(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111881092 B
(45)授权公告日 2024.07.09

(21)申请号 202010576064.1

(22)申请日 2020.06.22

(65)同一申请的已公布的文献号
申请公布号 CN 111881092 A

(43)申请公布日 2020.11.03

(73)专利权人 武汉绿色网络信息服务有限责任公司
地址 430000 湖北省武汉市东湖新技术开发区软件园中路4号光谷软件园六期2栋4层01室、5层01室、6层01室

(72)发明人 叶志钢 王化民 张本军 王赟 谭国权 赵雨佳

(74)专利代理机构 深圳市六加知识产权代理有限公司 44372
专利代理师 向彬

(51)Int.Cl.
G06F 16/16(2019.01)
G06F 16/182(2019.01)

(56)对比文件
CN 110727685 A,2020.01.24

审查员 熊晶

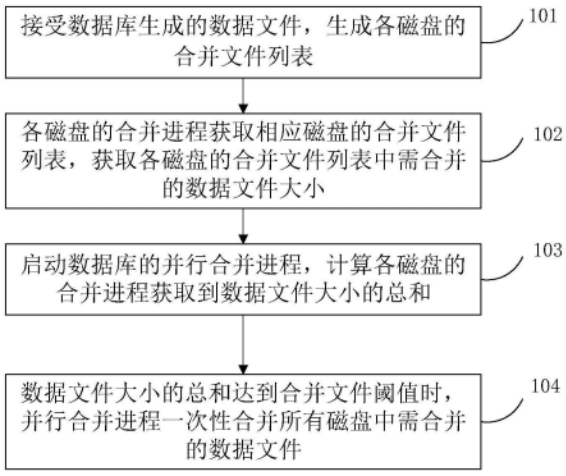
权利要求书1页 说明书8页 附图3页

(54)发明名称

一种基于cassandra数据库的文件合并的方法和装置

(57)摘要

本发明涉及数据库领域,特别是涉及一种基于cassandra数据库的文件合并的方法和装置。主要包括:接受数据库生成的数据文件,生成各磁盘的合并文件列表;各磁盘的合并进程获取相应磁盘的合并文件列表,获取各磁盘的合并文件列表中需合并的数据文件大小;启动;数据库的并行合并进程,计算各磁盘的合并进程获取到数据文件大小的总和数据文件大小的总和达到合并文件阈值时,并行合并进程一次性合并所有磁盘中需合并的数据文件。本发明可以在使用较少合并层次和临时文件的情况下及时对小文件进行合并,减少合并次数、减少磁盘中文件占用空间、减少磁盘IO次数和磁盘IO争抢,提高了文件合并的性能,提高了数据库的读写稳定性。



CN 111881092 B

1.一种基于cassandra数据库的文件合并的方法,其特征在于:

接受数据库生成的数据文件,生成各磁盘的合并文件列表;

各磁盘的合并进程获取相应磁盘的合并文件列表,获取各磁盘的合并文件列表中需合并的数据文件大小;

启动数据库唯一的并行合并进程,计算各磁盘的合并进程获取到数据文件大小的总和;

若需合并的文件总大小未达到合并文件阈值,不进行本次合并,等待接受数据库下一次生成的数据文件;

数据文件大小的总和达到合并文件阈值时,并行合并进程一次性合并所有磁盘中需合并的数据文件,按照文件大小阈值将需合并的数据文件合并为至少一个文件,剩余的小于文件大小阈值的部分不进行合并;

判断每个数据文件的大小是否小于文件数据量阈值;

若小于文件数据量阈值,合并文件阈值使用第一合并文件阈值;

若大于文件数据量阈值,合并文件阈值使用第二合并文件阈值;

其中,第一合并文件阈值大于第二合并文件阈值。

2.根据权利要求1所述的基于cassandra数据库的文件合并的方法,其特征在于:接受生成的数据文件前,禁用系统自带的合并策略。

3.根据权利要求1所述的基于cassandra数据库的文件合并的方法,其特征在于:指定需进行合并的目录,仅对需进行合并的目录启动合并进程进行文件合并。

4.根据权利要求1所述的基于cassandra数据库的文件合并的方法,其特征在于,一次性合并所有需合并的数据文件后,还包括:

并行合并进程统计合并时间;

判断合并时间是否超过合并时间阈值;

对合并时间超过合并时间阈值的磁盘进行告警。

5.根据权利要求1所述的基于cassandra数据库的文件合并的方法,其特征在于:若合并时间超过合并时间阈值,自动重启数据库进程。

6.根据权利要求1所述的基于cassandra数据库的文件合并的方法,其特征在于:将需合并的数据文件的源文件标记为删除,一次性合并所有需合并的数据文件后,将标记为删除的数据文件进行删除。

7.一种基于cassandra数据库的文件合并的装置,其特征在于:

包括至少一个处理器和存储器,所述至少一个处理器和存储器之间通过数据总线连接,所述存储器存储能被所述至少一个处理器执行的指令,所述指令在被所述处理器执行后,用于完成权利要求1-6中任一项所述的基于cassandra数据库的文件合并的方法。

