(19) Deutsches Patent- und Markenamt

(10) **DE 20 2015 009 875 U1** 2021.01.28

(12) # Gebrauchsmusterschrift

(21) Aktenzeichen: **20 2015 009 875.4**
(22) Anmeldetag: **18.02.2015**
(67) aus Patentanmeldung: **EP 15 75 1938.0**
(47) Eintragungstag: **23.12.2020**
(45) Bekanntmachungstag im Patentblatt: **28.01.2021**

(51) Int Cl.: ***G06F 16/22*** (2019.01)

(30) Unionspriorität:
61/941,986 **19.02.2014** **US**
14/518,931 **20.10.2014** **US**

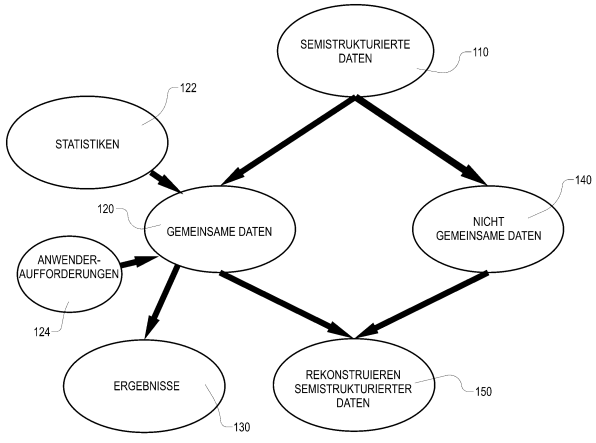
(73) Name und Wohnsitz des Inhabers:
Snowflake Inc., San Mateo, CA, US

(74) Name und Wohnsitz des Vertreters:
Betten & Resch Patent- und Rechtsanwälte PartGmbB, 80333 München, DE

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen.

(54) Bezeichnung: **Transparente Entdeckung eines semistrukturierten Datenschemas**

(57) Hauptanspruch: Ein Computerprogramm das Anweisungen umfasst, die dann, wenn sie von einem oder mehreren Prozessoren ausgeführt werden, veranlassen, dass der eine oder die mehreren Prozessoren Operationen zum Managen semistrukturierter Daten ausführen, die Folgendes umfassen:
Empfangen semistrukturierter Datenelemente von einer Datenquelle, die über ein Computernetzwerk verbunden ist;
Durchführen einer statistischen Analyse an Sammlungen der semistrukturierten Datenelemente, wenn sie der Datenbank über einen Computerprozessor hinzugefügt werden, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppen der semistrukturierten Datenelemente aufweisen, die extrahiert worden sind;
Identifizieren gemeinsamer Datenelemente von innerhalb der semistrukturierten Daten;
Kombinieren gemeinsamer Datenelemente von der Datenquelle in separate Pseudospalten;
Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte im Computerspeicher; und
Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse.



Beschreibung

QUERVERWEIS AUF ZUGEHÖRIGE ANMELDUNGEN

[0001] Diese Anmeldung beansprucht den Vorteil der provisorischen US-Anmeldung mit der seriellen Nr. 61/941,986 mit dem Titel „Apparatus and method for enterprise data warehouse data processing on cloud infrastructure“, eingereicht am 19. Februar 2014, deren Offenbarung hierin durch Bezugnahme ihrer Gesamtheit enthalten ist.

[0002] Unter Schutz gestellt werden und Gegenstand des Gebrauchsmusters sind dabei, entsprechend den Vorschriften des Gebrauchsmustergesetzes, lediglich Vorrichtungen wie in den beigefügten Schutzansprüchen definiert, jedoch keine Verfahren. Soweit nachfolgend in der Beschreibung gegebenenfalls auf Verfahren Bezug genommen wird, dienen diese Bezugnahmen lediglich der beispielhaften Erläuterung der in den beigefügten Schutzansprüchen unter Schutz gestellten Vorrichtung oder Vorrichtungen.

TECHNISCHES GEBIET

[0003] Die vorliegende Offenbarung betrifft Ressourcenmanagementsysteme, die eine Datenspeicherung und Computerressourcen managen.

HINTERGRUND

[0004] Viele existierende Datenspeicherungs- und -wiedergewinnungssysteme sind heutzutage verfügbar. Beispielsweise werden in einem System mit gemeinsam genutzter bzw. geteilter Platte auf einer gemeinsam genutzten bzw. geteilten Speichervorrichtung alle Daten gespeichert, die von allen Verarbeitungsknoten in einem Datencluster aus zugreifbar sind. Bei diesem Typ von System werden alle Datenänderungen zur geteilten Speichervorrichtung geschrieben, um sicherzustellen, dass alle Verarbeitungsknoten im Datencluster auf eine konsistente Version der Daten zugreifen. Wenn sich die Anzahl von Verarbeitungsknoten in einem System mit geteilter Platte erhöht, wird die geteilte Speichervorrichtung (und die Kommunikationsverbindungen zwischen den Verarbeitungsknoten und der geteilten Speichervorrichtung) ein Engpass, der Datenlese- und Datenschreiboperationen verlangsamt. Dieser Engpass wird mit dem Hinzufügen von mehr Verarbeitungsknoten weiter verschlimmert. Somit haben existierende Systeme mit geteilter Platte aufgrund dieses Engpassproblems eine beschränkte Skalierbarkeit.

[0005] Ein weiteres existierendes Datenspeicherungs- und -wiedergewinnungssystem wird „Shared-Nothing-Architektur“ genannt. Bei dieser Architektur werden Daten über mehrere Verarbeitungskonten so verteilt, dass jeder Knoten eine Untergruppe der Daten in der gesamten Datenbank speichert. Wenn ein neuer Verarbeitungsknoten hinzugefügt oder entfernt wird, muss die Shared-Nothing-Architektur Daten über die mehreren Verarbeitungsknoten neu anordnen. Diese Neuanordnung von Daten kann zeitaufwendig und störend für Datenlese- und -schreiboperationen sein, die während der Datenneuanordnung ausgeführt werden. Und die Affinität von Daten zu einem bestimmten Knoten kann „Hotspots“ auf dem Datencluster für populäre Daten erzeugen. Weiterhin erfordert diese Architektur deshalb, weil jeder Verarbeitungsknoten auch die Speicherfunktion durchführt, wenigstens einen Verarbeitungsknoten, um Daten zu speichern. Somit schlägt die Shared-Nothing-Architektur darin fehl, Daten zu speichern, wenn alle Verarbeitungsknoten entfernt sind. Zusätzlich ist ein Management von Daten in einer Shared-Nothing-Architektur aufgrund der Verteilung von Daten über viele unterschiedliche Verarbeitungsknoten komplex.

[0006] Die hierin beschriebenen Systeme und Verfahren stellen einen verbesserten Ansatz für eine Datenspeicherung und eine Datenwiedergewinnung zur Verfügung, der die oben identifizierten Beschränkungen existierender Systeme abmildert.

Figurenliste

[0007] Nicht beschränkende und nicht erschöpfende Ausführungsformen der vorliegenden Offenbarung werden unter Bezugnahme auf die folgenden Figuren beschrieben, wobei sich gleiche Bezugszeichen auf gleiche Teile in allen verschiedenen Figuren beziehen, solange nichts anderes spezifiziert ist.

Fig. 1 stellt ein Informationsfluss- und Beziehungsdiagramm dar, das die Verarbeitung von semistrukturierten Daten darstellt.

Fig. 2 ist ein Prozessablaufdiagramm, das eine Implementierung der hierin offenbarten Verfahren darstellt.

Fig. 3 stellt ein Blockdiagramm dar, das eine Ausführungsform einer Betriebsumgebung gemäß den Lehren der Offenbarung darstellt.

Fig. 4 stellt ein Blockdiagramm dar, das ein Beispiel einer Implementierung eines Ressourcenmanagers gemäß den Lehren der Offenbarung darstellt.

Fig. 5 stellt ein Blockdiagramm dar, das ein Beispiel einer Implementierung einer Ausführungsplattform gemäß den Lehren der Offenbarung darstellt.

Fig. 6 stellt ein Blockdiagramm dar, das eine beispielhafte Computervorrichtung gemäß den Lehren der Offenbarung darstellt.

DETAILLIERTE BESCHREIBUNG

[0008] Hierin sind Verfahren, Vorrichtungen und Systeme zum Managen semistrukturierter Daten offenbart. Zum Beispiel kann eine Implementierung eines Verfahrens zum Managen semistrukturierter Daten semistrukturierte Datenelemente von einer Datenquelle empfangen und kann eine statistische Analyse über Sammlungen der semistrukturierten Datenelemente durchführen, wenn sie zur Datenbank hinzugefügt werden. Zusätzlich können allgemeine bzw. gemeinsame Datenelemente von innerhalb der semistrukturierten Daten identifiziert werden und können sich weiterhin die gemeinsamen Datenelemente von der Datenquelle in separate Pseudospalten kombinieren, die in einem Cachespeicher gespeichert sind. Die Implementierung kann weiterhin Metadaten und Statistiken entsprechend den Pseudospalten für einen computerbasierten Anfragegenerator verfügbar machen und kann nicht allgemeine bzw. nicht gemeinsame Datenelemente in einer serialisierten Spalte für einen Überlauf in einem Computerspeicher speichern.

[0009] In der folgenden Beschreibung wird auf die beigefügten Zeichnungen Bezug genommen, die einen Teil von ihr bilden und in welchen anhand einer Darstellung spezifische beispielhafte Ausführungsformen gezeigt sind, bei welchen die Offenbarung ausgeführt werden kann. Diese Ausführungsformen sind in ausreichendem Detail beschrieben, um Fachleuten auf dem Gebiet zu ermöglichen, die hierin offenbarten Konzepte auszuführen, und es ist zu verstehen, dass Modifikationen an den verschiedenen offenbarten Ausführungsbeispielen durchgeführt werden können und andere Ausführungsbeispiele verwendet werden können, ohne vom Schutzumfang der vorliegenden Offenbarung abzuweichen. Die folgende detaillierte Beschreibung ist daher nicht in einem beschränkenden Sinn anzunehmen.

[0010] Eine Bezugnahme in dieser gesamten Beschreibung auf „eine einzige Ausführungsform“, „eine Ausführungsform“, „ein einziges Beispiel“ oder „ein Beispiel“ bedeutet, dass ein bestimmtes Merkmal, eine bestimmte Struktur oder eine bestimmte Charakteristik, das oder die in Verbindung mit der Ausführungsform oder dem Beispiel beschrieben ist, in wenigstens einer Ausführungsform der vorliegenden Offenbarung enthalten ist. Somit beziehen sich nicht notwendigerweise alle Erscheinungsformen der Phrasen „bei einer einzigen Ausführungsform“, „bei einer Ausführungsform“, „ein einziges Beispiel“ oder „ein Beispiel“ an verschiedenen Stellen in dieser gesamten Beschreibung auf dieselbe Ausführungsform oder dasselbe Beispiel. Zusätzlich sollte erkannt werden, dass die hiermit zur Verfügung gestellten Figuren zu Erklärungszwecken für Personen dienen, die Fachleute auf dem Gebiet sind, und dass die Zeichnungen nicht notwendigerweise maßstäblich gezeichnet sind.

[0011] Ausführungsformen gemäß der vorliegenden Offenbarung können als eine Vorrichtung, ein Verfahren oder ein Computerprogrammprodukt verkörpert sein. Demgemäß kann die vorliegende Offenbarung die Form einer gänzlich hardwareumfassten Ausführungsform, einer gänzlich softwareumfassten Ausführungsform (Firmware, residente Software, eine Mikrocodes, etc. enthaltend) oder einer Ausführungsform, die Software- und Hardwareaspekte kombiniert, auf welche alle hierin allgemein als „Schaltung“, „Modul“ oder „System“ Bezug genommen werden kann, annehmen. Weiterhin können Ausführungsformen der vorliegenden Offenbarung die Form eines Computerprogrammprodukts annehmen, das in irgendeinem konkreten Ausdrucksmedium verkörpert ist, das einen computernutzbaren Programmcode im Medium verkörpert hat.

