(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 108475216 B
(45)授权公告日 2022.07.22

(21)申请号 201780005289.8
(22)申请日 2017.08.03
(65)同一申请的已公布的文献号
申请公布号 CN 108475216 A
(43)申请公布日 2018.08.31
(30)优先权数据
15/227,899 2016.08.03 US
(85)PCT国际申请进入国家阶段日
2018.06.28
(86)PCT国际申请的申请数据
PCT/US2017/045288 2017.08.03
(87)PCT国际申请的公布数据
WO2018/027028 EN 2018.02.08
(73)专利权人 甲骨文国际公司
地址 美国加利福尼亚
(72)发明人 J•德拉瓦瑞恩 Y•多高夫
V•赫格德 S•弗玛
C•S•K•马赫德哈拉
A•纳玛奇瓦亚姆
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 边海梅
(51)Int.Cl.
G06F 9/50(2006.01)
(56)对比文件
US 2014324911 A1,2014.10.30
US 2014324911 A1,2014.10.30
US 2008250419 A1,2008.10.09
US 2015207758 A1,2015.07.23
CN 105808638 A,2016.07.27
CN 103646111 A,2014.03.19
CN 101042660 A,2007.09.26
CN 101183377 A,2008.05.21
US 8015524 B1,2011.09.06
CN 103503416 A,2014.01.08
US 7032003 B1,2006.04.18
US 7207041 B2,2007.04.17
审查员 朱晓岗

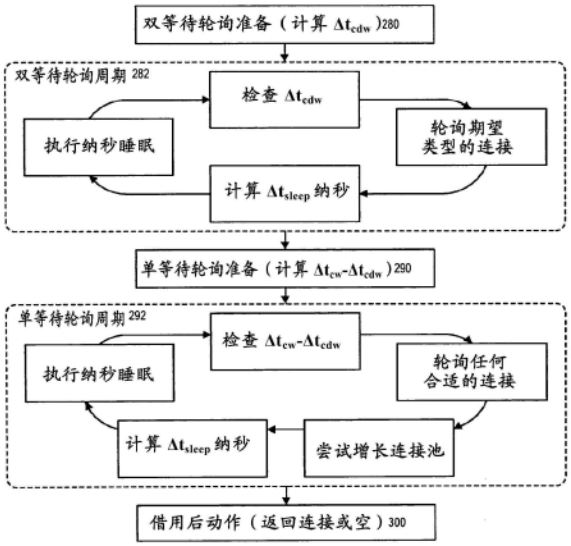
权利要求书3页 说明书10页 附图10页

(54)发明名称

用于在多租户数据库环境中的连接的高效再利用的系统和方法

(57)摘要

本文描述的是用于在多租户环境中提供对数据库的访问的系统和方法，包括连接池的使用，以及对连接的高效再利用的支持。根据实施例，软件应用可以请求提供连接以使得能够访问数据库。响应于接收到请求，连接池可以首先确定具有确切期望特性的特定连接是否已经存在于池内，但是在请求时被借用。如果存在这样的连接，则连接池可以等待该特定连接变得可用的时间段，在本文中称为双等待。随后，如果未使特定连接在双等待时间段内可用，则连接池例如通过再利用其它连接恢复其正常操作。



CN 108475216 B

1.一种用于提供对数据库的访问的系统，包括对连接的高效再利用的支持，所述系统包括：

计算机，包括处理器以及在其上执行的应用服务器或数据库环境中的至少一个；

连接池，使得软件应用能够从所述连接池中请求连接，并使用所提供的连接来访问数据库；

其中所述系统确定由可配置的连接池属性定义的连接等待时间段的值，并且动态地计算双等待时间段的值，该双等待时间段短于所述连接等待时间段，以及

其中所述系统使得能够再利用所述连接池中的连接，包括响应于接收到对具有期望特性的连接的请求，

在与所述请求相关联的所述双等待时间段期间，确定所述连接池是否包括在请求时被借用的具有所述期望特性的特定连接；以及

如果在所述连接池内存在这样的特定连接但是该特定连接在请求时被借用，则在短于所述连接等待时间段的所述双等待时间段期间等待，同时轮询具有期望特性的所述特定连接变得可用；

如果具有期望特性的所述特定连接在与所述请求相关联的所述双等待时间段期间未变得可用，则在与所述请求相关联的所述双等待时间段完成后，在再利用另一个连接以具有所述期望特性之前继续在剩余连接等待时段期间等待可用连接。

2.根据权利要求1所述的系统，其中如果未使所述特定连接在所述双等待时间段内可用，则所述连接池恢复正常操作，包括确定是否再利用其它连接。

3.根据权利要求1或2所述的系统，其中所述连接池能够使用包括单等待模式和双等待模式的不同模式进行轮询。

4.根据权利要求1或2所述的系统，其中所述连接池支持动态地确定与平均连接借用时间间隔成比例的所述双等待时间段的值的自动调谐处理。

5.根据权利要求1或2所述的系统，其中使得软件应用能够将特定标记与特定连接状态相关联。

6.根据权利要求1或2所述的系统，其中每个软件应用与一个或多个租户相关联。

7.一种用于提供对数据库的访问的方法，包括对连接的高效再利用的支持，所述方法包括：

在包括处理器和在其上执行的应用服务器或数据库环境中的至少一个的计算机处，提供包括连接对象的连接池，并且所述连接池使得软件应用能够从连接池中请求连接，并且使用所提供的连接来访问数据库；

确定由可配置的连接池属性定义的连接等待时间段的值，并且动态地计算双等待时间段的值，该双等待时间段短于所述连接等待时间段；以及

使得能够再利用所述连接池中的连接，包括响应于接收到对具有期望特性的连接的请求，

在与所述请求相关联的所述双等待时间段期间，确定所述连接池是否包括在请求时被借用的具有所述期望特性的特定连接；以及

如果在所述连接池内存在这样的特定连接但是该特定连接在请求时被借用，则在短于所述连接等待时间段的所述双等待时间段期间等待，同时轮询具有期望特性的所述特定连

接变得可用；

如果具有期望特性的所述特定连接在与所述请求相关联的所述双等待时间段期间未变得可用，则在与所述请求相关联的所述双等待时间段完成后，在再利用另一个连接以具有所述期望特性之前继续在剩余连接等待时段期间等待可用连接。

8.根据权利要求7所述的方法，其中如果未使所述特定连接在所述双等待时间段内可用，则所述连接池恢复正常操作，包括确定是否再利用其它连接。

9.根据权利要求7或8所述的方法，其中所述连接池能够使用包括单等待模式和双等待模式的不同模式进行轮询。

10.根据权利要求7或8所述的方法，其中所述连接池支持动态地确定与平均连接借用时间间隔成比例的所述双等待时间段的值的自动调谐处理。

11.根据权利要求7或8所述的方法，其中使得软件应用能够将特定标记与特定连接状态相关联。

12.根据权利要求7或8所述的方法，其中每个软件应用与一个或多个租户相关联。

13.一种非暂态计算机可读存储介质，包括存储在其上的指令，所述指令在由一个或多个计算机读取和执行时，使得所述一个或多个计算机执行包括以下的方法：

在包括处理器和在其上执行的应用服务器或数据库环境中的至少一个的计算机处，提供包括连接对象的连接池，并且使得软件应用能够从连接池中请求连接，并且使用所提供的连接来访问数据库；

确定由可配置的连接池属性定义的连接等待时间段的值，并且动态地计算双等待时间段的值，该双等待时间段短于所述连接等待时间段；以及

使得能够再利用所述连接池中的连接，包括响应于接收到对具有期望特性的连接的请求，

在与所述请求相关联的所述双等待时间段期间，确定所述连接池是否包括在请求时被借用的具有所述期望特性的特定连接；以及

如果在所述连接池内存在这样的特定连接但是该特定连接在请求时被借用，则在短于所述连接等待时间段的所述双等待时间段期间等待，同时轮询具有期望特性的所述特定连接变得可用；

如果具有期望特性的所述特定连接在与所述请求相关联的所述双等待时间段期间未变得可用，则在与所述请求相关联的所述双等待时间段完成后，在再利用另一个连接以具有所述期望特性之前继续在剩余连接等待时段期间等待可用连接。

14.根据权利要求13所述的非暂态计算机可读存储介质，其中如果未使所述特定连接在所述双等待时间段内可用，则连接池恢复正常操作，包括确定是否再利用其它连接。

15.根据权利要求13或14所述的非暂态计算机可读存储介质，其中所述连接池能够使用包括单等待模式和双等待模式的不同模式进行轮询。

16.根据权利要求13或14所述的非暂态计算机可读存储介质，其中所述连接池支持动态地确定与平均连接借用时间间隔成比例的所述双等待时间段的值的自动调谐处理。

17.根据权利要求13或14所述的非暂态计算机可读存储介质，其中使得软件应用能够将特定标记与特定连接状态相关联。

18.根据权利要求13或14所述的非暂态计算机可读存储介质，其中每个软件应用与一

个或多个租户相关联。

19.一种包括用于执行如权利要求7至12中任一项所述的方法的部件的装置。

用于在多租户数据库环境中的连接的高效再利用的系统和方法

[0001] 版权声明

[0002] 本专利文献的公开内容的一部分包含受版权保护的素材。版权拥有者不反对任何人对专利文献或专利公开内容按照其在专利商标局的专利文件或记录中出现的那样进行传真复制，但是除此之外在任何情况下都保留所有版权。

[0003] 要求优先权

[0004] 本申请要求于2016年8月3日提交的申请号为15/227,899的标题为“SYSTEM AND METHOD FOR EFFICIENT REPURPOSING OF CONNECTIONS IN A MULTI-TENTANT DATABASE ENVIRONMENT”的美国专利申请的优先权的权益，该申请通过引用并入本文。

技术领域

[0005] 本发明的实施例一般而言涉及软件应用服务器和数据库，并且特别地涉及用于提供对多租户环境中的数据库的访问的系统和方法，包括使用连接池以及支持连接的高效再利用。

背景技术

[0006] 在数据库环境中，一般描述的连接池作为连接对象的高速缓存进行操作，连接对象中的每个连接对象表示可以由软件应用用来连接到数据库的连接。在运行时，应用可以从连接池中请求连接。如果连接池包括可以满足特定请求的连接，则该连接池可以将该连接返回给应用以供其使用。在一些情况下，如果未找到合适的连接，则可以创建新连接并将该新连接返回给应用。应用可以借用连接来访问数据库并执行一些工作，然后将连接返回给池，其中可以使该连接然后可用于来自相同应用或来自其它应用的后续连接请求。

