# 数据库故障处理应急方案V1.0

云和恩墨(北京)信息技术有限公司

技术顾问 杨俊

http://www.enmotech.com

文档信息

|  |  |  |  |
| --- | --- | --- | --- |
| 文档名称 | 数据库故障处理应急方案V1.0 | | |
| 电子文档 | 数据库故障处理应急方案V1.0 | | |
| 文件状态 | □ 草 稿 ■ 正式发布 □ 正在修改 | | |
| 编 写 人 | 杨俊 | 日 期 | 2016年 3 月21日 |

**更改说明：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序** | **更改版本号** | **更改人** | **日期** | **备注** |
| 1 | 1.0版 | 杨俊 | 2016-3-21 | 初始版本 |
| 2 |  |  |  |  |
| 3 |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编制** | 杨俊 | （签字） | 日期 | 2016-3-21 |
| **校对** |  | （签字） | 日期 |  |
| **审核** |  | （签字） | 日期 |  |
| **批准** |  | （签字） | 日期 |  |

附件清单

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 附件文档名称 | 厂商 | 厂商签字 | 甲方签字 | 备 注 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[数据库故障处理应急方案V1.0 1](#_Toc460575659)

[等待事件类 5](#_Toc460575660)

[“TX,TM,DX”锁应急处理 5](#_Toc460575661)

[“Latch free”应急处理 6](#_Toc460575662)

[“Cache buffer chains”应急处理 6](#_Toc460575663)

[“Library cache lock”应急处理 7](#_Toc460575664)

[“gc buffer busy”应急处理 7](#_Toc460575665)

[“cursor: pin S wait on X”事件应急处理 8](#_Toc460575666)

[“latch: undo global data”事件应急处理 9](#_Toc460575667)

[“enq:US-content” or 回滚表空间使用过度事件应急处理 10](#_Toc460575668)

[“gc buffer busy”问题 11](#_Toc460575669)

[资源消耗类 12](#_Toc460575670)

[高资源消耗进程应急处理 12](#_Toc460575671)

[IO非常高 13](#_Toc460575672)

[PGA使用过大 14](#_Toc460575673)

[CPU使用过高应急处理 15](#_Toc460575674)

[大表索引被删除导致CPU 100%的应急处理 15](#_Toc460575675)

[网络连接类 17](#_Toc460575676)

[连接不上数据库 17](#_Toc460575677)

[ESB库客户端不定时断开连接 17](#_Toc460575678)

[X通过监听无法连接数据库 18](#_Toc460575679)

[空间管理类 18](#_Toc460575680)

[表空间故障应急处理 19](#_Toc460575681)

[SYSAUX表空间爆满没有存储扩容应急处理 20](#_Toc460575682)

[文件系统使用率达到或超过95%应急处理 20](#_Toc460575683)

[临时表空间过度事件应急处理 21](#_Toc460575684)

[后台undo表空间无法自动扩展 22](#_Toc460575685)

[undo表空间无法扩展 23](#_Toc460575686)

[后台库temporary段暴涨 24](#_Toc460575687)

[数据泵进程挂起导致磁盘爆满 25](#_Toc460575688)

[无法启动类 25](#_Toc460575689)

[RAC数据库CRS无法正常启动 25](#_Toc460575690)

[ORACLE GC OMS无法启动 26](#_Toc460575691)

[数据库HANG类 26](#_Toc460575692)

[数据库HANG异常处理 26](#_Toc460575693)

[话务告警库oracle进程HANG住 27](#_Toc460575694)

[数据库CRASH类 28](#_Toc460575695)

[网分后台库实例重启 28](#_Toc460575696)

[实例CRASH应急处理 29](#_Toc460575697)

[传输数据库脑裂问题 29](#_Toc460575698)

[应用操作类 30](#_Toc460575699)

[SQL执行计划变化应急处理 31](#_Toc460575700)

[使用绑定变量SQL突然运行缓慢的应急处理 32](#_Toc460575701)

[数据网管无法插入数据 32](#_Toc460575702)

[前端库应用程序timestamp错误 33](#_Toc460575703)

[其他分类 33](#_Toc460575704)

[服务切换应急处理 33](#_Toc460575705)

[升级PSU系统报错 35](#_Toc460575706)

[数据库在执行过程中使用动态采样 35](#_Toc460575707)

[通过srvctl查看数据库的状态时反应缓慢 36](#_Toc460575708)

# 等待事件类

### “TX,TM,DX”锁应急处理

**现象描述：**

数据库大量锁异常等待，系统资源消耗高，cpu负载高 （针对大量'TX,TM,DX'等类型的锁造成的大量异常等待）

**影响因素:**

多个事务争用造成。

**解决方法**

以下语句列出是谁造成了阻塞

column event format a30

column sess format a20

set linesize 250

set pagesize 0

break on id1 skip 1

select decode(request,0,'Holder:',' Waiter:') || s.inst\_id || ':' || s.sid||','|| s.serial# sess,

id1, id2, lmode, request, l.type, ctime, s.username,s.sql\_id, s.event

-- ,s.service\_name

from gv$lock l, gv$session s

where (id1, id2, l.type) in

(select id1, id2, type from gv$lock where request>0

)

and l.sid=s.sid

and l.inst\_id=s.inst\_id

order by id1, ctime desc, request

/

按照这个语句多查询几次，如果Holder不变，则KILL掉。操作前记录相关日志

### “Latch free”应急处理

**现象描述：**

数据库大量latch free等待，系统资源消耗高，cpu负载接近100%

**影响因素:**

**解决方法:**

1.手动执行hang查杀脚本：

/usr/bin/ksh /home/oracle/auto\_hang\_analyze.sh

观察几分钟，如果latch free等待事件一直没有下降，则执行第二步。

2.查询当前active的会话模块：

select username,machine,count(\*) from v$session where status='ACTIVE' having count(\*)>6 group by username,machine order by 1;