[0012] Irgendeine Kombination von einem oder mehreren computernutzbaren oder computerlesbaren Medien kann verwendet werden. Beispielsweise kann ein computerlesbares Medium eines oder mehreres von einer tragbaren Computerdiskette, einer Festplatte, einer Direktzugriffsspeicher-(RAM-) Vorrichtung, einer Nurlesespeicher-(ROM-)Vorrichtung, einer löschbaren programmierbaren Nurlesespeicher-(EPROM- oder Flash-Speicher-)Vorrichtung, einem tragbaren Kompaktdisk-Nurlesespeicher (CDROM), einer optischen Speichervorrichtung und einer magnetischen Speichervorrichtung enthalten. Ein Computerprogrammcode zum Ausführen von Operationen der vorliegenden Offenbarung kann in irgendeiner Kombination von einer oder mehreren Programmiersprachen geschrieben sein. Ein solcher Code kann von einem Quellcode zu einer computerlesbaren Assemblersprache oder einen Maschinencode, der für die Vorrichtung oder den Computer geeignet ist, auf der oder dem der Code ausgeführt werden wird, kompiliert werden.

[0013] Ausführungsformen können auch in Cloud-Computerumgebungen implementiert sein. In dieser Beschreibung und den folgenden Ansprüchen kann „Cloudcomputing“ als ein Modell zum Ermöglichen eines ubiquitären, angenehmen On-Demand-Netzwerkzugriffs auf einen gemeinsam genutzten bzw. geteilten Pool von konfigurierbaren Computerressourcen bzw. Rechnerressourcen (z.B. Netzwerken, Servern, Speichern, Anwendungen und Diensten) definiert sein, die über Virtualisierung schnell bereitgestellt und mit minimalem Managementaufwand oder minimaler Serviceprovider-Interaktion freigegeben und dann entsprechend skaliert werden können. Ein Cloud-Modell kann zusammengesetzt sein aus verschiedenen Eigenschaften (z.B. On-demand Self Service, Broad Network Access, Resource Pooling, Rapid Elasticity und Measured Services), Servicemo-

dellen (z.B. Software as a Service („SaaS“), Platform as a Service („PaaS“) und Infrastructure as a Service („IaaS“)) und Bereitstellungsmodellen (Deployment models) (z.B. Private Cloud, Community Cloud, Public Cloud und Hybrid Cloud).

[0014] Die Ablaufdiagramme und Blockdiagramme in den beigefügten Figuren stellen die Architektur, die Funktionalität und die Operation von möglichen Implementierungen von Systemen, Verfahren und Computerprogrammprodukten gemäß verschiedenen Ausführungsformen der vorliegenden Offenbarung dar. Diesbezüglich kann jeder Block in den Ablaufdiagrammen oder Blockdiagrammen ein Modul, ein Segment oder einen Teilbereich eines Codes darstellen, das oder der eine oder mehrere ausführbare Anweisungen zum Implementieren der spezifizierten logischen Funktion(en) umfasst. Es wird auch bemerkt werden, dass jeder Block der Blockdiagramme und/oder Ablaufdiagramme und Kombinationen von Blöcken in den Blockdiagrammen und/oder Ablaufdiagrammen durch spezielle hardwarebasierte Systeme, die die spezifizierten Funktionen oder Handlungen durchführen, oder Kombinationen von spezieller Hardware und Computeranweisungen implementiert sein können. Diese Computerprogrammanweisungen können auch in einem computerlesbaren Medium gespeichert sein, das einen Computer oder eine andere programmierbare Datenverarbeitungsvorrichtung anweisen kann, auf spezielle Weise zu funktionieren, so dass die in dem computerlesbaren Medium gespeicherten Anweisungen ein Herstellungsprodukt erzeugen, das Anweisungsmittel enthält, die die Funktion/Handlung implementieren, die in dem Block oder den Blöcken des Ablaufdiagramms und/oder des Blockdiagramms spezifiziert ist.

[0015] Die hierin beschriebenen Systeme und Verfahren stellen ein flexibles und skalierbares Datenlager unter Verwendung einer neuen Datenverarbeitungsplattform zur Verfügung. Bei einigen Ausführungsformen setzen die beschriebenen Systeme und Verfahren eine Cloud-Infrastruktur ein, die cloudbasierte Speicherressourcen, Rechner- bzw. Computerressourcen und ähnliches unterstützt. Beispielhafte cloudbasierte Speicherressourcen bieten eine signifikante Speicherkapazität, die on-demand bzw. auf Anforderung bei niedrigen Kosten verfügbar ist. Weiterhin können diese cloudbasierten Speicherressourcen fehlertolerant und hochskalierbar sein, was bei privaten Datenspeichersystemen teuer zu erreichen sein kann. Beispielhafte cloudbasierte Computerressourcen sind on-demand verfügbar und für sie kann ein Preis basierend auf tatsächlichen Ausmaßen einer Nutzung der Ressourcen festgesetzt werden. Typischerweise wird die Cloud-Infrastruktur auf eine schnelle Weise dynamisch bereitgestellt, rekonfiguriert und außer Betrieb gesetzt.

[0016] Bei den beschriebenen Systemen und Verfahren verwendet ein Datenspeichersystem eine semistrukturbasierte relationale Datenbank. Jedoch sind diese Systeme und Verfahren anwendbar auf irgendeinen Typ von Datenbank, der irgendeine Datenspeicherarchitektur verwendet und der irgendeine Sprache verwendet, um Daten innerhalb der Datenbank zu speichern und wiederzugewinnen. Wie es hierin verwendet ist, bedeuten semistrukturierte Daten, eine Form von strukturierten Daten weiterzuleiten, die nicht gemäß der typischen formalen Struktur von Datenmodellen sind, die mit einer Relation assoziiert sind, aber nichtsdestoweniger Tags oder andere Markierungen enthalten, um semantische Elemente zu trennen und Hierarchien von Aufzeichnungen und Feldern innerhalb der Daten zu verstärken. Die hierin beschriebenen Systeme und Verfahren stellen weiterhin ein mandantenfähiges System zur Verfügung, das eine Isolierung von Computerressourcen und Daten zwischen unterschiedlichen Kunden/Klienten und zwischen unterschiedlichen Anwendern innerhalb desselben Kunden/Klienten unterstützt.

[0017] Hierin sind Verfahren und Systeme offenbart, die eine Leistungsfähigkeit von Datenbanken und universellen Datenbanksystemen bzw. Lagersystemen signifikant verbessern, die große Mengen von semistrukturierten Daten handhaben. Existierende Datenbanksysteme sind entweder relational (d.h. SQL-Datenbanken) oder Schlüsselwertspeicher.

[0018] Relationale Datenbanken können effiziente Anfragen aufgrund eines Beschneidens eines Anfragedatenzugriffs (Ausschließen von Teilbereichen der Datenbank von der Suche basierend auf aggregierten Metadaten über Werte, die in spezifischen Spalten der Tabellen gespeichert sind) durchführen. Dies erfordert jedoch ein strenges tabulares Format der Daten, welches nicht verwendet werden kann, um semistrukturierte Daten darzustellen.

[0019] Andererseits sind die Schlüsselwertspeicher flexibler, führen aber schwerwiegende Leistungsfähigkeitsnachteile aufgrund eines Mangels an Beschneidung ein. Es gibt eine Anzahl von Wegen, um ein Handhaben von semistrukturierten Daten zu relationalen Datenbanken in existierenden Produkten und Forschungsprojekten hinzuzufügen:

1. Serialisiertes Codieren - eine semistrukturierte Datenaufzeichnung wird in einer Spalte als eine serialisierte Darstellung gespeichert. Jedes Mal wenn ein Wert von einem gewissen Feld verwendet wird, wird er extrahiert und zu einem elementaren Typ konvertiert. Dieses Verfahren ist flexibel, führt aber einen Zugriff auf diese durch Beschneiden zu verbessernde Daten durch, weil eine Extraktion aus einer serialisierten Darstellung kostspielig ist, und erfordert signifikant mehr CPU-Zeit als ein Arbeiten mit normalen relationalen Daten. Die gesamten seria-

lisierten Datenaufzeichnungen müssen aus einem persistenten Speicher gelesen und verarbeitet werden, selbst wenn nur ein äußerst geringer Teilbereich (wie beispielsweise ein einziges Element) von ihnen bei der Anfrage verwendet wird.

2. Umwandlung bzw. Konversion bei einer Aufnahme - die semistrukturierten Daten werden bei der Aufnahme in relationale Daten umgewandelt. Dies macht einen Zugriff auf diese Daten so schnell wie einen Zugriff auf irgendwelche anderen relationalen Daten, erfordert aber eine feststehende Spezifikation einer Datenstruktur bei der Aufnahme und dass ein entsprechendes Datenbankschema im Voraus vollständig spezifiziert ist. Dieses Verfahren macht ein Handhaben von Daten mit einer sich ändernden Struktur aufgrund der Notwendigkeit eines Änderns des Datenbankschemas sehr aufwendig bzw. kostspielig. Daten mit einer Struktur, die sich von Aufzeichnung zu Aufzeichnung ändert, sind unter Verwendung dieses Verfahrens unmöglich zu handhaben. Das Umwandlungsverfahren muss a priori spezifiziert werden und eine nicht triviale Änderung wird ein erneutes Aufnehmen der ursprünglichen semistrukturierten Daten erfordern.

3. Eine relationsartige Darstellung von strukturierten Daten äquivalent zu einer Objektattributenwert-Tripletdarstellung, gespeichert in einer herkömmlichen relationalen Datenbank. Dieses Verfahren ist flexibel, erfordert aber effektiv Verbindungsoperationen für einen Zugriff auf Datenunterkomponenten, was abhängig von Daten sehr langsam sein kann.

4. Nicht herkömmliche Erweiterungen für ein relationales Datenmodell, welche zulassen, dass Spalten mit unterschiedlicher Kardinalität in einer Hierarchie verbunden werden, die eine Struktur der Quelldaten wiedergibt. Die Anfrageerzeugungsverfahren für eine solche Datendarstellung werden nicht gut verstanden (und somit ist keine effektive Anfrageerzeugung mit dem gegenwärtigen Stand der Technik möglich). Dieses Verfahren erfordert auch, dass eingegebene Daten mit einem festen (obwohl nicht tabularen) Schema übereinstimmen, und ist somit nicht ausreichend flexibel, um beliebige semistrukturierte Daten zu handhaben.

[0020] Was nötig ist, ist ein System und ein Verfahren zum Arbeiten mit semistrukturierten Daten, das effizient ist, wenig kostet und reagierend ist, weil es die Semantiken der semistrukturierten Daten bewahren wird, während die Daten in wenigstens Pseudospalten gemanagt werden, die wie herkömmlichere Datenstrukturen verarbeitet und angefragt werden können.

[0021] Bei einer Implementierung der folgenden Offenbarung können Daten in der Form von Dateien, Elementen von Dateien, Teilbereichen von Dateien und ähnliches auftreten. Eine Datei kann eine Sammlung von Dokumenten umfassen und ein Teilbereich von Daten kann eine Datei, eine Vielzahl von Dokumenten aus einer Verbindung und/oder einen Teilbereich von Dokumenten umfassen. Weiterhin können bei der Implementierung Metadaten mit Dateien, Teilbereichen von Dateien und Teilbereichen von Daten assoziiert sein.

[0022] Wie es hierin verwendet ist, ist beabsichtigt, dass die Ausdrücke „gemeinsame Datenelemente“ Datenelemente bedeuten, die zu derselben Gruppe und einer Sammlung von logisch ähnlichen Elementen gehören.

