(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 106682139 B
(45)授权公告日 2019.09.13

(21)申请号 201611181717.6

(22)申请日 2016.12.19

(65)同一申请的已公布的文献号
申请公布号 CN 106682139 A

(43)申请公布日 2017.05.17

(73)专利权人 深圳盒子信息科技有限公司
地址 518000 广东省深圳市南山区粤海街道深圳市软件产业基地第5栋裙楼505室

(72)发明人 石祖恒 边浩男 黄利庆 韩昌雷

(74)专利代理机构 深圳中一专利商标事务所 44237
代理人 阳开亮

(51)Int.Cl.
G06F 16/2453(2019.01)

(56)对比文件
CN 105787118 A,2016.07.20,
Geosmart"s Notes.基于Solr的Hbase二级索引.《http://geosmart.github.io/2015/09/01/基于Solr的Hbase二级索引/》.2015,1—2、8.

审查员 杜锦锦

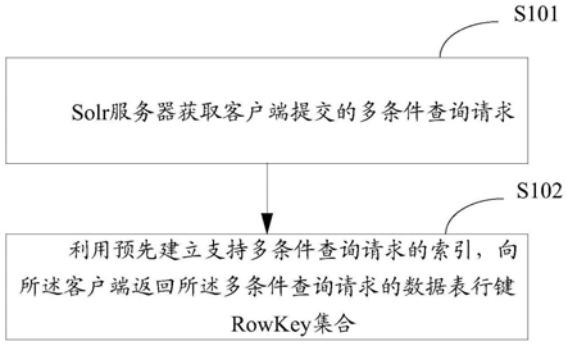
权利要求书2页 说明书7页 附图5页

(54)发明名称

一种基于Solr实现HBase多条件查询的方法及系统

(57)摘要

本发明适用于大数据领域,提供了一种基于Solr实现HBase多条件查询的方法及系统,所述方法包括:客户端向HBase数据库表提交多条件查询请求时,首先根据查询条件从Solr服务器中查询预先建立的索引,向客户端返回HBase数据库表行键RowKey集合,然后根据RowKey直接查询HBase表,返回最终结果;其中,所述RowKey集合的元素为数据库表行键。本发明解决了现有实现多条件查询功能的方式,查询HBase表时需要全表扫描,难以满足更多查询需求的问题。有益效果在于以下两方面,一方面,查询HBase表时,无需全表扫描,减少了查询时间,提高了查询效率,另一方面,具备足够的灵活性,能满足多条件查询的需求,提高了查询的智能程度。



CN 106682139 B

1.一种基于Solr实现HBase多条件查询的方法,其特征在于,包括:

Solr服务器获取客户端提交的多条件查询请求;

利用预先建立支持多条件查询请求的索引,向所述客户端返回所述多条件查询请求的数据表行键RowKey集合;

其中,所述RowKey集合的元素为数据表行键,所述数据表行键为开源数据库HBase中数据表的行键;

在利用预先建立支持多条件查询请求的索引,向所述客户端返回所述多条件查询请求的数据表行键RowKey集合之前,所述方法还包括:

将外部请求的操作码、数据表名称、当前时间戳和数据表行键,拼接成缓存表的行键,为数据行建立支持多条件查询请求的索引;

所述将外部请求的操作码、数据表名称、当前时间戳和数据表行键,拼接成缓存表的行键,为所述数据行建立支持多条件查询请求的索引,具体为:

接收拼接请求,将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键,拼接成缓存表的行键;

在缓存表入库线程启动之前,利用缓存表的preput()钩子先启动数据表的入库线程,当数据表入库线程完成之后,缓存表的preput()钩子结束返回,缓存表入库线程继续往下运行,同时数据表的postput()钩子先启动Solr入库线程、后启动确认线程,利用所述确认线程,确认数据表或Solr中都不存在所述数据行的记录时,将所述缓存表的行键插入到HBase的数据表中;

为所述数据行建立Solr中支持多条件查询请求的索引。

2.如权利要求1所述的方法,其特征在于,所述接收拼接请求,将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键,拼接成缓存表的行键,具体为:

在HBase中建立缓存表,接收拼接请求,将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键,拼接成缓存表的行键,将所述缓存表的行键插入到所述缓存表中。

3.如权利要求1或2所述的方法,其特征在于,在缓存表入库线程启动之前,利用缓存表的preput()钩子先启动数据表的入库线程,当数据表入库线程完成之后,缓存表的preput()钩子结束返回,缓存表入库线程继续往下运行,同时数据表的postput()钩子先启动Solr入库线程、后启动确认线程,利用所述确认线程,确认数据表或Solr中都不存在所述数据行的记录时,将所述缓存表的行键插入到HBase的数据表中之前,所述方法,还包括:

建立确认线程,所述确认线程具体为:

倘若所述数据行的操作码为ADD,则从数据表和Solr中查找之前的数据行记录,遍历之前的数据行记录,确定是否存在所述数据行的记录,若所述数据表和Solr中同时存在所述数据行的记录,从缓存表中删除所述数据行,否则,对所述数据行进行入库处理;