将会话数量过多的模块通知开发商，让他们切换部分业务到另外一个节点

然后进行系统资源监控和数据库监控

### “Cache buffer chains”应急处理

**现象描述：**

数据库大量cache buffer chains等待，系统资源消耗高，cpu负载高

**影响因素:**

A.低效的SQL语句是发生 cache buffers chains(热块争用），锁存器争用的最重要原因。

B.多个进程同时扫描大范围的索引或表时，可能广泛引发cache buffers chains 锁存器争用。

C.应用程序打开执行相同的低效率SQL语句的多个并发会话，并且这些SQL语句都设法得到相同的数据集，这种情景十分普遍。

**解决方法:**

1.查看 latch: cache buffers chains 事件相关的会话信息;

select sid,username,machine,program,p1raw,sql\_id,logon\_time,last\_call\_et from v$session where event='latch: cache buffers chains';

使用ora命令 ora get\_kill\_sh &sql\_id &username 进行查杀.

查杀后记录该sql语句,丢给相应的开发商处理

2、查看哪个SQL执行的次数最多

select sql\_id,count(\*) from v$session where event='latch: cache buffers chains' group by sql\_id order by 2;

### “Library cache lock”应急处理

**现象描述：**

数据库大量library cache lock等待，系统资源消耗高，cpu的idle为0

**影响因素:**

library cache lock出现的情况比较复杂，例如：

A、大量对某个对象访问；

B、shared pool有问题；

**解决方法:**

1、看看是不是某条SQL引起

select sql\_id,count(\*) from v$session where event='library cache lock' group by sql\_id order by 2;

然后分析SQL中的对象和执行计划等，再跟开发商确认，用ora get\_kill\_sh进行杀

2、shared pool的内部结构造成，再开一个窗口用topas监控系统资源，然后清理shared pool

alter system flush shared pool; （该操作需要向直属领导确认）

### “gc buffer busy”应急处理

**现象描述：**

一般的现象为CPU较高，IO较忙，处理方法与cache buffer chains应急处理一样

**影响因素:**

gc buffer busy出现在RAC中，出现概率并不高，因为BOSS是对业务做了分离的，是由于多节点同时大量访问某些数据块引起的

**解决方法:**

1.查看 latch: cache buffers chains 事件相关的会话信息;

select sid,username,machine,program,p1raw,sql\_id,logon\_time,last\_call\_et from v$session where event='gc buffer busy';

使用ora命令 ora get\_kill\_sh &sql\_id &username 进行查杀.

查杀后记录该sql语句,丢给相应的开发商处理

2、查看哪个SQL执行的次数最多

select sql\_id,count(\*) from v$session where event=' gc buffer busy ' group by sql\_id order by 2;

### “cursor: pin S wait on X”事件应急处理

**现象描述：**

**影响因素：**

一般包含以下几种：

1、常见硬解析

2、High Version Counts

3、BUG

**解决方法：**

1、查找等待事件的阻塞者：

select p2raw,to\_number(substr(to\_char(rawtohex(p2raw)),1,8),'XXXXXXXX') sid from v$session where event = 'cursor: pin S wait on X';

2、查看阻塞者在做什么：

select sid,serial#,SQL\_ID,BLOCKING\_SESSION,BLOCKING\_SESSION\_STATUS,EVENT

from v$session where SID=31;

3、根据阻塞者的SQL分析产生原因。

### “latch: undo global data”事件应急处理

**现象描述：**

一个大事务对某个表进行DML操作，使用大量undo空间。大量并发语句发起对这个表的操作，由于一致性读，需要使用undo记录进行回滚，产生latch:undo global data等待，cpu使用率上升

**影响因素：**

一般包含以下几种：

1、大事务对某个表进行DML操作

**解决方法：**

1、查找session使用undo量的SQL：

SELECT r.name rbs,

nvl(s.username, 'None') oracle\_user,

s.osuser client\_user,

p.username unix\_user,

s.sid,

s.serial#,

p.spid unix\_pid,

t.used\_ublk \* TO\_NUMBER(x.value) / 1024 / 1024 as undo\_mb ,

TO\_CHAR(s.logon\_time, 'mm/dd/yy hh24:mi:ss') as login\_time,

TO\_CHAR(sysdate - (s.last\_call\_et) / 86400, 'mm/dd/yy hh24:mi:ss') as last\_txn,

t.START\_TIME transaction\_starttime

FROM v$process p,

v$rollname r,

v$session s,

v$transaction t,

v$parameter x

WHERE s.taddr = t.addr

AND s.paddr = p.addr

AND r.usn = t.xidusn(+)

AND x.name = 'db\_block\_size'

ORDER by undo\_mb desc

/

2、大事务对数据库和应用影响还不大得情况下，可以采取的方法：

a.查找v$session\_longops，评估是让事务进行还是Kill发起大事务的session各自的代价，选择其中一个代价较低的方式。

b.如果是选择kill掉session，可以开启并发回滚事务的特性，加快事务回滚。

3、大量并发语句，大量’latch:undo global data’等待，应用已经无法响应，CPU使用90%以上的情况：

a.此时不管是采用何种回滚特性（并发回滚、单进程回滚）,由于已经没有cpu资源，回滚都非常耗时。

b.联系应用确认是否可以空表暂时代替，如果可以，可以再kill掉session后，将表rename掉，重新建一种空表，让应用临时使用。

c.后续使用分批提交的方式，将源表数据回插空表。

d.如不能空表代替，则只能暂停应用，kill掉等待session，cpu恢复正常后并发回滚，或建空表回插数据。

4、事件处理完毕后，对发起大事务的程序发给应用侧修改，如果是个人发起，则加强培训。

### “enq:US-content” or 回滚表空间使用过度事件应急处理

**现象描述：**

**影响因素：**

一般包含以下几种：

1. 回滚表空间使用过度，session发起新事务查找回滚段时需要排队等待
2. Oracle Bug

**解决方法：**

1、查找undo表空间使用情况的SQL：

select b.tablespace\_name,

nvl(used\_undo,0) "USED\_UNDO(M)",

total\_undo "Total\_undo(M)",

trunc(nvl(used\_undo,0) / total\_undo \* 100, 2) || '%' used\_PCT

from (select nvl(sum(bytes / 1024 / 1024), 0) used\_undo, tablespace\_name

from dba\_undo\_extents

where status in ( 'ACTIVE','UNEXPIRED')

