(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 105630860 A
(43)申请公布日 2016.06.01

(21)申请号 201510809717.5

(22)申请日 2015.11.20

(30)优先权数据
14/553,494 2014.11.25 US

(71)申请人 SAP欧洲公司
地址 德国瓦尔多夫

(72)发明人 I. 施雷特 J. 李 M. 安德雷

(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 邵亚丽 金玉洁

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

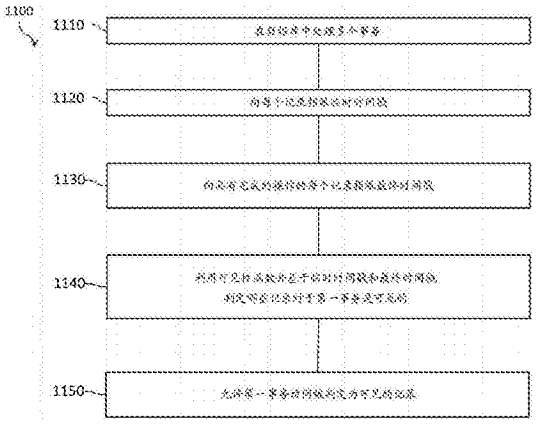
权利要求书2页 说明书8页 附图11页

(54)发明名称

具有事务控制块索引的数据库系统

(57)摘要

在数据库中处理多个事务。每个事务包括在数据库中的至少一个记录上的多个操作，其中这些事务中的至少两者被并发地处理。然后，向每个记录指派临时时间戳。临时时间戳至少部分基于相应的事务。另外，向具有提交操作的每个记录指派最终时间戳。随后可利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的。基于这种判定，可向第一事务提供对被判定为可见以便访问的那些记录的访问。



CN 105630860 A

1. 一种由形成至少一个计算设备的一部分的一个或多个硬件数据处理器实现的方法，该方法包括：

在数据库中处理多个事务，每个事务包括在所述数据库中的至少一个记录上的多个操作，所述事务中的至少两者被并发地处理；

向每个记录指派临时时间戳，该临时时间戳至少部分基于相应的事务；

向具有提交操作的每个记录指派最终时间戳；

利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的；以及

允许被判定为可见的记录被所述第一事务访问。

2. 如权利要求 1 所述的方法，其中，每个事务具有这种事务特定的相应事务控制块索引。

3. 如权利要求 1 所述的方法，其中，每个时间戳是由提交时间戳生成子生成的。

4. 如权利要求 3 所述的方法，还包括：

选择性地锁定所述提交时间戳生成子以防止其发出额外的时间戳。

5. 如权利要求 3 所述的方法，还包括：

通过以下步骤在提交操作执行时将用于第一记录的临时时间戳转换成最终时间戳：

进入提交锁定；

将用于所述第一记录的提交时间戳设置为来自提交时间戳生成子加 1 的值；

发出写入存储器屏障；

将所述提交时间戳生成子值增大 1；以及

释放所述提交锁定。

6. 如权利要求 1 所述的方法，其中，所述数据库包括存储器内数据库。

7. 如权利要求 6 所述的方法，其中，所述存储器内数据库是按列存储数据的存储器内列式数据库。

8. 一种存储指令的非暂态计算机程序产品，所述指令当被形成至少一个计算设备的一部分的至少一个硬件数据处理器执行时导致包括以下在内的操作：

在数据库中处理多个事务，每个事务包括在所述数据库中的至少一个记录上的多个操作，所述事务中的至少两者被并发地处理；

向每个记录指派临时时间戳，该临时时间戳至少部分基于相应的事务；

向具有提交操作的每个记录指派最终时间戳；

利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的；以及

允许被判定为可见的记录被所述第一事务访问。

9. 如权利要求 8 所述的计算机程序产品，其中，每个事务具有这种事务特定的相应事务控制块索引。

10. 如权利要求 8 所述的计算机程序产品，其中，每个时间戳是由提交时间戳生成子生成的。

11. 如权利要求 10 所述的计算机程序产品，其中，所述操作还包括：

选择性地锁定所述提交时间戳生成子以防止其发出额外的时间戳。

12. 如权利要求 10 所述的计算机程序产品，其中，所述操作还包括：

通过以下步骤在提交操作执行时将用于第一记录的临时时间戳转换成最终时间戳：

进入提交锁定；

将用于所述第一记录的提交时间戳设置为来自提交时间戳生成子加 1 的值；

发出写入存储器屏障；

将所述提交时间戳生成子值增大 1；以及

释放所述提交锁定。

13. 如权利要求 8 所述的计算机程序产品，其中，所述数据库包括存储器内数据库。

14. 如权利要求 13 所述的计算机程序产品，其中，所述存储器内数据库是按列存储数据的存储器内列式数据库。

15. 一种系统，包括：

至少一个硬件数据处理器；以及

存储指令的存储器，所述指令当被形成至少一个计算设备的一部分的至少一个硬件数据处理器执行时导致包括以下在内的操作：

在数据库中处理多个事务，每个事务包括在所述数据库中的至少一个记录上的多个操作，所述事务中的至少两者被并发地处理；

向每个记录指派临时时间戳，该临时时间戳至少部分基于相应的事务；

向具有提交操作的每个记录指派最终时间戳；

利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的；以及

允许被判定为可见的记录被所述第一事务访问。

16. 如权利要求 15 所述的系统，其中，每个事务具有这种事务特定的相应事务控制块索引。

17. 如权利要求 15 所述的系统，其中，每个时间戳是由提交时间戳生成子生成的。

18. 如权利要求 17 所述的系统，其中，所述操作还包括：

选择性地锁定所述提交时间戳生成子以防止其发出额外的时间戳。

19. 如权利要求 17 所述的系统，其中，所述操作还包括：

通过以下步骤在提交操作执行时将用于第一记录的临时时间戳转换成最终时间戳：

进入提交锁定；

将用于所述第一记录的提交时间戳设置为来自提交时间戳生成子加 1 的值；

发出写入存储器屏障；

将所述提交时间戳生成子值增大 1；以及

释放所述提交锁定。

20. 如权利要求 15 所述的系统，其中，所述至少一个处理器和所述存储器形成按列存储数据的存储器内列式数据库的一部分。

具有事务控制块索引的数据库系统

技术领域

[0001] 本文描述的主题涉及一种利用事务控制块索引的数据库系统，该事务控制块索引使得实时事务能够读取其自己对其他事务不可见的记录的写入。

背景技术

[0002] 一些数据库使用基于时间戳的多版本并发控制(multi-version concurrency control，MVCC)来最小化关于同时事务的冲突。利用此布置，数据库中的记录被（在逻辑上）标记以与创建和删除了该记录的事务相对应的创建和删除时间戳。如果记录尚未被删除，则删除时间戳被取消设置（这可由某个特殊值表示，例如可表示的最大整数值）。

[0003] 需要数据库上的一致视图（或快照）来确保SQL语句的或者整个事务的一致执行。一致视图是通过取被附加到或关联到每个记录的提交时间戳生成子（即，时间戳）的当前值来创建的。此时间戳随后被对照创建和删除时间戳(CTS、DTS)使用来决定记录的可见性。

[0004] 记录在以下情况下被认为可见：如果其时间戳（倘若被取消设置/不可见的时间戳被定义为可表示的最大整数值）小于或等于存储在一致视图(consistent view，CV)中的时间戳。

[0005] visible(CV，TS) ＝ TS ≤ CV.TS

[0006] 如果创建时间戳可见并且删除时间戳不可见，则记录因此是可见的。

[0007] visible(record) ＝ visible(CV，record.CTS) ∧ -visible(CV.record.DTS)

发明内容

[0008] 在一个方面中，在数据库中处理多个事务。每个事务包括在数据库中的至少一个记录上的多个操作，其中这些事务中的至少两者被并发地处理。然后，向每个记录指派临时时间戳。临时时间戳至少部分基于相应的事务。另外，向具有提交操作的每个记录指派最终时间戳。随后可利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的。基于这种判定，可向第一事务提供对被判定为可见以便访问的那些记录的访问。

[0009] 每个事务可具有这种事务特定的相应事务控制块索引。每个时间戳可由提交时间戳生成子生成。可选择性地锁定提交时间戳生成子以防止其发出额外的时间戳。可通过以下操作在提交操作执行时将用于第一记录的临时时间戳转换成最终时间戳：进入提交锁定，将用于第一记录的提交时间戳设置到来自提交时间戳生成子加1的值，发出写入存储器屏障，将提交时间戳生成子值增大1，并且释放提交锁定。

[0010] 也描述了存储指令的非暂态计算机程序产品（即，物理实现的计算机程序产品），这些指令在被一个或多个计算系统的一个或多个数据处理器执行时使得至少一个数据处理器执行这里的操作。类似地，也描述了计算机系统，这些计算机系统可包括一个或多个数据处理器和耦合到这一个或多个数据处理器的存储器。存储器可临时或永久地存储指令，

这些指令使得至少一个处理器执行本文描述的操作中的一个或多个。此外，方法可由单个计算系统内的或者分布在两个或更多个计算系统间的一个或多个数据处理器实现。这种计算系统可被连接并且可经由一个或多个连接、经由多个计算系统中的一个或多个之间的直接连接等等来交换数据和／或命令或其他指令等等，所述一个或多个连接包括但不限于通过网络（例如，因特网、无线广域网、局域网、广域网、有线网络等等）的连接。当前主题可包括并包含存储器内数据库，并且具体地包括列式存储器内数据库。

