

电子装置、数据链式归档的方法、系统及存储介质

优先权申明

本申请基于巴黎公约申明享有 2017 年 03 月 06 日递交的申请号为 CN 201810183496.9、名称为“电子装置、数据链式归档的方法及存储介质”中国专利申请的优先权，该中国专利申请的整体内容以参考的方式结合在本申请中。

技术领域

本申请涉及通信技术领域，尤其涉及一种电子装置、数据链式归档的方法、系统及存储介质。

背景技术

在企业机构中，随着业务量增长，数据库数据积压会越来越多。大量的数据积压会导致数据库系统性能下降，直接影响用户体验和用户操作时效。由此需要一个合理高效的方案来对数据进行归档。传统的数据归档方案是直接将数据搬到归档库，由于数据量非常大且跨及两个数据库，因此数据归档操作较慢，不能保持数据库系统性能的稳定性的。

发明内容

本申请的目的在于提供一种电子装置、数据链式归档的方法、系统及存储介质，旨在快速、高效地将数据库中的数据归档。

为实现上述目的，本申请提供一种电子装置，所述电子装置包括存储器及与所述存储器连接的处理器，所述存储器中存储有可在所述处理器上运行的处理系统，所述处理系统被所述处理器执行时实现如下步骤：

第一写入步骤，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的

已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

第二写入步骤，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

归档步骤，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

为实现上述目的，本申请还提供一种数据链式归档的方法，所述数据链式归档的方法包括：

S1，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

S2，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

S3，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

为实现上述目的，本申请还提供一种数据链式归档的系统，所述数据链式归档的系统包括：

第一写入模块，用于将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

第二写入模块，用于当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

归档模块，用于当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

本申请还提供一种计算机可读存储介质，所述计算机可读存储介质上存储有处理系统，所述处理系统被处理器执行时实现上述的方法的步骤。

本申请的有益效果是：本申请在对数据库中的数据归档时，采用链式多线程的方式进行归档，包括三步操作，每一步操作执行完成后方可执行下一步操作，在第一步操作中对数据原表进行分区并采用多个并行线程的方式将数据写入临时表，在第二步操作中用单线程筛选出符合条件的数据，在第三步操作中对临时表进行分区并采用多个并行线程的方式将数据归档至该数据库的归档表中，能够快速、高效地将数据库中的数据归档，保持数据库系统性能的稳定。

附图说明

图1为本申请电子装置一实施例的硬件架构的示意图；

图2为数据库的数据原表中父节点数据一实施例的示例图；

图 3 为图 2 所示父节点数据下的子节点数据一实施例的示例图；

图 4 为本申请数据链式归档的方法一实施例的流程示意图。

具体实施方式

为了使本申请的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本申请进行进一步详细说明。应当理解，此处所描述的具体实施例仅用以解释本申请，并不用于限定本申请。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

需要说明的是，在本申请中涉及“第一”、“第二”等的描述仅用于描述目的，而不能理解为指示或暗示其相对重要性或者隐含指明所指示的技术特征的数量。由此，限定有“第一”、“第二”的特征可以明示或者隐含地包括至少一个该特征。另外，各个实施例之间的技术方案可以相互结合，但是必须是以本领域普通技术人员能够实现为基础，当技术方案的结合出现相互矛盾或无法实现时应当认为这种技术方案的结合不存在，也不在本申请要求的保护范围之内。

参阅图 1 所示，是本申请电子装置以实施例的硬件结构示意图。电子装置 1 是一种能够按照事先设定或者存储的指令，自动进行数值计算和/或信息处理的设备。所述电子装置 1 可以是计算机、也可以是单个网络服务器、多个网络服务器组成的服务器组或者基于云计算的由大量主机或者网络服务器构成的云，其中云计算是分布式计算的一种，由一群松散耦合的计算机集组成的一个超级虚拟计算机。

在本实施例中，电子装置 1 可包括，但不限于，可通过系统总线相互通信连接的存储器 11、处理器 12、网络接口 13，存储器 11 存储有可在处理器 12 上运行的处理系统。需要指出的是，图 1 仅示出了具有组件 11-13 的电

