Oracle Database Online Documentation 11g Release 2 (11.2) / Database Administration/Database Concepts/10 Transactions/

Oracle Database Online Documentation 11g Release 2 (11.2) / Database Administration/Database Concepts/ 13 Oracle Database Instance

Installing, Executing and Interpreting output from the "scnhealthcheck.sql" script (Doc ID 1393363.1)

>概念

scn和checkpoint在数据库中无处不在，与控制文件，数据文件，日志文件，事务等息息相关。在概览scn和checkpoint之前，我们先来了解几个概念

System Change Numbers (SCNs)：系统更改号：是一个逻辑的，oracle内部的时钟。scn必须满足事务的ACID属性，每个事务的commit record中都有一个惟一的scn

scn总是单向递增长，scn用来指定一个逻辑时间点标记内部事件，如一个事件的scn低于另一个事件，则说明这个事件发生的时间早。

checkpoint：检查点：是个事件名称，是一个关键机制在数据库一致性上，包括关闭数据库，数据库恢复，数据库操作等

checkpoint position（ checkpoint scn）：检查点位置：是一个特殊时间点的scn，在redo流中指出恢复的起始点。

12.22日，我们又称为冬至日

>目的：

scn：

标记oracle内部发生事件的时间

checkpoint：

1 减少实例或介质恢复的时间   --checkpoint position

2 数据库运行中确保buffer cache中的dirty buffers写入到磁盘   --Incremental checkpoints

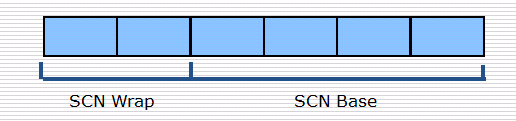
3 数据库关闭时确保所有commit的数据都写入到磁盘   --database checkpoint

========================================

》scn怎么理解

一个墙钟 wall clock

scn的结构：6个byte大小=高位2byte+低位4byte



scn16进制表示最大值=FFFF.FFFFFFFF  ---控制文件，日志文件等数据库文件中都是16进制表示scn的

低位使用

0000.00000000

0000.00000001

0000.00000011

......

0000.FFFFFFFF+1 ==》0001.0000000

高位使用

0001.0000003

......

0011.0000002

**scn: 0x0000.002736f0**

=====================================

》checkpoint怎么理解

checkpoint怎么理解

一个事件：人物： ckpt process

                 地点：rba

                 时间： scn

                 通道：thread

checkpoint：一个特殊时间点的scn，确保修改数据库被写入磁盘，修改数据对应产生的redo都以RBA形式 存放在了logfile中。

ckpt process=  通过thread#==》在scn时间点==》完成了操作（生成的redo写入磁盘，其对应数据位置为 rba）

检查点的存在就是为了缩短恢复时间

DBWn为异步懒惰写， 数据库可能因为断电而 Crash，那么内存中修改过的、尚未写入数据文件 的数据将会丢失

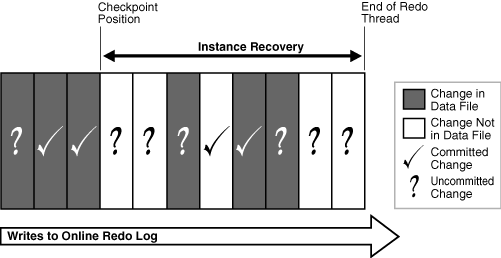
在下一次数据库启动之后，Oracle 可以通过重做日志（redo）进行事务重演 （也就是进行前滚），将数据库恢复到崩溃之前的状态

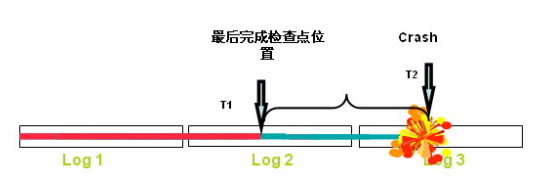
大家最关心的是数据库要经历多久才能打开。也就是需要读取 多少重做日志才能完成前滚。当然我们希望这个时间越短越好

