(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 103631831 A
(43)申请公布日 2014.03.12

(21)申请号 201210312721.7

(22)申请日 2012.08.29

(71)申请人 阿里巴巴集团控股有限公司
地址 英属开曼群岛大开曼岛资本大厦一座四层 847 号邮箱

(72)发明人 钱在晨

(74)专利代理机构 北京同达信恒知识产权代理有限公司 11291
代理人 郭润湘

(51)Int.Cl.
G06F 17/30(2006.01)
G06F 11/14(2006.01)

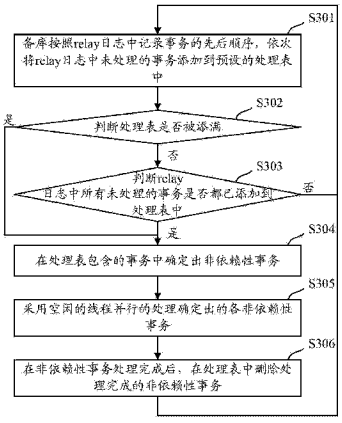
权利要求书3页 说明书11页 附图3页

(54)发明名称

一种数据备份方法及装置

(57)摘要

本申请公开了一种数据备份方法及装置，用以解决现有技术中浪费备库的系统资源的问题。该方法备库将 relay 日志中未处理的事务添加到处理表中，采用空闲的线程并行的对处理表中的各非依赖性事务进行处理，其中，针对处理表中的一个事务，如果处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则该事务为非依赖性事务。通过上述方法，备库对于互不依赖的各事务，并不根据各事务的特征值来确定用于处理各事务的线程，而是直接采用空闲的线程处理互不依赖的各事务，因此不会使线程长时间处于闲置状态，节省了备库的系统资源。



CN 103631831 A

1. 一种数据备份方法，其特征在于，包括：

备库按照 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到预设的处理表中；并

在所述处理表包含的事务中确定出非依赖性事务，其中，针对所述处理表中的一个事务，如果所述处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务；以及

采用空闲的线程并行的处理确定出的各非依赖性事务。

2. 如权利要求 1 所述的方法，其特征在于，依次将所述 relay 日志中未处理的事务添加到预设的处理表中，具体包括：

依次将所述 relay 日志中未处理的事务添加到预设的处理表中，直至添满所述处理表，或者将所述 relay 日志中所有未处理的事务都添加到所述处理表中为止；

采用空闲的线程并行的处理确定出的各非依赖性事务之后，所述方法还包括：

所述备库在非依赖性事务处理完成后，在所述处理表中删除处理完成的非依赖性事务，并继续按照所述 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到所述处理表中，基于更新后的处理表继续执行后续步骤。

3. 如权利要求 1 或 2 所述的方法，其特征在于，预设处理表具体包括：

预设 n 行 n 列的矩阵，n 为正整数；

将所述 relay 日志中未处理的事务添加到预设的处理表中，具体包括：

在预设的所述矩阵中选择满足指定条件的行和列，建立选择的行与所述未处理的事务的对应关系，建立选择的列与所述未处理的事务的对应关系，其中，满足指定条件的行和列为：行号与列号相等、且未与其他事务建立对应关系的行和列；并

针对所述矩阵中第 i 行第 j 列的元素 $E_{i,j}$，确定该元素 $E_{i,j}$ 所在的第 j 列对应的事务，确定该元素 $E_{i,j}$ 所在的第 i 行对应的事务，如果第 j 列对应的事务在所述 relay 日志中排在第 i 行对应的事务之前，且第 j 列对应的事务与第 i 行对应的事务冲突，则将该元素 $E_{i,j}$ 的值设置为第一数值，否则，将该元素 $E_{i,j}$ 的值设置为第二数值。

4. 如权利要求 3 所述的方法，其特征在于，在所述处理表包含的事务中确定出非依赖性事务，具体包括：

在所述矩阵中确定包含的所有元素的值均为第二数值的行所对应的事务，作为确定出的非依赖性事务。

5. 如权利要求 3 所述的方法，其特征在于，所述方法还包括：

针对位于所述矩阵的对角线上的元素 $E_{i,i}$，如果该元素 $E_{i,i}$ 所在的第 i 行和第 i 列已经建立了与未处理的事务的对应关系，则将该 $E_{i,i}$ 的值设置为所述第二数值，否则，将该元素 $E_{i,i}$ 的值设置为所述第一数值；

在预设的所述矩阵中选择满足指定条件的行和列，具体包括：

在位于所述矩阵的对角线上的元素中，选择值为所述第一数值的元素所在的行和列，作为选择的满足指定条件的行和列。

6. 如权利要求 5 所述的方法，其特征在于，在非依赖性事务处理完成后，在所述处理表中删除处理完成的非依赖性事务，具体包括：

在所述矩阵中确定处理完成的非依赖性事务对应的行和列，将确定的处理完成的非依

赖性事务对应的列中包含的所有元素的值都重新设置为第二数值，删除所述处理完成的非依赖性事务与确定的行和列的对应关系，并将所述矩阵中位于确定的行和列交叉位置处的元素的值设置为所述第一数值。

7. 如权利要求 6 所述的方法，其特征在于，所述方法还包括：

在删除所述处理完成的非依赖性事务与确定的行和列的对应关系时，对确定的列中包含的所有元素进行加锁处理，并在删除所述处理完成的非依赖性事务与确定的行和列的对应关系之后，对确定的列中包含的所有元素进行解锁处理。

8. 一种数据备份装置，其特征在于，包括：

添加模块，用于按照 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到预设的处理表中；

确定模块，用于在所述处理表包含的事务中确定出非依赖性事务，其中，针对所述处理表中的一个事务，如果所述处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务；

处理模块，用于采用空闲的线程并行的处理确定出的各非依赖性事务。

9. 如权利要求 8 所述的装置，其特征在于，所述添加模块具体用于，依次将所述 relay 日志中未处理的事务添加到预设的处理表中，直至添满所述处理表，或者将所述 relay 日志中所有未处理的事务都添加到所述处理表中为止；

所述添加模块还用于，在非依赖性事务处理完成后，在所述处理表中删除处理完成的非依赖性事务，并继续按照所述 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到所述处理表中，使所述确定模块和所述处理模块基于更新后的处理表继续执行后续步骤。

10. 如权利要求 8 或 9 所述的装置，其特征在于，所述添加模块包括：

预设单元，用于预设 n 行 n 列的矩阵作，其中，n 为正整数；

添加单元，用于在预设的所述矩阵中选择满足指定条件的行和列，建立选择的行与所述未处理事务的对应关系，建立选择的列与所述未处理事务的对应关系，其中，满足指定条件的行和列为：行号与列号相等、且未与其他事务建立对应关系的行和列；针对所述矩阵中第 i 行第 j 列的元素 $E_{i,j}$，确定该元素 $E_{i,j}$ 所在的第 j 列对应的事务，确定该元素 $E_{i,j}$ 所在的第 i 行对应的事务，如果第 j 列对应的事务在所述 relay 日志中排在第 i 行对应的事务之前，且第 j 列对应的事务与第 i 行对应的事务冲突，则将该元素 $E_{i,j}$ 的值设置为第一数值，否则，将该元素 $E_{i,j}$ 的值设置为第二数值。

