Oracle 数据库日常维护

【版本整理日期：2011/02/26 】

版本整理人：1634068400@QQ.COM

本文档包含以下内容：

/**********

HTTP://ERP100.TAOBAO.COM (若跳转不成功，请复制到浏览器或联系 Q)

http://item.taobao.com/item.htm?id=7437120468 Metalink Sharing

**********/

在 Oracle 数据库运行期间，DBA 应该对数据库的运行日志及表空间的使用情况进行监控，及早发现数据库中存在的问题。

一、Oracle 警告日志文件监控

Oracle 在运行过程中，会在警告日志文件(alert_SID.log)中记录数据库的一些运行情况：

- 数据库的启动、关闭，启动时的非缺省参数；
- 数据库的重做日志切换情况，记录每次切换的时间，及如果因为检查点(checkpoint)操作没有执行完成造成不能切换，会记录不能切换的原因；
- 对数据库进行的某些操作，如创建或删除表空间、增加数据文件；
- 数据库发生的错误，如表空间不够、出现坏块、数据库内部错误(ORA－600)

DBA 应该定期检查日志文件，根据日志中发现的问题及时进行处理

问题	处理
启动参数不对	检查初始化参数文件
因为检查点操作或归档操作没有完成造成重做日志不能切换	如果经常发生这样的情况，可以考虑增加重做日志文件组；想办法提高检查点或归档操作的效率；
有人未经授权删除了表空间	检查数据库的安全问题，是否密码太简单；如有必要，撤消某些用户的系统权限
出现坏块	检查是否是硬件问题(如磁盘本生有坏块)，如果不是，检查是那个数据库对象出现了坏块，对这个对象进行重建
表空间不够	增加数据文件到相应的表空间
出现 ORA-600	根据日志文件的内容查看相应的 TRC 文件，如果是 Oracle 的 bug，要及时打上相应的补丁

	CHUNKS	MAXCHUNK	TABLESPACE_NAME
1	1	118720	SYSTEM
2	1	325632	GCDATA_DEFAULT
3	1	65536	SIMULATOR
4	1	4608	USERS
5	2	2096128	ULTRASIM
6	3	1013760	ULTRACRUISER
7	4	1740800	ULTRANMS_JYFX
8	4	1306624	ULTRANMS_BIG
9	50	88000	SYSAUX
10	116	626624	ULTRANMS_SMALL
11	127	1571840	UNDOTBS1
12	709	1658880	ULTRANMS_DEFAULT

二、数据库表空间使用情况监控（字典管理表空间）

数据库运行了一段时间后，由于不断的在表空间上创建和删除对象，会在表空间上产生大量的碎片，DBA 应该及时了解表空间的碎片和可用空间情况，以决定是否要对碎片进行整理或为表空间增加数据文件。

```
select tablespace_name,
```

```
        count(*) chunks ,
        max(bytes/1024/1024) max_chunk
  from dba_free_space
 group by tablespace_name;
```

上面的 SQL 列出了数据库中每个表空间的空闲块情况,如下所示：

```
TABLESPACE_NAME          CHUNKS    MAX_CHUNK
-------------------- ---------- ----------
INDX                          1  57.9921875
RBS                           3   490.992188
RMAN_TS                       1   16.515625
SYSTEM                        1   207.296875
TEMP                         20   70.8046875
TOOLS                         1   11.8359375
USERS                        67   71.3671875
```

其中,CHUNKS 列表示表空间中有多少可用的空闲块(每个空闲块是由一些连续的 Oracle 数据块组成)，如果这样的空闲块过多，比如平均到每个数据文件上超过了 100 个，那么该表空间的碎片状况就比较严重了，可以尝试用以下的 SQL 命令进行表空间相邻碎片的接合：

```
alter tablespace 表空间名 cascade;
                                此处是有误吧，coalesce;
```

然后再执行查看表空间碎片的 SQL 语句，看表空间的碎片有没有减少。如果没有效果，并且表空间的碎片已经严重影响到了数据库的运行，则考虑对该表空间进行重建。

MAX_CHUNK 列的结果是表空间上最大的可用块大小，如果该表空间上的对象所需分配的空间(NEXT 值)大于可用块的大小的话，就会提示

ORA-1652、ORA-1653、ORA-1654 的错误信息，DBA 应该及时对表空间的空间进行扩充，以避免这些错误发生。

对表空间的扩充对表空间的数据文件大小进行扩展，或向表空间增加数据文件，具体操作见“存储管理”部份。

三、查看数据库的连接情况

DBA 要定时对数据库的连接情况进行检查，看与数据库建立的会话数目是不是正常，如果建立了过多的连接，会消耗数据库的资源。同时，对一些“挂死”的连接，可能会需要 DBA 手工进行清理。

以下的 SQL 语句列出当前数据库建立的会话情况：

```
select sid,serial#,username,program,machine,status
  from v$session;
```

输出结果为：

```
SID SERIAL#   USERNAME   PROGRAM      MACHINE         STATUS
---- ------- ---------- ----------- --------------- --------
   1       1             ORACLE.EXE   WORK3           ACTIVE
   2       1             ORACLE.EXE   WORK3           ACTIVE
   3       1             ORACLE.EXE   WORK3           ACTIVE
   4       1             ORACLE.EXE   WORK3           ACTIVE
   5       3             ORACLE.EXE   WORK3           ACTIVE
   6       1             ORACLE.EXE   WORK3           ACTIVE
   7       1            ORACLE.EXE   WORK3            ACTIVE
   8      27 SYS        SQLPLUS.EXE  WORKGROUP\WORK3  ACTIVE
  11       5 DBSNMP     dbsnmp.exe   WORKGROUP\WORK3 INACTIVE
```

其中，

SID　　会话(session)的 ID 号；

SERIAL#　　会话的序列号，和 SID 一起用来唯一标识一个会话；

USERNAME　　建立该会话的用户名；

PROGRAM	这个会话是用什么工具连接到数据库的；
STATUS	当前这个会话的状态，ACTIVE 表示会话正在执行某些任务，INACTIVE 表示当前会话没有执行任何操作；

如果 DBA 要手工断开某个会话，则执行：

```
alter system kill session 'SID,SERIAL#';
```

注意，上例中 SID 为 1 到 7(USERNAME 列为空)的会话，是 Oracle 的后台进程，不要对这些会话进行任何操作。

四、控制文件的备份

在数据库结构发生变化时，如增加了表空间，增加了数据文件或重做日志文件这些操作，都会造成 Oracle 数据库控制文件的变化，DBA 应及进行控制文件的备份，备份方法是：

执行 SQL 语句：

```
alter database
  backup controlfile to '/home/backup/control.bak';
```

或：

```
alter database
  backup controlfile to trace;
```

这样，会在 USER_DUMP_DEST(初始化参数文件中指定)目录下生成创建控制文件的 SQL 命令。

五、检查数据库文件的状态

DBA 要及时查看数据库中数据文件的状态（如被误删除），根据实际情况决定如何进行处理，检查数据文件的状态的 SQL 如下：

```
select file_name,status
from dba_data_files;
```

如果数据文件的STATUS列不是AVAILABLE，那么就要采取相应的措施，如对该数据文件进行恢复操作，或重建该数据文件所在的表空间。

六、检查数据库定时作业的完成情况

如果数据库使用了 Oracle 的 JOB 来完成一些定时作业，要对这些 JOB 的运行情况进行检查：

```
select job,log_user,last_date,failures
 from dba_jobs;
```

如果 FAILURES 列是一个大于 0 的数的话，说明 JOB 运行失败，要进一步的检查。

七、数据库坏块的处理

当 Oracle 数据库出现坏块时，Oracle 会在警告日志文件（alert_SID.log）中记录坏块的信息：

ORA-01578: ORACLE data block corrupted (file # 7, block # <BLOCK>)

ORA-01110: data file <AFN>: '/oracle1/oradata/V920/oradata/V816/users01.dbf'

其中，＜AFN＞代表坏块所在数据文件的绝对文件号，<BLOCK>代表坏块是数据文件上的第几个数据块

出现这种情况时，应该首先检查是否是硬件及操作系统上的故障导致 Oracle 数据库出现坏块。在排除了数据库以外的原因后，再对发生坏块的数据库对象进行处理。

1．确定发生坏块的数据库对象

```
SELECT tablespace_name,
       segment_type,
       owner,
       segment_name
FROM  dba_extents
WHERE  file_id = <AFN>
 AND <BLOCK> between block_id AND block_id+blocks-1;
```

2．决定修复方法

如果发生坏块的对象是一个索引，那么可以直接把索引 DROP 掉

后，再根据表里的记录进行重建；

如果发生坏块的表的记录可以根据其它表的记录生成的话，那么可以直接把这个表 DROP 掉后重建；

如果有数据库的备份，则恢复数据库的方法来进行修复；

如果表里的记录没有其它办法恢复，那么坏块上的记录就丢失了，只能把表中其它数据块上的记录取出来，然后对这个表进行重建。

3．用 Oracle 提供的 DBMS_REPAIR 包标记出坏块

```
exec DBMS_REPAIR.SKIP_CORRUPT_BLOCKS('<schema>','<tablename>');
```

4．使用 Create table as select 命令将表中其它块上的记录保存到另一张表上

```
create table corrupt_table_bak
 as
  select * from corrupt_table;
```

5．用 DROP TABLE 命令删除有坏块的表

```
drop table corrupt_table;
```

6．用 alter table rename 命令恢复原来的表

```
alter table corrupt_table_bak
 rename to corrupt_table;
```

7．如果表上存在索引，则要重建表上的索引

八、操作系统相关维护

DBA 要注意对操作系统的监控：

- 文件系统的空间使用情况（df -k），必要时对 Oracle 的警告日志及 TRC 文件进行清理
- 如果 Oracle 提供网络服务，检查网络连接是否正常
- 检查操作系统的资源使用情况是否正常
- 检查数据库服务器有没有硬件故障，如磁盘、内存报错

.数据字典和动态性能视图

数据字典是 oracle 数据库的最重要的组成部分，它提供了数据库的相关系统信息；动态性能视图记载了例程启动以来的相关性能信息。

数据字典记载了数据库的系统信息，它是只读表和视图的集会。数据字典包含数据字典基表和数据字典视图两部分，其中，基表存储数据库的基本信息，普通用户不能之间访问数据字典基表；数据字典视图是基于数据字典基表建立的视图，普通用户可以通过查询数据字典视图取得系统信息。数据字典视图主要包括 USER_XXX,ALL_XXX,DBA_XXX 三种类型。

USER_XXX 用于显示当前用户所拥有的所有对象，它只返回用户所对应的所有对象。

DBA_XXX 用于显示整个数据库范围内的详细系统信息，它会显示所有方案所拥有的数据库对象。

常用数据字典

DICT 用于显示当前用户可访问的所有数据字典视图，并给出了这些数据字典视图的作用。

DICT_COLUMNS 用于显示数据字典视图的每个列的作用。

DUAL 用于取得函数的返回值。

GLOBAL_NAME 用于显示当前数据库的全名。

IND 用于显示当前用户所拥有的所有索引和索引的统计信息。

OBJ 用于显示当前用户所拥有的所有对象。

SEQ 用于显示当前用户所拥有的所有序列。

SYN 用于显示当前用户所拥有的同义词和同义词所对应的数据库对象名。

TAB 用于显示当前用户所用于的表，视图和序列。

动态性能视图用于记录当前例程的活动信息。启动例程时，oracle 会自动建立动态性能视图；停止例程时，oracle 会自动删除动态性能视图。需要注意的时，数据字典的信息时从数据文件中取得，而动态性能视图时从 SGA 和控制文件中取得。通过查询动态性能视图，一方面可以获得性能数据，另一方面可以取得与磁

盘和内存结构相关的其他信息。所有的动态性能视图都是以 V_$开始的，oracle 为每个动态性能视图提供了相应的同义词（以 V$开始）

常用的动态性能视图

V$FIXED_TABLE 用于列出所有可用的动态性能视图和动态性能表。

V$INSTANCE 用于获取当前例程的详细信息。

V$SGA 用于取得 SGA 更详细的信息。

V$PARAMETER 用于取得初始化参数的详细信息。

V$VERSION 用于取得 oracle 版本的详细信息。

V$OPTION 用于显示已经安装的 oracle 选项。其中，TRUE 表示该选项已经安装，FALSE 表示该选项没有安装。

V$SESSION 用于显示会话的详细信息。

V$PROCESS 用于显示与 oracle 相关的所有进程的信息（包括后台进程和服务器进程）。

V$BGPROCESS 用于显示后台进程的详细信息。

V$DATABASE 用于取得当前数据库的详细信息（如数据库名，日志模式以及建立时间）。

V$CONTROLFILE 用于取得当前数据库所有控制文件的信息。

V$DATAFILE 用于取得当前数据库所有数据文件的详细信息。

V$DBFILE 用于取得数据文件编号及名称。

V$LOGFILE 用于显示重做日志成员的信息。

V$LOG 用于显示日志组的详细信息。

V$THREAD 用于取得重做线程的详细信息。

V$LOCK 用于显示锁信息。

V$LOCKED_OBJECT 用于显示被加锁的数据库对象。

V$ROLLNAME 和 V$ROLLSTAT

V$ROLLNAME 动态性能视图用于显示处于 online 状态的 undo 段，而 V$ROLLSTAT 用于显示 undo 段统计信息。通过在二者之间执行连接查询，可以显示 undo 段的详细统计信息。

V$TABLESPACE 用于显示表空间的信息。

V$TEMPFILE 用于显示当前数据库所包含的临时文件。

2.