Oracle正是通过检查点来优化这个过程，缩短恢复时间

检查点发生时会触发DBWn，将所有的dirty buffer 写入到磁盘， CKPT 进程则会相应更新控制文件和数据文件头，记录检查点信息，标识变 更。

也就是说，检查点之前的数据是永久的，所需要恢复的起始点是最后一个检查点位置（checkpoint scn）





由上图可知，crash到最后检查点之间的数据是需要恢复的，而这之间的距离越短，则恢复的数据越少，恢复时间越短

也就是说，检查点的频率越高，恢复时间越短，但是频率越高，性能也越低。

数据库参数fast\_start\_mttr\_target间接控制了检查点频率，值越小，恢复越快，检查点频率越高

**从 Oracle10g 开始，数据库可以实现自动调整的检查点（SelfTune Checkpoint）， 不建议修改此参数**

=================================

》增量检查点和全量检查点

为什么在shutdown时，执行这两个命令

SQL> alter system switch logfile;

SQL> alter system checkpoint;

体系结构中讲到dirty buffer到达LRU list冷端末尾，会被放到写列表中，这个列表叫做 脏缓冲列表（LRUW List）

Dirty Buffer 会被转移到 LRUW List，以便 将来执行的检查点可以将这些修改过的 Buffer 写出到数据文件上。

buffer在LRUW List上位置不固定，没有严格的顺序，如果buffer被反复修改，会由LRUW List》LUR list》LRUW List，这样其位置就从头部转移到尾部，

全量检查点：检查点发生时Oracle无法判断优先写哪些buffer，只能将全部脏buffer写到磁盘上

增量检查点：Oracle8开始引入，演进了新的算法， 引入了检查点队列（Buffer Checkpoint Queue - CKPTQ）机制，每一个dirty buffer

也会被记录到CKPTQ上，按照LRBA （Low RBA - 第一次对此数据块修改对应的 Redo Byte Address）的顺序来排列，

如果一个数据 块进行过多次修改，该数据块在检查点队列上的顺序并不会发生变化（相对于 LRBA，后面修 改的 RBA 被称为 HRBA）。

SQL>  select \* from v$sgastat where upper(name) like '%CHECKPOINT%';

POOL         NAME                              BYTES

------------ ------------------------------------------------------- ----------

shared pool  Checkpoint queue                         205312

shared pool  log\_checkpoint\_timeout                      12360

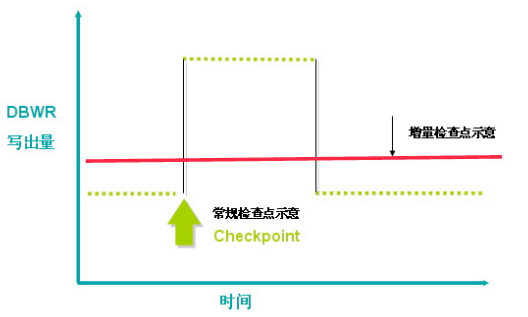
增量检查点时，DBWR 从检查点队列按照 Low RBA 的顺序写出，更新控制文件中Low RBA位置，

CKPT 在进行轻量级更新时，并不会改写控制文件中数 据文件的检查点信息以及数据文件头信息

通过增量检查点，数据库可以将以前的全量写出变更为增量渐进写出，从而可以极大的减少对于数据库性能的影响；

而检查点队列则进一步的将 RBA 和检查点关联起来， 从而可以通过检查点来确定恢复的起点。

除了（Buffer Checkpoint Queue - CKPTQ）外，oracle之后又引入了 文件检查点队列（ FILE QUEUE -FILEQ）和对象检擦点队列（Object Queue - OBJQ）



全量检查点仅在以下条件触发，其他时间都是增量检查点

ALTER SYSTEM CHECKPOINT

SHUTDOWN (除 ABORT 方式外)

