```
Sat Jan 28 19:38:04 2012
Completed: CREATE CONTROLFILE REUSE DATABASE "ORCL" RESETLOGS
```

这个创建是成功的，我们推测，用户或许认为，SYSTEM 表空间是数据库的核心，重建控制文件包含一个文件即可，但是显然这个想法是错误的。

此后，用户在参数文件中设置了一系列的隐含参数，用于强制 Open 打开数据库：

```
SYS auditing is disabled
Starting up ORACLE RDBMS Version: 9.2.0.1.0.
System parameters with non-default values:
  processes                = 500
  timed_statistics         = TRUE
  event                    = 10513 trace name context forever,level 10,
                             10512 trace name context forever,level 10,
                             10511 trace name context forever,level 10,
                             10510 trace name context forever,level 10,
                             10015 trace name context forever, level 10,
                             10269 trace name context forever, level 10,
                             10061 trace name context forever, level 10
  shared_pool_size         = 134217728
  large_pool_size          = 268435456
  java_pool_size           = 33554432
  control_files            = c:\zl\control01.ctl
  db_block_size            = 8192
  db_cache_size            = 629145600
  compatible               = 9.2.0.0.0
  db_file_multiblock_read_count= 16
  fast_start_mttr_target   = 300
  _allow_resetlogs_corruption= TRUE
  _allow_read_only_corruption= TRUE
  _offline_rollback_segments= _SYSSMU1$, _SYSSMU2$, _SYSSMU3$, _SYSSMU4$, _SYSSMU5$,
_SYSSMU6$, _SYSSMU7$, _SYSSMU8$, _SYSSMU9$, _SYSSMU10$
  _corrupted_rollback_segments= _SYSSMU1$, _SYSSMU2$, _SYSSMU3$, _SYSSMU4$, _SYSSMU5$,
_SYSSMU6$, _SYSSMU7$, _SYSSMU8$, _SYSSMU9$, _SYSSMU10$
```

```
undo_management          = MANUAL
undo_tablespace          = system
undo_retention           = 0
remote_login_passwordfile= NONE
db_domain                =
global_names             = FALSE
instance_name            = orcl
dispatchers              = (PROTOCOL=TCP) (SERVICE=orclXDB)
job_queue_processes      = 10
_system_trig_enabled     = FALSE
```

在之后强制打开数据库时，Oracle 进行自动的数据字典检查，发现大量控制文件中没有的数据文件，并强制将这些文件添加到控制文件中：

```
Sat Jan 28 19:44:35 2012
SMON: enabling cache recovery
Sat Jan 28 19:44:35 2012
Dictionary check beginning
<注意以下文件被添加入控制文件，删减部分信息>
Tablespace 'UNDOTBS1' #1 found in data dictionary,
but not in the controlfile. Adding to controlfile.
Tablespace 'TEMP' #2 found in data dictionary,
but not in the controlfile. Adding to controlfile.
Tablespace 'CWMLITE' #3 found in data dictionary,
but not in the controlfile. Adding to controlfile.
Tablespace 'DIGITALSCANDATA' #12 found in data dictionary,
but not in the controlfile. Adding to controlfile.
Tablespace 'DIGITALSCANDB' #13 found in data dictionary,
but not in the controlfile. Adding to controlfile.
<注意以下系列文件，被以 Offline 离线方式加入到数据库中>
File #2 found in data dictionary but not in controlfile.
Creating OFFLINE file 'MISSING00002' in the controlfile.
This file can no longer be recovered so it must be dropped.
File #3 found in data dictionary but not in controlfile.
Creating OFFLINE file 'MISSING00003' in the controlfile.
```

```
This file can no longer be recovered so it must be dropped.
File #4 found in data dictionary but not in controlfile.
Creating OFFLINE file 'MISSING00004' in the controlfile.
<注意，数据库提示这些文件不能被恢复，只能被删除>
File #32 found in data dictionary but not in controlfile.
Creating OFFLINE file 'MISSING00032' in the controlfile.
This file can no longer be recovered so it must be dropped.
<受限于创建控制文件的 MAXDATAFILES 参数设置，不能添加更多的文件到数据库中，添加数据文件工作中断>
Sat Jan 28 19:44:36 2012
Errors in file f:\oracle\admin\orcl\udump\orcl_ora_2056.trc:
ORA-01176: 控制文件允许数据字典具有多于 32 个文件

Error 1176 happened during db open, shutting down database
USER: terminating instance due to error 1176
Instance terminated by USER, pid = 2056
ORA-1092 signalled during: alter database open resetlogs...
```

