(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 600 04 385 T2** 2004.06.24

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 222 569 B1**
(21) Deutsches Aktenzeichen: **600 04 385.1**
(86) PCT-Aktenzeichen: **PCT/DK00/00354**
(96) Europäisches Aktenzeichen: **00 940 226.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 01/008041**
(86) PCT-Anmeldetag: **30.06.2000**
(87) Veröffentlichungstag der PCT-Anmeldung: **01.02.2001**
(97) Erstveröffentlichung durch das EPA: **17.07.2002**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **06.08.2003**
(47) Veröffentlichungstag im Patentblatt: **24.06.2004**

(51) Int Cl.[7]: **G06F 17/30**

(30) Unionspriorität:
104599 **21.07.1999** **DK**

(84) Benannte Vertragsstaaten:
AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LI, LU, MC, NL, PT, SE

(73) Patentinhaber:
Pedersen, Torben Bach, Arhus, DK; Jensen, Christian S., Gistrup, DK; Dyreson, Curtis E. S., Pullman, Wash., US

(74) Vertreter:
Samson & Partner, Patentanwälte, 80538 München

(72) Erfinder:
Pedersen, Torben Bach, 8200 Arhus N, DK; Jensen, Christian S., 9260 Gistrup, DK; Dyreson, Curtis E.S., Pullman, US

(54) Bezeichnung: **VERFAHREN UND SYSTEME UM OLAP HIERARCHIEN ZUSAMMENFASSBAR ZU MACHEN**

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

Beschreibung

Gebiet der Erfindung

[0001] Die vorliegende Erfindung betrifft ein Verfahren, ein Computersystem und ein Computerprogrammprodukt für ein Computersystem zur Umwandlung von Online-Analytical-Processing (OLAP) Hierarchien in summierbare Hierarchien, wodurch Vorab-Aggregationen ermöglicht werden. Dadurch werden schnelle Anfrage-Antwortzeiten für Aggregationsanfragen ohne übermäßige Speicherbenutzung ermöglicht, sogar wenn die Hierarchien ursprünglich irregulär sind.

Hintergrund der Erfindung

[0002] Online-Analytical-Processing (OLAP) Systeme, die darauf abzielen den Vorgang des Abrufens nützlicher Information aus großen Mengen detaillierter Transaktionsdaten zu erleichtern, haben in herkömmlichen Geschäftsanwendungen ebenso wie in neuen Anwendungen, wie im Gesundheitssystem, eine breite Akzeptanz erreicht. Diese Systeme bieten im allgemeinen eine dimensionale Sicht der Daten, in der gemessene Werte, Fakten genannt, durch deskriptive Werte beschrieben werden, die von einer Anzahl von Dimensionen abgeleitet sind; und die Werte von Dimensionen sind typischerweise in einer Hierarchie vom Containment-Typ organisiert. Eine prototypische Anfrage wendet eine Aggregationsfunktion, wie etwa Mittelung, auf die Fakten an, die durch bestimmte Werte aus den Dimensionen beschrieben sind.

[0003] Von diesen Systemen werden schnelle Antwortzeiten gefordert, selbst für Anfragen, die große Datenmengen aggregieren. Die vielleicht wichtigste Technik, die zur Erfüllung dieser Forderung verwendet wird, wird als Vorab-Aggregation (preaggregation) bezeichnet, wobei die Ergebnisse von Aggregationsanfragen vorab berechnet und gespeichert, das heißt für eine spätere Verwendung während der Anfragebearbeitung erzeugt werden. Vorab-Aggregation hat eine beträchtliche Aufmerksamkeit in der Forschungsgemeinde auf sich gezogen, wo untersucht wurde, wie vorab-aggregierte Daten optimal für die Anfrageoptimierung verwendet werden [7, 3] und wie die vorab-aggregierten Daten zu pflegen sind, wenn Basisdaten aktualisiert werden [19, 24]. Des weiteren bieten die neuesten Versionen kommerzieller RDBMS-Produkte eine Anfrageoptimierung, die auf vorab berechneten Aggregaten basiert, und eine automatische Pflege der gespeicherten Aggregate, wenn Basisdaten aktualisiert werden [30].

[0004] Die schnellsten Antwortzeiten werden erreicht, wenn Aggregationsergebnisse zu allen Kombinationen von Dimensionswerten über alle Dimensionen erzeugt werden, dies wird volle (oder eifrige, eager) Vorab-Aggregation genannt. Der benötigte Speicherplatz steigt jedoch rasch, um schnell untragbar zu werden, wenn die Komplexität der Anwendung steigt. Dieses Phänomen wird Datenexplosion genannt [4, 21, 27] und tritt auf, da die Anzahl möglicher Aggregationskombinationen schnell wächst wenn die Anzahl der Dimensionen steigt, während die dünne Besetzung des multidimensionalen Raums in höheren Dimensionsebenen abnimmt, was bedeutet, daß Aggregate in höheren Ebenen beinahe denselben Platz benötigen wie Aggregate niedriger Ebenen. In einigen kommerziellen Anwendungen benötigt eine volle Vorab-Aggregation das 200-fache des Speicherplatzes der Rohdaten [21]. Ein weiteres Problem mit der vollen Vorab-Aggregation ist es, daß das Aktualisieren der erzeugten Aggregate zu lange dauert, wenn Basisdaten sich ändern.

[0005] Mit dem Ziel die Datenexplosion zu vermeiden, hat sich die Forschung darauf konzentriert, wie, unter Berücksichtigung von Randbedingungen des Speicherplatzes [1, 9, 11, 26, 28, 32] oder der Pflegezeit [10], die beste Untermenge von Aggregationsebenen auszuwählen ist oder wie die beste Kombination von Aggregationsdaten und Indizes [8] auszuwählen ist. Dieser Ansatz wird allgemein als praktische (oder partielle oder halb-eifrige [5, 11, 29]) Vorab-Aggregation bezeichnet. Es existieren inzwischen auch kommerzielle OLAP-Systeme, die die praktische Vorab-Aggregation anwenden, beispielsweise Microsoft Decision Support Services (Plato) [18] und Informix MetaCube [13].

[0006] Ein neuer Datenbankoperator, der Aggregationen für den N-dimensionalen Datenraum verallgemeinert, wurde von Jim Gray et al. offenbart: „Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals", Data Mining and Knowledge Discovery 1, 1997, und es werden Lösungen vorgeschlagen, wie dieser Operator in die Ausführungs- und SQL-Sprach-Ebene zu integrieren ist. Es wird erwähnt, daß irreguläre Dimensionshierarchien die Vorab-Aggregation unmöglich machen, es wird jedoch keine Lösung für dieses Problem gegeben.

[0007] Die der Anwendbarkeit praktischer Vorab-Aggregation zugrunde liegende Vorraussetzung ist es, daß Aggregate niedriger Ebenen zur Berechnung von Aggregaten höherer Ebenen wiederverwendbar sind, was als Summierbarkeit bekannt ist [16]. Summierbarkeit tritt auf, wenn die Abbildungen in der Dimensionshierarchie surjektiv (alle Pfade in der Hierarchie haben gleiche Länge), abdeckend (nur direkte Eltern- und Kindwerte können verknüpft werden) und streng (jedes Kind in der Hierarchie hat nur einen Elternknoten) sind; und wenn ebenso die Beziehungen zwischen Fakten und Dimensionen viele-zu-eins sind und Fakten immer auf die niedrigste Ebene in den Dimensionen abgebildet werden [16]. Die in vielen realen Anwendungen angetroffenen Da-

ten erfüllen jedoch diese strengen Bedingungen nicht. Dies motiviert die Suche nach Techniken, die es ermöglichen praktische Vorab-Aggregation für einen weiteren Bereich von Anwendungen zu nutzen, was den Fokus der vorliegenden Erfindung darstellt.

Beschreibung der Erfindung

[0008] Angeregt durch die wachsende Anwendung von OLAP-Systemen in vielen verschiedenen Anwendungen, eingeschlossen solche in der Wirtschaft und dem Gesundheitssystem, stellt die vorliegende Erfindung Umwandlungstechniken für multidimensionale Datenbanken bereit, die die existierende leistungssteigernde Technik, bekannt als praktische oder partielle oder halb-eifrige Vorab-Aggregation, wirksam einsetzen, indem sie diese Technik auf einen viel größeren Kreis von realen Anwendungen anwendbar machen.

[0009] Derzeitige Vorab-Aggregationstechniken gehen davon aus, daß die dimensionalen Strukturen summierbar sind. Genauer müssen die Abbildungen in Dimensionshierarchien surjektiv, abdeckend und streng sein; die Beziehungen zwischen Fakten und Dimensionen müssen viele-zu-eins sein und die Fakten müssen immer auf die niedrigsten Kategorien in den Dimensionen abgebildet werden.

[0010] Die vorliegende Erfindung stellt neue Umwandlungustechniken vor, die Dimensionen mit Hierarchien, die nicht-surjektiv, nicht-abdeckend und nicht-streng sind, summierbar macht. Die Umwandlungen haben eine praktisch anwendbare, niedrige Komplexität, sie können unter Verwendung von Standard-Relationale-Datenbank-Technologien implementiert werden und es wird auch offenbart, wie die umgewandelten Hierarchien für den Anwender transparent in derzeitige OLAP-Systeme integriert werden.

[0011] Die vorliegende Erfindung offenbart auch, wie die Umwandlungen entsprechend der Erfindung auf die Fälle nicht-summierbarer Beziehungen zwischen Fakten und Dimensionen anzuwenden ist, wobei dies in realen Anwendungen oft auftritt. Schließlich wird gezeigt, wie die Algorithmen zu verändern sind, um die umgewandelten Hierarchien inkrementell zu pflegen, wenn die zugrundeliegenden Daten geändert werden. Nach unserem Kenntnisstand ist diese Arbeit die erste, die Algorithmen vorstellt, die automatisch eine Summierbarkeit für nicht-abdeckende und nicht-surjektive Hierarchien erreichen. Die hier vorgestellte Forschungsarbeit ist ebenso die erste, die Techniken und Algorithmen zum Erreichen der Summierbarkeit in nicht-strengen Hierarchien darstellt. Wir glauben, daß die für den Anwender transparente Integration der Techniken in derzeitige Systeme eine neue Eigenschaft ist.

[0012] Die multidimensionalen Datenbanken oder Objekte umfassen, wie der Name anzeigt, eine Anzahl von Dimensionen, von denen jede eine Hierarchie ist, und die Grundlage der Erfindung ist es, irreguläre, das heißt nicht-abdeckende und/oder nicht-surjektive und/oder nicht-strenge Dimensionen in Dimensionen umzuwandeln, die wenigstens teilweise aggregations-normalisiert sind, in den meisten praktischen Fällen vorzugsweise voll aggregations-normalisiert, um eine praktische Vorab-Aggregation der Dimensionen zu ermöglichen. Somit betrifft die vorliegende Erfindung ein Verfahren zur Umwandlung einer allgemeinen Online-Analytical-Processing-Dimension in eine zumindest teilweise aggregations-normalisierte Dimension, d. h. in eine Dimension mit verbesserter Summierbarkeit, mittels eines Computers, wobei die Dimension Dimensionswerte hat, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, wobei die Dimension Abbildungen von Verknüpfungen zwischen Dimensionswerten umfaßt, wobei das Verfahren folgende Schritte umfaßt:

Abrufen der Abbildung von einem zum Computer gehörenden Datenspeichermittel,

Analysieren der Abbildung zur Bestimmung von Irregularitäten der Dimension, d. h. die Dimension nicht-summierbar machenden Merkmalen, anhand von durch den Computer ausgeführten Analysemitteln,

Erzeugen neuer Dimensionswerte der Dimension und Modifizieren der Abbildung zwischen Dimensionswerten der Dimension entsprechend der Analyse, wodurch die Dimension zumindest teilweise aggregations-normalisiert ist, und

Sichern der neuen Dimensionswerte und der modifizierten Abbildungen auf dem Datenspeichermittel des Computers.

[0013] Die vorliegende Erfindung betrifft in einem weiteren Aspekt und in einer Alternative zum oben offenbarten Verfahren ein Verfahren zum zumindest teilweisen Aggregations-Normalisieren mittels eines Computers von einem allgemeinen multidimensionalen Online-Analytical-Processing-Objekt, welches einen Faktensatz umfaßt, welcher eine Anzahl von Fakten, die auf eine Anzahl von aggregationsnormalisierten Dimensionen abgebildet werden, umfaßt, wobei die Dimensionen Dimensionswerte aufweisen, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, wobei das multidimensionale Objekt Abbildungen von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension umfaßt,

wobei das Verfahren folgende Schritte umfaßt:

Abrufen der Abbildung von einem zum Computer gehörenden Datenspeichermittel,

Einfügen der Abbildung der Anzahl der Fakten auf dem multidimensionalen Objekt in die Abbildung des multidimensionalen Objekts, so daß die Abbildung Verknüpfungen von jedem der Fakten zu wenigstens einem Dimensionswert in jeder der Anzahl von Dimensionen umfaßt, und die Fakten die unterste Ebene jeder der Di-

mensionen des multidimensionalen Objekts bilden,
Analyse der Abbildung des multidimensionalen Objekts zur Bestimmung der Irregularitäten der Dimension mittels eines vom Computer ausgeführten Analysemittels,
Erzeugen neuer Dimensionswerte des multidimensionalen Objekts und Modifizieren der Abbildung zwischen Dimensionswerten des multidimensionalen Objekts entsprechend der Analyse, wodurch das multidimensionale Objekt zumindest teilweise aggregations-normalisiert ist, und
Speichern der neuen Dimensionen und der modifizierten Abbildung im Datenspeichermittel des Computers.

[0014] Die vorliegende Erfindung betrifft des weiteren ein Computersystem für onlineanalytisches Processing, das wenigstens einen universell einsetzbaren Computer mit einem zugehörigen Datenspeichermittel umfaßt, wobei auf dem Datenspeichermittel ein Computerprogrammprodukt gespeichert ist, das geeignet ist, den Computer zu befähigen eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekts durchzuführen entsprechend dem/den oben beschriebenen Verfahren, wobei das Computersystem Mittel zum Abrufen des Computerprogrammprodukts und zur entsprechenden Durchführung umfaßt.

[0015] Die vorliegende Erfindung betrifft des weiteren ein Computerprogrammprodukt, das geeignet ist, einen universell einsetzbaren Computer zu befähigen eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekt entsprechend dem/den oben beschriebenen Verfahren durchzuführen.

[0016] Ein weiterer Aspekt der vorliegenden Erfindung betrifft ein Computersystem zum Online-Analytical-Processing mit zugehörigem Datenspeichermittel, auf dem ein multidimensionales Objekt gespeichert wird, wobei das multidimensionale Objekt folgendes einschließt:
einen Faktensatz, der eine Anzahl von Fakten umfaßt,
eine erste Anzahl von Dimensionen mit Dimensionswerten, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, und die eine erste Abbildung von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension der ersten Anzahl von Dimensionen und ebenso Verknüpfungen zwischen den Fakten und den Dimensionen der ersten Anzahl von Dimensionen umfaßt, wobei wenigstens eine der Dimensionen der ersten Anzahl von Dimensionen irregulär ist, und
eine zweite Anzahl von Dimensionen mit Dimensionswerten, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, und die eine zweite Abbildung von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension der zweiten Anzahl von Dimensionen und ebenso Verknüpfungen zwischen den Fakten und den Dimensionen der zweiten Anzahl von Dimensionen umfaßt, wobei jede Dimension der zweiten Anzahl von Dimensionen aggregations-normalisiert ist,
wobei das Computersystem eine Anfragehandlerkomponente umfaßt, die geeignet ist, Antworten auf an das Computersystem gestellte und das multidimensionale Objekt betreffende Anfragen zu erstellen, wobei Antworten auf Navigationsanfragen auf dem ersten Satz von Dimensionen basieren und Antworten auf Aggregationsanfragen auf dem zweiten Satz von Dimensionen basieren. Dabei wird die Transparenz gegenüber dem Anwender und die Integrierbarkeit mit bekannten Systemen im allgemeinen erreicht. Des weiteren kann ein die zweite Anzahl von Dimensionen betreffender Satz von Vorab-Aggregationsdaten auf dem Datenspeichermittel gespeichert werden und die Antworten zu Aggregationsanfragen basieren des weiteren auf dem Satz von Vorab-Aggregationsdaten.

[0017] Die Anfragehandlerkomponente kann gemäß dem Computersystem zu einem weiteren Aspekt geeignet sein, Antworten auf Aggregationsanfragen zu erzeugen, wobei in den Antworten die Existenz der zweiten Anzahl von Dimensionen nicht sichtbar ist. Die Anfragehandlerkomponente kann des weiteren geeignet sein, Aggregationsanfragen, die an die erste Anzahl von Dimensionen gestellt wurden, in Anfragen an den zweiten Satz von Dimensionen umzuwandeln und Antworten, die auf dem zweiten Satz von Dimensionen basieren, in Antworten, wie solche, die auf dem ersten Satz von Dimensionen basieren, umzuwandeln, und somit die Existenz der zweiten Anzahl von Dimensionen in der erstellten Antwort nicht sichtbar zu machen. Zusätzlich kann das multidimensionale Objekt auf dem Datenspeichermittel des Computersystems in Tabellen gespeichert sein, die für den Teil des multidimensionalen Objekts, der nur strenge Abbildungen enthält, als eine Kombination von Star-Schemen organisiert sind, und zusätzlichen Tabellen, die den nicht-strengen Teil der Abbildungen enthalten, wobei die Anfragehandlerkomponente die Tabellen zur Umwandlung der Anfragen und Antworten verwendet.

[0018] Das Computersystem kann nach einem weiteren Aspekt der vorliegenden Erfindung geeignet mit dem/den oben beschriebenen Verfahren kombiniert werden und Mittel umfassen, die geeignet sind, eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekts entsprechend dem/der Verfahren durchzuführen.

Beschreibung der bevorzugten Ausführungsform

[0019] Die Erfindung wird in erster Linie als ein Verfahren beschrieben, um nichtsummierbare Hierarchien in multidimensionalen Datenbanken summierbar zu machen. Ein Fachmann wird jedoch erkennen, daß ein Gerät, wie etwa ein Datenverarbeitungssystem, das eine CPU, Speicher, I/O, Programmspeicher, einen Verbin-

dungsbus und andere geeignete Komponenten umfaßt, programmiert oder anderweitig ausgestaltet sein kann, um die Ausführung des Verfahrens der Erfindung zu ermöglichen. Solch ein System würde zur Ausführung des Verfahrens der Erfindung geeignete Pogrammmittel enthalten.

[0020] Ebenso kann ein Fabrikationserzeugnis zur Verwendung mit einem Datenverarbeitungssystem, wie etwa eine vorbeschriebene Festplatte oder andere ähnliche Computerprogrammprodukte, ein Speichermedium und darauf gespeicherte Programmmittel zur Steuerung des Datenverarbeitungssystems umfassen, um die Ausführung des Verfahrens der Erfindung zu ermöglichen. Solche Geräte und Fabrikationserzeugnisse fallen auch in den Anwendungsbereich der Erfindung.

Detaillierte Beschreibung der Erfindung

[0021] Die Erfindung wird jetzt im Detail beschrieben. Der nächste Abschnitt stellt eine reale klinische Fallstudie dar, die die nicht-summierbaren Eigenschaften realer Anwendungen beispielhaft zeigt. Der nächste Abschnitt fährt mit der Definition der Bestandteile eines multidimensionalen Datenmodells fort, die zur Beschreibung der neuen Technik notwendig sind und er definiert ebenso die Summierbarkeit betreffende wichtige Eigenschaften. Es werden Algorithmen zur Umwandlung von Dimensionshierarchien zum Erreichen der Summierbarkeit vorgestellt. Die Algorithmen werden dann zur Behebung nicht-summierbarer Beziehungen zwischen Fakten und Dimensionen angewendet. Es wird auch veranschaulicht, wie die Techniken für den Anwender transparent in derzeitige Systeme integriert werden können und wie die Algorithmen verändert werden, um eine inkrementelle Berechnung zu ermöglichen.

Kurze Beschreibung der Figuren

[0022] Zur detaillierten Beschreibung der Erfindung gehört ein Satz von Figuren, in denen

[0023] **Fig.** 1 ein ER-Diagramm zeigt, das die der Fallstudie zugrundeliegenden Daten darstellt,

[0024] **Fig.** 2 die Dimensionstypen der Fallstudie zeigt,

[0025] **Fig.** 3 zur Linken die durch den Abdeckendmachungs-Algorithmus auf die Hierarchie angewandten Umwandlungen und zur Rechten die durch den Surjektivmachungs-Algorithmus auf die Hierarchie angewandten Umwandlungen zeigt,

[0026] **Fig.** 4 die durch den Strengmachungs-Algorithmus auf die Hierarchie angewandten Umwandlungen zeigt,

[0027] **Fig.** 5 zur Linken ein weiteres Beispiel der durch den Surjektivmachungs-Algorithmus auf die Hierarchie angewandten Umwandlungen und zur Rechten die danach ausgeführten, durch den Strengmachungs-Algorithmus auf die Hierarchie angewandten Umwandlungen zeigt,

[0028] **Fig.** 6 zur Linken die Architektur eines gängigen OLAP-Systems und zur Rechten die Architektur der vorliegenden Erfindung zeigt, und

[0029] **Fig.** 7 die Implementierung der Systemarchitektur zeigt.

Motivation – eine Fallstudie

[0030] Dieser Abschnitt stellt eine Fallstudie vor, die die Eigenschaften realer Dimensionshierarchien veranschaulicht. Die Fallstudie betrifft Patienten in einem Krankenhaus, ihre zugehörigen Diagnosen und ihre Wohnsitze. Das Ziel der Datenanalyse ist herauszufinden, ob einige Diagnosen in einigen Gebieten häufiger auftreten als in anderen, in welchem Fall Umwelt- oder Lebensstilfaktoren zum Krankheitsbild beitragen könnten. Ein ER-Diagramm, das die zugrundeliegenden Daten darstellt ist in **Fig.** 1 zu sehen.

