基于数据块的关系数据库日志挖掘技术①

彭 巍[1], 邵佳炜[1], 雷振江[2], 吕旭明[2], 聂庆节[3], 刘 赛[3]

[1](国网上海市电力公司, 上海 200122)
[2](国网辽宁省电力有限公司, 沈阳 110004)
[3](国网电力科学研究院, 南京 210003)

摘 要: 在关系型数据库中数据库通过 Redo 日志来实现事物的快速提交, 并记录事物的操作过程与操作内容. 通过对 Redo 日志的分析与变化数据内容的捕获, 将变化数据传送到灾备端, 并在灾备端实现变化数据的写入, 是目前数据库复制最主要实现原理. 本文分析了 oralce 数据库 Redo 日志文件结构, 阐述了日志文件头标志位信息. 结合 Redo 日志文件头定位分析技术, 给出了一种基于数据块的数据库 Redo 日志挖掘算法. 通过测试分析, 验证了该 Redo 日志挖掘技术的可行性与可靠性. 最后展望了下一步的研究方向.

关键词: Redo 日志; 日志分析; 日志定位; 日志挖掘; 数据库复制

Relational Database RedoLog Mining Technology Based on Data Block

PENG Wei[1], SHAO Jia-Wei[1], LEI Zhen-Jiang[2], LV Xu-Ming[2], NIE Qing-Jie[3], LIU Sai[3]

[1](Shanghai Municipal Electric Power Company, Shanghai 200122, China)
[2](State Grid Liaoning Electric Power Co.Ltd, Shenyang 110004, China)
[3](State Grid Electric Power Research Institute, Nanjing 210003, China)

Abstract: The relational database can submit objects rapidly by means of the Redo log and also can record the operation process and the operation content. The most important method to realize database replication is transmitting and writing changed data in the disaster backup end through the analysis of the Redo log and the capturing of the changed data contents. The structure of Oracle Redo log file database is analyzed in this paper, and the header information of the log file is also stated. Considering the technology of Redo log file head positioning, the paper presents a Redo log database mining algorithm based on data block. The paper also validates the feasibility and reliability of the technology of the Redo log mining through the comparative analysis. Research direction is discussed at last.

Key words: redo log; log analysis; log location; log mining; database replication

国家电网公司信息系统灾备中心的建设是对灾害的超前预防、对信息化的前瞻性管理, 体现了公司信息化建设理念的进步和发展[1-2]. 在公司数据级灾备建设中, 采用了数据库复制与存储复制技术实现信息系统的异地灾备, 数据库复制技术使用了 Oracle Goldengate 关系数据库数据复制技术, 存储复制技术使用的是高端阵列持续复制技术[3]. 其中, 关系数据库复制技术灾备范围涉及国家电网公司多家单位的重要业务应用. 从灾备覆盖网省和业务系统范围、关键程度与数据量等方面来看, 关系数据库复制技术已是为国家电网灾备建设的关键技术[4], 而关系数据库复制技术的核心是数据库在线日志挖掘技术[5].

当前电力系统应用最广泛的是基于逻辑级日志抽取方式实现日志挖掘, 这类日志抽取方式对主备两端数据一致性与完整性无法有效保障[6,7]. 同时, 由于数据库的日志挖掘技术一直是各数据库厂商的核心技术之一, 在此类技术上壁垒严重, 鲜有技术细节披露[8]. 针对上述问题, 更加说明研究日志挖掘技术的值得深入研究探讨.

① 基金项目:国家电网公司总部科技项目
收稿时间:2015-10-27;收到修改稿时间:2015-11-27 [doi: 10.15888/j.cnki.csa.005209]

1 oracle数据库日志文件头分析

1.1 Redo 日志文件结构

Oracle 在线日志文件由日志块组成, 日志块大小会在日志文件中的 21、22 字中标识. DBA 可以根据需要设定文件的块大小, 块大小又会因操作系统的不同而不同. 在 Windows、Linux、Solaris, AIX 上, 块大小默认为 512byte, 而在 HP-UX 上, 块大小默认为 1024byte. Redolog 文件头及 Redolog 头占据了 Redolog 文件的一个块. 文件头之后, 是按照顺序写入的 Redo record. 一个 Redo record 记录的是一个原子操作 Redo.