遗憾的是，这些信息并未引起用户的重视，进一步的强制打开被尝试：

```
Sat Jan 28 19:46:31 2012
Database mounted in Exclusive Mode.
Completed: ALTER DATABASE   MOUNT
Sat Jan 28 19:49:43 2012
alter database open resetlogs
Sat Jan 28 19:49:43 2012
ORA-1139 signalled during: alter database open resetlogs...
Sat Jan 28 19:50:53 2012
ALTER DATABASE RECOVER  database until cancel
<在此执行 SYSTEM 表空间的恢复>
Sat Jan 28 19:50:53 2012
Media Recovery Start
Starting datafile 1 recovery in thread 1 sequence 1
Datafile 1: 'H:\2011\ORADATA\ORCL\SYSTEM01.DBF'
Media Recovery Log
ORA-279 signalled during: ALTER DATABASE RECOVER  database until cancel  ...
```

```
Sat Jan 28 19:51:03 2012
ALTER DATABASE RECOVER    CANCEL
Sat Jan 28 19:51:03 2012
ORA-1547 signalled during: ALTER DATABASE RECOVER    CANCEL  ...
Sat Jan 28 19:51:03 2012
ALTER DATABASE RECOVER CANCEL
ORA-1112 signalled during: ALTER DATABASE RECOVER CANCEL ...
<用户发出 resetlog 命令强制打开数据库>
Sat Jan 28 19:51:27 2012
alter database open resetlogs
Sat Jan 28 19:51:27 2012
RESETLOGS is being done without consistancy checks. This may result
in a corrupted database. The database should be recreated.
<强制打开的 SCN 时间点>
RESETLOGS after incomplete recovery UNTIL CHANGE 337553167
Resetting resetlogs activation ID 1301553802 (0x4d94228a)
Sat Jan 28 19:51:46 2012
Assigning activation ID 1301534482 (0x4d93d712)
<日志被重置，日志序列从 1 开始重新编号>
Thread 1 opened at log sequence 1
  Current log# 3 seq# 1 mem# 0: C:\ZL\REDO03.LOG
Successful open of redo thread 1.
Sat Jan 28 19:51:46 2012
SMON: enabling cache recovery
Sat Jan 28 19:51:46 2012
<数据字典检查，发现表空间离线>
Dictionary check beginning
File #2 is offline, but is part of an online tablespace.
data file 2: 'E:\ORACLE\DATABASE\MISSING00002'
File #3 is offline, but is part of an online tablespace.
data file 3: 'E:\ORACLE\DATABASE\MISSING00003'
File #4 is offline, but is part of an online tablespace.
……
```

```
File #32 is offline, but is part of an online tablespace.
data file 32: 'E:\ORACLE\DATABASE\MISSING00032'
Sat Jan 28 19:51:46 2012
Errors in file f:\oracle\admin\orcl\udump\orcl_ora_3892.trc:
ORA-01176: 控制文件允许数据字典具有多于 32 个文件
<数据库 Crash，无法正常开启>
Error 1176 happened during db open, shutting down database
USER: terminating instance due to error 1176
Instance terminated by USER, pid = 3892
ORA-1092 signalled during: alter database open resetlogs...
```

在之前的尝试中，ORA-00600 的 4194 错误也是常见的：

```
Fri Jan 20 02:02:04 2012
Errors in file e:\oracle\admin\orcl\udump\orcl_ora_4796.trc:
ORA-00600: 内部错误代码，参数: [4194], [34], [26], [], [], [], [], []

Fri Jan 20 02:02:04 2012
Recovery of Online Redo Log: Thread 1 Group 3 Seq 3 Reading mem 0
  Mem# 0 errs 0: F:\ORACLE\ORADATA\ORCL\REDO03.LOG
Recovery of Online Redo Log: Thread 1 Group 3 Seq 3 Reading mem 0
  Mem# 0 errs 0: F:\ORACLE\ORADATA\ORCL\REDO03.LOG
Fri Jan 20 02:02:05 2012
Errors in file e:\oracle\admin\orcl\udump\orcl_ora_4796.trc:
ORA-00604: 递归 SQL 层 1 出现错误
ORA-00607: 当更改数据块时出现内部错误
ORA-00600: 内部错误代码，参数: [4194], [34], [26], [], [], [], [], []

Error 604 happened during db open, shutting down database
USER: terminating instance due to error 604
Fri Jan 20 02:02:05 2012
Errors in file e:\oracle\admin\orcl\bdump\orcl_pmon_5704.trc:
ORA-00604: error occurred at recursive SQL level
```

