf Basistabellen durchgeführt wurden. Für den Einsatz an inneren Knoten konnten Schätzfehler gemessen werden, die auf Datenabhängigkeiten zurückzuführen waren.

Die Interpretation der Messergebnisse ermöglichte anschließend eine Bewertung der verschiedenen Einflussparameter und somit Aussagen über deren Auswirkungen auf die Schätzwerte des Optimierers. Die theoretischen Überlegungen wurden durch die Messungen weitgehend bestätigt. Besonders auffällig waren jedoch die große Anzahl an Schätzfehlern bei Vergleichsprädikaten mit Werten von Aggregatfunktionen und die großen Abweichungen bei der Verwendung von Standard-Filterfaktoren (z. B. für Anti-Verbunde und temporäre Tabellen im Ausführungsplan). Basierend auf den Messungen wurden die verschiedenen Bewertungsansätze beurteilt. Dabei erwies sich die Bewertung durch das Ungenauigkeitspotenzial UP als einzig brauchbarer Ansatz für Qualitätsaussagen an beliebigen Stellen im Ausführungsplan. Die dreistufige Bewertung (*niedrig, mittel, hoch*) ermöglicht jedoch keine differenzierten Aussagen, zumal die Stufe *niedrig* nur für serielle Histogramme angesetzt wird, die jedoch im verwendeten DBS nicht implementiert sind. Diese und weitere Unzulänglichkeiten wurden mit den Erkenntnissen aus dem Theorieteil sowie aus den Messungen korrigiert. Das Ergebnis ist ein fünfstufiges Bewertungsverfahren mit einem erweiterten Regelwerk für das Propagieren von Schätzfehlern. Das modifizierte UP ermöglicht eine breitere Streuung der Ergebnisse und eine genauere Abbildung der Schätzfehler durch das UP als der dreistufige Bewertungsansatz. Mit der Integration dieses Verfahrens in den Optimierungsprozess wurde abschließend ein Anwendungsszenario skizziert. Zusätzlich zur bisherigen Kostenschätzung ermöglicht die Bewertungskomponente eine Planauswahl, die nicht zwangsläufig den kostengünstigsten Plan wählt, sondern die Qualität der Schätzergebnisse berücksichtigt.

## 7.2 Ausblick

Im Mittelpunkt dieser Arbeit stand die Bewertung der Kosten- und Kardinalitätsschätzungen. Dem Einfluss der Kardinalitäten auf die Kosten  $K = E/A\text{-Kosten} + W \cdot \text{CPU-Kosten}$  wurde bereits im Theorieteil der Arbeit die größte Bedeutung beigemessen. Bei den Messungen wurden die geschätzten und die realen Werte der Kardinalitäten und der Ausführungszeiten von Zwischen- und Endergebnissen erfasst. Eine Analyse der Ausführungszeiten war nicht möglich, da die vom DBS gelieferten Einheiten nicht vergleichbar sind (geschätzte Laufzeit in *Timerons*, reale Laufzeit in *Sekunden*). Ein Vergleich zwischen geschätzter und realer Ausführungsdauer ermöglicht zusätzlich die Berücksichtigung der verschiedenen Planoperatoren, die keine Auswirkungen auf die Kardinalitäten haben.

Bei den Messungen konnte nur bedingt ein Zusammenhang zwischen Anfragekomplexität und Schätzqualität nachgewiesen werden. Das verwendete Maß für die Anfragekomplexität (Anzahl der Tabellen, Anzahl der Prädikate, Gruppierung sowie die Schachtelungstiefe) erwies sich als zu grob. Die Anfrageeigenschaften werden damit nicht differenziert genug erfasst. Das führt bei

einigen Anfragen dazu, dass die Komplexität ähnlich eingestuft wird, es jedoch Unterschiede bei den zugehörigen Schätzfehlern gibt. Für weitere Untersuchungen wird eine feingranularere Beschreibung der Anfragekomplexität empfohlen. Auf Grund der dadurch entstehenden Kombinationsmöglichkeiten ergibt sich die Notwendigkeit einer größeren Anfragemenge für nachfolgende Messungen. Das erfordert eine effiziente Ermittlung der realen Kardinalitäten an den Zwischenknoten im Ausführungsplan. DB2 gibt die realen Kardinalitäten innerer Knoten nicht aus. Deshalb wurden bisher für deren Bestimmung Ad-hoc-Anfragen spezifiziert. Für eine größere Anfragemenge, die für die Untersuchung einer detaillierteren Anfragekomplexität benötigt wird, ist dieses Vorgehen nicht mehr praktikabel.

Das in dieser Arbeit vorgestellte fünfstufige UP zeigt eine Möglichkeit, die Schätzqualität beliebiger Knoten im Ausführungsplan zu bewerten. Die Verlässlichkeit dieser Bewertung hängt im Wesentlichen davon ab, wie genau die Abbildung der Schätzfehler durch das UP gelingt. Bis auf einige wenige Ausnahmen signalisiert ein  $UP < 5$  eine gute Bewertung der Schätzqualität. Im Gegensatz dazu wurden einige Anfragen mit  $UP = 5$  bewertet, deren Schätzgenauigkeit hoch ist. Für nachfolgende Arbeiten empfiehlt es sich, die Ausnahmewerte, welche die Monotonie der Funktion  $UP(|F|)$  unterbrechen, genauer zu analysieren. Weitere Messreihen können an diesen Stellen ansetzen, um das Regelwerk des UP zu verfeinern und zu erweitern und somit eine verbesserte Abbildung des Schätzfehlers durch das UP zu erzielen.

Das vorgestellte Anwendungsszenario der Integration einer Bewertungskomponente in den Optimierer bietet ebenfalls einen Ansatzpunkt für weitere Forschungsarbeiten. Der Skalierungsfaktor  $s$  ermöglicht die Gewichtung der Bewertungskomponente gegenüber der herkömmlichen Kostenschätzung. Was ist eine sinnvolle Größe für diesen Wert? Außerdem ist es von Interesse, wie die modifizierte Kostenfunktion  $\tilde{K}$  die Planauswahl des Optimierers beeinflusst. In wie vielen Fällen führt die Verwendung von  $\tilde{K}$  zu einem anderen Plan als die Kostenfunktion ohne Bewertungskomponente? Eine Implementierung von  $\tilde{K}$  sowie weitere Messungen können dazu dienen, den Beitrag der Bewertung der Kosten- und Kardinalitätsschätzungen zur kostenbasierten Planauswahl der Optimierers zu quantifizieren.