redo record 的结构是由一个 redo record 头记录加上多个 change vector 组成. 一个 change vector 就是对一个 block 的一次修改 Redo. 一个原子操作可能包含对相关的几个 block 的修改, 例如 data block、undo block、undo header block 等. 关系型数据库 Redo 格式如下:

block0 文件头	block1 Redo 头	block2 Redo 记录 1	block3 Redo 记录 2	...	block n Redo 记录 N

Redo 记录(Record)由记录头以及多个操作(Change)组成. 每一个记录组成如下所示.

Redo Record Header	Change #1	Change #2	Change #3	….	Change #N

日志文件通常使用操作系统的块大小, 一般为 512bytes, Redo Logs 循环写. 日志格式依赖于操作系统与数据库版本, 一般是由日志文件头(Redo Logs Header)以及后面的日志记录(Redo Record)组成, 日志记录同样由记录头(Recod Header)以及多个日志操作(Redo Change)组成; 数据库的许多变化都放在记录(Record)中, 记录可以大于一个块, 也可以小于一个块, 记录的大小和数据库的块大小有关, 默认情况下最大的记录是三个数据块. 记录是Redo日志中表示数据库操作的一个原子单位. 每个 Redo 记录包含每个 Change 所需的 UNDO 和 REDO.

一个 Change 是 Record 中的一个操作信息, 一条记录可以记录多个操作, 这些操作是密不可分的. 如在记录更新时, 首先要标识一个事务开始, 其次要记录回滚段中原来的数据, 第三要记录真实更新的数据, 这三个操作组成了一个日志文件中的记录.

1.2 Redo 日志文件头信息

日志文件头记录了日志文件众多的信息, 不同平台, 不同版本的 oracle 日志头字节位置可能不同[9], 经过多年经验总结 linux 平台上 oracle10.2.0.5 版本的数据库文件头标志位信息如下.

表 1 文件头标志位信息表

日志文件头偏移量	说明
04-07	Block id
08-11	Log sequence
14-15	Disk checksum
20-23	Compatibility vsn
24-27	Db id
28-35	Dn name
36-39	Control seq
40-43	File size
44-47	Block size
48-49	Log file group number
50-51	Log file type
52-55	Activation ID
92-155	Threan 描述
156-159	Nab
160-163	Resetlog id
164-167	Resetlogs change
168-169	Resetlogs change wrap
172-175	Hws
176-177	Thread
180-183	Low scn
184-185	Low scn wrap
192-195	Next scn
196-197	Scn wrap
208-211	Enabled scn
212-213	Enable scn wrap
220-223	Thread closed scn
224-225	Close scn wrap
236-239	在线日志:0x00000000 归档日志:0x11000000
268-271	Largest LWN
284-287	Prev resetlogs scn
292-295	Prev resetlogs count

1.3 Redo block header

在 Linux 操作系统环境下, 通过 BBED[10]工具 DUMP 日志文件块查看 oracle 内部日志块内容如图 1.

通过实践分析总结, 图 1 中前 16bytes(01220000 02000000 33000000 1080d6f6)为 block header, 其中前八位(01220000)为 redo record signature, 后续字段描述信息如下表 2 所示.

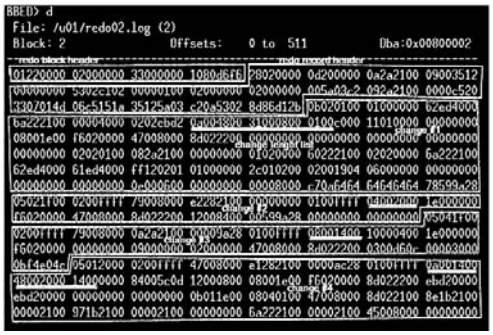

图 1 Redo block header

表 2 日志偏移信息表

日志偏移名称	二进制	字节数
redo log block number	00000002	4 个字节
redo log sequence number	00000033	4 个字节
redo log offset	0x10	占据 1 个字节

上述表 1 中的三部分合起来是 RBA:0x000033.00000002.0010, 表中 offset 指出了当前 block 中第一个 redo record 开始的位置(0x10), 即此 block 里第一个 redo record 开始于第 16 个字节, 也就是图 1 中的 28020000. 另外, check sum 校验和 f6d6 占据 2 个字节.