11. 如权利要求 10 所述的装置，其特征在于，所述确定模块具体用于，在所述矩阵中确定包含的所有元素的值均为第二数值的行所对应的事务，作为确定出的非依赖性事务。

12. 如权利要求 10 所述的装置，其特征在于，所述添加单元还用于，针对位于所述矩阵的对角线上的元素 $E_{i,i}$，如果该元素 $E_{i,i}$ 所在的第 i 行和第 i 列已经建立了与未处理的事务的对应关系，则将该 $E_{i,i}$ 的值设置为所述第二数值，否则，将该元素 $E_{i,i}$ 的值设置为所述第一数值；在选择满足指定条件的行和列时，在位于所述矩阵的对角线上的元素中，选择值为所述第一数值的元素所在的行和列，作为选择的满足指定条件的行和列。

13. 如权利要求 12 所述的装置，其特征在于，所述添加模块还包括：

删除单元，用于在所述矩阵中确定处理完成的非依赖性事务对应的行和列，将确定的

处理完成的非依赖性事务对应的列中包含的所有元素的值都重新设置为第二数值，删除所述处理完成的非依赖性事务与确定的行和列的对应关系，并将所述矩阵中位于确定的行和列交叉位置处的元素的值设置为所述第一数值。

14. 如权利要求 13 所述的装置，其特征在于，所述添加模块还包括：

加解锁单元，用于在删除所述处理完成的非依赖性事务与确定的行和列的对应关系时，对确定的列中包含的所有元素进行加锁处理，并在删除所述处理完成的非依赖性事务与确定的行和列的对应关系之后，对确定的列中包含的所有元素进行解锁处理。

一种数据备份方法及装置

技术领域

[0001] 本申请涉及通信技术领域，尤其涉及一种数据备份方法及装置。

背景技术

[0002] MySQL 是一种小型关系型数据库管理系统。一般的，MySQL 包括主库和备库，主库用于提供数据管理和数据查询等功能，备库用于备份主库中的数据，用以在主库发生故障时代替主库提供相应的功能。

[0003] 图 1 为现有技术中 MySQL 中的主库将自身的数据备份到备库的过程，具体包括以下步骤：

[0004] S101：主库在对自身的数据进行操作时，生成对应的操作记录并记录在二进制日志(binlog)中。

[0005] 其中，主库对自身的数据的操作包括：更新操作、插入操作、删除操作。

[0006] 例如，主库将自身的数据 A 更新为数据 B 时，生成将该数据 A 更新为数据 B 的操作记录，并记录在 binlog 日志中。

[0007] S102：备库创建一个 I/O 线程，通过该 I/O 线程读取主库保存的 binlog 日志。

[0008] S103：备库将读取到的 binlog 日志转换成中继日志(relay)。

[0009] 其中，主库保存的 binlog 日志中记录的操作记录，与备库转换的 relay 日志中记录的操作记录相同，只是 binlog 日志与 relay 日志的格式不同。

[0010] 继续沿用上例，由于主库的 binlog 日志中记录有将数据 A 更新为数据 B 的操作记录，因此备库转换的 relay 日志中也包含将数据 A 更新为数据 B 的操作记录。

[0011] S104：备库创建一个 SQL 线程，通过创建的 SQL 线程依次读取 relay 日志中记录的每个操作记录。

[0012] 继续沿用上例，备库读取到的操作记录即为将数据 A 更新为数据 B 的操作记录。

[0013] S105：备库根据读取到的操作记录，将该操作记录对应的数据读取到内存中，并根据该操作记录对内存中的该数据进行相应操作。

[0014] 继续沿用上例，由于读取到的操作记录为将数据 A 更新为数据 B 的操作记录，因此该操作记录对应的数据即为备库中保存的数据 A。备库则将该数据 A 读取到内存中，再将该操作记录(将数据 A 更新为数据 B 的操作记录)解析为相应的执行语句并执行，用以将内存中的该数据 A 更新为数据 B，完成对主库数据的备份。

[0015] 由上述图 1 所示的过程可以看出，备库在备份主库中的数据时，是通过创建的一个 SQL 线程依次读取 relay 日志中的操作记录进行操作的，也即备库是通过一个 SQL 线程串行的执行 relay 日志中各操作记录对应的执行语句，实现数据的备份的。而在实际应用中，主库通常是通过多个线程并行的对保存的各数据进行操作的，例如主库可以同时通过几十个线程并行的对相应数量的数据进行操作，而备库在备份这些数据时，只能通过一个线程串行的备份这些数据，这就会导致备库对主库数据的备份速度远远落后于主库对自身保存的数据进行操作的速度。

[0016] 为了提高备库对主库数据进行备份的速度，现有技术中主要采用以下方法：

[0017] 按照 relay 日志中记录各待处理事务的先后顺序，依次针对 relay 日志中的每个待处理事务，判断当前正在处理的事务中是否存在与该待处理事务冲突的事务，若是，则将该待处理事务加入到与该待处理事务冲突的事务所在的 SQL 线程的等待队列中等待处理，否则，确定该待处理事务的特征值，根据该特征值确定处理该待处理事务的 SQL 线程，将该待处理事务加入到确定的 SQL 线程的等待队列中等待处理。

[0018] 其中，binlog 日志和 relay 日志均是以事务的形式对操作记录进行记录的，一个事务中包含若干个操作记录，这若干个操作记录一般是用户在进行一个业务操作时，数据库根据这个业务操作对自身保存的数据所要做出的若干个操作所对应的若干个操作记录。并且，在 binlog 日志和 relay 日志中，每个事务均具有一个事务开始标记和事务结束标记。两个冲突的事务即为：包含的操作记录对应的数据相同的两个事务。

[0019] 可见，现有技术中对于互不冲突的各事务，是根据各事务的特征值来确定用于并行处理各事务的 SQL 线程的，例如可以通过哈希（hash）算法确定各事务的特征值，并根据预设的特征值与 SQL 线程的对应关系，将互不冲突的各事务添加到相应 SQL 线程的等待队列中，以等待相应 SQL 线程的处理。

[0020] 然而，由于备库需要备份的数据是海量的，也即备库需要处理大量的事务，因此很难找到一种可以使大量互不冲突的事务能够被均匀的分配到有限数量的 SQL 线程上进行处理 hash 算法，这就会导致备库虽然预先建立了多个用于并行处理互不冲突的事务的 SQL 线程，但是有些 SQL 线程的等待队列中长时间存在很多待处理事务，而有些 SQL 线程的等待队列中则长时间没有待处理事务而处于闲置状态，这显然浪费了备库的系统资源。

发明内容

[0021] 本申请实施例提供一种数据备份方法及装置，用以解决现有技术中浪费备库的系统资源的问题。

[0022] 本申请实施例提供的一种数据备份方法，包括：

[0023] 备库按照 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到预设的处理表中；并

[0024] 在所述处理表包含的事务中确定出非依赖性事务，其中，针对所述处理表中的一个事务，如果所述处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务；以及

[0025] 采用空闲的线程并行的处理确定出的各非依赖性事务。

[0026] 本申请实施例提供的一种数据备份装置，包括：

[0027] 添加模块，用于按照 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到预设的处理表中；

[0028] 确定模块，用于在所述处理表包含的事务中确定出非依赖性事务，其中，针对所述处理表中的一个事务，如果所述处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务；