[0011] 本文描述的主题提供了许多技术优点。例如，当前主题与传统技术相比就存储器消耗和执行时间而言更高效。例如，一种替换方案是明确地跟踪由事务创建和删除的一组记录和在这组记录中的查找。这种布置的不利之处在于当与当前主题相比较时其要求更复杂的代码并且提供更低的性能。这个替换方案的不利之处还在于随着多线程化而发生的问题，因为必须保护共享的结构；利用当前布置，不要求共享的结构。在另一替换方案中，可以使用一种不同的模型，其将事务标识符而不是提交时间戳存储在记录上，同时将开放事务列表保持为一致视图的一部分。与当前主题相比，利用开放事务列表创建一致视图和利用的可见性函数都比基于提交时间戳的解决方案更加资源昂贵。此外，利用当前主题，不需要在提交点对事务重定时间戳。另外，当前主题的有利之处也在于其允许实时事务读取其自己的写入，这些写入对其他事务是不可见的。

[0012] 本文描述的主题的一个或多个变化的细节在附图和下面的描述中记载。本文描述的主题的其他特征和优点将从描述和附图以及从权利要求中清楚显现。

附图说明

[0013] 图 1 是图示出业务软件系统体系结构的特征的图；

[0014] 图 2 是图示出业务软件系统体系结构的特征的另一幅图；

[0015] 图 3 是主存储中存储的片段的示意性表示；

[0016] 图 4 是图示出统一表格容器页链的特征的图；

[0017] 图 5 是图示出统一表格增量的特征的图；

[0018] 图 6 是图示出统一表格未排序字典的特征的图；

[0019] 图 7 是图示出利用统一表格执行增量合并操作和读取操作的功能框图；

[0020] 图 8 是图示出提交处理如何将临时时间戳转换成最终时间戳的过程流程图；

[0021] 图 9 是图示出进行提交的线程如何可以是若干个提交并行事务之一的过程流程图；

[0022] 图 10 是图示出提交线程如何可以是若干个提交并行事务之一的过程流程图；并且

[0023] 图 11 是用于在利用事务控制块索引的数据库系统中提交事务的过程流程图。

[0024] 各幅图中的相似附图标记指示相似的元素。

具体实施方式

[0025] 当前主题包括数个方面，这些方面可被单独应用或者将一个或多个这种方面组合应用，来支持一种统一数据库表格方案，该方案将存储器内数据库方案的性能优点与盘上数据库方案的降低的存储成本相集成，尤其对于并发事务的处理更是如此。当前主题可在

以下系统中实现：使用存储器内 OLAP 的数据库系统，例如包括大小为若干万亿字节（或更大）的数据库，具有数十亿（或更多）行的表格，等等；使用存储器内 OLTP 的系统（例如，企业资源规划或 ERP 系统等等），例如具有高事务量的大小为若干万亿字节（或更大）的数据库；以及使用盘上 OLAP 的系统（例如，“大数据”，用于高级分析的分析服务器、数据仓库、业务智能环境，等等），例如大小为若干千万亿字节或甚至更大的数据库、具有多达数万亿行的表格，等等。

[0026] 当前主题可实现为企业资源规划 (enterprise resource planning，ERP) 系统的核心软件平台、其他业务软件体系结构或者在特定组织的控制下的一个或多个处理器上运行的其他数据密集型计算应用或软件体系结构。这个布置对于这样的大规模组织可能是非常有效的：这些组织具有非常富有经验的内部信息技术 (IT) 职员，并且对于这些组织，在定制市售的业务软件解决方案以结合组织特定的业务流程和功能一起工作所需要的计算硬件和咨询服务上的相当大的资本投入是可行的。图 1 示出了符合这种实现方式的系统的图 100。计算系统 110 可包括提供业务软件系统的一个或多个特征的一个或多个核心软件平台模块 120。该计算系统也可聚合或以其他方式提供一网关，经由该网关，用户可访问由一个或多个外部软件组件 130 提供的功能。客户端机器 140 可经由直接连接、本地终端或者通过网络 150（例如，局域网、广域网、无线网络、因特网，等等）访问该计算系统。

[0027] 数据库管理代理 160 或其他相当的功能可访问数据库管理系统 170（有时就称为数据库），数据库管理系统 170 存储并提供对数据的访问（例如，业务场景、业务流程和一个或多个业务配置的定义，以及与业务场景、业务流程和一个或多个业务配置的定义有关的数据、元数据、主数据等等，和 / 或与业务场景或业务流程的特定实例相关的数据对象和 / 或业务对象的具体实例，等等）。数据库管理系统 170 可包括至少一个表格 180，并且还包括符合本文描述的那些的并行化特征。

[0028] 图 2 示出了图示可包括在符合当前主题的实现方式的数据库或数据库管理系统中的特征的体系结构 200 的框图。可被保存在多个数据卷 204 之间的表格数据存储 202 可包括以下各项中的一个或多个：增量存储 206（例如，分页增量部分，其可以可选地是 OLTP 优化的并且可以可选地包括合并过程 208）、索引存储 212（例如，一个或多个分段索引）以及主存储 210。主存储 210 可包括符合本文描述的特征的分成片段的主部分。

[0029] 为了实现最佳可能压缩并且也为了支持非常大的数据表格，表格的主部分可被划分成一个或多个片段。图 3 示出了主存储 210 中存储的各种片段的示意性表示。一个或多个主片段或片段 330 可用于数据库的每个表格或列。小的、易管理的表格可利用单个片段来表示。非常大的表格可被分割成两个或更多个表格分区 335。每个表格分区进而可包括两个或更多个片段 330。片段 330 可以是其所属的表格的水平切片。每个片段 330 可包括一个或多个列片段 340。每个列片段 340 可具有符合本文描述的特征的其自己的字典和值 ID 阵列。

[0030] 片段 330 可有利地充分大以获得由于片段的最优化压缩和聚合和扫描的高存储器内性能而产生的最大性能。相反，这种片段可充分地小以将任何给定片段的最大列加载到存储器中并且在存储器内对该片段排序。片段也可充分地小以能够将两个或更多个部分为空的片段合并成更小数目的片段。作为这个方面的说明性而非限制性示例，一片段可包含十亿行，每列最多有 100GB 的数据。其他片段大小也在当前主题的范围内。片段可以可

选地包括页的链。在一些实现方式中，列也可包括页的链。列数据可例如利用字典和 / 或任何其他压缩方法来加以压缩。表格片段可被实体化在存储器内的连续地址空间中以获得最大性能。数据库的所有片段可被存储在盘上，并且对这些片段的访问可基于对查询的数据访问要求的分析来作出。

[0031] 再次参考图 2，体系结构 200 的其他部分可包括数据操纵语言 (data manipulation language，DML) 处理模块或类似功能 214，一个或多个查询处理模块或类似功能 216（例如包括多版本并发控制）、支持索引存储 212 的索引构建器 220、查询语言引擎 222（其可以例如是 SQL 引擎）、用于接收来自用户 226 的输入的复杂事件处理模块（例如，事件处理程序、流处理模块，等等）224，等等。

[0032] 图 4 示出了图示统一表格容器页链 400 的示例的框图。如上所述，每个片段可以可选地包括页的链。一般地，容器可被表示为页链。页链一般可被表征为以给定顺序链接的页的集合。本文使用的术语“页”指的是数据库中的存储的基本单位。页大小一般是在构建数据库时确立的并且通常不能被改变。代表性的页大小可以是 2kB、4kB、8kB、16kB 等等级别的。一旦构建了服务器，在一些情况中该值通常就不能被改变，而在其他情况中，可以使用可变页大小。不同类型的页可存储不同类型的数据库对象。例如，数据页可以为表格存储数据行或列。索引页可以为索引的一个或多个级别存储索引行。大对象 (large object，LOB) 页可以为文本和图像列、为 Java 行外列等等存储数据。

[0033] 也如图 4 所示，可对增量部分、主部分、字典、索引段（可选，在图 2 中没有示出）等等定义页链的子链，使得这些实体的每一者的“整体”包含一个或多个页。在当前主题的一些实现方式中，增量部分可包括“热”增量片段 402 和“冷”增量片段 404 两者，这两者可分开存储。主部分也可被再分成主片段 330。包含字典压缩列式数据 410 的页可以指为它们包含字典的页。个体表格部分可根据需要被加载到主存储器中。合并过程可与事务处理解除耦合，使得合并过程可在恢复时（例如在日志重放期间）执行。页链，例如图 4 所示的示例，可由容器目录条目 (container directory entry，CDE)412 发起。

[0034] 单个 RowID 空间可在页链中跨页使用。一般指代数据库中的逻辑行的 RowID 可用于指代数据库的存储器内部分中的逻辑行并且也指代数据库的盘上部分中的物理行。行索引通常指的是表格中的行的物理的基于 0 的索引。基于 0 的索引可用于对连续阵列中的行物理寻址，其中逻辑 RowID 表示行的逻辑顺序，而不是物理位置。在一些存储器内数据库系统中，用于数据记录位置的物理标识符可被称为 UDIV 或 DocID。与逻辑 RowID 不同，UDIV 或 DocID（或者相当的参数）可指示行（例如数据记录）的物理位置，而 RowID 指示逻辑位置。为了允许表格的分区具有符合当前主题的实现方式的单个 RowID 和行索引空间，对于新插入的记录以及对于跨片段的更新记录的新版本，RowID 可被指派单调递增的 ID。换言之，例如，更新记录将改变其 RowID，因为更新实际上是（具有 RowID 的）旧记录的删除和（具有新 RowID 的）新记录的插入。利用此方案，表格的增量存储可按 RowID 来排序，这可用于访问路径的最优化。分开的物理表格实体可按分区来存储，并且这些分开的物理表格实体可在查询级别被连接成逻辑表格。

[0035] 当在列式合并操作期间执行最优化压缩以将增量存储中记录的改变添加到主存储时，表格中的行一般被重排序。换言之，合并操作之后的行通常不再是按其物理行 ID 排序的。因此，根据当前主题的一个或多个实现方式可使用稳定的行标识符。稳定行标识符

