V$性能视图学习大全

http://www.Database8.com

数据库吧整理

高级 DBA 经常告诉刚刚入行的 DBA，在 Oracle 6 年代，他们曾经将每一张 V$视图烂熟于心。在 Oracle 6 中，仅仅只有 23 个 V$视图，那时侯的 DBA 可以很轻松地使用它们。而在 Oracle 9i 中，有 259 个 V$视图以及近 400 个 X$表；现在 Oracle 10gR2 (10.2.0.1.0) 有 372 个 V$视图和 613 个 X$表。

几乎所有的出色调整或者 DBA 产品都有一个共性。它们中的大多数都是通过访问 V$视图来获取从数据库、单个查询、或者单个用户检索出来的内部信息。通过 Joe Trezzo 和其他 V$宗师们大量的介绍，访问 V$视图已经变得越来越普遍。只有在看过 V$视图之后，您才能体会之前的欠缺。V$视图可以全面、准确地展示 Oracle 数据库的核心信息。它是将普通水平的管理人员变为 DBA 专家的纽带。

第 13 章将更全面地介绍 X$表，该表是 V$视图的底层部分。附录 B 和 C 提供了 V$视图的相关信息，以及 X$表的创建脚本。遗憾的是，由于篇幅所限，我不能展示每一个 V$脚本，我也不想重复其他章节已经深入讨论过的内容。请查看我们的网站(www.tusc.com)来获得最新的可使用的 V$脚本。

本章主要内容：

- 创建 V$视图并设置其访问权限
- 获得所有 V$视图的列表
- 获得组成 V$视图的 X$脚本的列表
- 检查组成 DBA_视图的底层对象
- 查询 V$DATABASE，以获得数据库的创建时间和归档信息
- 了解自动工作量仓库（Automatic Workload Repository，简称 AWR)
- 查询 V$LICENSE，以查看许可限制和警告设置
- 访问 V$OPTIONS，以查看所有已经安装的选项
- 查询 V$SGA 来分配 Oracle 的基本内存
- 查询 V$SGSSTAT 来详细分配 Oracle 的内存
- 在 V$PARAMETER 中查找 init.ora 的设置
- 测定数据的命中率(V$SYSSTAT)
- 测定数据字典的命中率(V$ROWCACHE)
- 测定共享 SQL 和 PL/SQL 的命中率(V$LIBRARYCACHE)
- 识别哪个对象需要固定，以及是否有连续的空闲内存(V$DB_OBJECT_CACHE)
- 通过访问 V$SQLAREA、V$SQLTEXT、V$SESSION 以及 V$SESS_IO 来查找有问题的查询
- 检查用户的当前操作及其所使用的资源
- 识别锁定问题并关闭相应的会话
- 查找使用多会话的用户

- 使用视图 V$DATAFILE、V$FILESTAT 以及 DBA_DATA_FILES 来平衡 I/O
- 检查确认是否有足够的空闲列表
- 检查角色和特权设置
- 使用 V$SESSION、V$SESSION_WAIT、V$SESSION_EVENT、V$SESSION _WAIT _CLASS、V$SESSION_WAIT_HISTORY、V$SYSTEM_EVENT 和 V$SYSTEM_ WAIT_CLASS 查找等待
- 按所符合的类别使用表分组 V$视图

12.1 V$视图的创建和访问

V$视图是由 catalog.sql 脚本创建的。在 Oracle 10g 中，有将近 372 个 V$视图。实际的数量随版本和平台的不同而不同。下面是从 Oracle 6 到 Oracle 10gR2 V$视图和 X$表的具体数目的变化：

版　　本	V$ 视 图	X$ 表
6	23	(?)
7.1	72	126

(续表)

版　　本	V$ 视 图	X$ 表
8.0	132	200
8.1	185	271
9.0	227	352
9.2	259	394
10.1	340	543
10.2	372	613