一种基于cassandra数据库的文件合并的方法和装置

【技术领域】

[0001] 本发明涉及数据库领域,特别是涉及一种基于cassandra数据库的文件合并的方法和装置。

【背景技术】

[0002] Cassandra数据库是一种开源的分布式混合存储的存储方案,具备去中心化、可扩展性、高可用、容错性和可配置的一致性等特性。由于cassandra将缓存的数据顺序刷入磁盘时,会生成多个1-10MB左右的数据文件(Sorted String Table,简写为:sstable)。在cassandra数据库数据处理量很大时,巨大的文件数量会严重影响数据库的稳定性,并且拖慢查询速度。

[0003] 为了减小需处理文件的数量,cassandra数据库提供了文件合并(Compaction)机制将多个大文件合并为少量小文件。目前常用的文件合并策略有:

[0004] (1)Size Tiered Compaction Strategy

[0005] 基于文件大小进行多层合并,将小文件放入小文件合并线程进行合并,大文件放入大文件合并线程进行合并,最终获得需要的文件大小。每次合并文件数量由max_threshold参数决定,理论上该参数给使用者提供了控制合并规模的手段,但在实际生产环境中的效果是:当使用较小的max_threshold参数(如32)时,可快速减少文件数量,但是合并后的文件仍然很小,还会触发二次合并,而二次合并需要的磁盘IO资源更多,常常会和下一个批次的写入同时进行,抢夺磁盘IO资源。当使用较大的max_threshold参数(如128)时,不能快速减少文件数量,当达到合并条件时,常常会和下一个批次的写入同时进行,抢夺磁盘IO资源,导致写入性能严重抖动。该策略的不足在于:多个小文件合并后可能无法达到所需的文件大小,需要多次滚动合并,才能变成所需文件大小的大文件,这就导致一份数据需要多次合并,多次读写放大了磁盘读写负担。

[0006] (2)Leveled Compaction Strategy

[0007] 从最高的level上开始扫描是否需要进行compaction,如果有,将该层的sstable形成一个task。优先对高层sstable进行compaction能有效减少底层到高层compaction时合并的sstable的数。该策略的不足在于:1、优先对高Level的sstable进行压缩确实能减少参与compaction的sstable的数量,但是优先处理高层的sstable就会导致最底层L0层的sstable的堆积,单点管理的sstable过多,影响读写的性能。2、多个大文件合并未完成之前,被合并的数据一直存在磁盘上,极端情况下需要2倍的磁盘空间,可能存在磁盘空间不足的情况。

[0008] (3)Time Window CompactionStrategy

[0009] 基于时间窗口的合并,数据在时间窗口内,按照策略1进行合并,超过时间窗口的文件将不再合并,本质是当合并不能按时完成时,放弃合并。该策略的不足在于:1、因为使用了策略1,因此存在和策略1相同的不足之处。2、当入库频率较高,如达到5分钟/次的入库频率时,小文件急速增加,当时间窗口较大时(1天),也就失去了窗口策略的优势,同策略1

一样会产生较大的合并IO压力,严重影响数据入库,导致服务不可用;当时间窗口较小时(1小时),虽能优先保证写入,但未在窗口时间完成合并的文件放弃合并,会导致文件数量快速增加,每天的数据文件仍然接近10万级,当数据保存时间较长时,最终导致打开的文件数超过数据库进程处理极限,导致数据库进程中止。

[0010] 鉴于此,如何克服该现有技术所存在的缺陷,解决现有文件合并策略中的存在的缺陷,是本技术领域待解决的问题。

【发明内容】

[0011] 针对现有技术的以上缺陷或改进需求,本发明解决了现有文件合并策略中根据文件数量、合并层次或时间窗触发合并动作导致的磁盘读写负担大、磁盘空间占用较多、合并IO压力大等问题。

[0012] 本发明实施例采用如下技术方案:

[0013] 第一方面,本发明提供了一种基于cassandra数据库的文件合并的方法,具体为:接受数据库生成的数据文件,生成各磁盘的合并文件列表;各磁盘的合并进程获取相应磁盘的合并文件列表,获取各磁盘的合并文件列表中需合并的数据文件大小;启动数据库的并行合并进程,计算各磁盘的合并进程获取到数据文件大小的总和;数据文件大小的总和达到合并文件阈值时,并行合并进程一次性合并所有磁盘中需合并的数据文件。

[0014] 优选的,若需合并的文件总大小未达到合并文件阈值,不进行本次合并,等待接受数据库下一次生成的数据文件。

[0015] 优选的,判断每个数据文件的大小是否小于文件数据量阈值;若小于文件数据量阈值,合并文件阈值使用第一合并文件阈值;若大于文件数据量阈值,合并文件阈值使用第二合并文件阈值;其中,第一合并文件阈值大于第二合并文件阈值。

[0016] 优选的,接受生成的数据文件前,禁用系统自带的合并策略。

[0017] 优选的,若需合并的数据文件的大小超过文件大小阈值,按照文件大小阈值将需合并的数据文件合并为至少一个文件,剩余的小于文件大小阈值的部分不进行合并。

[0018] 优选的,指定需进行合并的目录,仅对需进行合并的目录启动合并进程进行文件合并。

[0019] 优选的,一次性合并所有需合并的数据文件后,还包括:并行合并进程统计合并时间;判断合并时间是否超过合并时间阈值;对合并时间超过合并时间阈值的磁盘进行告警。