子装置 1，但是应理解的是，并不要求实施所有示出的组件，可以替代的实施更多或者更少的组件。

其中，存储器 11 包括内存及至少一种类型的可读存储介质。内存为电子装置 1 的运行提供缓存；可读存储介质可为如闪存、硬盘、多媒体卡、卡型存储器（例如，SD 或 DX 存储器等）、随机访问存储器（RAM）、静态随机访问存储器（SRAM）、只读存储器（ROM）、电可擦除可编程只读存储器（EEPROM）、可编程只读存储器（PROM）、磁性存储器、磁盘、光盘等的非易失性存储介质。在一些实施例中，可读存储介质可以是电子装置 1 的内部存储单元，例如该电子装置 1 的硬盘；在另一些实施例中，该非易失性存储介质也可以是电子装置 1 的外部存储设备，例如电子装置 1 上配备的插接式硬盘，智能存储卡（Smart Media Card, SMC），安全数字（Secure Digital, SD）卡，闪存卡（Flash Card）等。本实施例中，存储器 11 的可读存储介质通常用于存储安装于电子装置 1 的操作系统和各类应用软件，例如存储本申请一实施例中的处理系统的程序代码等。此外，存储器 11 还可以用于暂时地存储已经输出或者将要输出的各类数据。

所述处理器 12 在一些实施例中可以是中央处理器（Central Processing Unit, CPU）、控制器、微控制器、微处理器、或其他数据处理芯片。该处理器 12 通常用于控制所述电子装置 1 的总体操作，例如执行与其他设备进行数据交互或者通信相关的控制和处理等。本实施例中，所述处理器 12 用于运行所述存储器 11 中存储的程序代码或者处理数据，例如运行处理系统等。

所述网络接口 13 可包括无线网络接口或有线网络接口，该网络接口 13 通常用于在所述电子装置 1 与其他电子设备之间建立通信连接。

所述处理系统存储在存储器 11 中，包括至少一个存储在存储器 11 中的计算机可读指令，该至少一个计算机可读指令可被处理器 12 执行，以实现本申请各实施例的方法；以及，该至少一个计算机可读指令依据其各部分

所实现的功能不同，可被划为不同的逻辑模块。

本申请的电子装置中安装有大型数据库，例如 Oracle 数据库、mysql 数据库等等，数据库的数据原表中存有大量的数据，为了使得数据原表无大量的数据积压，对数据进行归档。在归档时，采用链式多线程的方式进行归档，链式指的是包括多步操作，每一步操作完成后方可执行下一步操作：在第一步操作中对数据原表进行分区并采用多个并行线程的方式将已完成的数据快速写入临时表，为了不打散数据，在第二步操作中用单线程进一步筛选出符合预定规则的数据，在第三步操作中对临时表进行分区并采用多个并行线程的方式快速对数据进行归档处理。本申请能够快速、高效地将数据归档，保证数据库系统中数据的稳定。

在一实施例中，上述处理系统被所述处理器 12 执行时实现如下步骤：

第一写入步骤，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

在一具体实施例中，可以将数据原表分为 32 个分区，利用 8 个并行线程并按照每个线程处理 4 个分区中的数据的方式，将各个分区中的已完成数据写入第一临时表中。在其他实施例中，可以将数据原表分为其他数量的分区，并利用其他个数的并行线程对分区的数据进行处理，只要分区的数量大于等于并行线程的数量即可。预定的第一处理方式为每一并行线程处理至少一个分区的数据，这样就不会有空闲的线程，为了更快速地将各个分区中的已完成数据写入预先建立的第一临时表中，优选地，每一并行线程处理相同数量的分区。

其中，对于数据原表中的每一条数据，其包括对应的流程 id、是否处理完成的标识、上级流程 id、分区号及其他字段等等，如下表 1 所示：

流程 id	是否完成	上级流程 ID	分区号	其他字 段
parentProcessId	是	空	1	。。。

表 1

在每一并行线程处理数据原表中的数据时，按序逐条取数据，然后获取该条数据中的是否处理完成的标识，若该是否处理完成的标识为完成标识，则可以将该条数据完整写入第一临时表中，若该是否处理完成的标识为未完成标识，则将该条数据保留在数据原表中。在上述表 1 中，对于流程 id 为“parentProcessId”的数据，其是否处理完成的标识为“是”，则这条数据为已完成数据，将其写入第一临时表中。