发明内容

[0007] 本文描述的是用于在多租户环境中提供对数据库的访问的系统和方法，包括使用连接池以及支持连接的高效再利用。根据实施例，软件应用可以请求提供连接以使得能够访问数据库。响应于接收到请求，连接池可以首先确定具有确切期望特性的特定连接是否已经存在于池内，但是在请求时被借用。如果存在这样的连接，则连接池可以等待该特定连接变得可用的时间段，在本文中称为双等待。随后，如果未使该特定连接在双等待时间段内可用，则连接池例如通过再利用其它连接恢复其正常操作。

附图说明

[0008] 图1图示了根据实施例的包括连接池的系统。

[0009] 图2进一步图示了根据实施例的包括连接池的系统，包括支持使用分片数据库。

[0010] 图3进一步图示了根据实施例的包括连接池的系统，包括支持在多租户环境中使用。

[0011] 图4图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0012] 图5进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0013] 图6进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0014] 图7进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0015] 图8进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0016] 图9进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0017] 图10图示了根据实施例的在连接池环境中提供支持连接的高效再利用的方法。

具体实施方式

[0018] 如上所述，连接池作为连接对象的高速缓存进行操作，连接对象中的每个连接对象表示可以由软件应用用来连接到数据库的连接。在运行时，应用可以从连接池中请求连接。如果连接池包括可以满足特定请求的连接，则该连接池可以将该连接返回给应用以供其使用。在一些情况下，如果未找到合适的连接，则可以创建新连接并将该新连接返回给应用。应用可以借用该连接来访问数据库并执行一些工作，然后将该连接返回给池，在池中可以使该连接随后可用于来自相同应用或来自其它应用的后续连接请求。

[0019] 创建连接对象在时间和资源方面可能是昂贵的。例如，诸如网络通信、认证、事务征用和存储器分配的任务都会影响创建特定连接对象所花费的时间和资源量。由于连接池允许重复使用这些连接对象，因此它们有助于减少必须创建各种对象的次数。

[0020] 连接池的一个示例是Oracle通用连接池(UCP)，该UCP提供用于对Java数据库连接(JDBC)连接进行高度缓存的连接池。例如，连接池可以与JDBC驱动程序一起操作以创建到数据库的连接，所述连接随后由池进行维护；并且可以基于发出请求的软件应用的性能和可用性要求被配置具有用于进一步优化池行为的属性。

[0021] 连接标记(connection labeling)

[0022] 图1图示了根据实施例的包括连接池的系统。

[0023] 如图1所示，根据实施例，包括物理计算机资源101(例如，处理器/CPU、存储器和网络组件)的应用服务器或数据库环境100(例如Oracle WebLogic服务器、Oracle融合中间件(Oracle Fusion Middleware)或其它应用服务器或数据库环境)可以包括或提供对数据库102(例如Oracle数据库或其它类型的数据库)的访问。

[0024] 如图1进一步所示，根据实施例，该系统还包括连接池逻辑104或程序代码，连接池逻辑104或程序代码在由计算机执行时，控制105连接池106中的连接对象的创建和使用，连接对象包括例如当前由软件应用使用的连接108，以及空闲连接110或当前未在被使用的连接。

[0025] 软件应用可以在使用从连接池检索到的连接来访问或执行数据库处的工作之前初始化该连接。初始化的示例可以包括需要应用代码内的方法调用的简单的状态重新初始化，或者包括需要在网络上往返的数据库操作的更复杂的初始化。后面这些类型的初始化的计算成本可能很显著。

[0026] 一些连接池(例如，UCP)允许使用连接池属性来配置其连接池，这些连接池属性具有get和set方法，并且通过启用池的数据源实例而成为可用的。这些get和set方法提供了以编程方式配置池的简便方法。如果未设置池属性，则连接池使用默认的属性值。

[0027] 根据实施例，标记连接允许客户端软件应用将任意名称/值对附接到连接。然后，应用可以从连接池中请求具有期望标记的连接。通过将特定标记与特定连接状态相关联，应用可以潜在地从池中检索已经初始化的连接，并且避免了重新初始化的时间和成本。连接标记不会对用户定义的键或值强加任何含义；任何用户定义的键和值的含义仅仅由应用定义。

[0028] 例如，如图1所示，根据实施例，连接池可以包括当前由软件应用使用的多个连接，这里标示为连接A 112和B 114。每个连接可以被标记，例如，连接A被标记(为蓝色)并且连接B被标记(为绿色)。这些标记/颜色是为了图示的目的而提供，并且如上所述可以是由客户端应用附接到连接的任意名称/值对。根据各种实施例，可以使用不同类型的标记来区分不同的连接类型；并且不同的应用可以将不同的标记/颜色附接到特定的连接类型。

[0029] 如图1进一步所示，根据实施例，连接池还可以包括空闲的或者当前未在被软件应用使用的多个连接，这里标示为连接C 116、D 118、E 120、F 122、G 124和N 126。每个空闲连接可以被类似地标记，在该图示中被标记为(蓝色)或(绿色)，并且再次地，这些标记/颜色是为了图示的目的而提供的。

[0030] 如图1进一步所示，根据实施例，如果软件应用130希望使用特定类型的连接(例如(红色)连接)对数据库做出请求，则应用可以做出“getConnection(Red)”请求132。作为响应，连接池逻辑将创建新的(红色)连接，这里标示为X 134(红色)；或者将现有的空闲连接从(蓝色或绿色)再利用到(红色)，这里标示为E 135(红色)。

[0031] 分片数据库(sharded database)

[0032] 根据实施例，分片是使用跨多个独立物理数据库的数据的水平划分的数据库缩放技术。存储在每个物理数据库中的数据的部分被称为碎片(shard)。从软件客户端应用的角度来看，所有物理数据库的集合都表现为单个逻辑数据库。

[0033] 根据实施例，系统可以包括支持使用具有分片数据库的连接池。碎片引导器或侦听器提供软件客户端应用对数据库碎片的访问。连接池(例如，UCP)和数据库驱动程序(例如，JDBC驱动程序)可以被配置为允许客户端应用在连接检验(checkout)期间或者在以后的时间提供碎片键；识别由客户端应用指定的碎片键；以及启用由客户端应用到特定碎片或块的连接。该方法使得能够高效地重用连接资源，并且能够更快地访问适当的碎片。

[0034] 图2进一步图示了根据实施例的包括连接池的系统，包括支持使用分片数据库。

[0035] 根据实施例，可以使用碎片键(SHARD_KEY)将数据库表划分为，例如确定在特定碎片内每一行被存储在哪里的一个或多个列。碎片键可以在连接串或描述中被提供为连接数据(CONNECT_DATA)的特性。碎片键的示例可以包括数据库中的VARCHAR2、CHAR、DATE(日期)、NUMBER(数值)或TIMESTAMP(时间戳)。根据实施例，分片数据库还可以接受没有碎片键或碎片组键的连接。

[0036] 根据实施例，为了减少重新分片对系统性能和数据可用性的影响，每个碎片可以被细分成更小的片或块。每个块充当可以从一个碎片移动到另一个碎片的重新分片单元。通过向碎片键映射添加间接层，块也简化了路由。

[0037] 例如，每个块可以自动地与碎片键值的范围相关联。用户提供的碎片键可以映射到特定的块，并且该块可以映射到特定的碎片。如果数据库操作尝试在特定碎片上不存在的块上操作，则将会出现错误。当使用碎片组时，每个碎片组是具有碎片组标识符的特定值

的那些块的集合。

[0038] 碎片感知的客户端应用可以与分片数据库配置一起工作，包括连接到其中数据基于一个或多个分片方法被划分的一个或多个数据库碎片的能力。每次需要数据库操作时，客户端应用可以确定该客户端应用需要连接到的碎片。

[0039] 根据实施例，分片方法可以用于将碎片键值映射到各个碎片。可以支持不同的分片方法，例如：基于散列的分片，其中向每个块分配一定范围的散列值，使得在建立数据库连接时，系统将散列函数应用到分片键的给定值，并计算对应的散列值，该散列值随后基于该值所属的范围被映射到块；基于范围的分片，其中一定范围的碎片键值被直接分配给各个碎片；以及基于列表的分片，其中每个碎片与碎片键值列表相关联。

[0040] 如图2所示，根据实施例，分片数据库140可以包括第一数据库区域A(这里标示为“DB东”，DBE)141，DBE 141包括具有被存储为块A1、A2、...、An的碎片A的分片数据库实例“DBE-1”142；以及具有被存储为块B1、B2、...、Bn的碎片B的“DBE-2”143。

[0041] 如图2中进一步所示，根据实施例，第二数据库区域B(这里标示为“DB西”，DBW)144包括具有被存储为块C1、C2、...、Cn的碎片C的分片数据库实例“DBW-1”145；以及具有被存储为块D1、D2、...、Dn的碎片D的“DBW-2”146。

[0042] 根据实施例，每个数据库区域或分片数据库实例组可以与碎片引导器或侦听器(例如，Oracle全球服务管理器(GSM)侦听器或另一种类型的侦听器)相关联。例如，如图2所示，碎片引导器或侦听器147可以与第一数据库区域A相关联；并且另一个碎片引导器或侦听器148可以与第二数据库区域B相关联。系统可以包括维护碎片拓扑层154的数据库驱动程序(例如，JDBC驱动程序)152，其在一段时间内学习碎片键范围并且将碎片键范围高速缓存到分片数据库中每个碎片的位置。

[0043] 根据实施例，客户端应用可以在连接请求162期间向连接池提供一个或多个碎片键；并且基于该一个或多个碎片键以及由碎片拓扑层提供的信息，连接池可以将连接请求路由到正确或适当的碎片。

[0044] 根据实施例，连接池还可以通过其碎片键来标识到特定碎片或块的连接，并且当从特定客户端应用接收到对相同碎片键的请求时允许重用连接。

[0045] 例如，如图2所示，根据实施例，到特定块(例如，块A1)的连接可以用来连接174到该块。如果池中不存在到特定碎片或块的可用连接，则系统可以尝试将现有的可用连接再利用到另一个碎片或块，并重用该连接。可以使数据库中跨碎片和块的数据分布对客户端应用是透明的，这也最小化了客户端上块的重新分片的影响。

[0046] 当碎片感知的客户端应用向连接池提供与连接请求相关联的一个或多个碎片键时；则如果连接池或数据库驱动程序已经具有碎片键的映射，则可以将连接请求直接转发到适当的碎片和块，在这个示例中，转发到块C2。

[0047] 当碎片感知的客户端应用不提供与连接请求相关联的碎片键时；或者如果连接池或数据库驱动程序不具有所提供的碎片键的映射；则可以将连接请求转发到适当的碎片引导器或侦听器。

