(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 103092905 A
(43)申请公布日 2013.05.08

(21)申请号 201210433346.1

(22)申请日 2012.09.28

(30)优先权数据
13/290,866 2011.11.07 US

(71)申请人 SAP 股份公司
地址 德国瓦尔多夫

(72)发明人 I·施赖特 T·格列布 T·朔伊尔

(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 邵亚丽

(51)Int.Cl.
G06F 17/30(2006.01)

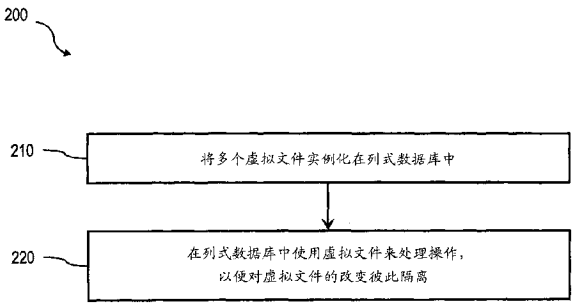
权利要求书2页 说明书9页 附图5页

(54)发明名称

使用虚拟文件数据对象的列式数据库

(57)摘要

多个虚拟文件被实例化在列式数据库中。列式数据库包括列式数据存储器，该列式数据存储器将其中包含的数据表保存到多个虚拟文件。每个虚拟文件存储在物理数据存储器中，一部分虚拟文件可能短暂高速缓存于列式数据存储器和物理数据存储器中间的持久层中。每个虚拟文件包括用于储存数据记录的数据库对象。在列式数据库中使用虚拟文件来处理操作，以便由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。相关设备、系统、技术和物品也被描述。



CN 103092905 A

1. 一种计算机程序产品，包括存储指令的非短暂的机器可读介质，当所述指令由至少一个可编程处理器执行时，使所述至少一个可编程处理器执行操作，所述操作包括：

在列式数据库中实例化多个虚拟文件，所述列式数据库包括列式数据存储器，所述列式数据存储器将包含在其中的数据表保存到所述多个虚拟文件，每个虚拟文件存储在物理数据存储器中并且每个虚拟文件的一部分能够短暂地高速缓存在所述列式数据存储器和所述物理数据存储器中间的持久层中，所述每个虚拟文件包括用于存储数据记录的数据库对象；以及

在所述列式数据库中使用所述虚拟文件来处理操作，其中由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。

2. 根据权利要求 1 所述的计算机程序产品，其中所述操作进一步包括：

当至少一个其它事务正在执行时，添加用于第一事务的至少一个数据记录，在所述第一事务提交之后，所述添加的至少一个数据记录可用于其它事务。

3. 根据权利要求 1 所述的计算机程序产品，其中所述操作进一步包括：

当至少一个其它事务正在执行时，删减用于第一事务的至少一个数据记录，在所述第一事务提交之前，所述经删减的至少一个数据记录可用于其它事务。

4. 根据权利要求 1 所述的计算机程序产品，其中每个虚拟文件存储在所述持久层中的数据库页面链中。

5. 根据权利要求 4 所述的计算机程序产品，其中所述数据库页面具有固定大小。

6. 根据权利要求 4 所述的计算机程序产品，其中所述数据库页面具有可变大小。

7. 根据权利要求 4 所述的计算机程序产品，其中所述操作进一步包括：

对于每个虚拟文件，将对相应开始数据库页面和相应结束数据库页面的引用存储在元数据中。

8. 根据权利要求 7 所述的计算机程序产品，其中所述操作进一步包括：

将对虚拟文件的最后重写记录的链接存储在元数据中，所述重写记录详细说明对所述虚拟文件做出的删减改变。

9. 根据权利要求 8 所述的计算机程序产品，其中所述操作进一步包括：

将所述元数据存储在容器目录中，所述容器目录被访问以识别最新版本的虚拟文件。

10. 一种方法，包括：

在列式数据库中实例化多个虚拟文件，所述列式数据库包括列式数据存储器，所述列式数据存储器将包含在其中的数据表保存到所述多个虚拟文件，每个虚拟文件存储在物理数据存储器中并且每个虚拟文件的一部分能够短暂地高速缓存在所述列式数据存储器和所述物理数据存储器中间的持久层中，所述每个虚拟文件包括用于存储数据记录的数据库对象；以及

在所述列式数据库中使用所述虚拟文件来处理操作，其中由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。

11. 根据权利要求 10 所述的方法，进一步包括：

当至少一个其它事务正在执行时，添加用于第一事务的至少一个数据记录，在所述第一事务提交之后，所述添加的至少一个数据记录可用于其它事务。

12. 根据权利要求 10 所述的方法，进一步包括：

当至少一个其它事务正在执行时，删减用于第一事务的至少一个数据记录，在所述第一事务提交之前，所述经删减的至少一个数据记录可用于其它事务。

13. 根据权利要求 10 所述的方法，其中每个虚拟文件存储在所述持久层中的数据库页面链中。

14. 根据权利要求 13 所述的方法，其中所述数据库页面具有固定大小。

15. 根据权利要求 13 所述的方法，其中所述数据库页面具有可变大小。

16. 根据权利要求 13 所述的方法，进一步包括：对于每个虚拟文件，将对相应开始数据库页面和相应结束数据库页面的引用存储在元数据中。

17. 根据权利要求 16 所述的方法，进一步包括：将对虚拟文件的最后重写记录的链接存储在元数据中，所述重写记录详细说明对所述虚拟文件做出的删减改变。

18. 根据权利要求 17 所述的方法，进一步包括：将所述元数据存储在容器目录中，所述容器目录被访问以识别最新版本的虚拟文件。

19. 一种系统，包括：

至少一个可编程处理器；

耦合到所述至少一个可编程处理器的存储器，所述存储器存储指令，当所述指令由所述至少一个可编程处理器执行时，使所述至少一个可编程处理器执行操作，所述操作包括：

在列式数据库中实例化多个虚拟文件，所述列式数据库包括列式数据存储器，所述列式数据存储器将包含在其中的数据表保存到所述多个虚拟文件，每个虚拟文件存储在物理数据存储器中并且每个虚拟文件的一部分能够短暂地高速缓存在所述列式数据存储器和所述物理数据存储器中间的持久层中，所述每个虚拟文件包括用于存储数据记录的数据库对象；以及

在所述列式数据库中使用所述虚拟文件来处理操作，其中由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。

20. 根据权利要求 19 所述的系统，其中所述操作进一步包括：

当至少一个其它事务正在执行时，添加用于第一事务的至少一个数据记录，在所述第一事务提交之后，所述添加的至少一个数据记录可用于其它事务；以及

当至少一个其它事务正在执行时，删减用于第二事务的至少一个数据记录，在所述第二事务提交之前，所述经删减的至少一个数据记录可用于其它事务。

使用虚拟文件数据对象的列式数据库

技术领域

[0001] 本文描述的主题涉及用于提供在列式 (columnar) 数据库中使用的虚拟文件数据对象的技术。

背景技术

[0002] 数据库能够执行包含相应数据的大量并发事务 (concurrent transaction)。这些并发事务可能导致对相同数据（例如，数据记录等）的改变，这使得必须在这种事务之间进行数据隔离。