[0023] **Fig. 1** stellt eine Schematik eines semistrukturierten Datenflusses in einem Computersystem dar, das semistrukturierte Daten verarbeitet. Wie es in der Figur gesehen werden kann, können semistrukturierte Daten **110** gemeinsame und nicht gemeinsame Datenelemente darin umfassen. Um gemeinsame Daten aus den semistrukturierten Daten zu extrahieren, können analytische Statistiken **122** gegenüber den semistrukturierten Daten laufengelassen werden, um gemeinsame Datenelemente zu bestimmen. Zusätzlich sollte bemerkt werden, dass auch ein Anwenderinteresse **124** an bestimmten Datenelementen verwendet werden kann, um gemeinsame Daten zu bestimmen. Wie es in der Figur dargestellt ist, können gemeinsame Daten in temporären spaltenartigen Strukturen gespeichert werden, die Pseudospalten **120** genannt werden. Datenelemente, für die nicht bestimmt wird, dass sie gemeinsam sind, können seriell in serialisierten „Überlauf“-Daten **140** gespeichert werden. Letztlich wird ein Anwender Ergebnisse **130** schneller und genau aus den Pseudospalten **120** mit gemeinsamen Daten empfangen.

[0024] Bei der Implementierung kann das angeforderte Datenelement, wenn es nicht in einer Pseudospalte **120** ist, aus den serialisierten „Überlauf“-Daten **140** extrahiert werden, und kann es, wenn eine gesamte semistrukturierte Datenaufzeichnung **150** angefordert ist, aus den extrahierten Datenelementen in Pseudospalten **120** und den „Überlauf“-Daten **140** und den erneut serialisierten rekonstruiert werden.

[0025] **Fig. 2** stellt ein Ablaufdiagramm eines Verfahrens **200** zum Handhaben semistrukturierter Daten dar. Wie es in der Figur gesehen werden kann, werden bei 210 semistrukturierte Datenelemente von einer semistrukturierten Datenquelle oder einer Vielzahl von Semistruktur-Datenquellen empfangen. Dies wird durch Durchführen einer statistischen Analyse der Sammlungen von semistrukturierten Datenaufzeichnungen, wenn sie zur Datenbank hinzugefügt werden, bei **220** und Identifizieren von gemein-

samen Datenelementen bei **230** erreicht. Das System, das die Anweisungen des Verfahrens **200** durchführt, kann weiterhin eine Speicherung von gemeinsamen Datenelementen aus den semistrukturierten Daten in separaten Pseudospalten bei **240** kombinieren, während die weniger gemeinsamen Datenelemente in einer serialisierten „Überlauf"-Darstellung bei **250** gespeichert werden. Diese Elemente werden aus den semistrukturierten Daten extrahiert und separat in spaltenartigem Format unsichtbar für Anwender bei **245** gespeichert, während bei **250** der Rest der semistrukturierten Daten in einem serialisierten Format in einer Hauptspalte bei **255** gespeichert wird. Es wird erkannt werden, dass die Metadaten und Statistiken (wie beispielsweise minimale und maximale Werte, eine Anzahl von eindeutigen Werten, etc.) dieser Pseudospalten dann bei **260** dem Anfragegenerator verfügbar gemacht werden. Es ist zu beachten, dass separate Sammlungen (d.h. Teile der Tabelle, die in den separaten Dateien gespeichert ist) unterschiedliche Untergruppen von extrahierten Datenelementen haben können.

[0026] Bei einer Implementierung kann, wenn Daten extrahiert werden, wenn ein Wert eines gemeinsamen Datenelements nötig ist, er direkt von der entsprechenden Pseudospalte unter Verwendung eines effizienten Spaltenzugriffs erhalten werden.

[0027] Bei einer Implementierung kann ein Bloomfilter verwendet werden, um eine Ressourcennutzung zu steuern. Bloomfilter können Identifizierer von Datenelementen innerhalb von semistrukturierten Daten verwenden, um Daten zu filtern, wenn sie durch das System und Prozesse aufgenommen und verbraucht werden.

[0028] Für einen Anwender kann dieses Verfahren nicht unterscheidbar von einem Speichern serialisierter Aufzeichnungen sein und erlegt einer Struktur von individuellen Datenaufzeichnungen keine Beschränkungen auf. Jedoch deshalb, weil die meisten gemeinsamen Datenelemente auf dieselbe Weise wie herkömmliche relationale Daten gespeichert sind, kann ein Zugriff auf sie zur Verfügung gestellt werden und kann kein Lesen und keine Extraktion der gesamten semistrukturierten Aufzeichnungen erfordern, wodurch die Geschwindigkeitsvorteile von herkömmlichen relationalen Datenbanken gewonnen werden.

[0029] Weil die unterschiedlichen Sammlungen von semistrukturierten Aufzeichnungen (von derselben Tabelle) unterschiedliche Gruppen von extrahierten Datenelementen haben können, sollte der Anfragegenerator und das Beschneiden dazu fähig sein, mit partiell verfügbaren Metadaten (d.h. Teilen der Tabelle, die Metadaten und Statistiken haben, die für ein bestimmtes Datenelement verfügbar sind, während anderen Teilen dies fehlen kann) zu arbeiten.

[0030] Ein Vorteil gegenüber dem Stand der Technik besteht in der Fähigkeit, die durch das Verfahren zum Verwenden einer hybriden Datenspeicherdarstellung (als sowohl eine serialisierte Speicherung von weniger gemeinsamen Elementen als auch eine Spaltenspeicherung von gemeinsamen Elementen) zur Verfügung gestellt ist. Dies lässt zu, dass Anwender sowohl eine Flexibilität als auch eine Fähigkeit erreichen, beliebige semistrukturierte Daten von Systemen unter Verwendung einer serialisierten Darstellung und einer hohen Leistungsfähigkeit von Datenanfragen, die durch herkömmliche relationale Datenbanken zur Verfügung gestellten sind, zu speichern.

[0031] Zusätzlich können semistrukturierte Daten gesamte Dateien, teilweise Dateien, Sammlungen von Dateien und teilweise Sammlungen von Dateien darstellen. Es sollte bemerkt werden, dass ein semistrukturiertes Datenelement eine Datei oder ein Teilbereich einer Datei sein kann. Bei einer Implementierung können Metadaten verwendet werden, um Daten zu definieren und um bei ihrer Organisation und Verwendung zu helfen.

[0032] Es wird von Fachleuten auf dem Gebiet erkannt werden, dass irgendeine Datenverarbeitungsplattform diesen Ansatz zum Handhaben von semistrukturierten Daten verwenden könnte. Es muss nicht auf ein DBMS-System beschränkt sein, das SQL laufen lässt.

[0033] In **Fig. 3** ist ein Computersystem zum Laufenlassen bzw. Durchführen der hierin offenbarten Verfahren dargestellt. Wie es in **Fig. 3** gezeigt ist, ist ein Ressourcenmanager **302** mit mehreren Anwendern **304**, **306** und **308** gekoppelt. Bei bestimmten Implementierungen kann der Ressourcenmanager **302** irgendeine Anzahl von Anwendern unterstützen, die auf eine Datenverarbeitungsplattform **300** zuzugreifen wünschen. Die Anwender **304-308** können beispielsweise Endanwender, die Datenspeicherungs- und -wiedergewinnungsaufforderungen zur Verfügung stellen, Systemadministratoren, die die Systeme managen, und Verfahren, die hierin beschrieben sind, Softwareanwendungen, die mit einer Datenbank interagieren, und andere Komponenten/Vorrichtungen, die mit dem Ressourcenmanager **302** interagieren, enthalten. Der Ressourcenmanager **302** stellt verschiedene Dienste und Funktionen zur Verfügung, die den Betrieb von allen Systemen und Komponenten innerhalb der Datenverarbeitungsplattform **300** unterstützen. Der Ressourcenmanager **302** ist auch mit Metadaten **310** gekoppelt, die mit der Gesamtheit von Daten assoziiert sind, die in der gesamten Datenverarbeitungsplattform **300** gespeichert sind. Weil der Ressourcenmanager mit den Metadaten entsprechend Gruppen von Dateien gekoppelt ist, können die Metadaten zum Erzeugen von Anwenderanfragen verwendet werden. Bei einigen Ausführungsformen enthalten die Metadaten **310**

eine Zusammenfassung von Daten, die in entfernten Datenspeichersystemen gespeichert sind, sowie Daten, die von einem lokalen Cache verfügbar sind. Zusätzlich können die Metadaten **310** Information diesbezüglich enthalten, wie Daten in den entfernten Datenspeichersystemen und den lokalen Caches organisiert sind. Die Metadaten **310** lassen zu, dass Systeme und Dienste bestimmen, ob ein Datenstück bzw. ein Datenwert verarbeitet werden muss, ohne die tatsächlichen Daten von einer Speichervorrichtung zu laden oder auf diese zuzugreifen.

[0034] Der Ressourcenmanager **302** ist weiterhin mit einer Ausführungsplattform **312** gekoppelt, die mehrere Computerressourcen zur Verfügung stellt, die verschiedene Datenspeicherungs- und Datenwiedergewinnungsaufgaben ausführen, wie es nachstehend ausführlicher diskutiert ist. Die Ausführungsplattform **312** ist mit mehreren Datenspeichervorrichtungen **316**, **318** und **320** gekoppelt, die Teil einer Speicherplattform **314** sind. Obwohl in **Fig. 3** drei Datenspeichervorrichtungen **316**, **318** und **320** gezeigt sind, kann die Ausführungsplattform **312** mit irgendeiner Anzahl von Datenspeichervorrichtungen kommunizieren. Bei einigen Ausführungsformen sind die Datenspeichervorrichtungen **316**, **318** und **320** cloudbasierte Speichervorrichtungen, die an einem oder mehreren geographischen Standorten angeordnet sind. Beispielsweise können die Datenspeichervorrichtungen **316**, **318** und **320** Teil einer öffentlichen Cloud-Infrastruktur oder einer privaten Cloud-Infrastruktur sein. Die Datenspeichervorrichtungen **316**, **318** und **320** können Festplattenlaufwerke (HDDs), Festkörperlaufwerke (SSDs), Speichercluster oder irgendeine andere Datenspeichertechnologie sein. Zusätzlich kann die Speicherplattform **314** verteilte Dateiensysteme (wie beispielsweise verteilte Hadoop-Dateiensysteme (HDFS)), Objektspeichersysteme und ähnliches sein.

[0035] Bei bestimmten Ausführungsformen sind die Kommunikationsverbindungen zwischen dem Ressourcenmanager **302** und den Anwendern **304-308**, den Metadaten **310** und der Ausführungsplattform **312** über ein oder mehrere Datenkommunikationsnetzwerke implementiert. Gleichermaßen sind die Kommunikationsverbindungen zwischen der Ausführungsplattform **312** und den Datenspeichervorrichtungen **316-320** in der Speicherplattform **314** über ein oder mehrere Datenkommunikationsnetzwerke implementiert. Diese Datenkommunikationsnetzwerke können irgendein Kommunikationsprotokoll und irgendeinen Typ von Kommunikationsmedium verwenden. Bei einigen Ausführungsformen sind die Datenkommunikationsnetzwerke eine Kombination aus zwei oder mehr Datenkommunikationsnetzwerken (oder Unternetzwerken), die miteinander gekoppelt sind. Bei alternativen Ausführungsformen sind diese Kommunikationsverbindungen unter Verwendung von irgendeinem Typ von Kommunikationsmedium und irgendeinem Kommunikationsprotokoll implementiert.

[0036] Wie es in **Fig. 3** gezeigt ist, sind die Datenspeichervorrichtungen **316**, **318** und **320** von den Computerressourcen entkoppelt, die mit der Ausführungsplattform **312** assoziiert sind. Diese Architektur unterstützt dynamische Änderungen an der Datenverarbeitungsplattform **300** basierend auf den sich ändernden Datenspeicherungs-/-wiedergewinnungs-Notwendigkeiten sowie den sich ändernden Notwendigkeiten der Anwender und Systeme, die auf die Datenverarbeitungsplattform **300** zugreifen. Die Unterstützung von dynamischen Änderungen lässt zu, dass sich die Datenverarbeitungsplattform **300** in Reaktion auf Änderungsanforderungen an die Systeme und Komponenten innerhalb der Datenverarbeitungsplattform **300** schnell skaliert. Die Entkopplung der Computerressourcen von den Datenspeichervorrichtungen unterstützt die Speicherung großer Mengen von Daten, ohne eine entsprechende große Menge von Computerressourcen zu erfordern. Gleichermaßen unterstützt diese Entkopplung von Ressourcen eine signifikante Erhöhung bezüglich der Computerressourcen, die zu einer bestimmten Zeit verwendet werden, ohne eine entsprechende Erhöhung bezüglich der verfügbaren Datenspeicherressourcen zu erfordern.