group by tablespace\_name) a,

(select tablespace\_name, sum(bytes / 1024 / 1024) total\_undo

from dba\_data\_files

where tablespace\_name in

(select value

from v$spparameter

where name = 'undo\_tablespace'

and (sid = (select instance\_name from v$instance) or

sid = '\*'))

group by tablespace\_name) b

where a.tablespace\_name (+)= b.tablespace\_name

/

2、回滚表空间中大部分都是ACTIVE extent的情况

a.查询是哪些session引起，用“二十”中提供的SQL

b.临时新增undo数据文件解决

c.事后根据查找到的session对提交应用侧修改程序。个人引起则加强培训

3、回滚表空间中大部分都是UNEXPIRED extent的情况

a.减小undo\_retention值

b.如果undo\_retention的值正常，则新增undo数据文件，加大undo表空间

1. 回滚表空间使用情况正常

a.查看undo相关的隐含参数是否设置正确。

### “gc buffer busy”问题

**现象描述：**

环境：SunOS 5.9(64bit, sparc)+RAC+oracle 10.2.0.3

场景：话务性能库不断报出 gc buffer busy ，系统资源消耗较高。

**影响因素:**

以下几点可导致此事件：

A、Hot Blocks；

B、低效率的查询；

**解决方法:**

1、查询事件相关用户，sql

select sid,username,machine,program,p1raw,sql\_id,logon\_time,last\_call\_et from v$session where event='gc buffer busy';

2、根据sql\_id,username查询出sql，分析执行效率，与厂商协商解决。

select sql\_text from v$session from v$sqlarea where sql\_id = 'xx';

# 资源消耗类

### 高资源消耗进程应急处理

**现象描述：**

某个oracle process CPU使用率非常高。

某个oracle process MEM使用率非常高。

**影响因素：**

暂无

**解决方法：**

1、使用TOPAS观察哪个进程CPU使用率高，找出相关进程号，通过以下命令定位数据库SID号. Select sid,sql\_id,event,status from v$session where paddr in (select addr from v$process where spid=&进程号);

2、使用以下命令查看oracle会话使用内存超过100M的用户

set line 180

col MACHINE for a10

col PROGRAM for a25

col USERNAME for a15

select s.sid, s.serial#, s.username, s.machine, s.program,s.process, s.sql\_id, p.pga\_alloc\_mem/1048576 size\_m, p.spid

from v$session s, v$process p where s.paddr=p.addr

and p.pga\_alloc\_mem>104857600 order by 7 desc;

3、对相关进程和会话进行分析，决定是否kill.

### IO非常高

**现象描述：**

数据库登陆缓慢，或者根本不可用，无法登陆数据库进行查询等操作。表空间满或者文件系统满了一般情况下不会发生，因为我们有监控告警。这里主要是针对异常sql引起数据库hang的情况

**影响因素:**

分析IO高的原因，例如：

A、大量的并行；

B、长事务；

C、物理读高

**解决方法:**

在操作系统使用命令: ps -ef|grep LOCAL=NO |awk '{print $2}' |xargs kill -9

kll所有非本地进程,然后检查系统资源状态，检查数据库状态

### PGA使用过大

**现象描述：**

影响SQL执行的效率

**影响因素:**

PGA使用过大

**解决方法：**

1、查询当前PGA使用大小：

select sum(pga\_alloc\_mem)/1048576/1024 size\_gb

from v$process ;

2.查询使用PGA较大的具体进程

例如以下语句可以查出具体占用内存大于100m的进程信息:(例如）

set line 180

col MACHINE for a10

col PROGRAM for a25

col USERNAME for a15

select s.sid, s.serial#, s.username, s.machine, s.program,s.process, s.sql\_id, p.pga\_alloc\_mem/1048576 size\_m, p.spid

from v$session s, v$process p where s.paddr=p.addr

and p.pga\_alloc\_mem>104857600 order by 7 desc;

3.咨询开发商可以删除语句

alter system kill session 'sid,serial#';

### CPU使用过高应急处理

**现象描述：**

usr%使用率达到90以上

**影响因素：**

CPU使用过高，一般表现在以下几点：

A、不良SQL造成的大量等待事件

B、大量的短连接造成CPU负载高，BOSS外围系统曾出现过。

C、BUG

**解决方法：**

1、当CPU出现高负载的时候，首先我们要检查当前的数据库里是否有大量异常等待，例如：latch free, library cache lock/pin 等待事件。

select event,count(\*),wait \_class from v$session group by event,wait\_class order by 2;