[0029] 处理模块，用于采用空闲的线程并行的处理确定出的各非依赖性事务。

[0030] 本申请实施例提供一种数据备份方法及装置，该方法备库将 relay 日志中未处理

的事务添加到处理表中，采用空闲的线程并行的对处理表中的各非依赖性事务进行处理，其中，针对处理表中的一个事务，如果处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则该事务为非依赖性事务。通过上述方法，备库对于互不依赖的各事务，并不根据各事务的特征值来确定用于处理各事务的线程，而是直接采用空闲的线程处理互不依赖的各事务，因此不会使线程长时间处于闲置状态，节省了备库的系统资源。

附图说明

[0031] 图 1 为现有技术中 MySQL 中的主库将自身的数据备份到备库的过程；

[0032] 图 2 为本申请实施例提供的数据备份的过程；

[0033] 图 3 为本申请实施例提供的数据备份的详细过程；

[0034] 图 4 为本申请实施例提供的在预设的矩阵中选择满足指定条件的行和列，并建立与未处理的事务的对应关系的示意图；

[0035] 图 5 为本申请实施例提供的数据备份装置结构示意图。

具体实施方式

[0036] 由于现有技术中通过 hash 算法来确定用于处理互不冲突的各事务的线程，因此不可避免的会出现某些线程的等待队列中长时间存在很多待处理事务，也即这些线程长时间处于繁忙状态，而另一些线程的等待队列中长时间没有任何待处理事务，也即这些线程长时间处于闲置状态，这显然浪费了备库为这些处于闲置状态的线程所分配的系统资源。本申请实施例为了节省备库的系统资源，将 relay 日志中记录的多个事务添加到处理表中，并直接采用空闲的线程对处理表中互不依赖的各事务进行并行处理，通过该方法，则不会出现某些线程长时间处理无事务可处理的闲置状态，充分利用了备库预先建立的多个线程，节省了备库的系统资源。

[0037] 下面结合说明书附图，对本申请实施例进行详细描述。

[0038] 图 2 为本申请实施例提供的数据备份的过程，具体包括以下步骤：

[0039] S201：备库按照 relay 日志中记录事务的先后顺序，依次将 relay 日志中未处理的事务添加到预设的处理表中。

[0040] 在本申请实施例中，备库预设一个处理表，该处理表中可以容纳有限数量的事务，例如可以容纳 n 个事务，n 为正整数。备库在读取主库的 binlog 日志，并将其转换为 relay 日志之后，则按照 relay 日志中记录事务的先后顺序，依次将 relay 日志中未处理的事务添加到该处理表中。

[0041] 例如，relay 日志中按照先后顺序依次记录了未处理的事务 1、事务 2、事务 3，则备库按照 relay 日志中记录事务的先后顺序，依次将事务 1、事务 2、事务 3 添加到处理表中。

[0042] S202：在处理表包含的事务中确定出非依赖性事务。

[0043] 其中，针对处理表中的一个事务，如果处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务。

[0044] 继续延用上例，当前添加到处理表中的事务为：事务 1、事务 2、事务 3。假设事务 1 中包含的两个操作记录分别对应的数据为数据 a 和数据 b，事务 2 中包含的两个操作记录分别对应的数据为数据 a 和数据 c，事务 3 中包含的两个操作记录分别对应的数据为数据 b

和数据 d，则：

[0045] 针对事务 1，由于事务 1 在 relay 日志中排在事务 2 和事务 3 之前，因此处理表中当前不存在在 relay 日志中排在事务 1 之前、且与事务 1 冲突的其他事务，因此事务 1 为非依赖性事务；

[0046] 针对事务 2，由于事务 2 在 relay 日志中排在事务 1 之后，因此，在当前的处理表中，存在在 relay 日志中排在事务 2 之前的事务 1。又因为事务 2 中包含了一个操作记录对应的数据为数据 a，事务 1 中也包含了一个操作记录对应的数据为数据 a，因此，事务 1 与事务 2 冲突。从而，在当前的处理表中，存在在 relay 日志中排在事务 2 之前、且与事务 2 冲突的事务 1，因此事务 2 为依赖性事务，也即事务 2 依赖于事务 1，需要在事务 1 处理完成之后再处理事务 2；

[0047] 针对事务 3，由于事务 3 在 relay 日志中排在事务 1 和事务 2 之后，因此，在当前的处理表中，存在在 relay 日志中排在事务 3 之前的事务 1 和事务 2。又因为事务 3 中包含了一个操作记录对应的数据为数据 b，事务 1 中也包含了一个操作记录对应的数据为数据 b，因此，事务 1 与事务 3 冲突。从而，在当前的处理表中，存在在 relay 日志中排在事务 3 之前、且与事务 3 冲突的事务 1，因此事务 3 为依赖性事务，也即事务 3 依赖于事务 1，需要在事务 1 处理完成之后再处理事务 3。

[0048] S203：采用空闲的线程并行的处理确定出的各非依赖性事务。

[0049] 其中，备库预先建立了多个 SQL 线程，当在处理表中确定出非依赖性事务时，则针对确定出的每个非依赖性事务，在预先建立的多个 SQL 线程中选择一个空闲的 SQL 线程处理该非依赖性事务，也即通过选择的该 SQL 线程对该非依赖性事务中包含的所有事务进行处理，完成数据的备份。当然，如果当前不存在空闲的线程，则可以等待至出现空闲的线程时，再采用空闲的线程对非依赖性事务进行处理。或者，当确定出的非依赖性事务的数量 x 大于空闲的线程的数量 y 时，可以先随机的从 x 个非依赖性事务中选择出 y 个非依赖性事务，并采用 y 个空闲的线程对选择出的非依赖性事务进行并行处理，对于剩余的 x-y 个非依赖性事务，则每次等待至出现空闲的线程时，随机从 x-y 个非依赖性事务中选择出一个，并采用空闲的线程进行处理。

[0050] 继续沿用上例，由于确定出的非依赖性事务为事务 1，因此备库采用一个空闲的 SQL 线程处理事务 1。

[0051] 由于上述方法备库在确定出处理表中的各非依赖性事务之后，并非是根据各非依赖性事务的特征值来确定用于处理各事务的线程，而是直接采用空闲的线程对确定出的各非依赖性事务进行处理，因此不会使备库预先建立的某些线程长时间处于闲置状态，可以充分利用备库预先建立的每个线程进行事务的处理，节省了备库的系统资源。

[0052] 进一步的，由于现有技术中仅仅是将未处理的事务与当前正在处理的事务进行对比，根据未处理的事务与当前正在处理的事务是否冲突，来决定是否对二者进行串行处理，因此在两个未处理的事务均与当前正在处理的事务冲突，但这两个未处理的事务并不冲突的情况下，现有技术中仍然会采用一个线程对这三个事务进行串行处理，从而降低了备库备份数据的速度。

[0053] 例如，备库当前正在通过线程 1 处理事务 1，事务 1 中包含的两个操作记录分别对应的数据为：数据 a 和数据 b。

[0054] 假设事务 2 中包含的两个操作记录分别对应的数据为 :数据 a 和数据 c,则事务 2 与事务 1 冲突(事务 2 与事务 1 均包含对应的数据为数据 a 的操作记录),因此,备库将事务 2 加入到线程 1 的等待队列中等待处理。