[0031] Die wichtigste Entität sind die Patienten, für die der Name gespeichert wird. Wir wollen immer die Anzahl von Patienten zählen, gruppiert nach bestimmten Eigenschaften der Patienten. Daher sind multidimensional ausgedrückt die Patienten die Fakten und die anderen, beschreibenden Entitäten bilden die Dimensionen.

[0032] Jeder Patient hat eine Anzahl von Diagnosen, was zu einer viele-zu-viele Beziehung zwischen den Fakten und der Diagnosedimension führt. Wenn Diagnosen von Patienten aufgenommen werden, verwenden Ärzte verschiedene Genauigkeitsebenen, die von sehr genauen Diagnosen, beispielsweise "Insulinabhängiger Diabetes während der Schwangerschaft", bis zu ungenaueren Diagnosen, beispielsweise "Diabetes", reichen und die große Bandbreiten von Patientenzuständen abdecken. Um dies abzubilden, ist die Beziehung von Patienten zu Diagnosen vom Obertyp "Diagnose", die drei Untertypen hat, die den verschiedenen Genauigkeitsebenen entsprechen, die Diagnose der niedrigen Ebene, die Diagnosefamilie und die Diagnosegruppe. Beispiele für diese sind jeweils "Insulinabhängiger Diabetes während der Schwangerschaft", "Insulinabhängiger Diabetes" und "Diabetes". Die Diagnosen der höheren Ebenen sind beide (ungenaue) Diagnosen aus eigenem Recht, dienen aber auch als Gruppen für die Diagnosen der niedrigeren Ebene.

[0033] Jede Diagnose hat einen alphanumerischen Code und einen beschreibenden Text, wobei die Texte durch einen bestimmten Standard festgelegt sind, hier die World Health Organisation's International Classifi-

cation of Diseases (ICD-10) [31], oder durch die Ärzte selber. Tatsächlich werden zwei Hierarchien aufgenommen: die durch die WHO festgelegte Standard-Hierarchie und die benutzerdefinierte Hierarchie, die zur Gruppierung von Diagnosen auf einer für diesen Zweck gegebenen Grundlage in anderer als der durch den Standard gegeben Weise dient. Das Typ-Attribut der Beziehung bestimmt, ob die Beziehung zwischen zwei Entitäten Teil des Standards oder der Benutzerdefinierten Hierarchie ist.

[0034] Die Hierarchie gruppiert Diagnosen der niedrigen Ebene in Diagnosefamilien, von denen jede aus 2–20 zugehörigen Diagnosen besteht. Zum Beispiel ist die Diagnose "Insulinabhängiger Diabetes während der Schwangerschaft" (der Grund dafür, daß es eine getrennte Diagnose im Zusammenhang mit einer Schwangerschaft gibt, ist, daß Diabetes während einer Schwangerschaft besonders intensiv überwacht und kontrolliert werden muß, um eine gute Gesundheit von Mutter und Kind zu garantieren) Teil der Familie "Diabetes während der Schwangerschaft". In der WHO-Hierarchie gehört eine Diagnose der niedrigen Ebene zu genau einer Familie, während es in der benutzerdefinierten Hierarchie diese Einschränkung nicht gibt. Daher kann eine Diagnose der niedrigen Ebene zu mehreren Diagnosefamilien gehören, beispielsweise gehört die „Insulinabhängiger Diabetes während der Schwangerschaft"-Diagnose sowohl zur "Diabetes während der Schwangerschaft"- wie auch zur "Insulinabhängiger Diabetes"-Familie. Als nächstes sind Diagnosefamilien in Diagnosegruppen gruppiert, die aus 2–10 Familien bestehen, und eine Familie kann in mehreren Gruppen sein. Beispielsweise kann die Familie "Diabetes während der Schwangerschaft" in der "Diabetes"- und der "Andere Krankheiten im Zusammenhang mit Schwangerschaft"-Gruppe sein.

[0035] In der WHO-Hierarchie gehört eine Familie zu genau einer Gruppe. In der WHO-Hierarchie gehört ein Wert einer niedrigeren Ebene zu genau einem Wert der höheren Ebene, wodurch sie streng und abdeckend wird. In der benutzerdefinierten Hierarchie kann ein Wert einer niedrigeren Ebene zu keinem oder mehreren Werten der höheren Ebene gehören, wodurch sie nicht-streng und nicht-abdeckend wird.

[0036] Eigenschaften der Hierarchien werden detaillierter im Abschnitt "Hierarchieeigenschaften" besprochen.

[0037] Es werden auch die Adressen der Patienten aufgenommen. Falls die Adresse in einer Stadt liegt, wird die Stadt aufgenommen; anderenfalls, wenn die Adresse in einem ländlichen Gebiet ist, wird der Verwaltungsbezirk aufgenommen, in dem die Adresse liegt. Eine Stadt liegt in genau einem Verwaltungsbezirk. Da nicht alle Adressen in Städten sind, können nicht alle Adressen eines Verwaltungsbezirks gefunden werden, indem man über die "Stadt liegt in"-Beziehung geht. Daher ist die Abbildung von Adressen auf Städte nicht-abdeckend in Bezug auf die Adressen. Für Städte und Verwaltungsbezirke wird nur der Namen aufgenommen. Nicht alle Verwaltungsbezirke haben Städte in sich, wodurch die Abbildung von Städten zu Verwaltungsbezirken in statt auf ist.

[0038] Um die Daten zu veranschaulichen, verwenden wir eine Standardabbildung der ER-Diagramms auf relationale Tabellen, das heißt eine Tabelle pro Entität und Beziehungstyp. Es werden auch Ersatz-Schlüssel, ID genannt, mit global eindeutigen Werten verwendet. Die drei Untertypen des Diagnosetyps werden auf eine gemeinsame Diagnosetabelle abgebildet und daher werden die "gehört zu"-und "Gruppierungs"-Beziehungen auf eine gemeinsame "Gruppierungs"-Tabelle abgebildet. Die resultierenden Tabellen mit Beispieldaten sind in Tabelle 1 gezeigt und werden in Beispielen im ganzen Schriftstück verwendet.

[0039] Wenn wir die Vorab-Aggregation auf die Daten der Fallstudie anwenden, treten mehrere Probleme auf. Werden beispielsweise die Patientenanzahlen pro Stadt vorab berechnet und diese zur Berechnung der Patientenanzahl in den Verwaltungsbezirken verwendet, tritt ein falsches Ergebnis auf. In den Daten liegen die Adressen "123 Rural Road" und "1 Sandy Dunes" (eine von diesen ist eine Patientenadresse) in keiner Stadt, wodurch die Abbildung von Städten auf Verwaltungsbezirke nicht-abdeckend in Bezug auf die Adressen wird.

[0040] Werden als nächstes die Patientenanzahlen für Diagnosen der niedrigsten Ebene vorab berechnet und diese zur Berechnung der Gesamtanzahl der Patienten verwendet, tritt wieder ein falsches Ergebnis auf. Erstens werden Patienten, die nur einen Lungentumor haben, nicht gezählt, da Lungentumor auf der Ebene der Diagnosen der niedrigen Ebene vorhanden ist; die Abbildung von Diagnosen der niedrigen Ebene auf Diagnosefamilien ist in. Zweitens haben Patienten wie etwa "Jim Doe" nur Diagnosen höherer Ebenen und werden nicht gezählt; die Fakten-zu-Dimensionen Abbildung hat variierende Genauigkeit. Drittens haben Patienten wie etwa "Jane Doe" mehrere Diagnosen und werden mehrfach gezählt; die Beziehung zwischen Fakten und Dimensionen ist viele-zu-viele. Viertens gehören Diagnosen der niedrigen Ebene wie "Insulinabhängiger Diabetes während der Schwangerschaft" zu mehreren Diagnosefamilien, was ebenso zu Doppelzählungen führen kann, wenn die Anzahlen auf den höheren Ebenen berechnet werden; die Dimensionshierarchie ist nicht-streng.

[0041] Diese Probleme ergeben "nicht-summierbare" Dimensionshierarchien, die die Anwendbarkeit der praktischen Vorab-Aggregation stark einschränken, wodurch nur die volle Vorab-Aggregation, die sehr große Speichermengen benötigt, oder keine Vorab-Aggregation, was zu langen Antwortzeiten auf Anfragen führt, bleibt.

ID	Name
1	John Doe
2	Jane Doe
3	Jim Doe

Patient

PatientID	AdressID
1	50
2	51
3	52

wohnhaft in

ID	Adresse
50	21 Central Street
51	34 Main Street
52	123 Rural Road
53	1 Sandy Dunes

Adresse

ID	Name
20	Sydney
21	Melbourne

Stadt

PatientID	DiagnoseID	Typ
1	9	primär
2	5	sekundär
2	9	primär
3	11	primär

Hat

ElternID	KindID	Typ
4	5	WHO
4	6	WHO
9	5	benutzerdefiniert
10	6	benutzerdefiniert
11	9	WHO
11	10	WHO
12	4	WHO
13	14	WHO

Gruppierung

ID	Code	Text	Typ
4	O24	Diabetes während Schwangerschaft	Familie
5	O24.0	Insulinabhängige Diabetes während Schwangerschaft	niedrige Ebene
6	O24.1	Nicht insulinabhängige Diabetes während Schwangerschaft	niedrige Ebene
9	E10	Insulinabhängige Diabetes	Familie
10	E11	Nicht insulinabhängige Diabetes	Familie
11	E1	Diabetes	Gruppe
12	O2	Andere Krankheiten im Zusammenhang mit Schwangerschaft	Gruppe
13	A1	Krebs	Gruppe
14	A11	Lunkenkrebs	Familie

Diagnose

AdressID	StadtID
50	20
51	21

liegt in Stadt

ID	Name
30	Sydney
31	Melbourne
32	Outback

Verwalt.bezirk

ID	Name
52	31
53	32

liegt in ländlichem Gebiet

StadtID	VerwBezID
20	30
21	31

Stadt liegt in

Tabelle 1
Tabellen zur Fallstudie

[0042] Die oben beschriebenen Eigenschaften sind in vielen anderen realen Anwendungen anzutreffen. Viele-zu-viele Beziehungen zwischen Fakten und Dimensionen treten zwischen Bankkunden und Konten, zwischen Firmen und Standard Industry Classifications (SIC) und zwischen Studenten und Fachbereichen auf [15, 16]. Nicht-strenge Dimensionshierarchien treten von Städten zu Staaten in einer Geographiedimension auf [25] und von Wochenenden zu Monaten in einer Zeitdimension. Außerdem sind Hierarchien, in denen die zeitliche Veränderung festgehalten wird im allgemeinen nicht-streng. Die Abbildung von Feiertagen zu Wochen ebenso wie Organisationshierarchien veränderlicher Tiefe [12] bieten Beispiele für "in"-Abbildungen. Nicht-abdeckende Beziehungen existieren für Tage-Feiertage-Wochen und für Verwaltungsbezirke-Städte-Staaten und ebenso für Organisationshierarchien (12).

[0043] Obwohl viele reale Fälle die oben beschriebenen Eigenschaften besitzen, erfordern derzeitige Techniken zur praktischen Vorab-Aggregation, daß Fakten in viele-zu-viele Beziehungen zu Dimensionen stehen und daß alle Hierarchien streng, auf und abdeckend sind. Daher können derzeitige Techniken nicht angewendet werden, wenn die Hierarchien diese Eigenschaften haben.

Verfahrenskontext

[0044] Dieser Abschnitt definiert die Bestandteile eines multidimensionalen Datenmodells, die für die Defini-

tion von Techniken notwendig sind, die die praktische Vorab-Aggregation in Anwendungen wie der gerade beschriebenen ermöglichen. Das vollständige Modell ist anderenorts beschrieben [22]. Als nächstes wird der Datenmodellkontext verwendet, um die für die Techniken wichtigen Eigenschaften von Hierarchien zu definieren.
[0045] Das bestimmte Datenmodell wurde statt anderer multidimensionaler Datenmodelle ausgewählt, weil es explizite Konzepte von Dimensionen und Dimensionshierarchien umfaßt, was für eine klare Darstellung der Techniken sehr wichtig ist. Die Techniken sind jedoch auch auf andere multidimensionale oder statistische Datenmodelle anwendbar, was im Abschnitt "Architekturkontext" besprochen wird.

Ein konkreter Datenmodellkontext

[0046] Für jeden Teil des Modells wird die Intension und die Extension definiert und es werden veranschaulichende Beispiele gegeben.
[0047] Ein n-dimensionales Faktenschema ist ein Zwei-Tupel s = (F,D), wobei F ein Fakttyp und D = {T_i, i = 1,...n} seine zugehörigen Dimensionstypen sind.

Beispiel 1

[0048] In der Fallstudie ist Patient der Fakttyp und Diagnose, Wohnsitz und Name sind die Dimensionstypen. Die Vorstellung ist, daß alles was einen Fakttyp beschreibt als Dimension aufgefaßt wird.
[0049] Ein Dimensionstyp T ist ein Vier-Tupel (c, $\leq_T$, $\lceil_T$, $\lfloor_T$), wobei C = {C_j, j = 1,...,k} die Kategorientypen von T sind, $\leq_T$ ist eine teilweise Sortierung auf den C_j, wobei $\lceil_T \in$ c und $\lfloor_T \in$ c das oberste beziehungsweise unterste Element der Sortierung ist. Die Kategorien bilden daher ein Gitter. Die Vorstellung ist, daß ein Kategorietyp "größer als" ein anderer Kategorietyp ist, falls Elemente der Erweiterung des Ersteren logisch Elemente der Erweiterung des Letzteren beinhalten, das heißt sie haben eine größere Wertgröße. Das oberste Element der Sortierung entspricht der größtmöglichen Wertgröße, das heißt es gibt nur einen Wert in seiner Erweiterung, der logisch alle anderen Werte beinhaltet.
[0050] Wir sagen, daß c_j eine Kategorie vom Typ T ist, in Zeichen $c_j \in$ T falls $C_j \in$ C.

Beispiel 2

[0051] Diagnosen der unteren Ebene sind in Diagnosefamilien enthalten, die wiederum in Diagnosegruppen enthalten sind. Der Diagnose-Dimensionstyp hat die folgende Sortierung auf seinen Kategorietypen: $\lfloor_{Diagnose}$ = Diagnosen der niedrigen Ebene<Diagnosefamilie<Diagnosegruppe<$\lceil_{Diagnose}$. Andere Beispiele von Kategorietypen sind Adresse, Stadt und Verwaltungsbezirk. **Fig.** 2, die später genauer besprochen wird, stellt die Dimensionstypen der Fallstudie dar.
[0052] Eine Kategorie C_j vom Typ c_j ist ein Satz von Dimensionswerten e. Eine Dimension D vom Typ T = ({c_j}, $\leq_T$, $\lceil_T$, $\lfloor_T$), ist ein Zwei-Tupel D = (C,≤), wobei C = {C_j} ein Satz von Kategorien C_j ist so, daß Type(C_j) = c_j und ≤ ist eine teilweise Sortierung auf $\cup_j C_j$, der Vereinigung aller Dimensionswerte in den einzelnen Kategorien. Sei Pred: C → 2^c eine Funktion, die den Satz der direkten Vorgänger einer Kategorie C_j gibt. Sei in ähnlicher Weise Desc: C → 2^c eine Funktion, die den Satz der direkten Nachfolger einer Kategorie C_j gibt. Sowohl für Pred als auch für Desc "zählen" wir ab der Kategorie $\lceil_T$ (vom Typ $\lceil_T$), so daß die Kategorie $\lceil_T$ keinen Vorgänger hat und die Kategorie $\lfloor_T$ (vom Typ $\lfloor_T$) keine Nachfolger hat. In ähnlicher Weise sei ImmPred: C → 2^c eine Funktion, die nur die unmittelbaren Vorgänger einer Kategorie C_j ergibt. Unmittelbare Vorgänger sind in der Hierarchie unmittelbar darüberliegend, das heißt Stadt ist ein unmittelbarer Vorgänger von Adresse, aber Verwaltungsbezirk ist es nicht. In ähnlicher Weise sei ImmDesc: C → 2^c eine Funktion, die die unmittelbaren Nachfolger einer Kategorie C_j gibt.
[0053] Die Definition der teilweisen Sortierung ist: gegeben zwei Werte e_1,e_2, dann ist $e_1 \leq e_2$,, falls e_1 logisch in e_2 enthalten ist. Wir sagen, daß C_j eine Kategorie von D ist, in Zeichen $C_j \in$ D falls $C_j \in$ C. Für einen Dimensionswert e sagen wir, daß e ein Dimensionswert von D ist, in Zeichen e∈D, falls e $\in \cup_j C_j$,
[0054] Die Kategorie vom Typ $\lfloor$T in der Dimension vom Typ T enthält die Werte mit der kleinsten Wertgröße. Die Kategorie mit der größten Wertgröße, vom Typ $\lceil_T$, enthält genau einen Wert, mit $\lceil$ bezeichnet. Für alle Werte e der Dimension D ist e ≤$\lceil$. Der Wert $\lceil$ ist ähnlich dem ALL-Konstrukt in Gray et al. [6]. Wenn der Kontext klar ist, bezeichnen wir eine Kategorie vom Typ $\lceil_T$ als eine $\lceil$-Kategorie, nicht zu verwechseln mit dem $\lceil$-Dimensionswert.

Beispiel 3

[0055] in unserer Diagnosedimension gibt es die folgenden Kategorien, benannt nach ihrem Typ. Die Nummern in Klammern sind ihre ID-Werte aus der Diagnosetabelle in Tabelle 1. Diagnosen niedriger Ebene = {"Insulinabhängiger Diabetes während einer Schwangerschaft" (5), "Nicht-insulinabhängiger Diabetes während ei-

ner Schwangerschaft" (6)}, Diagnosefamilie = {"Diabetes während der Schwangerschaft" (4), "Insulinabhängiger Diabetes" (9), "Nicht-insulinabhängiger Diabetes" (10), "Lungenkrebs" (14)}, Diagnosegruppe = {"Diabetes" (11), "Andere mit einer Schwangerschaft zusammenhängende Krankheiten" (12), "Krebs" (13)}, und $\top_{Diagnose} = \{\top\}$. Es gilt Pred(Diagnose niedriger Ebene) = {Diagnosefamilie}. Die teilweise Sortierung ≤ wird durch eine Kombination der WHO-Hierarchie mit benutzerdefinierten Hierarchien erhalten, so wie durch die Gruppierungstabelle in Tabelle 1 gegeben. Zusätzlich ist der oberste Wert $\top$ größer als alle anderen Diagnosewerte, d. h. er enthält sie logisch.

[0056] Sei F ein Satz von Fakten und D = (C = $\{C_j\}$, ≤) eine Dimension. Eine Fakt-Dimensions-Relation zwischen F und D ist eine Menge R = {(f,e)}, wobei f ∈ F und e $\in \cup_j C_j$. R verbindet demnach Fakten mit Dimensionswerten. Wir sagen, daß der Fakt f durch den Dimensionswert e beschrieben wird, in Zeichen f ⇝ e, falls $\exists e_1 \in D$ ((f,e_1) ∈ R ∧ $e_1 \leq e$). Wir fordern, daß $\forall f \in F(\exists e \in \cup_j C_j$ ((f,e) ∈ R)); somit werden alle Fakten zu wenigstens einem Dimensionswert in jeder Dimension. Der $\top$-Wert wird verwendet, um einen unbekannten, fehlenden Wert darzustellen, da $\top$ logisch alle Dimensionswerte enthält und somit wird Fakt f auf $\top$ abgebildet, falls er in der bestimmten Dimension nicht beschrieben werden kann.

Beispiel 4

[0057] Die Fakt-Dimensions-Relation R verbindet Patientenfakten mit Diagnose-Dimensionswerten, wie durch die Hat-Tabelle aus der Fallstudie gegeben, so daß R = {("John Doe" (1), "Insulinabhängiger Diabetes" (9), ("Jane Doe" (2), "Insulinabhängiger Diabetes während einer Schwangerschaft" (5)), ("Jane Doe" (2), "Insulinabhängiger Diabetes" (9)), ("Jim Doe" (3), "Diabetes" (11))}. Es ist zu bemerken, daß Fakten mit Werten in Kategorien höherer Ebenen verbunden werden können. Wir fordern nicht, daß e zu $\top_{Diagnose}$ gehört. Beispielsweise ist der Fakt "John Doe" (1) mit der Diagnose "Insulinabhängiger Diabetes" (5) verbunden, die zur Diagnosefamilie-Kategorie gehört. Diese Eigenschaft wird später verwendet werden, um explizit die verschiedenen Genauigkeiten in den Daten aufzunehmen. Falls keine Diagnose für den Patienten "John Doe" (1) bekannt wäre, würden wir das Paar ("John Doe" (1), $\top$) zu R hinzufügen.

[0058] Ein multidimensionales Objekt (MO) ist ein Vier-Tupel M = (s,F,D,R), wobei s = (F,D = $\{T_i\}$) das Faktschema ist, F = {f} ist ein Satz von Fakten f, wobei Type(f) = F, D = $\{D_i$, i = 1,...,n} ist ein Satz von Dimensionen, wobei Type(D_i) = T_i, und R = $\{R_i$, i = 1,...,n} ist ein Satz von Fakt-Dimensions-Relationen derart, daß $\forall i((f,e) \in R_i \Rightarrow f \in F \wedge \exists C_j \in D_i(e \in C_j))$.