如果有，根据相关等待事件分析问题。 也可以通过HANG分析，进行阻塞源头会话定位。

2、当CPU出现高负载的时候，检查发现当前数据库并无任何异常等待事件，我们就要参考平时的CPU使用率指标，然后通过会话、事务量来衡量。

3、当CPU出现高负载的时候，检查发现当前数据库并无任何异常等待事件，当前活动SQL语句与平时差别很大，我们可以关闭监听，检查是否由于连接造成的。

### 大表索引被删除导致CPU 100%的应急处理

**现象描述：**

CPU使用率90%以上

**影响因素：**

一般包含以下几种：

1、大表上索引被删除，部分SQL全表扫描导致资源集聚消耗

**解决方法：**

1、查找某个实例上等待session的信息，定位低效SQL：

select to\_char(a.logon\_time,'yyyy-mm-dd hh24:mi') logon\_time,

a.sql\_id,

a.event,

a.username,

a.osuser,

a.process,

a.machine,

a.program,a.module,

b.sql\_text,

b.LAST\_LOAD\_TIME,

to\_char(b.last\_active\_time,'yyyy-mm-dd hh24:mi:ss') last\_active\_time,

c.owner,c.object\_name,

a.last\_call\_et,

a.sid,a.SQL\_CHILD\_NUMBER,

c.object\_type,p.PGA\_ALLOC\_MEM,a.p1,a.p2,a.p3,

'kill -9 '||p.spid killstr

from v$session a, v$sql b, dba\_objects c,v$process p

where a.wait\_class <> 'Idle' and a.status='ACTIVE' and p.addr=a.paddr

and a.sql\_id = b.sql\_id(+)

and a.sql\_child\_number = b.CHILD\_NUMBER(+)

and a.row\_wait\_obj# = c.object\_id(+)

and a.type='USER'

order by a.sql\_id,a.event;

2、非rac环境

a.部分大表索引删除，导致全表扫面，cpu 100%

b.此时数据库基本处于hang状态，应用已经无法响应

c.停止相关应用，kill掉这些等待session, 新建索引（parallel nologging）

d.如果无法停止应用，停止数据库的监听，kill掉等待session，新建索引（parallel nologging）

3、rac环境

a.部分大表索引删除，导致全表扫面，cpu 100%

b.此时该节点基本处于hang状态，应用已经无法响应

c.停止相关应用，kill掉这些等待session, 新建索引（parallel nologging）

d.如果无法停止应用，停止所有节点的监听，kill掉等待session，新建索引（parallel nologging）

# 网络连接类

### 连接不上数据库

**现象描述：**

数据库无法连接

**影响因素:**

客户报连接不上数据库，例如：

1、连接数达到上限；

2、监听有问题；

**解决方法:**

1、看看连接数

show parameter session;

select count(\*) from v$session;

若我们用sqlplus / as sysdba也无法连接，则先KILL掉几个LOCAL=NO的会话，然后再进去分析是哪个模块的连接较多，然后通知开发商处理

ps -ef | grep LOCAL=NO －－找出PID，KILL掉几个

select machine,program,count(\*) from v$session group by machine,program order by 3; --找出较多的模块

2、查看监听

lsnrctl status

看看状态是不是ready，或者reload一次，看看故障是否能恢复

### 客户端不定时断开连接

**现象描述：**

环境：Aix+RAC+ORACLE 11.2.0.2.0

现象：PL/SQL等开发工具连接数据库后，会不定时的断开连接，断开后界面无反应，经过长时间才能重连，影响开发工作。根据现场观察到的故障现象：

1、检查数据库的各项相关设置，包括系统超时限制、用户资源限制和网络超时限制等方面，均未发现异常。

2、在不同网段进行模拟，发现所有网段均有断开连接现象发生。

3、经过长时间和全方面的模拟测试，发现断开规律为idle超过10分钟的数据库连接会被自动断开。

**分析思路：**

**注释： 这里最好配上测试的过程。**

**影响因素:**

一般情况下，为网络、防火墙策略限制.

**解决方法:**

重新修改网管防火墙相关限制项。

### 通过监听无法连接数据库

**现象描述：**

环境：Redhat+RAC+ORACLE 11.2.0.1.0

现象：数据库工作状态正常，监听器工作状态也正常。并且，监听服务也已经注册了。但即使在服务器本地，通过监听无法访问数据库。通过tns访问数据库时，总是报lost connect的错误信息。直接登陆时没有问题。

**现象分析:**

这个问题可能是oracle的一个bug，在排除了常用的解决办法办法之后，可以考虑修改$ORACLE\_HOME/bin/oracle文件的权限。

但这个问题在许多数据库并无法重现，在遇到这样的问题时，可以偿试这样解决一下。

**解决方法:**

将$ORACLE\_HOME/bin/oracle的权限更改为6571

# 空间管理类

### 表空间故障应急处理

**现象描述：**

场景一：在RAC环境下进行表空间扩容（添加数据文件）时，只在一个节点上对数据文件建立了软连接，另一个节点没有建立软连接。

场景二： 在RAC环境下进行表空间扩容（添加数据文件）时，两个节点都没有建立软连接，只在一个节点的本地文件系统添加了数据文件，或者添加数据文件时有空格等特殊字符

场景三： 不小心将其他环境的裸设备加到到当前的环境中。（**绝不允许出现此类错误**）

**影响因素:**

一般情况下，都属于人为错误.

**解决方法:**

**（场景一）解决方法**：

1、 将两个节点数据文件改为离线状态

alter database datafile 'XXX' offline;

2、 在问题节点对数据文件建立软连接

ln –s 裸设备 数据文件

3、 在问题节点恢复数据文件

recover datafile 'XXX';

4、 将数据文件改为在线状态

alter database datafile 'XXX' online;

5、 确认数据库告警日志无报错。

**（场景二）解决方法：**

1、 将问题节点数据文件改为离线状态

alter database datafile 'XXX' offline;

2、 在各节点对数据文件建立软连接

ln –s 裸设备 数据文件

3、 通过ALTER DATABASE CREATE DATAFILE ‘源文件’ AS ‘目标文件’; copy数据文件至目标位置

ALTER DATABASE CREATE DATAFILE '源文件' AS '目标文件';

4、 恢复数据文件

recover datafile '目标文件';

5、 将数据文件改为在线状态

alter database datafile '目标文件' online;

