(19) Deutsches Patent- und Markenamt

(10) **DE 20 2015 104 126 U1** 2015.12.31

(12) # Gebrauchsmusterschrift

(21) Aktenzeichen: **20 2015 104 126.8**
(22) Anmeldetag: **06.08.2015**
(47) Eintragungstag: **11.11.2015**
(45) Bekanntmachungstag im Patentblatt: **31.12.2015**

(51) Int Cl.: ***G06F 17/30*** (2006.01)
G06F 21/62 (2013.01)

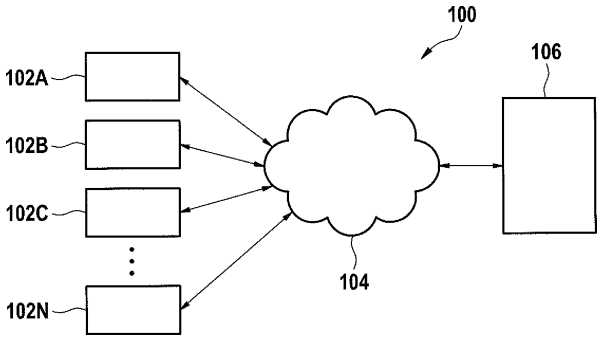
(73) Name und Wohnsitz des Inhabers:
CompuGroup Medical AG, 56070 Koblenz, DE

(74) Name und Wohnsitz des Vertreters:
Richardt Patentanwälte PartG mbB, 65185 Wiesbaden, DE

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Datenverarbeitungssystem**

(57) Hauptanspruch: Computersystem zum Abfragen einer Datenbank, die sich in einem Servercomputer des Computersystems befindet, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Computersystem ferner einen Client-Computer aufweist,
wobei der Client-Computer konfiguriert ist zum Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, an den Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
der Servercomputer konfiguriert ist zum Bestimmen, ob ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist,
wobei im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: der Client-Computer konfiguriert ist zum Traversieren der partiell geordneten Menge zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei zum Traversieren:
der Client-Computer konfiguriert ist zum:
Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer;
Erhalten der angeforderten Teile von dem Servercomputer;
Entschlüsseln der Datenelemente der erhaltenen Teile;
Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Teile;
Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde;
und der Servercomputer zum Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, konfiguriert ist;
der Servercomputer im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, zum Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache konfiguriert ist;
wobei der Servercomputer zum Bereitstellen der angeforderten Datensätze ferner konfiguriert ist zum
Identifizieren aller verschlüsselten Datenelemente, die mit Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und
Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.



Beschreibung

HINTERGRUND DER ERFINDUNG

[0001] Die vorliegende Offenbarung betrifft ein System zur Datenverwaltung.

[0002] Speziell betrifft die Offenbarung ein medizinisches System zur Ermöglichung des Datenaustauschs zwischen einer Vielzahl von Vorrichtungen.

[0003] Die Geheimhaltung von Datensätzen ist für Systeme wie medizinische Systeme, die sensible Patientendaten pflegen, von höchster Bedeutung. Im Stand der Technik erfordern die zunehmenden Mengen derartiger Daten aber Verbesserungen, um ein Gleichgewicht zwischen der Sicherheit von Gesundheitsdaten und der Zugriffsgeschwindigkeit auf die Gesundheitsdaten zu erzielen.

KURZDARSTELLUNG DER ERFINDUNG

[0004] Verschiedene Ausführungsformen sehen ein Client-Server-System, ein Client-System und ein Server-System vor, wie vom Gegenstand der unabhängigen Ansprüche beschrieben. Vorteilhafte Ausführungsformen werden in den abhängigen Ansprüchen bschrieben. Ausführungsformen der vorliegenden Erfindung können frei miteinander kombiniert werden, wenn sie sich nicht gegenseitig ausschließen.

[0005] In einem Aspekt betrifft die Erfindung ein Computersystem zum Abfragen einer Datenbank, die sich in einem Servercomputer des Computersystems befindet, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Computersystem ferner einen Client-Computer aufweist,
wobei der Client-Computer konfiguriert ist zum Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, an den Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
der Servercomputer konfiguriert ist zum Bestimmen, ob ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, wobei im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: der Client-Computer konfiguriert ist zum Traversieren der partiell geordneten Menge zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei zum Traversieren:
der Client-Computer konfiguriert ist zum: Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer; Erhalten der angeforderten Teile von dem Servercomputer; Entschlüsseln der Datenelemente der erhaltenen Teile; Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Teile; Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde;
der Servercomputer zum Traversieren ferner zum Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, konfiguriert ist;
im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, ist der Servercomputer zum Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache konfiguriert;
wobei der Servercomputer zum Bereitstellen der angeforderten Datensätze ferner konfiguriert ist zum Identifizieren aller verschlüsselten Datenelemente, die mit Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

[0006] In einem weiteren Aspekt betrifft die Erfindung einen Servercomputer, wobei der Servercomputer eine Datenbank aufweist, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet,

wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung im der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert werden. Der Servercomputer ist konfiguriert zum: Erhalten einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, von einem Client-Computer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet; für jedes verschlüsselte Datenelement, das die erste und die zweite Intervallgrenze bildet: Bestimmen, ob ein Dateneintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, Empfangen einer Anforderung von dem Client-Computer, die die Dateneinheit der linearen Ordnung anzeigt, die an das jeweilige verschlüsselte Datenelement annotiert ist, das die Intervallgrenze bildet; Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde; im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache; wobei der Servercomputer ferner konfiguriert ist zum Identifizieren aller verschlüsselten Datenelemente, an die die Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

[0007] In einem weiteren Aspekt betrifft die Erfindung einen Client-Computer eines Computersystems nach Anspruch 1, wobei das Computersystem einen Servercomputer aufweist, wobei der Servercomputer eine Datenbank aufweist, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Datenelementen einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind. Der Client-Computer ist konfiguriert zum: Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet; für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet: im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: Traversieren der partiell geordneten Menge zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei für das Traversieren: Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge von dem Servercomputer; Erhalten der angeforderten Teile von dem Servercomputer; Entschlüsseln der Datenelemente der erhaltenen Teile; Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der entschlüsselten Elemente; Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde.

[0008] Ferner wird ein Verfahren zum Abfragen einer Datenbank, die sich in einem Servercomputer befindet, beschrieben, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Verfahren ferner Folgendes aufweist:
Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, von einem Client-Computer zu dem Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
Bestimmen, ob ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, wobei im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die

Intervallgrenze bildet, assoziiert ist: Traversieren der partiell geordneten Menge durch den Client-Computer zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei das Traversieren Folgendes aufweist:
Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer;
Erhalten der angeforderten Teile von dem Servercomputer;
Entschlüsseln der Datenelemente der erhaltenen Teile;
Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Elemente;
Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde;
Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde;
im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache aus dem Servercomputer;
Identifizieren aller verschlüsselten Datenelemente, die mit den Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden;
Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

KURZE BESCHREIBUNG DER MEHREREN ANSICHTEN DER ZEICHNUNGEN

[0009] Im Folgenden werden Ausführungsformen der Erfindung nur beispielhaft unter Bezugnahme auf die Zeichnungen ausführlicher beschrieben, wobei:

[0010] Fig. 1 ein schematisches Blockdiagramm eines Client-Serversystems zeigt,

[0011] Fig. 2 ein schematisches Blockdiagramm eines Serversystems zeigt,

[0012] Fig. 3 ein schematisches Blockdiagramm eines Client-Systems zeigt,

[0013] Fig. 4 ein Flussdiagramm eines Verfahrens zum Abfragen einer Datenbank ist,

[0014] Fig. 5 ein Flussdiagramm eines Verfahrens zum Beschränken des Zugriffs auf gecachte Teile eines Indexbaums an dem Client-System ist,

[0015] Fig. 6 ein Flussdiagramm eines Verfahrens zum Löschen veralteter gecachter Teile des Indexbaums ist,

[0016] Fig. 7 ein Flussdiagramm eines beispielhaften Verfahrens zum Traversieren gecachter Teile des Indesbaums ist,

[0017] Fig. 8 ein Flussdiagramm eines weiteren beispielhaften Verfahrens zum Traversieren gecachter Teile des Indexbaums ist.

AUSFÜHRLICHE BESCHREIBUNG DER ERFINDUNG

[0018] Die Beschreibungen der verschiedenen Ausführungsformen der vorliegenden Erfindung werden zum Zweck der Veranschaulichung vorgelegt, es ist aber nicht vorgesehen, dass sie umfassend sind oder auf die offenbarten Ausführungsformen beschränkt sind. Für den Durchschnittsfachmann sind mehrere Modifikationen und Variationen offensichtlich, ohne vom Umfang und Sinn der beschriebenen Ausführungsformen abzuweichen. Die hierin verwendete Terminologie wurde gewählt, um die Grundlagen der Ausführungsformen, die praktische Anwendung oder technische Verbesserung gegenüber im Markt angetroffenen Technologien am besten zu erläutern oder um andere Durchschnittsfachleute zu befähigen, die hierin offenbarten Ausführungsformen zu verstehen.

[0019] Der Begriff Relation betrifft eine Datenstruktur, die einer Datentabelle, wie etwa einer Datentabelle eines Datenbanksystems, äquivalent ist.

[0020] Eine partiell geordnete Menge versteht sich als eine Menge Dateneinheiten, die das Konzept eines Ordnens, Sequenzierens oder Anordnens der Elemente einer Menge formalisiert. Eine partiell geordnete Menge besteht aus einer Menge zusammen mit einer binären Relation, die anzeigt, dass bei gewissen Elementepaaren in der Menge eines der Elemente dem anderen vorangeht. Eine derartige Relation wird eine partielle Ordnung genannt, um die Tatsache zu reflektieren, dass nicht jedes Elementepaar miteinander in Beziehung stehen muss: bei einigen Paare kann es sein, dass keines der Elemente dem anderen in der partiell geordneten Menge direkt vorangeht. Die Datenelemente bilden die partiell geordnete Menge in der Relation über einen gewurzelten Baum. Das heißt, die partiell geordnete Menge kann eine Datenstruktur betreffen, bei der Datenelemente in der Relation miteinander in Beziehung stehen. Die Datenstruktur kann eine Baumstruktur wie einen gewurzelten Baum oder eine ähnliche indexierende Datenstruktur mit Zeigern oder Referenzen zu den Da-

tensätzen aufweisen, die die entsprechenden Werte enthalten, die mit den Datenelementen assoziiert sind. Zum Beispiel kann die partiell geordnete Menge einen AVL-Baum oder einen binären Suchbaum umfassen. Der Baum hält die Datenelemente in sortierter Ordnung, da dies den Baumdurchlauf beschleunigen kann.

[0021] Das erste Verschlüsselungsverfahren kann ein beliebiges Verschlüsselungsverfahren umfassen. Zum Beispiel kann das erste Verschlüsselungsverfahren ein nichtdeterministisches Verschlüsselungsverfahren, ein homomorphes Verschlüsselungsverfahren oder ein ordnungserhaltendes Verschlüsselungsverfahren umfassen. Das erste Verschlüsselungsverfahren verschlüsselt bei Ausführung für ein vorgegebenes Datenelement das Datenelement unter Verwendung von beispielsweise einem kryptografischen Schlüssel. Gemäß einer Ausführungsform ist der kryptografische Schlüssel ein symmetrischer oder ein asymmetrischer Schlüssel. Ein zweites Verschlüsselungsverfahren, das das gleiche wie oder ein anderes als das erste Verschlüsselungsverfahren ist, kann zum Verschlüsseln der Datensätze verwendet werden, die die entsprechenden Werte enthalten, die den Datenelementen der partiell geordneten Menge zugeordnet sind.

[0022] Ein „kryptografischer Schlüssel", wie hierin verwendet, umschließt einen symmetrischen Schlüssel, der sowohl zur Verschlüsselung und Entschlüsselung als auch als ein asymmetrisches kryptografisches Schlüsselpaar dient, wobei der öffentliche Schlüssel zur Verschlüsselung verwendet wird und der private Schlüssel zur Entschlüsselung verwendet wird.

[0023] Der Begriff „Datenelement", wie hierin verwendet, bezieht sich auf einen Datenwert einer Dateneinheit eines Dateneinheitensatzes, wie z. B. ein Tupel, der eine oder mehr Dateneinheiten aufweist, wie etwa ein Datenwert eines Datenfelds eines Datensatzes, der mehrere Datenfelder hat, in einer Datenbank, wie etwa einer relationalen Datenbank, objektorientierten Datenbanken, Objekt-relationalen Datenbanken, hierarchischen Datenbanken, noSQL-Datenbanken oder einer In-Memory-Datenbank. Zum Beispiel können die medizinischen Aufzeichnungen eines Patienten verschiedene Dateneinheiten, z. B. Datenfelder, wie etwa Namens-, Adress-, Telefonnummer- und medizinische Datenfelder dieses Patienten, aufweisen, wobei Datenwerte für Namen, Adresse und Telefonnummer beispielhafte Datenelemente sind. Ein Datenelement kann eine Zeichenfolge aufweisen, die eine Vielzahl von Buchstaben, Zahlen oder beiden aufweisen kann. Das Datenelement kann eine Zahl mit einem ganzzahligen Wert oder Gleitkommazahlenwert usw. aufweisen.

[0024] Die verschlüsselten Datenelemente, die die erste und die zweite Intervallgrenze bilden, können Teil der Relation sein oder nicht.

[0025] Die obigen Merkmale können den Vorteil haben, dass sie sowohl schnellen als auch sicheren Zugriff auf im Serversystem gespeicherte Daten maximieren. Das vorliegende Verfahren und System können Verarbeitungszeit und Ressourcen sparen, die ansonsten bei der Verarbeitung wiederholt abgefragter Suchintervalle erforderlich wären. Diese Ausführungsform kann für verteilte Systeme besonders vorteilhaft sein, wie etwa die Cloud, wo täglich eine sehr hohe Anzahl von Abfragen und Suchintervallen verarbeitet werden, so dass die Wahrscheinlichkeit, dass die gleichen Suchintervalle angefordert werden, zunimmt.

[0026] Das vorliegende Verfahren und System können einen sicheren Zugriff auf die Daten über die partiell geordnete Menge ermöglichen. Dies kann für sensible Daten eines Gesundheitssystems besonders wichtig sein. Die Manipulation der Datenelemente der partiell geordneten Menge auf der Serverseite kann durchgeführt werden, ohne die mehreren Datenelemente am Servercomputer entschlüsseln zu müssen, d. h. der Servercomputer kann keinen Zugriff auf den kryptografischen Schlüssel oder den unverschlüsselten Inhalt mehrerer Datenelemente haben. Das Risiko einer Beeinträchtigung der Datenelemente am Servercomputer kann daher abgemildert werden.

[0027] Das vorliegende Verfahren und System können ferner den Vorteil haben, dass sie einen (Fern-) Steuerungszugriff auf Daten über Client-Systeme ermöglichen. Dies kann für Daten mit medizinischem Bezug, für welche ein zentralisierter Steuerungszugriff auf Daten möglicherweise nicht geeignet ist, besonders vorteilhaft sein. Zum Beispiel kann die Vertraulichkeit von medizinischen Daten (z. B. Datenelemente und zugeordnete Datensätze) beeinträchtigt werden, wenn ein unberechtigter Benutzer Zugriff auf unverschlüsselten Inhalt der medizinischen Daten hat.