倘若所述数据行的操作码为DEL,则从数据表和Solr中查找之前的数据行记录,遍历之前的数据行记录,确定是否存在所述数据行的记录,若所述数据表和Solr中都不存在所述数据行的记录,从缓存表中删除所述数据行,否则,启动数据删除线程,对所述数据行进行删除处理。

4.一种基于Solr实现HBase多条件查询的系统,其特征在于,包括:

多条件查询请求获取模块,用于获取客户端提交的多条件查询请求;

RowKey返回模块，用于利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合；

其中，所述RowKey集合的元素为数据表行键，所述数据表行键为开源数据库HBase中数据表的行键；

所述系统还包括：

索引建立模块，用于将外部请求的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，为所述数据行建立支持多条件查询请求的索引；

所述索引建立模块包括：

数据行拼接模块，用于接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键；

数据行插入模块，用于在缓存表入库线程启动之前，利用缓存表的preput()钩子先启动数据表的入库线程，当数据表入库线程完成之后，缓存表的preput()钩子结束返回，缓存表入库线程继续往下运行，同时数据表的postput()钩子先启动Solr入库线程、后启动确认线程，利用所述确认线程，确认数据表或Solr中都不存在所述数据行的记录时，将所述缓存表的行键插入到HBase的数据表中；

索引建立模块，用于为所述数据行建立Solr中支持多条件查询请求的索引。

5.如权利要求4所述的系统，其特征在于，所述系统还包括：

缓存表入库模块，用于在HBase中建立缓存表，接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，将所述缓存表的行键插入到所述缓存表中。

6.如权利要求4或5所述的系统，其特征在于，所述系统还包括：

数据行确认模块，用于建立确认线程，所述确认线程具体为：

倘若所述数据行的操作码为ADD，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中同时存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据表入库线程，对所述数据行进行入库处理；

倘若所述数据行的操作码为DEL，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中都不存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据删除线程，对所述数据行进行删除处理。

一种基于Solr实现HBase多条件查询的方法及系统

技术领域

[0001] 本发明属于大数据领域，尤其涉及一种基于Solr实现HBase多条件查询的方法及系统。

背景技术

[0002] HBase是一个分布式的、面向列的开源数据库，根据数据表行键字典排序存储数据，使用单一数据表行键进行查询时，查询效率十分高效。但是这种单一的查询方式不能满足更多的查询需求，如果需要实现类似于关系型数据库那样随意组合的多条件查询功能，可以采用以下两种方式：

[0003] 1.第一种方式：

[0004] 使用HBase提供的过滤器filter原生应用程序编程接口(Application Programming Interface,API)。这种方式使用起来方便简单，但是局限性很大，当表的数据量比较大时，直接全表扫描记录，查询速度会非常慢。

[0005] 2.第二种方式：

[0006] 优化数据表行键RowKey，将查询条件拼接在RowKey。这种方式在一定程度上可以满足多条件查询的需求，但是如果业务需求改变的时候，扩展性和灵活性太差。

[0007] 综上所述，现有实现多条件查询功能的方式，存在以下不足，详述如下：

[0008] 1.查询HBase表时需要全表扫描。

[0009] 2.查询的扩展性和灵活性太差。

发明内容

[0010] 本发明实施例的目的在于提供一种基于Solr实现HBase多条件查询的方法，旨在解决现有实现多条件查询功能的方式，查询HBase表时需要全表扫描，难以满足更多查询需求的问题。

[0011] 本发明实施例是这样实现的，一种基于Solr实现HBase多条件查询的方法，包括：

[0012] Solr服务器获取客户端提交的多条件查询请求；

[0013] 利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合；

[0014] 其中，所述RowKey集合的元素为数据表行键，所述数据表行键为开源数据库HBase中数据表的行键。

[0015] 本发明实施例的另一目的在于提供一种基于Solr实现HBase多条件查询的系统，包括：

[0016] 多条件查询请求获取模块，用于获取客户端提交的多条件查询请求；

[0017] RowKey返回模块，用于利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合；

[0018] 其中，所述RowKey集合的元素为数据表行键，所述数据表行键为开源数据库HBase

中数据表的行键。

[0019] 在本发明实施例中，利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合。使得客户端可以通过RowKey集合，查询到符合条件查询请求的结果集。解决了现有实现多条件查询功能的方式，查询HBase表时需要全表扫描，难以满足更多查询需求的问题。有益效果在于以下两方面，一方面，查询HBase表时，无需全表扫描，减少了查询时间，提高了查询效率，另一方面，具备足够的灵活性，能满足多条件查询的需求，提高了查询的智能程度。

附图说明

[0020] 图1是本发明实施例提供的基于Solr实现HBase多条件查询的方法的实现流程图；

[0021] 图2是本发明实施例提供的建立支持多条件查询请求的索引的实现流程图；

[0022] 图3是本发明实施例提供的查询时序图；

[0023] 图4是本发明实施例提供的插入数据建立索引时序图；

[0024] 图5是本发明实施例提供的删除数据删除索引时序图；

[0025] 图6是本发明实施例提供的基于Solr实现HBase多条件查询的系统的结构框图。

具体实施方式

[0026] 为了使本发明的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本发明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仅用以解释本发明，并不用于限定本发明。