发明内容

[0003] 在一个方面，多个虚拟文件被实例化在列式数据库中。列式数据库包括列式数据存储器，该列式数据存储器将其中包含的数据表保存到多个虚拟文件。每个虚拟文件存储在物理数据存储器中，一部分虚拟文件可能短暂高速缓存于列式数据存储器和物理数据存储器中间的持久层中。每个虚拟文件包括用于储存数据记录的数据库对象。在列式数据库中使用虚拟文件来处理操作，以便由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。

[0004] 当至少一个其它事务正在执行时，可以添加用于第一事务的至少一个数据记录。在第一事务提交之后，添加的至少一个数据记录可用于其它事务。当至少一个其它事务正在执行时，可以删减用于第二事务的至少一个数据记录，在第二事务提交之前，所述经删减的至少一个数据记录（即没有被删减的记录）可用于其它事务。

[0005] 每个虚拟文件可以存储在持久层中的一系列数据库页面中。数据库页面可以具有固定大小或可变大小。对相应开始数据库页面和相应结束数据库页面的引用可以存储在元数据中。对虚拟文件的最后重写记录的链接可以存储在元数据中，以便重写记录详细说明对虚拟文件做出的删减改变。元数据可以存储在容器目录中，该容器目录可以被访问以识别最新版本的虚拟文件。

[0006] 也描述包括持久存储在非短暂计算机可读介质上的计算机可执行指令的制品，当所述计算机可执行指令由计算机执行时，使计算机执行本文中的操作。相似地，也描述可包括处理器和耦合到处理器的存储器的计算机系统。存储器可暂时或持久地存储使处理器执行本文中描述的一个或多个操作的一个或多个程序。此外，用方法说明的操作可以由单个计算系统内的或分布在两个或多个计算系统中的一个或多个数据处理器来实现。

[0007] 本文描述的主题提供许多优点。例如，本文描述的虚拟文件提供依次允许数据库事务被正确处理的ACID（原子性、一致性、隔离性、持久性）属性，与此同时还由列式数据存储器提供对大量（数百万）虚拟文件的快速和可伸缩的访问。此外，实施分布式的列式数据库系统所需的各种特殊操作可以在虚拟文件上容易地实现。

[0008] 本文描述的主题的一个或多个变化的细节在下文的附图和说明中进行阐述。基于所述说明和附图以及基于权利要求，本文描述的主题的其它特征和优点将是显而易见的。

附图说明

[0009] 图 1 是说明包括数据存储应用程序的系统的框图；

[0010] 图 2 是说明由列式数据库使用虚拟文件的过程流程图；

[0011] 图 3 是说明图 1 的系统的细节的框图；

[0012] 图 4 是说明当事务正在被并发执行时添加虚拟文件的框图；以及

[0013] 图 5 是说明当事务正在被并发执行时删减虚拟文件的框图。

[0014] 在各图中相同的参考符号表示相同的单元。

具体实施方式

[0015] 图 1 示出系统 100 的示例，其中可以包括可配置、链接在一个或多个网络上的一个或多个可编程处理器等的计算系统 102 执行数据存储应用程序 104 的一个或多个模块、软件组件等。数据存储应用程序 104 可以包括数据库、企业资源程序、分布式存储系统（例如，可以从美国加州的 Sunnyvale 的 NetApp 公司获得的 NetApp Filer）等中的一个或多个。

[0016] 所述一个或多个模块、软件组件等可以是计算系统 102 的本地用户以及从网络连接 110 上的一个或多个客户端机器 106 访问计算系统 102 的远程用户可以访问的。由一个或多个第一模块生成的一个或多个用户接口屏幕可以经由本地显示器或经由与一个客户端机器 106 相关的显示器来显示给用户。数据存储应用程序 104 的数据单元可以短暂存储在持久层 112（例如，页面缓冲器或其它类型的临时持久层），其可以例如经由输入 / 输出组件 116 来以存储页面的形式将数据写到一个或多个存储器 114 中。一个或多个存储器 114 可以包括被配置用于写数据以用于较长期存储的一个或多个物理存储介质或装置（例如，硬盘驱动器、持久性闪存、随机存取存储器、光介质、磁介质等）。应该注意到，存储器 114 和输入 / 输出组件 116 可以被归入计算系统 102 中，尽管在图 1 中它们显示在计算系统 102 之外。

[0017] 保留在较长期存储器 114 的数据可以被组织为页面，每个页面具有分配给它的规定数量的存储空间。在一些实施方式中，分配给每个页面的存储空间的数量可以是常量和固定的。然而，分配给每个页面的存储空间的数量可以变化的其它实施方式也在当前主题的范围内。

[0018] 图 2 是过程流程图 200，其中在 210，多个虚拟文件被实例化 (instantiate) 在列式数据库中。列式数据库包括列式数据存储器，该列式数据存储器将其中包含的数据表保存到多个虚拟文件。每个虚拟文件存储在辅助存储器中，每个虚拟文件中的一部分可以短暂高速缓存于在列式数据存储器和物理数据存储器中间的持久层中。此外，每个虚拟文件包括用于储存数据记录的数据库对象。在 220 中，在实例化了虚拟文件之后，在列式数据库中使用虚拟文件来处理操作，以便由一个事务的操作引起的对虚拟文件的改变与由其它事务的操作引起的对相同虚拟文件的改变隔离。

[0019] 图 3 显示和当前主题的一个或多个特征一致的软件体系结构 300。可以实现在一个或多个硬件和软件中的数据存储应用程序 104 可以包括数据库应用程序、网络连接存储系统等中的一个或多个。根据当前主题的至少一些实施方式，这种数据存储应用程序 104 可以包括持久层 112 或其它类型的存储缓冲器，或者要不然例如经由持久接口 302 与持久

层 112 或其它类型的存储缓冲器连接。持久层 112 中的页面缓冲器 304 可以存储一个或多个逻辑页面 306，并且可以可选地包括影子页面 311、活动页面 313 和虚拟文件的数据页面 315（即，像文件一样起作用的数据对象）等。保留在持久层 112 中的逻辑页面 306 可以经由输入 / 输出组件 116 写到存储器（例如，较长期存储器等）114，输入 / 输出组件 116 可以是软件模块、用一个或多个软件和硬件实现的子系统等。存储器 114 可以包括一个或多个数据卷 (volume)310，其中以物理存储块来分配存储页面 312。

[0020] 在一些实施方式中，数据存储应用程序 104 可以包括行存储器 303 和列存储器 305。行存储器 303 可以包括页面管理器 314 和 / 或保存点管理器 316，或者可以用别的方法与页面管理器 314 和 / 或保存点管理器 316 通信。页面管理器 314 可以与持久层 112 的页面管理模块 320 进行通信，所述模块可以包括监视页面状态信息 324 的空闲块管理器 322，所述页面状态信息例如是存储器 114 中的物理页面以及持久层 112（以及可选地在页面缓冲器 304）中的逻辑页面的状态。保存点管理器 316 可以与持久层 204 的保存点协调器 326 进行通信以处理保存点，所述保存点被用于在可能的崩溃之后创建一致的数据库持久状态以重新启动。行存储器 303 可以经由绝对 (absolute) 页面 API 307 来访问持久接口 302。可以在连续存储中存储列的列存储器 305 可以经由虚拟文件 API309 访问持久接口 302。

