ParnassusData 诗檀软件 – Oracle数据库修复专家

Maclean Liu

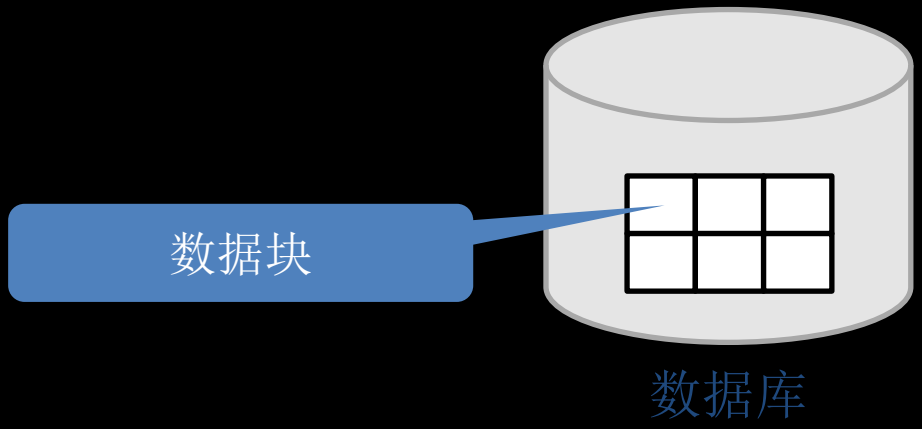
Oracle数据块损坏知识

Know More about Oracle Database Block Corruption

古希腊的Delphi(世界中心)，屹立着Parnassus Mount(诗檀山)，山上有一座阿波罗神庙，庙中住着女祭司(Oracle)

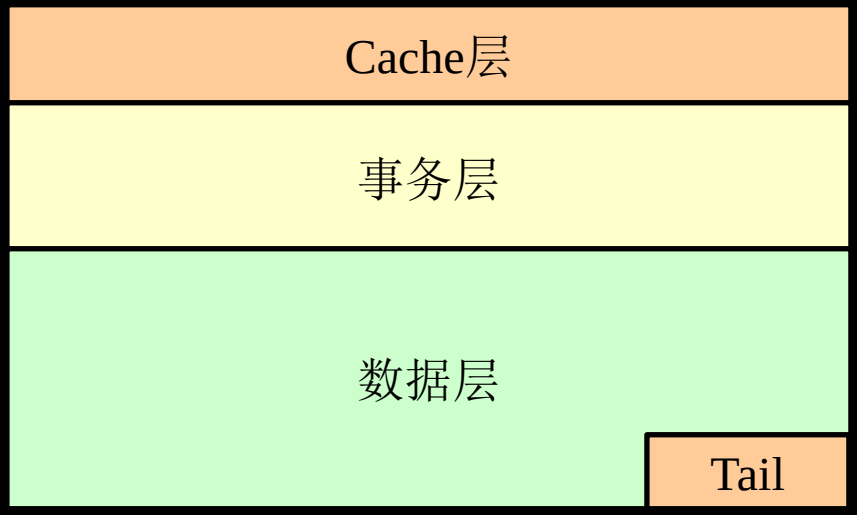
Oracle Data Block

- Oracle中最小I/O单位
- RDBMS具体存放数据的区域
- 一个Block对应磁盘上的一定区域，可能跨多个磁盘
- 微观的说，空间分配以块为单位



Oracle Block的构成

- 主要由Cache层，事务层和数据层组成

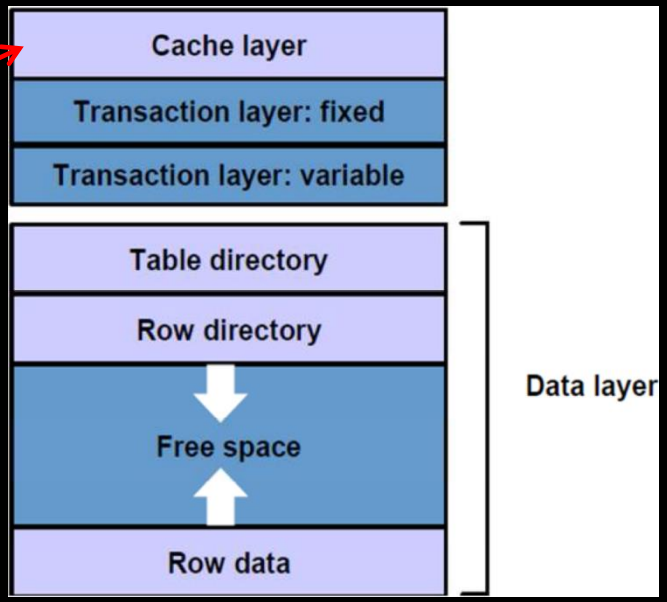


Oracle Block构成 Cache层 Kcbh

- Cache层包含用来检测块损坏的信息
  - Cache层的chkval_kcbh用来确认，上一次的写入到磁盘，到现在读取出来磁盘上的Block的内容是否一致
- Cache层存放的信息
  - 本数据块存放何种数据type_kcbh(Data,Index)
  - RDBA和格式版本
  - 本数据块的更新情况
- 几乎所有的数据块都有Cache层，KCBH

```
BBED> MAP
 File: o1_mf_system_9tdt5cf3_.dbf (0)
 Block: 500                                  Dba:0x00000000
------------------------------------------------------------
 KTB Data Block (Table/Cluster)

 struct kcbh, 20 bytes                       @0
 struct ktbbh, 72 bytes                      @20
 struct kdbh, 14 bytes                       @92
 struct kdbt[1], 4 bytes                     @106
 sb2 kdbr[94]                                @110
 ub1 freespace[825]                          @298
 ub1 rowdata[7065]                           @1123
 ub4 tailchk                                 @8188
```

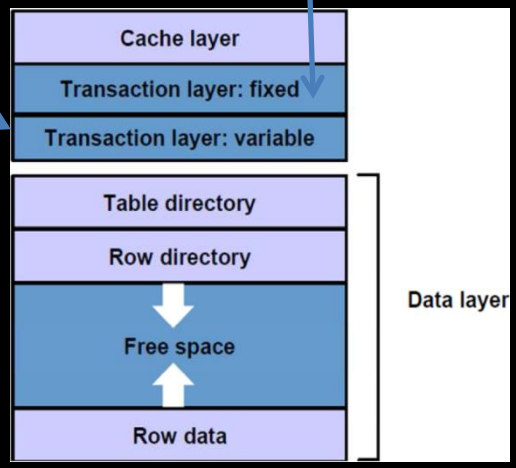


Oracle Block构成 事务层 ktbbh

- 在本数据块上执行着的或者执行过的事务信息
- 存放
  - SCN (System Change Number)
  - ITL 一种事务列表 ktbbhitl
  - 存放的数据类型 ktbbhtyp ，KDDBTINDEX代表索引 ，KDDBTDATA 代表数据
  - ktbbhflg ， MSSM or ASSM, KTBFONFL or KTBFBSEG
- 不是每个块都有ktbbh，例如Undo block就没有ktbbh

```
BBED> MAP
 File: o1_mf_system_9tdt5cf3_.dbf (0)
 Block: 500                                   Dba:0x00000000
------------------------------------------------------------
 KTB Data Block (Table/Cluster)

 struct kcbh, 20 bytes                        @0
 struct ktbbh, 72 bytes                       @20
 struct kdbh, 14 bytes                        @92
 struct kdbt[1], 4 bytes                      @106
 sb2 kdbr[94]                                 @110
 ub1 freespace[825]                           @298
 ub1 rowdata[7065]                            @1123
 ub4 tailchk                                  @8188
```



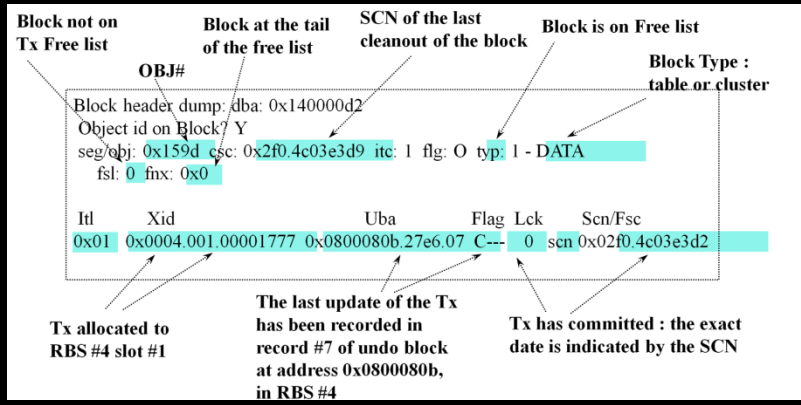
ITL(Interested Transaction List)

- XID:ktbitxid 事务ID
- UBA: 指向事务最后一次修改对应的Undo
- Flag:是否提交了，提交的情形 C、B、U、T
- Lck: 在本数据块锁住的行数
- Scn/Fsc: commit scn或control scn _ktbitfsc、_ktbitwrp、ktbitbas

Flag	is-set Function	Flag	Description
C	ktbico()	KTBFCOM	is committed transaction
B	ktbibi()	KTBFIBI	Rolling back this uba will give a before image of the itl / [never seen this!]
U	ktbiup()	KTBFUPB	commit time is an upper bound
T	ktbita()	KTBFTAC	active as of last cleanout (csc) [again, never seen now]

Regarding KTBFUPB: (ktbc3.h)

```
/*
* For a cleaned-out committed itl, 0x2000 indicates the commit time is
* an upperbound time.
* For a uncleaned-out active itl, 0x2000 indicates that the itl has
* been delayed-logged cleaned out with an scn base set, row locks are not
* cleared
*/
```