创建时均以 v_$作为这些视图的前缀。catldr.sql 脚本创建了两张视图，用于 SQL*Loader 的直接加载的统计信息。每个 V$视图的底层视图定义(从技术角度讲，这些视图从没有被创建，它们的定义只是以二进制形式硬编码)可以通过名为 V$FIXED_VIEW_DEFINITION 的 V$视图查看。视图是通过选取一个或多个 X$表中的信息来创建的。系统为每一个 v_$视图创建了一个可以允许用户访问的视图。用户不能访问实际的 v$视图(他们实际上是访问 v_$视图；v$对象只对 SYS 用户可见)，所以，该方法通过在一个视图上创建另一个视图的方法提供了对这些视图的访问。然后，视图的前缀改为了 V$。最后，因为 SYS 用户拥有这些表，每个视图上就创建了一个公共同名视图。下面的程序清单展示了一个用 catalog.sql 创建 V$视图的示例

```
create or replace view gv_$datafile as
select *
from    gv$datafile;
create or replace public synonym gv$datafile for gv_$datafile;
```

下文详细描述了整个事件的完整步骤：

(1) 当创建数据库时，根据 X$表创建 GV$视图的定义：

```
create or replace view gv$fixed_table as
```

```
select inst_id,kqftanam, kqftaobj, 'TABLE', indx
from    X$kqfta
union all
select inst_id,kqfvinam, kqfviobj, 'VIEW', 65537
from    X$kqfvi
union all
select inst_id,kqfdtnam, kqfdtobj, 'TABLE', 65537
from    X$kqfdt;
```

(2) 执行版本特定的目录脚本：

```
SQL> @catalog
```

(3) 根据 V$视图创建 v_$视图：

```
create or replace view v_$fixed_table
as
select *
from v$fixed_table;
```

(4) 再根据 v_$视图创建新的 V$同名视图：

```
create or replace public synonym v$fixed_table for v_$fixed_table;
```

技巧：

SYSTEM 访问的 V$视图实际上是指向 v_$视图的同名视图，而 v_$视图是以根据 X$表创建的原始 V$视图为基础而建立的视图(最好将上面的话多读几遍)。

这些视图上唯一可以执行的操作就是 SELECT。为了用户能够访问 V$视图，必须授于用户访问底层的 v_$视图的权限。不能授于用户访问 V$视图的权限(即使是 SYS 用户)：

```
connect      sys/change_on_install as sysdba
Grant select on v$fixed_table to richn;
ORA-02030: can only select from fixed tables/views.
```

尽管在尝试授权访问 V$FIXED_TABLE 时返回的错误消息反映了不正确的信息（紧接前面的代码），但授权仍然无法实现。然而，可以授权访问 V$视图底层的 v_$视图。

要与 SYS 超级用户连接，请使用下面的语句：

```
Connect sys/change_on_install as sysdba
Connected.
```

要向指定用户授于访问底层视图的权限，请使用下列语句：

```
grant select on v_$fixed_table to richn;
Grant succeeded.
```

要以指定的用户身份连接，请使用下列语句：

```
conn richn/tusc
Connected.
```

使用下列的语句来访问 V$FIXED_TABLE 视图，它是根据 V_$FIXED_TABLE 创建的同名视图：

```
select  count(*)
from    v$fixed_table;
COUNT(*)
--------
1224
```

尽管已经过授权，仍然不能访问 v_$fixed_table：

```
select count(*)
from v_$fixed_table;
ORA-00942: table or view does not exist.
```

如果加上前缀 SYS，就可以访问 v_$fixed_table 了：

```
conn    richn/tusc
select  count(*)
from    SYS.v_$fixed_table;
COUNT(*)
--------
1224
```

为了避免混淆，最好是授于对 v_$表的访问权限，并通知 DBA 用户可访问 V$视图。通过这个方法 ，就可以授权他人访问 V$视图的信息，而不用向他们提供 SYS 或者 SYSTEM 账户的密码。关键是对他 SYS 所有的原始 v_$视图授予 SELECT 权限。