[0037] Der Ressourcenmanager **302**, die Metadaten **310**, die Ausführungsplattform **312** und die Speicherplattform **314** sind in **Fig. 3** als einzelne Komponenten gezeigt. Jedoch kann jedes von dem Ressourcenmanager **302**, den Metadaten **310**, der Ausführungsplattform **312** und der Speicherplattform **314** als ein verteiltes System implementiert sein (z.B. verteilt über mehrere Systeme/Plattformen an mehreren geographischen Standorten). Zusätzlich kann jedes von dem Ressourcenmanager **302**, den Metadaten **310**, der Ausführungsplattform **312** und der Speicherplattform **314** (unabhängig voneinander) in Abhängigkeit von Änderungen an den von den Anwendern **304-308** empfangenen Aufforderungen und den sich ändernden Notwendigkeiten der Datenverarbeitungsplattform **300** vergrößert oder verkleinert werden. Somit ist die Datenverarbeitungsplattform **300** bei den beschriebenen Ausführungsformen dynamisch und unterstützt reguläre Änderungen, um die aktuellen Datenverarbeitungsnotwendigkeiten zu erfüllen.

[0038] **Fig. 4** ist ein Blockdiagramm, das eine Ausführungsform des Ressourcenmanagers **302** darstellt. Wie es in **Fig. 3** gezeigt ist, enthält der Ressourcenmanager **302** einen Zugriffsmanager **402** und einen Schlüsselmanager **404**, die mit einer Datenspeichervorrichtung **406** gekoppelt sind. Der Zugriffsmanager **402** handhabt Authentifizierungs- und Autorisierungsaufgaben für die hierin beschriebenen Systeme. Der Schlüsselmanager **404** managt

eine Speicherung und Authentifizierung von während Authentifizierungs- und Autorisierungsaufgaben verwendeten Schlüsseln. Ein Aufforderungsverarbeitungsservice bzw. -dienst 408 managt empfangene Datenspeicherungsaufforderungen und Datenwiedergewinnungsaufforderungen. Ein Managementkonsolenservice bzw. -dienst 410 unterstützt einen Zugriff auf verschiedene Systeme und Prozesse durch Administratoren und andere Systemmanager.

[0039] Der Ressourcenmanager **302** enthält auch einen SQL-Compiler **412**, einen SQL-Optimierer **414** und einen SQL-Ausführer **410**. Der SQL-Compiler **412** zerlegt SQL-Anfragen und erzeugt den Ausführungscode für die Anfragen. Der SQL-Optimierer **414** bestimmt das beste Verfahren, um Anfragen auszuführen, basierend auf den Daten, die verarbeitet werden müssen. Der SQL-Ausführer **416** führt den Anfragecode für durch den Ressourcenmanager **302** empfangene Anfragen aus. Ein Anfrageplaner und -koordinator 418 sendet empfangene Anfragen zu den geeigneten Diensten oder Systemen zur Kompilation, Optimierung und Abfertigung zu einer Ausführungsplattform. Ein virtueller universeller Datenbankmanager bzw. virtueller Lagermanager bzw. Manager für virtuelle Lager **420** managt die Operation von mehreren virtuellen Lagern, die in einer Ausführungsplattform implementiert sind.

[0040] Zusätzlich enthält der Ressourcenmanager **302** einen Konfigurations- und Metadatenmanager **422**, der die Information in Bezug auf die Daten managt, die in den entfernten Datenspeichervorrichtungen und in den lokalen Caches gespeichert sind. Ein Überwachungs- und Arbeitsbelastungsanalysator **424** überblickt die durch den Ressourcenmanager **302** durchgeführten Prozesse und managt die Verteilung von Aufgaben (z.B. Arbeitsbelastung) über die virtuellen Lager und Ausführungsknoten in der Ausführungsplattform. Der Konfigurations- und Metadatenmanager **422** und der Überwachungs- und Arbeitsbelastungsanalysator **424** sind mit einer Datenspeichervorrichtung **426** gekoppelt.

[0041] Der Ressourcenmanager **302** enthält auch ein Transaktionsmanagement- und Zugriffssteuermodul **428**, das die verschiedenen Aufgaben und andere Aktivitäten, die mit der Verarbeitung von Datenspeicheranfragen und Datenzugriffsanfragen assoziiert sind, managt. Beispielsweise stellt das Transaktionsmanagement- und Zugriffssteuermodul **428** einen konsistenten und synchronisierten Zugriff auf Daten durch mehrere Anwender oder Systeme zur Verfügung. Da mehrere Anwender/Systeme auf dieselben Daten gleichzeitig zugreifen können, müssen Änderungen an den Daten synchronisiert werden, um sicherzustellen, dass jeder Anwender/jedes System mit der aktuellen Version der Daten arbeitet. Das Transaktionsmanagement- und Zugriffssteuermodul **428** stellt eine Steuerung von verschiedenen Datenverarbeitungsaktivitäten an einem einzelnen, zentralisierten Standort im Ressourcenmanager **302** zur Verfügung.

[0042] **Fig. 5** ist ein Blockdiagramm, das eine Ausführungsform einer Ausführungsplattform darstellt. Wie es in **Fig. 5** gezeigt ist, enthält die Ausführungsplattform **512** mehrere virtuelle universelle Datenbanken bzw. Lager **502**, **504** und **506**. Jedes virtuelle Lager enthält mehrere Ausführungsknoten, die jeweils einen Cache und einen Prozessor enthalten. Obwohl jedes in **Fig. 5** gezeigte virtuelle Lager **502-506** drei Ausführungsknoten enthält, kann ein bestimmtes virtuelles Lager irgendeine Anzahl von Ausführungsknoten enthalten. Weiterhin ist die Anzahl von Ausführungsknoten in einem virtuellen Lager dynamisch, so dass neue Ausführungsknoten erzeugt werden, wenn eine zusätzlicher Bedarf vorhanden ist, und existierende Ausführungsknoten gelöscht werden, wenn sie nicht länger nötig sind.

[0043] Jede virtuelle Lager **502-506** kann auf irgendeine der in **Fig. 3** gezeigten Datenspeichervorrichtungen **316-320** zugreifen. Somit sind die virtuellen Lager **502**, **506** nicht notwendigerweise einer spezifischen Datenspeichervorrichtung **316-320** zugeordnet und können stattdessen auf Daten von irgendeiner der Datenspeichervorrichtungen **316-320** zugreifen. Gleichermaßen kann jeder der in **Fig. 5** gezeigten Ausführungsknoten auf Daten von irgendeiner der Datenspeichervorrichtungen **316-320** zugreifen. Bei einigen Ausführungsformen kann ein bestimmtes virtuelles Lager oder ein bestimmter Ausführungsknoten temporär einer spezifischen Datenspeichervorrichtung zugeordnet sein, aber das virtuelle Lager oder der Ausführungsknoten können später auf Daten von irgendeiner anderen Datenspeichervorrichtung zugreifen.

[0044] Bei dem Beispiel der **Fig. 5** enthält das virtuelle Lager **502** drei Ausführungsknoten **508**, **510** und **512**. Der Ausführungsknoten **508** enthält einen Cache **514** und einen Prozessor **516**. Der Ausführungsknoten **510** enthält einen Cache **518** und einen Prozessor **520**. Der Ausführungsknoten **512** enthält einen Cache **522** und einen Prozessor **524**. Jeder Ausführungsknoten **508-512** ist mit einer Verarbeitung von einer oder mehreren Datenspeicherungs- und/oder Datenwiedergewinnungsaufgaben assoziiert. Beispielsweise kann ein bestimmtes virtuelles Lager mit einem bestimmten Anwender oder Kunden assoziierte Datenspeicherungs- und Datenwiedergewinnungsaufgaben handhaben. Bei anderen Implementierungen kann ein bestimmtes virtuelles Lager mit einem bestimmten Datenspeichersystem oder einer bestimmten Kategorie von Daten assoziierte Datenspeicherungs- und Datenwiedergewinnungsaufgaben handhaben.

[0045] Gleich der oben diskutierten virtuellen Lager **502** enthält das virtuelle Lager **504** drei Ausführungsknoten **526**, **528** und **530**. Der Ausführungsknoten **526** enthält einen Cache **532** und einen Prozessor **534**. Der Ausführungsknoten **528** enthält einen Cache **536** und einen Prozessor **538**. Der Ausführungsknoten **530** enthält einen Cache **540** und einen Prozessor **542**. Zusätzlich enthält das virtuelle Lager **506** drei Ausführungsknoten **544**, **546** und **548**. Der Ausführungsknoten **544** enthält einen Cache **550** und einen Prozessor **552**. Der Ausführungsknoten **546** enthält einen Cache **554** und einen Prozessor **556**. Der Ausführungsknoten **548** enthält einen Cache **558** und einen Prozessor **560**.

[0046] Obwohl der in **Fig. 5** gezeigte Ausführungsknoten jeweils einen Cache und einen Prozessor enthält, können alternative Ausführungsformen Ausführungsknoten enthalten, die irgendeine Anzahl von Prozessoren und irgendeine Anzahl von Caches enthalten. Zusätzlich können die Caches bezüglich einer Größe unter den unterschiedlichen Ausführungsknoten variieren. Die in **Fig. 5** gezeigten Caches speichern in dem lokalen Ausführungsknoten Daten, die aus einer oder mehreren Datenspeichervorrichtungen in einer Speicherplattform **314** (**Fig. 3**) wiedergewonnen wurden. Somit reduzieren oder eliminieren die Caches die Engpassprobleme, die bei Plattformen auftreten, die Daten von entfernten Speichersystemen konsistent wiedergewinnen. Anstelle eines wiederholten Zugreifens auf Daten aus den entfernten Speichervorrichtungen greifen die hierin beschriebenen Systeme und Verfahren auf Daten aus den Caches in den Ausführungsknoten zu, was signifikant schneller ist und das oben diskutierte Engpassproblem vermeidet. Bei einigen Ausführungsformen sind die Caches unter Verwendung von Hochgeschwindigkeitsspeichervorrichtungen implementiert, die einen schnellen Zugriff auf die im Cache gespeicherten Daten zur Verfügung stellen. Jeder Cache kann Daten von irgendeiner der Speichervorrichtungen in der Speicherplattform **314** speichern.

[0047] Weiterhin können die Cacheressourcen und Computerressourcen zwischen unterschiedlichen Ausführungsknoten variieren. Beispielsweise kann ein Ausführungsknoten signifikante Computerressourcen und minimale Cacheressourcen enthalten, was den Ausführungsknoten nützlich für Aufgaben macht, die signifikante Computerressourcen erfordern. Ein weiterer Ausführungsknoten kann signifikante Cacheressourcen und minimale Computerressourcen enthalten, was diesen Ausführungsknoten nützlich für Aufgaben macht, die ein Speichern von großen Mengen von Daten im Cache erfordern. Bei einigen Ausführungsformen werden die Cacheressourcen und Computerressourcen, die mit einem bestimmten Ausführungsknoten assoziiert sind, basierend auf den durch den Ausführungsknoten durchzuführenden erwarteten Aufgaben bestimmt, wenn der Ausführungsknoten erzeugt wird.

[0048] Zusätzlich können sich die Cacheressourcen und Computerressourcen, die mit einem bestimmten Ausführungsknoten assoziiert sind, im Laufe der Zeit basierend auf sich ändernden Aufgaben ändern, die durch den Ausführungsknoten durchgeführt werden. Beispielsweise kann ein bestimmter Ausführungsknoten mehr prozessorintensiv sein. Gleichermaßen können einem Ausführungsknoten mehr Cacheressourcen zugeordnet sein, wenn die durch den Ausführungsknoten durchgeführten Aufgaben eine größere Cachekapazität erfordern.