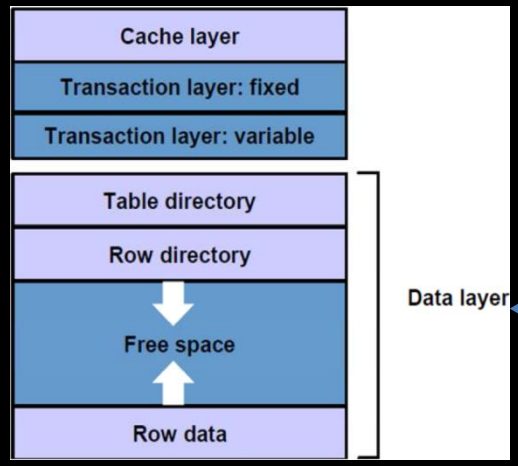


Oracle Block构造 数据层

- 数据的存储信息
- 数据头信息kdbh
- Table Direcotry kdbt ，针对簇表 有多条记录
- Row Directory kdbr ，记录了每一个row piece的块内偏移量
- Row piece 数据行片，一个row piece最多255字段，因为CC column count是单字节最大0xFF
- 根据存储数据类型的不同，其结构也会有区别

```
BBED> MAP
 File: o1_mf_system_9tdt5cf3_.dbf (0)
 Block: 500                                   Dba:0x00000000
------------------------------------------------------------
 KTB Data Block (Table/Cluster)

 struct kcbh, 20 bytes                        @0
 struct ktbbh, 72 bytes                       @20
 struct kdbh, 14 bytes                        @92
 struct kdbt[1], 4 bytes                      @106
 sb2 kdbr[94]                                 @110
 ub1 freespace[825]                           @298
 ub1 rowdata[7065]                            @1123
 ub4 tailchk                                  @8188
```



kdbh 数据头信息

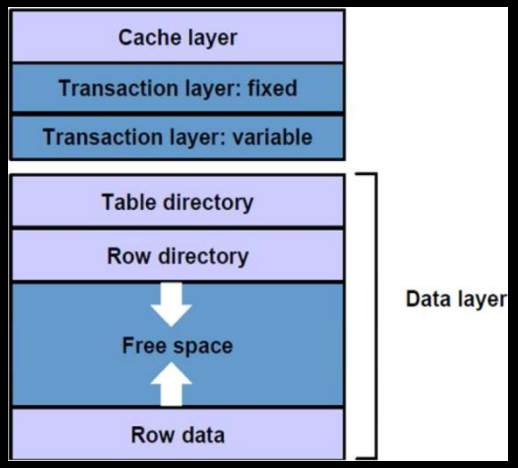
- kdbhflag bit位， KDBHFFK 0x01 Flushable Key
- kdbhntab Number of tables in the table index
- kdbhnrow Number of rows in the row index
- kdbhfrre first FRee Row index Entry
- kdbhfsbo Free Space Beginning Offset
- kdbhfseo Free Space Ending Offset
- kdbhavsp AVailable SPace in the block
- kdbhtosp TOtal Space that will be available

```
BBED> MAP
 File: o1_mf_system_9tdt5cf3_.dbf (0)
 Block: 500                                   Dba:0x00000000
------------------------------------------------------------
 KTB Data Block (Table/Cluster)

 struct kcbh, 20 bytes                        @0
 struct ktbbh, 72 bytes                       @20
 struct kdbh, 14 bytes                        @92
 struct kdbt[1], 4 bytes                      @106
 sb2 kdbr[94]                                 @110
 ub1 freespace[825]                           @298
 ub1 rowdata[7065]                            @1123
 ub4 tailchk                                  @8188
```

Table Directory kdbt

- kdbtoffs 对应表记录的偏移量
- kdbtnrow 对应表在本块的行数
- 通常情况下一张表仅仅有一条kdbt记录
- 当时簇表Cluster Table情况下，一个块中存放多张表记录，KDBT中有多条记录



Row Directory kdbr

- 一个kdbr数组，每一个成员为2个字节的signed bytes
- 每一个成员代表一行数据的块内相对偏移量
- 相对偏移量从数据层的起始位置开始
- 相对偏移量+环境偏移量=块内的绝对偏移量

数据块的大小

- 默认数据块的大小取决于建库时的DB_BLOCK_SIZE
- 可选项为 2k, 4k, 8k, 16k, 32k
- 默认数据块大小在建库后不可修改
- 可以混用非标准块大小的表空间，例如8k默认块大小下建16k大小的表空间
- 注意：非官方的看法是8k是最稳定，测试最多的块大小。 少数情况下非8k的Block Size更容易触发部分BUG。

数据块的损坏

- 什么是Oracle数据块损坏？

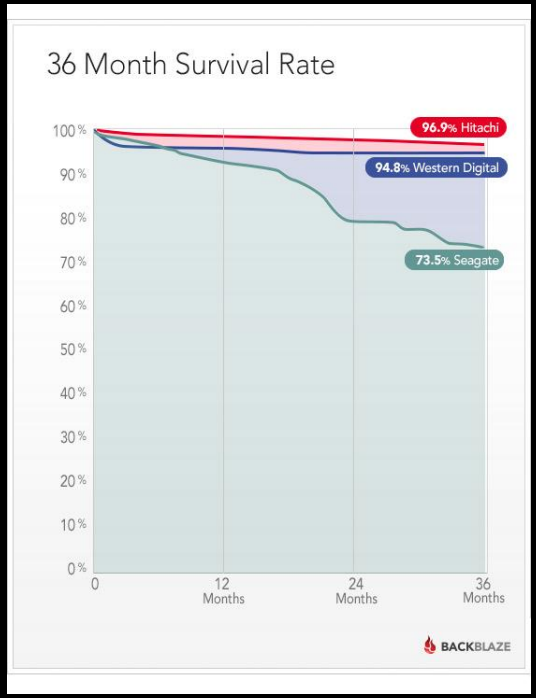
- 根据不同的情况，可以分成三种：
  - 物理损坏 Physical Corrupt
  - 逻辑损坏 Logical Corrupt
  - 写丢失　 Lost Write

造成数据块损坏的原因

- 存储/硬盘断电，机械老化故障
- 卷管理软件或文件系统bug
- OS I/O API bug
- 人为的误操作 dd/lvm/bbed
- Oracle RDBMS/ASM自身的bug，越来越少了

存储/硬盘故障

- 右图为36个月的硬盘留任率
- 一些案例：
  - ORA-1578 Transient Corruption - Caused by Parity Error on EMC DMX4 (Doc ID 1486903.1)
  - ORA-1578 Data Block Corruptions when using EMC Storage. Blocks with 0xc9 byte (Doc ID 1323108.1)
  - Database Corruption due to Lost IO on Hitachi storage. ORA-600 [kdsgrp1] ORA-1499 ORA-1410 ORA-600 [3020] (Doc ID 1512717.1)



卷管理软件或文件系统bug

- 一些案例：
  - ORA-354 Redo log corruption when using Xisgo Driver (Doc ID 1498389.1)
  - ORA-1578 ORA-353 ORA-19599 Corrupt blocks with zeros when filesystemio_options=SETALL on ext4 file system using Linux (Doc ID 1487957.1)
  - ORA-1578 Misplaced blocks against datafiles stored in NFS filers (Doc ID 1525108.1)
  - ORA-1578 ORA-354 ORA-600 [3020] Misplaced blocks by Symantec / Veritas after adding LUN (Doc ID 1323532.1)
  - ORA-1578 Block overwritten with string “DiskDescription cyl alt hd sec” when using Symantec / Veritas (Doc ID 1313454.1)
  - Block Corruption (ALL ZERO) detected after reclaiming space on Veritas Filesystem (Doc ID 1587427.1)

遇到过的VxCFS问题：

32位指针问题导致内存泄露

卷同步存在漏洞，将实际未完成同步的数据返回给Oracle使用

数据块损坏的影响

- 乐观：不发生显性错误，或仅少量SQL报错
- 悲观：业务几乎不可用，大量ORA-1578、ORA-8103、ORA-1410错误出现，或者数据库干脆打不开
- 部分数据可能丢失
- 数据库宕机
- 长时间的数据恢复流程，带库恢复几T数据？
- 上报 保监会、银监会？

数据块损坏检测参数 DB_BLOCK_CHECKSUM

- 三种不同作用的检测 保护机制，但都不负责修复现有问题。

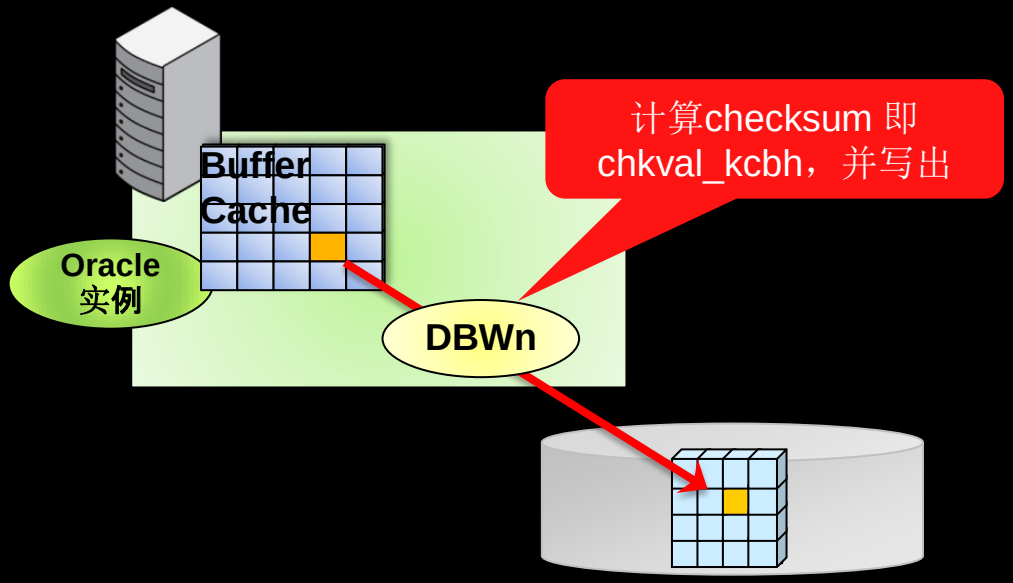
- DB_BLOCK_CHECKSUM负责控制块的cache层的chkval_kcbh是否在块被写出时计算并写入到磁盘中

```
BBED> sum
Check value for File 0, Block 500:
current = 0x5327, required = 0x5327

BBED> p chkval_kcbh
ub2 chkval_kcbh                         @16       0x5327
```