常用 DBA 管理脚本

一、数据库构架体系

1、表空间的监控是一个重要的任务，我们必须时刻关心表空间的设置，是否满足现在应用的需求，以下的语句可以查询到表空间的详细信息

```
SELECT TABLESPACE_NAME,INITIAL_EXTENT,NEXT_EXTENT,MIN_EXTENTS,
MAX_EXTENTS,PCT_INCREASE,MIN_EXTLEN,STATUS,
CONTENTS,LOGGING,
EXTENT_MANAGEMENT, -- Columns not available in v8.0.x
ALLOCATION_TYPE, -- Remove these columns if running
PLUGGED_IN, -- against a v8.0.x database
SEGMENT_SPACE_MANAGEMENT --use only in v9.2.x or later
FROM DBA_TABLESPACES
ORDER BY TABLESPACE_NAME;
```

2、对于某些数据文件没有设置为自动扩展的表空间来说，如果表空间满了，就将意味着数据库可能会因为没有空间而停止下来。监控表空间，最主要的就是监控剩余空间的大小或者是使用率。以下是监控表空间使用率与剩余空间大小的语句

```
SELECT D.TABLESPACE_NAME,SPACE "SUM_SPACE(M)",BLOCKS
SUM_BLOCKS,SPACE-NVL(FREE_SPACE,0) "USED_SPACE(M)",
ROUND((1-NVL(FREE_SPACE,0)/SPACE)*100,2) "USED_RATE(%)",FREE_SPACE
"FREE_SPACE(M)"
FROM
(SELECT TABLESPACE_NAME,ROUND(SUM(BYTES)/(1024*1024),2)
SPACE,SUM(BLOCKS) BLOCKS
FROM DBA_DATA_FILES
GROUP BY TABLESPACE_NAME) D,
(SELECT TABLESPACE_NAME,ROUND(SUM(BYTES)/(1024*1024),2) FREE_SPACE
```

```
FROM DBA_FREE_SPACE
GROUP BY TABLESPACE_NAME) F
WHERE D.TABLESPACE_NAME = F.TABLESPACE_NAME(+)
UNION ALL --if have tempfile
SELECT D.TABLESPACE_NAME,SPACE "SUM_SPACE(M)",BLOCKS SUM_BLOCKS,
USED_SPACE "USED_SPACE(M)",ROUND(NVL(USED_SPACE,0)/SPACE*100,2)
"USED_RATE(%)",
NVL(FREE_SPACE,0) "FREE_SPACE(M)"
FROM
(SELECT TABLESPACE_NAME,ROUND(SUM(BYTES)/(1024*1024),2)
SPACE,SUM(BLOCKS) BLOCKS
FROM DBA_TEMP_FILES
GROUP BY TABLESPACE_NAME) D,
(SELECT TABLESPACE_NAME,ROUND(SUM(BYTES_USED)/(1024*1024),2)
USED_SPACE,
ROUND(SUM(BYTES_FREE)/(1024*1024),2) FREE_SPACE
FROM V$TEMP_SPACE_HEADER
GROUP BY TABLESPACE_NAME) F
WHERE D.TABLESPACE_NAME = F.TABLESPACE_NAME(+)
```

3、除了监控表空间的剩余空间，有时候我们也有必要了解一下该表空间是否具有自动扩展空间的能力，虽然我们建议在生产系统中预先分配空间。以下语句将完成这一功能

```
SELECT T.TABLESPACE_NAME,D.FILE_NAME,
D.AUTOEXTENSIBLE,D.BYTES,D.MAXBYTES,D.STATUS
FROM DBA_TABLESPACES T,
DBA_DATA_FILES D
WHERE T. TABLESPACE_NAME =D. TABLESPACE_NAME
ORDER BY TABLESPACE_NAME,FILE_NAME
```

4、我相信使用字典管理的表空间的也不少吧，因为字典管理的表空间中，每个表的下一个区间的大小是不可以预料的，所以我们必须监控那些表在字典管理的表空间中的下一个区间的分配将会引起性能问题或由于是非扩展的表空间而导致系统停止。以下语句检查那些表的扩展将引起表空间的扩展。

```
SELECT A.OWNER,A.TABLE_NAME,A.NEXT_EXTENT,A.TABLESPACE_NAME
FROM ALL_TABLES A,
(SELECT TABLESPACE_NAME, MAX(BYTES) BIG_CHUNK
FROM DBA_FREE_SPACE
```

```
GROUP BY TABLESPACE_NAME) F
WHERE F.TABLESPACE_NAME = A.TABLESPACE_NAME
AND A.NEXT_EXTENT > F.BIG_CHUNK
```

5、段的占用空间与区间数也是很需要注意的一个问题，如果一个段的占用空间太大，或者跨越太多的区间（在字典管理的表空间中，将有严重的性能影响），如果段没有可以再分配的区间，将导致数据库错误。所以，段的大小与区间监控也是一个很重要的工作

```
SELECT S.OWNER,S.SEGMENT_NAME,S.SEGMENT_TYPE,S.PARTITION_NAME,
ROUND(BYTES/(1024*1024),2) "USED_SPACE(M)",
EXTENTS USED_EXTENTS,S.MAX_EXTENTS,S.BLOCKS ALLOCATED_BLOCKS,
S.BLOCKS USED_BOLCKS,S.PCT_INCREASE,S.NEXT_EXTENT/1024
"NEXT_EXTENT(K)"
FROM DBA_SEGMENTS S
WHERE S.OWNER NOT IN ('SYS','SYSTEM')
ORDER BY Used_Extents DESC
```

6、对象的空间分配与空间利用情况，除了从各个方面的分析，如分析表，查询 rowid 等方法外，其实 oracle 提供了一个查询空间的包 dbms_space，如果我们稍封装一下，将是非常好用的一个东西。

```
CREATE OR REPLACE PROCEDURE show_space
(p_segname in varchar2,
p_type in varchar2 default 'TABLE' ,
p_owner in varchar2 default user)
AS
v_segname varchar2(100);
v_type varchar2(10);
l_free_blks number;
l_total_blocks number;
l_total_bytes number;
l_unused_blocks number;
l_unused_bytes number;
l_LastUsedExtFileId number;
l_LastUsedExtBlockId number;
l_LAST_USED_BLOCK number;
PROCEDURE p( p_label in varchar2, p_num in number )
IS
BEGIN
dbms_output.put_line( rpad(p_label,40,'.')|| p_num );
```

```
END;
BEGIN
v_segname := upper(p_segname);
v_type := p_type;
if (p_type = 'i' or p_type = 'I') then
v_type := 'INDEX';
end if;
if (p_type = 't' or p_type = 'T') then
v_type := 'TABLE';
end if;
if (p_type = 'c' or p_type = 'C') then
v_type := 'CLUSTER';
end if;
--以下部分不能用于 ASSM
dbms_space.free_blocks
( segment_owner => p_owner,
segment_name => v_segname,
segment_type => v_type,
freelist_group_id => 0,
free_blks => l_free_blks );
--以上部分不能用于 ASSM
dbms_space.unused_space
( segment_owner => p_owner,
segment_name => v_segname,
segment_type => v_type,
total_blocks => l_total_blocks,
total_bytes => l_total_bytes,
unused_blocks => l_unused_blocks,
unused_bytes => l_unused_bytes,
LAST_USED_EXTENT_FILE_ID => l_LastUsedExtFileId,
LAST_USED_EXTENT_BLOCK_ID => l_LastUsedExtBlockId,
LAST_USED_BLOCK => l_LAST_USED_BLOCK );
--显示结果
p( 'Free Blocks', l_free_blks );
p( 'Total Blocks', l_total_blocks );
p( 'Total Bytes', l_total_bytes );
p( 'Unused Blocks', l_unused_blocks );
p( 'Unused Bytes', l_unused_bytes );
p( 'Last Used Ext FileId', l_LastUsedExtFileId );
p( 'Last Used Ext BlockId', l_LastUsedExtBlockId );
p( 'Last Used Block', l_LAST_USED_BLOCK );
END;
```

执行结果将如下所示

```
SQL> set serveroutput on;
SQL> exec show_space('test');
Free Blocks.............................1
Total Blocks............................8
Total Bytes.............................65536
Unused Blocks...........................6
Unused Bytes............................49152
Last Used Ext FileId....................1
Last Used Ext BlockId...................48521
Last Used Block.........................2
PL/SQL procedure successfully completed
```

8、数据库的索引如果有比较频繁的 Delete 操作，将可能导致索引产生很多碎片，所以，在有的时候，需要对所有的索引重新 REBUILD，以便合并索引块，减少碎片，提高查询速度。

```
SQL> set heading off
SQL> set feedback off
SQL> spool d:index.sql
SQL> SELECT 'alter index ' || index_name || ' rebuild '
||'tablespace INDEXES storage(initial 256K next 256K pctincrease 0);'
FROM all_indexes
WHERE ( tablespace_name != 'INDEXES'
OR next_extent != ( 256 * 1024 )
)
AND owner = USER
SQL>spool off
```

这个时候，我们打开 spool 出来的文件，就可以直接运行了。

9、表的主键是必要的，没有主键的表可以说是不符合设计规范的，所以我们需要监控表是否有主键

```
SELECT table_name
FROM all_tables
WHERE owner = USER
MINUS
SELECT table_name
FROM all_constraints
```

```
WHERE owner = USER
AND constraint_type = 'P'
```

二、性能监控

1、数据缓冲区的命中率已经不是性能调整中的主要问题了，但是，过低的命中率肯定是不可以的，在任何情况下，我们必须保证有一个大的 data buffer 和一个高的命中率。
这个语句可以获得整体的数据缓冲命中率，越高越好

```
SELECT a.VALUE + b.VALUE logical_reads,
c.VALUE phys_reads,
round(100*(1-c.value/(a.value+b.value)),4) hit_ratio
FROM v$sysstat a,v$sysstat b,v$sysstat c
WHERE a.NAME='db block gets'
AND b.NAME='consistent gets'
AND c.NAME='physical reads'
```

2、库缓冲说明了 SQL 语句的重载率，当然，一个 SQL 语句应当被执行的越多越好，如果重载率比较高，就考虑增加共享池大小或者是提高 Bind 变量的使用
以下语句查询了 Sql 语句的重载率，越低越好

```
SELECT SUM(pins) total_pins,SUM(reloads) total_reloads,
SUM(reloads)/SUM(pins)*100 libcache_reload_ratio
FROM v$librarycache
```

3、用户锁，数据库的锁有的时候是比较耗费资源的，特别是发生锁等待的时候，我们必须找到发生等待的锁，有可能的话，杀掉该进程。
这个语句将查找到数据库中所有的 DML 语句产生的锁，还可以发现，任何 DML 语句其实产生了两个锁，一个是表锁，一个是行锁。
可以通过 alter system kill session ‘sid,serial#’ 来杀掉会话

```
SELECT /*+ rule */ s.username,
decode(l.type,'TM','TABLE LOCK',
'TX','ROW LOCK',
NULL) LOCK_LEVEL,
o.owner,o.object_name,o.object_type,
s.sid,s.serial#,s.terminal,s.machine,s.program,s.osuser
FROM v$session s,v$lock l,dba_objects o
WHERE l.sid = s.sid
```

```
AND l.id1 = o.object_id(+)
AND s.username is NOT NULL
```

4、锁与等待，如果发生了锁等待，我们可能更想知道是谁锁了表而引起谁的等待
以下的语句可以查询到谁锁了表，而谁在等待。

```
SELECT /*+ rule */ lpad(' ',decode(l.xidusn ,0,3,0))||l.oracle_username
User_name,
o.owner,o.object_name,o.object_type,s.sid,s.serial#
FROM v$locked_object l,dba_objects o,v$session s
WHERE l.object_id=o.object_id
AND l.session_id=s.sid
ORDER BY o.object_id,xidusn DESC
```

以上查询结果是一个树状结构，如果有子节点，则表示有等待发生。如果想知道锁用了哪个回滚段，还可以关联到 V$rollname，其中 xidusn 就是回滚段的 USN

5、如果发生了事务或锁，想知道哪些回滚段正在被使用吗？其实通过事务表，我们可以详细的查询到事务与回滚段之间的关系。同时，如果关联会话表，我们则可以知道是哪个会话发动了这个事务。

```
SELECT s.USERNAME,s.SID,s.SERIAL#,t.UBAFIL "UBA filenum",
t.UBABLK "UBA Block number",t.USED_UBLK "Number os undo Blocks Used",
t.START_TIME,t.STATUS,t.START_SCNB,t.XIDUSN RollID,r.NAME RollName
FROM v$session s,v$transaction t,v$rollname r
WHERE s.SADDR=t.SES_ADDR
AND t.XIDUSN=r.usn
```