[0028] Gemäß einer Ausführungsform ist das Datenelement, das die Intervallgrenze bildet, das Datenelement der Relation, das der Intervallgrenze am nächsten ist, wobei das Datenelement innerhalb des Suchintervalls ist. Diese Ausführungsform kann den Vorteil haben, dass sie genaue Bereichssuchergebnisse liefert.

[0029] Gemäß einer Ausführungsform wird ferner offenbart, dass ein neues Datenelement in dem gewurzelten Baum am Servercomputer eingefügt wird; und nach Einfügung werden die gecachten Einträge vom Servercomputer gelöscht. Die vorhergehenden Bereichssuchergebnisse können unter Verwendung

der Cache-Einträge, die vor der Änderung des Indexbaums, z. B. durch Einfügen des neuen Datenelements, generiert werden, nicht reproduziert werden.

[0030] Gemäß einer Ausführungsform bilden die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei der eine oder die mehreren Teile Teilbäume des gewurzelten Baums umfassen, wobei die Teilbäume alle eine gemeinsame vordefinierte Höhe haben, wobei die Höhe die Anzahl von Kanten am längsten Abwärtsweg zwischen dem Wurzelknoten des Teilbaums und der Blattebene des Teilbaums festsetzt. Diese Ausführungsform kann den Vorteil haben, dass sie einen beschränkten Zugriff auf den gewurzelten Baum durch den Client-Computer bereitstellt. Der Client-Computer kann zum Beispiel die erforderliche Höhe der vom Serversystem zu erhaltenden Teile definieren. Zum Beispiel kann das Client-System im Fall, dass das Client-System Zugang zu einer begrenzten Netzbandbreite hat, die erforderliche Höhe der Teilbäume entsprechend anpassen. In einem anderen Beispiel können die Teilbäume verschiedene vordefinierte Höhen haben.

[0031] Gemäß einer Ausführungsform werden die erhaltenen Teile der partiell geordneten Menge durch den Client-Computer gecacht. Diese Ausführungsform kann den Vorteil haben, dass sie den Zugriff auf die Datenelemente weiter beschleunigt. Ferner kann dies Netzbandbreite sparen, die ansonsten zum erneuten Senden der gecachten Teile der partiell geordneten Menge benötigt würde.

[0032] Gemäß einer Ausführungsform bilden die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei ferner offenbart wird, dass: ein neues verschlüsseltes Datenelement am Servercomputer in den gewurzelten Baum eingefügt wird; im Fall, dass an einem vorgegebenen Knoten eine Ungleichmäßigkeit festgestellt wird, der gewurzelte Baum an einem von dem vorgegebenen Knoten definierten Rotationspunkt neu ausgeglichen wird; gecachte Teile, die an einem Abwärtsweg entlang, angefangen am Wurzelknoten eines gecachten Teils der gecachten Teile, der den vorgegebenen Knoten enthält, bis zu den Blättern des gewurzelten Baums, Knoten haben, vom Client-Computer gelöscht werden. Diese Ausführungsform kann die Nutzung veralteter Teile am Client-System verhindern, die falsche Suchergebnisse von Bereichssuchen ergeben können.

[0033] Gemäß einer Ausführungsform weist das Löschen der gecachten Teile Folgendes auf: erneutes Traversieren des gewurzelten Baums, angefangen mit dem neuen eingefügten verschlüsselten Datenelement aufwärts zum Wurzelknoten des gewurzelten Baums, wobei das erneute Traversieren einen Durchlaufweg zur Folge hat; dem Client-Computer Mitteilen des Durchlaufwegs, wobei der Durchlaufweg den Rotationspunkt und die traversierten Knoten anzeigt; Verwenden des Durchlaufwegs zum Traversieren der gecachten Teile durch den Client-Computer zum Identifizieren eines gecachten Teils, der den Rotationspunkt enthält, in den gecachten Teilen; und Identifizieren jedweder Knoten an einem Abwärtsweg entlang, angefangen am Wurzelknoten des identifizierten gecachten Teils bis zu den Blättern des gewurzelten Baums; Löschen des identifizierten gecachten Teils und der identifizierten Knoten. Diese Ausführungsform kann ein genaues und systematisches Verfahren zum Aktualisieren des Cache-Inhalts am Client-Computer bereitstellen. Dies kann für große Indexe (z. B. für Massendaten) besonders vorteilhaft sein, da das Löschen gemäß dieser Ausführungsform systematisch durchgeführt werden kann.

[0034] Gemäß einer Ausführungsform weist das Traversieren das Erstellen einer Bitmaske durch den Servercomputer auf, wobei die Bitmaske Bits aufweist, die mit jeweiligen Knoten des Durchlaufwegs in der Ordnung, in der die Knoten im Durchlaufweg traversiert werden, assoziiert sind, wobei jedes Bit der Bitmaske für einen vorgegebenen Knoten einen Wert hat, der anzeigt, ob der nächste Knoten im Durchlaufweg dem vorgegebenen Knoten mit Bezug auf die von der partiellen Ordnung repräsentierten Ordnung vorangeht oder nachfolgt; Senden der Bitmaske an den Client-Computer; Verwenden der Bitmaske durch den Client-Computer, um die gecachten Teile zu traversieren. Dies kann ein genaues Verfahren zum Traversieren der Teile bereitstellen.

[0035] Gemäß einer Ausführungsform weist der Durchlaufweg eine oder mehr Kanten auf, wobei jede Kante von einem Paar von zwei aufeinanderfolgend angeordneten Knoten definiert wird, wobei die zwei aufeinanderfolgend angeordneten Knoten einen Quellknoten und einen Zielknoten des Durchlaufwegs aufweisen, wobei das Traversieren das Erstellen einer Bitmaske durch den Servercomputer aufweist, wobei die Bitmaske Bits aufweist, die mit Kanten des Durchlaufwegs in der Ordnung, in der die Kanten in dem Durchlaufweg traversiert werden, assoziiert sind, wobei jedes Bit der Bitmaske für eine vorgegebene Kante einen Wert hat, der anzeigt, ob der Zielknoten der Kante dem Quellknoten der Kante mit Bezug auf die von der partiellen Ordnung repräsentierte Ordnung vorangeht oder nachfolgt; Senden der Bitmaske an den Client-Computer; Verwenden der Bitmaske durch den Client-Computer, um die gecachten Teile zu traversieren.

[0036] Gemäß einer Ausführungsform wird ferner offenbart, dass: das verschlüsselte Datenelement, das

dem Rotationspunkt entspricht, dem Client-Computer mitgeteilt wird; das mitgeteilte Datenelement vom Client-Computer entschlüsselt wird; das entschlüsselte Datenelement im Cache als der den Rotationspunkt bildende vorgegebene Knoten identifiziert wird.

[0037] Gemäß einer Ausführungsform wird ferner offenbart, dass: bestimmt wird, ob ein Neuausgleichen des Baums unter Verwendung des erneuten Durchlaufs erforderlich ist. Dies kann durch Ausnutzen des erneuten Durchlaufs, um die Ausgeglichenheit des Baums zu prüfen, Verarbeitungsressourcen sparen. Dies kann einen zusätzlichen Durchlauf für das Neuausgleichen vermeiden.

[0038] Gemäß einer Ausführungsform umfasst die partiell geordnete Menge einen AVL-Baum. Diese Ausführungsform kann nahtlos in bestehende Systeme integriert werden.

[0039] Gemäß einer Ausführungsform zeigt der generierte Cache-Eintrag ferner an, dass die Intervallgrenze eine untere oder eine obere Intervallgrenze ist.

[0040] Gemäß einer Ausführungsform wird ferner offenbart, dass: eine Suchanforderung am Client erhalten wird, wobei die Suchanforderung eine Anforderung für eine Präfixsuche nach einem Suchkriterium aufweist, wobei der Client funktionell ist, um das Suchintervall durch Umwandeln der Präfixsuche in ein entsprechendes Intervall zu bestimmen, das das Suchkriterium als die untere Intervallgrenze und einen rechnerisch von dem Suchkriterium abgeleiteten Wert als die obere Intervallgrenze aufweist.

[0041] Fig. 1 zeigt ein schematisches Blockdiagramm eines Client-Serversystems **100** gemäß der vorliegenden Offenbarung. Das Client-Serversystem **100** weist ein oder mehr Client-Systeme (oder Client-Computer) **102A**–N und ein Serversystem (oder einen Servercomputer) **106** auf. Das Client-System **102A**–N und das Serversystem **106** sind über eine Netzverbindung **104** miteinander verbunden. Das Client-System **102** kann als ein Computersystem betrachtet werden, das über die Netzverbindung **104**, die zum Beispiel eine drahtlose Nahverkehrsnetz-(WLAN)-verbindung, eine WAN-(Weitverkehrsnetz)-Verbindung oder eine Kombination davon umfassen kann, mit dem Serversystem **106** kommunizieren kann. Das Serversystem **106** kann als ein Computersystem betrachtet werden, das Datenzugang zu dem Client-System **102** bereitstellen kann. Um auf im Server **106** gespeicherte Daten zuzugreifen, sendet das Client-System **102** eine Anforderung an den Server **106**, wobei die Anforderung im Serversystem **106** aufgenommen und verarbeitet wird.

[0042] Das Client-Serversystem **100** kann zum Beispiel ein medizinisches System sein, bei dem das Client-System **102** Teil einer Arztpraxis oder eines Krankenhauses sein kann, und das Serversystem **106** kann sich in einem externen Datenbankzentrum wie etwa einem Cloud-Rechenzentrum befinden.

[0043] Zum Beispiel kann ein Client-System **102** eine tragbare elektronische Telekommunikationsvorrichtung aufweisen, wie etwa ein Mobilfunkgerät oder ein digitales Mobilfunktelefon, wie etwa ein Smartphone oder ein Tablet. Dies kann für Anwendungen, an denen terrestrische Benutzer in Fahrzeugen oder zu Fuß beteiligt sind, wie etwa Ersthelferorganisationen oder Organisationen für öffentliche Bauarbeiten, besonders nützlich sein. Auf dem Gebiet der Bereitstellung von Gesundheitsfürsorgediensten kann dies besonders nützlich sein, da es die Verwendung einer batteriebetriebenen mobilen tragbaren Client-Vorrichtung im Kontext medizinischer Notdienste und/oder Hausbesuche bei Patienten durch Ärzte unter Erhaltung hoher Datenvertraulichkeitsmaßstäbe ermöglichen kann.

[0044] Der Begriff „Server", wie hierin verwendet, betrifft ein(e) computerisierte(s) Komponente (z. B. eine Softwarekomponente), System oder Entität, ungeachtet der Form, die/das zum Geben von Daten, Dateien, Anwendungen, Inhalt oder anderer Dienste an eine oder mehr andere Vorrichtungen oder Entitäten ausgeführt ist.

[0045] Fig. 2 zeigt ein schematisches Blockdiagramm eines Client-Systems **102** gemäß der vorliegenden Offenbarung. Zu den Komponenten des Client-Systems **102** können unter anderem ein oder mehr Prozessoren oder Verarbeitungseinheiten **203**, ein Speichersystem **211**, eine Speichereinheit **205** und ein Bus **207**, der verschiedene Systemkomponenten einschließlich der Speichereinheit **205** mit dem Prozessor **203** koppelt, zählen. Das Speichersystem **211** kann z. B. ein Festplattenlaufwerk (HDD) beinhalten. Die Speichereinheit **205** kann computersystemlesbare Medien in der Form von flüchtigem Speicher, wie etwa Direktzugriffsspeicher (RAM) und/oder Cache-Speicher, beinhalten. Das Client-System **102** beinhaltet gewöhnlich verschiedene computersystemlesbare Medien. Derartige Medien können jedwede Medien sein, auf die das Client-System **102** zugreifen kann, und zu ihnen zählen flüchtige und nichtflüchtige Medien, entfernbare und nichtentfernbare Medien.

[0046] Das Client-System **102** kann auch mit einer oder mehr externen Vorrichtungen wie einer Tastatur, einem Zeigegerät, einer Anzeige **213** usw.; einer oder mehr Vorrichtungen, die einem Benutzer die Interaktion mit dem Client-System **102** ermöglichen; und/oder jedweden Vorrichtungen (z. B. Netzwerkkarte, Modem usw.), die dem Client-System **102** die Kommunikation mit einer oder mehr anderen Rechenvorrichtungen ermöglichen, kommunizieren. Eine sol-

che Kommunikation kann über die E/A-Schnittstelle (n) **219** stattfinden. Und das Client-System **102** kann über den Netzwerkadapter **209** auch noch mit einem oder mehr Netzen, wie etwa einem lokalen Netz (LAN), einem allgemeinen Weitverkehrsnetz (WAN) und/oder einem öffentlichen Netz (z. B. dem Internet), kommunizieren. Wie abgebildet, kommuniziert der Netzadapter **209** über den Bus **207** mit den anderen Komponenten des Client-Systems **102**.

[0047] Die Speichereinheit **205** ist zum Speichern von Anwendungen konfiguriert, die im Prozessor **203** ausführbar sind. Zum Beispiel kann das Speichersystem **205** ein Betriebssystem sowie Anwendungsprogramme aufweisen. Die Anwendungsprogramme weisen eine Datenzugriffsanwendung **208** auf. Die Datenzugriffsanwendung **208** weist Anweisungen auf, die, wenn sie ausgeführt werden, es einem Benutzer des Client-Systems **102** ermöglichen, im Serversystem **106** befindliche Daten anzufordern. Zum Beispiel kann die Ausführung der Anweisungen den Prozessor **203** zum Anzeigen einer grafischen Benutzeroberfläche **220** veranlassen. Die grafische Benutzeroberfläche **220** weist Suchfelder **226** auf, die zur Aufnahme von Eingaben konfiguriert sind, die z. B. eine Bereichs- oder Intervallsuche erkennen lassen. Die Intervallsuche kann beispielsweise eine Präfixsuche nach Zeichenketten repräsentieren, wie etwa „Ped*". Zum Beispiel kann der Benutzer eine Präfixsuche Ped* in ein Suchfeld **226** der grafischen Benutzeroberfläche **220** eingeben. Die Präfixsuche kann durch die Datenzugriffsanwendung **208** in eine intervallsuche, z. B. ["Ped"; "Pee"[, umgewandelt werden, die die angeforderten Daten abdecken würde, die Benutzern mit einem Namen, der mit „Ped" anfängt, zugeordnet sind.

[0048] Die Speichereinheit **205** ist ferner zum Speichern eines kryptografischen Schlüssels **222** konfiguriert. Der kryptografische Schlüssel **222** kann zum Verschlüsseln von Datenelementen verwendet werden, auf die das Client-System **102** zugreifen kann, z. B. kann es, da das Client-System **102** und das Serversystem **106** zur Handhabung medizinischer Aufzeichnungen verwendet werden können, entscheidend sein, dass alle Informationen, die das Client-System **102** und somit die Umgebung einer Arztpraxis verlassen, am Zielort, z. B. Serversystem **106**, geheim gehalten werden. Es muss sichergestellt werden, dass kein unberechtigter Benutzer am Serversystem **106** Zugriff auf Patientendatensätze haben kann.