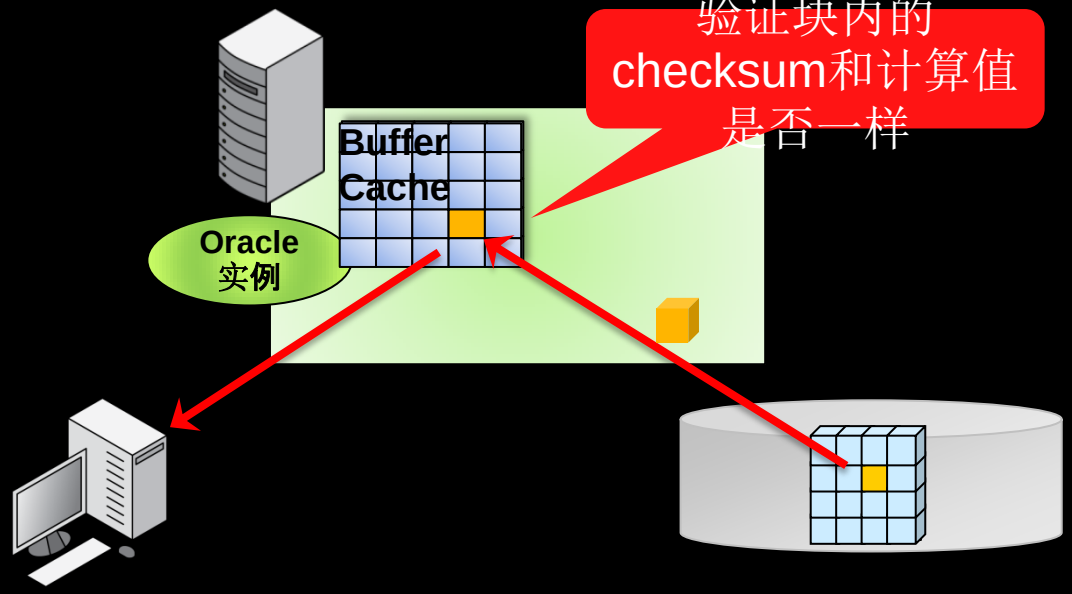
DB_BLOCK_CHECKSUM

数据块从buffer cache中写出时



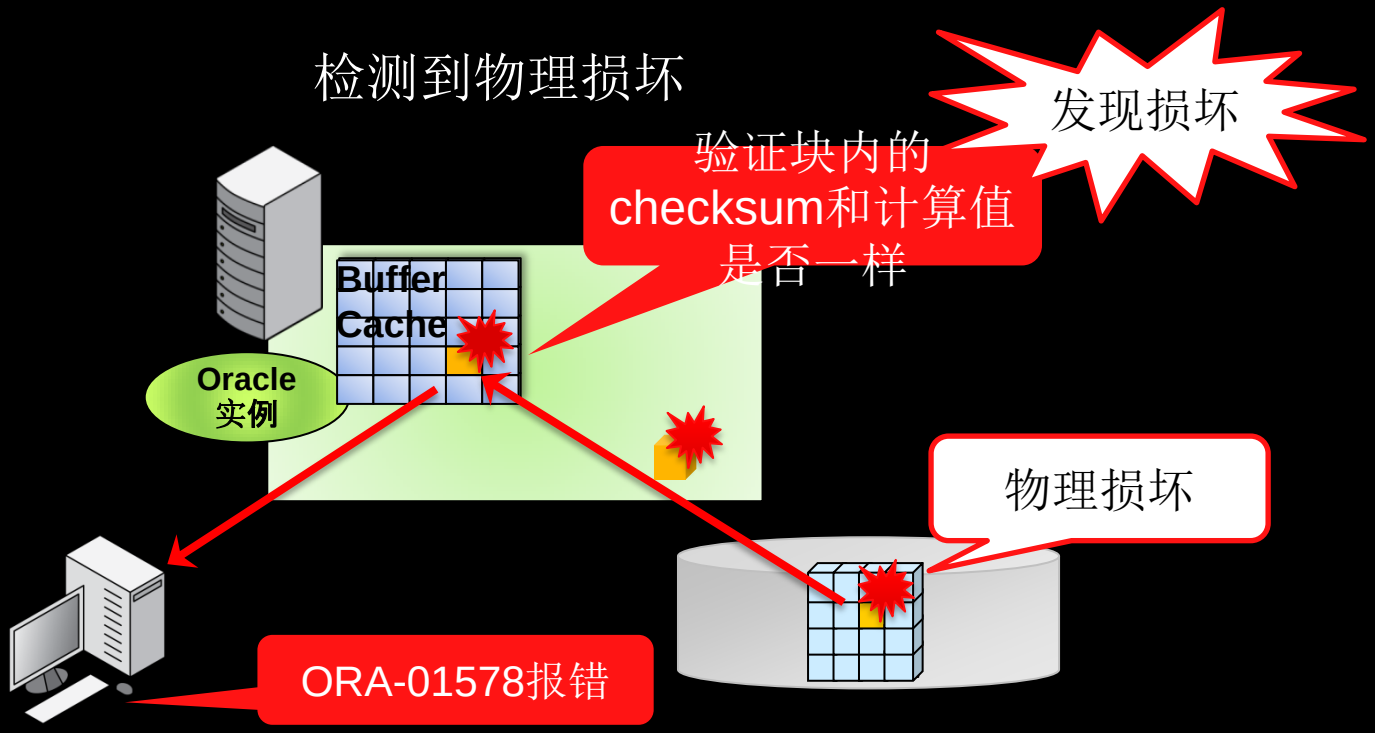
DB_BLOCK_CHECKSUM

数据块被从磁盘读取到buffer cache中时



DB_BLOCK_CHECKSUM

- 检验发现checksum和计算值不一样



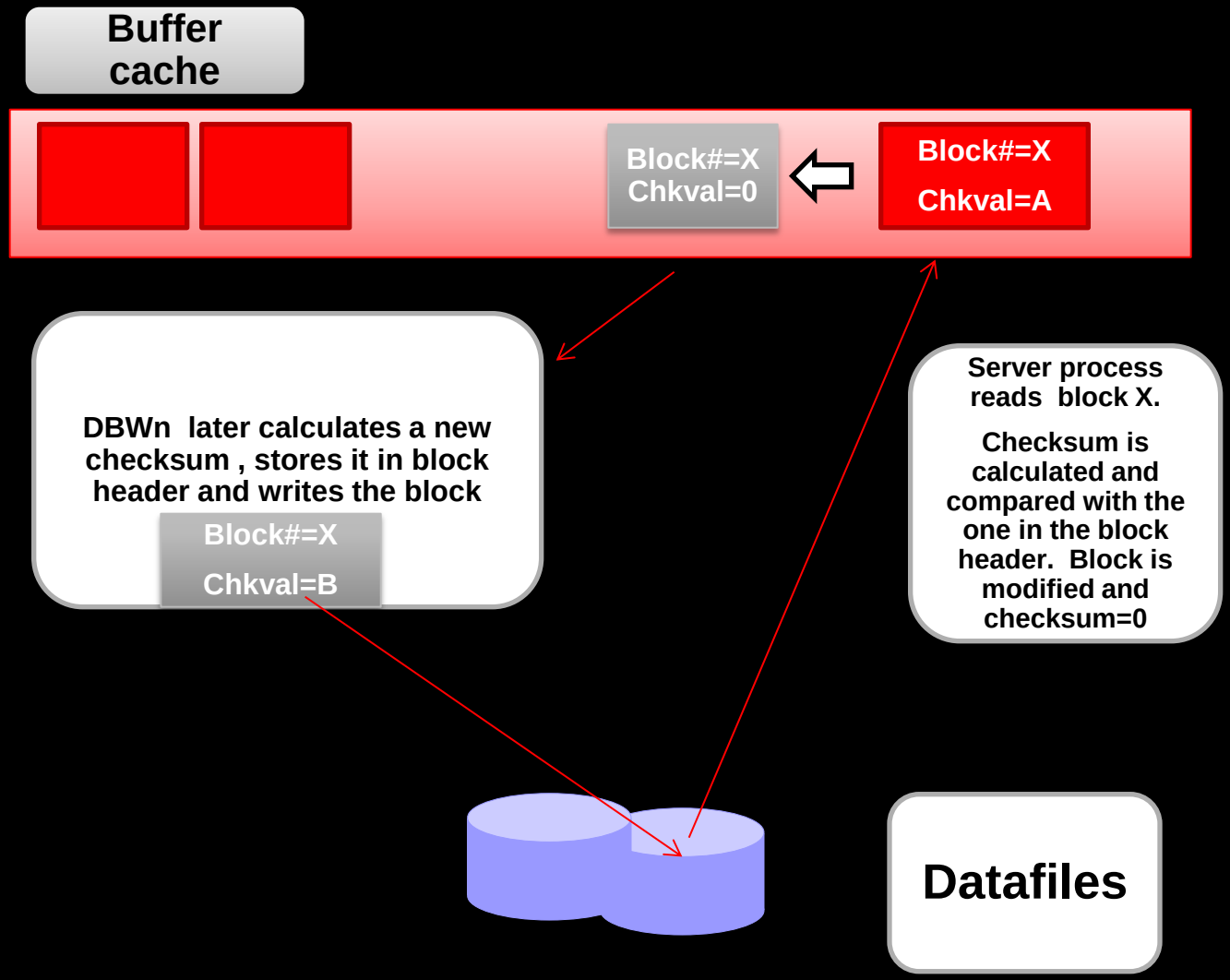
DB_BLOCK_CHECKSUM

- 通过一个校验值来检测出块损坏:
  - 由于断电或硬盘故障，一个块仅写出了一半或更少内容，此时chkval必然不等于整个块checksum
  - 从写出到下一次读取的时间中硬盘发生了故障，块中的部分内容不正确
- DB_BLOCK_CHECKSUM控制是否计算和写chkval_kcbh，如果不写则chkval_kcbh为0000，读取时也不做检测