在这个案例的处理过程中，主要的错误来自于控制文件的创建，如果我们判断是控制文件的损坏，则完全

可以用正确的脚本重建，并且使用 Noresetlogs 方式来尝试恢复数据，打开数据库，如果日志文件完好，这个恢复会非常的顺利。

以下让我们来分析一下这个过程。

尝试启动挂载数据库，可以发现数据库提供控制文件不一致，其中 1 号控制文件比较新，2 号控制文件陈旧，数据库因为控制文件不一致无法 Mount：

```
SQL> startup mount;
ORACLE instance started.

Total System Global Area  126950956 bytes
Fixed Size                   454188 bytes
Variable Size              92274688 bytes
Database Buffers           33554432 bytes
Redo Buffers                 667648 bytes
ORA-00214: controlfile 'C:\ORACLE\ORADATA\ORA9I\CONTROL01.CTL' version 121362
inconsistent with file 'C:\ORACLE\ORADATA\ORA9I\CONTROL02.CTL' version 121347
```

使用 10046 事件跟踪一下数据库 Mount 的后台过程：

```
SQL> alter session set events '10046 trace name context forever,level 12';
Session altered.
SQL> alter database mount;
alter database mount
*
ERROR at line 1:
ORA-00214: controlfile 'C:\ORACLE\ORADATA\ORA9I\CONTROL01.CTL' version 121362
inconsistent with file 'C:\ORACLE\ORADATA\ORA9I\CONTROL02.CTL' version 121347
```

摘录后台的跟踪文件，与前面的案例类似，可以看到在控制文件一致性判断的过程中，仅读取控制文件的第一个数据块即可完成：

```
PARSING IN CURSOR #1 len=20 dep=0 uid=0 oct=35 lid=0 tim=42638703883 hv=1379354989
ad='6a3c8b8c'
alter database mount
END OF STMT
PARSE #1:c=0,e=1324,p=0,cr=0,cu=0,mis=1,r=0,dep=0,og=4,tim=42638703877
```

```
BINDS #1:
WAIT #1: nam='rdbms ipc reply' ela= 3 p1=5 p2=2147483647 p3=0
WAIT #1: nam='rdbms ipc reply' ela= 331 p1=5 p2=900 p3=0
WAIT #1: nam='rdbms ipc reply' ela= 2102 p1=5 p2=900 p3=0
WAIT #1: nam='control file sequential read' ela= 276 p1=0 p2=1 p3=1
WAIT #1: nam='control file sequential read' ela= 191 p1=1 p2=1 p3=1
WAIT #1: nam='control file sequential read' ela= 210 p1=2 p2=1 p3=1
```

由于数据库无法 Mount，此时无法生成创建控制文件的脚本，我们尝试使用单一的控制文件进行加载数据库：

```
SQL> startup mount;
ORACLE instance started.

Total System Global Area  126950956 bytes
Fixed Size                   454188 bytes
Variable Size              92274688 bytes
Database Buffers           33554432 bytes
Redo Buffers                 667648 bytes
Database mounted.
SQL> show parameter control
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
control_file_record_keep_time        integer     7
control_files                        string      c:\oracle\oradata\ora9i\contro
                                                 l01.ctl
```

此时查询 v$datafile 视图，会发现数据库提示 ORA-00235 错误，控制文件由于并发更新而不允许查询，这说明控制文件上存在不一致的事务，更新状态未完成：

```
SQL> select count(*) from v$datafile;
select count(*) from v$datafile
                     *
ERROR at line 1:
ORA-00235: controlfile fixed table inconsistent due to concurrent update
```

但是这不妨碍生成创建控制文件的脚本，以下一条命令就可以获得准确无误的控制文件创建脚本：

```
SQL> alter database backup controlfile to trace;
Database altered.
```

对于这个数据库，摘录一下重要的信息：

```
CREATE CONTROLFILE REUSE DATABASE "ORCL" NORESETLOGS  NOARCHIVELOG
--  SET STANDBY TO MAXIMIZE PERFORMANCE
    MAXLOGFILES 50
    MAXLOGMEMBERS 5
    MAXDATAFILES 100
    MAXINSTANCES 1
    MAXLOGHISTORY 14520
LOGFILE
  GROUP 1 'F:\ORACLE\ORADATA\ORCL\REDO01.LOG'  SIZE 100M,
  GROUP 2 'F:\ORACLE\ORADATA\ORCL\REDO02.LOG'  SIZE 100M,
  GROUP 3 'F:\ORACLE\ORADATA\ORCL\REDO03.LOG'  SIZE 100M
-- STANDBY LOGFILE
DATAFILE
  'F:\ORACLE\ORADATA\ORCL\SYSTEM01.DBF',
  'F:\ORACLE\ORADATA\ORCL\UNDOTBS01.DBF',
  'F:\ORACLE\ORADATA\ORCL\CWMLITE01.DBF',
  'G:\ORADATA\DIGITALSCANDATA.DAT',
  'G:\ORADATA\DIGITALSCANDB.DAT',
  'F:\ORACLE\ORADATA\ORCL\REPAIRDB.DBF',
  'F:\ORACLE\ORADATA\ORCL\STANDARDDB.DBF',
  'G:\ORADATA\DIGITIALSCANDB.DAT',
  'G:\ORADATA\DIGITALSCANDB01.DAT',
  'F:\ORACLE\ORA92\DATABASE\NETAPPLYTS.DAT',
  'F:\ORADATA\CSETS.DBF',
  'H:\ORADATA\DIGITALSCANDB60.DAT'
CHARACTER SET ZHS16GBK
;
```