[0027] 应当理解，当在本说明书和所附权利要求书中使用时，术语“包括”和“包含”指示所描述特征、整体、步骤、操作、元素和/或组件的存在，但并不排除一个或多个其它特征、整体、步骤、操作、元素、组件和/或其集合的存在或添加。

[0028] 还应当理解，在此本发明说明书中所使用的术语仅仅是出于描述特定实施例的目的而并不意在限制本发明。如在本发明说明书和所附权利要求书中所使用的那样，除非上下文清楚地指明其它情况，否则单数形式的“一”、“一个”及“该”意在包括复数形式。

[0029] 还应当进一步理解，在本发明说明书和所附权利要求书中使用的术语“和/或”是指相关联列出的项中的一个或多个的任何组合以及所有可能组合，并且包括这些组合。

[0030] 如在本说明书和所附权利要求书中所使用的那样，术语“倘若”可以依据上下文被解释为“当...时”或“一旦”或“响应于确定”或“响应于检测到”。类似地，短语“倘若确定”或“倘若读取到[所描述条件或事件]”可以依据上下文被解释为意指“一旦确定”或“响应于确定”或“一旦检测到[所描述条件或事件]”或“响应于检测到[所描述条件或事件]”。

[0031] 实施例一

[0032] 图1是本发明实施例提供的基于Solr实现HBase多条件查询的方法的实现流程图，详述如下：

[0033] 在步骤S101中，Solr服务器获取客户端提交的多条件查询请求；

[0034] 在步骤S102中，利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合；

[0035] 其中，所述RowKey集合的元素为数据表行键，所述数据表行键为开源数据库HBase

中数据表的行键。

[0036] 其中，数据行包括数据表行键。

[0037] 通过将支持多条件查询请求的索引关联数据行，完成支持多条件查询请求的索引与数据行中数据表行键的关联。

[0038] 在步骤S102之前，所述方法包括：

[0039] 建立与开源数据库HBase中的数据表行键相对应，且支持多条件查询请求的索引。

[0040] 在本发明实施例中，有益效果在于以下两方面，一方面，查询HBase表时，无需全表扫描，减少了查询时间，提高了查询效率，另一方面，具备足够的灵活性，能满足多条件查询的需求，提高了查询的智能程度。

[0041] 实施例二

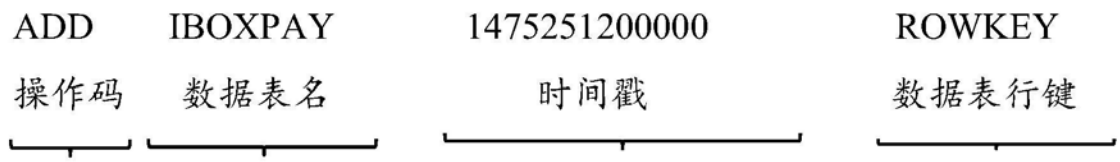
[0042] 本发明实施例描述了建立支持多条件查询请求的索引的实现流程，详述如下：

[0043] 将外部请求的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，为所述数据行建立支持多条件查询请求的索引。

[0044] 其中，缓存表的行键包括以下特征：

[0045] 操作码+数据表名+时间戳+数据表行键(RowKey)，采用的结构举例如下：

[0046]



[0047] 其中，操作码、数据表名、时间戳和数据表行键在数据行中的位置关系，需要在建立索引过程、删除索引过程和查询过程三者之间约定次序。

[0048] 实施例三

[0049] 图2是本发明实施例提供的建立支持多条件查询请求的索引的实现流程图，详述如下：

[0050] 在步骤S201中，接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键；

[0051] 在步骤S202中，在缓存表入库线程启动之前，利用缓存表的preput()钩子先启动数据表的入库线程，当数据表入库线程完成之后，缓存表的preput()钩子结束返回，缓存表入库线程继续往下运行，同时数据表的postput()钩子先启动Solr入库线程、后启动确认线程，利用所述确认线程，确认数据表或Solr中都不存在所述数据行的记录时，将所述缓存表的行键插入到HBase的数据表中；

[0052] 当增加数据行或更新数据行时，配置操作码为ADD；

[0053] 当删除数据行时，配置操作码为DEL；

[0054] 当操作码为ADD时，启动数据表入库线程，利用数据表入库线程将所述缓存表的行键插入到数据表中，其中，时间戳与所述缓存表中的保持一致；

[0055] 当操作码为DEL时，启动数据删除线程，利用数据删除线程将数据行从数据表中删除。

[0056] preput()钩子和postput()钩子是常用的钩子函数，在此不做赘述。

[0057] 在步骤S203中，为所述数据行建立Solr中支持多条件查询请求的索引。

[0058] 利用Solr入库线程，为所述数据行建立支持多条件查询请求的索引；

[0059] 利用Solr删除线程，删除所述数据行对应的支持多条件查询请求的索引。

[0060] 在本发明实施例中，通过缓存表和确认线程，解决了HBase与Solr数据一致性问题，能避免以下两种情况的发生：

[0061] (1)HBase中数据写入成功，Solr中没有写入成功，导致HBase中出现一些根本无法查询到的漂浮数据；

[0062] (2)HBase中数据没有写入成功，但是Solr中写入成功了，导致根据索引查询时出现空集。

[0063] 实施例四

[0064] 本发明实施例描述了步骤S201的实现流程，详述如下：

[0065] 在HBase中建立缓存表，接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，将所述缓存表的行键插入到所述缓存表中。