可以可选地是逻辑 RowID。对稳定的逻辑（而不是物理）RowID 的使用可允许在提前写入日志和事务撤销日志中的 REDO/UNDO 条目中寻址行。此外，以这种方式可促进跨合并稳定、而不会保持对数据库的旧主版本的引用的光标。为了使能这些特征，可存储存储器内逻辑 RowID 到物理行索引的映射和相反方向的映射。在当前主题的一些实现方式中，RowID 列可被添加到每个表格。RowID 列在当前主题的一些实现方式中也可服从压缩。

[0036] 图 5 示出了符合当前主题的一个或多个实现方式的统一表格增量 500 的框图。在一些示例中，可使用“热”和“冷”增量方案，其中未压缩的数据被保存在“热”增量部分中，而字典压缩的数据被保存在“冷”增量部分中，在热和冷部分之间执行微型合并。这种增量部分可被认为是单个容器。如图 5 所示，每个增量子链可具有其自己的暂态结构。换言之，对于每个增量可使用单独的结构。页向量 502 可保持到个体页 504 的页句柄并且可允许页 504 上的快速迭代（例如作为列或表格扫描的一部分）。到个体页 504 的页句柄可包括被保持在存储器中的钉 (pin) 等等。当在本文中使用时，术语“钉”指的是将特定数据页（其也可能已被存储在盘上）保持在存储器中。作为示例，如果页未被钉住，则其可被从存储器清除。钉住通常在被活跃访问的数据页上进行以避免与将页从盘读取到存储器中相关联的潜在性能劣化。

[0037] RowID 索引 506 可充当搜索结构以允许基于 RowID 值的给定间隔来找到页 504。搜索时间可以是 log n 级别的，其中 n 非常小。RowID 索引可经由 RowID 值提供对数据的快速访问。为了优化，“新”页在 RowID 和行索引之间可具有 1:1 关联，使得简单数学（没有查找）操作成为可能。在当前主题的至少一些实现方式中，只有被合并过程重组织的页需要 RowID 索引。

[0038] 图 6 示出了统一表格未排序字典 600 的框图。根据当前主题的一个或多个实现方式，增量部分中的列数据可使用未排序字典。对每个增量列字典可提供暂态结构。页向量 602 可处理页在存储器中的钉住。从其他结构经由指针可提供直接访问。值向量间接 602 可允许每个字典块 604 相同数目的值。此能力对于按 ValueID 查找值可支持 1 级别的性能成本。字典可向每个唯一值指派唯一的 ValueID（通常是数值），使得唯一值（通常在存储器大小上大于 ValueID）可被存储一次而不是多次。值阵列是被字典用来在给定 ValueID 的情况下检索值或者在给定值的情况下检索 ValueID 的结构。此技术可减小在值不唯一的情况下存储一组值所需要的存储器量，其通常被称为字典压缩。值到 ValueID 映射 606 对于按值查找 ValueID 可支持 1 级别的或者 log n 级别的散列或 B 树大小。B 树是一种使数据保持被排序并允许对数时间的搜索、顺序访问、插入和删除的树数据结构。此能力对于字典压缩可能是必要的。B 树对于范围扫描可能更好，但维护起来可能是更昂贵的。

[0039] 图 7 示出了用于在统一表格上执行增量合并操作 710 的功能框图 700。新的事务或变化可最初被写入到增量存储 206 中。主存储 210 可包括一个活跃片段 712 和一个或多个关闭片段 716。当更新被从增量存储 206 合并到主存储 210 中时，关闭片段 716 中的现有记录不能被改变。反而，记录的新版本可被添加到活跃片段 712，并且旧版本可被标记为无效。

[0040] 功能框图 700 也图示了读取操作 720。一般地，读取操作可具有对所有片段（即，活跃片段 712 和关闭片段 716）的访问。可通过仅加载包含来自特定查询的数据的片段来优化读取操作。不包含这种数据的片段可被排除。为了做出此决策，可对每个片段存储容

器级元数据（例如，最小值和/或最大值）。可将此元数据与查询相比较来判定片段是否包含请求的数据。

[0041] 数据库 170 可向每个记录写入时间戳以允许作出关于这种记录是否作为一致视图的一部分可用的判定。这些时间戳可被表示为整数值（例如，64 比特等等）。每个实时事务可由事务索引（例如，32 比特长度等等）表示，该事务索引可例如是事务控制块在事务控制块的阵列中的索引号（本文中称为事务控制块索引或简称为 TCB 索引）。在一些情况中，事务控制块的阵列可以是预分配的（而不是动态分配的）。TCB 阵列可以被静态存储（例如，静态大小的静态阵列）或者可以是动态分配的。可以使用基数树 (radix tree) 来动态地分配阵列的固定大小的组块并且在这个基数树中管理它们。此布置允许了经由索引的快速访问，而不要求阵列在存储器中是连续的。

[0042] 为了允许实时事务读取其自己的写入（即，事务写入到的记录等等），一致视图可不仅基于时间戳，而且基于事务的 TCB 索引。参考以下的表 1，每个时间戳可被编码为至少一个比特是指示出其是最终时间戳还是临时时间戳的标志。最终时间戳也可包括封装提交时间戳的部分。临时时间戳也可包括封装相应的 TCB 索引值的部分。

[0043]

最终时间戳	63	62..0	
	标志 =0	提交时间戳	
临时时间戳	63	62..32	31..0
	标志 =1	（保留）	TCB 索引

[0044] 表 1

[0045] 利用以上提供的时间戳编码，可对照一致视图的时间戳检查最终时间戳和临时时间戳两者。如果利用上述可见性公式时间戳不可见（例如，64 比特值高于一致视图的时间戳，等等），则其可能是事务的临时时间戳。时间戳中的 TCB 索引与运行中的事务的 TCB 索引的额外比较确定自身写入的可见性。可见性函数可如下：

[0046] $visible(CV,TS) = TS \leqslant CT.TS \vee (TS.Flag = 1 \wedge TS.TCBIndex = CV.TCBIndex)$

[0047] 参考图 8 的图 800，为了将临时时间戳转换成最终时间戳，事务的提交处理可如下。最初，在 810，可进入提交锁定。然后，在 820，提交时间戳 (CommitTS) 被确定为等于提交时间戳生成子 (CommitTSGenerator) 加 1。接下来，在 830，对于形成事务的一部分的每个被修改的时间戳，将其设置为提交时间戳 (CommitTS)。随后，在 840，可发出写入存储器屏障（例如，对于不遵循总存储顺序 (total store orde，TSO) 存储器模型的那些系统）。这种存储器屏障可使得数据库 170 的处理器或者编译器对在屏障指令之前和之后发出的存储器操作实施排序约束。此时，在 850，可以使提交时间戳生成子 (CommitTSGenerator) 等于提交时间戳 (CommitTS)。然后在 860 释放提交锁定。

[0048] 记录在一些情况中可形成事务日志的一部分，该事务日志可用于促进恢复。日志恢复系统的一个示例可在美国专利 8,768,891 号中找到，其内容在此被通过引用完全并入。另外，可向客户端 140 提供通知以识别对于各种事务哪些记录是可见的。

[0049] 利用当前布置，在提交处理期间开始的任何一致视图仍将读取提交时间戳生成子的旧值，因此进行提交的事务（即，发起了提交处理的事务）做出的任何改变在提交处理期间和该事务最终提交之后都将不是可见的（临时时间戳和最终提交时间戳两者对于此一致视图都将被评估为不可见）。在提交处理结束之后开始的任何一致视图将看到该事务完成的所有改变，因为重定时间戳（即，改变时间戳）到提交时间戳生成子被设置到新值时已完成。

[0050] 由于上述可要求将提交时间戳生成子锁定很长的一段时间（长运行中事务可能修改了大量时间戳），所以可进行进一步优化。例如，如下文提供的，可以并行提交若干个事务。

[0051] 参考图 9 的图 900，对于进行提交的线程，在 910，首先利用原子指令将事务排队到提交队列。接下来，在 920，等待提交线程以便允许时间戳处理。然后，在 930，提交时间戳 (CommitTS) 被设置为提交时间戳生成子值 (CommitTSGenerator) 加 1。从而，在 940，被事务设置 / 修改的时间戳被设置为 CommitTS。然后在 950 发出写入存储器屏障。然后在 960 通知提交线程。进行提交的线程随后在 970 等待提交线程确认该提交。

[0052] 此外，对于图 10 的图 1000，在 1010，提交线程等待被排队的任何元素。随后，在 1020，所有线程被从队列中撤出。然后，在 1030，所有被撤出队列的线程被唤醒以处理其时间戳。接下来，在 1040，提交线程等待来自所有撤出队列的线程的通知。提交时间戳生成子 (CommitTSGenerator) 随后在 1050 前进一个值。所有撤出队列的线程在 1060 随后被唤醒以继续处理。该过程随后可在 1070 被重复。

[0053] 图 11 是过程流程图 1100，其中，在 1110，处理数据库中的多个事务。每个事务包括在数据库中的至少一个记录上的多个操作，其中这些事务中的至少两者被并发地处理。然后，在 1120，向每个记录指派临时时间戳，该临时时间戳至少部分基于相应事务。另外，在 1130，向具有提交操作的每个记录指派最终时间戳。随后在 1140 利用可见性函数并且基于指派的临时时间戳和最终时间戳判定哪些记录在用于第一事务的一致视图中是可见的。然后，在 1150，第一事务可访问被判定为可见的记录。