在一优选的实施例中，所述将数据库中的数据原表分为预设的第一数量的分区的步骤包括：建立分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该分区函数中，基于输入所述边界值及分区依据列的分区函数对所述数据原表进行分区。其中，分区函数是数据库中的一个独立对象，其将数据原表的行映射到一组分区，在创建分区函数时，指明数据分区的边界值以及分区依据列，执行该分区函数，以对数据原表进行分区。

第二写入步骤，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

当各个分区中的已完成数据全部写入所述第一临时表后，即上一步写入操作完成后，执行本步骤的写入操作。其中，数据库中的数据很多数据在流程上存在关联的关系，即存在上下级关联关系，若这些数据被打散处理则失去其具体意义，如果采用多线程处理的方式，则可能会将有关联的数据分到不同的线程进行处理，数据被打散。由于这些数据不能打散，因此本步骤的写入操作中利用单线程对第一临时表中的数据进行预处理，将所述具有上下

级关联的数据写入预先建立的第二临时表，能够保证有关联的数据不被打散。

在一实施例中，该第二写入步骤，具体包括：

利用单线程获取第一临时表中的各条数据的流程 id 标识，分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据；若存在父节点数据及与该父节点数据相关联的子节点数据，则获取该父节点数据的处理标识；若该父节点数据的处理标识为已完成标识，则将该条数据作为具有上下级关联的数据写入该第二临时表中。

在一具体的实例中，如图 2 所示，父节点数据例如为图 2 中的查勘、进入理算条件检查、理算、质检、支付下发。如图 3 所示，以“进入理算条件检查”作为父节点数据，则对应的子节点数据为进入理算检查规则、等待进入理算条件满足。

其中，在整个流程的数据中，每一父节点分配一父节点处理 ID (parentProcessId)，父节点下的子节点分配一子节点处理 ID (sonProcessId)，父节点下的数据与其对应的子节点下的数据为存在上下级关联关系的数据。

在一场景下产生的数据如下述表 2 所示：

流程 id	是否完成	上级流程 ID	分区号	其他字段
parentProcessId	是	空	1	。 。 。
sonProcessId	是	parentProcessId	1	。 。 。

表 2

表 2 这两条数据均为已完成的数据，经第一写入步骤进入第一临时表中，第二写入步骤对这两条数据进行分析，首先经流程 ID 获取到该数据，其存在父节点数据及子节点数据（即存在上下级关联关系的数据），经确认父节点数据为已完成的数据，则其子节点数据必然为已完成的数据，将这两条数据写入第二临时表中。

在另一场景下产生的数据如下述表 3 所示：

流程 id	是否完成	上级流程 ID	分区号	其他字段
parentProcessId	否	空	1	。 。 。
sonProcessId	是	parentProcessId	1	。 。 。

表 3

表 3 中的两条数据中子节点数据为已完成的数据，经第一写入步骤进入第一临时表中，父节点数据为未完成的数据，不能写入第一临时表中。第二写入步骤对该条子节点数据进行分析，发现该条数据为存在上下级关联关系的数据，但是对应的父节点数据不存在第一临时表中，其为未完成的数据，因此，这条子节点数据不能写入第二临时表中。

归档步骤，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

当利用单线程对所述第一临时表中的全部数据处理完成后，即上一步写入操作完成后，执行本步骤的归档操作。

在一具体实施例中，可以将第二临时表分为 32 个分区，利用 8 个并行线程并按照每个线程处理 4 个分区中的数据的方式，将第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中。在其他实施例中，可以将第二临时表分为其他数量的分区，并利用其他个数的并行线程对分区的数据进行处理，只要分区的数量大于等于并行线程的数量即可。预定的第二处理方式与预定的第一处理方式可以相同或不同，预定的第二处理方式为每一并行线程处理至少一个分区的数据，这样就不会有空闲的线程，为了更快速地将各个分区中的数据归档至归档表，优选地，每一并行线程处理相同数量的分区。

在一优选的实施例中，与上述的数据原表分区的原理相同，所述将所述第二临时表分为预设的第三数量的分区的步骤包括：建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

在一优选的实施例中，为了方便后续查询数据，以便能够快速查找到所需的数据，在将第二临时表中各个分区中的数据归档至该数据库中的归档表之后，还可以对归档表中的数据进行分类保存。