[0020] 优选的,若合并时间超过合并时间阈值,自动重启数据库进程。

[0021] 优选的,将需合并的数据文件的源文件标记为删除,一次性合并所有需合并的数据文件后,将标记为删除的数据文件进行删除

[0022] 另一方面,本发明提供了一种基于cassandra数据库的文件合并的方法的装置,具体为:包括至少一个处理器和存储器,至少一个处理器和存储器之间通过数据总线连接,存储器存储能被至少一个处理器执行的指令,指令在被处理器执行后,用于完成第一方面中的基于cassandra数据库的文件合并的方法。

[0023] 与现有技术相比,本发明实施例的有益效果在于:通过使用动态合并的方式,使用合并文件阈值作为合并动作的触发项,在小文件大小达到合并文件阈值时对生成的小文件进行及时的一次性合并,提供了一种快速稳定的小文件合并方法。在优选方案中,通过调整

合并文件阈值获取优化的合并后文件数量和写入性能组合,通过合并时间检测和告警及时反馈合并过程异常,通过垃圾回收机制优化文件存储结构和IO效率。

【附图说明】

[0024] 为了更清楚地说明本发明实施例的技术方案,下面将对本发明实施例中所需要使用的附图作简单地介绍。显而易见地,下面所描述的附图仅仅是本发明的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以根据这些附图获得其他的附图。

[0025] 图1为本发明实施例提供的基于cassandra数据库的文件合并的方法流程图;

[0026] 图2为本发明实施例提供的基于cassandra数据库的文件合并的方法流程图;

[0027] 图3为本发明实施例提供的基于cassandra数据库的文件合并的方法流程图;

[0028] 图4为本发明实施例提供的基于cassandra数据库的文件合并的方法流程图;

[0029] 图5为本发明实施例提供的一种基于cassandra数据库的文件合并的装置结构示意图。

【具体实施方式】

[0030] 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。

[0031] 本发明是一种特定功能系统的体系结构,因此在具体实施例中主要说明各结构模组的功能逻辑关系,并不对具体软件和硬件实施方式做限定。

[0032] 此外,下面所描述的本发明各个实施方式中所涉及到的技术特征只要彼此之间未构成冲突就可以相互组合。下面就参考附图和实施例结合来详细说明本发明。

[0033] 实施例1:

[0034] 在cassandra数据库中,当客户端写入数据时,客户端程序根据集群上的token范围确定本条数据应该发送到的服务器节点,服务端多线程并行接受数据,每个线程会对自己接收的数据做排序,生小于10M的数据文件。在某些实施场景中,当cassandra数据库单机处理的数据量达到4TB/天时,数据库进程需要打开的数据文件数量会超过20万,当要求数据存储7天时,进程需要打开的数据文件达到140万,需要对数据文件进行合并以减少文件数量。因此,本实施例提供了一种新的动态小文件合并方法,避免现有文件合并策略中存在的缺陷。

[0035] Cassandra是一款NoSQL分布式数据库,采用Log Structured Merge Tree(简写为:LSM Tree)架构。LSM是一种分层,有序,面向磁盘的数据结构,利用磁盘批量的顺序写远比随机写性能高的特性,通过分层批量读写实现。为了实现文件的分批读写,LSM有两个重要过程:数据顺序刷入磁盘生成数据文件(sstable)和数据文件合并(Compaction)。Cassandra数据库读入用户写入的数据并且缓存在内存中。当缓存满的时候,会将缓存中的数据刷入磁盘,也就生成了sstable文件。通过写缓存,可以将数量批量写入,利用磁盘顺序写性能更好的特点。另一方面,缓存中的数据读入磁盘之前会预先排序,因此可以通过二分法快速进行查找,提高数据的查找效率。

[0036] 如图1所示,本发明实施例提供的基于cassandra数据库的文件合并的方法具体步骤如下:

[0037] 步骤101:接受数据库生成的数据文件,生成各磁盘的合并文件列表。

[0038] Cassandra数据库利用分布式的存储引擎将数据分布式存储在大量普通商用级别服务器上面的海量的结构化数据中,以避免单点故障造成的数据损失或服务异常。客户端每一次写入数据时,cassandra数据库的客户端程序会启动数据顺序刷入磁盘生成数据文件的过程,获取集群上的token范围信息,根据token范围确定本条数据应该发送到的服务器节点,服务端多线程并行接受数据,每个线程会对自己接收的数据做排序,然后生成大量的sstable,以便于后续写入磁盘中。在进行文件合并前,需要获取分布式存储中每个节点中各磁盘的需要合并的文件,生成各磁盘的合并文件列表,用于后续合并时根据合并文件列表获取文件。在具体实施方式中,需要进行合并的文件可以为索引文件、数据文件、bloom filter文件等。