=======================

> SCN获取

记住scn和checkpoint无处不在，因为时间无处不在

**系统当前SCN并不是在任何的数据库操作发生时都会改变，SCN通常在事务提交或回滚时改变** **,在控制文件,数据文件头** **,数据块,日志文件头** **,日志文件change vector中都有** **SCN,但其作用各不相同** 。

1 控制文件当前全局scn（10进制）

set linesize 200

select dbid,checkpoint\_change#,current\_scn,scn\_to\_timestamp(current\_scn) from v$database;

      DBID CHECKPOINT\_CHANGE# CURRENT\_SCN SCN\_TO\_TIMESTAMP(CURRENT\_SCN)

---------- ------------------ ----------- ---------------------------------------------------------------------------

1395555907          2452427      2485243 10-JUN-16 12.13.20.000000000 PM

or

SQL> oradebug setmypid

SQL> oradebug DUMPvar SGA kcsgscn\_

kcslf kcsgscn\_ [060019598, 0600195C8) = 00180EAC 00000000 00000000 00000000 000004D0 00000000 00000000 00000000 00000000 00000000 60019278 00000000

Little Endian： SCN base在前，SCN wrap在后

SQL> select to\_number('180EAC','xxxxxx') SCN from dual;

       SCN

----------

   1576620

**注：** **只能对最近5天的scn进行时间转换**

2 日志中所含SCN范围

可以通过视图v$log 查看日志保留scn的范围

对于 Current的日志文件（当前正在被使用的 Redo Logfile） ，其最终 SCN 不可知，所以 Next SCN 被置为无穷大，也就是 0xffffffff。

SQL> select GROUP#,sequence#,STATUS,FIRST\_CHANGE#,to\_char(FIRST\_TIME,'yyyy/mm/dd:hh24:mi:ss') time,next\_change#,to\_char(next\_time,'yyyy/mm/dd:hh24:mi:ss') ntime from V$log;

    GROUP#  SEQUENCE# STATUS           FIRST\_CHANGE# TIME         NEXT\_CHANGE#

---------- ---------- ---------------- ------------- ------------------- ------------

     1       70 INACTIVE             2437034 2016/06/09:22:00:20      2452427

     2       71 CURRENT             2452427 2016/06/10:01:50:49   2.8147E+14

     3       69 INACTIVE             2405652 2016/06/09:12:00:42      2437034

**注：日志文件头中包含了** **Low SCN，Next SCN  --会在日志文件概览中介绍**

3 控制文件中记录的数据文件checkpoint scn

当前数据文件检查点SCN。来自控制文件信息，在控制文件中,表示该数据文件最近一次执行检查点操作时的SCN

set linesize 200

col name format a55

select file#,name,checkpoint\_change#,to\_char(checkpoint\_time,'yyyy-mm-dd hh24:mi:ss'),last\_change# from v$datafile;

     FILE# NAME                           CHECKPOINT\_CHANGE# TO\_CHAR(CHECKPOINT\_ LAST\_CHANGE#

---------- -------------------------------------------------- ------------------ ------------------- ------------

     1 +DATA/orcl/datafile/system.256.867775087             2452427 2016-06-10 01:50:49

     2 +DATA/orcl/datafile/sysaux.257.867775089             2452427 2016-06-10 01:50:49

......

**注： LAST\_CHANGE#，如果数据库非正常关闭值为NULL。正常关闭：设置为关闭时的检查点SCN=关闭时CHECKPOINT\_CHANGE#**

4 数据文件的checkpoint scn

查询数据文件头检查点SCN，来自数据文件头信息

SQL> select name,checkpoint\_change#,CHECKPOINT\_COUNT from v$datafile\_header;

NAME                           CHECKPOINT\_CHANGE# CHECKPOINT\_COUNT

-------------------------------------------------- ------------------ ----------------