[0048] 多租户环境

[0049] 根据实施例，系统可以包括使用连接标记支持基于云的环境或多租户环境。例如，多租户云环境可以包括应用服务器或数据库环境，该应用服务器或数据库环境包括或提供

在基于云的环境中对数据库的访问以供多个租户或租户应用使用。

[0050] 图3进一步图示了根据实施例的包括连接池的系统，包括支持在多租户环境中使用。

[0051] 类似于上述环境，可以由租户经由云或其它网络访问的软件应用可以在使用连接之前对从连接池检索到的连接进行初始化。

[0052] 如上所述，初始化的示例可以包括需要应用代码内的方法调用的简单的状态重新初始化，或者包括需要在网络上往返的数据库操作的更复杂的初始化。

[0053] 还如上所述，标记连接允许应用将任意名称/值对附接到连接，使得应用然后可以从连接池中请求具有期望标记的连接，包括从池中检索已经初始化的连接的能力，并且避免了重新初始化的时间和成本。

[0054] 如图3所示，根据实施例，多租户数据库环境180可以包括例如容器数据库(CDB)181和一个或多个可插拔数据库(PDB)，这里示为“PDB-1”182、“PDB-2”183和“PDB-3”184。

[0055] 根据实施例，每个PDB可以与多租户应用的租户(这里标示为“租户-1”、“租户-2”和“租户-3”)相关联，多租户应用由应用服务器或数据库环境185托管或者被提供为为外部客户端应用186，并且通过使用一个或多个Oracle真正应用集群(Real Application Cluster,RAC)实例186、188(在这个示例中包括“RAC-Instance-1”和“RAC-Instance-2”)；一个或多个服务(在这个示例中包括“服务-1”、“服务-2”和“服务-3”)以及租户到服务的映射190来提供对数据库环境的访问。

[0056] 在图3所示的示例中，由租户用来访问数据库环境的应用可以做出与该租户的数据源192、194、196相关联的连接请求，并且系统可以在需要的情况下切换服务198以利用到现有RAC实例或PDB的连接。

[0057] 服务器侧连接池

[0058] 根据实施例，系统可以利用服务器侧连接池标签(tagging)特征，诸如例如由Oracle数据库驻留连接池(DRCP)提供的特征。服务器侧连接池标签特征允许用户应用或客户端基于使用该数据库环境所理解的单个标签来选择性地获取到数据库环境的连接。

[0059] 根据实施例，每个连接仅关联一个标签。数据库服务器不将标签值传送给用户应用或客户端，而是传送标签匹配(例如，作为布尔(Boolean)值)。

[0060] 池中的连接的高效再利用

[0061] 根据实施例，系统可以包括对连接的高效再利用的支持。软件应用可以请求提供连接以使得能够访问数据库。响应于接收到请求，连接池可以首先确定具有确切期望特性的特定连接是否已经存在于池内，但是在请求时被借用。如果存在这样的连接，则连接池可以等待该特定连接变得可用的时间段，在本文中称为双等待。随后，如果未使特定连接在双等待时间段内可用，则连接池例如通过再利用其它连接恢复其正常操作。

[0062] 利用传统的连接池，无论何时由软件应用做出对具有具体期望特性(例如，具有特定连接标记或特定会话状态)的连接的请求，对于连接池的正常过程为：(1)确定是否存在具有期望特性的可用连接；或者如果没有，则(2)确定是否存在具有不同特性但可通过修改其特性被再利用以满足请求的可用连接；或者(3)尝试创建具有期望特性的全新连接。

[0063] 如果已经达到允许的连接的最大数量，则连接池一般将等待现有的虽然繁忙的连接被释放，然后再利用所释放的连接。

[0064] 在这样的环境中，连接池支持单等待功能，即，等待合适的连接变得可用，然后相应地动作。

[0065] 然而，根据实施例，系统可以包括双等待功能，该功能使得连接池能够代替上述单等待功能或除了上述单等待功能之外还首先确定具有确切期望特性的特定连接是否已经存在，但是在请求时被借用；并且如果是的话，则在如上所述恢复其正常过程之前，等待该连接变得可用的时间段(在本文中称为连接双等待超时(Δt_{cdw}))。

[0066] 利用双等待功能，增加了重用连接的可能性，而不需要再利用或以其它方式必须修改这些连接的特性。由于改变连接特性是计算上昂贵的操作，因此就往返/延时和服务器侧CPU使用而言，双等待的使用提供了响应于请求而提供和/或再利用连接的高效手段。

[0067] 例如，根据实施例，系统可以包括支持基于云的环境或多租户环境，包括包含或提供对对数据库的访问的应用服务器或数据库，以供多个租户或租户应用使用。

[0068] 在这样的环境中，当对于给定租户在池中不存在可用连接时，则代替再利用与另一个租户相关联的可用连接(这将是计算上昂贵的操作)，连接池可以双等待，包括首先确定该池是否已经包括针对给定租户的连接，即使该特定连接当前被借用(即，具有确切期望特性的特定连接)。如果是的话，则连接池可以等待该特定(即，给定租的)连接变得可用的时间段，使得可以在不改变特性的情况下提供该连接。如果没有具有确切期望特性的连接可用，则连接池将仅再利用另一个租户的连接。

[0069] 图4-图8图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0070] 如上所述，标记连接允许客户端软件应用将任意名称/值对附加到连接。然后该应用可以请求具有期望标记的连接，该连接将由连接池提供。通过将特定标记与特定连接状态相关联，应用可以潜在地从池中检索已初始化的连接，并避免重新初始化的时间和成本。例如，如果软件应用希望使用特定类型的连接(例如，(红色)连接)在数据库上做出请求，则连接池逻辑可以创建新的(红色)连接；或者将现有的空闲连接从例如(蓝色或绿色)再利用到(红色)。

[0071] 然而，连接池的借用操作通常是复杂的，包括若干协调的连接选择机制-例如，RAC负载均衡、高可用性、亲和性、连接验证、标签处理、收获或各种超时的处理。虽然上面的示例使用不同的颜色示出了再利用可以包括例如(蓝色)连接到(红色)连接的变换；但是在实际实践中，这种变换可能需要关闭一个RAC实例上的连接，并在另一个RAC实例上创建替换连同实现这种操作所需的各种过程。

[0072] 对于计算上不昂贵的特性，这可以是相对直接的处理。然而，计算昂贵的特性改变(例如，改变容器数据库内的可插拔数据库，如可能在多租户环境中实现的)可能需要更多考虑的方法。

[0073] 如图4中所示，根据实施例，连接池可以使用双等待功能250来执行等待252，如果适当，接下来在等待之后再利用连接254，其处理在下面进一步详细描述。

[0074] 根据实施例，所描述的方法认识到的是，即使对于借用请求可以存在匹配的连接，并且借用线程对于这个连接是双等待，也不能保证连接在连接双等待超时(Δt_{cdw})内被返回。

[0075] 为了解决这个问题，根据实施例，可以将连接双等待超时设置为小于连接等待超时(Δt_{cw})，因此如果双等待不成功，则系统可以还原回到其正常过程，即，回到单等待借用

模式。

[0076] 例如，根据实施例，连接双等待超时（Δt_{cdw}）可以被设置为50毫秒，其中（总）连接等待超时或时间段（Δt_{cw}）被设置为500毫秒。

[0077] 此外，根据实施例，所描述的方法认识到系统一般可能无法预测被借用的连接何时将被返回到池。然而，依靠信号事件获得这些信息在性能方面将是昂贵的。例如，如果系统要等待信号API，则对于每个返回的连接，这将减缓连接借用/返回机制的性能。

[0078] 为了解决这个问题，根据实施例，连接池支持将少量等待信号API调用与池的繁忙等待轮询组合以获取所需连接的借用操作。一般而言，借用操作等待被发信号通知的唯一情况是当前没有连接可借用。有效可用连接的出现触发所有当前借用线程开始轮询。

[0079] 一般而言，线程应当以小的（例如，纳秒或零时间）间隔来轮询池以尽可能快地尝试获取适当的连接。然而，在实践中，零时间轮询间隔可能导致昂贵的CPU使用。为了解决这个问题，根据实施例，连接池支持自动调谐处理，如下面进一步描述的，用以提供CPU负载与这些间隔的耐久性之间的均衡。

[0080] 例如，根据实施例，轮询间睡眠间隔可以用以下来计算：

[0081] $\Delta t_{sleep} = \Delta t_{o}(|CPU_{expected} - CPU_{current}|)$

[0082] 其中Δt_{0}是经验恒定时间间隔，$CPU_{expected}$是预期的CPU利用率的百分比，并且$CPU_{current}$是当前CPU利用率的百分比。根据实施例，CPU利用率的值是所有参与线程的CPU利用率的积分值。

[0083] 单等待模式和双等待模式

[0084] 根据实施例，系统被配置为使用两种不同模式来轮询连接池：在本文中一般称为连接单等待模式和连接双等待模式。

[0085] 在连接单等待模式中，连接池逻辑假设借用请求预期任何连接显示可用于保留（例如，（红色）、（蓝色）或（绿色）连接中的任何连接，特定的连接类型无关紧要），如果需要就进行再利用（例如，（红色）连接变换为（绿色）），然后借用。这种单等待模式持续连接等待超时（Δt_{cw}）的持续时间，可以将该持续时间定义为可配置的连接池属性。

[0086] 在连接双等待模式中，连接池假设借用请求仅预期所需质量或类型的连接，即，具有确切期望特性的连接（例如，仅（红色）连接），使得不需要再利用。这种双等待模式持续连接双等待超时（Δt_{cdw}）的持续时间，可以将该持续时间定义为可配置的连接池属性，或者由系统自动/动态地计算。

[0087] 通过定义，连接等待超时（Δt_{cw}）是借用线程为连接而轮询池的最长间隔。

[0088] 根据实施例，系统可以被配置为首先轮询双等待模式，然后轮询单等待模式。一旦双等待确定不保证连接在Δt_{cdw}时间段内被释放，则设置（$\Delta t_{cdw} = \Delta t_{cw}$）的用处不大，因为在许多情况下，在不可避免的再利用之前，这将需要花费时间而什么都没有实现。相反，设置值$\Delta t_{cdw} = 0$将意味着关闭双等待模式。

[0089] 以上指出，根据实施例，双等待超时Δt_{cdw}的值一般应当在一些界限之间浮动，例如：

[0090] $0 \leqslant \Delta t_{cdw} \leqslant \Delta t_{cw}$

[0091] 一旦值$\Delta t_{cdw} \leqslant \Delta t_{cw}$，对于任何连接的单等待就要花费某个非零时间间隔，并且如果双等待不成功，则对该连接进行再利用。