Beispiel 5

[0059] Für die Fallstudie erhalten wir ein dreidimensionales MO M = (s,F,D,R), wobei S = (Patient, {Diagnose, Name, Wohnsitz}) und F = {"John Doe" (1), "Jane Doe" (2), "Jim Doe" (3)}. Die Definition der Diagnosedimension und ihrer zugehörigen Fakt-Dimensions-Relation wurde in den vorherigen Beispielen gegeben. Die Wohnsitz-Dimension hat die Kategorien Adresse (= $\bot_{Wohnsitz}$), Stadt, Verwaltungsbezirk und $\top_{Wohnsitz}$. Die Werte der Kategorien sind durch die entsprechenden Tabellen in Tabelle 1 gegeben. Die teilweise Sortierung ist durch die Beziehungstabellen gegeben. Zusätzlich ist der einzige Wert in der $\top_{Wohnsitz}$-Kategorie $\top$, das logisch alle anderen Werte der Wohnsitz-Dimension enthält. Die Name-Dimension ist einfach, das heißt sie hat nur eine Name Kategorie (= $\bot_{Name}$) und eine $\top$-Kategorie. Wir beziehen uns auf diese MO als die "Patienten"-MO. Eine graphische Veranschaulichung des Schemas der "Patienten"-MO ist in **Fig.** 2 zu sehen. Da einige Adressen direkt auf Verwaltungsbezirke abgebildet werden, ist Verwaltungsbezirk ein unmittelbarer Vorgänger von Adresse.

[0060] Die Fakten in einer MO sind Objekte mit wert-unabhängiger Identität. Wir können Fakten auf Gleichheit testen, aber keine Sortierung auf den Fakten haben. Die Kombination von Dimensionswerten, die die Fakten eines Faktensatzes beschreiben ist kein "Schlüssel" für den Faktensatz. Daher können mehrere Fakten durch dieselbe Kombination von Dimensionswerten beschrieben sein. Die Fakten eines MO sind jedoch ein Satz. Somit hat ein MO keine doppelten Fakten. Das Modell definiert formal recht allgemeine Konzepte von Dimensionen und Dimensionshierarchien, was für die Darstellung unserer Techniken ideal ist. Die vorgestellten Techniken sind nicht durch die Wahl des Datenmodells eingeschränkt.

Hierarchieeigenschaften

[0061] In diesem Abschnitt werden wichtige Eigenschaften, die die Verwendung vorab berechneter Aggregate betreffen, definiert. Die Eigenschaften werden in den folgenden Abschnitten dafür verwendet genau anzugeben, welche Probleme die vorgeschlagenen Algorithmen lösen. Das erste wichtige Konzept ist die Summierbarkeit, was intuitiv bedeutet, daß Aggregate höherer Ebenen direkt aus Aggregaten niedrigerer Ebenen erhalten werden können.

Definition 1

[0062] Gegeben sei ein Typ T, ein Satz S = {S_j, j = 1,...,k}, wobei $S_j \in 2^T$, und eine Funktion $g:2^T \rightarrow T$, wir sagen, daß g für S summierbar ist, wenn $g(\{\{g(S_1),...,g(S_k)\}\}) = g(S_1 \cup ... \cup S_k)$. Der Ausdruck auf der linken Seite der Gleichung ist ein Multisatz, das heißt derselbe Wert kann mehrfach auftreten.
[0063] Summierbarkeit ist wichtig als eine Bedingung für den flexiblen Einsatz vorab berechneter Aggregationen. Ohne Summierbarkeit können Ergebnisse niedrigerer Ebenen im allgemeinen nicht direkt in Ergebnisse höherer Ebenen eingearbeitet werden. Das bedeutet, daß wir nicht lediglich eine relevanten Auswahl der möglichen Aggregate berechnen und dann diese dazu (effizient) verwenden können, Aggregate höherer Ebenen im Vorbeigehen zu berechnen. Statt dessen müssen alle Aggregatergebnisse von Anfragen, für die wir schnelle Antworten benötigen, vorab berechnet werden, während andere Aggregate ausgehend von den Basisdaten zu berechnen sind. Platz- und Zeitbeschränkungen können eine Vorabberechnung aller Ergebnisse verbieten, während die Berechnung von Aggregaten aus den Basisdaten oft ineffizient ist.
[0064] Es wurde gezeigt, daß Summierbarkeit gleichbedeutend damit ist, daß die Aggregatfunktion (g) distributiv ist, alle Pfade streng sind und die Abbildungen zwischen Dimensionswerten in den Hierarchien abdeckend und auf sind [16]. Diese Konzepte sind untenstehend formal definiert. Die Definition geht von einer Dimension D = (C,≤) und einem MO M = (s,F,D,R) aus.

Definition 2

[0065] Gegeben seien zwei Kategorien, C_1, C_2, derart, daß $C_2 \in Pred(C_1)$, wir sagen die Abbildung von C_1 nach C_2 ist dann und genau dann auf, wenn $\forall e_2 \in C_2(\exists e_1 \in C_1\ (e_1 \leq e_2))$. Andernfalls ist sie in. Wenn alle Abbildungen in einer Dimension auf sind, sagen wir die Dimensionshierarchie ist auf. Gegeben ein MO M, falls alle Dimensionshierarchien in M auf sind, sagen wir M ist auf.
[0066] Abbildungen, die in sind, treten typischerweise auf, wenn die Dimensionshierarchie eine variierende Höhe hat. In der Fallstudie gibt es keine Krebsdiagnose auf der niedrigen Ebene, was bedeutet, daß einige Teile der Hierarchie die Höhe 2 haben, während die meisten die Höhe 3 haben. Es ist daher nicht möglich Aggregate auf der niedrigen Diagnoseebene zur Berechnung von Aggregaten auf den zwei höheren Ebenen zu verwenden. Ebenso treten Abbildungen, die in sind, häufig in Organisationshierarchien auf.

Definition 3

[0067] Gegeben seien drei Kategorien, C_1, C_2, C_3, derart, daß $Type(C_1) < Type(C_2) < Type(C_3)$, wir sagen die Abbildung von C_2 nach C_3 ist abdeckend bezüglich C_1, dann und genau dann, wenn $\forall e_1 \in C_1\ (\forall e_3 \in C_3(e_1 \leq e_3 \Rightarrow \exists e_2 \in C_2\ (e_1 \leq e_2 \wedge e_2 \leq e_3)))$. Andernfalls ist sie nicht-abdeckend bezüglich C_1. Sind alle Abbildungen in einer Dimension abdeckend bezüglich jeder Kategorie, sagen wir die Dimensionshierarchie ist abdeckend. Sind alle Dimensionshierarchien in den Dimensionen von M abdeckend und alle Abbildungen zwischen Kategorien in den Dimensionen von M bezüglich F abdeckend, sagen wir M ist abdeckend.
[0068] Nicht-abdeckende Abbildungen treten auf, wenn einige der Verknüpfungen zwischen Dimensionswerten eine oder mehrere Ebenen überspringen und direkt auf einen Wert abbilden, der höher in der Hierarchie liegt. In der Fallstudie passiert dies für die "1 Sandy Dunes" Adresse, die direkt auf "Outback-Verwaltungsbezirk" abgebildet wird (es gibt keine Städte im Outback-Verwaltungsbezirk). Daher können wir zur Berechnung von Aggregaten auf der Verwaltungsbezirk-Ebene keine Aggregate auf der Stadt-Ebene verwenden.

Definition 4

[0069] Gegeben sei ein MO M = (s,F,D,R) und zwei Kategorien C_1 und C_2, die zur selben Dimension $D_i \in D$ gehören, so daß $Type(C_1) < Type(C_2)$, wir sagen die Abbildung von C_1 nach C_2 ist abdeckend bezüglich F, dem Faktensatz, dann und genau dann, wenn $\forall f \in F(\forall e_2 \in C_2(f\ {}^{TM}_{i}\ e_2 \Rightarrow \exists e_1 \in C_1\ (f\ {}^{TM}_{i}\ e_1 \wedge e_1 \leq_i e_2)))$.
[0070] Dieser Fall ist ähnlich zum vorherigen Fall, jedoch sind es nun die Abbildungen zwischen den Fakten und Dimensionswerten, die eine oder mehrere Ebenen überspringen können und Fakten direkt auf Dimensionswerte in Kategorien über der untersten Ebene abbilden. In der Fallstudie können Patienten auf Diagnosen irgendwo in der Diagnosedimension abgebildet werden, nicht nur auf die Diagnosen der niedrigen Ebene. Dies bedeutet, daß wir Aggregate auf der niedrigen Diagnoseebene nicht zur Berechnung von Aggregaten weiter oben in der Hierarchie verwenden können.

Definition 5

[0071] Gegeben seien zwei Kategorien C_1 und C_2, so daß $C_2 \in Pred(C_1)$, wir sagen die Abbildung von C_1 nach C_2 ist streng, dann und genau dann, wenn $\forall e_1 \in C_1\ (\forall e_2, e_3 \in C_2\ (e_1 \leq e_2 \wedge e_1 \leq e_3 \Rightarrow e_2 = e_3))$. Andernfalls ist sie

nicht-streng. Die Hierarchie in der Dimension D ist streng wenn alle Abbildungen in ihr streng sind; andererseits ist sie nicht-streng. Gegeben sei ein MO M = (s,F,D,R) und eine Kategorie C_j in einer Dimension $D_i \in D$, wir sagen, daß es einen strengen Pfad vom Faktensatz F nach C_j gibt, dann und genau dann, wenn $\forall f \in F$ ($f \ ^{TM}_i \ e_1 \wedge f \ ^{TM}_i \ e_2 \wedge e_1 \in C_j \wedge e_2 \in C_j \Rightarrow e1 = e2$). (Es ist zu bemerken, daß die Pfade zur $\lceil_T$-Kategorie immer streng sind). Die Hierarchie in der Dimension D wird aggregations-streng genannt, wenn das folgende gilt: wenn eine Abbildung von Kategorie C zu einer Kategorie P, beide in D, nicht-streng ist, dann Pred(P) = φ, das heißt P hat keine Eltern in der Hierarchie.

[0072] Nicht-strenge Hierarchien treten auf, wenn ein Dimensionswert mehrere Eltern hat. Dies tritt in der Diagnosedimension in der Fallstudie auf, bei der die "Insulinabhängiger Diabetes während einer Schwangerschaft"-Diagnose der niedrigen Ebene sowohl zur "Insulinabhängiger Diabetes"- als auch zur "Diabetes während einer Schwangerschaft"-Diagnosefamilie gehört, die ihrerseits beide zur "Diabetes"-Diagnosegruppe gehören. Dies bedeutet, daß wir Aggregate auf der Diagnosefamilie-Ebene nicht zur Berechnung von Aggregaten auf der Diagnosegruppe-Ebene verwenden können, da Daten für "Insulinabhängiger Diabetes während einer Schwangerschaft" doppelt gezählt würden. Aggregationsstrenge Hierarchien gestatten es eine praktische Vorab-Aggregation anzuwenden, da die Werte in der Elternkategorie selber keine Eltern haben. Dies bedeutet, daß eine Doppelzählung von Daten nicht auftritt, da die Werte in der Elternkategorie, die möglicherweise überlappende Daten enthält, nicht zusammengezählt werden.

Definition 6

[0073] Wenn die Dimensionshierarchie für eine Dimension D auf, abdeckend und streng ist, sagen wir D ist normalisiert. Andernfalls ist sie un-normalisiert. Für ein MO M = (s,D,F,R), wenn alle Dimensionen $D_i \in D$ normalisiert sind und $\forall R_i \in R$ ($(f,e) \in R_i \Rightarrow e \in_{L_D}$) (d. h. alle Fakten werden auf Dimensionswerte in der untersten Kategorie abgebildet) sagen wir M ist normalisiert. Andernfalls ist es un-normalisiert. Wenn die Hierarchie in einer Dimension D abdeckend, auf und aggregations-streng ist, sagen wir D ist aggregations-normalisiert. Wenn alle Dimensionen in einem MO M aggregations-normalisiert sind und alle Beziehungen von Fakten zu Dimensionen viele-zu-eins und nur zur untersten Kategorie sind, sagen wir M ist aggreggations-normalisiert.

[0074] Für normalisierte Hierarchien und MOs sind alle Abbildungen summierbar, was bedeutet, daß Werte in jeder Kombination von Dimensionsebenen vorab-aggregiert werden können und die vorab-aggregierten Werte können gefahrlos zur Berechnung von Aggregationsergebnissen höherer Ebenen wiederverwendet werden. Daher wollen wir die Dimensionshierarchien und MOs, auf die wir die praktische Vorab-Aggregation anwenden wollen, normalisieren.

[0075] Wir beschreiben nun, wie die Normalisierung der für die Aggregation verwendeten Dimensionshierarchien und MOs erreicht wird. Wir zeigen zuerst, wie Umwandlungen auf Dimensionshierarchien ausgeführt werden, danach beschreiben wir, wie dieselben Techniken angewendet werden können, um die nicht-summierbaren Eigenschaften von Fakt-Dimensions-Relationen zu beseitigen.

Dimensions-Umwandlungstechniken

[0076] Dieser Abschnitt beschreibt, wie Dimensionen umgewandelt werden können, um Summierbarkeit zu erreichen. Die Umwandlungen von Dimensionen alleine, getrennt von den Fakten, ergibt sich gut verhaltende Dimensionen, die in einer Anzahl verschiedener Systeme anwendbar oder an Third-Party-Nutzer verkaufbar sind. Die Umwandlung der Dimensionshierarchien ist eine dreiphasige Operation. Zuerst werden alle Abbildungen durch das Einführen zusätzlicher "Zwischen"-Werte umgewandelt, um abdeckend zu sein. Zweitens werden alle Abbildungen durch das Einführen von "Platzhalter"-Werfen auf niedrigeren Ebenen für Werte ohne Kinder umgewandelt, um auf zu sein. Drittens werden die Abbildungen streng gemacht, indem Werte "vereinigt" werden. Die drei Phasen werden in getrennten Abschnitten behandelt. Keiner der Algorithmen führt nicht-summierbare Eigenschaften ein, daher ist es ausreichend jeden nur einmal anzuwenden.

[0077] Im Allgemeinen nehmen die Algorithmen als Eingabe einen Satz von Tabellen Rc_1,c_2, die die Abbildung von Dimensionswerten in der Kategorie C_1 auf Werte in der Kategorie C_2 festlegen. Die Eingabe braucht nicht alle Paare von Vorgängern und Nachfolgern zu beinhalten – nur direkte Eltern-Kind-Beziehungen sind erforderlich. Falls es nicht-abdeckende Abbildungen in der Hierarchie gibt, haben wir Kategorien C,P,H derart, daß $(P,H) \subseteq Pred(C)$ und $Type(P) < Type(H)$. In diesem Fall muß die Eingabe auch $R_{P,H}$-Tabellen beinhalten, die P-Werte auf H-Werte abbilden.

[0078] Die Algorithmen verwenden Rekursion. Sie können statt dessen auch leicht auf Iterationen umgeschrieben werden.

Nicht-Abdeckende Hierarchien

[0079] Der erste Algorithmus macht alle Abbildungen in einer Dimensionshierarchie bezüglich jeder Kategorie

abdeckend. Wenn ein Dimensionswert direkt auf einen anderen Wert in einer höheren Kategorie, als der in der Hierarchie unmittelbar darüberliegenden, abgebildet ist, wird ein neuer Zwischen-Wert in die unmittelbar darüberliegende Kategorie eingefügt und die zwei ursprünglichen Dimensionswerte werden zu diesem neuen Wert verknüpft anstatt miteinander.

Beispiel 6

[0080] In der Hierarchie für die Wohnsitz-Dimension gehen zwei Verknüpfungen von Adresse direkt zu Verwaltungsbezirk. Die Adresse "123 Rural Road" (52) ist in "Melbourne Verwaltungsbezirk" (31), aber nicht in einer Stadt, und die Adresse "1 Sandy Dunes" (53) ist in "Outback Verwaltungsbezirk" (32), der überhaupt keine Stadt hat. Der Algorithmus fügt zwei neue Dimensionswerte in die Stadt-Kategorie ein, C31 und C32, die Melbourne- beziehungsweise Outback-Verwaltungsbezirk repräsentieren, und verknüpft sie zu ihren jeweiligen Verwaltungsbezirken. Die Adressen "123 Rural Road" und "1 Sandy Dunes" werden dann mit C31 beziehungsweise C32 verknüpft. Dies geschieht beim ersten Aufruf der Abdeckendmachungs-Prozedur (MakeCovering) (auf der Adressen-Kategorie; die Prozedur wird weiter unten dargestellt). Wenn MakeCovering rekursiv auf der Stadt-, Verwaltungsbezirk- und ⌜-Kategorie aufgerufen wird, passiert nichts, da bereits alle Abbildungen abdeckend sind. Die Umwandlung ist graphisch in **Fig.** 3 auf der linken Seite veranschaulicht. Die gepunkteten Linien zeigen die "problematischen" Verknüpfungen und die fett gedruckten Werte und dicken Linien zeigen die neuen Dimensionswerte und Verknüpfungen.

[0081] Im Algorithmus ist C eine Kindkategorie, P ist eine unmittelbare Elternkategorie, H ist eine "höhere" Kategorie, L sind die nicht-abdeckenden Verknüpfungen von C nach H und N sind die "höheren" Dimensionswerte in L. Der ⊗-Operator bezeichnet die natürliche Hintereinanderschaltung. Der Algorithmus arbeitet wie folgt. Bei gegebener Eingabe-Kategorie C (anfänglich die unterste Kategorie) in Zeile (1) geht der Algorithmus durch alle unmittelbaren Elternkategorien P von C (2). Für jede unmittelbare Elternkategorie P sucht er nach Vorgängerkategorien H von C, die in der Hierarchie "höher" sind als P (4). Wenn solch ein N existiert, können Verknüpfungen in der Abbildung von C nach H vorhanden sein, die nicht verfügbar sind, wenn man über P geht. Zeile (6) findet diese "nicht-abgedeckten" Verknüpfungen, L, in der Abbildung von C nach H, indem es die Verknüpfungen, die verfügbar sind, wenn man über P geht, von allen Verknüpfungen in der Abbildung von C nach H "subtrahiert". Zeile (7) verwendet L, um die Dimensionswerte N in H zu finden, die an den "nicht-abgedeckten" Abbildungen beteiligt sind. Für jeden Wert in N, fügt Zeile (8) einen entsprechenden markierten Wert in P ein; diese markierten Werte repräsentieren die N Werte in P. Die markierten Werte in P werden dann mit den ursprünglichen Werten in H (9) und C (10) verknüpft. Zeile (12) ändert das Schema, so daß H nicht länger ein Vorgänger von C ist. Dies kann gefahrlos ausgeführt werden, da die Zwischenwerte und Zwischenverknüpfungen bedeuten, daß die Werte in H und C, die vorher verbunden waren immer noch über P verbunden sind. Linie (13) beinhaltet einen rekursiven Aufruf des Algorithmus P und korrigiert so Abbildungen weiter oben in der Hierarchie. Der Algorithmus terminiert, wenn er die ⌜-Kategorie erreicht, die keinen Vorgänger hat.

```
(1)  procedure MakeCovering(C)
(2)      for each P ∈ ImmPred(C) do
(3)      begin
(4)          for each H ∈ Pred(C) where Type(H) > Type(P) do
(5)          begin
(6)              L ← R_{C,H} \ Π_{C,H}(R_{C,P} ⊗ R_{P,H})
(7)              N ← Π_H(L)
(8)              P ← P ∪ {Mark(h) | h ∈ N}
(9)              R_{P,H} ← R_{P,H} ∪ {(Mark(h), h) | h ∈ N}
(10)             R_{C,P} ← R_{C,P} ∪ {(c, Mark(h)) | (c, h) ∈ L}
(11)         end
(12)         Pred(C) ← Pred(C) \ {H}
(13)         MakeCovering(P)
(14)     end
(15) end
```

[0082] Alle Schritte im Algorithmus sind durch übliche Operatoren aus der relationalen Algebra ausgedrückt. Die allgemeine Komplexität der Verbindung ist im ungünstigsten Fall $O(n^2)$, wobei n die Größe der Eingabe ist. Da die Eingabe des Algorithmus jedoch Hierarchiedefinitionen sind, ist die Komplexität der Verbindung im Algorithmus nur O(n log n). Alle verwendeten Operatoren können daher in der Zeit O(n log n) ausgeführt werden, wobei n die Größe der Eingabe ist. Die Mark-Operation kann in O(1) ausgeführt werden. Die innere Schleife des Algorithmus wird höchstens einmal für jede Verknüpfung zwischen Kategorien ausgewertet, das heißt höchstens $k^2/2$ mal, wobei k die Anzahl der Kategorien ist (wenn alle Kategorien direkt mit allen anderen verknüpft sind). Daher ist die gesamte Big-O-Komplexität des Algorithmus $O(k^2 n \log n)$, wobei k die Anzahl der Kategorien und n die Größe der größten beteiligten $R_{C1,C2}$-Relation ist. Die Komplexität des ungünstigsten Falls wird nicht sehr oft auftreten; in den meisten Fällen wird die innere Schleife höchstens k mal ausgewertet.

[0083] Der Algorithmus fügt neue Werte in die P-Kategorie ein, um sicherzustellen, daß Abbildungen von P zu höheren Kategorien summierbar sind, das heißt, daß für P vorab aggregierte Ergebnisse direkt in Aggregationsergebnisse höherer Ebenen eingearbeitet werden können. Die neuen Werte in P bedeuten, daß die Kosten zur Erzeugung von Aggregationsergebnissen für P mit der umgewandelten Hierarchie höher als für die ursprüngliche sind. Wäre die Hierarchie jedoch nicht zum Erreichen der Summierbarkeit umgewandelt, müßten Aggregate für G und eventuell auch für Kategorien höherer Ebenen erzeugt werden. Für jeden Wert in G wird in P höchstens ein neuer Wert eingefügt, was bedeutet, daß die zusätzlichen Kosten zum Erzeugen der Ergebnisse in P niemals größer als die Kosten des (andernfalls notwendigen) Erzeugens der Ergebnisse in G sind. Dies ist der sehr unwahrscheinliche ungünstigste Fall – in den gängigsten Fällen sind die zusätzlichen Kosten für P viel geringer als die Kosten zum Erzeugen von Ergebnissen für G und die Einsparungen sind noch größer, da auch das Erzeugen von Ergebnissen für Kategorien höherer Ebenen eingespart werden kann.