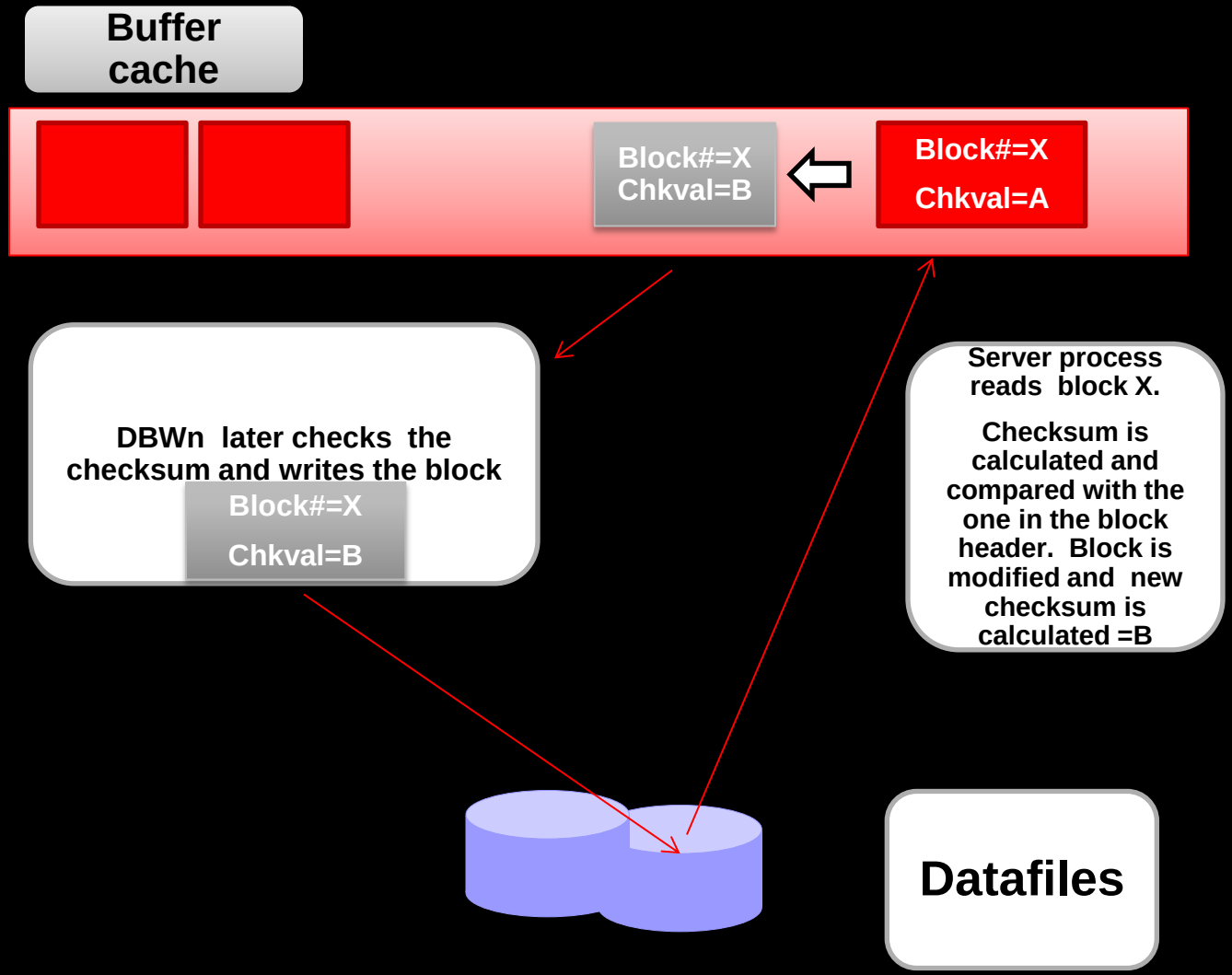
DB_BLOCK_CHECKSUM

- 默认值 TRUE(10g),Typical(11g)
- 当块要写出到磁盘时，DBWn从块中的每一个字节计算checksum并存放到块头的chkval_kcbh
- 有了checksum后，Oracle能判断由底层磁盘存储系统引起的损坏
- 每一次数据块从磁盘中被读取都会检查checksum
- 更新过程如下：
  - 服务进程修改数据块，checksum被置0
  - DBW负责计算checksum并写出块
  - 在上述改数据块和checksum之间存在时间，DBW不清楚这中间的问题，也不负责

db_block_checksum=TRUE



db_block_checksum=FULL



db_block_checksum=False

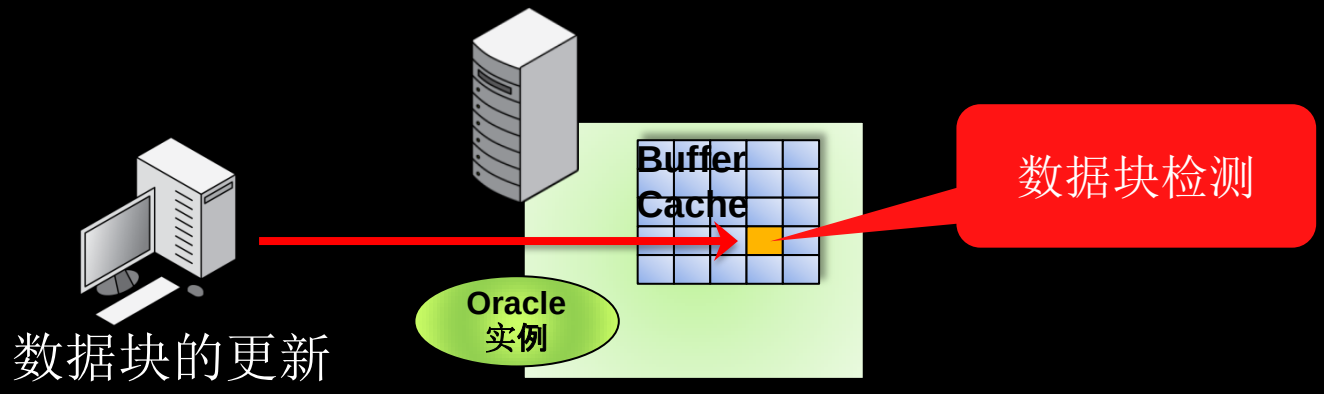
- db_block_checksum=False，那么仅SYSTEM表空间上的block还做checksum
- _db_always_check_system_ts Always perform block check and checksum for System tablespace,该隐藏参数控制是否对SYSTEM表空间做checksum

数据块损坏检测参数 DB_BLOCK_CHECKING

- 对内存中的块内容做修改前，是否对块做检测
- 考虑这样一个问题，假设使用FPE工具(一种古老的游戏内存修改工具）修改了内存中一个块的内容，这意味着这个块已经错了。然后Oracle不做任何检测继续在这个块上做修改，那么就是错上加错了。
- DB_BLOCK_CHECKING的义务