[0092] 根据实施例，双等待超时Δt_{cdw}的值可以被定义为可配置的属性。如上所述，另一种方法是使用自动调谐处理。例如，可以使双等待间隔与统计平均借用(“ab”)时间间隔Δt_{ab}成比例，该Δt_{ab}可以通过以下公式计算：

[0093] $$\Delta t_{ab} = \frac{\sum_{i=0}^{i=n} \Delta t_i}{n}$$

[0094] 其中n是最近Δt_i的某个量-对于池中所需类型的所有连接的个体(i)借用时间间隔(即，在连接借用和返回到池之间经过的时间)。Δt_{ab}的值与符合未决借用请求的被借用连接的数量(N_c)成反比，并且与类似的未决借用请求(即，寻求相同类型的连接的那些请求)的数量(N_{br})成比例。

[0095] 根据实施例，系统根据自动调谐处理可以自动/动态地计算双等待超时(Δt_{cdw})的值为：

[0096] $$\Delta t_{cdw} = \min\left(F\Delta t_{cw}, \frac{N_{br}\Delta t_{ab}}{N_c}\right)$$

[0097] 其中F是在[0..1]的范围内的$\Delta t_{cdw}/\Delta t_{cw}$的最大允许值。

[0098] 虽然有可能在每次单轮询之前调整Δt_{cdw}的值，但在实践中，这种方法将需要一些分析来确定所需类型的被借用连接的数量并且将需要一些资源。相反，系统可以在每次特定的借用操作中计算一次Δt_{cdw}，然后将其用作该特定借用操作的常数。

[0099] 例如，如图5中所示，根据实施例，连接池可以接收对特定类型(例如，(蓝色)标记)连接的请求260。

[0100] 如图6中所示，根据实施例，系统可以包括双等待功能，该功能使得连接池能够代替上述单等待功能或除了上述单等待功能之外还首先确定具有确切期望特性的特定连接是否已经存在，但是在请求时被借用；并且如果是的话，则等待该连接变得可用的时间段。如果在双等待超时(Δt_{cdw})内使期望连接可用，则将该期望连接返回262到客户端应用以供其使用。

[0101] 如图7中所示，根据实施例，如果合适的连接在双等待超时(Δt_{cdw})内未变得可用，则连接池可以在单等待模式期间恢复其正常过程。

[0102] 如图8中所示，根据实施例，在(剩余或单等待)连接超时(Δt_{cw})期间，在这个示例中，连接池可以确定是否将现有的空闲连接再利用264为(蓝色)连接266，然后将其返回268到客户端应用。

[0103] 以下附加示例说明上述内容：

[0104] 示例1：考虑100个连接的池，其中Δt_{cw}是10秒，存在两种类型的连接：50个(绿色)连接和50个(蓝色)连接。当前存在全部50个(绿色)连接和25个(蓝色)连接被借用，25个(蓝色)连接可用，并且存在150个(绿色)借用请求未决。现在Δt_{ab}是1秒。考虑其中F为0.5的示例。因此，Δt_{cdw}=min((0.5×10),((150+1)/50))，即，等于3秒。借用请求将双等待3秒，预期确切所需类型的连接，然后等待另外7秒以获取任何连接并对该连接进行再利用。

[0105] 示例2：考虑具有相同输入的上述示例，其中存在300个借用请求而不是150个。则Δt_{cdw}=min((0.5×10),((300+1)/50))，即，等于5秒。这意味着要借用(绿色)连接，借用线程将双等待5秒，如果没有借用，则将单等待另外5秒，如果需要就对该连接进行再利用。

[0106] 示例3:考虑使用相同输入的上述示例,并且存在5000个借用请求而不是300个。Δt_{cdw}=5秒。这导致与示例2相同的行为。

[0107] 连接借用处理

[0108] 图9进一步图示了根据实施例的在连接池环境中支持连接的高效再利用。

[0109] 如图9中所示,根据实施例,连接借用操作可以被视为从双等待轮询准备步骤280前进到双等待轮询周期282、前进到单等待轮询准备步骤290、前进到单等待轮询周期292、前进到一个或多个借用后动作步骤300,其中每一个步骤在下面进一步描述。

[0110] 双等待轮询准备:

[0111] 根据实施例,双等待轮询准备可以包括:

[0112] a.计算Δt_{cdw}。

[0113] b.保存当前的时间戳,并开始双等待轮询。

[0114] 双等待轮询周期:

[0115] 根据实施例,双等待轮询周期可以包括:

[0116] a.确定Δt_{cdw}为双等待轮询的时间间隔。检查它是否已经过去。如果过去了,则退出双等待周期并开始单等待轮询准备。

[0117] b.轮询适当类型的可用有效连接。如果找到适当类型的可用有效连接,则跳转到借用后动作;借用请求结束。如果没有找到,则前往下一步。

[0118] c.计算Δt_{sleep}。

[0119] d.睡眠,并且前往(a)。

[0120] 单等待轮询准备:

[0121] 根据实施例,单等待轮询准备可以包括:

[0122] a.计算$\Delta t_{cw}-\Delta t_{cdw}$。

[0123] b.保存当前的时间戳,并开始单等待轮询。

[0124] 单等待轮询周期:

[0125] 根据实施例,单等待轮询周期可以包括:

[0126] a.$\Delta t_{cw}-\Delta t_{cdw}$是用于单等待轮询的时间间隔。检查它是否已经过去。如果过去了,则退出单等待周期并开始执行借用后动作。

[0127] b.轮询任何类型的可用有效连接。如果找到任何类型的可用有效连接,则跳转到借用后动作;借用请求结束。如果没有找到,则前往下一步。

[0128] c.如果存在连接池增长的空间,则利用所需类型的连接进行增长。如果新的连接已经增长,则获得该连接并跳转到借用后动作;借用请求结束。如果没有增长,则前往下一步。

[0129] d.计算Δt_{sleep}。

[0130] e.睡眠,前往(a)。

[0131] 借用后动作:

[0132] 根据实施例,借用后动作可以包括:

[0133] a.对连接上执行一些借用后动作。

[0134] b.返回连接,或者空。

[0135] 根据实施例,对于双等待机制的连接返回操作,常规连接返回逻辑根据需要工作,

从而从应用返回连接，因为不再需要该连接。返回增加可用连接的数量，标记池中可用的连接，并且如果正在等待至少一个可用连接则向池发送信号。

[0136] 图10图示了根据实施例的在连接池环境中提供支持连接的高效再利用的方法。

[0137] 如图10中所示，根据实施例，在步骤310处，在应用服务器或数据库环境处，提供控制创建和使用连接池中的连接对象的连接池逻辑或程序代码，其中软件应用可以从连接池中请求连接，并使用所提供的连接来访问数据库。

[0138] 在步骤312处，提供与连接池一起使用的双等待功能，其中响应于对具有期望特性的特定连接的请求，连接池首先确定具有这些确切期望特性的连接是否已经存在于连接池内，但是在请求时被借用；并且如果是的话，则连接池等待双等待超时或该连接变得可用的时间段。

[0139] 在步骤314处，当双等待超时或时间段到期时，连接池还原到单等待模式以确定是否存在具有不同特性但是可以通过修改其特性被再利用以满足请求的可用连接；或尝试创建具有期望特性的全新连接。

[0140] 在步骤316处，连接池提供以下中的一个：具有确切期望特性的连接、再利用的连接、新连接，或者返回空值或其它值。

[0141] 本发明的实施例可以使用一个或多个常规的通用或专用数字计算机、计算设备、机器或微处理器(包括根据本公开的教导进行编程的一个或多个处理器、存储器和/或计算机可读存储介质)来方便地实现。如对软件领域技术人员将显而易见的，基于本公开的教导，适当的软件编码可以由熟练的程序员容易地准备。

[0142] 在一些实施例中，本发明包括计算机程序产品，计算机程序产品是其上/其中存储有指令的非暂态存储介质或计算机可读介质，指令可以用来对计算机进行编程以执行本发明的任何处理。存储介质的示例可以包括但不限于任何类型的盘(包括软盘、光盘、DVD、CD-ROM、微驱动器和磁光盘)、ROM、RAM、EPROM、EEPROM、DRAM、VRAM、闪存设备，磁卡或光卡、纳米系统(包括分子存储器IC)，或适于存储指令和/或数据的任何类型的介质或设备。

[0143] 本发明的实施例的前述描述是为了说明和描述的目的而提供的。它不旨在是详尽的或者将本发明限于所公开的精确形式。许多修改和变化对于本领域技术人员将是显而易见的。修改和变化包括所公开特征的任何相关组合。实施例的选择和描述是为了最好地解释本发明的原理及其实际应用，由此使本领域的其他技术人员能够理解本发明的各种实施例以及适于预期的特定用途的各种修改。