[0084] Die Korrektheitsaussage für den Algorithmus hat zwei Gesichtspunkte. Erstens sollten die Abbildungen in der Hierarchie bei der Terminierung abdeckend sein. Zweitens sollte der Algorithmus nur Umwandlungen ausführen, die semantisch korrekt sind, das heißt wir sollten bei der Berechnung von Ergebnissen mit der neuen und mit der alten Hierarchie dieselben Ergebnisse erhalten. Die Korrektheit folgt aus den folgenden Theoremen 1 und 2. Da neue Werte in die P-Kategorie eingefügt werden erhalten wir Aggregatwerte sowohl für die neuen als auch für die ursprünglichen Werte, wenn in P "gruppiert" wird. Ergebnisse für die ursprünglichen Werte sind dieselben wie zuvor, womit der ursprüngliche Ergebnissatz eine Untermenge der mit der umgewandelten Hierarchie erhaltenen Ergebnisse ist.

[0085] Theorem 1 Der Algorithmus MakeCovering terminiert und die Hierarchie der sich ergebenden Dimension D' ist abdeckend.

Beweis

[0086] Durch Induktion in der Höhe des Dimensionsgitters. Anfang: die Höhe ist 0 wodurch die Aussage trivialerweise wahr ist. Induktionsschritt: Wir nehmen an, die Aussage sei für Dimensionsgitter der Höhe n wahr und betrachten Gitter der Höhe n + 1. Zur Terminierung bemerken wir, daß es eine endliche Anzahl von (P,H)-Paaren gibt, daß alle Operatoren in der inneren Schleife terminieren und daß der Algorithmus rekursiv auf P aufgerufen wird, was die Wurzel eines Gitters der Höhe n ist. Zur Abdeckend-Eigenschaft bemerken wir, daß das Einfügen von markierten Zwischenwerten in P bedeutet, daß die Abbildung von P nach H abdeckend bezüglich C ist. Mit der Induktionshypothese werden die höher in der Hierarchie gelegenen Abbildungen durch den rekursiven Aufruf des Algorithmus korrigiert.

Theorem 2

[0087] Gegeben seien Dimensionen D und D; wobei D' das Ergebnis der Ausführung von MakeCovering auf D ist, dann ist ein unter Verwendung von D erhaltenes Aggregationsergebnis eine Untermenge des unter Verwendung von D` erhaltenen Ergebnisses.

Beweis

[0088] Folgt leicht aus dem nachfolgenden Lemma 1, da die eingefügten Werte "intern" in der Hierarchie sind.

Lemma 1

[0089] Für die Dimension D' = (C', ≤'), die sich aus einer Anwendung des MakeCovering-Algorithmus auf die Dimension D = (C,≤) ergibt, gilt das folgende: $\forall e_1,e_2 \in D\ (e_1 \leq' e_2 \Leftrightarrow e_1 \leq e_2)$ (zwischen zwei beliebigen ursprünglichen Dimensionswerten in der neuen Dimensionshierarchie gibt es dann und genau dann einen Pfad, wenn es zwischen ihnen einen Pfad in der ursprünglichen Hierarchie gab).

[0090] Beweis: Durch Induktion in der Höhe des Dimensionsgitters. Anfang: die Höhe ist 0 wodurch die Aussage trivialerweise wahr ist. Induktionsschritt: Wir nehmen an, die Aussage sei für Dimensionsgitter der Höhe n wahr und betrachten Gitter der Höhe n + 1. Beim Betrachten der inneren Schleife sehen wir, daß das Einfügen von Zwischenwerten in P und das Verknüpfen von Werten in C und H mit diesen nur Werte in C und H verknüpft, die bereits zuvor verknüpft waren. Keine Verknüpfungen oder Werte werden durch die innere Schleife zerstört. Daher ist die Aussage für Verknüpfungen von C nach P und von C nach N wahr. Mit der Induktionshypothese ist die Aussage für die durch den rekursiven Aufruf auf P durchgeführten Umwandlungen wahr.

[0091] Wir sehen, daß die ursprünglichen Werte in der Hierarchie weiterhin genau mit denselben ursprünglichen Werten wie zuvor verknüpft sind, wie durch Lemma 1 festgestellt, obwohl eventuell neue Werte zwischen den ursprünglichen Werten eingefügt wurden. Daher sind bei der Auswertung einer Anfrage unter Verwendung der umgewandelten Hierarchie die Ergebnisse für die ursprünglichen Werte dieselben wie bei Verwendung der ursprünglichen Hierarchie.

[0092] Unter der Annahme, daß nur der ursprüngliche Ergebnissatz gewünscht ist, müssen die Ergebnisse für die neuen Werte ausgeschlossen werden, was leicht zu erreichen ist. Die neuen "internen" Werte sind mit "mark=intern" markiert, wogegen die ursprünglichen Werte "mark= ursprünglich" haben. Um die neuen, internen Werte vom Ergebnissatz auszuschließen, wird die Entsprechung einer SQL HAVING-Bedingung mit "mark= ursprünglich" in die ursprüngliche Anfrage eingefügt.

Nicht-Auf-Hierarchien

[0093] Der zweite Algorithmus macht alle Abbildungen in Hierarchien auf, das heißt alle Dimensionswerte in nicht-untersten Kategorien haben Kinder. Dies wird durch das Einfügen von Platzhalterwerten in niedrigere Kategorien zur Repräsentation der Kindwerte erreicht. Diese neuen Werte werden mit den ursprüngliche Werten markiert, wodurch es möglich wird Fakten auf die neuen Platzhalterwerte anstatt auf die ursprünglichen Werte abzubilden. Dadurch können Fakten ausschließlich auf die unterste Kategorie abgebildet werden.

Beispiel 7

[0094] In der Diagnosedimension hat die "Lungenkrebs"-Diagnosefamile (ID = 14) keine Kinder. Wenn der Algorithmus die Diagnosefamilie erreicht, fügt er einen Platzhalterwert (L14) in die Diagnosekategorie der niedrigen Ebene ein, der die "Lungenkrebs"-Diagnose repräsentiert, und verknüpft ihn mit dem ursprünglichen Wert. Auf den "Lungenkrebs"-Wert abgebildete Fakten können dann statt dessen auf den neuen Platzhalterwert abgebildet werden, wodurch erreicht wird, daß Fakten nur auf Diagnosekategorien der niedrigen Ebene

abgebildet werden. Eine graphische Veranschaulichung der Umwandlung ist rechts in der **Fig.** 3 zu sehen. Der fett gedruckte L14-Wert ist der neue eingefügte Wert und die dicke Linie zwischen 14 und L14 ist die eingefügte neue Verknüpfung.

[0095] Im untenstehenden Algorithmus ist P eine Elternkategorie, C eine Kindkategorie und N beinhaltet die Elternwerte ohne Kinder. Der Algorithmus arbeitet wie folgt. Gegeben eine Kategorie P (anfänglich die $\lceil$-Kategorie): in Zeile (1) geht der Algorithmus durch alle Kategorien C, die (unmittelbare) Nachfolger von P sind (2). Für jedes C findet Zeile (4) die Werte N in P, die keine Kinder in C haben, indem die Werte mit Kindern in C von den Werten in P "subtrahiert" werden. Für jeden "kinderlosen" Wert in N fügen Zeile (5) beziehungsweise (6) in C einen Platzhalterwert ein, der mit dem Elternwert markiert wird, und verknüpfen den neuen mit dem ursprünglichen Wert. Die Surjektivmachungs-Prozedur (MakeOnto) wird dann rekursiv auf C aufgerufen (7). Der Algorithmus terminiert, wenn er die $\lfloor$-Kategorie erreicht, die keine Nachfolger hat.

```
(1)  procedure MakeOnto(P)
(2)      for each C ∈ Desc(P) do
(3)      begin
(4)          N ← P \ Π_P(R_{C,P})
(5)          C ← C ∪ {Mark(p) | p ∈ N}
(6)          R_{C,P} ← R_{C,P} ∪ {(Mark(p), p) | p ∈ N}
(7)          MakeOnto(C)
(8)      end
(9)  end
```

[0096] Der Argumentation im vorhergehenden Abschnitt folgend, finden wir, daß die gesamte Big-O-Komplexität $O(k^2 n \log n)$ ist, wobei k die Anzahl der Kategorien und n die Größe der größten beteiligten RC_1,C_2-Relation. In den gängigsten Fällen ist die Komplexität jedoch nur $O(kn \log n)$.

[0097] Der MakeOnto-Algorithmus fügt neue Werte in C ein, um zu Erreichen, daß die Abbildung von C nach P summierbar ist. Dies bedeutet wieder, daß die Kosten zum Erzeugen von Ergebnissen für C für die umgewandelte Hierarchie höher sind als für die ursprüngliche. Ohne das Einfügen der neuen Werte, hätten wir jedoch Ergebnisse für P und eventuell auch für höhere Kategorien und ebenso für C zu erzeugen. Für jeden Wert in P wird höchstens ein Wert in C eingefügt, was bedeutet, daß die zusätzlichen Kosten für C niemals größer als die Kosten zum Erzeugen von Ergebnissen in P sind. Wie zuvor ist dies ein sehr unrealistischer Fall, da er dem Fall entspricht, daß keine Werte in P Kinder in C haben. In den meisten Fällen sind die zusätzlichen Kosten für C ein kleiner Prozentsatz der Kosten zum Erzeugen von Ergebnissen für P und die möglichen Einsparungen sind sogar größer, da die Vorab-Aggregationen für Kategorien höherer Ebenen vermieden werden.

[0098] Wie zuvor hat die Korrektheitsaussage für den Algorithmus zwei Gesichtspunkte. Erstens sollten die Abbildungen in der Hierarchie bei der Terminierung auf sein. Zweitens sollte der Algorithmus nur Umwandlungen durchführen, die semantisch korrekt sind. Die Korrektheit folgt aus den nachstehenden Theoremen 3 und 4. Wieder ist der unter Verwendung der ursprünglichen Hierarchie erhaltene Ergebnissatz für die ursprünglichen Werte eine Teilmenge des Ergebnissatzes, den man unter Verwendung der umgewandelten Hierarchie erhält. Die Ergebnisse für die neuen Werte können vom Ergebnissatz durch das Hinzufügen einer HAVING-Bedingung ausgeschlossen werden.

Theorem 3

[0099] Der Algorithmus MakeOnto terminiert und die Hierarchie der sich ergebenden Dimension D' ist auf.

Beweis

[0100] Durch Induktion in der Höhe des Dimensionsgitters. Anfang: die Höhe ist 0 wodurch die Aussage trivialerweise wahr ist. Induktionsschritt: Wir nehmen an, die Aussage sei für Dimensionsgitter der Höhe n wahr und betrachten Gitter der Höhe n + 1. Zur Terminierung bemerken wir, daß es für jedes P eine endliche Anzahl von Nachfolgern C gibt, daß alle Operationen in der Schleife terminieren und, daß der Algorithmus rekursiv auf

C aufgerufen wird, das das oberste Element in einem Gitter der Höhe n ist. Für die Auf-Eigenschaft bemerken wir, daß das Einfügen von Platzhalterwerten in C die Abbildung von C nach P auf macht. Mit der Induktionshypothese werden die im Gitter tiefer gelegenen Abbildungen durch den rekursiven Aufruf behandelt.

Theorem 4

[0101] Gegeben seien Dimensionen D und D', wobei D' das Ergebnis der Ausführung des MakeOnto-Algorithmus auf D ist, dann ist ein unter Verwendung von D erhaltenes Aggregationsergebnis eine Untermenge des unter Verwendung von D' erhaltenen Ergebnisses.

Beweis

[0102] Folgt leicht aus der Beobachtung, daß "kinderlose" Dimensionswerte mit neuen Platzhalterwerten in niedrigeren Kategorien in eins-zu-eins Beziehungen verknüpft werden, was bedeutet, daß Daten für kinderlose Werte in Aggregationsberechnungen, die die neue Dimension verwenden, weiterhin genau einmal gezählt werden.

Nicht-Strenge Hierarchien

[0103] Der dritte Algorithmus macht Abbildungen in Hierarchien streng (strict), was bedeutet, daß Probleme mit "Doppelzählungen" nicht auftreten. Nicht-strenge Hierarchien treten auf, wenn ein Dimensionswert mehrerer Elternwerte hat.

[0104] Die grundlegende Idee ist es, eine Satz von Elternwerten in einen "vereinigten" (fused) Wert zu vereinigen und dann den Kindwert statt dessen zu diesem neuen Wert zu verknüpfen. Die vereinigten Werte werden in eine neue Kategorie eingefügt, die zwischen den Kind- und Eltern-Kategorien liegt. Daten für die neue vereinigte Kategorie können gefahrlos zur Berechnung von Aggregationsergebnissen höherer Ebenen wiederverwendet werden, da die zur neuen Kategorie hinaufführende Hierarchie streng ist.

[0105] Der vereinigte Wert wird auch mit den relevanten Elternwerten verknüpft. Diese Abbildung ist der Natur nach nicht-streng, aber diese Nicht-Strengheit ist kein Problem, da wir es verhindern, daß Aggregationsergebnisse der Elternkategorie weiter oben in der Hierarchie wiederverwendet werden. Dies wird durch das Auflösen der Verknüpfungen der Elternkategorie mit ihren Vorgänger-Kategorien erreicht.

[0106] Die weiter oben gelegenen Kategorien werden statt dessen über die vereinigte Kategorie erreicht. Das bedeutet, daß wir weiterhin für jede ursprüngliche Kategorie Ergebnisse erhalten können, während wir gleichzeitig in der Lage sind über die ganze Hierarchie die praktische Vorab-Aggregation anzuwenden. Bezüglich der Vorab-Aggregation bedeutet das Auflösen der Verknüpfungen der Elternkategorien, daß wir es verhindern müssen Ergebnisse zu Erzeugen, die diese Kategorie einschließen – nur "sichere" (safe) Kategorien können beitragen. Dies sollte dem Vorab-Aggregationssystem, das die beteiligten Ebenen auswählt, als Randbedingung gegeben werden.

[0107] Wir bemerken, daß der Algorithmus nicht mehr Ebenen in der Hierarchie einführt, sondern nur mehr Kategorien, und daß die Anzahl der "sicheren" Kategorien im Ergebnis dieselbe wie die Anzahl der ursprünglichen Kategorien ist. Das bedeutet, daß die Komplexität der Auswahl der optimalen beteiligten Aggregationsebenen durch den Algorithmus unverändert bleibt.

Beispiel 8

[0108] Das Ergebnis des Ausführens des Algorithmus auf der Diagnosedimension ist in **Fig.** 4 zu sehen. Durch die Nicht-Strengheit in der Abbildung von den Diagnosen niedriger Ebene zu den Diagnosefamilien und von den Diagnosefamilien zu den Diagnosegruppen werden zwei neue Kategorietypen und die entsprechenden Kategorien eingeführt. Das dritte Bild zeigt die Eingabe des Algorithmus; und zusätzlich zeigen seine gepunkteten Linien die durch den Algorithmus gelöschten Verknüpfungen. Das vierte Bild zeigt das Ergebnis der Anwendung des Algorithmus; hier zeigen die fett gedruckten Werte und die dicken Linien die durch den Algorithmus eingefügten Werte und Verknüpfungen.

[0109] Im ersten Aufruf des Algorithmus werden die drei Diagnosewerte der niedrigen Ebene – "(niedrige Ebene) Lungenkrebs" (L14); "Insulinabhängiger Diabetes während einer Schwangerschaft" (5); und "Nicht-insulinabhängiger Diabetes während einer Schwangerschaft" (6) – mit den drei neuen vereinigten Werten – "(niedrige Ebene) Lungenkrebs" (14); "Diabetes während einer Schwangerschaft, Insulinabhängiger Diabetes" (4, 9); und "Diabetes während einer Schwangerschaft, Nicht-insulinabhängiger Diabetes" (4,10) – verknüpft und diese werden wiederum mit "Lungenkrebs" (14); "Diabetes während einer Schwangerschaft" (4); "Insulinabhängiger Diabetes" (9); und "Nichtinsulinabhängiger Diabetes" (10) verknüpft. Die Verknüpfungen letzterer vier Werte in der Diagnosefamilie zu ihren Eltern werden aufgelöst, da die Diagnosefamilie-Kategorie "unsi-

cher" ist.

[0110] Beim rekursiven Aufrufen auf der Satz-von-Diagnosefamilien-Kategorie erzeugt der Algorithmus in der Satz-von-Diagnosegruppen-Kategorie die neuen vereinigten Werte "Krebs" (13) und "Diabetes, Andere Krankheiten im Zusammenhang mit Schwangerschaft" (11, 12). Diese neuen Werte werden mit den Werten "Krebs" (13), "Diabetes" (11) und "Andere Krankheiten im Zusammenhang mit Schwangerschaft" (12) in der Diagnosegruppe-Kategorie und mit dem ⌈-Wert verknüpft; und die Verknüpfungen der Werte in der Diagnosegruppe-Kategorie mit ihren Eltern werden aufgelöst. Man beachte wie wichtig es ist einen ⌈-Wert zu haben: die nicht mit ⌈ verknüpften Werte sind genau die unsicheren Werte, für die keine Aggregationsergebnisse wiederverwendet werden sollten.

[0111] Der Algorithmus geht davon aus, daß alle Pfade in der Dimensionshierarchie gleiche Länge haben, das heißt alle direkten Verknüpfungen gehen von Kindern zu ihren unmittelbaren Eltern. Dies wird durch die MakeCovering- und MakeOnto-Algorithmen erreicht. Im untenstehenden Algorithmus ist C eine Kindkategorie, P ist eine Elternkategorie, G ist eine Großelternkategorie, N ist die neue Kategorie, die zur Aufnahme der "vereinigten" Werte eingeführt wird, und $\otimes$ bezeichnet die natürliche Hintereinanderschaltung.

[0112] Der Algorithmus nimmt eine Kategorie C (anfänglich die ⌊-Kategorie) als Eingabe. Er geht dann durch die den Satz der unmittelbaren Elternkategorien P von C (Zeile(2)), Zeile (4) testet, ob in der Abbildung von C nach P Nicht-Strengheit auftritt und ob P Eltern hat (4). Wenn dieser Test fehlschlägt, gibt es kein Problem, da Aggregationsergebnisse für P entweder gefahrlos wiederverwendet werden können oder es sichergestellt ist, daß sie nicht wiederverwendet werden; und der Algorithmus wird dann in Zeile (20) rekursiv aufgerufen.

[0113] Wenn der Test erfolgreich ist, erzeugt der Algorithmus eine neue vereinigte Kategorie. Zuerst wird eine neue leere Kategorie N mit dem Wertbereich 2^P in Zeile (6) erzeugt. Die in diese Kategorie eingesetzten Werte repräsentieren Sätze von Werten aus P. Beispielsweise repräsentiert der Wert "1,2" den Satz, der genau aus 1,2 besteht. Werte in C werden dann zu den neuen vereinigten Werten verknüpft, die deren bestimmte Eltern-Kombination in P repräsentieren (7). Die neuen Werte werden unter Verwendung einer Vereinigungsfunktion (Fuse-Funktion) konstruiert, die einen gesonderten Wert für jede Kombination von P-Werten erzeugt und die dazugehörigen P-Werte mit diesen abspeichert.

[0114] Die sich ergebenden Verknüpfungen werden in Zeile (8) verwendet, um die vereinigten Werte in ihre Kategorie N einzufügen, und eine "Trenn"-Funktion (Unfuse-Funktion), die vereinigte Werte in N auf zugehörige Werte in P abbildet, wird in Zeile (9) verwendet, um die Werte in N auf die in P abzubilden. In Zeile (10) wird N in den Satz der Vorgänger von C eingeschlossen und P wird davon ausgeschlossen. Der Satz von Vorgängern von N wird in Zeile (11) auf P gesetzt, was bedeutet, daß die neue Kategorie N in der Hierarchie zwischen C und P liegt.

```
(1)   procedure MakeStrict (C)
(2)       for each P ∈ Pred(C) do
(3)       begin
(4)           if (∃e1 ∈ C (∃e2, e3 ∈ P (e1 ≤ e2 ∧ e1 ≤ e3 ∧ e2 ≠ e3)))
                  ∧ Pred(P) ≠ 0 then
(5)           begin
(6)               N ← CreateCategory(2^P)
(7)               R_{C,N} ← {(e1, Fuse({e2 | (e1, e2) ∈ R_{C,P}}))}
(8)               N ← Π_N(R_{C,N})
(9)               R_{N,P} ← {(e1, e2) | e1 ∈ N ∧ e2 ∈ Unfuse(e1)}
(10)              Pred(C) ← Pred(C) ∪ {N} \ {P}
(11)              Pred(N) ← {P}
(12)              for each G ∈ Pred(P) do
(13)              begin
(14)                  R_{N,G} ← Π_{N,G}(R_{N,P} ⊗ R_{P,G})
(15)                  Pred(N) ← Pred(N) ∪ {G}
(16)                  Pred(P) ← Pred(P) \ {G}
(17)              end
(18)              MakeStrict(N)
(19)          end
(20)          else MakeStrict(P)
(21)      end
(22)  end
```

[0115] Für jede Großelternkategorie G verknüpft der Algorithmus in Zeile (14) Werte in N mit Werten in G, fügt in Zeile (15) G den Vorgängern von N hinzu und schließt G in Zeile (16) von den Vorgängern von P aus, wobei auch die Verknüpfungen von P nach G aus der Hierarchie entfernt werden. Der Ausschluß der G-Kategorien von den Vorgängern von P bedeutet, daß die Aggregationsergebnisse für P nicht zur Berechnung von Ergebnissen für die G-Kategorien wiederverwendet wird.

