(19) Deutsches Patent- und Markenamt

(10) **DE 11 2018 005 620 T5** 2020.07.23

(12) # Veröffentlichung

der internationalen Anmeldung mit der
(87) Veröffentlichungs-Nr.: **WO 2019/111188**
in der deutschen Übersetzung (Art. III § 8 Abs. 2 IntPatÜG)
(21) Deutsches Aktenzeichen: **11 2018 005 620.1**
(86) PCT-Aktenzeichen: **PCT/IB2018/059692**
(86) PCT-Anmeldetag: **06.12.2018**
(87) PCT-Veröffentlichungstag: **13.06.2019**
(43) Veröffentlichungstag der PCT Anmeldung in deutscher Übersetzung: **23.07.2020**

(51) Int Cl.: ***G06F 16/00*** (2019.01)

(30) Unionspriorität:
15/835,824 **08.12.2017** **US**

(71) Anmelder:
International Business Machines Corporation, Armonk, N.Y., US

(74) Vertreter:
Richardt Patentanwälte PartG mbB, 65185 Wiesbaden, DE

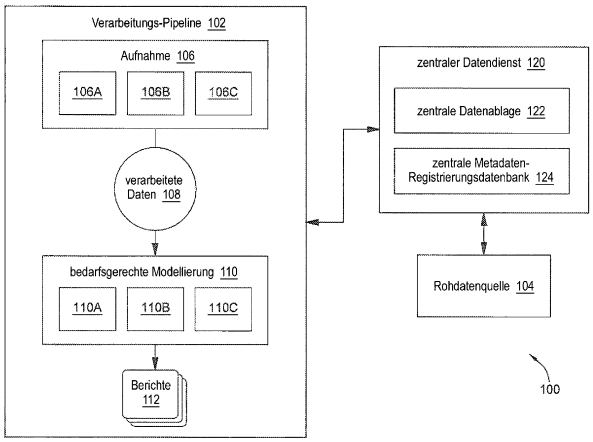
(72) Erfinder:
Yang, Jingwei, Cambridge, MA, US; Mahatma, Shilpa, Yorktown Heights, N.Y., US; Chandra, Rachita, Cambridge, MA, US; Tran, Kevin, Costa Mesa, CA, US; Wei, Dennis, Yorktown Heights, NY, US; Natesan Ramamurthy, Karthikeyan, Yorktown Heights, NY, US; Yuen-Reed, Gigi, Daytona Beach, FL, US

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen.

(54) Bezeichnung: **AUFTRAGSVERWALTUNG IN EINEM DATENVERARBEITUNGSSYSTEM**

(57) Zusammenfassung: Modularisierte Datenverarbeitungssysteme und Verfahren für deren Verwendung werden bereitgestellt. Bei einer Verarbeitung eines aktuellen Auftrags können Daten, die für einen zuvor verarbeiteten Auftrag erzeugt worden sind, in dem Umfang wiederverwendet werden, in dem die beiden Parameterkonfigurationen gemeinsam nutzen. In ähnlicher Weise können Ausgaben von Verarbeitungsmodulen, die während einer Verarbeitung des zuvor verarbeiteten Auftrags erzeugt worden sind, als Eingaben in Verarbeitungsmodule verwendet werden, die einen aktuellen Auftrag verarbeiten, wenn die beiden Aufträge einige Parameterkonfigurationen gemeinsam nutzen.



Beschreibung

HINTERGRUND

[0001] Ausführungsformen der Erfindung beziehen sich allgemein auf Datenmodellierung und maschinelles Lernen und im Besonderen auf eine Auftragsverwaltung in Datenverarbeitungs-Pipelines für Datenmodellierung und maschinelles Lernen.

[0002] Ein Datenverarbeitungssystem wie zum Beispiel ein System zur vorausschauenden Datenmodellierung oder ein System für maschinelles Lernen verarbeitet einen oder mehrere Aufträge. Ein Auftrag bezieht sich auf einen Satz von Daten und auf einen Satz von Parameterkonfigurationen zum Verarbeiten durch eine Anwendungsprogrammierschnittstelle (application programming interface, API); wobei die API einen Satz von Programmieranweisungen zum Verarbeiten des Satzes von Daten des Auftrags, wie durch die Parameterkonfigurationen des Auftrags geregelt, enthält, um ein Datenverarbeitungsziel (zum Beispiel ein Erzeugen eines Datenmodells) zu erreichen. Die Parameterkonfigurationen des Auftrags können in einer Parameterdatei definiert sein. Der Satz von Parameterkonfigurationen kann als Teil des Auftrags oder als diesem zugehörig betrachtet werden. Ein Beispiel für eine Konfiguration eines Auftrags sind die jeweiligen APIs, die zu seiner Ausführung erforderlich sind, die Datensätze, die die API verarbeiten sollte, und sonstige Verarbeitungskonfigurationen.

[0003] Datenwissenschaftler befassen sich häufig versuchsweise mit Datenverarbeitungssystemen, indem sie Aufträge mit unterschiedlichen Konfigurationen und Datensätzen verarbeiten. Auf Grundlage von Verarbeitungsergebnissen, die aus Unterschieden in Parameterkonfigurationen gewonnen werden, können Datenwissenschaftler Erkenntnisse über die Daten ableiten, die sie analysieren. Beispielsweise kann ein Datenwissenschaftler versuchsweise zahlreiche Aufträge mit unterschiedlichen Parameterkonfigurationen ausführen, um Sätze von Datenmerkmalen zu erkennen, die sich auf Verarbeitungsergebnisse auswirken. Einige Unterschiede in Parameterkonfigurationen können eine große Auswirkung auf Ergebnisse haben; einige können eine geringe Auswirkung aufweisen; andere haben möglicherweise keine Auswirkung. Mithilfe von beobachteten Varianten können Datenwissenschaftler vorausschauende Datenmodelle erstellen, und sie können maschinelle Lernprozesse für einen bestimmten Zweck steuern.

KURZDARSTELLUNG

[0004] Ausführungsformen der Erfindung sehen Verfahren, Computerprogrammprodukte und Systeme zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform vor. Gemäß einem Aspekt der Erfindung empfängt die Datenverarbeitungsplattform einen ersten Auftrag zur Verarbeitung. Der erste Auftrag weist einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform auf. Die Datenverarbeitungsplattform führt zumindest einen Abschnitt des ersten Auftrags über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline aus. Die Ausführung geht mit einem Verwenden zumindest eines Daten-Shards einher, wobei es sich bei einem Daten-Shard um eine Partition von Daten aus einer Datenbank handelt, die während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und geht des Weiteren mit einem Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule einher. Die Ausgabe wird während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt.

[0005] Gemäß einem Aspekt der Erfindung werden ein oder mehrere Abschnitte des zweiten Auftrags vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet.

[0006] Gemäß einem Aspekt der Erfindung enthalten die Verarbeitungsmodule eine oder mehrere Anwendungsprogrammierschnittstellen (APIs).

[0007] Gemäß einem Aspekt der Erfindung sind Parameterkonfigurationen von Aufträgen, die durch das Datenverarbeitungssystem verarbeitet werden, in jeweiligen Parameterdateien definiert, die den Aufträgen zugehörig sind.

[0008] Gemäß einem Aspekt der Erfindung werden Daten-Shards, die den zumindest einen Daten-Shard enthalten, der während der Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, durch einen zentralen Datendienst verwaltet.

[0009] Gemäß einem Aspekt der Erfindung ist der zentrale Datendienst virtualisiert.

[0010] Gemäß einem Aspekt der Erfindung enthält der zentrale Datendienst eine zentrale Datenablage und eine zentrale Metadatenablage.

[0011] Gemäß einem Aspekt der Erfindung enthält die zentrale Metadatenablage einen Satz von Metadatendateien für Daten-Shards (Partitionen), die in der zentralen Datenablage gespeichert sind, und enthält des Weiteren einen zusätzlichen Satz von Metadatendateien für Ausgaben von Verarbeitungsmodulen der Verarbeitungs-Pipeline.

[0012] Gemäß einem Aspekt der Erfindung ruft das Datenverarbeitungssystem den zumindest einen Daten-Shard, der während der Ausführung des Abschnitts des zweiten Auftrags erzeugt wird, von einer Position ab, die in einer Metadatendatei für den Daten-Shard angegeben wird, und ruft jegliche Daten, die nicht durch einen bekannten Daten-Shard definiert sind, aus einer Rohdatenquelle ab.

[0013] Gemäß einem Aspekt der Erfindung wird das Abrufen des zumindest einen Daten-Shards und das Abrufen der Daten, die nicht durch einen bekannten Daten-Shard definiert sind, auf Grundlage einer Benutzerauswahl über eine graphische Benutzeroberfläche (graphical user interface, GUI) ausgelöst.

[0014] Gemäß einem Aspekt der Erfindung empfängt ein Verfahren zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform einen ersten Auftrag zur Verarbeitung. Der erste Auftrag weist einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform auf. Die Verarbeitungsmodule enthalten eine oder mehrere Anwendungsprogrammierschnittstellen (APIs). Das Verfahren führt zumindest einen Abschnitt des ersten Auftrags über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline aus. Das Ausführen bezieht sich auf ein Verwenden zumindest eines Daten-Shards, der während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und auf ein Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule. Die Ausgabe wird während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt, und ein oder mehrere Abschnitte des zweiten Auftrags werden vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet. Daten-Shards, die den zumindest einen Daten-Shard enthalten, der während der Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, werden durch einen zentralen Datendienst verwaltet, der eine zentrale Datenablage und eine zentrale Metadatenablage aufweist. Das Ausführen des zumindest einen Abschnitts des ersten Auftrags enthält des Weiteren ein Abrufen des zumindest einen Daten-Shards, der während der Ausführung des Abschnitts des zweiten Auftrags erzeugt wird, aus einer Position, die in einer Metadatendatei für den Daten-Shard angegeben wird, und ein Abrufen jeglicher Daten, die nicht durch einen bekannten Daten-Shard definiert sind, aus einer Rohdatenquelle.

Figurenliste

Fig. 1 ist ein Blockschaubild eines Datenverarbeitungssystems **100** gemäß einer Ausführungsform der Erfindung.

Fig. 2 stellt ein Beispiel für eine Organisation der inkrementellen Daten, die durch das Verarbeitungssystem **100** (**Fig. 1**) verwendet werden, gemäß einer Ausführungsform der Erfindung bereit.

Fig. 3 stellt ein Beispiel für eine Metadatendatei für einen der in **Fig. 2** dargestellten Daten-Shards gemäß einer Ausführungsform der Erfindung bereit.

Fig. 4 stellt ein Beispiel für ein Verfahren **400** für bestimmte Funktionen des Verarbeitungssystems **100** (**Fig. 1**) gemäß einer Ausführungsform der Erfindung bereit.

Fig. 5 stellt ein Beispiel für ein Verfahren **500** zum Wiederverwenden von Ausgaben von API-Komponenten über Aufträge hinweg gemäß einer Ausführungsform der Erfindung bereit.

Fig. 6 stellt eine beispielhafte graphische Benutzeroberfläche (GUI) **600** zum Wiederverwenden von Daten-Shards und API-Ausgaben, die durch Verarbeiten eines vorhergehenden Auftrags erzeugt worden sind, gemäß einer Ausführungsform der Erfindung bereit.

Fig. 7 stellt eine beispielhafte GUI **700** zum Überwachen von Auftragsstatus gemäß einer Ausführungsform der Erfindung bereit.

Fig. 8 stellt eine veranschaulichende Datenverarbeitungseinheit zum Implementieren verschiedener Komponenten des Datenverarbeitungssystems von **Fig. 1** gemäß einer Ausführungsform der Erfindung bereit.

AUSFÜHRLICHE BESCHREIBUNG

[0015] Datenverarbeitungssysteme nach dem Stand der Technik wie zum Beispiel ein System zur vorausschauenden Datenmodellierung oder ein System für maschinelles Lernen weisen aus mehreren Gründen Beschränkungen auf. Zunächst sind solche Systeme ineffizient beim Verarbeiten von Streaming- und inkrementellen Daten. Streaming- und inkrementelle Daten beziehen sich auf einen Prozess eines Datenzuflusses, bei dem nicht sämtliche der zu verarbeitenden Daten unmittelbar verfügbar sind. Dies kann zum Beispiel der Fall sein, wo Daten einigermaßen regelmäßig erfasst werden, und es ist wünschenswert, neu verfügbare Daten zu verarbeiten, um vorhergehende Verarbeitungsergebnisse zu verbessern. Mit anderen Worten, die typischerweise durchgeführte Verarbeitung muss inkrementell aktualisiert werden, wenn neue Daten in das System eingehen. Nach dem Stand der Technik laden solche Systeme typischerweise Daten aus verschiedenen Datenquellen oder „rufen“ diese „erneut ab“ und wiederholen vergangene Verarbeitungsschritte an einigen derselben Daten. Das erneute Abrufen der Daten erhöht die Ressourcen-Kosten und verlangsamt Verarbeitungszeiten. Diese Nachteile wirken sich ungünstig auf Betriebsabläufe einer Online-Transaktionsverarbeitung (online transaction processing, OLTP) aus.