# Literaturverzeichnis

- [AGPR99] Acharya, S.; Gibbons, P. B.; Poosala, V.; Ramaswamy, S.: Join Synopses for Approximate Query Answering. In: Proceedings of the 1999 ACM SIGMOD international conference on Management of data, Philadelphia, Pennsylvania, USA: ACM Press 1999
- [BC02] Bruno, N.; Chaudhuri, S.: Exploiting Statistics on Query Expressions for Optimization. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, ACM Press 2002
- [CG94] Cole, R. L.; Graefe, G.: Optimization of Dynamic Query Evaluation Plans. In: Proceedings of the 1994 ACM SIGMOD international conference on Management of data, Minneapolis, Minnesota, USA: ACM Press 1994
- [CGRS01] Chakrabarti, K.; Garofalakis, M.; Rastogi, R.; Shim, K.: Approximate query processing using wavelets. In: The VLDB Journal Vol. 10, No. 2-3, 199-223, 2001 – The International Journal on Very Large Data Bases, Secaucus, NJ, USA: Springer-Verlag New York, Inc. 2001
- [Chr84] Christodoulakis, S.: Implications of Certain Assumptions in Database Performance Evaluation. In: ACM Transactions on Database Systems, Vol. 9, No. 2, 163-186, 1984, New York, NY, USA: ACM Press 1984
- [CHS99] Chu, F.; Halpern, J. Y.; Seshadrit, P.: Least Expected Cost Query Optimization: An Exercise in Utility. In: Proceedings of the 18th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, Philadelphia, Pennsylvania, USA: ACM Press 1999
- [CR94] Chen, C. M.; Roussopoulos, N.: Adaptive Selectivity Estimation Using Query Feedback. In: Proceedings of the 1994 ACM SIGMOD international conference on Management of data, Minneapolis, Minnesota, USA: ACM Press 1994
- [DR99] Donjerkovic, D.; Ramakrishnan, R.: Probabilistic Optimization of Top N Queries. In: Proceedings of 25th International Conference on Very Large Data Bases, Edinburgh, Scotland, UK: Morgan Kaufmann 1999
- [EN89] Elmasri, R.; Navathe, S.: Fundamentals of Database Systems. Redwood City, California: Benjamin Cummings 1989
- [GJWW03] Galindo-Legaria, C. A.; Joshi, M.; Waas, F.; Wu, M.: Statistics on Views. In: Proceedings of the 29th International Conference on Very Large Data Bases, 09-12 September, 2003, Berlin, Germany: Morgan Kaufmann 2003

- [GUW02] Garcia-Molina, H.; Ullman, J. D.; Widom, J.: Database Systems – The Complete Book. Upper Saddle River New Jersey: Prentice Hall 2002
- [HR01] Härder, T.; Rahm, E.: Datenbanksysteme – Konzepte und Techniken der Implementierung. Berlin u. a. : Springer 2001
- [HS92] Haas, P. J.; Swami, A. N.: Sequential Sampling Procedures for Query Size Estimation. In: Proceedings of the 1992 ACM SIGMOD international conference on Management of data, San Diego, California, USA: ACM Press 1992
- [HSN97] Harangsri, B.; Shepherd, J.; Ngu, A.: Selectivity Estimation for Joins Using Systematic Sampling. In: Proceedings of the 8th International Workshop on Database and Expert Systems Applications, 01-02 September, 1997, Toulouse, France: IEEE Computer Society 1997
- [IBM04] IBM: Online-Dokumentation von IBM's DB2 Universal Database Version 8. <http://publib.boulder.ibm.com/infocenter/db2v8luw/index.jsp>, Zugriff: August 2004
- [IK84] Ibaraki, T.; Kameda, T.: On the Optimal Nesting Order for Computing N-relational Joins. In: ACM Transactions on Database Systems, Vol. 9, No. 3, 482-502, 1984, New York, NY, USA: ACM Press 1984
- [IC91] Ioannidis, Y. E.; Christodoulakis, S.: On the Propagation of Errors in the Size of Join Results. In: Proceedings of the 1991 ACM SIGMOD international conference on Management of data, Denver, Colorado, USA: ACM Press 1991
- [IC93] Ioannidis, Y. E.; Christodoulakis, S.: Optimal Histograms for Limiting Worst-Case Error Propagation in the Size of Join Results. In: ACM Transactions on Database Systems, Vol. 18, No. 4, 709-748, 1993, New York, NY, USA: ACM Press 1993
- [Ioa03] Ioannidis, Y. E.; The History of Histograms (abridged). In: Proceedings of the 29th International Conference on Very Large Data Bases, 09-12 September, 2003, Berlin, Germany: Morgan Kaufmann 2003
- [IP95a] Ioannidis, Y. E.; Poosala, V.: Histogram-Based Solutions to Diverse Database Estimation Problems. In: IEEE Data Engineering Bulletin, Vol. 18, No. 3, 10-18, 1995
- [IP95b] Ioannidis, Y. E.; Poosala, V.: Balancing Histogram Optimality and Practicality for Query Result Size Estimation. In: Proceedings of the 1995 ACM SIGMOD international conference on Management of data, San Jose, California, USA: ACM Press 1995
- [IP99] Ioannidis, Y. E.; Poosala, V.: Histogram-Based Approximation of Set-Valued Query Answers. In: Proceedings of the 25th International Conference on Very Large Data Bases, Edinburgh, Scotland, UK: Morgan Kaufmann 1999