[0039] 在现有的LSM架构中,数据顺序刷入磁盘生成数据文件后,会自动启动基于现有合并策略的数据文件合并的过程。为了在cassandra数据库中仅使用本实施例提供的文件合并的方法进行合并,启动数据顺序刷入磁盘生成数据文件的过程前,即接受生成的数据文件前,需要禁用系统自带的所有合并策略,以避免数据文件合并策略冲突。具体的,可以通过在建立数据库表时指定参数建表时指定参数:compaction＝{'enabled':'false'}来禁用系统自带的所有合并策略。

[0040] 在某些具体实施场景中,仅需要合并某些目录下的文件,或将合并后的文件存放在指定目录下。因此,在生成各磁盘的合并文件列表之前,可以根据实施场景的实际需要,指定需进行合并的目录,仅对需进行合并的目录启动合并进程进行文件合并,以减少需要处理的文件数量,提高文件处理效率。

[0041] 步骤102:各磁盘的合并进程获取相应磁盘的合并文件列表,获取各磁盘的合并文件列表中需合并的数据文件大小。

[0042] 由于cassandra数据库将存储的文件组织为分布式文件系统(Distributed File System,简写为:DFS),每个客户端或服务器端所使用的物理存储资源不一定直接连接在本地节点上,而是通过计算机网络与节点相连;或是若干不同的逻辑磁盘分区或卷标组合在一起而形成的完整的有层次的文件系统。DFS为分布在网络上任意位置的资源提供一个逻辑上的树形文件系统结构,从而使用户访问分布在网络上的共享文件更加简便。在进行文件合并时,为了对不同节点上的文件进行合并,需要对每个节点中不同磁盘的合并文件目录进行读取。在本实施例中,在各磁盘中分别启动一个合并进程,对该磁盘中的合并文件列表进行读取,并获取需合并的数据文件大小,便于后续步骤使用。

[0043] 步骤103:启动数据库的并行合并进程,计算各磁盘的合并进程获取到数据文件大小的总和。

[0044] 为了对集群上所有磁盘中需合并的文件进行统一管理和合并,数据库系统中还需要启动唯一的并行合并进程,并行合并进程可以对各磁盘的合并进程进行扫描,对各磁盘的合并进程获取到的数据文件大小进行汇总,计算数据文件大小的总和,作为是否启动数据文件合并过程的依据。在本实施例的实际使用场景中,各磁盘中的数据文件仅会在客户端每一次写入数据,并生成数据文件后产生变化,因此也仅需在客户端每一次写入数据并

生成数据文件后,对各磁盘合并进程获取到的数据文件大小的总和进行计算,并判断是否需要进行文件合并。

[0045] 步骤104:数据文件大小的总和达到合并文件阈值时,并行合并进程一次性合并所有磁盘中需合并的数据文件。

[0046] 为了减少文件合并次数,严格保证一份数据一次完成合并,需要指定合并后的目标文件大小,即合并文件阈值。数据库的并行合并线程在客户端每一次写入数据并生成数据文件后,对各磁盘合并进程获取到的数据文件大小的总和进行计算,动态判断各磁盘中的文件是否达到了合并文件阈值。一旦需合并的文件大小超过文件达到合并文件阈值,立刻启动该次数据写入所对应的数据文件合并过程,对生成的数据文件进行合并,并将合并后的文件移动至下一层。另一方面,若需合并的文件总大小未达到合并文件阈值,不进行本次合并,等待接受数据库下一次生成的数据文件,计算两次生成的数据文件大小总和是否达到合并文件阈值,若达到合并文件阈值再进行合并。在某些具体实施场景中,客户端每次写入的数据量较小,需要多次写入后,数据文件大小之和才能达到合并文件阈值,该场景中,需要进行多次等待后才会启动数据文件合并过程。

[0047] 通过步骤101-步骤104,在每一次接收到客户端写入的数据时,对各磁盘中生成的sstable进行扫描,对文件数据和合并文件阈值进行比较,仅在接收到的数据需合并时进行合并,不需合并时不进行合并,避免多次滚动合并,减少了磁盘的读写负担。

[0048] 在本实施例的具体实施方式中,为了确保合并的文件大小达到合并文件阈值,如图2所示,数据文件合并的过程可以通过以下步骤实现:

[0049] 步骤201:接受数据库生成的数据文件,生成各磁盘的合并文件列表。

[0050] 步骤202:各磁盘的合并进程获取相应磁盘的合并文件列表,获取各磁盘的合并文件列表中需合并的数据文件大小。

[0051] 步骤203:启动数据库的并行合并进程,计算各磁盘的合并进程获取到数据文件大小的总和。

[0052] 步骤204:判断需合并的数据文件大小的总和是否小于合并文件阈值。

[0053] 步骤205:若需合并的数据文件大小的总和小于合并文件阈值,等待接受数据库下一次生成的数据文件。

[0054] 步骤206:并行合并进程一次性合并所有磁盘中需合并的数据文件。

[0055] 其中,步骤201-步骤203对应步骤101-步骤103,步骤206对应步骤104。通过步骤201-步骤206,通过比较需合并的数据文件大小的总和与合并文件阈值的大小关系,动态决定是否启动数据文件合并过程,以减少文件合并的次数,并且合并后的文件必然达到了需要的大小,避免因合并后的数据文件过小导致的二次合并。