[0066] 实施例五

[0067] 本发明实施例描述了提供的建立确认线程的实现流程，详述如下：

[0068] 建立确认线程，所述确认线程具体为：

[0069] 倘若所述数据行的操作码为ADD，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中同时存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据表入库线程，对所述数据行进行入库处理；

[0070] 其中，否则，启动数据表入库线程，对所述缓存表的行键进行入库处理，包含以下3种情况：

[0071] 1.数据表不存在所述数据行记录，Solr中存在所述数据行记录；

[0072] 2.数据表存在所述数据行，Solr中不存在所述数据；

[0073] 3.数据表和Solr中都不存在所述数据行。

[0074] 倘若所述数据行的操作码为DEL，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中都不存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据删除线程，对所述数据行进行删除处理。

[0075] 其中，否则，启动数据删除线程，对所述缓存表的行键进行删除处理，包含以下3种情况：

[0076] 1.数据表不存在所述数据行记录，Solr中存在所述数据行记录；

[0077] 2.数据表存在所述数据行，Solr中不存在所述数据；

[0078] 3.数据表和Solr中同时存在所述数据行。

[0079] 实施例六

[0080] 图3是本发明实施例提供的查询时序图，详述如下：使用HBase的Observer类型的协处理器(Coprocessor)结合Solr实现二级索引的功能。业务层多条件查询时，先查询Solr索引，返回HBase表的行键集合，然后通过行键直接命中HBase表的行，返回结果集，避免查询HBase表时的全表扫描或者部分扫描。

[0081] 实施例七

[0082] 图4是本发明实施例提供的插入数据建立索引时序图，详述如下：

[0083] 图4包括：外部请求、缓存表入库线程、确认线程、缓存表、数据表入库线程、数据表、Solr入库线程以及Solr。

[0084] 1.外部请求新增数据，启动缓存表入库线程；

[0085] 2.缓存表的prePut()钩子启动数据表入库线程，数据表入库线程将数据入库，数据表向数据表入库线程发出入库返回的结果，prePut()钩子返回；

[0086] 3.缓存表入库，缓存表入库返回；

[0087] 4.数据表的postPut()钩子启动Solr入库，入库返回；

[0088] 5.数据表的postPut()钩子启动确认线程，确认线程的实施过程如下：

[0089] 查询时间戳比此记录的时间戳小的记录，返回数据行集；

[0090] 循环部分：foreach(数据行)，获取行锁，检查数据表是否存在此记录，检查Solr是否存在此记录；

[0091] 替换部分：if(数据表和Solr都存在此记录)，删除记录行，else启动数据表入库线程；

[0092] 释放行锁。

[0093] 其中，缓存表和数据表增加Observer类型的协处理器(Coprocessor)，并为缓存表的prePut()钩子事件增添启动数据表入库线程和数据删除线程的业务逻辑处理功能；

[0094] 为数据表的postPut钩子事件，负责启动Solr入库线程和确认线程。

[0095] 当操作码为ADD时，启动数据表入库线程；

[0096] 当操作码为DEL时，启动数据删除线程。

[0097] 实施例八

[0098] 图5是本发明实施例提供的删除数据删除索引时序图，详述如下：

[0099] 图5包括：外部请求、缓存表入库线程、确认线程、缓存表、数据删除线程、数据表、Solr删除线程以及Solr。

[0100] 1.外部请求删除数据，启动缓存表入库线程；

[0101] 2.prePut()钩子启动，数据入库，数据删除线程删除数据行，删除返回，prePut()钩子返回；

[0102] 3.缓存表入库，缓存表入库返回；

[0103] 4.数据表的preDelete()钩子启动Solr删除线程，Solr删除线程删除Solr记录，删除返回，preDelete()钩子返回，删除数据行，删除返回；

[0104] 5.数据表的postDelete()钩子启动确认线程，确认线程的实施过程如下：

[0105] 查询时间戳比此记录的时间戳小的记录，返回数据行集；

[0106] 循环部分：foreach(数据行)，获取行锁，检查数据表是否存在此记录，检查Solr是否存在此记录；

[0107] 替换部分：if(数据表和Solr都不存在此记录)，删除缓存表记录行，else启动数据删除线程；

[0108] 释放行锁。

[0109] 实施例九

[0110] 图6是本发明实施例提供的基于Solr实现HBase多条件查询的系统的结构框图。为了便于说明，仅示出了与本实施例相关的部分。

[0111] 参照图6，该基于Solr实现HBase多条件查询的系统，包括：

[0112] 多条件查询请求获取模块61，用于获取客户端提交的多条件查询请求；

[0113] RowKey返回模块62，用于利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合；

[0114] 其中，所述RowKey集合的元素为数据表行键，所述数据表行键为开源数据库HBase中数据表的行键。

[0115] 作为本实施例的一种实现方式，所述系统还包括：

[0116] 索引建立模块，用于将外部请求的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，为所述数据行建立支持多条件查询请求的索引。