在一优选的实施例中，处理系统被所述处理器执行时，还实现如下步骤：当接收到对该数据库中的数据进行查询的指令时，根据所述查询指令查询该数据库中的数据原表，若在所述数据原表中未查询到对应的数据，则对该数据库中的归档表中的数据进行查询。

其中，传统方案是将数据库中的数据归档至归档库中，也就是另一数据库中，数据原表与归档表处于不同的数据库中，因此无法在查询数据原表后直接查询归档表。本实施例的归档表建立在与该数据原表相同的数据库中，由于数据原表与归档表均在该数据库中，因此在查询该数据库中的数据时，可以查询数据原表及归档表中的数据，首先查询数据库中的数据原表，若在数据原表中未查询到对应的数据，则再查询该数据库中的归档表，使得其支持归档表查询，提高归档表的可用性，提高服务性能。

与现有技术相比，本申请在对数据库中的数据归档时，采用链式多线程的方式进行归档，包括三步操作，每一步操作执行完成后方可执行下一步操作，在第一步操作中对数据原表进行分区并采用多个并行线程的方式将数据写入临时表，在第二步操作中用单线程筛选出符合条件的数据，在第三步操作中对临时表进行分区并采用多个并行线程的方式将数据归档至该数据库的归档表中，能够快速、高效地将数据库中的数据归档，保持数据库系统性

能的稳定性。

如图 4 所示,图 4 为本申请数据链式归档的方法一实施例的流程示意图,该数据链式归档的方法包括以下步骤:

步骤 S1,将数据库中的数据原表分为预设的第一数量的分区,利用预设的第二数量的并行线程并按照预定的第一处理方式,将各个分区中的已完成数据写入预先建立的第一临时表中,所述第一数量大于等于所述第二数量;

在一具体实施例中,可以将数据原表分为 32 个分区,利用 8 个并行线程并按照每个线程处理 4 个分区中的数据的方式,将各个分区中的已完成数据写入第一临时表中。在其他实施例中,可以将数据原表分为其他数量的分区,并利用其他个数的并行线程对分区的数据进行处理,只要分区的数量大于等于并行线程的数量即可。预定的第一处理方式为每一并行线程处理至少一个分区的数据,这样就不会有空闲的线程,为了更快速地将各个分区中的已完成数据写入预先建立的第一临时表中,优选地,每一并行线程处理相同数量的分区。

其中,对于数据原表中的每一条数据,其包括对应的流程 id、是否处理完成的标识、上级流程 id、分区号及其他字段等等,如上述表 1 所示,此处不再赘述。

在每一并行线程处理数据原表中的数据时,按序逐条取数据,然后获取该条数据中的是否处理完成的标识,若该是否处理完成的标识为完成标识,则可以将该条数据完整写入第一临时表中,若该是否处理完成的标识为未完成标识,则将该条数据保留在数据原表中。在上述表 1 中,对于流程 id 为“parentProcessId”的数据,其是否处理完成的标识为“是”,则这条数据为已完成数据,将其写入第一临时表中。

在一优选的实施例中，所述将数据库中的数据原表分为预设的第一数量的分区的步骤包括：建立分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该分区函数中，基于输入所述边界值及分区依据列的分区函数对所述数据原表进行分区。其中，分区函数是数据库中的一个独立对象，其将数据原表的行映射到一组分区，在创建分区函数时，指明数据分区的边界值以及分区依据列，执行该分区函数，以对数据原表进行分区。

步骤 S2，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

当各个分区中的已完成数据全部写入所述第一临时表后，即上一步写入操作完成后，执行本步骤的写入操作。其中，数据库中的数据很多数据在流程上存在关联的关系，即存在上下级关联关系，若这些数据被打散处理则失去其具体意义，如果采用多线程处理的方式，则可能会将有关联的数据分到不同的线程进行处理，数据被打散。由于这些数据不能打散，因此本步骤的写入操作中利用单线程对第一临时表中的数据进行预处理，将所述具有上下级关联的数据写入预先建立的第二临时表，能够保证有关联的数据不被打散。

在一实施例中，该步骤 S2，具体包括：

利用单线程获取第一临时表中的各条数据的流程 id 标识，分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据；若存在父节点数据及与该父节点数据相关联的子节点数据，则获取该父节点数据的处理标识；若该父节点数据的处理标识为已完成标识，则将该条数据作为具有上下级关联的数据写入该第二临时表中。

