浅析undo

——分享技术　分享快乐

张记

Phone：15628888210

QQ：771919563

Email ：smilejiodb@gmail.com

议题

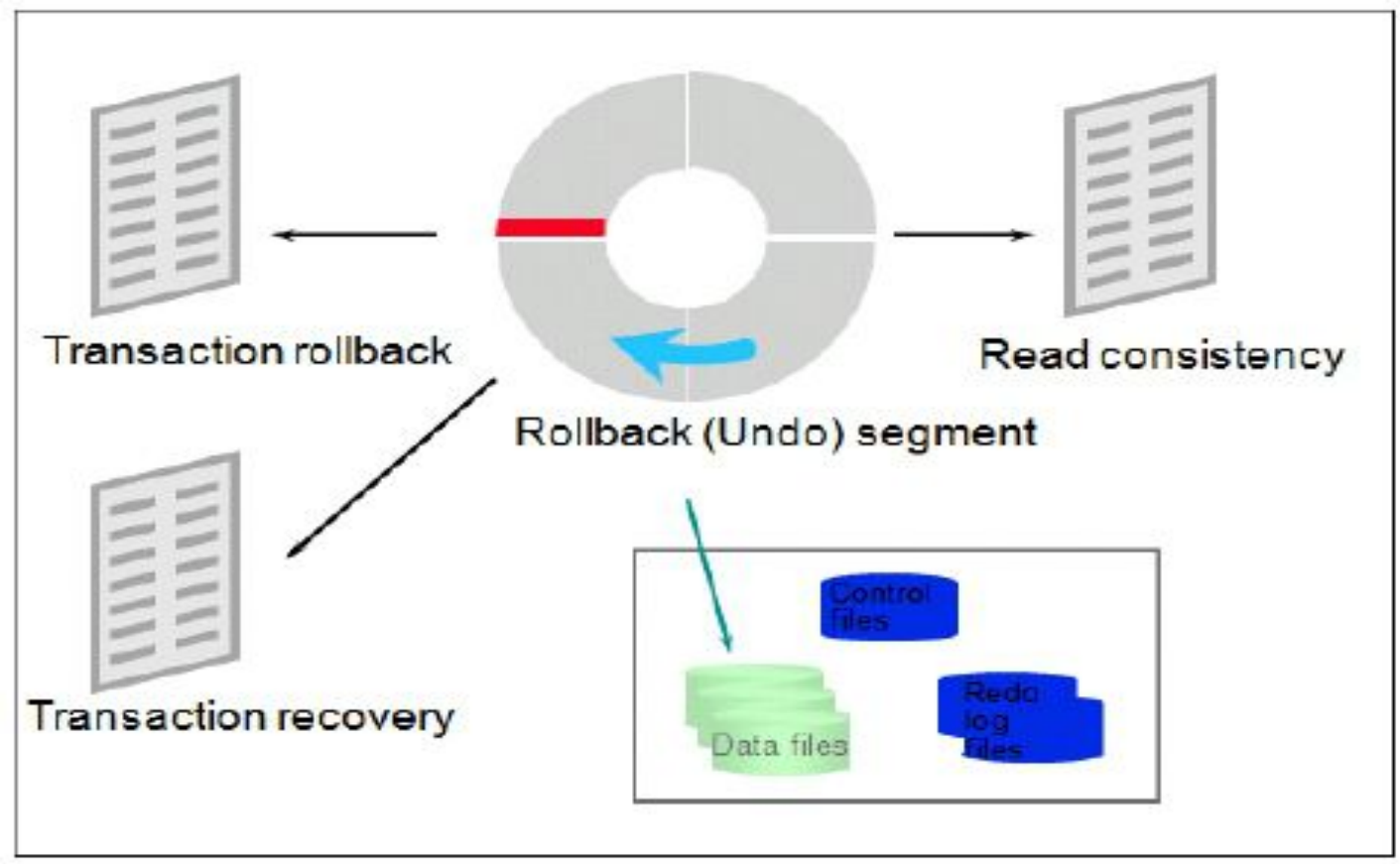
议题

1、简单介绍undo作用
2、orade是怎么找到前镜像的？
3、undo 相关初始化参数
4、undo 表空间的优化方法
5、undo 表空间损坏案例分享

Undo介绍

为了保证数据库的读一致性和能够回退事务，Oracle 必须拥有一种机制来实现保存修改之前的旧数据，以遍回退或撤消对数据库所作的修改，同时为数据恢复以及一致性读服务。