[0049] Das Client-System **102** kann ferner einen Cache **224** aufweisen. Der Begriff „Cache", wie hierin verwendet, bezieht sich auf einen Zwischenspeicherbereich, der ein Schnellzugriffsbereich ist und entweder ein Hintergrundspeicher oder ein Disk-Cache sein kann. Der Cache kann zum Beispiel ein Speicherteil eines schnellen statischen RAM (SRAM) sein oder Teil des Hauptspeichers sein, z. B. aus dynamischem RAM (DRAM) bestehend.

[0050] Fig. 3 zeigt ein schematisches Blockdiagramm eines Serversystems **106** gemäß der vorliegenden Offenbarung.

[0051] Zu den Komponenten des Serversystems **106** können unter anderem ein oder mehr Prozessoren oder Verarbeitungseinheiten **303**, ein Speichersystem **311**, eine Speichereinheit **305** und ein Bus **307**, der verschiedene Systemkomponenten einschließlich Speichereinheit **305** mit dem Prozessor **303** koppelt, zählen. Das Speichersystem **311** kann z. B. ein Festplattenlaufwerk (HDD) beinhalten. Die Speichereinheit **305** kann ein computersystemlesbares Medium in der Form von flüchtigem Speicher, wie etwa Direkzugriffsspeicher (RAM) und/oder Cache-Speicher, beinhalten. Zum Serversystem **106** zählen gewöhnlich verschiedene computersystemlesbare Medien. Derartige Medien können jedwede Medien sein, auf die das Serversystem **106** zugreifen kann, und zu ihnen zählen flüchtige und nichtflüchtige Medien, entfernbare und nichtentfernbare Medien.

[0052] Das Serversystem **106** kann auch mit einer oder mehr externen Vorrichtungen wie einer Tastatur, einem Zeigegerät, einer Anzeige **313** usw.; einer oder mehr Vorrichtungen, die einem Benutzer die Interaktion mit dem Serversystem **106** ermöglichen; und/oder jedweden Vorrichtungen (z. B. Netzwerkkarte, Modem usw.), die dem Serversystem **106** die Kommunikation mit einer oder mehr anderen Rechenvorrichtungen ermöglichen, kommunizieren. Eine solche Kommunikation kann über die E/A-Schnittstelle(n) **319** stattfinden. Das Serversystem **106** kann über einen Netzwerkadapter **309** auch noch mit einem oder mehr Netzen, wie etwa einem lokalen Netz (LAN), einem allgemeinen Weitverkehrsnetz (WAN) und/oder einem öffentlichen Netz (z. B. dem Internet) kommunizieren. Wie abgebildet, kommuniziert der Netzadapter **309** über den Bus **307** mit den anderen Komponenten des Serversystems **102**.

[0053] Die Speichereinheit **305** ist zum Speichern von Anwendungen konfiguriert, die im Prozessor **303** ausführbar sind. Zum Beispiel kann das Speichersystem **305** ein Betriebssystem sowie Anwendungsprogramme aufweisen. Die Anwendungsprogramme weisen ein Datenbankmanagementsystem (DBMS) **308** auf, das Speichern, Modifizieren und Entnehmen von Daten aus dem Datenbankspeicher **320** des Serversystems **106** ermöglicht. Anforderungen für Informationen aus dem Datenbankspeicher **320** werden zum Beispiel in der Form von Abfragen getätigt. Die Abfrage kann zum Beispiel eine SQL-Abfrage sein. Der Datenbankspeicher **320** kann eine Datenbank **322** aufweisen, die Daten, z. B. Patientendaten, aufweist. Der Inhalt der Datenbank **322** kann z. B. unter Verwendung des kryptografischen Schlüs-

sels **222** am Client-System **102** verschlüsselt werden. Der Datenzugriff (z. B. Datenabruf) auf die Datenbank **322** kann unter Verwendung eines gewurzelten Baums **324** durchgeführt werden. Zum Beispiel kann der gewurzelte Baum **324** Patientennamen aufweisen, die zum Zugreifen auf Daten in der Datenbank **322**, die zu jedem der Namen gehören, verwendet werden können. Der gewurzelte Baum **324** kann beispielsweise ein Index sein, wobei er dadurch hierin als Indexbaum **324** bezeichnet wird. Der Indexbaum **324** kann mehrere Datenelemente aufweisen, die von Knoten **326** repräsentiert werden. Der Einfachheit der Beschreibung halber werden Datenelemente und Knoten hierin austauschbar verwendet. Der Indexbaum **324** weist Datenelemente auf, die an Computersystemen **102** verschlüsselt werden. Die Verschlüsselung der Datenelemente des Indexbaums **324** kann unter Verwendung eines kryptografischen Schlüssels **222** durchgeführt werden, der in dem Client-System **102** gespeichert wird, wo die Verschlüsselung durchgeführt wird. Dies kann das Serversystem **106** am Zugriff auf den verschlüsselten Inhalt der Datenelemente **326** des Indexbaums **324** hindern.

[0054] Die Datenelemente **326** des Indexbaums **324** können in numerischer oder lexikografischer Ordnung oder einer Kombination davon sortiert werden. Das Sortieren kann unter Verwendung des unverschlüsselten Inhalts der Datenelemente des Indexbaums **324** durchgeführt werden. Dies kann am Client-System **102** durchgeführt werden, da das Client-System **102** berechtigten Zugriff auf den unverschlüsselten Inhalt der Datenelemente des Indexbaums **324** haben kann. In einem weiteren Beispiel kann das Serversystem **106** das Sortieren der Datenelemente **326** des Indexbaums **324** auf Basis ihres verschlüsselten Inhalts durchführen, wobei die Datenelemente unter Verwendung eines ordnungserhaltenden Verschlüsselungsverfahrens verschlüsselt werden. Dies ist besonders vorteilhaft, weil das Serversystem **106** möglicherweise keinen berechtigten Zugriff auf den unverschlüsselten Inhalt der Datenelemente **326** hat.

[0055] Zum Beispiel kann jedes Datenelement **326** des Indexbaums **324** einen Patientennamen in dem verschlüsselten Format und eine Referenz (oder referentielle Verbindung) auf mit diesem Patienten in Beziehung stehenden Daten in der Datenbank **322** aufweisen. Die Referenzen teilen jedes Datenelement **326** entsprechenden in der Datenbank **322** gespeicherten Daten c_data zu. Die Daten c_data können unter Verwendung desselben oder eines anderen Verschlüsselungsverfahren verschlüsselt werden, das zum Verschlüsseln der Datenelemente **326** verwendet wird. Der Einfachheit der Beschreibung halber werden die Datenelemente **326** in Klartext gezeigt, obwohl die Datenelemente **326** verschlüsselt sind und das Serversystem **106** keinen Zugriff auf den unverschlüsselten (d. h. Klartext-)Inhalt der Datenelemente **326** hat.

[0056] Zum Beispiel ist der Indexbaum **324** gemäß der lexikografischen Ordnung der Patientennamen in der unverschlüsselten Form sortiert. Der Indexbaum **324** kann ein ausgeglichener Baum sein oder nicht. Zum Beispiel kann der Indexbaum **324** einen AVL-Baum oder einen binären Suchbaum umfassen.

[0057] Das Serversystem **106** kann ferner eine lineare Liste **330** aufweisen. Die lineare Liste weist den Datenelementen **326** zugeordnete Dateneinheiten auf. Die Einheiten linearer Ordnung werden in ganzzahligen Nummern, z. B. 2, 5, 11 usw. gezeigt. Die Dateneinheiten haben in der Liste eine lineare Ordnung, die der Ordnung entspricht, in der die verschlüsselten Datenelemente **326** im Indexbaum **324** gespeichert werden. Die lineare Liste **330** kann zum Identifizieren der Datenelemente verwendet werden, die zu einem bestimmten Bereich oder Suchintervall gehören. Jede Einheit der linearen Liste **330** kann eine Referenz auf Daten der Datenbank **322** aufweisen, die dem Datenelement **326** entsprechen, das von diesem Element repräsentiert wird. Die Nutzung der linearen Liste **330** kann vorteilhaft sein, da der Zugriff auf die lineare Liste schneller als der Zugriff auf den Indexbaum **324** sein kann.

[0058] Das Serversystem **106** kann ferner einen Cache **344** aufweisen.

[0059] Das Client-Serversystem **100**, das Client-System **102** und das Serversystem **106** werden mit Bezug auf die **Fig.** 4 bis **Fig.** 8 ausführlich beschrieben.

[0060] Im Folgenden werden die Begriffe „Relation", „gewurzelter Baum", „partiell geordnete Menge" und „Indexbaum" austauschbar verwendet.

[0061] **Fig.** 4 ist ein Flussdiagramm eines Verfahrens zum Abfragen einer Datenbank, z. B. **322**, die sich in einem Serversystem, z. B. **106**, befindet.

[0062] In Schritt **401** kann das Client-System **102** eine Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, an das Serversystem **106** senden. Die Anforderung kann zum Beispiel eine SQL-Abfrage umfassen. Die Anforderung zeigt verschlüsselte Datenelemente an, die eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bilden. Das Suchintervall kann beispielsweise eine Präfixsuche nach Zeichenketten repräsentieren, wie etwa „B*". Zum Beispiel kann der Benutzer eine Präfixsuche „B*" in das Suchfeld **226** der grafischen Benutzeroberfläche **220** eingeben. Die Präfixsuche kann durch die Datenzugriffsanwendung **208** in eine Intervallsuche, z. B. ["B"; "C"[, umgewandelt werden, die die angeforderten Daten-

sätze (c_data) abdecken würde, die Benutzern mit einem Namen, der mit „B" und „C" anfängt, zugeordnet sind. Die zwei Intervallgrenzen können in diesem Beispiel die Zeichenketten „B" und „C" in ihrem verschlüsselten Format E(B) bzw. E(C) aufweisen.

[0063] In einem weiteren Beispiel kann das Suchintervall eine Reihe von ID-Nummern repräsentieren, z. B. von ID = 1 bis ID = 10, wobei die ID eine Benutzerkennungsnummer anzeigt. Dafür kann der Benutzer die zwei Intervallgrenzen 1 und 10 in die Felder **226** der grafischen Benutzeroberfläche **220** eingeben. Die Anforderung, die vom Client-System **102** gesendet wird, ist zur Anforderung von Datensätzen (c_data), die mit Benutzern assoziiert sind, die eine zwischen 1 und 10 liegende ID haben.

[0064] Das Serversystem **106** kann den Cache **344** prüfen (Anfrage **403**), um zu bestimmen, ob mit dem Datenelement, das die Intervallgrenze bildet (wie unten beschrieben, kann das Serversystem **106** zuvor angeforderte Suchintervalle cachen), ein Cache-Eintrag assoziiert ist. Zum Beispiel kann das Serversystem **106** das verschlüsselte Datenelement E(B) oder E(C) mit dem Inhalt von Cache-Einträgen in dem Cache **344** vergleichen. In einem Beispiel kann der Vergleich Folgendes aufweisen: Nach Bestimmen, dass ein Cache-Eintrag E(B) aufweist, Prüfen, ob der Cache-Eintrag, der E(B) entspricht, auch eine Untergrenze ist. Zum Beispiel kann festgestellt werden, dass E(B) in einem Cache-Eintrag gespeichert ist, dass der Cache-Eintrag aber mit einer Obergrenze in Beziehung steht. Dies kann besonders vorteilhaft sein, falls die Intervallgrenzen nicht paarweise gespeichert werden, z. B. jede Intervallgrenze separat gecacht wird.

[0065] Im Fall, dass ein Cache-Eintrag nicht mit dem Datenelement assoziiert ist, das die Intervallgrenze bildet, z. B. kein Cache-Eintrag im Cache, der das verschlüsselte Datenelement E(B) aufweist, kann das Serversystem **106** in Schritt **405** eine Benachrichtung an das Client-System **102** senden, die anzeigt, dass das gesuchte Intervall nicht im Cache zwischengespeichert ist. In einem weiteren Beispiel kann es sein, dass das Client-System **102** die Benachrichtigung vom Serversystem **106** nicht erfordert. Das Client-System **102** kann nach einer verstrichenen vorgegebenen Zeitspanne, angefangen von dem Zeitpunkt, an dem die Anforderung von Schritt **401** gesendet wurde, automatisch bestimmen, dass das angeforderte Suchintervall nicht im Cache zwischengespeichert ist. Zum Beispiel kann der fehlende Cache-Eintrag auf der Tatsache beruhen, dass der Cache-Eintrag nicht bereits für das verschlüsselte Datenelement E(B) erstellt wurde, oder das Serversystem **106** kann in einem anderen Beispiel diesen Cache-Eintrag gelöscht haben, nachdem vom Serversystem **106** ein neues Datenelement in den Indexbaum **324** eingefügt wurde. Grund dafür ist, dass die früher angeforderten Ergebnisse nicht reproduziert werden können, da der Indexbaum **324** sich verändert hat.

[0066] Beim Bestimmen, dass das Datenelement, das die Intervallgrenze bildet, am Serversystem **106** nicht im Cache zwischengespeichert ist, kann das Client-System **102** den Indexbaum **324** traversieren, um in Schritt **407** ein jeweiliges Datenelement des Indexbaums **324**, das die Intervallgrenze bildet, z. B. E(B), zu bestimmen. Das Traversieren von Schritt **407** kann wie mit den Schritten **407A**–**407G** beschrieben durchgeführt werden.

[0067] Das Client-System **102** kann in Schritt **407A** einen oder mehr Teile (oder Teilbäume) **421A**–B von Datenelementen des Indexbaums **324** anfordern. Ein Teil des Indexbaums **324** kann ein oder mehr Datenelemente **326** aufweisen. Zum Beispiel kann im Fall, dass der angeforderte Teil des Indexbaums **324** mehr als einen Knoten aufweist, die jeweilige Anforderung ferner die Höhe des angeforderten Teils anzeigen. Die angeforderten Teile oder Teilbäume können eine gleiche oder verschiedene vordefinierte Höhe haben, wobei die Höhe die Anzahl von Kanten am längsten Abwärtsweg zwischen dem Wurzelknoten eines Teilbaums und der Blattebene dieses Teilbaums festsetzt. Zum Beispiel kann das Client-System **102** in der Anforderung eines vorgegebenen Teils des Indexbaums **324** die Höhe dieses Teils ungeachtet der Position des Teils im Indexbaum **324** festsetzen. Ferner oder alternativ kann das Client-System **102** in der Anforderung den Wurzelknoten oder Anfangsknoten eines vorgegebenen Teils des Indexbaums **324** festsetzen (z. B. kann unter Verwendung der Blätter eines traversierten Teils der Anfangsknoten (oder Wurzelknoten) eines anzufordernden nachfolgenden Teils bestimmt werden). Das Client-System **102** kann über das Netz **104** eine Verbindung mit dem Serversystems **106** herstellen und kann unter Verwendung eines Protokolls wie HTTP aufeinanderfolgende Anforderungen an das Serversystem **106** senden. Für jede der aufeinanderfolgenden Anforderungen kann das Client-System **102** einen einzelnen Teil des Indexbaums **324** anfordern.