1.4 **Redo record header**

Redo record header 是一个固定的结构如图 2 所示. 第一个 redo record 的开始位置可以根据上述 block header 中的 RBA 的 offset 得到. 每个 redo record 以 redo record length 开始, 由于每个 record 的长度不是固定的, 将长度放在最前面, 就可以快速的的根据长度来查找到下一个 record 的位置.

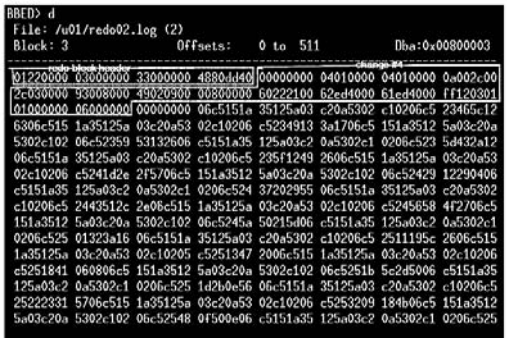

图 2 Redo block header

从上图 2 可以得知这个 block RBA 的 offset 位于第 12 个 bytes, 占据 1 个字节, 即 offset = 0x48=72bytes. 根据这个可以找到第 3 个 block 中第一个 record 的开头, 即 00000000, 第一个 change 的长度是 0, 说明这是一个无用的块. 同时表明 block header 和第 72byte 之间的这 56 个字节是不属于当前 block.

2 在线日志挖掘技术

在线日志挖掘技术主要涉及到获取数据库日志信息、在线日志挖掘流程、在线日志挖掘算法等内容, 下面对上述内容作详细的阐述.

2.1 在线日志挖掘对表的调用关系

在线日志数据块挖掘前需要获取数据库在线日志信息, 如数据库系统数据块大小、重置日志 ID、thread 号、在线日志组数量、在线日志所在路径等. 操作进程在 DBA 权限下从数据库视图表 v$archived_log、v$log、v$logfile、v$asm_disk 中获取上述相关信息. 在线日志挖掘对数据库表的具体调用关系如表 3 所示.

表 3 在线日志挖掘对数据库表的关系

表名称	操作权限	关键操作栏位
v$archived_log	dba	block_size,resetlogs_id
v$log	dba	thread
v$logfile	dba	group, type
v$asm_disk	dba	path

2.2 在线日志挖掘流程描述

关系数据库 oracle 运行时会有多个日志组文件, 数据库管理系统定时让某个日志组处于在线状态, 日志组状态变化时数据库对应的 sequence 号会随着改变. 当某一个日志组处于在线状态时, 挖掘进程实时捕获在线日志组变化信息, 并将挖掘到的变化数据块按照规则[thread 号+sequence 号+resetlogs_id+数据块编号]命名后存入到文件目录中, 在当前在线日志组即将切换到另外一个日志组时, 最后一次挖掘的变化数据块命名为[thread 号+sequence 号+resetlogs_id+final], 即结束此 sequence 号的数据块挖掘, 挖掘进程开始对新的在线日志组开始日志挖掘, 相应的 sequence 号也会自动加 1. 在线日志挖掘流程逻辑如图 3 所示.