- [JK+98] Jagadish, H. V.; Koudas, N.; Muthukrishnan, S.; Poosala, V.; Sevcik, K.; Suel, T.: Optimal Histograms with Quality Guarantees. In: Proceedings of the 24th International Conference on Very Large Data Bases, New York, NY, USA: Morgan Kaufmann 1998
- [KE01] Kemper, A.; Eickler, A.: Datenbanksysteme – Eine Einführung. München Wien: Oldenbourg 2001
- [Koe01] König, A. C.: Query Estimation Techniques in Database Systems. Universität des Saarlandes Saarbrücken, Naturwissenschaftlich-Technische Fakultät I, Dissertation, Saarbrücken 2001
- [KSRM03] Kraft, T.; Schwarz, H.; Rantza, R.; Mitschang, B.: Coarse-Grained Optimization: Techniques for Rewriting SQL Statement Sequences. In: Proceedings of 29th International Conference on Very Large Data Bases, September 9-12, Berlin: Morgan Kaufmann 2003
- [KW98] Kabra, N.; De Witt, D. J. Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, Seattle, Washington, USA: ACM Press 1998
- [LG03] Lorentz, D.; Gregoire, J.: Oracle Database SQL Reference 10g Release 1, Part No. B10759-01. <https://cwisdb.cc.kuleuven.ac.be/ora10doc/index.htm>, Zugriff: August 2004
- [LKC99] Lee, J.; Kim, D.; Chung, C.: Multi-dimensional Selectivity Estimation Using Compressed Histogram Information. In: Proceedings of the 1999 ACM SIGMOD international conference on Management of data Philadelphia, Pennsylvania, USA: ACM Press 1999
- [LN90] Lipton, R. J.; Naughton, J. F.: Query Size Estimation by Adaptive Sampling (Extended Abstract). In: Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, April 02-04, 1990, Nashville, Tennessee, USA: ACM Press 1990
- [MCS88] Mannino, M. V.; Chu, P.; Sager, T.: Statistical Profile Estimation in Database Systems In: ACM Computing Surveys, Vol. 20, No. 3, 192-221, 1988: ACM Press 1988
- [Mit95] Mitschang, B.: Anfrageverarbeitung in Datenbanksystemen – Entwurf und Implementierungskonzepte. Braunschweig Wiesbaden: Vieweg 1995
- [MLR03] Markl, V.; Lohman, G. M.; Raman, V.: An Autonomic Query Optimizer for DB2 In: IBM Systems Journal, Vol. 42, No. 1, 98-106, 2003
- [MS02] Melton, J.; Simon, A. R.: SQL:1999 – Understanding Relational Language Components. San Diego: Academic Press 2002
- [MVW98] Matias, Y.; Vitter, J. S.; Wang, M.: Wavelet-based Histograms for Selectivity Estimation. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, Seattle, Washington, USA: ACM Press 1998

- [OR01] Oommen, B. J.; Rueda, L. G.: Histogram Methods in Query Optimization: The Relation between Accuracy and Optimality. In: Proceedings of the 7th International Conference on Database Systems for Advanced Applications (DASFAA 2001), Hong-Kong, China: IEEE Computer Society 2001
- [OT99] Oommen, B. J.; Thiyagarajah, M.: Query Result Size Estimation Using a Novel Histogram-like Technique: The Rectangular Attribute Cardinality Map. In: Proceedings of the 1999 International Symposium on Database Engineering & Applications, Washington, DC, USA: IEEE Computer Society 1999
- [OT00] Oommen, B. J.; Thiyagarajah, M.: Query Result Size Estimation Using the Trapezoidal Attribute Cardinality Map In: 2000 International Database Engineering and Applications Symposium (IDEAS'00), Yokohama, Japan: IEEE Computer Society 2000
- [PI97] Poosala, V.; Ioannidis, Y. E.: Selectivity Estimation Without the Attribute Value Independence Assumption. In: Proceedings of the 23rd International Conference on Very Large Data Bases, Athnes, Greece: Morgan Kaufmann 1997
- [PIHS96] Poosala, V.; Ioannidis, Y. E.; Haas, P. J.; Shekita, E. J.: Improved Histograms for Selectivity Estimation of Range Predicates. In: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, Montreal, Quebec, Canada: ACM Press 1996
- [PC84] Piatetsky-Shapiro, G.; Connell, C.: Accurate Estimation of the Number of Tuples Satisfying a Condition. In: Proceedings of the 1984 ACM SIGMOD international conference on Management of data, Boston, Massachusetts, USA: ACM Press 1984
- [SH99] Saake, G.; Heuer, A.: Datenbanken – Entwurf und Implementierungstechniken. Bonn: MITP 1999
- [SI93] Swami, A.; Iyer, B. R.: A Polynomial Time Algorithm for Optimizing Join Queries. In: Proceedings of the 9th International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria: IEEE Computer Society 1993
- [SLMK01] Stillger, A.; Lohman, G.; Markl, V.; Kandil, M.: LEO – DB2's LEarning Optimizer. In: Proceedings of the 27th International Conference on Very Large Data Bases, Roma, Italy: Morgan Kaufmann 2001
- [SLRD93] Sun, W.; Ling, Y.; Rische, N.; Deng, Y.: An Instant and Accurate Size Estimation Method for Joins and Selection in a Retrieval-Intensive Environment. In: Proceedings of the 1993 ACM SIGMOD international conference on Management of data, Washington, D.C., USA: ACM Press 1993
- [SMK97] Steinbrunn, M.; Moerkotte, G.; Kemper, A.: Heuristic and Randomized Optimization for the Join Ordering Problem. In: The VLDB Journal Vol. 6, No. 3, 191-208, 1997, Springer-Verlag Heidelberg, 1997

- [TPCH04] TPC Benchmark H, (Decision Support for Ad Hoc Queries). <http://www.tpc.org>, Zugriff: August 2004
- [Wag00] Wagner, R.: Realisierung und Optimierung einer OLAP-Anwendung im Handelsbereich. Universität Stuttgart, Fakultät Informatik, Studienarbeit Nr. 1770, 2000
- [WG00] Waas, F.; Galindo-Legaria, C.: Counting, Enumerating, and Sampling of Execution Plans in a Cost-Based Query Optimizer. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, Dallas, Texas, USA: ACM Press 2000
- [WKW94] Whang, K.; Kim, S., Wiederhold, G.: Dynamic Maintenance of Data Distribution for Selectivity Estimation. In: The VLDB Journal Vol. 3, No. 1, 29-51, 1994, Secaucus, NJ, USA: Springer-Verlag New York, Inc. 1994

# Anhang A: Anfragen

Für die Messungen wurden drei verschiedene Anfragegruppen verwendet (s. Abschnitt 5.1.2): Anfragen des TPC Benchmark H ( $Q_t$ ), selbst spezifizierte Anfragen ( $Q_e$ ) sowie Anfragen aus dem Projekt ORBIT ( $Q_o$ ). Im Folgenden sind die Anfragegruppen  $Q_e$  und  $Q_o$  aufgelistet. Die TPC-H-Anfragen finden sich auf den Internet-Seiten des Transaction Processing Performance Council [TPCH04].