技巧：

当其他的 DBA 需要访问 V$视图的信息，但是没有 SYS 或 SYSTEM 密码时，可以授予他们访问 v_$视图的权限。然后这些用户就可以访问与 v_$视图具有公共同名的 V$视图。然而，所写的脚本必须直接查询 SYS.V_$视图，以避免重引用公共同名视图造成的性能损失。

警告：

应该仅在需要的时候才授于非 DBA 用户访问 V$视图的权限，并且要小心谨慎。记住，查询 V$视图会造成性能损失，并且您的环境越复杂，这些损失就越多。

12.1.1 获得所有 V$视图的数量和列表

要获得某个版本的 Oracle 的所有 V$视图的数量，可以查询 V$FIXED_TABLE 视图。即使是同一个版本，V$视图的数量也会变化。下面的示例显示了针对 Oracle 10g 的 V$视图查询。每一个新版本Oracle中的V$视图的数量一直在不断膨胀。下面程序清单展示了获得V$视图数量的查询：

```
select count(*)
```

```
from    v$fixed_table
where   name like 'V%';
COUNT(*)
--------
372
```

许多 V$视图仍然没有记入技术资料文档。随着视图数据的增长，在 Oracle 中探索数据的方法变得越来越多。在 Oracle 8 中，引入了 GV$视图。GV$视图(全局 V$)与 V$视图基本一样，只是附加了一个实例 ID 列。

下面的程序清单给出了 GV$视图的部分列表(仅是部分列表；完整的列表见附录 B)。

```
select    name
from      v$fixed_table
where     name like 'GV%'
order by name;
NAME
--------------------------------
GV$ACCESS
GV$ACTIVE_INSTANCES
GV$ACTIVE_SERVICES
GV$ACTIVE_SESSION_HISTORY
GV$ACTIVE_SESS_POOL_MTH
GV$ADVISOR_PROGRESS
GV$ALERT_TYPES
GV$AQ1
GV$ARCHIVE
GV$ARCHIVED_LOG
GV$ARCHIVE_DEST
GV$ARCHIVE_DEST_STATUS
GV$ARCHIVE_GAP
GV$ARCHIVE_PROCESSES
...
```

技巧：

查询V$FIXED_TABLE视图可以获得数据库中所有的 GV$视图和 V$视图。GV$视图和V$视图几乎完全一样，但实例 ID 列包含了一个标识符。

12.1.2 查找用于创建 V$视图的 X$表

为了理解 V$视图的信息是从何处得到的，可以查询底层的 X$表(查阅第 13 章了解更详细的信息)。有时候，查询底层的 X$表可能更有利，因为 V$视图通常根据多个 X$表连接得到。X$表非常隐秘，因为它和 Oracle 数据字典的底层表结构很相似。Oracle 在 SGA 中创建 V$视图，使用户可以以一种更便于阅读的格式来检查 X$表中存储的信息。实际上，当在 V$视图上执行 SELECT 操作时，SELECT 操作负责从 SGA 中提取信息—— 更明确地说，是从 X$表中提取信息。

在了解 SELECT 所操作的底层 V$视图后，您就有能力创建自定义视图；直接复制 Select 语句底层的 V$视图，然后修改或者创建一个针对 X$表的新定制的 SELECT 语句。这个技术可创建出更具选择性和更加优化的查询。下面的程序清单用于访问对底层 X$表的查询。为了获得组成 V$视图的 X$表的列表，必须访问 V$FIXED_TABLE_DEFINITION 视图(输出结果已经过格式设置，更便于阅读)。

```
select *
from    v$fixed_view_definition
where    view_name = 'GV$FIXED_TABLE';
```

输出结果

```
VIEW_NAME       VIEW_DEFINITION
--------------  ----------------------------------------------
GV$FIXED_TABLE select inst_id,kqftanam, kqftaobj, 'TABLE', indx
from    X$kqfta
union all
select inst_id,kqfvinam, kqfviobj, 'VIEW', 65537
from    X$kqfvi
union all
select inst_id,kqfdtnam, kqfdtobj, 'TABLE', 65537
from    X$kqfdt
```