6、 将错误的本地数据文件移到其他路径，避免“/oracle”文件系统使用比率达到告警值。

7、 确认数据库告警日志无报错。

**（场景三）解决方法：**

1、 除了恢复，没有太好的方法。需要备份、归档都在。

### SYSAUX表空间爆满没有存储扩容应急处理

**现象描述：**

如果SYSAUX 表空间不可用时， 数据库的核心功能还是可以继续运行的。只是一些存放在SYSAUX表空间里的功能收到限制，比如OEM。

**影响因素：**

一般包含以下几种：

A、业务表被放在SYSAUX表空间

B、一些辅助表的数据量太大

**解决方法：**

检查SYSAUX中的对象，是哪个对象占用了大量空间.检查时业务表还是辅助表，确认该对象是否可以清理.

col segment\_name format a30

col segment\_type format a30

set linesize 300

select OWNER,SEGMENT\_NAME,SEGMENT\_TYPE,BYTES/1024/1024/1024 GB

from dba\_segments

where TABLESPACE\_NAME='SYSAUX'

order by 4;

### 文件系统使用率达到或超过95%应急处理

**现象描述：**

使用df 命令查看文件系统使用率达到95%

**影响因素：**

备份文件、dump文件、trace文件等大量产生造成.

**解决方法：**

检查相关文件系统下面的文件大小：

例如：查看/u01目录下每个子目录的大小

du -sh /u01/\*

根据需求进行清理.

### 临时表空间过度事件应急处理

**现象描述：**

临时表空间使用过度，需要使用临时表空间的SQL运行缓慢

**影响因素：**

一般包含以下几种：

1、部分SQL大量使用临时段

**解决方法：**

1、查找单节点temp表空间使用情况的SQL：

SELECT A.tablespace\_name tablespace,

D.mb\_total,

SUM(A.used\_blocks \* D.block\_size) / 1024 / 1024 mb\_used,

D.mb\_total - SUM(A.used\_blocks \* D.block\_size) / 1024 / 1024 mb\_free

FROM v$sort\_segment A,

(SELECT B.name, C.block\_size, SUM(C.bytes) / 1024 / 1024 mb\_total

FROM v$tablespace B, v$tempfile C

WHERE B.ts# = C.ts#

GROUP BY B.name, C.block\_size) D

WHERE A.tablespace\_name = D.name

GROUP by A.tablespace\_name, D.mb\_total;

2、查询session使用temp段的情况：

SELECT S.sid || ',' || S.serial# sid\_serial,

S.username,

S.osuser,

P.spid,

S.module,

S.program,

s.sql\_id

SUM(T.blocks) \* TBS.block\_size / 1024 mb\_used,

T.tablespace,

COUNT(\*) sort\_ops

FROM v$sort\_usage T, v$session S, dba\_tablespaces TBS, v$process P

WHERE T.session\_addr = S.saddr

AND S.paddr = P.addr

AND T.tablespace = TBS.tablespace\_name

GROUP BY S.sid,

S.serial#,

S.username,

S.osuser,

P.spid,

S.module,

S.program,

TBS.block\_size,

T.tablespace

ORDER BY sid\_serial;

1. 如果是某些session使用temp段过度引起
2. 联系应用停止相关程序 or kill掉这些session
3. 如果session使用temp情况都属正常情况
4. 加大临时表空间

### 后台undo表空间无法自动扩展

**现象描述：**

环境：Aix+RAC+oracle 10.2.0.3.0

场景：数据库节点一，后台alter日志报：ORA-1628: max # extents reached for rollback segment错误信息，使数据库的调度任务无法完成。

**影响因素:**

回滚段的maxextens数达到参数设置的最大值，造成不能分配下一个extent

**分析思路：**

**注释： 这里需要将如何看maxextens 的查询方法记录上。**

**解决方法**

1.新建立一个表空间undo2，修改节点1的undo表空间为undo2后问题排除。

2.修改增大undo的maxextents参数值

### undo表空间无法扩展

**现象描述：**

环境：Linux+ oracle 11.2.0.1.0

场景：在向数据库插入一条记录时，数据库频繁出现ORA-30036: unable to extend segment by 8 in undo tablespace 'UNDOTBS1'的报错信息，无法插入记录。

**影响因素:**

产生该错误，通常是由于当前的UNDO表空间很小，并且在同时有一个很大的事务存在。并且，在Oracle看来，这种提示并不是一种异常。因此，您无法在该ORA报错的同时在alert文件中看到相关的信息，你只能通过相应的trace文件来看到这类的信息。

当数据库产生上述信息时，在alert中会留下这样的信息：

kdidelleafblks -- encountered error 30036. See trace file for details

相应的trace文件(vmdb\_j000\_3784.trc)中，会有这样的描述：

ORA-30036 DIAGNOSTIC

This diagnostic information is dumped to trace file at

most once every 24 hours, it does not indicate any error.

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

ORA-30036 happens when trying to extend undo segment \_SYSSMU12\_710107609$ (usn=12) by 8 blocks

Reason: Cannot Wrap, Fail to Steal

Current undo tablespace UNDOTBS1 (tsn=2)

undo tablespace current size 262144 blks, maxsize 262144 blks, fixed sized

Undo Retention (reactive):568, Max Query Length:1383

Parameter Undo Retention:900, Tuned Undo Retention:568, High threshold Undo Retention:-2 autotune:1

Retention Guarantee FALSE

Current Time is 1351615156

同时，在该trace文件的Info about Undo Segments in this undo tablespace信息列表中，将会很容易找到当前存在的大事务。如：

Undo Segment 1: HW Ext=2, 1st Active Ext=-1, Total Extents=3

Extent Info of Undo Segment 1

ExtID Status Size CmtTime

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

0 Active 7 1351614703

1 Active 8 1351614760

2 Unexpired 128 1351614760

2 Unexp wmql 128 1351614760

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

\_SYSSMU1\_2861643892$ (usn 1) is INUSE (act=16, une=128, exp=0)