[0068] In Schritt **407B** kann das Serversystem **106** den angeforderten Teil an das Client-System **102** senden. Zum Beispiel kann das Serversystem **106** den angeforderten Teil in einer Antwort an das Client-System **102** senden. Das Serversystem **106** kann bestimmen, in welcher Ordnung der Teil an das Client-System **102** zu senden ist. Zum Beispiel kann das Serversystem **106** beginnen, den Teil **421A**, der den Wurzelknoten des Indexbaums **324** enthält, zu senden. Dies kann den Durchlauf der Teile am Client-System **102** erleichtern, da es sein kann, dass das Client-System **102** zur Durchführung des Durchlaufs mit dem Wurzelknoten beginnen muss.

[0069] In Schritt **407C** kann das Client-System **102** die angeforderten Teile **421A**–B in einer oder mehr Antworten von dem Serversystem **106** erhalten.

[0070] Da die Datenelemente der Teile **421A**–B verschlüsselt sind, kann das Client-System **102** im Schritt **407D** die Datenelemente **326** der erhaltenen Teile **421A**–B entschlüsseln, z. B. unter Verwendung des kryptografischen Schlüssels **222**, der am Client-System **102** zum Verschlüsseln der Datenelemente des Indexbaums **324** verwendet wurde.

[0071] Beim Auslesen des unverschlüsselten Inhalts der Datenelemente **326** der erhaltenen Teile **421** kann das Client-System **102** in Schritt **407E** das jeweilige Datenelement, das die Intervallgrenze E(B) bildet, anhand der entschlüsselten Elemente bestimmen. Dafür kann das Client-System **102** das die Intervallgrenze (z. B. E(B)) bildende Datenelement entschlüsseln, wenn der unverschlüsselte Inhalt von E (B) am Client-System **102** (nach Durchführen von Schritt **401**) nicht gepflegt wird, und kann den unverschlüsselten Inhalt von E(B), der „B" ist, mit dem unverschlüsselten Inhalt der Datenelemente **326** der erhaltenen Teile **421A**–B vergleichen. Das Client-System **102** kann daher bestimmen, dass das „B" am nächsten liegende Datenelement des Indexbaums **324** das Datenelement „Barton" ist.

[0072] Nach Bestimmen des die Intervallgrenze E (B) bildenden Datenelements kann das Client-System **102** in Schritt **407F** eine Anforderung an das Serversystem **106** senden zum Abrufen der Dateneinheit (II_B) der linearen Ordnung oder der linearen Liste **330**, die an das jeweilige verschlüsselte Datenelement annotiert ist, das in Schritt **407E** als die Intervallgrenze bildend bestimmt wurde. Zum Beispiel kann das Client-System **102** das Datenelement „Barton" unter Verwendung des kryptografischen Schlüssels **222** verschlüsseln, um E(Barton) zu erhalten, und kann E(Barton) in der Anforderung an das Server-System **106** senden, um z. B. durch das Serversystem **106** die Dateneinheit der linearen Liste **330** zu bestimmen, die E(Barton) entspricht. Zum Beispiel kann das Serversystem **106** bestimmen, dass die Dateneinheit II_B, die dem E(Barton) entspricht, die Ganzzahl 2 ist, wie im Indexbaum **324** angezeigt wird.

[0073] Als Reaktion auf den Erhalt der Anforderung von Schritt **407F** kann das Serversystem **106** in Schritt **407G** für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze E(B) bildend bestimmt wurde, einen Cache-Eintrag generieren, der das jeweilige verschlüsselte Datenelement E(B), das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit II_B aufweist. Zum Beispiel kann im Cache-Eintrag ein Tripel (E(B), untere, II_B) gespeichert werden, um die Intervallgrenze anzuzeigen. Die Zeichenfolge „untere" („obere") zeigt an, dass die Intervallgrenze eine untere (obere) Intervallgrenze ist.

[0074] Im Fall (Anfrage **403**), dass ein Cache-Eintrag mit einem Datenelement assoziiert ist, das die Intervallgrenze (E(B)) bildet, kann das Serversystem **106** in Schritt **409** die Dateneinheit II_B, die mit der Intervallgrenze „B" assoziiert ist, aus dem Cache **344** abrufen, zum Beispiel durch Auslesen des Tripels (E(B), untere, II_B). In diesem Fall kann das vorliegende Verfahren Ressourcen sparen, die anderweitig zum Durchführen der Schritte **405** bis **407** benötigt würden.

[0075] Die Schritte **405** bis **409** können für die andere Intervallgrenze „C" wiederholt werden, um das Datenelement **326** des Indexbaums **324** zu bestimmen, das die Intervallgrenze „C" bildet. In dem in **Fig.** 4 gezeigten Beispiel kann das Datenelement „Bronte" ein Datenelement sein, das die Intervallgrenze „C" bildet.

[0076] In Schritt **411** können alle verschlüsselten Datenelemente, die annotierte Einheiten der linearen Liste haben, die zwischen den Datenelementen (z. B. II_B und II_C) liegen, die mit den Datenelementen („Barton" und „Bronte") assoziiert sind, die die Intervallgrenzen „B" und „C" bilden, identifiziert werden. Zum Beispiel kann die lineare Liste **330** zum Entnehmen von Datenelementen verwendet werden, die zwischen II_B und II_C in der linearen Liste **330** liegen. Diese Dateneinheiten können im Beispiel von **Fig.** 4 die Datenelemente „Ben", „Blyte" repräsentieren, da sie Ganzzahlen haben, die zwischen 2 und 7 des „Barton" bzw. „Bronte" liegen. Die Einheiten der linearen Liste **330**, die mit den Datenelementen „Barton", „Ben", „Blyte" und „Bronte" assoziiert sind, können ihre jeweiligen Referenzen r3, r7, r1 und r4 anzeigen. Diese Referenzen können vom Serversystem **106** in Schritt **413** zum Abrufen der Datensätze verwendet werden, die angefordert sind und die dem gesuchten Intervall ["B", "C"[entsprechen.

[0077] In Schritt **415** kann das Serversystem **106** die angeforderten Datensätze an das Client-System **102** senden.

[0078] In einem Beispiel kann der Schritt **411** von dem Serversystem **106** durchgeführt werden. Dies kann im Fall von begrenzter Netzbandbreite im Client-Server-System **100** vorteilhaft sein, da es möglicherweise nicht erforderlich ist, die Elemente II_B und II_C an das Client-System **102** zu senden.

[0079] In einem weiteren Beispiel kann der Schritt **411** von dem Client-System **102** durchgeführt werden. Dafür kann das Client-System **102** die Dateneinheiten II_B und II_C, die mit den intervallgrenzen „B" und „C" assoziiert sind, vom Serversystem **106** erhalten und kann die Identifizierung unter Verwendung der linearen Liste **330** durchführen, die auch am Client-System gespeichert sein kann. Danach kann das Cli-

ent-System **102** die identifizierten Dateneinheiten an das Serversystem **106** senden. Dies kann den Vorteil haben, dass der Sicherheitsaspekt des vorliegenden Verfahrens durchgesetzt wird, da das Client-System uneingeschränkte Kontrolle über den Datenzugriff am Serversystem **106** haben kann.

[0080] Fig. 5 ist ein Flussdiagramm eines Verfahrens zum Einschränken des Zugriffs auf im Cache zwischengespeicherte Teile **421A**–B an dem Client-System **102**. Zum Beispiel kann das Client-System **102** erhaltene Teile **421A**–B des Indexbaums **324** (von Schritt **407C**) im Cache **224** zwischenspeichern.

[0081] In Schritt **501** kann ein neues verschlüsseltes Datenelement **326** am Serversystem **106** in den Indexbaum **324** eingefügt werden.

[0082] Nach Einfügen des neuen verschlüsselten Datenelements und im Fall, das an einem vorgegebenen Knoten **326** des Indexbaums **324** eine Ungleichmäßigkeit festgestellt wird, kann das Serversystem **106** in Schritt **503** den Indexbaum **326** an einem von dem vorgegebenen Knoten definierten Rotationspunkt neu ausgleichen. Da der Index aktualisiert und neu ausgeglichen worden ist, kann das Serversystem **106** eine Benachrichtigung an das Client-System **102** senden, die anzeigt, dass das neue verschlüsselte Datenelement in den Indexbaum **324** eingefügt worden ist. Das Client-System **102** kann daher in Schritt **505** gecachte Teile löschen, die Knoten entlang eines Abwärtswegs haben, angefangen vom Wurzelknoten eines gecachten Teils der gecachten Teile, der den vorgegebenen Knoten enthält, bis zu den Blättern des gewurzelten Baums. Dies kann die Nutzung veralteter Teile des Indexbaums **324** am Client-System **102** verhindern.

[0083] Fig. 6 ist ein Flussdiagramm eines Verfahrens, das den Schritt **505** weiter ausführlicher darlegt. In Schritt **601** kann das Serversystem **106** den Indexbaum **324**, angefangen mit dem neuen eingefügten verschlüsselten Datenelement **621** aufwärts zum Wurzelknoten **623** des Indexbaums **324**, erneut traversieren. Das erneute Traversieren hat dabei einen Durchlaufweg **625** zur Folge, wie in **Fig.** 6 veranschaulicht. Das erneute Traversieren kann vom Serversystem **106** automatisch durchgeführt werden. In einem weiteren Beispiel kann das Serversystem **106** eine Anweisung vom Client-System **102** zum Durchführen des erneuten Traversieren erhalten.

[0084] In Schritt **603** kann das Serversystem **106** den Durchlaufweg **625** dem Client-System mitteilen. Der Durchlaufweg kann die traversierten Knoten anzeigen/aufweisen und kann auch den Rotationspunkt anzeigen. Zum Beispiel kann einer der traversierten Knoten des Durchlaufs, der dem Rotationspunkt entspricht, markiert werden, um anzuzeigen, dass er der Rotationspunkt ist. In einem weiteren Beispiel kann der Knoten (das Datenelement), der dem Rotationspunkt entspricht, außerdem auch zum Durchlaufweg gesendet werden. Der Durchlaufweg **625** kann in der Form einer Textdatei oder einer Datenstruktur (oder unter Verwendung einer Bitmaske, wie mit Bezug auf **Fig.** 7 beschrieben) gesendet werden, die die Datenelemente anzeigt, die Knoten des Indexbaums **324** entsprechen, die erneut traversiert wurden, und die Ordnung anzeigen, in der die Knoten des Durchlaufwegs traversiert werden können. Zum Beispiel kann eine Nummerierung verwendet werden, um es dem Client-System zu ermöglichen, zum Traversieren der Knoten des Durchlaufwegs der Nummerierung zu folgen. Die Nummerierung weist dabei jedem Knoten (Datenelement) eine entsprechende Nummer zu, z. B. kann der erste (oder Anfangs-)Knoten des Durchlaufwegs als erster nummeriert (z. B. 1) nummeriert werden, der zweite traversierte Knoten kann als 2 nummeriert werden und so weiter.

[0085] Bei Erhalt des Traversierungswegs **625** kann das Client-System **102** in Schritt **605** den Traversierungsweg **625** (z. B. in der Textdatei angezeigte Daten) zum Traversieren der gecachten Teile **421A**–B zum Identifizieren eines gecachten Teils durch das Client-System **102** verwenden, der den Rotationspunkt **627** in den gecachten Teilen **421A**–B enthält.

[0086] In Schritt **607** kann das Client-System **102** in den gecachten Teilen **421A**–B Knoten entlang eines Abwärtswegs, angefangen an dem Wurzelknoten des identifizierten gecachten Teils bis zu den Blättern des Indexbaums **324**, identifizieren.

[0087] In Schritt **609** kann das Client-System **102** den identifizierten gecachten Teil und die identifizierten Knoten löschen.

[0088] Fig. 7 ist ein Flussdiagramm eines beispielhaften Verfahrens zum Durchführen des Traversierens von Schritt **605**. Der Einfachheit halber wird die Beschreibung des Durchlaufwegs **721** als Kanten, die traversierte Knoten **723A**–D verbinden, aufweisend gezeigt. Jede Kante wird von einem Paar von zwei aufeinanderfolgend geordneten Knoten definiert. Zum Beispiel wird die erste geordnete Kante von den Knoten **723A** und **723B** definiert. Die zweite Kante wird von den Knoten **723B** und **723C** definiert. Die dritte Kante wird von den Knoten **723C** und **723D** definiert.

[0089] In Schritt **701** kann das Serversystem **106** eine Bitmaske erstellen, um den Durchlaufweg und die Knoten des Durchlaufwegs zu beschreiben oder anzuzeigen. Die Bitmaske weist Bits auf, die jeweiligen Kanten des Durchlaufwegs in der Ordnung, in der die Knoten im Durchlaufweg traversiert werden, assoziiert sind. Jedes Bit der Bitmaske für eine vorgegebene Kante hat einen Wert, der anzeigt, ob der Zielknoten im Durchlaufweg dem Quellknoten mit Be-

zug auf die von der partiellen Menge repräsentierten Ordnung vorangeht oder nachfolgt. Der Ziel- und der Quellknoten definieren die vorgegebene Kante. Im Folgenden wird das Beispiel des Durchlaufwegs **721** betrachtet, der durch die Knoten **723A**–D verläuft. Die Bitmaske kann in diesem Fall 3-Bits aufweisen, die jeweils mit einer der drei Kanten assoziiert sind, die von den Paaren **723A**–B, **723B**–C bzw. **723C**–D definiert werden. Der erste Bitwert für die von den Knoten **723A**–B definierte Kante kann anzeigen, dass der Knoten **723B** eine Ordnung hat, die kleiner als die Ordnung des Knoten **723A** ist (d. h. „Blythe" geht in der für den Indexbaum **324** verwendeten Ordnung „Elwood" voraus). Zum Beispiel kann der erste Bitwert gleich 0 sein, um anzuzeigen, dass der Knoten **723B** dem Knoten **723B** in der Ordnung vorausgeht. Der zweite Bitwert für die zweite Kante, die von den Knoten **723B**–C definiert wird, kann aus demselben Grund wie für das erste Bit beschrieben den gleichen Wert 0 haben. Das dritte Bit der dritten Kante, die von den Knoten **723C**–D definiert wird, kann aber einen anderen Bitwert, z. B. 1, haben, um anzuzeigen, dass der Knoten **723D** eine dem Knoten **723C** nachfolgende Ordnung hat.

[0090] In Schritt **703**, kann das Serversystem **106** die erstellte Bitmaske an das Client-System **102** senden. Und beim Erhalt der Bitmaske durch das Client-System **102** vom Serversystem **106** kann das Client-System **102** in Schritt **705** die Bitmaske zum Traversieren der gecachten Teile verwenden. Das Client-System **106** kann zum Beispiel vom Serversystem **106** Anweisungen erhalten, um die Bitwerte wie oben beschrieben zu interpretieren. Zum Beispiel kann der Teil **421A** vom Wurzelknoten ausgehend traverisiert werden, indem jeder der Bitwerte des Client-Systems ausgelesen wird, kann das Client-System den Teil **421A** am Durchlaufweg **721** entlang traversieren.

[0091] Fig. 8 ist ein Flussdiagramm eines Verfahrens zum Durchführen des Traversierens von Schritt **605**.

[0092] In Schritt **801** kann das Serversystem **106** das verschlüsselte Datenelement, das dem Rotationspunkt **627** entspricht, dem Client-System **102** über ein Netzwerk **104** mitteilen.

