

基于日志的数据热备份的实现

申新鹏¹, 李战怀², 赵晓南²

(1 西安工业大学 计算机科学与工程学院, 陕西 西安 710032; 2 西北工业大学 计算机学院, 陕西 西安 710072)

摘 要: 对于业务不能中断的系统来说, 热备份是保证数据可靠性的有效方式。但是传统的热备份系统, 备份文件较大, 不利于进行网络传输, 为此实现了一个增量热备份系统。当某个数据块发生变化时, 将文件修改点的位置信息和修改的内容记录到一个备份文件中, 然后将备份文件通过网络传递到其他地方。当需要恢复数据时, 以一次冷备份的结果为基础, 按照备份的先后顺序逐条恢复备份数据。使用这种方法, 可以在不停止工作流程的情况下, 实现数据文件的异地备份。

关键词: 网络传输; 热备份; 增量备份

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2011)04-0045-04

The Implement of Hot Backup Based on Logs

SHEN Xin-peng¹, LI Zhan-huai², ZHAO Xiao-nan²

(1 Computer Science and Engineering College, Xi'an Technological University, Xi'an 710032, China;

2 College of Computer, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: If the operation can not be interrupted in a system, the hot backup would be a effective backup method. To the traditional hot backup system, the backup document is large and it is difficult to transport it to other computer through the network. In order to improve it, the author implemented an incremental hot backup system. When a block of data is changes, the information of the location and the content of the amending point would be written in a backup file, then the backup file can be transported through the network. When it is necessary to restore data, based on the results of the cold, the data would be restored correctly using the backup files. Thus the backup would be performed safely without interrupting the operation of the system.

Key words: network transmission; hot backup; incremental backup

1 引言

随着企业信息化的推进, 对于已经高度电子化的现代企业来说, 运行过程中积累起来的大量数据资料是它们最可宝贵的财富^[1]。如果这些数据一旦丢失或损坏, 企业将遭受沉重的打击, 因此, 能够在有效地管理和利用这些数据的基础上保证它们的安全, 就变得越来越重要了^[2-3]。

传统的数据备份分为冷备份和热备份两种。冷备份是使用内置或外置的磁带机等设备在系统停止工作的基础上进行重要数据的复制。这种方法备份

数据量大, 且会造成业务的暂时停止。

对于业务不能中断的系统来说, 热备份是一种有效的备份方式。传统的热备份系统是以一个冷备份的结果为基础, 当文件的某个数据块发生变化时, 采用写前拷贝的办法同时修改备份文件和工作文件。这样备份文件和工作文件始终保持一致, 从而保证数据的完整性^[4-5]。这是一种全备份的方法, 备份文件较大, 给备份文件的管理造成一定的困难。

备份是为了在出现问题时进行数据恢复, 以保证业务的正常运行。如果将备份数据与原始数据保存在同一地点, 甚至是同一台机器上, 将很难达到预

期的效果. 例如在 CIH 等病毒侵害电脑时, 同一台电脑上的所有数据都将受到破坏. 在发生地震等自然灾害时, 处于同一位置的所有数据都将丢失.

因此, 必须使用一定的方式进行异地备份, 将备份结果通过网络传送到其他地理位置, 以提高数据的安全性. 但传统的数据备份方法产生的备份文件都很大, 不利于进行网络传输.

为此, 本文对传统的数据热备份方式进行了研究, 并借鉴了数据库系统在数据变化时, 记录日志的特点^[6], 提出了一种基于日志的数据热备份的方法来解决这个问题.

2 基于日志的热备份系统的设计

传统的数据备份方法产生的备份文件太大, 无法满足异地数据备份的要求, 必须寻找一种新的方法解决这个问题. 这种新的方法要求能够记录数据的所有变化, 并且备份文件的大小可以根据需要任意变化, 满足网络传输的要求.

数据库系统中的日志文件具有类似的特点. 数据库中的所有数据变化信息(变化的内容及修改用户等)都保存在日志文件中^[7]. 日志文件的大小由设定的参数决定, 可以进行更改. 使用日志文件可以将数据库的内容恢复到指定的状态^[8].