[0055] 假设事务 3 中包含的两个操作记录分别对应的数据为 :数据 b 和数据 d,则事务 3 与事务 1 也冲突(事务 3 与事务 1 均包含对应的数据为数据 b 的操作记录),因此,备库将事务 3 加入到线程 1 的等待队列中等待处理。

[0056] 此时,线程 1 的等待队列中就存在两个等待处理的事务,即为事务 2 和事务 3,也即,事务 1、事务 2、事务 3 是通过线程 1 串行处理的。

[0057] 然而,在上例中,由于事务 2 中包含的两个操作记录分别对应数据 a 和数据 c,事务 3 中包含的两个操作记录分别对应数据 b 和数据 d,因此,虽然事务 2 和事务 3 均与事务 1 冲突,但是事务 2 与事务 3 并不冲突,而现有技术中通过一个 SQL 线程串行处理这两个不冲突的事务,显然也会降低备库备份数据的速度。

[0058] 本申请实施例为了提高备库对主库数据进行备份的速度,将 relay 日志中记录的多个事务添加到处理表中后,实时分析处理表中添加的所有事务之间的依赖关系,也即不仅要分析未处理的事务与当前正在处理的事务之间的依赖关系,还要分析处理表中各个未处理的事务之间的依赖关系,并对非依赖性事务进行并行处理,因此,在两个未处理的事务均与当前正在处理的事务冲突,但这两个未处理的事务并不冲突的情况下,如果该当前正在处理的事务处理完毕,则两个不冲突的未处理事务之间没有依赖关系,备库则可以采用两个线程并行的处理这两个事务,因此提高了备库备份数据的速度。

[0059] 图 3 为本申请实施例提供的数据备份的详细过程,具体包括以下步骤 :

[0060] S301 :备库按照 relay 日志中记录事务的先后顺序,依次将 relay 日志中未处理的事务添加到预设的处理表中。

[0061] 例如,relay 日志中按照先后顺序依次记录了未处理的事务 1、事务 2、事务 3,则备库按照 relay 日志中记录事务的先后顺序,依次将事务 1、事务 2、事务 3 添加到处理表中。

[0062] S302 :判断处理表是否被添满,若是,则执行步骤 S304,否则执行步骤 S303。

[0063] S303 :判断 relay 日志中所有未处理的事务是否都已添加到处理表中,若是,则执行步骤 S304,否则返回步骤 S301。

[0064] 由于预设的该处理表中可以容纳的事务的数量是有限的,因此备库每次向该处理表中添加一个事务,就判断该处理表是否已经添满,如果被添满,则执行后续步骤 S304,如果尚未被添满,则继续判断 relay 日志中是否还有未添加到处理表中的未处理的事务,若存在,则继续将 relay 日志中未处理的事务添加到处理表中,也即返回步骤 S301,若不存在,则执行后续步骤 S304。

[0065] S304 :在处理表包含的事务中确定出非依赖性事务。

[0066] 其中,针对处理表中的一个事务,如果处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务,则确定该事务为非依赖性事务。

[0067] 继续延用上例,当前添加到处理表中的事务为 :事务 1、事务 2、事务 3。假设事务 1 中包含的两个操作记录分别对应的数据为数据 a 和数据 b,事务 2 中包含的两个操作记录分别对应的数据为数据 a 和数据 c,事务 3 中包含的两个操作记录分别对应的数据为数据 b 和数据 d,则 :

[0068] 针对事务 1，由于事务 1 在 relay 日志中排在事务 2 和事务 3 之前，因此处理表中当前不存在在 relay 日志中排在事务 1 之前、且与事务 1 冲突的其他事务，因此事务 1 为非依赖性事务；

[0069] 针对事务 2，由于事务 2 在 relay 日志中排在事务 1 之后，因此，在当前的处理表中，存在在 relay 日志中排在事务 2 之前的事务 1。又因为事务 2 中包含了一个操作记录对应的数据为数据 a，事务 1 中也包含了一个操作记录对应的数据为数据 a，因此，事务 1 与事务 2 冲突。从而，在当前的处理表中，存在在 relay 日志中排在事务 2 之前、且与事务 2 冲突的事务 1，因此事务 2 为依赖性事务，也即事务 2 依赖于事务 1，需要在事务 1 处理完成之后再处理事务 2；

[0070] 针对事务 3，由于事务 3 在 relay 日志中排在事务 1 和事务 2 之后，因此，在当前的处理表中，存在在 relay 日志中排在事务 3 之前的事务 1 和事务 2。又因为事务 3 中包含了一个操作记录对应的数据为数据 b，事务 1 中也包含了一个操作记录对应的数据为数据 b，因此，事务 1 与事务 3 冲突。从而，在当前的处理表中，存在在 relay 日志中排在事务 3 之前、且与事务 3 冲突的事务 1，因此事务 3 为依赖性事务，也即事务 3 依赖于事务 1，需要在事务 1 处理完成之后再处理事务 3。

[0071] S305：采用空闲的线程并行的处理确定出的各非依赖性事务。

[0072] 继续沿用上例，由于确定出的非依赖性事务为事务 1，因此备库采用一个空闲的 SQL 线程处理事务 1。

[0073] S306：在非依赖性事务处理完成后，在处理表中删除处理完成的非依赖性事务，并返回步骤 S301。

[0074] 也即，在处理完非依赖性事务之后，在处理表中删除处理完的该非依赖性事务，并继续按照 relay 日志中记录事务的先后顺序，依次将 relay 日志中未处理的事务添加到处理表中，并基于更新后的处理表继续执行后续步骤，直至将 relay 日志中所有未处理的事务都处理完成为止。

[0075] 继续延用上例，当事务 1 处理完成之后，在处理表中删除事务 1，并返回步骤 S301。假设此时 relay 日志中已经不存在未添加到处理表中的未处理的事务，则此时更新后的处理表中就只包括事务 2 和事务 3。而由于事务 2 中包含的两个操作记录分别对应数据 a 和数据 c，事务 3 中包含的两个操作记录分别对应数据 b 和数据 d，因此这两个事务并不冲突，通过步骤 S304 则可以确定事务 2 和事务 3 均为非依赖性事务，从而，在步骤 S305 中，备库则可以通过两个 SQL 线程并行的处理事务 2 和事务 3。

[0076] 由上例可见，由于本申请实施例中删除处理表中已经处理完成的事务，并实时的分析处理表中各事务之间的依赖关系，对非依赖性事务进行处理，因此上例中备库处理事务 1、事务 2、事务 3 的过程为：先处理事务 1，再同时并行处理事务 2 和事务 3，相比与现有技术中串行的处理事务 1、事务 2、事务 3 的方法，本申请实施例提供的上述数据备份方法提高了备库备份数据的速度。

[0077] 在本申请实施例中，备库预设一个矩阵作为预设的处理表，具体的，备库预设 n 行 n 列的矩阵作为预设的处理表，n 为正整数。

[0078] 当备库以矩阵的形式预设处理表时，则图 3 所示的步骤 S301 中备库将 relay 日志中未处理的事务添加到处理表中的具体方法为，在预设的该矩阵中选择满足指定条件的行

