数据库在open时，可能会因为undo损坏导致open失败，如undo出现坏块

没有备份的话，可尝试event 10513来打开数据库

event 10513的作用不是跳过undo坏块，而是disable transaction recovery，在如下internal note中有说明：

EVENT: 10513 "disable transaction recovery" ( Doc ID 123558.1 )

Explanation:

This event disables deferred transaction recovery which is initiated

by the SMON process.

Levels:

level 2 = transaction recovery disabled

数据库异常关闭后，启动时需要做实例恢复，实例恢复分为两个阶段，cache recover 前滚用到redo，transaction recovery回滚用到undo，回退那些未提交的数据，而此时读取的undo块是损坏的，则smon无法完成实例恢复，报无法open

数据库非常规恢复时，可通过一些EVENT或隐含参数改变数据库的机制，从而达到open的目的

但这些方法使用后，数据库会存在不一致情况，建议将数据库导出，导入重建

Undo损坏导致无法open时，可做以下尝试非常规恢复

1 event 10513

\*.undo\_management='MANUAL'

\*.event = '10513 trace name context forever, level 2'

放弃实例恢复，不读取undo，尝试open

如果以上方式依然无法open，尝试以下方法

2 隐含参数允许undo损坏

UNDO\_MANAGEMENT=MANUAL

 \_CORRUPTED\_ROLLBACK\_SEGMENTS=(SYSSMU1$, \_SYSSMU2$, \_SYSSMU3$, ...etc)

\_ALLOW\_RESETLOGS\_CORRUPTION = TRUE

括号中的Undo段列表从system数据文件中获取

$strings /u01/database/system01.dbf | grep \_SYSSMU | cut -d $ -f 1 | sort -u > listSMU

\_OFFLINE\_ROLLBACK\_SEGMENTS和\_CORRUPTED\_ROLLBACK\_SEGMENTS

请参考：

FAQ On Undo Corruption ( Doc ID 405006.1 )

7.What is the difference between \_offline\_rollback\_segments and \_corrupted\_rollback\_segments?

There are differences between these two parameters:

If the rollback segment is listed in the \_offline parameter list, the transaction table is still read accessible. This is important for delayed block cleanout. If a select statement reads a data block with an open ITL which points to the transaction table of the \_offline rollback segment, the table is still checked.

If the transaction is committed, delayed block clean out occurs. If the transaction is uncommitted,

it will generate ORA-1578. It will not allow to select the block.

Rollback segments listed in \_OFFLINE\_ROLLBACK\_SEGMENTS are not subject to all the the normal safeguards. In particular there is no check for active transactions before dropping the rollback segment.

If the rollback segment is listed in the \_corrupted parameter list, the transaction table is not read accessible.

All transactions are assumed COMMITTED and delayed block cleanout will occur but there is

no consistent read view of uncommitted transactions (logical corruption).

 Be aware that the use of \_OFFLINE\_ROLLBACK\_SEGMENTS may lead to the recreation

of the database, depending on whether there were active transactions in the

dropped undo segments. If so, then this may lead to logical corruption, and

hence to the recreation of the database. (Refer Note:106638.1 that explains

how to check the transaction table : you can use the same SELECT statements)

Be aware that the use of \_CORRUPTED\_ROLLBACK\_SEGMENTS requires the recreation

of the database.

Handling Rollback Segment Corruptions in Oracle7.3 to 8.1.7 ( Doc ID 106638.1 )

\_offline\_rollback\_segment 和\_corrupt\_rollback\_segment 的作用不一致，有些时候必须用\_corrupt\_rollback\_segment 才能强制打开损坏的数据库。

\_OFFLINE\_ROLLBACK\_SEGMENTS 的作用：

允许数据库的逻辑corruption，用了这个参数之后，在删除回滚段时，Oracle不会检查这个回滚段是否有active的 transaction, 你可以删除一个拥有active transaction的rollback segment，这时会导致字典表corruption。 这个参数也可以阻止SMON对某些回滚段中的事务进行恢复，所以当SMON在recovery transaction时如果发生异常导致数据库不断crash，那么这时使用这个参数有用。

\_CORRUPTED\_ROLLBACK\_SEGMENTS 的作用：

Oracle不会访问这个参数中列出的任何回滚段的段头，而且认为这些回滚段的所有事物都是commit的。

对于使用了\_OFFLINE\_ROLLBACK\_SEGMENTS的回滚段，它们的 transaction table 仍然可以读取。但是对于\_CORRUPTED\_ROLLBACK\_SEGMENTS 中列出的回滚段，它们的transaction table就不被读取了。

如果不是万不得已，要谨慎使用上面的这些参数，尤其是\_CORRUPTED\_ROLLBACK\_SEGMENTS，因为使用了\_CORRUPTED\_ROLLBACK\_SEGMENTS之后的数据库Oracle就不支持了，必须要重建。

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

 不行的话只能建议在不offline数据文件的情况下尝试强制打开的步骤，因为这种情况理论上还是有可能通过强制打开解决的（只要数据文件自身没有损坏）：

 低端存储异常掉电会丢失写cache内容，那么由于redo的缺失会使数据文件无法recover，所以此时强制打开应该可以解决。

 如果强制打开（包括推进SCN之后）也不行，那么就只能是使用bbed或者dul等外部工具尽可能恢复数据了，而bbed或者dul是我们ACS支持的，global support这边只提供有备份的、以及误删数据文件后通过拷贝文件句柄来恢复的恢复支持。

注意：

++ force open的数据库属于不稳定的状态，如果成功打开，请务必立刻导出并重建这个库。

Dear customer,

 Since no available backup, please try to use the following steps to force open database

 0.Please shutdown your database and backup all the datafiles belongs to the database before any actions below !!!!