[0117] 作为本实施例的一种实现方式，在所述系统中，所述索引建立模块包括：

[0118] 数据行拼接模块，用于接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键；

[0119] 数据行插入模块，用于在缓存表入库线程启动之前，利用缓存表的preput()钩子先启动数据表的入库线程，当数据表入库线程完成之后，缓存表的preput()钩子结束返回，缓存表入库线程继续往下运行，同时数据表的postput()钩子先启动Solr入库线程、后启动确认线程，利用所述确认线程，确认数据表或Solr中都不存在所述数据行的记录时，将所述缓存表的行键插入到HBase的数据表中；

[0120] 索引建立模块，用于为所述数据行建立Solr中支持多条件查询请求的索引。

[0121] 作为本实施例的一种实现方式，所述系统还包括：

[0122] 缓存表入库模块，用于在HBase中建立缓存表，接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成缓存表的行键，将所述缓存表的行键插入到所述缓存表中。

[0123] 作为本实施例的一种实现方式，所述系统还包括：

[0124] 数据行确认模块，用于建立确认线程，所述确认线程具体为：

[0125] 倘若所述数据行的操作码为ADD，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中同时存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据表入库线程，对所述数据行进行入库处理；

[0126] 倘若所述数据行的操作码为DEL，则从数据表和Solr中查找之前的数据行记录，遍历之前的数据行记录，确定是否存在所述数据行的记录，若所述数据表和Solr中都不存在所述数据行的记录，从缓存表中删除所述数据行，否则，启动数据删除线程，对所述数据行进行删除处理。

[0127] 本发明实施例方法中的步骤可以根据实际需要进行顺序调整、合并和删减。

[0128] 本发明实施例系统和系统中的单元可以根据实际需要进行合并、划分和删减。

[0129] 本发明实施例提供的系统可以应用在前述对应的方法实施例中，详情参见上述实施例的描述，在此不再赘述。

[0130] 通过以上的实施方式的描述，所属领域的技术人员可以清楚地了解到本发明可借

助软件加必需的通用硬件的方式来实现。所述的程序可以存储于可读取存储介质中，所述的存储介质，如随机存储器、闪存、只读存储器、可编程只读存储器、电可擦写可编程存储器、寄存器等。该存储介质位于存储器，处理器读取存储器中的信息，结合其硬件执行本发明各个实施例所述的方法。

[0131] 以上所述，仅为本发明的具体实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，可轻易想到的变化或替换，都应涵盖在本发明的保护范围之内。因此，本发明的保护范围应以权利要求的保护范围为准。

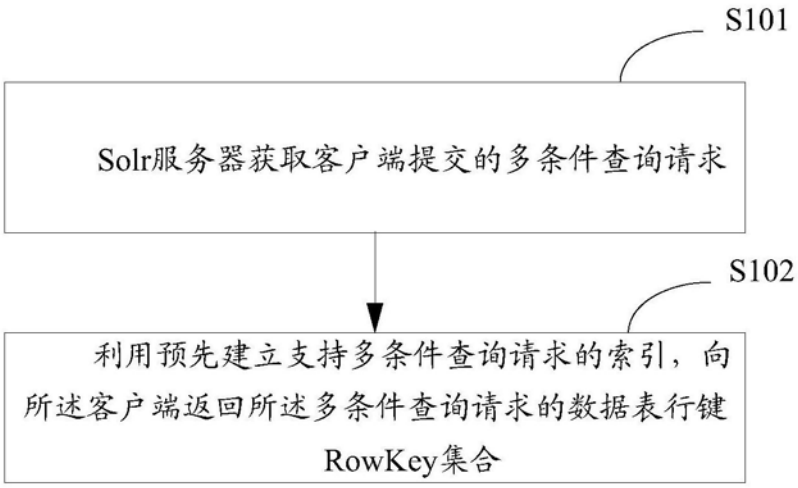

S101
Solr服务器获取客户端提交的多条件查询请求
S102
利用预先建立支持多条件查询请求的索引，向所述客户端返回所述多条件查询请求的数据表行键RowKey集合