[0116] Schließlich wird der Algorithmus rekursiv auf der neuen Kategorie N aufgerufen. Es ist zu bemerken, daß der Test auf Pred(P) ≠ 0 in Zeile (4) sicherstellt, daß die Abbildung von N nach P nicht verändert wird, da P nun keinen Vorgänger hat.

[0117] Der Argumentation in den vorhergehenden Abschnitten folgend, finden wir, daß die gesamte Big-O-Komplexität O(pnk log n log k) ist, wobei p die Anzahl der unmittelbaren Eltern- und Kindkategorien im Dimensionstypgitter ist, n die Größe der größten Abbildung in der Hierarchie ist und k die maximale Anzahl miteinander vereinigter Werte ist. Für die meisten realistischen Fälle sind p und k kleine Konstanten, was zu einer niedrige O(n log n)-Komplexität für den Algorithmus führt.

[0118] Der Strengmachungs-Algorithmus (MakeStrict-Algorithmus) erzeugt eine neue Kategorie N und fügt vereinigte Werte in N ein, um die Summierbarkeit der Abbildung von N nach P und von N nach G zu erreichen. Der Algorithmus fügt die vereinigten Werte nur für die Kombinationen ein, die tatsächlich in der Abbildung von C nach P auftreten. Das bedeutet, daß die Kosten zum Erzeugen von Ergebnissen für N niemals höher als die Kosten zum Erzeugen von Ergebnissen für C sind. Das ist der ungünstigste Fall, für die gängigsten Fälle sind die Kosten zum Erzeugen von Ergebnissen für N nahe bei den Kosten zum Erzeugen von Ergebnissen für P. Ohne die Einführung von N hätten wir jedoch nicht nur Ergebnisse für P zu Erzeugen, sondern auch für G und

alle Kategorien höherer Ebenen. Daher ist das Einsparpotential in den Erzeugungskosten tatsächlich sehr groß.

[0119] In Bezug auf die Korrektheit, sollten die Abbildungen in den Hierarchien beim Terminieren streng sein und der Algorithmus sollte nur Umwandlungen ausführen, die semantisch korrekt sind. Genauer ist es akzeptabel, daß einige Abbildungen nicht-streng sind, nämlich diejenigen von der neuen vereinigten Kategorie zu den unsicheren Elternkategorien. Dies ist so, da unsichere Kategorien keine Vorgänger in der sich ergebenden Hierarchie haben, was bedeutet, daß die Aggregationsergebnisse für diese Kategorien nicht wiederverwendet werden.

[0120] Die Korrektheit folgt aus den untenstehenden Theoremen 5 und 6. Bei der Auswertung von Anfragen erhalten wir für die ursprünglichen Werte dasselbe Ergebnis wie bei der Auswertung auf den alten Hierarchien. Die durch den Algorithmus gelöschten Werte waren mit keinen Fakten verknüpft, was bedeutet, daß diese Werfe zum Ergebnis in der ursprünglichen Hierarchie nicht beigetragen haben. Da alle neuen Werte in neue Kategorien eingefügt werden, die dem Benutzer unbekannt sind, ist das erzielte Aggregationsergebnis für die ursprüngliche und die umgewandelte Hierarchie dasselbe. Die ursprüngliche Anfrage muß daher nicht verändert werden.

Theorem 5

[0121] Sei D' die Dimension, die sich aus der Anwendung des MakeStrict-Algorithmus auf die Dimension D ergibt. Dann gilt das folgende: der MakeStrict-Algorithmus terminiert und die Hierarchie für die Dimension D", die sich ergibt, wenn die unsicheren Kategorien von D' entfernt werden, ist streng.

Beweis

[0122] Durch Induktion in der Höhe des Dimensionsgitters. Anfang: die Höhe ist 0 wodurch die Aussage trivialerweise wahr ist. Induktionsschritt: Wir nehmen an, die Aussage sei für Dimensionsgitter der Höhe n wahr und betrachten Gitter der Höhe n + 1. Alle Schritte im Algorithmus terminieren und der Algorithmus wird rekursiv entweder auf P (im strengen Fall) oder auf N (im nicht-strengen Fall) aufgerufen, die beide Wurzeln eines Gitters der Höhe n sind und daher die Terminierung sicherstellen.

[0123] Die Strengheits-Eigenschaft betreffend gibt es drei Fälle. Wenn die Abbildung von C nach P bereits streng ist, wird diese Abbildung nicht verändert und mit der Induktionshypothese gilt die Aussage für den rekursiven Aufruf auf P. Wenn die Abbildung von C nach P nicht-streng ist, aber P keine Eltern hat, ist die Strengheit gesichert, da P von D" ausgeschlossen wird. Wenn die Abbildung nicht-streng ist und P Eltern hat, ist die sich ergebende Abbildung von C nach N streng. Mit der Induktionshypothese gilt die Aussage für den rekursiven Aufruf auf N, da die Einführung von N die Höhe des Gitters nicht vergrößert hat.

Theorem 6

[0124] Gegeben seien Dimensionen D und D', wobei D' das Ergebnis der Ausführung des MakeStrict-Algorithmus auf D ist, dann ist ein unter Verwendung von D' erhaltenes Aggregat dasselbe wie das unter Verwendung von D erhaltene.

Beweis

[0125] Folgt aus Lemma 2, da alle Fakten auf Werte in der $_{\perp}$-Kategorie abgebildet werden, die eine sichere Kategorie ist. Daher gibt es von einem Fakt f einen Pfad zu einem ursprünglichen Dimensionswert e dann und genau dann, wenn es einen Pfad in der ursprünglichen Hierarchie gab, was bedeutet, daß Aggregationsergebnisse, die unter Verwendung der ursprünglichen und der neuen Hierarchie berechnet wurden dieselben sind.

Lemma 2

[0126] Für die Dimension D' = (C', ≤'), die sich aus einer Anwendung des MakeStrict-Algorithmus auf die Dimension D = (C,≤) ergibt, gilt das folgende: $\forall e_1,e_2 \in D\ (e_1 \in C_1 \wedge \text{Safe}(C_1) \wedge e_1 \leq' e_2 \Leftrightarrow e_1 \leq e_2)$ (zwischen einem ursprünglichen Dimensionswert in einer sicheren (safe) Kategorie und einem beliebigen anderen ursprünglichen Dimensionswert in der neuen Dimensionshierarchie gibt es dann und genau dann einen Pfad, wenn es zwischen ihnen einen Pfad in der ursprünglichen Hierarchie gegeben hat).

Beweis

[0127] Durch Induktion in der Höhe des Dimensionsgitters. Anfang: die Höhe ist 0 wodurch die Aussage tri-

vialerweise wahr ist. Induktionsschritt: Wenn die Abbildung von C nach P streng ist oder P keine Eltern hat, verändert der Algorithmus die Abbildungen nicht und mit der Induktionshypothese ist die Aussage für alle rekursiven Aufrufe auf P wahr. Andernfalls stellen wir fest, daß die Erzeugung vereinigter Werte N und die Verknüpfung von C-, P- und G-Werten mit diesen nur genau die Werte in C und P oder C und G verknüpft, die zuvor verknüpft waren. Da P nicht sicher ist können die Verknüpfungen von P nach G gelöscht werden. Mit der Induktionshypothese ist die Aussage für den rekursiven Aufruf auf N wahr.

[0128] Im oben beschriebenen Verfahren bleiben sowohl die ursprünglichen als auch die neuen Kategorien in derselben Hierarchie. Eine Alternative wäre es die "unsicheren" Kategorien und die Abbildungen zu ihnen in einer getrennten Hierarchie zu halten, so daß nur "sichere" Kategorien in der Haupthierarchie verbleiben.

Fakt-Dimensions-Umwandlungstechniken

[0129] Dieser Abschnitt erläutert wie der Satz von Algorithmen aus dem Abschnitt "Dimensions-Umwandlungstechniken" auch auf die Beziehungen zwischen Fakten und Dimensionen angewendet werden kann und damit eine Grundlage bietet, um praktische Vorab-Aggregation auf konkrete MOs anzuwenden, die Faktdaten einschließen.

[0130] Die grundlegende Idee ist es, den Faktensatz F als die unterste Ebene (Genauigkeitsstufe) in einem Gitter zu betrachten. Die Eingabe der Algorithmen besteht dann aus den Fakten, F, den $R_{F,C}$-Tabellen, die die Abbildungen von den Fakten zu den Dimensionswerten beschreiben, und den C- und RC_1,C_2-Tabellen, die die Dimensionskategorien und die Abbildungen zwischen ihnen beschreiben.

[0131] Nur die Abdeckend- und Strengheits-Eigenschaften werden berücksichtigt, da für die Fakt-Dimensions-Beziehungen eine Abbildung zwischen Fakten und Dimensionswerten, die in ist bedeutet, daß nicht alle Dimensionswerte in der untersten Kategorie zugehörige Fakten haben, was sich auf die Summierbarkeit nicht auswirkt. Wie zuvor wenden wir zuerst den MakeCovering-Algorithmus an, dann den MakeStrict-Algorithmus.

[0132] Die Komplexität des Algorithmus wird nun durch die Größe n der Abbildung zwischen den Fakten und Dimensionswerten dominiert, das heißt die Komplexität ist O(n log n), wenn wir annehmen, daß die Höhe des Gitters und die maximale Anzahl miteinander vereinigter Werte kleine Konstanten sind. Das bedeutet, daß der Algorithmus selbst auf sehr große Datenbanken angewendet werden kann.

Abbildungen gemischter Genauigkeitsstufen

[0133] Der erste zu betrachtende Fall ist derjenige, bei dem einige der Abbildungen nichtabdeckend bezüglich den Fakten sind, was bedeutet, daß nicht alle Fakten über diese Abbildungen erreicht werden und daher dazu führen, daß diese Fakten in Aggregationsberechnungen nicht berücksichtigt werden. Dies tritt auf, wenn einige Fakten direkt auf Dimensionswerte in höheren Kategorien als der $_L$-Kategorie abgebildet werden, das heißt die Fakten werden auf Werte gemischter Genauigkeitsstufen abgebildet.

[0134] Wir verwenden den MakeCovering-Algorithmus, um die Abbildungen abdeckend zu machen, wobei er zuerst auf F aufgerufen wird, das jetzt die unterste Ebene des Gitters ist. Der Algorithmus macht die Abbildungen abdeckend bezüglich den Fakten, indem er neue markierte Werte, Repräsentanten der Elternwerte, in die Zwischenkategorien einfügt und indem er die Fakten mit den neuen Werten anstatt mit den Elternwerten verknüpft. Wie im Abschnitt "Nicht-Abdeckende Hierarchien" behalten die markierten Werfe die Information über ihre ursprünglichen Werte, so daß beim Hinzufügen neuer Fakt-Dimensions-Abbildungen, die Verknüpfungen, die dafür gedacht sind direkt mit den ursprünglichen Elternwerten verknüpft zu werden, statt dessen nun mit dem markierten Wert in der $_L$-Kategorie verknüpft werden können.

Beispiel 9

[0135] in der Fallstudie hat die Abbildung zwischen Patienten und Diagnosen gemischte Genauigkeitsstufen: "John Doe" (1) und "Jane Doe" werden beide auf die Diagnosefamilie "Insulinabhängiger Diabetes" (9) abgebildet, "Jane Doe" wird zusätzlich auf die Diagnose der niedrigen Ebene "Insulinabhängiger Diabetes während einer Schwangerschaft" (5) abgebildet und "Jim Doe" wird auf "Diabetes" (11) abgebildet, eine Diagnosegruppe.

[0136] Im ersten Aufruf des Algorithmus werden zwei neue Diagnosen der niedrigen Ebene eingefügt: "L9" für "Insulinabhängiger Diabetes" und "L11" für "Diabetes"; und die Fakten werden auf diese anstatt auf die ursprünglichen Werte abgebildet. Im rekursiven Aufruf auf den Diagnosen der niedrigen Ebene wird ein "F11 "-Wert für "Diabetes" auf der Diagnosefamilien-Ebene zwischen "Diabetes" und dem Wert "L11" eingefügt.

[0137] Die Umwandlungen sind in **Fig.** 5 dargestellt, wobei gepunktete Linien Verknüpfungen bezeichnen, die durch den Algorithmus gelöscht wurden, und fett gedruckte Werte und dicke Linien die durch den Algorithmus eingefügten Dimensionswerte und Verknüpfungen bezeichnen.

Viele-zu-Viele Beziehungen

[0138] Der zweite Fall tritt auf, wenn Beziehungen zwischen Fakten und Dimensionswerten viele-zu-viele sind. Das bedeutet, daß die Hierarchie, mit den Fakten als der untersten Kategorie, nicht-streng ist, was zu möglichen Doppelzählungen von Fakten führt. Es ist ausreichend die Hierarchie teilweise streng zu machen, wie im Abschnitt "Nicht-Strenge Hierarchien" beschrieben ist. Der MakeStrict-Algorithmus wird zuerst auf F aufgerufen, das nun die unterste Ebene der Hierarchiegitters ist. Da der MakeCovering-Algorithmus bereits angewendet wurde, haben alle Pfade von Fakten zum ⌜-Wert gleiche Länge, wie vom MakeStrict-Algorithmus gefordert.
[0139] Einige Dimensionswerte haben keine auf sie abgebildeten Fakten, was zu einem interessanten Seiteneffekt des Algorithmus führt. Wenn der Algorithmus Werte vereinigt und die vereinigten Werte zwischen den ursprünglichen Werten anordnet, löscht er auch die Kind-zu-Eltern- und Eltern-zu-Großeltern-Verknüpfungen. Die faktenlosen Dimensionswerte sind dann abgetrennt vom Rest der Hierarchie, ohne Verknüpfungen zu anderen Werten.
[0140] Diese faktenlosen Dimensionswerte tragen zu keinen Aggregationsberechnungen bei und sind daher überflüssig. Zur Minimierung der Dimensionen wird in einem Nachbearbeitungslauf ein "Lösche-unverbundene"-Algorithmus angewendet, der die faktenlosen Dimensionswerte löscht, indem der angefangen bei den Fakten die Hierarchie durchläuft. Für eine Hierarchie der Höhe k kann dies in O(kn log n) ausgeführt werden, wobei n die Größe der Abbildung zwischen Fakten und Dimensionen ist. Daher ist die gesamte Komplexität unverändert.

Beispiel 10

[0141] Die Beziehung zwischen Patienten und Diagnosen ist viele-zu-viele. In Beispiel 9 wurde das MO so umgewandelt, daß alle Abbildungen abdeckend sind, wie in **Fig.** 5 zu sehen ist, der MakeStrict-Algorithmus wird auf dieses MO angewendet. Das Endergebnis der Anwendung des Algorithmus ist in **Fig.** 5 zu sehen. Werte in kursiv, beispielsweise L14, und gepunktete Linien zeigen gelöschte Werte und Verknüpfungen an. Fett gedruckte Werte und dicke Linien bezeichnen durch den Algorithmus eingefügte Werte und Verknüpfungen.
[0142] Drei neue Kategorien werden eingeführt: "Satz-von Diagnose niedriger Ebene", "Satz-von Diagnosenfamilie" und "Satz-von Diagnosegruppe", da Nicht-Strengheit auf allen Ebenen auftritt. Vereinigte Werte werden in diese vereinigten Kategorien eingefügt. Beispielsweise werden die Werte "(niedrige Ebene) Lungenkrebs" (L14), "Insulinabhängiger Diabetes während einer Schwangerschaft, (niedrige Ebene) Insulinabhängiger Diabetes" (5,L9) und "(niedrige Ebene) Insulinabhängiger Diabetes" (L9) in die "Satz-von Diagnose niedriger Ebene"-Kategorie eingefügt; und die ursprünglichen Werte werden mit den neuen Werten verknüpft.
[0143] Die Werte "(niedrige Ebene) Lungenkrebs" (L14), "Lungenkrebs" (14), "Krebs" (13), "Nicht-insulinabhängiger Diabetes während einer Schwangerschaft" (6) und "Nicht-insulinabhängiger Diabetes" (10) beschreiben keine Fakten und werden durch "Lösche-unverbundene" gelöscht.

Architekturkontext

Überblick

[0144] Die in diesem Schriftstück dargestellte übergeordnete Idee ist es nicht-normalisierte MOs in normalisierte MOs umzuwandeln, die durch die in derzeitigen OLAP-Systemen verfügbaren praktischen Vorab-Aggregationstechniken gut unterstützt werden. Anfragen werden dann auf den umgewandelten MOs ausgewertet. Der Anwender soll jedoch weiterhin lediglich die ursprünglichen MOs sehen, da sie das Verständnis des Anwenders von dem Gebiet Wiederspiegeln. Dies verlangt nach Mitteln, die es ermöglichen sowohl die ursprünglichen wie auch die umgewandelten MOs zu bearbeiten. Dieser Abschnitt untersucht diese Koexistenz.
[0145] Ein derzeitiger Trend in kommerziellen OLAP-Technologien ist die Trennung der Front-End-Präsentationsschicht von den Back-End-Datenbankservern. Moderne OLAP-Anwendungen bestehen aus einem OLAP-Client, der die Benutzerschnittstelle handhabt, und einem OLAP-Server, der die Daten handhabt und Anfragen bearbeitet. Der Client kommuniziert mit dem Server über eine standardisierte Anwendungs-Programmierschnittstelle (application programming interface, API), beispielsweise OLE DB für OLAP von Microsoft [17] oder die OLAP Council's MDAPI [20]. Die Architektur eines solchen Systems wird links in **Fig.** 6 dargestellt.
[0146] Diese Trennung eines Clients und Servers erleichtert unseren Wunsch, daß der Anwender das ursprüngliche MO sieht, während Anfragen auf dem umgewandelten MO ausgewertet werden. Studien haben gezeigt, daß Anfragen auf einem Data-Warehouse zu 80% aus Navigationsanfragen, die die Dimensionshierarchien erkunden, und zu 20% aus Aggregationsanfragen, die die Daten auf verschiedenen Detailebenen zusammenfassen, bestehen [14]. Beispiele von Navigationsanfragen und Aggregationsanfragen sind "Zeige mir

die Diagnosen niedriger Ebene, die in der Insulinabhängiger Diabetes Diagnosefamilie enthalten sind" beziehungsweise "Zeige mir die nach Diagnosefamilien gruppierten Patientenzahlen". Die Navigationsanfragen müssen auf dem ursprünglichen MO ausgeführt werden, während die Aggregationsanfragen auf dem umgewandelten MO ausgeführt werden müssen. Dies wird durch das Einführen einer zusätzlichen "Anfragehandler"-Komponente zwischen dem Client und dem Server erreicht. Der OLAP-Client sendet eine Anfrage an einen Anfragehandler, dessen primäre Aufgabe es ist, zu ermitteln, ob die Anfrage eine Navigationsanfrage (intern zu einer Dimension) oder eine Aggregationsanfrage (die Fakten einschließend) ist. Navigationsanfragen werden an den OLAP-Server weitergereicht, der die ursprünglichen (Navigations-) Daten handhabt, während Aggregationsanfragen an einen weiteren OLAP-Server weitergereicht werden, der die umgewandelten (Aggregations-) Daten handhabt. Diese erweiterte Systemarchitektur ist rechts in **Fig.** 6 zu sehen.

[0147] Der OLAP-Server für Navigationsdaten muß Dimensionshierarchien unterstützen, die nicht-summierbare Eigenschaften haben, eine Forderung die derzeit von nicht vielen kommerziellen Systemen unterstützt wird. Relationale OLAP-Systeme, die Snow-Flake-Schemata verwenden [14], sind jedoch in der Lage diesen Hierarchietyp zu unterstützen, genauso wie einige andere OLAP-Systeme, beispielsweise Hyperion (Arbor) Essbase [12]. Wenn das verfügbare OLAP-System keine ausreichend flexible Hierarchie-Unterstützung bietet, ist eine Lösung, einen speziellen OLAP-Server zu bauen, der konform zur gegebenen API ist. Diese Aufgabe ist weniger aufwendig als es auf den ersten Blick scheinen mag, da nur Navigationsanfragen unterstützt werden müssen, was bedeutet, daß die multidimensionalen Anfragen in einfache SQL-"lookup"-Anfragen übersetzt werden können.

[0148] Es ist zu bemerken, daß die einzigen für die Beantwortung von Navigationsanfragen notwendigen Daten, die Hierarchiedefinitionen sind. Daher müssen die Faktdaten (Fakten und Fakt-Dimensions-Relationen in unserem Modell) nur einmal in den Aggregationsdaten abgespeichert werden, was bedeutet, daß der Gesamtspeicherplatzbedarf nur geringfügig größer ist, als die Speicherung der Aggregationsdaten alleine. Navigationsanfragen werden auf den ursprünglichen Hierarchiedefinitionen ausgewertet und müssen vom Anfragehandler nicht umgeschrieben werden.

[0149] Wie im Abschnitt "Dimensions-Umwandlungstechniken" beschrieben, müssen Aggregationsanfragen leicht umgeschrieben werden, indem eine HAVING-Bedingung hinzugefügt wird, um Ergebnisse für die von den Umwandlungsalgorithmen eingefügten neuen Werte auszuschließen. Dies kann leicht automatisch vom Anfragehandler, wodurch sich für den Anwender eine vollständige Transparenz ergibt. Obwohl die hinzugefügten HAVING-Bedingungen nur für die Abeckend- und Auf-Umwandlungen notwendig sind, können sie auch auf Hierarchien angewendet werden, die Umgewandelt wurden, um Strengheit zu erreichen; dies hat keine Auswirkung, vereinfacht aber das Umschreiben der Anfragen. Die neuen Werte können ebenso durch die Anwendung einer modifizierten WHERE-Bedingung herausgefiltert werden, indem eine innere Verbindung mit einer Tabelle ausgeführt wird, die nur die originalen Werte enthält, oder indem verschachtelte SELECT-Anweisungen ausgeführt werden, so wie im nächsten Abschnitt beschrieben.