[0021] 在数据存储应用程序 104 的一些实施方式中，持久层 112 的页面管理模块可以实现影子页面调度。页面管理模块 320 中的空闲块管理器 322 可以保存物理页面的状态。页面缓冲器 304 可以包括如本文所讨论的一样操作的固定页面状态缓冲器。可以是页面管理模块 320 一部分或与其通信的转换器组件 340 可以对写到存储器 114 的逻辑与物理页面之间的映射负责。转换器 340 可以在转换器表格 342 中维护逻辑页面与相应物理页面的当前映射。转换器 340 可以在一个或多个转换器表 342 中维护逻辑页面 306 与相应物理页面的当前映射。当逻辑页面 306 从存储器 114 中读入时，可以使用转换器 340 来从一个或多个转换器表 342 中查找出将被加载的存储页面。当逻辑页面在保存点之后第一次被写到存储器 114 时，新的空闲物理页面被分配给逻辑页面。空闲块管理器 322 将新的物理页面标记为“已使用”，新的映射被存储在一个或多个转换器表 342 中。

[0022] 持久层 112 可以确保数据存储应用程序 104 中做出的改变是持久的，并且数据存储应用程序 104 可以在重新启动之后被恢复到最近的提交 (commit) 的状态。将数据写到存储器 114 不必与写入事务的结束同步。同样地，当完成写入事务时，未提交的改变可以被写到磁盘，而提交的改变却也许不被写到磁盘。在系统崩溃以后，由未完成的事务进行的改变可以重新运算 (rollback)。在该过程中不应该丢失由已经提交的事务产生的改变。还可以包括记录器组件 344 以在线性日志中存储对数据存储应用程序的数据做出的改变。记录器组件 344 能在重放从最近的保存点之后的操作的恢复期间使用，以确保所有操作被用于数据以及具有被登记的“提交”记录的事务在恢复过程最后重新运算仍然打开的事务之前被提交。

[0023] 利用一些数据存储应用程序，将数据写入磁盘不必与写入事务的结束同步。可能发生当完成写入事务时未提交的改变被写入磁盘而同时提交的改变却未被写入磁盘的情况。在系统崩溃之后，由未完成的事务做出的改变必须重新运算，由已提交事务作出的改变不可以丢失。

[0024] 为确保提交的改变不丢失，可以由记录器组件 344 在进行改变的任何时候写入重

做日志信息。该信息可以最晚在事务结束时写入磁盘。日志项目可以当标准数据被写入数据卷 310 时存留在独立的日志卷 317 中。即使相应数据页面未写到磁盘中，利用重做日志可以恢复提交的改变。为了撤消未提交的改变，持久层 112 可以使用（来自一个或多个日志的）撤消日志项目和影子页面调度的组合。

[0025] 持久接口 302 可以处理存储（例如，内存中存储等）的读写请求。持久接口 302 还可以提供利用记录以及不利用记录两者来写入数据的写入方法。如果使用已记录的写入操作，则持久接口 302 调用记录器 344。此外，记录器 344 提供允许存储（例如，内存中存储等）直接添加日志项目到日志队列的接口。记录器接口也提供请求内存中日志队列中的日志项目被刷新 (flush) 到磁盘的方法。

[0026] 日志项目包含日志序号、日志项目的类型和事务的标识符。根据操作类型，记录器 344 记录附加信息。例如，对于项目类型“更新”，这将会是受影响的记录的标识和修改数据的余象 (after image)。

[0027] 当重新启动数据应用程序 104 时，需要处理日志项目。为加速该过程，并不总是从头开始处理重做日志。改为如上所述地，定期执行将从最近的保存点之后做出的（例如，在存储器等中的）所有改变写到磁盘中的保存点。当起动系统时，只有在最近的保存点之后创建的日志需要被处理。在接下来的备份操作之后，所述保存点位置之前的旧的日志项目可以被删除。

[0028] 当记录器 344 被调用以用于写入日志项目时，它没有立即写到磁盘。它改为可以将日志项目放入存储器中的日志队列。日志队列中的项目可以最迟在相应事务完成（提交或失败）时写到磁盘中。为保证提交的改变不丢失，在相应日志项目被刷新到磁盘之前提交操作不算成功完成。将日志队列项目写入磁盘还可以由其它事件触发，例如当日志队列页面已满或当执行保存点时。

[0029] 列存储器 305 可以将其表保存到由持久层 112 经由虚拟文件 API 307 提供的虚拟文件。持久层 112 可以在内部将虚拟文件映射到存储在页面缓冲器 304 中的一系列链接页面 315。属于一个列式表格的数据可以存储在多个虚拟文件：用于主存储的每列一个虚拟文件以及用于增量 (delta) 日志的一个虚拟文件。此外，对于表格的历史部分的主存储可以可选地每列存储一个虚拟文件，和 / 或对于表格的历史部分的增量可以可选地每个表格存储一个虚拟文件。持久层 112 可以维护目录，所述目录对于每个虚拟文件存储启动页面和诸如虚拟文件的大小和类型的附加信息。

[0030] 如上所述，虚拟文件能够用于存储列式表的主部分和增量部分。这些文件可以在第一次访问相应表格时读入存储器。利用一些实施方式，当读访问只在数据的内存中表示上发生时，更新、添加、重写和删减还可以写到磁盘上的虚拟文件中。在将虚拟文件从源节点移动到目标节点之后，虚拟文件可以在第一次访问目标节点时读入存储器。为支持从日志备份中恢复，将虚拟文件从一个节点移动到另一个节点（如果不包括如下所述的技术）可能或者需要为所有移动数据在目标节点上写入重做日志或者需要在几个节点上进行恢复时的明显昂贵的同步，在两种情况下这都是太大的性能损失。

[0031] 当增量合并操作执行时，只能改变主存储的内容。因此当合并完成时，只能写入主虚拟文件。注意到，这并不意味着在合并操作期间主数据被写到磁盘中：当列存储器 305 写到虚拟文件时，数据可能被写入持久层 112 的页面缓冲器 304 中。确定虚拟文件中的数据

何时被实际刷新到磁盘（例如，在页面替换期间或者最晚在写入下一保存点时等）是持久层 112 的职责。

[0032] 增量合并操作是只有列存储器 305 才有的，并且不与持久层 112 的保存点同步。增量合并主要是在单个表格颗粒度上执行的内存中结构的优化。另一方面，保存点在整个数据库上工作，其目的是将改变保存到磁盘中。

[0033] 在列存储器 305 上执行的所有改变进入数据卷 310 中的增量存储器。增量存储器可能只存在于内存中，而不写到磁盘中。然而，列存储器 305 可经由记录器 344 来写入包含在增量存储器上执行的所有操作的逻辑重做日志项目的所保存的增量日志。在这个上下文中，逻辑日志是指操作及其参数被记录但没有存储物理映像。当执行增量合并操作时，增量存储器中的改变可以被合并到主存储器中而增量日志虚拟文件可以被删减。