为此, 可以借鉴数据库日志文件的特性, 设计一个数据热备份系统, 达到预期的效果.

基于日志的数据热备份系统是以一个存在的冷备份结果为基础的, 通过将冷备份后数据的变化情况记录到日志文件中的方法来实现数据的完整备份. 这是一种增量备份的思想.

日志文件中记录了冷备份后所有的数据变化信息, 日志文件的大小由参数确定. 两次冷备份之间可能有多个日志文件. 数据恢复是以前一次的冷备份为基础的. 冷备份后的所有日志文件的大小决定了数据恢复的时间. 如果冷备份间隔时间太长, 需要恢复的日志文件很多, 将严重影响数据恢复的效率, 因此必须每隔一段时间(例如一个星期)进行一次冷备份操作. 然后将冷备份的结果和日志文件复制到其他地理位置, 以保证数据的安全.

以下以最常见的索引文件存储为例来介绍基于日志的数据热备份的思想. 索引文件存储是将一个文件的信息存放在若干不连续物理块中, 系统为每个文件建立一个专用数据结构(索引表), 表中的每一条信息记录了一个数据块所在的逻辑块号与与之对应的物理块号. 因此, 一个文件建立后跟踪它的索

引表就可以跟踪到文件中各数据块内容的变化情况.

基于日志的数据热备份系统使用的数据结构如图 1 所示.

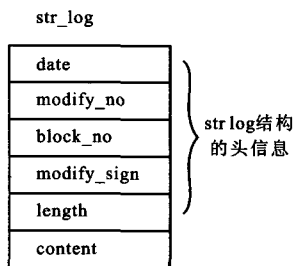


图 1 数据结构图

其中, date 是写 log 的时间, modify_no 是该 log 的序号; block_no 是内容发生变化的数据块的逻辑块号; modify_sign 是修改标志, 可能的取值有 1: 修改逻辑块中的内容, 2: 删除逻辑块, 3: 新增逻辑块; length 是数据块内容的长度(数据备份和恢复可能不在同一台计算机上进行, 不同的系统使用的数据块大小可能不同, 因此, 必须记录备份时数据块的大小); content 是逻辑块中最新的内容(二进制字符流), 若是删除逻辑块 content 的内容全为 0.

3 基于日志的热备份系统的实现

下面简单介绍一下索引文件使用基于日志的数据热备份系统进行数据备份和恢复的过程.

启动备份系统时, 设置两个全局变量 g_length 和 g_modify_no. 其中 g_length 等于操作系统使用的数据块的大小. 对全局变量 g_modify_no, 初始值设为 0, 最大值为 MAX_MODIFY_NO (取值可以根据需要设定, 将决定每个日志文件中日志的条数, 从而影响日志文件的大小).

当有数据块的内容发生变化时, $g_modify_no = (g_modify_no + 1) \% MAX_MODIFY_NO$, 即 g_modify_no 每次加 1, 当 g_modify_no 的值达到最大值(MAX_MODIFY_NO)后, 重新从 0 开始取值.

数据文件内容的变化主要有修改、删除、增加三种操作. 热备份系统对这三种操作分别设计了不同的算法来记录日志.

(1) 修改一个数据块(块号是 B_m)

要修改一个数据块 B_m 的内容时, 进行如下工作:

- ① 设置 str_log. date 等于系统时间,
str_log. modify_no = g_modify_no,
str_log. block_no = B_m ,

str_log.modify_no = 1(修改逻辑块的内容),
str_log.length = g_length,
str_log.content 的内容为该数据块的新的内容
(二进制字符流).

② 将 str_log 中的内容输出到 log 文件中.

③ 修改 g_modify_no 的值:

$g_modify_no = (g_modify_no + 1) \% MAX_MODIFY_NO$.

④ 真正修改工作文件的数据块 B_m 的内容.

(2) 删除一个数据块(块号是 B_d)

要删除一个数据块 B_d 时,进行如下工作:

① 设置 str_log.date 等于系统时间,

str_log.modify_no = g_modify_no,

str_log.block_no = B_d ,

str_log.modify_no = 2(删除一个逻辑块),

str_log.length = g_length,

str_log.content 的内容全部置为 0.