[0056] 进一步的,客户端每一次写入的数据大小若不为文件大小阈值的整数倍,生成的sstable按照文件大小阈值合并后,会产生小于文件大小阈值的文件,导致文件需要二次合并。若需合并的数据文件的大小超过文件大小阈值,按照文件大小阈值将需合并的数据文件合并为至少一个文件,剩余的小于文件大小阈值的部分不进行合并。在某个具体实施场景中,各磁盘的合并进程获取到的需合并的数据文件大小总和为2.6G,文件大小阈值为1G。此时,并行合并进程将需合并的数据文件合并为2个1G的文件,剩余0.6G文件不进行本次合并,等待下次接受数据库生成的数据库文件后进行合并。上述处理方式对需合并的文件进

行部分合并,既确保了每次合并后的文件大小都不小于文件大小阈值,又通过部分合并减少了需合并的数据文件在磁盘中的存留,节省了磁盘空间的占用。

[0057] 为了适应不同的实际使用场景,可以通过调整合并文件阈值,在文件数量和写入性能之间取得优化的平衡。如图3所示,具体步骤如下:

[0058] 步骤301:判断每个数据文件的大小是否小于文件数据量阈值。

[0059] 步骤302:若小于文件数据量阈值,合并文件阈值使用第一合并文件阈值。

[0060] 步骤303。若大于文件数据量阈值,合并文件阈值使用第二合并文件阈值,其中,第一合并文件阈值大于第二合并文件阈值。

[0061] 通过步骤301-步骤303,根据文件数据量阈值调整合并文件阈值的大小,以调节文件数量和写入性能。当接受到的业务文件数据量小于文件数据量阈值时,使用较大的第一合并文件阈值,将较多的小文件合并为一个大文件,将合并后的文件数量进一步减小;当接受到的业务数据量大于文件数据量阈值时,使用较小的第二合并文件阈值,虽然文件数增加,但可以优先保障写入性能,确保磁盘读写性能稳定,数据库业务平稳运行。在本实施例的具体实施方式中,每台数据服务器每天处理4TB原始数据,进行压缩后的数据为1.3TB,设置合并文件阈值为1G的情况下,合并后的文件数=1.3TB*1024=1331,业务的数据文件保存15天,总文件数=15*1331=19965个。合并后的文件个数完全在可控范围内数据写入,合并均在规定时间内平稳完成,数据库稳定运行。

[0062] 为了进一步确保数据库运行的稳定性,避免因文件合并异常导致的内存占用过多、查询时延增加等性能问题,本实施例中的小文件合并方法还包括了合并性能检测和告警机制。

[0063] 在本实施例的具体实施场景中,如图4所示,进行合并性能检测和告警的步骤如下:

[0064] 步骤401:并行合并进程统计合并时间。

[0065] 步骤402:判断合并时间是否超过合并时间阈值。

[0066] 步骤403:对合并时间超过合并时间阈值的磁盘进行告警。

[0067] 在实际使用中,在系统性能稳定的情况下,对文件进行合并的时间长度也较为稳定,若实际进行合并的时间远超过理论上进行合并的时间或历史合并的平均时间,则表明合并过程可能出现异常,需要进行异常处理。在具体实施场景中,可以通过估算理论上进行合并的时间或计算历史合并的平均时间确定合并时间阈值。

[0068] 为了及时处理发生异常的合并进程,在并行合并进程发现某个合并进程的合并时间超过合并时间阈值,即合并发生异常时,需要将发现的合并异常进行告警。在具体使用场景中,可以通过cassandra数据库的管理界面显示告警信息。为了避免与cassandra数据库的其它功能互相等待,及时显示告警信息,也可以使用专用的监控进程进行合并状态的显示和告警信息显示。为了使数据库的管理者和用户能够获取更详细的合并异常信息,还可以通过异常日志等方式将出现异常的合并线程的具体情况进行输出,提供具体的异常数据,以便于数据库管理者或用户进行处理。

[0069] 为了提高异常处理的自动化程度,也可以通过预设的异常处理程序对出现合并异常的线程进行自动处理。根据具体实施场景的需要,可以根据异常的具体数据进行针对性处理,也可以直接对数据库进程进行重启,以避免因合并异常导致的内存占用过多和查询

时延增加等性能问题,提高数据库运行的稳定性。

[0070] 在cassandra的存储管理机制中,为了减少磁盘IO的次数,使用了垃圾回收机制,对于需要删除的数据并不会直接删除,而是仅标记为删除,但仍保存在磁盘中,并在进行合并过程时进行标记为删除的数据进行统一的真正删除。本实施例中提供的小文件合并方法也使用垃圾回收机制对需要删除的数据进行删除,在进行合并时,将需合并的数据文件的源文件标记为删除,一次性合并所有需合并的数据文件后,将标记为删除的数据文件进行删除。通过该方式,可以避免对大量文件分别删除造成的多次磁盘IO,减少了磁盘负载,提高了合并效率。

[0071] 本实施例提供的基于cassandra数据库的文件合并的方法,通过对接收到的文件的大小动态监控,并在需合并的文件大小达到合并大小阈值时立即进行合并,在使用较少合并层次和临时文件的情况下及时对小文件进行合并,减少合并次数、减少磁盘中文件占用空间、减少磁盘IO次数和磁盘IO争抢,提高了文件合并的性能,提高了数据库的读写稳定性。