[0049] Obwohl die virtuellen Lager **502-506** mit derselben Ausführungsplattform **312** der **Fig. 3** assoziiert sind, können die virtuellen Lager unter Verwendung von mehreren Computersystemen an mehreren geographischen Standorten implementiert sein. Beispielsweise kann das virtuelle Lager **502** durch ein Computersystem an einem ersten geographischen Standort implementiert sein, während die virtuellen Lager **504** und **506** durch ein anderes Computersystem an einem zweiten geographischen Standort implementiert sind. Bei einigen Ausführungsbeispielen sind diese unterschiedlichen Computersysteme cloudbasierte Computersysteme, die durch eine oder mehrere unterschiedliche Einheiten gewartet bzw. erhalten bzw. gepflegt werden.

[0050] Zusätzlich ist in **Fig. 5** jedes virtuelle Lager derart gezeigt, dass es mehrere Ausführungsknoten hat. Die mehreren Ausführungsknoten, die mit jedem virtuellen Lager assoziiert sind, können unter Verwendung von mehreren Computersystemen an mehreren geographischen Standorten implementiert sein. Beispielsweise implementiert ein bestimmter Fall eines virtuellen Lagers **502** Ausführungsknoten **508** und **510** auf einer Computerplattform an einem bestimmten geographischen Standort und implementiert den Ausführungsknoten **512** einer anderen Computerplattform an einem anderen geographischen Standort. Ein Auswählen bestimmter Computersysteme, um einen Ausführungsknoten zu implementieren, kann von verschiedenen Faktoren abhängen, wie beispielsweise dem Ausmaß an Ressourcen, die für einen bestimmten Ausführungsknoten nötig sind (z.B. Verarbeitungsressourcenerfordernisse und Cache-Erfordernisse), den Ressourcen, die bei bestimmten Computersystemen verfügbar sind, den Kommunikationsfähigkeiten von Netzwerken innerhalb eines geographischen Standorts oder zwischen geographischen Standorten, und welche Computersysteme bereits andere Ausführungsknoten im virtuellen Lager implementieren. Die Ausführungsplattform **312** ist auch fehlertolerant. Wenn beispielsweise ein virtuelles Lager ausfällt, wird dieses virtuelle Lager schnell durch ein anderes virtuelles Lager an einem anderen geographischen Standort ersetzt.

[0051] Eine bestimmte Ausführungsplattform **312** kann irgendeine Anzahl von virtuellen Lagern **502-506** enthalten. Zusätzlich ist die Anzahl von virtuellen Lagern bei einer bestimmten Ausführungsplattform dynamisch, so dass neue virtuelle Lager erzeugt werden, wenn zusätzliche Verarbeitungs- und/oder Cache-Ressourcen nötig sind. Gleichermaßen können existierende virtuelle Lager gelöscht werden, wenn die mit dem virtuellen Lager assoziierten Ressourcen nicht länger nötig sind.

[0052] Fig. 6 ist ein Blockdiagramm, das eine beispielhafte Computervorrichtung **600** darstellt. Bei einigen Ausführungsbeispielen wird die Computervorrichtung **600** verwendet, um eines oder mehreres von den Systemen und den Komponenten zu implementieren, die hierin diskutiert sind. Beispielsweise kann die Computervorrichtung **600** zulassen, dass ein Anwender oder ein Administrator auf den Ressourcenmanager **302** zugreift. Weiterhin kann die Computervorrichtung **600** mit irgendeinem der Systeme und der Komponenten, die hierin beschrieben sind, interagieren. Demgemäß kann die Computervorrichtung **600** verwendet werden, um verschiedene Prozeduren und Aufgaben durchzuführen, wie beispielsweise diejenigen, die hierin diskutiert sind. Die Computervorrichtung **600** kann als Server, Client oder irgendeine andere Computereinheit fungieren. Die Computervorrichtung **600** kann irgendetwas von einer breiten Vielfalt von Computervorrichtungen sein, wie beispielsweise ein Desktopcomputer, ein Notebookcomputer, ein Servercomputer, ein Handheld-Computer bzw. in der Hand gehaltener Computer, ein Tablet und ähnliches.

[0053] Die Computervorrichtung **600** enthält einen oder mehrere Prozessoren **602**, eine oder mehrere Speichervorrichtungen **604**, eine oder mehrere Schnittstellen **606**, eine oder mehrere Massenspeichervorrichtungen **608** und eine oder mehrere Eingabe/Ausgabe-(I/O-)Vorrichtungen **610**, von welchen alle mit einem Bus **612** gekoppelt sind. Der Prozessor/die Prozessoren **602** enthält (enthalten) einen oder mehrere Prozessoren oder Steuerungen, die Anweisungen ausführen, die in der Speichervorrichtung (den Speichervorrichtungen) 604 und/oder der Massenspeichervorrichtung (den Massenspeichervorrichtungen) 608 gespeichert sind. Der Prozessor (die Prozessoren) **602** kann (können) auch verschiedene Typen von computerlesbaren Medien enthalten, wie beispielsweise einen Cachespeicher.

[0054] Die Speichervorrichtung(en) **604** enthält (enthalten) verschiedene computerlesbare Medien, wie beispielsweise einen flüchtigen Speicher (z.B. einen Direktzugriffsspeicher (RAM)) und/oder einen nichtflüchtigen Speicher (z.B. einen Nurlesespeicher (ROM)). Die Speichervorrichtung(en) **604** kann (können) auch einen wiederbeschreibbaren ROM, wie beispielsweise einen Flash-Speicher, enthalten.

[0055] Die Massenspeichervorrichtung(en) **608** enthält (enthalten) verschiedene computerlesbare Medien, wie beispielsweise Magnetbänder, magnetische Platten, optische Platten, einen Festkörperspeicher (z.B. einen Flash-Speicher), und so weiter. Verschiedene Vorrichtungen können auch in der Massenspeichervorrichtung (den Massenspeichervorrichtungen) 608 enthalten sein, um ein Lesen von und/oder ein Schreiben zu den verschiedenen computerlesbaren Medien zu ermöglichen. Die Massenspeichervorrichtung(en) **608** enthält (enthalten) entfernbare Medien und/oder nicht entfernbare Medien.

[0056] Die I/O-Vorrichtung(en) **610** enthält (enthalten) verschiedene Vorrichtungen, die zulassen, dass Daten und/oder andere Information zu der Computervorrichtung **600** eingegeben oder daraus wiedergewonnen werden (wird). Eine beispielhafte I/O-Vorrichtung (Beispielhafte I/O-Vorrichtungen) **610** enthält (enthalten) Cursor-Steuervorrichtungen, Tastaturen, Keypads bzw. kleine Tastaturen bzw. Folientastaturen, Mikrofone, Monitore oder andere Anzeigevorrichtungen, Lautsprecher, Drucker, Netzwerkschnittstellenkarten, Modems, Optik bzw. Objektive, CCDs oder andere Bilderfassungsvorrichtungen und ähnliches.

[0057] Die Schnittstelle(n) **606** enthält (enthalten) verschiedene Schnittstellen, die zulassen, dass die Computervorrichtung **600** mit anderen Systemen, Vorrichtungen oder Computerumgebungen interagiert. Eine beispielhafte Schnittstelle (beispielhafte Schnittstellen) **606** enthält (enthalten) irgendeine Anzahl von unterschiedlichen Netzwerkschnittstellen, wie beispielsweise Schnittstellen zu lokalen Netzen (LAN), Weitverkehrsnetzen (WAN), drahtlosen Netzen und dem Internet.

[0058] Der Bus **612** lässt zu, dass der Prozessor (die Prozessoren) **602**, die Speichervorrichtung(en) **604**, die Schnittstelle(n) **606**, die Massenspeichervorrichtung(en) **608** und die I/O-Vorrichtung(en) **610** miteinander sowie mit anderen Vorrichtungen oder Komponenten, die mit dem Bus **612** gekoppelt sind, kommunizieren. Der Bus stellt einen oder mehrere von einigen Typen von Busstrukturen dar, wie beispielsweise einen Systembus, einen PCI-Bus, einen IEEE-1394-Bus, einen USB-Bus und so weiter.

[0059] Zu Darstellungszwecken sind Programme und andere ausführbare Programmkomponenten hierin als diskrete Blöcke gezeigt, obwohl es verstanden wird, dass sich solche Programme und Komponenten zu verschiedenen Zeiten in unterschiedlichen Speicherkomponenten der Computervorrichtung **660** befinden können, und werden durch den Prozessor (die Prozessoren) **602** ausgeführt. Alternativ dazu können die hierin beschriebenen Systeme und Prozeduren in Hardware implementiert sein, oder einer Kombination aus Hardware, Software und/

oder Firmware. Beispielsweise kann oder können eine oder mehrere anwendungsspezifische integrierte Schaltungen (ASICs) programmiert sein, um eines oder mehreres von den hierin beschriebenen Systemen und Prozeduren auszuführen.

[0060] Obwohl die vorliegende Offenbarung in Bezug auf bestimmte bevorzugte Ausführungsformen beschrieben ist, werden Fachleuten auf dem Gebiet andere Ausführungsformen offensichtlich werden, die den Vorteil dieser Offenbarung ergeben, einschließlich von nicht alle der hierin aufgezeigten Vorteile und Merkmale zur Verfügung stellenden Ausführungsformen, die auch innerhalb des Schutzumfangs dieser Offenbarung sind. Es ist zu verstehen, dass andere Ausführungsformen verwendet werden können, ohne vom Schutzumfang der vorliegenden Offenbarung abzuweichen.

[0061] Weitere Implementierungen sind in den folgenden Beispielen zusammengefasst:

Beispiel 1. Computerprogramm mit Anweisungen, die dann, wenn sie durch einen oder mehrere Prozessoren ausgeführt werden, veranlassen, dass der eine oder die mehreren Prozessoren Operationen durchführt oder durchführen, die folgendes umfassen: Empfangen semistrukturierter Datenelemente von einer Datenquelle, die über ein Computernetzwerk angeschlossen ist; Durchführen einer statistischen Analyse an Sammlungen der semistrukturierten Datenelemente, wenn sie über einen Computerprozessor zur Datenbank hinzugefügt werden; Identifizieren gemeinsamer Datenelemente von innerhalb der semistrukturierten Daten; Kombinieren gemeinsamer Datenelemente aus der Datenquelle in separate Pseudospalten; Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte in einem Computerspeicher; und Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse.

Beispiel 2. Computerprogramm nach Beispiel 1, wobei die gemeinsamen Datenelemente aus den semistrukturierten Daten extrahiert und separat in einem Spaltenformat gespeichert werden.

Beispiel 3. Computerprogramm nach Beispiel 2, wobei die Pseudospalten nicht sichtbar für Anwender gemacht werden.

Beispiel 4. Computerprogramm nach Beispiel 1, wobei die Operationen weiterhin ein Speichern der nicht gemeinsamen semistrukturierten Daten in einem serialisierten Format in einer Hauptspalte umfassen.

Beispiel 5. Computerprogramm nach Beispiel 1, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppen der semistrukturierten Datenelemente haben, die extrahiert worden sind.

Beispiel 6. Computerprogramm nach Beispiel 1, wobei die Operationen weiterhin ein Filtern mit einem Bloomfilter umfassen, das Identifizierer von Datenelementen umfasst, die innerhalb der separaten Sammlungen enthalten sind.

Beispiel 7. Computerprogramm nach Beispiel 1, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen: minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen.

Beispiel 8. Computerprogramm nach Beispiel 1, wobei die Operationen weiterhin ein Extrahieren von Datenelementen von den serialisierten Überlaufdaten umfassen, wenn das angeforderte Datenelement nicht in einer Pseudospalte ist.

Beispiel 9. Computerprogramm nach Beispiel 1, wobei die Operationen weiterhin ein Rekonstruieren von semistrukturierten Daten zu einer ursprünglichen Form durch Extrahieren von Datenelementen aus Pseudospalten und aus den serialisierten Überlaufdaten umfassen.

Beispiel 10. Computerprogramm nach Beispiel 1, wobei die Operationen weiterhin ein Updaten der Metadaten entsprechend der Pseudospalten umfassen, wenn zusätzliche semistrukturierte Daten empfangen werden.