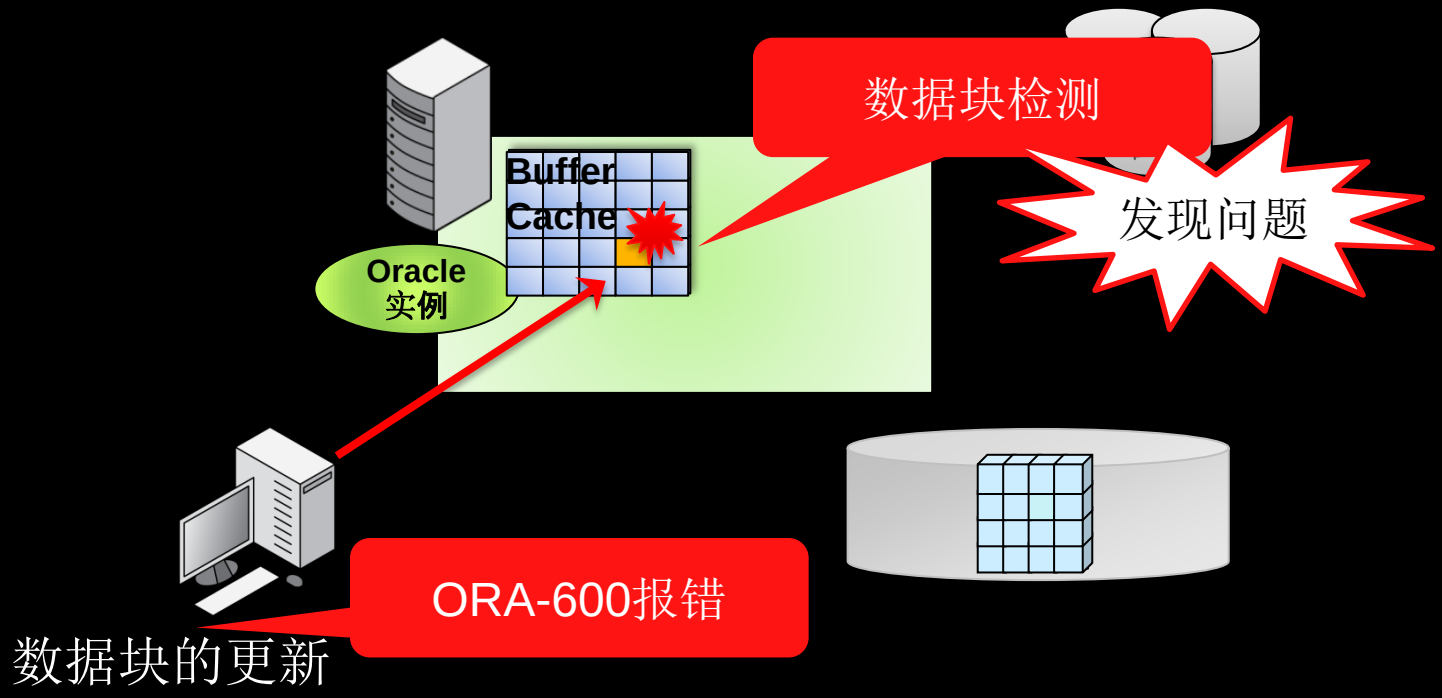
DB_BLOCK_CHECKING

- 当数据块在内存中被修改时，检测块的事务层和数据层的完整性

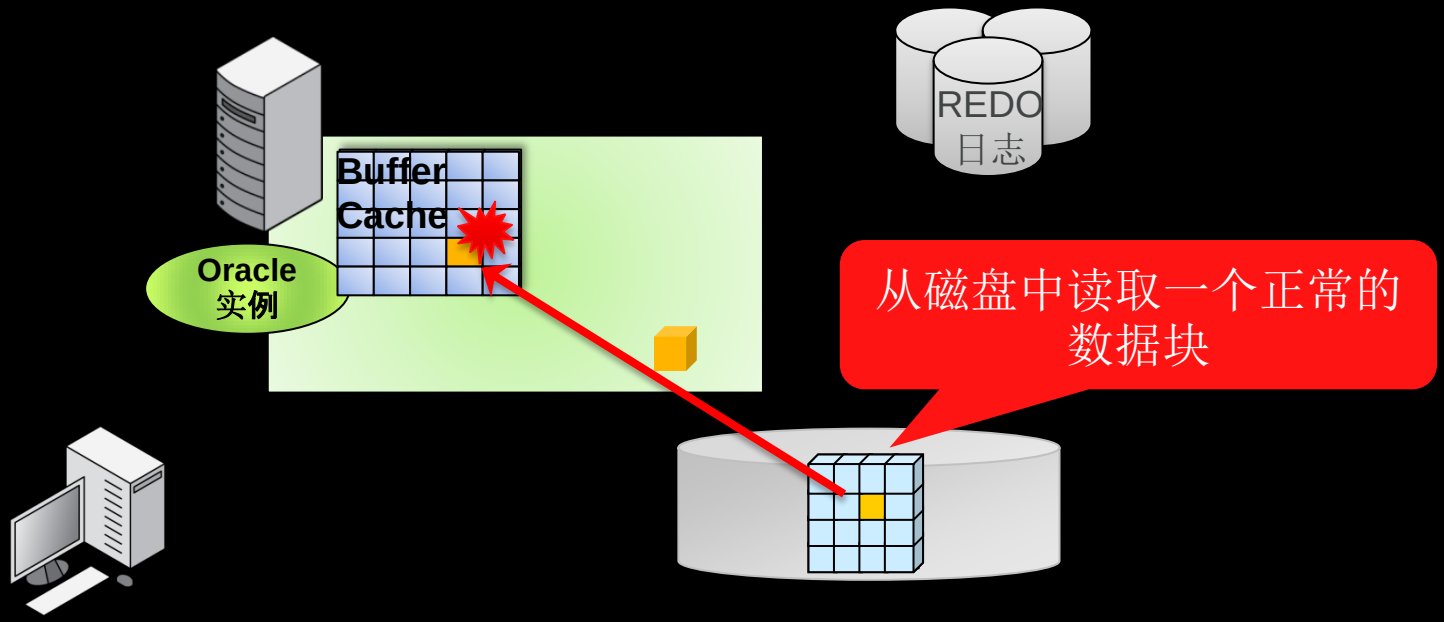


DB_BLOCK_CHECKING

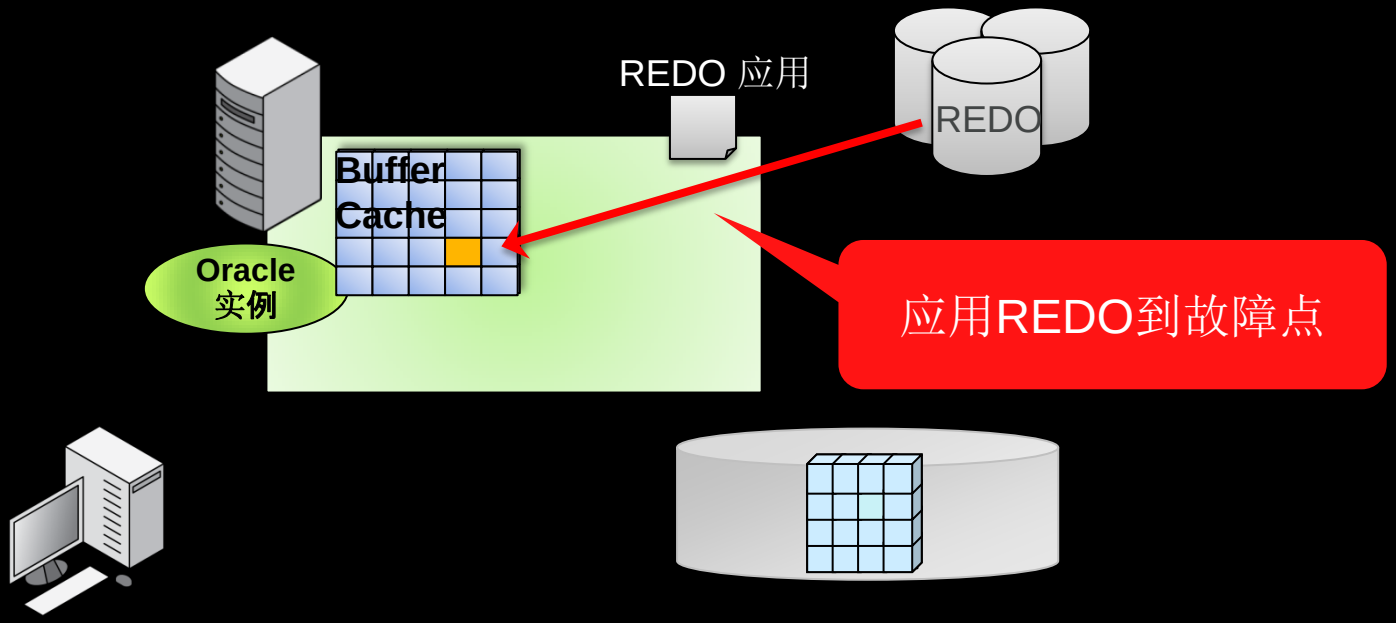
对于更新过程中检测发现问题时，阻止在这个缓存块中继续更新和回滚，直接报ORA-00600错误



DB_BLOCK_CHECKING

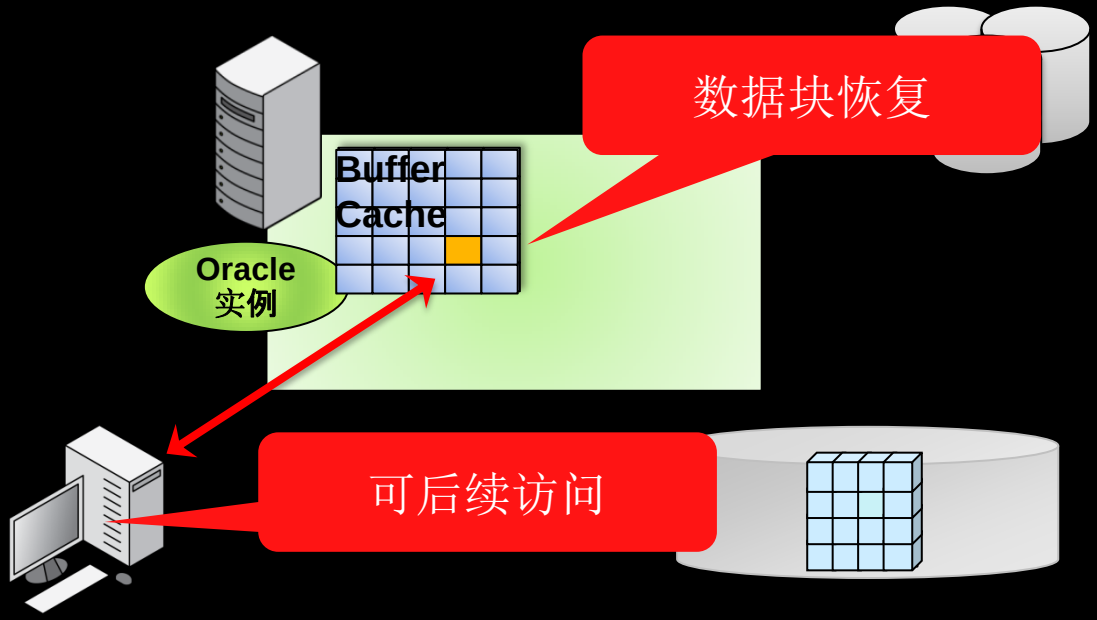


DB_BLOCK_CHECKING



DB_BLOCK_CHECKING

- 对于仅仅发生在buffer cache里的损坏，通过db_block_checking检测并保护，不让它写出到磁盘上，不让错上加错发生。则大事化小小事化了



DB_BLOCK_CHECKING

当数据块在内存中被修改时，检测块的事务层和数据层的完整性

- Row piece没有重叠在一起
- 可用空间的大小是合理的
- ITL锁定的行数，和实际有lock bit的行数是对应的
- Oracle定义了上百个检测C语言宏