[0072] 与Size Tiered Compaction Strategy策略相比,本实施例提供的小文件合并方法每次合并后的文件都必然达到所需的大小,不会出现合并后的文件仍然很小的问题,不需要多次滚动合并,减少了磁盘IO抢夺,导致磁盘性能抖动和磁盘读写负担增加。

[0073] 与Leveled Compaction Strategy策略相比,本实施例提供的小文件合并方法对不同level的文件等同处理,不会导致最底层L0层的sstable堆积,需要单点管理的sstable数量较为稳定,达到合并标准的sstable能够及时进行合并,不会占用过多磁盘空间。

[0074] 与Time Window CompactionStrategy策略相比,本实施例提供的小文件合并方法对于接收到的数据文件大致上为顺序处理,不会存在因时间窗结束而放弃合并的情况,使得所有需合并的文件都能得到处理,避免了文件数量快速增加。进一步的,通过异常检测和告警机制处理发生异常的合并进程,确保系统的稳定性;通过调整合并文件阈值大小优化合并后文件数量,提高合并性能;通过使用垃圾回收机制进一步减少磁盘IO次数,提高合并效率。

[0075] 实施例2:

[0076] 在上述实施例1提供的基于cassandra数据库的文件合并的方法的基础上,本发明还提供了一种可用于实现上述方法的基于cassandra数据库的文件合并的装置,如图5所示,是本发明实施例的装置架构示意图。本实施例的基于cassandra数据库的文件合并的装置包括一个或多个处理器21以及存储器22。其中,图5中以一个处理器21为例。

[0077] 处理器21和存储器22可以通过总线或者其他方式连接,图5中以通过总线连接为例。

[0078] 存储器22作为一种基于cassandra数据库的文件合并方法非易失性计算机可读存储介质,可用于存储非易失性软件程序、非易失性计算机可执行程序以及模块,如实施例1中的基于cassandra数据库的文件合并方法。处理器21通过运行存储在存储器22中的非易失性软件程序、指令以及模块,从而执行基于cassandra数据库的文件合并的装置的各种功能应用以及数据处理,即实现实施例1的基于cassandra数据库的文件合并的方法。

[0079] 存储器22可以包括高速随机存取存储器,还可以包括非易失性存储器,例如至少一个磁盘存储器件、闪存器件、或其他非易失性固态存储器件。在一些实施例中,存储器22

可选包括相对于处理器21远程设置的存储器,这些远程存储器可以通过网络连接至处理器21。上述网络的实例包括但不限于互联网、企业内部网、局域网、移动通信网及其组合。

[0080] 程序指令/模块存储在存储器22中,当被一个或者多个处理器21执行时,执行上述实施例1中的基于cassandra数据库的文件合并的方法,例如,执行以上描述的图1-图4所示的各个步骤。

[0081] 本领域普通技术人员可以理解实施例的各种方法中的全部或部分步骤是可以通过程序来指令相关的硬件来完成,该程序可以存储于一计算机可读存储介质中,存储介质可以包括:只读存储器(ROM,Read Only Memory)、随机存取存储器(RAM,Random Access Memory)、磁盘或光盘等。

[0082] 以上所述仅为本发明的较佳实施例而已,并不用以限制本发明,凡在本发明的精神和原则之内所作的任何修改、等同替换和改进等,均应包含在本发明的保护范围之内。