技巧：

访问V$FIXED_TABLE_DEFINITION视图可以获得组成 V$视图的底层 X$表的所有信息。

同样需要注意的是，在 Oracle 8 中，底层的 X$表存在索引，以使在 V$视图上执行的查询可以更快地执行。可以通过 V$INDEXED_FIXED_COLUMN 视图来查看在底层 X$表上的索引信息(请查阅第 13 章以了解更详细的信息)。

12.1.3 查找组成 DBA_视图的底层对象

有些人认为 DBA_视图也是从 X$表和/或者 V$视图得到的。它们实际上是从 Oracle 底层数据库的表中得到的(当然也有些是通过访问 X$表得到的)。下面的程序清单通过访问 DBA_VIEWS 来查看组成 DBA_视图的对象。

注意：

可能需要将长度设置为 2000000 来查看所有的输出结果。

```
select text
from   dba_views
where  view_name='DBA_IND_PARTITIONS' ;
TEXT
--------------------------------------------------------------------------------
select u.name, io.name, 'NO', io.subname, 0, ip.hiboundval, ip.hiboundlen
SQL> set long 2000000
(RUN IT AGAIN)
select text
from   dba_views
where view_name='DBA_IND_PARTITIONS';
TEXT
--------------------------------------------------------------------------------
select u.name, io.name, 'NO', io.subname, 0, ip.hiboundval,ip.hiboundlen,
ip.part#,decode(bitand(ip.flags, 1), 1, 'UNUSABLE', 'USABLE'), ts.name,
ip.pctfree$,ip.initrans, ip.maxtrans, s.iniexts * ts.blocksize,
decode(bitand(ts.flags, 3), 1, to_number(NULL), s.extsize *
ts.blocksize),
s.minexts,s.maxexts,decode(bitand(ts.flags,3),1,to_number(NULL),s
.extpct),
decode(bitand(ts.flags, 32), 32, to_number(NULL),
decode(s.lists, 0, 1, s.lists)),
decode(bitand(ts.flags, 32), 32, to_number(NULL),
decode(s.groups, 0, 1, s.groups)),
decode(mod(trunc(ip.flags / 4), 2), 0, 'YES', 'NO'),
decode(bitand(ip.flags, 1024), 0, 'DISABLED', 1024, 'ENABLED', null),
ip.blevel, ip.leafcnt, ip.distkey, ip.lblkkey, ip.dblkkey,
ip.clufac, ip.rowcnt, ip.samplesize, ip.analyzetime,
decode(s.cachehint, 0, 'DEFAULT', 1, 'KEEP', 2, 'RECYCLE', NULL),
```

```
decode(bitand(ip.flags, 8), 0, 'NO', 'YES'), ip.pctthres$,
decode(bitand(ip.flags, 16), 0, 'NO', 'YES'),'',''
from   obj$ io, indpart$ ip, ts$ ts, sys.seg$ s, user$ u
where  io.obj# = ip.obj# and ts.ts# = ip.ts# and ip.file#=s.file# and
ip.block#=s.block# and ip.ts#=s.ts# and io.owner# = u.user#
union all
select u.name, io.name, 'YES', io.subname, icp.subpartcnt,
icp.hiboundval, icp.hiboundlen, icp.part#, 'N/A', ts.name,
icp.defpctfree, icp.definitrans, icp.defmaxtrans,
icp.definiexts, icp.defextsize, icp.defminexts, icp.defmaxexts,
icp.defextpct, icp.deflists, icp.defgroups,
decode(icp.deflogging, 0, 'NONE', 1, 'YES', 2, 'NO', 'UNKNOWN'),
'N/A', icp.blevel, icp.leafcnt, icp.distkey, icp.lblkkey, icp.dblkkey,
icp.clufac, icp.rowcnt, icp.samplesize, icp.analyzetime,
decode(icp.defbufpool, 0, 'DEFAULT', 1, 'KEEP', 2, 'RECYCLE', NULL),
decode(bitand(icp.flags, 8), 0, 'NO', 'YES'), TO_NUMBER(NULL),
decode(bitand(icp.flags, 16), 0, 'NO', 'YES'),'',''
from   obj$ io, indcompart$ icp, ts$ ts, user$ u
where  io.obj# = icp.obj# and icp.defts# = ts.ts# (+) and u.user# = io.owner#
union all
select u.name, io.name, 'NO', io.subname, 0,
ip.hiboundval, ip.hiboundlen, ip.part#,
decode(bitand(ip.flags, 1), 1, 'UNUSABLE',
decode(bitand(ip.flags, 4096), 4096, 'INPROGRS', 'USABLE')),
null, ip.pctfree$, ip.initrans, ip.maxtrans,
0, 0, 0, 0, 0, 0, 0,
decode(mod(trunc(ip.flags / 4), 2), 0, 'YES', 'NO'),
decode(bitand(ip.flags, 1024), 0, 'DISABLED', 1024, 'ENABLED', null),
ip.blevel, ip.leafcnt, ip.distkey, ip.lblkkey, ip.dblkkey,
ip.clufac, ip.rowcnt, ip.samplesize, ip.analyzetime,
'DEFAULT',
decode(bitand(ip.flags, 8), 0, 'NO', 'YES'), ip.pctthres$,
decode(bitand(ip.flags, 16), 0, 'NO', 'YES'),
decode(i.type#,
9, decode(bitand(ip.flags, 8192), 8192, 'FAILED', 'VALID'),''),
ipp.parameters
from   obj$ io, indpartv$ ip, user$ u, ind$ i, indpart_param$ ipp, tab$ t
```

```
where io.obj# = ip.obj# and io.owner# = u.user# and
ip.bo# = i.obj# and ip.obj# = ipp.obj# and i.bo# = t.obj# and
bitand(t.trigflag, 1073741824) != 1073741824
and io.namespace = 4 and io.remoteowner IS NULL and iolinkname IS NULL
```

