

12. GI-Fachtagung für Datenbanksysteme in
Business, Technologie und Web (BTW 2007)
5. bis 9. März 2007 - Aachen, Germany
<http://www.btw2007.de/>

Kosten und Nutzen von Datenbankreorganisationen: Grundlagen, Modelle, Leistungsuntersuchungen

Stefan Dorendorf
Berufsakademie Gera /
Friedrich-Schiller-Universität Jena – Institut für Informatik
Lehrstuhl für Datenbanken und Informationssysteme
Ernst-Abbe-Platz 1-4
07743 Jena
Stefan.Dorendorf@informatik.uni-jena.de

Abstract: Stetig wachsende Datenmengen und hohe Verfügbarkeitsanforderungen an Datenbanken führen dazu, dass Wartungsarbeiten (wie z.B. Datenbankreorganisationen) oft nicht mehr (oder nicht mehr vollständig) offline in zur Verfügung stehenden Zeitfenstern durchgeführt werden können bzw. dass die Kosten oder die Behinderungen des normalen Datenbankbetriebs bei online durchgeführten Wartungsarbeiten nicht mehr vernachlässigt werden können. Es ist daher wichtig, die Datenbankobjekte zu lokalisieren und einzugrenzen, bei denen ein hoher Wartungsbedarf besteht. Weiterhin ist es wünschenswert, zunächst den durch Wartungsmaßnahmen erreichbaren Nutzen vor ihrer Durchführung quantifizieren zu können. Dieser Nutzen ist insbesondere von der Workload abhängig, also von den gegen die Datenbankobjekte gerichteten Anweisungen und deren Ausführungshäufigkeiten. Der zur Workload-Abarbeitung anfallende I/O-Aufwand, der einen dominierenden Anteil am Gesamtaufwand ausmacht, lässt sich mit einer Datenbankreorganisation u.U. wesentlich beeinflussen (reduzieren).

In diesem Beitrag wird eine Methode vorgestellt, die es ermöglicht, die Auswirkungen von Datenbankreorganisationen auf den zur Workload-Abarbeitung notwendigen I/O-Aufwand abzuschätzen und damit den hauptsächlichen Nutzen der Reorganisationsmaßnahmen zu quantifizieren. Die von der Reorganisationsdurchführung verursachten Kosten sollten ebenfalls vorab in die Entscheidungsfindung einbezogen und dem Nutzen gegenübergestellt werden. Wie solche Kostenabschätzungen durchgeführt werden können, wird am Beispiel von In-Place-Reorganisationen gezeigt. Weiterhin werden die Ergebnisse von in einer Beispielumgebung angestellten Messungen präsentiert.

1 Einleitung

Die Herausforderungen, die sich beim Betrieb von größeren datenbankgestützten Anwendungssystemen ergeben, sind in den vergangenen Jahren trotz immer höherer Leistungsfähigkeit der den Systemen zu Grunde liegenden Hardware nicht geringer geworden – im Gegenteil. Die Betreiber sehen sich mit ständig steigenden Datenmengen, mit teilweise stark angestiegenen Benutzerzahlen und damit mit einem stark gestiegenen Transaktionsvolumen sowie hohen Verfügbarkeitsanforderungen konfrontiert.

Durch neue Anwendungen ist oftmals die Zahl der Transaktionen stark gestiegen, die gegen die zu Grunde liegenden Datenbanken ausgeführt werden. Im Bankbereich z.B. steigt die Zahl anfallender Buchungen durch die erhebliche Zunahme des bargeldlosen Zahlungsverkehrs [Joc05]. Die Zugangsmöglichkeit über das Internet und Automaten hat die Zahl der Banktransaktionen (inkl. Kontoabfragen etc.) ebenfalls sehr stark ansteigen lassen. Die Transaktionszahlen der Volks- und Raiffeisenbanken in Deutschland stiegen allein im Geldkarten-System im Jahr 2002 um 28% [BVR03]. Auf dem Gebiet der Telekommunikation ist die Zahl der Mobilfunknutzer in den letzten Jahren enorm angestiegen. Die Vermittlung der Gespräche und das Sammeln der Verbindungsdaten erfolgen hier über datenbankgestützte Softwarelösungen.

Große Logistikunternehmen betreiben u.a. Systeme zur Sendungsverfolgung, die jetzt Informationen in zentralen Datenbanken speichern, die an Sortier- und Weiterleitungsstellen anfallen und die früher nur dort lokal verwendet und nach deren „Gebrauch“ gelöscht wurden. Die Deutsche Post World Net betreibt im Unternehmensbereich BRIEF ein System zur Sammlung und Analyse produktionsbezogener Daten mit ca. 1000 Tabellen, einer Datenbankgröße von über 5 Terabyte und täglichen Datenbewegungen von ca. 4,5 Gigabyte [DPW05]. Im Bereich großer integrierter betrieblicher Informationssysteme (z.B. SAP-Systeme) sind Datenbanken in der Größenordnung von mehreren hundert Gigabyte und mehr sowie mit vielen tausend Nutzern keine Seltenheit.

Mit steigender Nutzerzahl und damit wachsendem Transaktionsvolumen steigt bei OLTP-Systemen i.d.R. auch der Wartungsbedarf der Datenbanken. Einfüge-, Löscho- und Änderungsoperationen führen u.U. zu Degenerierungen in den zur Speicherung verwendeten internen physischen Strukturen. Bei der oft üblichen Ablage von Daten in als Heap organisierten Bereichen werden bei Einfügeoperationen evtl. wünschenswerte interne Sortierreihenfolgen i.d.R. nicht beachtet, Speicherbereiche für logisch zusammengehörige Daten müssen über Datenträger verstreut reserviert werden etc. Bei Löschooperationen (z.B. beim Verschieben von Daten in Archive) entstehen oft weitere Lücken im Datenbestand, weil der frei werdende Speicher nicht ohne größeren Aufwand sofort wiederverwendet werden kann. Solche *Degenerierungen* führen also zu einem erhöhten Speicher- und Verarbeitungsaufwand und damit zu höheren Kosten (allg. Leistungseinbußen). Damit kommt es im Laufe der Zeit zu einer schleichenden Leistungsver schlechterung. Eine Beseitigung vorhandener Degenerierungen kann und muss durch *Datenbankreorganisationen* erfolgen.

Hohe Verfügbarkeitsanforderungen lassen die für Wartungsarbeiten zur Verfügung stehenden Zeitfenster, innerhalb derer zumindest Teile der Datenbanken nicht verfügbar sind, schrumpfen oder nahezu vollständig verschwinden. Durch Bemühungen der Hersteller von Datenbank-Management-Systemen (DBMS) und von Drittanbietern wurden deshalb Mechanismen zur Durchführung von Wartungsarbeiten an Datenbanken (z.B. Backup-Erstellung, Archivierung von Daten, Konfiguration von DBMS, Datenbankreorganisation) zur Verfügung gestellt, die online und parallel zum laufenden Betrieb ausgeführt werden können. Trotzdem sind solche Wartungen mit Einschränkungen des normalen Datenbankbetriebs verbunden, denn der Wartungsaufwand bleibt erhalten. Hinzu kommt der Aufwand zur Aufrechterhaltung der nahezu vollständigen Verfügbarkeit der Daten. (Ähnliche Ansätze – online versus offline – sind aus dem Bereich der Backup- und Recovery-Durchführung bekannt). Hier ist eine

wohlüberlegte Planung zwingend erforderlich. Wartungsarbeiten, die einen entsprechenden Nutzen erwarten lassen oder aus unterschiedlichen Gründen als „dringend“ eingestuft werden, sind durchzuführen, unnötige oder noch nicht nötige Arbeiten müssen vermieden bzw. zurückgestellt werden. Dadurch besteht die Notwendigkeit, den zu erwartenden Nutzen der Wartungsarbeiten vorab möglichst genau einzuschätzen. Dazu ist es wichtig, das Ausmaß vorhandener Degenerierungen zu kennen und die daraus resultierenden Auswirkungen quantifizieren zu können.

In der Praxis sind diese Probleme nicht neu und Datenbankadministratoren sind gezwungen (oft pragmatisch), Lösungen zu finden, bei denen die Beeinflussung des normalen Datenbankbetriebs während einer Reorganisation möglichst gering ist. Auf Grund

- immer größer werdender Datenbanken auf der einen Seite und hoher Verfügbarkeitsanforderungen auf der anderen Seite,
- größerer Freiheitsgrade beim physischen Datenbankentwurf durch neue Speicherkonzepte (objektrelational, XML usw.) bzw. durch neue Kombinationsmöglichkeiten und
- der Tatsache, dass Wartungsarbeiten wegen des damit verbundenen teilweise hohen Aufwands nicht einfach „auf Verdacht“ (online oder offline) ausgeführt werden können [Sch04]

ergibt sich eine Verschärfung der Problematik. Vorhandene Problemlösungen in Praxis und Wissenschaft sind nach wie vor nicht immer überzeugend. Es erscheint daher opportun, dieses Thema weiter aufzugreifen. Dies zeigt sich auch an den wachsenden Anstrengungen der Anbieter von DBMS-Produkten zur Verbesserung von Werkzeugen zur Datenbankadministration [AF06] und dem Trend der letzten Jahre zur verstärkten Entwicklung und Verbesserung von Self-Managing Database Systems [Ruf05].