[0093] In Schritt **803** kann das Client-System als Reaktion auf den Erhalt des verschlüsselten Datenelements, das dem Rotationspunkt **627** entspricht, das erhaltene verschlüsselte Datenelement entschlüsseln. Dies kann zum Beispiel unter Verwendung des kryptografischen Schlüssels **222** erfolgen.

[0094] In Schritt **805** kann das Client-System **102** das entschlüsselte Datenelement im Cache **224** als den vorgegebenen Knoten identifizieren, der den Rotationspunkt **627** bildet.

[0095] Aspekte der vorliegenden Erfindung werden hierin mit Bezug auf Flussdiagrammdarstellungen und/oder Blockdiagramme von Verfahren, Vorrichtungen (Systemen) und Computerprogrammprodukten gemäß Ausführungsformen der Erfindung beschrieben. Es versteht sich, dass jeder Block der Flussdiagrammdarstellungen und/oder Blockdiagramme und Kombinationen von Blöcken in den Flussdiagrammdarstellungen und/oder Blockdiagrammen durch computerlesbare Programmanweisungen implementiert werden

[0096] Die vorliegende Erfindung kann ein System, ein Verfahren und/oder ein Computerprogrammprodukt sein. Das Computerprogrammprodukt kann ein computerlesbares Speichermedium (oder -medien) mit computerlesbaren Programmanweisungen darin zum Veranlassen eines Prozessors zur Durchführung von Aspekten der vorliegenden Erfindung beinhalten.

[0097] Das computerlesbare Speichermedium kann eine reale Vorrichtung sein, die Anweisungen zur Verwendung durch eine Anweisungsausführungsvorrichtung behalten und speichern kann. Das computerlesbare Speichermedium kann zum Beispiel unter anderem eine elektronische Speichervorrichtung, eine magnetische Speichervorrichtung, eine optische Speichervorrichtung, eine elektromagnetische Speichervorrichtung, eine Halbleiterspeichervorrichtung oder ein geeignete Kombination der Vorangehenden sein. Eine nicht umfassende Liste spezifischerer Beispiel für das computerlesbare Speichermedium beinhaltet die Folgenden: eine portable Computerdiskette, eine Festplatte, einen Direktzugriffsspeicher (RAM), einen Festwertspeicher (ROM), einen löschbaren programmierbaren Nur-Lese-Speicher (EPROM oder Flash-Speicher), einen statischen Direktzugriffsspeicher (SRAM), einen portablen Compact Disc-Festwertspeicher (CD-ROM), eine Digital Versatile Disk (DVD), einen Speicher-Stick, eine Diskette, eine mechanisch codierte Vorrichtung, wie etwa Lochkarten oder erhabene Strukturen in einer Rille mit daran aufgezeichneten Anweisungen, und jedwede geeignete Kombination der Vorangehenden. Ein computerlesbares Speichermedium, wie hierin verwendet, darf nicht als kurzzeitige Signale an sich, wie etwa Funkwellen oder andere sich frei ausbreitende elektromagnetische Wellen, elektromagnetische Wellen, die sich durch einen Wellenleiter oder andere Übertragungsmedien ausbreiten (z. B. Lichtimpulse, die durch einen Lichtwellenleiter laufen), oder durch einen Draht übertragene elektrische Signale, ausgelegt werden.

[0098] Hierin beschriebene computerlesbare Programmanweisungen können über ein Netz, z. B. das Internet, ein lokales Netz, ein Weitverkehrsnetz und/oder ein drahtloses Netz, von einem computerlesbaren Speichermedium zu jeweiligen Rechen-/Verar-

beitungsvorrichtungen oder zu einem externen Rechner oder einer externen Speichervorrichtung heruntergeladen werden. Das Netz kann Kupferübertragungskabel, Lichtleitfasern, drahtlose Übertragung, Router, Firewalls, Switches, Gateway-Computer und/oder Edge-Server aufweisen. Eine Netzwerkadapterkarte oder Netzschnittstelle in jeder Rechen-/Verarbeitungsvorrichtung empfängt computerlesbare Programmanweisungen aus dem Netz und leitet die computerlesbaren Programmanweisungen zur Speicherung in einem computerlesbaren Speichermedium innerhalb der jeweiligen Rechen-/Verarbeitungsvorrichtung weiter.

[0099] Computerlesbare Programmanweisungen zur Durchführung von Operationen der vorliegenden Erfindung können Assembleranweisungen, ISA-Anweisungen (ISA: Instruction-Set-Architecture), Maschinenanweisungen, maschinenabhängige Anweisungen, Mikrocode, Firmware-Anweisungen, zustandssetzende Daten oder entweder Quellcode oder Objektcode sein, der in einer Kombination von einer oder mehr Programmiersprachen geschrieben ist, einschließlich einer objektorientierten Programmiersprache wie Smalltalk C++ oder dergleichen und konventionellen Sprachen für die prozedurale Programmierung wie die Porgrammiersprache „C” oder ähnliche Programmiersprachen. Die computerlesbaren Programmanweisungen können vollkommen im Rechner des Benutzers, teilweise im Rechner des Benutzers, als separates Softwarepaket, teilweise im Rechner des Benutzers und teilweise in einem abgesetzten Rechner oder völlig in dem abgesetzten Rechner oder Server ausgeführt werden. In der letzteren Situation kann der abgesetzte Rechner durch irgendeinen Typ von Netz, einschließlich einem lokalen Netz (LAN) oder einem Weitverkehrsnetz (WAN), mit dem Rechner des Benutzers verbunden sein oder die Verbindung kann zu einem externen Rechner (z. B. unter Verwendung eines Internet Service Providers durch das Internet) hergestellt werden. In einigen Ausführungsformen können elektronische Schaltungsanordnungen, einschließlich z. B. programmierbare Logikschaltungen, Field Programmable Gate Arrays (FPGA) oder programmierbare logische Anordnungen (PLA), die computerlesbaren Programmanweisungen durch Nutzen von Zustandsinformationen der computerlesbaren Programmanweisungen zum Individualisieren der elektronischen Schaltung, um Aspekte der vorliegenden Erfindung durchzuführen, ausführen.

[0100] Aspekte der vorliegenden Erfindung werden hierin mit Bezug auf Flussdiagrammdarstellungen und/oder Blockdiagramme von Verfahren, Vorrichtungen (Systemen) und Computerprogrammprodukten gemäß Ausführungsformen der Erfindung beschrieben. Es versteht sich, dass jeder Block der Flussdiagrammdarstellungen und/oder Blockdiagramme und Kombinationen von Blöcken in den Flussdiagrammdarstellungen und/oder Blockdiagrammen durch computerlesbare Programmanweisungen implementiert werden können.

[0101] Diese computerlesbaren Programmanweisungen können einem Prozessor eines Universalrechners, eines Spezialrechners oder einer anderen programmierbaren Datenverarbeitungsvorrichtung zum Erzeugen einer Maschine bereitgestellt werden, so dass die Anweisungen, die über den Prozessor des Rechners oder der anderen programmierbaren Datenverarbeitungsvorrichtung ausgeführt werden, Mittel zum Implementieren der Funktionen/Handlungen, die in dem/den Flussdiagramm- und/oder Blockdiagrammblock oder -blöcken angegeben sind, schaffen. Diese computerlesbaren Programmanweisungen können auch in einem computerlesbaren Speichermedium, das einen Rechner, eine programmierbare Datenverarbeitungsvorrichtung und/oder andere Einrichtungen zum Funktionieren auf eine bestimmte Art und Weise anleiten kann, gespeichert werden, so dass das computerlesbare Speichermedium mit darin gespeicherten Anweisungen einen Fertigungsartikel aufweist, der Anweisungen beinhaltet, die Aspekte der in dem/den Fluss- und/oder Blockdiagrammblock oder -blöcken vorgegebenen Funktion/Handlung implementieren.

[0102] Die computerlesbaren Programmanweisungen können auch in einen Rechner, eine andere programmierbare Datenverarbeitungsvorrichtung oder eine andere Einrichtung geladen werden, um die Durchführung einer Reihe von Operationsschritten in dem Rechner, der anderen programmierbaren Vorrichtung oder der anderen Einrichtung zum Erzeugen eines computerimplementierten Prozesses zu veranlassen, so dass die Anweisungen, die in dem Rechner, der anderen programmierbaren Vorrichtung oder der anderen Einrichtung ausgeführt werden, die in dem/den Flussdiagramm- und/oder Blockdiagrammblock oder -blöcken vorgegebenen Funktionen/Handlungen implementieren.

[0103] Die Fluss- und Blockdiagramme in den Figuren veranschaulichen die Architektur, Funktionalität und Funktionsweise möglicher Implementierungen der Systeme, Verfahren und Computerprogrammprodukte gemäß verschiedenen Ausführungsformen der vorliegenden Erfindung. In dieser Hinsicht kann jeder Block in den Fluss- oder Blockdiagrammen ein Modul, ein Segment oder einen Teil von Anweisungen darstellen, das/der eine oder mehr ausführbare Anweisungen zur Implementierung der vorgegebenen logischen Funktion(en) aufweist. In einigen alternativen Ausführungsformen können die in dem Block angegebenen Implementierungen außerhalb der in den Figuren angegebenen Ordnung stattfinden. Zum Beispiel können zwei nacheinander gezeigte Blöcke in Wirklichkeit im Wesentlichen gleichzeitig ausgeführt werden oder die Blöcke können manch-

mal in umgekehrter Reihenfolge ausgeführt werden, je nach der betreffenden Funktionalität. Es ist auch zu beachten, dass jeder Block der Blockdiagramme und/oder Flussdiagrammdarstellung und Kombinationen von Blöcken in den Blockdiagrammen und/oder der Flussdiagrammdarstellung von hardwaremäßigen Spezialsystemen implementiert werden können, die vorgegebene Funktionen oder Handlungen durchführen oder Kombinationen von Spezialhardware und Computeranweisungen ausführen.

[0104] Im Folgenden wird ein Beispielverfahren für Server- und Client-Caching beschrieben.

[0105] Ein erster Caching-Ansatz ist die Speicherung zeitweiliger Bereichssuchergebnisse im Servercomputer. Man erwäge zum Beispiel einen Client-Computer, der eine Bereichssuche für alle Werte von Index **324** (als IX bezeichnet), die in [a, b] liegen, beantragt, die zwei Baumdurchläufe einleiten können, wobei das Ergebnis ein Paar der am stärksten übereinstimmenden Kandidaten x und y für a bzw. b ist, wobei x und y Datenelemente des Indexes **324** sind.

[0106] Dem kann eine Auswahl aller Indexelemente folgen, deren Nummer in der linearen Liste (z. B. in der linearen Liste **330**) zwischen den Nummern x und y der linearen Liste liegt:

$$O_{Value\in[a,b]}(IX) = O_{ll\in[ll_x,ll_y]}(IX) = rs$$

[0107] Die resultierende Tupelmenge rs könnte im Servercomputer gespeichert werden, falls ein anderer Client-Computer genau dieselbe Anfrage für in [a, b] liegende Indexwerte ausgibt.

[0108] Ein kompakteres Caching-Verfahren wird ebenfalls beschrieben: Für [a, b] anstatt von rs ein Tripel für jeden Intervallgrenzwert: (E(a), untere, ll_x) und (E(b), obere, ll_y), wobei

- E(a) und E(b) der verschlüsselte Grenzwert sind, der als Identifikator verwendet wird,
- "untere" und "obere" der Grenztyp ist (kann auch als Teil eines Identifikators verwendet werden)
- ll_x und ll_y der Wert in der linearen Liste ist, der in der aktuellen Baumerweiterung für die vorgegebene Intervallgrenze, a und b, die größte Übereinstimmung hat.

[0109] Wenn ein Client-Computer eine Bereichssuche für [a, b] beantragt, verschlüsselt der Client-Computer zunächst a und b und sendet (E(a), untere) und (E(b), obere) zum Servercomputer, wo der Cache auf mögliche Treffer abgesucht wird. Wenn ein oder beide Cache-Treffer stattfinden, wird die Nummer in der linearen Liste für die endgültige Auswahl der Bereichssuchoperation abgerufen. Wenn nicht, beginnt der Baumdurchlauf wie oben beschrieben (z. B. zum Identifizieren von ll_x und ll_y).

[0110] Ein gecachtes Tripel kann veraltet sein, wenn der Index **324** geändert wird, z. B. wenn ein Indexelementwert eingefügt wird, dessen Wert eine stärkere Übereinstimmung für die verschlüsselte Intervallgrenze des Tripels ist. In diesem Fall muss das Tripel aus dem Cache gelöscht werden. Dies kann zum Beispiel durch Löschen des Server-Caches **344** jedesmal dann, wenn sich der Index ändert, erfolgen.

[0111] Im Folgenden wird Client-Caching beschrieben.

Client-Caching

[0112] Um so viele redundante Teilbaumabrufe wie möglich zu vermeiden, können Teilbäume, die während vorheriger Suchen abgerufen wurden, zur späteren Verwendung im Client-Computer gespeichert werden. Der „Wurzel-Teilbaum" zum Beispiel, d. h. der höchste Teilbaum unter dem Wurzelknoten des gesamten Baums, wird möglicherweise immer benötigt, weil jeder Durchlauf hier starten kann. Andere gecachte Teilbäume können ebenfalls nützlich sein, wenn der Client-Computer nach Knoten sucht oder beabsichtigt, Knoten innerhalb geringer Entfernung von vorherigen Baumzugriffen einzufügen. Es ist zu beachten, dass für Lese- und für Schreibzugriffe die gleiche Art von Teilbäumen abgerufen werden kann.

[0113] Nach seinem Abrufen wird der Teilbaum also unter dem Knoten mit Kennung i als Cache-Eintrag im Client-Computer gespeichert. Der Cache-Eintrag kann die Knoten $k \leq 2^h - 1$ des Teilbaums enthalten und die Kennung i seines Wurzelknotens kann als Identifikator verwendet werden. Es kann auch vorteilhaft sein, einen Cache-Eintrag als den Wurzelteilbaum r_{root} zu markieren, so dass ein Cache-Eintrag r_i die folgende Struktur hat:

r_i = (i, {($id_{i,1}$, $left_{i,1}$, $right_{i,1}$, $cData_{i,1}$), ..., (idi,k, lefti,k, righti,k, cDatai,k)}, isRoot)

[0114] Wenn also während des Baumdurchlaufs ein Teilbaum mit Wurzelknotenkennung i benötigt wird, kann der Client-Computer zunächst seinen Cache **224** dahingehend prüfen, ob er einen Eintrag ri enthält; wenn nicht, wird ein Serveraufruf getSubtree(i) eingesendet.

[0115] Das Löschen des Cache kann notwendig sein, wenn der Index **324** so modifiziert wurde, dass die gecachten Teilbäume nicht mehr mit ihren Gegenstücken in der aktuellen Erweiterung des Indexes in der Datenbank übereinstimmend sind, d. h. wenn ein neues Element in den bzw. aus dem Index **324** eingefügt oder gelöscht wird. Ohne Einschränkung der Allgemeingültigkeit betrachten wir im Folgenden dieses Teilabschnitts nur Indexeinfügungen; die vorgelegten Konzepte gelten aber trotzdem auch für Indexe, die Löschungen zulassen.