[0016] Zum Zweiten sind solche Systeme nicht modularisiert und sind daher nicht dazu ausgestattet, Teilergebnisse von Verarbeitungsmodulen wiederzuverwenden. Mit anderen Worten, das Verarbeitungskonzept bei den Systemen nach dem Stand der Technik besteht darin, sämtliche Verarbeitungsoperationen jedes Mal zu wiederholen, wenn neue Daten eingehen, ohne Ergebnisse einer vergangenen Verarbeitung wiederzuverwenden.

[0017] Zum Dritten trainieren solche Systeme nach dem Stand der Technik unnötigerweise vorausschauende Lerner erneut, wo solche Lerner verwendet werden. Das Training umfasst kein inkrementelles Training, sondern stützt sich stattdessen auf ein erneutes Trainieren von vorausschauenden Lernermodellen mithilfe vollständiger Datensätze (alten Datensätzen und inkrementell neuen Datensätzen).

[0018] Daher besteht ein Bedarf an einer Lösung für eine Verwaltung von modularisierten Datenverarbeitungs-Pipelines, die inkrementelle Daten effizient verarbeitet.

[0019] Die obigen Beschränkungen nach dem Stand der Technik sind Beispiele und sollen keine erschöpfende Liste von Beschränkungen nach dem Stand der Technik sein. Des Weiteren ist das Behandeln eines bestimmten Mangels des Standes der Technik kein notwendiges Merkmal einer bestimmten Ausführungsform der Erfindung. Die Erfindung des Anmelders wird durch die Ansprüche definiert.

[0020] Aspekte der offenbarten Erfindung sehen eine modularisierte Verarbeitung von Aufträgen in einem Datenverarbeitungssystem vor. Um eine Erörterung von Ausführungsformen der beanspruchten Erfindung zu erleichtern, wird zunächst eine Übersicht über die verschiedenen Figuren bereitgestellt. Anschließend werden jede Figur oder Sätze von Figuren ausführlicher erörtert.

[0021] Beispielsweise stellt **Fig. 1** eine Übersicht über ein Datenverarbeitungssystem **100** bereit. Das Datenverarbeitungssystem **100** empfängt im Allgemeinen Aufträge zur Verarbeitung. Im Gegensatz zu Systemen nach dem Stand der Technik ist das Datenverarbeitungssystem **100** modularisiert und ermöglicht allgemein eine Wiederverwendung von inkrementell erneut abgerufenen Daten (d.h., Daten, die in Inkrementen über vorhergehende Zeitinkremente abgerufen worden sind) und eine Wiederverwendung von Ausgaben von Pipeline-Modulen.

[0022] **Fig. 2** stellt ein Beispiel für eine Organisation der inkrementellen Daten, die durch das Verarbeitungssystem **100** (**Fig. 1**) verwendet werden, bereit. Die Daten in dem dargestellten Beispiel sind nach Datum zu Shards organisiert.

[0023] **Fig. 3** stellt ein Beispiel für eine Metadatendatei für einen der in **Fig. 2** dargestellten Daten-Shards bereit.

[0024] **Fig. 4** stellt ein Beispiel für ein Verfahren **400** für bestimmte Funktionen des Verarbeitungssystems **100** (**Fig. 1**) bereit.

[0025] **Fig. 5** stellt ein Beispiel für ein Verfahren **500** zum Wiederverwenden von Ausgaben von API-Komponenten über Aufträge hinweg bereit.

[0026] **Fig. 6** stellt eine beispielhafte graphische Benutzeroberfläche (GUI) **600** zum Wiederverwenden von Daten-Shards und API-Ausgaben, die durch Verarbeiten eines vorhergehenden Auftrags erzeugt worden sind, bereit.

[0027] **Fig. 7** stellt eine beispielhafte GUI **700** zum Überwachen von Auftragsstatus bereit.

[0028] **Fig. 8** stellt eine veranschaulichende Datenverarbeitungseinheit zum Implementieren verschiedener Komponenten des Datenverarbeitungssystems von **Fig. 1** bereit.

[0029] Unter Bezugnahme auf **Fig. 1** handelt es sich bei dem Datenverarbeitungssystem **100** um ein Datenverarbeitungssystem zum Verarbeiten von Daten gemäß einer Ausführungsform der Erfindung. Bei dem Datenverarbeitungssystem **100** kann es sich um eine einzelne physische Einheit oder um eine Zusammenstellung von physischen und virtuellen Datenverarbeitungs-Ressourcen handeln, wie ausführlicher im Folgenden in Verbindung mit **Fig. 13** beschrieben.

[0030] Das Datenverarbeitungssystem **100** enthält allgemein eine Datenverarbeitungs-Pipeline **102**, eine Rohdatenbank **104** (die auch als „ursprüngliche Datenquelle“ bezeichnet wird), einen zentralen Datendienst **120** und einen oder mehrere Clients **150** (Clients stellen Benutzereinheiten dar, die durch Datenwissenschaftler verwendet oder betrieben werden, um sich mit verschiedenen Aspekten des Datenverarbeitungssystems **100** zu befassen). Dies alles wird im Folgenden ausführlicher beschrieben.

[0031] Bei der Datenverarbeitungs-Pipeline **102** handelt es sich um eine Verarbeitungs-Pipeline zur Ausführung eines oder mehrerer Aufträge gemäß einer Ausführungsform der Erfindung. Bei der dargestellten Ausführungsform enthält die Datenverarbeitungs-Pipeline **102** zwei Teilverarbeitungs-Pipelines, eine Aufnahme-Pipeline **106** und eine Pipeline **110** zur bedarfsgerechten Modellierung.

[0032] Die Aufnahme-Pipeline **106** empfängt im Allgemeinen Rohdaten und Metadaten von dem zentralen Datendienst **120**, verarbeitet die Daten durch eine oder mehrere Verarbeitungsstufen oder -module und gibt die Ergebnisse als verarbeitete Daten **108** aus. Zu den Modulen in der Aufnahme-Pipeline **106** zählen ein Datenauswahlmodul **106A** (zum Auswählen der zu verarbeitenden Daten), ein Validierungsmodul **106B** (zum Überprüfen der Datenintegrität), ein Zusammenfassungs- und Merkmalextraktionsmodul **106C** (das Merkmale in den Datensätzen erkennt) und ein Modul **106D** zur Umwandlung von dünnbesetzten Matrizen (zum Verwalten von dünnbesetzten Matrizen).

[0033] Die Pipeline **110** zur bedarfsgerechten Modellierung empfängt im Allgemeinen verarbeitete Daten **108** von der Aufnahme-Pipeline **106** und verarbeitet die Daten durch eine oder mehrere Verarbeitungsstufen oder -module und gibt einen oder mehrere Berichte **112** aus. Zu den Modulen in der Pipeline **110** zur bedarfsgerechten Modellierung zählen ein Erzeugungsmodul **110A** zum Erzeugen von Trainings- und Testdatensätzen aus den verarbeiteten Daten **108**, ein Merkmalvorauswahlmodul **110B**, das Merkmale aus den Trainings- und Testdatensätzen vorab auswählt, ein Modelltrainingsmodul **110C**, das Datenmodelle auf Grundlage der vorab ausgewählten Merkmale trainiert, und ein Berichterstellungsmodul **110D**, das das trainierte Modell auf einen bestimmten Datensatz, der analysiert wird, anwendet, um einen oder mehrere Berichte **112** zu erzeugen.

[0034] Weiterhin unter Bezugnahme auf **Fig. 1** werden gemäß der dargestellten Ausführungsform Datenquellen in dem Datenverarbeitungssystem **100** logisch oder physisch in die Rohdatenquelle **104** (die auch als „ursprüngliche Datenquelle“ bezeichnet wird) und in den zentralen Datendienst **120** aufgeteilt. Bei der Rohdatenquelle **104** handelt es sich um eine Datenbank, die Daten speichert, die ein Datenwissenschaftler (über einen Client) speichern, überwachen, untersuchen, prüfen oder auf andere Weise verarbeiten möchte. Die in der Rohdatenquelle **104** gespeicherten Daten liegen in einer einigermaßen unverarbeiteten Form vor (d.h., sie werden nicht zwingend für einen bestimmten Auftrag oder ein bestimmtes Verarbeitungsmodul gepflegt); jedoch können verarbeitete Formen einiger oder sämtlicher dieser Rohdaten in einer Datenbank des zentralen Datendienstes **120** gespeichert sein.

[0035] Bei dem zentralen Datendienst **120** handelt es sich um eine physische oder eine virtualisierte Schicht (zum Beispiel eine Datenbank), die in die Rohdatenquelle **104** eingebaut ist. Im Gegensatz zu dem Stand der Technik setzen Clients den zentralen Datendienst **120** für ihre Datenanforderungen bei Operationen der Auftragsbearbeitung ein, statt die Rohdatenquelle **104** direkt einzusetzen. Diese Organisation und Struktur trägt zu der neuartigen Fähigkeit zum Wiederverwenden von vorverarbeiteten Daten in einer modularisierten Verarbeitungs-Pipeline wie zum Beispiel denjenigen in dem Datenverarbeitungssystem **100** bei.

[0036] Die Funktionen des zentralen Datendienstes **120** werden durch eine zentrale Datenablage **122** und eine zentrale Metadaten-Registrierungsdatenbank **124** ermöglicht. Bei der zentralen Datenablage **122** handelt es sich um ein Funktionselement, das einen Zugriff auf zuvor verarbeitete Daten (d.h., Rohdaten von der Rohdatenquelle **104**, nachdem sie durch ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline **102** verarbeitet worden sind) erleichtert. In der zentralen Datenablage **122** gespeicherte Daten sind zu Daten-Shards organisiert. Bei der zentralen Datenablage **122** kann es sich um eine physische oder virtuelle Datenbank, ein Hadoop Distributed File System (HDFS), ein beliebiges sonstiges Dateisystem oder gar um eine Daten-Shard-Positionsregistrierungsdatenbank handeln.

[0037] Die Daten-Shards können auf vielfältige Weise organisiert sein, zum Beispiel nach einem Datums- und Zeitwert (etwa entsprechend dem Datum und der Zeit, an denen der Daten-Shard erzeugt wird). Der Einfachheit halber wird in der dargestellten Ausführungsform ein bestimmter Daten-Shard mit einer Zeichenfolgen-ID mit dem Format „JJJJ - MM“, d.h., einer vierstelligen Zahl, die das Jahr darstellt, und einer zweistelligen Zahl, die den Monat darstellt, in denen der bestimmte Daten-Shard erzeugt wird, gekennzeichnet.

[0038] Im Allgemeinen „ruft“ der zentrale Datendienst **120** Rohdaten nach Bedarf, d.h., wenn die Daten nicht als Daten-Shard in der zentralen Datenablage **124** vorhanden sind, aus der Rohdatenquelle **104** „ab“. Diese Funktion ermöglicht dem Datenverarbeitungssystem **100**, ein „erneutes Abrufen“ und ein erneutes Verarbeiten von Rohdaten zu vermeiden, die bereits durch Module der Verarbeitungs-Pipeline abgerufen und verarbeitet worden sind. Diese und sonstige Merkmale von Ausführungsformen der offenbarten Erfindung werden in der folgenden Erörterung weiter verdeutlicht.

[0039] Weiterhin unter Bezugnahme auf **Fig. 1** ermöglicht die offenbarte Konfiguration des Datenverarbeitungssystems **100** und seiner verschiedenen Bestandteile im Gegensatz zum Stand der Technik zwei Typen einer Wiederverwendung von Daten: eine Wiederverwendung von abgerufenen inkrementellen Daten und eine Wiederverwendung von Ausgaben von Modulen. Bevor auf Einzelheiten in Verbindung mit veranschaulichenden Beispielen in den verschiedenen sonstigen Figuren eingegangen wird, wird hier eine Kurzdarstellung bereitgestellt.