切勿修改底层的对象；许多 DBA 就是因为修改了这些对象而导致他们的数据库崩溃。不要执行以下代码，但要注意它是可以执行的：

```
Connect sys/change_on_install as sysdba
Connected.
DELETE FROM OBJAUTH$; -- Don’t do this! If you commit this, your database is over!
13923 rows deleted.
Rollback;
Rollback complete.
```

技巧：

DBA_视图不是从X$表或者V$视图派生的。事实上，可以从 obj$中删除行数据这一特性尽量不要以 SYS 超级用户的身份执行类似操作。

12.1.4 使用有帮助的 V$脚本

本章的剩余部分将致力于介绍有助于分析 Oracle 数据库的不同方面的脚本。大部分脚本是动态的，并且能够针对需要分析的数据库的相应领域提供有价值的内部信息，以确定在一个时间点上是否有资源争用现象。通常情况下，DBA 立即执行一些操作，通过调整查询或者增加 init.ora 参数值来减少未来资源争用。撤消特定查询用户的访问权限，或者通过配置文件来限制其占用的系统资源，也可作为一个紧急的处理措施。下文的三个部分包括了可以检索以下信息的脚本：

- 基本的数据库信息
- 有关自动工作量仓库（AWR）的信息
- 基本的许可信息
- 数据库中已安装的数据库选项

1. 基本的数据库信息

获得实例的基本信息通常就像登录 SQL*Plus 一样简单，因为所有的信息都显示在标题栏(banner)上。如果想查看完整的标题栏的题头，可以访问 V$VERSION 视图来显示标题栏。下面的程序清单显示了一种快速的方法，以查看您使用的数据库版本信息和其他相关信息：

版本信息：

```
SQL> select * from v$version;
BANNER
```

```
----------------------------------------------------------------
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Prod
PL/SQL Release 10.2.0.1.0 - Production
CORE    10.2.0.1.0      Production
TNS for 32-bit Windows: Version 10.2.0.1.0 - Production
NLSRTL Version 10.2.0.1.0 - Production
数据库信息：
select name, created, log_mode
from v$database;
NAME        CREATED       LOG_MODE
--------- ---------    ---------------
ORCL        03-DEC-04    ARCHIVELOG
```

