(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 113632073 A
(43)申请公布日 2021.11.09

(21)申请号 202180001735.4
(22)申请日 2021.03.09
(30)优先权数据
16/812,940 2020.03.09 US
(85)PCT国际申请进入国家阶段日
2021.06.30
(86)PCT国际申请的申请数据
PCT/US2021/021516 2021.03.09
(87)PCT国际申请的公布数据
WO2021/183520 EN 2021.09.16
(71)申请人 斯诺弗雷克公司
地址 美国蒙大拿州
(72)发明人 伊斯特凡·切里 泰勒·琼斯 迪内希·钱德拉坎特·库尔卡尼 伊戈尔·津科夫斯基
(74)专利代理机构 北京安信方达知识产权代理有限公司 11262
代理人 周靖 杨明钊
(51)Int.Cl.
G06F 16/18(2006.01)
G06F 16/182(2006.01)
G06F 16/23(2006.01)
G06F 16/2455(2006.01)
G06F 16/2453(2006.01)

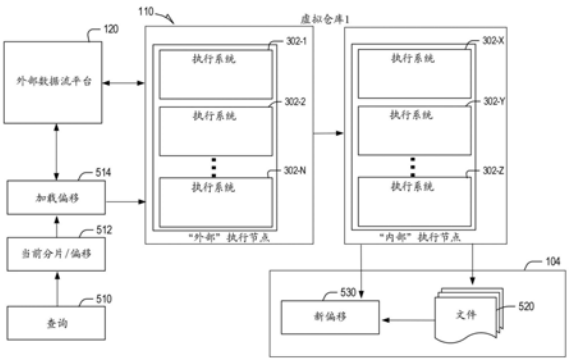
权利要求书3页 说明书20页 附图9页

(54)发明名称

数据源上的可扩展流

(57)摘要

本主题技术确定对应于可通过网络访问的外部数据源的分片集合，该外部数据源由与基于网络的数据库系统分离的外部系统托管。本主题技术使用流对象来确定分片集合中每个分片的偏移集合。本主题技术识别要在该分片集合上执行的操作，该操作包括读操作或写操作。本主题技术基于分片集合和偏移集合，对外部数据源执行操作。



CN 113632073 A

1.一种基于网络的数据库系统,包括:

至少一个硬件处理器;以及

存储器,所述存储器存储使所述至少一个硬件处理器执行包括以下项的操作的指令:

确定对应于通过网络可访问的外部数据源的分片集合,所述外部数据源由与所述基于网络的数据库系统分离的外部系统托管;

使用流对象确定所述分片集合中每个分片的偏移集合;

识别要对所述分片集合执行的操作,所述操作包括读操作或写操作;和

基于所述分片集合和所述偏移集合,对所述外部数据源执行所述操作。

2.根据权利要求1所述的基于网络的数据库系统,其中所述分片集合的每个分片包括由所述外部数据源存储的数据记录序列。

3.根据权利要求2所述的基于网络的数据库系统,其中确定所述分片集合中每个分片的偏移集合包括:

执行包括在所述流对象中的用户定义函数,以确定所述偏移集合。

4.根据权利要求3所述的基于网络的数据库系统,其中,所述用户定义函数在被执行时,确定相对于在先前事务期间对应于所述流对象的可扩展流被访问的先前时间的偏移,并且在所述先前事务完成时预先存储了相对于所述先前时间的偏移。

5.根据权利要求4所述的基于网络的数据库系统,其中所述操作还包括:

确定在所述可扩展流被访问的所述先前时间之后的新偏移的集合。

6.根据权利要求5所述的基于网络的数据库系统,其中所述操作还包括:

确定在所述可扩展流被访问的所述先前时间之后创建了新分片;

确定对应于所述新分片的新偏移;和

提供所述新偏移与所述新偏移的集合。

7.根据权利要求1所述的基于网络的数据库系统,其中所述操作还包括:

确定多个偏移对应于所述分片集合中的特定分片;

将所述分片集合中的所述特定分片分成至少两个分片;和

在访问所述至少两个分片中的第二分片之前访问第一分片。

8.根据权利要求3所述的基于网络的数据库系统,其中,所述用户定义函数在被执行时,至少部分地基于所述分片集合和所述偏移集合向所述外部数据源请求数据。

9.根据权利要求8所述的基于网络的数据库系统,其中所述用户定义函数使用基于所述分片集合的一组并行操作来请求数据。

10.根据权利要求3所述的基于网络的数据库系统,其中所述用户定义函数在被执行时,至少部分地基于来自所述分片集合的特定分片和来自内部数据源的要存储在所述特定分片中的一组行,将数据存储到所述外部数据源中。

11.根据权利要求1所述的基于网络的数据库系统,其中所述流对象包括:

第一函数,其提供存储的偏移的集合;

第二函数,其存储偏移的第二集合;和

第三函数,其提供与外部对象的集成,从而实现与所述外部数据源的通信。

12.一种方法,包括:

使用至少一个硬件处理器确定对应于通过网络可访问的外部数据源的分片集合,所述

外部数据源由与基于网络的数据库系统分离的外部系统托管；

使用流对象确定所述分片集合中每个分片的偏移集合；

识别要对所述分片集合执行的操作，所述操作包括读操作或写操作；和

基于所述分片集合和所述偏移集合，对所述外部数据源执行所述操作。

13.根据权利要求12所述的方法，其中所述分片集合的每个分片包括由所述外部数据源存储的数据记录序列。

14.根据权利要求13所述的方法，其中确定所述分片集合中的每个分片的偏移集合包括：

执行包括在所述流对象中的用户定义函数，以确定所述偏移集合。

15.根据权利要求14所述的方法，其中，所述用户定义函数在被执行时，确定相对于在先前事务期间对应于所述流对象的可扩展流被访问的先前时间的偏移，并且在所述先前事务完成时预先存储了相对于所述先前时间的偏移。

16.根据权利要求15所述的方法，还包括：

确定在所述可扩展流被访问的所述先前时间之后的新偏移的集合。

17.根据权利要求16所述的方法，还包括：

确定在所述可扩展流被访问的所述先前时间之后创建了新分片；

确定对应于所述新分片的新偏移；和

提供所述新偏移与所述新偏移的集合。

18.根据权利要求12所述的方法，还包括：

确定多个偏移对应于所述分片集合中的特定分片；

将所述分片集合中的所述特定分片分成至少两个分片；和

在访问所述至少两个分片中的第二分片之前访问第一分片。

19.根据权利要求14所述的方法，其中，所述用户定义函数在被执行时，至少部分地基于所述分片集合和所述偏移集合向所述外部数据源请求数据，并且所述用户定义函数使用基于所述分片集合的一组并行操作来请求数据。

20.一种计算机存储介质，其包括指令，当由处理设备执行时，所述指令配置所述处理设备来执行操作，所述操作包括：

确定对应于通过电子网络可访问的外部数据源的分片集合，所述外部数据源由与基于网络的数据库系统分离的外部系统托管；

使用流对象确定所述分片集合中每个分片的偏移集合；

识别要对所述分片集合执行的操作，所述操作包括读操作或写操作；和

基于所述分片集合和所述偏移集合，对所述外部数据源执行所述操作。

21.根据权利要求20所述的计算机存储介质，其中所述分片集合的每个分片包括由所述外部数据源存储的数据记录序列。

22.根据权利要求21所述的计算机存储介质，其中确定所述分片集合中的每个分片的偏移集合包括：

执行包括在所述流对象中的用户定义函数，以确定所述偏移集合。

23.根据权利要求22所述的计算机存储介质，其中，所述用户定义函数在被执行时，确定相对于在先前事务期间对应于所述流对象的可扩展流被访问的先前时间的偏移，并且在

所述先前事务完成时预先存储了相对于所述先前时间的偏移。

24.根据权利要求23所述的计算机存储介质,其中所述操作还包括:

确定在所述可扩展流被访问的所述先前时间之后的新偏移的集合。

25.根据权利要求24所述的计算机存储介质,其中所述操作还包括:

确定在所述可扩展流被访问的所述先前时间之后创建了新分片;

确定对应于所述新分片的新偏移;和

提供所述新偏移与所述新偏移的集合。

26.根据权利要求20所述的计算机存储介质,其中所述操作还包括:

确定多个偏移对应于所述分片集合中的特定分片;

将所述分片集合中的所述特定分片分成至少两个分片;和

在访问所述至少两个分片中的第二分片之前访问第一分片。

27.根据权利要求22所述的计算机存储介质,其中,所述用户定义函数在被执行时,至少部分地基于所述分片集合和所述偏移集合向所述外部数据源请求数据。

28.根据权利要求27所述的计算机存储介质,其中所述用户定义函数使用基于所述分片集合的一组并行操作来请求数据。

29.根据权利要求22所述的计算机存储介质,其中所述用户定义函数在被执行时,至少部分地基于来自所述分片集合的特定分片和来自内部数据源的要存储在所述特定分片中的一组行,将数据存储到所述外部数据源中。

30.根据权利要求20所述的计算机存储介质,其中所述流对象包括:

第一函数,其提供存储的偏移的集合;

第二函数,其存储偏移的第二集合;和

第三函数,其提供与外部对象的集成,从而实现与所述外部数据源的通信。

数据源上的可扩展流

[0001] 优先权申请的交叉引用

[0002] 本申请要求2020年3月9日提交的序列号为16/812,940的美国专利申请的优先权，其内容通过引用整体结合于此。

技术领域

[0003] 本公开的实施例一般涉及数据库，更具体地说，涉及利用数据流平台来访问存储在数据库中的数据。

[0004] 背景

[0005] 数据库被广泛用于计算应用中的数据存储和访问。数据库存储的目标是以有组织的方式提供大量信息，以便可以对其进行访问、管理和更新。在数据库中，数据可以组织成行、列和表。各种实体和公司都使用数据库来存储可能需要访问或分析的信息。

[0006] 云数据仓库(也称为“基于网络的数据仓库”或简称为“数据仓库”)是一种类型的基于网络的数据系统，用于数据分析和报告，包括来自一个或更多个不同来源的集成数据的中央储存库。云数据仓库通常是一个在线分析处理(OLAP)数据库，它可以基于存储在企业数据库中的数据，存储可用于为企业创建分析报告的当前和历史数据。为此，数据仓库通常提供商业智能工具、提取、变换数据并将数据加载到储存库中的工具、以及管理和检索元数据的工具。还有其他类型的基于网络的数据系统，如在线事务处理(OLTP)数据库，以及具有多种传统数据库系统特征的数据系统。

[0007] 当要从数据库中提取某些信息时，可以对数据库数据执行查询语句。云数据仓库系统处理该查询，并根据指示该查询应该返回什么信息的一个或更多个查询谓词(query predicates)返回某些数据。数据仓库系统从数据库中提取具体的数据，并将这些数据格式化为可读的形式。但是，在非常大的表上执行查询可能很有挑战性，至少因为需要大量的时间和计算资源来扫描整个表以识别满足查询的数据。

[0008] 附图简述

[0009] 从下面给出的详细描述和本公开的各种实施例的附图中，将更全面地理解本公开。

[0010] 图1示出了根据本公开一些实施例的示例计算环境，其包括与存储平台通信的基于网络的数据仓库系统。

[0011] 图2是示出根据本公开一些实施例的计算服务管理器的组件的框图。

[0012] 图3是示出根据本公开一些实施例的执行平台的组件的框图。

[0013] 图4是示出根据一些示例实施例的用于外部数据流平台的流对象的示例的概念图。

[0014] 图5是示出根据一些示例实施例，使用可扩展流来执行用于从外部数据流平台读取数据的查询的概念图。

[0015] 图6是示出根据一些示例实施例，使用可扩展流来执行用于在外部数据流平台上存储数据的命令的概念图。

[0016] 图7是示出根据本公开的一些实施例，在执行用于从外部数据流平台读取数据的方法时，基于网络的数据仓库系统的操作的流程图。

[0017] 图8是示出根据本公开的一些实施例，在执行用于在外部数据流平台上存储数据的方法时，基于网络的数据仓库系统的操作的流程图。

[0018] 图9示出了根据示例实施例的以计算机系统形式的机器的示意图表示，用于使机器执行本文所讨论方法中的任何一个或更多个的指令集合可以在该计算机系统内执行。

[0019] 详细描述

[0020] 现在将详细参考用于实现本发明主题的具体示例实施例。这些具体实施例的示例在附图中示出，并且具体细节在以下描述中阐述，以便提供对主题的透彻理解。应当理解，这些示例并不旨在将权利要求的范围限制于所示的实施例。相反，它们旨在覆盖可能包含在本公开范围内的替代、修改和等同物。

[0021] 如上所述，处理针对非常大的表的查询是具有挑战性的，因为需要大量的时间和计算资源来扫描整个表以识别满足查询的数据。因此，在不扫描整个表的情况下执行查询可以提高数据仓库系统的性能，并改善返回查询结果的延迟。在一些现有的数据仓库系统中，一种方法是提供对表流(table stream)的支持，在一个示例中，表流可以实现为从表、视图和实例化视图等公开变化数据捕获(Change Data Capture，CDC)信息的对象。这种CDC信息表示数据的显著变化，包括插入、更新和删除，以及关于每个变化的元数据。特别是，单个表流跟踪对源表中的行所做的变化。表流(此处称为“流”)生成“变化表”，该“变化表”具有指示表中两个事务时间点之间在行级的变化的信息。当执行对应于当前事务的查询时，数据仓库系统可以利用该流来确定自先前事务(例如，先前查询)以来的变化。通过这种方式，该流能够以事务方式查询和消费一系列变化记录，这为连续处理新的或变化的数据提供了一种方便的方法。

[0022] 然而，在当今不断扩展的云计算环境中，许多用户可能会在不同的数据源中处理查询(例如，提取、转换数据并将数据加载到新的主机源中，以对数据执行数据分析)，有时会跨不同的存储平台，这些平台可能由第三方托管。在一些现有的数据仓库系统中，虽然可以提供用于数据的流，但是不支持通过流访问外部数据源。本公开的各方面通过有利地使本主题系统能够利用外部数据流平台中提供的数据流，解决了在外部数据源中处理查询的上述和其他挑战。更具体地，如本文的实施例中所述，使用流对象来实现可扩展流，该可扩展流便于响应于查询，以事务方式在外部数据流平台上执行操作(例如，读或写)，并跟踪外部数据源中的变化的增量。