[0034] 从持久层 112 的观点来看，增量日志虚拟文件实际上不是日志。对于持久层 112 来说，它们仅是数据。实际的重做日志和撤消项目可以写入持久层 112 中的日志卷 317。用于增量日志的虚拟文件可以被配置为已记录。每当列存储器 305 写到增量日志虚拟文件时，持久层接口 302 调用记录器 344 和撤消管理器来写入重做日志项目和撤消信息。这确保增量日志虚拟文件可以在重新启动之后就像任意其它数据一样被恢复。在增量日志虚拟文件被恢复之后，它们就准备好将由列存储器 305 处理以从逻辑增量日志项目重建内存中增量存储器。

[0035] 在增量合并操作期间，可以重写受影响的（多个）表格的主文件，并且可以删减增量日志文件。对于所有这些操作而言，持久层 112 都没有写日志。这是可能的，因为当增量文件作为初始的改变操作的一部分被写入时，已经记录了在表上执行的所有操作。合并操作没有改变、创建或删除数据库中的任何信息。它仅仅是存储现有信息的方法的重新组织。为防止为合并操作而写日志，虚拟的主文件被配置为不被记录，并且特殊的不被记录的操作被用于增量日志删减。

[0036] 在重新启动期间，持久层 112 可以从最近的保存点恢复主虚拟文件。增量日志虚拟文件可以从最近的保存点以及从重做日志中被恢复。当持久层 112 已经完成其部分时，列的主存储器可以从虚拟文件加载到列存储器内存中。这包含在持久层 112 的页面缓冲器 304 中的数据高速缓存和列存储器 305 中的连续内存区之间的内存复制操作。列存储器 305 能够因此由增量日志虚拟文件执行逻辑重做项目并重建内存中增量存储器。如上所述，存在允许为每个列式表格定义在系统启动期间它是否将被加载的元数据。如果表格被配置为在要求时加载，则那个表格的恢复序列在第一次访问时执行。

[0037] 如本文中所使用的，虚拟文件可以被表征为模拟文件系统中的文件的数据库对象。虚拟文件可以由诸如 8 字节 ID 的数字标识符（或间接由名称空间 + 名称）来标识。虚拟文件可以利用被认为是（关于并行写入的）原子块的一个写入流来支持流读写 I/O 请求。如上所述，虚拟文件可以（利用每列的一个虚拟文件和用于增量信息的虚拟文件）提供存储列式数据的基础。虚拟文件另外可以提供对二进制大型对象 (BLOB)、任意数据、SAP liveCache 可变大小对象等的支持。此外，虚拟文件可以提供一般的事务支持和事务隔离。读取虚拟文件遵守事务隔离标准，所以只有提交的、（根据事务隔离标准）已经可见的数据被读取。此外，虚拟文件具有充分的重新运算支持。

[0038] 虚拟文件还可以与数据存储应用程序 104 的剩余部分一起提供备份和恢复支持。

它们也提供包括最迟由保存点留存的改变的明确控制的重做日志以及对列式合并的支持。最后，虚拟文件可以提供对诸如（经由链接处理的）“缓慢的”横向文件移动的其它高级操作的支持。

[0039] 如本文中所描述的，虚拟文件可以固定在某种容器目录中，所述容器目录包含关于数据库对象的元数据并且被访问以确定最近的虚拟文件或其状态等。当虚拟文件存储在可以具有相同或不同的大小的数据库页面的链（chain）中时，开始和结束页面存储在元数据中。对所述链中的前一个和后一个页面的链接可以存储在独立的页头中。另外，对（随后会看到的）最后的重写记录的链接也可以存储在元数据中。在虚拟文件上的每个操作可以在该页面链上进行，并且在该页面链中创建具有小的头部和用户数据的（多个）记录。虚拟文件的每个页面通常可以包含至少一个记录。同样，最低（即最佳）数目的记录能够用于在虚拟文件中储存数据。

[0040] 添加到虚拟文件。为允许高度并行的操作，添加操作可以首先将对其流式传送的数据假脱机（spool）到内存中缓冲器。当流被结束时，该数据可以受制于在页面链末端的锁而被添加为一个或一系列数据记录（每个页面一个记录）。当受制于锁而添加数据时，整个添加是原子性的。

[0041] 为支持事务隔离和一致读取，每个数据记录可以在头部包括以下内容：(i) 创建 TID；(ii) 删除 TID；以及记录数据大小（数据直接接在头部之后）。创建 TID 可以被设置为创建数据记录（添加新数据）的事务的事务 ID。删除 TID 可以在创建时清除，并可以在删减 / 重写时被设置为删减事务的事务 ID（如下文进一步描述的）。

[0042] 当添加数据到虚拟文件的事务中止时，它可以仅仅清除由该事务添加的记录的创建 TID，有效地使新数据记录无效。当撤消操作检测到在页面上没有有效记录（即，所有记录被撤消等）时，页面可以从页面链中删除并直接回收（但这不是绝对必需的）。

[0043] 图 4 是说明了（跨越暂时部分重叠的三个时间线 410、420、430 的）一系列事务 T1-T7 的图 400，其中存在事务水平上的快照隔离。在 T1（时间线 410 中）的开始时，相应的虚拟文件最初是空的并且随后添加了记录 R1。但是在添加记录 R1 之前以及 T1 的开始之后，（在时间线 420 中）启动第二事务 T2，其中添加了第二记录 R2。第一记录 R1 作为第一事务 T1 的一部分被读取，第二记录 R2 作为第二事务 T2 的一部分被读取（第二事务 T2 只能读取第二记录 R2，因为第一记录 R1 还没有被提交）。此后，第一事务 T1 被提交，这导致第三事务 T3 被启动。并行地，（在时间线 430 中）第四事务 T4 被随后启动。此时，只有记录 R1 已经被提交，因此此时第四事务 T4 只能读取第一记录 R1。随后（在时间线 420 中），第二事务被提交并且第五事务 T5 被启动。此时，只有第一记录 R1 和第二记录 R2 已经被提交，因此第二事务 T2 读取这些记录 R1、R2。并行地，第三记录 R3 作为第四事务 T4 的一部分被添加，因此读操作读取第一记录 R1 和第三记录 R3（因为第四事务 T4 在提交添加第二记录 R2 的第二事务 T2 之前被启动）。此后，第四事务 T4 被提交，产生读取记录 R1、R2、R3（所有这些记录此时已经被提交）的第六事务 T6。相似地，第五事务 T5 被提交，产生同样读取记录 R、R2、R3 的第七事务 T7。

[0044] 删减 / 重写虚拟文件。删减和重写操作还可以受制于虚拟文件上的锁来操作。利用这样的安排，每个虚拟文件可以具有相关的锁，当文件上的改变操作在进行中时采用该锁。这包括完成添加操作（将数据从内存中缓冲器传送到虚拟文件）、删减操作和重写操作

（其是删减 + 添加新数据）。这确保在虚拟文件上的操作是串行化的并且内部结构不被破坏。

[0045] 删减操作能够添加特定删减记录，并设置在最近的可见位置（或者如下文进一步描述的，如果不存在最近的可见位置时的页面链的起点）和新的删减记录之间的可见数据记录的删减 TID。在重写情况下，流式数据可以作为虚拟文件末端的新数据记录而被写入（和添加数据一样）。