Eine Einordnung des Beitrags und einen kurzen Überblick über relevante Literatur bietet *Kapitel 2*. *Kapitel 3* befasst sich zunächst allgemein mit der Thematik Datenbankreorganisation, bevor in *Kapitel 4* die Methoden zur Bestimmung von Nutzen und Kosten solcher Reorganisationen beschrieben werden. Weiterhin wird eine Möglichkeit zur gezielten Auswahl von Reorganisationskandidaten vorgestellt. Die Ergebnisse von in einer Beispielumgebung angestellten Messungen werden in *Kapitel 5* präsentiert. *Kapitel 6* bietet eine kurze Zusammenfassung und einen Ausblick auf weitere Arbeiten.

2 Stand in Wissenschaft und Technik

Das Thema Reorganisation von physischen Speicherstrukturen, egal ob im Zusammenhang mit Datenbanken oder auch Dateisystemen, wurde in der Forschung in den vergangenen Jahren immer wieder mit unterschiedlicher Intensität und Zielstellung behandelt. Die genannten Beiträge stellen dabei nur eine beispielhafte Auswahl dar.

Einige ältere Beiträge befassen sich zunächst mit dem Begriff Datenbankreorganisation und dem Herausarbeiten von Ebenen, auf denen Datenbankreorganisationen ansetzen

können [NF76, SG79], sowie mit der Bestimmung von Reorganisationszeitpunkten bzw. der Festlegung von Reorganisationsintervallen [BG82, Tue78].

Mit verschiedenen Methoden zur Reorganisation von physischen Speicherungsstrukturen befasst sich ein Großteil der Beiträge in der Literatur. Dabei liegen die Schwerpunkte auf Methoden zur Online-Reorganisation von Datenbank- bzw. Dateistrukturen [OLS94, SPO89, ZS98] und auf Methoden, die parallel zum laufenden Betrieb quasi eine permanente Reorganisation des Datenbestands vornehmen [Soe81].

Weiterhin werden Möglichkeiten behandelt, durch die Verwendung geeigneter Speicherungsstrukturen und -konzepte auf der Definitionsebene den Aufwand für die Abarbeitung einer erwarteten Workload möglichst gering zu halten [AON96, GBG04, Omi89, SWS+05]. Dabei werden oft auch Verfahren zur Konvertierung der Strukturen beschrieben.

Das Thema der Umstellung von Datenbankschemata (Restrukturierung, bzw. Schemaevolution) wird in der Literatur ebenfalls ausführlich behandelt, soll hier nicht weiter aufgegriffen werden, da sich der Schwerpunkt der dortigen Betrachtungen wesentlich von dem des vorliegenden Beitrags unterscheidet.

Derzeit verbreitete Werkzeuge, die u.a. Unterstützung bei Reorganisationsentscheidungen bieten, wie z.B. REORGCHK/REORG für DB2 [IBM02], Oracle Enterprise Manager [Ora03] oder BMC Space Expert [BMC04] arbeiten rein kennzahlenbasiert, ohne Berücksichtigung der gegen die Datenbank gerichteten Workload. Typischerweise werden hier aus statistischen Daten über die Datenbankobjekte Degenerierungsgrade berechnet, wie z.B. der prozentuale Anteil ausgelagerter Sätze oder der Freiplatzanteil. Überschreiten diese Zahlen bestimmte Grenzwerte, die entweder fest vorgegeben sind oder vom DBA festgelegt werden können, wird vom Werkzeug eine Reorganisation empfohlen. Eine Quantifizierung des damit erreichbaren Nutzens, besonders bezüglich der Systemleistung, wird aber nicht vorgenommen, weil keine Workload-Informationen verwendet werden.

Die Grundidee aus [BG82], einen eventuell vorhandenen Reorganisationsbedarf aus Informationen über Workload und den Zustand von Speicherungsstrukturen zu bestimmen, wird im vorliegenden Beitrag aufgegriffen. Allerdings soll hier nicht vordergründig die Dauer von Reorganisationsintervallen bestimmt, sondern der zum Zeitpunkt der Analyse erreichbare *Nutzen* der Datenbankreorganisation *quantifiziert* werden, ohne dabei auf Erfahrungswerte zurückgreifen zu müssen. Weiterhin ergeben sich einige neue Herausforderungen gegenüber der in [BG82] betrachteten Situation, die berücksichtigt werden müssen. Die dort beschriebene Methode zielt auf die *Minimierung der gesamten Verarbeitungskosten*. Ob dieses „hehre“ Ziel bei Anwendungen im Produktivbetrieb konsequent verfolgt werden kann, erscheint fraglich. Verfügbarkeitsanforderungen und Speicherkosten können dafür sorgen, dass die automatisch bestimmten Reorganisationsintervalle oder Speicherungsparameter nicht eingehalten werden können. Die *Workload-Informationen* beruhen auf Annahmen bzw. Erfahrungen aus der Vergangenheit. Dies setzt voraus, dass entsprechende Informationen über längere Zeiträume hinweg gesammelt wurden (und auch für die absehbare Zukunft weiter Aussagekraft besitzen).

3 Reorganisationsgründe und Reorganisationsmethoden

3.1 Degenerierungen

Sich in den Bereichen zur Speicherung von Daten und Indexen entwickelnde Degenerierungen sind oft die Ursache für steigenden Speicherbedarf und die Verschlechterung der Systemleistung. Eingehendere Betrachtungen zu verschiedenen Speicherungsformen und möglichen Degenerierungen werden bspw. in [Dor05] angestellt. Hier soll nur kurz auf einige häufig auftretende Degenerierungen eingegangen werden.

Datenlöschungen oder tupelverkürzende Änderungsoperationen verursachen *eingestreute Freiplatzfragmente* in Datenbereichen und zu *dünn besetzten Indexknoten*. Der freigewordene Platz wird oft nicht oder nicht sofort wieder benutzt. Dies bewirkt eine Verschlechterung der Speicherauslastung und eine Erhöhung des I/O-Aufwands bei Suchoperationen, da die Freiplatzfragmente mit gelesen werden müssen.

Änderungsoperationen können eine Verlängerung von Tupeln verursachen. Das kann dazu führen, dass Tupel nicht mehr im ursprünglichen Datenblock gespeichert werden können und in andere Datenblöcke verschoben werden. Zur Vermeidung eines u.U. aufwendigen Nachpflegens von Indexen wird am ursprünglichen Speicherort üblicherweise ein Zeiger (Auslagerungszeiger) auf den neuen Speicherort abgelegt (*migrierte Tupel*). Dieser Zeiger muss bei Zugriffen auf Daten verfolgt werden und verursacht eine Erhöhung des Aufwands.

Durch die Speicherung von Daten in Zugangsreihenfolge und durch die Möglichkeit, Daten nach unterschiedlichen Kriterien zu indexieren, entspricht die *physische Sortierung* von Daten i.d.R. nicht den Ordnungskriterien der verschiedenen zur Tabelle gehörenden Indexe. Werden Daten nach einem bestimmten Kriterium sortiert benötigt, so steigt der Sortieraufwand, wenn diese nur wenig vorsortiert gespeichert sind. Auch bei Bereichssuchen unter Nutzung von Indexen (Index Scans) steigt der Aufwand, weil Datenblöcke evtl. mehrfach gelesen werden müssen.

3.2 Reorganisationsmethoden

Bei der Reorganisation von Datenbankobjekten können unterschiedliche Methoden (*Abbildung 1*) verwendet werden. *Zyklisch (I)* werden Reorganisationsmethoden nach einer bestimmten Nutzungsphase angewendet, um während dieser Nutzungsphase entstandene Degenerierungen zu beseitigen. Solche Reorganisationen sind typischerweise nach einer kurzen Zeitspanne beendet.

Parallel (II) arbeitende Reorganisationsmethoden nehmen Bereinigungen bzw. Veränderungen physischer Speicherungsstrukturen parallel zum laufenden Datenbankbetrieb vor. Dabei werden meist sehr kleine Granulate „nach und nach“ bearbeitet und die Reorganisationsoperationen u.U. mit anderen (Änderungs-) Operationen (z.B. des normalen Datenbankbetriebs) verknüpft.