通过这个脚本我们可以看到，用户的数据文件分布在多个磁盘目录上，甚至还有文件创建于

$ORACLE_HOME/database 目录下，这样混乱的管理，在备份时想不漏掉文件也是困难的，手工编辑控制文件创建脚本也会变得相当复杂。

但是如果我们能够生成这样一个脚本，通过脚本来创建控制文件，则一切问题都可以避免。

所以我们说，对于一个技术人员，扎实的基本功是非常重要的。

通过转储控制文件的 2 进制内容可以看到数据库更为详细的信息：

```
SQL> alter session set events 'immediate trace name controlf level 12';
Session altered.
```

以下是数据库的一些关键信息，通过检查点信息可以看到数据库最后的 On Disk SCN 时间是 2011 年 12 月 30 日，从那之后，数据库就没有被正常打开过：

```
*******************************************************************************
CHECKPOINT PROGRESS RECORDS
*******************************************************************************
 (blkno = 0x4, size = 104, max = 1, in-use = 1, last-recid= 0)
THREAD #1 - status:0x1 flags:0x0 dirty:0
low cache rba:(0xffffffff.ffffffff.ffff) on disk rba:(0x5501.580c.0)
on disk scn: 0x0000.2e999200 12/30/2011 16:05:47
resetlogs scn: 0x0000.0002e872 07/07/2009 15:53:52
heartbeat: 773955197 mount id: 1301712020
MTTR statistics status: 3
Init time: Avg: 8603925, Times measured: 3
File open time: Avg: 10283, Times measured: 174
Log block read time: Avg: 2, Times measured: 80611
Data block handling time: Avg: 247, Times measured: 10661
```

这是一次处置不当带来的复杂恢复，属于不应当出现的故障，由于部分文件在备份时以后，后期的恢复时又改写了部分文件，使得完全恢复不再可能，我们通过本书前面描述的方法修改文件头部信息可以完成恢复。

尺有所短，物有不足

硬件故障导致的灾难一则

我们在和系统打交道时，要清楚的知道各种设备的不足之处，扬长避短，这样才能够规避一些可能存在的问题，维持系统的稳定和安全。

2011 年遇到的一个金融客户案例，客户的存储系统，持续运做了近 7 年，最终彻底崩溃，硬盘灯常绿，但是热备盘已经损坏，Raid 5 里又同时损坏 2 块硬盘。

实际上大家都应该具备这样的经验：通常硬件的稳定期在 3 年左右，通常这段时间系统硬件会相当稳定，故障率低，而一旦超过 3 年，就进入了故障多发期，可能会遇到较多的故障。在规划 IT 系统时，应当在 3 ~ 5 年左右时考虑更替硬件，规避可能出现的重大故障。

硬件和人一样，过劳都会出问题。如果我们忽视硬件的生命周期，则很有可能在关键时刻遭遇灾难。

灾难描述

这次故障发生的过程很简单：

1. 客户数据库崩溃
2. 检查故障时发现系统读写存储时报警，存储离线
3. 数据库未进行及时有效备份
4. 客户寻找第三方支持寻求故障处理

硬件故障不是人力所能控制的，套用一句俗话：所有的系统要么已经出现故障，要么正在走向故障。我们能够做的工作就是加强维护监控，并且在必要时及时更新硬件，将故障点不断推迟下去。

案例警示

这一则案例给我们的借鉴有：

1. **关键时刻要保护现场寻求支持**

这个客户具备非常专业的素质和判断，在数据库出现问题时，首先保护现场，进行认真的分析，当客户觉得事故超出现有资源的控制能力后，接下来就通过各种渠道寻找技术专家来进行判断决策，在此过程中客户未进行任何盲目的恢复尝试。