在一具体的实例中，如图 2 所示，父节点数据例如为图 2 中的查勘、进

入理算条件检查、理算、质检、支付下发。如图 3 所示，以“进入理算条件检查”作为父节点数据，则对应的子节点数据为进入理算检查规则、等待进入理算条件满足。

其中，在整个流程的数据中，每一父节点分配一父节点处理 ID (parentProcessId)，父节点下的子节点分配一子节点处理 ID (sonProcessId)，父节点下的数据与其对应的子节点下的数据为存在上下级关联关系的数据。

在一场景下产生的数据如上述表 2 所示，表 2 这两条数据均为已完成的数据，经步骤 S1 进入第一临时表中，步骤 S2 对这两条数据进行分析，首先经流程 ID 获取到该数据，其存在父节点数据及子节点数据（即存在上下级关联关系的数据），经确认父节点数据为已完成的数据，则其子节点数据必然为已完成的数据，将这两条数据写入第二临时表中。

在另一场景下产生的数据如上述表 3 所示，表 3 中的两条数据中子节点数据为已完成的数据，经步骤 S1 进入第一临时表中，父节点数据为未完成的数据，不能写入第一临时表中。步骤 S2 对该条子节点数据进行分析，发现该条数据为存在上下级关联关系的数据，但是对应的父节点数据不存在第一临时表中，其为未完成的数据，因此，这条子节点数据不能写入第二临时表中。

步骤 S3，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

当利用单线程对所述第一临时表中的全部数据处理完成后，即上一步写入操作完成后，执行本步骤的归档操作。

在一具体实施例中，可以将第二临时表分为 32 个分区，利用 8 个并行

线程并按照每个线程处理 4 个分区中的数据的方式，将第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中。在其他实施例中，可以将第二临时表分为其他数量的分区，并利用其他个数的并行线程对分区的数据进行处理，只要分区的数量大于等于并行线程的数量即可。预定的第二处理方式与预定的第一处理方式可以相同或不同，预定的第二处理方式为每一并行线程处理至少一个分区的数据，这样就不会有空闲的线程，为了更快速地将各个分区中的数据归档至归档表，优选地，每一并行线程处理相同数量的分区。

在一优选的实施例中，与上述的数据原表分区的原理相同，所述将所述第二临时表分为预设的第三数量的分区的步骤包括：建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

在一优选的实施例中，为了方便后续查询数据，以便能够快速查找到所需的数据，在将第二临时表中各个分区中的数据归档至该数据库中的归档表中之后，还可以对归档表中的数据进行分类保存。

在一优选的实施例中，处理系统被所述处理器执行时，还实现如下步骤：当接收到对该数据库中的数据进行查询的指令时，根据所述查询指令查询该数据库中的数据原表，若在所述数据原表中未查询到对应的数据，则对该数据库中的归档表中的数据进行查询。

其中，传统方案是将数据库中的数据归档至归档库中，也就是另一数据库中，数据原表与归档表处于不同的数据库中，因此无法在查询数据原表后直接查询归档表。本实施例的归档表建立在与该数据原表相同的数据库中，由于数据原表与归档表均在该数据库中，因此在查询该数据库中的数据时，可以查询数据原表及归档表中的数据，首先查询数据库中的数据原表，若在

数据原表中未查询到对应的数据，则再查询该数据库中的归档表，使得其支持归档表查询，提高归档表的可用性，提高服务性能。

与现有技术相比，本申请在对数据库中的数据归档时，采用链式多线程的方式进行归档，包括三步操作，每一步操作执行完成后方可执行下一步操作，在第一步操作中对数据原表进行分区并采用多个并行线程的方式将数据写入临时表，在第二步操作中用单线程筛选出符合条件的数据，在第三步操作中对临时表进行分区并采用多个并行线程的方式将数据归档至该数据库的归档表中，能够快速、高效地将数据库中的数据归档，保持数据库系统性能的稳定。