## Anfrage $Q_{e01}$

```
SELECT l.l_orderkey
FROM   tpch4.lineitem l, tpch4.orders o, tpch4.customer c, tpch4.nation n1, tpch4.nation n2,
       tpch4.region r1, tpch4.region r2, tpch4.partsupp ps, tpch4.part p, tpch4.supplier s
WHERE  l.l_orderkey = o.o_orderkey AND o.o_custkey = c.c_custkey AND
       c.c_nationkey = n1.n_nationkey AND n1.n_regionkey = r1.r_regionkey AND
       l.l_partkey = ps.ps_partkey AND l.l_suppkey = ps.ps_suppkey AND
       ps.ps_partkey = p.p_partkey AND ps.ps_suppkey = s.s_suppkey AND
       s.s_nationkey = n2.n_nationkey AND n2.n_regionkey = r2.r_regionkey;
```

## Anfrage $Q_{e02}$

```
SELECT  n.n_nationkey, r.r_regionkey
FROM    tpch4.customer c, tpch4.nation n, tpch4.region r
WHERE   c.c_nationkey = n.n_nationkey AND n.n_regionkey = r.r_regionkey
GROUP BY n.n_nationkey, r.r_regionkey HAVING COUNT(*) > 610;
```

## Anfrage $Q_{e03}$

```
SELECT l_orderkey
FROM   tpch4.lineitem l, tpch4.orders o
WHERE  l.l_orderkey = o.o_orderkey AND l.l_receiptdate > o.o_orderdate;
```

## Anfrage $Q_{e04}$

```
SELECT o.o_totalprice
FROM   tpch4.orders o
WHERE  o.o_totalprice < (
  SELECT SUM(l_extendedprice)
  FROM   tpch4.lineitem l
  WHERE  l.l_orderkey = o.o_orderkey
  GROUP BY l.l_orderkey
);
```

### Anfrage Q<sub>e</sub>05

```
SELECT l.l_orderkey
FROM   tpch4.lineitem l, tpch4.orders o, tpch4.customer c, tpch4.nation n1, tpch4.nation n2,
      tpch4.region r1, tpch4.region r2, tpch4.partsupp ps, tpch4.part p, tpch4.supplier s
WHERE  l.l_orderkey = o.o_orderkey AND o.o_custkey = c.c_custkey AND
      c.c_nationkey = s.s_nationkey AND c.c_nationkey = n1.n_nationkey AND
      n1.n_regionkey = r1.r_regionkey AND l.l_partkey = ps.ps_partkey AND
      l.l_suppkey = ps.ps_suppkey AND ps.ps_partkey = p.p_partkey AND
      ps.ps_suppkey = s.s_suppkey AND s.s_nationkey = n2.n_nationkey AND
      n2.n_regionkey = r2.r_regionkey;
```

### Anfrage Q<sub>e</sub>06

```
SELECT c.c_custkey
FROM   tpch4.customer c
WHERE  c.c_comment LIKE '%gr%';
```

### Anfrage Q<sub>e</sub>07

```
SELECT s.s_suppkey
FROM   tpch4.supplier s, tpch4.nation n, tpch4.region r, tpch4.partsupp ps
WHERE  s.s_nationkey = n.n_nationkey AND n.n_regionkey = r.r_regionkey AND
      r.r_name = 'EUROPE' AND s.s_suppkey = ps.ps_suppkey AND ps.ps_availqty > 1000;
```

### Anfrage Q<sub>e</sub>08

```
SELECT DISTINCT(l_quantity)
FROM   tpch4.lineitem l;
```

### Anfrage Q<sub>e</sub>09

```
SELECT DISTINCT(p.p_partkey)
FROM   tpch4.part p, tpch4.partsupp ps, tpch4.lineitem l, tpch4.orders o
WHERE  p.p_partkey = ps.ps_partkey AND ps.ps_partkey = l.l_partkey AND
      l.l_orderkey = o.o_orderkey AND o.o_orderdate = '1997-12-14';
```

### Anfrage Q<sub>e</sub>10

```
SELECT *
FROM (
  SELECT o1.o_custkey AS cust1
  FROM tpch4.orders o1
  WHERE o1.o_orderdate > '1996-12-01' AND o1.o_orderdate < '1996-12-31'
  UNION
  SELECT o2.o_custkey AS cust2
  FROM tpch4.orders o2
  WHERE o2.o_orderdate > '1997-09-14' AND o2.o_orderdate < '1997-10-10'
) AS neu;
```

### Anfrage Q<sub>e</sub>11

```
SELECT *
FROM tpch4.orders o, tpch4.lineitem l, tpch4.customer c
WHERE l.l_shipmode = 'asdfaasdfd' AND l.l_orderkey = o.o_orderkey AND
      o.o_comment LIKE '%a' AND o.o_custkey = c.c_custkey AND c.c_acctbal > 1000.1;
```

### Anfrage Q<sub>e</sub>12

```
SELECT *
FROM tpch4.orders o, tpch4.lineitem l, tpch4.customer c
WHERE l.l_shipmode = 'SHIP' AND l.l_orderkey = o.o_orderkey AND o.o_comment LIKE '%a' AND
      o.o_custkey = c.c_custkey AND c.c_acctbal > 1000.1;
```

### Anfrage Q<sub>e</sub>13

```
SELECT *
FROM tpch4.lineitem
WHERE DAY(l_commitdate) = 31;
```

### Anfrage Q<sub>e</sub>14

```
SELECT *
FROM tpch4.lineitem l
WHERE l.l_shipinstruct = 'NONE';
```

### Anfrage Q<sub>e</sub>15

```
SELECT *
FROM tpch4.lineitem l1
WHERE 0.8 * l1.l_quantity > (
  SELECT AVG(l2.l_quantity)
  FROM tpch4.lineitem l2
);
```

### Anfrage Q<sub>e</sub>16

```
SELECT SUM(l_quantity), AVG(l_tax), MIN(l_extendedprice)
FROM tpch4.lineitem l
GROUP BY l.l_shipmode, l.l_linestatus, l.l_discount;
```

### Anfrage Q<sub>e</sub>17

```
SELECT *
FROM tpch4.lineitem l
WHERE l.l_shipdate IN ('1996-11-11', '1997-12-12', '1998-12-12') OR
      l.l_commitdate IN (
  SELECT o.o_orderdate
  FROM tpch4.orders o
  WHERE o.o_orderkey = l.l_orderkey
);
```

### Anfrage Q<sub>e</sub>18

```
SELECT l.l_orderkey
FROM   tpch4.lineitem l
GROUP BY l.l_orderkey HAVING COUNT(*) > 6;
```

### Anfrage Q<sub>e</sub>19

```
SELECT p.p_brand, p.p_type, s.s_suppkey
FROM   tpch4.part p, tpch4.partsupp ps, tpch4.supplier s
WHERE  p.p_partkey = ps.ps_partkey AND ps.ps_suppkey = s.s_suppkey
GROUP BY p.p_brand, p.p_type, s.s_suppkey HAVING MAX(p.p_retailprice) < 100;
```

### Anfrage Q<sub>e</sub>20

```
SELECT c.c_phone
FROM   tpch4.customer c
WHERE  EXISTS (
    SELECT *
    FROM   tpch4.supplier s
    WHERE  SUBSTR(s.s_phone,1,6) = SUBSTR(c.c_phone,1,6)
);
```