[0054] 本文描述的主题的一个或多个方面或特征可在数字电子电路、集成电路、特殊设计的专用集成电路 (application specific integrated circuit，ASIC)、现场可编程门阵列 (field programmable gate array，FPGA)、计算机硬件、固件、软件和 / 或其组合中实现。这些各种方面或特征可包括在可编程系统上可执行和 / 或可解释的一个或多个计算机程序中的实现方式，该系统包括至少一个可编程处理器，该处理器可以是专用或通用的，其被耦合来从存储系统、至少一个输入设备和至少一个输出设备接收数据和指令并且向它们发送数据和指令。可编程系统或计算系统可包括客户端和服务器。客户端和服务器一般可彼此远离并且通常通过通信网络来交互。客户端和服务器的关系是由于计算机程序在相应计算机上运行且相互之间具有客户端 - 服务器关系而发生的。

[0055] 这些计算机程序——也可称为程序、软件、软件应用、应用、组件或代码——包括用于可编程处理器的机器指令，并且可以用高级别过程语言、面向对象的编程语言、功能性编程语言、逻辑编程语言和 / 或用汇编 / 机器语言来实现。当在本文中使用时，术语“机器可读介质”指的是用于向可编程处理器提供机器指令和 / 或数据的任何计算机程序产品、装置和 / 或设备，例如磁盘、光盘、存储器和可编程逻辑器件 (Programmable Logic Device，

PLD)，包括以机器可读信号的形式接收机器指令的机器可读介质。术语“机器可读信号”指的是用于向可编程处理器提供机器指令和/或数据的任何信号。机器可读介质可非暂时地存储这种机器指令，例如像非暂态固态存储器或磁性硬盘驱动器或者任何等同的存储介质那样。机器可读介质可以替换地或额外地以暂态方式存储这种机器指令，例如像与一个或多个物理处理器核心相关联的处理器缓存或其他随机访问存储器那样。

[0056] 为了支持与用户的交互，本文描述的主题的一个或多个方面或特征可在具有诸如阴极射线管(cathode ray tube，CRT)或液晶显示器(liquid crystal display，LCD)或发光二极管(light emitting diode，LED)监视器之类的用于向用户显示信息的显示设备以及用户可用来向计算机提供输入的键盘和诸如鼠标或轨迹球之类的指点设备的计算机上实现。其他种类的设备也可用于支持与用户的交互。例如，向用户提供的反馈可以是任何形式的感觉反馈，例如视觉反馈、听觉反馈或触觉反馈；并且可以以任何形式接收来自用户的输入，包括但不限于音响、话音或触觉输入。其他可能的输入设备包括——但不限于——触摸屏或其他触摸敏感设备，例如单点或多点电阻或电容触控板、语音识别硬件和软件、光学扫描仪、光学指示器、数字图像捕捉设备和关联的解释软件，等等。

[0057] 在以上描述中和权利要求中，诸如“……中的至少一者”或者“……中的一个或多个”之类的短语可跟在元素或特征的连接列表之后出现。术语“和/或”也可出现在两个或更多个元素或特征的列表中。除非与其所用于的上下文隐含地或明确地矛盾，这种短语打算意指列出的元素或特征中的任何单独的一个或者任何记载的元素或特征与任何其他记载的元素或特征的组合。例如，短语“A和B中的至少一者”、“A和B中的一个或多个”和“A和/或B”各自打算意指“A单独、B单独或者A和B一起”。类似的解释也打算用于包括三个或更多个项目的列表。例如，短语“A、B和C中的至少一者”、“A、B和C中的一个或多个”和“A、B和/或C”各自打算意指“A单独、B单独、C单独、A和B一起、A和C一起、B和C一起或者A和B和C一起”。此外，上文和权利要求中对术语“基于”的使用打算意指“至少部分基于”，使得未记载的特征或元素也是可允许的。

[0058] 本文描述的主题取决于期望的配置可在系统、装置、方法和/或物品中实现。前文描述中记载的实现方式不表示符合本文描述的主题的所有实现方式。反而，它们只是符合与描述的主题有关的方面的一些示例。虽然上文详细描述了几个变化，但其他修改或添加是可能的。具体地，除了本文记载的那些以外，也可提供另外的特征和/或变化。例如，上文描述的实现方式可针对公开的特征的各种组合和子组合和/或上文公开的若干进一步特征的组合和子组合。此外，附图中描绘的和/或本文描述的逻辑流程要实现期望的结果并非必然要求所示出的特定顺序或者先后顺序。其他实现方式可在所附权利要求的范围内。

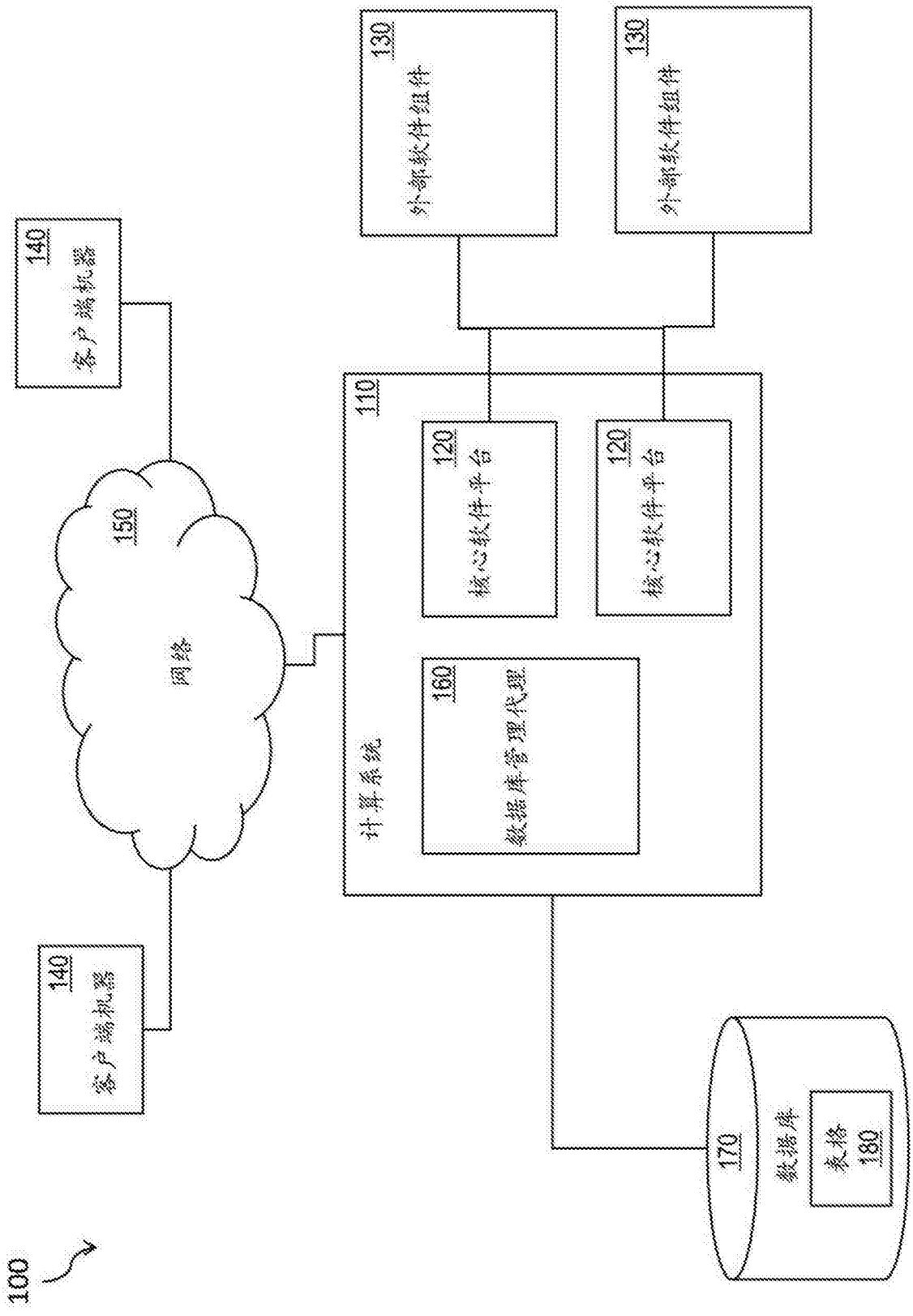

130
外部软件组件
130
外部软件组件
140
客户端机器
110
120
核心软件平台
120
核心软件平台
150
网络
160
数据库管理代理
计算系统
140
客户端机器
170
数据库
表格
180
100