[0023] 图1示出了根据本公开一些实施例的示例计算环境100，其包括与存储平台104通信的基于网络的数据仓库系统102。为了避免用不必要的细节模糊本发明主题，图1中省略了与传达对本发明主题的理解不密切相关的各种功能组件。然而，本领域技术人员将容易认识到，各种附加功能组件可以被包括作为计算环境100的一部分，以促进本文没有具体描述的附加功能。

[0024] 如图所示，计算环境100包括基于网络的数据仓库系统102和存储平台104(例如，AWS®、微软Azure Blob存储®或谷歌云存储®)。基于网络的数据仓库系统102用于报告和分析来自一个或更多个不同源的集成数据，该一个或更多个不同源包括存储平台104内的存储设备106-1至106-N。存储平台104包括多个计算机器，并向基于网络的数据仓库系统

102按需提供计算机系统资源,如数据存储和计算能力。

[0025] 基于网络的数据仓库系统102包括计算服务管理器108、执行平台110和数据库114。基于网络的数据仓库系统102托管并向多个客户端账户提供服务。管理用户可以创建和管理身份(例如,用户、角色和组),并使用权限来允许或拒绝身份对资源和服务的访问。

[0026] 计算服务管理器108协调和管理基于网络的数据仓库系统102的操作。计算服务管理器108还执行查询优化和编译,以及管理提供计算资源的计算服务的集群(也称为“虚拟仓库”)。计算服务管理器108可以支持任意数量的客户端账户,如提供数据存储和检索请求的终端用户、管理本文描述的系统和方法的系统管理员以及与计算服务管理器108交互的其他组件/设备。

[0027] 计算服务管理器还与用户设备112通信。用户设备112对应于由基于网络的数据仓库102支持的多个客户端账户之一的用户。在一些实施例中,计算服务管理器108不从用户设备112接收任何直接通信,并且仅从基于网络的数据仓库系统102内的队列接收关于作业的通信。

[0028] 计算服务管理器108还耦合到数据库114,数据库114与存储在计算环境100中的数据相关联。数据库114存储数据,该数据涉及与基于网络的数据仓库系统102及其用户相关联的各种功能和方面。在一些实施例中,数据库114包括存储在远程数据存储系统中的数据以及可从本地高速缓存获得的数据的概要。另外,数据库114可以包括关于数据如何在远程数据存储系统(例如,存储平台104)和本地高速缓存中组织的信息。数据库114允许系统和服务确定是否需要在不从存储设备加载或访问实际数据的情况下访问一段数据。

[0029] 例如,数据库114可以包括对应于微分区集合的信息。如本文所讨论的,“微分区(micro-partition)”是批量单元,并且每个微分区具有连续的存储单元。举例来说,每个微分区可以包含50MB到500MB之间的未压缩数据(请注意,存储中的实际大小可能较小,因为数据可能会被压缩存储)。表中的行的组可被映射到以列形式组织的个体微分区中。这种大小和结构允许对待扫描的微分区进行极细粒度选择,该微分区可以包括数百万甚至数亿个微分区。这种针对要进行扫描的微分区的粒度选择过程在本文被称为“修剪”。修剪涉及使用元数据来确定表的哪些部分(包括表中的哪些微分区或微分区组)与查询无关,然后在响应查询时避免那些无关的微分区且只扫描相关微分区以对查询进行响应。可以自动收集关于微分区中存储的所有行的元数据,包括:微分区中每个列的值范围;不同值的数量;和/或用于优化和高效查询处理的其他属性。在一个实施例中,可以在所有表上自动执行微分区。例如,可以使用插入/加载数据时发生的排序对表进行透明分区。然而,应当理解,微分区的这种公开仅仅是示例性的,并且应当被认为是非限制性的。应当理解,微分区可以包括其他数据库存储设备,而不脱离本公开的范围。

[0030] 在一些实施例中,计算服务管理器108可以基于来自数据库114的数据来确定应该执行作业。在这样的实施例中,计算服务管理器108可以扫描数据并确定应该执行作业来改善数据组织或数据库性能。例如,计算服务管理器108可以确定源表的新版本已经生成,并且修剪索引没有被刷新以反映源表的新版本。数据库114可以包括事务性变化跟踪流(transactional change tracking stream),该事务性变化跟踪流指示源表的新版本何时被生成以及修剪索引何时被最后刷新。基于该事务流,计算机服务管理器108可以确定应该执行作业。在一些实施例中,计算服务管理器108基于触发事件确定应该执行作业,并将作

业存储在队列中，直到计算服务管理器108准备好调度和管理作业的执行。在本公开的实施例中，计算服务管理器108基于正在执行的一个或更多个DML命令来确定是否需要对表或修剪索引重新聚类，其中该一个或更多个DML命令构成触发事件。

[0031] 计算服务管理器108还耦合到执行平台110，执行平台110提供执行各种数据存储和数据检索任务的多个计算资源。执行平台110耦合到存储平台104的存储平台104。存储平台104包括多个数据存储设备106-1至106-N。在一些实施例中，数据存储设备106-1至106-N是位于一个或更多个地理位置的基于云的存储设备。例如，数据存储设备106-1至106-N可以是公共云基础设施或私有云基础设施的一部分。数据存储设备106-1至106-N可以包括硬盘驱动(HDD)、固态驱动(SSD)、存储集群、亚马孙(Amazon)S3TM存储系统或任何其它数据存储技术。另外，储存平台104可以包括分布式文件系统(例如Hadoop分布式文件系统(HDFS))、对象储存系统等等。

[0032] 执行平台110包括多个计算节点。计算节点上的进程集合执行由计算服务管理器108编译的查询计划。进程集合可以包括：执行查询计划的第一进程；使用最近最少使用(least recently used，LRU)策略来监视和删除微分区文件，并实现内存不足(OOM)错误缓解进程的第二进程；从进程日志和状态提取健康信息以发送回计算服务管理器108的第三进程；在系统引导(boot)之后建立与计算服务管理器108的通信的第四进程；以及第五进程，该第五进程处理与由计算服务管理器108提供的给定作业的计算集群的所有通信，并将信息传送回计算服务管理器108和执行平台110的其他计算节点。

[0033] 在一些实施例中，计算环境100的元件之间的通信链路经由一个或更多个数据通信网络来实现。这些数据通信网络可以利用任何通信协议和任何类型的通信介质。在一些实施例中，数据通信网络是彼此耦合的两个或更多个数据通信网络(或子网络)的组合。在替代实施例中，使用任何类型的通信介质和任何通信协议来实现这些通信链路。

[0034] 如图1所示，数据存储设备106-1至106-N从与执行平台110相关联的计算资源解耦。该架构支持基于网络的数据仓库系统102的动态变化，该动态变化基于变化的数据存储/检索需求以及变化的用户和系统需求。对动态变化的支持允许基于网络的数据仓库系统102响应于对基于网络的数据仓库系统102内的系统和组件的不断变化的需求而快速变化规模(scale)。计算资源与数据存储设备的分离支持大量数据的存储，而无需相应的大量计算资源。类似地，资源的这种分离支持在特定时间使用的计算资源的显著增加，而无需相应增加可用数据存储资源。

[0035] 计算服务管理器108、数据库114、执行平台110、存储平台104以及认证和身份管理系统118在图1中显示为单独的分立组件。然而，计算服务管理器108、数据库114、执行平台110和存储平台104中的每一个都可以被实现为分布式系统(例如，分布在多个地理位置处的多个系统/平台上)。另外，计算服务管理器108、数据库114、执行平台110和存储平台104中的每一个可以根据所接收请求的变化和基于网络的数据仓库系统102的变化的需求而(彼此独立地)使其规模放大或缩小。因此，在所描述的实施例中，基于网络的数据仓库系统102是动态的，并且支持频繁变化以满足当前的数据处理需求。

[0036] 在典型操作期间，基于网络的数据仓库系统102处理由计算服务管理器108确定的多个作业。这些作业由计算服务管理器108调度和管理，以确定何时以及如何执行该作业。例如，计算服务管理器108可以将作业划分为多个离散任务，并且可以确定需要什么数据来

执行多个离散任务中的每个离散任务。计算服务管理器108可以将多个离散任务中的每个离散任务分配给执行平台110的一个或更多个节点以处理任务。计算服务管理器108可以确定需要哪些数据来处理任务，并且进一步确定执行平台110内的哪些节点最适合于处理该任务。一些节点可能已经高速缓存了处理任务所需的数据，因此是用于处理任务的不错的候选者。存储在数据库114中的元数据帮助计算服务管理器108确定执行平台110中的哪些节点已经缓存了处理该任务所需的至少一部分数据。执行平台110中的一个或更多个节点使用由节点高速缓存的数据以及必要时从存储平台104检索到的数据来处理任务。期望从执行平台110内的高速缓存中检索尽可能多的数据，因为检索速度通常比从存储平台104中检索数据快得多。

[0037] 如图1所示，计算环境100将执行平台110与存储平台104分开。在这种布置中，执行平台110中的处理资源和高速缓存资源独立于存储平台104中的数据存储设备106-1至106-N进行操作。因此，计算资源和高速缓存资源不限于具体的数据存储设备106-1至106-N。相反，所有计算资源和所有高速缓存资源可以从存储平台104中的任何数据存储资源检索数据，并将数据存储到其中。

[0038] 如进一步所示，基于网络的数据仓库系统102能够与外部数据流平台120通信。在该示例中，外部数据流平台可由第三方提供，并且可由基于网络的数据仓库系统102的一个或更多个组件(例如，计算服务管理器108和/或执行平台110)访问。这种数据流平台的一些例子包括Apache Flink®、Apache Kafka®、Amazon Kinesis®、Apache Pulsar®等。这种外部数据流平台提供外部数据源，然后基于网络的数据仓库系统102的组件可以访问这些外部数据源。然而，应当理解，本主题技术也可以实现利用内部数据源用于流的方面。

[0039] 如本文所述，流对象跟踪对表进行的数据操作语言(DML)变化，包括插入、更新和删除，以及关于每个变化的元数据，以便可以使用变化的数据采取行动。这个过程被称为变化数据捕获(CDC)。各个表流跟踪对源表中行所做的变化。如前所述，流提供了“变化表”，其指示在表中的两个事务时间点之间的行级的这种变化，从而能够以事务方式查询和消费一系列变化记录。

[0040] 在一个实施例中，流在源表的事务性版本化时间线中保持一个时间点，称为偏移，它在流内容被最后使用DML语句消费时的事务性点开始。该流可以提供从源表的当前偏移到当前事务时间(例如，表的当前版本)的一组变化。在一个例子中，流保持变化的增量；如果多个DML语句更改了一行，则该流只包含对该行采取的最新动作。在一个实施例中，当在事务中使用时，偏移前进(例如，更新)。

[0041] 图2是示出根据本公开一些实施例的计算服务管理器108的组件的框图。如图2所示，计算服务管理器108包括耦合到数据存储设备206的访问管理器202和密钥管理器204。访问管理器202处理针对本文描述的系统的认证和授权任务。密钥管理器204管理在认证和授权任务期间使用的密钥的存储和认证。例如，访问管理器202和密钥管理器204管理用于访问存储在远程存储设备(例如，存储平台104中的数据存储设备)中的数据的密钥。如本文所用，远程存储设备也可以被称为“永久存储设备”或“共享存储设备”。另外，访问管理器202处理流对象的授权和认证任务，如这里进一步讨论的。

[0042] 请求处理服务208管理接收到的数据存储请求和数据检索请求(例如，将对数据库数据执行的作业)。例如，请求处理服务208可以确定处理接收到的查询(例如，数据存储请

求或数据检索请求)所需的数据。数据可以存储在执行平台110内的高速缓存中,或者存储在存储平台104中的数据存储设备中。

[0043] 管理控制台服务210支持管理员和其他系统管理员对各种系统和进程的访问。另外,管理控制台服务210可以接收执行作业并监视系统上的工作负荷的请求。

[0044] 计算服务管理器108还包括作业编译器212、作业优化器214和作业执行器216。作业编译器212将作业解析为多个离散任务,并为多个离散任务中的每个离散任务生成执行代码。作业优化器214基于需要处理的数据来确定执行多个离散任务的最佳方法。作业优化器214还处理各种数据修剪操作和其他数据优化技术,以提高执行作业的速度和效率。作业执行器216执行从队列接收的或由计算服务管理器108确定的作业的执行代码。

[0045] 作业调度器和协调器218将接收到的作业发送到适当的服务或系统,以进行编译、优化并分派到执行平台110。例如,可以按该优先顺序对作业进行优先考虑和处理。在实施例中,作业调度器和协调器218确定由计算服务管理器108与其他“外部”作业(诸如可以由数据库中的其他系统调度但可以利用执行平台110中的相同处理资源的用户查询)调度的内部作业的优先级。在一些实施例中,作业调度器和协调器218识别或分配执行平台110中的特定节点以处理特定任务。虚拟仓库管理器220管理在执行平台110中实现的多个虚拟仓库的操作。如下所述,每个虚拟仓库都包括多个执行节点,每个执行节点都包括一个高速缓存和一个处理器。

[0046] 另外,计算服务管理器108包括配置和元数据管理器222,配置和元数据管理器222管理与存储在远程数据存储设备和本地高速缓存(例如,执行平台110中的高速缓存)中的数据相关的信息。配置和元数据管理器222使用元数据来确定需要访问哪些数据微分区来检索用于处理特定任务或作业的数据。监视器和工作负载分析器224监管由计算服务管理器108执行的进程,并管理任务(例如,工作负载)在执行平台110中的执行节点和虚拟仓库之间的分配。监视器和工作负载分析器224还根据需要,基于整个数据仓库系统102中不断变化的工作负载来重新分配任务,并且还可以基于也可以由执行平台110处理的用户(例如,“外部”)查询工作负载来重新分配任务。配置和元数据管理器222以及监视器和工作负荷分析器224耦合到数据存储设备226。图2中的数据存储设备226表示数据仓库系统102中的任何数据存储设备。例如,数据存储设备226可以表示执行平台110中的高速缓存、存储平台104中的存储设备或任何其他存储设备。