图1

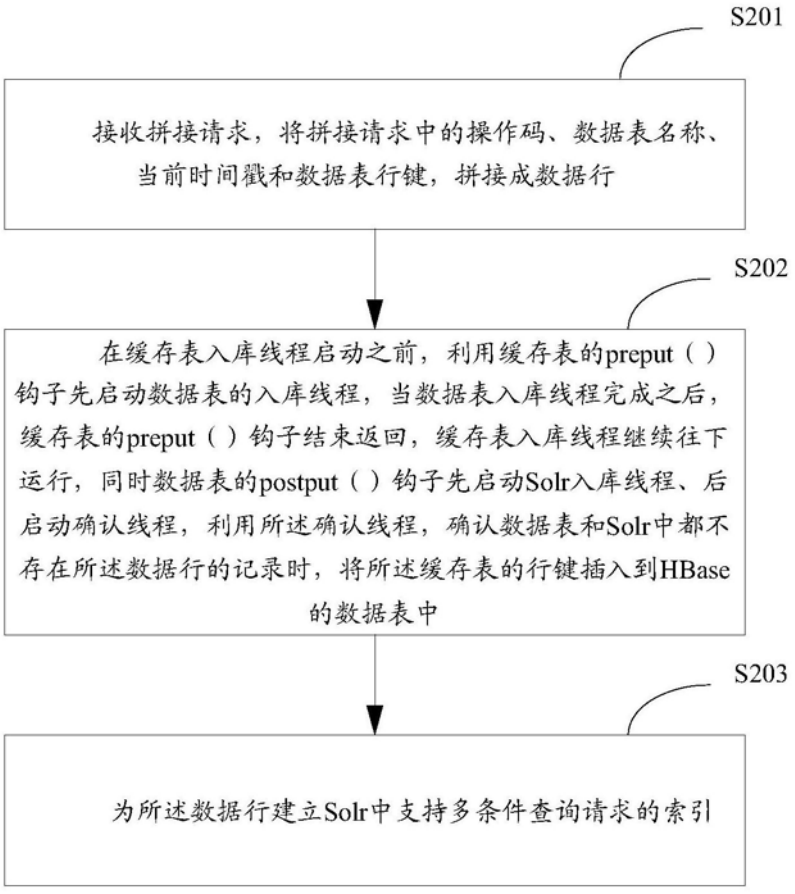

S201
接收拼接请求，将拼接请求中的操作码、数据表名称、当前时间戳和数据表行键，拼接成数据行
S202
在缓存表入库线程启动之前，利用缓存表的preput（）钩子先启动数据表的入库线程，当数据表入库线程完成之后，缓存表的preput（）钩子结束返回，缓存表入库线程继续往下运行，同时数据表的postput（）钩子先启动Solr入库线程、后启动确认线程，利用所述确认线程，确认数据表和Solr中都不存在所述数据行的记录时，将所述缓存表的行键插入到HBase的数据表中
S203
为所述数据行建立Solr中支持多条件查询请求的索引