图 1

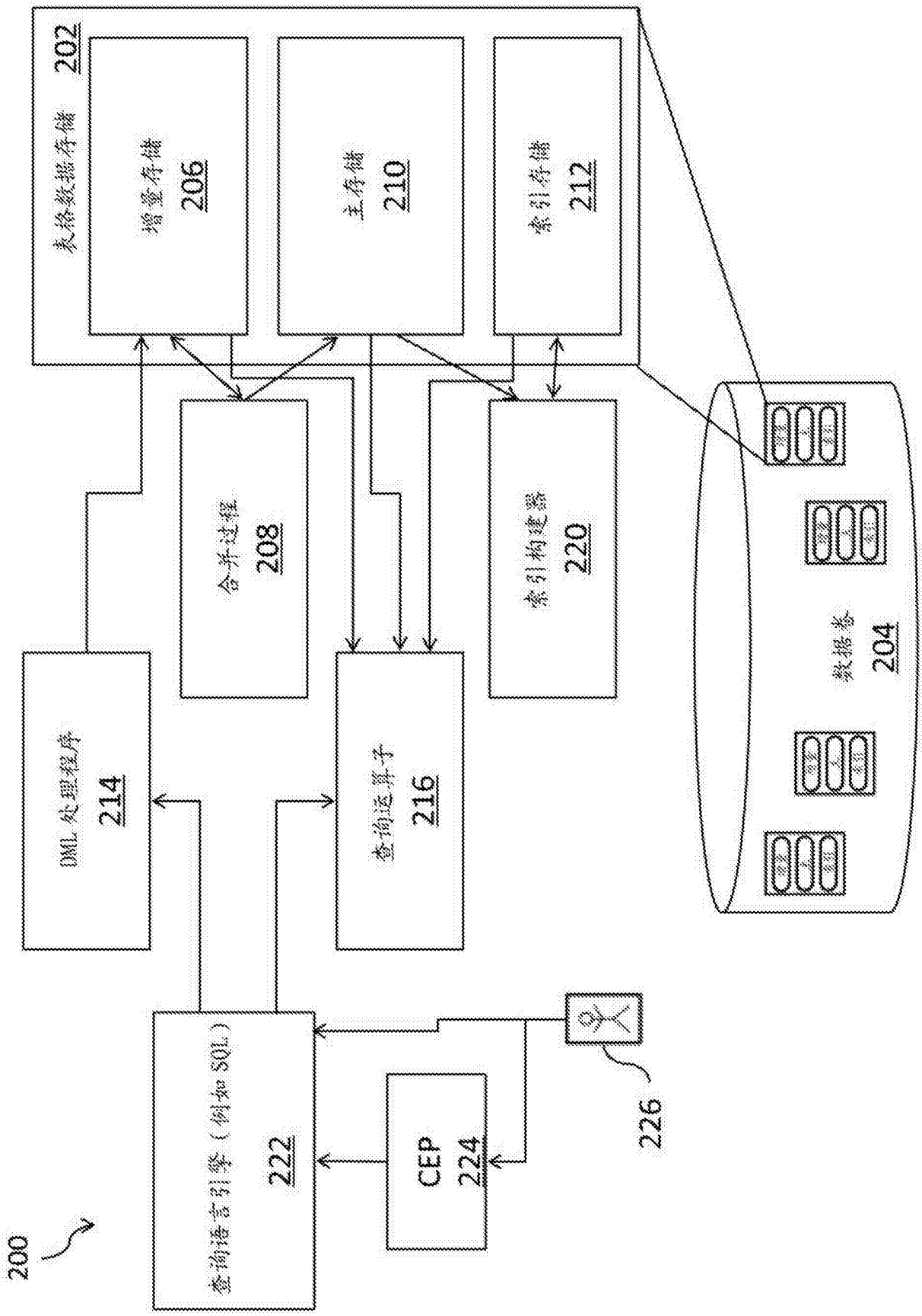

表格数据存储 202
增量存储 206
主存储 210
索引存储 212
合并过程 208
索引构建器 220
数据卷 204
DML 处理程序 214
查询运算子 216
查询语言引擎（例如 SQL） 222
CEP 224
226
200