+DATA/orcl/datafile/system.256.867775087              2452427           184

+DATA/orcl/datafile/sysaux.257.867775089              2452427           184

+DATA/orcl/datafile/undotbs1.258.867775089              2452427           108

**注： 这里的CHECKPOINT\_CHANGE#，CHECKPOINT\_COUNT都是很重要的数据结构**

v$database: current\_scn  当前scn

v$datafile：CHECKPOINT\_CHANGE# ：控制文件起始检查点，LAST\_CHANGE#： 控制文件结束检查点

v$datafile\_header :CHECKPOINT\_CHANGE#  ： 数据文件头检查点信息，CHECKPOINT\_COUNT 数据文件头检查点计数

之前在控制文件概览中已经讨论过数据库初始化的验证，这里结合检查点重申

1 首先根据控制文件头CHECKPOINT\_COUNT和数据文件头CHECKPOINT\_COUNT对比判断，如果相等,进行第二次检查.

2 需要对比v$database： **LAST\_CHANGE#** 和v$datafile\_header: checkpoint\_change# 来是否需要进行恢复。

3 根据low SCN 和next SCN 来确定需要的恢复信息位于哪一个在线日志或归档日志中。

注意：详细的对比过程会在数据库初始化一节中介绍 。

=====================================

**》ORA-19706: invalid SCN！！！！！！！！！！！！！！！！！**

**什么是SCN最大值！**

SQL> select trunc(power(2,48)/12/31/24/3600/16/1024,2) Year from dual;

YEAR

--------------------

       534.51

SCN 的 6 Bytes 记录, 理论上可以存储 281 trillion (兆) 的数值,这是 SCN 的极限值:

理论上SCN 至少可以用 500 年左右，使用完之后，需要重新建库。

**什么是SCN Limit**

Oracle 11g 以前， 内部控制SCN的合理生成频率大约为16K/秒，这样就有一个当前最大合理值 （MRS）

这个数字在 11g 中增加到32K/秒，最大值可以达到 256K/秒。

\_max\_reasonable\_scn\_rate         Max reasonable SCN rate                32768               TRUE      FALSE    FALSE

**什么是SCN最大合理值（MRS）**

Oracle用约4G的空间来表示scn，SCN的时间范围为 01/01/1988 00:00:00 ~ 08/18/2121 06:28:15，每个月31天，每增加1s，scn增加1

根据每秒允许的最大scn增长率16k/s来算，当前有一个**SCN最大合理值**

**什么是SCN Headroom**

当前的scn值到最大合理值（MRS）之间的距离，11g之后因为SCN合理生成率变大，SCN headroom也变得更大

**scn异常增长问题**

当通过dblink进行业务提交的时候，由于数据库之间存在不同的SCN，因此，为了让事务一致，Oracle将会以两者之间较大的SCN来进行同步

SCN的异常就会通过dblink传染到其他相关的数据库，而dblink使用的频率越大，这种传染的速度也就越快。

**scn bug**

如果SCN异常增长的很大，会导致SCN Headroom变小，如果SCN Headroom变为0 则表示达到了最大合理值（MRS）不允许再有新的scn产生，会造成数据库hang，无法工作

且SCN异常增长可能导致SCN被很快的用尽

**ORA-19706: invalid SCN**

2012年1月之后发布的CPU或PSU补丁修复了此bug，安装补丁后，其有一个隐含参数

\_external\_scn\_rejection\_threshold\_hours 默认值为24h

此隐含参数避免SCN Headroom变为0，其scn变化如果超过隐含参数值，则拒绝DBLINK连接，报错**ORA-19706: invalid SCN，即这个错误是我们所愿意看到的**