作用：一致读、事务回退



UNDO怎么找到前镜像的？

1、首先解释下ITL

ITL(Interested Transaction List)是Oracle数据块内部的一个组成部分，位于数据块的头部（block header），由xid，uba，flag，lck和scn/fsc组成，用来记录该块所有发生的事务，一个ITL可以看作是一条事务记录，但是这条记录为oracle内部使用。

转储的ITL信息：

```
Itl           Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0x000e.02a.0000026f  0x01400117.0066.0c  ----    1  fsc 0x0001.00000000
0x02   0x000b.007.00000276  0x014000e8.005c.14  C---    0  scn 0x0000.004fcd83
```

Xid：事务id

Uba：回滚段地址

Flag：事务标志位

Lck：影响的记录数

Scn/Fsc：快速提交（Fast Commit Fsc）的SCN或者Commit SCN

UNDO怎么找到前镜像的？

2、理论介绍

我们都知道，当用户发出一条SQL语句，ORACLE就知道了它的结果，其实原因就在于ITL中记录的SCN,和Uba.

当发出一条sql语句时，ORACLE会记录下这个时刻(SCN),然后在buffer cache中查找需要的BLOCK,或者从磁盘上读，当别的会话修改了数据，或者正在修改数据，就会在相应的block上记录ITL,此时ORACLE发现ITL中记录的SCN大于SELECT时刻的SCN，

那么ORACLE就会根据ITL中的Uba找到UNDO信息获得该block的前镜像，然后在buffer cache 中构造出CR块，此时ORALCE 也会检查构造出来的BLOCK中ITL记录的SCN，如果SCN还大于select时刻的SCN，那么一直重复构造前镜像，然后ORACLE找到前镜像BLOCK中的ITL的SCN是否小于select的SCN,同时检查这个事物有没有提交或者回滚，如果没有，那么继续构造前镜像，直到找到需要的BLOCK，如果在构造前镜像的过程中所需的UNDO信息被覆盖了，就会报快照过旧的错误。

这样ORACLE就实现了多版本，这就是ORACLE多版本的本质。

实验解释(1)

介绍理论一般听起来就比较晕，下面根据具体的实验来验证一下：

大体的实验步骤：

1、创建测试表

```
Create table t1(a number,b varchar2(20)) pctfree 0;
```

2、测试

```
--查看块号、相对文件号
select dbms_rowid.rowid_relative_fno(rowid)file_id,
dbms_rowid.rowid_block_number(rowid)block_id
from t1  where a=88;
Update t1 set a=88888 where a=88; --不提交
```

3、dump数据块

```
oradebug setmypid
alter system dump datafile 1 block 73546;
oradebug tracefile_name
```

实验解释(2)

4、在trace 文件里找到我们的ITL信息：

```
Itl       Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0x000e.02a.0000026f  0x01400117.0066.0c  ----    1  fsc 0x0001.00000000
0x02   0x000b.007.00000276  0x014000e8.005c.14  C---    0  scn 0x0000.004fcd83
```

可以看到01槽里面有一个活跃的事务0x000e.02a.0000026f

验证：

ITL中的XID的格式为:usn#.slot#.wrap#

```
select to_number('000e','XXXXX') from dual;
select to_number('02a','XXXXX')from dual;
select to_number('0000026f','XXXXXXXXX')from dual;
select xidusn,xidslot,xidsqn,ubafil,ubablk,ubasqn,ubarec
from v$transaction;
```

结果和V$TRANSACTION中记录的XIDUSN,XIDSLOT,XIDSQN相同.