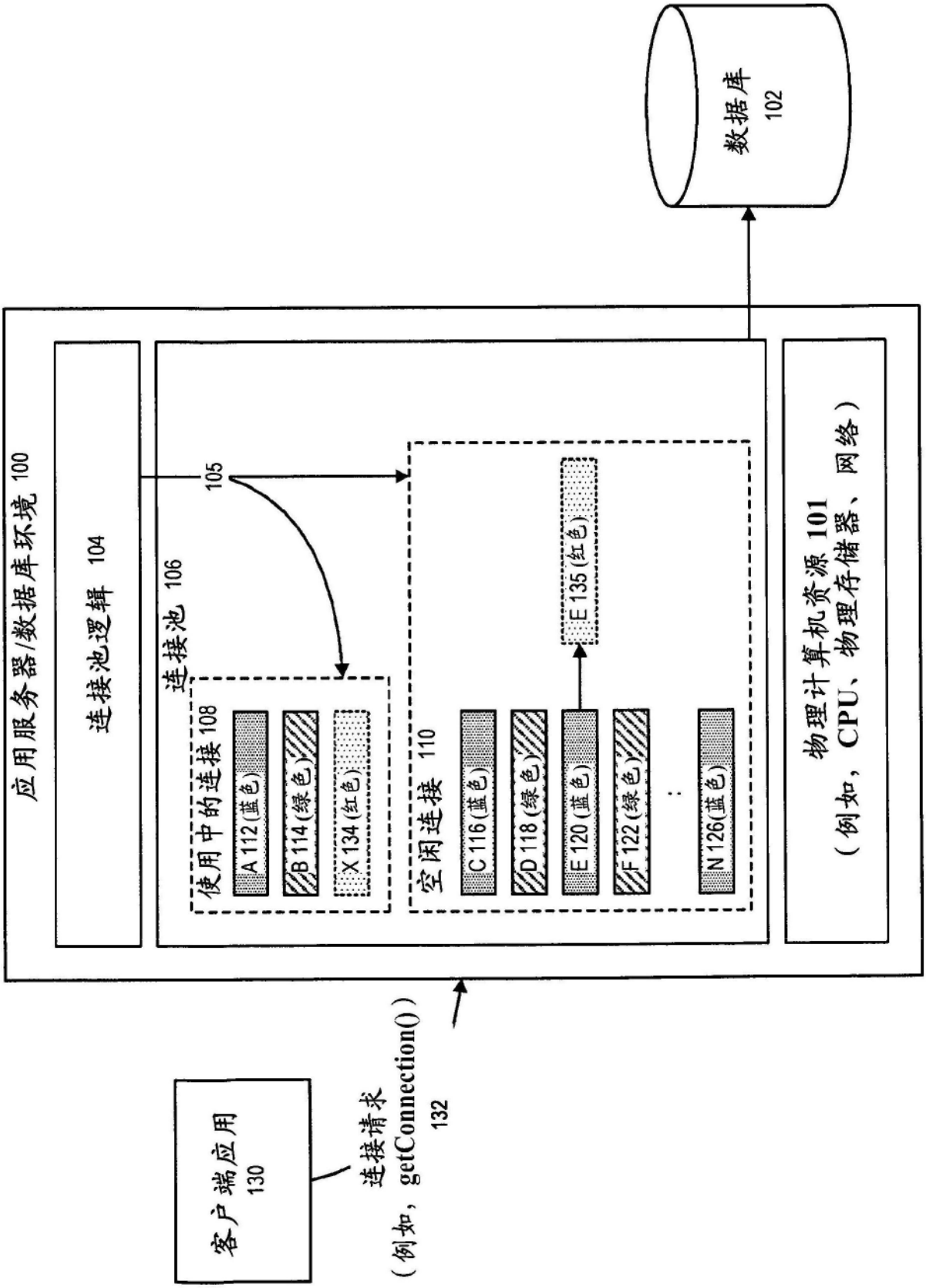

应用服务器/数据库环境 100
连接池逻辑 104
连接池 106
105
使用中的连接 108
A 112(蓝色)
B 114(绿色)
X 134(红色)
空闲连接 110
C 116(蓝色)
D 118(绿色)
E 120(蓝色)
E 135(红色)
F 122(绿色)
N 126(蓝色)
物理计算机资源 101
(例如，CPU、物理存储器、网络)
数据库
102
客户端应用
130
连接请求
(例如，getConnection())
132

图1

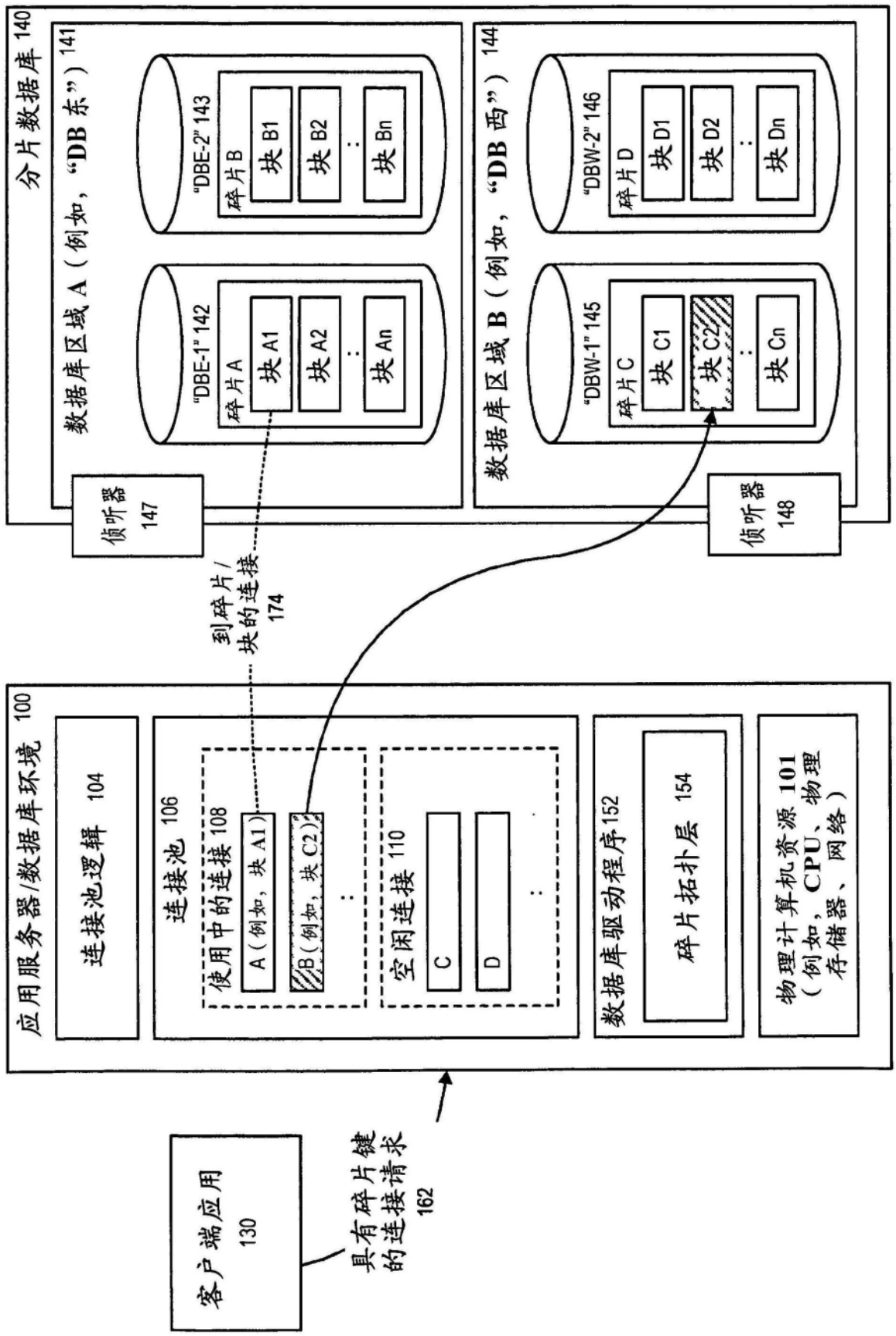

分片数据库 140
数据库区域 A（例如，“DB 东”）141
“DBE-1”142
碎片 A
块 A1
块 A2
块 An
“DBE-2”143
碎片 B
块 B1
块 B2
块 Bn
数据库区域 B（例如，“DB 西”）144
“DBW-1”145
碎片 C
块 C1
块 C2
块 Cn
“DBW-2”146
碎片 D
块 D1
块 D2
块 Dn
侦听器 147
侦听器 148
到碎片/块的连接 174
应用服务器/数据库环境 100
连接池逻辑 104
连接池 106
使用中的连接 108
A（例如，块 A1）
B（例如，块 C2）
空闲连接 110
C
D
数据库驱动程序 152
碎片拓扑层 154
物理计算机资源 101（例如，CPU、物理存储器、网络）
客户端应用 130
具有碎片键的连接请求 162

图2

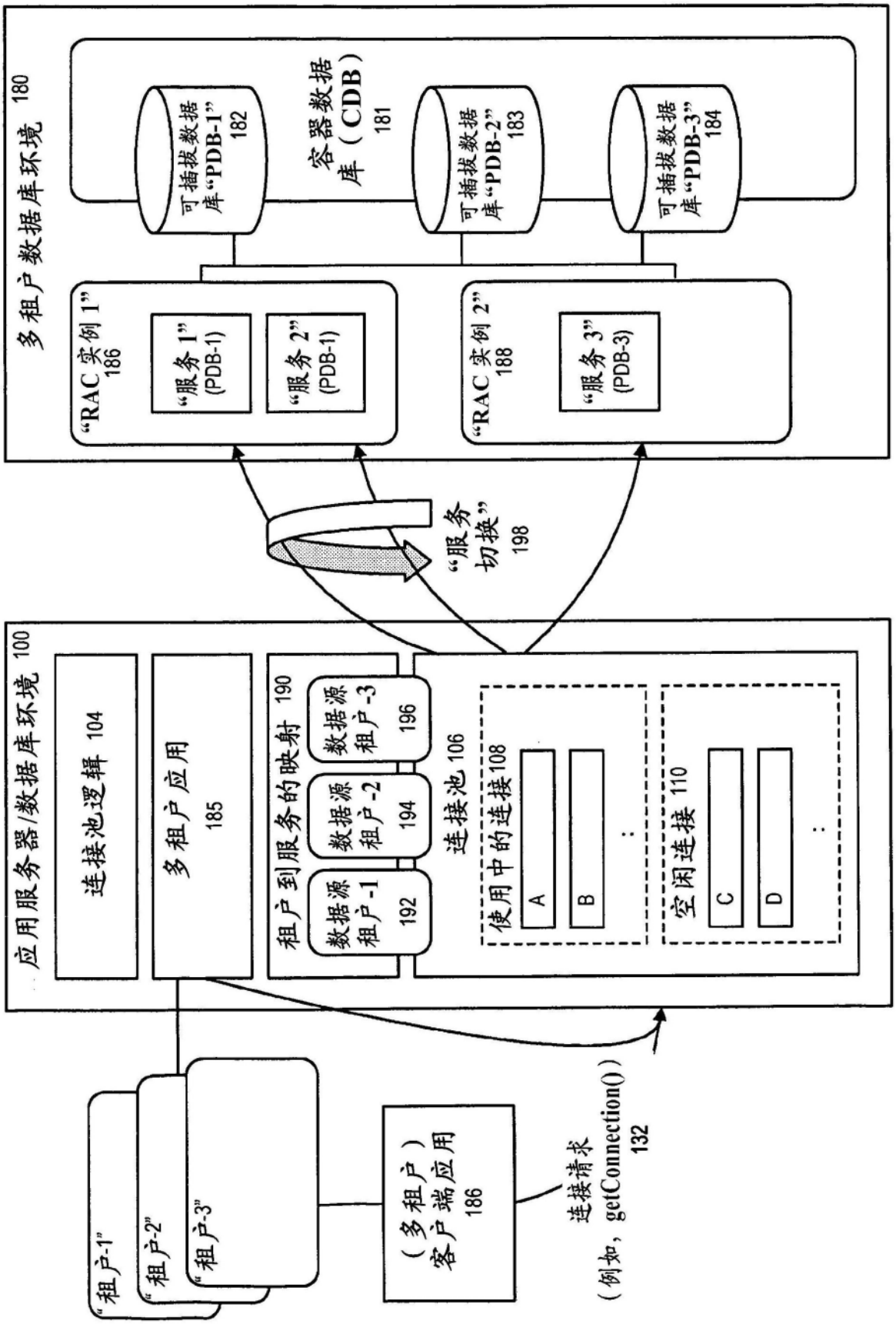

多租户数据库环境 180
容器数据库（CDB）181
可插拔数据库“PDB-1”182
可插拔数据库“PDB-2”183
可插拔数据库“PDB-3”184
“RAC 实例 1”186
“服务 1”(PDB-1)
“服务 2”(PDB-1)
“RAC 实例 2”188
“服务 3”(PDB-3)
“服务切换”198
应用服务器/数据库环境 100
连接池逻辑 104
多租户应用 185
租户到服务的映射 190
数据源租户-1 192
数据源租户-2 194
数据源租户-3 196
连接池 106
使用中的连接 108
A
B
空闲连接 110
C
D
“租户-1”
“租户-2”
“租户-3”
（多租户）客户端应用 186
连接请求（例如，getConnection()）132