[0040] Im Hinblick auf die Wiederverwendung von inkrementell abgerufenen Daten ruft das Datenverarbeitungssystem **100** Rohdaten nach Bedarf durch Module der Verarbeitungs-Pipeline **102** ab, organisiert die angerufenen Daten zu Daten-Shards und erzeugt Metadatendateien für die Daten-Shards. Wenn ein Modul der Verarbeitungs-Pipeline **102** einige Daten benötigt, überprüft das Datenverarbeitungssystem **100** (z.B. überprüft der zentrale Datendienst **120** die Metadatendateien für die verschiedenen Daten-Shards), ob die Daten bereits als Daten-Shard vorhanden sind. Statt die Daten erneut abzurufen, stattet das Datenverarbeitungssystem **100** das Modul einfach mit einem Zeiger (z.B. einem Abrufpfad) auf die relevanten Daten-Shards aus. Jegliche Daten, die nicht bereits in einem Daten-Shard in der zentralen Datenablage **122** vorhanden sind, werden aus der Rohdatenquelle **104** abgerufen (entsprechende Daten-Shards und Metadatendateien werden anschließend für die abgerufenen Daten erzeugt).

[0041] Im Hinblick auf die Wiederverwendung von und die Wiederverwendung von Ausgaben von Modulen; werden zwei Aufträge betrachtet: Auftrag A und Auftrag B, die zur Verarbeitung mithilfe von Modulen oder APIs 1 bis 7 an das Datenverarbeitungssystem **100** gesendet worden sind. In diesem Beispiel werde angenommen, dass die beiden Aufträge dieselben Parameterkonfigurationen für die APIs 1 bis 4, jedoch unterschiedliche Parameterkonfigurationen für die übrigen APIs aufweisen. Darüber hinaus werde angenommen, dass der Auftrag A vollständig verarbeitet worden ist. Der Client (ein Datenwissenschaftler oder allgemeiner ein Benutzer) kann das Datenverarbeitungssystem **100** dazu einsetzen, den Auftrag B durch Wiederverwenden der Ausgaben der APIs 1 bis 4, wie sie während der Verarbeitung des Auftrags A erzeugt worden sind, zu verarbeiten; und eine neue Verarbeitung des Auftrags B mit der API 5 zu beginnen. Das Datenverarbeitungssystem **100** kann API- und Auftragstopologien und -abhängigkeiten erkennen, um zu ermitteln, welche Module welche Ausgaben von zuvor verarbeiteten Aufträgen und Modulen verwenden können. Bei diesen Ermittlungen kann das Datenverarbeitungssystem **100** die Metadatendateien für jeden Daten-Shard verwenden, um zu ermitteln, welcher Auftrag oder welches Modul sie erzeugt hat und welche sonstigen Aufträge oder Module diese verwenden können.

[0042] Unter Bezugnahme auf **Fig. 2** wird gemäß einer Ausführungsform ein Beispiel 200 für eine Organisation der inkrementellen Daten (d.h., der Rohdaten, die inkrementell abgerufen und durch verschiedene Verarbeitungs-Pipelines **102** verwendet werden) bereitgestellt, die durch das Verarbeitungssystem **100** verwendet worden sind (**Fig. 1**). Die Daten in dem dargestellten Beispiel sind nach Datum zu Shards (Fragmenten oder Partitionen) geordnet. Die Daten werden als Liste von Packlistendateien („.pkl“) angezeigt, wobei jeder Datei-

name einen Modulnamen (z.B. „dfCode“, „dfCost“ und „dfpmpm“) gefolgt von einer vierstelligen Kennung für das Jahr und einer zweistelligen Kennung für den Monat enthält.

[0043] Unter Bezugnahme auf **Fig. 3** wird ein Beispiel für eine Metadatendatei **300** für einen der in **Fig. 2** dargestellten Daten-Shards; im Besonderen für den Daten-Shard mit der Bezeichnung „dfpmpm2013-06.pkl“ bereitgestellt. Bei der Metadatendatei in **Fig. 3** handelt es sich lediglich um ein Beispiel dafür, wie einige Daten über einen bestimmten Daten-Shard verfolgt werden können. Bei TABELLE 1 handelt es sich um eine kommentierte Form der beispielhaften Metadatendatei von **Fig. 3**. Die Metadatendatei für diesen Daten-Shard beschreibt den Daten-Shard, wann er erstellt wurde, wo er gespeichert ist (z.B. über einen Abrufpfad; der Daten-Shard muss nicht von dort, wo er ursprünglich gespeichert wurde, erneut abgerufen oder kopiert werden), den Typ der Verarbeitungs-Pipeline, die ihn erzeugt hat, und das Format der Daten, die der Daten-Shard enthält. Diese Daten können durch Ausführungsformen der Erfindung dazu verwendet werden zu ermitteln, ob die Daten des Daten-Shards für eine Wiederverwendung durch ein weiteres Modul einer Verarbeitungs-Pipeline geeignet sind.

TABELLE 1

Inhalt der Metadatendatei	Anmerkungen
Ausgabe-API: 2	die jeweilige API oder das Modul, die/das die Daten in diesem Daten-Shard erzeugt hat
Datenbereich: 2013-06	der Datenbereich der Rohdaten, der zum Erzeugen dieses Daten-Shards verwendet worden ist
Erstellungszeit: 2017-08-09 10:18:16.598385253	
Pipeline-Typ: Inkrementelles Training	der jeweilige Typ der Verarbeitungs-Pipeline, die diesen Daten-Shard erzeugt hat
Abrufpfad: hdfs://localhost:9010/2017-08-09/api2-2011-10-datashard	der/das physische oder virtualisierte Server/Verzeichnis, wo der Daten-Shard verfügbar ist
Typ: Datei	
Datenbank:	Daten, die den Typ der Datenbank, zu der der Shard gehört, und das Schema beschreibt, das sie verwendet; diese Daten können dazu verwendet werden zu ermitteln, ob der Daten-Shard für sonstige Verarbeitungs-Pipelines oder ihre Module wiederverwendbar ist
-Host: „192.168.1.1“	
-Datenbank_Name: „WATSON“	
-Schema: „MARKETSCAN“	
Tabellen:	Daten, die die in dem Daten-Shard gespeicherten Daten beschreiben; in diesem Beispiel enthalten die Daten in diesem Daten-Shard Versicherungsdaten wie etwa Versicherungsansprüche, Versicherungsmitgliedschaften usw.
-Ansprüche: „Anspruchs Tabelle_1m“	
-Mitglied: „Mitgliedschafts Tabelle_1m“	
Filter:	Daten, die Filter beschreiben, die zum Erzeugen der Daten verwendet werden
-Mitglied: „Mitgliedschafts Tabelle_1m“	
-Spalte: pharmben_ja'	
-Operation: ‚=='	
-Wert: 1	
-Tabelle: ‚Mitgliedschafts Tabelle_1m'	

Inhalt der Metadatendatei	Anmerkungen
-Spalte: ‚Prod_Typ' -Operation: ‚notin' -Wert: [‚HMO','POS mit Zahlung']	

[0044] **Fig. 4** stellt ein Beispiel für ein Verfahren **400** zum Abrufen von Daten aus der Rohdatenquelle **104** und dem zentralen Datendienst **120** des Datenverarbeitungssystems **100** (**Fig. 1**) gemäß einer Ausführungsform der Erfindung bereit. Die verschiedenen Funktionen oder Schritte des Verfahrens **400** werden durch Programmieranweisungen ermöglicht, die durch einen oder mehrere Prozessoren des Datenverarbeitungssystems **100** ausführbar sind. In der folgenden Erörterung kann jeder Schritt als durch eine bestimmte Komponente des Datenverarbeitungssystems **100** ausführbar bezeichnet werden; dies dient jedoch lediglich der Veranschaulichung. Die Schritte oder Funktionen können auf andere Weise im Wesentlichen genauso ausgeführt werden, um im Wesentlichen dieselben Ergebnisse zu erzielen, ohne vom Wesensgehalt oder Umfang der Erfindung abzuweichen.

[0045] Weiterhin unter Bezugnahme auf **Fig. 4** beginnt (Schritt **402**) die Verarbeitungs-Pipeline **102** eine Ausführung eines Auftrags in einer bestimmten API (d.h., einem ihrer Module). Das Beginnen der Verarbeitung des Auftrags kann ein Erkennen einer Auftragsanforderung enthalten.

[0046] Die Verarbeitungs-Pipeline **102** empfängt die Auftragsanforderung und ihre Parameterdatei und liest (Schritt **404**) die Parameterdatei, um die Vorgaben des Auftrags zu ermitteln.

[0047] Die Verarbeitungs-Pipeline **102** erzeugt (Schritt **406**) eine Liste von Daten-Shards, deren Verwendung bei der Verarbeitung des Auftrags erwartet wird. Das Erzeugen der Liste von Daten-Shards, deren Verwendung erwartet wird, kann im Austausch mit dem zentralen Datendienst **120** durchgeführt werden. Beispielsweise kann die Verarbeitungs-Pipeline **102** die Vorgaben des Auftrags an den zentralen Datendienst **120** übertragen, die die Daten angeben, die die Verarbeitung des Auftrags für die jeweilige API erfordert. Der zentrale Datendienst **120** durchsucht die zentrale Metadaten-Registrierungsdatenbank **124** mithilfe der Parameterkonfigurationen des Auftrags. Die Suche ergibt möglicherweise keine Ergebnisse, oder die Suche ergibt eine oder mehrere passende Metadatendateien (d.h., Metadatendateien, die Daten-Shards entsprechen, deren Daten zum Verarbeiten des aktuell anstehenden Auftrags wiederverwendet werden können). Wenn die Suche keine Ergebnisse ergibt, werden die entsprechenden Daten aus der Rohdatenquelle **104** abgerufen. Wenn die Suche jedoch einige Ergebnisse ergibt, werden die erforderlichen Daten aus entsprechenden Daten-Shards abgerufen. Auf diese Weise braucht die aktuell ausgeführte API keine Daten „erneut abzurufen“, sondern verwendet vielmehr zuvor abgerufene Daten wieder. Dies ist erheblich effizienter, und es ist möglich, weil die zuvor abgerufenen Daten bereits auf Grundlage der Anforderungen der API gepflegt worden sind. Sie brauchen nicht in Vorbereitung auf die Verarbeitung neu erstellt zu werden.

[0048] Dementsprechend überprüft (Schritt **408**) der zentrale Datendienst **120** auf Grundlage der erzeugten (Schritt **406**) Liste von erwarteten Daten-Shards die vorhandenen Daten-Shards in seiner zentralen Metadaten-Registrierungsdatenbank **122** und ruft (Schritt **410**) fehlende Daten aus der Rohdatenquelle **104** ab und erzeugt (Schritt **412**) entsprechende Daten-Shards und Metadatendateien für die neu abgerufenen Daten und fügt sie der zentralen Datenablage **122** bzw. der zentralen Metadaten-Registrierungsdatenbank **124** hinzu.

[0049] Dadurch stellt das Verfahren **400** einen Mechanismus zum Wiederverwenden von zuvor abgerufenen Daten bereit und verarbeitet neu abgerufene Daten so, dass sie im Folgenden wiederverwendet werden können. Unter Bezugnahme zurück auf das Beispiel von Auftrag A und Auftrag B; beginnt das Datenverarbeitungssystem **100** beim Verarbeiten des Auftrags B eine Ausführung der API 1 für den Auftrag B. Das Datenverarbeitungssystem **100** erkennt, dass die Daten-Shards und Parameterkonfigurationen für den Auftrag B, API 1, mit denjenigen für den Auftrag A, API 1, übereinstimmen. Das Datenverarbeitungssystem **100** führt Schritte des Verfahrens **400** aus, um zuvor für den Auftrag A abgerufene Daten-Shards wiederzuverwenden, so dass die Daten nicht „erneut abgerufen“ werden müssen. Das Datenverarbeitungssystem **100** wiederholt die Ausführung einiger oder sämtlicher Schritte des Verfahrens **400** für den Auftrag B, APIs 2 bis 4, auf dieselbe Weise (es ist zu beachten, dass diese Reihenfolge der Ausführung lediglich zur Veranschaulichung dient; beispielsweise können zahlreiche oder sämtliche der Schritte des Verfahrens gleichzeitig oder in unterschiedlichen Reihenfolgen für mehr als eine API gleichzeitig durchgeführt werden). Irgendwann ermittelt das Datenverarbeitungssystem **100**, dass sich die Parameterkonfigurationen für die API 5 für Auftrag A und Auftrag B unterscheiden. Daher verarbeitet das Datenverarbeitungssystem **100** Auftrag B mithilfe der API 5, ohne sich

auf die vorab abgerufenen Daten von Auftrag A zu stützen, verwendet jedoch solche Daten zum Verarbeiten von Auftrag B mithilfe der APIs 1 bis 4.