(unewmql=128, expwmql=0)

**解决方法：**

1、通过分析上述trace文件，可以找到当前存在的大事务，对该事务进行适当分解，减少对UNDO表空间的资源需求量。

2、增大UNDO表空间的大小。

**紧急措施：**

1、增大UNDO表空间

### 后台库temporary段暴涨

**现象描述：**

环境：AIX Version 6.1+RAC+oracle 10.2.0.3

场景：后台库表空间tbs\_tmp\_data出现暴涨，一小时大约增长30-40G.经查询，发现tbs\_tmp\_data表空间出现大量的SEGMENT\_NAME类似于35.75187这样的temporary段。

**影响因素:**

以下操作会占用大量的temporary：

1、 用户执行imp/exp 导入导出操作时，会使用大量的temporary段

2、 用户在rebuild index时

3、 执行create table ...... as 语句时

4、 移动用户下的数据到别的表空间时

**解决方法:**

找出产生temporary段的用户：

SQL> select OWNER,SEGMENT\_NAME,SEGMENT\_TYPE,BYTES/1024/1024/1024 from dba\_segments where TABLESPACE\_NAME=upper('tbs\_tmp\_data') and SEGMENT\_TYPE=upper('temporary');

找出操作：

SQL> select sid from v$session where username='ETL' and STATUS='ACTIVE' and program in ('IMP','EXP','IMPDP','EXPDP');

SQL> select p.spid,to\_char(SQL\_FULLTEXT) from v$sqlarea a,v$session s,v$process p where p.addr=s.paddr and s.sql\_id=a.sql\_id and s.username='ETL' and s.STATUS='ACTIVE' and (SQL\_FULLTEXT like '%alter%' or SQL\_FULLTEXT like '%create table%as select%');

我们也可以上面的方法找出导致temporary段暴涨的原因，跟开发商沟通，可以选择kill掉sid对应的操作系统进程。尽量减少在业务高峰期进行这些操作。

### 数据泵进程挂起导致磁盘爆满

**现象描述：**

环境：SunOS 5.10+RAC+ORACLE 10.2.0.3

场景： 管线系统数据库，由于系统磁盘空间涨到100%,数据库无法继续使用。

**影响因素:**

检查发现磁盘空间一直在增长，但是数据库没有用户做操作。最后检查有一个数据库expdp进程在running，导出的文件使磁盘空间一直增长

**解决方法:**

select job\_name,state from dba\_datapump\_jobs;找出running的job\_name,

expdp 用attach参数连入数据库，输入命令：kill\_job，即杀掉expdp进程，磁盘空间停止增长，清理不必要的数据，重启库恢复正常。

# 无法启动类

### RAC数据库CRS无法正常启动

**现象描述：**

环境：redhat linux +RAC+oracle 11.1.0.7

场景：在作了十八大安全相关整改后，数据库CRS无法正常启来。

**影响因素:**

影响crs正常启动的因素较多，不能正常启动，数据库集群无法使用。但检查crs各项配置均正常。

**解决方法:**

检查crs日志发现报系统权限相关的错误，再次确认在安全整改过程中，/etc/security目录的权限被更改为700。正常应该为755权限,执行chmod 755 /etc/security再进行执行 crsctl start crs 可正常启动。

### ORACLE GC OMS无法启动

**现象描述：**

环境：OS Version: Windows 2003 x86，GC Version: 11.0.1.0，WebLogic Version: 10.3.2.0

Oracle DB Version: 11.2.0.1

场景：OMS服务在主机启动以后无法起来 OMS日志中有一些报错信息：

weblogic.diagnostics.lifecycle.DiagnosticComponentLifecycleException: weblogic.store.PersistentStoreException: [Store:280020]There was an error while reading from the log file

Caused by: java.io.IOException: Error reading from file, 已到文件结尾， errno=38

**影响因素:**

oms主机所在的39.189机器此次故障前一周,出现过因GC安装路径所在磁盘空间耗尽导致OMS宕机的,当时清理空间后,重新启动OMS,正常启动.但主机重启后,OMS无法启动.根据搜索的一些信息:当OMS日志所在目录爆满一次后，下次重新启动时日志将无法写入。

**解决方法:**

1.E:\app\Middleware\oms11g\BIN>emctl stop oms -all #停止后才能删除DAT文件

2.E:\app\gc\_inst\user\_projects\domains\GCDomain\servers\EMGC\_OMS1\data\store\diagnostics目录下的DAT文件备份后删除

3. .E:\app\Middleware\oms11g\BIN>emctl start oms

之后，OMS可正常启动，问题解决。

# 数据库HANG类

### 数据库HANG异常处理

**现象描述:**

数据库无法登陆

**影响因素:**

A、大量异常阻塞

B、资源耗尽

**解决方法:**

1.进行HANG分析，查找顶级阻塞的会话。

普通的HANG分析：

参考业支脚本.

全局的HANG分析：

oradebug setmypid (oradebug setospid 3188)

oradebug unlimit

oradebug setinst all

oradebug -g def hanganalyze 3

oradebug tracefile\_name

2.紧急情况下，在操作系统使用命令: ps -ef|grep LOCAL=NO |awk '{print $2}' |xargs kill -9

kill所有非本地进程,然后检查系统资源状态，检查数据库状态

### 话务告警库oracle进程HANG住

**现象描述：**

环境：SunOS+RAC+oracle 10.2.0.3.0

场景：话务告警库打补丁后，开启各个资源后，用crs\_stat -t检查个资源均为online状态，实际上2节点数据库实例未启动。

**影响因素:**

当天凌晨数据库打补丁，再startup数据库实例。在重启过程过程中看后台日志，发现使用未有 ’instance open‘出现，实际上ORACLE进程HANG住，实例未正常启动。