### Anfrage Q<sub>e</sub>21

```
SELECT *
FROM   tpch4.lineitem
WHERE  MOD(MONTH(l_shipdate), 131) > 13;
```

### Anfrage Q<sub>e</sub>22

```
SELECT DISTINCT(ROUND(c_acctbal,1))
FROM   tpch4.customer
WHERE  c_acctbal = ROUND(c_acctbal,1);
```

### Anfrage Q<sub>e</sub>23

```
SELECT (DAYS(l_shipdate) - DAYS(o_orderdate))
FROM   tpch4.lineitem l, tpch4.orders o
WHERE  l.l_orderkey = o.o_orderkey AND (DAYS(l_shipdate) - DAYS(o_orderdate)) < 1
ORDER BY (DAYS(l_shipdate) - DAYS(o_orderdate));
```

### Anfrage Q<sub>e</sub>24

```
SELECT (DAYS(l_shipdate) - DAYS(o_orderdate))
FROM   tpch4.lineitem l, tpch4.orders o
WHERE  l.l_orderkey = o.o_orderkey AND (DAYS(l_shipdate) - DAYS(o_orderdate)) < 10
ORDER BY (DAYS(l_shipdate) - DAYS(o_orderdate))
FETCH FIRST 1000 ROWS ONLY;
```

### Anfrage Q<sub>e</sub>25

```
SELECT *
FROM   tpch4.lineitem l LEFT JOIN tpch4.orders o ON l.l_comment = o.o_comment
WHERE  o.o_comment IS NULL;
```

### Anfrage Q<sub>e</sub>26

```
SELECT ps_suppkey
FROM   tpch2.partsupp
WHERE  ps_availqty > (
      SELECT SUM(l_quantity)
      FROM   tpch2.lineitem
      );
```

### Anfrage Q<sub>o</sub>a1

```
SELECT a3.ordermonthkey, a3.ordermonthname, a4.orderyearkey, a4.orderyear, a5.partkey,
       a5.partname, a1.sumquantity, a2.sumquantity,
       (CASE WHEN a1.sumquantity IS NULL THEN 0 ELSE a1.sumquantity END) -
       (CASE WHEN a2.sumquantity IS NULL THEN 0 ELSE a2.sumquantity END)
       AS incrquantity,
       ((CASE WHEN a1.sumquantity IS NULL THEN 0 ELSE a1.sumquantity END) -
        (CASE WHEN a2.sumquantity IS NULL THEN 0 ELSE a2.sumquantity END)) /
       (CASE a2.sumquantity WHEN 0 THEN NULL ELSE a2.sumquantity END)
       AS incrquantity2
FROM   (SELECT a2.orderyearkey, a2.ordermonthkey, a1.partkey, SUM(a1.quantity)
      FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
      WHERE  a2.orderdate = a1.orderdate AND a2.ordermonthkey IN (199401, 199402)
      GROUP BY a2.orderyearkey, a2.ordermonthkey, a1.partkey
      ) AS a1 (orderyearkey, ordermonthkey, partkey, sumquantity),
       (SELECT a2.ordermonthkey, a1.partkey, SUM(a1.quantity)
      FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
      WHERE  a2.lastmonthdate = a1.orderdate AND a2.ordermonthkey IN (199401, 199402)
      GROUP BY a2.ordermonthkey, a1.partkey
      ) a2 (ordermonthkey, partkey, sumquantity),
       tpch4.lookup_ordermonth a3, tpch4.lookup_orderyear a4, tpch4.lookup_part a5
WHERE  a1.ordermonthkey = a2.ordermonthkey AND a1.partkey = a2.partkey AND
       a1.ordermonthkey = a3.ordermonthkey AND a1.orderyearkey = a4.orderyearkey AND
       a1.partkey = a5.partkey AND
       ((CASE WHEN a1.sumquantity IS NULL THEN 0 ELSE a1.sumquantity END) -
        (CASE WHEN a2.sumquantity IS NULL THEN 0 ELSE a2.sumquantity END)) /
       (CASE a2.sumquantity WHEN 0 THEN NULL ELSE a2.sumquantity END) >= 9.800000000000000e+01;
```

### Anfrage Q<sub>o</sub>a1\_seq1

```
CREATE TABLE temptableA010 (
  orderyearkey   INTEGER,
  ordermonthkey  INTEGER,
  partkey        INTEGER,
  sumquantity    DECIMAL(10, 2)
) IN data1;

INSERT INTO temptableA010
SELECT a2.orderyearkey, a2.ordermonthkey, a1.partkey, SUM(a1.quantity)
FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
WHERE  a2.orderdate = a1.orderdate AND a2.ordermonthkey IN (199401, 199402)
GROUP BY a2.orderyearkey, a2.ordermonthkey, a1.partkey;
```

### Anfrage Q<sub>o</sub>a1\_seq2

```
CREATE TABLE temptableA020 (
  ordermonthkey  INTEGER,
  partkey        INTEGER,
  sumquantity    DECIMAL(10, 2)
) IN data1;

INSERT INTO temptableA020
  SELECT  a2.ordermonthkey, a1.partkey, SUM(a1.quantity)
  FROM    tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
  WHERE   a2.lastmonthdate = a1.orderdate AND a2.ordermonthkey IN (199401, 199402)
  GROUP BY a2.ordermonthkey, a1.partkey;
```

### Anfrage Q<sub>o</sub>a1\_seq3

```
CREATE TABLE temptableA030 (
  ordermonthkey  INTEGER,
  ordermonthname VARCHAR(10),
  orderyearkey   INTEGER,
  orderyear      INTEGER,
  partkey        INTEGER,
  partname       VARCHAR(55),
  sumquantity    DECIMAL(10, 2),
  lmsumquantity  DECIMAL(10, 2),
  incrquantity   FLOAT,
  incrquantity2  FLOAT
) IN data1;

INSERT INTO temptableA030
  SELECT a3.ordermonthkey, a3.ordermonthname, a4.orderyearkey, a4.orderyear,
  a5.partkey, a5.partname, a1.sumquantity, a2.sumquantity,
  (CASE WHEN a1.sumquantity IS NULL THEN 0 ELSE a1.sumquantity END) -
  (CASE WHEN a2.sumquantity IS NULL THEN 0 ELSE a2.sumquantity END),
  ((CASE WHEN a1.sumquantity IS NULL THEN 0 ELSE a1.sumquantity END) -
  (CASE WHEN a2.sumquantity IS NULL THEN 0 ELSE a2.sumquantity END)) /
  (CASE a2.sumquantity WHEN 0 THEN NULL ELSE a2.sumquantity END)
FROM temptableA010 a1,
  temptableA020 a2,
  tpch4.lookup_ordermonth a3,
  tpch4.lookup_orderyear a4,
  tpch4.lookup_part a5
WHERE a1.ordermonthkey = a2.ordermonthkey
  AND a1.partkey = a2.partkey
  AND a1.ordermonthkey = a3.ordermonthkey
  AND a1.orderyearkey = a4.orderyearkey
  AND a1.partkey = a5.partkey;
```