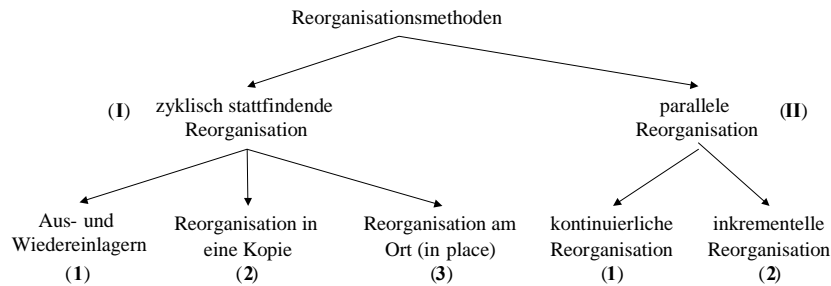


Abbildung 1: Reorganisationsmethoden

Bei der Methode **I.1** wird zunächst der Inhalt der zu reorganisierenden Datenbankobjekte aus der Datenbank *ausgelagert* (entladen bzw. exportiert). Danach werden die Datenbankobjekte zerstört und wieder neu angelegt. Anschließend werden die zuvor ausgelagerten Daten wieder *eingelagert* (importiert bzw. geladen). Nach dem Auslagern der Daten stehen diese allerdings bis nach Abschluss des Wiedereinlagerns nicht für Anwendungsprozesse zur Verfügung. Diese relativ starke Einschränkung steht oft im Widerspruch zu hohen Verfügbarkeitsanforderungen. Weiterhin unterliegen die Daten in dieser Zeit – also außerhalb der Datenbank befindlich – auch nicht den Integritätssicherungsmechanismen des DBMS.

Bei Methode **I.2**, die bspw. von Oracle oder DB2 angewendet wird, werden die Daten intern an einen anderen Speicherort *umkopiert*. Ist die Methode als Online-Methode implementiert, so werden nach dem Kopieren der Daten zwischenzeitlich an den „Originaldaten“ vorgenommene Veränderungen auch auf die Daten der Kopie angewendet. Lediglich zum Abschluss der Reorganisation werden Benutzerzugriffe auf die Reorganisationskandidaten kurzzeitig unterbunden und die Originale der Datenbankobjekte werden durch die reorganisierten Kopien ersetzt.

Das wesentliche Merkmal der Methode **I.3** ist, dass die die Tupel einer Tabelle darstellenden Sätze bzw. Indexeinträge einzeln innerhalb der bereits von der Tabelle belegten Speicherbereiche (quasi *am Ort* bzw. *in place*) so umkopiert werden, dass Freispeicherlücken wieder gefüllt oder angestrebte interne Sortierreihenfolgen wieder erreicht werden.

Bei einer *parallelen* Reorganisation kann die Wartung der Speicherstrukturen durch einen *kontinuierlich* arbeitenden Reorganisationsprozess parallel zum laufenden Datenbankbetrieb durchgeführt werden (**II.1**). Bei der *inkrementellen* Methode (**II.2**) erfolgt die Reorganisation nach und nach. Sie wird in kleinen Einheiten durchgeführt und der aktuell vom Reorganisationsprozess bearbeitete Teil des Datenbestands wird für Zugriffe von Benutzertransaktionen gesperrt, während der Rest voll verfügbar bleibt.

4 Auswahl von Reorganisationskandidaten

4.1 Grundlegende Betrachtungen

Für die Bestimmung eines evtl. vorhandenen Reorganisationsbedarfs sind Kennzahlen über den Zustand der zur Speicherung verwendeten physischen Strukturen notwendig. Die Kennzahlen, die für Zwecke der Anfrageoptimierung gesammelt und im Datenbankkatalog hinterlegt werden, sind dafür größtenteils ausreichend. Bspw. ist der sog. Clustering Factor ein Maß für den Grad der Sortierung von Daten nach dem Ordnungskriterium eines Index. Als hinderlich für eine möglichst produktübergreifende Betrachtung erweist sich allerdings die fehlende Standardisierung der Teile der Datenbankkataloge, die die entsprechenden Statistikdaten enthalten. Daher wurde während der im Rahmen von [Dor06] angestellten Untersuchungen ein Ansatz für ein vereinheitlichtes Speicher- und Verhaltensmodell, das „eInformationsschema“ entwickelt. Dabei steht das „e“ schlicht für „erweitert“, um Verwechslungen mit dem Informationsschema der SQL-Norm zu vermeiden. Aus den Kennzahlen lässt sich zunächst grob ableiten, welche Auswirkungen die vorhandenen Degenerierungen auf bestimmte Operationen haben. So führt (evtl. vorhandene Nebenbedingungen hier außer Acht gelassen) z.B. eine Speicherplatzauslastung von nur 50 % in den Datenblöcken zu einer Verdoppelung des Aufwands für das sequenzielle Suchen gegenüber dem Suchaufwand, der bei einer vollen Belegung der Blöcke anfällt. Das klingt zunächst nach einem dringenden Reorganisationsbedarf. Allerdings bleibt die Frage, ob und in welchem Umfang bei der gegebenen Workload intern sequenzielle Suchoperationen auf das entsprechende Datenbankobjekt angewendet werden. Machen sequenzielle Suchoperationen über dem betrachteten Datenbankobjekt einen nennenswerten Anteil an der Gesamt-I/O-Last aus? Der „statische Blick“ allein reicht also i.d.R. nicht aus.

Eine Berücksichtigung der Workload im Rahmen von Reorganisationsbedarfsanalysen, bei denen auch die Auswirkungen von Degenerierungen auf die Systemleistung quantifiziert werden sollen, ist also unerlässlich. Das heißt, Informationen über die gegen die Datenbankobjekte gerichteten Anweisungen müssen in die Betrachtungen einbezogen werden. Nur so ist eine Quantifizierung des Nutzens von Datenbankreorganisationen mit einer ausreichenden Genauigkeit möglich. Derartige Vorgehensweisen sind teilweise bei Werkzeugen (Advisors, Wizards), die den DBA bei der Indexierung von Datenbeständen unterstützen sollen, bereits üblich und z.B. für den Microsoft SQL Server oder DB2 verfügbar [CN98].

Zur Bereitstellung der Informationen über die Workload stehen prinzipiell zwei unterschiedliche Varianten zur Verfügung. Ein Weg ist die manuelle Erstellung einer Workload-Beschreibung. Hier besteht aber das Problem, dass bei Datenbanken, die von unterschiedlichsten Anwendungen genutzt werden, nur mit einem erheblichen Aufwand ein Gesamtüberblick über die Operationen (SQL-Statements, Häufigkeiten usw.) aller Anwendungen erreicht werden kann, der dann in die Workload-Beschreibung einfließt. Eine andere mögliche Vorgehensweise stellt die Protokollierung der gegen die Datenbank gerichteten Anweisungen durch den Query-Prozessor oder eine aufgesetzte Komponente in einem repräsentativen Zeitraum dar. Hier besteht zwar die Gefahr, dass

das Protokoll einen erheblichen Umfang erreicht, allerdings existieren auch Arbeiten, die sich mit diesem Problem auseinandersetzen und Lösungsmöglichkeiten präsentieren (z.B. [CGN02]). Eine einfache, aber durchaus wirkungsvolle Methode besteht bereits darin, die ausgeführten SQL-Anweisungen nur einmal im Protokoll zu speichern und zusätzlich einen Zähler mitzuführen, der die Häufigkeit der Ausführungen angibt. Abhängig davon, welche Datenbankobjekte (z.B. einzelne Tabellen oder alle Tabellen eines Table Space usw.) im Rahmen der Analyse betrachtet werden sollen, schließt sich an die Protokollierung evtl. noch eine Aufbereitung des Protokolls an, bei der nicht relevante Anweisungen entfernt werden. Die entstandene Workload-Beschreibung kann dann fortan verwendet werden, solange keine signifikanten Änderungen im Operations-Mix auftreten.

Aus den einzelnen SQL-Anweisungen des Anweisungsprotokolls kann nicht direkt darauf geschlossen werden, wie diese Anweisungen vom DBMS intern realisiert werden. Es ist die Aufgabe des Anfrageoptimierers, hier einen geeigneten kostengünstigen Weg (Ausführungsplan, Anfragegraph) zu bestimmen. Ausführungspläne enthalten Folgen von einfachen Operationen (Planoperatoren), über die die SQL-Anweisungen realisiert werden. Bei der Abarbeitung dieser Planoperatoren entstehen Kosten. Insbesondere die anfallenden I/O-Kosten können durch Datenbankreorganisationen u.U. wesentlich beeinflusst werden und stehen daher im Mittelpunkt unserer Betrachtungen. Zur Ermittlung der Ausführungspläne im Rahmen der vorgeschlagenen Methode werden die einzelnen Anweisungen an den *Anfrageoptimierer* übergeben. Dies bewirkt implizit auch die Berücksichtigung der konkreten Systemumgebung.