和列，建立选择的行与该未处理的事务的对应关系，建立选择的列与该未处理的事务的对应关系，其中，满足指定条件的行和列为：行号与列号相等、且未与其他事务建立对应关系的行和列；针对该矩阵中第 i 行第 j 列的元素 $E_{i,j}$，确定该元素 $E_{i,j}$ 所在的第 j 列对应的事务，确定该元素 $E_{i,j}$ 所在的第 i 行对应的事务，如果第 j 列对应的事务在 relay 日志中排在第 i 行对应的事务之前，且第 j 列对应的事务与第 i 行对应的事务冲突，则将该元素 $E_{i,j}$ 的值设置为第一数值，否则，将该元素 $E_{i,j}$ 的值设置为第二数值。其中，预设的该矩阵的行和列的数量 n 不小于备库预先建立的 SQL 线程的数量。

[0079] 继续以上例中的事务 1、事务 2、事务 3 为例进行说明。假设预设的矩阵为 3 行 3 列的矩阵 $M_{3\times3}=\begin{bmatrix} E_{1,1} & E_{1,2} & E_{1,3} \\ E_{2,1} & E_{2,2} & E_{2,3} \\ E_{3,1} & E_{3,2} & E_{3,3} \end{bmatrix}$，当前该矩阵 $M_{3\times3}$ 中没有任何一行和任何一列与其他事务建立对应关系，则该矩阵 $M_{3\times3}$ 中满足指定条件的行和列为：第 1 行和第 1 列、第 2 行和第 2 列、第 3 行和第 3 列。

[0080] 针对事务 1，选择满足指定条件的第 1 行和第 1 列，建立第 1 行与事务 1 的对应关系，建立第 1 列与事务 1 的对应关系。

[0081] 针对事务 2，选择满足指定条件的第 2 行和第 2 列，建立第 2 行与事务 2 的对应关系，建立第 2 列与事务 2 的对应关系。

[0082] 针对事务 3，选择满足指定条件的第 3 行和第 3 列，建立第 3 行与事务 3 的对应关系，建立第 3 列与事务 3 的对应关系。

[0083] 建立了对应关系后的矩阵如图 4 所示。图 4 为本申请实施例提供的在预设的矩阵中选择满足指定条件的行和列，并建立与未处理的事务的对应关系的示意图。

[0084] 在图 4 中，第 1 行和第 1 列均与事务 1 对应，第 2 行和第 2 列均与事务 2 对应，第 3 行和第 3 列均与事务 3 对应，因此，该预设的矩阵中可以容纳的未处理的事务的数量就是该矩阵的行或列的数量。

[0085] 假设第一数值设定为 1，第二数值设定为 0，则在建立了如图 4 所示的对应关系后，针对矩阵 $M_{3\times3}$ 中第 1 行第 1 列的元素 $E_{1,1}$，该元素 $E_{1,1}$ 所在的第 1 列对应事务 1，该元素 $E_{1,1}$ 所在的第 1 行也对应事务 1，第 1 列对应的事务（事务 1）在 relay 日志中的排序与第 1 行对应的事务（事务 1）相同，因此将该元素 $E_{1,1}$ 的值设置为 0（第二数值）。

[0086] 针对矩阵 $M_{3\times3}$ 中第 1 行第 2 列的元素 $E_{1,2}$，该元素 $E_{1,2}$ 所在的第 2 列对应事务 2，该元素 $E_{1,2}$ 所在的第 1 行对应事务 1，第 2 列对应的事务 2 在 relay 日志中排在第 1 行对应的事务 1 之后，因此将该元素 $E_{1,2}$ 的值也设置为 0（第二数值）。

[0087] 针对矩阵 $M_{3\times3}$ 中第 2 行第 1 列的元素 $E_{2,1}$，该元素 $E_{2,1}$ 所在的第 1 列对应事务 1，该元素 $E_{2,1}$ 所在的第 2 行对应事务 2，第 1 列对应的事务 1 在 relay 日志中排在第 2 行对应的事务 2 之前，且第 1 列对应的事务 1 与第 2 行对应的事务 2 冲突，因此将该元素 $E_{2,1}$ 的值设置为 1（第一数值）。

[0088] 类似的，矩阵 $M_{3\times3}$ 中的其他元素也按照上述方法进行赋值，赋值后的矩阵即为 $M_{3\times3}=\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$。

[0089] 也即，在本申请实施例中，预设的矩阵中的第 i 行第 j 列的元素 $E_{i,j}$ 的值如果为第一数值，则表示第 i 行对应的事务在 relay 日志中排在第 j 行对应的事务之后、且二者冲突，说明第 i 行对应的事务依赖于第 j 行对应的事务，进一步说明第 i 行对应的事务需要在第 j 行对应的事务处理完成之后再进行处理；相反的，元素 $E_{i,j}$ 的值如果为第二数值，则表示第 i 行对应的事务在 relay 日志中未排在第 j 行对应的事务之后，或者二者并不冲突，说明第 i 行对应的事务并不依赖于第 j 行对应的事务，进一步说明第 i 行对应的事务不需要在第 j 行对应的事务处理完成之后再进行处理。

[0090] 采用上述方法将未处理的事务添加到处理表中时，图 3 所示的步骤 S304 中备库在处理表包含的事务中确定出非依赖性事务的方法可以为：在矩阵中确定包含的所有元素的值均为第二数值的行所对应的事务，作为确定出的非依赖性事务。这是由于对于矩阵中的任一元素 $E_{i,j}$，如果 $E_{i,j}$ 的值为第二数值，则说明第 i 行对应的事务并不依赖于第 j 行对应的事务，进而，如果第 i 行包含的所有元素 $E_{i,1}$、$E_{i,2}$、……$E_{i,n}$ 均为第二数值，则说明第 i 行对应的事务不依赖于当前矩阵（处理表）中添加的任何其他事务，因此第 i 行对应的事务即为非依赖性事务。

[0091] 另外，在本申请实施例中，针对位于矩阵的对角线上的元素 $E_{i,i}$，如果该元素 $E_{i,i}$ 所在的第 i 行和第 i 列已经建立了与未处理的事务的对应关系，则将该 $E_{i,i}$ 的值设置为所述第二数值，否则，将该元素 $E_{i,i}$ 的值设置为所述第一数值。

[0092] 采用上述方法设置矩阵中位于对角线上的元素的值时，则备库在该矩阵中选择满足指定条件的行和列的方法具体可以为：在位于该矩阵的对角线上的元素中，选择值为第一数值的元素所在的行和列，作为选择的满足指定条件的行和列。也即，由于该矩阵对角线上的元素所在的行的行号等于该元素所在的列的列号，并且如果该对角线上的元素的值为第一数值，则说明该元素所在的行和列均为与其他事务建立对应关系，因此可以选择该对角线上的元素所在的行和列，并建立与 relay 日志中未处理的事务的对应关系。当然，建立对应关系后，也要将该对角线上的元素的值重新设置为第二数值。

[0093] 进一步的，当以上述矩阵的形式设置处理表时，备库在对确定出的非依赖性事务处理完成之后，在处理表中删除处理完成的非依赖性事务的方法具体为：在矩阵中确定处理完成的非依赖性事务对应的行和列，将确定的处理完成的非依赖性事务对应的列中包含的所有元素的值都重新设置为第二数值，删除该处理完成的非依赖性事务与确定的行和列的对应关系，并将该矩阵中位于确定的行和列交叉位置出的元素的值设置为第一数值。