Konkrete Implementierung

[0150] Wir zeigen nun, wie die oben beschriebene abstrakte Architektur unter Anwendung von gängiger relationaler Datenbanktechnologie implementiert werden kann.

[0151] Die Transparenz wird erreicht, indem mit zwei Versionen jeder anwender-spezifizierten Hierarchie gearbeitet wird und indem ein Anfrage-Umschreib-Mechanismus eingesetzt wird. Dies wird im Detail später in diesem Abschnitt beschrieben. Die gesamte Systemarchitektur ist in **Fig.** 7 zu sehen.

DiagID	niedrige Ebene	Familie	Gruppe
5	Insulinabhängiger Diabetes während Schwangerschaft	Diabetes während Schwangerschaft	Diabetes
5	Insulinabhängiger Diabetes während Schwangerschaft	Diabetes während Schwangerschaft	schwangerschafts-bezogen
5	Insulinabhängiger Diabetes während Schwangerschaft	Insulinabhängiger Diabetes	Diabetes
6	Nicht-insulinabhängiger Diabetes während Schwangerschaft	Diabetes während Schwangerschaft	Diabetes
6	Nicht-insulinabhängiger Diabetes während Schwangerschaft	Diabetes während Schwangerschaft	schwangerschafts-bezogen
6	Nicht-insulinabhängiger Diabetes während Schwangerschaft	Nicht-insulinabhängiger Diabetes	Diabetes
100	!niedrige Ebene! Lungenkrebs	Lungenkrebs	Krebs

Tabelle 2
Diagnose-Dimensions-Tabelle

[0152] Das ROLAP-Client-Werkzeug, in diesem Fall das ROLAP-Werkzeug Synchrony, das seine Ursprung in Kimball's Startracker Tool [14] hat, macht SQL-Anfragen an die ROLAP-Datenbank, in diesem Fall eine Oracle8 RDBMS, unter Verwendung des ODBC-Standards. Wir haben einen speziellen anfrageumwandelnden ODBC-Treiber (QTOD) entwickelt, der basierend auf Fallspezifischen Metadaten die SQL-Anfragen in Anfragen umwandelt, die die Umwandlungen vor dem Benutzer verstecken, wobei die Anfrageergebnisse zurückgegeben werden, die der Benutzer basierend auf der ursprünglichen Hierarchie erwartet. Eine umgewandelte Anfrage wird an die OLAP-DB gesendet unter Verwendung eines RDBMS-spezifischen ODBC-Treibers. Die QTOD-Komponente ist allen RDBM gemeinsam, so daß Oracle8 durch eine andere RDBMS, wie etwa IBM DB2, Informix, oder MS SQL Server, ersetzt werden kann. Es kann auch ein anderes ROLAP-Werkzeug verwendet werden, wodurch die Lösung relativ allgemein und flexibel ist.
[0153] Wir haben uns entschlossen den Prototyp auf einer RDBMS (Oracle8) aufzubauen, da RDBMSs die gängigsten Plattformen für Data-Warehouse- und OLAP-Anwendungen sind. Außerdem verwenden die wichtigsten RDBMS derzeit, wie dedizierte multidimensionale DBMSes (MDDBs), vorab-aggregierte Daten für schnellere Anfrageantworten [30]. Der Ansatz könnte jedoch auch unter Verwendung multidimensionaler Technologie implementiert werden, beispielsweise basierend auf dem Microsoft OLE DB für OLAP-Standard [17].
[0154] Die Umwandlungsalgorithmen sind in der PL/SQL Programmiersprache von Oracle implementiert. Die Umwandlungen sind relativ schnell, sie benötigen höchstens einige Minuten, selbst für große Dimensionen. Sind die Dimensionshierarchien einmal umgewandelt, wandelt der QTOD die Anfragen und Ergebnisse zwischen den ursprünglichen und den umgewandelten Hierarchien um. Der QTOD ist eine dünne Schicht und fügt den Anfragen sehr wenig Mehraufwand hinzu. Er ist unter Verwendung von GNU Flex++/Bison++ Scanner/Parser-Generatoren und dem MS Visual C++ Compiler implementiert.
[0155] Die zwei Arten von Anfragen, Navigationsanfragen und Aggregationsanfragen, werden verschieden behandelt, um dem Benutzer die Illusion zu geben, daß die Dimensionshierarchien ihre ursprüngliche Form haben.
[0156] Die multidimensionalen Daten werden in einem Star-Schema festgehalten [14]. Die Dimensionstabelle für die Diagnosedimension ist in Tabelle 2 gegeben, die eine Spalte für die ID der Diagnose niedriger Ebene zusätzlich zu den Spalten für die textlichen Beschreibungen der Diagnose niedriger Ebene, Diagnosefamilie und Diagnosegruppe aufweist.
[0157] Die in der Tabelle festgehaltene Hierarchie ist teilweise normalisiert, das heißt es wurden Platzhalterwerte eingeführt, um die Hierarchie auszugleichen (sie bleibt aber nicht-streng). Genauer wurde der "!niedrige Ebene! Lungenkrebs"-Platzhalterwert in die Ebene der Diagnosen niedriger Ebene eingefügt. Wir geben solchen Werten einen Präfix "!" und ihre Ebene, um anzuzeigen, daß sie durch den Umwandlungsprozeß eingefügt wurden. Es ist das mehrfache Auftreten von Werten niedrigerer Ebenen zu bemerken, das durch die Nicht-Strengheit der Hierarchie verursacht ist. Das ist die Tabelle, die für die Benutzernavigation in der Hierarchie verwendet wird. Ihr Name ist mit dem Präfix "D" versehen um sie von einer anderen "Diagnose"-Dimen-

sionstabelle (nachfolgend beschrieben), die für Aggregationsanfragen verwendet wird, zu unterscheiden.
[0158] Wir beschreiben nun wie die Transparenz für Navigationsanfragen erreicht wird. Die untenstehende Anfrage ruft alle Diagnosenamen niedriger Ebene ab.

```
SELECT DISTINCT niedrige Ebene
FROM Diagnose
```

[0159] Von ROLAP-Werkzeugen ausgegebene Navigationsanfragen haben exakt dieses Format. Die Anfrage wird vom QTOD in die untenstehende Anfrage umgewandelt, die auf der Tabelle DDiagnose operiert. Die umgewandelte Anfrage liefert das in Tabelle 3 gezeigte Ergebnis zurück.

```
SELECT DISTINCT niedrige Ebene
FROM DDiagnose
WHERE niedrige Ebene NOT LIKE '!%'
```

niedrige Ebene
Insulinabhängiger Diabetes während Schwangerschaft
Nicht-insulinabhängiger Diabetes während Schwangerschaft

Tabelle 3
Navigationsanfrage-Ergebnis

[0160] Durch die Verwendung von DISTINCT als Quantifikator, werden keine Duplikate zurückgegeben. Das NOT LIKE Prädikat entfernt die zum Ausgleich in die Hierarchie eingefügten Platzhalterwerte, in diesem Fall der Wert "!niedrige Ebene! Lungenkrebs". Wie gewünscht ist das Ergebnis unbeeinflußt von den Übersetzungen.
[0161] Für Aggregationsanfragen ist es ebenso möglich Umwandlungstransparenz zu erreichen, auch wenn dies schwieriger ist. Für nicht-strenge Dimensionen wird eine spezielle Dimensionstabelle eingeführt, die nur den Teil der normalisierten Hierarchie enthält, der keine nicht-strengen Anteile hat. In der normalisierten Hierarchie rechts im linken Teil der **Fig.** 4 ist dieser Teil die Kategorie der Diagnosen niedriger Ebene und die beiden speziellen Kategorien, die durch den Normalisierungsprozeß eingeführt wurden, um Sätze von Diagnosefamilien beziehungsweise Sätze von Diagnosegruppen aufzunehmen. Dieser Teil der Hierarchie ist in der Diagnose-Dimensionstabelle implementiert, die in Tabelle 4 zu sehen ist.

DiagID	niedrige Ebene	Familie	Gruppe
1000020	!Diagnose niedriger Ebene! Lungenkrebs	14	13
5	Insulinabhängiger Diabetes während Schwangerschaft	4,9	11,12
6	Nicht-insulinabhängiger Diabetes während Schwangerschaft	4,10	11,12

Diagnose

Gruppe	SGruppe
Krebs	13
Diabetes	11,12
schwangerschafts-bezogen	11,12

SGruppe

Tabelle 4

Dimensions- und Gruppen-Tabelle für Aggregation

[0162] Die "niedrige Ebene"-Spalte enthält die normale textliche Diagnosebeschreibung, während die speziellen "Familie" und "Gruppe"-Spalten komma-getrennte sortierte Listen der IDs der Wertesätze enthalten, die durch die Spaltenwerte repräsentiert werden. Beispielsweise repräsentiert der Wert "4,9" den Satz {4,9}.
[0163] Wir müssen den verbleibenden Teil der Hierarchie festhalten, der aus nichtstrengen Abbildungen von einer "Satz-von-X"-Kategorie zu einer "X"-Kategorie besteht, beispielsweise die Abbildung der "Satz-von-Diagnosegruppe"-Kategorie zu der "Diagnosegruppe"-Kategorie rechts in der **Fig.** 4, die {13} auf 13 (Krebs) und {11, 12} auf 11 (Diabetes) und 12 (schwangerschafts-bezogen) abbildet. Dies wird durch die Einführung einer speziellen Tabelle für jede derartige Abbildung erreicht, benannt nach der Kategorie mit einem "S"-Präfix (für Satz-von). Beispielsweise bildet für die Diagnosegruppe-Kategorie die Tabelle "SGruppe" in Tabelle 4 Sätze von Diagnosegruppen auf die einzelnen Diagnosegruppen im Satz ab. Die "Gruppe"-Spalte repräsentiert die Diagnosegruppe, während die "SGruppe"-Spalte den zugehörigen Satz von Diagnosegruppen repräsentiert.
[0164] Mit Hilfe dieser Tabellen ist es möglich Umwandlungstransparenz für Aggregationsanfragen zu erhalten. Eine ROLAP-Aggregationsanfrage hat das nachfolgende Anfrageformat, das die Patientenanzahl pro Diagnosegruppe berechnet.

```
SELECT Diagnose . Gruppe, SUM ( Patient. Count )
FROM Diagnose, Patient
WHERE Diagnose.DiagID = Patient.DiagID
GROUP BY Diagnose. Group
```

[0165] Dies wird in die nachfolgende komplexere Anfrage umgewandelt.

```
SELECT SGruppe. Gruppe, SUM(QQQQQQQ. Count)
FROM SGruppe,
(SELECT Diagnose.Gruppe,
SUM(Patient. Count) AS Count
FROM Diagnose, Patient
WHERE Diagnose.DiagID = Patient.DiagID
GROUP BY Diagnose. Gruppe) QQQQQQQ
WHERE QQQQQQQ. Gruppe = SGruppe. SGruppe AND
SGruppe. SGruppe NOT LIKE '!%'
GROUP BY SGRuppe.SGruppe
```

[0166] Die umgewandelte Aggregationsanfrage hat zwei Teile. Der verschachtelte Tabellenausdruck berechnet die Anzahl der Patienten pro Satz von Diagnosegruppe, was über den Korrelationsname QQQQQQQ ermöglicht wird. Dieser Teil der Hierarchie ist ein ausgewogener Baum, so daß die RDBMS gefahrlos vorabaggregierte Daten zur Optimierung der Anfrageleistung verwenden kann. Das Ergebnis des verschachtelten Tabellenausdrucks wird in der äußern Anfrage verwendet, die unter Verwendung der "SGruppe"-Tabelle den letzten Teil des Wegs aufwärts zu den Diagnosegruppen aggregiert. Die äußere Anfrage filtert auch alle Platzhalterwerte heraus, die durch den Normalisierungsprozeß eingefügt wurden ("!"-Präfix). Als ein Ergebnis wird das Client-OLAP-Werkzeug das erwartete Ergebnis liefern.
[0167] Eine gute Anfrageleistung ohne den Einsatz übermäßigen Speicherplatzes für vorab-aggregierte Daten wird erreicht, indem für den in der "Diagnose"-Dimensionstabelle festgehaltenen "netten" Teil der Hierarchie praktische Vorab-Aggregation verwendet wird. Die hier veranschaulichte Anfrageumwandlung kann für alle ROLAP-Aggregationsanfragen durchgeführt werden, was die Lösung relativ allgemein macht.

Inkrementelle Berechnung

[0168] Wenn Dimensionshierarchien oder Faktdaten aktualisiert werden, müssen die umgewandelten Hierarchien dementsprechend aktualisiert werden. Eine Lösung ist die Neuberechnung der Hierarchien mit den neuen Daten. Diese direkte Lösung ist attraktiv, wenn kleine Dimensionshierarchien aktualisiert werden, die sich nur selten ändern, oder wenn große Mengen an Aktualisierungen durchgeführt werden. Für große Hierarchien und häufige Aktualisierungen sowie für Aktualisierungen von kleinen Teilen der Hierarchie im allgemeinen, ist es jedoch wünschenswert, daß der Algorithmus nur die geänderten Teile der Daten zu berücksichtigen braucht, was nur ein kleiner Bruchteil des ganzen Datenvolumens ist. Dieser Abschnitt beschreibt, wie der Algorithmus inkremental angewendet werden kann.

[0169] Zusätzlich zur Abänderung der umgewandelten Hierarchien ist es auch notwendig die tatsächlichen vorab-aggregierten Daten zu aktualisieren, wenn die zugrundeliegenden Basisdaten geändert wurden. Die geänderten Hierarchien, die sich aus den in diesem Abschnitt gegebenen Algorithmen ergeben unterscheiden sich nur lokal von den Eingabehierarchien. Das bedeutet, daß die Kosten zur Aktualisierung der vorab-aggregierten Daten durch die Hierarchieumwandlungen nicht stark beeinflußt werden.

[0170] In den inkrementellen Algorithmen sind Aktualisierungen als Löschungen gefolgt von Einfügungen modelliert, daher berücksichtigen wir nur die zwei letzteren Modifikationsoperationen. Wir verwenden den Präfix Δ_i zur Bezeichnung von eingefügten Werten, Δ_d bezeichnet gelöschte Werte und Δ bezeichnet alle Modifikationen. Beispielsweise bezeichnet $\Delta_i C$ die in C eingefügten Werte. Die Kategorie- und Verknüpfungs-Tabellen in den Algorithmen beziehen sich auf die Zustände nach der Modifikation; und wenn ein Hierarchiewert gelöscht wird, werden auch alle Verknüpfungen zu diesem Wert im selben Satz von Modifikationen gelöscht.

Abdeckende Hierarchien

[0171] Modifikationen können abdeckende Hierarchien auf mehrere Weisen nichtabdeckend machen. Die ganz linke Tabelle in Tabelle 5, "Abdeckend" benannt und nachfolgend besprochen, zeigt an, ob eine Einfügung ("Einfügen") oder eine Löschung ("Löschen") in den verschiedenen Teilen der Eingabe zu MakeCovering die modifizierte Hierarchie nicht-abdeckend macht.

	Einfügen	Löschen
C	ja	nein
P	nein	ja
H	nein	ja
$R_{C,H}$	ja	nein
$R_{C,P}$	nein	ja
$R_{P,H}$	nein	ja

Abdeckend

	Einfügen	Löschen
C	nein	ja
P	ja	nein
$R_{C,P}$	nein	ja

Auf

	Einfügen	Löschen
C	nein	ja
P	nein	ja
G	nein	ja
$R_{C,P}$	ja	ja
$R_{P,G}$	ja	ja

Streng

Tabelle 5

Auswirkung von Einfügungen und Löschungen auf die Abdeckend-, Auf- und Strengheits-Eigenschaften

[0172] Probleme können auftreten, wenn Verknüpfungen in $R_{C,H}$ eingefügt werden, die nicht durch Einfügungen in $R_{C,P}$ und $R_{P,H}$ abgedeckt sind, oder wenn Verknüpfungen in $R_{C,P}$ oder $R_{C,H}$ gelöscht werden, aber die zugehörigen C-zu-H-Verknüpfungen in $R_{C,H}$ nicht gelöscht werden. Wenn Werte in P oder H gelöscht werden, werden auch ihre Verknüpfungen gelöscht, was durch den oben genannten Fall behandelt wird. Werte können nicht ohne irgendwelche Verknüpfungen in C eingefügt werden, da alle Werte in der ursprünglichen Hierarchie wenigstens zum ⌜-Wert verknüpft sein müssen.

[0173] Die inkrementelle Version des MakeCovering-Algorithmus beginnt damit (in Zeile (6)), die Verknüpfungen L von C nach H zu finden, die nicht durch die Verknüpfungen von C nach P und P nach H abgedeckt sind. Diese Verknüpfungen werden als Basis für den Rest der Umwandlung verwendet. Daher wird Zeile (6) des Algorithmus folgender Ausdruck.

$$L \leftarrow \Delta_i R_{C,H} \cup \Pi_{C,H}(\Delta_d R_{C,P} \otimes R_{P,H}) \cup \Pi_{C,H}(R_{C,P} \otimes \Delta_d R_{P,H}) \setminus \Pi_{C,H}(\Delta_i R_{C,P} \otimes \Delta_i R_{P,H}) \setminus \Delta d R_{C,H}$$

Auf-Hierarchien

[0174] Die Auswirkungen von Einfügungen und Löschungen auf die Auf-Eigenschaft sind in der mittleren Tabelle in Tabelle 5 umrissen. Einfügen von Werten in P, Löschen von Werten in C und Löschen von Verknüpfungen in $R_{C,P}$ kann verursachen, daß die Hierarchie nicht-auf wird. Die inkrementelle Version des MakeOnto-Algorithmus beginnt daher damit (in Zeile (4)), die "kinderlosen" Werte N aus P mit keinen Kindern in C zu finden. Als Ergebnis wird Zeile (4) des Algorithmus folgender Ausdruck.

$$N \leftarrow \Delta_i P \cup \Pi_P (\Delta_d R_{C,P}) \setminus \Pi_P (\Delta_d P) \setminus \Pi_P (\Delta i R_{C,P})$$

Strenge Hierarchien

[0175] Die Pflege der Strengheits-Eigenschaft von Hierarchien ist komplizierter, da im Algorithmus eine neue Kategorie N eingefügt wird. Wir nehmen an, alle neuen Kategorien seien bereits erzeugt bevor der inkrementelle Algorithmus eingesetzt wird, das heißt, falls in neuen Teilen der Hierarchie Nicht-Strengheit eingeführt

wird, müssen wir die umgewandelte Hierarchie neu berechnen. Die Einführung von Nicht-Strengheit erfordert eine größere Umarbeitung sowohl der Hierarchie als auch der vorab-aggregierten Daten, wenn dies vernünftig ist.

[0176] Einen Überblick über die Auswirkungen von Einfügungen und Löschungen in der Eingabe des Make-Strict-Algorithmus auf die Strengheit ist in der ganz rechten Tabelle in Tabelle 5 gegeben. Wenn Verknüpfungen in $R_{C,P}$ oder $R_{P,G}$ eingefügt oder daraus gelöscht werden, müssen die Verknüpfungen nach N für die betroffenen C-, P- und G-Werte neu berechnet werden.

[0177] Einfügungen in oder Löschungen aus C, P oder G werden durch die entsprechenden Einfügungen und Löschungen von Verknüpfungen begleitet, weshalb sie vom oben genannten Fall behandelt werden. Der untenstehende inkrementelle MakeStrict-Algorithmus (IncrementalMakeStrict) findet die betroffenen C-, P- und G-Werte, berechnet dann deren Verknüpfungen nach N neu und löscht die alten Verknüpfungen und fügt schließlich die neuen Verknüpfungen ein. Wie zuvor wird er gefolgt von einer Anweisung, die die abgetrennten Teile der Hierarchie löscht.

```
(1)  procedure IncrementalMakeStrict(C)
(2)      for each P ∈ Pred(C) such that Pred(P) ≠ 0 do
(3)      begin
(4)          dC ← Π_C(ΔR_{C,P})
(5)          dR_{C,N} ← {(c, Fuse({p | (c,p) ∈ dC ⊗ R_{C,P}})}
(6)          dN ← Π_N(dR_{C,N})
(7)          N ← N ∪ dN
(8)          R_{C,N} ← R_{C,N} \ {(c,n) | c ∈ dC} ∪ dR_{C,N}
(9)          dP ← Π_P(ΔR_{C,P})
(10)         dR_{N,P} ← {(n,p) | n ∈ dN ∧ p ∈ dP ∩ UnFuse(n)}
(11)         R_{N,P} ← R_{N,P} \ {(n,p) | p ∈ dP} ∪ dR_{N,P}
(12)         for each G ∈ Pred(P) do
(13)         begin
(14)             dG ← Π_G(ΔR_{P,G} ∪ (dP ⊗ R_{P,G}))
(15)             R_{N,G} ← R_{N,G} {(n,g) | g ∈ dG} ∪ Π_{N,G}(R_{N,P} ⊗ R_{P,G} ⊗ dG)
(16)         end
(17)         IncrementalMakeStrict(N)
(18)     end
(19) end
```

Literaturverzeichnis

[1] E. Baralis, S. Paraboschi and E. Teniente. Materialized View Selection in a Multidimensional Database. In Proceedings of the Twenty-Third International Conference on Very Large Data Bases, S. 156–165,1997.

[2] E. F Codd. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report, E.F. Codd and Associates, 1993.

[3] S. Dar, H.V. Jagadish, A.Y Levy, and D. Srivastava. Answering SQL Queries Using Views. In Proceedings of the Twenty-Second International Conference on Very Large Data Bases, S. 318–329, 1996.

[4] P.M. Deshpande, J.F. Naughton, K. Ramasamy, A. Shukla, K. Tufte and Y Zhao. Cubing Algorithms, Storage Estimation, and Storage and Processing Alternatives for OLAP. IEEE Data Engineering Bulletin, 20(1):3–11, 1997.