访问 V$DATABASE 视图可以获得数据库的基本信息。输出结果中最重要的信息确保您正处在所需要的 ARCHIVELOG 模式中。查看数据库的归档日志状态的另一种方法是使用 ARCHIVE LOG LIST 命令作为 SQL*Plus 中的 SYS 用户。如下所示，输出结果还提供了数据库创建的准确日期。

技巧：

查询 V$VERSION 和 V$DATABASE 视图，可以获得数据库的基本信息，例如版本信息，以确定创建数据库的信息以及基本的归档信息。

2. 自动工作量仓库(AWR) 的基本信息

随着自动工作量仓库(AWR)的出现，需要注意许多地方。默认情况下，仓库用小时填充，保留期是 7 天。对于 AWR 而言，需要知道一些查询(MMON 后台进程用来让 AWR 数据从内存流到磁盘)。关于 AWR 的详细信息请参考第 5 章，它依据 AWR 与 V$相关的视图以及为了调整而使用来自 AWR 的信息。

AWR 使用多少空间？

```
    Select occupant_name, occupant_desc, space_usage_kbytes
from   v$sysaux_occupants
where occupant_name like '%AWR%';
OCCUPANT_NAME
OCCUPANT_DESC                                    SPACE_USAGE_KBYTES
------------- ---------------------------------------- ------------------
SM/AWR        Server Manageability - Automatic Workload Repository 44352
```

系统上最原始的 AWR 信息是什么？

```
select dbms_stats.get_stats_history_availability from dual;
GET_STATS_HISTORY_AVAILABILITY
```

```
---------------------------------------------------------------------------
03-NOV-06 09.30.34.291000000 AM -06:00
什么是 AWR 信息的保留期?
select dbms_stats.get_stats_history_retention from dual;
GET_STATS_HISTORY_RETENTION
---------------------------
31
将 AWR 信息的保留期更改为 15 天?
EXEC dbms_stats.alter_stats_history_retention(15);
GET_STATS_HISTORY_RETENTION
---------------------------
15
```

技巧：

查询V$SYSAUX_OCCUPANTS以确保自动工作量仓库(AWR)没有占用过多空间。使用 Use DBMS_STATS 检查历史和保留。

3. 基本的许可信息

V$LICENSE 视图允许 DBA 监控系统内任何时候有关数据库数量的所有系统活动的数量。它提供了一个记载任何时候最大并发会话数的日志，这就允许公司确保他们获得了正确的许可。当前会话数量与会话警告级别和会话最大级别一起显示。会话警告级别为 0 表示没有设置 init.ora 会话警告参数，所以系统不会显示警告信息。会话最大级别为 0 表示没有设置 init.ora 会话最大参数，所以系统不会限制会话的数量。

应该定期执行脚本，以向 DBA 提供系统一天中实际的会话数量，从而保证正确的许可授权。设置 init.ora 参数 LICENSE_MAX_SESSIONS = 110，将会话数限制为 110。设置 init.ora 参数 LICENSE_SESSIONS_WARNING = 100，系统将向每位在第 100 个会话之后的用户显示警告信息，这样他们就会通知 DBA，系统因遇到问题而关闭(希望能如此)。init.ora 参数 LICENSE_MAX_USERS 用于设置数据库中可以创建的已命名的用户数。在以下程序清单中，该值为 0，所以没有限制。

```
select *
from    v$license;
SESS_MAX   SESS_WARNING   SESS_CURRENT   SESS_HIGHWATER   USERS_MAX
--------   ------------   ------------   --------------   ---------
110             100              44               105             0
(selected columns listed above)
```