图 2

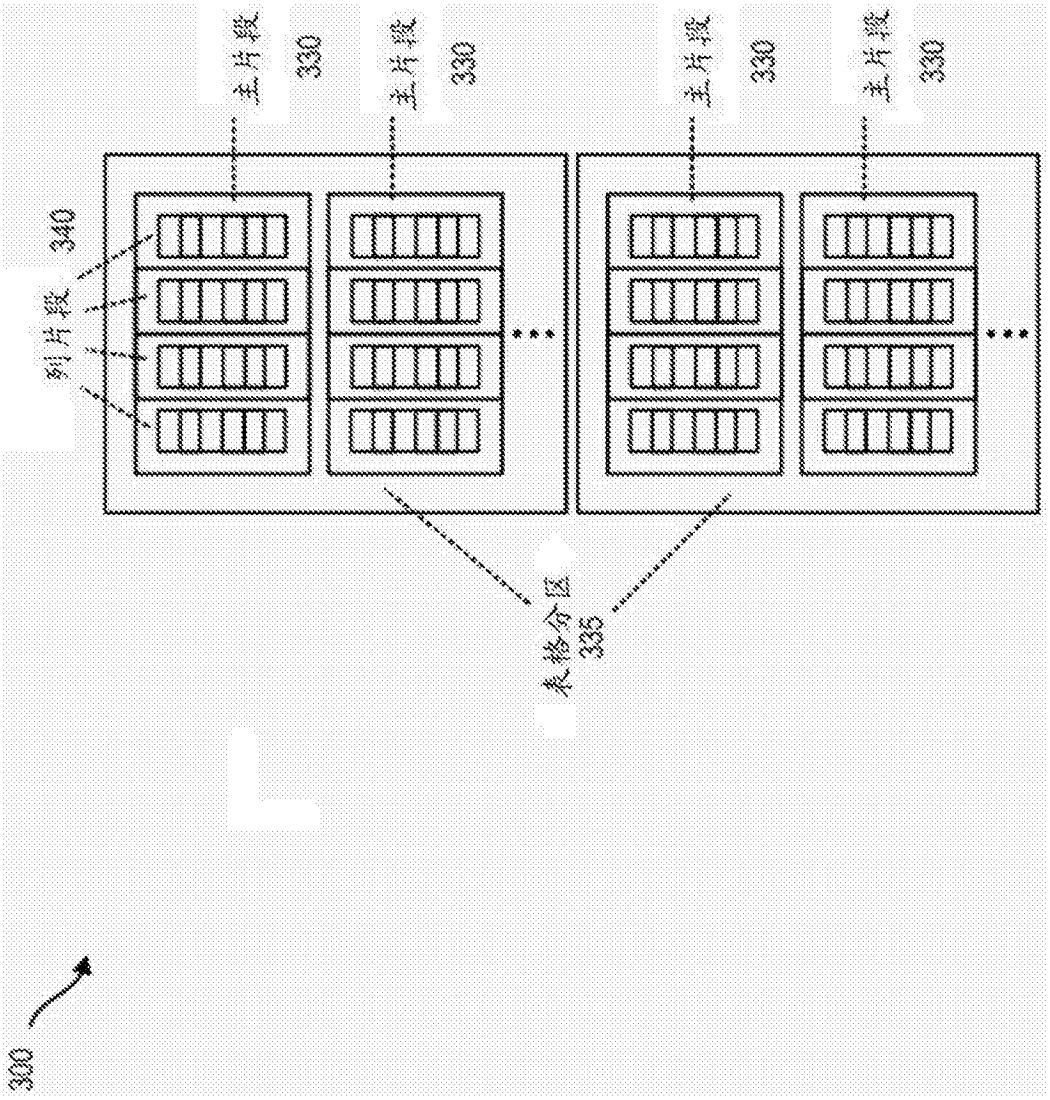


图 3

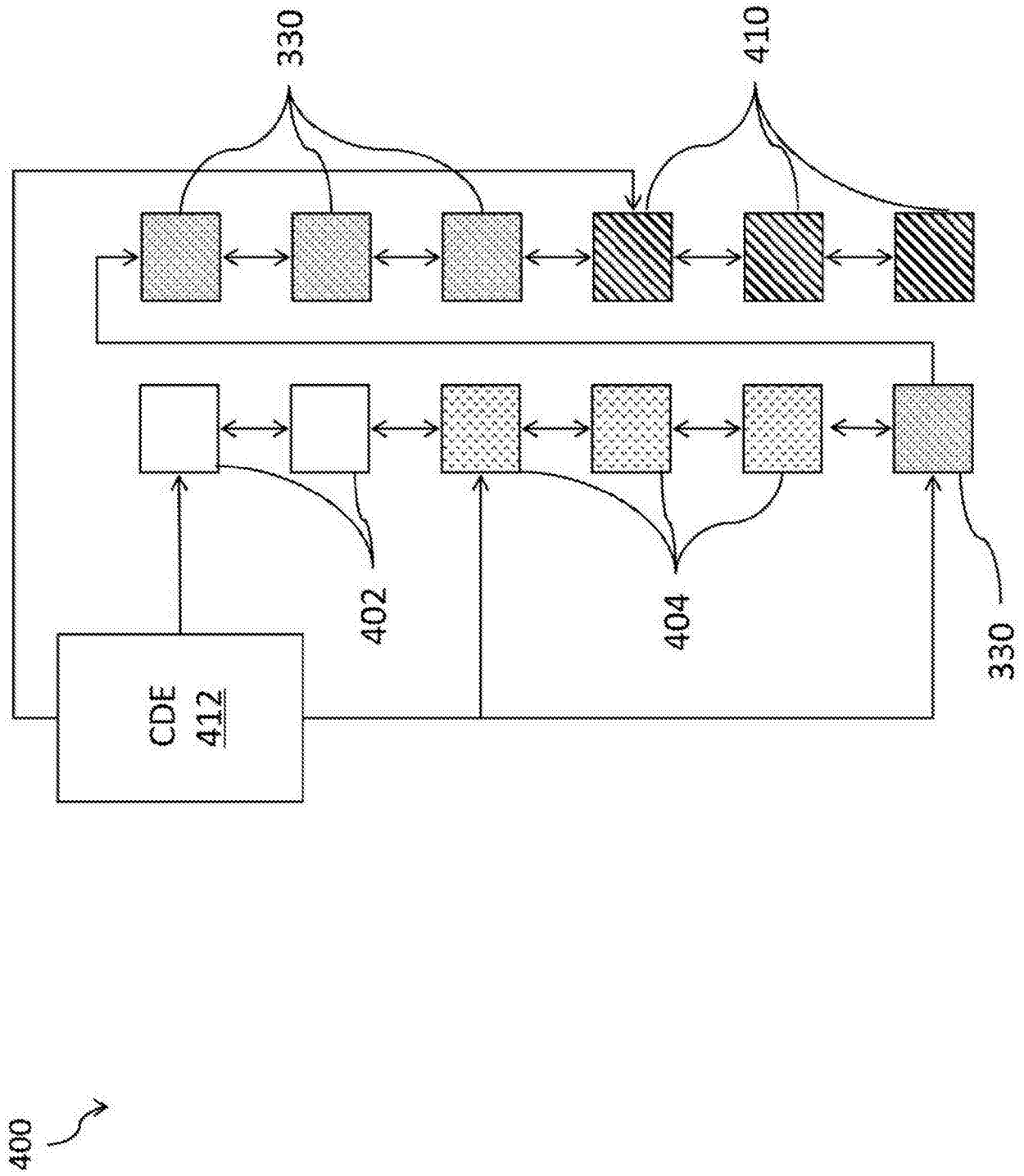


图 4

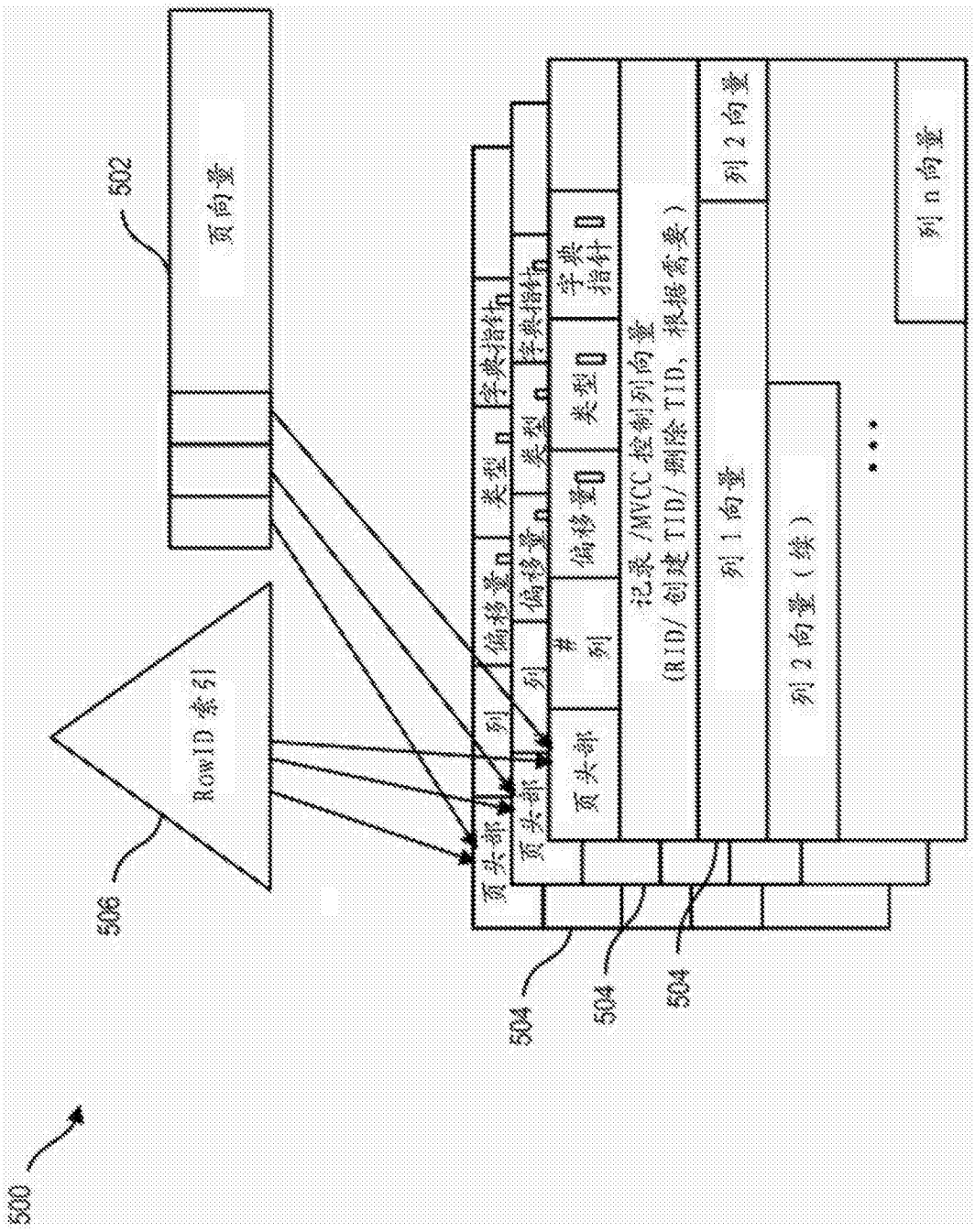

500
502
页向量
506
RowID 索引
504
504
504
页头部
列
偏移量0
偏移量n
类型0
类型n
字典指针0
字典指针n
#列
记录/MVCC 控制列向量
(RID/ 创建 TID/ 删除 TID, 根据需要)
列1向量
列2向量
列2向量(续)
列n向量

图 5

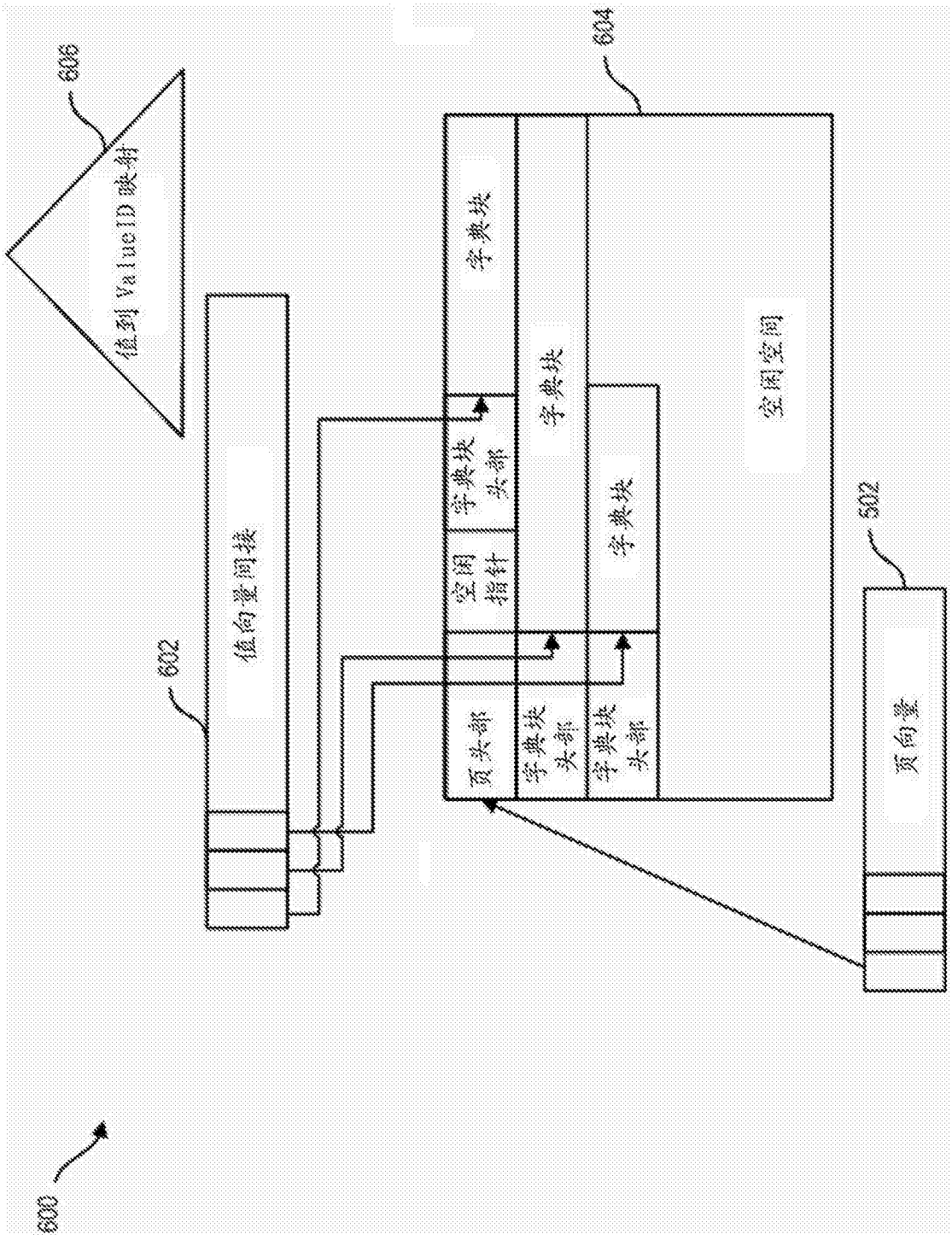

600
606
值到 ValueID 映射
602
值向量间接
604
字典块
字典块
字典块
字典块
字典块
字典块
空闲空间
空闲指针
页头部
602
页向量