[0050] **Fig. 5** stellt ein Beispiel für ein Verfahren **500** zum Wiederverwenden von API-Ausgaben über Verarbeitungsaufträge hinweg in dem Verarbeitungssystem **100** (**Fig. 1**) gemäß einer Ausführungsform der Erfindung bereit. Die verschiedenen Funktionen oder Schritte des Verfahrens **500** werden durch Programmieranweisungen ermöglicht, die durch einen oder mehrere Prozessoren des Datenverarbeitungssystems **100** ausführbar sind. In der folgenden Erörterung kann jeder Schritt als durch eine bestimmte Komponente des Datenverarbeitungssystems **100** ausführbar bezeichnet werden; dies dient jedoch lediglich der Veranschaulichung. Die Schritte oder Funktionen können auf andere Weise im Wesentlichen genauso ausgeführt werden, um im Wesentlichen dieselben Ergebnisse zu erzielen, ohne vom Wesensgehalt oder Umfang der Erfindung abzuweichen.

[0051] Unter Bezugnahme auf die **Fig. 1** und **Fig. 5** wird eine API-Ausgabe ausgewählt (Schritt **502**). Bei der ausgewählten API-Ausgabe handelt es sich um eine Ausgabe, die während einer Verarbeitung eines ersten Auftrags durch eine API erzeugt worden ist. Die Auswahl kann (manuell durch einen Benutzer oder automatisch) durch die Verarbeitungs-Pipeline **102** zur Unterstützung einer oder mehrerer APIs vorgenommen werden, die einen zweiten Auftrag verarbeiten. Unter Bezugnahme zurück auf das Beispiel von Auftrag A und Auftrag B und die APIs bis 17; wird Auftrag A mithilfe der APIs 1 bis 7 verarbeitet, wobei jede API eine Ausgabe auf Grundlage ihrer Verarbeitung erzeugt. Auftrag B trifft in dem Datenverarbeitungssystem **100** zur Verarbeitung ein. Da seine Parameterkonfigurationen mit denjenigen für Auftrag B im Hinblick auf die APIs 1 bis 4 übereinstimmen, kann die Ausgabe der API 4 für Auftrag B wiederverwendet werden. Daher wird die Ausgabe der API 4 zur Wiederverwendung ausgewählt (Schritt **502**).

[0052] Die Verarbeitungs-Pipeline **102** fragt (Schritt **504**) den zentralen Datendienst **120** im Hinblick auf die Metadaten für die API ab, deren Ausgabe wiederzuverwenden ist. Ein Beispiel für eine Metadatendatei, die auf Grundlage einer Verarbeitung eines Auftrags über eine API erzeugt wird, wird in TABELLE 2 bereitgestellt. In diesem Beispiel handelt es sich bei den APIs, deren Ausgaben in der Metadatendatei gespeichert werden, um die APIs 1 bis 4.

TABELLE 2

Ausgabe-API: 4
Datenbereich: 2015-03 - 2017-06
Erstellungszeit: 2017-08-09 10:18:16.598385253
Pipeline-Typ: Normal
Abrufpfad: data@192.168.9.9:/ home/data/api4-output
Typ: Verzeichnis

API1

API2
...
API3
...
API4
Daten:
Train_Merkmal_Beginn: - „2015-03“
Merkmal_Erstellung:
Ziel: „zul_Menge:Summe“
...
...

[0053] Der zentrale Datendienst **120** überprüft (Entscheidungsschritt **506**), ob ein erster Auftrag (der bereits verarbeitet worden ist) und der zweite Auftrag (der zu verarbeiten ist) dieselben Parameterkonfigurationen gemeinsam nutzen. Die Überprüfung wird durch Abfragen der Parameterdateien jedes Auftrags und der Metadatendateien der betreffenden APIs (zum Beispiel der Metadatendatei in TABELLE 2) durchgeführt.

[0054] Wenn die Überprüfung fehlschlägt (keine Verzweigung des Entscheidungsschritts **510**), d.h., wenn der erste Auftrag und der zweite Auftrag nicht dieselben Parameterkonfigurationen gemeinsam nutzen, endet der Prozess (Schritt **514**). Die Beendigung kann zum Beispiel ein Übertragen einer Nachricht an einen Client **150** (in der Annahme eines Schritts **502** zur manuellen Auswahl) enthalten, dass die ausgewählte API-Ausgabe, wie sie auf Grundlage der Verarbeitung des ersten Auftrags erzeugt worden ist, nicht zum Verarbeiten des zweiten Auftrags verwendet werden kann.

[0055] Wenn die Überprüfung erfolgreich ist (Ja-Verzweigung des Entscheidungsschritts **506**), bezieht (Schritt **508**) der zentrale Datendienst **120** demgegenüber den Abrufpfad (z.B. Daten über die Adresse und das Verzeichnis des Servers) für die ausgewählte Ausgabe der API. Die Verarbeitung geht zu einer Überprüfung (Entscheidungsschritt **510**) eines Vorhandenseins und einer Konsistenz von Daten über. Wenn die Daten nicht vorhanden sind oder nicht intakt sind (Nein-Verzweigung des Entscheidungsschritts **510**), endet die Verarbeitung (ähnlich wie die Beendigung nach dem Entscheidungsschritt **506**). Wenn die Daten vorhanden sind und intakt sind (Ja-Verzweigung des Entscheidungsschritts **510**), liest (Schritt **512**) der zentrale Datendienst **120** die ausgewählte Ausgabe der APIs für den ersten Auftrag (die beim Verarbeiten des ersten Auftrags ausgegeben worden ist) und beginnt eine Ausführung der nächsten API für den zweiten Auftrag mithilfe dieser Ausgabe.

[0056] Wenn die Verarbeitungs-Pipeline **120** ihre Verarbeitungsfunktionen für den zweiten Auftrag mithilfe der nächsten API durchführt, überträgt sie ihre Ausgabe zum Speichern an den zentralen Datendienst **120**. TABELLE 3 stellt ein Beispiel für eine Ausgabe der API 5 bereit, die beim Ausführen des Auftrags B auf Grundlage von Ausgaben der APIs 1 bis 4, die beim Ausführen von Auftrag A erzeugt worden sind, gespeichert wird.

TABELLE 3

Ausgabe-API: 5
Datenbereich: 2015-03 - 2017-06
Erstellungszeit: 2017-08-09 10:18:16.598385253 Pipeline-
Type: Inkrementelles Training
Abrufpfad: data@202.168.9.9:/ home/data/api5-output
Typ: Verzeichnis
API1
...
API2
...
API3
...
API4
...
API5
...

[0057] Unter Bezugnahme auf **Fig. 6** wird eine beispielhafte graphische Benutzeroberfläche (GUI) **600** zum Wiederverwenden von Daten-Shards und API-Ausgaben, die durch Verarbeiten eines vorhergehenden Auftrags erzeugt worden sind, bereitgestellt. In dem dargestellten Beispiel werden zwei Aufträge bezeichnet als: „aktueller Auftrag“ und „wiederverwendeter Auftrag“. An anderer Stelle sind diese beiden Aufträge abhängig vom Zusammenhang als erster Auftrag und zweiter Auftrag oder Auftrag A und Auftrag B bezeichnet worden. „Wiederverwendete API-Ergebnisse“ in **Fig. 6** bezieht sich auf die Ausgabe (in diesem Fall) der API 4.1. Die GUI **600** stellt einem Client **510** ein Dropdown-Menü zum Auswählen einer von zwei APIs mit verfügbaren

Ausgaben bereit: API 4.1 und API 5. Zu Beginn der Ausführung des „aktuellen Auftrags“ mithilfe dieser Einstellungen verarbeitet das Datenverarbeitungssystem **100** entweder den „aktuellen Auftrag“ mithilfe vorhandener Daten-Shards und API-Ausgaben oder, wenn die Parameterkonfigurationen des „aktuellen Auftrags“ und des „wiederverwendeten Auftrags“ nicht übereinstimmen, beendet den Prozess mit einer Nachricht an den Client, dass die Verarbeitung nicht erfolgreich war oder nicht durchgeführt werden kann. Ausführungsformen der Erfindung können vorgeschlagene APIs, vorgeschlagene Daten-Shards oder vorgeschlagene sonstige Aufträge bereitstellen, die stattdessen zu verwenden sind. Bei einer verwandten Ausführungsform werden diese Vorschläge zu Beginn gemacht.

[0058] Unter Bezugnahme auf **Fig. 7** wird eine beispielhafte GUI **700** zum Überwachen von Auftragsstatus bereitgestellt. Ein Client kann die GUI **700** dazu verwenden, Auftragsstatus zu überwachen, um API-Ausgaben von geeigneten Aufträgen auszuwählen. In dem dargestellten Beispiel werden zwei Aufträge gezeigt. Der erste Auftrag ist den APIs 1 bis 7 (einige werden nicht dargestellt, und einige weisen Unter-APIs auf) zugehörig, wobei einige von ihnen für den ersten Auftrag ausgeführt worden sind und zugehörige Ausgaben aufweisen. Der zweite Auftrag ist denselben APIs zugehörig, wobei keine von ihnen im Hinblick auf den zweiten Auftrag ausgeführt worden ist. In der Annahme, dass die beiden dargestellten Aufträge Konfigurationsparameter für die APIs 1 bis 4.1 (jedoch nicht für 4.2) gemeinsam nutzen, kann die Datenverarbeitungsplattform **100** eine Verarbeitung des zweiten Auftrags über die API 5 mithilfe von Ausgaben der APIs 1 bis 4.1 beginnen, die beim Verarbeiten des ersten Auftrags erzeugt worden sind.

[0059] Unter Bezugnahme auf **Fig. 8** wird eine schematische Darstellung einer beispielhaften Datenverarbeitungseinheit (bei der es sich um einen Cloud-Computing-Knoten handeln kann) gemäß einer Ausführungsform der Erfindung dargestellt. Die Datenverarbeitungseinheit **10** ist lediglich ein Beispiel für einen geeigneten Cloud-Computing-Knoten und soll den Umfang der Nutzung oder der Funktionalität von Ausführungsformen der hierin beschriebenen Erfindung nicht einschränken. Die Datenverarbeitungseinheit **10** ist ein Beispiel für eine oder mehrere der physischen und virtuellen Einheiten des Datenverarbeitungssystems **100** (**Fig. 1**).

[0060] In der Datenverarbeitungseinheit **10** befindet sich ein Computersystem/Server **12**, das/der mit zahlreichen sonstigen Universal- oder Spezial-Datenverarbeitungssystem-Umgebungen oder -Konfigurationen betrieben werden kann. Zu Beispielen für allgemein bekannte Datenverarbeitungssysteme, -Umgebungen und/oder -Konfigurationen, die zur Verwendung mit dem Computersystem/Server **12** geeignet sein können, zählen Personal-Computersysteme, Server-Computersysteme, Thin Clients, Thick Clients, Hand- oder Laptop-Einheiten, Mehrprozessorsysteme, Systeme auf Grundlage von Mikroprozessoren, Set-Top-Boxen, programmierbare Unterhaltungselektronik, Netzwerk-PCs, Minicomputersysteme, Großrechnersysteme und verteilte Cloud-Computing-Umgebungen, die beliebige der obigen Systeme oder Einheiten enthalten, und dergleichen, ohne auf diese beschränkt zu sein.

[0061] Das Computersystem/der Server **12** kann im allgemeinen Zusammenhang von Anweisungen beschrieben werden, die durch ein Computersystem ausführbar sind, wie zum Beispiel Programmmodule, die durch ein Computersystem ausgeführt werden. Im Allgemeinen können Programmmodule Routinen, Programme, Objekte, Komponenten, Logik, Datenstrukturen usw. enthalten, die bestimmte Aufgaben durchführen oder bestimmte abstrakte Datentypen implementieren. Das Computersystem/der Server **12** kann in verteilten Cloud-Computing-Umgebungen angewendet werden, in denen Aufgaben durch entfernt angeordnete Verarbeitungseinheiten durchgeführt werden, die durch ein Datenübertragungsnetzwerk miteinander verbunden sind. Bei einer verteilten Cloud-Computing-Umgebung können sich Programmmodule sowohl in lokalen als auch in entfernt angeordneten Computersystem-Speichermedien befinden, darunter in Speichereinheiten.