7、如果利用会话跟踪或者是想查看某个会话的跟踪文件，那么查询到 OS 上的进程或线程号是非常重要的，因为文件的令名中，就包含这个信息，以下的语句可以查询到进程或线程号，由此就可以找到对应的文件。

```
SELECT p1.value||''||p2.value||'_ora_'||p.spid filename
FROM
v$process p,
v$session s,
v$parameter p1,
v$parameter p2
WHERE p1.name = 'user_dump_dest'
AND p2.name = 'db_name'
```

```
AND p.addr = s.paddr
AND s.audsid = USERENV ('SESSIONID');
```

8、在 ORACLE 9i 中，可以监控索引的使用，如果没有使用到的索引，完全可以删除掉，减少 DML 操作时的操作。
以下就是开始索引监控与停止索引监控的脚本

```
set heading off
set echo off
set feedback off
set pages 10000
spool start_index_monitor.sql

SELECT 'alter index '||owner||'.'||index_name||' monitoring usage;'
FROM dba_indexes
WHERE owner = USER;

spool off
set heading on
set echo on
set feedback on
-----------------------------------------------
set heading off
set echo off
set feedback off
set pages 10000
spool stop_index_monitor.sql

SELECT 'alter index '||owner||'.'||index_name||' nomonitoring usage;'
FROM dba_indexes
WHERE owner = USER;

spool off
set heading on
set echo on
set feedback on
```

如果需要监控更多的用户，可以将 owner=User 改写成别的
监控结果在视图 v$object_usage 中查询
感谢 fenng，他提供了一个更新版的 show_space 脚本

```
CREATE OR REPLACE PROCEDURE show_space
```

```
( p_segname IN VARCHAR2,
p_owner IN VARCHAR2 DEFAULT USER,
p_type IN VARCHAR2 DEFAULT 'TABLE',
p_partition IN VARCHAR2 DEFAULT NULL )
-- This procedure uses AUTHID CURRENT USER so it can query DBA_*
-- views using privileges from a ROLE and so it can be installed
-- once per database, instead of once per user who wanted to use it.
AUTHID CURRENT_USER
as
l_free_blks number;
l_total_blocks number;
l_total_bytes number;
l_unused_blocks number;
l_unused_bytes number;
l_LastUsedExtFileId number;
l_LastUsedExtBlockId number;
l_LAST_USED_BLOCK number;
l_segment_space_mgmt varchar2(255);
l_unformatted_blocks number;
l_unformatted_bytes number;
l_fs1_blocks number; l_fs1_bytes number;
l_fs2_blocks number; l_fs2_bytes number;
l_fs3_blocks number; l_fs3_bytes number;
l_fs4_blocks number; l_fs4_bytes number;
l_full_blocks number; l_full_bytes number;

-- Inline procedure to print out numbers nicely formatted
-- with a simple label.
PROCEDURE p( p_label in varchar2, p_num in number )
IS
BEGIN
dbms_output.put_line( rpad(p_label,40,'.') ||
to_char(p_num,'999,999,999,999') );
END;
BEGIN
-- This query is executed dynamically in order to allow this procedure
-- to be created by a user who has access to DBA_SEGMENTS/TABLESPACES
-- via a role as is customary.
-- NOTE: at runtime, the invoker MUST have access to these two
-- views!
-- This query determines if the object is an ASSM object or not.
BEGIN
EXECUTE IMMEDIATE
'select ts.segment_space_management
```

```
FROM dba_segments seg, dba_tablespaces ts
WHERE seg.segment_name = :p_segname
AND (:p_partition is null or
seg.partition_name = :p_partition)
AND seg.owner = :p_owner
AND seg.tablespace_name = ts.tablespace_name'
INTO l_segment_space_mgmt
USING p_segname, p_partition, p_partition, p_owner;
EXCEPTION
WHEN too_many_rows THEN
dbms_output.put_line
( 'This must be a partitioned table, use p_partition => ');
RETURN;
END;

-- If the object is in an ASSM tablespace, we must use this API
-- call to get space information; else we use the FREE_BLOCKS
-- API for the user managed segments.
IF l_segment_space_mgmt = 'AUTO'
THEN
dbms_space.space_usage
( p_owner, p_segname, p_type, l_unformatted_blocks,
l_unformatted_bytes, l_fs1_blocks, l_fs1_bytes,
l_fs2_blocks, l_fs2_bytes, l_fs3_blocks, l_fs3_bytes,
l_fs4_blocks, l_fs4_bytes, l_full_blocks, l_full_bytes, p_partition);

p( 'Unformatted Blocks ', l_unformatted_blocks );
p( 'FS1 Blocks (0-25) ', l_fs1_blocks );
p( 'FS2 Blocks (25-50) ', l_fs2_blocks );
p( 'FS3 Blocks (50-75) ', l_fs3_blocks );
p( 'FS4 Blocks (75-100)', l_fs4_blocks );
p( 'Full Blocks ', l_full_blocks );
ELSE
dbms_space.free_blocks(
segment_owner => p_owner,
segment_name => p_segname,
segment_type => p_type,
freelist_group_id => 0,
free_blks => l_free_blks);

p( 'Free Blocks', l_free_blks );
END IF;

-- And then the unused space API call to get the rest of the
```

```
-- information.
dbms_space.unused_space
( segment_owner => p_owner,
segment_name => p_segname,
segment_type => p_type,
partition_name => p_partition,
total_blocks => l_total_blocks,
total_bytes => l_total_bytes,
unused_blocks => l_unused_blocks,
unused_bytes => l_unused_bytes,
LAST_USED_EXTENT_FILE_ID => l_LastUsedExtFileId,
LAST_USED_EXTENT_BLOCK_ID => l_LastUsedExtBlockId,
LAST_USED_BLOCK => l_LAST_USED_BLOCK );

p( 'Total Blocks', l_total_blocks );
p( 'Total Bytes', l_total_bytes );
p( 'Total MBytes', trunc(l_total_bytes/1024/1024) );
p( 'Unused Blocks', l_unused_blocks );
p( 'Unused Bytes', l_unused_bytes );
p( 'Last Used Ext FileId', l_LastUsedExtFileId );
p( 'Last Used Ext BlockId', l_LastUsedExtBlockId );
p( 'Last Used Block', l_LAST_USED_BLOCK );
END;
```

隐含参数：

```
select a.ksppinm "parameter ", a.ksppdesc "descriptoin "
from x$ksppi a,x$ksppcv b,x$ksppsv c
where a.indx=b.indx and a.indx=c.indx and a.ksppinm like '/_%' escape
'/';
```

Check OS process id from Oracle sid

```
select spid from v$process
where addr in ( select paddr from v$session where sid=[$sid) ]
```

Check Oracle sid from OS process id

```
select sid from v$session
where paddr in ( select addr from v$process where spid=[$pid) ]
```

Check current SQL in a session

```
select SQL_TEXT from V$SQLTEXT
where HASH_VALUE =
( select SQL_HASH_VALUE from v$session
where sid = &sid)
order by PIECE
```

Checking v$session_wait

```
select * from v$session_wait
where event not like 'rdbms%'
and event not like 'SQL*N%'
and event not like '%timer';
```

Dictionary Cache Hits

```
SELECT sum(getmisses)/sum(gets) FROM v$rowcache;
/*It should be < 15%, otherwise Add share_pool_size*/
```

Check DB object name from file id and block#

```
select owner,segment_name,segment_type
from dba_extents
where file_id = [$fno and &dno between block_id and block_id + blocks -
1 ]
```

```
#寻找 hot block
select /*+ ordered */
e.owner ||'.'|| e.segment_name segment_name,
e.extent_id extent#,
x.dbablk - e.block_id + 1 block#,
x.tch,
l.child#
from
sys.v$latch_children l,
sys.x$bh x,
sys.dba_extents e
where
l.name = 'cache buffers chains' and
l.sleeps > &sleep_count and
x.hladdr = l.addr and
e.file_id = x.file# and
```

```
x.dbablk between e.block_id and e.block_id + e.blocks - 1;

#找出每个文件上的等待事件
select df.name,kf.count from v$datafile df,x$kcbfwait kf where
(kf.indx+1)=df.file#;

#找出引起等待事件的 SQL 语句.
select sql_text from v$sqlarea a,v$session b,v$session_wait c where
a.address=b.sql_address and b.sid=c.sid and c.event=[$11]

#监控共享池中哪个对象引起了大的内存分配
SELECT * FROM X$KSMLRU WHERE ksmlrsiz > 0;
```

判断你是从 pfile 启动还是 spfile 启动的简单方法！！！

判断你是从 pfile 启动还是 spfile 启动的简单方法！！！

```
select decode(count(*), 1, 'spfile', 'pfile' )
from v$spparameter
where rownum=1
and isspecified='TRUE'
/

DECODE
------
spfile
```

ORACLE 常用技巧和脚本

ORACLE 常用技巧和脚本

1.如何查看 ORACLE 的隐含参数?
ORACLE 的显式参数，除了在 INIT.ORA 文件中定义的外，在 svrmgrl 中用"show parameter *"，可以显示。但 ORACLE 还有一些参数是以“_”，开头的。如我们非常熟悉的“_offline_rollback_segments”等。
这些参数可在 sys.x$ksppi 表中查出。
语句：“select ksppinm from x$ksppi where substr(ksppinm,1,1)='_'; ”

2.如何查看安装了哪些 ORACLE 组件?
进入${ORACLE_HOME}/orainst/，运行./inspdver，显示安装组件和版本号。

3.如何查看 ORACLE 所占用共享内存的大小?
可用 UNIX 命令“ipcs”查看共享内存的起始地址、信号量、消息队列。

在 svrmgrl 下，用“oradebug ipc”，可看出 ORACLE 占用共享内存的分段和大小。
example:
SVRMGR> oradebug ipc
-------------- Shared memory --------------
Seg Id Address Size
1153 7fe000 784
1154 800000 419430400
1155 19800000 67108864

4.如何查看当前 SQL*PLUS 用户的 sid 和 serial#?
在 SQL*PLUS 下，运行：
“select sid, serial#, status from v$session
where audsid=userenv('sessionid');”

5.如何查看当前数据库的字符集？
在 SQL*PLUS 下，运行：
“select userenv('language') from dual;”
或：“select userenv('lang') from dual; ”

6.如何查看数据库中某用户，正在运行什么 SQL 语句？
根据 MACHINE、USERNAME 或 SID、SERIAL#，连接表 V$SESSION 和 V$SQLTEXT，可查出。
SQL*PLUS 语句：
“SELECT SQL_TEXT FROM V$SQL_TEXT T, V$SESSION S WHERE
T.ADDRESS=S.SQL_ADDRESS
AND T.HASH_VALUE=S.SQL_HASH_VALUE
AND S.MACHINE='XXXXX' OR USERNAME='XXXXX' -- 查看某主机名，或用户名
/”
7.如何删除表中的重复记录？
例句：
DELETE
FROM table_name a
WHERE rowid > (SELECT min(rowid)
FROM table_name b
WHERE b.pk_column_1 = a.pk_column_1
and b.pk_column_2 = a.pk_column_2);

8.手工临时强制改变服务器字符集
以 sys 或 system 登录系统，sql*plus 运行：“create database character set us7ascii;”.
有以下错误提示：
* create database character set US7ASCII
ERROR at line 1:

ORA-01031: insufficient privileges
实际上，看 v$nls_parameters，字符集已更改成功。但重启数据库后，数据库字符集又变回原来的了。
该命令可用于临时的不同字符集服务器之间数据倒换之用。
9.怎样查询每个 instance 分配的 PCM 锁的数目
用以下命令：
select count(*) "Number of hashed PCM locks" from v$lock_element where bitand(flags,4)<>0
/
select count(*) "Number of fine grain PCM locks" from v$lock_element where bitand(flags,4)=0
/
10. 怎么判断当前正在使用何种 SQL 优化方式？
用 explain plan 产生 EXPLAIN PLAN，检查 PLAN_TABLE 中 ID=0 的 POSITION 列的值。
e.g.
select decode(nvl(position,-1),-1,'RBO',1,'CBO') from plan_table where id=0
/
11. 做 EXPORT 时，能否将 DUMP 文件分成多个？
ORACLE8I 中 EXP 增加了一个参数 FILESIZE，可将一个文件分成多个：
EXP SCOTT/TIGER FILE=(ORDER_1.DMP,ORDER_2.DMP,ORDER_3.DMP) FILESIZE=1G TABLES=ORDER;

其他版本的 ORACLE 在 UNIX 下可利用管道和 split 分割：
mknod pipe p
split -b 2048m pipe order & #将文件分割成，每个 2GB 大小的，以 order 为前缀的文件：
#orderaa,orderab,orderac,... 并将该进程放在后台。
EXP SCOTT/TIGER FILE=pipe tables=order
户如何有效地利用数据字典

用户如何有效地利用数据字典

ORACLE 的数据字典是数据库的重要组成部分之一，它随着数据库的产生而产生,随着数据库的变化而变化,
体现为 sys 用户下的一些表和视图。数据字典名称是大写的英文字符。

数据字典里存有用户信息、用户的权限信息、所有数据对象信息、表的约束条件、统计分析数据库的视图等。
我们不能手工修改数据字典里的信息。

很多时候，一般的 ORACLE 用户不知道如何有效地利用它。

dictionary 全部数据字典表的名称和解释，它有一个同义词 dict
dict_column 全部数据字典表里字段名称和解释

如果我们想查询跟索引有关的数据字典时，可以用下面这条 SQL 语句:

```
SQL>select * from dictionary where instr(comments,'index')>0;
```