[0047] 如图所示,计算服务管理器108还包括流处理引擎228。在一个实施例中,流处理引擎228负责生成和管理流,这些流被实现为对象,在一个示例中,这些对象公开来自表、视图、实例化视图(例如,包含相应查询的结果的数据库对象)和/或外部表的变化数据捕获(CDC)信息。此外,流处理引擎228被配置为支持例如外部数据流平台120的外部源上的流。关于使用外部源处理流的进一步细节将在下面进一步讨论。

[0048] 图3是示出根据本公开一些实施例的执行平台110的组件的框图。如图3所示,执行平台110包括多个虚拟仓库,其包括虚拟仓库1、虚拟仓库2和虚拟仓库n。每个虚拟仓库都包括多个执行节点,每个执行节点都包括一个数据高速缓存和一个处理器。虚拟仓库可以通过使用多个执行节点并行执行多个任务。如本文所讨论的,执行平台110可以基于系统和用户的当前处理需求实时添加新的虚拟仓库和删除现有的虚拟仓库。这种灵活性允许执行平台110在需要时快速部署大量计算资源,而不必在不再需要它们时继续为那些计算资源付

费。所有虚拟仓库都可以访问任何数据存储设备(例如,存储平台104中的任何存储设备)中的数据。

[0049] 尽管图3中所示的每个虚拟仓库包括三个执行节点,但是特定的虚拟仓库可以包括任意数量的执行节点。此外,虚拟仓库中执行节点的数量是动态的,使得在存在额外需求时创建新的执行节点,并在不再需要现有执行节点时将其删除。

[0050] 每个虚拟仓库能够访问图1中所示的数据存储设备106-1至106-N中的任何一个。因此,虚拟仓库没有必要被分配给特定的数据存储设备124-1到124-n,而是可以从存储平台104内的数据存储设备106-1至106-N中的任何一个访问数据。类似地,图3中所示的每个执行节点可以从数据存储设备106-1至106-N中的任何一个访问数据。在一些实施例中,特定的虚拟仓库或特定的执行节点可以被临时分配给具体的数据存储设备,但是该虚拟仓库或执行节点稍后可以从任何其他数据存储设备访问数据。

[0051] 在图3的示例中,虚拟仓库1包括三个执行节点302-1、302-2和302-n。执行节点302-1包括高速缓存304-1和处理器306-1。执行节点302-2包括高速缓存304-2和处理器306-2。执行节点302-n包括高速缓存304-n和处理器306-n。每个执行节点302-1、302-2和302-n与处理一个或更多个数据存储和/或数据检索任务相关联。例如,虚拟仓库可以处理与内部服务(诸如,聚类服务、实例化视图刷新服务、文件压缩服务、存储程序服务或文件升级服务)相关联的数据存储和数据检索任务。在其他实施方式中,特定虚拟仓库可以处理与特定数据存储系统或特定类别的数据相关联的数据存储和数据检索任务。

[0052] 类似于上面讨论的虚拟仓库1,虚拟仓库2包括三个执行节点312-1、312-2和312-n。执行节点312-1包括高速缓存314-1和处理器316-1。执行节点312-2包括高速缓存314-2和处理器316-2。执行节点312-n包括高速缓存314-n和处理器316-n。另外,虚拟仓库3包括三个执行节点322-1、322-2和322-n。执行节点322-1包括高速缓存324-1和处理器326-1。执行节点322-2包括高速缓存324-2和处理器326-2。执行节点322-n包括高速缓存324-n和处理器326-n。

[0053] 在一些实施例中,相对于执行节点正在高速缓存的数据,图3所示的执行节点是无状态的。例如,这些执行节点不存储或以其他方式维护关于执行节点或由特定执行节点缓存的数据的状态信息。因此,在执行节点发生故障的情况下,可以用另一个节点透明地替换发生故障的节点。由于没有与发生故障的执行节点相关联的状态信息,因此新的(替换)执行节点可以轻松替换发生故障的节点,而无需考虑重新创建特定状态。

[0054] 尽管图3中所示的执行节点包括一个数据高速缓存和一个处理器,但是替代实施例可以包括包含任意数量的处理器和任意数量的高速缓存的执行节点。另外,高速缓存的大小在不同执行节点之间可能有所不同。图3中所示的高速缓存在本地执行节点中存储从存储平台104中的一个或更多个数据存储设备检索到的数据。因此,高速缓存减小或消除了在从远程存储系统中不断检索数据的平台中出现的瓶颈问题。代替从远程存储设备重复访问数据,本文描述的系统和方法从执行节点中的高速缓存访问数据,这明显更快并且避免了上面讨论的瓶颈问题。在一些实施例中,使用提供对高速缓存的数据的快速访问的高速存储器设备来实现高速缓存。每个高速缓存可以存储来自存储平台104中的任何存储设备的数据。

[0055] 此外,高速缓存资源和计算资源可以在不同的执行节点之间变化。例如,一个执行

节点可包含大量的计算资源和最少的高速缓存资源，从而使该执行节点可用于需要大量计算资源的任务。另一个执行节点可包含大量的高速缓存资源和最少的计算资源，从而使该执行节点可用于需要高速缓存大量数据的任务。又一个执行节点可包含提供更快的输入-输出操作的高速缓存资源，这对于需要快速扫描大量数据的任务很有用。在一些实施例中，基于执行节点将要执行的预期任务，在创建执行节点时确定与特定执行节点相关联的高速缓存资源和计算资源。

[0056] 另外，与特定执行节点相关联的高速缓存资源和计算资源可以基于由执行节点执行的改变的任务而随时间改变。例如，如果执行节点执行的任务变得更加处理器密集，则可以为执行节点分配更多的处理资源。类似地，如果执行节点执行的任务需要更大的高速缓存容量，则可以为执行节点分配更多的高速缓存资源。

[0057] 尽管虚拟仓库1、2和n与同一执行平台110相关联，但是可以使用在多个地理位置处的多个计算系统来实现虚拟仓库。例如，虚拟仓库1可以由第一地理位置处的计算系统来实现，而虚拟仓库2和n由第二地理位置处的另一计算系统来实现。在一些实施例中，这些不同的计算系统是由一个或更多个不同的实体维护的基于云的计算系统。

[0058] 另外，每个虚拟仓库在图3中被示为具有多个执行节点。可以使用位于多个地理位置处的多个计算系统来实现与每个虚拟仓库相关联的多个执行节点。例如，虚拟仓库1的实例在一个地理位置处的一个计算平台上实现执行节点302-1和302-2，并且在另一个地理位置处的不同计算平台上实现执行节点302-n。选择特定的计算系统以实施执行节点可以取决于各种因素，诸如特定的执行节点所需的资源水平(例如，处理资源要求和高速缓存要求)、特定计算系统处可用的资源、地理位置内或地理位置之间的网络的通信能力以及哪些计算系统已在虚拟仓库中实现其他执行节点。

[0059] 执行平台110也是容错的。例如，如果一个虚拟仓库发生故障，那么该虚拟仓库将很快被位于不同地理位置处的不同虚拟仓库替换。

[0060] 特定执行平台110可以包括任意数量的虚拟仓库。另外，特定执行平台中虚拟仓库的数量是动态的，使得在需要附加的处理和/或高速缓存资源时创建新的虚拟仓库。类似地，当不再需要与虚拟仓库关联的资源时，可以删除现有的虚拟仓库。

[0061] 在一些实施例中，虚拟仓库可以在存储平台104中的相同数据上操作，但是每个虚拟仓库具有其自己的具有独立处理和高速缓存资源的执行节点。此配置允许对不同虚拟仓库上的请求进行独立处理，并且请求之间不会相互干扰。这种独立的处理与动态添加和移除虚拟仓库的能力相结合，支持为新用户添加新的处理能力，而不会影响现有用户观察到的性能。

[0062] 根据一些实施例，以下讨论涉及生成可扩展流。为了提供与诸如外部数据流平台120之类的外部源的互操作性，如下面进一步讨论的，流处理引擎228被配置成生成可扩展流，该可扩展流用于响应外部数据流平台120上的查询。如下面进一步讨论的，流对象可以用来表示可扩展流，并提供用于在外部数据源(例如外部数据流平台120)上执行操作的函数和/或方法。

[0063] 在一个实施例中，以下示例语法(例如伪代码语句)使得流处理引擎228能够生成可扩展流并使用该可扩展流执行附加操作：

[0064] CREATE[OR REPLACE]EXTENSIBLE STREAM STREAM_NAME

[0065] LOAD_OFFSETS＝load_offset_function(streamObj,action)

[0066] LOAD_DATA＝load_data_function(streamObj,partition_offset_pair,rowwriter)

[0067] STORE_DATA＝store_data_function(streamObject,rows,shard)

[0068] [USING EXTERNAL INTEGRATION MY_EXTERNAL_INTEGRATION]

[0069] 应当理解,上述示例语法是一个示例,并且在另一个示例中,可以由实现特定接口协定的JAVA类来表示,该特定接口协定对应于以下数据协定:

[0070] ●可扩展流实现能够从流数据源中检索概念偏移信息。在Kafka®主题的情况下,这是分区/偏移对集合,表示主题中上次检索信息的点。此外,流的用户可以用下面说明的AT子句指定该偏移信息。

[0071] ●可扩展流实现必须能够使用该偏移信息,并使用偏移信息从流数据源检索数据。除了流化数据之外,还必须返回新的偏移信息,该信息指示流是从多“远”的地方被读取的。可扩展流实现必须能够将数据写入流数据源,并返回该操作的成功或失败。

[0072] 在一些实施例中,为了给流提供进一步的可扩展性,可以提供提供实现接口的类的JAVA JAR(用于聚集文件的JAVA归档),或者具有带适当方法的类的Python®库,并将其与可扩展流相关联。

[0073] 在上述示例语法中,“加载偏移(LOAD_OFFSETS)”对应于用户定义的函数(例如,JavaScript®UDF或JS-UDF),该函数被调用来从流对象加载偏移信息以及在外部数据流平台120上执行分区发现以确定新的偏移。在一个例子中,“LOAD_OFFSETS”函数接受表示流的对象(例如,JavaScript®对象),以及作为字符串的动作(例如,“STORE_DATA”或“LOAD_DATA”)。此外,前述的JavaScript®对象提供了将当前偏移集合提供为字符串的函数(例如,“getStreamOffsets”),该字符串可以实现为blob数据(例如JavaScript®对象标记(JSON)blob)、逗号分隔的字符串等。在一个示例中,从“getStreamOffsets”函数返回的偏移集合可以由用户指定,或者对应于相对于使用流执行读操作的先前时间(例如,最后一次)的偏移,并且在读取流之后存储与流相关的信息(例如,更新最后读取偏移等)。在一个实施例中,上述函数可以被配置为提供新的偏移,而不是当前的偏移集合。

[0074] 在另一个实施例中,流处理引擎228可以确定偏移的范围并将偏移分成单独的集合。例如,当流处理引擎228发现分区和值有为零(0)的偏移,但是存在值为一百(100)的高水位线(例如,成功复制到所有日志副本的最后消息的偏移)的指示时,流处理引擎228可以将该分区分成两个(逻辑)集合,第一集合覆盖从零(0)到五十(50)的偏移,第二集合覆盖从五十一(51)到一百(100)的偏移。如这里所讨论的,特定分区可以与特定偏移相关联,从而形成用于执行流的分区/偏移对。如这里所提到的,流的执行对应于在流上执行事务。在一个例子中,当事务完成时,偏移可以前进以反映来自流的最后读取偏移。

[0075] 在另一个例子中,如上所述,当加载数据时,流处理引擎228可以不处理每个分区/偏移对中的所有50个可用偏移。在这种情况下,流处理引擎228在成功加载数据之后存储两个条目,其中每个条目具有相同的分区,但是具有表示开始和结束偏移的不同的偏移范围(例如,35-50和85-100)。流对象表现为“无序”流,因为当从流对象读取时返回的记录相对

于外部数据流平台120中的逻辑排序可能是无序的。在一个示例中，流处理引擎228可以聚合两个分区/偏移对，使得新的一个生成自分区[35,50,85-100]，其旨在供一个并行单元进行处理，因为该分区要处理的偏移总数相对较低。

[0076] 在另一个例子中，流处理引擎228确定在外部数据流平台120上创建了新的分区，或者在外部数据流平台120上合并了两个分区。在这种情况下，如果创建了新的分区，则流处理引擎228生成新的分区/偏移对，或者如果合并了两个分区，则流处理引擎228将两个分区聚合成单个分区。

[0077] 同样如上所述，“LOAD_DATA”函数对应于从外部数据流平台120读取数据的函数。在一个实施例中，流处理引擎228可以并行执行N个该函数，其中N是通过加载分区/偏移对来确定的。“LOAD_DATA”函数接收流对象和分片(shard)/偏移字符串，该字符串可在函数中用于进行外部调用。例如，这个分片/偏移字符串可以是Kafka®分区/偏移对，或者是Kinesis®分片/序列号。“LOAD_DATA”函数在执行时提供行集(例如，可用作表或视图的对象)，其中该行集的维度可以基于用户配置。此外，“LOAD_DATA”函数在执行时提供一个字符串，该字符串指示最后读取的分区/偏移对以及其他类型的信息。例如，这可以是为分区(例如，Kafka®)处理的最后一个偏移，或者可以是序列号(例如，Kinesis®)。

[0078] 如本文所用，术语“分片”是指数据库中的数据的分区。例如，在一个实现中，分片可以对应于存储在给定数据库系统中的数据的水平分区。在一些数据库系统中，根据特定的数据库系统实现(例如Kafka®)，这种分片可以被称为分区。此外，给定的分片可以包括一系列数据记录。