主要流程描述如下:

① 从生产库获取当前数据库系统数据块大小;

② 获取生产库一条在线日志全路径名, 根据全路径名判断是存储在文件系统中还是 ASM 上;

③ 如果日志文件存在 ASM 上, 则从表 v$asm_disk 中取得路径;

④ 访问生产库, 从 vlog,vlogfile 中获取当前日

志组信息, 获取操作信息, 获取当前系统的数据块大小;

⑤ 根据当前在线日志的格式, 启动多个子进程, 对每一个在线日志进行单独进程挖掘;

⑥ 挖掘日志文件头信息, 比对 thread 和 dbid, 是否和生产库一致, 如果不一致, 则返回错误;

⑦ 从日志文件头中获取当前的日志序列 curSeq, 检查日志序列是否存在, 不存在则返回错误;

⑧ 检查当前日志文件头中序列是否和之前获取的 curSeq 一致, 如果不一致, 返回错误信息, 终止当前日志挖掘, 等待归档流程处理, 开始挖掘新的日志文件. 出现上述错误表明在线日志挖掘速度小于在线日志组切换速度, 正常情况下在线日志挖掘速度和数据库写日志速度等同, 而在线日志组切换的速度远小于数据库写日志的速度, 即步骤 8 中的错误基本上不存在. 另外, 在极端情况下出现上述情况后, 可以通过从数据库生成的归档文件中获取遗漏的数据库日志文件, 获取归档日志的方法此处不作详述.

⑨ 开始挖掘日志文件头后的数据块, 每次挖掘 1M 数据, 记录当前挖掘的截止位置 index, 将数据块内容写入新建的临时文件, 临时文件名根据业务系统 id, 在线日志组号, sequence 以及当前数据块编号进行命名; 如果获取失败则返回错误信息, 终止当前挖掘;

⑩ 从文件头获取文件结束标志, 检查是否已经挖掘结束, 如果没有结束, 则跳到步骤 5, 挖掘下一数据块; 如果文件已经结束, 重新挖掘日志文件头, 写入日志头数据块临时文件, 存入结束标志, 结束当前文件挖掘, 开始挖掘下一日志文件.

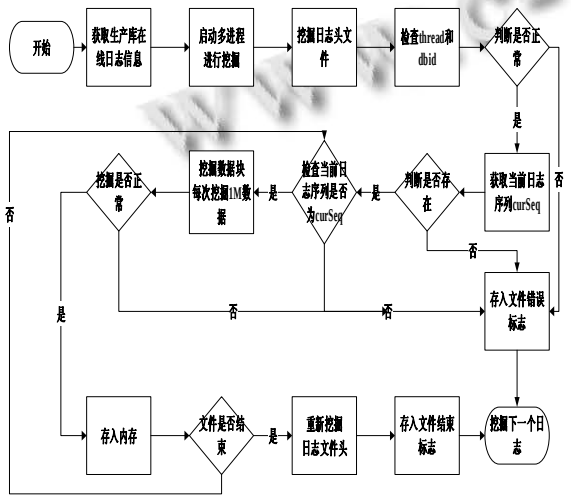

图 3　在线日志组数据挖掘流程

2.3 在线日志挖掘算法描述

在线日志挖掘算法描述如下:

① 初始化:设定当前打开日志文件的 sequence 为 cur_Sequence; 读取在线日志文件头第 8-11 个字节内容为 online_Sequence; 当前挖掘的截止位置为 index; 数据块大小为 blockSize; 日志文件头第 24-27 字节为日志文件数据块数 redo_blockSize; 日志文件头第 236-239 字节为日志类型 redo_Type; 数据库版本为 version; 日志头总大小为 redo_Size; 分割后的日志头字段为 R_i; 校验参数 C 初始值等于 0;

② WHILE 挖掘进程数小于等于日志组数
　　生成新的挖掘子进程
END WHILE

③ IF cur_Sequence 不等于 online_Sequence
　　当前在线日志文件写入较快, 文件已经被新内容覆盖, 动态调整挖掘文件速度.
END IF

④ IF 日志文件块挖掘结束
　　redo_blockSize = index / blockSize – 1;
END IF

⑤ IF version 等于 oracle 10g
　　redo_Type= 17
　else if version 等于 11g
　　redo_Type=8388625
END IF

⑥ 日志文件预处理
(R_1、R_2、$R_3 \ldots R_{n-1}$)redo_Size/64;
R_n= redo_Size%64;