4.2 Quantifizierung des Nutzens einer Reorganisation

Eine Bewertung der Kosten, die für die Ausführung einer Anweisung anfallen, erfolgt bei DBMS-Produkten meist durch den *Anfrageoptimierer*. Eine Schätzung wird i.d.R. für jeden Planoperator vorgenommen. Problematisch ist hier derzeit allerdings, dass die Kostenschätzungen bei vorhandenen DBMS-Produkten „von außen“ kaum im Detail nachvollziehbar sind. Die Hersteller legen die im Anfrageoptimierer verwendeten Berechnungsvorschriften i.d.R. nicht offen. Unklar bleibt insbesondere, wie vorhandene Degenerierungen im Kostenmodell berücksichtigt werden. Wie (und ob) z.B. die Anzahl migrierter Tupel in die Kostenschätzung für einen Datenzugriff über einen Index mit einfließt oder inwiefern der Clustering Factor bei Index Scans berücksichtigt wird, bleibt ebenso unklar wie Annahmen über die Einflüsse von Blockpufferung und vorausschauendem Lesen, die teilweise in die Kostenermittlungen einfließen. Um eine Nutzung der Kostenschätzungen für Reorganisationsnutzensanalysen zu ermöglichen, müssten DBMS-Anbieter ihre Kostenfunktionen entsprechend erweitern. Für einen *Index Range Scan* werden solche Erweiterungen nachfolgend beispielhaft ausführlicher dargestellt. Für *sequenzielles Suchen* entspricht der I/O-Aufwand der Zahl der vom Datenbereich einer Tabelle belegten Blöcke. Bei einem *Index Lookup* über einen eindeutigen Index entspricht der Aufwand der Zahl der Ebenen des Index-Baums. Hinzu kommt ein Zugriff auf einen Datenblock sowie ein Mehraufwand, der dem Anteil migrierter Tupel an der Gesamtzahl der Tupel entspricht.

Während der im Rahmen von [Dor06] angestellten Untersuchungen wurde (quasi mangels Alternativen) ein *eigenes I/O-Kostenmodell* verwendet, das auch die

Auswirkungen vorhandener Degenerierungen auf die Planoperatoren berücksichtigt. Die im Modell enthaltenen Kostenfunktionen basieren auf den in eInformationsschema enthaltenen Daten und stellen größtenteils Erweiterungen der aus dem Bereich der Anfrageoptimierung bekannten Kostenfunktionen dar. Die Kostenfunktionen sind weitgehend allgemein gültig gehalten. Allerdings lassen sich Systemabhängigkeiten hier nicht vollständig vermeiden. Dies gilt bspw. bei der Berücksichtigung spezieller Adressierungskonzepte in Sekundärindizes indexorganisierter Tabellen oder bei der Abschätzung der Kosten für die Durchführung von Datenbankreorganisationen, weil die Hersteller von DBMS-Produkten hier verschiedene Implementierungsmöglichkeiten umgesetzt haben. Im Rahmen der Quantifizierung des Nutzens von Datenbankreorganisationen werden I/O-Kosten betrachtet, da diese einen erheblichen Anteil an dem bei Datenzugriffen anfallenden Gesamtaufwand ausmachen und mit einer Datenbankreorganisation maßgeblich beeinflusst werden können. Die prozentuale Verringerung des zur Workload-Abarbeitung anfallenden I/O-Aufwands stellt hier den Nutzen einer Datenbankreorganisation bezüglich der Verarbeitungskosten dar.

Bei der Bestimmung des Nutzens werden zunächst je Planoperator (P) über eine zugehörige Kostenfunktion die Kosten, die vor einer Reorganisation anfallen (K_{Pvor}) und der jeweils durch vorhandene Degenerierungen verursachte prozentuale Mehraufwand (M_p) ermittelt. In *Abbildung 2* ist die Arbeitsweise eines *Index Range Scan* dargestellt.

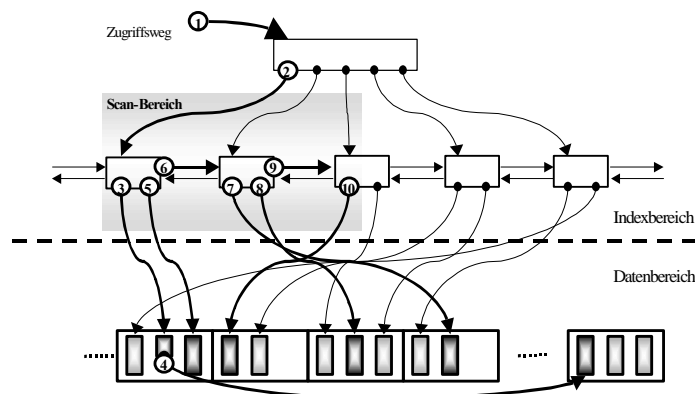


Abbildung 2: Blockzugriffe bei einem Index Range Scan

Die Anzahl notwendiger Blockzugriffe (*Gleichung 1*) ergibt sich hier zunächst aus der Summe der Anzahl Indexebenen (I_{LEV} – ohne Blattebene) und der Anzahl Blätter des Indexbaums, die entlang der Verkettung durchlaufen werden müssen (I_{LEAF} bzw. ein Teil davon). Über die in den Blättern des Indexbaums gespeicherten Verweise kann dann auf die einzelnen Tupel zugegriffen werden. Dabei müssen eventuell vorhandene Zeiger auf migrierte Tupel verfolgt werden (Zugriff Nr. 4 in *Abbildung 2*), deren Anzahl über A berücksichtigt wird. Weiterhin werden Informationen über den Grad der Übereinstimmung der Ordnung eines Index und der Sätze, auf die die Index-Einträge verweisen (*Clustering Factor* – I_{CLUST}), benötigt. Hier wird in den Statistikdaten von DBMS oftmals direkt angegeben, wie oft bei einem Index Scan über die gesamte Tabelle der Datenblock wechselt, in der der nächste zu lesende Satz gespeichert ist. Der Wert steht also zwischen „fast gar nicht springen“ und „extremem Springen“.

S (Selektivität) gibt den Anteil des Suchbereichs an der Gesamtdatenmenge an, über den der Index Scan ausgeführt wird. Die Schätzung solcher Selektivitäten kann unter Nutzung von im Datenbankkatalog hinterlegten Informationen zu Werteverteilungen von Schlüsseln erfolgen. Die Werte I_{BRi} , I_{BRh} sowie P_{BR} stehen für die Wahrscheinlichkeiten, dass die angeforderten Blöcke im Puffer gefunden werden (Pufferungsgrade). Bei B- und B*-Baum-Indexten werden im eInformationsschema diese Pufferungsgrade einzeln für jede Indextebene geführt. In [Kli05] wurde eine Methode entwickelt und prototypisch für Oracle 10g implementiert, die die Ermittlung solcher feingranularen Pufferungsgrade ermöglicht¹. Deren Berücksichtigung gestattet eine Bestimmung des Nutzens von Datenbankreorganisationen bezüglich der erreichbaren Reduzierung von physischen I/O-Operationen (Zugriffe auf Datenträger).

$$K_{P_{vor}} = \underbrace{\sum_{i=1}^{LEV-1} (1 - I_{BRi})}_{\text{innere Ebenen}} + \underbrace{\max(S \cdot I_{LEAF} \cdot (1 - I_{BRh}), (1 - I_{BRh}))}_{\text{Blattebene}} + \underbrace{(I_{CLUST} + A) \cdot (1 - P_{BR}) \cdot S}_{\text{Datenteil}} \quad \text{Gleichung 1}$$

Der in den Kosten für die jeweiligen Planoperatoren enthaltene I/O-Mehraufwand in Prozent kann planoperatorspezifisch errechnet werden. Für den betrachteten Index Range Scan kann der enthaltene Mehraufwand wie in *Gleichung 2* gezeigt ermittelt werden.

$$M_P = \frac{A \cdot (1 - P_{BR})}{\underbrace{\sum_{i=1}^{LEV-1} (1 - I_{BRi}) + I_{LEAF} \cdot (1 - I_{BRh})}_{\text{Indexteil}} + \underbrace{I_{CLUST} \cdot (1 - P_{BR})}_{\text{Datenteil}}} \cdot 100 \quad \text{Gleichung 2}$$

Sind die vor der Reorganisation anfallenden Kosten und der darin enthaltene Mehraufwand bekannt, so können die zu erwartenden Kosten ($K_{P_{nach}}$) für den Planoperator nach der Reorganisation über

$$K_{P_{nach}} = \frac{K_{P_{vor}}}{1 + \frac{M_P}{100}} \quad \text{Gleichung 3}$$

näherungsweise vorhergesagt werden. Implizit fließt hier die Annahme ein, dass die Ausführungspläne – Operatorverwendung, -reihenfolgen etc. – für die SQL-Anweisungen vor und nach einer Reorganisation gleich sind. Diese Annahme wird meist auch zutreffen. Darüber hinaus kann, wenn der Anfrageoptimierer korrekt arbeitet, davon ausgegangen werden, dass durch Veränderungen (Verbesserungen) in den Ausführungsplänen eine noch höhere Kostenreduzierung erreicht wird als zunächst angenommen. Das heißt, dass das bei der hier vorgestellten Form einer Reorganisationsnutzenbestimmung ermittelte Einsparungspotenzial bei der Reorganisationsdurchführung u.U. übertroffen werden kann.