Beispiel 11. System zum Aggregieren bzw. Anhäufen semistrukturierter Daten, umfassend: ein Empfangsmodul, das konfiguriert ist, um semistrukturierte Datenelemente von einer Datenquelle zu empfangen; ein Statistikmodul, das konfiguriert ist, um eine statistische Analyse an Sammlungen der semistrukturierten Datenelemente durchzuführen, wenn sie zur Datenbank hinzugefügt werden; ein Aggregationsmodul, das konfiguriert ist, um gemeinsame Datenelemente von innerhalb der semistrukturierten Daten zu identifizierten und die gemeinsamen Datenelemente von der Datenquelle in separate Pseudospalten zu kombinieren; ein Serialisierungsmodul, das konfiguriert ist, um nicht gemeinsame semistrukturierte Datenelemente in einer serialisierten Überlaufspalte zu speichern; und wobei das Aggregationsmodul weiterhin konfiguriert ist, um Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse abzuleiten.

Beispiel 12. System nach Beispiel 11, wobei die gemeinsamen Datenelemente von den semi-

strukturierten Daten extrahiert und separat in einem Spaltenformat gespeichert werden.

Beispiel 13. System nach Beispiel 11, wobei das Spaltenformat für Anwender unsichtbar ist.

Beispiel 14. System nach Beispiel 11, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppen der semistrukturierten Datenelemente haben, die extrahiert worden sind.

Beispiel 15. System nach Beispiel 11, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen: minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, und
eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen

Beispiel 16. Vorrichtung zum Aggregieren semistrukturierter Daten, umfassend: ein Empfangsmodul, das konfiguriert ist, um semistrukturierte Datenelemente von einer Datenquelle zu empfangen; ein Statistikmodul, das konfiguriert ist, um eine statistische Analyse an Sammlungen der semistrukturierten Datenelemente durchzuführen, wenn sie zur Datenbank hinzugefügt werden; eine Aggregationseinrichtung zum Identifizieren gemeinsamer Datenelemente von innerhalb der semistrukturierten Daten und zum Kombinieren der gemeinsamen Datenelemente von der Datenquelle in separate Pseudospalten; wobei die Aggregationseinrichtung weiterhin zum Serialisieren und Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte dient; und wobei die Aggregationseinrichtung weiterhin zum Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse dient.

Beispiel 17. Vorrichtung nach Beispiel 16, wobei die gemeinsamen Datenelemente aus den semistrukturierten Daten extrahiert und separat in einem Spaltenformat gespeichert werden. [0088] Beispiel 18. Vorrichtung nach Beispiel 16, wobei das Spaltenformat für Anwender unsichtbar ist.

Beispiel 19. Vorrichtung nach Beispiel 16, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppe der semistrukturierten Datenelemente haben, die extrahiert worden sind.

Beispiel 20. Vorrichtung nach Beispiel 16, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen: minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, und eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen.

Beispiel 21. Vorrichtung nach Beispiel 16, wobei die Aggregationseinrichtung weiterhin zum Updaten der Metadaten entsprechend den Pseudospalten dient, wenn zusätzliche semistrukturierte Daten empfangen werden.

Schutzansprüche

1. Ein Computerprogramm das Anweisungen umfasst, die dann, wenn sie von einem oder mehreren Prozessoren ausgeführt werden, veranlassen, dass der eine oder die mehreren Prozessoren Operationen zum Managen semistrukturierter Daten ausführen, die Folgendes umfassen:
Empfangen semistrukturierter Datenelemente von einer Datenquelle, die über ein Computernetzwerk verbunden ist;
Durchführen einer statistischen Analyse an Sammlungen der semistrukturierten Datenelemente, wenn sie der Datenbank über einen Computerprozessor hinzugefügt werden, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppen der semistrukturierten Datenelemente aufweisen, die extrahiert worden sind;
Identifizieren gemeinsamer Datenelemente von innerhalb der semistrukturierten Daten;
Kombinieren gemeinsamer Datenelemente von der Datenquelle in separate Pseudospalten;
Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte im Computerspeicher; und
Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse.

2. Computerprogramm nach Anspruch 1, wobei die gemeinsamen Datenelemente aus den semistrukturierten Daten extrahiert und separat in einem Spaltenformat gespeichert werden.

3. Das Computerprogramm nach Anspruch 2, wobei die Pseudospalten für die Anwender nicht sichtbar gemacht werden.

4. Das Computerprogramm nach Anspruch 1, wobei die Operationen ferner ein Speichern der nicht gemeinsamen semistrukturierten Daten in einem serialisierten Format in einer Hauptspalte umfassen.

5. Das Computerprogramm nach Anspruch 1, wobei die Operationen ferner ein Filtern mit einem Bloomfilter umfassen, das Identifizierer von Datenelementen umfasst, die innerhalb der separaten Sammlungen enthalten sind.

6. Computerprogramm nach Anspruch 1, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen: minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen.

7. Computerprogramm nach Anspruch 1, wobei die Operationen weiterhin ein Extrahieren von Datenelementen von den serialisierten Überlaufdaten umfassen, wenn das angeforderte Datenelement nicht in einer Pseudospalte ist.

8. Computerprogramm nach Anspruch 1, wobei die Operationen ferner die Rekonstruktion halbstrukturierter Daten in eine Originalform durch Extrahieren von Datenelementen aus Pseudospalten und aus den überlaufserialisierten Daten umfassen.

9. Computerprogramm nach Anspruch 1, wobei die Operationen weiterhin ein Rekonstruieren von semistrukturierten Daten zu einer ursprünglichen Form durch Extrahieren von Datenelementen aus Pseudospalten und aus den serialisierten Überlaufdaten umfassen.

10. System zum Aggregieren semistrukturierter Daten, umfassend:
einen oder mehrere Prozessoren; einen Speicher, der betriebsfähig mit dem einen oder den mehreren Prozessoren verbunden ist; und den Speicher, der ein oder mehrere Module speichert, programmiert zum:
Empfangen von semistrukturierten Datenelementen von einer Datenquelle;
Durchführen einer statistischen Analyse an Sammlungen der semistrukturierten Datenelemente, wenn sie der Datenbank hinzugefügt werden, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert werden, die unterschiedliche Untergruppen der semistrukturierten Datenelemente aufweisen, die extrahiert worden sind;
Identifizieren gemeinsamer Datenelemente aus den semistrukturierten Daten und Kombinieren der gemeinsamen Datenelemente aus der Datenquelle in separate Pseudospalten;
Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte; und
Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse.

11. System nach Anspruch 10, wobei das Kombinieren der gemeinsamen Datenelemente aus der Datenquelle in separate Pseudospalten das Extrahieren gemeinsamer Datenelemente aus den semistrukturierten Daten und das separate Speichern der gemeinsamen Datenelemente in einem Spaltenformat umfasst.

12. Das System nach Anspruch 10, wobei das Spaltenformat für die Anwender unsichtbar ist.

13. System nach Anspruch 10, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen: minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen.

14. Vorrichtung zum Aggregieren semistrukturierter Daten, umfassend:
einen oder mehrere Prozessoren; einen Speicher, der betriebsfähig mit dem einen oder den mehreren Prozessoren verbunden ist; und den Speicher, der speichert:
ein Empfangsmodul, das konfiguriert ist, semistrukturierte Datenelemente von einer Datenquelle zu empfangen;
ein Statistikmodul, das konfiguriert ist, eine statistische Analyse an Sammlungen der semistrukturierten Datenelemente durchzuführen, wenn sie zu der Datenbank hinzugefügt werden, wobei separate Sammlungen, die Teilbereiche der semistrukturierten Daten umfassen, in separaten Dateien gespeichert sind, die unterschiedliche Untergruppen der semistrukturierten Datenelemente aufweisen, die extrahiert worden sind;
eine Aggregationseinrichtung zum Identifizieren gemeinsamer Datenelemente von innerhalb der semistrukturierten Daten und zum Kombinieren der gemeinsamen Datenelemente von der Datenquelle in separate Pseudospalten;
wobei die Aggregationseinrichtung weiterhin zum Serialisieren und Speichern nicht gemeinsamer semistrukturierter Datenelemente in einer serialisierten Überlaufspalte dient; und
wobei die Aggregationseinrichtung weiterhin zum Ableiten von Metadaten entsprechend den Pseudospalten der gemeinsamen Datenelemente aus der statistischen Analyse dient

15. Vorrichtung nach Anspruch 14, wobei das Kombinieren der gemeinsamen Datenelemente aus der Datenquelle in separate Pseudospalten das Extrahieren gemeinsamer Datenelemente aus den semistrukturierten Daten und das separate Speichern der gemeinsamen Datenelemente in einem Spaltenformat umfasst.

16. Vorrichtung nach Anspruch 14, wobei das Spaltenformat für Anwender unsichtbar ist.

17. Vorrichtung nach Anspruch 14, wobei Metadaten für jede Pseudospalte wenigstens eines von Folgendem umfassen:

minimale und maximale Werte innerhalb einer entsprechenden Pseudospalte, eine Zahl, die die Anzahl von Malen darstellt, für welche eindeutige Werte in der Spalte erscheinen.

18. Vorrichtung nach Anspruch 14, wobei die Aggregationseinrichtung weiterhin zum Updaten der Metadaten entsprechend den Pseudospalten dient, wenn zusätzliche semistrukturierte Daten empfangen werden.