[0079] 同样如上所述，流处理引擎228执行“STORE_DATA”函数以将数据存储在外部数据流平台120中。在一个实施例中，“STORE_DATA”函数接收流对象、从存储平台104写入哪些行的指示、以及可选地指示特定分片将被写入外部数据流平台120的分片。当被执行时，该函数返回一个布尔值，该值指示是否所有行都被成功存储在外部数据流平台120中。

[0080] 同样如上所述，“USING EXTERNAL INTEGRATION”代表可选的外部集成，其可用于初始化到外部数据流平台120的连接，其可包括不同于偏移信息的安全和配置信息。示例配置信息可以包括重试超时、要处理的最大偏移量、请求超时等。

[0081] 在一个实施例中，使用字符串“EXTERNAL”来表示该集成旨在与外部源一起使用，并且使用“USER_DEFINED”类型来表示用户负责定义属性。两个保留属性包括ENABLED和SECRET_PROPERTIES。所有给定的选项都作为键值对传入和存储。下面的伪代码就是一个例子：

[0082] CREATE EXTERNAL INTEGRATION MY_INTEGRATION

[0083] TYPE＝USER_DEFINED

[0084] ENABLED＝{TRUE|FALSE}

[0085] SECRET_PROPERTIES＝[PROP1,PROP2]

[0086] PROP1＝VAL1

[0087] PROP2＝VAL2

[0088] PROPN＝VAL2

[0089] 在上述内容中，ENABLED指的是在使用之前必须启用集成的要求。如果ENABLED设

置为假，则该集成不可用，并且在到达任何执行节点之前，从流中读取会失败。

[0090] 在上文中，SECRET_PROPERTIES是秘密和/或敏感属性的列表。例如，OAuth客户端信息存储在SECRET_PROPERTIES中。SECRET_PROPERTIES是加密存储的。该列表中的任何属性都将被安全地存储，并且基于网络的数据仓库系统102在描述集成时仅显示键名和修订的属性值。

[0091] 在一个实施例中，基于网络的数据仓库系统102实现“CREDENTIALS”对象，该对象接受单个字符串(例如，JSON blob或其他blob数据)或一组属性，用于实现可扩展流的外部集成。

[0092] 图4是示出根据一些示例实施例的用于与外部数据流平台(例如，外部数据流平台120)一起使用的流对象410的示例的概念图。

[0093] 如图所示，流对象410包括方法450和对应于偏移452、偏移修改454和外部集成456的一组属性。在一个实施例中，流对象410可以被实现为表示流的JavaScript®对象，并且包括以下方法450：

[0094] getOffsets()--以字符串形式返回当前存储的当前偏移集

[0095] storeOffsets(newOffsetInfo)-接收表示偏移的字符串，并将偏移持久化到流中

[0096] getExternalIntegration()-提供外部集成(例如，指示用于连接到外部数据流平台120的安全性的信息和/或配置信息)

[0097] 出于描述可扩展流的目的，流对象410被示为包括如图4所示的一组属性。然而，应当理解，在一些实施例中，这样的属性可以存储在基于网络的数据仓库系统102中的其他地方，例如存储平台104和/或数据库114中，并且与流对象410相关联。

[0098] 如进一步所示，在一个实施例中，流对象410包括一组属性。例如，偏移452对应于在流上设置的存储的偏移字符串。此外，offsets_modified_on 454对应于流偏移被修改的最后时间。每次更新流上的偏移时，都会更新此属性。外部集成456对应于流对象410使用的外部集成的名称(如果设置的话)。虽然未示出，但是在一个示例中，提供指示流对象410的版本的属性来区分不同的流对象，其中每当使用外部数据流平台120完成一组成功的操作(例如，读取或写入)时，版本可以递增。

[0099] 在一个实施例中，可以修改流对象410的一个或更多个属性。例如，可以更新外部集成456，和/或可以重写偏移452。在一个实现中，偏移452可以仅在没有流的并发执行时被更新。

[0100] 图5是示出根据一些示例实施例，使用可扩展流来执行用于从外部数据流平台(例如，外部数据流平台120)读取数据的查询的概念图。

[0101] 以下是在可扩展流上执行读操作的示例伪代码：

[0102] SELECT...

[0103] FROM STREAM_NAME...

[0104] {

[0105] <AT({OFFSET=><time_difference>})>

[0106] <USING EXTERNAL INTEGRATION INTEGRATION_NAME>

[0107] }

[0108] 在上述示例中，“AT”子句是可选的，因为该流使用该流被消耗时(例如，之前)前进

的最后一个偏移。“USING”集成子句也是可选的，因为集成可以在创建流时与流相关联。

[0109] 如图所示，计算服务管理器108的请求处理服务208接收查询510，其分析查询510并确定流处理引擎228应该处理查询510，因为它涉及对可扩展流执行操作(例如，SELECT* from STREAM STREAM)。在该示例中，查询510包括对应于可扩展(外部)流(例如，在外部数据流平台120上)的字符串。

[0110] 在一个实施例中，在向流处理引擎228发送查询510之前，访问管理器202执行授权检查，以确定提交查询的相应用户或客户端具有执行查询510的授权或足够特权。

[0111] 在被授权之后，流处理引擎228确定流以及与查询510相关联的安全集成对象。在一个示例中，利用安全集成对象来生成访问令牌，以使用户能够访问基于网络的数据仓库系统102。

[0112] 流处理引擎228使用流对象来确定当前分片(或在一些实施例中的分区)/偏移512。如前所述，“LOAD_OFFSETS”函数接受表示流的对象(例如，JavaScript®对象)，其中JavaScript®对象提供函数(例如，“getStreamOffsets”)，该函数以字符串形式提供当前偏移集合。在一个示例中，从“getStreamOffsets”函数返回的偏移集合可以由用户指定，或者对应于使用流对象执行读操作的先前时间(例如，最后一次)的偏移，并且与流对象相关的信息被存储以反映该读操作的结果。因此，该函数确定相对于在先前事务期间访问对应于流对象的可扩展流的先前时间的偏移，并且在先前事务完成时预先存储了相对于先前时间的偏移。此外，前面提到的“getStreamOffsets”函数可以配置为提供新的偏移，而不是当前的偏移集合。

[0113] 使用当前分片/偏移512，流处理引擎228然后调用函数514(例如，“LOAD_OFFSETS”)来在外部数据流平台120上加载偏移，以确定用于加载(例如，读入数据仓库系统)的分区/偏移对集合。在一个示例中，集合的大小可以确定用于执行这种偏移的加载的期望并行度(DoP)。外部数据流平台120将该分区/偏移对集合返回给流处理引擎228。

[0114] 在一个实施例中，流处理引擎228有利地以动态和可伸缩的方式执行分区/偏移对集合的加载。例如，本主题系统能够分配计算资源，并根据总体负载按需增加/缩小这些资源以处理分区/偏移加载。在查询时，本主题系统可以动态地执行前述操作，使得资源不是固定的。

[0115] 流处理引擎228生成查询计划(例如，用于访问数据的有序步骤集)和/或规范和描述语言(SDL)，其基于上述分区集合的大小向执行平台110指示所请求的DoP。此外，流处理引擎228可以在查询计划中包括串行化流和安全集成对象，以使用该对象启用后续函数。

[0116] 在该示例中，执行平台110处理所请求的DoP，并基于虚拟仓库(例如，如图所示的“虚拟仓库1”)的大小，从执行节点302-1、302-2和302-n启动一组执行节点。在一个实施例中，执行平台110可以相应地“时隙化(slot)”分区(例如，每个处理器一个分区)。在一个示例中，执行平台为每个处理器提供多个分区的映射。在这种情况下，这被理解为DoP降级，并且执行平台110然后启用分区窃取，使得如果处理器完成了对其分区的处理，并且处理器确定其他节点具有未处理的分区，则处理器将从尚未处理该分区的另一个节点获取一个分区。在多核处理器的情况下，上述实现也可以在每个核的基础上应用，其中这种分区窃取是在多核处理器的核级实现的。

[0117] 来自执行节点302-1、302-2和302-n的该组执行节点然后调用外部函数(例如，LOAD_DATA函数)以相应地从外部数据流平台120检索数据。该组执行节点将从外部数据流平台120返回的行发送到第二组执行节点302-X、302-Y和302-Z，然后第二组执行节点将返回的行存储在存储平台104中。出于说明的目的，在第二组执行节点中示出了三个执行节点，但是应当理解，基于虚拟仓库的配置和/或容量，可以在该第二组中使用更多或更少的执行节点。

[0118] 在这个阶段，查询510成功完成。执行平台110在第二组执行节点302-X、302-Y和302-Z上执行“store offsets”函数，并将对应于新偏移530的信息作为用于存储在文件520中的字符串发送到存储平台104，文件520可以对应于由存储平台104提供的相应微分区。在一个实施例中，当提交事务时，执行平台110将字符串作为事务的一部分存储在流上。可选地，新偏移530被存储在具有新字段的现有流偏移中，例如“external0ffset”。

[0119] 图6是示出根据一些示例实施例，使用可扩展流来执行用于在外部数据流平台(例如，外部数据流平台120)上存储数据的查询的概念图。

[0120] 如图所示，命令610由计算服务管理器108的请求处理服务208接收，其分析命令610并确定流处理引擎228应该处理命令610，因为它涉及对可扩展流执行操作(例如，INSERT INTO EXTENSIBLE_EXTERNAL_STREAM_NAME as SELECT*from MY_TABLE)。在该示例中，命令610包括对应于可扩展流的字符串和存储在存储平台104中的表的名称。

[0121] 在一个实施例中，在向流处理引擎228发送命令610之前，访问管理器202执行授权检查，以确定提交查询的相应用户或客户端具有执行命令610的授权或足够特权。在被授权之后，流处理引擎228确定流以及与命令610相关联的安全集成对象612。

[0122] 流处理引擎228使用流对象，执行LOAD_OFFSETS函数614来确定(例如，在外部数据流平台120上的)目的分片集合。外部数据流平台120将该目的分片集合例如，分区)返回给流处理引擎228。因为DoP可以由目的分片集合中的分片数量来确定。这使得执行平台110能够并行化存储在外部数据流平台120上的数据(当被外部数据流平台120支持时)。给定N个分片，执行平台110可以在N个执行节点中分配一组正被写入的行。

[0123] 流处理引擎228生成查询计划(例如，用于访问数据的有序步骤集)和/或规范和描述语言(SDL)，该查询计划基于上述目的分片集合的大小向执行平台110指示所请求的DoP。此外，流处理引擎228可以在查询计划中包括串行化流和安全集成对象，以使用该对象启用后续函数。

[0124] 在该示例中，执行平台110处理所请求的DoP，并基于虚拟仓库(例如，如图所示的“虚拟仓库1”)的大小，从执行节点302-1、302-2和302-n启动一组执行节点。基于命令610和/或前述查询计划，来自执行节点302-1、302-2和302-n的一组执行节点从存储平台104读取数据(例如，来自表的行)。执行节点302-1、302-2和302-n将数据转发到第二组执行节点302-X、302-Y和302-Z，它们负责将数据存储到外部数据流平台120。出于说明的目的，在第二组执行节点中示出了三个执行节点，但是应当理解，基于虚拟仓库的配置和/或容量，可以在该第二组中使用更多或更少的执行节点。

[0125] 第二组执行节点302-X、302-Y和302-Z随后为每个分片调用外部函数(例如，STORE_DATA函数)，其在执行LOAD_OFFSETS函数614期间确定，以相应地将数据存储到外部数据流平台120。如前所述，当被执行时，STORE_DATA函数返回一个布尔值，指示是否所有行

都被成功存储在外部数据流平台120中。如果布尔值为真，在此阶段，命令610成功完成，并且事务被记录为在存储平台104中提交。在一个示例中，如果(例如，相反地，通过假值被返回)指示存储操作失败，例如，此时外部数据流平台120当前不可用，则可以回滚命令610，并且在后续时间重试命令610将是有益的。

[0126] 尽管图5和图6中描述的上述示例涉及外部数据流平台(例如，具有外部数据源)，但是本主题技术可以应用于内部数据源，以有利地为这样的内部数据源提供流。在一个实施例中，用户可以定义基于网络的数据仓库系统102内提供的数据上的流，例如查询历史。在这个例子中，可以从访问(例如读取)查询历史的最后时间存储时间戳，并且可以利用流中的时间戳运行查询历史，然后处理结果行，并且提交(例如存储)查询历史中看到的最后时间，这被用作后续时间的偏移。在这个例子中，系统因此可以放弃为每个内部目录实体定义新的流，从而使用户能够执行这样的任务(例如，定义流)。

[0127] 图7是示出根据本公开的一些实施例，在执行用于从外部数据流平台读取数据的方法700时，基于网络的数据仓库系统102的操作的流程图。方法700可以体现为由一个或更多个硬件组件(例如，一个或更多个处理器)执行的计算机可读指令，使得方法700的操作可以由基于网络的数据仓库系统102的组件来执行。相应地，方法700在下面通过参考其示例来描述。然而，应当理解，方法700可以部署在各种其他硬件配置上，并且不旨在限于部署在基于网络的数据仓库系统102内。

[0128] 在操作702，流处理引擎228接收在外部数据流平台(例如，外部数据流平台120)上读取数据的查询。在一个示例中，如前所述，查询可以包括用于从外部数据流平台120读取数据的SELECT语句，其还指定可扩展流和外部集成(例如，提供用于连接和访问外部数据流平台120的信息)。

[0129] 在操作704，流处理引擎228至少部分基于查询(例如，指定的可扩展流)来确定分片和偏移的当前集合。在一个示例中，分片和偏移的当前集合是以各自分片/偏移对的形式，其中特定分片可以与来自偏移范围的特定当前偏移相关联。在本例中，当前偏移可以对应于从可扩展流中读取的最后一个偏移。因此，流处理引擎228可以确定从该最后偏移到由外部数据流平台120提供的源表的当前事务时间的变化。

[0130] 在操作706，流处理引擎228基于分片和偏移的当前集合生成具有所请求的并行度(DoP)的查询计划(例如，用于完成查询的一组任务)。如前所述，所请求的DoP可以基于对应于分片和偏移的当前集合的分片的数量。