Durch die Summierung der durch die einzelnen Planoperatoren vor bzw. nach einer Reorganisation verursachten Kosten können die Gesamtkosten (bezogen auf die Anzahl

¹ Die Pufferungsgrade beziehen sich auf einzelne Tabellen sowie auf die einzelnen Ebenen von Index-Bäumen. Bei der Nutzung von Partitionierungskonzepten können diese Informationen auch für die einzelnen Partitionen gesammelt werden. Damit können zumindest Überprüfungen des Kostenmodells durchgeführt werden. Für einen Einsatz großen Produktivumgebungen muss allerdings die Effizienz der aufgesetzt realisierten Methode noch verbessert werden.

notwendiger Blockzugriffe) vor (K_{Wvor}) und nach der Reorganisation (K_{Wnach}) ermittelt werden. Der Nutzen der Reorganisation (N) kann anschließend mit

$$N = \left(1 - \frac{K_{Wnach}}{K_{Wvor}}\right) * 100 \quad \text{Gleichung 4}$$

berechnet werden. Er entspricht der prozentualen Reduzierung der I/O-Kosten zur Abarbeitung der Workload durch die Datenbankreorganisation (Reorganisationsnutzen).

4.3 Abschätzung der Reorganisationskosten

Neben dem Nutzen sollten bei der Entscheidung über die Durchführung von Datenbankreorganisationen auch die durch sie verursachten Kosten berücksichtigt werden. Zur Reorganisation von Tabellen und den zugehörigen Indexen werden im Wesentlichen die drei in *Abschnitt 3.2* beschriebenen Methoden angewendet. Beispielhaft soll hier kurz die Abschätzung der während einer In-Place-Reorganisation anfallenden Kosten vorgestellt werden. Diese Kostenschätzungen sind von der konkreten Implementierung der Reorganisationsmethode abhängig. Hier wird die für Oracle beschriebene Implementierung [Ora03] betrachtet.

Bei den In-Place-Verfahren erfolgt die Reorganisation durch ein Verschieben von Datensätzen innerhalb der von den Reorganisationskandidaten belegten Speicherbereiche und jeweils das Pflegen der betroffenen Indexeinträge. Unter Oracle werden In-Place-Reorganisationen zur Beseitigung von eingestreuten Freiplatzfragmenten eingesetzt. Die Verschiebefunktionalität wird hier über Löschen und Wiedereinfügen von Sätzen realisiert [GK03, Ora03]. Die Prüfung von Constraints und die Ausführung von Triggern muss dabei unterdrückt werden. Die prinzipielle Vorgehensweise zeigt *Abbildung 3* schematisch an einer Tabelle mit sechs Datenblöcken.

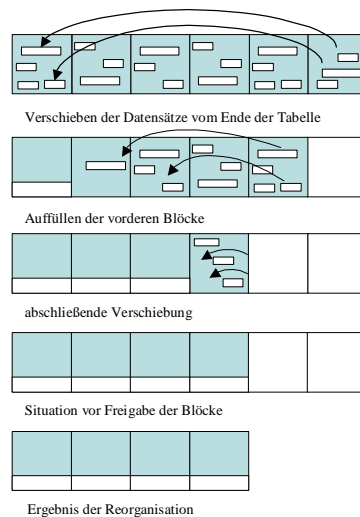


Abbildung 3: Verschieben von Datensätzen zur Beseitigung von eingestreutem Freiplatz

Freier Speicher ist in der Abbildung weiß dargestellt. Zunächst werden in den vorderen Datenblöcken vorhandene Freispeicherlücken aufgefüllt. Dazu werden Sätze aus den hinteren Datenblöcken nach und nach in die vorderen Blöcke verschoben (wie im oberen Teil von Abbildung 3 angedeutet).

Beim Auffüllen wird der in den vorderen Blöcken vorhandene Freispeicher zunächst zusammengeführt. Dies ist im Datenbankpuffer effizient realisierbar. Beim Auffüllen der Blöcke werden auch definierte Füllungsgrade berücksichtigt und die vorgesehenen Speicherreserven freigehalten. Dies ist jeweils im unteren Teil der bereits aufgefüllten Blöcke angedeutet. Durch das Verschieben werden die hinteren Datenblöcke geleert und können am Ende der Reorganisation freigegeben werden. Indexe werden im Rahmen der In-Place-Reorganisation innerhalb der Operationskette (Lesen, Einfügen, Löschen) für jeden Datensatz einzeln aktualisiert.

Für diese Implementierung lässt sich der anfallende Aufwand relativ einfach bestimmen, wenn die Zahl der zu verschiebenden Datensätze bekannt ist. Dazu muss zunächst die Zahl der nach der Reorganisation belegten Datenblöcke (P_{IDEAL}) näherungsweise bestimmt werden. Vorgehensweisen bei solchen Speicherplatzabschätzungen sind üblicherweise für DBMS-Produkte beschrieben. Die Differenz zur aktuell belegten Anzahl Datenblöcke (P_{USED}) entspricht der Anzahl frei werdender Blöcke. Damit kann über Gleichung 5 die Anzahl der zu verschiebenden Datensätze geschätzt werden.

$$T_{MOVE} = \left[(P_{USED} - P_{IDEAL}) \cdot \frac{P_{ANZT}}{P_{USED}} \right] \quad \text{Gleichung 5}$$

Durch die Multiplikation der Zahl zu verschiebender Sätze mit dem bei Lösch- und Einfügeoperationen anfallenden Aufwand können die Kosten für die Reorganisationsdurchführung berechnet werden (Gleichung 6).

$$C_{R\ org} = (C_{Insert} + C_{Delete}) \cdot T_{MOVE} \quad \text{Gleichung 6}$$

Genauere Ausführungen zur Abschätzung der Kosten für Einfüge- (C_{Insert}) und Löschoptionen (C_{Delete}) finden sich in [Dor06, Wie05].

4.4 Auswahl von Reorganisationskandidaten

Üblicherweise stehen bei einer Datenbankreorganisation mehrere (viele) Datenbankobjekte (Reorganisationskandidaten) zur Auswahl. Zur Reorganisationsdurchführung stehen aber nur bestimmte Zeitfenster und Ressourcen zur Verfügung. Innerhalb der Zeitfenster muss die Reorganisation jeweils abgeschlossen sein. Damit ist der betreibbare Aufwand begrenzt. Dies gilt i.d.R. auch, wenn die Reorganisation parallel zum laufenden Betrieb durchgeführt wird. Hier behindert die Belegung von Ressourcen den laufenden Datenbankbetrieb. Ein möglicher Ansatz zur Unterstützung von Datenbankadministratoren bei der Auswahl von Reorganisationskandidaten ist, den mit dem betreibbaren Aufwand erreichbaren Nutzen zu maximieren. Dieses Ziel kann mit Hilfe von Optimierungsverfahren (bspw. Branch-and-Bound-Verfahren) erreicht werden.

Die Problemstellung entspricht dabei einem aus der mathematischen Optimierung bekannten *Rucksackproblem*. Die einzelnen Reorganisationskandidaten stellen die

Gegenstände dar, die in den Rucksack gepackt werden können. Die Beschränkung des betreibbaren Aufwands entspricht der Kapazität des Rucksacks.

Die zum Packen des Rucksacks verwendbaren Gegenstände sind *unteilbar* und stehen jeweils nur einmal zur Verfügung. Auf die Reorganisationsproblematik übertragen bedeutet die „Unteilbarkeit“, dass sich das Reorganisationsgranulat nach dem bei den Kosten- und Nutzenanalysen verwendeten Granulat richtet. Wurden bspw. bei einer solchen Analyse eine Tabelle und die ihr zugeordneten Indexe als Einheit betrachtet, so wird für die Optimierung angenommen, dass eine Reorganisation auch auf die Tabelle und die ihr zugeordneten Indexe angewendet wird. Sollen evtl. auch nur einzelne Indexe reorganisiert werden, so müssen die Kosten- und Nutzenanalysen von vornherein einzeln für die Indexe und (evtl.) den Datenbereich der Tabelle durchgeführt werden.

Bei der Ermittlung der *Nutzenwerte* werden die einzelnen Reorganisationskandidaten als unabhängig angesehen. Die Reorganisation eines Kandidaten zur Beseitigung von Degenerierungen hat keinen Einfluss auf den durch die Reorganisation eines anderen Kandidaten erreichbaren Nutzen. Würde eine Reorganisation allerdings zur Veränderung der physischen Repräsentation von Datenbankobjekten durchgeführt, so können (bspw. bei hierarchisch strukturierten Objekten) hier durchaus Abhängigkeiten existieren. Dieser Fall soll hier aber nicht weiter betrachtet werden.