[0062] Wie in **Fig. 8** gezeigt, wird das Computersystem/der Server **12** in der Datenverarbeitungseinheit **10** in Form einer Universal-Datenverarbeitungseinheit dargestellt. Zu den Komponenten des Computersystems/Servers **12** können ein oder mehrere Prozessoren oder Verarbeitungseinheiten **16**, ein Systemspeicher **28** und ein Bus **18** zählen, der verschiedene Systemkomponenten wie etwa den Systemspeicher **28** mit dem Prozessor **16** verbindet, ohne auf diese beschränkt zu sein.

[0063] Der Bus **18** stellt einen oder mehrere von mehreren beliebigen Typen von Busstrukturen dar, darunter einen Speicherbus oder eine Speichersteuereinrichtung, einen Peripheriebus, einen Accelerated Graphics Port und einen Prozessor- oder einen lokalen Bus unter Verwenden einer beliebigen von einer Vielfalt von Busarchitekturen. Beispielsweise, und ohne einschränkend zu wirken, enthalten solche Architekturen einen Industry-Standard-Architecture(ISA)-Bus, einen Micro-Channel-Architecture(MCA)-Bus, einen Enhanced-ISA (EISA)-Bus, einen lokalen Video-Electronics-Standards-Association(VESA)-Bus und einen Peripheral-Component-Interconnects(PCI)-Bus.

[0064] Das Computersystem/der Server **12** enthält üblicherweise eine Vielfalt von durch ein Computersystem lesbaren Medien. Bei solchen Medien kann es sich um beliebige verfügbare Medien handeln, auf die durch das Computersystem/den Server **12** zugegriffen werden kann, und sie enthalten sowohl flüchtige als auch nichtflüchtige Medien sowie austauschbare und nichtaustauschbare Medien.

[0065] Der Systemspeicher **28** kann durch ein Computersystem lesbare Medien in Form eines flüchtigen Speichers wie zum Beispiel eines Direktzugriffsspeichers (random access memory, RAM) **30** und/oder eines Cache-Speichers **32** enthalten. Das Computersystem/der Server **12** kann des Weiteren sonstige austauschbare/nicht austauschbare, flüchtige/nichtflüchtige Computersystem-Speichermedien enthalten. Lediglich als Beispiel kann ein Speichersystem **34** zum Lesen von einem nicht austauschbaren, nichtflüchtigen (nicht dargestellten und üblicherweise als „Festplatte“ bezeichneten) Magnetmedium und zum Schreiben darauf bereitgestellt werden. Wenngleich es nicht dargestellt wird, kann ein Magnetplattenlaufwerk zum Lesen von einer austauschbaren, nichtflüchtigen Magnetplatte (z.B. einer „Diskette“) und zum Schreiben darauf und ein optisches Plattenlaufwerk zum Lesen von einer austauschbaren, nichtflüchtigen optischen Platte wie zum Beispiel einer CD-ROM, DVD-ROM oder sonstigen optischen Medien und zum Schreiben darauf bereitgestellt werden. In solchen Fällen kann jedes durch eine oder mehrere Datenmedien-Schnittstellen mit dem Bus **18** verbunden sein. Wie im Folgenden näher dargestellt und beschrieben wird, kann der Speicher **28** zumindest ein Programmprodukt enthalten, das einen Satz (z.B. zumindest eins) von Programmmodulen aufweist, die dazu konfiguriert sind, die Funktionen von Ausführungsformen der Erfindung auszuführen.

[0066] Ein Programm/Dienstprogramm **40**, das einen Satz (zumindest eins) von Programmmodulen **42** aufweist, kann als Beispiel, das keine Einschränkung darstellen soll, in dem Speicher **28** gespeichert werden, wie auch ein Betriebssystem, ein oder mehrere Anwendungsprogramme, sonstige Programmmodule und Programmdaten. Von dem Betriebssystem, dem einen oder den mehreren Anwendungsprogrammen, den sonstigen Programmmodulen und Programmdaten und einigen Kombinationen von diesen kann jedes eine Implementierung einer Netzwerkumgebung enthalten. Die Programmmodule **42** führen im Allgemeinen die Funktionen und/oder Methoden von Ausführungsformen der Erfindung aus, wie sie hierin beschrieben wird.

[0067] Das Computersystem/der Server **12** kann außerdem mit einer oder mehreren externen Einheiten **14** wie zum Beispiel einer Tastatur, einer Zeigeeinheit, einer Anzeige **24** usw.; einer oder mehreren Einheiten, die einem Benutzer ermöglichen, mit dem Computersystem/dem Server **12** in Wechselwirkung zu treten; und/oder beliebigen Einheiten (z.B. einer Netzwerkkarte, einem Modem usw.) Daten austauschen, die dem Computersystem/Server **12** ermöglichen, Daten mit einer oder mehreren sonstigen Datenverarbeitungseinheiten auszutauschen. Ein solcher Datenaustausch kann über Eingabe-/Ausgabe(E/A)-Schnittstellen **22** durchgeführt werden. Weiterhin kann das Computersystem/der Server **12** mit einem oder mehreren Netzwerken wie zum Beispiel einem lokalen Netzwerk (lokal area network, LAN), einem allgemeinen Weitverkehrsnetzwerk (wide area network, WAN) und/oder einem öffentlichen Netzwerk (z.B. dem Internet) über einen Netzwerkadapter **20** Daten austauschen. Wie dargestellt, tauscht der Netzwerkadapter **20** Daten mit den anderen Komponenten des Computersystems/Servers **12** über den Bus **18** aus. Es versteht sich, wenngleich dies nicht dargestellt wird, dass sonstige Hardware- und/oder Software-Komponenten zusammen mit dem Computersystem/Server **12** verwendet werden könnten. Zu Beispielen zählen Mikrocode, Einheitentreiber, redundante Verarbeitungseinheiten, externe Plattenlaufwerk-Arrays, RAID-Systeme, Bandlaufwerke und Datenarchivierungs-Speichersysteme usw., ohne auf diese beschränkt zu sein.

[0068] Unter allgemeiner Bezugnahme auf Ausführungsformen der vorliegenden Erfindung kann es sich bei den Ausführungsformen um ein System, ein Verfahren und/oder ein Computerprogrammprodukt mit einem beliebigen Integrationsgrad technischer Details handeln. Das Computerprogrammprodukt kann (ein) durch einen Computer lesbare(s) Speichermedium (oder -medien) enthalten, auf dem/denen durch einen Computer lesbare Programmanweisungen gespeichert ist/sind, um einen Prozessor dazu zu veranlassen, Aspekte der vorliegenden Erfindung auszuführen.

[0069] Bei dem durch einen Computer lesbaren Speichermedium kann es sich um eine physische Einheit handeln, die Anweisungen zur Verwendung durch eine Einheit zur Ausführung von Anweisungen behalten und speichern kann. Bei dem durch einen Computer lesbaren Speichermedium kann es sich zum Beispiel um eine elektronische Speichereinheit, eine magnetische Speichereinheit, eine optische Speichereinheit, eine elektromagnetische Speichereinheit, eine Halbleiterspeichereinheit oder jede geeignete Kombination daraus handeln, ohne auf diese beschränkt zu sein. Zu einer nicht erschöpfenden Liste spezifischerer Beispiele des durch einen Computer lesbaren Speichermediums gehören die Folgenden: eine tragbare Computerdiskette, eine Festplatte, ein Direktzugriffsspeicher (RAM), ein Festwertspeicher (ROM), ein löschbarer programmierbarer Festwertspeicher (EPROM bzw. Flash-Speicher), ein statischer Direktzugriffsspeicher (SRAM), ein tragba-

rer Kompaktspeicherplatten-Festwertspeicher (CD-ROM), eine DVD (digital versatile disc), ein Speicher-Stick, eine Diskette, eine mechanisch codierte Einheit wie zum Beispiel Lochkarten oder erhabene Strukturen in einer Rille, auf denen Anweisungen gespeichert sind, und jede geeignete Kombination daraus. Ein durch einen Computer lesbares Speichermedium soll in der Verwendung hierin nicht als flüchtige Signale an sich aufgefasst werden, wie zum Beispiel Funkwellen oder andere sich frei ausbreitende elektromagnetische Wellen, elektromagnetische Wellen, die sich durch einen Wellenleiter oder ein anderes Übertragungsmedium ausbreiten (z.B. durch ein Glasfaserkabel geleitete Lichtimpulse) oder durch einen Draht übertragene elektrische Signale.

[0070] Hierin beschriebene, durch einen Computer lesbare Programmanweisungen können von einem durch einen Computer lesbaren Speichermedium auf jeweilige Datenverarbeitungs-/Verarbeitungseinheiten oder über ein Netzwerk wie zum Beispiel das Internet, ein lokales Netzwerk, ein Weitverkehrsnetz und/oder ein drahtloses Netzwerk auf einen externen Computer oder eine externe Speichereinheit heruntergeladen werden. Das Netzwerk kann Kupferübertragungskabel, Lichtwellenübertragungsleiter, drahtlose Übertragung, Router, Firewalls, Vermittlungseinheiten, Gateway-Computer und/oder Edge-Server aufweisen. Eine Netzwerkadapterkarte oder Netzwerkschnittstelle in jeder Datenverarbeitungs-/Verarbeitungseinheit empfängt durch einen Computer lesbare Programmanweisungen aus dem Netzwerk und leitet die durch einen Computer lesbaren Programmanweisungen zur Speicherung in einem durch einen Computer lesbaren Speichermedium innerhalb der entsprechenden Datenverarbeitungs-/Verarbeitungseinheit weiter.

[0071] Bei durch einen Computer lesbaren Programmanweisungen zum Ausführen von Arbeitsschritten der vorliegenden Erfindung kann es sich um Assembler-Anweisungen, ISA-Anweisungen (Instruction-Set-Architecture), Maschinenanweisungen, maschinenabhängige Anweisungen, Mikrocode, Firmware-Anweisungen, zustandssetzende Daten, Konfigurationsdaten für integrierte Schaltungen oder entweder Quellcode oder Objektcode handeln, die in einer beliebigen Kombination aus einer oder mehreren Programmiersprachen geschrieben werden, darunter objektorientierte Programmiersprachen wie Smalltalk, C++ o.ä. sowie herkömmliche prozedurale Programmiersprachen wie die Programmiersprache „C“ oder ähnliche Programmiersprachen. Die durch einen Computer lesbaren Programmanweisungen können vollständig auf dem Computer des Benutzers, teilweise auf dem Computer des Benutzers, als eigenständiges Software-Paket, teilweise auf dem Computer des Benutzers und teilweise auf einem entfernt angeordneten Computer oder vollständig auf dem entfernt angeordneten Computer oder Server ausgeführt werden. In letzterem Fall kann der entfernt angeordnete Computer mit dem Computer des Benutzers durch eine beliebige Art Netzwerk verbunden sein, darunter ein lokales Netzwerk (LAN) oder ein Weitverkehrsnetz (WAN), oder die Verbindung kann mit einem externen Computer hergestellt werden (zum Beispiel über das Internet unter Verwenden eines Internet-Dienstanbieters). In einigen Ausführungsformen können elektronische Schaltungen, darunter zum Beispiel programmierbare Logikschaltungen, im Feld programmierbare Gatter-Anordnungen (FPGA, field programmable gate arrays) oder programmierbare Logikanordnungen (PLA, programmable logic arrays) die durch einen Computer lesbaren Programmanweisungen ausführen, indem sie Zustandsdaten der durch einen Computer lesbaren Programmanweisungen nutzen, um die elektronischen Schaltungen zu personalisieren, um Aspekte der vorliegenden Erfindung durchzuführen.

[0072] Aspekte der vorliegenden Erfindung sind hierin unter Bezugnahme auf Ablaufpläne und/oder Blockschaubilder von Verfahren, Vorrichtungen (Systemen) und Computerprogrammprodukten gemäß Ausführungsformen der Erfindung beschrieben. Es wird darauf hingewiesen, dass jeder Block der Ablaufpläne und/oder der Blockschaubilder sowie Kombinationen von Blöcken in den Ablaufplänen und/oder den Blockschaubildern mittels durch einen Computer lesbare Programmanweisungen ausgeführt werden können.