### Anfrage Q<sub>o</sub>a1\_seq4

```
SELECT a1.ordermonthkey, a1.ordermonthname, a1.orderyearkey, a1.orderyear, a1.partkey,
  a1.partname, a1.sumquantity, a1.lmsumquantity, a1.incrquantity, a1.incrquantity2
FROM temptableA030 a1
WHERE a1.incrquantity2 >= 9.800000000000000e+01;
```

### Anfrage Q<sub>o</sub>b1

```

SELECT  a4.partkey, MAX(a4.partname), a5.custregionkey, MAX(a5.custregionname),
        a6.custnationkey, MAX(a6.custnationname), a7.orderyearkey, MAX(a7.orderyear),
        SUM(a1.quantity)
FROM    tpch4.lineitem_orders a1, tpch4.lookup_customer a2, tpch4.lookup_orderday a3,
        tpch4.lookup_part a4, tpch4.lookup_custregion a5, tpch4.lookup_custnation a6,
        tpch4.lookup_orderyear a7
WHERE   a2.custkey = a1.custkey AND a3.orderdate = a1.orderdate AND
        a1.partkey = a4.partkey AND a2.custregionkey = a5.custregionkey AND
        a2.custnationkey = a6.custnationkey AND a3.orderyearkey = a7.orderyearkey AND
        a4.partkey <= 3 AND a7.orderyearkey = 1994
GROUP BY a4.partkey, a5.custregionkey, a6.custnationkey, a7.orderyearkey;

```

### Anfrage Q<sub>o</sub>c1

```

SELECT  a6.custkey, a6.custname, a4.stdeviation, a5.stdeviation,
        a1.turnover1992, a2.turnover1993, a3.turnover1994
FROM    (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
        GROUP BY a1.custkey
        ) AS a1 (custkey, turnover1992),
        (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993
        GROUP BY a1.custkey
        ) AS a2 (custkey, turnover1993),
        (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994
        GROUP BY a1.custkey
        ) AS a3 (custkey, turnover1994),
        (SELECT  a1.custkey, STDDEV(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2,
        (SELECT  a1.custkey, a1.turnover1992, a2.turnover1993, a3.turnover1994
        FROM    (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
        GROUP BY a1.custkey
        ) AS a1 (custkey, turnover1992),
        (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993
        GROUP BY a1.custkey
        ) AS a2 (custkey, turnover1993),
        (SELECT  a1.custkey, SUM(a1.endprice)
        FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
        WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994
        GROUP BY a1.custkey
        ) AS a3 (custkey, turnover1994)
        WHERE a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND
        a1.turnover1992 >= 500000 AND a2.turnover1993 >= 500000 AND
        a3.turnover1994 >= 500000
        ) AS a3 (custkey, turnover1992, turnover1993, turnover1994)

```

```

WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994) AND
a1.custkey = a3.custkey
GROUP BY a1.custkey
) AS a4 (custkey, stddeviation),
(SELECT a1.custkey, STDDEV(a1.endprice) /
(CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)
FROM tpch2.lineitem_orders a1, tpch2.lookup_orderday a2,
(SELECT a1.custkey, a1.turnover1992, a2.turnover1993, a3.turnover1994
FROM (SELECT a1.custkey, SUM(a1.endprice)
FROM tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
GROUP BY a1.custkey
) AS a1 (custkey, turnover1992),
(SELECT a1.custkey, SUM(a1.endprice)
FROM tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993
GROUP BY a1.custkey
) AS a2 (custkey, turnover1993),
(SELECT a1.custkey, SUM(a1.endprice)
FROM tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994
GROUP BY a1.custkey
) AS a3 (custkey, turnover1994)
WHERE a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND
a1.turnover1992 >= 500000 AND a2.turnover1993 >= 500000 AND
a3.turnover1994 >= 500000
) AS a3 (custkey, turnover1992, turnover1993, turnover1994)
WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994) AND
a1.custkey = a3.custkey
GROUP BY a1.custkey
) AS a5 (custkey, stddeviation),
tpch2.lookup_customer a6
WHERE a4.custkey = a1.custkey AND a4.custkey = a2.custkey AND a4.custkey = a3.custkey AND
a4.custkey = a5.custkey AND a4.custkey = a6.custkey AND
a5.stddeviation <= 0.66794004454646;

```

### Anfrage Q<sub>o</sub>c1\_seq1

```

CREATE TABLE temptableC010 (
  custkey      INTEGER,
  turnover1992  DECIMAL(31, 6)
) IN data1;

INSERT INTO temptableC010
SELECT a1.custkey, SUM(a1.endprice)
FROM tpch2.lineitem_orders a1, tpch2.lookup_orderday a2
WHERE a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
GROUP BY a1.custkey;

```

### Anfrage Q<sub>oc1</sub>\_seq2

```
CREATE TABLE temptableC020 (  
  custkey          INTEGER,  
  turnover1993    DECIMAL(31, 6)  
) IN data1;
```

```
INSERT INTO temptableC020  
  SELECT  a1.custkey, SUM(a1.endprice)  
  FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>oc1</sub>\_seq3

```
CREATE TABLE temptableC030 (  
  custkey          INTEGER,  
  turnover1994    DECIMAL(31, 6)  
) IN data1;
```

```
INSERT INTO temptableC030  
  SELECT  a1.custkey, SUM(a1.endprice)  
  FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>oc1</sub>\_seq4

```
CREATE TABLE temptableC040 (  
  custkey          INTEGER,  
  turnover1992    FLOAT,  
  turnover1993    FLOAT,  
  turnover1994    FLOAT  
) IN data1;
```

```
INSERT INTO temptableC040  
  SELECT a1.custkey, a1.turnover1992, a2.turnover1993, a3.turnover1994  
  FROM   temptableC010 a1, temptableC020 a2, temptableC030 a3  
  WHERE  a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND  
         a1.turnover1992 >= 500000 AND a2.turnover1993 >= 500000 AND a3.turnover1994 >= 500000;
```

### Anfrage Q<sub>oc1</sub>\_seq5

```
CREATE TABLE temptableC050 (  
  custkey          INTEGER,  
  stddeviation    DECIMAL(31, 6)  
) IN data1;
```

```
INSERT INTO temptableC050  
  SELECT  a1.custkey, STDDEV(a1.endprice)  
  FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2, temptableC040 a3  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994) AND  
         a1.custkey = a3.custkey  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>o</sub>c1\_seq6