技巧：

查询 V$LICENSE 视图，以查看所允许的最大会话数。也可以在接近最大数时设置警告。

4. 数据库中已安装的产品项

下面程序清单中的脚本描述了数据库中已经安装的产品项，以及可用状况。如果您购买了一项产品，该产品没有显示在列表中，则可能是您没有正确地安装它。查询 V$OPTION 视图来检查已经安装的产品，或者登录到 SQL*Plus 来检查已经安装的产品(开源数据库能做到吗？)。

```
select *
from    v$option;
```

为了获得以下的输出结果，需要按 PARAMETER 排序。

输出结果

```
PARAMETER                                VALUE
---------------------------------------- -----------------------
Partitioning                             TRUE
Objects                                  TRUE
Real Application Clusters                FALSE
Advanced replication                     TRUE
Bit-mapped indexes                       TRUE
Connection multiplexing                  TRUE
Connection pooling                       TRUE
Database queuing                         TRUE
Incremental backup and recovery          TRUE
Instead-of triggers                      TRUE
Parallel backup and recovery             TRUE
Parallel execution                       TRUE
Parallel load                            TRUE
Point-in-time tablespace recovery        TRUE
Fine-grained access control              TRUE
Proxy authentication/authorization       TRUE
Change Data Capture                      TRUE
Plan Stability                           TRUE
Online Index Build                       TRUE
Coalesce Index                           TRUE
Managed Standby                          TRUE
```

```
Materialized view rewrite                          TRUE
Materialized view warehouse refresh                TRUE
Database resource manager                          TRUE
Spatial                                            TRUE
Visual Information Retrieval                       TRUE
Export transportable tablespaces                   TRUE
Transparent Application Failover                   TRUE
Fast-Start Fault Recovery                          TRUE
Sample Scan                                        TRUE
Duplexed backups                                   TRUE
Java                                               TRUE
OLAP Window Functions                              TRUE
Block Media Recovery                               TRUE
Fine-grained Auditing                              TRUE
Application Role                                   TRUE
Enterprise User Security                           TRUE
Oracle Data Guard                                  TRUE
Oracle Label Security                              FALSE
OLAP                                               TRUE
Table compression                                  TRUE
Join index                                         TRUE
Trial Recovery                                     TRUE
Data Mining                                        TRUE
Online Redefinition                                TRUE
Streams Capture                                    TRUE
File Mapping                                       TRUE
Block Change Tracking                              TRUE
Flashback Table                                    TRUE
Flashback Database                                 TRUE
Data Mining Scoring Engine                         FALSE
Transparent Data Encryption                        FALSE
Backup Encryption                                  FALSE
Unused Block Compression                           TRUE
54 rows selected.
```

上面的数据库提供了 Partitioning 选项，但它没有安装实时应用群集(Real Application Clusters，RAC)。

技巧：

查询 V$OPTION 视图，可以获取您已安装的 Oracle 产品项。V$VERSION 视图将给出已安装的基本产品项的版本。

12.1.5 内存分配摘要(V$SGA)

如下所示，V$SGA 视图给出了系统的系统全局区(System Global Area，SGA)内存结构的摘要信息。Data Buffers 是在内存中分配给数据的字节数量。它根据 init.ora 的参数 DB_CACHE_SIZE 得到。Redo Buffers 主要是依据 init.ora 参数 LOG_BUFFER 计算得到，每当 COMMIT 命令提交数据时，它被用于缓存已改变的记录并将它们保存到重做日志中。

```
COLUMN value FORMAT 999,999,999,999
select *
from   v$sga;
NAME                                 VALUE
---------------------------- ----------------
Fixed Size                              734,080
Variable Size                       352,321,536
Database Buffers                  2,667,577,344
Redo Buffers                        1,335,296
```

如果使用 SGA_TARGET—— 内部动态调整大小：

```
select (
(select sum(value) from v$sga) -
(select current_size from v$sga_dynamic_free_memory)
) "SGA_TARGET"
from dual;
SGA_TARGET
----------
138412032
```