2000		
undo data block	(type# 2)	
2001	KCBTUNDA_EC_START + KTUCDBFT	offset to FooTer has wrong value
2002	KCBTUNDA_EC_START + KTUCDBNR	# of undo record in this block exceed allowed
2003	KCBTUNDA_EC_START + KTUCDBRB	RollBack record index exceed max
2004	KCBTUNDA_EC_START + KTUCDBCL	CoLlecting record index exceed max
2005	KCBTUNDA_EC_START + KTUCDBAL	ALignment error
2006	KCBTUNDA_EC_START + KTUCDBCC	undo Count Corrupted

DB_BLOCK_CHECKING

- False/OFF 除了SYSTEM表空间的块还做部分检测外，其他表空间的块不做此类检测
- LOW 基本的块头检测(如rbba,seqno)会在内存中块内存修改后被检测，例如UPDATE/INSERT，磁盘读或RAC节点间传输
- MEDIUM 所有的LOW级别检测，同时对堆表的数据块做语义检测
- FULL/TRUE 所有LOW和MEDIUM检测的，同时也对索引块做语义检测
- DB_BLOCK_CHECKING的表现实际收到_DB_BLOCK_ADJCHK_LEVEL（默认为7）的影响

DB_BLOCK_CHECKING

- 默认值为FALSE
- 为了检测和避免逻辑损坏
- 典型情况下block checking的性能损耗为1%到10%，但更视乎合计负载类型，某银行启用DB_BLOCK_CHECKING=FULL后CPU使用率增长了20%~30%
- DB_BLOCK_CHECKING=MEDIUM时性能损耗较少，原因是其不检测索引块，也不检测IOT
- 如果发现了损坏，报错常为ORA-600 [kddummy_blkchk] 或ORA-600 [kdBlkCheckError]，进一步损坏被避免

_db_block_check_for_debug=TRUE

- 默认为False
- 和DB_BLOCK_CHECKING配合使用，如果没设DB_BLOCK_CHECKING那么不要用
- 当发现损坏时转出块以便诊断
- 该参数控制在修改块前拷贝块到PGA中，若发现问题则将该拷贝转出到跟踪文件中
- 性能损耗在于每次修改块前都会拷贝块到PGA中，其损耗实际并不大

DB_LOST_WRITE_PROTECT
啥是Lost Write?

- 存储或卷管理软件(如Veritas)都可能是引发Lost Write的层面,表现为:
  - 因为是Lost Write，所以上一次写入的Block结构完全正常，Oracle默认不会检测报错
  - 提高了返回给用户不正确数据的可能
  - 不正确的数据可能进一步污染其他业务数据
  - Lost Write的业务影响
    - 缺货还超卖
    - 下单的记忆似乎不那么真实

DB_LOST_WRITE_PROTECT

- 针对写丢失的保护
- 需要DataGuard
- 版本11g中出现
- 默认为NONE
- 当在DG环境的Primary上设置为TYPICAL，则将read-write表空间上的buffer cache读记录到redolog中，以便判断写丢失
- 则Lost Write可以在数据库recovery时被发现，如物理备库

DB_LOST_WRITE_PROTECT

- DG物理备库Lost Write检测机制
- Primary Database从磁盘读取Block的操作记录到redo中
  - Data File Number
  - Data Block Address(DBA)

  唯一指向一个数据块
  - System Change Number(SCN)

  Block更新时会增长
- DataGuard中redo将传输到Standby Database
- 在Standby上检验redo内的SCN和本地block

如果SCN不一致，则说明有Lost Write发生的可能

DB_LOST_WRITE_PROTECT

- Primary和Standby2侧都要设置

Value	Default	Lost Write检测对象	Primary Database	Standby Database
NONE	✔	N/A	N/A	N/A
TYPICAL		Read / Write表空间	从磁盘读取block到buffer cache中时生成检测用Redo	REDO APPLY的一部分，MRP将负责对比REDO里的SCN
FULL		Read / Write表空间 Read Only表空间		

- Primary或Standby上设置为NONE时是无效的
  - Primary不生成相关redo，Standby无法验证
  - Primary生成redo,Standby不验证

数据块损坏检测参数
DB_LOST_WRITE_PROTECT

- Lost Write检测出的时机？
- 存在这样一种可能性，即发生Lost Write后Primary立即发生了读取和修改，那么Standby上MRP的验证已经晚了