正是由于客户的冷静判断，认真分析，正确决策，才有了故障的快速恢复与解决，这是任何客户都应该学习借鉴的。

2. **要尊重硬件的使用年限和生命周期**

在系统规划时一定不能忽视硬件的生命周期，有时候可能超限使用的硬件一切看起来正常无误，但是一旦听之任之，等着硬件去出现故障，就有可能遭遇严重灾难。

所以，即便在系统完全稳定运行时，也应当做出足够的提前预估，进行硬件设备的维护升级、更新换代。

在这个案例中，硬件是故障根源，但是用户的准确判断最终成为了数据安全的保障。

技术回放

在操作系统的日志中，我们可以找到硬件的出错信息，出现了 IO 读写错误：

```
Nov 19 01:08:03 DBSVRA scsi:
[ID 107833 kern.warning] WARNING: /pci@8,600000/pci@1/scsi@4/sd@0,0 (sd31):
Nov 19 01:08:03 DBSVRA    SCSI transport failed: reason 'reset': retrying command
Nov 19 01:08:03 DBSVRA scsi:
[ID 107833 kern.warning] WARNING: /pci@8,600000/pci@1/scsi@4/sd@0,1 (sd63):
Nov 19 01:08:03 DBSVRA    SCSI transport failed: reason 'reset': retrying command
Nov 19 01:08:09 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: read error on /dev/did/dsk/d4s0
Nov 19 01:08:09 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d102: read error on /dev/did/dsk/d5s0
Nov 19 01:08:15 DBSVRA last message repeated 1 time
Nov 19 01:08:15 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: read error on /dev/did/dsk/d4s0
Nov 19 01:08:21 DBSVRA md_stripe:
```

```
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: write error on /dev/did/dsk/d4s0
Nov 19 01:08:21 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d102: read error on /dev/did/dsk/d5s0
Nov 19 01:08:27 DBSVRA last message repeated 1 time
Nov 19 01:08:27 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: read error on /dev/did/dsk/d4s0
Nov 19 01:08:36 DBSVRA last message repeated 1 time
Nov 19 01:08:39 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: write error on /dev/did/dsk/d4s0
Nov 19 01:08:45 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: read error on /dev/did/dsk/d4s0
Nov 19 01:09:00 DBSVRA last message repeated 6 times
Nov 19 01:09:00 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: write error on /dev/did/dsk/d4s0
Nov 19 01:09:03 DBSVRA md_stripe:
[ID 641072 kern.warning] WARNING: md: dbsvr/d101: read error on /dev/did/dsk/d4s0
Nov 19 01:09:03 DBSVRA scsi:
[ID 107833 kern.warning] WARNING: /pci@8,600000/pci@1/scsi@4/sd@0,0 (sd31):
```

在这样的情况下，虽然 IO 读写出现问题，但是通常物理硬盘没有损坏，通过磁盘级别的恢复和数据提取可以恢复出数据文件，再对文件进行简单修复即可挽救数据。

这个用户的操作系统是 Solaris ，使用 UFS 文件系统，我们首先通过独立的盘阵挂接故障盘，进行磁盘镜像，保护原有数据盘，然后通过镜像盘进行恢复，非常安全和可靠。

恢复完成的数据在启动时遇到 4193 错误，通过重建 UNDO 表空间得以解决，最后数据库顺利的恢复了运行：

```
Mon Nov 21 18:24:19 2011
Errors in file /oracle/admin/db/bdump/db_smon_13638.trc:
ORA-01595: error freeing extent (3) of rollback segment (25))
ORA-00607: Internal error occurred while making a change to a data block
ORA-00600: internal error code, arguments: [4193], [31866], [5416], [], [], [], [],
[]
SMON: about to recover undo segment 27
SMON: mark undo segment 27 as needs recovery
```

这个案例的顺利恢复得益于用户的正确判断，未进行任何混乱的尝试，值得其他用户借鉴。

附录一 BBED 的说明

在 Linux 和 Unix 上，BBED 缺省未安装，需要通过编译生成可执行文件，以下 make 用于生成 BBED，范例为 Oracle 10g：

```
[oracle@danaly lib]$cd $ORACLE_HOME/rdbms/lib
[oracle@danaly lib]$make -f ins_rdbms.mk $ORACLE_HOME/rdbms/lib/bbed
Linking BBED utility (bbed)
rm -f /opt/oracle/product/10.2.0/rdbms/lib/bbed
gcc -o /opt/oracle/product/10.2.0/rdbms/lib/bbed ...
   -L/opt/oracle/product/10.2.0/lib
```