Es folgen 6 Seiten Zeichnungen

Anhängende Zeichnungen

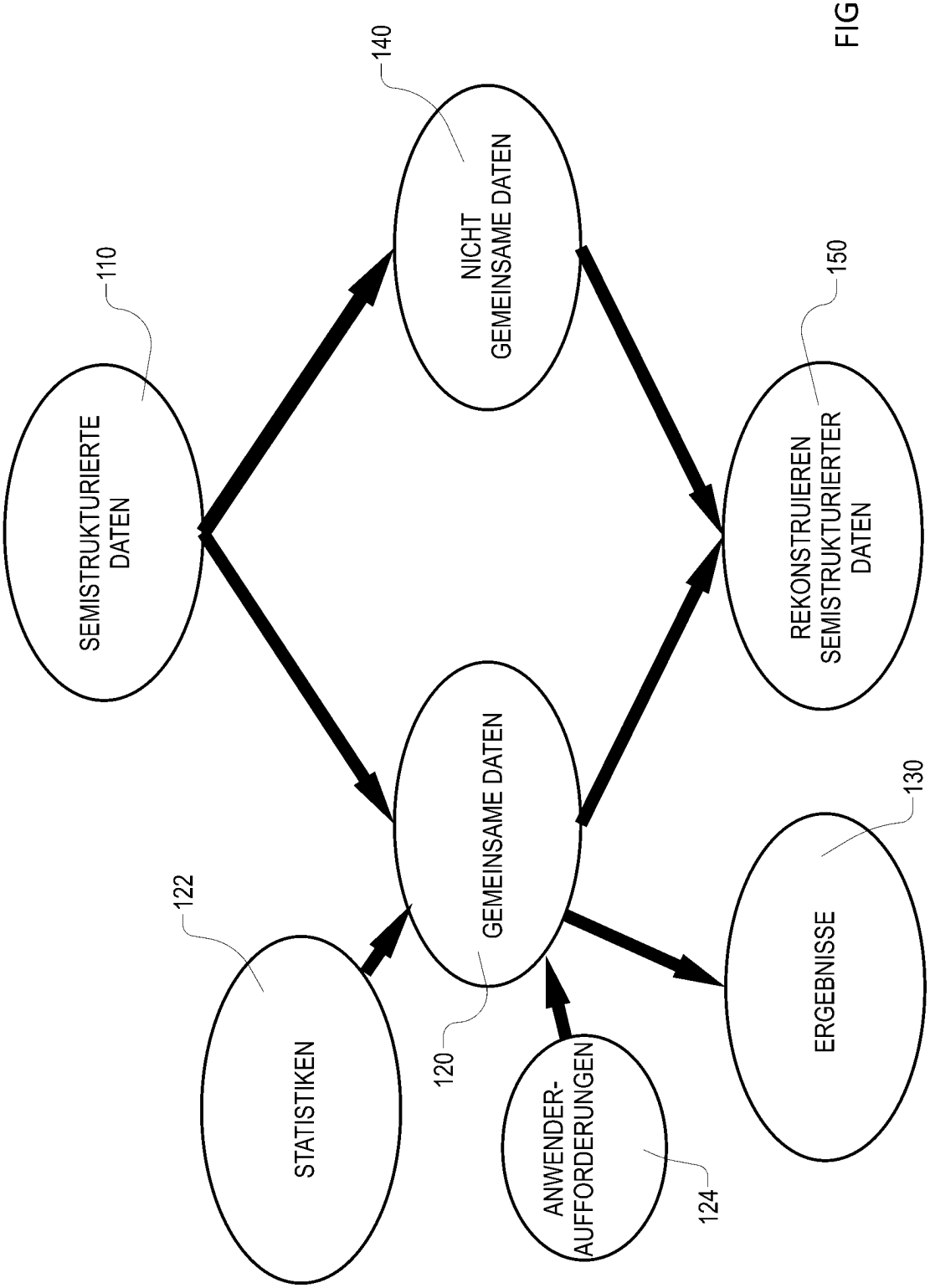


FIG. 1

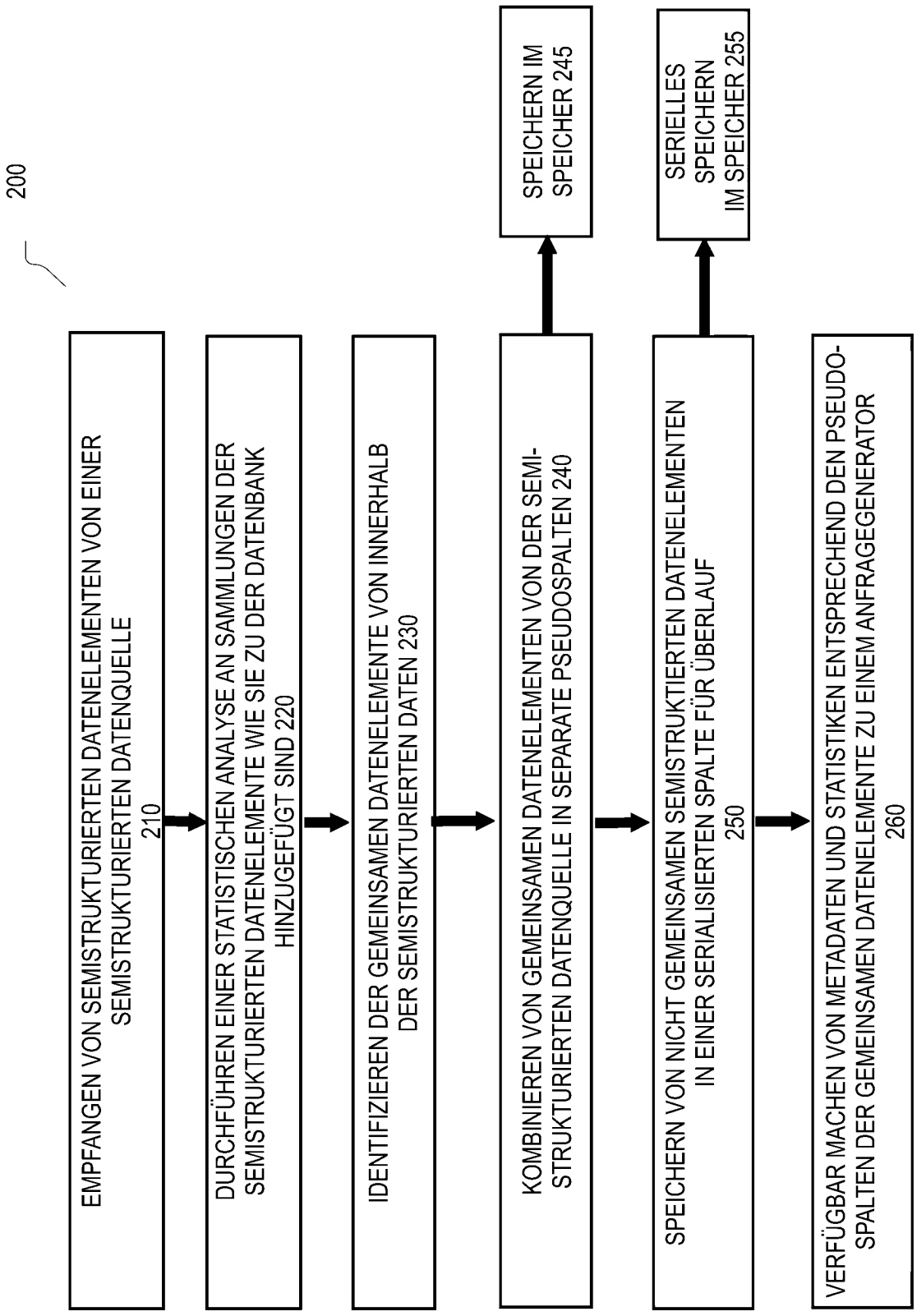

200
EMPFANGEN VON SEMISTRUKTURIERTEN DATENELEMENTEN VON EINER SEMISTRUKTURIERTEN DATENQUELLE 210
DURCHFÜHREN EINER STATISTISCHEN ANALYSE AN SAMMLUNGEN DER SEMISTRUKTURIERTEN DATENELEMENTE WIE SIE ZU DER DATENBANK HINZUGEFÜGT SIND 220
IDENTIFIZIEREN DER GEMEINSAMEN DATENELEMENTE VON INNERHALB DER SEMISTRUKTURIERTEN DATEN 230
KOMBINIEREN VON GEMEINSAMEN DATENELEMENTEN VON DER SEMI-STRUKTURIERTEN DATENQUELLE IN SEPARATE PSEUDOSPALTEN 240
SPEICHERN IM SPEICHER 245
SPEICHERN VON NICHT GEMEINSAMEN SEMISTRUKTIERTEN DATENELEMENTEN IN EINER SERIALISIERTEN SPALTE FÜR ÜBERLAUF 250
SERIELLES SPEICHERN IM SPEICHER 255
VERFÜGBAR MACHEN VON METADATEN UND STATISTIKEN ENTSPRECHEND DEN PSEUDO-SPALTEN DER GEMEINSAMEN DATENELEMENTE ZU EINEM ANFRAGEGENERATOR 260

FIG. 2

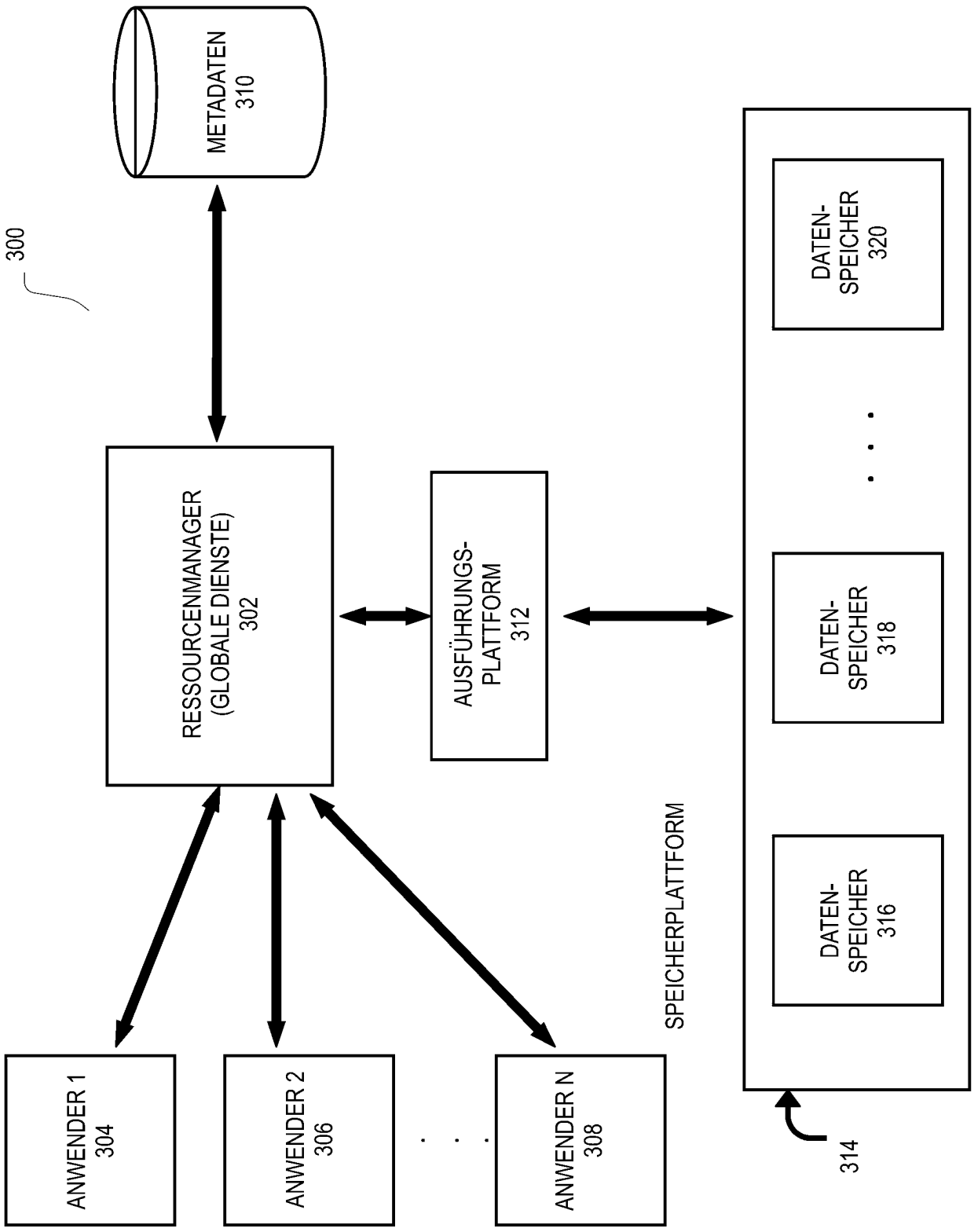

300
METADATEN
310
RESSOURCENMANAGER
(GLOBALE DIENSTE)
302
AUSFÜHRUNGS-
PLATTFORM
312
SPEICHERPLATTFORM
DATEN-
SPEICHER
320
DATEN-
SPEICHER
318
DATEN-
SPEICHER
316
ANWENDER 1
304
ANWENDER 2
306
ANWENDER N
308
314