[0046] 为促进事务隔离和一致性，删减 / 重写虚拟文件可以给特定删减记录添加以下头部但不添加数据 :(i) 创建 TID ;(ii) 对最早的可读位置的链接（页面 + 偏移量）;以及 (iii) 对前一个删减记录的链接（页面 + 偏移量）。

[0047] 同时，对删减记录的链接可以在容器目录中的虚拟文件元数据中被更新为指向最新的删减记录。

[0048] 在一个实施方式中，删减 / 重写会如下所述地进行。当最近的已知删减记录存储在容器目录中的虚拟文件元数据中时，算法能够从最近的已知删减记录开始。如果该删减记录不可见并且因此删减本身不可见（即，删减记录的创建 TID 属于并行的或较新的事务），那么这是一个错误（并行事务删减虚拟文件）。如果不存在删减记录，则算法将虚拟文件的开始假设为一种删减记录。

[0049] 新的删减记录可以写入虚拟文件的末端，其具有被设为删减事务的 TID 的创建 TID，不设置对最早的可读位置的链接，并且如果前一删减记录存在，则对前一删减记录的链接可以设置为前一删减记录的位置，否则可以设置为虚拟文件的开始。对虚拟文件元数据中删减记录的链接可以被更新为指向新的删减记录。

[0050] 对最早的可读位置的链接可以从前一删减记录中读取，如果它存在的话（否则使用虚拟文件的开始）。该位置在前一文件删减操作期间被存储并指向最早的数据记录，其由比前一删减事务更新的或与其并行的事务写入。

[0051] 然后，删减记录的标记可以由该位置开始。对于找到的每个数据记录，该数据记录的创建 TID 被检查可见性。如果数据记录是可见的（即其创建 TID 属于比删减事务早的事务并且创建事务不并行于删减事务），那么其删减 TID 可以被设置为删减事务的 TID。如果数据记录不可见，其删减 TID 没有被设置，并且它是第一个这种不可见的记录，则在新的删减记录中对其位置的链接被存储为最早的可读位置。算法可以重复用于所有记录，直到新的删减记录。

[0052] 如果没有找到最早的可读位置，则意味着在并行事务中已经没有记录被添加。在这种情况下，对最早的可读位置的链接可以设置为指向新的删减记录。

[0053] 在重写情况下，在删减结束之后，新的数据可以几乎与如上所述的添加情况下一样地被添加。

[0054] 当删减虚拟文件的事务中止时，它可以仅仅清除在删减操作期间设置删减 TID 的所有记录的删减 TID，然后它可以将虚拟文件元数据中的链接重置为指向前一删减记录，然后它可以使新的删减记录无效（例如，通过清除创建 TID，所以记录将被其它操作忽略）。

[0055] 图 5 是说明跨越暂时部分重叠的两个时间线 510 和 520 的一系列事务 T1-T5 的图 500。当（在时间线 510 中）启动第一事务 T1 的时候，相应虚拟文件包含三个记录 R1、R2、R3。此后，添加第四记录 R4（并且由读取所有四个记录 R1、R2、R3、R4 的第一事务 T1 接着执

行读操作）。此后，（在时间线 520 中）启动第二事务 T2。此时，第二事务读取所有三个记录 R1、R2、R3。第二事务 T2 启动使所有记录被清除的删减操作。然而，该删减不影响第一事务 T1，直到它已经被提交。同时，第五记录已经添加 R5 并被提交。因此，在第二事务 T2 被提交时，虚拟文件包含记录 R4 和 R5（因为它们不是删减的一部分），所述记录 R4 和 R5 作为（时间线 520 中）随后的第四事务 T4 和（时间线 510 中）随后的第五事务 T5 的一部分被读取。

[0056] 读取虚拟文件。首先，可以确定用于读取事务的最早的可见数据记录。读取事务可以“了解”从最近的可见删减记录开始的所有可见数据记录以及当删减运行时在并行事务中添加的任意数据记录。为确定最初读取位置，当最近的已知删减记录存储在容器目录中的虚拟文件元数据中时，算法能够从最近的已知删减记录开始。如果该删减记录不可见并且因此删减本身不可见（即，删减记录的创建 TID 属于并行的或较新的事务），那么可以遵循对前一删减记录的链接并且算法可以利用前一删减记录进行重复。

[0057] 当确定最近的可见删减记录时，对最早的可读位置的链接可以从该删减记录中读取。该位置可以在文件删减操作期间被存储并指向最早的数据记录，其由比删减事务更新的或与其并行的事务写入（参见上文）。这也是第一记录，读取从这里开始。

[0058] 如果没有找到删减记录，那么读取可以由虚拟文件的起点开始。

[0059] 利用该信息，虚拟文件的读取者可以仅仅读取起始于对该读者最早可见的数据记录的所有记录，其中创建 TID 可能是可见的（即，属于不比自己的 TID 更新的并且不与读取事务并行运行的事务），并且删减 TID 可以被清除或不可见（即，属于比自己的 TID 更新的或与读取事务并行运行的事务）。该读操作可以重复任意多次，并且对于同一个事务将总是返回相同的数据（除了当事务本身修改虚拟文件的时候）。

[0060] 虚拟文件的垃圾收集。由于总是在最后添加数据，因此虚拟文件将无限地增长，甚至在存在删减的时候。但是只有任意运行事务所看得见的数据是需要的，旧的数据可以被垃圾收集。当所有读取事务的最低 TID 超过删减记录的 TID 时，可以为每个删减记录执行垃圾收集。可以由例如数据库的历史管理器来执行最小读取 TID 的追踪和垃圾收集本身的执行。重要的是，仅为记录执行垃圾收集，任意读取事务均不需要垃圾收集。

[0061] 虚拟文件的垃圾收集可能非常简单 - 当存储在删减记录中时，最早的可读位置之前的虚拟文件的页面链中的所有页面可以从页面链中删减并返回空闲空间管理。在影子页面调度数据库（如上所述）中，这些页面可以作为影子页面继续留在磁盘上，直到执行将随后回收空闲空间的下一保存点。换句话说，当垃圾收集时逻辑空间被释放，而当保存点时物理空间被释放。

[0062] 虚拟文件上的特殊操作。虚拟文件可以支持许多特殊操作。为了支持这种操作，可以给虚拟文件添加表示所述操作的扩展数据记录。然后，这种操作可以像任意普通添加操作一样被删减和垃圾收集。

[0063] 本文描述的主题的各方面可以体现在根据期望配置的系统、设备、方法和 / 或物品中。特别地，本文描述的主题的各种实施方式可以以数字电子电路、集成电路、特别设计的专用集成电路 (ASIC)、计算机硬件、固件、软件和 / 或其组合的形式来实现。这些各种实施方式可以包括用一个或多个计算机程序的实施方式，所述计算机程序在包括至少一个可编程处理器的可编程系统上是可执行和 / 或可解释的，所述可编程处理器可以是专用或通