以下为 BBED 的默认帮助：

```
[oracle@danaly lib]$ bbed
Password:

BBED: Release 2.0.0.0.0 - Limited Production on Sun Sep 3 12:42:59 2006

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

************* !!! For Oracle Internal Use only !!! ***************
BBED> help ALL
SET DBA [ dba | file#, block# ]
SET FILENAME 'filename'
SET FILE file#
SET BLOCK [+/-]block#
SET OFFSET [ [+/-]byte offset | symbol | *symbol ]
SET BLOCKSIZE bytes
SET LIST[FILE] 'filename'
SET WIDTH character_count
SET COUNT bytes_to_display
```

```
SET IBASE [ HEX | OCT | DEC ]
SET OBASE [ HEX | OCT | DEC ]
SET MODE  [ BROWSE | EDIT ]
SET SPOOL [ Y | N ]
SHOW [  | ALL ]
INFO
MAP[/v] [ DBA | FILENAME | FILE | BLOCK ]
DUMP[/v] [ DBA | FILENAME | FILE | BLOCK | OFFSET | COUNT ]
PRINT[/x|d|u|o|c] [ DBA | FILE | FILENAME | BLOCK | OFFSET | symbol | *symbol ]
EXAMINE[/Nuf] [ DBA | FILE | FILENAME | BLOCK | OFFSET | symbol | *symbol ]
:
N - a number which specifies a repeat count.
u - a letter which specifies a unit size:
  b - b1, ub1 (byte)
  h - b2, ub2 (half-word)
  w - b4, ub4(word)
  r - Oracle table/index row
f - a letter which specifies a display format:
  x - hexadecimal
  d - decimal
  u - unsigned decimal
  o - octal
  c - character (native)
  n - Oracle number
  t - Oracle date
  i - Oracle rowid
FIND[/x|d|u|o|c] numeric/character string [ TOP | CURR ]
COPY [ DBA | FILE | FILENAME | BLOCK ] TO [ DBA | FILE | FILENAME | BLOCK ]
MODIFY[/x|d|u|o|c] numeric/character string
      [ DBA | FILE | FILENAME | BLOCK | OFFSET | symbol | *symbol ]
ASSIGN[/x|d|u|o] =
 : [ DBA | FILE | FILENAME | BLOCK | OFFSET | symbol | *symbol ]
 : [ value |  ]
```

```
SUM [ DBA | FILE | FILENAME | BLOCK ] [ APPLY ]
PUSH [ DBA | FILE | FILENAME | BLOCK | OFFSET ]
POP [ALL]
REVERT [ DBA | FILE | FILENAME | BLOCK ]
UNDO
HELP [  | ALL ]
VERIFY [ DBA | FILE | FILENAME | BLOCK ]
CORRUPT [ DBA | FILE | FILENAME | BLOCK ]
BBED> exit
```

Oracle Database 11g 中缺省的未提供 BBED 库文件，但是可以用 10g 的文件编译出来，参考如下步骤：

1.复制 Oracle 10g 文件

```
Copy $ORA10g_HOME/rdbms/lib/ssbbded.o to $ORA11g_HOME/rdbms/lib
Copy $ORA10g_HOME/rdbms/lib/sbbdpt.o to $ORA11g_HOME/rdbms/lib

Copy $ORA10g_HOME/rdbms/mesg/bbedus.msb to $ORA11g_HOME/rdbms/mesg
Copy $ORA10g_HOME/rdbms/mesg/bbedus.msg to $ORA11g_HOME/rdbms/mesg
Copy $ORA10g_HOME/rdbms/mesg/bbedar.msb to $ORA11g_HOME/rdbms/mesg
```

2.编译

```
make -f $ORA11g_HOME/rdbms/lib/ins_rdbms.mk BBED=$ORACLE_HOME/bin/bbed $ORACLE_HOME/bin/bbed
```

对于 Windows 平台，BBED 在 9i 的某些版本缺省（如 9.2.0.6）提供，可以提取供以后版本使用。从 Oracle 10g 开始，Windows 平台不再提供 BBED 工具。