图2

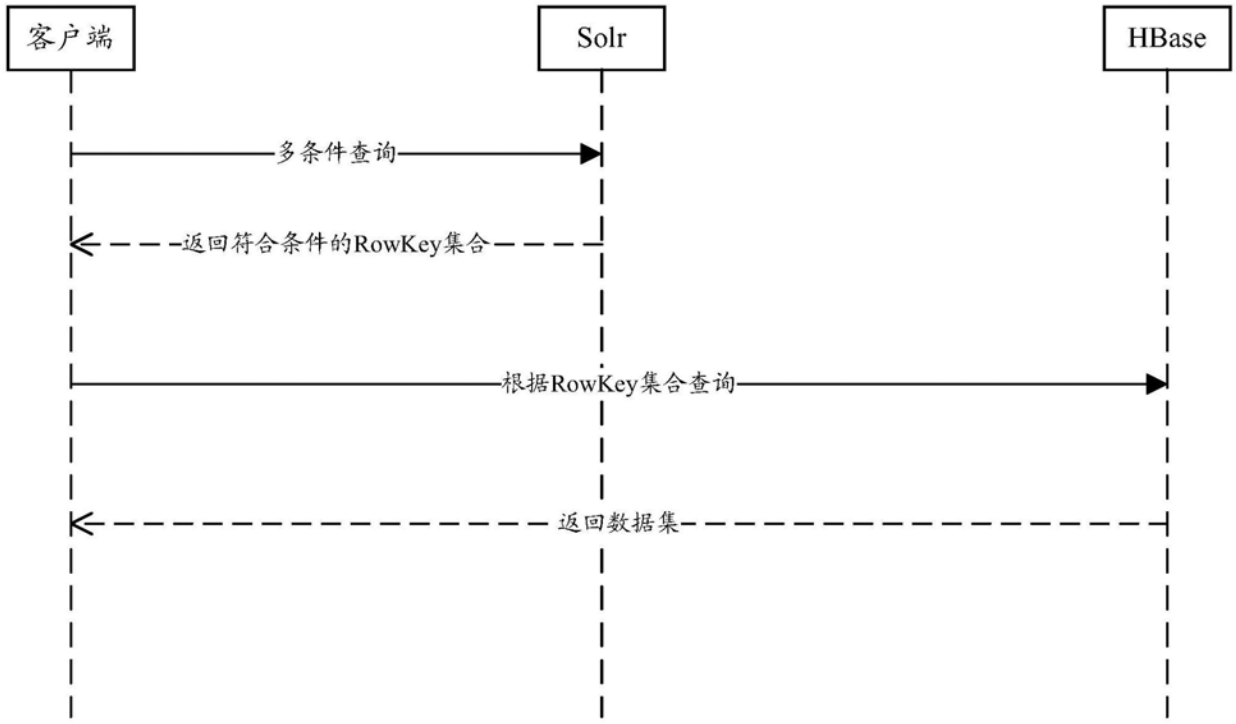


图3

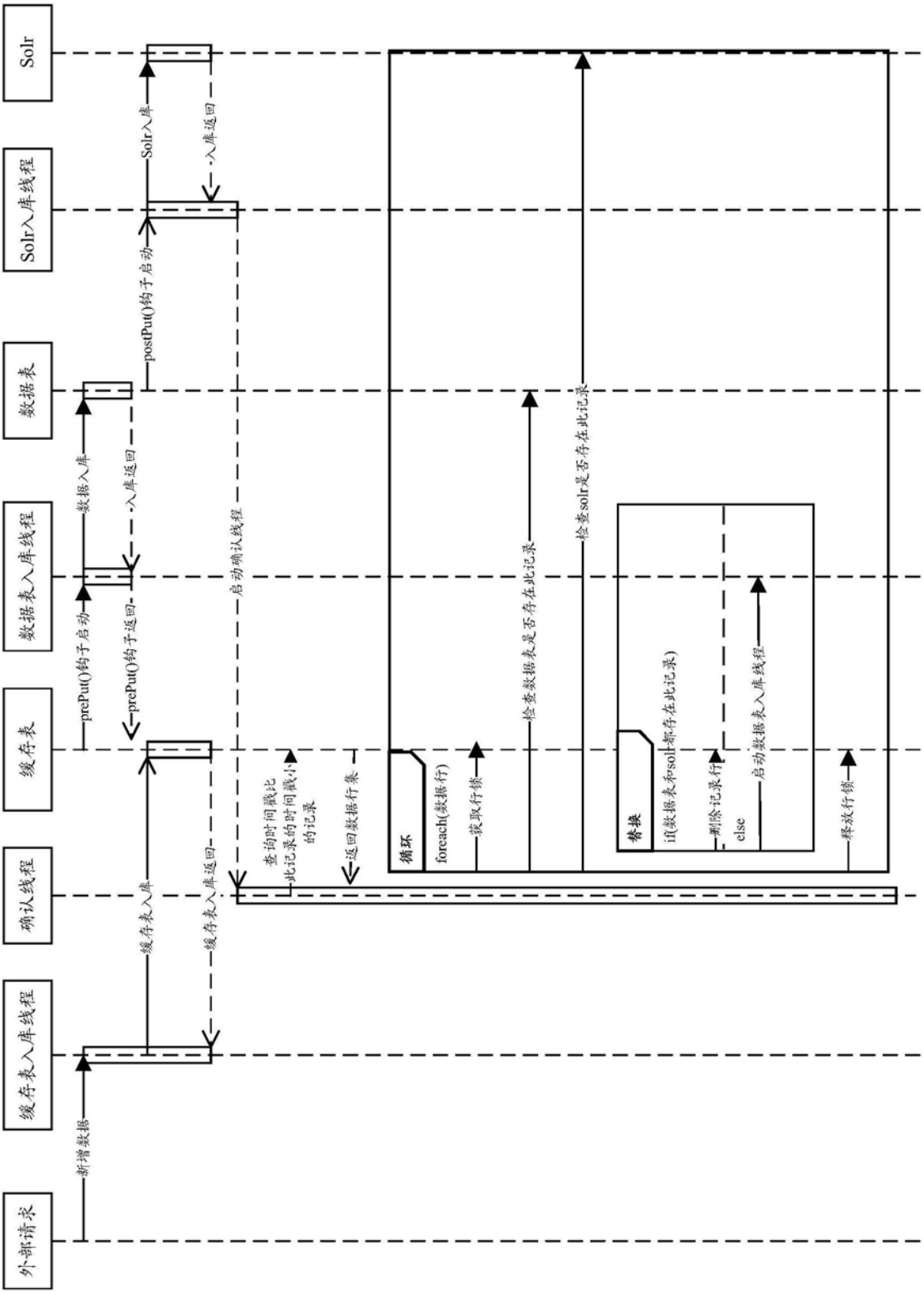

外部请求
缓存表入库线程
确认线程
缓存表
数据表入库线程
数据表
Solr入库线程
Solr
新增数据
缓存表入库
缓存表入库返回
prePut()钩子启动
prePut()钩子返回
数据入库
入库返回
postPut()钩子启动
Solr入库
入库返回
启动确认线程
查询时间戳比此记录的时间戳小的记录
返回数据行集
循环
foreach(数据行)
获取行锁
检查数据表是否存在此记录
检查solr是否存在此记录
替换
if(数据表和solr都存在此记录)
删除记录行
else
启动数据表入库线程
释放行锁