用的，被耦合以从存储系统、至少一个输入装置和至少一个输出装置接收数据与指令并发送数据与指令到存储系统、至少一个输入装置和至少一个输出装置。

[0064] 还可以称为程序、软件、软件应用程序、应用程序、组件或代码的这些计算机程序包括用于可编程处理器的机器指令，并且可以以高级过程和 / 或面向对象的编程语言和 / 或以汇编 / 机器语言来实现。如本文中所使用的，术语“机器可读介质，，是指任意计算机程序产品、设备和 / 或装置，例如诸如磁盘、光盘、内存和可编程逻辑器件 (PLD)，用于提供机器指令和 / 或数据给可编程处理器，包括将机器指令接收为机器可读信号的机器可读介质。术语“机器可读信号”是指用于提供机器指令和 / 或数据给可编程处理器的任意信号。机器可读介质可以非短暂地存储这种机器指令，例如诸如非短暂的固态存储器或磁硬盘或任意等效存储介质那样。机器可读介质可以可替换地或另外以短暂的方式存储这种机器指令，例如诸如处理器高速缓存或与一个或多个物理处理器核心相关的其它随机存取存储器那样。

[0065] 本文描述的主题可以实现在计算系统中，所述计算系统包括例如诸如用于一个或多个数据服务器的后端组件，或包括例如诸如用于一个或多个应用服务器的中间件组件，或包括例如诸如用于具有图形用户界面或网页浏览器（用户可以通过图形用户界面或网页浏览器与本文描述的主题的实施方式互动）的一个或多个客户端计算机的前端组件，或者这种后端、中间件或前端组件的任意组合。客户机和服务器通常，但不是全部彼此远离，并且通常通过通信网络相互配合，尽管系统的组件可以通过任意形式或介质的数字数据通信来互连。通信网络的示例包括但不限于局域网（“LAN”）、广域网（“WAN”）和互联网。客户机和服务器的关系由于运行在各自的计算机上并具有相对彼此的客户机 - 服务器关系的计算机程序的功效而发生。

[0066] 在上述说明部分阐述的实施方式不代表和本文描述的主题一致的所有实施方式。相反，它们仅仅是和与描述主题相关的某些方面一致的一些示例。尽管本文已经详细描述了几个变化，但其它修改或增加也是可能的。尤其是，除了本文阐述的那些之外可以提供更多的特征和 / 或变化。例如，如上所述的实施方式可以用于所公开特征的不同组合以及子组合和 / 或相对于本文所公开的那些特征更多的一个或多个特征的组合以及子组合。此外，在附图中描述的和 / 或本文中描述的逻辑流程不是必须要所显示的特定顺序或连续顺序来达到想要的结果。随后的权利要求的范围可包括其它实施方式或实施例。