图3

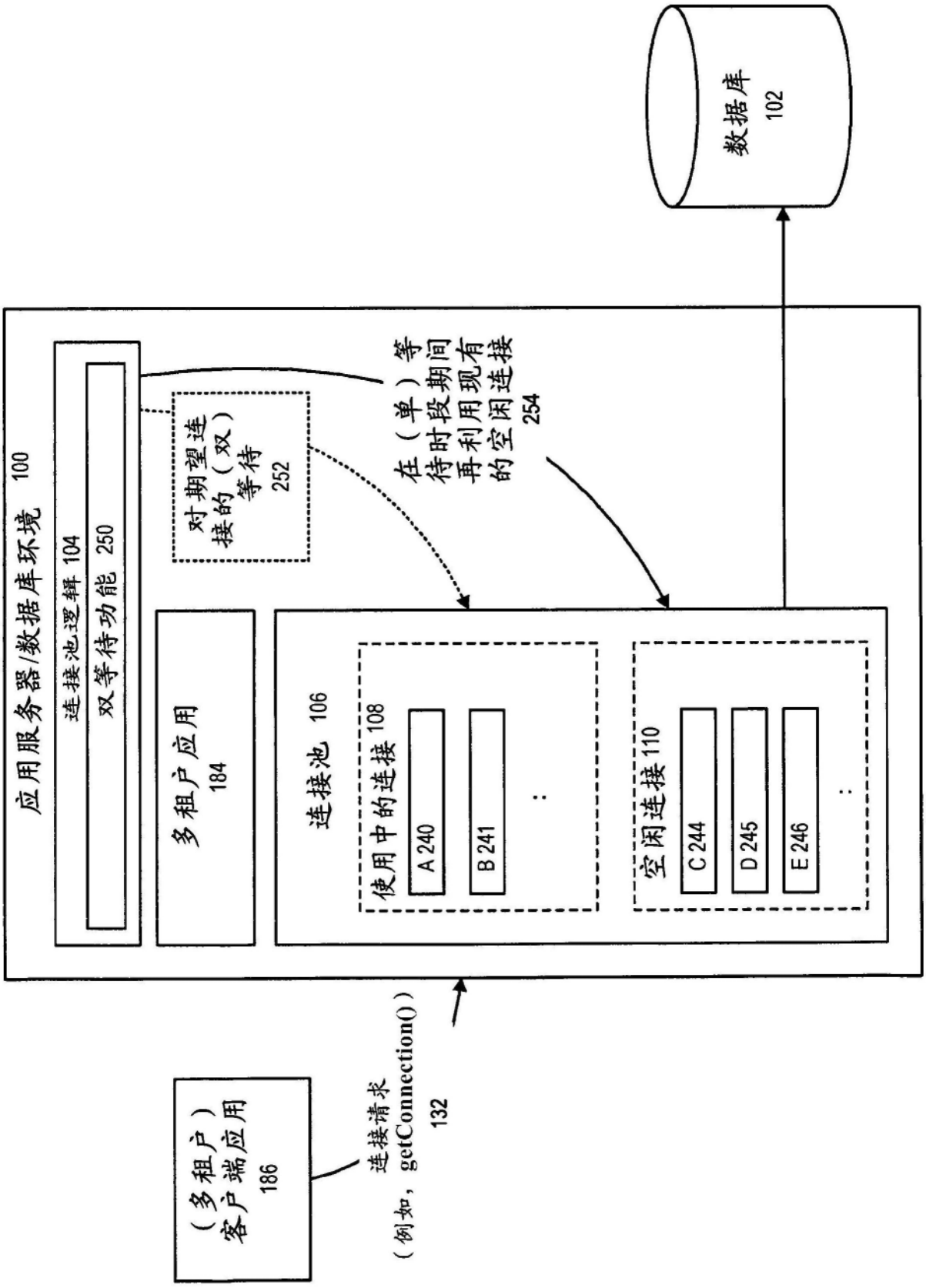

数据库
102
应用服务器/数据库环境 100
连接池逻辑 104
双等待功能 250
对期望连接的（双）等待 252
在（单）等待时段期间再利用现有的空闲连接 254
多租户应用 184
连接池 106
使用中的连接 108
A 240
B 241
空闲连接 110
C 244
D 245
E 246
（多租户）客户端应用 186
连接请求（例如，getConnection()） 132

图4

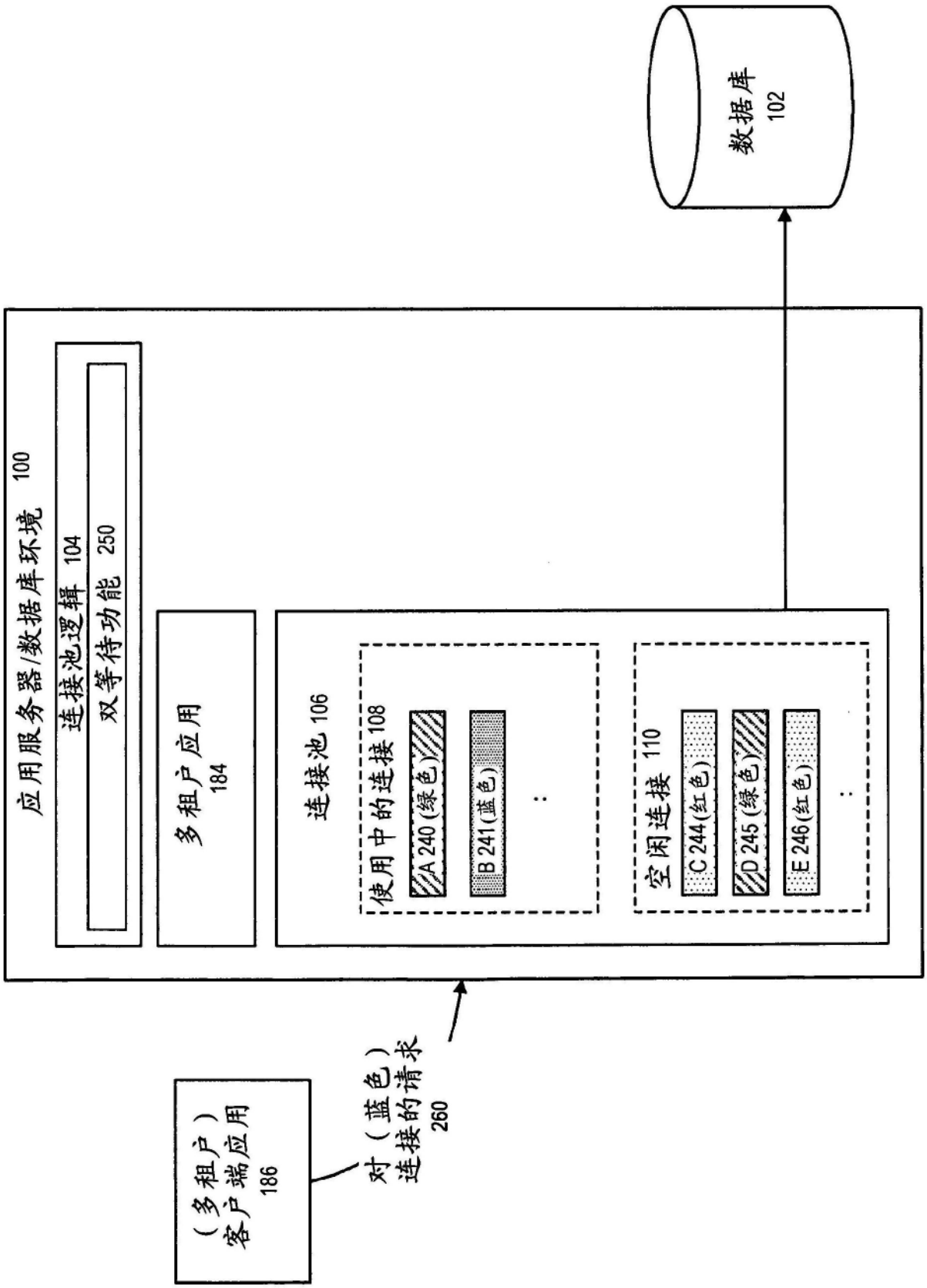

数据库
102
应用服务器/数据库环境 100
连接池逻辑 104
双等待功能 250
多租户应用
184
连接池 106
使用中的连接 108
A 240 (绿色)
B 241(蓝色)
空闲连接 110
C 244(红色)
D 245 (绿色)
E 246(红色)
(多租户)
客户端应用
186
对(蓝色)
连接的请求
260