[0131] 在操作708，执行平台110基于分片和偏移的当前集合，在外部数据流平台120上执行查询。如前所述，执行平台110处理所请求的DoP，并基于虚拟仓库的大小启动一组执行节点。在一个实施例中，执行平台110可以相应地“时隙化”分片(例如，每个处理器一个分片)。该组执行节点调用外部函数(例如，LOAD_DATA函数)来相应地从外部数据流平台120检索数据。

[0132] 在操作710，执行平台110从外部数据流平台120接收数据。该组执行节点将从外部数据流平台120返回的行发送到第二组执行节点，然后第二组执行节点将返回的行存储在存储平台104中。

[0133] 在操作712，执行平台110指示查询完成。在一个实施例中，执行平台110在第二组执行节点上执行“store offsets”函数，并将该信息作为存储在存储平台104中的字符串发

送到存储平台104。在一个实施例中，当提交事务时，执行平台110将字符串作为事务的一部分存储在流上。

[0134] 图8是示出根据本公开的一些实施例，在执行用于在外部数据流平台上存储数据的方法800时，基于网络的数据仓库系统102的操作的流程图。方法800可以体现为由一个或更多个硬件组件(例如，一个或更多个处理器)执行的计算机可读指令，使得方法800的操作可以由基于网络的数据仓库系统102的组件来执行。相应地，方法800在下面通过参考其示例来描述。然而，应当理解，方法800可以部署在各种其他硬件配置上，并且不旨在限于部署在基于网络的数据仓库系统102内。

[0135] 在操作802，流处理引擎228接收用于在外部数据流平台(例如，外部数据流平台120)上存储数据的命令。在一个示例中，该命令可以包括查询语句，该查询语句具有INSERT命令和对应于存储在存储平台104中用于读取要存储的数据的表的字符串。

[0136] 在操作804，流处理引擎228确定外部数据流平台120上的目的分片集合。如前所述，流处理引擎228使用流对象，执行LOAD_OFFSETS函数614来确定(例如，在外部数据流平台120上的)目的分片集合。外部数据流平台120将该目的分片集合(例如，分区)返回给流处理引擎228。

[0137] 在操作806，流处理引擎228基于该目的分片集合生成具有所请求的并行度(DoP)的查询计划(例如，用于完成命令的一组任务)。如前所述，所请求的DoP可以基于对应于目的分片的分片的数量。

[0138] 在操作808，执行平台110基于目的分片集合，在外部数据流平台120上执行命令。如前所述，执行平台110处理所请求的DoP，并基于虚拟仓库的大小启动一组执行节点。基于该命令和/或前述查询计划，该组执行节点从存储平台104读取数据(例如，来自表的行)。执行节点将数据转发到第二组执行节点，第二组执行节点负责将数据存储到外部数据流平台120。然后，第二组执行节点为每个目的分片调用外部函数(例如，STORE_DATA函数)，以相应地将数据存储到外部数据流平台120。

[0139] 在操作810，执行平台110指示命令完成。如前所述，当被执行时，STORE_DATA函数返回一个布尔值，指示是否所有行都被成功存储在外部数据流平台120中。如果布尔值为真，在此阶段，命令610成功完成，并且事务被记录为在存储平台104中提交。

[0140] 根据示例实施例，图9以计算机系统的形式示出了机器900的示意图表示，用于使机器900执行本文所讨论方法中的任何一个或更多个的指令集合可以在该计算机系统内执行。具体地，图9以计算机系统的示例形式示出了机器900的示意图表示，用于使机器900执行本文所讨论方法中的任何一个或更多个的指令916(例如，软件、程序、应用、小程序、小应用(app)或其他可执行代码)在该计算机系统内执行。例如，指令916可以使机器900执行上述方法中的任何一个或更多个的任何一个或更多个操作。作为另一个示例，指令916可以使机器900实现图1-8中的任何一个或更多个中所示的部分功能。以这种方式，指令916将通用的、未编程的机器转换成特定的机器900(例如，计算服务管理器108、执行平台110和用户设备112)，该特定的机器900被具体配置成以本文描述的方式执行所描述和示出的功能中的任何一个。

[0141] 在替代实施例中，机器900作为独立的设备操作或者可以耦合(例如，联网)到其它机器。在联网部署中，机器900可以在服务器-客户端网络环境中作为服务器机器或客户端

机器来操作，或者作为对等(或分布式)网络环境中的对等机器来操作。机器900可以包括但不限于服务器计算机、客户端计算机、个人计算机(PC)、平板计算机、膝上型计算机、上网本(netbook)、智能电话、移动设备、网络路由器、网络交换机、网络桥接器或能够顺序地或以其它方式执行指令916的任何机器，指令916指定由机器900采取的动作(action)。此外，虽然仅示出了单个机器900，但术语“机器”还应被理解为包括单独或联合地执行指令916以实施本文所讨论方法中的任何一个或更多个的机器900的集合。

[0142] 机器900包括处理器910、存储器930和输入/输出(I/O)组件950，它们被配置成例如经由总线902彼此通信。在示例实施例中，处理器910(例如，中央处理单元(CPU)、精简指令集计算(RISC)处理器、复杂指令集计算(CISC)处理器、图形处理单元(GPU)、数字信号处理器(DSP)、专用集成电路(ASIC)、射频集成电路(RFIC)、另一处理器或其任何合适的组合)可以包括例如可以执行指令916的处理器912和处理器914。术语“处理器”旨在包括多核处理器910，多核处理器910可以包括可以同时执行指令916的两个或更多个独立处理器(有时称为“核”)。尽管图9示出了多个处理器910，但是机器900可以包括具有单核的单个处理器、具有多核的单个处理器(例如，多核处理器)、具有单核的多个处理器、具有多核的多个处理器或者它们的任意组合。

[0143] 存储器930可以包括主存储器932、静态存储器934和存储单元936，它们都可以由处理器910例如经由总线902访问。主存储器932、静态存储器934和存储单元936存储指令916，指令916体现本文描述的方法或功能中任何一种或更多种。在由机器900执行期间，指令916也可以全部或部分地驻留在主存储器932内、静态存储器934内、存储单元936内、至少一个处理器910内(例如，在处理器的高速缓冲存储器内)或其任何合适的组合内。

[0144] I/O组件950包括用于接收输入、提供输出、产生输出、传输信息、交换信息、捕获测量值等的组件。包含在特定机器900中的具体的I/O组件950将取决于机器的类型。例如，诸如移动电话的便携式机器将可能包括触摸输入设备或其他这样的输入机制，而无头服务器机器可能不包括这样的触摸输入设备。将认识到的是，I/O组件950可以包括图9中未示出的许多其他组件。I/O组件950根据功能进行分组仅仅是为了简化下面的讨论，并且该分组决不是限制性的。在各种示例实施例中，I/O组件950可以包括输出组件952和输入组件954。输出组件952可以包括视觉组件(例如，诸如等离子体显示面板(PDP)、发光二极管(LED)显示器、液晶显示器(LCD)、投影仪或阴极射线管(CRT)的显示器)、声学组件(例如，扬声器)、其他信号发生器等。输入组件954可以包括字母数字输入组件(例如，键盘、被配置成接收字母数字输入的触摸屏、光电键盘或其他字母数字输入组件)、基于点的输入组件(例如，鼠标、触控板、轨迹球、操纵杆、运动传感器或另一指向仪器)、触觉输入组件(例如，物理按钮、提供触摸或触摸手势的位置和/或力的触摸屏或其他触觉输入组件)、音频输入组件(例如，麦克风)等。

[0145] 通信可以使用多种技术来实现。I/O组件950可以包括通信组件964，通信组件964可操作来分别经由耦合982和耦合972将机器900耦合到网络980或设备970。例如，通信组件964可以包括网络接口组件或与网络980接口的另一合适的设备。在进一步的示例中，通信组件964可以包括有线通信组件、无线通信组件、蜂窝通信组件和经由其他模态提供通信的其他通信组件。设备970可以是另一台机器或多种外围设备中的任一种(例如，经由通用串行总线(USB)耦合的外围设备)。例如，如上所述，机器900可以对应于计算服务管理器108、

执行平台110和设备970中的任何一个，可以包括用户设备112或本文描述的与基于网络的数据仓库系统102或存储平台104通信的任何其他计算设备。

[0146] 可执行指令和机器存储介质

[0147] 各种存储器(例如，1430、1432、1434和/或处理器1410和/或存储单元1436的存储器)可以存储指令1416和数据结构(例如，软件)的一个或更多个集合，这些指令1416和数据结构体现本文所描述方法或功能中的任何一种或更多种或被本文所描述方法或功能中的任何一种或更多种所利用。当由处理器1410执行时，这些指令1416促使各种操作来实现所公开的实施例。

[0148] 如本文所使用的，术语“机器存储介质”、“设备存储介质”和“计算机存储介质”意思相同，并且可以在本公开中互换使用。这些术语指的是存储可执行指令和/或数据的单个或多个存储设备和/或介质(例如，集中式或分布式数据库和/或相关联的高速缓存和服务器)。相应地，这些术语应当被认为包括但不限于包括处理器内部或外部的存储器的固态存储器以及光和磁介质。机器存储介质、计算机存储介质和/或设备存储介质的具体示例包括非易失性存储器，非易失性存储器包括例如半导体存储器设备，诸如可擦除可编程只读存储器(EPROM)、电可擦除可编程只读存储器(EEPROM)、现场可编程门阵列(FPGA)和闪存设备；磁盘，如内部硬盘和可移除磁盘；磁光盘；以及CD-ROM和DVD-ROM光盘。术语“机器存储介质”、“计算机存储介质”和“设备存储介质”具体排除了载波、调制数据信号和其他这样的介质(其中的至少一些包含在下面讨论的术语“信号介质”中)。

[0149] 传输介质

[0150] 在各种示例实施例中，网络980的一个或更多个部分可以是自组织网络(ad hoc network)、内联网、外联网、虚拟专用网络(VPN)、局域网(LAN)、无线LAN(WLAN)、广域网(WAN)、无线WAN(WWAN)、城域网(MAN)、互联网、互联网的一部分、公共交换电话网络(PSTN)的一部分、普通老式电话服务(POTS)网络、蜂窝电话网络、无线网络、Wi-Fi®网络、另一种类型的网络、或两种或更多种这样的网络的组合。例如，网络980或网络980的一部分可以包括无线或蜂窝网络，并且耦合982可以是码分多址(CDMA)连接、全球移动通信系统(GSM)连接或另一种类型的蜂窝或无线耦合。在该示例中，耦合982可以实现多种类型的数据传输技术中的任何一种，如单载波无线电传输技术(1xRTT)、演进数据优化(EVDO)技术、通用分组无线电服务(GPRS)技术、增强型数据速率GSM演进(EDGE)技术、包括3G、第四代无线(4G)网络、通用移动电信系统(UMTS)、高速分组接入(HSPA)、微波接入全球互操作性(WiMAX)、长期演进(LTE)标准的第三代合作伙伴计划(3GPP)、由各种标准设置组织定义的其他技术、其他长程协议、或其他数据传输技术。

[0151] 指令916可以使用传输介质经由网络接口设备(例如，包括在通信组件964中的网络接口组件)并利用多种众所周知的传输协议(例如，超文本传输协议(HTTP))中的任何一种在网络980上传输或接收。类似地，指令916可以使用传输介质经由耦合972(例如，对等耦合)传输或接收到设备970。术语“传输介质”和“信号介质”意思相同，在本公开中可以互换使用。术语“传输介质”和“信号介质”应被理解为包括能够存储、编码或携带由机器900执行的指令916的任何无形介质，并且包括数字或模拟通信信号或便于这种软件的通信的其他无形介质。因此，术语“传输介质”和“信号介质”应被理解为包括任何形式的调制数据信号、载波等。术语“调制数据信号”意指这样的信号：其具有以对信号中的信息进行编码的这样

的方式设置或更改的其特性中的一个或更多个特性。

[0152] 计算机可读介质

[0153] 术语“机器可读介质”、“计算机可读介质”和“设备可读介质”意思相同，并且在本公开中可以互换使用。这些术语被定义为包括机器存储介质和传输介质。因此，这些术语包括存储设备/介质和载波/调制数据信号。

[0154] 本文所述的示例方法的各种操作可以至少部分地由(例如，通过软件)临时配置或永久地配置为执行相关操作的一个或更多个处理器来执行。类似地，本文描述的方法可以至少部分地是处理器实现的。例如，方法300和400的至少一些操作可以由一个或更多个处理器执行。某些操作的执行可以分布在一个或更多个处理器中，不仅驻留在单个机器内，而且部署在多个机器上。在一些示例实施例中，一个或更多个处理器可以位于单个位置(例如，在家庭环境、办公室环境或服务器场内)中，而在其他实施例中，处理器可以分布在多个位置上。

[0155] 尽管参考具体的示例实施例描述了本公开的实施例，但显然地，在不背离本发明主题的更广泛范围的情况下，可以对这些实施例做出各种修改和变化。相应地，说明书和附图被认为是说明性的而不是限制性的。形成其一部分的附图通过说明方式并且不以限制方式显示了其中可以实施主题的具体实施例。所示出的实施例以充分细节描述，以便使本领域那些技术人员能够实施本文披露的教导。可以使用其他实施例和从其中衍生的实施例，使得可以作出结构性或逻辑性替换和变化而不偏离本公开的范围。因此不以限制性意义理解这一详细说明，并且各种实施例的范围仅由所附权利要求连同授权的此类权利要求的完整范围的等同物限定。

[0156] 发明主题的此类实施例可以在本文中单独地和/或共同地由术语“发明”提到，该术语仅出于便利目的并且不意在自愿将本申请的范围限于任何单一发明或发明构思，如果实际上披露的多于一种的话。因此，虽然已经在此说明和描述了特定实施例，但是应当理解，可以用经计算以实现相同目的的任何布置替换用于所示的具体实施例。本公开旨在覆盖各种实施例的任何和所有修改或变型。在阅读以上描述后，本领域的技术人员将明白以上实施例的组合以及本文未具体描述的其他实施例。