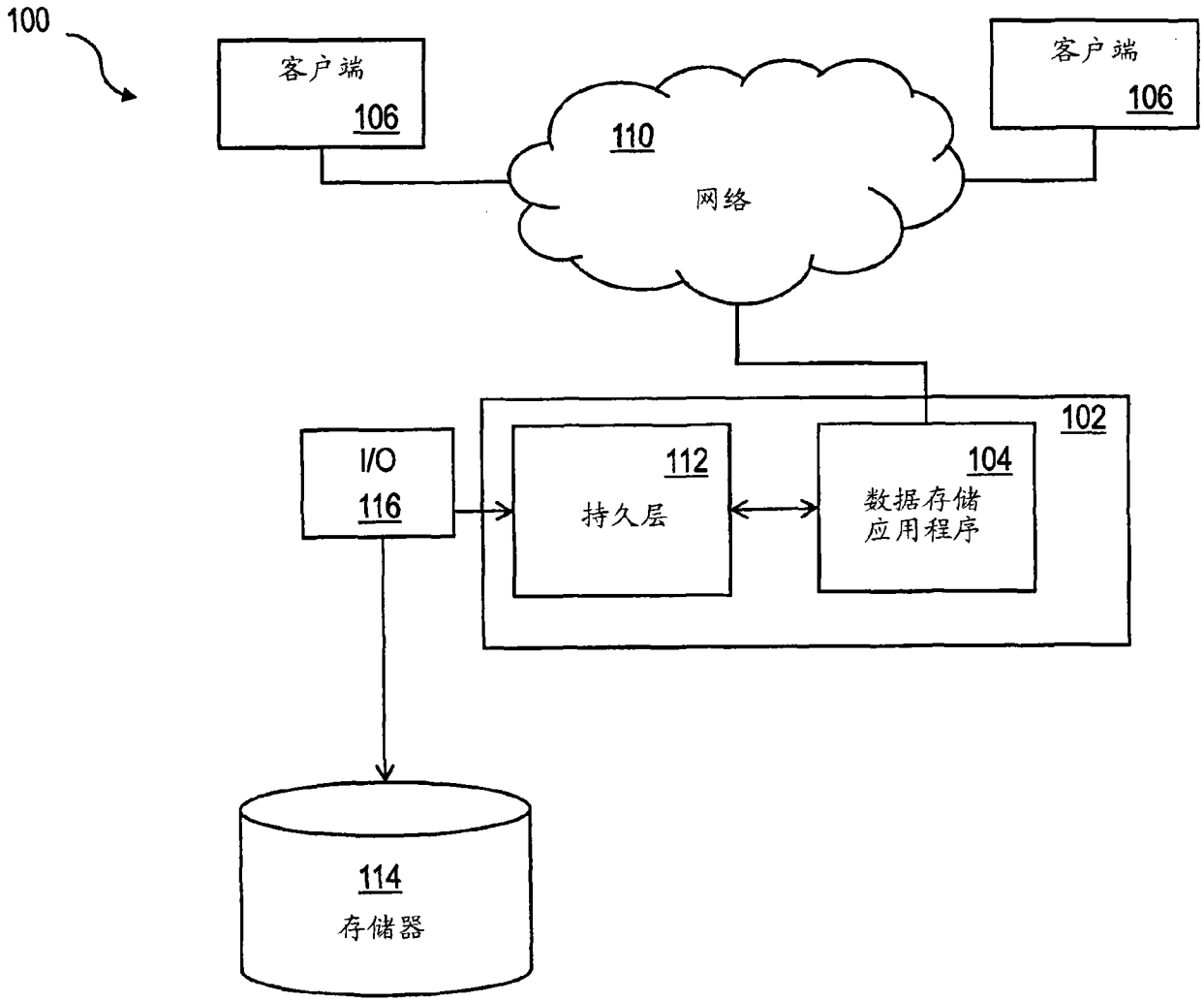

100
客户端
106
客户端
106
110
网络
102
I/O
116
112
持久层
104
数据存储
应用程序
114
存储器

图 1

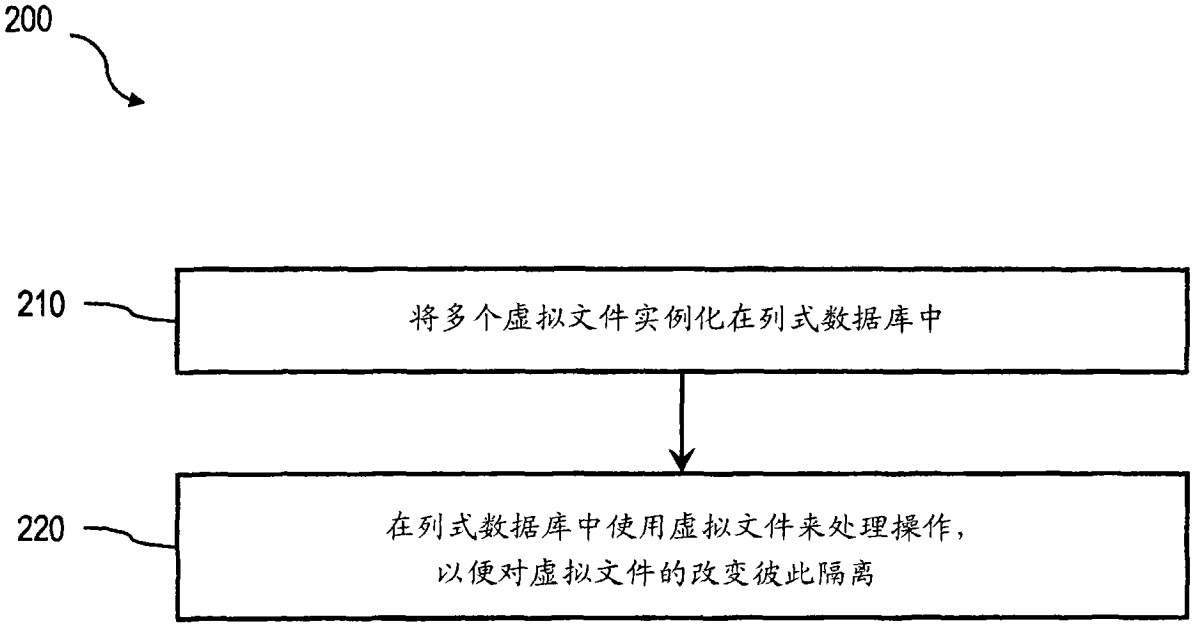

200
210
将多个虚拟文件实例化在列式数据库中
220
在列式数据库中使用虚拟文件来处理操作，
以便对虚拟文件的改变彼此隔离

图 2

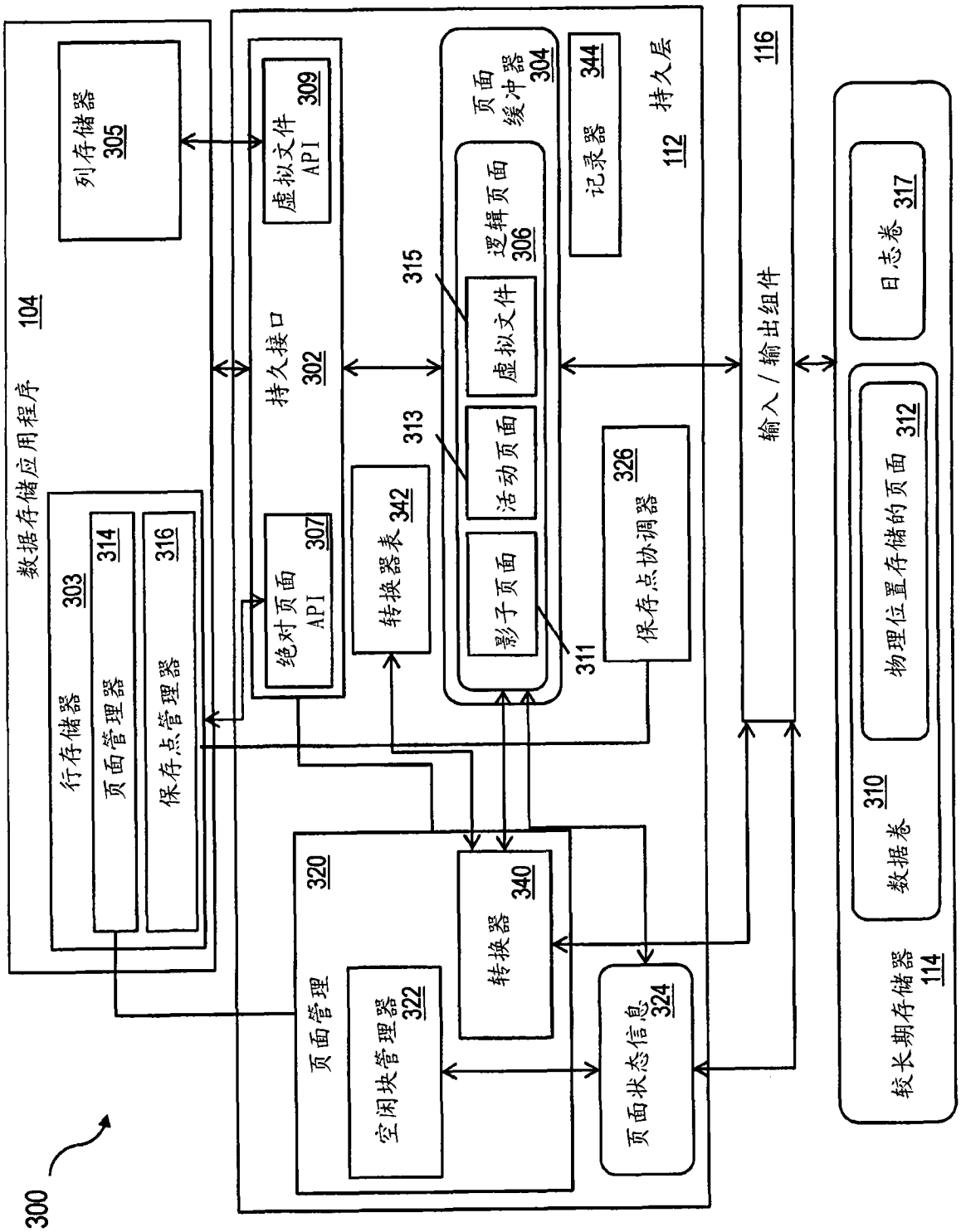

数据存储应用程序 104
行存储器 303
页面管理器 314
保存点管理器 316
列存储器 305
持久接口 302
虚拟文件 API 309
绝对页面 API 307
转换器表 342
315
313
页面缓冲器 304
逻辑页面 306
虚拟文件
活动页面
影子页面
311
记录器 344
持久层 112
保存点协调器 326
输入 / 输出组件 116
日志卷 317
物理位置存储的页面 312
数据卷 310
较长期存储器 114
页面管理 320
空闲块管理器 322
转换器 340
页面状态信息 324
300