附录二 函数 f_get_from_dump

```
CREATE OR REPLACE FUNCTION F_GET_FROM_DUMP
(
P_DUMP IN VARCHAR2,
P_TYPE IN VARCHAR2
)
RETURN VARCHAR2 AS
   V_LENGTH_STR VARCHAR2(10);
   V_LENGTH NUMBER DEFAULT 7;
   V_DUMP_ROWID VARCHAR2(30000);

   V_DATE_STR VARCHAR2(100);
   TYPE T_DATE IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;
   V_DATE T_DATE;

   FUNCTION F_ADD_PREFIX_ZERO (P_STR IN VARCHAR2, P_POSITION IN NUMBER) RETURN VARCHAR2
   AS
      V_STR VARCHAR2(30000) := P_STR;
      V_POSITION NUMBER := P_POSITION;
      V_STR_PART VARCHAR2(2);
      V_RETURN VARCHAR2(30000);
   BEGIN
      WHILE (V_POSITION != 0) LOOP
         V_STR_PART := SUBSTR(V_STR, 1, V_POSITION - 1);
         V_STR := SUBSTR(V_STR, V_POSITION + 1);

         IF V_POSITION = 2 THEN
            V_RETURN := V_RETURN || '0' || V_STR_PART;
         ELSIF V_POSITION = 3 THEN
```

```
            V_RETURN := V_RETURN || V_STR_PART;
          ELSE
            RAISE_APPLICATION_ERROR(-20002, 'DUMP ERROR CHECK THE INPUT ROWID');
          END IF;

          V_POSITION := INSTR(V_STR, ',');
        END LOOP;
        RETURN REPLACE(V_RETURN , ',');
      END F_ADD_PREFIX_ZERO;

    BEGIN
      IF SUBSTR(P_DUMP, 1, 3) = 'Typ' THEN
        V_DUMP_ROWID := SUBSTR(P_DUMP, INSTR(P_DUMP, ':') + 2);
      ELSE
        V_DUMP_ROWID := P_DUMP;
      END IF;

      IF P_TYPE = 'VARCHAR2' OR P_TYPE = 'CHAR' THEN

        V_DUMP_ROWID :=F_ADD_PREFIX_ZERO(V_DUMP_ROWID || ',', INSTR(V_DUMP_ROWID, ','));

        RETURN(UTL_RAW.CAST_TO_VARCHAR2(V_DUMP_ROWID));

      ELSIF P_TYPE = 'NVARCHAR2' OR P_TYPE = 'NCHAR' THEN

        V_DUMP_ROWID :=F_ADD_PREFIX_ZERO(V_DUMP_ROWID || ',', INSTR(V_DUMP_ROWID, ','));

        RETURN(UTL_RAW.CAST_TO_NVARCHAR2(V_DUMP_ROWID));

      ELSIF P_TYPE = 'NUMBER' THEN

        V_DUMP_ROWID :=F_ADD_PREFIX_ZERO(V_DUMP_ROWID || ',', INSTR(V_DUMP_ROWID, ','));
```

```
    RETURN(TO_CHAR(UTL_RAW.CAST_TO_NUMBER(V_DUMP_ROWID)));

  ELSIF P_TYPE = 'DATE' THEN

    V_DUMP_ROWID := ',' || V_DUMP_ROWID || ',';

    FOR I IN 1..7 LOOP
      V_DATE(I) := TO_NUMBER(SUBSTR(V_DUMP_ROWID, INSTR(V_DUMP_ROWID, ',', 1, I) + 1,
        INSTR(V_DUMP_ROWID, ',', 1, I + 1) - INSTR(V_DUMP_ROWID, ',', 1, I) - 1), 'XXX');
    END LOOP;

    V_DATE(1) := V_DATE(1) - 100;
    V_DATE(2) := V_DATE(2) - 100;

    IF ((V_DATE(1) < 0) OR (V_DATE(2) < 0)) THEN
      V_DATE_STR := '-' || LTRIM(TO_CHAR(ABS(V_DATE(1)), '00')) || LTRIM(TO_CHAR(ABS(V_DATE(2)),
'00'));
    ELSE
      V_DATE_STR := LTRIM(TO_CHAR(ABS(V_DATE(1)), '00')) || LTRIM(TO_CHAR(ABS(V_DATE(2)),'00'));
    END IF;

    V_DATE_STR := V_DATE_STR || '-' || TO_CHAR(V_DATE(3)) || '-' || TO_CHAR(V_DATE(4)) || ' ' ||
      TO_CHAR(V_DATE(5) - 1) || ':' || TO_CHAR(V_DATE(6) - 1) || ':' || TO_CHAR(V_DATE(7) - 1);
    RETURN (V_DATE_STR);

  ELSIF ((P_TYPE LIKE 'TIMESTAMP(_)') OR (P_TYPE = 'TIMESTAMP')) THEN

    V_DUMP_ROWID := ',' || V_DUMP_ROWID || ',';

    FOR I IN 1..11 LOOP
      V_DATE(I) := TO_NUMBER(SUBSTR(V_DUMP_ROWID, INSTR(V_DUMP_ROWID, ',', 1, I) + 1,
        INSTR(V_DUMP_ROWID, ',', 1, I + 1) - INSTR(V_DUMP_ROWID, ',', 1, I) - 1), 'XXX');
    END LOOP;
```

```
    V_DATE(1) := V_DATE(1) - 100;
    V_DATE(2) := V_DATE(2) - 100;

    IF ((V_DATE(1) < 0) OR (V_DATE(2) < 0)) THEN
      V_DATE_STR := '-' || LTRIM(TO_CHAR(ABS(V_DATE(1)), '00')) || LTRIM(TO_CHAR(ABS(V_DATE(2)),
'00'));
    ELSE
      V_DATE_STR := LTRIM(TO_CHAR(ABS(V_DATE(1)), '00')) || LTRIM(TO_CHAR(ABS(V_DATE(2)),'00'));
    END IF;

    V_DATE_STR := V_DATE_STR || '-' || TO_CHAR(V_DATE(3)) || '-' || TO_CHAR(V_DATE(4)) || ' ' ||
      TO_CHAR(V_DATE(5) - 1) || ':' || TO_CHAR(V_DATE(6) - 1) || ':' || TO_CHAR(V_DATE(7) - 1) || '.' ||
      SUBSTR(TO_CHAR(V_DATE(8) * POWER(256, 3) + V_DATE(9) * POWER(256, 2) + V_DATE(10) * 256 +
V_DATE(11)),
        1, NVL(TO_NUMBER(SUBSTR(P_TYPE, 11, 1)), 6));
    RETURN (V_DATE_STR);

  ELSIF P_TYPE = 'RAW' THEN

    V_DUMP_ROWID :=F_ADD_PREFIX_ZERO(V_DUMP_ROWID || ',', INSTR(V_DUMP_ROWID, ','));

    RETURN(V_DUMP_ROWID);

  ELSIF P_TYPE = 'ROWID' THEN

    V_DUMP_ROWID :=F_ADD_PREFIX_ZERO(V_DUMP_ROWID || ',', INSTR(V_DUMP_ROWID, ','));

    RETURN (DBMS_ROWID.ROWID_CREATE(
      1,
      TO_NUMBER(SUBSTR(V_DUMP_ROWID, 1, 8), 'XXXXXXXXXXX'),
      TRUNC(TO_NUMBER(SUBSTR(V_DUMP_ROWID, 9, 4), 'XXXXXX')/64),
      TO_NUMBER(MOD(TO_NUMBER(SUBSTR(V_DUMP_ROWID, 9, 4), 'XXXXXX'), 64) ||
```

```
            TO_NUMBER(SUBSTR(V_DUMP_ROWID, 13, 4), 'XXXXXXXXXX')),
          TO_NUMBER(SUBSTR(V_DUMP_ROWID, 17, 4), 'XXXXXX')));

    ELSE
      RAISE_APPLICATION_ERROR(-20001, 'TYPE NOT VALID OR CAN''T TRANSALTE ' || P_TYPE || ' TYPE');
    END IF;

END;
/
```