[0157] 在本文档中，术语“一个(a)”或“一个(an)”，如专利文档中常见的那样，用于包括一个或多于一个，与“至少一个”或“一个或更多个”的任何其他实例或用法无关。在本文档中，术语“或”用于指非排他性的或，因此“A或B”包括“A但不包括B”、“B但不包括A”和“A和B”，除非另有说明。在所附权利要求中，术语“包括(including)”和“其中(in which)”被用作相应术语“包含(comprising)”和“其中(wherein)”的纯英语等效词。此外，在所附权利要求中，术语“包括”和“包含”是开放式的；也就是说，包括除了权利要求中这样的术语之后列出的那些元素之外的元素的系统、设备、物品或过程仍然被认为落入该权利要求的范围内。

[0158] 示例

[0159] 示例1是基于网络的数据库系统，包括：至少一个硬件处理器；以及存储器，其存储使至少一个硬件处理器执行操作的指令，所述操作包括：确定对应于可通过网络访问的外部数据源的分片集合，外部数据源由与基于网络的数据库系统分离的外部系统托管；使用流对象确定该分片集合中每个分片的偏移集合；识别要对该分片集合执行的操作，该操作包括读操作或写操作；并且基于该分片集合和该偏移集合，对外部数据源执行操作。

[0160] 在示例2中，根据示例1的主题，其中分片集合的每个分片可选地包括由外部数据源存储的数据记录序列

[0161] 在示例3中，根据示例1和示例2中任一个的主题，其中确定该分片集合中的每个分片的偏移集合可选地包括：执行包括在流对象中的用户定义函数以确定该偏移集合。

[0162] 在示例4中，根据示例1-3中任一个的主题，其中用户定义的函数在被执行时可选地确定相对于在先前事务期间对应于流对象的可扩展流被访问的先前时间的偏移，并且在先前事务完成时相对于先前时间的偏移被预先存储。

[0163] 在示例5中，根据示例1-4中任一个的主题，其中操作可选地进一步包括确定可扩展流被访问的先前时间之后的新偏移的集合。

[0164] 在示例6中，根据示例1-5中任一个的主题，其中操作可选地进一步包括：确定在可扩展流被访问的先前时间之后创建了新分片；确定对应于新分片的新偏移；以及提供新偏移与该新偏移的集合。

[0165] 在示例7中，根据示例1-6中任一个的主题，其中操作可选地进一步包括：确定多个偏移对应于来自分片集合的特定分片；将该分片集合中的特定分片分成至少两个分片；以及在访问至少两个分片中的第二分片之前访问第一分片。

[0166] 在示例8中，根据示例1-7中任一个的主题，其中用户定义的函数在被执行时，可选地至少部分基于该分片集合和该偏移集合向外部数据源请求数据。

[0167] 在示例9中，根据示例1-8中任一个的主题，其中用户定义函数可选地使用基于该分片集合的一组并行操作来请求数据。

[0168] 在示例10中，根据示例1-9中任一个的主题，其中流对象可选地包括：提供存储的偏移的集合的第一函数；存储偏移的第二集合的第二函数；以及第三函数，其提供与外部对象的集成，使得能够与外部数据源通信。

[0169] 在示例11中，根据示例1-10中任一个的主题，其中外部数据源可选地包括外部数据流平台。

[0170] 示例12是一种方法，包括：使用至少一个硬件处理器，确定对应于可通过网络访问的外部数据源的分片集合，该外部数据源由与基于网络的数据库系统分开的外部系统托管；使用流对象确定该分片集合中每个分片的偏移集合；识别要对该分片集合执行的操作，该操作包括读操作或写操作；并且基于该分片集合和该偏移集合，对外部数据源执行操作。

[0171] 在示例13中，根据示例12的主题，其中分片集合的每个分片可选地包括由外部数据源存储的数据记录序列。

[0172] 在示例14中，根据示例12-13的主题，其中确定分片集合中的每个分片的偏移集合可选地包括：执行包括在流对象中的用户定义函数来确定偏移集合。

[0173] 在示例15中，根据示例12-14中任一个的主题，其中用户定义的函数在被执行时可选地确定相对于在先前事务期间对应于流对象的可扩展流被访问的先前时间的偏移，并且在先前事务完成时相对于先前时间的偏移被预先存储。

[0174] 在示例16中，根据示例12-15中任一个的主题进一步可选地包括确定可扩展流被访问的先前时间之后的新偏移的集合。

[0175] 在示例17中，根据示例12-16中任一个的主题进一步可选地包括：确定在可扩展流被访问的先前时间之后创建了新分片；确定对应于新分片的新偏移；以及提供新偏移与该

新偏移的集合。

[0176] 在示例18中，根据示例12-17中任一个的主题进一步可选地包括确定多个偏移对应于来自分片集合的特定分片；将该分片集合中的特定分片分成至少两个分片；以及在访问至少两个分片中的第二分片之前访问第一分片。

[0177] 在示例19中，根据示例12-18中任一个的主题，其中用户定义函数在被执行时，可选地至少部分地基于该分片集合和该偏移集合向外部数据源请求数据，并且用户定义函数使用基于该分片集合的一组并行操作来请求数据。

[0178] 在示例20中，非暂时性计算机存储介质包括指令，当由处理器执行时，所述指令配置处理器执行操作，所述操作包括确定对应于可通过电子网络访问的外部数据源的分片集合，外部数据源由与基于网络的数据库系统分开的外部系统托管；使用流对象确定该分片集合中每个分片的偏移集合；识别要对该分片集合执行的操作，该操作包括读操作或写操作；并且基于该分片集合和该偏移集合，对外部数据源执行操作。

[0179] 在示例21中，根据示例20的主题，其中分片集合的每个分片可选地包括由外部数据源存储的数据记录序列。

[0180] 在示例22中，根据示例20-21的主题，其中确定分片集合中的每个分片的偏移集合可选地包括：执行包括在流对象中的用户定义函数来确定偏移集合。

[0181] 在示例23中，根据示例20-22的主题，其中用户定义的函数在被执行时可选地确定相对于在先前事务期间对应于流对象的可扩展流被访问的先前时间的偏移，并且在先前事务完成时相对于先前时间的偏移被预先存储。

[0182] 在示例24中，根据示例20-23的主题，其中操作进一步可选地包括：确定可扩展流被访问的先前时间之后的新偏移的集合。

[0183] 在示例25中，根据示例20-24的主题，其中操作可选地进一步包括：确定在可扩展流被访问的先前时间之后创建了新分片；确定对应于新分片的新偏移；以及提供新偏移与该新偏移的集合。

[0184] 在示例26中，根据示例20-25的主题，其中操作进一步可选地包括：确定多个偏移对应于来自分片集合的特定分片；将该分片集合中的特定分片分成至少两个分片；以及在访问至少两个分片中的第二分片之前访问第一分片。

[0185] 在示例27中，根据示例20-26的主题，其中用户定义的函数在被执行时，可选地至少部分基于该分片集合和该偏移集合向外部数据源请求数据。

[0186] 在示例28中，根据示例20-27的主题，其中用户定义的函数可选地使用基于该分片集合的一组并行操作来请求数据。

[0187] 在示例29中，根据示例20-28的主题，其中用户定义的函数在被执行时，至少部分地基于来自分片集合的特定分片和来自内部数据源的要存储在特定分片中的一组行，可选地将数据存储到外部数据源中。

[0188] 在示例30中，根据示例20-29的主题，其中流对象可选地包括：提供存储的偏移的集合的第一函数；存储偏移的第二集合的第二函数；以及第三函数，其提供与外部对象的集成，使得能够与外部数据源通信。

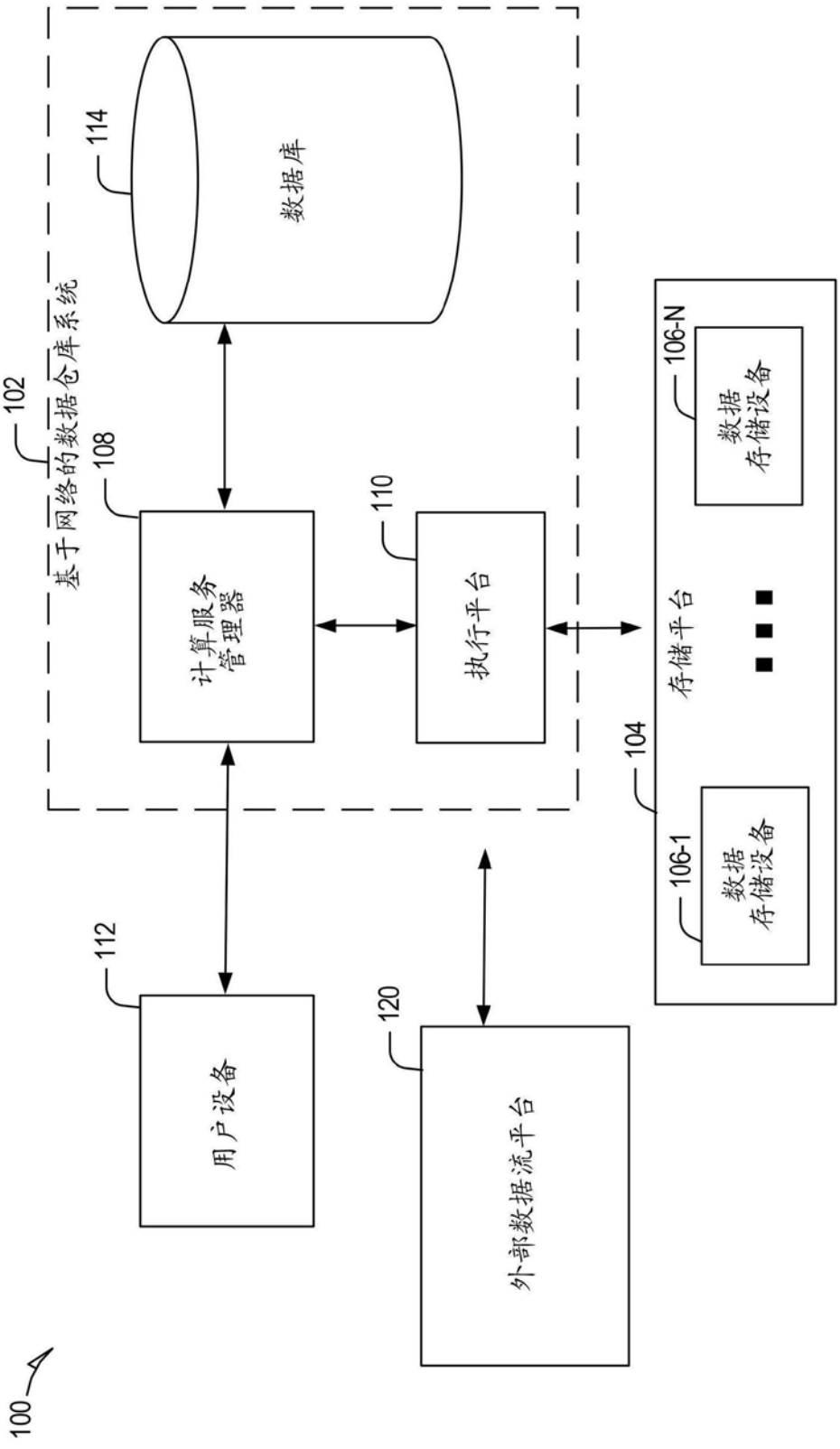

100
102
基于网络的数据仓库系统
114
数据库
108
计算服务管理器
110
执行平台
104
存储平台
106-N
数据存储设备
106-1
数据存储设备
112
用户设备
120
外部数据流平台

图1

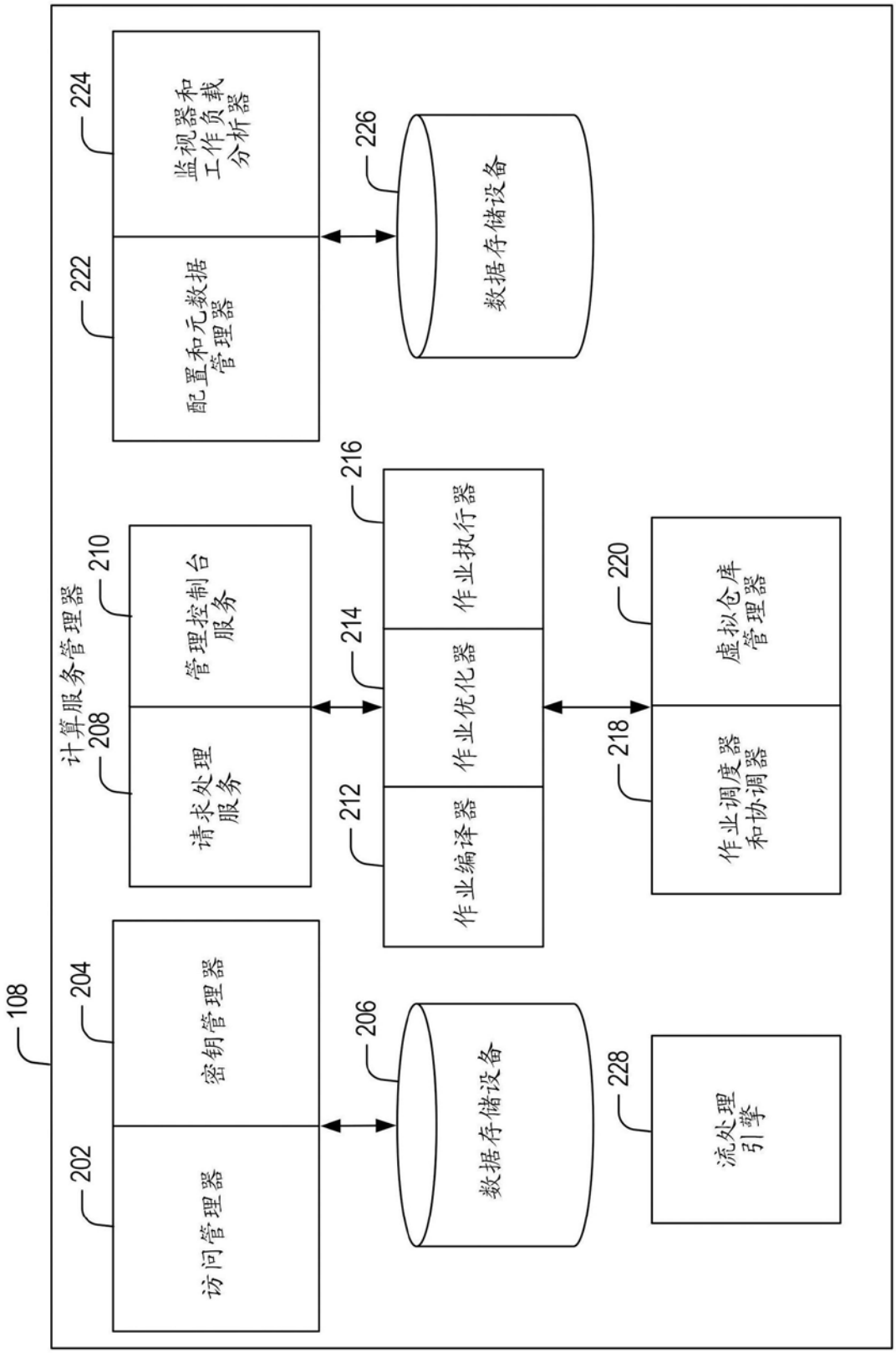

108
计算服务管理器
202
访问管理器
204
密钥管理器
206
数据存储设备
208
请求处理服务
210
管理控制台服务
212
作业编译器
214
作业优化器
216
作业执行器
218
作业调度器和协调器
220
虚拟仓库管理器
222
配置和元数据管理器
224
监视器和工作负载分析器
226
数据存储设备
228
流处理引擎