FIG. 3

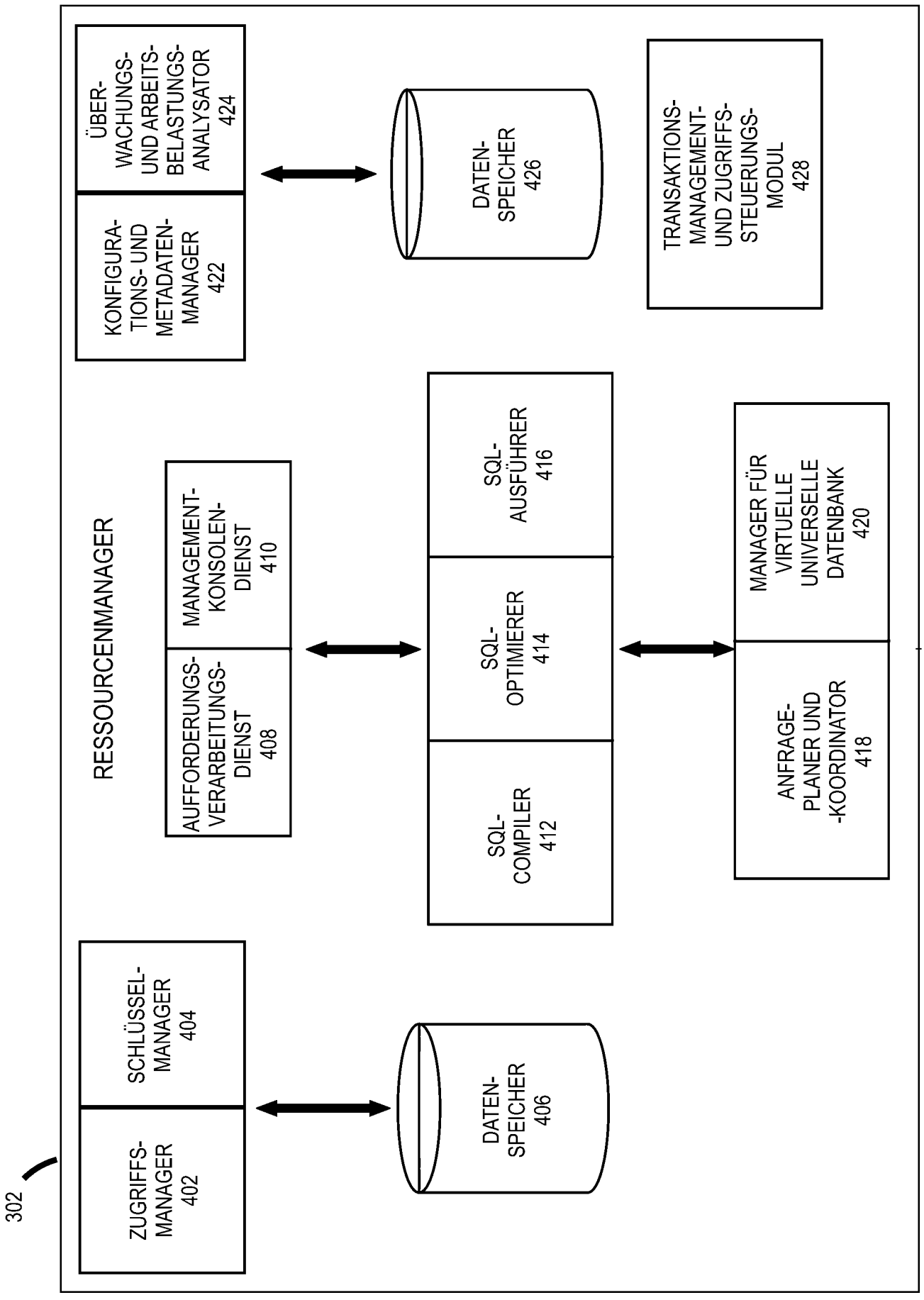

302
ZUGRIFFS-MANAGER 402
SCHLÜSSEL-MANAGER 404
DATEN-SPEICHER 406
RESSOURCENMANAGER
AUFFORDERUNGS-VERARBEITUNGS-DIENST 408
MANAGEMENT-KONSOLEN-DIENST 410
SQL-COMPILER 412
SQL-OPTIMIERER 414
SQL-AUSFÜHRER 416
ANFRAGE-PLANER UND -KOORDINATOR 418
MANAGER FÜR VIRTUELLE UNIVERSELLE DATENBANK 420
KONFIGURATIONS- UND METADATEN-MANAGER 422
ÜBER-WACHUNGS- UND ARBEITS-BELASTUNGS-ANALYSATOR 424
DATEN-SPEICHER 426
TRANSAKTIONS-MANAGEMENT- UND ZUGRIFFS-STEUERUNGS-MODUL 428

FIG. 4

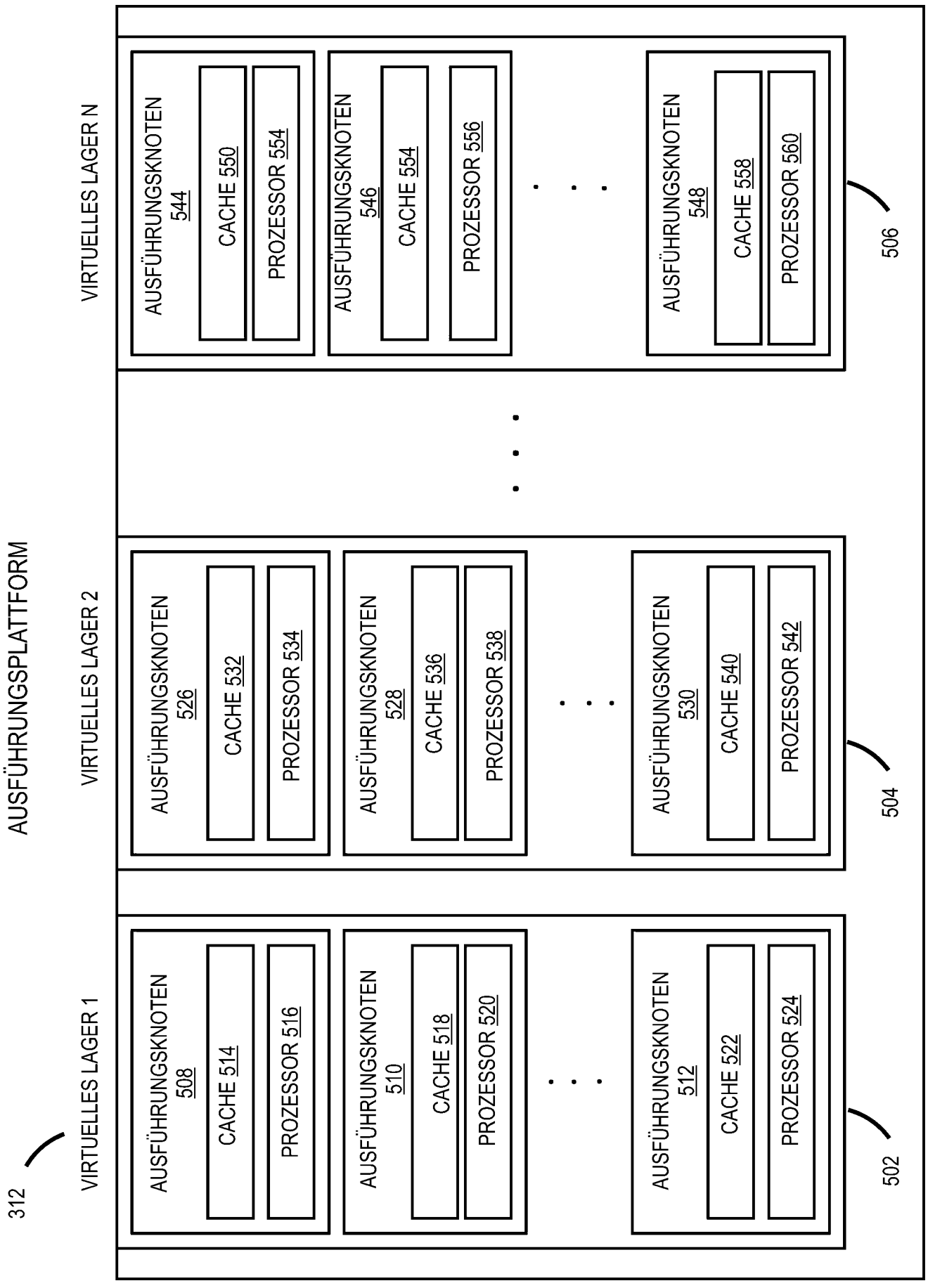

AUSFÜHRUNGSPLATTFORM
312
VIRTUELLES LAGER 1
VIRTUELLES LAGER 2
VIRTUELLES LAGER N
AUSFÜHRUNGSKNOTEN 508
CACHE 514
PROZESSOR 516
AUSFÜHRUNGSKNOTEN 510
CACHE 518
PROZESSOR 520
AUSFÜHRUNGSKNOTEN 512
CACHE 522
PROZESSOR 524
502
AUSFÜHRUNGSKNOTEN 526
CACHE 532
PROZESSOR 534
AUSFÜHRUNGSKNOTEN 528
CACHE 536
PROZESSOR 538
AUSFÜHRUNGSKNOTEN 530
CACHE 540
PROZESSOR 542
504
AUSFÜHRUNGSKNOTEN 544
CACHE 550
PROZESSOR 554
AUSFÜHRUNGSKNOTEN 546
CACHE 554
PROZESSOR 556
AUSFÜHRUNGSKNOTEN 548
CACHE 558
PROZESSOR 560
506

FIG. 5

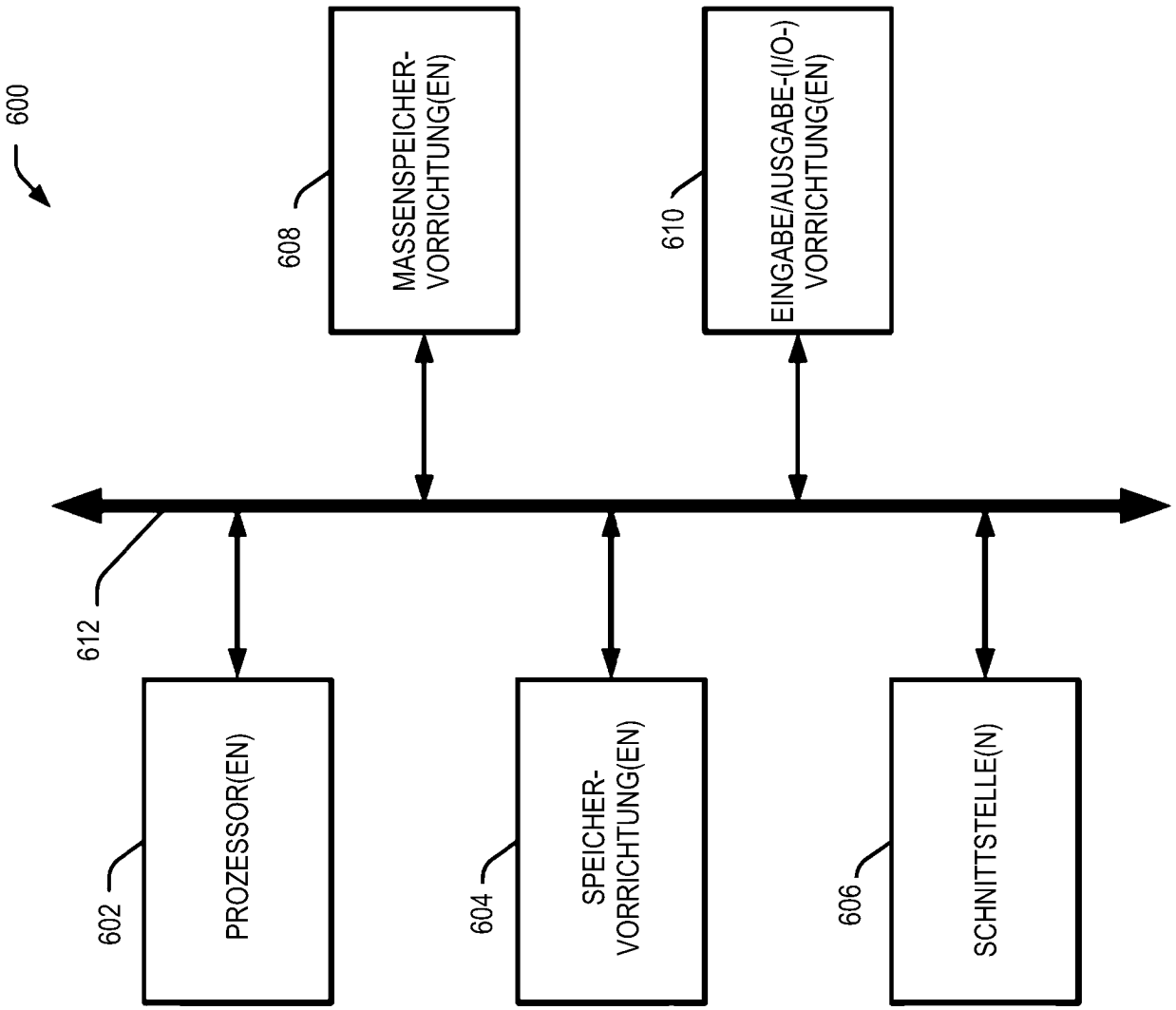


FIG. 6

Das Dokument wurde durch die Firma Jouve erstellt.