如果我们想知道 user_indexes 表各字段名称的详细含义，可以用下面这条 SQL 语句:

```
SQL>select column_name,comments from dict_columns where
table_name='USER_INDEXES';
```

依此类推，就可以轻松知道数据字典的详细名称和解释，不用查看 ORACLE 的其它文档资料了。

下面按类别列出一些 ORACLE 用户常用数据字典的查询使用方法。
一、用户

查看当前用户的缺省表空间
```
SQL>select username,default_tablespace from user_users;
```

查看当前用户的角色
```
SQL>select * from user_role_privs;
```

查看当前用户的系统权限和表级权限
```
SQL>select * from user_sys_privs;
SQL>select * from user_tab_privs;
```

SQL-1 =====查看控制文件

SQL-2 =====查看日志文件

SQL-3 =====查看表空间使用情况

SQL-4 =====查看数据库库对象

SQL-5 =====查看数据库的版本

SQL-6 =====查看数据库创建日期和归档方式

SQL-7 =====捕捉运行很久的 SQL

SQL-8 =====查看数据表的参数信息

SQL-9 =====查看表空间的名称及大小

SQL-10 =====表空间相关查询方法

SQL-11 =====查看回滚段名称及大小

SQL-12 =====查看当前 SQL*PLUS 用户的 sid 和 serial

SQL-13 =====如何查看当前数据库的字符集

SQL-14 =====查询 SQL 优化方式

SQL-15 =====查看系统当前最新的 SCN 号

SQL-16 =====查看 TRACE 文件脚本

SQL-17 =====查看客户端登陆 IP

SQL-18 =====创建追踪客户端 IP 地址

SQL-19 =====查看数据库当前日期

SQL-20 =====查看 Disk Read 最高的 SQL

SQL-21 =====查找前十条性能差的 sql

SQL-22 =====获取等待时间最多的 5 个系统等待事件

SQL-23 =====检查 Oracle 回滚段状态

SQL-24 =====检查 Oracle 回滚段扩展信息

SQL-25 =====Oracle 杀会话的脚本

SQL-26 =====查看排序段的性能

SQL-27 =====查看数据库对象

SQL-28 =====查看尚未提交的事务

SQL-29 =====查找 object 为哪些进程所用

SQL-30 =====查看回滚段

SQL-31 =====耗资源的进程(top session)

SQL-32 =====根据 PID 查找相应的语句

SQL-33 =====监控当前数据库谁在运行什么 SQL 语句

SQL-34 =====监控数据库某用户在运行什么 SQL

SQL-35 =====查询前台正在发出的 sql 语句

SQL-36 =====查询当前所执行的 SQL 语句

SQL-37 =====监控消耗 CPU 最高的进程所对应的 SQL 语句

SQL-38 =====监控 CPU 使用率最高的 2 条 SQL 语句

SQL-39 =====查询锁（Lock）情况

SQL-40 =====DBA 监控数据库死锁

SQL-41 =====查看等待（wait）情况

SQL-42 =====查看 sga 情况

SQL-43 =====查看 catched object

SQL-44 =====查看 V$SQLAREA

SQL-45 =====查看 object 分类数量

SQL-46 =====查看 connection 的相关信息

SQL-47 =====查询有哪些数据库实例在运行

SQL-48 =====查看表是否是分区表

SQL-49 =====查看分区表的分区名和相应的表空间名

SQL-50 =====查看索引是否是分区索引

SQL-51 =====Dual 表的用法，常用在没有目标表的 Select 中

SQL-52 =====查看索引段中 extent 的数量

SQL-53 =====查看系统表空间中的非管理员索引

SQL-54 =====查看 system 表空间内的索引的扩展情况

SQL-55 =====查看表空间数据文件的读写性能

SQL-56 =====转换表空间为 local 方式

SQL-57 =====查看一下哪个用户在用临时段

SQL-58 =====查看占 io 较大的正在运行的 session

SQL-59 =====查找前十条性能差的 sql

SQL-60 =====删除用户下所有表的语句

SQL-61 =====查看 LOCK 并杀掉会话

SQL-62 =====识别 IO 竞争和负载平衡

SQL-63 =====查看哪些 session 正在使用哪些回滚段

SQL-64 =====查看 WACOS 表空间下所有的索引

==========

SQL-1　=====查看控制文件

==========

```
select name from v$controlfile;
```

==========

SQL-2 =====查看日志文件

==========

select member from v$logfile;

==========

SQL-3 =====查看表空间使用情况

==========

select sum(bytes)/(1024*1024) as free_space,tablespace_name from dba_free_space group by tablespace_name;

==========

SQL-4 =====查看数据库库对象

==========

select owner, object_type, status, count(*) count# from all_objects group by owner, object_type, status;

==========

SQL-5 =====查看数据库的版本

==========

Select version FROM Product_component_version

Where SUBSTR(PRODUCT,1,6)='Oracle';

==========

SQL-6 =====查看数据库创建日期和归档方式

==========

Select Created, Log_Mode, Log_Mode From V$Database;

==========

SQL-7 =====捕捉运行很久的 SQL

```
============================================================
column username format a12
column opname format a16
column progress format a8

select username,sid,opname, round(sofar*100 / totalwork,0) || '%' as progress,
time_remaining,sql_text
   from v$session_longops , v$sql
  where time_remaining <> 0
     and sql_address = address
     and sql_hash_value = hash_value

============================================================
```

SQL-8 =====查看数据表的参数信息

```
============================================================
SELECT     partition_name, high_value, high_value_length, tablespace_name,
              pct_free, pct_used, ini_trans, max_trans, initial_extent,
              next_extent, min_extent, max_extent, pct_increase, FREELISTS,
              freelist_groups, LOGGING, BUFFER_POOL, num_rows, blocks,
              empty_blocks, avg_space, chain_cnt, avg_row_len, sample_size,
              last_analyzed
       FROM dba_tab_partitions
     --WHERE table_name = :tname AND table_owner = :towner
ORDER BY partition_position

============================================================
```

SQL-9 =====查看表空间的名称及大小

```
============================================================
select t.tablespace_name, round(sum(bytes/(1024*1024)),0) ts_size
```

from dba_tablespaces t, dba_data_files d where t.tablespace_name = d.tablespace_name group by t.tablespace_name;

==========

SQL-10 =====表空间相关查询方法

==========

SQL>select tablespace_name, file_id, file_name,round(bytes/(1024*1024),0) total_space from dba_data_files order by tablespace_name;

SQL>select distinct file_name,tablespace_name,AUTOEXTENSIBLE from dba_data_files;

**查询表空间使用情况

SQL>select sum(bytes)/(1024*1024) as free_space,tablespace_name from dba_free_space group by tablespace_name;

SQL>SELECT A.TABLESPACE_NAME,A.BYTES TOTAL,B.BYTES USED, C.BYTES FREE,(B.BYTES*100)/A.BYTES "% USED",(C.BYTES*100)/A.BYTES "% FREE"

FROM SYS.SM$TS_AVAIL A,SYS.SM$TS_USED B,SYS.SM$TS_FREE C

WHERE A.TABLESPACE_NAME=B.TABLESPACE_NAME AND A.TABLESPACE_NAME=C.TABLESPACE_NAME;

SQL>column tablespace_name format a18;

SQL>column Sum_M format a12;

SQL>column Used_M format a12;

SQL>column Free_M format a12;

SQL>column pto_M format 9.99;

SQL>select s.tablespace_name,ceil(sum(s.bytes/1024/1024))||'M' Sum_M,ceil(sum(s.UsedSpace/1024/1024))||'M' Used_M,ceil(sum(s.FreeSpace/1024/1024))||'M' Free_M,

sum(s.UsedSpace)/sum(s.bytes) PTUSED

from (select b.file_id,b.tablespace_name,b.bytes, (b.bytes-sum(nvl(a.bytes,0))) UsedSpace, sum(nvl(a.bytes,0)) FreeSpace,(sum(nvl(a.bytes,0))/(b.bytes)) * 100 FreePercentRatio

from sys.dba_free_space a,sys.dba_data_files b

where a.file_id(+)=b.file_id group by b.file_id,b.tablespace_name,b.bytes order by b.tablespace_name) s

group by s.tablespace_name

order by sum(s.FreeSpace)/sum(s.bytes) desc;

数据库各个表空间增长情况的检查：

SQL>select A.tablespace_name,(1-(A.total)/B.total)*100 used_percent

From (select tablespace_name,sum(bytes) total from dba_free_space group by tablespace_name) A,(select tablespace_name,sum(bytes) total from dba_data_files group by tablespace_name) B where A.tablespace_name=B.tablespace_name;

SQL>SELECT UPPER(F.TABLESPACE_NAME) "表空间名",

D.TOT_GROOTTE_MB "表空间大小(M)",

D.TOT_GROOTTE_MB - F.TOTAL_BYTES "已使用空间(M)", TO_CHAR(ROUND((D.TOT_GROOTTE_MB - F.TOTAL_BYTES) / D.TOT_GROOTTE_MB * 100, 2), '990.99') "使用比", F.TOTAL_BYTES "空闲空间(M)",

F.MAX_BYTES "最大块(M)" FROM (SELECT TABLESPACE_NAME,

ROUND(SUM(BYTES) / (1024 * 1024), 2) TOTAL_BYTES,

ROUND(MAX(BYTES) / (1024 * 1024), 2) MAX_BYTES

FROM SYS.DBA_FREE_SPACE GROUP BY TABLESPACE_NAME) F,

(SELECT DD.TABLESPACE_NAME,ROUND(SUM(DD.BYTES) / (1024 * 1024), 2) TOT_GROOTTE_MB FROM SYS.DBA_DATA_FILES DD

GROUP BY DD.TABLESPACE_NAME) D WHERE D.TABLESPACE_NAME = F.TABLESPACE_NAME

```
ORDER BY 4 DESC;
```

查看各个表空间占用磁盘情况:

```
SQL>col tablespace_name format a20;
SQL>select    b.file_id    file_ID,
b.tablespace_name    tablespace_name,
b.bytes    Bytes,
(b.bytes-sum(nvl(a.bytes,0)))    used,
sum(nvl(a.bytes,0))    free,
sum(nvl(a.bytes,0))/(b.bytes)*100 Percent
        from dba_free_space a,dba_data_files b
        where a.file_id=b.file_id
        group by b.tablespace_name,b.file_id,b.bytes
        order by b.file_id;
```

数据库对象下一扩展与表空间的 free 扩展值的检查：

```
SQL>select a.table_name, a.next_extent, a.tablespace_name
from all_tables a,(select tablespace_name, max(bytes) as big_chunk
from dba_free_space group by tablespace_name ) f where f.tablespace_name =
a.tablespace_name and a.next_extent > f.big_chunk
union select a.index_name, a.next_extent, a.tablespace_name
from all_indexes a,(select tablespace_name, max(bytes) as big_chunk
from dba_free_space group by tablespace_name ) f where f.tablespace_name =
a.tablespace_name and a.next_extent > f.big_chunk;
```

查询表空间使用情况:

```
select a.tablespace_name "表空间名称",
100-round((nvl(b.bytes_free,0)/a.bytes_alloc)*100,2) "占用率(%)",
round(a.bytes_alloc/1024/1024,2) "容量(M)",
```

```
round(nvl(b.bytes_free,0)/1024/1024,2) "空闲(M)",
round((a.bytes_alloc-nvl(b.bytes_free,0))/1024/1024,2) "使用(M)",
Largest "最大扩展段(M)",to_char(sysdate,'yyyy-mm-dd hh24:mi:ss') "采样时间"
from (select f.tablespace_name,sum(f.bytes) bytes_alloc,
sum(decode(f.autoextensible,'YES',f.maxbytes,'NO',f.bytes)) maxbytes
from dba_data_files f group by tablespace_name) a,
(select f.tablespace_name,sum(f.bytes) bytes_free
from dba_free_space f group by tablespace_name) b,
(select round(max(ff.length)*16/1024,2) Largest,ts.name tablespace_name
from sys.fet$ ff, sys.file$ tf,sys.ts$ ts
where ts.ts#=ff.ts# and ff.file#=tf.relfile# and ts.ts#=tf.ts#
group by ts.name, tf.blocks) c where a.tablespace_name = b.tablespace_name and
a.tablespace_name = c.tablespace_name;

SELECT UPPER(F.TABLESPACE_NAME) "表空间名",
       D.TOT_GROOTTE_MB "表空间大小(M)",
       D.TOT_GROOTTE_MB - F.TOTAL_BYTES "已使用空间(M)",
       TO_CHAR(ROUND((D.TOT_GROOTTE_MB - F.TOTAL_BYTES) /
D.TOT_GROOTTE_MB * 100,
                     2),
               '990.99') "使用比",
       F.TOTAL_BYTES "空闲空间(M)",
       F.MAX_BYTES "最大块(M)"
  FROM (SELECT TABLESPACE_NAME,
               ROUND(SUM(BYTES) / (1024 * 1024), 2) TOTAL_BYTES,
               ROUND(MAX(BYTES) / (1024 * 1024), 2) MAX_BYTES
          FROM SYS.DBA_FREE_SPACE
         GROUP BY TABLESPACE_NAME) F,
       (SELECT DD.TABLESPACE_NAME,
               ROUND(SUM(DD.BYTES) / (1024 * 1024), 2)
```

```
TOT_GROOTTE_MB
                FROM SYS.DBA_DATA_FILES DD
               GROUP BY DD.TABLESPACE_NAME) D
WHERE D.TABLESPACE_NAME = F.TABLESPACE_NAME
ORDER BY 4 DESC;
```