⑦ WHILE R_i 等于 64 字节
　R_i 分成 4 个 16 字节的(R_{i1}、R_{i2}、R_{i3}、R_{i4})
　A = $R_{i1} \wedge R_{i2}$;
　B = $R_{i3} \wedge R_{i4}$;
　C=A∧C;
　C=C∧B;
END WHILE

⑧ 16 字节的 C 分为 4 个 4 字节段 r1、r2、r3、r4, 将上述四个值分别按位异或得到值 r0, 将 r0 按位右移 16 位得到 g,将 r0 与 g 进行异或得到 m, 将 m 与 0xFFFF 按位与得到最终的校验和;

r0=r0^r1;
r0=r0^r2;

```
r0=r0^r3;
r0=r0^r4;
r1=r0;
r0=r0>>16;
r0=r0^r1;
r0=r0&0xFFFF;
return r0;
```

⑨ 将计算的校验和 r0 重写到日志头的第 14-15 字节，规定挖掘到的数据块命名方式:thread+sequence+ resetlogs_id+数据块编号;

⑩ 继续在线日志挖掘.

3 测试结果与分析

为验证在线日志挖掘的可行性与可靠性，搭建如图 4 所示的测试环境. 本文设计的在线日志挖掘算法应用到自主研发的数据库数据复制产品中，测试软件系统架构由两部分组成，包括数据库复制层与数据库复制管理层. 其中数据库复制层由数据库复制工具组成，复制工具分别部署在源端数据库服务器(32 核 128GB 内存、oracle11g)与灾备端数据库服务器(32 核 128GB 内存、oracle11g)；数据库复制管理层由数据库复制管理工具组成，复制管理工具部署在管理库服务器(16 核 64GB 内存、oracle11g)与应用服务器(16 核 64GB 内存). 测试环境中网络带宽 100Mb/s，其余测试参数不再详述.

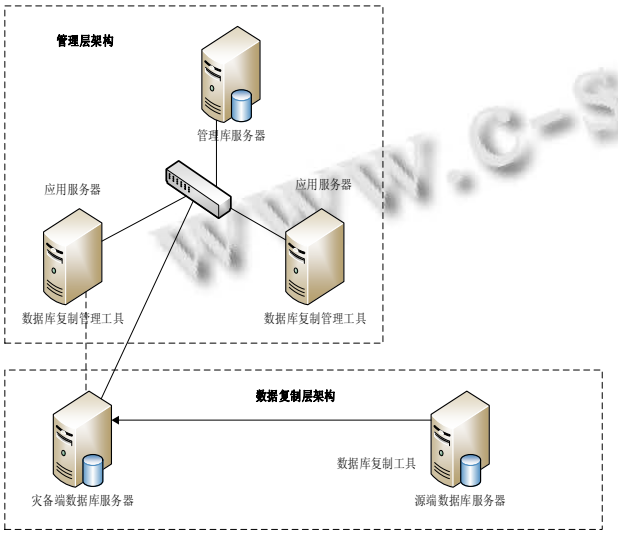

图 4 测试环境拓扑

测试生产端数据库分别每秒生成 0.5M、1M、5M 数据的情况下，挖掘出一个数据块所需时间如图 5 所示，从图 5 可以看出在生成数据量较小时挖掘一个数据块所需时间平均 1.6 微秒左右，生成数据量大于等于 1M 时挖掘一个数据块所需时间基本没有变化，表明整个挖掘过程基本处于稳定的状态.