图 6

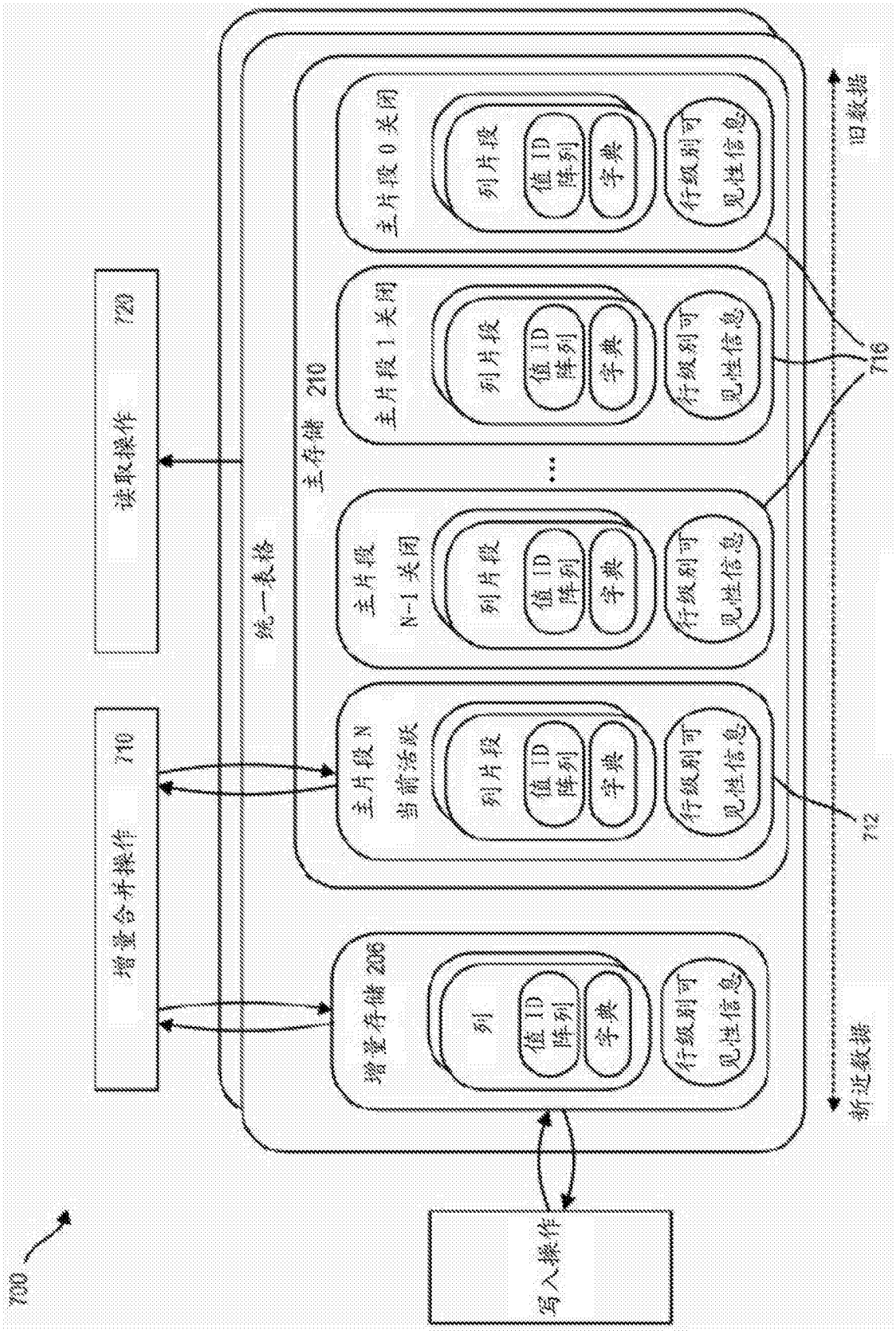

730
写入操作
增量合并操作 710
读取操作 720
统一表格
主存储 210
主片段 0 关闭
主片段 1 关闭
主片段 N-1 关闭
主片段 N 当前活跃
列片段
值 ID 阵列
字典
行级别可见性信息
增量存储 206
列
716
712
旧数据
新近数据

图 7

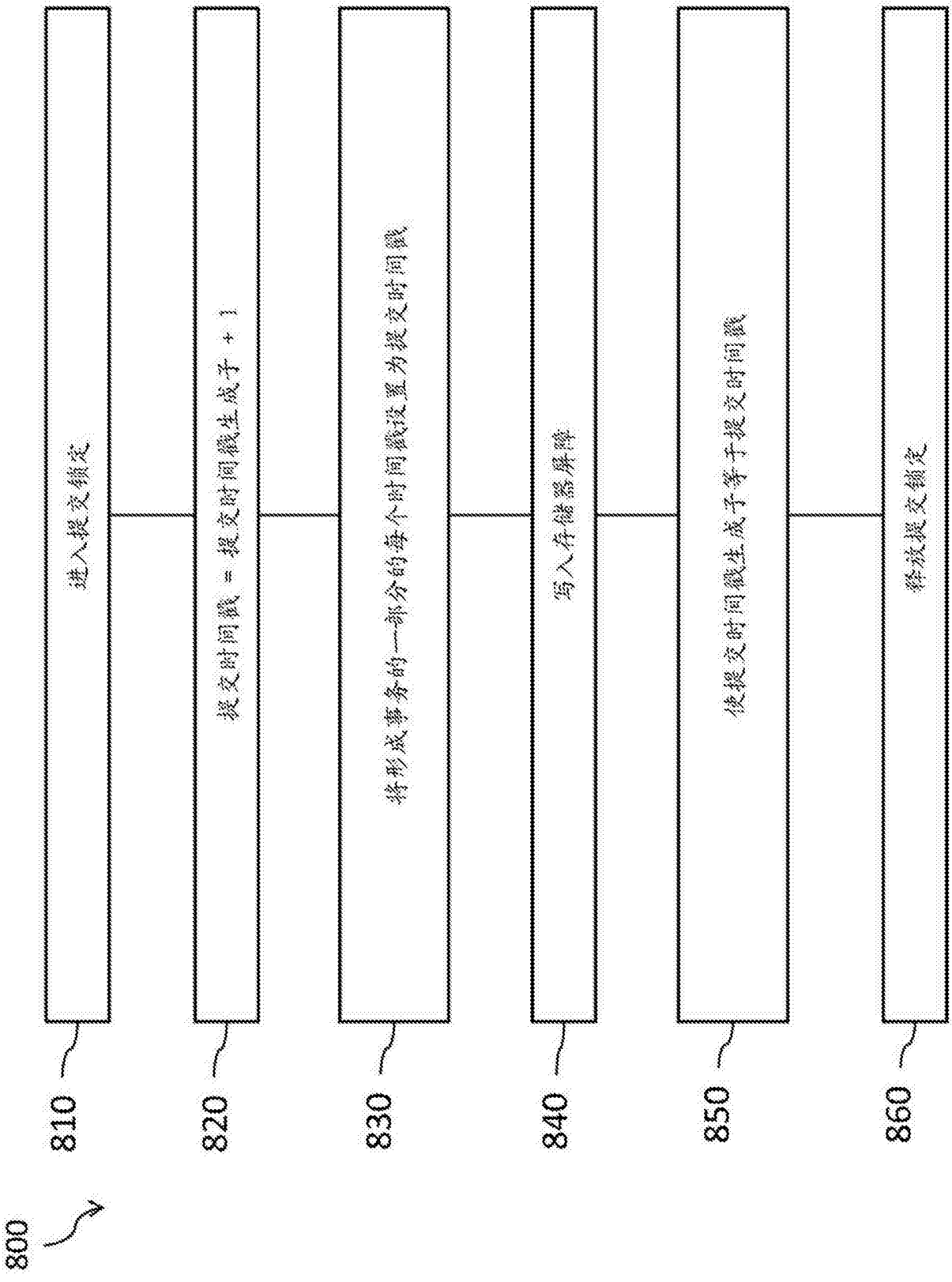

800
810
进入提交锁定
820
提交时间戳 = 提交时间戳生成子 + 1
830
将形成事务的一部分的每个时间戳设置为提交时间戳
840
写入存储器屏障
850
使提交时间戳生成子等于提交时间戳
860
释放提交锁定

图 8

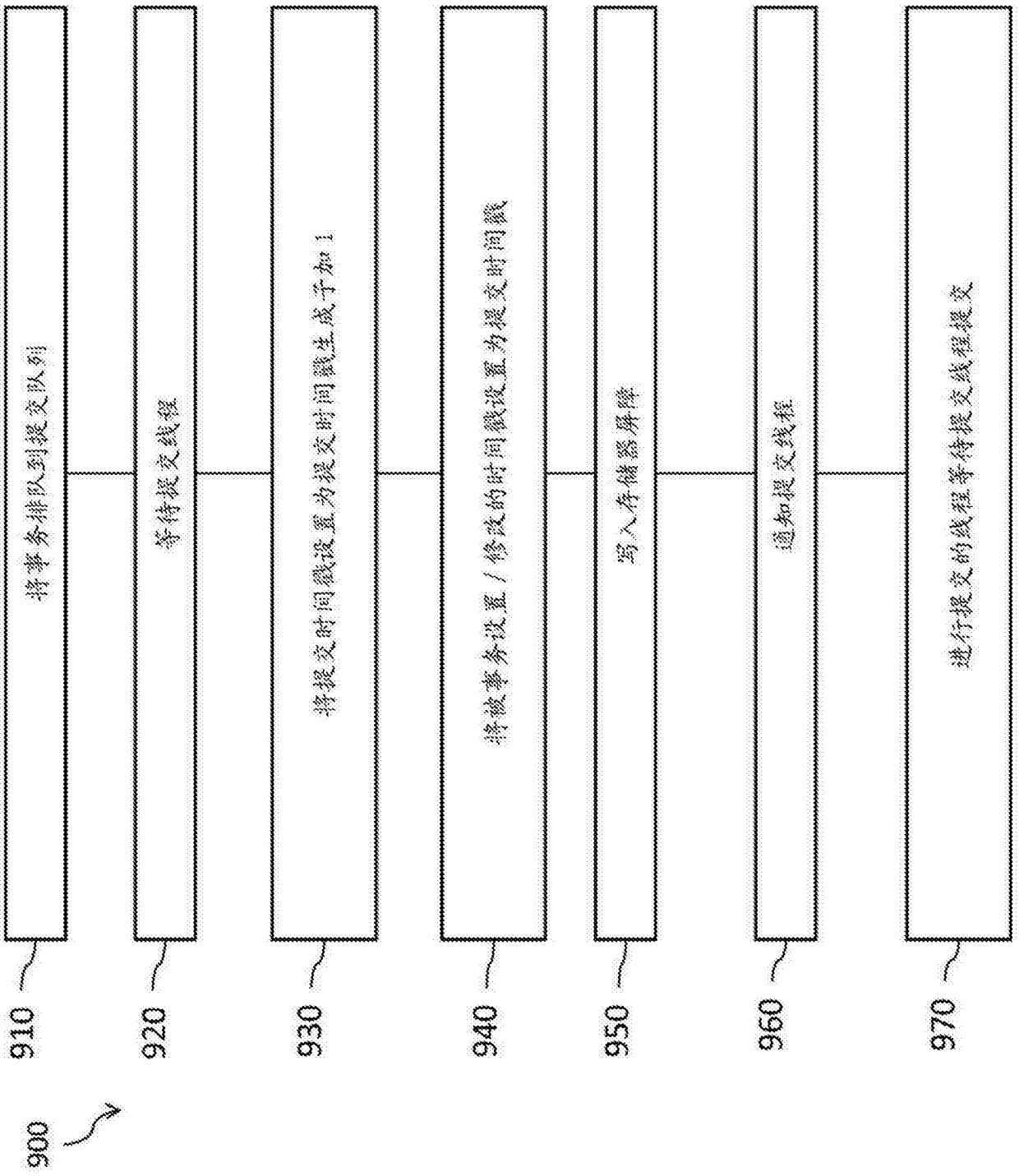

900
910
将事务排队到提交队列
920
等待提交线程
930
将提交时间戳设置为提交时间戳生成子加 1
940
将被事务设置 / 修改的时间戳设置为提交时间戳
950
写入存储器屏障
960
通知提交线程
970
进行提交的线程等待提交线程提交