实验解释(2)

5、查看undo block address

UBA的格式为：DBA.seq#.rec#

上面的活跃的事务的地址为：0x01400117.0066.0c

将DBA 解析成file_id 和 block_id

```
select
dbms_utility.data_block_address_file(to_number('01400117','xxxxxxxxxxxx')) file_id,
dbms_utility.data_block_address_block(to_number('01400117','xxxxxxxxxxxx')) block_id from dual;
select to_number('0066','xxxx') from dual;
select to_number('0c','xxxx') from dual;
select xidusn,xidslot,xidsqn,ubafil,ubablk,ubasqn,ubarec
from v$transaction;
```

结果和V$TRANSACTION中记录的ubafil,ubablk,ubasqn,ubarec一致

实验解释(3)

5、转储undo segment

根据上面的块号和文件号查看undo 块信息

oradebug setmypid

alter system dump datafile x block x;

oradebug tracefile_name

然后在trc中直接找0x000e.02a.0000026f

会看到以下信息：

UNDO BLK:

xid: 0x000e.02a.0000026f seq: 0x66 cnt: 0xc irb: 0xc icl: 0x0 flg: 0x0000

数据块的uba =0x01400117.0066.0c

UBA的格式：DBA.seq#.rec#。

cnt 为序列号，对应uba中的rec#。

可以通过Rec #0xc 找到与之对应的Rec号，根据Rec号的内容计算出真实的内容

实验解释(4)

7、根据Rec号的内容计算出真实的内容

以下是在trc中找到的对应的Rec序列内容

```
*----------------------------
Rec #0xc  slt: 0x2a  objn: 54770(0x0000d5f2)  objd: 54770  tbl
……省略部分
col  0: [ 2]  c1 59
col  1: [ 3]  44 42 41
```

根据列值计算值，得出的结果就是事务的更新值：

```
declare n number;
 begin
 dbms_stats.convert_raw_value('c159',n);
  dbms_output.put_line(n);
end;
```

UNDO相关参数

1、DNDO_MANAGEMENT

2、UNDO_TABLESPACE

3、UNDO_RETENTION

4、_UNDO_AUTOTUNE

默认是true

重建undo表空间

1、show parameter undo

2、create undo tablespace new_undo_tablespace

3、alter system set undo_tablespace=new_undo scope=both;

4、show parameter undo

5、drop tablespace old_undo

UNDO表空间优化

我们的数据库经常会遇到的问题：

1、undo表空间过大，浪费空间

2、undo表空间使用率长时间100%

3、经常遇到ORA-01555错误

遇到以上问题改怎么办呢？

UNDO表空间优化

关于undo数据的保留时间：

目前我们的数据库基本上都是按照undo自动管理方式，因为oracle会存在一些bug，所以我们会经常遇到undo保留时间过长、undo表空间过大的问题.

在目前的oracle 版本中undo保留时间其实是动态调整的，在这里列出了三种方法来对undo表空间的调整：

(1)使用(MAXQUERYLEN secs + 300)来确定UNDO_RETENTION

设置方法：需要把所有的undo datafile设置为可自动扩展，为了避免空间过大，需要设置最大值限制，例如：

ALTER DATABASE DATAFILE '<datafile_flename>' AUTOEXTEND ON MAXSIZE <current_size> ;

(2)使用TUNED_UNDORETENTION来确定UNDO_RETENTION

设置方法：undo数据文件为不可扩展

(3)使用UNDO_RETENTION初始化参数

设置方法：修改隐含参数：_undo_autotune = false

建议：设置_undo_autotune=false + 重建undo表空间 + 设置undo_retention=较大值 + 启用undo guarantee

UNDO表空间优化

选择合适的undo 大小：

1、使用脚本

2、根据AWR中的undo 统计

Undo Statistics

- Undo Segment Summary
- Undo Segment Stats

Back to Top

Undo Segment Summary

- Min/Max TR (mins) - Min and Max Tuned Retention (minutes)
- STO - Snapshot Too Old count, OOS - Out of Space count
- Undo segment block stats:
- uS - unexpired Stolen, uR - unexpired Released, uU - unexpired reUsed
- eS - expired Stolen, eR - expired Released, eU - expired reUsed