**解决方法:**

关闭emagent，mnagent程序及其他所有oracle相关进程，重启数据库恢复正常。再打开emagent。

# 数据库CRASH类

### 网分后台库实例重启

**现象描述：**

环境：Aix+RAC+oracle 10.2.0.3.0

场景：数据库三个实例自动重启，startup命令可以启动数据库，但运行一个多小时候会自动重启数据库。

**影响因素:**

最后检查到是undo表空间中有一个事务回滚段出错。

**分析思路：**

**注释： 请描述出错误内容，以及找出具体查询的方法。**

**解决方法:**

找出次回滚的具体数据库操作，与开发商协商解决。

### 实例CRASH应急处理

**现象描述：**

RAC环境单节点crash ，RAC环境所有节点crash

**影响因素：**

暂无,CRASH问题涉及到的问题比较复杂，以后将逐步更新。

**解决方法：**

遇到实例crash之后，首先看是否能重启启动：

1、能重新启动的情况下，知会相关领导, 事后分析重启原因。

2、不能重启的情况下：

A、RAC之单节点无法重启，通知相关领导， 评估正常节点负载，应用切换至正常节点。相关技术人员进行故障处理。

B、RAC之全节点无法重启，通知相关领导， 考虑是否切换应急库。相关技术人员进行故障处理。

### 传输数据库脑裂问题

**现象描述：**

环境：Aix+RAC+oracle 10.2.0.4

场景:传输网管数据库实例脑裂，主机和CRS无异常，但是节点二实例自动重启，并且在脑裂的过程中影响节点一数据库hang，应用不能正常连接。

数据库日志可看到以下信息：

IPC Send timeout detected,Reciver ospid 2123460;Warting for clusterware split-brain resolution; ORA-29740: evicted by member , group incarnation;

**影响因素:**

脑裂会导致数据库自动实例重启。通常出现ORA-29740是由以下原因引起：

1. RAC节点间时间不同步
2. 数据库主机资源负载过高
3. 心跳网络异常
4. Oracle bug

**解决方法：**

经过检查先后发现了以下问题：

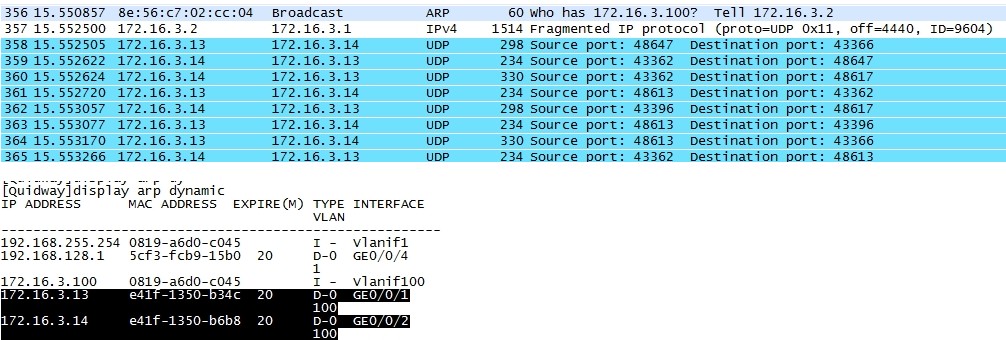
1、主机/var目录空间不足，系统dump生成失败

2、Oracle用户环境变量设置不正确

3、操作系统未打APAR IV04047补丁

4、RAC两节点的操作系统版本不一致，cs\_01版本6100-07，cs\_02版本为6100-06

5、分析发现心跳交换机配置存在异常，交换机IP为172.16.3.1和cs\_01的心跳网卡IP冲突。经协商后更改交换机IP为172.16.3.100。并且发现在主机上无法ping通交换机IP，在交换机上cs\_01和cs\_02连接的端口上抓包，发现抓到172.16.3.13和172.16.3.14的数据包，确认这两个IP为物联网数据库的心跳IP。我们检查物联网数据库后发现，该系统也出现了ORA-29740引起的脑裂。抓包信息如下



经过和惠讯、IBM、亿阳协商，更换一台交换机进行测试，进一步定位是否由于心跳交换机配置问题导致该故障。在几次更换心跳交换机后问题解决。

至此，前后耗时约一个月的传输脑裂问题解决，此类长时间无法定位的问题，应该扩大分析范围不仅着眼于数据库，应该和主机网络人员一道分析定位。

# 应用操作类

### SQL执行计划变化应急处理

**现象描述：**

一个SQL的执行计划的不稳定，

**影响因素:**

常见原因包含以下两种：

1、统计信息的变化

2、SQL语句的变化

**解决方法**：

1、通过SQL\_ID确认统计信息是否一致 （该语句会将AWR中所有的信息查找出来）

set lines 155

col execs for 999,999,999

col avg\_etime for 999,999.999

col avg\_lio for 999,999,999.9

col begin\_interval\_time for a30

col node for 99999

break on plan\_hash\_value on startup\_time skip 1

select ss.snap\_id,

ss.instance\_number node,

begin\_interval\_time,

sql\_id,

plan\_hash\_value,

nvl(executions\_delta, 0) execs,

(elapsed\_time\_delta /

decode(nvl(executions\_delta, 0), 0, 1, executions\_delta)) / 1000000 avg\_etime,