[0116] Zum Beispiel kann eine Cache-Löschung nach jeder Indexmodifikation getätigt werden; immer, wenn ein Element in den Index **324** eingefügt wird, werden alle aktuell angeschlossenen Clients benachrichtigt, alle ihre Cache-Einträge zu löschen.

[0117] In einem weiteren Beispiel kann es sein, wenn ein neues Element in den Index **324** eingefügt wird, dass die binäre Baumdarstellung der linearen Ordnung der Indexelemente, die zum Beispiel unter Verwendung eines AVL-Baums implementiert sein kann, neu ausgeglichen werden muss. Der Servercomputer führt dies durch erneutes Traversieren des Durchlaufwegs des neuen Knotens zurück zur Baumwurzel durch, wobei er die Gleichmäßigkeit jedes Knotens entlang des Wegs kontrolliert.

[0118] Im Fall einer Ungleichmäßigkeit, je nach ihrem Typ, wird an einem Knoten (dieser Knoten wird im Folgenden als der „Rotationspunkt" bezeichnet) eine zutreffende Neuausgleichoperation („Rotation") des Indexes durchgeführt. Sie ändert die Positionen, Kind-Eltern-Relationen und Teilbaumhöhen mehrerer Knoten um den Rotationspunkt. Während des Neuausgleichs kann höchstens eine derartige Rotation stattfinden, wenn also eine Rotation stattgefunden hat, ist der gesamte AVL-Baum neu ausgeglichen und der erneute Durchlauf wird abgebrochen. Die Gleichmäßigkeit aller Knoten, die außerhalb des Durchlaufwegs liegen, wird von der gesamten Einfügungs- und Neuausgleichsoperation nicht beeinflusst.

[0119] Das Kombinieren der Rotation auf dem Durchlaufweg mit gecachten Teilbäumen lässt sich am besten mit einem Beispiel veranschaulichen. **Fig.** 6 zeigt eine Menge von Cache-Einträgen ri, die mit ihrer jeweiligen Wurzelknotenkennung id gekennzeichnet sind, und einen Durchlaufweg eines neuen Knotens v_x, der auf der Blattebene r_{916} eingefügt ist. Da der Durchlaufweg nur r_{916}, r_{2824} und r_{1741} durchkreuzt, sind die anderen drei Cache-Einträge r_{1053}, r_{2549} und r_{10} von der Einfügung nicht betroffen. Angenommen, das die Einfügung von v_x eine Rotation in r_{2824} verursacht hat, betrifft dies darüber hinaus nur r_{2824} selbst, während die anderen Cache-Einträge, die den Durchlaufweg hosten, r_{1741} und r_{916}, unverändert bleiben.

[0120] Daher braucht in einem Beispiel nur derjenige Cache-Eintrag gelöscht zu werden (in diesem Beispiel r_{2824}), in dem die Neuausgleichsoperation stattfindet, und alle anderen können bleiben. In einem weiteren Beispiel kann es aus folgendem Grund sein, dass mehr Cache-Einträge gelöscht werden:

[0121] Möglicherweise ändert eine Rotation, die in einem Teilbaum stattfindet, Gleichmäßigkeiten und Höhen der Knoten in den Kind-Teilbäumen des Teilbaums nicht, kann aber die Baumebenen einiger dieser Knoten verändern. Weil die Teilbäume mit einer definierten Höhe abgerufen werden, kann dies gecachte Kind-Teilbäume eines Teilbaums nutzlos und in einigen Fällen sogar fehlerhaft machen.

[0122] In diesem Beispiel, wobei eine Teilbaumhöhe h = 5 angenommen wird, befinden sich die Wurzelknoten v_{1053} und v_{916} von r_{1053} und r_{916} vor der Rotation beide auf Baumebene 11. Nach der Rotation können sich diese Baumebenen zu 10 und 12 oder umgekehrt geändert haben. Während aller Baumdurchläufe werden aber nur Teilbäume mit Wurzelknoten auf Baumebene 1, 6, 11, 16 usw. angefordert und r_{1053} und r_{916} würden als tote Einträge im Client-Cache bleiben.

[0123] Daher werden der Cache-Eintrag mit dem Teilbaum, der den Rotationspunkt enthält, sowie alle Cache-Einträge mit den transitiven Kind-Teilbäumen dieses Teilbaums, z. B. Cache-Einträge r_{2824}, r_{1053} und r_{916}, gelöscht. Alle anderen Teilbäume und insbesondere alle Teilbäume am Durchlaufweg entlang über dem Teilbaum, der den Rotationspunkt enthält, können im Cache bleiben.

[0124] Die Erfindung kann mit den folgenden Merkmalskombinationen beschrieben werden.

1. Ein Verfahren zum Abfragen einer Datenbank, die sich in einem Servercomputer befindet, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Verfahren ferner Folgendes aufweist:
   Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, von einem Client-Computer zu dem Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
   für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
   Bestimmen, ob ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgren-

ze bildet, assoziiert ist, wobei im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: Traversieren der partiell geordneten Menge durch den Client-Computer zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei das Traversieren Folgendes aufweist:
Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer;
Erhalten der angeforderten Teile von dem Servercomputer;
Entschlüsseln der Datenelemente der erhaltenen Teile;
Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Elemente;
Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde,
Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde;
im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache aus dem Servercomputer;
Identifizieren aller verschlüsselten Datenelemente, die mit den Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die intervallgrenzen bilden;
Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

2. Das Verfahren mit Merkmalskombination 1, wobei das Datenelement, das die Intervallgrenze bildet, das Datenelement der Relation ist, das der Intervallgrenze am nächsten ist, wobei das Datenelement innerhalb des Suchintervalls ist.

3. Das Verfahren mit Merkmalskombination 1, das ferner Folgendes aufweist:
Einfügen eines neuen Datenelements in den gewurzelten Baum am Servercomputer und
nach dem Einfügen Löschen der gecachten Einträge durch den Servercomputer.

4. Das Verfahren mit Merkmalskombination 1, wobei die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum bilden, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei der eine oder die mehreren Teile Teilbäume des gewurzelten Baums umfassen, wobei die Teilbäume alle eine gemeinsame vordefinierte Höhe haben, wobei die Höhe die Anzahl von Kanten am längsten Abwärtsweg zwischen dem Wurzelknoten des Teilbaums und der Blattebene des Teilbaums festsetzt.

5. Ein Verfahren mit Merkmalskombination 1, das ferner Folgendes aufweist: Cachen der erhaltenen Teile der partiell geordneten Menge durch den Client-Computer.

6. Ein Verfahren mit Merkmalskombination 5, wobei die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum bilden, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei das Verfahren ferner Folgendes aufweist:
Einfügen eines neuen verschlüsselten Datenelement in den gewurzelten Baum am Servercomputer;
im Fall, dass an einem vorgegebenen Knoten eine Ungleichmäßigkeit festgestellt wird, Neuausgleichen des gewurzelten Baums an einem von dem vorgegebenen Knoten definierten Rotationspunkt;
Löschen gecachter Teile durch den Client-Computer, die an einem Abwärtsweg entlang, angefangen am Wurzelknoten eines gecachten Teils, der den vorgegebenen Knoten enthält, bis zu den Blättern des gewurzelten Baums, Knoten haben.

7. Ein Verfahren mit Merkmalskombination 6, wobei das Löschen der gecachten Teile Folgendes aufweist:
erneutes Traversieren des gewurzelten Baums, angefangen mit dem neuen eingefügten verschlüsselten Datenelement aufwärts zum Wurzelknoten des gewurzelten Baums, wobei das erneute Traversieren einen Durchlaufweg zur Folge hat;
dem Client-Computer Mitteilen des Durchlaufwegs, wobei der Durchlaufweg den Rotationspunkt und die traversierten Knoten anzeigt;
Verwenden des Durchlaufwegs zum Traversieren der gecachten Teile durch den Client-Computer zum Identifizieren eines gecachten Teils, der den Rotationspunkt enthält;
in den gecachten Teilen Identifizieren jedweder Knoten an einem Abwärtsweg entlang, angefangen am Wurzelknoten des identifizierten gecachten Teils bis zu den Blättern des gewurzelten Baums;
Löschen des identifizierten gecachten Teils und der identifizierten Knoten.

8. Das Verfahren mit Merkmalskombination 7, wobei der Durchlaufweg eine oder mehr Kanten aufweist, wobei jede Kante von einem Paar von zwei aufeinanderfolgend angeordneten Knoten definiert wird, wobei die zwei aufeinanderfolgend angeordneten Knoten einen Quellknoten und einen Zielknoten des Durchlaufwegs aufweisen, wobei das Traversieren aufweist: Erstellen einer Bitmaske durch den Servercomputer, wobei die Bitmaske Bits aufweist, die mit Kanten des

Durchlaufwegs in der Ordnung, in der die Kanten in dem Durchlaufweg traversiert werden, assoziiert sind, wobei jedes Bit der Bitmaske für eine vorgegebene Kante einen Wert hat, der anzeigt, ob der Zielknoten der Kante dem Quellknoten der Kante mit Bezug auf die von der partiellen Ordnung repräsentierte Ordnung vorangeht oder nachfolgt; Senden der Bitmaske an den Client-Computer; Verwenden der Bitmaske durch den Client-Computer, um die gecachten Teile zu traversieren.

9. Das Verfahren mit Merkmalskombination 7, das ferner Folgendes aufweist:
dem Client-Computer Mitteilen des verschlüsselten Datenelements, das dem Rotationspunkt entspricht;
Entschlüsseln des mitgeteilten Datenelements durch den Client-Computer; Identifizieren des entschlüsselten Datenelements in dem Cache als den vorgegebenen Knoten, der den Rotationspunkt bildet.

10. Das Verfahren mit Merkmalskombination 7, das ferner das Bestimmen aufweist, ob ein Neuausgleichen des Baums unter Verwendung des erneuten Durchlaufs erforderlich ist.

11. Das Verfahren mit Merkmalskombination 1, wobei die partiell geordnete Menge einen AVL-Baum umfasst.

12. Das Verfahren mit Merkmalskombination 1, wobei der erzeugte Cache-Eintrag ferner anzeigt, dass die Intervallgrenze eine untere oder eine obere Intervallgrenze ist.

13. Das Verfahren mit Merkmalskombination 1, das ferner das Erhalten einer Suchanforderung an dem Client aufweist, wobei die Suchanforderung eine Anforderung für eine Präfixsuche nach einem Suchkriterium aufweist, wobei der Client funktionell ist zum Bestimmen des Suchintervalls durch Umwandeln der Präfixsuche in ein entsprechendes Intervall, das das Suchkriterium als die untere Intervallgrenze und einen rechnerisch von dem Suchkriterium abgeleiteten Wert als die obere Intervallgrenze aufweist.

[0125] Ein Verfahren für einen Servercomputer, wobei der Servercomputer eine Datenbank aufweist, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Verfahren ferner Folgendes aufweist:
Erhalten einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, von einem Client-Computer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Intervallgrenze bildet:
Bestimmen, ob ein Dateneintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist,
im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist:
Empfangen einer Anforderung von dem Client-Computer, die die Dateneinheit der linearen Ordnung anzeigt, die an das jeweilige verschlüsselte Datenelement annotiert ist, das die Intervallgrenze bildet;
Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde;
im Fall, dass ein Cache-Eintrag mit einem Datenelement, das die Intervallgrenze bildet, assoziiert ist, Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache;
wobei der Servercomputer ferner konfiguriert ist zum Identifizieren aller verschlüsselten Datenelemente, an die die Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und
Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

[0126] Ein Verfahren für einen Client-Computer eines Computersystems nach den vorherigen Ausführungsformen, wobei das Computersystem einen Servercomputer aufweist, wobei der Servercomputer eine Datenbank aufweist, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Datenein-

heiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert werden, wobei das Verfahren Folgendes aufweist:
Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, an den Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: Traversieren der partiell geordneten Menge zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei zum Traversieren:
Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer;
Erhalten der angeforderten Teile von dem Servercomputer; Entschlüsseln der Datenelemente der erhaltenen Teile;
Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Teile;
Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde.

[0127] Ein Computerprogrammprodukt, wobei das Computerprogrammprodukt ein computerlesbares Speichermedium aufweist, das damit ausgestaltete Programmanweisungen hat, wobei die Programmanweisungen durch einen Prozessor ausführbar sind, um den Prozessor zu veranlassen, eine der vorhergehenden Merkmalskombinationen durchzuführen.

Schutzansprüche

1. Computersystem zum Abfragen einer Datenbank, die sich in einem Servercomputer des Computersystems befindet, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Computersystem ferner einen Client-Computer aufweist,
wobei der Client-Computer konfiguriert ist zum Senden einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, an den Servercomputer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Grenze bildet:
der Servercomputer konfiguriert ist zum Bestimmen, ob ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, wobei im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist: der Client-Computer konfiguriert ist zum Traversieren der partiell geordneten Menge zum Bestimmen eines jeweiligen Datenelements der partiell geordneten Menge, das die Intervallgrenze bildet, wobei zum Traversieren:
der Client-Computer konfiguriert ist zum:
Anfordern von einem oder mehr Teilen von Datenelementen der partiell geordneten Menge vom Servercomputer;
Erhalten der angeforderten Teile von dem Servercomputer;
Entschlüsseln der Datenelemente der erhaltenen Teile;
Bestimmen des jeweiligen Datenelements, das die Intervallgrenze bildet, anhand der verschlüsselten Teile;
Senden einer Anforderung an den Servercomputer zum Abrufen der Dateneinheit der linearen Ordnung, die an das jeweilige verschlüsselte Datenelement annotiert ist, das als die Intervallgrenze bildend bestimmt wurde;
und der Servercomputer zum Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, konfiguriert ist;
der Servercomputer im Fall, dass ein Cache-Eintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist, zum Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache konfiguriert ist;
wobei der Servercomputer zum Bereitstellen der angeforderten Datensätze ferner konfiguriert ist zum
Identifizieren aller verschlüsselten Datenelemente, die mit Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und

Abrufen der angeforderten Datensätze unter Verwendung der identifizierten Dateneinheiten über ihre referentiellen Verbindungen.

2. System nach Anspruch 1, wobei das Datenelement, das die Intervallgrenze bildet, das Datenelement der Relation ist, das der Intervallgrenze am nächsten ist, wobei das Datenelement innerhalb des Suchintervalls ist.

3. System nach einem der vorhergehenden Ansprüche, wobei der Servercomputer ferner konfiguriert ist zum:
– Einfügen eines neuen Datenelements in den gewurzelten Baum am Server-Computer und
– nach Einfügung Löschen der gecachten Einträge.

4. System nach einem der vorhergehenden Ansprüche, wobei die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum bilden, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei der eine oder die mehreren Teile Teilbäume des gewurzelten Baums umfassen, wobei die Teilbäume alle eine gemeinsame vordefinierte Höhe haben, wobei die Höhe die Anzahl von Kanten am längsten Abwärtsweg zwischen dem Wurzelknoten des Teilbaums und der Blattebene des Teilbaums festsetzt.

5. System nach einem der vorhergehenden Ansprüche, wobei der Client-Computer ferner zum Cachen der erhaltenen Teile der partiell geordneten Menge konfiguriert ist.