Die einzelnen Nutzenwerte stehen für die durch die Reorganisation erreichbare Reduzierung des I/O-Aufwands in Prozent, bezogen auf die betrachtete Workload. Nutzenwerte kleiner als 0 treten i.d.R. nicht auf, da dies bedeuten würde, dass der Aufwand zur Workload-Abarbeitung durch die Reorganisation steigt. Da dies nicht das Ziel von Datenbankreorganisationen ist, wird davon ausgegangen, dass Kandidaten, deren Reorganisation eine Aufwandserhöhung verursachen würde, vorab von den Betrachtungen ausgeschlossen werden.

Für die Optimierung mit einem Branch-and-Bound-Verfahren wird *je Kandidat ein Kostenwert* berücksichtigt. Das heißt, dass vorab feststehen muss, mit welcher Methode die Reorganisation eines Datenbankobjekts durchgeführt wird. Damit kann das Modell für das Optimierungsproblem einfach gehalten werden, was die Zahl der möglichen Lösungen in engeren Grenzen hält. Aus praktischer Sicht stellt diese Vereinfachung sicherlich kein größeres Problem dar, da die Entscheidung für eine Reorganisationsmethode i.d.R. nicht nur auf der Basis der zur Reorganisationsdurchführung anfallenden I/O-Kosten gefällt wird. Verfügbarkeitsanforderungen lassen oftmals die Nutzung kostengünstiger durchführbarer Offline-Verfahren nicht zu. Auch der Umstand, ob genügend Speicherplatz zur Verfügung steht, um eine Online-Reorganisation in eine Kopie durchzuführen, kann im Optimierungsmodell der Problemstellung nicht berücksichtigt werden. Ähnlich verhält es sich mit anderen Nebenbedingungen (wie bspw. dem Vorhandensein eines definierten Primärschlüssels). Bevor also das Optimierungsverfahren angewendet werden kann, muss zunächst im konkreten Fall geprüft werden, mit welchen Methoden die einzelnen Datenbankobjekte überhaupt reorganisiert werden können. Solche Prüfungen können oftmals mit den derzeit verfügbaren Werkzeugen zur Reorganisationsdurchführung (teilweise in unterschiedlicher Tiefe) angestellt werden. Für die *anwendbaren Methoden* können dann jeweils die anfallenden Kosten abgeschätzt und dem Administrator zur Entscheidung angeboten werden. Dieser kann festlegen, welche Reorganisationsmethode

unter den gegebenen Umständen den Verfügbarkeitsanforderungen und Forderungen nach geringen Kosten bestmöglich entspricht. Der Kostenwert für die ausgewählte Methode fließt dann in das Optimierungsverfahren ein.

5 Überprüfung in einer Beispielumgebung

Zu Überprüfungszwecken wurde das in *Abbildung 4* dargestellte Tabellenschema unter Oracle 10g implementiert. Hier wird ein vereinfachter Ausschnitt der Daten modelliert, die zur Abwicklung von mobiler Kommunikation im Global System for Mobile Communications (GSM) benötigt werden.

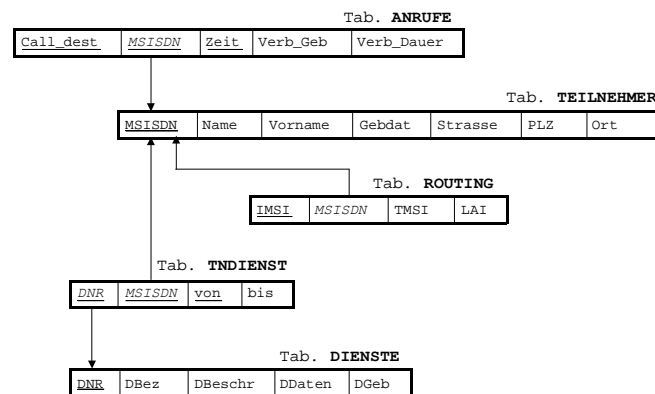


Abbildung 4: Tabellenschema des Beispiels

Im sog. Home Location Register (HLR), einer zentralen Datenbank des Mobilfunkanbieters, werden die Daten über jeden Mobilfunkteilnehmer gespeichert. Dazu gehören u.a. die eigentlichen Teilnehmerdaten, Daten über die Dienste, die vom Mobilfunkanbieter zur Verfügung gestellt und von den Teilnehmern genutzt werden können sowie die Daten über geführte Anrufe, um die entsprechenden Abrechnungen zu erstellen. Im Unterschied zu Festnetzanschlüssen sind die Teilnehmer im Mobilfunkbereich frei beweglich. Diesem Aspekt wird dadurch Rechnung getragen, dass temporär Routing-Daten gespeichert werden, über die mit einer sog. MSISDN² die für die Funkschnittstelle zum Mobiltelefon notwendigen Daten (LAI, TMSI)³ zugeordnet werden können. Die Tabellen wurden im Beispiel mit per Zufallszahlengenerator erzeugten Daten gefüllt. Anschließend wurden verschiedene Änderungsoperationen auf die Beispieldatenbank angewendet, um Degenerierungen (wie eingestreuten Freiplatz, migrierte Tupel und nicht eingehaltene interne Sortierreihenfolgen) zu erzeugen.

² Die MSISDN ist die „normale“ Telefonnummer, die gewählt werden muss, um einen Teilnehmer zu erreichen.

³ Die Location Area Identity (LAI) und die Temporary Mobile Station Identity (TMSI) dienen zur Identifizierung der Mobilfunkzelle, in der sich der Teilnehmer aufhält und zur Identifizierung des Mobiltelefons in der Zelle.

Aus den in *Abbildung 5* dargestellten SQL-Anweisungen wurden vier unterschiedlich zusammengesetzte Beispiel-Workloads mit je 5000 Anweisungen generiert, die anschließend auf die Tabellen angewendet wurden.

```

0: SELECT * FROM teilnehmer WHERE msisdn=???;
1: SELECT msisdn, name, vorname, gebdat FROM teilnehmer;
2: SELECT * FROM dienste WHERE dbez=???;
3: SELECT * FROM anrufe WHERE call_dest=???;
4: SELECT T.msisdn, T.name, T.plz, T.ort, T.strasse, A.call_dest,
A.zeit, A.verb_gib, A.verb_dauer FROM teilnehmer T, anrufe A
WHERE T.msisdn= ??? AND T.msisdn=A.msisdn;
5: SELECT R.tmsi, R.lai FROM teilnehmer T, routing R
WHERE T.msrn= ??? AND T.msisdn=R.msisdn;
6: SELECT T.name, T.vorname, T.ort, D.dbez, D.daten
FROM teilnehmer T, tndienst TN, dienste D
WHERE T.msisdn= ??? AND T.msisdn=TN.msisdn AND TN.dnr=d.dnr;
7: SELECT * FROM teilnehmer WHERE msisdn>=??? and msisdn<=???
order by msisdn;

```

Abbildung 5: Anweisungen zur Generierung von Beispiel-Workload

Dabei werden in den verschiedenen generierten Workloads nicht immer alle Anweisungen verwendet. Enthalten sind

- in der ersten Workload die Anweisungen 0 bis 6,
- in der zweiten Workload die Anweisungen 0 und 1,
- in Workload Nr. 3 die Anweisungen 0, 4, 5 und 6 und
- in der vierten Workload alle acht aufgeführten Anweisungen.

Ziel ist es, die Workload-Abhängigkeit des Nutzens von Datenbankreorganisationen zu verdeutlichen. Die Reihenfolge der Ausführung der enthaltenen Anweisungen wurde per Zufallszahlengenerator festgelegt.

In der Beispielumgebung werden die Anweisungen aufgrund entsprechender Entscheidungen des kostenbasierten Oracle-Optimierers wie folgt realisiert:

- Bei Anweisung 0 wird ein *Index Lookup* über ausgeführt.
- Die Anweisungen 1 und 2 führen jeweils eine *sequenzielle Suche* über den entsprechenden Tabellen aus.
- Anweisung 3 wird über einen *Index Lookup* realisiert.
- Anweisung 4 wird mittels eines *Nested Loop Joins* umgesetzt. Dabei wird zunächst über einen *Index Lookup* auf die Tabelle TEILNEHMER zugegriffen. Danach wird über einen *Index Range Scan* die Verbindung zur Tabelle ANRUFEN hergestellt.
- Für Anweisung 5 werden die Tabellen TEILNEHMER und ROUTING zunächst *sequenziell* gelesen und danach über einen *Hash Join* verbunden.
- Bei Anweisung 6 wird zunächst über einen *Index Lookup* eine Selektion auf der Tabelle TEILNEHMER durchgeführt. Das Ergebnis wird über einen *Nested Loop Join* mit der Tabelle TNDIENST verbunden, auf die mittels sequenziellem Suchen zugegriffen wird. Über einen weiteren *Nested Loop Join* wird anschließend die

Verbindung zur Tabelle DIENSTE hergestellt, auf die mittels *Index Lookup* zugegriffen wird.

- Anweisung 7 wird über einen *Index Range Scan* über der Tabelle TEILNEHMER realisiert.