如果因为此补丁影响了业务的运行，可将隐含参数值改为1，最大限度的允许dblink连接

Oracle Database 12.2 中，为了更彻底的解决SCN问题，Oracle 通过引入 BigSCN 的新特性，最终改变了 SCN 的算法。

[https://mp.weixin.qq.com/s?\_\_biz=MjM5MDAxOTk2MQ==&mid=2650278190&idx=1&sn=e3752b62a11d1751a4aed5653dfd6a85&chksm=be4791388930182e9266f5340815a760d6b8b00bf2831bb85cc204642715c36c9934860e2656&mpshare=1&scene=23&srcid=03267OTJJddu2qEta9AkZcop#rd](https://mp.weixin.qq.com/s?__biz=MjM5MDAxOTk2MQ==&mid=2650278190&idx=1&sn=e3752b62a11d1751a4aed5653dfd6a85&chksm=be4791388930182e9266f5340815a760d6b8b00bf2831bb85cc204642715c36c9934860e2656&mpshare=1&scene=23&srcid=03267OTJJddu2qEta9AkZcop" \l "rd)

##########################################

SQL> select current\_scn,scn\_to\_timestamp(current\_scn) from v$database;

CURRENT\_SCN SCN\_TO\_TIMESTAMP(CURRENT\_SCN)

----------- ----------------------------------------------------------

    5743808 23-JUL-15 04.30.09.000000000 PM

1s=1000ms

1ms=1000μs

1μs=1000ns

10的-9次方 达到了ns级！！

##########################################

下面看一下控制文件中的scn记录

dump control file(dump转储控制文件)

alter session set tracefile\_identifier='ctl';

alter session set events 'immediate trace name controlf  level 8';

select \* from v$diag\_info where lower(name) like '%default%';

以上看到的控制文件中的检查点信息，在此文件中均可找到对应关系，包括检查点scn

DUMP OF CONTROL FILES, Seq # 7619 = 0x1dc3

\*\*\* 2016-06-09 21:26:55.178

V10 STYLE FILE HEADER:

    Compatibility Vsn = 186646528=0xb200000

    Db ID=1395555907=0x532e7e43, Db Name='ORCL'

    Activation ID=0=0x0

**Control Seq=7619=0x1dc3**, File size=594=0x252    --控制文件发生改变时，控制文件的版本号

    File Number=0, Blksiz=16384, File Type=1 CONTROL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DATABASE ENTRY

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 (size = 316, compat size = 316, section max = 1, section in-use = 1,

  last-recid= 0, old-recno = 0, last-recno = 0)

 (extent = 1, blkno = 1, numrecs = 1)

 12/31/2014 16:39:31

 DB Name "ORCL"

 Database flags = 0x40404000 0x00001000

 Controlfile Creation Timestamp  12/31/2014 16:39:34

 Incmplt recovery scn: 0x0000.00000000

 Resetlogs scn: 0x0000.000f30dc Resetlogs Timestamp  12/31/2014 16:39:34

 Prior resetlogs scn: 0x0000.00000001 Prior resetlogs Timestamp  09/17/2011 09:46:04

 Redo Version: compatible=0xb200000

 #Data files = 6, #Online files = 6

**Database checkpoint: Thread=1 scn: 0x0000.002736f0  --全局检查点scn**

 Threads: #Enabled=1, #Open=1, Head=1, Tail=1

 enabled  threads:  01000000 00000000 00000000 00000000 00000000 00000000

scn: 0x0000.002736f0

select to\_number('2736f0','xxxxxxxxxx') from dual;     ---scn 16进制转换成10进制

TO\_NUMBER('2736F0','XXXXXXXXXX')

--------------------------------

             2569968

SQL> select current\_scn from v$database;

CURRENT\_SCN

-----------

    2577686

checkpoint scn 肯定小于current scn

......