[5] C.E. Dyreson. Information Retrieval from an Incomplete Data Cube. In Proceedings of the Twenty-Second Conference on Very Large Databases, S. 532–543,1996.

[6] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F Pellow and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals. Data Mining and Knowledge Discovery, 1 (1):29–54, 1997.

[7] A. Gupta, V. Harinarayan and D. Quass. Aggregate Query Processing in Data Warehousing Environments.

In Proceedings of the Twenty-First International Conference on Very Large Data Bases, S. 358–369, 1995.
[8] H. Gupta, V. Harinarayan, A. Rajaraman and J. Ullman. Index Selection for OLAP In Proceedings of he Thirteenth International Conference on Data Engineering, S. 208–219, 1997.
[9] H. Gupta. Selection of Views to Materialize in a Data Warehouse. In Proceedings of the Sixth International Conference on Database Theory, S. 98–112, 1997.
[10] H. Gupta and I.S. Mumick. Selection of Views to Materialize Under a Maintenance-Time Constraint. In Proceedings of the Seventh International Conference on Database Theory, S. 453–470, 1999.
[11] V. Harinarayan, A. Rajaraman and J. D. Ullman. Implementing Data Cubes Efficiently. In Proceedings of the ACM SIGMOD international Conference on the Management of Data, S. 205–216, 1996.
[12] Hyperion Corporation. Hyperion Essbase OLAP Server. URL: <www.hyperion.com/downloads/essbaseolap.pdf>. Current as of February 17,1999.
[13] Informix Corporation. Data Warehouse Administrator's Guide: MetaCube ROLAP Option for Informix Dynamic Server. URL: <www.informix.com/answers/ english/pdf docs/metacube/4189.pdf>. Current as of February 15,1999.
[14] R. Kimball. The Data Warehouse Toolkit. Wiley Computer Publishing, 1996.
[15] R. Kimball. Data Warehouse Architect: Help with Multi-Valued Dimension. DBMS Magazine, 11(9), 1998.
[16] N. Lenz and A. Shoshani. Summarizability in OLAP and Statistical Data Bases. In Proceedings of the Ninth International Conference on Statistical and Scientific Database Management, S. 39–48,1997.
[17] Microsoft Corporation. OLE DB for OLAP Version 1.0 Specification. Microsoft Technical Document, 1998.
[18] Microsoft Corporation. OLAP Services White Paper. URL: <www.microsoft.com/ sql/70/whpprs/olapoverview.htm>. Current as of February 9, 1999.
[19] I.S. Mumick, D. Quass and B.S. Mumick. Maintenance of data cubes and summary tables in a warehouse. In Proceedings of the ACM SIGMOD International Conference on the Management of Data, S. 100–111, 1997.
[20] The OLAP Council. MDAPI Specification Version 2.0. OLAP Council Technical Document, 1998.
[21] The OLAP Report. Database Explosion. URL: <www.olapreport.com/Database-Explosion.htm>. Current as of February 10, 1999.
[22] T.B. Pedersen and C.S. Jensen. Multidimensional Data Modeling for Complex Data. In Proceedings of the Fifteenth International Conference on Data Engineering, 1999. Extended version available as TimeCenter Technical Report TR-37, URL: <www.cs.auc.dk/TimeCenter>, 1998.
[23) T.B. Pedersen, C.S. Jensen and C.E. Dyreson. Extending Practical Pre-Aggregation in On-Line Analytical Processing. In Proc. VLDB, S. 663–674, 1999. Extended version available as TR R-99-5004, Dept. of Comp. Sci. Aalborg University, <www.cs.auc.dk/Ntbp/articles/R995004.ps>, 1999.
[24] D. Quass and J. Widom. On-Line Warehouse View Maintenance for Batch Updates. In Proceedings of the ACM SIGMOD International Conference on the Management of Data, S. 393–404, 1997.
[25] M. Rafanelli and A. Shoshani. STORM: A Statistical Object Representation Model. In Proceedings of the Fifth International Conference on Statistical and Scientific Database Management, S. 14–29, 1990.
[26] A. Segev and J. L. Zhao. Selective View Materialization in Data Warehousing Systems. Working paper, URL: <ftpa/segev.lbl.gov/pubILBL_DB_PUBLICATIONS/ 1997/aggreg-dw.ps>. Current as of February 9, 1999.
[27] A. Shukla, P.M. Deshpande, J.F Naughton and K. Ramasamy. Storage Estimation for Multidimensional Aggregates in the Presence of Hierarchies. In Proceedings of the Twenty-Second International Conference on Very Large Data Bases, S. 522–531, 1996.
[28] D. Theodoratos and T. Sellis. Data Warehouse Configuration. In Proceedings of the Twenty-Third International Conference on Very Large Data Bases, S. 126–135, 1997.
[29] J. Widom. Research Problems in Data Warehousing. In Proceedings of the Fourth International Conference on Information and Knowledge Management., S. 25–30, 1995.
[30] R. Winter. Databases: Back in the OLAP game. Intelligent Enterprise Magazine, 1 (4):60–64, 1998.
[31] World Health Organization. International Classification of Diseases (ICD-10). Tenth Revision, 1992.
[32] J. Yang, K. Karlapalem and Q.Li. Algorithms for materialized view design in a data warehousing environment. In Proceedings of the Twenty-Third International Conference on Very Large Data Bases, S. 136–145, 1997.

Legende zu den Figuren

	Englisch	Deutsch
Fig. 1		
	adress	Adresse
	located in city	liegt in Stadt
	Name	Name
	city	Stadt
	city located in	Stadt liegt in
	county	Verwaltungsbezirk
	lives at	wohnhaft in
	located in rural area	liegt in ländlicher Gegend
	patient	Patient
	has	hat
	diagnosis	Diagnose
	code	Code
	Text	Text
	low-level diagnosis	Diagnose niedriger Ebene
	belongs to	gehört zu
	type	Typ
	diagnosis family	Diagnosefamilie
	grouping	Gruppierung
	diagnosis group	Diagnosegruppe
Fig. 2		
	diagnosis dimension	Diagnosedimension
	diagnosis group	Diagnosegruppe
	diagnosis family	Diagnosefamilie
	low-level diagnosis	Diagnose niedriger Ebene
	residence dimension	Wohnsitzdimension
	county	Verwaltungsbezirk

	city	Stadt
	adress	Adresse
	name dimension	Namensdimension
	patient	Patient
Fig. 4		
	diagnosis group	Diagnosegruppe
	diagnosis family	Diagnosefamilie
	low-level diagnosis	Diagnose niedriger Ebene
	set-of diagnosis group	Satz-von Diagnosegruppe
	set-of diagnosis family	Satz-von Diagnosefamilie
Fig. 6		
	OLAP client	OLAP-Client
	queries	Anfragen
	OLAP server	OLAP-Server
	data	Daten
	query handler	Anfragehandler
	navigational queries	Navigationsanfragen
	aggregation queries	Aggregationsanfragen
	navigational data	Navigationsdaten
	aggregational data	Aggregationsdaten
Fig. 7		
	ROLAP client tool	ROLAP-Client-Werkzeug
	generic SQL	allgemeines SQL
	query-transforming ODBC driver (QTOD)	anfrage-umwandelnder ODBC-Treiber (QTOD)
	generic transformed SQL	allgemeines umgewandeltes SQL
	DBMS specific ODBC driver	DBMS-spezifischer ODBC-Treiber
	DBMS specific transformed SQL	DBMS-spezifisch umgewandeltes SQL

	ROLAP DB (Oracle)	ROLAP DB (Oracle)

Patentansprüche

1. Verfahren zur Umwandlung einer allgemeinen Online-Analytical-Processing-Dimension in eine zumindest teilweise aggregations-normalisierte Dimension, d.h. in eine Dimension mit verbesserter Summierbarkeit, mittels eines Computers, wobei die Dimension Dimensionswerte hat, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, wobei die Dimension Abbildungen von Verknüpfungen zwischen Dimensionswerten umfaßt, wobei das Verfahren folgende Schritte umfaßt:
Abrufen der Abbildung von einem zum Computer gehörenden Datenspeichermittel,
Analysieren der Abbildung zur Bestimmung von Irregularitäten der Dimension, d. h. die Dimension nicht-summierbar machenden Merkmalen, anhand von durch den Computer ausgeführten Analysemitteln,
Erzeugen neuer Dimensionswerte der Dimension und Modifizieren der Abbildung zwischen Dimensionswerten der Dimension entsprechend der Analyse, wodurch die Dimension zumindest teilweise aggregations-normalisiert ist, und
Sichern der neuen Dimensionswerte und der modifizierten Abbildungen auf dem Datenspeichermittel des Computers.

2. Verfahren nach Anspruch 1, bei dem der Schritt des Erzeugers neuer Dimensionswerte und des Modifizierens der Abbildung folgende Schritte umfaßt:
Untersuchen, ob die Dimension abdeckend ist, d.h. daß nur direkte Elternwerte und Kindwerte verknüpft sind, und surjektiv ist, d. h. daß alle Pfade in der Hierarchie gleiche Länge haben, und wenn dies der Fall ist,
Ausführen einer Strengmachungs-Prozedur, um die Dimension aggregations-streng zu machen, d. h. daß jedes Kind in der Hierarchie nur einen Elternknoten hat, und dadurch die nicht-strenge Dimension aggregations-normalisiert, d. h. summierbar zu machen.

3. Verfahren nach Anspruch 1 oder 2, bei dem der Schritt des Erzeugens neuer Dimensionswerte und des Modifizierens der Abbildung folgende Schritte umfaßt:
Untersuchen, ob die Dimension abdeckend ist, und wenn dies der Fall ist,
Ausführen einer Surjektivmachungs-Prozedur, um die Dimension surjektiv zu machen, wodurch eine nicht-surjektive Dimension zumindest teilweise aggregations-normalisiert gemacht wird.

4. Verfahren nach einem der Ansprüche 1 bis 3, bei dem der Schritt des Erzeugens neuer Dimensionswerte und des Modifizierens der Abbildung folgenden Schritt umfaßt:
Ausführen einer Abdeckendmachungs-Prozedur, um die Dimension abdeckend zu machen, wodurch eine nicht-abdeckende Dimension zumindest teilweise aggregations-normalisiert gemacht wird.

5. Verfahren nach einem der Ansprüche 2 bis 4, bei dem die Strengmachungs-Prozedur, anfangend bei der untersten Kategorie und nacheinander fortschreitend in Richtung der obersten Kategorie, folgende Schritte umfaßt:
Identifizieren von Kombinationen von Dimensionswerten derselben Kategorie, wobei für jede dieser Kombinationen zumindest ein Dimensionswert einer Kategorie unterhalb dieser Kategorie mit jedem der Dimensionswerte der Kombination verknüpft ist,
Erzeugen einer oder mehrerer neuer Dimensionswerte, von denen jeder eine der identifizierten Kombinationen von Dimensionswerten repräsentiert, und Erzeugen von Verknüpfungen von den neuen Dimensionswerten zu Dimensionswerten oberhalb gelegener Kategorien in Übereinstimmung mit existierenden Verknüpfungen von jedem der vom neuen Dimensionswert repräsentierten Dimensionswerte, und
Identifizieren von Dimensionswerten, die mit identifizierten Kombinationen von Dimensionswerten derselben Kategorie verknüpft sind und Ersetzen der Verknüpfungen mit Verknüpfungen zu neuen Dimensionswerten, die diese Kombinationen von Dimensionswerten repräsentieren.

6. Verfahren nach einem der Ansprüche 2 bis 5, bei dem die Strengmachungs-Prozedur folgende aufeinanderfolgenden Schritte umfaßt:
(i) Setzen der untersten Kategorie der Dimension als die Kindkategorie,
(ii) für jede Kategorie, die eine direkte Vorgängerkategorie der Kindkategorie ist, für die zumindest ein Dimensionswert der Kindkategorie mit einem Dimensionswert verknüpft ist, Setzen dieser Kategorie als Elternkategorie und Ausführen der Schritte
(iia) Beenden der Strengmachungs-Prozedur mit der Elternkategorie, falls die Elternkategorie die oberste Kategorie der Dimension ist,

(iib) Beenden der Strengmachungs-Prozedur mit der Elternkategorie, falls kein Dimensionswert der Elternkategorie mit einem Dimensionswert einer höheren Kategorie verknüpft ist,
(iic) Erzeugen einer neuen vereinigten Kategorie in der Dimension direkt unterhalb der Elternkategorie, falls zumindest einer der Dimensionswerte der Kindkategorie mit mehr als einem Dimensionswert der Elternkategorie verknüpft ist,
(iid) Ausführen der folgenden Schritte für jeden Dimensionswert der Kindkategorie: Erzeugen eines neuen Dimensionswerts der neuen vereinigten Kategorie, der den einen oder die mehreren Werte der Elternkategorie repräsentiert, mit denen der Dimensionswert der Kindkategorie verknüpft ist, und Erzeugen von Verknüpfungen von diesem neuen Dimensionswert zu den Werten in der Elternkategorie, wobei die Erzeugung des neuen Dimensionswerts davon abhängig ist, daß kein Dimensionswert der neuen vereinigten Kategorie bereits existiert, der genau diese Verknüpfungen) aufweist, und
für jede Kategorie, die eine direkte Vorgängerkategorie der Elternkategorie ist, für die zumindest ein Dimensionswert der Elternkategorie mit einem Dimensionswert verknüpft ist, Setzen der Kategorie als eine Großelternkategorie und Erzeugen von Verknüpfungen vom neuen Dimensionswert zu dem einen oder den mehreren Dimensionswerten der Großelternkategorie, zu welcher der eine oder die mehreren Dimensionswerte der Elternkategorie verknüpft sind,
(iie) Entfernen der Verknüpfungen von der Elternkategorie zu der einen oder den mehreren Großelternkategorien, wodurch die Großelternkategorien keine direkten Vorgänger der Elternkategorie mehr sind,
(iif) Erzeugen von Verknüpfungen von jedem Dimensionswert der Kindkategorie zu den Dimensionswerten der neuen vereinigten Kategorie, der die gleichen Verknüpfungen zu den Dimensionswerten der Elternkategorie aufweist, wodurch die neue vereinigte Kategorie ein direkter Vorgänger der Kindkategorie wird, und Entfernen der Verknüpfungen von den Dimensionswerten der Kindkategorie zur Elternkategorie, wodurch die Elternkategorie kein direkter Vorgänger der Kindkategorie mehr ist, und
(iig) Setzen der neuen vereinigten Kategorie als Kindkategorie und zurückgehen zu Schritt (ii).

7. Verfahren nach einem der Ansprüche 3 bis 6, bei dem die Surjektivmachungs-Prozedur, anfangend bei der obersten Kategorie und nacheinander fortschreitend in Richtung der untersten Kategorie, folgende Schritte umfaßt:
für jeden Dimensionswert jeder Kategorie oberhalb der untersten Kategorie, der nicht mit irgendeinem Dimensionswert der Kategorie direkt unterhalb verknüpft ist, Erzeugen eines neuen Dimensionswerts in der direkt unterhalb liegenden Kategorie und Erzeugen einer Verknüpfung zwischen dem neuen Dimensionswert und dem Dimensionswert besagter Kategorie.

8. Verfahren nach einem der Ansprüche 3 bis 7, bei dem die Surjektivmachungs-Prozedur folgende aufeinanderfolgende Schritte umfaßt:
(i) Setzen der obersten Kategorie der Dimension als Elternkategorie,
(ii) für jede Kategorie direkt unterhalb der Elternkategorie, die mit Dimensionswerten der Elternkategorie verknüpfte Dimensionswerten aufweist, Setzen der Kategorie als Kindkategorie und Ausführen der Schritte
(iia) für jeden Dimensionswert der Elternkategorie, der nicht mit einem Dimensionswert der Kindkategorie verknüpft ist, Erzeugen eines neuen Dimensionswerts in der Kindkategorie und Erzeugen einer Verknüpfung zwischen dem neuen Dimensionswert und dem Dimensionswert der Elternkategorie,
(iib) Setzen der Kindkategorie als Elternkategorie,
(iic) Beenden der Surjektivmachungs-Prozedur, falls die Elternkategorie die unterste Kategorie der Dimension ist, andernfalls Zurückgehen zu Schritt (ii) der Surjektivmachungs-Prozedur.

9. Verfahren nach einem der Ansprüche 4 bis 8, bei dem die Abdeckendmachungs-Prozedur folgende aufeinanderfolgende Schritte umfaßt:
Bestimmen von Verbindungen zwischen Dimensionswerten zweier Kategorien, die zumindest eine dazwischenliegende Kategorie aufweisen,
Erzeugen eines neuen Dimensionswerts in jeder der dazwischenliegenden Kategorien für jede dieser Verknüpfungen, für die keine Verknüpfungspfade existieren, die nur über direkte Kind-Eltern-Verknüpfungen von niedrigeren zu höheren Kategorien führen und eine Verknüpfung mit einem Dimensionswert der dazwischenliegenden Kategorie einschließen, und
Ersetzen dieser Verknüpfungen mit Verknüpfungen zwischen den Dimensionswerten dieser Verknüpfungen und den neuen Dimensionswerten.

10. Verfahren nach einem der Ansprüche 4 bis 9, bei dem die Abdeckendmachungs-Prozedur die aufeinanderfolgenden Schritte
(i) Setzen der untersten Kategorie der Dimension als Kindkategorie,
(ii) für jede Kategorie direkt oberhalb der Kindkategorie, für die zumindest eine Verknüpfung zwischen einem

Dimensionswert der Kategorie und einem Dimensionswert der Kindkategorie existiert, Setzen der Kategorie als Elternkategorie und Ausführen der Schritte:
(iia) Beenden der Abdeckendmachungs-Prozedur mit der Elternkategorie, falls die Elternkategorie die oberste Kategorie der Dimension ist,
(iib) für jede höhere Kategorie, die eine direkte Vorgängerkategorie der Kindkategorie ist und in der Hierarchie höher als die Elternkategorie liegt, Ausführen der Schritte
(iiba) Bestimmen von Sätzen von Dimensionswerten der höheren Kategorie und Dimensionswerten der Kindkategorie, wobei für die Sätze eine Verknüpfung existiert, und kein Verknüpfungspfad existiert, der nur von niedrigeren zu höheren Kategorien führt und eine Verknüpfung zu einem Dimensionswert der Elternkategorie einschließt, und
(iibb) für jeden bestimmten Satz von Dimensionswerten Erzeugen eines neuen Dimensionswerts in der Elternkategorie, Erzeugen von Verknüpfungen zwischen jedem der Dimensionswerte des Satzes und dem neuen Dimensionswert, und Entfernen der Verknüpfung zwischen den zwei Dimensionswerten des bestimmten Satzes, wodurch die höhere Kategorie kein Vorgänger der Kindkategorie mehr ist,
(iic) Setzen der Elternkategorie als Kindkategorie und Zurückgehen zu Schritt (ii).

11. Verfahren zum zumindest teilweisen Aggregations-Normalisieren mittels eines Computers eines allgemeinen multidimensionalen Online-Analytical-Processing-Objekts, welches einen Faktensatz umfaßt, welcher eine Anzahl von Fakten, die auf eine Anzahl von Dimensionen abgebildet werden umfaßt, wobei die Dimensionen Dimensionswerte aufweisen, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, wobei das multidimensionale Objekt Abbildungen von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension umfaßt, durch die Anwendung des Verfahrens eines der Ansprüche 1 bis 10 auf zumindest eine der Dimensionen des multidimensionalen Objekts.

12. Verfahren nach Anspruch 11, bei dem das multidimensionale Objekt eine Anzahl von Fakten umfaßt und die Abbildung Verknüpfungen von jedem der Fakten zu wenigstens einem Dimensionswert in jeder der Anzahl von Dimensionen umfaßt, wobei die Fakten die unterste Ebene jeder der Dimensionen des multidimensionalen Objekts bilden.

13. Verfahren nach Anspruch 11 oder 12, das folgende Schritte umfaßt: Auswählen einer Untermenge von Kategorien der einen oder mehreren Dimensionen, die aggregations-normalisiert werden sollen, und Ausführen einer Aggregations-Normalisierung der ausgewählten Untermenge, wobei eine oder mehrere der Dimensionen des multidimensionalen Objekts nur teilweise aggregations-normalisiert ist/sind.

14. Verfahren nach einem der Ansprüche 11 bis 13, das folgende Schritte umfaßt: Auswählen bestimmter, auf dem multidimensionalen Objekt auszuführender, Aggregationsfunktionen, und
Auswählen von auszuführenden Normalisierungsschritten mittels eines Computers, basierend auf der Auswahl von bestimmten auszuführenden Aggregationsfunktionen,
wobei eine oder mehrere der Dimensionen des multidimensionalen Objekts nur teilweise aggregations-normalisiert ist/sind.

15. Verfahren zum zumindest teilweisen Aggregations-Normalisieren mittels eines Computers von einem allgemeinen multidimensionalen Online-Analytical-Processing-Objekt, welches einen Faktensatz umfaßt, welcher eine Anzahl von Fakten, die auf eine Anzahl von aggregations-normalisierten Dimensionen abgebildet werden umfaßt, wobei die Dimensionen Dimensionswerte aufweisen, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, wobei das multidimensionale Objekt Abbildungen von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension umfaßt,
wobei das Verfahren folgende Schritte umfaßt:
Abrufen der Abbildung von einem zum Computer gehörenden Datenspeichermittel,
Einfügen der Abbildung der Anzahl der Fakten auf dem multidimensionalen Objekt in die Abbildung des multidimensionalen Objekts, so daß die Abbildung Verknüpfungen von jedem der Fakten zu wenigstens einem Dimensionswert in jeder der Anzahl von Dimensionen umfaßt, und die Fakten die unterste Ebene jeder der Dimensionen des multidimensionalen Objekts bilden, Analyse der Abbildung des multidimensionalen Objekts zur Bestimmung der Irregularitäten der Dimension mittels eines vom Computer ausgeführten Analysemittels,
Erzeugen neuer Dimensionswerte des multidimensionalen Objekts und Modifizieren der Abbildung zwischen Dimensionswerten des multidimensionalen Objekts entsprechend der Analyse, wodurch das multidimensionale Objekt zumindest teilweise aggregations-normalisiert ist, und
Speichern der neuen Dimensionen und der modifizierten Abbildung im Datenspeichermittel des Computers.