[0094] 继续沿用上例，由矩阵 $M_{3\times3}=\begin{bmatrix}0&0&0\\1&0&0\\1&0&0\end{bmatrix}$ 可见，第 1 行包含的所有元素的值均为第二数值 0，因此该矩阵 $M_{3\times3}$ 的第 1 行对应的事务 1 为非依赖性事务，则备库采用一个空闲的 SQL 线程处理该事务 1，事务 2 和事务 3 等待后续处理。

[0095] 当备库对该事务 1 处理完成后，则备库首先确定该事务 1 对应的该矩阵中的第 1 行和第 1 列，将确定的第 1 列中包含的所有元素的值都重新设置为 0（第二数值）。这是由于此时事务 1 已经处理完成，之前（事务 1 尚未处理完成之前）依赖于该事务 1 的其他事务（事务 2 和事务 3）此时已经不再依赖于事务 1，也即，之前需要等待事务 1 处理完成之后再进行处理的其他事务此时已经可以处理了，因此解除其他事务对事务 1 的依赖关系，将事务 1

对应的第 1 列中包含的所有元素的值重置为第二数值。将第 1 列中包含的所有元素的值重置为第二数值的矩阵$M_{3\times3}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}$，可见，此时第 2 行第 1 列以及第 3 行第 1 列的元素的值均为 0，说明此时事务 2 和事务 3 已经可以处理，不再依赖于事务 1。

[0096] 然后，备库删除该事务 1 与矩阵 $M_{3\times3}$ 的第 1 行和第 1 列的对应关系，并位于将第 1 行和第 1 列交叉位置处的元素设置为 1（第一数值），也即，将矩阵 $M_{3\times3}$ 中第 1 行第 1 列的元素的值设置为第一数值，亦即，将第 1 行（或第 1 列）中位于对角线上的元素的值设置为第一数值，表示此时矩阵 $M_{3\times3}$ 中的第 1 行和第 1 列均未与任何事务建立对应关系，此时的矩阵 $M_{3\times3}=\begin{bmatrix}1&0&0\\0&0&0\\0&0&0\end{bmatrix}$。

[0097] 如果 relay 日志中还存在未处理的事务 4，则备库继续建立事务 4 与矩阵 $M_{3\times3}$ 中的第 1 行和第 1 列的对应关系，将 $M_{3\times3}$ 中的第 1 行第 1 列的元素的值设置为第二数值（表示第 1 行和第 1 列均已经与事务建立了对应关系），并根据事务 4 与事务 2 和事务 3 在 relay 日志中的排序以及冲突关系，对矩阵 $M_{3\times3}$ 中的每个元素的值进行设置。实质上，建立矩阵 $M_{3\times3}$ 中第 1 行和第 1 列与事务 4 的对应关系之后，由于事务 4 在 relay 日志中排在事务 2 和事务 3 之后，因此事务 2 和事务 3 均不依赖于事务 4，从而矩阵 $M_{3\times3}$ 中第 2 行第 1 列以及第 3 行第 1 列元素的值仍然为第二数值，因此只需要根据事务 4 与事务 2 和事务 3 的冲突关系，重新调整该事务 4 所在的第 1 行的元素的值即可。

[0098] 由上例可以看出，无论是否在矩阵中添加事务 4，矩阵中事务 2 对应的第 2 行包含的所有元素的值均为 0（第二数值），事务 3 对应的第 3 行包含的所有元素的值也均为 0（第二数值），因此，此时备库则可以在预先建立的 SQL 线程中，选择两个空闲的 SQL 线程，通过这两个空闲的 SQL 线程并行的对事务 2 和事务 3 进行处理，相比于现有技术对事务 1、事务 2、事务 3 进行串行处理的方法，本申请实施例提供的上述方法有效的提高了备库备份数据的速度。

[0099] 在本申请实施例中，备库可以建立一个矩阵管理线程，通过该矩阵管理线程建立预设的矩阵中的行和列与 relay 日志中未处理的事务的对应关系，并通过该矩阵管理线程实时的更新矩阵中的各个元素的值，也可以通过该矩阵管理线程删除处理完成的非依赖性事务与矩阵中相应的行和列的对应关系。此时，为了避免矩阵管理线程与 SQL 线程对于矩阵中的元素的调整发生冲突，备库在通过矩阵管理线程删除处理完成的非依赖性事务与相应的行和列的对应关系时，可以对该列中包含的所有元素进行加锁处理，并在删除该处理完成的非依赖性事务与相应的行和列的对应关系之后，再对该列中包含的所有元素进行解锁处理。其中，当矩阵管理线程对矩阵中的某一列中所有元素进行加锁处理时，则只有该矩阵管理线程可以对该列中的元素进行调整，其他线程均不可以对该列中的元素进行调整。

[0100] 另外，在本申请实施例中，备库在以矩阵的形式预设处理表时，具体可以通过预设 n 行 n 列的位（bit）矩阵作为预设的处理表。

[0101] 在本申请实施例中，冲突的两个事务除了可以是分别包含的操作记录对应的数据相同的两个事务之外，还可以是分别包含的操作记录对应的数据的主键相同、或者数据所

属的表相同、或者数据所属的子库相同的两个事务，这里就不再一一赘述。

[0102] 图 5 为本申请实施例提供的数据备份装置结构示意图，具体包括：

[0103] 添加模块 501，用于按照 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到预设的处理表中；

[0104] 确定模块 502，用于在所述处理表包含的事务中确定出非依赖性事务，其中，针对所述处理表中的一个事务，如果所述处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则确定该事务为非依赖性事务；

[0105] 处理模块 503，用于采用空闲的线程并行的处理确定出的各非依赖性事务。

[0106] 所述添加模块 501 具体用于，依次将所述 relay 日志中未处理的事务添加到预设的处理表中，直至添满所述处理表，或者将所述 relay 日志中所有未处理的事务都添加到所述处理表中为止；

[0107] 所述添加模块 501 还用于，在非依赖性事务处理完成后，在所述处理表中删除处理完成的非依赖性事务，并继续按照所述 relay 日志中记录事务的先后顺序，依次将所述 relay 日志中未处理的事务添加到所述处理表中，使所述确定模块 502 和所述处理模块 503 基于更新后的处理表继续执行后续步骤。

[0108] 所述添加模块 501 包括：

[0109] 预设单元 5011，用于预设 n 行 n 列的矩阵作，其中，n 为正整数；

[0110] 添加单元 5012，用于在预设的所述矩阵中选择满足指定条件的行和列，建立选择的行与所述未处理事务的对应关系，建立选择的列与所述未处理事务的对应关系，其中，满足指定条件的行和列为：行号与列号相等、且未与其他事务建立对应关系的行和列；针对所述矩阵中第 i 行第 j 列的元素 $E_{i,j}$，确定该元素 $E_{i,j}$ 所在的第 j 列对应的事务，确定该元素 $E_{i,j}$ 所在的第 i 行对应的事务，如果第 j 列对应的事务在所述 relay 日志中排在第 i 行对应的事务之前，且第 j 列对应的事务与第 i 行对应的事务冲突，则将该元素 $E_{i,j}$ 的值设置为第一数值，否则，将该元素 $E_{i,j}$ 的值设置为第二数值。