[0073] Diese durch einen Computer lesbaren Programmanweisungen können einem Prozessor eines Universalcomputers, eines Spezialcomputers oder einer anderen programmierbaren Datenverarbeitungsvorrichtung bereitgestellt werden, um eine Maschine zu erzeugen, so dass die über den Prozessor des Computers bzw. der anderen programmierbaren Datenverarbeitungsvorrichtung ausgeführten Anweisungen Mittel zur Umsetzung der in dem Block bzw. den Blöcken der Ablaufpläne und/oder der Blockschaubilder festgelegten Funktionen/Schritte erzeugen. Diese durch einen Computer lesbaren Programmanweisungen können auch auf einem durch einen Computer lesbaren Speichermedium gespeichert sein, das einen Computer, eine programmierbare Datenverarbeitungsvorrichtung und/oder andere Einheiten so steuern kann, dass sie auf eine bestimmte Art funktionieren, so dass das durch einen Computer lesbare Speichermedium, auf dem Anweisungen gespeichert sind, ein Herstellungsprodukt aufweist, darunter Anweisungen, welche Aspekte der/des in dem Block bzw. den Blöcken des Ablaufplans und/oder der Blockschaubilder angegebenen Funktion/Schritts umsetzen.

[0074] Die durch einen Computer lesbaren Programmanweisungen können auch auf einen Computer, eine andere programmierbare Datenverarbeitungsvorrichtung oder eine andere Einheit geladen werden, um das

Ausführen einer Reihe von Prozessschritten auf dem Computer bzw. der anderen programmierbaren Vorrichtung oder anderen Einheit zu verursachen, um einen auf einem Computer ausgeführten Prozess zu erzeugen, so dass die auf dem Computer, einer anderen programmierbaren Vorrichtung oder einer anderen Einheit ausgeführten Anweisungen die in dem Block bzw. den Blöcken der Ablaufpläne und/oder der Blockschaubilder festgelegten Funktionen/Schritte umsetzen.

[0075] Die Ablaufpläne und die Blockschaubilder in den Figuren veranschaulichen die Architektur, die Funktionalität und den Betrieb möglicher Ausführungen von Systemen, Verfahren und Computerprogrammprodukten gemäß verschiedenen Ausführungsformen der vorliegenden Erfindung. In diesem Zusammenhang kann jeder Block in den Ablaufplänen oder Blockschaubildern ein Modul, ein Segment oder einen Teil von Anweisungen darstellen, die eine oder mehrere ausführbare Anweisungen zur Ausführung der bestimmten logischen Funktion(en) aufweisen. In einigen alternativen Ausführungen können die in dem Block angegebenen Funktionen in einer anderen Reihenfolge als in den Figuren gezeigt stattfinden. Zwei nacheinander gezeigte Blöcke können zum Beispiel in Wirklichkeit im Wesentlichen gleichzeitig ausgeführt werden, oder die Blöcke können manchmal je nach entsprechender Funktionalität in umgekehrter Reihenfolge ausgeführt werden. Es ist ferner anzumerken, dass jeder Block der Blockschaubilder und/oder der Ablaufpläne sowie Kombinationen aus Blöcken in den Blockschaubildern und/oder den Ablaufplänen durch spezielle auf Hardware beruhende Systeme umgesetzt werden können, die die festgelegten Funktionen oder Schritte durchführen, oder Kombinationen aus Spezial-Hardware und Computeranweisungen ausführen.

Patentansprüche

1. Verfahren zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform, das aufweist:
Empfangen eines ersten Auftrags zur Verarbeitung, wobei der erste Auftrag einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform aufweist; und
Ausführen zumindest eines Abschnitts des ersten Auftrags über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline, wobei das Ausführen ein Verwenden zumindest eines Daten-Shards aufweist, der während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und des Weiteren ein Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule aufweist, wobei die Ausgabe während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird.

2. Verfahren nach Anspruch 1, wobei ein oder mehrere Abschnitte des zweiten Auftrags vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet werden.

3. Verfahren nach Anspruch 1, wobei die Verarbeitungsmodule eine oder mehrere Anwendungsprogrammierschnittstellen (APIs) aufweisen.

4. Verfahren nach Anspruch 1, wobei Parameterkonfigurationen von Aufträgen, die durch das Datenverarbeitungssystem verarbeitet werden, in jeweiligen Parameterdateien definiert sind, die den Aufträgen zugehörig sind.

5. Verfahren nach Anspruch 1, wobei Daten-Shards, die den zumindest einen Daten-Shard enthalten, der während der Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, durch einen zentralen Datendienst verwaltet werden.

6. Verfahren nach Anspruch 1, wobei der zentrale Datendienst virtualisiert ist.

7. Verfahren nach Anspruch 1, wobei der zentrale Datendienst eine zentrale Datenablage und eine zentrale Metadatenablage aufweist.

8. Verfahren nach Anspruch 7, wobei die zentrale Metadatenablage einen Satz von Metadatendateien für Daten-Shards aufweist, die in der zentralen Datenablage gespeichert sind, und des Weiteren einen zusätzlichen Satz von Metadatendateien für Ausgaben von Verarbeitungsmodulen der Verarbeitungs-Pipeline aufweist.

9. Verfahren nach Anspruch 1, das des Weiteren aufweist:
Abrufen des zumindest einen Daten-Shards, der während der Ausführung des Abschnitts des zweiten Auftrags erzeugt wird, aus einer Position, die in einer Metadatendatei für den Daten-Shard angegeben wird; und

Abrufen jeglicher Daten, die nicht durch einen bekannten Daten-Shard definiert sind, aus einer Rohdatenquelle.

10. Verfahren nach Anspruch 9, wobei das Abrufen des zumindest einen Daten-Shards und das Abrufen der Daten, die nicht durch einen bekannten Daten-Shard definiert sind, auf Grundlage einer Benutzerauswahl über eine graphische Benutzeroberfläche (GUI) ausgelöst wird.

11. Computerprogrammprodukt zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform, wobei das Computerprogrammprodukt eine nichttransitorische physische Speichereinheit aufweist, in der ein Programmcode verkörpert ist, wobei der Programmcode so durch einen Prozessor eines Computers ausführbar ist, dass er ein Verfahren ausführt, wobei das Verfahren aufweist:
Empfangen eines ersten Auftrags zur Verarbeitung durch den Prozessor, wobei der erste Auftrag einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform aufweist; und
Ausführen zumindest eines Abschnitts des ersten Auftrags durch den Prozessor über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline, wobei das Ausführen ein Verwenden zumindest eines Daten-Shards aufweist, der während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und des Weiteren ein Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule aufweist, wobei die Ausgabe während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird.

12. Computerprogrammprodukt nach Anspruch 11, wobei ein oder mehrere Abschnitte des zweiten Auftrags vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet werden.

13. Computerprogrammprodukt nach Anspruch 11, wobei die Verarbeitungsmodule eine oder mehrere Anwendungsprogrammierschnittstellen (APIs) aufweisen.

14. Computerprogrammprodukt nach Anspruch 11, wobei Parameterkonfigurationen von Aufträgen, die durch das Datenverarbeitungssystem verarbeitet werden, in jeweiligen Parameterdateien definiert sind, die den Aufträgen zugehörig sind.

15. Computerprogrammprodukt nach Anspruch 11, wobei Daten-Shards, die den zumindest einen Daten-Shard enthalten, der während der Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, durch einen zentralen Datendienst verwaltet werden.

16. Computerprogrammprodukt nach Anspruch 11, wobei der zentrale Datendienst virtualisiert ist.

17. Computerprogrammprodukt nach Anspruch 11, wobei der zentrale Datendienst eine zentrale Datenablage und eine zentrale Metadatenablage aufweist.

18. Computerprogrammprodukt nach Anspruch 17, wobei die zentrale Metadatenablage einen Satz von Metadatendateien für Daten-Shards aufweist, die in der zentralen Datenablage gespeichert sind, und des Weiteren einen zusätzlichen Satz von Metadatendateien für Ausgaben von Verarbeitungsmodulen der Verarbeitungs-Pipeline aufweist.

19. Computerprogrammprodukt nach Anspruch 11, das des Weiteren aufweist:
Abrufen des zumindest einen Daten-Shards, der während der Ausführung des Abschnitts des zweiten Auftrags erzeugt wird, durch den Prozessor aus einer Position, die in einer Metadatendatei für den Daten-Shard angegeben wird; und
Abrufen jeglicher Daten, die nicht durch einen bekannten Daten-Shard definiert sind, durch den Prozessor aus einer Rohdatenquelle.

20. Computerprogrammprodukt nach Anspruch 19, wobei das Abrufen des zumindest einen Daten-Shards und das Abrufen der Daten, die nicht durch einen bekannten Daten-Shard definiert sind, auf Grundlage einer Benutzerauswahl über eine graphische Benutzeroberfläche (GUI) ausgelöst wird.

21. Computersystem zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform, das aufweist:
eine oder mehrere Computereinheiten, die jeweils einen oder mehrere Prozessoren und eine oder mehrere physische Speichereinheiten aufweisen; und

ein Programm, das in zumindest einer der einen oder mehreren Speichereinheiten verkörpert ist, wobei das Programm eine Mehrzahl von Programmanweisungen zur Ausführung durch den einen oder die mehreren Prozessoren aufweist, wobei die Programmanweisungen Anweisungen aufweisen zu einem:
Empfangen eines ersten Auftrags zur Verarbeitung, wobei der erste Auftrag einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform aufweist; und
Ausführen zumindest eines Abschnitts des ersten Auftrags über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline, wobei das Ausführen ein Verwenden zumindest eines Daten-Shards aufweist, der während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und des Weiteren ein Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule aufweist, wobei die Ausgabe während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird.

22. Computersystem nach Anspruch 21, wobei ein oder mehrere Abschnitte des zweiten Auftrags vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet werden.

23. Computersystem nach Anspruch 21, wobei die Verarbeitungsmodule eine oder mehrere Anwendungsprogrammierschnittstellen (APIs) aufweisen.

24. Computersystem nach Anspruch 21, wobei die Datenverarbeitungsplattform einen zentralen Datendienst aufweist, wobei der zentrale Datendienst eine zentrale Datenablage und eine zentrale Metadatenablage aufweist, wobei die zentrale Metadatenablage einen Satz von Metadatendateien für Daten-Shards aufweist, die in der zentralen Datenablage gespeichert sind, und des Weiteren einen zusätzlichen Satz von Metadatendateien für Ausgaben von Verarbeitungsmodulen der Verarbeitungs-Pipeline aufweist.

25. Verfahren zum Verarbeiten eines Auftrags auf einer Datenverarbeitungsplattform, das aufweist:
Empfangen eines ersten Auftrags zur Verarbeitung, wobei der erste Auftrag einen Satz von Parameterkonfigurationen zur Ausführung des ersten Auftrags durch eine Verarbeitungs-Pipeline der Datenverarbeitungsplattform aufweist, wobei die Verarbeitungsmodule eine oder mehrere Anwendungsprogrammierschnittstellen (APIs) aufweisen; und
Ausführen zumindest eines Abschnitts des ersten Auftrags über ein oder mehrere Verarbeitungsmodule der Verarbeitungs-Pipeline, wobei das Ausführen ein Verwenden zumindest eines Daten-Shards aufweist, der während einer Ausführung eines Abschnitts eines zweiten Auftrags erzeugt wird, und des Weiteren ein Verwenden einer Ausgabe von zumindest einem Verarbeitungsmodul des einen oder der mehreren Verarbeitungsmodule aufweist, wobei die Ausgabe während einer Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, wobei ein oder mehrere Abschnitte des zweiten Auftrags vor einem Ausführen jeglichen Abschnitts des ersten Auftrags verarbeitet werden, wobei Daten-Shards, die den zumindest einen Daten-Shard enthalten, der während der Ausführung eines Abschnitts des zweiten Auftrags erzeugt wird, durch einen zentralen Datendienst verwaltet werden, der eine zentrale Datenablage und eine zentrale Metadatenablage aufweist, wobei das Ausführen des zumindest einen Abschnitts des ersten Auftrags des Weiteren aufweist:
Abrufen des zumindest einen Daten-Shards, der während der Ausführung des Abschnitts des zweiten Auftrags erzeugt wird, aus einer Position, die in einer Metadatendatei für den Daten-Shard angegeben wird; und
Abrufen jeglicher Daten, die nicht durch einen bekannten Daten-Shard definiert sind, aus einer Rohdatenquelle.