16. Verfahren nach Anspruch 15, bei dem der Schritt der Erzeugung neuer Dimensionswerte und der Mo-

difizierung der Abbildung folgenden Schritt umfaßt: Ausführen einer Strengmachungs-Prozedur, um das multidimensionale Objekt aggregations-streng zu machen, wodurch das nicht-strenge multidimensionale Objekt aggregations-normalisiert wird, wobei die Strengmachungs-Prozedur unter der Bedingung ausgeführt wird, daß das multidimensionale Objekt vor der Ausführung abdeckend ist.

17. Verfahren nach Anspruch 15 oder 16, bei dem der Schritt der Erzeugung neuer Dimensionswerte und der Modifizierung folgenden Schritt umfaßt:
Ausführen einer Abdeckendmachungs-Prozedur, um das multidimensionale Objekt abdeckend zu machen, wodurch das nicht-abdeckende multidimensionale Objekt zumindest teilweise aggregations-normalisiert wird.

18. Verfahren nach einem der Ansprüche 15 bis 17, bei dem das Verfahren den anfänglichen Schritt umfaßt, jede der Anzahl der Dimensionen mittels des Verfahrens nach einem der Ansprüche 1 bis 10 aggregations-normalisiert zu machen.

19. Verfahren nach einem der Ansprüche 11 bis 18, bei dem die erzeugten neuen Dimensionswerte als solche markiert werden, eine Vorab-Aggregation auf einem mittels des Computers zu normalisierenden multidimensionalen Objekts ausgeführt wird und das Verfahren des weiteren folgenden Schritt umfaßt: Erzeugen einer Antwort auf eine an das System gestellte und das multidimensionale Objekt betreffende Anfrage, Aggregationsanfragen, Untersuchung der Dimensionshierarchien und ebenso Navigationsanfragen, die die Daten auf verschiedenen Detailebenen summieren, wobei in der Antwort die Existenz der erzeugten neuen Dimensionswerten transparent ist.

20. Verfahren nach einem der Ansprüche 11 bis 19, das des weiteren folgende Schritte umfaßt:
Einfügen neuer Fakten in das aggregations-normalisierte multidimensionale Objekt, einschließlich der Abbildung der Fakten auf die Dimension neuer Dimensionswerte der Dimensionen, oder einer neuen Abbildung zwischen einigen der Dimensionswerte, wobei durch die Einfügung Irregularitäten des multidimensionalen Objekts eingeführt werden,
Analyse der eingeführten Irregularitäten der Dimensionen des multidimensionalen Objekts,
Erzeugen neuer Dimensionswerte des multidimensionalen Objekts und Modifizieren der Abbildung zwischen Dimensionswerten des multidimensionalen Objekts entsprechend der Analyse, wodurch das multidimensionale Objekt aggregations-normalisiert wird, und
Speichern der neuen Dimensionen und der modifizierten Abbildung im Datenspeichermittel des Computers.

21. Computersystem für online-analytisches Processing, das wenigstens einen universell einsetzbaren Computer mit einem zugehörigen Datenspeichermittel umfaßt, wobei auf dem Datenspeichermittel ein Computerprogrammprodukt gespeichert ist, das geeignet ist, den Computer zu befähigen eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekts durchzuführen entsprechend dem Verfahren nach einem der Ansprüche 11 bis 20, wobei das Computersystem Mittel zum Abrufen des Computerprogrammprodukts und zur entsprechenden Durchführung umfaßt.

22. Computerprogrammprodukt, das geeignet ist, einen universell einsetzbaren Computer zu befähigen eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekt entsprechend dem Verfahren nach einem der Ansprüche 11 bis 20 durchzuführen.

23. Computersystem zum Online-Analytical-Processing mit zugehörigem Datenspeichermittel, auf dem ein multidimensionales Objekt gespeichert wird, wobei das multidimensionale Objekt folgendes einschließt:
einen Faktensatz, der eine Anzahl von Fakten umfaßt,
eine erste Anzahl von Dimensionen mit Dimensionswerten, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, und die eine erste Abbildung von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension der ersten Anzahl von Dimensionen und ebenso Verknüpfungen zwischen den Fakten und den Dimensionen der ersten Anzahl von Dimensionen umfaßt, wobei wenigstens eine der Dimensionen der ersten Anzahl von Dimensionen irregulär ist, und
eine zweite Anzahl von Dimensionen mit Dimensionswerten, die, basierend auf einer teilweisen Sortierung, in Kategorien von Dimensionswerten organisiert sind, und die eine zweite Abbildung von Verknüpfungen zwischen Dimensionswerten innerhalb jeder Dimension der zweiten Anzahl von Dimensionen und ebenso Verknüpfungen zwischen den Fakten und den Dimensionen der zweiten Anzahl von Dimensionen umfaßt, wobei jede Dimension der zweiten Anzahl von Dimensionen aggregations-normalisiert ist,
wobei das Computersystem eine Anfragehandlerkomponente umfaßt, die geeignet ist, Antworten auf an das Computersystem gestellte und das multidimensionale Objekt betreffende Anfragen zu erstellen, wobei Antworten auf Navigationsanfragen auf dem ersten Satz von Dimensionen basieren und Antworten auf Aggregations-

anfragen auf dem zweiten Satz von Dimensionen basieren.

24. Computersystem nach Anspruch 23, bei dem ein die zweite Anzahl von Dimensionen betreffender Satz von Vorab-Aggregationsdaten auf dem Datenspeichermittel gespeichert wird und Antworten zu Aggregationsanfragen zusätzlich auf dem Satz von Vorab-Aggregationsdaten basieren.

25. Computersystem nach Anspruch 23 oder 24, bei dem die Anfragehandlerkomponente geeignet ist, Antworten auf Aggregationsanfragen zu erzeugen, wobei in den Antworten die Existenz der zweiten Anzahl von Dimensionen nicht sichtbar ist.

26. Computersystem nach Anspruch 25, bei dem die Anfragehandlerkomponente geeignet ist, Aggregationsanfragen, die an die erste Anzahl von Dimensionen gestellt wurden, in Anfragen an den zweiten Satz von Dimensionen umzuwandeln und Antworten, die auf dem zweiten Satz von Dimensionen basieren, in Antworten; wie solche, die auf dem ersten Satz von Dimensionen basieren, umzuwandeln, und somit die Existenz der zweiten Anzahl von Dimensionen in der erstellten Antwort nicht sichtbar zu machen.

27. Computersystem nach Anspruch 26, bei dem das multidimensionale Objekt auf dem Datenspeichermittel des Computersystems in Tabellen gespeichert ist, die für den Teil des multidimensionalen Objekts, der nur strenge Abbildungen enthält, als eine Kombination von Star-Schemen organisiert sind, und zusätzlichen Tabellen, die den nicht-strengen Teil der Abbildungen enthalten, wobei die Anfragehandlerkomponente die Tabellen zur Umwandlung der Anfragen und Antworten verwendet.

28. Computersystem nach einem der Ansprüche 23 bis 27, das des weiteren Mittel umfaßt, die geeignet sind, eine zumindest teilweise Aggregations-Normalisierung eines multidimensionalen Objekts entsprechend dem Verfahren nach einem der Ansprüche 11 bis 20 durchzuführen.

Es folgen 4 Blatt Zeichnungen

Anhängende Zeichnungen

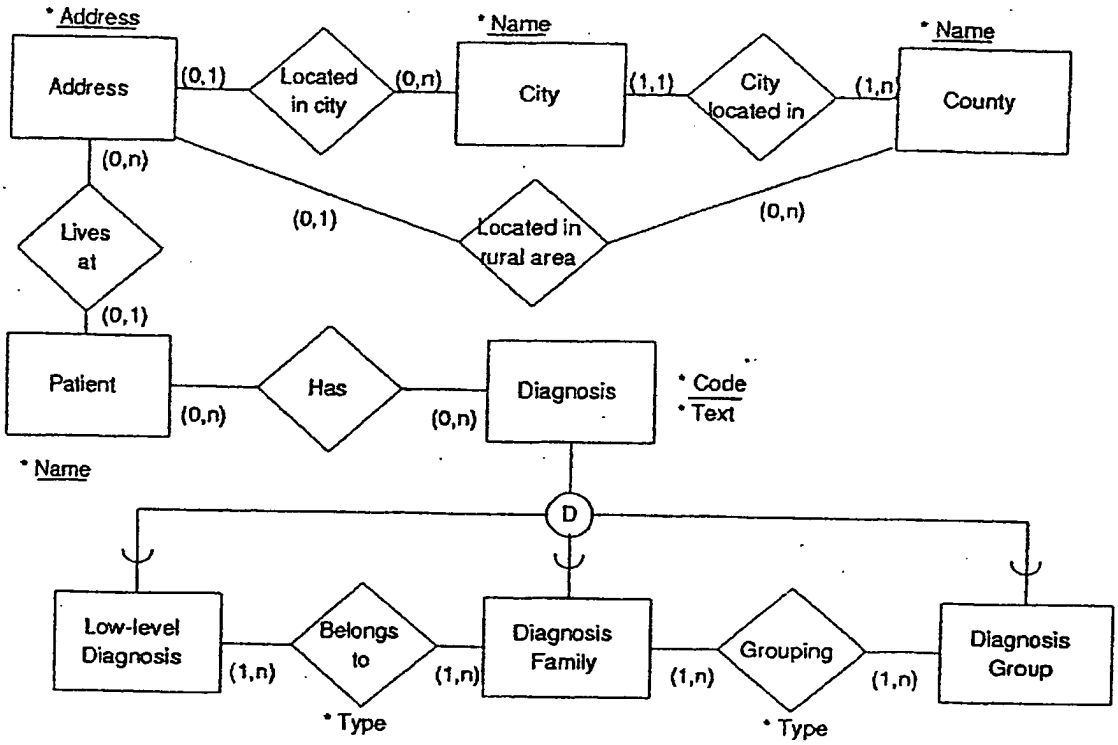


Fig. 1

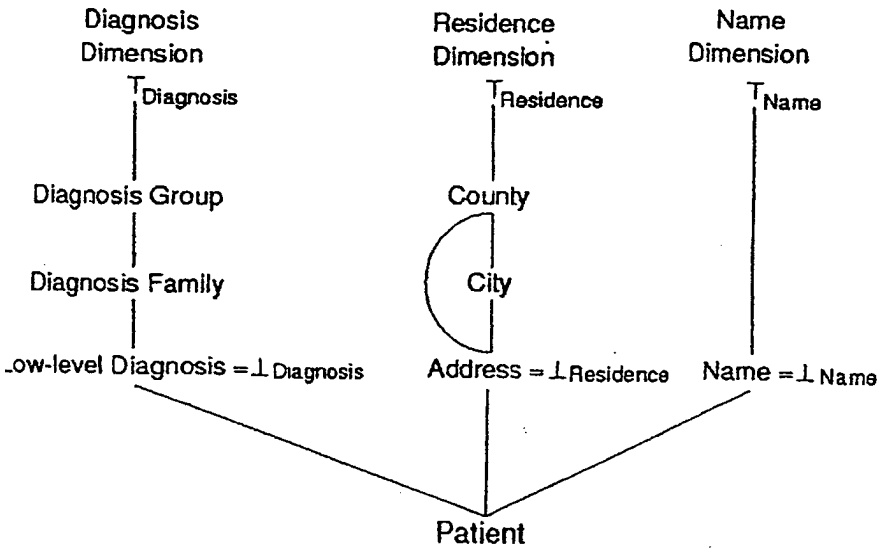


Fig. 2

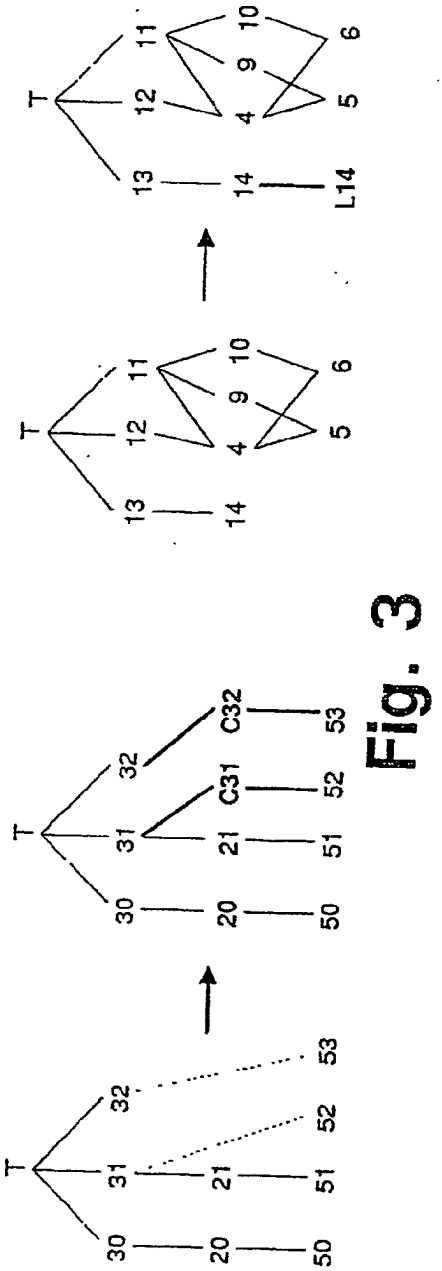

T
30
31
32
20
21
C31
C32
50
51
52
53
T
13
12
11
14
4
9
10
5
6
L14

Fig. 3

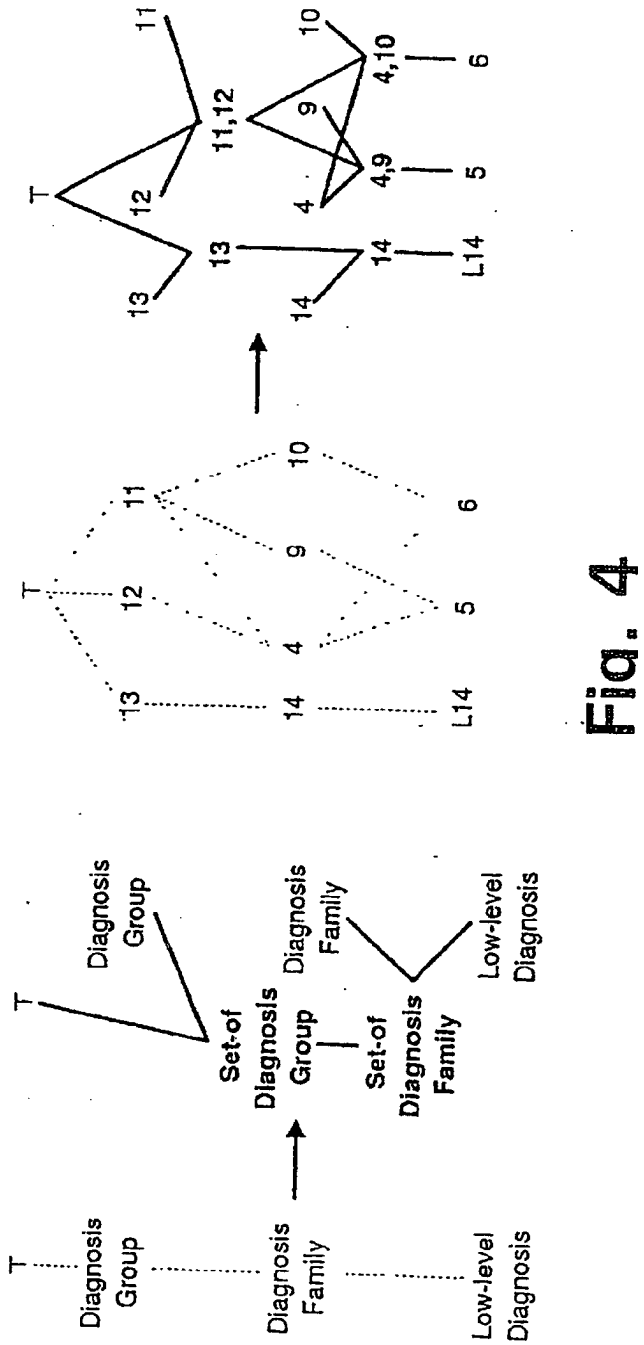

T
13
12
11
14
4
9
10
5
6
L14
11,12
4,9
4,10
Diagnosis Group
Diagnosis Family
Low-level Diagnosis
Set-of Diagnosis Group
Set-of Diagnosis Family

Fig. 4

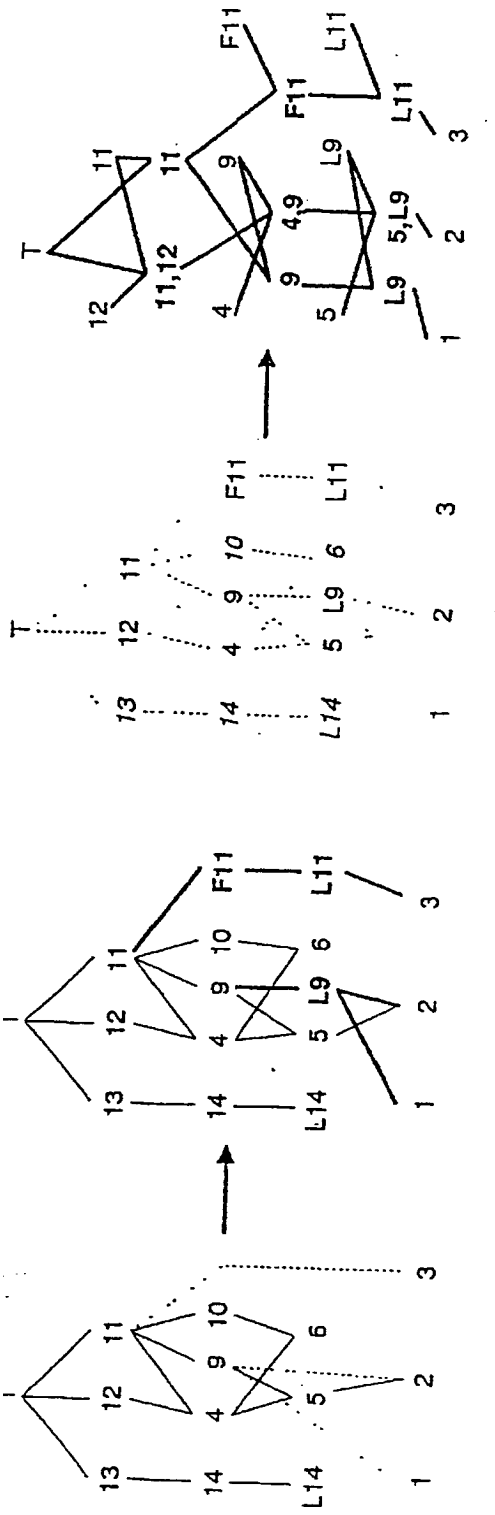

T
12
11,12
11
11
9
4
4,9
9
5
L9
5,L9
L9
F11
F11
L11
L11
1
2
3
T
12
13
14
L14
11
10
9
4
6
5
L9
F11
L11
1
2
3

Fig. 5

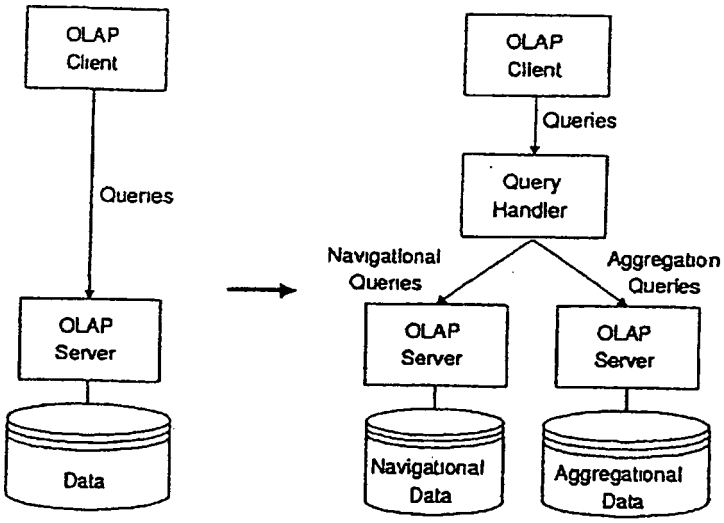

OLAP Client
Queries
OLAP Server
Data
OLAP Client
Queries
Query Handler
Navigational Queries
Aggregation Queries
OLAP Server
OLAP Server
Navigational Data
Aggregational Data

Fig. 6

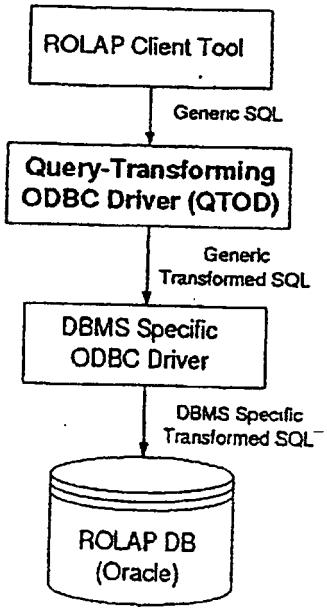

ROLAP Client Tool
Generic SQL
Query-Transforming ODBC Driver (QTOD)
Generic Transformed SQL
DBMS Specific ODBC Driver
DBMS Specific Transformed SQL
ROLAP DB (Oracle)

Fig. 7