查询表空间的碎片程度:

```
SQL>select tablespace_name,count(tablespace_name) from dba_free_space group by tablespace_name having count(tablespace_name)>10;

SQL>alter tablespace name coalesce;
SQL>alter table table_name deallocate unused;

SQL>create or replace view ts_blocks_v as
select tablespace_name,block_id,bytes,blocks,'free space' segment_name from dba_free_space union all
select tablespace_name,block_id,bytes,blocks,segment_name from dba_extents;

SQL>select * from ts_blocks_v;

SQL>select tablespace_name,sum(bytes),max(bytes),count(block_id) from dba_free_space group by tablespace_name;

SQL>select 'alter tablespace '||TABLESPACE_NAME||' coalesce;'
from DBA_FREE_SPACE_COALESCED where
PERCENT_EXTENTS_COALESCED<100
or PERCENT_BLOCKS_COALESCED<100;
```

由于自由空间碎片是由几部分组成，如范围数量、最大范围尺寸等，我们可

用 fsfi--free space fragmentation index（自由空间碎片索引）值来直观体现:

```
fsfi=100*sqrt(max(extent)/sum(extents))*1/sqrt(sqrt(count(extents)))
rem fsfi value compute
rem fsfi.sql
column fsfi format 999,99
select tablespace_name,sqrt(max(blocks)/sum(blocks))*
(100/sqrt(sqrt(count(blocks)))) fsfi
from dba_free_space
group by tablespace_name order by 1;
spool fsfi.rep;
/
spool off;
```

可以看出，fsfi 的最大可能值为 100（一个理想的单文件表空间）。随着范围的增加，fsfi 值缓慢下降，而随着最大范围尺寸的减少，fsfi 值会迅速下降。

比如，在某数据库运行脚本 fsfi.sql,得到以下 fsfi 值：

```
tablespace_name fsfi
------------------------------ -------
rbs 74.06
system 100.00
temp 22.82
tools 75.79
users 100.00
user_tools 100.00
ydcx_data 47.34
ydcx_idx 57.19
ydjf_data 33.80
ydjf_idx 75.55
```

---- 统计出了数据库的 fsfi 值，就可以把它作为一个可比参数。在一个有着足够有效自由空间，且 fsfi 值超过 30 的表空间中，很少会遇见有效自由空间的问题。当一个空间将要接近可比参数时，就需要做碎片整理了。

==========

SQL-11 =====查看回滚段名称及大小

==========

SQL>select segment_name, tablespace_name, r.status,
(initial_extent/1024) InitialExtent,(next_extent/1024) NextExtent,
max_extents, v.curext CurExtent From dba_rollback_segs r, v$rollstat v
Where r.segment_id = v.usn(+) order by segment_name;

==========

SQL-12 =====查看当前 SQL*PLUS 用户的 sid 和 serial

==========

SQL>select sid, serial#, status from v$session where audsid=userenv('sessionid');

==========

SQL-13 =====如何查看当前数据库的字符集

==========

SQL>select userenv('language') from dual;
SQL>select userenv('lang') from dual;

==========

SQL-14 =====查询 SQL 优化方式

==========

用 explain plan 产生 EXPLAIN PLAN，检查 PLAN_TABLE 中 ID=0 的 POSITION 列的值。

SQL>select decode(nvl(position,-1),-1,'RBO',1,'CBO') from plan_table where id=0;

==========

SQL-15 =====查看系统当前最新的 SCN 号

==========

SQL>select max(ktuxescnw * power(2,32) + ktuxescnb) from x$ktuxe;

==========

SQL-16 =====查看 TRACE 文件脚本

==========

在 ORACLE 中查找 TRACE 文件的脚本:

select u_dump.value || '/' || instance.value || '_ora_' ||
v$process.spid || nvl2(v$process.traceid, '_' || v$process.traceid, null) || '.trc'"Trace File" from v$parameter u_dump cross join v$parameter instance cross join v$process join v$session on v$process.addr = v$session.paddr where u_dump.name = 'user_dump_dest' and
instance.name = 'instance_name' and
v$session.audsid=sys_context('userenv','sessionid');

==========

SQL-17 =====查看客户端登陆 IP

==========

SQL>select sys_context('userenv','ip_address') from dual;

==========

SQL-18 =====创建追踪客户端 IP 地址

==========

```
SQL>create or replace trigger on_logon_trigger
after logon on database
begin
   dbms_application_info.set_client_info(sys_context('userenv', 'ip_address'));
end;
```

==========

SQL-19 =====查看数据库当前日期

==========

SQL> select to_char(sysdate,'yyyy-mm-dd,hh24:mi:ss') from dual;

==========

SQL-20 =====查看 Disk Read 最高的 SQL

==========

SQL>select sql_text from (select * from v$sqlarea order by disk_reads)
where rownum<=5;

==========

SQL-21 =====查找前十条性能差的 sql

==========

SELECT * FROM (SELECT PARSING_USER_ID
EXECUTIONS,SORTS,COMMAND_TYPE,DISK_READS,
sql_text FROM v$sqlarea ORDER BY disk_reads DESC)
WHERE ROWNUM<10 ;

==========

SQL-22 =====获取等待时间最多的 5 个系统等待事件

==========

SQL>select * from (select * from v$system_event where event not like 'SQL%' order
by total_waits desc) where rownum<=5;

==========

SQL-23 =====检查 Oracle 回滚段状态

==========

SQL>select

segment_name,owner,tablespace_name,initial_extent,next_extent,dba_rollback_segs.status from dba_rollback_segs,v$datafile where file_id=file#;

==========

SQL-24 =====检查 Oracle 回滚段扩展信息

==========

```
col name format a10
set linesize 140
select substr(name,1,40) name,extents,rssize,optsize,aveactive,extends,wraps,shrinks,hwmsize
from v$rollname rn,v$rollstat rs where (rn.usn=rs.usn);
```

extents:回滚段中的盘区数量。

Rssize:以字节为单位的回滚段的尺寸。

optsize：为 optimal 参数设定的值。

Aveactive:从回滚段中删除盘区时释放的以字节为单位的平均空间的大小。

Extends:系统为回滚段增加的盘区的次数。

Shrinks:系统从回滚段中清除盘区（即回滚段收缩）的次数。回滚段每次清除盘区时，系统可能会从这个回滚段中消除一个或多个盘区。

Hwmsize:回滚段尺寸的上限，即回滚段曾经达到的最大尺寸。

(如果回滚段平均尺寸接近 OPTIMAL 的值，那么说明 OPTIMAL 的值设置正确，如果回滚段动态增长次数或收缩次数很高，那么需要提高 OPTIMAL 的值)

==========

SQL-25 =====Oracle 杀会话的脚本

==========

```
select
A.SID,B.SPID,A.SERIAL#,a.lockwait,A.USERNAME,A.OSUSER,a.logon_time,a.last_call_et/3600 LAST_HOUR,A.STATUS,
'orakill '||sid||' '||spid HOST_COMMAND,
```

'alter system kill session '''||A.sid||','||A.SERIAL#||''' SQL_COMMAND

from v$session A,V$PROCESS B where A.PADDR=B.ADDR AND SID>6;

==========

SQL-26 =====查看排序段的性能

==========

SQL>SELECT name, value FROM v$sysstat WHERE name IN ('sorts (memory)', 'sorts (disk)');

==========

SQL-27 =====查看数据库对象

==========

select owner, object_type, status, count(*) count# from all_objects group by owner, object_type, status;

==========

SQL-28 =====查看尚未提交的事务

==========

select * from v$locked_object;

select * from v$transaction;

==========

SQL-29 =====查找 object 为哪些进程所用

==========

select p.spid,s.sid,s.serial# serial_num,s.username user_name,

a.type object_type,s.osuser os_user_name,a.owner,a.object object_name,decode(sign(48 - command),1,

to_char(command), 'Action Code #' || to_char(command)) action,

p.program oracle_process,s.terminal terminal,s.program program,s.status session_status from v$session s, v$access a, v$process p where s.paddr = p.addr

and s.type = 'USER' and a.sid = s.sid and a.object='SUBSCRIBER_ATTR'order by s.username, s.osuser;

==========

SQL-30 =====查看回滚段

==========

SQL>col name format a10

SQL>set linesize 100

SQL>select rownum, sys.dba_rollback_segs.segment_name Name, v$rollstat.extents Extents, v$rollstat.rssize Size_in_Bytes, v$rollstat.xacts XActs, v$rollstat.gets Gets, v$rollstat.waits Waits, v$rollstat.writes Writes, sys.dba_rollback_segs.status status from v$rollstat, sys.dba_rollback_segs, v$rollname where v$rollname.name(+) = sys.dba_rollback_segs.segment_name and v$rollstat.usn (+) = v$rollname.usn order by rownum;

==========

SQL-31 =====耗资源的进程(top session)

==========

select s.schemaname schema_name,decode(sign(48 - command), 1, to_char(command), 'Action Code #' || to_char(command)) action,status session_status,s.osuser os_user_name,s.sid,p.spid,s.serial# serial_num,nvl(s.username,'[Oracle process]') user_name,s.terminal terminal,s.program program,st.value criteria_value from v$sesstat st,v$session s,v$process p where st.sid = s.sid and st.statistic# = to_number('38') and ('ALL'='ALL' or s.status ='ALL') and p.addr=s.paddr order by st.value desc,p.spid asc,s.username asc,s.osuser asc;

==========

SQL-32 =====根据 PID 查找相应的语句

==========

```
SELECT a.username,
        a.machine,a.program,a.sid,a.serial#,a.status,c.piece,c.sql_text
  FROM v$session a,v$process b,v$sqltext c WHERE b.spid=spid
   AND b.addr=a.paddr AND a.sql_address=c.address(+) ORDER BY c.piece;
```

==========

SQL-33 =====监控当前数据库谁在运行什么 SQL 语句

==========

```
SQL>SELECT osuser, username, sql_text from v$session a, v$sqltext b
where a.sql_address =b.address order by address, piece;
```

==========

SQL-34 =====监控数据库某用户在运行什么 SQL

==========

```
SQL>SELECT SQL_TEXT FROM V$SQLTEXT T, V$SESSION S WHERE
T.ADDRESS=S.SQL_ADDRESS
AND T.HASH_VALUE=S.SQL_HASH_VALUE AND S.MACHINE='XXXXX'
OR USERNAME='WACOS';
```

==========

SQL-35 =====查询前台正在发出的 sql 语句

==========

```
SQL> select user_name,sql_text from v$open_cursor where sid in(select sid from
(select sid,serial# from v$session where status='ACTIVE'));
```

==========

SQL-36 =====查询当前所执行的 SQL 语句

==========

步骤 1 查询 SQL 执行地址*

```
SQL> select program ,sql_address from v$session where paddr in (select addr
```

from v$process where spid=3556);

PROGRAM SQL_ADDRESS

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

sqlplus@ctc20 (TNS V1-V3) 000000038FCB1A90

步骤 2 根据 SQL 执行的内存地址，查询 SQL 语句*

SQL> select sql_text from v$sqlarea where address='000000038FCB1A90';

==========

SQL-37 =====监控消耗 CPU 最高的进程所对应的 SQL 语句

==========

set line 240

set verify off

column sid format 999

column pid format 999

column S_# format 999

column username format A9 heading "ORA User"

column program format a29

column SQL format a60

COLUMN OSname format a9 Heading "OS User"

SQL>SELECT P.pid pid,S.sid sid,P.spid spid,S.username username,S.osuser osname,P.serial# S_#,P.terminal,P.program program,P.background,S.status,RTRIM(SUBSTR(a.sql_text, 1, 80)) SQL

FROM v$process P, v$session S,v$sqlarea A WHERE P.addr = s.paddr AND S.sql_address = a.address (+) AND P.spid LIKE '%&1%';

Enter value for 1: PID（这里输入占用 CPU 最高的进程对应的 PID）

set termout off

spool maxcpu.txt

SQL>SELECT '++'||S.username

username,RTRIM(REPLACE(a.sql_text,chr(10),''))||';'

FROM v$process P, v$session S,v$sqlarea A WHERE P.addr = s.paddr AND S.sql_address = a.address (+) AND P.spid LIKE '%&&1%';

Enter value for 1: PID（这里输入占用 CPU 最高的进程对应的 PID）

spool off(这句放在最后执行)

==========

SQL-38 =====监控 CPU 使用率最高的 2 条 SQL 语句

==========

执行：top，通过 top 获得 CPU 占用率最高的进程的 pid。

SQL>select sql_text,spid,v$session.program,process from v$sqlarea,v$session,v$process where v$sqlarea.address=v$session.sql_address and v$sqlarea.hash_value=v$session.sql_hash_value

and v$session.paddr=v$process.addr and v$process.spid in (pid);