- 一般情况下写丢失仅仅发生在少量块上，只要不适用这些块：
  - 对其他块的查询更新都是正常的
  - 错误的数据不会污染别的块

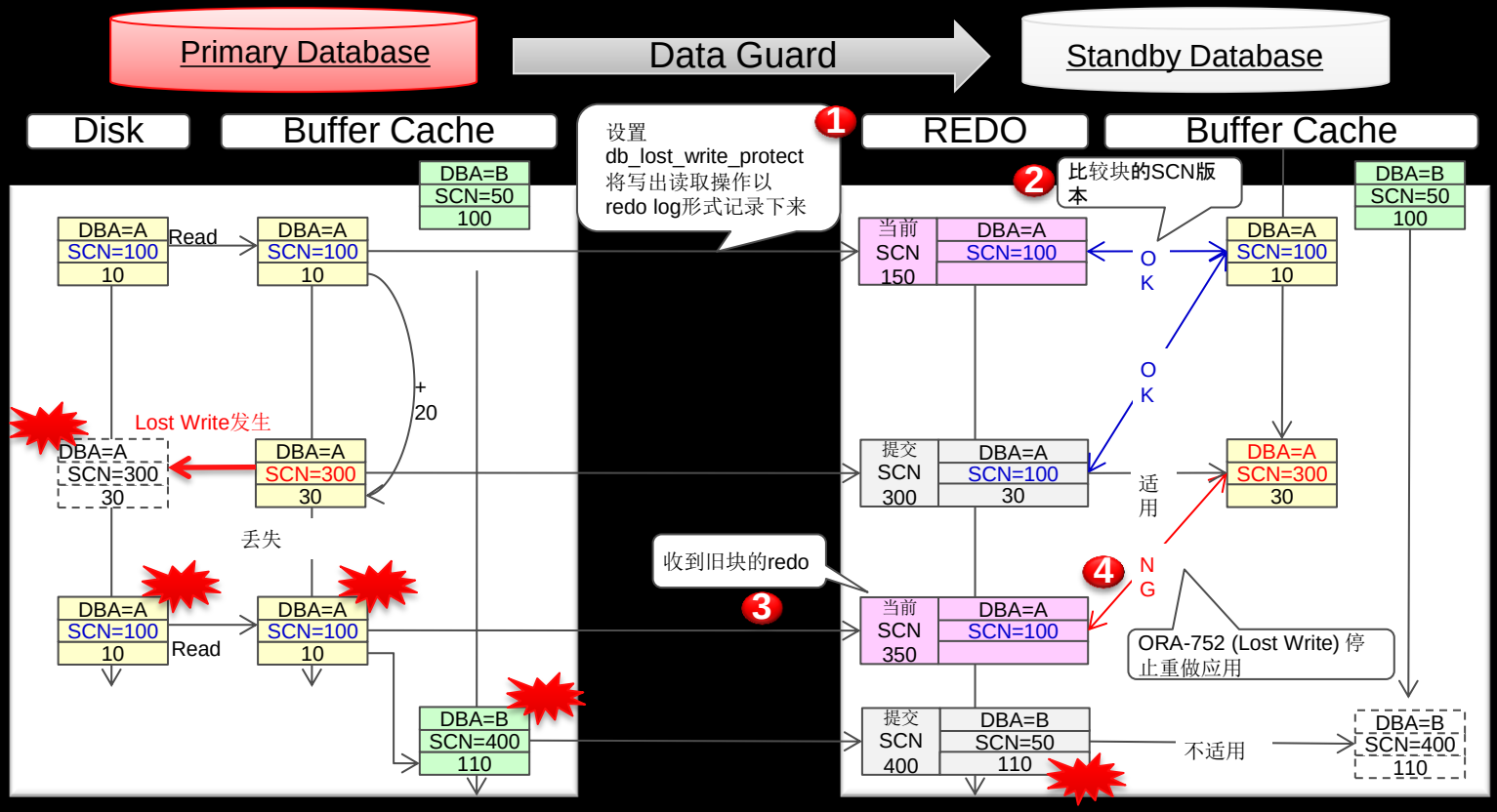
DB_LOST_WRITE_PROTECT 补充

- 相关验证redo仅限于buffer cache读
  - 如果是direct path read直接路径读，则不会生成

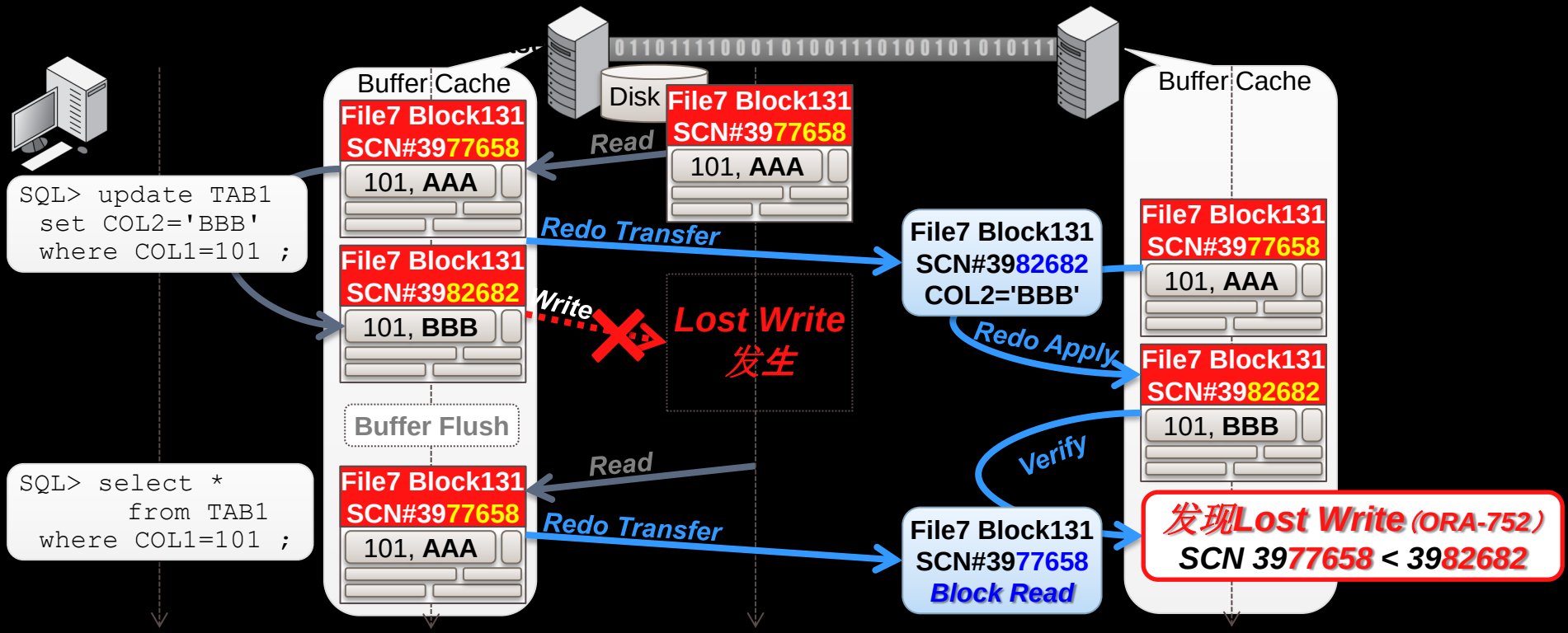
- 非DataGuard环境下也可以生成验证Redo
  - 可以在前滚时rolling forward时做检验

- Standby上也可能检测出LostWrite
  - 还可通过ASM Mirror或ABR自动修复

DB_LOST_WRITE_PROTECT



发现Lost Write流程(Primary侧Lost Write)



DB_LOST_WRITE_PROTECT
发现Lost Write后的动作

- 负责通知Lost Write的是Physical Standby中的一个备库
- 其在alert.log中记录ORA-752
- 为了保护备库，自动停止MRP进程
- 后续的redo不被应用，不用担心进一步数据污染

- PRXX进程对应的跟踪文件中记录了详细信息
  - 包括对相关redo和block的转储
  - 这些信息帮助DBA追查Lost Write问题

DB_LOST_WRITE_PROTECT
发现Lost Write的Standby端的alert.log

```
Wed Oct 23 19:18:08 2013
Hex dump of (file 7, block 131) in trace file
    /u01/app/oracle/diag/rdbms/orcls/orcls1/trace/orcls1_pr02_1401.trc
Reading datafile '+DATA/orcls/datafile/lw.281.829593935' for corruption at rdba: 0x01c00083
    (file 7, block 131)
Read datafile mirror 'DATA_0004' (file 7, block 131) found same corrupt data (logically
    corrupt)
Read datafile mirror 'DATA_0006' (file 7, block 131) found same corrupt data (logically
    corrupt)
STANDBY REDO APPLICATION HAS DETECTED THAT THE PRIMARY DATABASE
LOST A DISK WRITE OF BLOCK 131, FILE 7
NO REDO AT OR AFTER SCN 3987667 CAN BE USED FOR RECOVERY.
Recovery of Online Redo Log: Thread 1 Group 7 Seq 103 Reading mem 0
Slave exiting with ORA-752 exception
Errors in file /u01/app/oracle/diag/rdbms/orcls/orcls1/trace/orcls1_pr02_1401.trc:
ORA-00752: recovery detected a lost write of a data block
ORA-10567: Redo is inconsistent with data block (file# 7, block# 131, file offset is 1073152
    bytes)
ORA-10564: tablespace LW
ORA-01110: data file 7: '+DATA/orcls/datafile/lw.281.829593935'
ORA-10561: block type 'TRANSACTION MANAGED DATA BLOCK', data object# 87637
Wed Oct 23 19:18:12 2013
Recovery Slave PR02 previously exited with exception 752
Wed Oct 23 19:18:12 2013
MRP0: Background Media Recovery terminated with error 448
Errors in file /u01/app/oracle/diag/rdbms/orcls/orcls1/trace/orcls1_pr00_1395.trc:
ORA-00448: normal completion of background process
MRP0: Background Media Recovery process shutdown (orcls1)
```

DB_LOST_WRITE_PROTECT
PRXX进程TRACE

Block Dump

Main Message

Redo Dump

```
Hex dump of (file 7, block 131)
Dump of memory from 0x00000000F03B0000 to 0x00000000F03B2000
0F03B0000 0000A206 01C00083 003CC55A 04010000  [........Z.<.....]
...
STANDBY REDO APPLICATION HAS DETECTED THAT THE PRIMARY DATABASE
LOST A DISK WRITE OF BLOCK 131, FILE 7
The block read on the primary had SCN 3977658 (0x0000.003cb1ba) seq 1 (0x01)
 while expected to have SCN 3982682 (0x0000.003cc55a) seq 1 (0x01)
The block was read at SCN 3987667 (0x0000.003cd8d3), BRR:
CHANGE #1 TYP:2 CLS:6 AFN:7 DBA:0x01c00083 OBJ:87637 SCN:0x0000.003cb1ba SEQ:1 OP:23.2 ENC:0 RBL:1
...
REDO RECORD - Thread:1 RBA: 0x000067.00000128.0010 LEN: 0x0034 VLD: 0x10
SCN: 0x0000.003cd8d3 SUBSCN:  1 10/23/2013 19:18:16
(LWN RBA: 0x000067.00000126.0010 LEN: 0003 NST: 0001 SCN: 0x0000.003cd8cf)
CHANGE #1 TYP:2 CLS:6 AFN:7 DBA:0x01c00083 OBJ:87637 SCN:0x0000.003cb1ba SEQ:1 OP:23.2 ENC:0 RBL:1
 Block Read - afn: 7 rdba: 0x01c00083 BFT:(1024,29360259) non-BFT:(7,131)
              scn: 0x0000.003cb1ba seq: 0x01
              flags: 0x00000006 ( dlog ckval )
```