本申请还提供一种计算机可读存储介质，所述计算机可读存储介质上存储有处理系统，所述处理系统被处理器执行时实现上述的数据链式归档的方法的步骤。

上述本申请实施例序号仅仅为了描述，不代表实施例的优劣。

通过以上的实施方式的描述，本领域的技术人员可以清楚地了解到上述实施例方法可借助软件加必需的通用硬件平台的方式来实现，当然也可以通过硬件，但很多情况下前者是更佳的实施方式。基于这样的理解，本申请的技术方案本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质（如 ROM/RAM、磁碟、光盘）中，包括若干指令用以使得一台终端设备（可以是手机，计算机，服务器，空调器，或者网络设备等）执行本申请各个实施例所述的方法。

以上仅为本申请的优选实施例，并非因此限制本申请的专利范围，凡是利用本申请说明书及附图内容所作的等效结构或等效流程变换，或直接或间接运用在其他相关的技术领域，均同理包括在本申请的专利保护范围内。

权利要求书

1.一种电子装置，其特征在于，所述电子装置包括存储器及与所述存储器连接的处理器，所述存储器中存储有可在所述处理器上运行的处理系统，所述处理系统被所述处理器执行时实现如下步骤：

第一写入步骤，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

第二写入步骤，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

归档步骤，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

2.根据权利要求1所述的电子装置，其特征在于，所述处理系统被所述处理器执行时，还实现如下步骤：

当接收到对该数据库中的数据进行查询的指令时，根据所述查询指令查询该数据库中的数据原表，若在所述数据原表中未查询到对应的数据，则对该数据库中的归档表中的数据进行查询。

3.根据权利要求1或2所述的电子装置，其特征在于，所述第二写入步骤，具体包括：

利用单线程获取第一临时表中的各条数据的流程id标识，分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据；

若存在父节点数据及与该父节点数据相关联的子节点数据，则获取该父节点数据的处理标识；

若该父节点数据的处理标识为已完成标识，则将该条数据作为具有上下级关联的数据写入该第二临时表中。

4.根据权利要求 1 或 2 所述的电子装置，其特征在于，所述将数据库中的数据原表分为预设的第一数量的分区的步骤包括：

建立第一分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第一分区函数中，基于输入所述边界值及分区依据列的第一分区函数对所述数据原表进行分区；

所述将所述第二临时表分为预设的第三数量的分区的步骤包括：

建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

5.根据权利要求 1 或 2 所述的电子装置，其特征在于，所述处理系统被所述处理器执行时，还实现如下步骤：对归档表中的数据进行分类保存。

6.一种数据链式归档的方法，其特征在于，所述数据链式归档的方法包括：

S1，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

S2，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

S3，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并

按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表中的数据删除，所述第三数量大于等于所述第四数量。

7.根据权利要求 6 所述的数据链式归档的方法，其特征在于，所述步骤 S3 之后，还包括：

当接收到对该数据库中的数据进行查询的指令时，根据所述查询指令查询该数据库中的数据原表，若在所述数据原表中未查询到对应的数据，则对该数据库中的归档表中的数据进行查询。

8.根据权利要求 6 或 7 所述的数据链式归档的方法，其特征在于，所述步骤 S2，具体包括：

利用单线程获取第一临时表中的各条数据的流程 id 标识，分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据；

若存在父节点数据及与该父节点数据相关联的子节点数据，则获取该父节点数据的处理标识；

若该父节点数据的处理标识为已完成标识，则将该条数据作为具有上下级关联的数据写入该第二临时表中。

9.根据权利要求 6 或 7 所述的数据链式归档的方法，其特征在于，所述将数据库中的数据原表分为预设的第一数量的分区的步骤包括：

建立第一分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第一分区函数中，基于输入所述边界值及分区依据列的第一分区函数对所述数据原表进行分区；

所述将所述第二临时表分为预设的第三数量的分区的步骤包括：

建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

10.根据权利要求 6 或 7 所述的数据链式归档的方法,其特征在于,所述数据链式归档的方法,还包括:对归档表中的数据进行分类保存。

11.一种数据链式归档的系统,其特征在于,所述数据链式归档的系统包括:

第一写入模块,用于将数据库中的数据原表分为预设的第一数量的分区,利用预设的第二数量的并行线程并按照预定的第一处理方式,将各个分区中的已完成数据写入预先建立的第一临时表中,所述第一数量大于等于所述第二数量;

第二写入模块,用于当各个分区中的已完成数据全部写入所述第一临时表后,利用单线程获取所述第一临时表中具有上下级关联的数据,并将所述具有上下级关联的数据写入预先建立的第二临时表;