② 将 str_log 中的内容输出到 log 文件中.

③ 修改 g_modify_no 的值:

$g_modify_no = (g_modify_no + 1) \% MAX_MODIFY_NO$.

④ 真正删除工作文件的数据块 B_d .

(3) 新增一个逻辑块(块号是 B_n)

要新增一个数据块 B_n 时,进行如下工作:

① 设置 str_log.date 等于系统时间,

str_log.modify_no = g_modify_no,

str_log.block_no = B_n ,

str_log.modify_no = 3(新增逻辑块),

str_log.length = g_length,

str_log.content 的内容为新块的内容(二进制字符流).

② 将 str_log 中的内容输出到 log 文件中.

③ 修改 g_modify_no 的值:

$g_modify_no = (g_modify_no + 1) \% MAX_MODIFY_NO$.

④ 真正在工作文件中新增数据块 B_n .

(4) 恢复文件

使用上述算法记录数据变化的日志后,在需要的时候,可以结合最近的一次冷备份结果和日志文件实现数据的恢复.

在进行上面的写 log 的过程中,可以设定 log 的保存方式,一种是一条 log 对应一个 log 文件,文件名为 str_log.date 和 str_log.modify_no 的组合,一种是指定 log 文件的最大值,写 log 时判断 log 文

件的大小,若 log 文件已经超过了最大值,就新建一个 log 文件,新的 log 文件的文件名就是当前这条 log 的 str_log.date 和 str_log.modify_no 的组合.通过这种命名体系产生的 log 文件很容易排序.

恢复文件时,需要一次冷备份的内容及其以后的所有日志文件.进行恢复时,先将日志文件排序,然后按如下流程恢复文件:

① 设置全局变量 g_modify_no = 0.

② 打开第一个日志文件.读取其中第一条记录的 str_log 结构的头信息,设置全局变量 g_length 等于 str_log.length,即原来的操作系统使用的数据块的大小.

③ 读取一个 str_log 结构,判断 str_log.modify_no 是否等于 g_modify_no,若不等,出错退出,若相等,判断 str_log.modify 的值:

- 若 str_log.modify 为 1,表示要修改一个逻辑块,则根据 str_log.block_no(修改的数据块的逻辑块号)到备份文件的文件索引表中找到该逻辑块对应的物理地址,然后 str_log.content 中的内容复制到该物理地址对应的数据块中.

- 若 str_log.modify 为 2,表示要删除一个逻辑块,则根据 str_log.block_no(修改的数据块的逻辑块号)到备份文件的文件索引表中找到该逻辑块对应的物理地址,释放该地址对应的磁盘空间,然后从文件索引表中删除该逻辑块,若备份和恢复是在不同的操作系统上进行的,要注意区分两个系统删除一个逻辑块后的工作是否相同,即是否都会自动修改该逻辑块后所有逻辑块的块号,若处理方式不相同,要进行相应的处理保证逻辑块号不发生混乱.

- 若 str_log.modify 为 3,表示要新增一个逻辑块,先在文件索引表中查找该逻辑块,若能找到,异常退出,否则就新申请一个磁盘块,将 str_log.content 中的内容写入该块中,然后将该逻辑块和物理块之间的对应关系写入文件索引表.

④ 修改 g_modify_no 的值:

$g_modify_no = (g_modify_no + 1) \% MAX_MODIFY_NO$

⑤ 读取下一个 str_log 结构,重复第③、④步的处理.

⑥ 若读到了日志文件的结束标志,就关闭这个日志文件,打开下一个日志文件,重复第③~⑤步的处理.