Abbildung 6 zeigt beispielhaft die Bestimmung des Nutzens der Reorganisation der Tabelle TEILNEHMER unter der Annahme, dass Workload 2 ausgeführt wird. Workload 2 wurde hier der Übersichtlichkeit halber ausgewählt. Die in der Abbildung rechts angegebenen Statistikdaten von Tabelle und Index wurden dazu in die für *Index Lookups* bzw. *sequenzielles Suchen* eingesetzt. Der Einfluss der Datenbankpufferung in der Beispielrechnung aus Vereinfachungsgründen nicht weiter berücksichtigt.

Benötigte Daten zur Tabelle TEILNEHMER	Aufwandsschätzung für Anweisung 0 (Index Lookup - wurde ca. 2500 mal ausgeführt)
belegte Blöcke (vor Reorg.): 6233	$C_P = \left(\underbrace{3+1}_{\text{normaler Aufwand}} + \frac{2844}{\underbrace{104762}_{\text{Mehraufwand } d}} \right) \cdot 2500 = (3+1,027) \cdot 2500 = 10068$
Tupelzahl: 104762	Schätzung des enthaltenen Mehraufwands
Anzahl migr. Tupel (vor Reorg.): 2844	$M_P = \frac{\overbrace{2844}^{\text{Zusatzteile und}}}{104762 \cdot \underbrace{\left(\frac{3}{\text{Indexteil}} + \frac{1}{\text{Datenzeit}} \right)}_{\text{mind. notwendiger Aufwand}}} \cdot 100 = \frac{2844}{628572} \cdot 100 = 0,452\%$
belegte Blöcke (nach Reorg.): 4047	Berechnung der Kosten nach der Reorganisation
Benötigte Daten zum Primärschlüsselindex der Tabelle TEILNEHMER	$K_{P_{\text{neu}}} = \frac{10068}{1+0,00452} = 10023$
Anzahl Ebenen des Index: 3	Berechnung des Nutzens der Reorganisation
Aufwandsschätzung für Anweisung 1 (sequenzielle Suche - wurde ca. 2500 mal ausgeführt)	$N = \left(1 - \frac{10323334}{15592568} \right) \cdot 100 = 33,8\%$
$C_P = 6233 \cdot 2500 = 15582500$	
Schätzung des enthaltenen Mehraufwands	
$M = \left(\frac{6233}{4047} - 1 \right) \cdot 100 = 54\%$	
Berechnung der Kosten nach der Reorganisation	
$K_{P_{\text{neu}}} = \frac{15582500}{1+0,54} = 10313311$	

Abbildung 6: Beispielrechnung für Workload 2

Zur Überprüfung der errechneten Nutzenwerte wurden die Beispiel-Workloads vor und nach einer Reorganisation auf die Beispieldatenbank angewendet. Dabei wurden die tatsächlich anfallenden logischen (im Datenbankpuffer stattfindenden) und physischen Blockzugriffe unter Nutzung der Zahlen aus den von Oracle zur Verfügung gestellten dynamischen Performance Views gemessen und den berechneten Werten gegenübergestellt. Die Ergebnisse zeigt *Abbildung 7*.

Die unterschiedlichen Werte für die Workloads verdeutlichen gut die Workload-Abhängigkeit des Nutzens von Datenbankreorganisationen. Der Vergleich der errechneten Werte mit dem tatsächlich erreichten Nutzen zeigt, dass es mit der vorgestellten Methode möglich ist, den voraussichtlichen Nutzen einer Datenbankreorganisation relativ genau vorab zu errechnen. Ungenauigkeiten sind hierbei im wesentlichen auf Ungenauigkeiten bei den Abschätzungen des Speicherplatzbedarfs der Tabellen TEILNEHMER und DIENSTE nach der Reorganisation zurückzuführen, deren Tupel jeweils mehrere variabel lange Zeichenketten enthalten.

Die Abweichungen der errechneten Werte für physische Blockzugriffe (jeweils dritte Säule) von den gemessenen Werten (jeweils vierte Säule) sind etwas größer als bei den

Werten für logische Blockzugriffe. Dies liegt hauptsächlich an Ungenauigkeiten bei der Ermittlung von datenbankobjektbezogenen Pufferungsgraden. Bei der verwendeten Methode werden die Pufferungsgrade zu bestimmten Zeitpunkten ermittelt. Durch die Bildung des Mittelwerts der letzten drei Messwerte wird zwar versucht, über längere Zeiträume hinweg auftretende Schwankungen mit zu berücksichtigen und die Ungenauigkeiten zu verringern, allerdings lassen sie sich damit nicht gänzlich vermeiden.

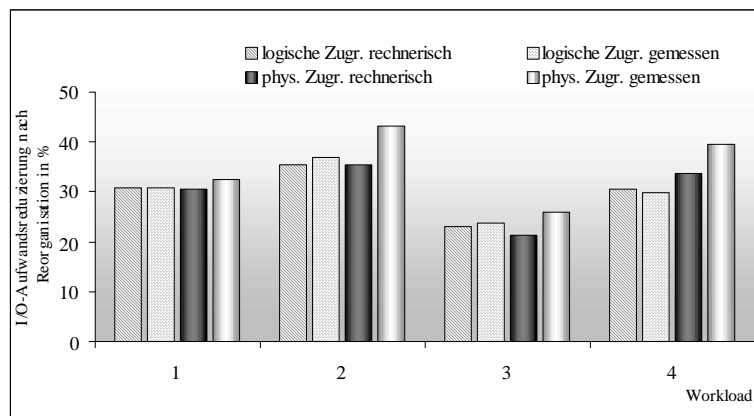


Abbildung 7: Vergleich errechneter und gemessener Nutzen von Datenbankreorganisationen bei gleichem Degenerierungsgrad und unterschiedlicher Workload-Zusammensetzung

Zur Überprüfung des Berechnungsschemas zur Schätzung der bei In-Place-Reorganisationen anfallenden Kosten wurden (ebenfalls unter Nutzung von Oracle 10g) Untersuchungen durchgeführt und Messreihen aufgenommen. Basis war die Tabelle TEILNEHMER, die zunächst mit Beispieldaten geladen wurde. Anschließend wurden durch die Ausführung von DELETE-Anweisungen über die gesamte Tabelle hinweg Freispeicherfragmente erzeugt. Über den Attributen MSISDN, Name und Ort wurde jeweils ein Index angelegt. Alle Indexe besaßen während der Messreihen eine Höhe von 3 Ebenen. Der durchschnittliche Pufferungsgrad von Indexblöcken lag während der Messreihen bei ca. 98%, der der Datenblöcke bei ca. 82%. Die von Oracle angewendete asynchrone Ausführung von Schreiboperationen führte dazu, dass in der Beispielumgebung ca. 6,7 Änderungen an Indexblöcken bzw. ca. 4,3 Änderungen an Datenblöcken jeweils zu einer physischen Schreiboperation führten. *Tabelle 1* zeigt einen Vergleich des rechnerisch ermittelten und des gemessenen physischen I/O-Aufwands zur Durchführung einer In-Place-Reorganisation der Tabelle TEILNEHMER bei verschiedenen Anteilen von eingestreutem Freiplatz und damit verschiedenen Anzahlen zu verschiebender Sätze.

Die gemessenen Aufwandswerte liegen dabei bei der ersten Messreihe etwas über und bei der zweiten Messreihe etwas unter den errechneten Werten. Dies ist darauf zurückzuführen, dass die Durchschnittswerte beider Messreihen in den Berechnungen für die Pufferungsgrade und die Anteile tatsächlich erfolgter Schreiboperationen eingeflossen sind. Durch die größere Anzahl zu verschiebender Sätze im Rahmen der zweiten Messreihe ergibt sich da auch eine höhere Lokalität der Zugriffe als bei der

ersten Messreihe. Durch die höhere Lokalität können anteilig mehr Zugriffe im Puffer erfolgen. Bei der ersten Messreihe liegt die Lokalität unter der, die sich aus den Durchschnittswerten für Pufferungsgrade und Schreiboperationen ergibt. Detailliertere Ausführungen zu den durchgeführten Messreihen finden sich in [Dor06].

	ca. 28 400 zu verschiebende Sätze (Freiplatzanteil ca. 27%)		ca. 42 300 zu verschiebende Sätze (Freiplatzanteil ca. 41%)	
	errechnete Werte	gemessene Werte	errechnete Werte	gemessene Werte
gelesene Datenblöcke	5112	6438	7614	6868
geschriebene Datenblöcke	6532	7734	9729	7919
gelesene Indexblöcke	1704	2521	2538	1459
geschriebene Indexblöcke	12695	12886	18909	18634
Summen	26043	29579	38790	34880

Tabelle 1: Gegenüberstellung von errechnetem und gemessenem I/O-Aufwand

6 Zusammenfassung und Ausblick

In diesem Beitrag wurde eine Methode zur Unterstützung von Datenbankadministratoren bei der Auswahl von Kandidaten für die Durchführung von Datenbankreorganisationen vorgestellt. Dazu wird zunächst der Nutzen der Reorganisation von Datenbankobjekten auf der Ebene der physischen Speicherungsstrukturen bezüglich der Systemleistung ermittelt. Eine mögliche Vorgehensweise hierzu wurde beschrieben. Als Nutzen wird hauptsächlich die durch eine Datenbankreorganisation erreichbare prozentuale Verringerung (bezogen auf den zur Workload-Abarbeitung anfallenden Gesamtaufwand) des I/O-Aufwands betrachtet. Insbesondere die Workload-Berücksichtigung unterscheidet die hier vorgestellte Methode von den Vorgehensweisen von derzeit am Markt verfügbaren Produkten für Reorganisationsbedarfsanalysen. Der Vorteil der vorgeschlagenen Methode liegt aber auch darin, dass benötigte Eingabegrößen (Workload, Zustand der Speicherungsstrukturen, Systemumgebung) auf den jeweiligen Systemen unter realistischen Bedingungen ermittelt werden, für die die Analysen durchgeführt werden sollen.