6. System nach Anspruch 5, wobei die Datenelemente die partiell geordnete Menge in der Relation über einen gewurzelten Baum bilden, wobei jedes der Datenelemente des gewurzelten Baums von einem jeweiligen Knoten repräsentiert wird, wobei der Servercomputer ferner konfiguriert ist zum:
– Einfügen eines neuen verschlüsselten Datenelement in den gewurzelten Baum am Servercomputer;
– im Fall, dass an einem vorgegebenen Knoten eine Ungleichmäßigkeit festgestellt wird, Neuausgleichen des gewurzelten Baums an einem von dem vorgegebenen Knoten definierten Rotationspunkt;
– wobei der Client-Computer ferner konfiguriert ist zum Löschen gecachter Teile, die an einem Abwärtsweg entlang, angefangen am Wurzelknoten eines gecachten Teils, der den vorgegebenen Knoten enthält, bis zu den Blättern des gewurzelten Baums, Knoten haben.

7. System nach Anspruch 6, wobei zum Löschen der gecachten Teile:
der Servercomputer konfiguriert ist zum:
erneuten Traversieren des gewurzelten Baums, angefangen mit dem neuen eingefügten verschlüsselten Datenelement aufwärts zum Wurzelknoten des gewurzelten Baums, wobei das erneute Traversieren einen Durchlaufweg zur Folge hat;
dem Client-Computer Mitteilen des Durchlaufwegs, wobei der Durchlaufweg den Rotationspunkt und die traversierten Knoten anzeigt;
wobei der Client-Computer ferner konfiguriert ist zum:
– Verwenden des Durchlaufwegs zum Traversieren der gecachten Teile zum Identifizieren eines gecachten Teils, der den Rotationspunkt enthält;
– in den gecachten Teilen Identifizieren jedweder Knoten an einem Abwärtsweg entlang, angefangen am Wurzelknoten des identifizierten gecachten Teils bis zu den Blättern des gewurzelten Baums;
– Löschen des identifizierten gecachten Teils und der identifizierten Knoten.

8. System nach Anspruch 7, wobei der Durchlaufweg eine oder mehr Kanten aufweist, wobei jede Kante von einem Paar von zwei aufeinanderfolgend angeordneten Knoten definiert wird, wobei die zwei aufeinanderfolgend angeordneten Knoten einen Quellknoten und einen Zielknoten des Durchlaufwegs aufweisen, wobei für das Traversieren:
der Servercomputer konfiguriert ist zum:
das Erstellen einer Bitmaske, wobei die Bitmaske Bits aufweist, die mit Kanten des Durchlaufwegs in der Ordnung, in der die Kanten in dem Durchlaufweg traversiert werden, assoziiert sind, wobei jedes Bit der Bitmaske für eine vorgegebene Kante einen Wert hat, der anzeigt, ob der Zielknoten der Kante dem Quellknoten der Kante mit Bezug auf die von der partiellen Ordnung repräsentierte Ordnung vorangeht oder nachfolgt;
Senden der Bitmaske an den Client-Computer;
und der Client-Computer konfiguriert ist zum Verwenden der Bitmaske, um die gecachten Teile zu traversieren.

9. System nach Anspruch 7, wobei der Servercomputer ferner konfiguriert ist zum
dem Client-Computer Mitteilen des verschlüsselten Datenelements, das dem Rotationspunkt entspricht;
wobei der Client-Computer ferner konfiguriert ist zum:
Entschlüsseln des mitgeteilten Datenelements durch den Client-Computer;
Identifizieren des entschlüsselten Datenelements in dem Cache als den vorgegebenen Knoten, der den Rotationspunkt bildet.

10. System nach einem der vorhergehenden Ansprüche 7 bis 9, wobei der Servercomputer ferner konfiguriert ist zum Bestimmen, ob ein Neuausgleichen des Baums unter Verwendung des erneuten Durchlaufs erforderlich ist.

11. System nach einem der vorhergehenden Ansprüche, wobei die partiell geordnete Menge einen AVL-Baum umfasst.

12. System nach Anspruch 1, wobei der erzeugte Cache-Eintrag ferner anzeigt, dass die Intervallgrenze eine untere oder eine obere Intervallgrenze ist.

13. System nach einem der vorhergehenden Ansprüche, wobei der Client-Computer ferner konfiguriert ist zum:
Erhalten einer Suchanforderung, wobei die Suchanforderung eine Anforderung für eine Präfixsuche nach einem Suchkriterium aufweist,
Bestimmen des Suchintervalls durch Umwandeln der Präfixsuche in ein entsprechendes Intervall, das das Suchkriterium als die untere Intervallgrenze und einen rechnerisch von dem Suchkriterium abgeleiteten Wert als die obere Intervallgrenze aufweist.

14. Servercomputer, der eine Datenbank aufweist, wobei die genannte Datenbank Datensätze speichert, wobei die Datenbank ferner eine Relation aufweist, wobei die Relation Datenelemente aufweist, wobei die Datenelemente mit einem ersten Verschlüsselungsverfahren in der Relation verschlüsselt sind, wobei die Datenelemente eine partiell geordnete Menge in der Relation bilden, wobei die partielle Ordnung mit Bezug auf die Datenelemente in nichtverschlüsselter Form gebildet ist, wobei eine referentielle Verbindung besteht, die jedes verschlüsselte Datenelement in der Relation einem jeweiligen Datensatz der Datensätze zuordnet, wobei die verschlüsselten Datenelemente mit Dateneinheiten einer linearen Ordnung in der Datenbank annotiert sind, wobei die lineare Ordnung der Ordnung entspricht, in der die verschlüsselten Datenelemente in der Relation mit Bezug auf die partiell geordnete Menge gespeichert sind, wobei das Computersystem ferner konfiguriert ist zum:
Erhalten einer Anforderung für Datensätze, deren assoziierte Datenelemente innerhalb eines Suchintervalls liegen, von einem Client-Computer, wobei die Anforderung ein verschlüsseltes Datenelement anzeigt, das eine erste bzw. eine zweite Intervallgrenze des Suchintervalls bildet;
für jedes verschlüsselte Datenelement, das die erste und die zweite Intervallgrenze bildet:
Bestimmen, ob ein Dateneintrag mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist,
im Fall, dass ein Cache-Eintrag nicht mit dem verschlüsselten Datenelement, das die Intervallgrenze bildet, assoziiert ist:
Empfangen einer Anforderung von dem Client-Computer, die die Dateneinheit der linearen Ordnung anzeigt, die an das jeweilige verschlüsselte Datenelement annotiert ist, das die Intervallgrenze bildet;
Generieren eines Cache-Eintrags, der das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde, und die jeweilige Dateneinheit aufweist, für das jeweilige verschlüsselte Datenelement, das als die Intervallgrenze bildend bestimmt wurde;
im Fall, dass ein Cache-Eintrag mit einem Datenelement, das die Intervallgrenze bildet, assoziiert ist, Abrufen der Dateneinheit, die mit der Intervallgrenze assoziiert ist, aus dem Cache;
wobei der Servercomputer ferner konfiguriert ist zum Identifizieren aller verschlüsselten Datenelemente, an die die Einheiten linearer Ordnung zwischen den Dateneinheiten annotiert sind, die mit den Datenelementen assoziiert sind, die die Intervallgrenzen bilden; und
Abrufen der angeforderten Datensätze über ihre referentiellen Verbindungen unter Verwendung der identifizierten Dateneinheiten.

Es folgen 5 Seiten Zeichnungen

Anhängende Zeichnungen

Fig. 1

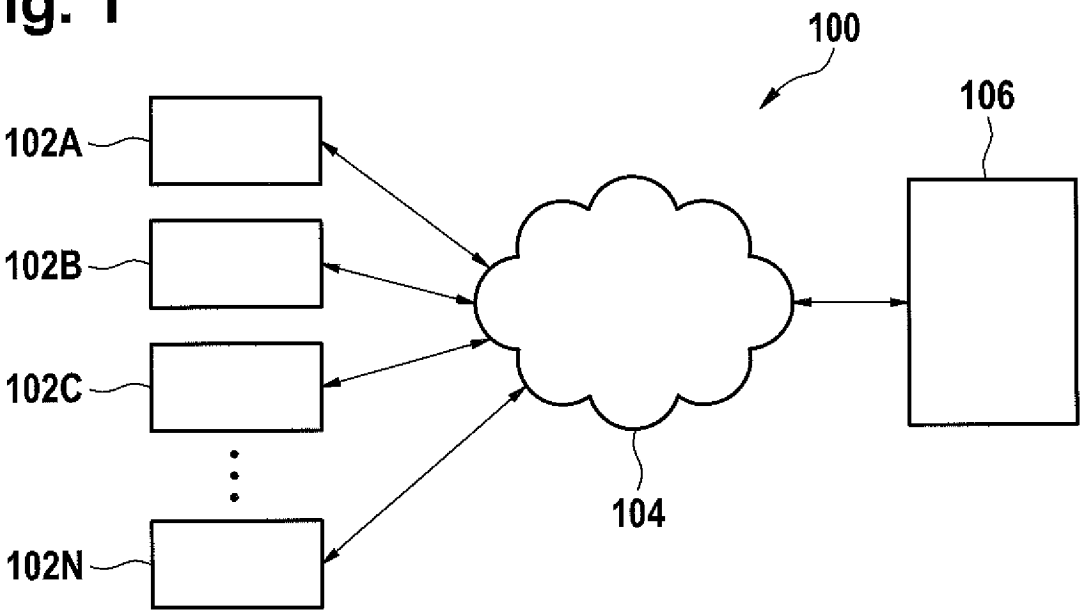


Fig. 2

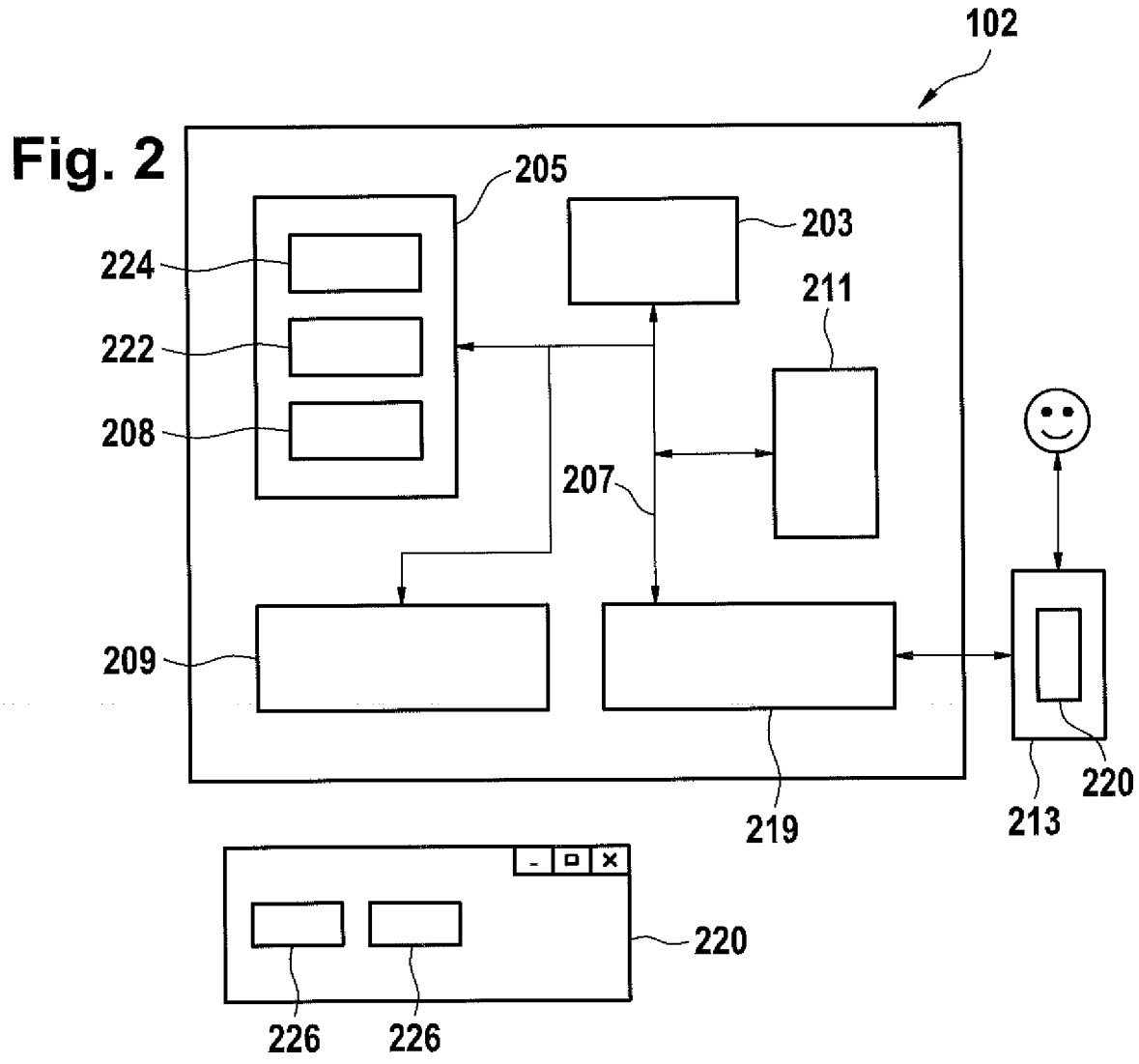
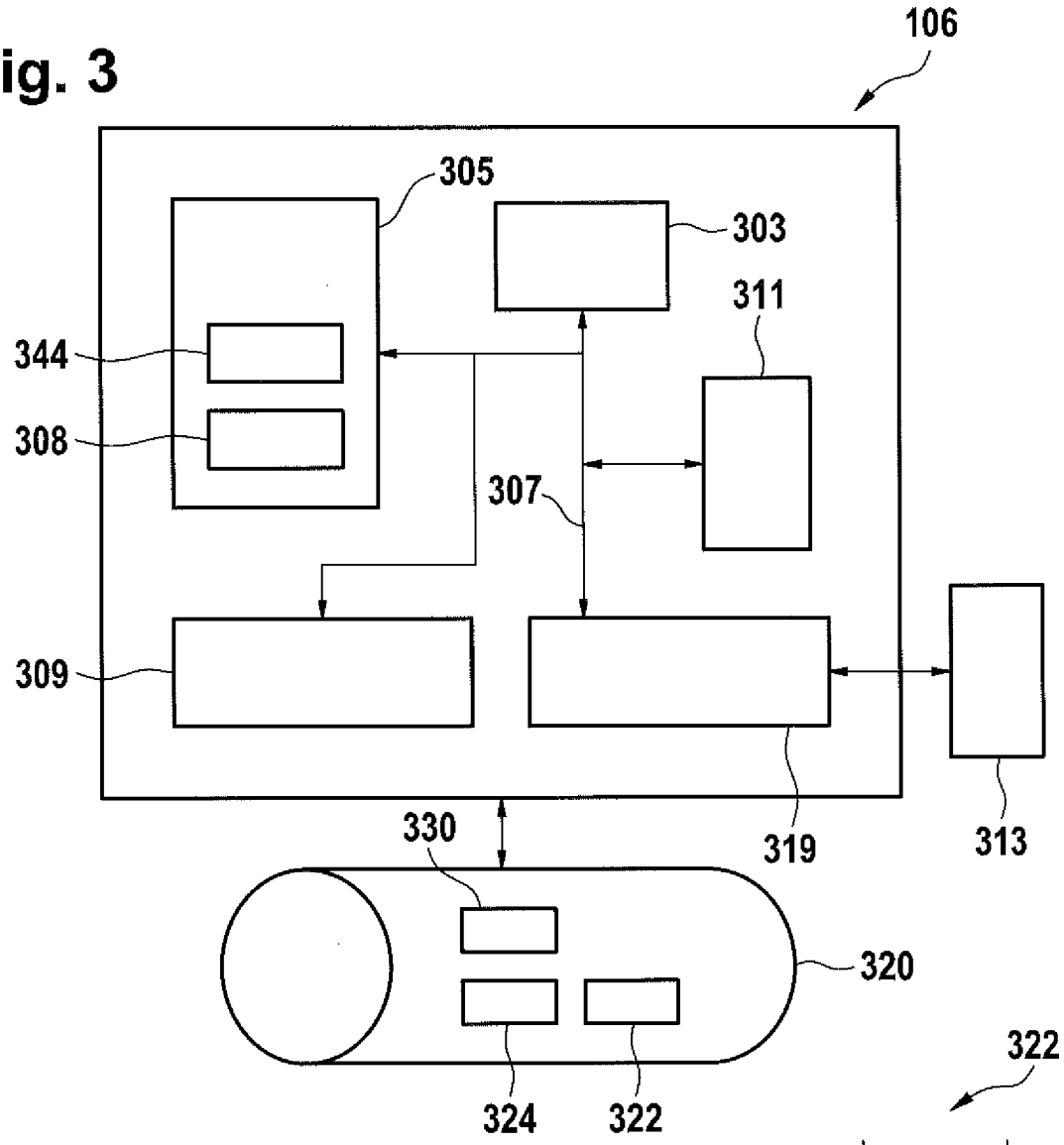
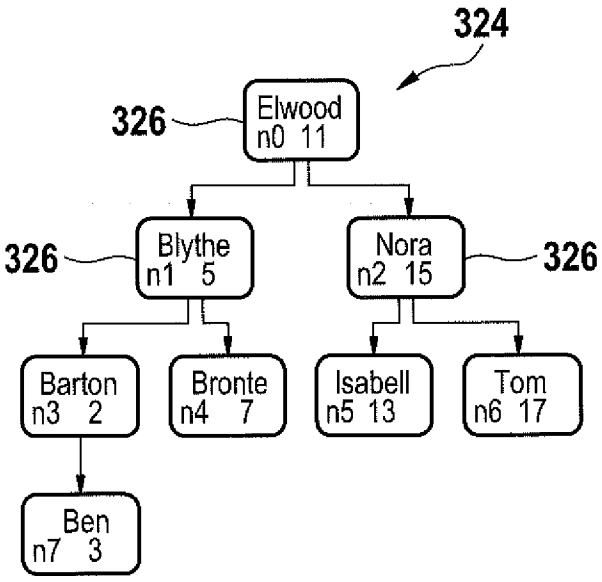