⑦ 若所有的日志文件都已经恢复完毕,就关闭所有文件,恢复成功.若要恢复到某个时间点,就在

每次读取 str_log 结构后,先判断 str_log.date 是否大于指定的时间,若小于该时间点,就按照上面的流程恢复数据,若大于该时间点,就说明恢复已经完成,成功退出。

4 结束语

通过以上的分析可以看出基于日志文件的热备份方法具有如下优点:

(1) 它是以二进制字符流的形式按数据块来进行备份的,因此每一条日志与文件真实的数据结构之间没有关系,难以破译。以数据库文件为例,一个数据块可能包括一个表中的多条记录,也有可能仅包括半条记录,甚至也可能包括一个完整的小表。因此备份内容可读性很差,安全性很高。

(2) 该算法是以日志为基础实现的,日志文件的大小可以任意控制,可以方便的实现异地实时备份。要备份的文件每有一处改动,就生成一个日志文件,然后立即通过网络传输到异地的其他机器上,这样,无论何时这台机器出现故障,都可以通过备份机器上的数据进行恢复。

(3) 恢复点是任意的。该算法备份了任意时刻的中间数据,并且它的恢复完全模拟了系统修改数据的全过程,因此,可以恢复数据到任意时刻。这对于安全性要求很高的数据是很有必要的,因为它可以追查对数据的所有改动。

(4) 该算法经过的改进后,可以跨平台,跨文件实现方式应用。不同系统的文件块的大小可能不同,但是经过计算可以找到其中的对应关系,并将 str_

log.content 的内容分几次进行复制就可以实现。一种病毒一般是针对某种操作系统下的文件才会发作,因此实现跨平台备份后,可以更有效地保证数据的安全。

参考文献:

- [1] Farley M. SAN 存储区域网络[M]. 孙功星,蒋文保,范勇,等译. 北京:机械工业出版社,2002.
- [2] 窦万峰,朱鸣,陈卉. 实时协同编辑中群组可靠性和容错研究[J]. 微电子学与计算机,2007,24(9):95-97.
- [3] 刘茂诚,刘展,袁海兰,等. 数据备份系统的关键参数估计[J]. 计算机工程,2008,34(13):70-71,74.
- [4] 高航,罗宇. RTEMS 上基于软件重算的容错支撑设计研究[J]. 微电子学与计算机,2008,25(10):151-154.
- [5] 颜洪梅,战守义,杨方廷. 网络系统中数据备份技术的研究[J]. 计算机应用研究,2006,23(12):154-156.
- [6] 范丽雅,王万森,关永. 数据库中数据恢复的研究[J]. 计算机工程与设计,2008,29(3):563-565.
- [7] 贺亚茹. Oracle 数据库日志文件损坏时修复方法的实验研究[J]. 计算机应用,2009,29(12):393-395.
- [8] 廖国琼,刘云生. 基于实时日志的嵌入式实时数据库恢复策略[J]. 计算机学报,2007,30(4):672-679.

作者简介:

申新鹏 男,(1975-),讲师,博士.研究方向为对等计算、网络计算。

李战怀 男,(1961-),教授,博士生导师.研究方向为数据库、数据挖掘。

赵晓南 女,(1979-),博士研究生.研究方向为海量数据存储、数据管理。

(上接第 44 页)

的快速部署,未来可以对目前的散体文件存储方式进行改进,使用海量图片打包方式,解决海量栅格图片的快速部署问题。

参考文献:

- [1] 彭纪奎,胡平. 基于 Web 的空间数据可视化研究[J]. 微电子学与计算机,2010,27(1):147-150.
- [2] Joe Schwartz. Bing Maps Tile System[EB/OL]. (2010-06-22)[2010-08-12]. <http://msdn.microsoft.com/en-us/library/bb259689.aspx>.
- [3] OSGEO. Tile Map Service Specification[EB/OL]. (2010

-03-06)[2010-08-12]. http://wiki.osgeo.org/wiki/Tile_Map_Service_Specification.

- [4] 朱玉周,杨芳,魏璐,等. 瓦片网络地图技术在河南交通气象服务系统中的应用[C]// 2007 年中国气象学会年会. 广州:中国气象学会,2007.

- [5] 胡双双,秦杰,谢惠. 一种实用元搜索引擎的设计与实现[J]. 微电子学与计算机,2009,26(4):67-69.

作者简介:

李晓斌 男,(1971-),博士研究生.研究方向为智能交通、交通诱导。