(buffer\_gets\_delta /

decode(nvl(buffer\_gets\_delta, 0), 0, 1, executions\_delta)) avg\_lio,

(disk\_reads\_delta /

decode(nvl(buffer\_gets\_delta, 0), 0, 1, executions\_delta)) avg\_pio

from DBA\_HIST\_SQLSTAT S, DBA\_HIST\_SNAPSHOT SS

where sql\_id = nvl('&sql\_id', '4dqs2k5tynk61')

and ss.snap\_id = S.snap\_id

and ss.instance\_number = S.instance\_number

and s.instance\_number like nvl('&instance\_number', s.instance\_number)

and executions\_delta > 0

order by 1, 2, 3

/

2、如果不一致，我们需要检查统计信息是否最新，如果统计信息更新时间比当前时间有些遥远，收集统计信息。

3、如果统计信息正常，我们需要确认数据量是否有大量增加。

4、如果数据量增加的情况下，我们就要考虑清理数据或SQL优化。

### 使用绑定变量SQL突然运行缓慢的应急处理

**现象描述：**

一个session可能一某种方式变的非常消耗CPU,并且没有任何有意义的wait event. SQL的执行计划没有改变过

**影响因素：**

一般包含以下几种：

1. SQL代入的绑定变量值极端
2. 等待的块有问题

**解决方法：**

1. 查看SQL的执行计划，及等待事件。
2. 等待事件是否正常，如果为块等待，则可以采样等待的块，通过采样的块，定位对应的表和索引，进行分析。
3. 使用errorstack定位这个SQL的绑定变量值进行分析。

### 数据网管无法插入数据

**现象描述：**

环境：SunOS+oracle 11.2.0.1

数据网管数据库中插入数据失败。

**影响因素:**

主要问题来自于I/O的问题，由于业务增加，I/O争用严重，导致数据库dbwr进程写入慢，从而影响到缓冲区不能快速释放，在用户请求时无法分配到空闲的缓冲块。产生较高的"free buffer waits"的等待事件。

**解决方法:**

找出正在进行的io读写频繁的数据库操作，与开发商协商解决。

### 前端库应用程序timestamp错误

**现象描述：**

环境：Aix+RAC+oracle 10.2.0.5.0

场景：开发商反映前端数据库，应用程序的timestamp一直出错

**影响因素:**

最后检查到是主机时区与数据库的时区设置错误导致这个错误。

**分析思路：**

**注释： 记录错误内容。**

**解决方法:**

调整主机时区问题解决。

# 其他分类

### 服务切换应急处理

**现象描述：**

场景一：当我们在进行服务切换时，无法从一个节点切换到另外一个节点。

**影响因素:**

一般包含以下几种：

A、目标节点实例异常,服务无法切换。

B、持有服务节点无法释放，服务无法切换

C、BUG

分析思路：

**解决方法**：

**(场景一) 解决方法**：

1.首先使用 srvctl start service -d <db\_name> [-s <service\_name> [-i <inst\_name>]] 命令切换。

例如： srvctl start service -d fsdb -s smm –i fsdb1

2.当上面命令, 可以使用以下命令进行强制服务切换: alter system set service\_names=‘s1,s2’;

例如：alter system set service\_names='sfs,smm' scope=memory sid='FSDB1';

然后查看是否注册成功：

SQL> show parameter service

NAME TYPE VALUE

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

service\_names string sfs,smm

3.检查监听并测试连接测试是否生效

$lsnrctl service

$sqlplus sys/oracle@node\_vip:1521/service\_name

4、如果连接不成功，按照以下命令排查.

检查实例启动时间: select instance\_number, status, startup\_time from gv$instance;

检查数据库状态: select inst\_id, open\_mode from gv$database;

检查alert.log日志:Completed: ALTER DATABASE OPEN (是否完成)

检查服务运行情况: srvctl status service -d <db\_name>

检查服务配置情况: srvctl config service -d <db\_name> -a

### 升级PSU系统报错

**现象描述：**

在打补丁的过程中，出现如下的报错信息：

Running prerequisite checks...

Prerequisite check "CheckActiveFilesAndExecutables" failed.

The details are:

Following executables are active :

/oracle/app/oracle/product/10.2.0/bin/tnslsnr

UtilSession failed: Prerequisite check "CheckActiveFilesAndExecutables" failed.

OPatch failed with error code 73

**影响因素:**

通过日志可以发现，导致当前升级PSU出错的原因是当前的监听器没有关闭。因此，当务之急，是将监听手工关闭。

但事实上，在出现上述错误之前，工程已经将监听关闭了。但由于该PSU只跟数据库相关，与集群没有关系。因此，他在升级该PSU时，只是通过CRS关闭了监听，过了一段时间以后，监群又自动将监听拉起来了，这就导致升级PSU时，监听仍然是ACTIVE状态。

因而，在升级补丁或PSU时，需要通过lsnrctl stop的方式关闭监听器。

**解决方法**：

通过lsnrctl stop的方式关闭监听器。

### 数据库在执行过程中使用动态采样

**现象描述：**

一个SQL在执行过程中异常缓慢，通过查看其执行计划发现，该SQL在执行过程中使用动态采样。

Note

-----

- dynamic sampling used for this statement

**现象分析:**

动态采样本身是Oracle的一个特性，用于在没有统计信息的情况下，自动收集相关表的统计信息。但在某些情况下，他仍然使用动态采样，并进而导致执行效率低下。

**解决方法**：

1、通过hint禁用动态采样。/\*+dynamic\_sampling(t 0) \*/

2、这是Oracle的一个bug，请打补丁9272549。

### 通过srvctl查看数据库的状态时反应缓慢

**现象描述：**

数据库状态正常，但在执行"srvctl status database -d uwnms1"时，其节点一需要很长时间才能出结果，节点二正常。查看alert日志，也并未发现有告警的信息。

**现象分析:**

通过类似的命令查看或启动某个资源时，由于并无相应的日志，因此给我们的判断带来一些麻烦。解决此类问题，我们可以采用跟踪的办法来分析其内部的相应过程。

**解决方法**：

1、strace -frT -o /tmp/strace2.log srvctl status database -d uwnms1，根据其生成的trace文件分析内部过程，并解决。

2、使用sh -x shell.sh的方式去debug命令的执行过程，发现问题。

3、使用truss等命令进行时间分析。