图 9

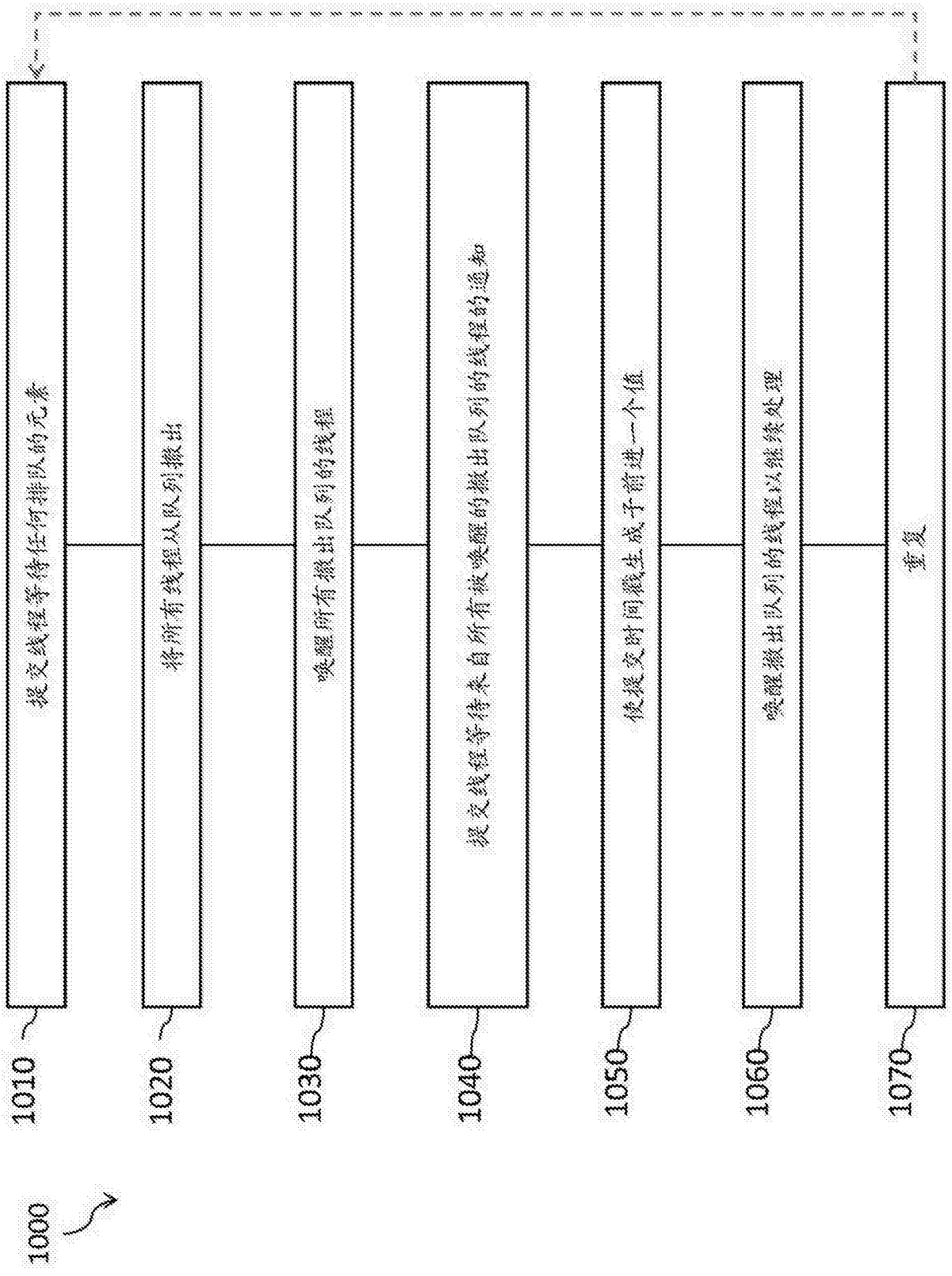

提交线程等待任何排队的元素
将所有线程从队列撤出
唤醒所有撤出队列的线程
提交线程等待来自所有被唤醒的撤出队列的线程的通知
使提交时间戳生成子前进一个值
唤醒撤出队列的线程以继续处理
重复
1010
1020
1030
1040
1050
1060
1070
1000

图 10

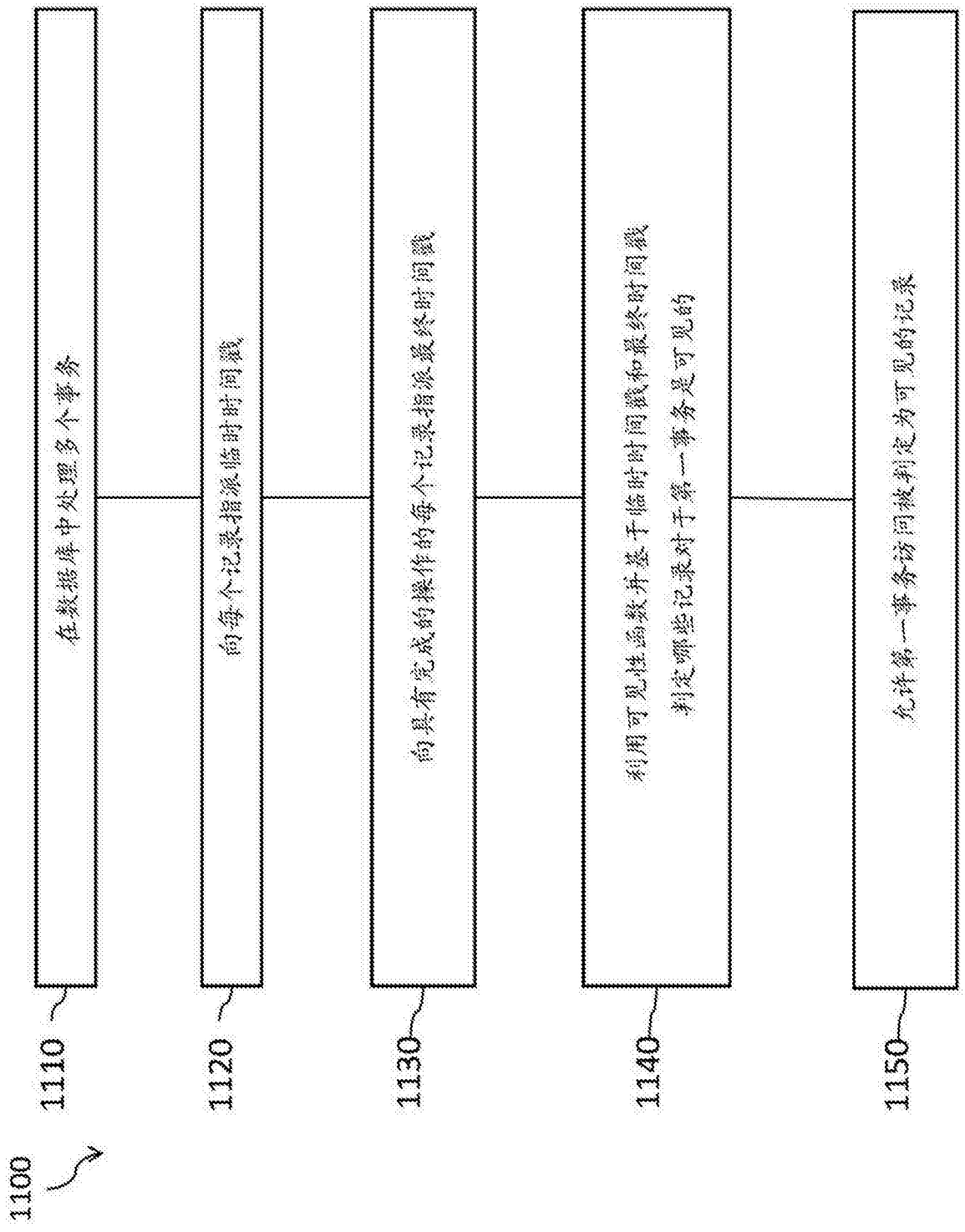

1100
1110
在数据库中处理多个事务
1120
向每个记录指派临时时间戳
1130
向具有完成的操作的每个记录指派最终时间戳
1140
利用可见性函数并基于临时时间戳和最终时间戳
判定哪些记录对于第一事务是可见的
1150
允许第一事务访问被判定为可见的记录

图 11