图2

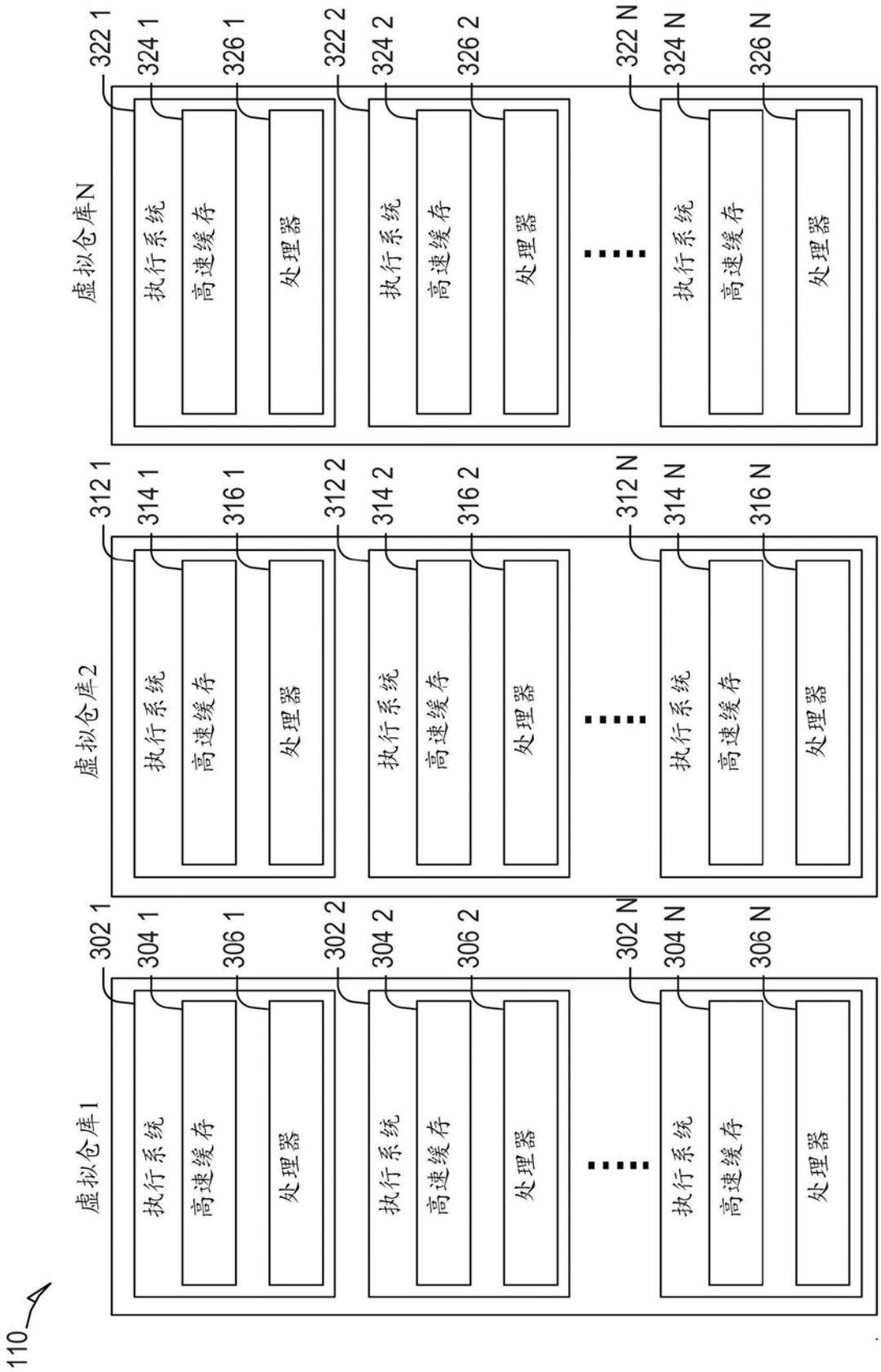

110
虚拟仓库1
虚拟仓库2
虚拟仓库N
302 1
304 1
306 1
302 2
304 2
306 2
302 N
304 N
306 N
312 1
314 1
316 1
312 2
314 2
316 2
312 N
314 N
316 N
322 1
324 1
326 1
322 2
324 2
326 2
322 N
324 N
326 N
执行系统
高速缓存
处理器

图3

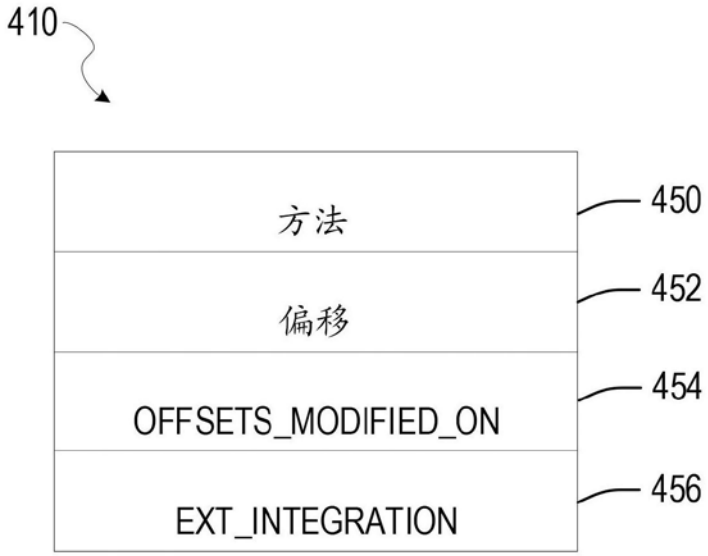

410
方法
450
偏移
452
OFFSETS_MODIFIED_ON
454
EXT_INTEGRATION
456

图4

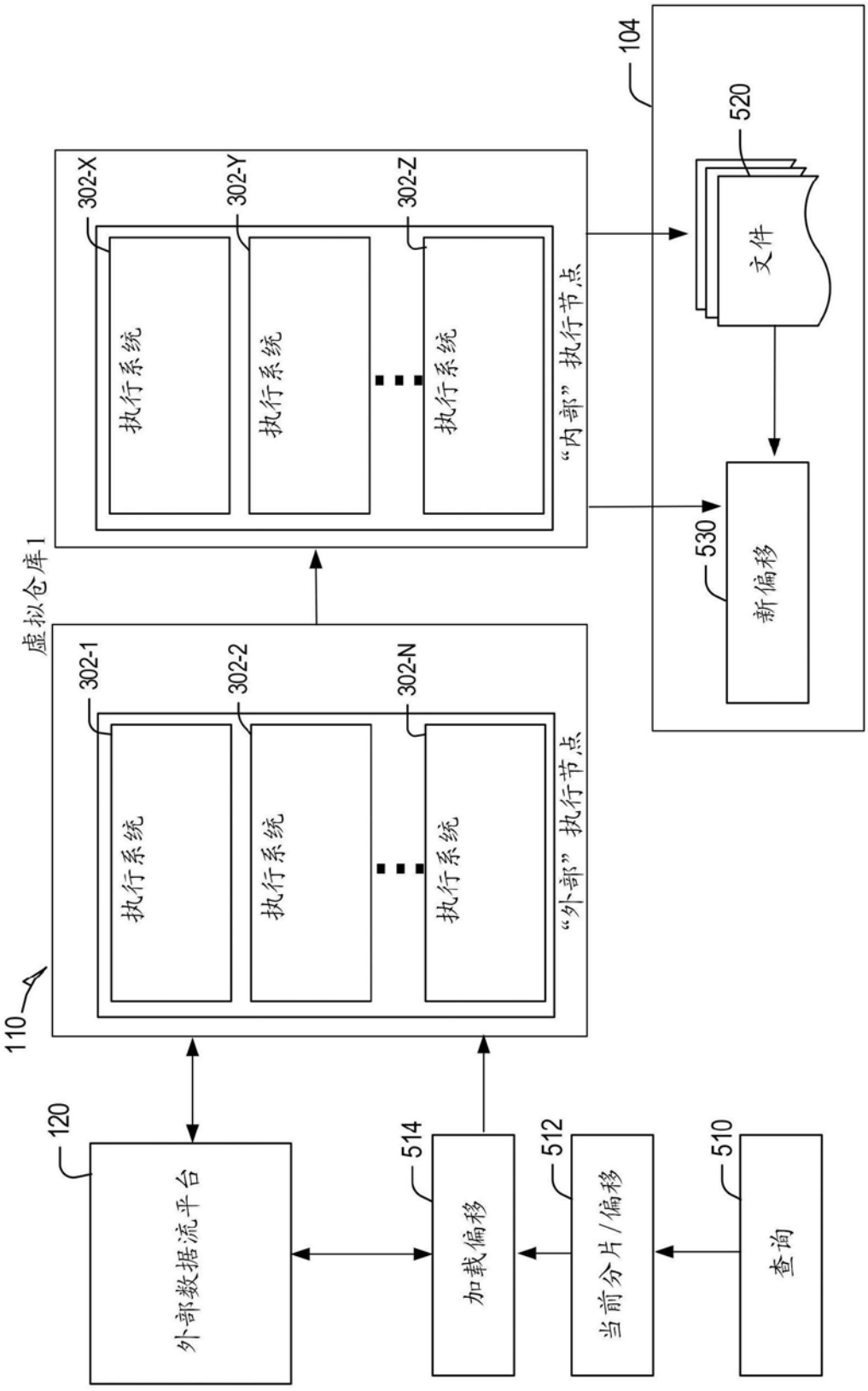

104
520
文件
302-X
302-Y
302-Z
执行系统
执行系统
执行系统
“内部”执行节点
虚拟仓库1
530
新偏移
302-1
302-2
302-N
执行系统
执行系统
执行系统
“外部”执行节点
110
120
外部数据流平台
514
加载偏移
512
当前分片/偏移
510
查询

图5

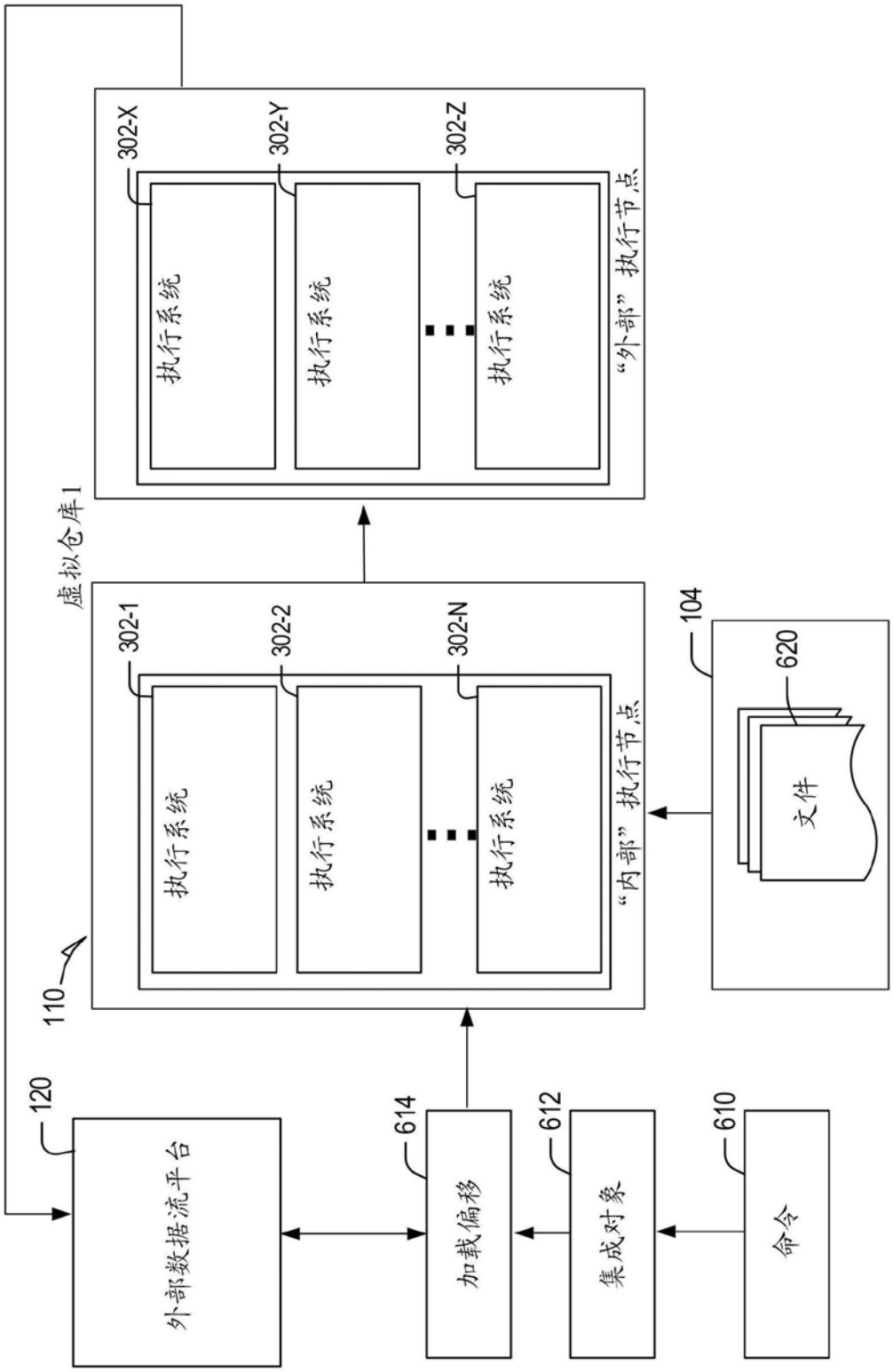

虚拟仓库1
302-X
302-Y
302-Z
执行系统
执行系统
执行系统
“外部”执行节点
302-1
302-2
302-N
执行系统
执行系统
执行系统
“内部”执行节点
110
104
620
文件
120
外部数据流平台
614
加载偏移
612
集成对象
610
命令