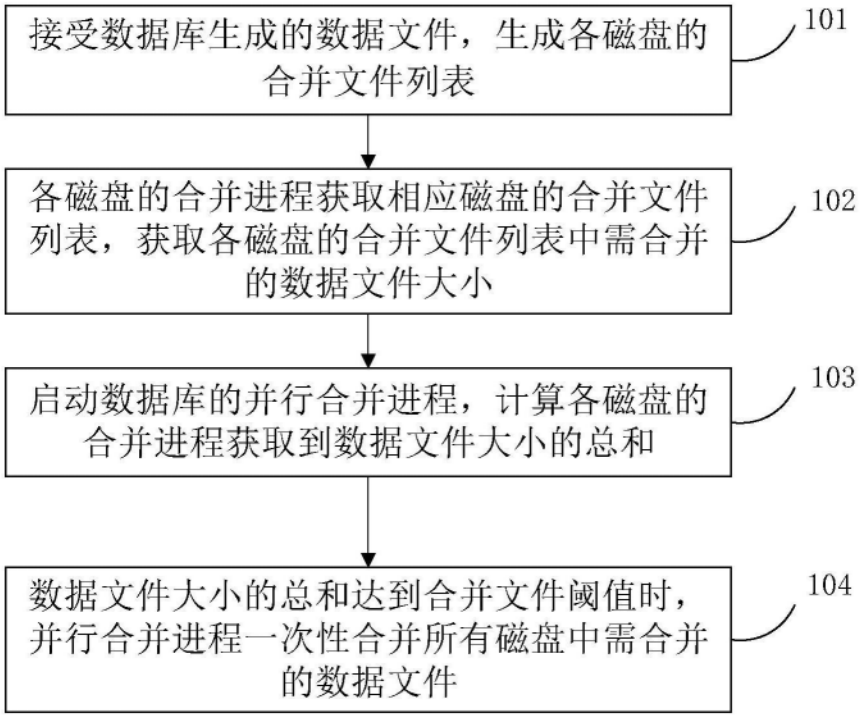

接受数据库生成的数据文件，生成各磁盘的合并文件列表
101
各磁盘的合并进程获取相应磁盘的合并文件列表，获取各磁盘的合并文件列表中需合并的数据文件大小
102
启动数据库的并行合并进程，计算各磁盘的合并进程获取到数据文件大小的总和
103
数据文件大小的总和达到合并文件阈值时，并行合并进程一次性合并所有磁盘中需合并的数据文件
104

图1

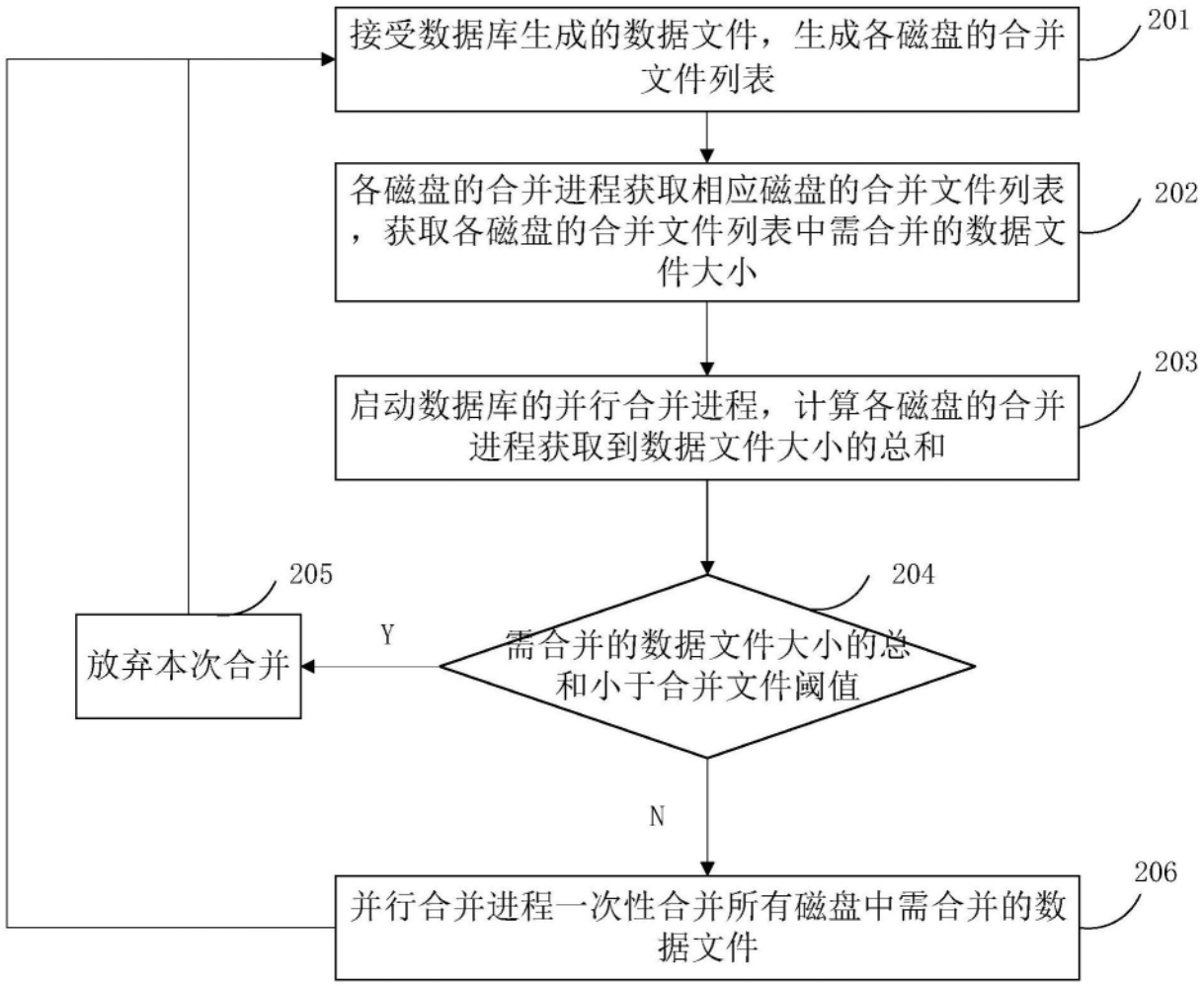

接受数据库生成的数据文件，生成各磁盘的合并文件列表
201
各磁盘的合并进程获取相应磁盘的合并文件列表，获取各磁盘的合并文件列表中需合并的数据文件大小
202
启动数据库的并行合并进程，计算各磁盘的合并进程获取到数据文件大小的总和
203
需合并的数据文件大小的总和小于合并文件阈值
204
Y
放弃本次合并
205
N
并行合并进程一次性合并所有磁盘中需合并的数据文件
206