图 3

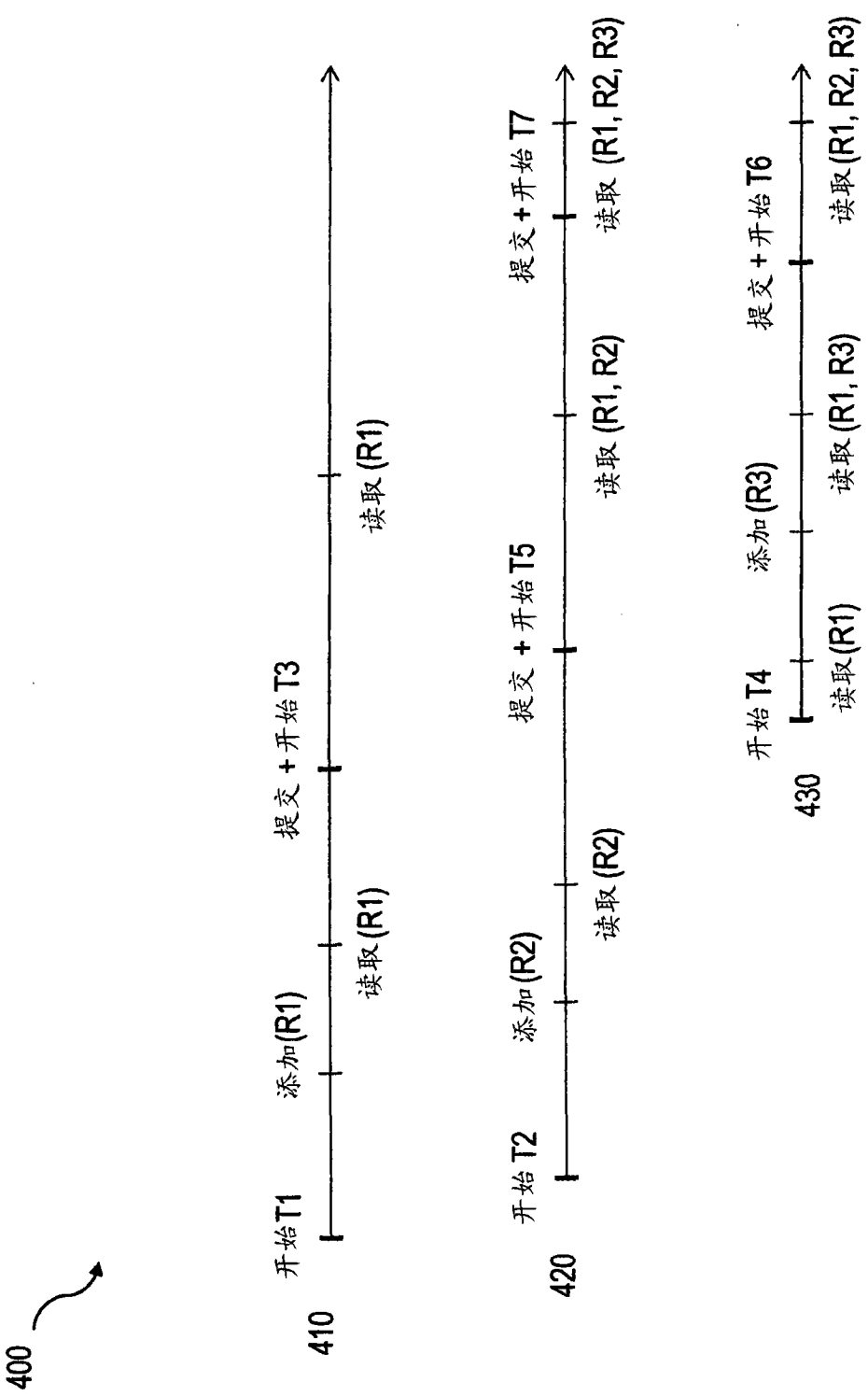

400
410
开始T1
添加(R1)
提交 + 开始 T3
读取(R1)
读取 (R1)
420
开始 T2
添加(R2)
提交 + 开始 T5
读取 (R2)
读取 (R1, R2)
提交 + 开始 T7
读取 (R1, R2, R3)
430
开始 T4
添加(R3)
提交 + 开始 T6
读取(R1)
读取 (R1, R3)
读取(R1, R2, R3)

图 4

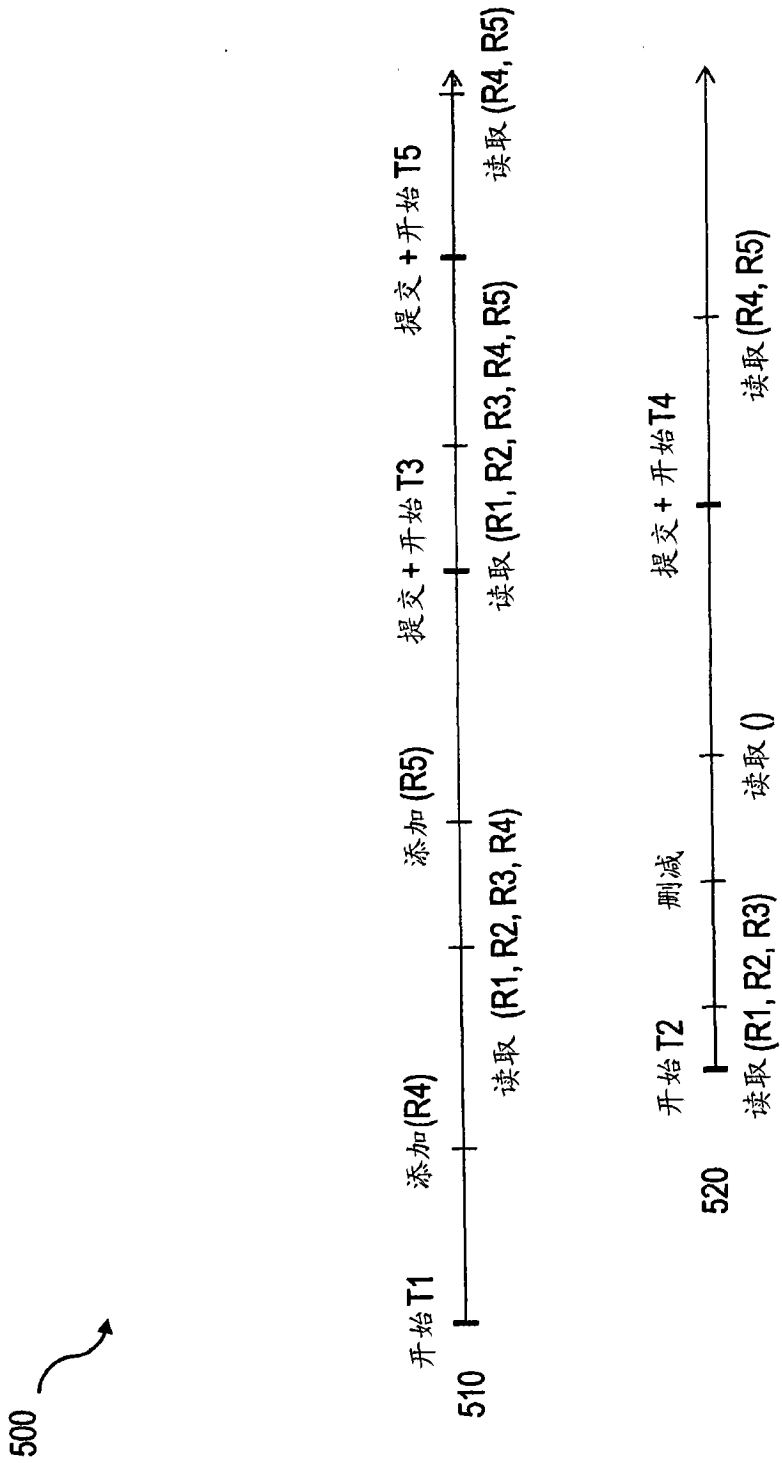

500
开始 T1
添加(R4)
读取 (R1, R2, R3, R4)
添加 (R5)
提交 + 开始 T3
读取 (R1, R2, R3, R4, R5)
提交 + 开始 T5
读取 (R4, R5)
510
开始 T2
删减
读取 (R1, R2, R3)
提交 + 开始T4
读取 ()
读取 (R4, R5)
520

图 5