Undo TS#	Num Undo Blocks (K)	Number of Transactions	Max Qry Len (s)	Max Tx Concurcy	Min/Max TR (mins)	STO/ OOS	uS/uR/uU/ eS/eR/eU
543	1,348.96	1,955,147	46,370	48	196.1/209.7	0/0	0/0/0/6/14208/0

Back to Undo Statistics
Back to Top

Undo Segment Stats

- Most recent 35 Undostat rows, ordered by Time desc

End Time	Num Undo Blocks	Number of Transactions	Max Qry Len (s)	Max Tx Concy	Tun Ret (mins)	STO/ OOS	uS/uR/uU/ eS/eR/eU
24-Apr 17:24	492,131	220,254	46,370	36	196	0/0	0/0/0/0/0/0
24-Apr 17:14	706,776	1,675,329	45,766	48	199	0/0	0/0/0/0/0/0
24-Apr 17:04	150,052	59,564	45,162	22	210	0/0	0/0/0/6/14208/0

UNDO表空间损坏

环境介绍：

数据库版本：11.2.0.2.8

OS ： AIX 6.1

系统类型：OLAP

数据量：大约13T

有无备份：无

UNDO表空间损坏

1、错误信息

Errors in file /u01/app/oracle/diag/rdbms/xxx/xxx/incident/incdir_2081014/xxx_p000_17695096_i2081014.trc:

ORA-10388: parallel query server interrupt (failure)

ORA-00600: internal error code, arguments: [2023], [0], [0], [], [], [], [], [], [], [], [], []

Errors in file /u01/app/oracle/diag/rdbms/xxx/xxx/incident/incdir_2081014/xxx_p000_17695096_i2081014.trc:

ORA-10388: parallel query server interrupt (failure)

ORA-00600: internal error code, arguments: [2023], [0], [0], [], [], [], [], [], [], [], [], []

Sat Feb 14 00:42:49 2015

Errors in file /u01/app/oracle/diag/rdbms/xxx/xxx/trace/xxx_p064_13828282.trc (incident=2083214):

ORA-00600: internal error code, arguments: [2023], [0], [0], [], [], [], [], [], [], [], [], []

UNDO表空间损坏

处理方法：

重建undo表空间

create undo tablespace undotbs3 datafile ‘+DG1' size 1G;

alter system set undo_tablespace=undotbs3 scope=both sid=‘nwom21’;

create undo tablespace undotbs3 datafile ‘+DG1' size 1G;

alter system set undo_tablespace=undotbs4 scope=both sid=‘nwom22’;

删除有问题的：

drop tablespace UNDOTB2 including contents and datafiles;

报错：

ORA-01548: active rollback segment '_SYSSMU52$' found, terminate dropping tablespace

后续研究

查看事务信息，其实这里没有查看有活跃的事务信息，也没有定时任务.

首先查看dba_rollback_segs视图中的undo段信息，看哪些没有offline的转储没有offline的segment header

oradebug setmypid

alter system dump undo header ‘xxx';

oradebug tracefile_name

oradebug close_trace

在trc中查看：

```
index  state cflags  wrap#    uel         scn            dba          parent-xid    nub     stmt_num
------------------------------------------------------------------------------------------------
0x23   10    0x80  0x0b2d  0x0000  0x0000.017f73e0  0x0080006d  0x0000.000.00000000  0x00000001   0x00000000
```

state列9:inactive,10:active

如果有活跃的事务可以分居index查看到块的具体数据，但是在这里下面没有UNDO BLK信息。

Thanks