图6

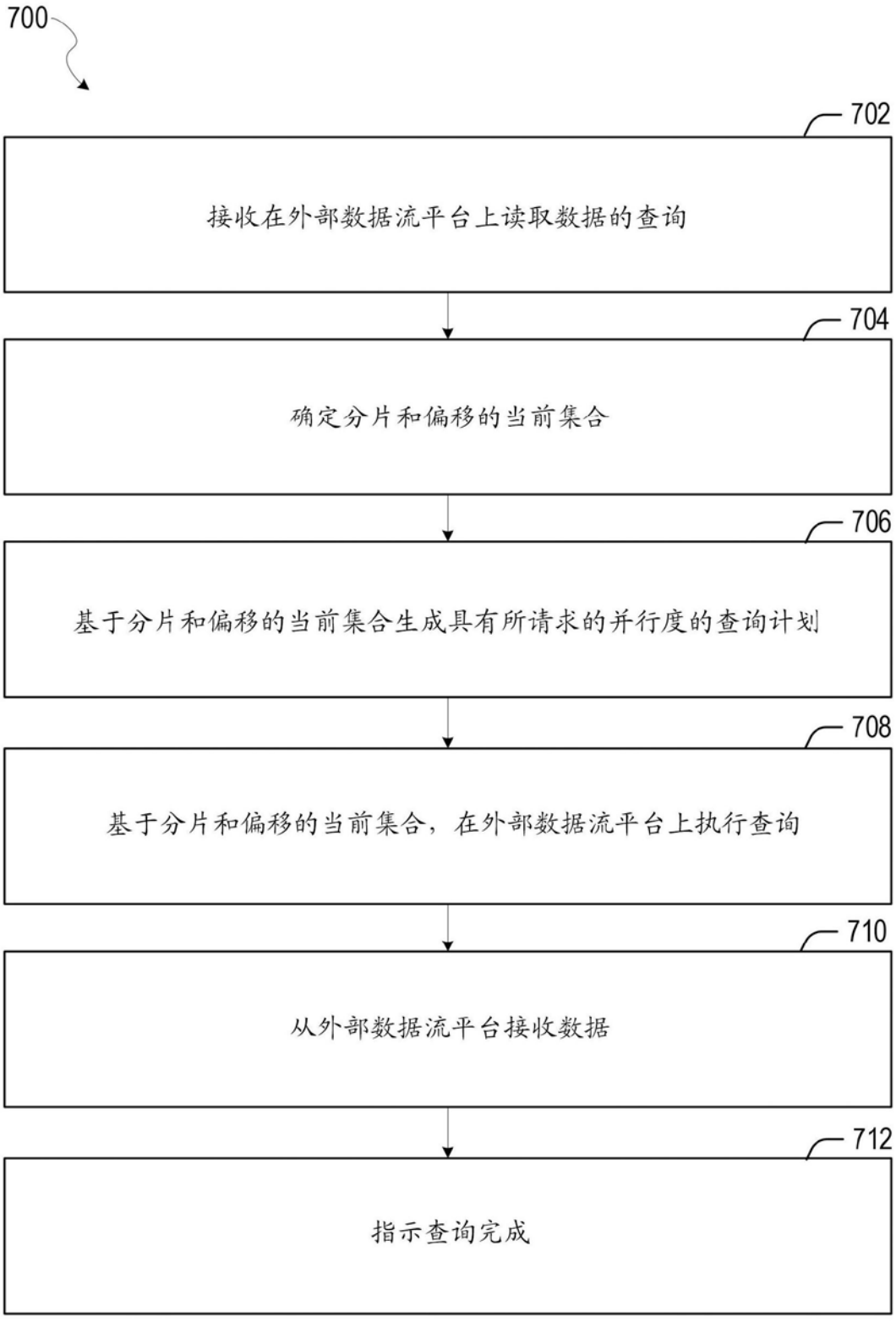

700
702
接收在外部数据流平台上读取数据的查询
704
确定分片和偏移的当前集合
706
基于分片和偏移的当前集合生成具有所请求的并行度的查询计划
708
基于分片和偏移的当前集合，在外部数据流平台上执行查询
710
从外部数据流平台接收数据
712
指示查询完成

图7

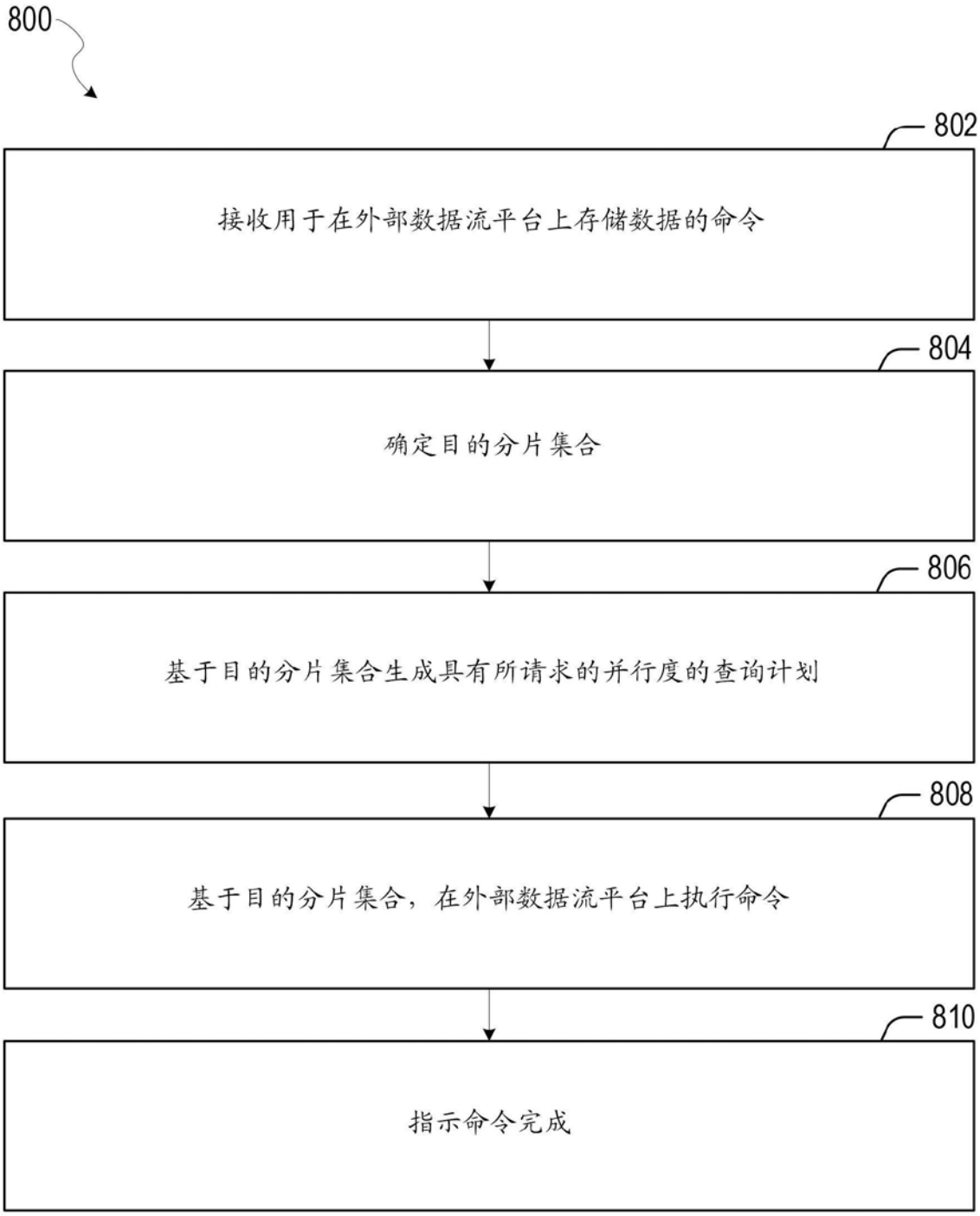

800
802
接收用于在外部数据流平台上存储数据的命令
804
确定目的分片集合
806
基于目的分片集合生成具有所请求的并行度的查询计划
808
基于目的分片集合，在外部数据流平台上执行命令
810
指示命令完成

图8

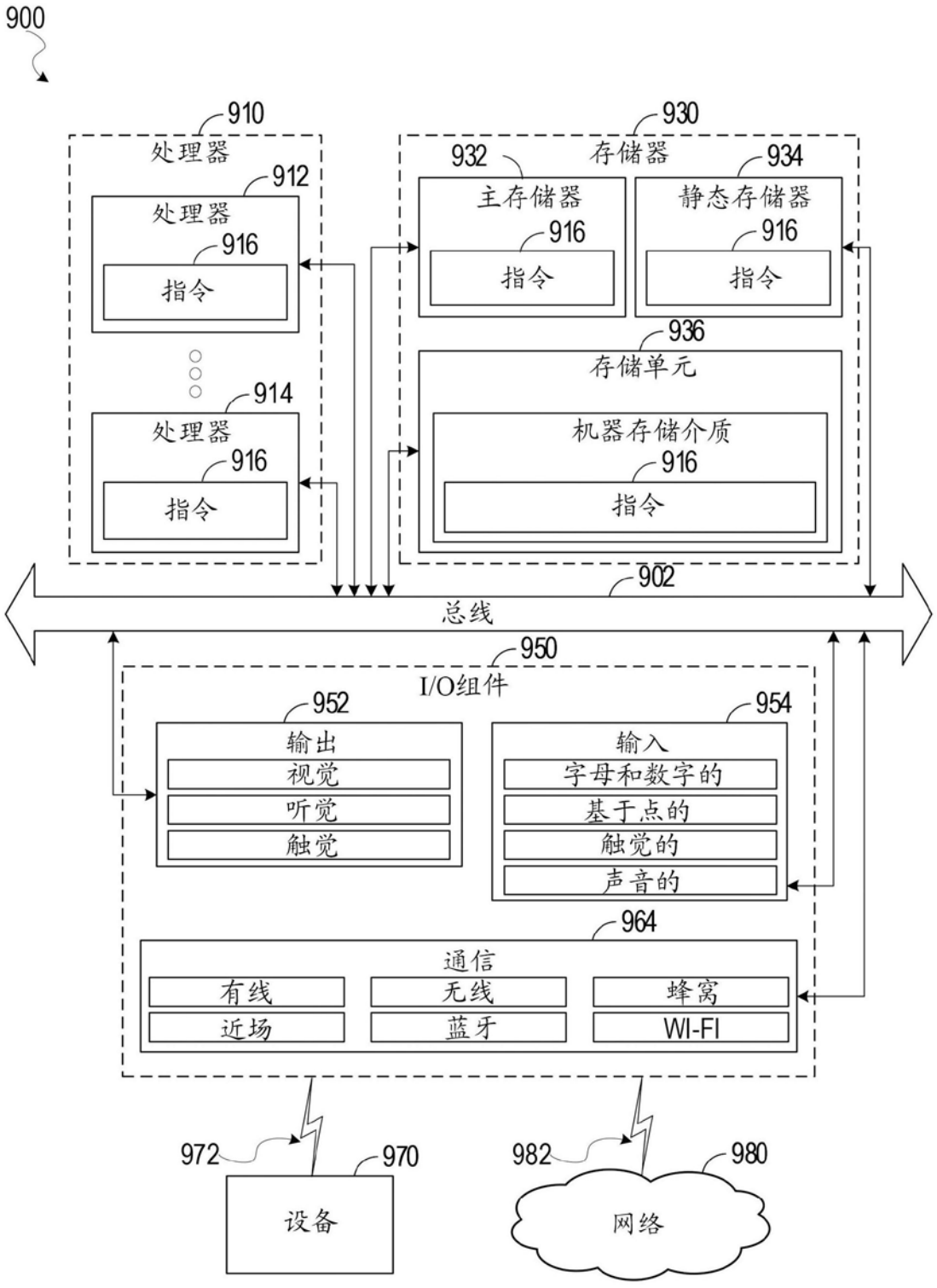

900
910
处理器
912
处理器
916
指令
914
处理器
916
指令
930
存储器
932
主存储器
916
指令
934
静态存储器
916
指令
936
存储单元
机器存储介质
916
指令
902
总线
950
I/O组件
952
输出
视觉
听觉
触觉
954
输入
字母和数字的
基于点的
触觉的
声音的
964
通信
有线
无线
蜂窝
近场
蓝牙
WI-FI
972
970
设备
982
980
网络

图9