图2

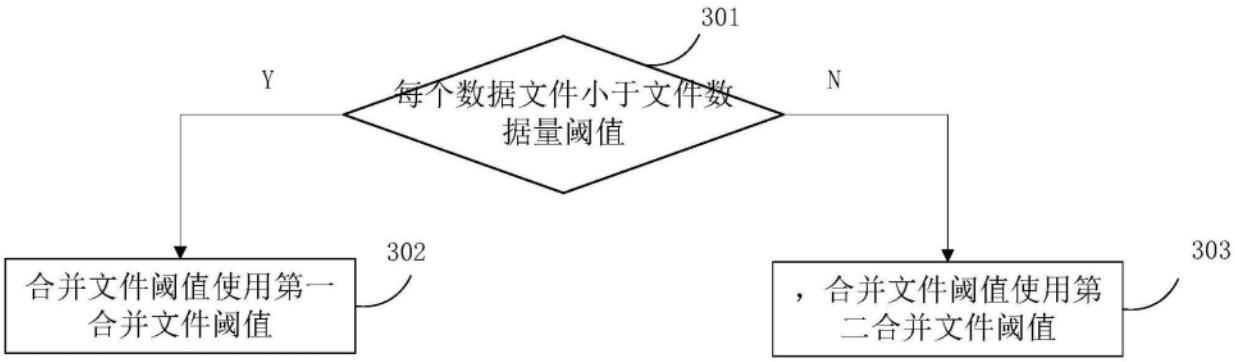

每个数据文件小于文件数据量阈值
301
Y
N
合并文件阈值使用第一合并文件阈值
302
，合并文件阈值使用第二合并文件阈值
303

图3

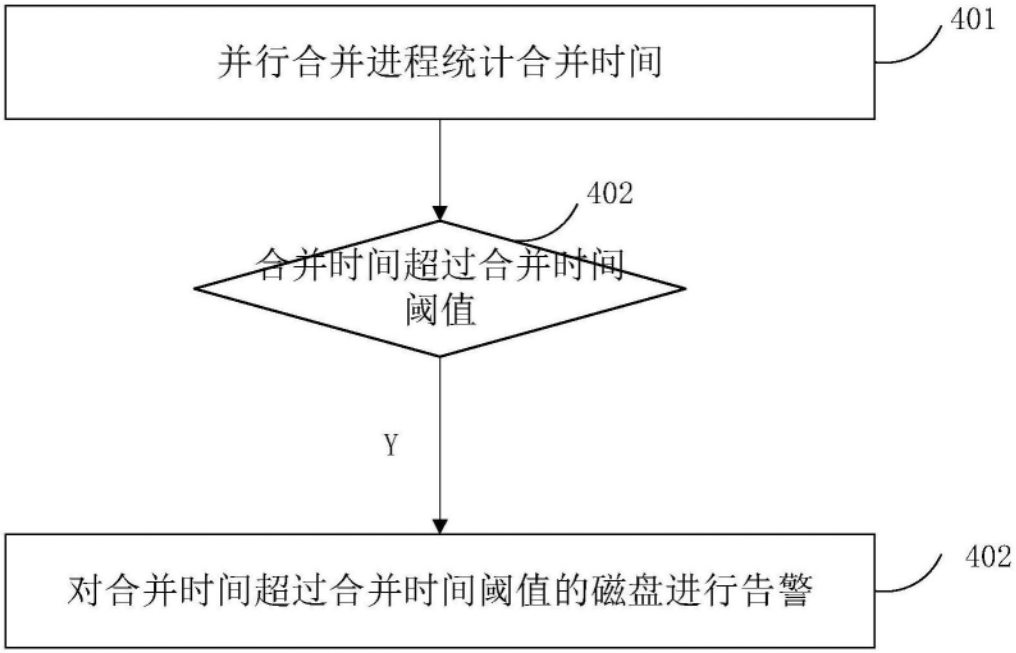

并行合并进程统计合并时间
401
402
合并时间超过合并时间阈值
Y
对合并时间超过合并时间阈值的磁盘进行告警
402

图4

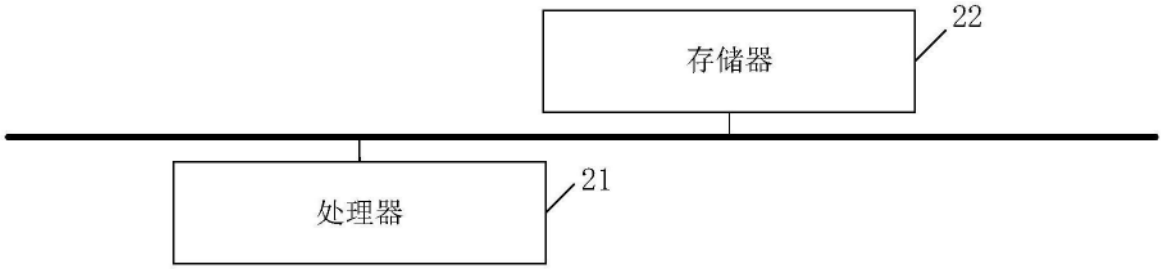

存储器
22
处理器
21

图5