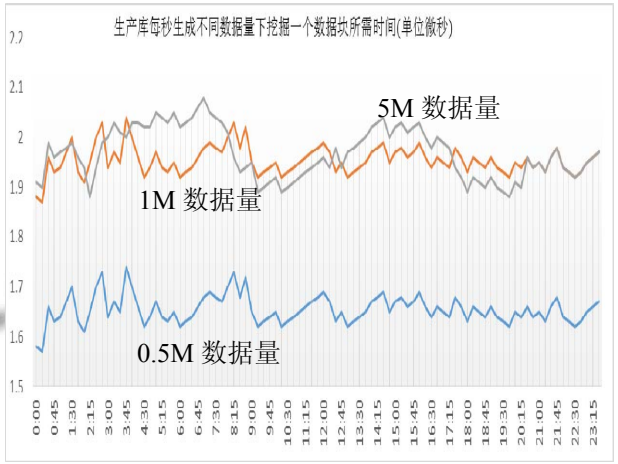

图 5 挖掘一个数据块所需时间

为进一步验证日志挖掘的可行性，某一时间段从生产库在线日志组中挖掘到同 sequence 号的数据块文件列表如图 6 所示. 从图 6(1)中可以看出 thread 号为 1、sequence 号为 18462、resetlogs_id 为 857297259 的日志文件存放在 59 个数据块中，图 6(2)是其中一个文件，可以看出即使数据块文件被截获，也不会造成数据泄密.

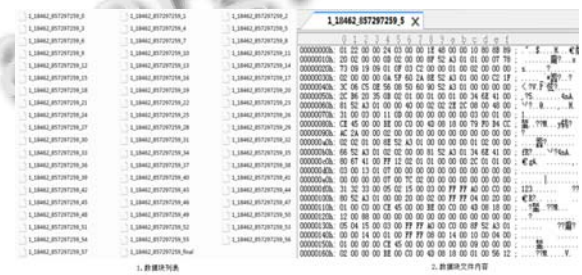

图 6 数据库复制延时对比

为验证挖掘到的数据块的可用性，在测试环境灾备端把挖掘到的同 sequence 号的数据块合成日志文件，图 6(1)中的数据块文件，sequence 号为 18462 的 1_18462_857297259.dbf 归档日志文件. 把合成的归档日志文件注册应用到灾备库，查询灾备库发现 sequence 号为 18462 的归档已存在 v$archived_log 视图中，进而验证了日志挖掘技术的有效性.

4 结语

数据库在线日志挖掘技术是信息灾备领域中的关键技术, 当前业界主要是以 goldenGate 为代表的逻辑级数据库日志抽取产品. 基于数据块级别的数据库在线日志挖掘技术支持所有数据库对象, 不但可以从根本上保障生产端与灾备端的数据的一致性、完整性与有效性, 还可以消除灾备信息安全隐患. 下一步我们将优化在线日志挖掘算法, 完善不同数据库版本下的日志挖掘技术, 更好的支持智能电网建设.

参考文献

1 朱征,顾中坚,吴金龙,等.云计算在电力系统数据灾备业务中的应用研究.电网技术,2012,9(9):43−50.

2 岳峻松,张磊,聂庆节,等.灾备数据恢复的验证方法. 电力信息与通信技术,2013,2(2):26−30.[doi:10.3969/j.issn.1672-4844.2013.02.009.]

3 吴金龙,马悦皎.基于 GoldenGate 数据库复制技术的容灾虚拟化解决方案.电力信息与通信技术,2012,10:52−56.[doi:10.3969/j.issn.1672-4844.2012.10.009.]

4 胡扬波,陈咏秋,周红林.服务器日志挖掘在电力业务系统功能推荐中的应用.计算机系统应用,2015,24(3):256−259.

5 陈三川,吴国全,魏峻,等.基于日志挖掘的移动应用用户访问模型建模技术研究.计算机科学,2014,(11):25−30.[doi:10.11896/j.issn.1002-137X.2014.11.006.]

6 邹先霞,贾维嘉,潘久辉.基于数据库日志的变化数据捕获研究.小型微型计算机系统,2012,(3):531−536.[doi:10.3969/j.issn.1000-1220.2012.03.017.]

7 孙昆,张琍,王子亨,等.多级多域视频监控系统数据同步机制的设计.计算机应用, 2015.

8 李芳,陈勇,张松树,等.大电网统一数据库建设相关技术研究.电网技术,2013,37(2):417−424.

9 马友忠,孟小峰.云数据管理索引技术研究.软件学报,2015,26:145−166.[doi:10.13328/j.cnki.jos.004688.]

10 陈飞.Oracle 数据库块损坏的恢复——浅析 BBED 在数据库恢复中的应用.计算机光盘软件与应用,2011,21:37−38.