发现Lost Write的数据块

① Primary侧 Block的SCN

② Standby侧Block的SCN

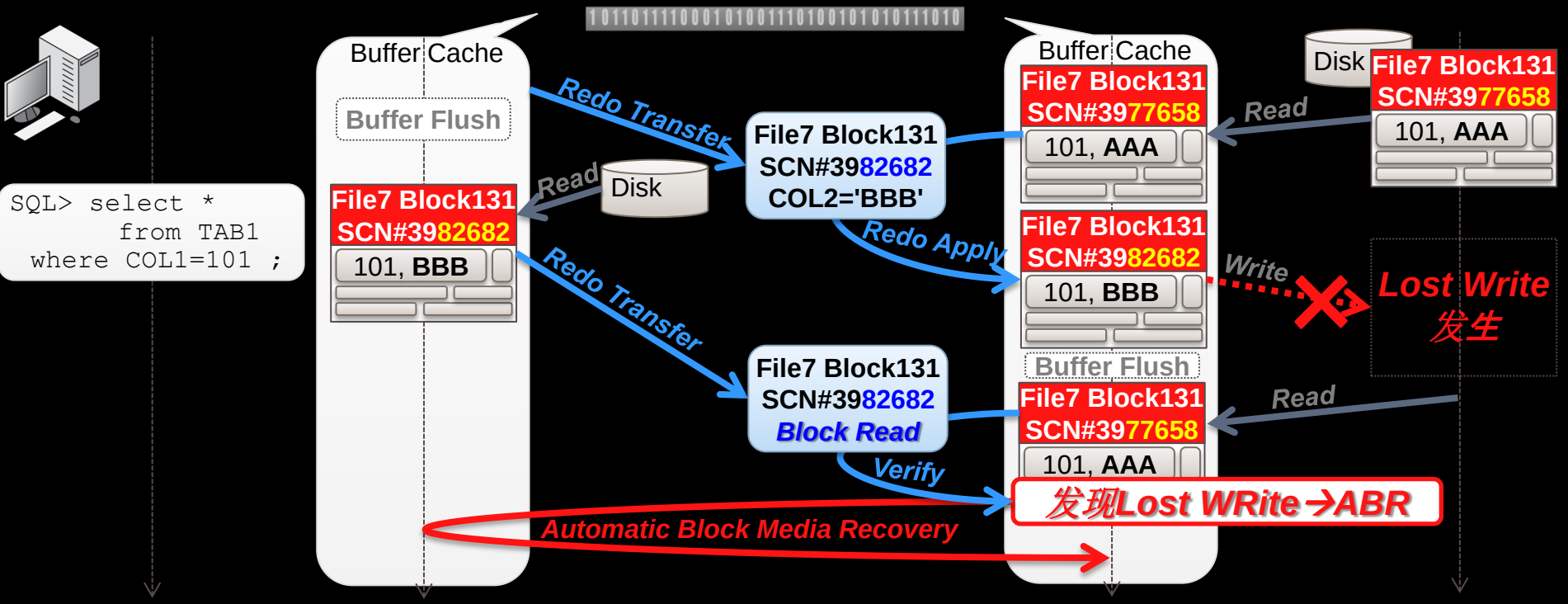
③ Primary侧读取该block时的SCN

3977658 (①) < 3982682 (②) 可判定Primary发生Lost Write

3982682 (②)的更新被丢失了

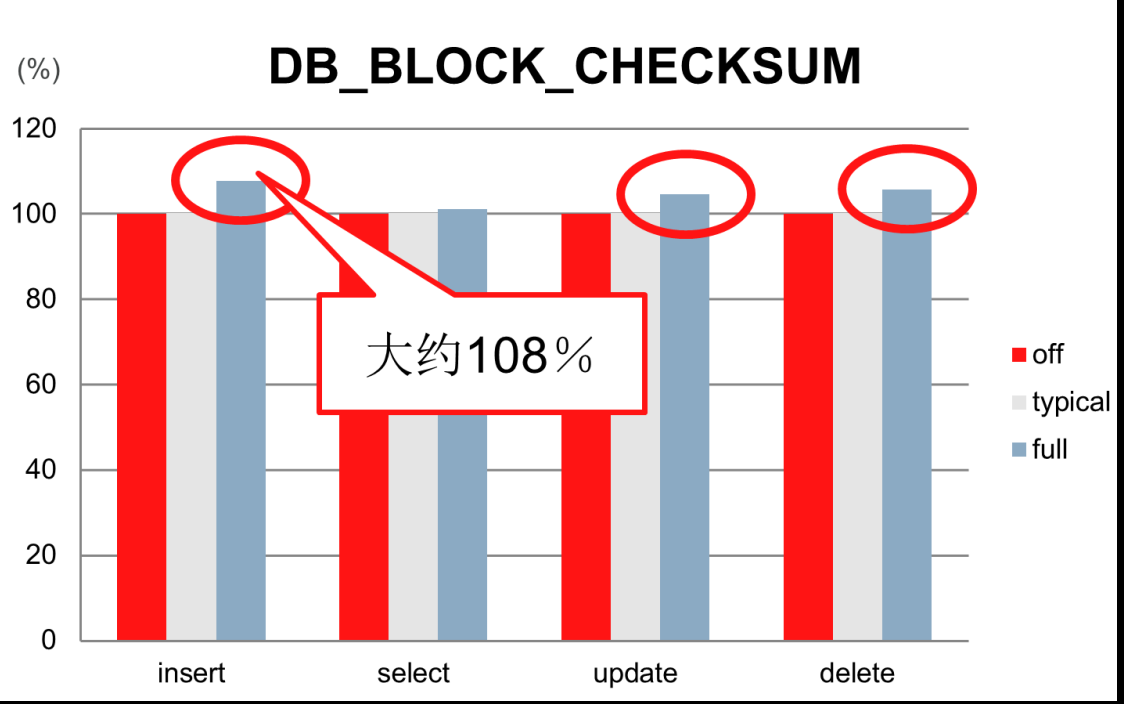
3987667 (③)读取数据块时被检测到

DB_LOST_WRITE_PROTECT
Standby侧检测出Lost Write



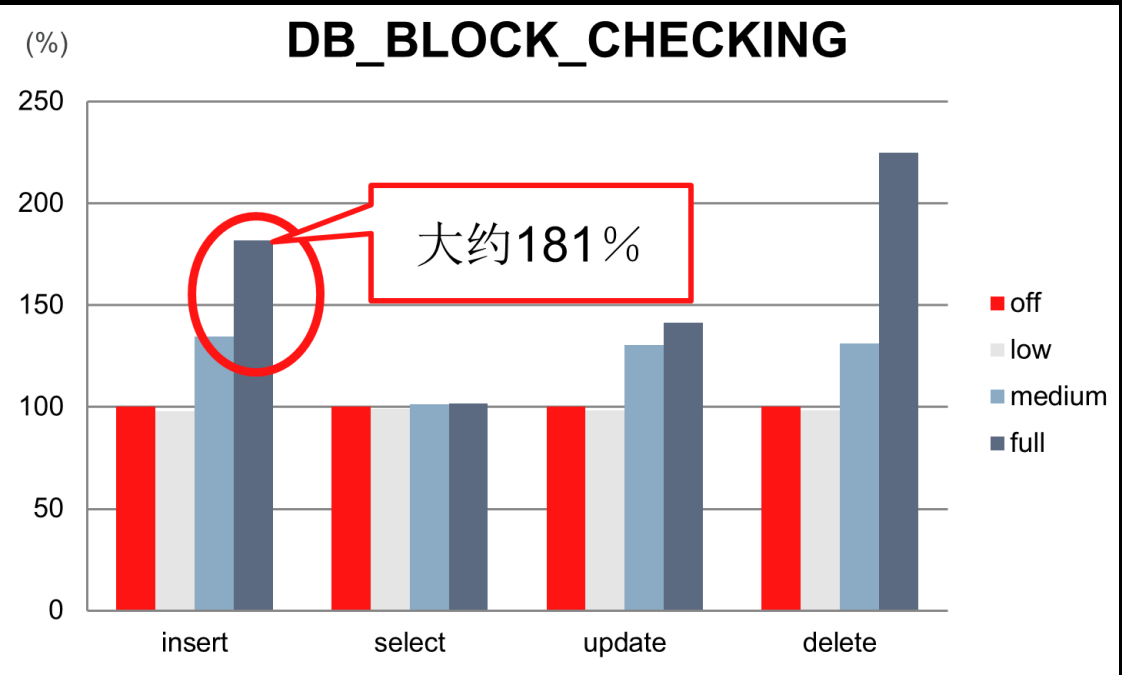
DB_BLOCK_CHECKSUM性能损耗测试与设置为OFF时的执行耗时对比

1. 开销较小
2. FULL时大约8%



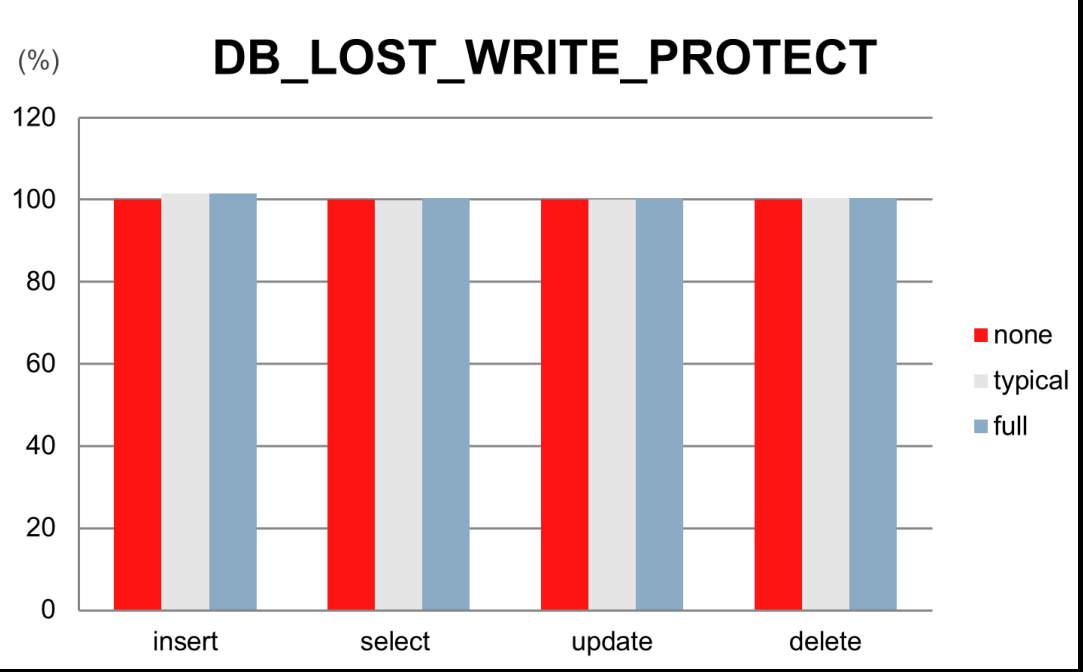
DB_BLOCK_CHECKING性能损耗测试与设置为OFF时的执行耗时对比

1. LOW级别影响很小
2. SELECT查询几乎都不受影响
3. MEDIUM · FULL级别下有较大的损耗(最大约181%)
4. FULL级别下INSERT和UPDATE的区别主要体现在索引的更新量上



DB_LOST_WRITE_PROTECT性能损耗测试与设置为OFF时的执行耗时对比

1. 几乎没有特别的开销
2. 不同语句之间几乎没有差异



ParnassusData™

www.parnassusdata.com

400-690-3643

Thank You