Es folgen 8 Seiten Zeichnungen

Anhängende Zeichnungen

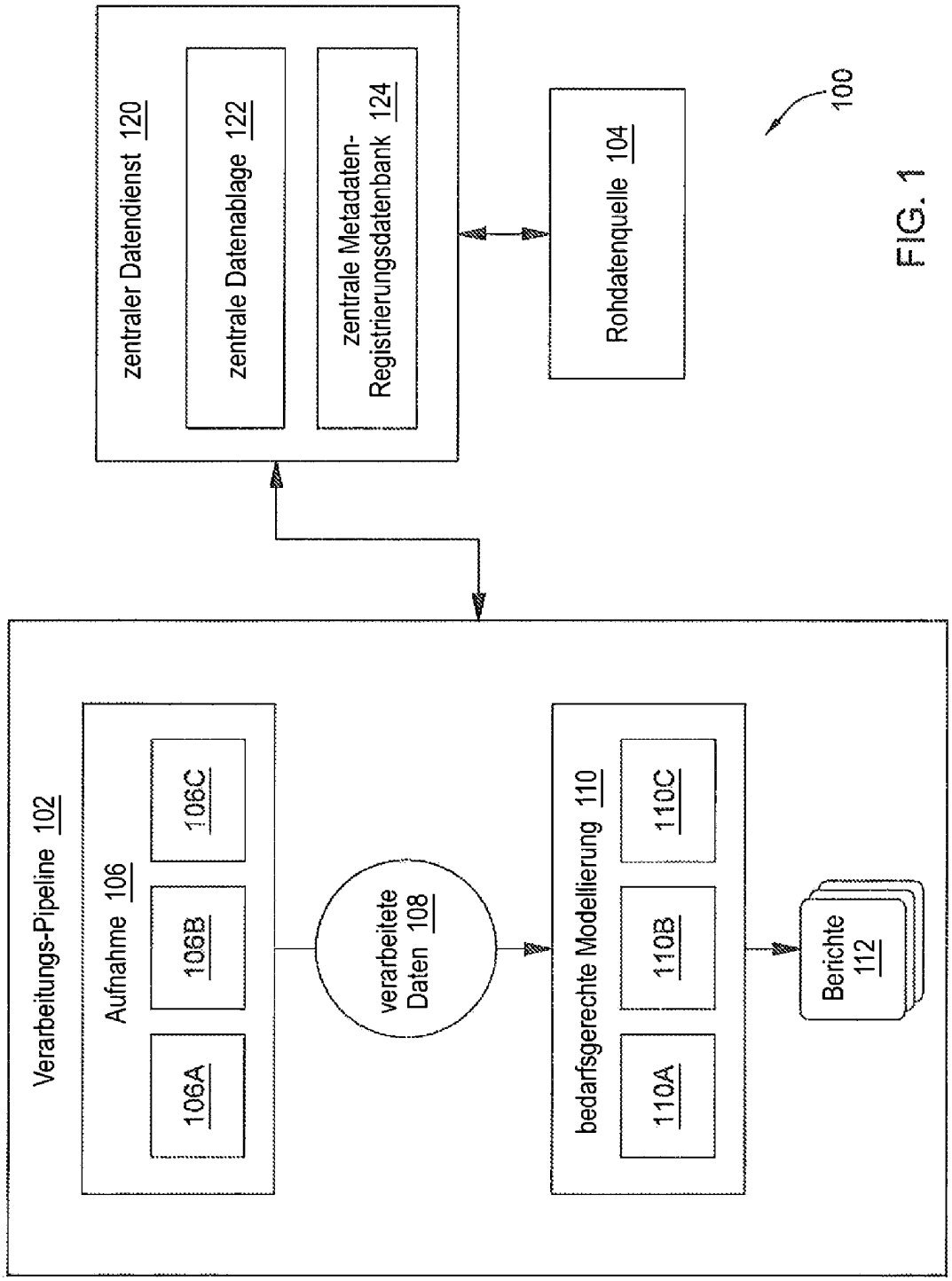


FIG. 1

```
[jingwei@watsonetnx03 db_pkl]$ ls
dfCode2011-01.pkl  dfCode2013-01.pkl  dfCode2015-01.pkl  dfCost2012-07.pkl  dfCost2014-07.pkl  dfpmpm2012-01.pkl  dfpmpm2014-01.pkl
dfCode2011-02.pkl  dfCode2013-02.pkl  dfCode2015-02.pkl  dfCost2012-08.pkl  dfCost2014-08.pkl  dfpmpm2012-02.pkl  dfpmpm2014-02.pkl
dfCode2011-03.pkl  dfCode2013-03.pkl  dfCode2015-03.pkl  dfCost2012-09.pkl  dfCost2014-09.pkl  dfpmpm2012-03.pkl  dfpmpm2014-03.pkl
dfCode2011-04.pkl  dfCode2013-04.pkl  dfCode2015-04.pkl  dfCost2012-10.pkl  dfCost2014-10.pkl  dfpmpm2012-04.pkl  dfpmpm2014-04.pkl
dfCode2011-05.pkl  dfCode2013-05.pkl  dfCode2015-05.pkl  dfCost2012-11.pkl  dfCost2014-11.pkl  dfpmpm2012-05.pkl  dfpmpm2014-05.pkl
dfCode2011-06.pkl  dfCode2013-06.pkl  dfCode2015-06.pkl  dfCost2012-12.pkl  dfCost2014-12.pkl  dfpmpm2012-06.pkl  dfpmpm2014-06.pkl
dfCode2011-07.pkl  dfCode2013-07.pkl  dfCode2011-01.pkl  dfCost2013-01.pkl  dfCost2015-01.pkl  dfpmpm2012-07.pkl  dfpmpm2014-07.pkl
dfCode2011-08.pkl  dfCode2013-08.pkl  dfCode2011-02.pkl  dfCost2013-02.pkl  dfCost2015-02.pkl  dfpmpm2012-08.pkl  dfpmpm2014-08.pkl
dfCode2011-09.pkl  dfCode2013-09.pkl  dfCode2011-03.pkl  dfCost2013-03.pkl  dfCost2015-03.pkl  dfpmpm2012-09.pkl  dfpmpm2014-09.pkl
dfCode2011-10.pkl  dfCode2013-10.pkl  dfCode2011-04.pkl  dfCost2013-04.pkl  dfCost2015-04.pkl  dfpmpm2012-10.pkl  dfpmpm2014-10.pkl
dfCode2011-11.pkl  dfCode2013-11.pkl  dfCode2011-05.pkl  dfCost2013-05.pkl  dfCost2015-05.pkl  dfpmpm2012-11.pkl  dfpmpm2014-11.pkl
dfCode2011-12.pkl  dfCode2013-12.pkl  dfCode2011-06.pkl  dfCost2013-06.pkl  dfCost2015-06.pkl  dfpmpm2012-12.pkl  dfpmpm2014-12.pkl
dfCode2012-01.pkl  dfCode2014-01.pkl  dfCode2011-07.pkl  dfCosi2013-07.pkl  dfCost2011-01.pkl  dfpmpm2013-01.pkl  dfpmpm2015-01.pkl
dfCode2012-02.pkl  dfCode2014-02.pkl  dfCode2011-08.pkl  dfCosi2013-08.pkl  dfCost2011-02.pkl  dfpmpm2013-02.pkl  dfpmpm2015-02.pkl
dfCode2012-03.pkl  dfCode2014-03.pkl  dfCode2011-09.pkl  dfCost2013-09.pkl  dfCost2011-03.pkl  dfpmpm2013-03.pkl  dfpmpm2015-03.pkl
dfCode2012-04.pkl  dfCode2014-04.pkl  dfCode2011-10.pkl  dfCost2013-10.pkl  dfCost2011-04.pkl  dfpmpm2013-04.pkl  dfpmpm2015-04.pkl
dfCode2012-05.pkl  dfCode2014-05.pkl  dfCode2011-11.pkl  dfCost2013-11.pkl  dfCost2011-05.pkl  dfpmpm2013-05.pkl  dfpmpm2015-05.pkl
dfCode2012-06.pkl  dfCode2014-06.pkl  dfCode2011-12.pkl  dfCost2013-12.pkl  dfCost2011-06.pkl  dfpmpm2013-06.pkl  dfpmpm2015-06.pkl
dfCode2012-07.pkl  dfCode2014-07.pkl  dfCode2012-01.pkl  dfCost2014-01.pkl  dfCost2011-07.pkl  dfpmpm2013-07.pkl  uCostCols.pkl
dfCode2012-08.pkl  dfCode2014-08.pkl  dfCode2012-02.pkl  dfCost2014-02.pkl  dfCost2011-08.pkl  dfpmpm2013-08.pkl  uPmpmCols.pkl
dfCode2012-09.pkl  dfCode2014-09.pkl  dfCode2012-03.pkl  dfCost2014-03.pkl  dfCost2011-09.pkl  dfpmpm2013-09.pkl
dfCode2012-10.pkl  dfCode2014-10.pkl  dfCode2012-04.pkl  dfCost2014-04.pkl  dfCost2011-10.pkl  dfpmpm2013-10.pkl
dfCode2012-11.pkl  dfCode2014-11.pkl  dfCode2012-05.pkl  dfCost2014-05.pkl  dfCost2011-11.pkl  dfpmpm2013-11.pkl
dfCode2012-12.pkl  dfCode2014-12.pkl  dfCode2012-06.pkl  dfCost2014-06.pkl  dfCost2011-12.pkl  dfpmpm2013-12.pkl
```

200

FIG. 2

Ausgabe-API: 2
Datenbereich: 2013-06
Erstellungszeit: 2017-08-09 10:18:16.598385253
Pipeline-Typ: Inkrementelles Training
Abrufpfad: hdfs://localhost:9010/2017-08-09/api2-2011-10-datashard
Typ: Datei

Datenbank:
-Host: „192.168.1.1“
-Datenbank_Name: „WATSON“
-Schema: „MARKETSCAN“

Tabellen:
-Ansprüche: „Anspruchs_Tabelle_1m“
-Mitglied: „Mitgliedschafts_Tabelle_1m“

Filter:
-Mitglied: „Mitgliedschafts_Tabelle_1m“
-Spalte: ‚pharmben_ja‘
-Operation: ‚==‘
-Wert: 1

-Tabelle: ‚Mitgliedschafts_Tabelle_1m‘
-Spalte: ‚Prod_Typ‘
-Operation: ‚notin‘
-Wert: [‚HMO‘,‘POS mit Zahlung‘]

FIG. 3

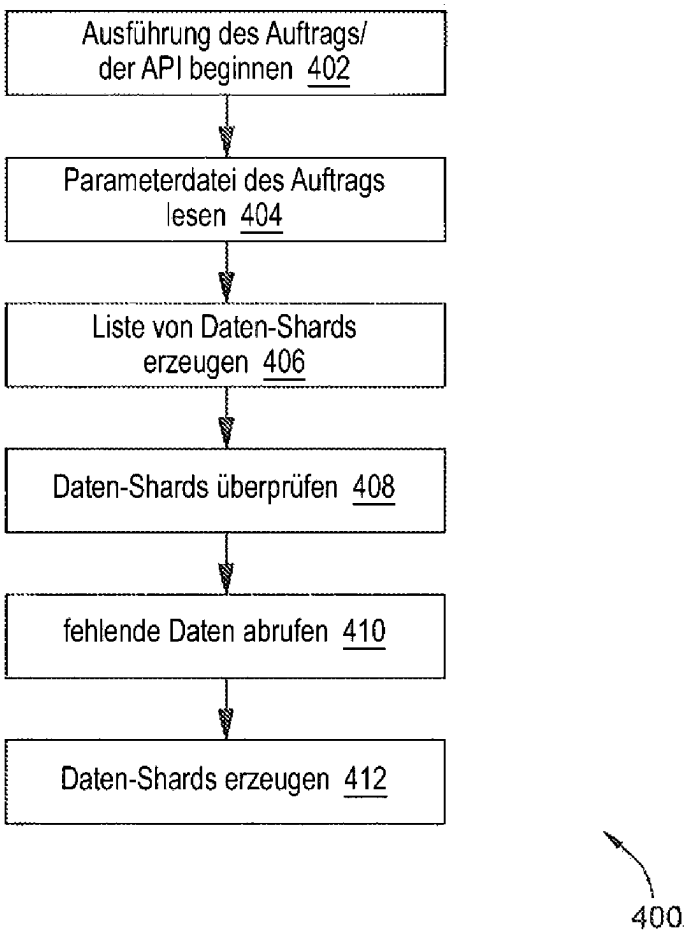

Ausführung des Auftrags/ der API beginnen 402
Parameterdatei des Auftrags lesen 404
Liste von Daten-Shards erzeugen 406
Daten-Shards überprüfen 408
fehlende Daten abrufen 410
Daten-Shards erzeugen 412
400