[0111] 所述确定模块 502 具体用于，在所述矩阵中确定包含的所有元素的值均为第二数值的行所对应的事务，作为确定出的非依赖性事务。

[0112] 所述添加单元 5012 还用于，针对位于所述矩阵的对角线上的元素 $E_{i,i}$，如果该元素 $E_{i,i}$ 所在的第 i 行和第 i 列已经建立了与未处理的事务的对应关系，则将该 $E_{i,i}$ 的值设置为所述第二数值，否则，将该元素 $E_{i,i}$ 的值设置为所述第一数值；在选择满足指定条件的行和列时，在位于所述矩阵的对角线上的元素中，选择值为所述第一数值的元素所在的行和列，作为选择的满足指定条件的行和列。

[0113] 所述添加模块 501 还包括：

[0114] 删除单元 5013，用于在所述矩阵中确定处理完成的非依赖性事务对应的行和列，将确定的处理完成的非依赖性事务对应的列中包含的所有元素的值都重新设置为第二数值，删除所述处理完成的非依赖性事务与确定的行和列的对应关系，并将所述矩阵中位于确定的行和列交叉位置处的元素的值设置为所述第一数值。

[0115] 所述添加模块 501 还包括：

[0116] 加解锁单元 5014，用于在删除所述处理完成的非依赖性事务与确定的行和列的对应关系时，对确定的列中包含的所有元素进行加锁处理，并在删除所述处理完成的非依赖

性事务与确定的行和列的对应关系之后，对确定的列中包含的所有元素进行解锁处理。

[0117] 具体的上述数据备份装置可以位于备库中。

[0118] 本申请实施例提供一种数据备份方法及装置，该方法备库将 relay 日志中未处理的事务添加到处理表中，采用空闲的线程并行的对处理表中的各非依赖性事务进行处理，其中，针对处理表中的一个事务，如果处理表中不存在在 relay 日志中排在该事务之前、且与该事务冲突的其他事务，则该事务为非依赖性事务。通过上述方法，备库对于互不依赖的各事务，并不根据各事务的特征值来确定用于处理各事务的线程，而是直接采用空闲的线程处理互不依赖的各事务，因此不会使线程长时间处于闲置状态，节省了备库的系统资源。

[0119] 显然，本领域的技术人员可以对本申请进行各种改动和变型而不脱离本申请的精神和范围。这样，倘若本申请的这些修改和变型属于本申请权利要求及其等同技术的范围之内，则本申请也意图包含这些改动和变型在内。

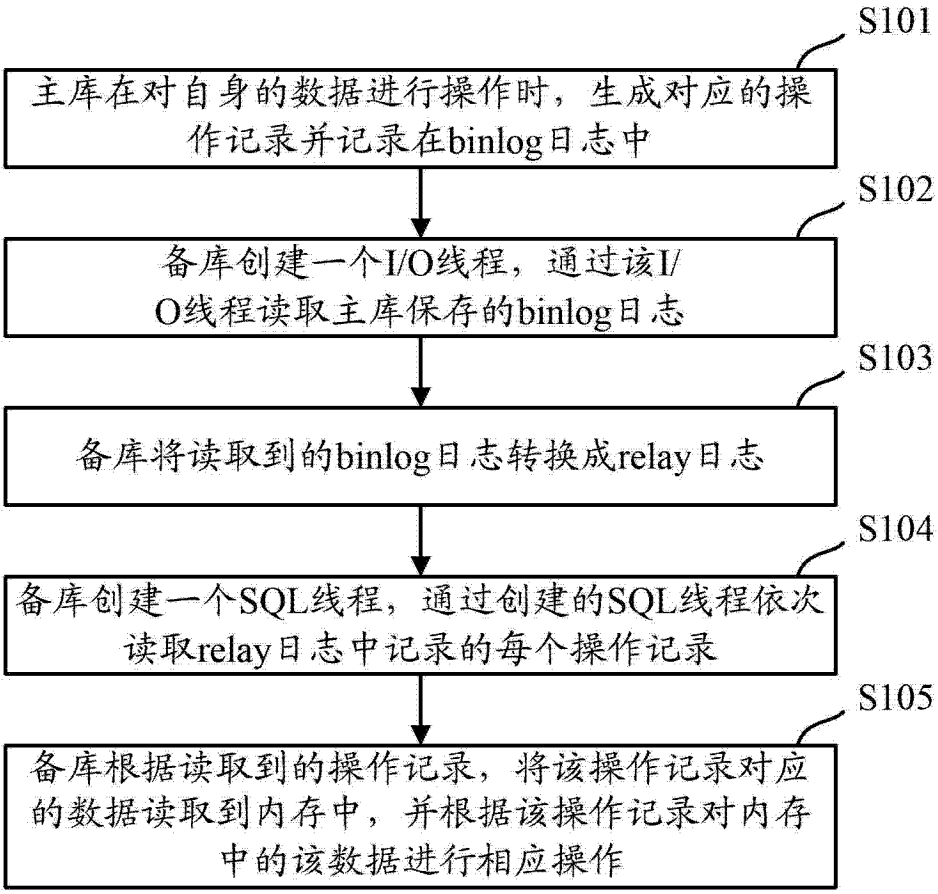

S101
主库在对自身的数据进行操作时，生成对应的操作记录并记录在binlog日志中
S102
备库创建一个I/O线程，通过该I/O线程读取主库保存的binlog日志
S103
备库将读取到的binlog日志转换成relay日志
S104
备库创建一个SQL线程，通过创建的SQL线程依次读取relay日志中记录的每个操作记录
S105
备库根据读取到的操作记录，将该操作记录对应的数据读取到内存中，并根据该操作记录对内存中的该数据进行相应操作

图 1

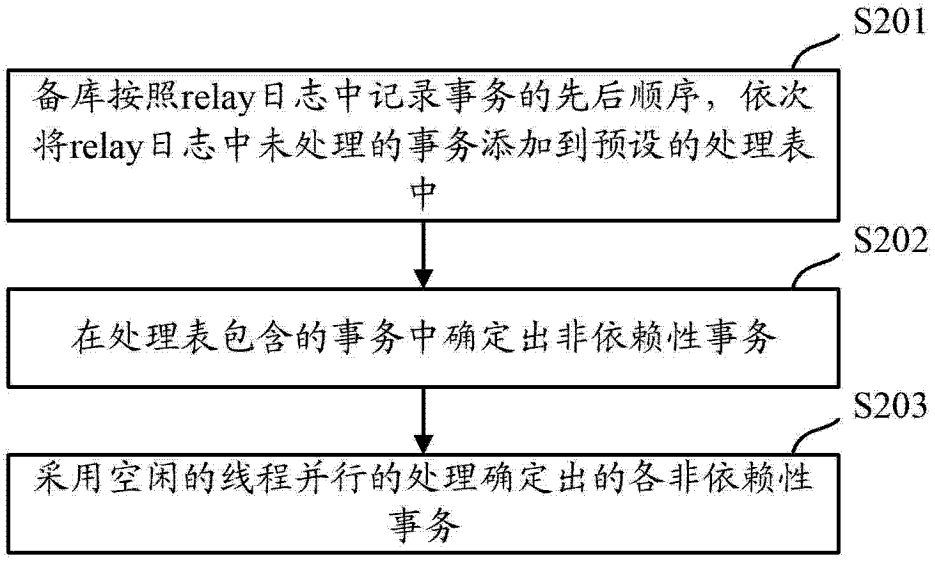

S201
备库按照relay日志中记录事务的先后顺序，依次将relay日志中未处理的事务添加到预设的处理表中
S202
在处理表包含的事务中确定出非依赖性事务
S203
采用空闲的线程并行的处理确定出的各非依赖性事务