==========

SQL-39 =====查询锁（Lock）情况

==========

SQL>select /*+ RULE */ ls.osuser os_user_name, ls.username user_name,

decode(ls.type,'RW','Row wait enqueue lock','TM','DML enqueue lock',

'TX','Transaction enqueue lock','UL','User supplied lock') lock_type,

o.object_name object,decode(ls.lmode, 1, null, 2,'Row Share', 3,'Row Exclusive',4,'Share',5,'Share Row Exclusive',6,'Exclusive',null) lock_mode

,o.owner,ls.sid,ls.serial# serial_num,ls.id1,ls.id2

from sys.dba_objects o,(select s.osuser,s.username,l.type,l.lmode,s.sid,s.serial#,l.id1,l.id2 from v$session s,v$lock l where s.sid=l.sid) ls

where o.object_id=ls.id1 and o.owner<>'SYS' order by o.owner, o.object_name;

```
SQL>select sys.v_$session.osuser,sys.v_$session.machine,v$lock.sid,
sys.v_$session.serial#,
       decode(v$lock.type,'MR','Media Recovery','RT','Redo
Thread','UN','User Name','TX', 'Transaction','TM','DML','UL','PL/SQL User
Lock','DX','Distributed Xaction','CF','Control File', 'IS','Instance State','FS','File
Set','IR','Instance Recovery', 'ST','Disk Space Transaction','TS','Temp
Segment','IV','Library Cache Invalida-tion','LS','Log Start or Switch','RW','Row
Wait','SQ','Sequence Number','TE','Extend Table','TT','Temp Table','Unknown')
LockType,
       rtrim(object_type)||' '||rtrim(owner)||'.'|| object_name object_name,
       decode(lmode, 0, 'None',1, 'Null',2, 'Row-S',3, 'Row-X',4, 'Share', 5,
'S/Row-X',6, 'Exclusive','Unknown') LockMode,decode(request, 0, 'None',1, 'Null',2,
'Row-S',3, 'Row-X', 4, 'Share',5, 'S/Row-X', 6, 'Exclusive', 'Unknown') RequestMode,
       ctime, block b
  from v$lock, all_objects, sys.v_$session
 where v$Lock.sid > 6 and sys.v_$session.sid = v$lock.sid and v$lock.id1 =
all_objects.object_id;

==========================================================
SQL-40 =====DBA 监控数据库死锁
==========================================================
col owner for a12
col object_name for a16
select b.owner,b.object_name,l.session_id,l.locked_mode
   from v$locked_object l, dba_objects b
  where b.object_id=l.object_id;

SQL>select t2.username,t2.sid,t2.serial#,t2.logon_time
    from v$locked_object t1,v$session t2
   where t1.session_id=t2.sid order by t2.logon_time;
```

SQL>Select sql_address from v$session where sid=<sid>;

SQL>Select * from v$sqltext where address=<sql_address>;

SQL>select COMMAND_TYPE,PIECE,sql_text from v$sqltext where address=(select sql_address from v$session a where sid=18);

SQL>select object_id from v$locked_object;

SQL>select object_name,object_type from dba_objects where object_id='';

select object_id,session_id,locked_mode from v$locked_object;
select t2.username,t2.sid,t2.serial#,t2.logon_time from v$locked_object t1,v$session t2 where t1.session_id=t2.sid order by t2.logon_time;

如果有长期出现的一列，可能是没有释放的锁。我们可以用下面 SQL 语句杀掉长期没有释放非正常的锁*
SQL>alter system kill session 'sid,serial#';

==========

SQL-41 =====查看等待（wait）情况

==========

SQL>SELECT v$waitstat.class,v$waitstat.count count, SUM(v$sysstat.value) sum_value FROM v$waitstat,v$sysstat WHERE v$sysstat.name IN('db block gets','consistent gets') group by v$waitstat.class,v$waitstat.count;

==========

SQL-42 =====查看 sga 情况

==========

SQL>SELECT NAME, BYTES FROM SYS.V_$SGASTAT ORDER BY NAME ASC;

==========

SQL-43 =====查看 catched object

==========

SQL>SELECT owner,name,db_link,namespace,type,sharable_mem,loads, executions,locks,pins,kept FROM v$db_object_cache;

==========

SQL-44 =====查看 V$SQLAREA

==========

SQL>SELECT SQL_TEXT,SHARABLE_MEM,PERSISTENT_MEM,RUNTIME_MEM,SORTS, VERSION_COUNT,LOADED_VERSIONS,OPEN_VERSIONS,USERS_OPENING, EXECUTIONS, USERS_EXECUTING,LOADS,FIRST_LOAD_TIME,INVALIDATIONS,PARSE_CALLS,DISK_READS,BUFFER_GETS,ROWS_PROCESSED FROM V$SQLAREA;

==========

SQL-45 =====查看 object 分类数量

==========

select decode(o.type#,1,'INDEX',2,'TABLE',3,'CLUSTER',4,'VIEW',5,'SYNONYM',6, 'SEQUENCE','OTHER') object_type , count(*) quantity from sys.obj$ o where o.type# > 1 group by decode(o.type#,1,'INDEX',2,'TABLE',3,'CLUSTER' ,4,'VIEW',5,'SYNONYM',6,'SEQUENCE','OTHER') union select 'COLUMN', count(*) from sys.col$ union select 'DB LINK' , count(*) from all_objects;

==========

SQL-46 =====查看 connection 的相关信息

==========

1）查看有哪些用户连接

```
select s.osuser os_user_name,decode(sign(48 - command),1,to_char(command),
'Action Code #' || to_char(command))action,p.program oracle_process,
status session_status,s.terminal terminal,s.program program,
s.username user_name,s.fixed_table_sequence activity_meter,''query,
0 memory,0 max_memory,0 cpu_usage,s.sid,s.serial# serial_num
from v$session s,v$process p where s.paddr=p.addr and s.type = 'USER'
order by s.username, s.osuser;
```

2）根据 v.sid 查看对应连接的资源占用等情况

```
select n.name,v.value,n.class,n.statistic#
from v$statname n,v$sesstat v where v.sid=18 and v.statistic# = n.statistic# order by
n.class, n.statistic#;
```

3）根据 sid 查看对应连接正在运行的 sql

```
select /*+ PUSH_SUBQ */ command_type,sql_text,sharable_mem,
persistent_mem,runtime_mem,sorts,version_count,
loaded_versions,open_versions,users_opening,executions,
users_executing,loads,first_load_time,invalidations,
parse_calls,disk_reads,buffer_gets,rows_processed,sysdate start_time,sysdate
finish_time,'>'|| address sql_address,
'N' status from v$sqlarea where address = (select sql_address from v$session where
sid=8);
```

4）根据 pid 查看 sql 语句:

```
select sql_text from v$sql
where address in
```

(select sql_address from v$session

where sid in

(select sid from v$session where paddr in (select addr from v$process where spid=&pid)));

==========

SQL-47 =====查询有哪些数据库实例在运行

==========

select inst_name from v$active_instances;

==========

SQL-48 =====查看表是否是分区表

==========

select TABLE_NAME,PARTITIONED from user_tables where TABLE_NAME='LOCALUSAGE';

TABLE_NAME PAR

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

LOCALUSAGE YES

==========

SQL-49 =====查看分区表的分区名和相应的表空间名

==========

select TABLE_NAME, PARTITION_NAME,TABLESPACE_NAME from user_tab_partitions where table_name like '%USAGE%';

==========

SQL-50 =====查看索引是否是分区索引

==========

步骤 1 查询是否是分区索引**

SELECT INDEX_NAME, TABLE_NAME, STATUS, PARTITIONED FROM

USER_INDEXES WHERE TABLE_NAME LIKE '%USAGE';

步骤 2 如果返回的 PATITIONED 为 YES，请再执行如下语句来查询分区索引的类型**

SELECT index_name,table_name,locality FROM user_part_indexes;

==========

SQL-51 =====Dual 表的用法，常用在没有目标表的 Select 中

==========

查看系统时间**

select to_char(sysdate,'yy-mm-dd hh24:mi:ss') shijian from dual;

==========

SQL-52 =====查看索引段中 extent 的数量

==========

select segment_name,count(*) from dba_extents

where segment_type='INDEX' and owner='SCOTT' group by segment_name;

==========

SQL-53 =====查看系统表空间中的非管理员索引

==========

SQL>select count(*) from dba_indexes where tablespace_name='SYSTEM' and owner NOT IN('SYS','SYSTEM');

==========

SQL-54 =====查看 system 表空间内的索引的扩展情况

==========

SELECT SUBSTR(segment_name,1,20) "SEGMENT NAME",bytes, COUNT(bytes)

FROM dba_extents WHERE segment_name IN(SELECT index_name FROM dba_indexes

WHERE tablespace_name = 'SYSTEM') GROUP BY segment_name,bytes ORDER

BY segment_name;

==========

SQL-55 =====查看表空间数据文件的读写性能

==========

SQL>Select name,phyrds,phywrts,avgiotim,miniotim,maxiowtm,maxiortm from v$filestat,v$datafile where v$filestat.file#=v$datafile.file#;

SQL>Select fs.name name,f.phyrds,f.phyblkrd,f.phywrts,f.phyblkwrt ,f.readtim,f.writetim from v$filestat f, v$datafile fs where f.file# = fs.file# order by fs.name;

(注意：如果 phyblkrd 与 phyrds 很接近的话，则表明这个表空间中存在全表扫描的表，这些表需要调整索引或优化 SQL 语句)

==========

SQL-56 =====转换表空间为 local 方式

==========

SQL> exec sys.dbms_space_admin.tablespace_migrate_to_local('TBS_TEST') ;

==========

SQL-57 =====查看一下哪个用户在用临时段

==========

SELECT username,sid,serial#,sql_address,machine,program,tablespace,segtype, contents FROM v$session se,v$sort_usage su WHERE se.saddr=su.session_addr;

==========

SQL-58 =====查看占 io 较大的正在运行的 session

==========

SELECT se.sid,se.serial#,pr.SPID,se.username,se.status,se.terminal,se.program, se.MODULE,se.sql_address,st.event,st.p1text,si.physical_reads,si.block_changes

FROM v$session se,v$session_wait st,v$sess_io si,v$process pr WHERE st.sid=se.sid AND st.sid=si.sid AND se.PADDR=pr.ADDR AND se.sid>6 AND st.wait_time=0 AND st.event NOT LIKE '%SQL%' ORDER BY physical_reads DESC;

==========

SQL-59 =====查找前十条性能差的 sql

==========

SELECT * FROM(SELECT PARSING_USER_ID EXECUTIONS,SORTS,COMMAND_TYPE,DISK_READS,sql_text FROM v$sqlarea ORDER BY disk_reads DESC) WHERE ROWNUM<10;

==========

SQL-60 =====删除用户下所有表的语句

==========

select 'drop table '||table_name||' cascade constraints;' from user_tables;

==========

SQL-61 =====查看 LOCK 并杀掉会话

==========

```
set linesize 132 pagesize 66
break on Kill on username on terminal
column Kill heading 'Kill String' format a13
column res heading 'Resource Type' format 999
column id1 format 9999990
column id2 format 9999990
column lmode beading 'Lock Held' format a20
column request heading 'Lock Requested' format a20
column serial# format 99999
column username format a10 heading "Username"
```

```
column terminal heading Term format a6
column tab format a35 heading "table Name"
column owner format a9
column Address format a18

select nvl(S.USERNAME,'Internal') username,
                nvl(S.TERMINAL,'None') terminal,
                L.SID||','||S.SERIAL# Kill,
                U1.NAME||','||substr(T1.NAME,1,20) tab,
                decode(L.LMODE, 1,'No Lock',
                                                2,'Row Share',
                                                3,'Row Exclusive',
                                                4,'Share',
                                                5,'Share Row Exclusive',
                                                6,'Exclusive',null) lmode,
                decode(L.REQUEST,1,'No Lock',
                                        2,'Row Share',
                                        3,'Row Exclusive',
                                        4,'Share',
                                        5,'Share Row Exclusive',
                                        6,'Exclusive',null) request
                from V$LOCK             L,
                     V$SESSION S,
                     SYS.USER$ U1,
                     SYS.OBJ$    T1
                where L.SID = S.SID
                and T1.OBJ# = decode(L.ID2,0,L.ID1,L.ID2)
                and U1.USER#= T1.OWNER#
                and S.TYPE != 'BACKGROUND'
                order by 1,2,5;
```

```
--alter system kill session ' , ';

column username format A15
column sid          format 9990 heading SID
column type         format A4
column lmode        format 990    heading 'HELD'
column request    format 990    heading 'REQ'
column id1          format 9999990
column id2                      format 9999990
break on id1 skip 1 dup
spool tfslckwt.lst
select sn.username,
                m.sid,
                m.type,
                DECODE(m.lmode,0,'None',
                                                1,'Null',
                                                2,'Row Share',
                                                3,'Row Excl.',
                                                4,'Share',
                                                5,'S/Row Excl.',
                                                6,'Exclusive',
                                        lmode,ltrim(to_char(lmode,'990'))) lmode,
            DECODE(m.request,0,'None',
                                                        1,'Null',
                                                        2,'Row Share',
                                                        3,'Row Excl.',
                                                        4,'Share',
                                                        5,'S/Row Excl.',
```

```
                                                    6,'Exclusive',

request,ltrim(to_char(m.request,'990'))) request,
                m.id1,
                m.id2
                from v$session sn,
                        v$lock        m
                where (sn.sid = m.sid and m.request!= 0)
                  or      (sn.sid = m.sid and
                            m.request = 0 and lmode != 4 and
                            (id1 ,id2) in (select s.id1,
                                                                    s.id2
                                                            from v$lock s
                            where request != 0 and s.id1 = m.id1 and s.id2 = m.id2)
                          )
                  order by id1,id2,m.request;
                  spool off
                  clear breaks

============================================================
SQL-62 =====识别 IO 竞争和负载平衡
============================================================
col  文件名  format a35
select    df.name    文  件  名  ,fs.phyrds    读  次  数  ,fs.phywrts    写  次  数  ,
(fs.readtim/decode(fs.phyrds,0,-1,fs.phyrds))          读          时          间          ,
(fs.writetim/decode(fs.phywrts,0,-1,fs.phywrts))  写时间
    from v$datafile df, v$filestat fs
  where df.file#=fs.file# order by df.name
/
```