 1. Get all the rollback segment information:

 $strings /u01/database/system01.dbf | grep \_SYSSMU | cut -d $ -f 1 | sort -u > listSMU

 ==>Please replace the path and system01.dbf to the datafile name of the system tablespace.

If files are in ASM, you can use RMAN command to copy them to filesystem before using strings command, e.g. :

 RMAN> backup as copy datafile 1 format '/tmp/system01.dbf' ;

/\*或者，尝试复制一份system.01.dbf，将新的复制的文件用windows的记事本工具打开，并搜索关键字为 \_SYSSMU 并保留所有 \_SYSSMU的字段。

结果应该是类似这样的

\_SYSSMU1$, \_SYSSMU2$, \_SYSSMU3$, \_SYSSMU4$, \_SYSSMU5$, \_SYSSMU6$, \_SYSSMU7$, \_SYSSMU8$, \_SYSSMU9$, \_SYSSMU10$

或者版本不同结果会是

\_SYSSMU10\_1999223151,\_SYSSMU11\_1398542477......

------\*/

 2. Create pfile:

 $sqlplus / as sysdba

 SQL>create pfile from spfile;

 3. Add the following parameter in init.ora:

 UNDO\_MANAGEMENT=MANUAL     --允许undo损坏

 \_CORRUPTED\_ROLLBACK\_SEGMENTS=(SYSSMU1$, \_SYSSMU2$, \_SYSSMU3$, ...etc)   --允许undo损坏

 ==>In the generated file listSMU, there will be many rollback segments listed, please add these rollback segments to the ()

 Please don't forget to rename the \_SYSSMU9 to \_SYSSMU9$ (add $ in the end)...

\_ALLOW\_RESETLOGS\_CORRUPTION = TRUE    --允许redo损坏

/\*Automatic Undo segments 的内容来自第一步的输出，

只保留\_SYSSMU的字段即可（如\_SYSSMU10\_1999223151 =》\_SYSSMU10$）

不过多填一些没有关系。比如，实际上是到\_SYSSMU10$，但是填一些不存在\_SYSSMU11$，\_SYSSMU12$之类的，不会影响后面的操作。

如果strings命令没有获得结果，可以自己多填一些。\*/

 4. Startup the database using pfile:

 SQL>startup mount pfile='/..../initORCL.ora'

 SQL>show parameter corrupt;

 5. Recover and open the database:

 SQL>RECOVER DATABASE UNTIL CANCEL;

 Cancel

 SQL>ALTER DATABASE OPEN RESETLOGS;

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

其他情况  前调scn

 6.If the ora-00600 [2662] errors reported again then adjust the scn as below :

/\*ora-00600 [2662]是由于SCN需要推进（块的SCN比CURRENT SCN大，这是open resetlogs导致的结果），可以用10015 event推进下，如果推进多次仍然不够大，也可以通过设置\_minimum\_giga\_scn隐含参数来大幅推进（但\_minimum\_giga\_scn在11.2版本已被废弃）。 \*/

 6.1.Shutdown the database :

 SQL>shutdown immediate;

 6.2.Startup the database to mount:

 SQL>startup mount pfile=<the file has been modified already >Z

 6.3.Adjust it again

 SQL> ALTER SESSION SET EVENTS '10015 TRACE NAME ADJUST\_SCN LEVEL 1';

 SQL> ALTER SESSION SET EVENTS '10015 TRACE NAME ADJUST\_SCN LEVEL 1';

This event is also useful for triggering the SCN to be bumped on database open to avoid SCN mismatch problems.

LEVEL: Level 1 is usually sufficient - it raises the SCN to 1 billion (1024\*1024\*1024)

adjust scn 如何确定前调还是推进scn？

scn只能向前推进的，event 10015适用于需要推进current SCN的情况

 6.4.Open the database

 SQL>alter database open;

 7. export the database using exp then import it to a new instance using imp;

 Thanks!

 Regards,

 Simon

\_minimum\_giga\_scn隐含参数来大幅推进scn的方法

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

来自 <<https://support.oracle.com/epmos/faces/SrDetail?_adf.ctrl-state=uq6oui6s2_4&srDetailRelativeDateParam=false&queryModeName=Technical&srNumber=3-11195200851&needSrDetailRefresh=true&_afrLoop=486444553677607>>