Neben dem Nutzen müssen bei der Entscheidung über die Durchführung von Datenbankreorganisationen auch die durch sie verursachten Kosten berücksichtigt werden. Wie die Ermittlung solcher Kosten erfolgen kann, wurde am Beispiel der für das DBMS-Produkt Oracle implementierten In-Place-Reorganisationsfunktionalität dargestellt.

Anhand von durchgeführten Messreihen wurde die Funktionstüchtigkeit der vorgeschlagenen Methoden zur Kosten- und Nutzenermittlung gezeigt.

Basierend auf der Möglichkeit der näherungsweisen Vorhersage von Kosten und Nutzen von Datenbankreorganisationen wurde ein Ansatz zur Unterstützung von Datenbankadministratoren bei der Auswahl von Reorganisationskandidaten vorgestellt, der auf die Maximierung des von einer Datenbankreorganisation zu erwartenden Nutzens unter Berücksichtigung einer Kostenobergrenze für die Reorganisation abzielt.

Die Genauigkeit der Kosten- und Nutzenabschätzungen ist vom verwendeten Kostenmodell abhängig. In den Berechnungen wird von einer Gleichverteilung vorliegender Degenerierungen und anfallender Datenzugriffe ausgegangen. In den Beispielumgebungen konnten diese Bedingungen auch eingehalten werden. In realen Anwendungsumgebungen kann von solchen "idealen" Bedingungen sicher nicht immer ausgegangen werden. Weiterführende Untersuchungen müssen sich u.a. damit befassen, wie in den Kosten-/Nutzenbetrachtungen die in realen Anwendungsumgebungen vorhandenen Schiefen in Zugriffsmustern bzw. bezüglich des Vorhandenseins von Degenerierungen berücksichtigt werden können.

Literaturverzeichnis

- [AF06] J. Albrecht, M. Fiedler. Datenbank-Tuning – einige Aspekte am Beispiel von Oracle 10g. In *Datenbank-Spektrum, Heft 16/2006*. dpunkt.verlag, Heidelberg, Februar 2006
- [AON96] K. J. Achyutuni, E. Omiecinski, S. B. Navathe. Two Techniques for On-Line Index Modification in Shared Nothing Parallel Databases. In *Proc. of the 1996 ACM SIGMOD International Conference on Management of Data*. Montreal, Kanada Juni 1996
- [BG82] D. S. Batory, C. C. Gotlieb. A Unifying Model of Physical Databases. In *ACM Transactions on Database Systems (TODS), Vol. 7, No. 4*. ACM Press, Dez. 1982
- [BMC04] *SmartDBA Performance Management Solutions for Oracle, Version 2.6*. White Paper. BMC Software Inc., 2004
- [BVR03] *Jahresbericht des Bundesverbands der Deutschen Volksbanken und Raiffeisenbanken 2002*. BVR, Berlin, 2003
- [CGN02] S. Chaudhuri, A. Gupta, V. Narasayya. Compressing SQL Workloads. In *Proc. of the 2002 ACM SIGMOD International Conference on Management of Data*. Madison, USA, 2002
- [CN98] S. Chaudhuri, V. Narasayya. AutoAdmin 'What-if' Index Analysis Utility. In *Proc. of the 1998 ACM SIGMOD International Conference on Management of Data*. Seattle, USA, Juni 1998
- [Dor05] S. Dorendorf. Quantifizierung des zu erwartenden Nutzens von Datenbankreorganisationen. In *Informatik-Forschung und Entwicklung, Bd. 20, Nr. 1-2*. Springer-Verlag, Berlin/Heidelberg, Okt. 2005
- [Dor06] S. Dorendorf. *Reorganisation von Datenbanken - Auslöser, Verfahren, Nutzenermittlung* -. Dissertation. Friedrich-Schiller-Universität Jena, Okt. 2006
- [DPW05] *ZEBRA-Zentrale Briefdatenbank, Architekturüberblick*. Projektunterlagen. Deutsche Post World Net, 2003-2005
- [GBG04] P. Ganesan, M. Bawa, H. Garcia-Molina. Online Balancing of Range-Partitioned Data with Applications to Peer-to-Peer Systems. In *Proc. of the 30th International Conference on Very Large Data Bases*. Toronto, Kanada, Sept. 2004
- [GK03] A. Ganesh, S. Kumar. *The Self-Managing Database: Proactive Space & Schema Object Management*. White Paper. Oracle Corporation, Nov. 2003
- [IBM02] *IBM DB2 Universal Database Command Reference Version 8*. International Business Machines Corporation, 2002

- [Joc05] C. Jochum. Versinkt das IT-Management in der Bedeutungslosigkeit? In *Informatik-Spektrum*, Bd. 28, Nr. 4. Springer-Verlag, Berlin/Heidelberg, Aug. 2005
- [Kli05] O. Klinger. *Abschätzung von I/O-Kosten für Zugriffsoperationen bei RDBMS*. Diplomarbeit. Institut für Informatik, Friedrich-Schiller-Universität Jena, Sept. 2005
- [NF76] S. B. Navathe, J. P. Fry. Restructuring for Large Data Bases: Three Levels of Abstraction. In *ACM Transactions on Database Systems (TODS)*, Vol. 1, No. 2. ACM Press, Juni 1976
- [OLS94] E. Omiecinski, L. Lee, P. Scheuermann. Performance Analysis of a Concurrent File Reorganization Algorithm for Record Clustering. In *IEEE Transactions on Knowledge and Data Engineering*, Vol. 6, No. 2. IEEE Computer Society, Apr. 1994
- [Omi89] E. Omiecinski. Concurrent File Conversion between B+-Tree and Linear Hash Files. In *Information Systems*, Vol. 14, No. 5. Pergamon Press, Nov. 1989
- [Ora03] *Oracle Database 10g: The Self-Managing Database*. White Paper. Oracle Corporation, Nov. 2003
- [Ruf05] T. Ruf. *Datenbanken heute: Was hat/braucht die betriebliche Praxis seit/in 25 Jahren – und was nicht?*. Vortrag. Informatik-Kolloquium der Friedrich-Schiller-Universität Jena, der Regionalgruppe Ostthüringen der Gesellschaft für Informatik (GI) und der Fachhochschule Jena, Jena, 25. Apr. 2005
- [Sch04] R. Schumacher. Why You Need Capacity Planning. In *Database Trends And Applications*, Vol. 18, No. 7. www.dbta.com, Juli 2004
- [SG79] G. H. Sockut, R. P. Goldberg. Database Reorganization - Principles and Practice. In *ACM Computing Surveys*, Vol. 11, No. 4. ACM Press, Dez.. 1979
- [Soe81] L. Söderlund. Concurrent Data Base Reorganization – Assessment of a Powerful Technique through Modeling. In *Proc. of the 7th International Conference on Very Large Data Bases*. Cannes, Frankreich, Sept. 1981
- [SPO89] P. Scheuermann, Y. C. Park, E. Omiecinski. A Heuristic File Reorganization Algorithm based on Record Clustering. In *BIT Computer Science and Numerical Mathematics*, Vol. 29, No. 3. Springer-Verlag Niederlande, Sept. 1989
- [SWS+05] X. Sun, R. Wang, B. Salzberg, C. Zou. Online B-Tree Merging. In *Proc. of the 2005 ACM SIGMOD International Conference on Management of Data*. Baltimore, USA, Juni 2005
- [Tue78] W. G. Tuel. Optimum Reorganization Points for Linearly Growing Files. In *ACM Transactions on Database Systems (TODS)*, Vol. 3, No. 1. ACM Press, März 1978
- [Wie05] F. Wiczorek. *Datenbankreorganisationen: Verfahren und Kostenvorhersagen*. Diplomarbeit. Institut für Informatik, Friedrich-Schiller-Universität Jena, Dez. 2005
- [ZS98] C. Zou, B. Salzberg. Safely and Efficiently Updating References During On-line Reorganization. In *Proc. of the 24th International Conference on Very Large Data Bases*. New York, USA, Aug. 1998