归档模块,用于当利用单线程对所述第一临时表中的全部数据处理完成后,将所述第二临时表分为预设的第三数量的分区,利用预设的第四数量的并行线程并按照预定的第二处理方式,将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中,并将数据原表中对应已写入该归档表中的数据删除,所述第三数量大于等于所述第四数量。

12.根据权利要求 11 所述的数据链式归档的系统,其特征在于,还包括查询模块,用于当接收到对该数据库中的数据进行查询的指令时,根据所述查询指令查询该数据库中的数据原表,若在所述数据原表中未查询到对应的数据,则对该数据库中的归档表中的数据进行查询。

13.根据权利要求 11 或 12 所述的数据链式归档的系统,其特征在于,所述第二写入模块,具体用于:利用单线程获取第一临时表中的各条数据的流程 id 标识,分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据;若存在父节点数据及与该父节点数据相关联的子节点数据,则获取该父节点数据的处理标识;若该父节点数据的处理标识为已完成标识,则

将该条数据作为具有上下级关联的数据写入该第二临时表中。

14.根据权利要求 11 或 12 所述的数据链式归档的系统，其特征在于，所述第一写入模块具体用于：建立第一分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第一分区函数中，基于输入所述边界值及分区依据列的第一分区函数对所述数据原表进行分区；

所述归档模块具体用于：建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

15.根据权利要求 11 或 12 所述的数据链式归档的系统，其特征在于，还包括分类模块，用于对归档表中的数据进行分类保存。

16.一种计算机可读存储介质，其特征在于，所述计算机可读存储介质上存储有处理系统，所述处理系统被处理器执行时实现步骤：

第一写入步骤，将数据库中的数据原表分为预设的第一数量的分区，利用预设的第二数量的并行线程并按照预定的第一处理方式，将各个分区中的已完成数据写入预先建立的第一临时表中，所述第一数量大于等于所述第二数量；

第二写入步骤，当各个分区中的已完成数据全部写入所述第一临时表后，利用单线程获取所述第一临时表中具有上下级关联的数据，并将所述具有上下级关联的数据写入预先建立的第二临时表；

归档步骤，当利用单线程对所述第一临时表中的全部数据处理完成后，将所述第二临时表分为预设的第三数量的分区，利用预设的第四数量的并行线程并按照预定的第二处理方式，将所述第二临时表中各个分区中的数据归档至该数据库中预先建立的归档表中，并将数据原表中对应已写入该归档表

中的数据删除，所述第三数量大于等于所述第四数量。

17.根据权利要求 16 所述的计算机可读存储介质，其特征在于，所述处理系统被所述处理器执行时，还实现如下步骤：

当接收到对该数据库中的数据进行查询的指令时，根据所述查询指令查询该数据库中的数据原表，若在所述数据原表中未查询到对应的数据，则对该数据库中的归档表中的数据进行查询。

18.根据权利要求 16 或 17 所述的计算机可读存储介质，其特征在于，所述第二写入步骤，具体包括：

利用单线程获取第一临时表中的各条数据的流程 id 标识，分析各条数据是否存在父节点数据及与该父节点数据相关联的子节点数据；

若存在父节点数据及与该父节点数据相关联的子节点数据，则获取该父节点数据的处理标识；

若该父节点数据的处理标识为已完成标识，则将该条数据作为具有上下级关联的数据写入该第二临时表中。

19.根据权利要求 16 或 17 所述的计算机可读存储介质，其特征在于，所述将数据库中的数据原表分为预设的第一数量的分区的步骤包括：

建立第一分区函数，基于所述第一数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第一分区函数中，基于输入所述边界值及分区依据列的第一分区函数对所述数据原表进行分区；

所述将所述第二临时表分为预设的第三数量的分区的步骤包括：

建立第二分区函数，基于所述第三数量确定分区的边界值及分区依据列，将所确定的边界值及分区依据列输入该第二分区函数中，基于输入所述边界值及分区依据列的第二分区函数对所述第二临时表进行分区。

20.根据权利要求 16 或 17 所述的计算机可读存储介质，其特征在于，所述处理系统被所述处理器执行时，还实现如下步骤：对归档表中的数据进行

分类保存。