图 2

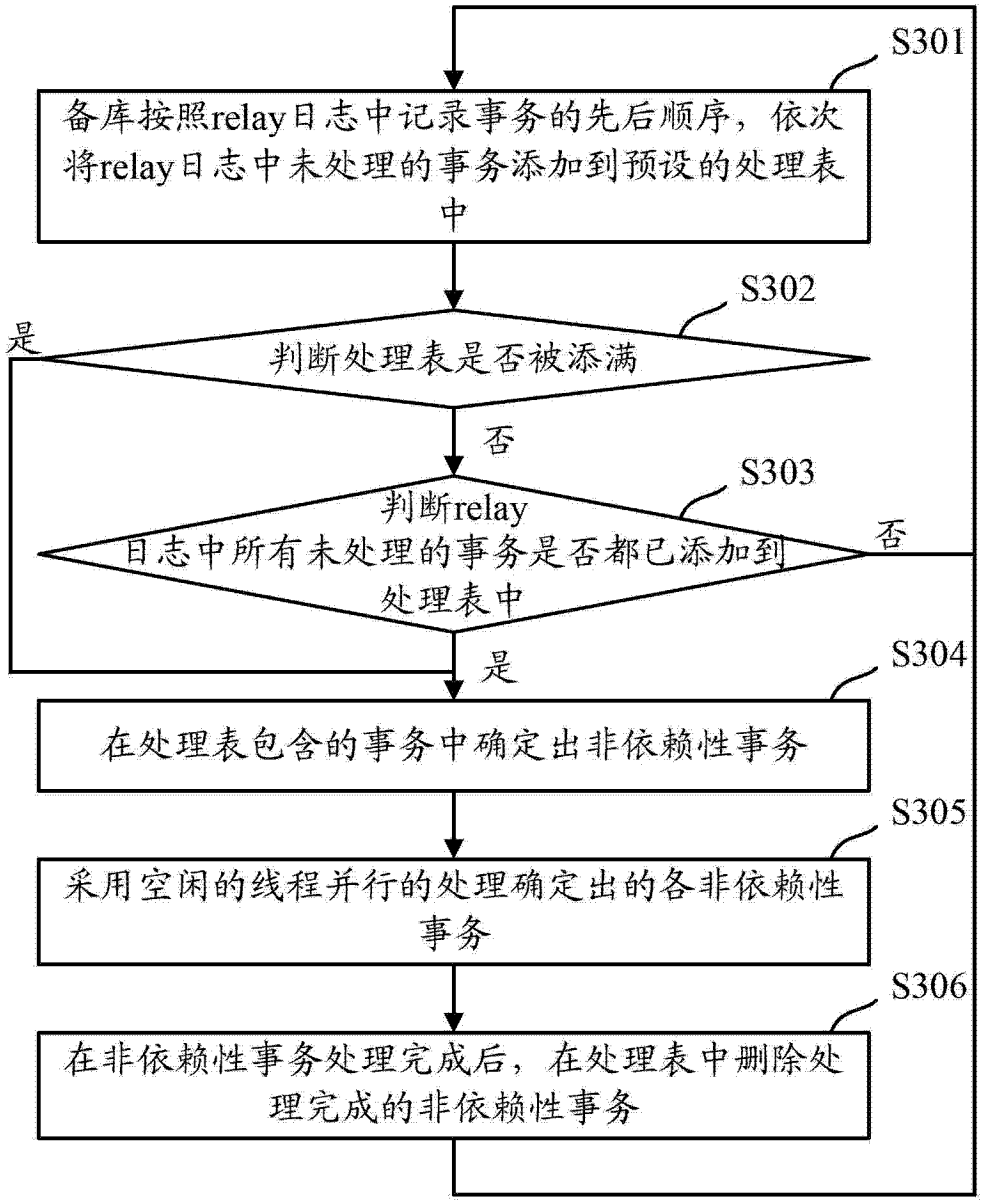

S301
备库按照relay日志中记录事务的先后顺序，依次将relay日志中未处理的事务添加到预设的处理表中
S302
判断处理表是否被添满
是
否
S303
判断relay日志中所有未处理的事务是否都已添加到处理表中
否
是
S304
在处理表包含的事务中确定出非依赖性事务
S305
采用空闲的线程并行的处理确定出的各非依赖性事务
S306
在非依赖性事务处理完成后，在处理表中删除处理完成的非依赖性事务

图 3

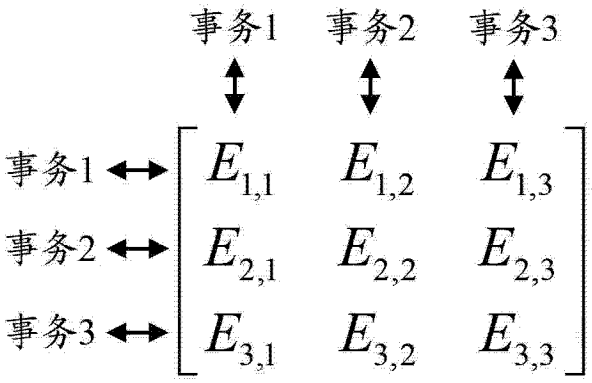

事务1
事务2
事务3
事务1
事务2
事务3
$E_{1,1}$ $E_{1,2}$ $E_{1,3}$
$E_{2,1}$ $E_{2,2}$ $E_{2,3}$
$E_{3,1}$ $E_{3,2}$ $E_{3,3}$

图 4

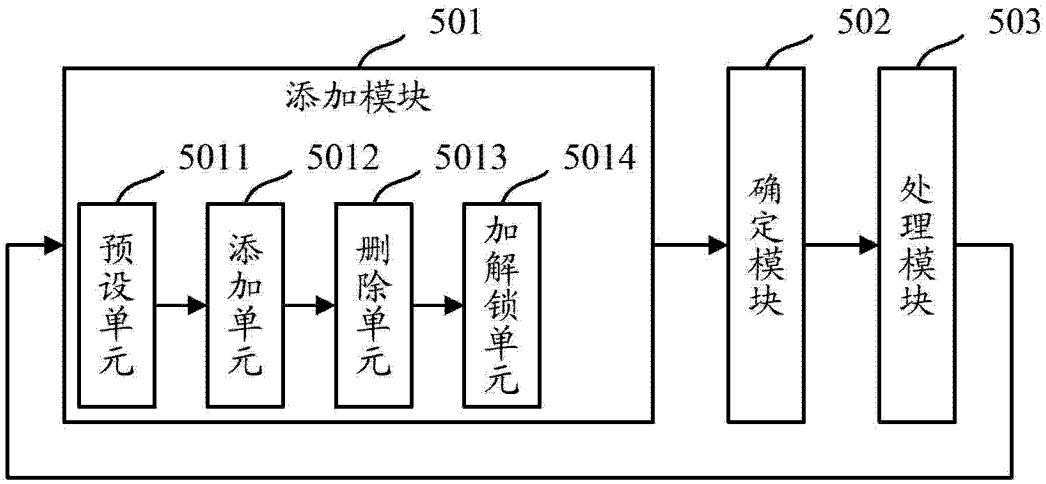

501
添加模块
5011
5012
5013
5014
预设单元
添加单元
删除单元
加解锁单元
502
确定模块
503
处理模块

图 5