参考资料

[1]Bumblebee 项目, https://github.com/MrMEEE/bumblebee/commit/a047be85247755cdbe0acce6#diff-1.

[2]一个空格引发的惨剧, http://coolshell.cn/articles/4875.html.

[3]ITPUB 论坛技术讨论：职业生涯误操作篇, http://www.itpub.net/thread-911086-1-1.html.

[4]猜测的力量，刘磊 ACOUG 主题演讲.

九载耕耘奠定专业地位

博文视点诚邀精锐作者加盟

以书为证彰显卓越品质

《代码大全》、《Windows内核情景分析》、《加密与解密》、《编程之美》、《VC++深入详解》、《SEO实战密码》、《PPT演义》……

“圣经”级图书光耀夺目，被无数读者朋友奉为案头手册传世经典。

潘爱民、毛德操、张亚勤、张宏江、咎辉Zac、李刚、曹江华……

“明星”级作者济济一堂，他们的名字熠熠生辉，与IT业的蓬勃发展紧密相连。

九年的开拓、探索和励精图治，成就**博**古通今、**文**圆质方、**视**角独特、**点**石成金之计算机图书的风向标杆：博文视点。

“凤翱翔于千仞兮，非梧不栖”，博文视点欢迎更多才华横溢、锐意创新的作者朋友加盟，与大师并列于IT专业出版之巅。

英雄帖

江湖风云起，代有才人出。
IT界群雄并起，逐鹿中原。
博文视点诚邀天下技术英豪加入，
指点江山，激扬文字
传播信息技术，分享IT心得

• 专业的作者服务 •

博文视点自成立以来一直专注于 IT 专业技术图书的出版，拥有丰富的与技术图书作者合作的经验，并参照IT技术图书的特点，打造了一支高效运转、富有服务意识的编辑出版团队。我们始终坚持：

善待作者——我们会把出版流程整理得清晰简明，为作者提供优厚的稿酬服务，解除作者的顾虑，安心写作，展现出最好的作品。

尊重作者——我们尊重每一位作者的技术实力和生活习惯，并会参照作者实际的工作、生活节奏，量身制定写作计划，确保合作顺利进行。

提升作者——我们打造精品图书，更要打造知名作者。博文视点致力于通过图书提升作者的个人品牌和技术影响力，为作者的事业开拓带来更多的机会。

联系我们

博文视点官网：http://www.broadview.com.cn　　CSDN官方博客：http://blog.csdn.net/broadview2006/
新浪官方微博：http://weibo.com/broadviewbj　　腾讯官方微博：http://t.qq.com/bowenshidian
投稿电话：010-51260888　88254368　　投稿邮箱：jsj@phei.com.cn