Fig. 3





322

Referenz	Daten
50	c_data0
51	c_data1
52	c_data2
53	c_data3
"	"
"	"
"	"
"	"

Fig. 4

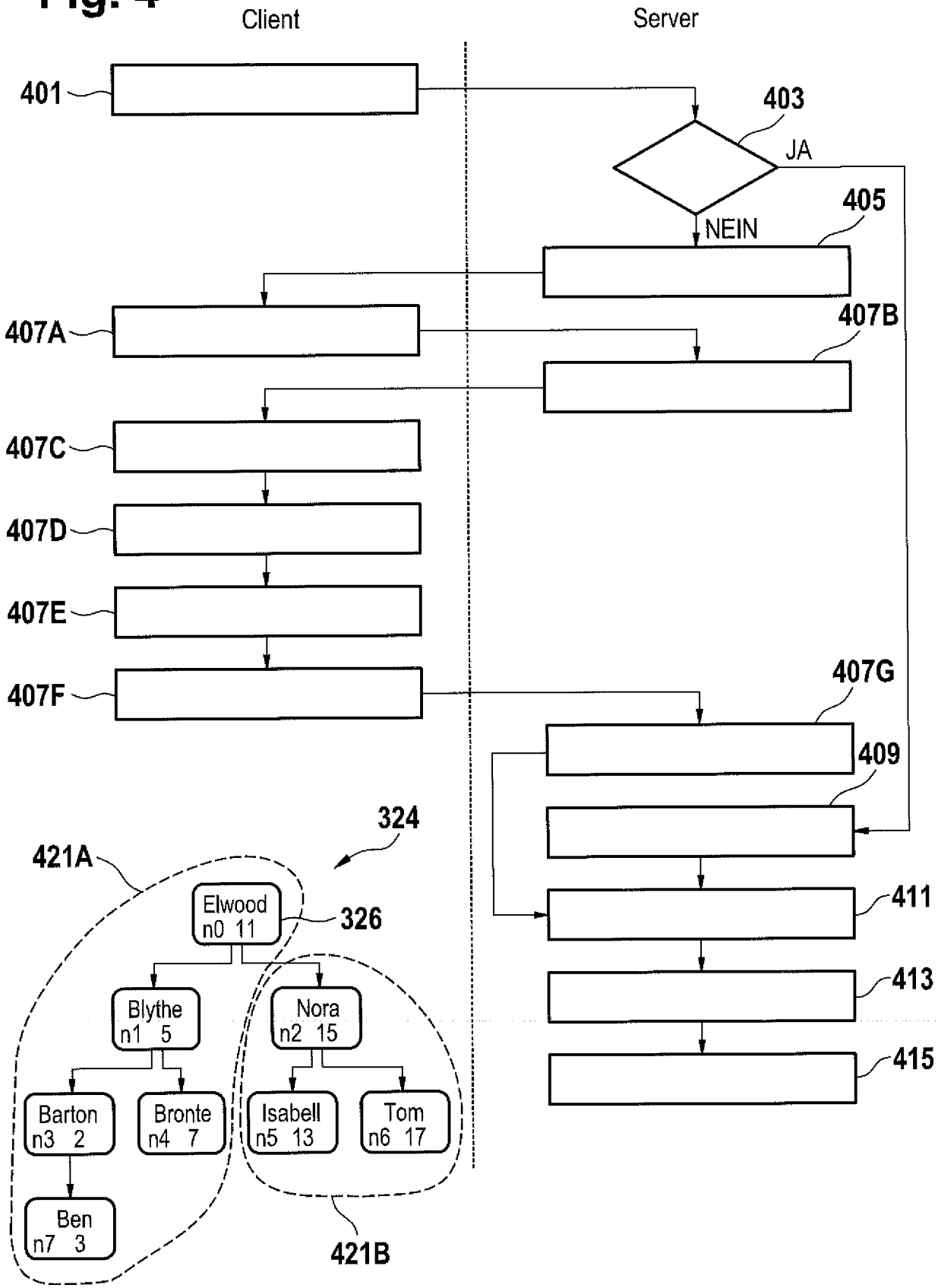

Client
Server
401
403
JA
NEIN
405
407A
407B
407C
407D
407E
407F
407G
409
411
413
415
324
421A
Elwood
n0 11
326
Blythe
n1 5
Nora
n2 15
Barton
n3 2
Bronte
n4 7
Isabell
n5 13
Tom
n6 17
Ben
n7 3
421B

Fig. 5

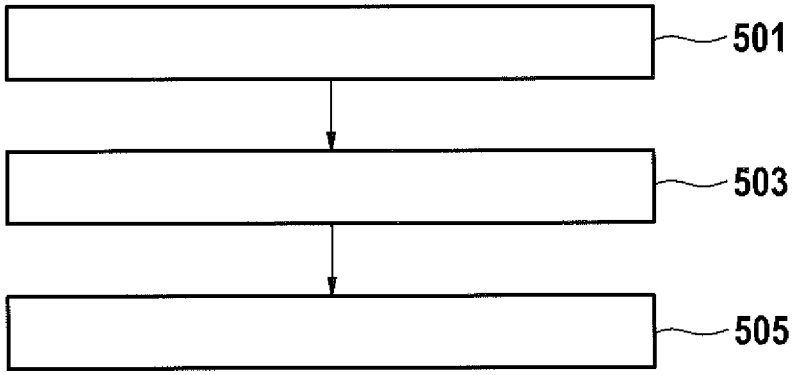

501
503
505

Fig. 6

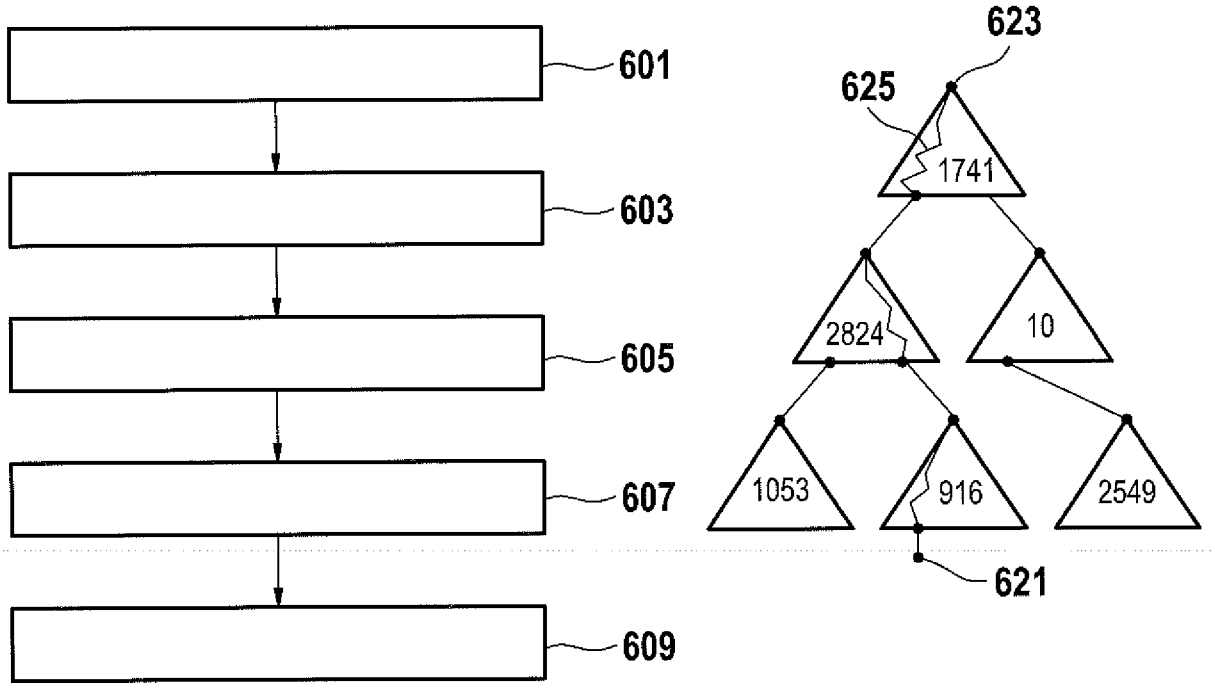

601
603
605
607
609
623
625
1741
2824
10
1053
916
2549
621

Fig. 7

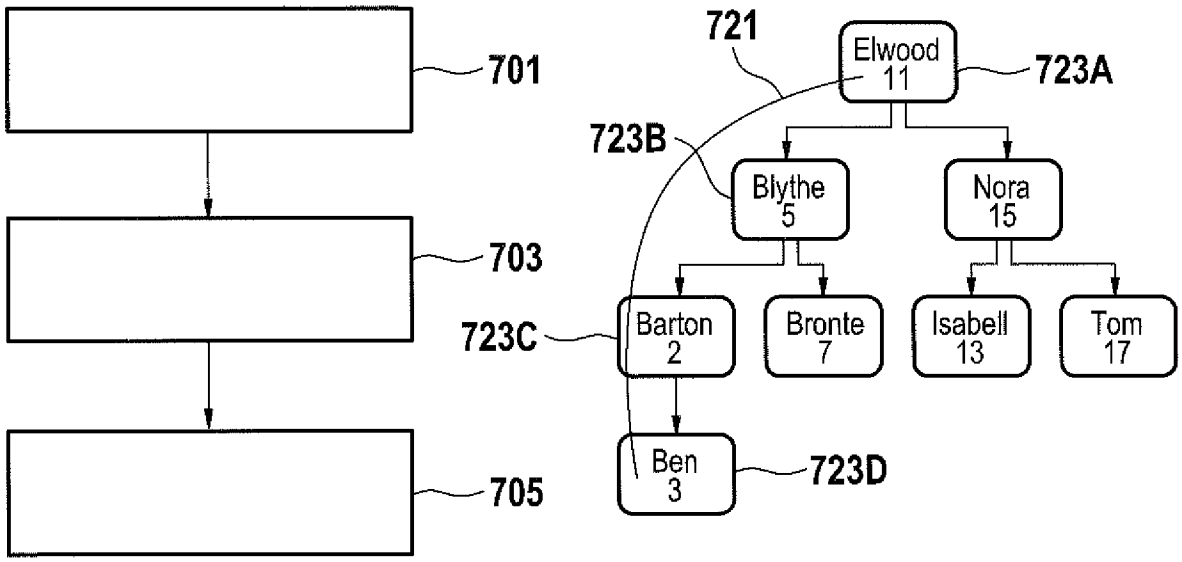
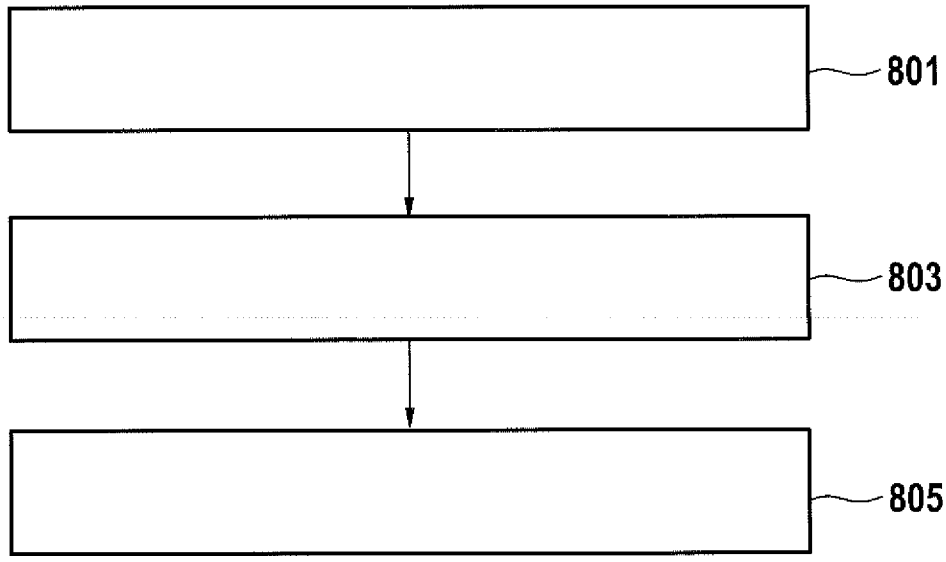

701
703
705
721
Elwood
11
723A
723B
Blythe
5
Nora
15
723C
Barton
2
Bronte
7
Isabell
13
Tom
17
Ben
3
723D

Fig. 8


801
803
805