文件名	读次数	写次数	读时间	写时间
C:\ORACLE\ORADATA\ORADB\DR01.DBF	885	883	0	0
C:\ORACLE\ORADATA\ORADB\INDX01.DBF	885	883	0	0
C:\ORACLE\ORADATA\ORADB\OEM_REPOSITORY.ORA	885	883	0	0
C:\ORACLE\ORADATA\ORADB\RBS01.DBF	925	22306	0	0
C:\ORACLE\ORADATA\ORADB\SYSTEM01.DBF	50804	155025	0	0
C:\ORACLE\ORADATA\ORADB\TEMP01.DBF	887	894	0	0
C:\ORACLE\ORADATA\ORADB\TOOLS01.DBF	886	892	0	0
C:\ORACLE\ORADATA\ORADB\USERS01.DBF	885	883	0	0

已选择 8 行。

其中：ORADB 为数据库名，因为本例中数据库使默认安装，没有进行过优化、调整，所以一直在 system 表空间上做操作，导致 system 表空间所在的数据文件 SYSTEM01.DBF 被读写的次数最多，

这也说明了，尽量不要在 system 表空间做与系统无关的操作，应给各个用户建立单独的表空间。

==========

SQL-63 =====查看哪些 session 正在使用哪些回滚段

==========

col 回滚段名 format a10

col SID format 9990

col 用户名 format a10

col 操作程序 format a80

col status format a6 trunc

SELECT r.name 回滚段名, s.sid, s.serial#, s.username 用户名, t.status, t.cr_get, t.phy_io, t.used_ublk, t.noundo, substr(s.program, 1, 78) 操作程序

FROM sys.v_$session s,sys.v_$transaction t,sys.v_$rollname r

WHERE t.addr = s.taddr and t.xidusn = r.usn ORDER BY t.cr_get,t.phy_io;

==========

SQL-64 =====查看 WACOS 表空间下所有的索引

==========

SQL>select 'analyze index '||segment_name||' validate structure;' from dba_segments where tablespace_name='WACOS'and segment_type='INDEX';

==========

SQL-65 =====查看数据文件的 hwm（可以 resize 的最小空间）和文件头大小

==========

SQL>SELECT v1.file_name,v1.file_id,num1 totle_space,num3 free_space,num1-num3 "USED_SPACE(HWM)",nvl(num2,0) data_space,num1-num3-nvl(num2,0) file_head

FROM (SELECT file_name,file_id,SUM(bytes) num1 FROM Dba_Data_Files GROUP BY file_name,file_id) v1,

(SELECT file_id,SUM(bytes) num2 FROM dba_extents GROUP BY file_id) v2,

(SELECT file_id,SUM(BYTES) num3 FROM DBA_FREE_SPACE

GROUP BY file_id) v3

WHERE v1.file_id=v2.file_id(+) AND v1.file_id=v3.file_id(+);

==========

SQL-66 =====查看数据文件大小及头大小

==========

SQL>SELECT v1.file_name,v1.file_id,

num1 totle_space,

num3 free_space,

num1-num3 Used_space,

nvl(num2,0) data_space,

num1-num3-nvl(num2,0) file_head

FROM

(SELECT file_name,file_id,SUM(bytes) num1 FROM Dba_Data_Files GROUP BY file_name,file_id) v1,

(SELECT file_id,SUM(bytes) num2 FROM dba_extents GROUP BY file_id) v2,

(SELECT file_id,SUM(BYTES) num3 FROM DBA_FREE_SPACE GROUP BY file_id) v3

WHERE v1.file_id=v2.file_id(+)

AND v1.file_id=v3.file_id(+);

(运行以上查询，我们可以如下信息：

Totle_pace:该数据文件的总大小，字节为单位

Free_space:该数据文件的剩于大小，字节为单位

Used_space:该数据文件的已用空间，字节为单位

Data_space:该数据文件中段数据占用空间，也就是数据空间，字节为单位

File_Head:该数据文件头部占用空间，字节为单位)

数据库各个表空间增长情况的检查：

SQL>select A.tablespace_name,(1-(A.total)/B.total)*100 used_percent

From (select tablespace_name,sum(bytes) total from dba_free_space group by

tablespace_name) A,(select tablespace_name,sum(bytes) total from dba_data_files group by tablespace_name) B where A.tablespace_name=B.tablespace_name;

SQL>SELECT UPPER(F.TABLESPACE_NAME) "表空间名",
D.TOT_GROOTTE_MB "表空间大小(M)",
D.TOT_GROOTTE_MB - F.TOTAL_BYTES "已使用空间(M)",
TO_CHAR(ROUND((D.TOT_GROOTTE_MB - F.TOTAL_BYTES) / D.TOT_GROOTTE_MB * 100, 2), '990.99') "使用比", F.TOTAL_BYTES "空闲空间(M)",
F.MAX_BYTES "最大块(M)" FROM (SELECT TABLESPACE_NAME,
ROUND(SUM(BYTES) / (1024 * 1024), 2) TOTAL_BYTES,
ROUND(MAX(BYTES) / (1024 * 1024), 2) MAX_BYTES
FROM SYS.DBA_FREE_SPACE GROUP BY TABLESPACE_NAME) F,
(SELECT DD.TABLESPACE_NAME,ROUND(SUM(DD.BYTES) / (1024 * 1024), 2) TOT_GROOTTE_MB FROM SYS.DBA_DATA_FILES DD
GROUP BY DD.TABLESPACE_NAME) D WHERE D.TABLESPACE_NAME = F.TABLESPACE_NAME
ORDER BY 4 DESC;

查看各个表空间占用磁盘情况:

```
SQL>col tablespace_name format a20;
SQL>select  b.file_id  file_ID,
b.tablespace_name  tablespace_name,
b.bytes  Bytes,
(b.bytes-sum(nvl(a.bytes,0)))  used,
 sum(nvl(a.bytes,0))  free,
 sum(nvl(a.bytes,0))/(b.bytes)*100 Percent
      from dba_free_space a,dba_data_files b
      where a.file_id=b.file_id
      group by b.tablespace_name,b.file_id,b.bytes
```

order by b.file_id;

数据库对象下一扩展与表空间的 free 扩展值的检查：

SQL>select a.table_name, a.next_extent, a.tablespace_name
from all_tables a,(select tablespace_name, max(bytes) as big_chunk
from dba_free_space group by tablespace_name) f where f.tablespace_name = a.tablespace_name and a.next_extent > f.big_chunk
union select a.index_name, a.next_extent, a.tablespace_name
from all_indexes a,(select tablespace_name, max(bytes) as big_chunk
from dba_free_space group by tablespace_name) f where f.tablespace_name = a.tablespace_name and a.next_extent > f.big_chunk;

Disk Read 最高的 SQL 语句的获取：

SQL>select sql_text from (select * from v$sqlarea order by disk_reads)
where rownum<=5;

查找前十条性能差的 sql:

SQL>SELECT * FROM (SELECT PARSING_USER_ID
EXECUTIONS,SORTS,COMMAND_TYPE,DISK_READS,
sql_text FROM v$sqlarea ORDER BY disk_reads DESC)
WHERE ROWNUM<10 ;

等待时间最多的 5 个系统等待事件的获取：

SQL>select * from (select * from v$system_event where event not like 'SQL%' order by total_waits desc) where rownum<=5;

查看当前等待事件的会话:

SQL>col username format a10
SQL>set line 120
SQL>col EVENT format a30

SQL>select
SE.Sid,s.Username,SE.Event,se.Total_Waits,SE.Time_Waited,SE.Average_Wait
from v$session S,v$session_event SE where S.Username is not null and SE.Sid=S.Sid
and S.Status='ACTIVE' and SE.Event not like '%SQL*Net%';

SQL>select sid, event, p1, p2, p3, wait_time, seconds_in_wait, state from v$session_wait where event not like '%message%' and event not like 'SQL*Net%' and event not like '%timer%' and event != 'wakeup time manager';

找到与所连接的会话有关的当前等待事件：
SQL>select
SW.Sid,S.Username,SW.Event,SW.Wait_Time,SW.State,SW.Seconds_In_Wait
SEC_IN_WAIT
from v$session S,v$session_wait SW where S.Username is not null and SW.Sid=S.Sid
and SW.event not like '%SQL*Net%' order by SW.Wait_Time Desc;

Oracle 所有回滚段状态的检查：
SQL>select
segment_name,owner,tablespace_name,initial_extent,next_extent,dba_rollback_segs.status from dba_rollback_segs,v$datafile where file_id=file#;

Oracle 回滚段扩展信息的检查:
SQL>col name format a10
SQL>set linesize 140
SQL>select substr(name,1,40)
name,extents,rssize,optsize,aveactive,extends,wraps,shrinks,hwmsize
from v$rollname rn,v$rollstat rs where (rn.usn=rs.usn);

extents:回滚段中的盘区数量。

Rssize:以字节为单位的回滚段的尺寸。

optsize：为 optimal 参数设定的值。

Aveactive:从回滚段中删除盘区时释放的以字节为单位的平均空间的大小。

Extends:系统为回滚段增加的盘区的次数。

Shrinks:系统从回滚段中清除盘区（即回滚段收缩）的次数。回滚段每次清除盘区时，系统可能会从这个回滚段中消除一个或多个盘区。

Hwmsize:回滚段尺寸的上限，即回滚段曾经达到的最大尺寸。

(如果回滚段平均尺寸接近 OPTIMAL 的值，那么说明 OPTIMAL 的值设置正确，如果回滚段动态增长次数或收缩次数很高，那么需要提高 OPTIMAL 的值)

查看回滚段的使用情况，哪个用户正在使用回滚段的资源:

```
SQL>select s.username, u.name from v$transaction t,v$rollstat r,
v$rollname u,v$session s where s.taddr=t.addr and
t.xidusn=r.usn and r.usn=u.usn order by s.username;
```

如何查看一下某个 shared_server 正在忙什么:

```
SQL>SELECT a.username,a.machine,a.program,a.sid,
a.serial#,a.status,c.piece,c.sql_text
FROM v$session a,v$process b,v$sqltext c
WHERE b.spid=13161 AND b.addr=a.paddr
AND a.sql_address=c.address(+) ORDER BY c.piece;
```

数据库共享池性能检查:

```
SQL>Select namespace,gets,gethitratio,pins,pinhitratio,reloads,
Invalidations from v$librarycache where namespace in
('SQLAREA','TABLE/PROCEDURE','BODY','TRIGGER');
```

检查数据重载比率:

```
SQL>select sum(reloads)/sum(pins)*100 "reload ratio" from
v$librarycache;
```

检查数据字典的命中率:

SQL>select 1-sum(getmisses)/sum(gets) "data dictionary hit ratio" from v$rowcache;

(对于 library cache, gethitratio 和 pinhitratio 应该大于 90%,对于数据重载比率,reload ratio 应该小于 1%,对于数据字典的命中率,data dictionary hit ratio 应该大于 85%)

检查共享内存的剩余情况:

SQL>select request_misses, request_failures from v$shared_pool_reserved;

(对于共享内存的剩余情况, request_misses 和 request_failures 应该接近 0)

数据高速缓冲区性能检查:

SQL>select 1-p.value/(b.value+c.value) "db buffer cache hit ratio" from v$sysstat p,v$sysstat b,v$sysstat c where p.name='physical reads' and b.name='db block gets' and c.name='consistent gets';

检查 buffer pool HIT_RATIO 执行

SQL>select name, (physical_reads/(db_block_gets+consistent_gets)) "MISS_HIT_RATIO" FROM v$buffer_pool_statistics WHERE (db_block_gets+ consistent_gets)> 0;

(正常时 db buffer cache hit ratio 应该大于 90%, 正常时 buffer pool MISS_HIT_RATIO 应该小于 10%)

数据库回滚段性能检查:

检查 Ratio 执行

SQL>select sum(waits)* 100 /sum(gets) "Ratio", sum(waits) "Waits", sum(gets) "Gets" from v$rollstat;

检查 count/value 执行:

SQL>select class,count from v$waitstat where class like '%undo%';

SQL>select value from v$sysstat where name='consistent gets';

(两者的 value 值相除)

检查 average_wait 执行:

SQL>select event,total_waits,time_waited,average_wait from v$system_event
where event like '%undo%';

检查 RBS header get ratio 执行:

SQL>select n.name,s.usn,s.wraps, decode(s.waits,0,1,1- s.waits/s.gets)"RBS
header get ratio" from v$rollstat s,v$rollname n where s.usn=n.usn;

(正常时 Ratio 应该小于 1%, count/value 应该小于 0.01%,average_wait 最好为 0，该值越小越好,RBS header get ratio 应该大于 95%)

杀会话的脚本:

SQL>select
A.SID,B.SPID,A.SERIAL#,a.lockwait,A.USERNAME,A.OSUSER,a.logon_time,a.la
st_call_et/3600 LAST_HOUR,A.STATUS,
'orakill '||sid||' '||spid HOST_COMMAND,
'alter system kill session '''||A.sid||','||A.SERIAL#||''' SQL_COMMAND
from v$session A,V$PROCESS B where A.PADDR=B.ADDR AND SID>6;

查看排序段的性能:

SQL>SELECT name, value FROM v$sysstat WHERE name IN ('sorts (memory)',
'sorts (disk)');

查看数据表的参数信息:

SQL>SELECT partition_name, high_value, high_value_length, tablespace_name,pct_free, pct_used, ini_trans, max_trans, initial_extent,next_extent, min_extent, max_extent, pct_increase, FREELISTS,freelist_groups, LOGGING, BUFFER_POOL, num_rows, blocks,empty_blocks, avg_space, chain_cnt, avg_row_len, sample_size,last_analyzed FROM dba_tab_partitions

--WHERE table_name = :tname AND table_owner = :towner

ORDER BY partition_position;

2）根据 v.sid 查看对应连接的资源占用等情况

select n.name,v.value,n.class,n.statistic#

from v$statname n,v$sesstat v where v.sid=18 and v.statistic# = n.statistic# order by n.class, n.statistic#;

3）根据 sid 查看对应连接正在运行的 sql

select /*+ PUSH_SUBQ */ command_type,sql_text,sharable_mem, persistent_mem,runtime_mem,sorts,version_count,

loaded_versions,open_versions,users_opening,executions,

users_executing,loads,first_load_time,invalidations,

parse_calls,disk_reads,buffer_gets,rows_processed,sysdate start_time,sysdate finish_time,'>'|| address sql_address,

'N' status from v$sqlarea where address = (select sql_address from v$session where sid=8);

根据 pid 查看 sql 语句:

SQL>select sql_text from v$sql

where address in

(select sql_address from v$session

where sid in

(select sid from v$session where paddr in (select addr from v$process where spid=&pid)));

oracle 数据库性能监控的 SQL

监控事例的等待

SQL>select event,sum(decode(wait_Time,0,0,1)) "Prev",sum(decode(wait_Time,0,1,0)) "Curr",count(*) "Tot" from v$session_Wait group by event order by 4;

回滚段的争用情况

SQL>select name, waits, gets, waits/gets "Ratio" from v$rollstat a, v$rollname b where a.usn = b.usn;

监控表空间的 I/O 比例

SQL>select df.tablespace_name name,df.file_name "file",f.phyrds pyr,
f.phyblkrd pbr,f.phywrts pyw, f.phyblkwrt pbw from v$filestat f, dba_data_files df where f.file# = df.file_id
order by df.tablespace_name;

监控文件系统的 I/O 比例

SQL>select substr(a.file#,1,2) "#", substr(a.name,1,30) "Name",
a.status,a.bytes,b.phyrds,b.phywrts from v$datafile a, v$filestat b
where a.file# = b.file#;

在某个用户下找所有的索引

SQL>select user_indexes.table_name, user_indexes.index_name,uniqueness, column_name from user_ind_columns, user_indexes where user_ind_columns.index_name = user_indexes.index_name
and user_ind_columns.table_name = user_indexes.table_name
order by user_indexes.table_type, user_indexes.table_name,

user_indexes.index_name, column_position;

监控 SGA 的命中率

SQL>select a.value + b.value "logical_reads", c.value "phys_reads",
round(100 * ((a.value+b.value)-c.value) / (a.value+b.value)) "BUFFER HIT RATIO"
from v$sysstat a, v$sysstat b, v$sysstat c where a.statistic# = 38 and b.statistic# = 39
and c.statistic# = 40;

监控 SGA 中字典缓冲区的命中率

SQL>select parameter, gets,Getmisses , getmisses/(gets+getmisses)*100 "miss ratio",(1-(sum(getmisses)/ (sum(gets)+sum(getmisses))))*100 "Hit ratio" from v$rowcache where gets+getmisses <>0 group by parameter, gets, getmisses;

监控 SGA 中共享缓存区的命中率，应该小于 1%

SQL>select sum(pins) "Total Pins", sum(reloads) "Total Reloads",
sum(reloads)/sum(pins) *100 libcache from v$librarycache;

SQL>select sum(pinhits-reloads)/sum(pins) "hit radio",sum(reloads)/sum(pins) "reload percent" from v$librarycache;

显示所有数据库对象的类别和大小

SQL>select count(name) num_instances ,type ,sum(source_size) source_size,sum(parsed_size) parsed_size ,sum(code_size) code_size ,sum(error_size) error_size,sum(source_size) +sum(parsed_size) +sum(code_size) +sum(error_size) size_required from dba_object_size group by type order by 2;

监控 SGA 中重做日志缓存区的命中率，应该小于 1%

SQL>SELECT name, gets, misses, immediate_gets, immediate_misses,
Decode(gets,0,0,misses/gets*100) ratio1,
Decode(immediate_gets+immediate_misses,0,0,
immediate_misses/(immediate_gets+immediate_misses)*100) ratio2
FROM v$latch WHERE name IN ('redo allocation', 'redo copy');

监控内存和硬盘的排序比率，最好使它小于 .10，增加 sort_area_size

SQL>SELECT name, value FROM v$sysstat WHERE name IN ('sorts (memory)', 'sorts (disk)');

监控当前数据库谁在运行什么 SQL 语句

SQL>SELECT osuser, username, sql_text from v$session a, v$sqltext b

where a.sql_address =b.address order by address, piece;

监控字典缓冲区

SQL>SELECT (SUM(PINS - RELOADS)) / SUM(PINS) "LIB CACHE" FROM V$LIBRARYCACHE;

SQL>SELECT (SUM(GETS - GETMISSES - USAGE - FIXED)) / SUM(GETS) "ROW CACHE" FROM V$ROWCACHE;

SQL>SELECT SUM(PINS) "EXECUTIONS", SUM(RELOADS) "CACHE MISSES WHILE EXECUTING" FROM V$LIBRARYCACHE;(后者除以前者,此比率小于 1%,接近 0%为好)

SQL>SELECT SUM(GETS) "DICTIONARY GETS",SUM(GETMISSES) "DICTIONARY CACHE GET MISSES" FROM V$ROWCACHE;

查找 ORACLE 字符集

SQL>select * from sys.props$ where name='NLS_CHARACTERSET';

监控 MTS

SQL>select busy/(busy+idle) "shared servers busy" from v$dispatcher;

(此值大于 0.5 时，参数需加大)

SQL>select sum(wait)/sum(totalq) "dispatcher waits" from v$queue where type='dispatcher';

SQL>select count(*) from v$dispatcher;

SQL>select servers_highwater from v$mts;

(servers_highwater 接近 mts_max_servers 时，参数需加大)

碎片程度

SQL>select tablespace_name,count(tablespace_name) from dba_free_space group by tablespace_name having count(tablespace_name)>10;

SQL>alter tablespace name coalesce;

SQL>alter table name deallocate unused;

SQL>create or replace view ts_blocks_v as

select tablespace_name,block_id,bytes,blocks,'free space' segment_name from

dba_free_space

union all

select tablespace_name,block_id,bytes,blocks,segment_name from dba_extents;

select * from ts_blocks_v;

SQL>select tablespace_name,sum(bytes),max(bytes),count(block_id) from dba_free_space group by tablespace_name;

查看碎片程度高的表

SQL>SELECT segment_name table_name,COUNT(*) extents

FROM dba_segments WHERE owner NOT IN ('SYS', 'SYSTEM') GROUP BY segment_name HAVING COUNT(*)=(SELECT MAX(COUNT(*)) FROM dba_segments GROUP BY segment_name);

17. 表、索引的存储情况检查

SQL>select segment_name,sum(bytes),count(*) ext_quan from dba_extents where tablespace_name='&tablespace_name' and segment_type='TABLE' group by tablespace_name,segment_name;

SQL>select segment_name,count(*) from dba_extents where segment_type='INDEX' and owner='&owner' group by segment_name;

18、找使用 CPU 多的用户 session

SQL>select a.sid,spid,status,substr(a.program,1,40) prog,a.terminal,osuser,value/60/100 value from v$session a,v$process b,v$sesstat c where c.statistic#=12 and c.sid=a.sid and a.paddr=b.addr order by value desc;

(12 是 cpu used by this session)

表空间统计

A、 脚本说明：

这是我最常用的一个脚本，用它可以显示出数据库中所有表空间的状态，如表空间的大小、已使用空间、使用的百分比、空闲空间数及现在表空间的最大块是多大。

B、脚本原文:

```
SELECT upper(f.tablespace_name) "表空间名",
       d.Tot_grootte_Mb "表空间大小(M)",
       d.Tot_grootte_Mb - f.total_bytes "已使用空间(M)",
       to_char(round((d.Tot_grootte_Mb - f.total_bytes) / d.Tot_grootte_Mb * 100,2),'990.99') "使用比",
       f.total_bytes "空闲空间(M)",
       f.max_bytes "最大块(M)"
  FROM
    (SELECT tablespace_name,
            round(SUM(bytes)/(1024*1024),2) total_bytes,
            round(MAX(bytes)/(1024*1024),2) max_bytes
       FROM sys.dba_free_space
      GROUP BY tablespace_name) f,
    (SELECT dd.tablespace_name, round(SUM(dd.bytes)/(1024*1024),2) Tot_grootte_Mb
       FROM sys.dba_data_files dd
       GROUP BY dd.tablespace_name) d
WHERE d.tablespace_name = f.tablespace_name
ORDER BY 4 DESC;
```

查看无法扩展的段

A、 脚本说明：

ORACLE 对一个段比如表段或索引无法扩展时，取决的并不是表空间中剩余的空间是多少，而是取于这些剩余空间中最大的块是否够表比索引的“NEXT”值大，所以有时一个表空间剩余几个 G 的空闲空间，在你使用时 ORACLE 还是提示某个表或索引无法扩展，就是由于这一点，这时说明空间的碎片太多了。这个脚本是找出无法扩展的段的一些信息。

B、脚本原文：

```
SELECT segment_name,
       segment_type,
```

```
owner,
a.tablespace_name "tablespacename",
initial_extent/1024 "inital_extent(K)",
next_extent/1024 "next_extent(K)",
pct_increase,
b.bytes/1024 "tablespace max free space(K)",
b.sum_bytes/1024 "tablespace total free space(K)"
  FROM dba_segments a,
       (SELECT  tablespace_name,MAX(bytes)  bytes,SUM(bytes)  sum_bytes
FROM dba_free_space GROUP BY tablespace_name) b
 WHERE a.tablespace_name=b.tablespace_name
   AND next_extent>b.bytes
 ORDER BY 4,3,1;
```

查看段(表段、索引段)所使用空间的大小

A、 脚本说明：

有时你可能想知道一个表或一个索引占用多少 M 的空间，这个脚本就是满足你的要求的，把<>中的内容替换一下就可以了。

B、脚本原文：

```
SELECT owner,
                segment_name,
                SUM(bytes)/1024/1024
    FROM dba_segments
   WHERE owner=
         And segment_name=
 GROUP BY owner,segment_name
 ORDER BY 3 DESC;
```

查看数据库中的表锁

A、 脚本说明：

这方面的语句的样式是很多的，各式一样，不过我认为这个是最实用的，不信你就用一下，无需多说，锁是每个 DBA 一定都涉及过的内容，当你相知道某个表被哪个 session 锁定了，你就用到了这个脚本。

B、脚本原文：

```
SELECT A.OWNER,
                A.OBJECT_NAME,
                B.XIDUSN,
               B.XIDSLOT,
               B.XIDSQN,
               B.SESSION_ID,
               B.ORACLE_USERNAME,
               B.OS_USER_NAME,
               B.PROCESS,
               B.LOCKED_MODE,
               C.MACHINE,
               C.STATUS,
               C.SERVER,
               C.SID,
               C.SERIAL#,
               C.PROGRAM
  FROM ALL_OBJECTS A,
       V$LOCKED_OBJECT B,
       SYS.GV_$SESSION C
WHERE ( A.OBJECT_ID = B.OBJECT_ID )
   AND (B.PROCESS = C.PROCESS )
--   AND
ORDER BY 1,2;
```

处理存储过程被锁

A、 脚本说明：

实际过程中可能你要重新编译某个存储过程理总是处于等待状态，最后会报无法锁定对象，这时你就可以用这个脚本找到锁定过程的那个 sid，需要注意的是查 v$access 这个视图本来就很慢，需要一些耐心。

B、脚本原文：

SELECT * FROM V$ACCESS WHERE owner=<object owner> And object=<procedure name>;