```
CREATE TABLE temptableC060 (  
  custkey      INTEGER,  
  stddeviation  FLOAT  
) IN data1;  
  
INSERT INTO temptableC060  
  SELECT  a1.custkey, STDDEV(a1.endprice) /  
          (CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)  
  FROM    tpch2.lineitem_orders a1, tpch2.lookup_orderday a2, temptableC040 a3  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994) AND  
          a1.custkey = a3.custkey  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>o</sub>c1\_seq7

```
CREATE TABLE temptableC070 (  
  custkey      INTEGER,  
  custname     VARCHAR(25),  
  stddev1     DECIMAL(10, 2),  
  stddev2     FLOAT,  
  turnover1992  FLOAT,  
  turnover1993  FLOAT,  
  turnover1994  FLOAT  
) IN data1;  
  
INSERT INTO temptableC070  
  SELECT a6.custkey, a6.custname, a4.stddeviation, a5.stddeviation,  
         a1.turnover1992, a2.turnover1993, a3.turnover1994  
  FROM   temptableC010 a1, temptableC020 a2, temptableC030 a3, temptableC050 a4,  
         temptableC060 a5, tpch2.lookup_customer a6  
  WHERE  a4.custkey = a1.custkey AND a4.custkey = a2.custkey AND  
         a4.custkey = a3.custkey AND a4.custkey = a5.custkey AND a4.custkey = a6.custkey;
```

### Anfrage Q<sub>o</sub>c1\_seq8

```
SELECT a1.custkey, a1.custname, a1.stddev1, a1.stddev2,  
       a1.turnover1992, a1.turnover1993, a1.turnover1994  
FROM   temptableC070 a1  
WHERE  a1.stddev2 <= 0.66794004454646;
```

### Anfrage Q<sub>oc2</sub>

```

SELECT a1.custkey, a3.custname, a2.stddev1, a2.stddev2,
       a1.turnover1992, a1.turnover1993, a1.turnover1994
FROM   (SELECT a1.custkey, a1.turnover1992, a2.turnover1993, a3.turnover1994
        FROM (SELECT a1.custkey, SUM(a1.endprice)
              FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
              WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
              GROUP BY a1.custkey) a1(custkey, turnover1992),
          (SELECT a1.custkey, SUM(a1.endprice)
              FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
              WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993
              GROUP BY a1.custkey) a2(custkey, turnover1993),
          (SELECT a1.custkey, SUM(a1.endprice)
              FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
              WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994
              GROUP BY a1.custkey) a3 (custkey, turnover1994)
        WHERE a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND
              a1.turnover1992 >= 500000 AND a2.turnover1993 >=500000 AND
              a3.turnover1994 >= 500000
        ) a1 (custkey, turnover1992, turnover1993, turnover1994),
       (SELECT a1.custkey, STDDEV(a1.endprice), STDDEV(a1.endprice) /
              (CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)
        FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
        WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994)
        GROUP BY a1.custkey
        ) a2 (custkey, stddev1, stddev2),
       tpch4.lookup_customer a3
WHERE a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND a2.stddev2 <= 0.66794004454646;

```

### Anfrage Q<sub>oc2\_seq1</sub>

```

CREATE TABLE temptableC010 (
  custkey      INTEGER,
  turnover1992 DECIMAL(31, 6)
) IN data1;

INSERT INTO temptableC010
SELECT a1.custkey, SUM(a1.endprice)
FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey = 1992
GROUP BY a1.custkey;

```

### Anfrage Q<sub>oc2\_seq2</sub>

```

CREATE TABLE temptableC020 (
  custkey      INTEGER,
  turnover1993 DECIMAL(31, 6)
) IN data1;

INSERT INTO temptableC020
SELECT a1.custkey, SUM(a1.endprice)
FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2
WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey = 1993
GROUP BY a1.custkey;

```

### Anfrage Q<sub>oc2</sub>\_seq3

```
CREATE TABLE temptableC030 (  
  custkey      INTEGER,  
  turnover1994 DECIMAL(31, 6)  
) IN data1;
```

```
INSERT INTO temptableC030  
  SELECT  a1.custkey, SUM(a1.endprice)  
  FROM    tpch4.lineitem_orders a1, tpch4.lookup_orderday a2  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey = 1994  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>oc2</sub>\_seq4

```
CREATE TABLE temptableC040 (  
  custkey      INTEGER,  
  turnover1992  FLOAT,  
  turnover1993  FLOAT,  
  turnover1994  FLOAT  
) IN data1;
```

```
INSERT INTO temptableC040  
  SELECT a1.custkey, a1.turnover1992, a2.turnover1993, a3.turnover1994  
  FROM   temptableC010 a1, temptableC020 a2, temptableC030 a3  
  WHERE  a1.custkey = a2.custkey AND a1.custkey = a3.custkey AND a1.turnover1992 >= 500000 AND  
        a2.turnover1993 >=500000 AND a3.turnover1994 >= 500000 ;
```

### Anfrage Q<sub>oc2</sub>\_seq5

```
CREATE TABLE temptableC050 (  
  custkey      INTEGER,  
  stddev1      DECIMAL(31, 6),  
  stddev2      FLOAT  
) IN data1;
```

```
INSERT INTO temptableC050  
  SELECT a1.custkey, STDDEV(a1.endprice), STDDEV(a1.endprice) /  
        (CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)  
  FROM   tpch4.lineitem_orders a1, tpch4.lookup_orderday a2  
  WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994)  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>oc2</sub>\_seq6

```
CREATE TABLE temptableC070 (  
  custkey      INTEGER,  
  custname     VARCHAR(25),  
  stddev1     DECIMAL(10, 2),  
  stddev2     FLOAT,  
  turnover1992  FLOAT,  
  turnover1993  FLOAT,  
  turnover1994  FLOAT  
) IN data1;  
  
INSERT INTO temptableC070  
  SELECT a1.custkey, a3.custname, a2.stddev1, a2.stddev2,  
         a1.turnover1992, a1.turnover1993, a1.turnover1994  
  FROM   temptableC040 a1, temptableC050 a2, tpch4.lookup_customer a3  
  WHERE  a1.custkey = a2.custkey AND a1.custkey = a3.custkey;
```

### Anfrage Q<sub>oc2</sub>\_seq7

```
SELECT a1.custkey, a1.custname, a1.stddev1, a1.stddev2,  
       a1.turnover1992, a1.turnover1993, a1.turnover1994  
FROM   temptableC070 a1  
WHERE  a1.stddev2 <= 0.66794004454646;
```