图5

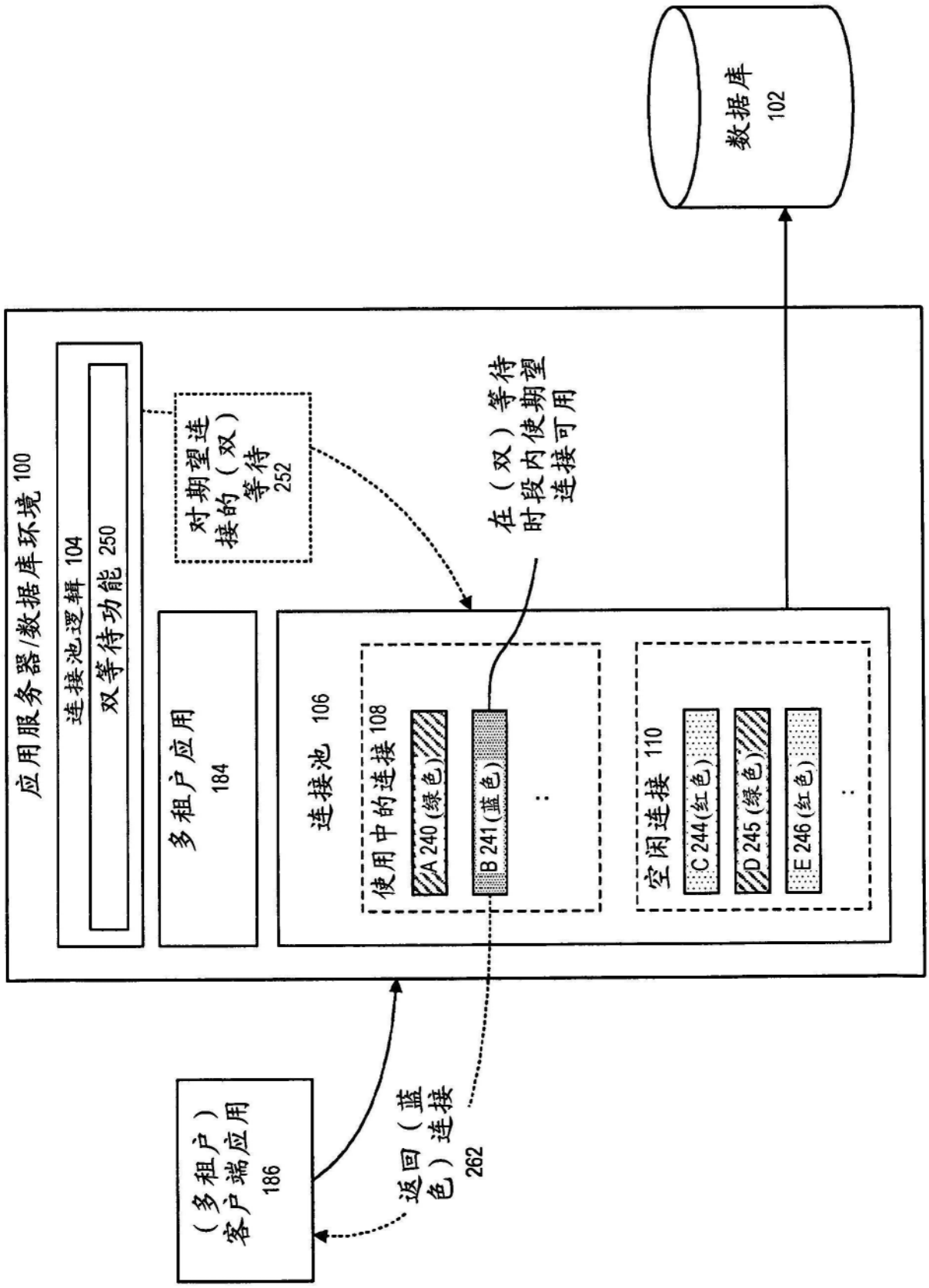

数据库
102
应用服务器/数据库环境 100
连接池逻辑 104
双等待功能 250
对期望连接的（双）等待 252
多租户应用 184
连接池 106
使用中的连接 108
A 240 (绿色)
B 241(蓝色)
空闲连接 110
C 244(红色)
D 245 (绿色)
E 246 (红色)
在（双）等待时段内使期望连接可用
（多租户）客户端应用 186
返回（蓝色）连接 262

图6

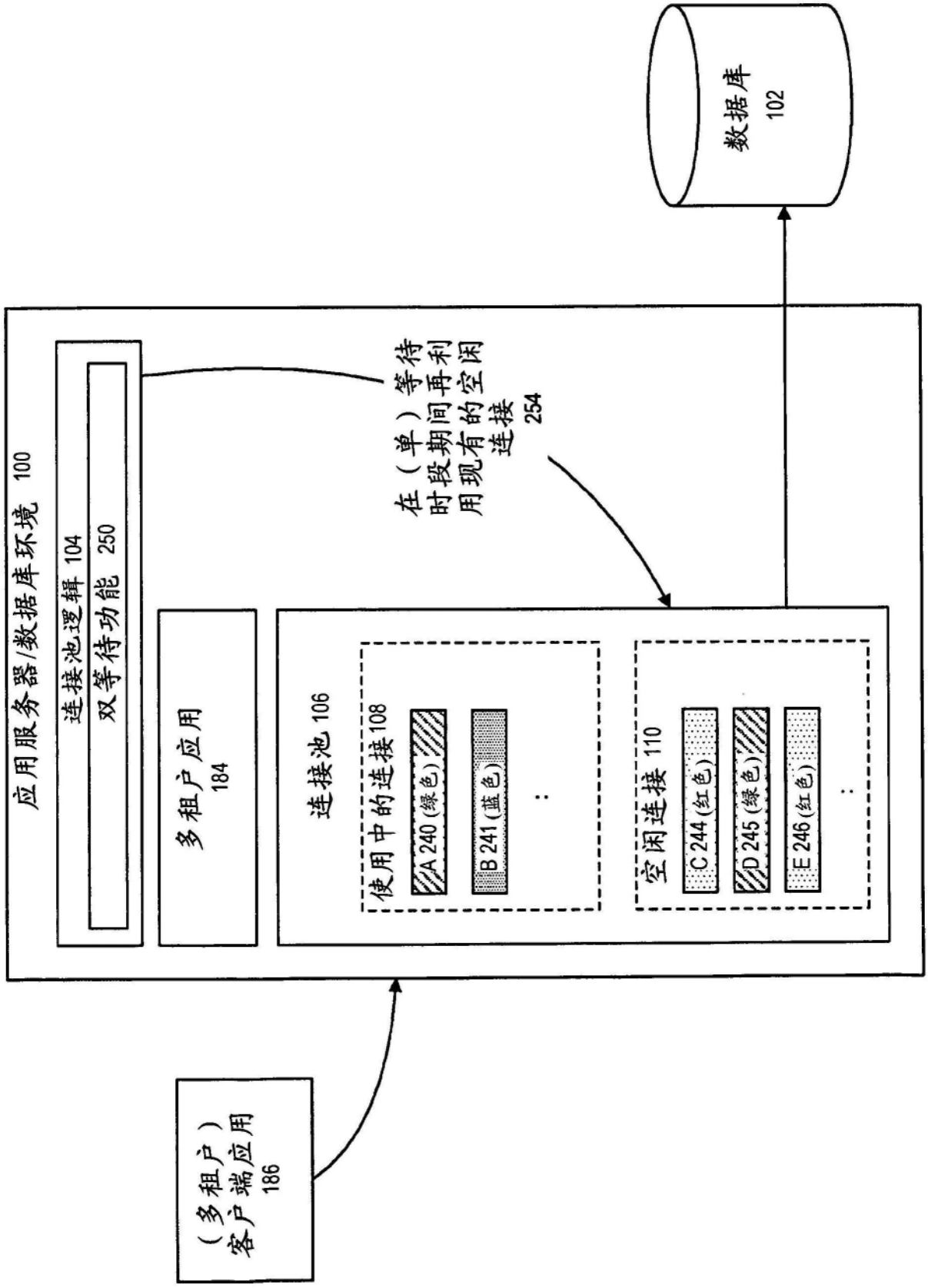

数据库
102
应用服务器/数据库环境 100
连接池逻辑 104
双等待功能 250
多租户应用
184
连接池 106
使用中的连接 108
A 240 (绿色)
B 241 (蓝色)
空闲连接 110
C 244 (红色)
D 245 (绿色)
E 246 (红色)
在（单）等待时段期间再利用现有的空闲连接
254
（多租户）客户端应用
186

图7

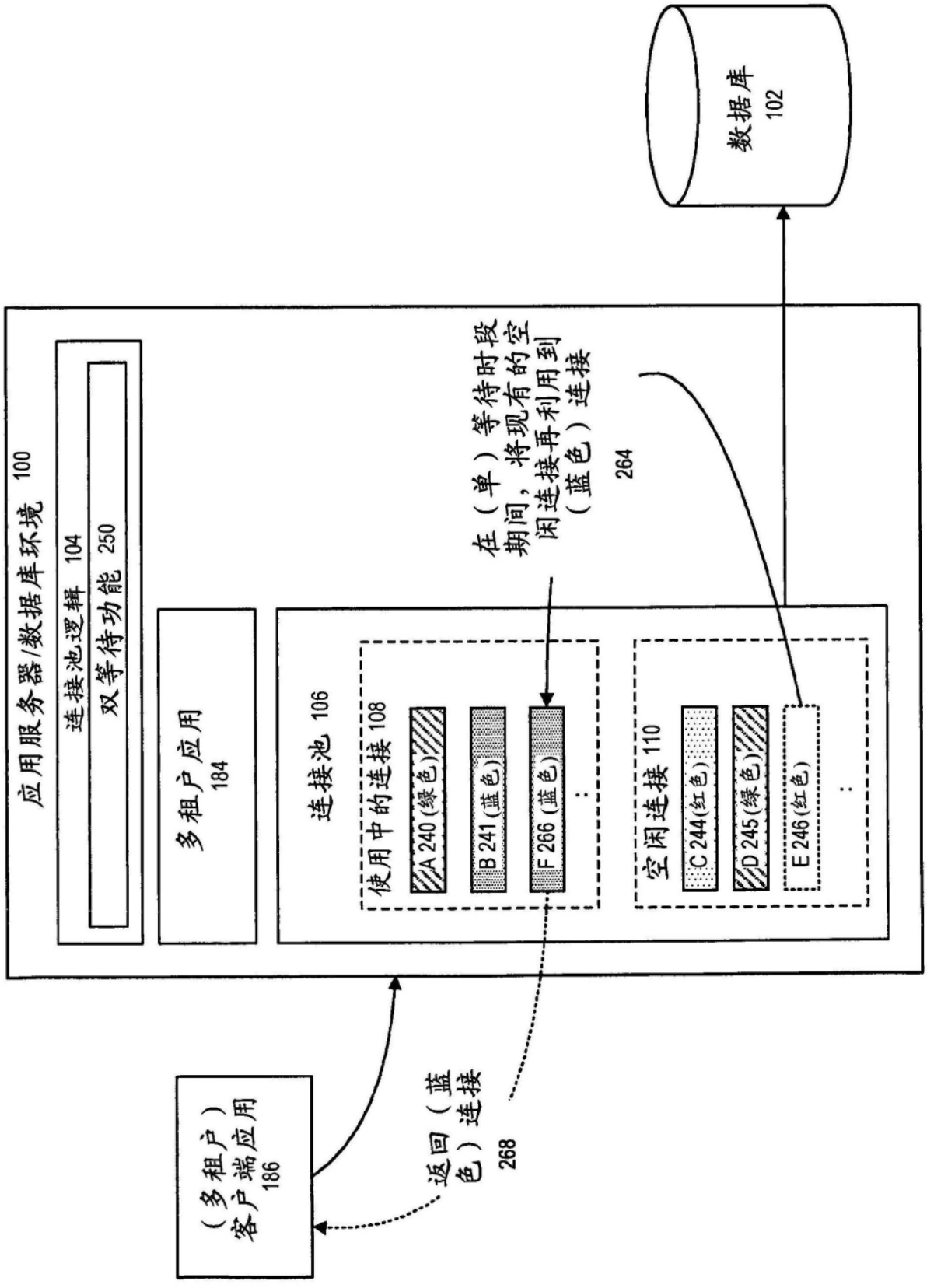

数据库
102
应用服务器/数据库环境 100
连接池逻辑 104
双等待功能 250
多租户应用
184
连接池 106
使用中的连接 108
A 240 (绿色)
B 241(蓝色)
F 266 (蓝色)
空闲连接 110
C 244(红色)
D 245(绿色)
E 246(红色)
在（单）等待时段期间，将现有的空闲连接再利用到（蓝色）连接
264
（多租户）客户端应用
186
返回（蓝色）连接
268