**Controlfile Checkpointed at scn:  0x0000.00181915 04/12/2018 10:29:08**

**thread:0 rba:(0x0.0.0)**               -- 增量检查点记录之后是 RBA 信息，检查点和 Redo 相关联在这里实现

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CHECKPOINT PROGRESS RECORDS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

(size = 8180, compat size = 8180, section max = 11, section in-use = 0,

  last-recid= 0, old-recno = 0, last-recno = 0)

(extent = 1, blkno = 2, numrecs = 11)

THREAD #1 - status:0x2 flags:0x0 **dirty:35**

**low cache rba:(0x45.de7a.0) on disk rba:(0x45.dee5.0)**

**on disk scn: 0x0000.00251ee4** 06/09/2016 21:26:37

这里 low cache rba（recovery block address）指在 cache 中，最低的 rba 地址，在实例恢复 或者崩溃恢复中，需要从这里开始恢复。

on disk rba 是磁盘上的最高的重做值，在进行恢复时 应用重做至少要达到这个值。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LOG FILE RECORDS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

(size = 72, compat size = 72, section max = 16, section in-use = 3,

  last-recid= 9, old-recno = 0, last-recno = 0)

(extent = 1, blkno = 10, numrecs = 16)

LOG FILE #1:

  name #3: +DATA/orcl/onlinelog/group\_1.261.867775175

Thread 1 redo log links: forward: 2 backward: 0

siz: 0x19000 seq: 0x00000043 hws: 0x2 bsz: 512 nab: 0x1393d flg: 0x0 dup: 1

Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.00237636

**Low scn: 0x0000.0023fcc0 06/08/2016 22:00:17    --日志记录的start scn和next scn**

**Next scn: 0x0000.00242b9a 06/09/2016 00:16:59**

LOG FILE #2:

  name #2: +DATA/orcl/onlinelog/group\_2.262.867775175

Thread 1 redo log links: forward: 3 backward: 1

siz: 0x19000 seq: 0x00000044 hws: 0x2 bsz: 512 nab: 0x139a3 flg: 0x0 dup: 1

Archive links: fwrd: 0 back: 0 Prev scn: 0x0000.0023fcc0

Low scn: 0x0000.00242b9a 06/09/2016 00:16:59

Next scn: 0x0000.0024b514 06/09/2016 12:00:42

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DATA FILE RECORDS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

(size = 520, compat size = 520, section max = 100, section in-use = 6,

  last-recid= 92, old-recno = 0, last-recno = 0)

(extent = 1, blkno = 11, numrecs = 100)

DATA FILE #1:

  name #7: +DATA/orcl/datafile/system.256.867775087

creation size=0 block size=8192 status=0xe head=7 tail=7 dup=1

tablespace 0, index=1 krfil=1 prev\_file=0

unrecoverable scn: 0x0000.00000000 01/01/1988 00:00:00

**Checkpoint cnt:182 scn: 0x0000.0024b514 06/09/2016 12:00:42  --控制文件中记录的数据文件检查点计数和检查点scn和结束scn**

**Stop scn: 0xffff.ffffffff 06/06/2016 09:13:30**

数据文件头中scn记录

DATA FILE #1:

  name #7: +DATA/orcl/datafile/system.256.867775087

creation size=0 block size=8192 status=0xe head=7 tail=7 dup=1

tablespace 0, index=1 krfil=1 prev\_file=0

unrecoverable scn: 0x0000.00000000 01/01/1988 00:00:00

**Checkpoint cnt**:145 **scn: 0x0000.001804d3** 04/12/2018 09:08:11

**Stop scn:** 0xffff.ffffffff 04/11/2018 19:06:22

**Creation Checkpointed at scn**:  0x0000.00000007 09/17/2011 09:46:08

thread:0 rba:(0x0.0.0)

>有关checkpoint queue，可以理解为由脏块组成的一个LRU链表 会在buffer cache>checkpoint queue中介绍

>有关low scn 和next scn 及rba会在 日志文件和rab概览中介绍

>整个恢复过程scn的对比

通过以下参数配置可将检查点信息写入alert log

alter system set log\_checkpoints\_to\_alert=true;

Current log# 3 seq# 43 mem# 0: /opt/oracle/oradata/alexhell/redo03.log

Beginning log switch checkpoint up to RBA [0x2c.2.10], SCN: 8914464526139

文档原创，转载请注明出处-------------------