Anfrage Q<sub>oc3</sub>

```

SELECT a1.custkey, a1.custname, a1.stddev1, a1.stddev2,
       a1.turnover1992, a1.turnover1993, a1.turnover1994
FROM   (SELECT a1.custkey, a3.custname, a2.stddev1, a2.stddev2,
              a1.turnover1992, a1.turnover1993, a1.turnover1994
        FROM   (SELECT a4.custkey, a1.turnover, a2.turnover, a3.turnover
                FROM   (SELECT a1.custkey, a2.orderyearkey, SUM(a1.endprice)
                        FROM   tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
                        WHERE  a2.orderdate = a1.orderdate AND
                              a2.orderyearkey IN (1992, 1993, 1994)
                        GROUP BY a1.custkey, a2.orderyearkey HAVING SUM(a1.endprice) >= 500000
                        ) AS a1(custkey,year,turnover),
                (SELECT a1.custkey, a2.orderyearkey, SUM(a1.endprice)
                        FROM   tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
                        WHERE  a2.orderdate = a1.orderdate AND
                              a2.orderyearkey IN (1992, 1993, 1994)
                        GROUP BY a1.custkey, a2.orderyearkey HAVING SUM(a1.endprice) >= 500000
                        ) AS a2(custkey,year,turnover),
                (SELECT a1.custkey, a2.orderyearkey, SUM(a1.endprice)
                        FROM   tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
                        WHERE  a2.orderdate = a1.orderdate AND
                              a2.orderyearkey IN (1992, 1993, 1994)
                        GROUP BY a1.custkey, a2.orderyearkey HAVING SUM(a1.endprice) >= 500000
                        ) AS a3(custkey,year,turnover),
                (SELECT a1.custkey
                        FROM   (SELECT a1.custkey, a2.orderyearkey, SUM(a1.endprice)
                                FROM   tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
                                WHERE  a2.orderdate = a1.orderdate AND
                                      a2.orderyearkey IN (1992, 1993, 1994)
                                GROUP BY a1.custkey, a2.orderyearkey
                                HAVING  SUM(a1.endprice) >= 500000
                                ) AS a1(custkey,year,turnover)
                        GROUP BY a1.custkey HAVING COUNT(*) = 3
                        ) AS a4(custkey)
        WHERE a1.custkey = a2.custkey AND a2.custkey = a3.custkey AND
              a3.custkey = a4.custkey AND a1.year = 1992 AND a2.year = 1993 AND
              a3.year = 1994
        ) AS a1(custkey,turnover1992,turnover1993,turnover1994),
        (SELECT a1.custkey, STDDEV(a1.endprice), STDDEV(a1.endprice) /
              (CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)
        FROM   tpch3.lineitem_orders a1, tpch3.lookup_orderday a2
        WHERE  a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994)
        GROUP BY a1.custkey
        ) AS a2(custkey,stddev1,stddev2), tpch3.lookup_customer a3
WHERE  a1.custkey = a2.custkey AND a1.custkey = a3.custkey
) AS a1(custkey,custname,stddev1,stddev2,turnover1992,turnover1993,turnover1994)
WHERE  a1.stddev2 <= 0.66794004454646;

```

### Anfrage Q<sub>oc3</sub>\_seq1

```
CREATE TABLE temptableC010 (  
  custkey      INTEGER,  
  year         INTEGER,  
  turnover     DECIMAL(31, 6)  
) IN data1;  
  
INSERT INTO temptableC010  
  SELECT  a1.custkey, a2.orderyearkey, SUM(a1.endprice)  
  FROM    tpch3.lineitem_orders a1, tpch3.lookup_orderday a2  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994)  
  GROUP BY a1.custkey, a2.orderyearkey HAVING SUM(a1.endprice) >= 500000;
```

### Anfrage Q<sub>oc3</sub>\_seq2

```
CREATE TABLE temptableC020 (  
  custkey      INTEGER  
) IN data1;  
  
INSERT INTO temptableC020  
  SELECT  a1.custkey  
  FROM    temptableC010 a1  
  GROUP BY a1.custkey HAVING COUNT(*) = 3;
```

### Anfrage Q<sub>oc3</sub>\_seq3

```
CREATE TABLE temptableC040 (  
  custkey      INTEGER,  
  turnover1992  FLOAT,  
  turnover1993  FLOAT,  
  turnover1994  FLOAT  
) IN data1;  
  
INSERT INTO temptableC040  
  SELECT a4.custkey, a1.turnover, a2.turnover, a3.turnover  
  FROM   temptableC010 a1, temptableC010 a2, temptableC010 a3, temptableC020 a4  
  WHERE  a1.custkey = a2.custkey AND a2.custkey = a3.custkey AND a3.custkey = a4.custkey AND  
         a1.year = 1992 AND a2.year = 1993 AND a3.year = 1994;
```

### Anfrage Q<sub>oc3</sub>\_seq4

```
CREATE TABLE temptableC050 (  
  custkey      INTEGER,  
  stddev1      DECIMAL(31, 6),  
  stddev2      FLOAT  
) IN data1;  
  
INSERT INTO temptableC050  
  SELECT  a1.custkey, STDDEV(a1.endprice), STDDEV(a1.endprice) /  
         (CASE AVG(a1.endprice) WHEN 0 THEN NULL ELSE AVG(a1.endprice) END)  
  FROM    tpch3.lineitem_orders a1, tpch3.lookup_orderday a2  
  WHERE   a2.orderdate = a1.orderdate AND a2.orderyearkey IN (1992, 1993, 1994)  
  GROUP BY a1.custkey;
```

### Anfrage Q<sub>oc3</sub>\_seq5

```
CREATE TABLE temptableC070 (  
  custkey      INTEGER,  
  custname     VARCHAR(25),  
  stddev1     DECIMAL(10, 2),  
  stddev2     FLOAT,  
  turnover1992  FLOAT,  
  turnover1993  FLOAT,  
  turnover1994  FLOAT  
) IN data1;  
  
INSERT INTO temptableC070  
  SELECT a1.custkey, a3.custname, a2.stddev1, a2.stddev2,  
         a1.turnover1992, a1.turnover1993, a1.turnover1994  
  FROM   temptableC040 a1, temptableC050 a2, tpch3.lookup_customer a3  
  WHERE  a1.custkey = a2.custkey AND a1.custkey = a3.custkey;
```

### Anfrage Q<sub>oc3</sub>\_seq6

```
SELECT a1.custkey, a1.custname, a1.stddev1, a1.stddev2,  
       a1.turnover1992, a1.turnover1993, a1.turnover1994  
FROM   temptableC070 a1  
WHERE  a1.stddev2 <= 0.66794004454646;
```

## **Erklärung**

Hiermit versichere ich, diese Arbeit selbstständig verfasst und nur die angegebenen Quellen benutzt zu haben.

---

(Thilo Marquardt)