图8

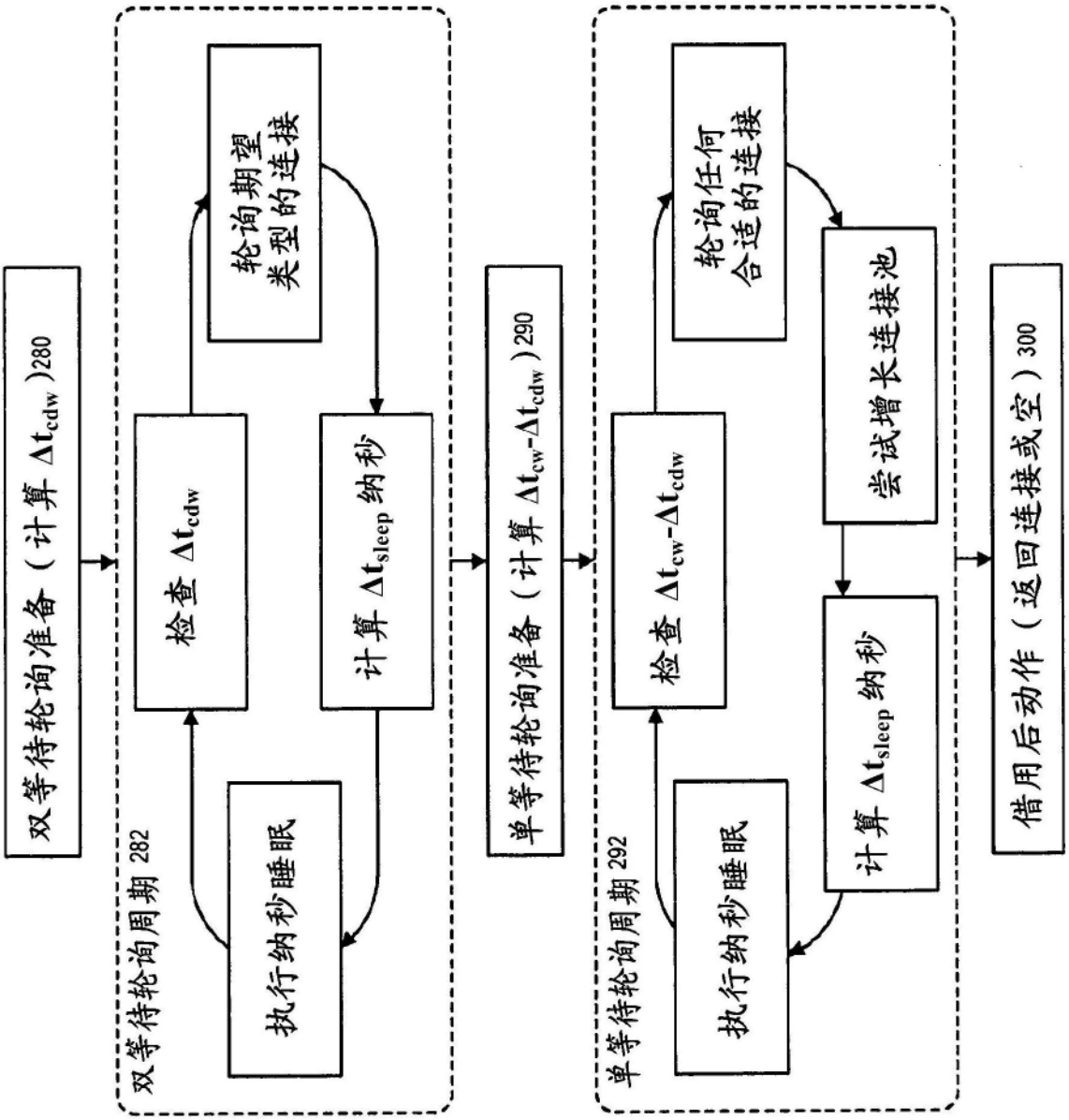

双等待轮询准备（计算 Δt_{cdw}）280
双等待轮询周期 282
检查 Δt_{cdw}
轮询期望类型的连接
计算 Δt_{sleep} 纳秒
执行纳秒睡眠
单等待轮询准备（计算 Δt_{cw}-Δt_{cdw}）290
单等待轮询周期 292
检查 Δt_{cw}-Δt_{cdw}
轮询任何合适的连接
尝试增长连接池
计算 Δt_{sleep} 纳秒
执行纳秒睡眠
借用后动作（返回连接或空）300

图9

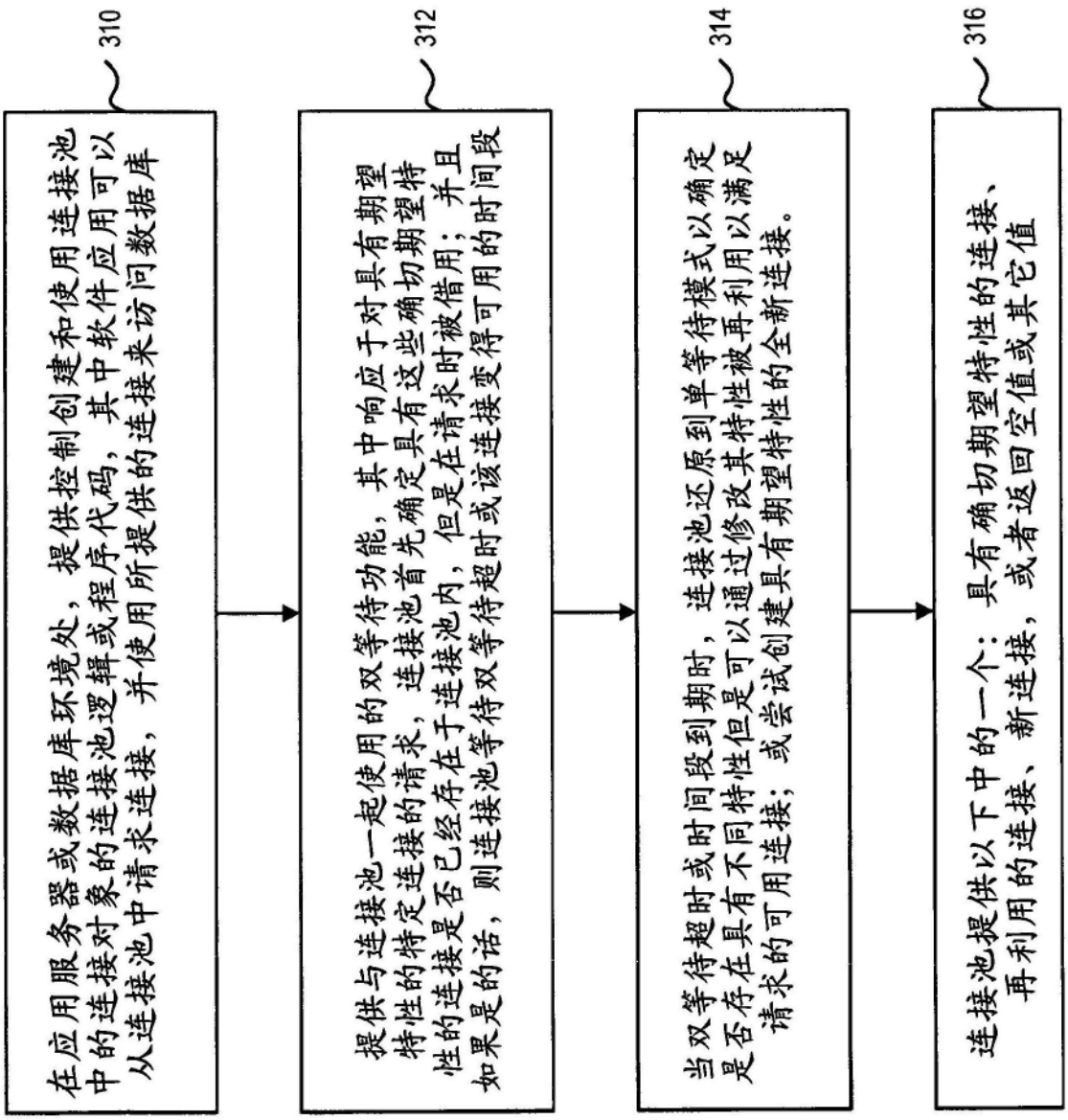

310
在应用服务器或数据库环境处，提供控制创建和使用连接池中的连接对象的连接池逻辑或程序代码，其中软件应用可以从连接池中请求连接，并使用所提供的连接来访问数据库
312
提供与连接池一起使用的双等待功能，其中响应于对具有期望特性的特定连接的请求，连接池首先确定具有这些确切期望特性的连接是否已经存在于连接池内，但是在请求时被借用；并且如果是的话，则连接池等待双等待超时或该连接变得可用的时间段
314
当双等待超时或时间段到期时，连接池还原到单等待模式以确定是否存在具有不同特性但是可以通过修改其特性被再利用以满足请求的可用连接；或尝试创建具有期望特性的全新连接。
316
连接池提供以下中的一个：具有确切期望特性的连接、再利用的连接、新连接，或者返回空值或其它值

图10