FIG. 4

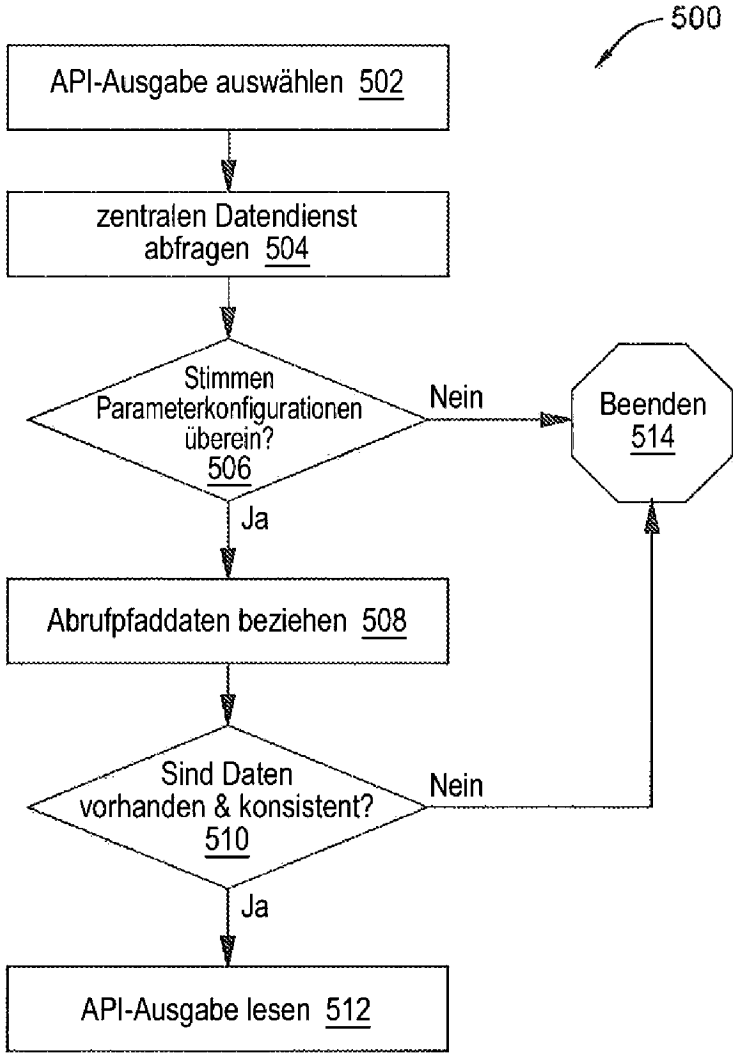

500
API-Ausgabe auswählen 502
zentralen Datendienst abfragen 504
Stimmen Parameterkonfigurationen überein? 506
Nein
Beenden 514
Ja
Abrufpfaddaten beziehen 508
Sind Daten vorhanden & konsistent? 510
Nein
Ja
API-Ausgabe lesen 512

FIG. 5

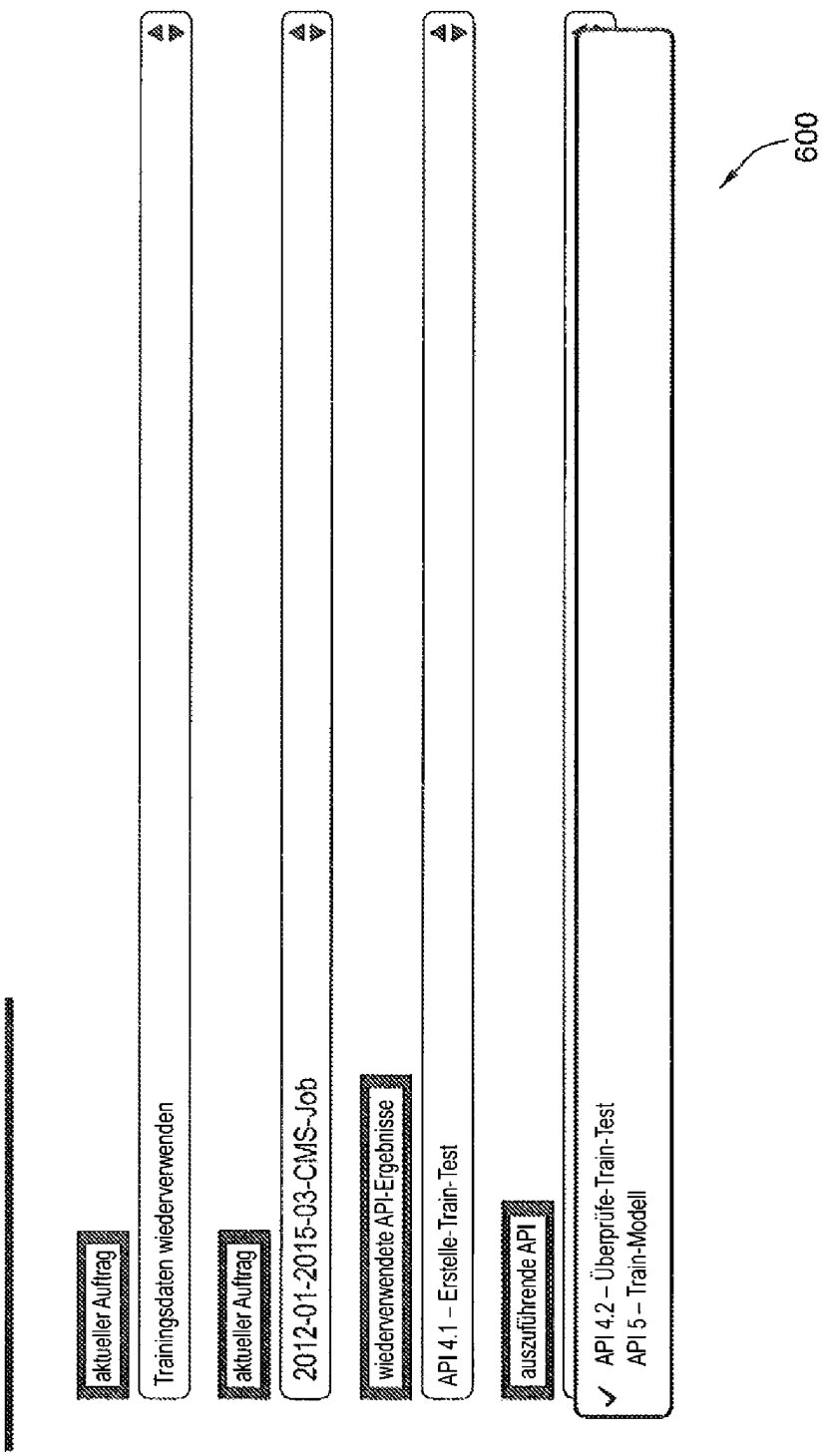

API ausführen (wiederverwenden)
API-Status
aktueller Auftrag
Trainingsdaten wiederverwenden
aktueller Auftrag
2012-01-2015-03-CMS-Job
wiederverwendete API-Ergebnisse
API 4.1 – Erstelle-Train-Test
auszuführende API
API 4.2 – Überprüfe-Train-Test
API 5 – Train-Modell
600

FIG. 6

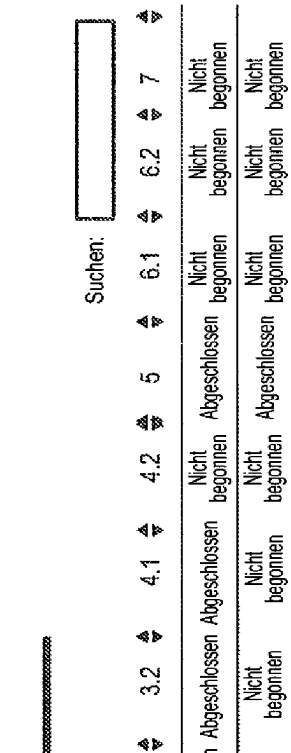

API ausführen (wiederverwenden)
API-Status
Suchen:
Auftragsname
Erstellungszeit
2
3.1
3.2
4.1
4.2
5
6.1
6.2
7
2012-01-2015-03-CMS-Job
Mi., 16. Aug. 2017 15:44:17 GMT
Abgeschlossen
Abgeschlossen
Abgeschlossen
Abgeschlossen
Nicht begonnen
Abgeschlossen
Nicht begonnen
Nicht begonnen
Nicht begonnen
Trainingsdaten wiederverwenden
Mi., 16. Aug. 2017 17:31:02 GMT
Nicht begonnen
Nicht begonnen
Nicht begonnen
Nicht begonnen
Nicht begonnen
Abgeschlossen
Nicht begonnen
Nicht begonnen
Nicht begonnen
Zurück
1
Weiter
700

FIG. 7

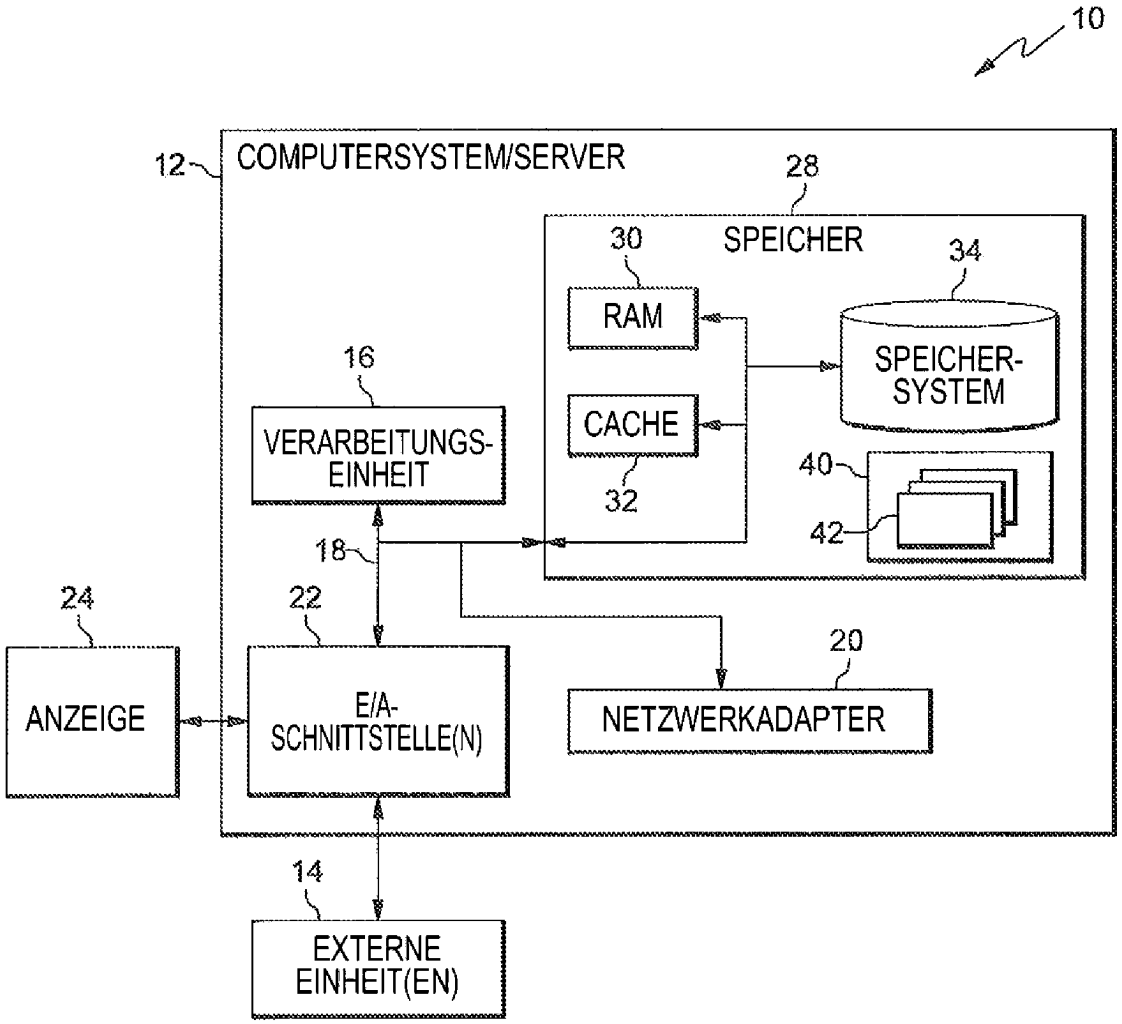


FIG. 8