图4

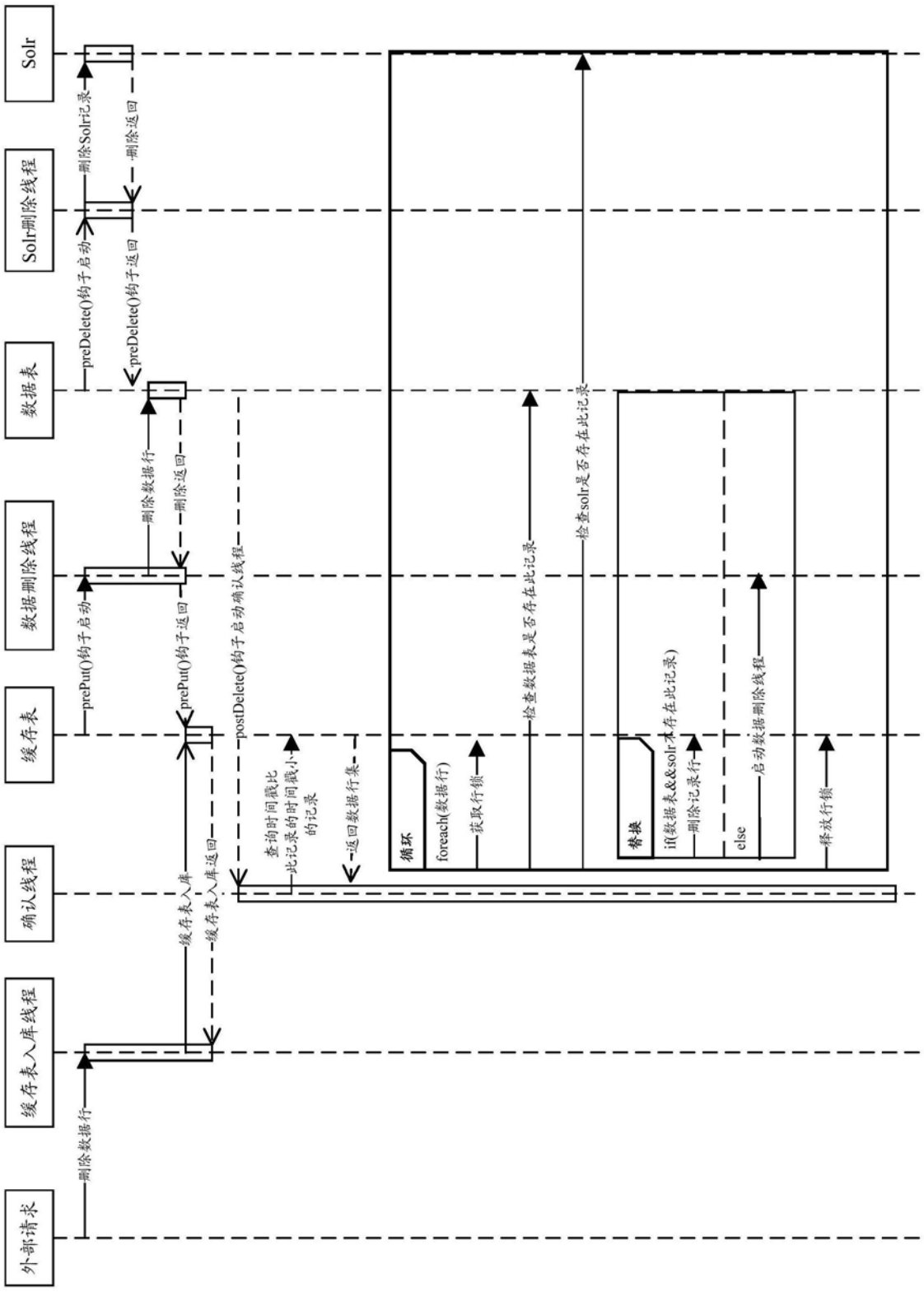

外部请求
缓存表入库线程
确认线程
缓存表
数据删除线程
数据表
Solr删除线程
Solr
删除数据行
缓存表入库
缓存表入库返回
prePut()钩子启动
prePut()钩子返回
postDelete()钩子启动确认线程
查询时间戳比
此记录的时间戳小
的记录
返回数据行集
循环
foreach(数据行)
获取行锁
检查数据表是否存在此记录
检查solr是否存在此记录
替换
if(数据表&&solr不存在此记录)
删除记录行
else
启动数据删除线程
释放行锁
删除数据行
删除返回
preDelete()钩子启动
preDelete()钩子返回
删除Solr记录
删除返回

图5

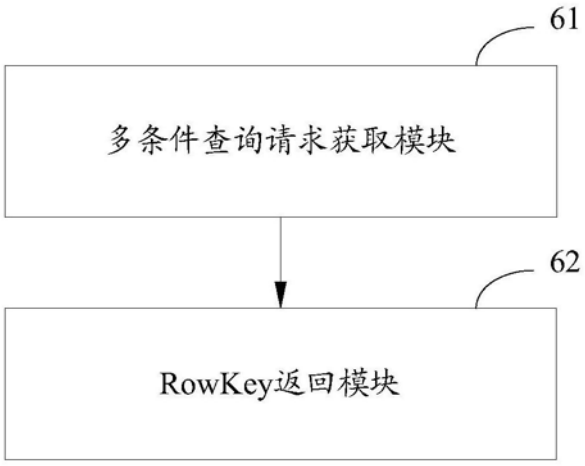


图6