这个输出结果说明一个相对较大的 SGA 有超过 2.5GB 的缓冲区，该缓冲区缓存包括 DB_CACHE_SIZE、DB_KEEP_CACHE_SIZE 和 DB_RECYCLE_CACHE_SIZE。如第 1 章和第 4 章所述，只将 SGA_TARGET 设置为 3G，将其他参数设置为强制最小值。Variable Size 项中较为突出的部分是共享池(该 SGA 的共享池略大于 200MB)。程序清单中的 SGA 使用了大约 3GB 的实际系统的物理内存。这些信息也可以在 Statspack 报告(参阅第 14 章)中给出，并且 SYS 超级用户使用 SHO SGA 命令也可以显示该信息。

技巧：

访问 V$SGA 视图可以得到系统的物理内存分配的基本概念，包括在 Oracle 中为数据、共享池、large 池、java 池以及日志缓冲区分配的内存。

12.1.6 内存分配的细节(V$SGASTAT)

在 V$视图中，可以查询 V$SGASTAT 视图来提供有关 SGA 更详细的内存分配信息。这个视图提供了 SGA 和内存资源的动态信息(访问数据库时会出现相应变化)。这个语句非常详细地描述了 SGA 的尺寸。在 V$SGA 和 V$SGASTAT 视图中均包含记录 FIXED_SGA、BUFFER_CACHE 和 LOG_BUFFER，且它们在这两个视图中的值均相等。V$SGASTAT 视图中的剩余记录组成了 V$SGA 视图中唯一的其他记录(Variable Size 或 Shared Pool 记录)。

```
Fixed Size (V$SGA)                = fixedsga (V$SGASTAT)
Database Buffers (V$SGA)            = buffercache (V$SGASTAT)
Redo Buffers (V$SGA)                = logbuffer (V$SGASTAT)
Variable Size (V$SGA)             = 39 Other Records (V$SGASTAT)
```

在 Oracle 9.2 中，V$SGASTAT 视图共有 43 个记录，如下程序清单所示

```
select *
from    v$sgastat;
POOL           NAME                              BYTES
-----------  -------------------------  ----------
fixed_sga                                  787828
buffer_cache                             16777216
log_buffer                                 262144
shared pool    KQR L SO                      76800
shared pool    KQR M PO                    1414752
shared pool    KQR M SO                     242688
shared pool    KQR S PO                     157508
shared pool    KQR S SO                        512
shared pool    KTI-UNDO                    1235304
shared pool    sessions                     781324
shared pool    sql area                   11719164
...etc.
597 rows selected
```

在 Statspack 报告中(见第 14 章)也同样可显示这个信息，连同 Statspack 报告持续期间的开始值和结束值一并给出。

技巧：

访问 V$SGASTAT 视图可获取 Oracle SGA 详细的分类列表以及共享池分配中各存储容器的详细信息。

12.1.7 在 V$PARAMETER 中发现 init.ora 的设置

程序清单中的脚本显示了系统中的 init.ora 参数。它还提供了有关参数的信息，确定每一个参数的当前值是否就是默认值(ISDEFAULT=TRUE)。该脚本还显示了该参数是否可以通过 alter session 命令修改，以及是否可以通过 alter system 命令修改(ISSYS_MODIFIABLE=IMMEDIATE)。那些可以被 alter session 和 alte system 命令修改的参数，是通过修改初始化文件然后关闭并重启实例来实现的。程序清单中的示例(来自 Oracle 9.2)显示了可以被 alter 命令修改的部分初始化参数(IMMEDIATE 意味着它可以被修改并立即生效)。注意，您可以使用 ALTER 命令，但对于有些参数而言，例如 o7_dictionary_accessibility，就只能使用 ALTER SYSTEM . . . SCOPE=SPFILE 命令来修改它，然后弹出数据库让它生效。

```
select       name, value, isdefault, isses_modifiable,
issys_modifiable
from          v$parameter
order by     name;
```

查询 V$PARAMETER

```
NAME                              VALUE       ISDEFAULT  ISSES ISSYS_MOD
--------------------------- ---------- --------- ----- ---------
O7_DICTIONARY_ACCESSIBILITY   FALSE       TRUE       FALSE   FALSE
__db_cache_size                 16777216    FALSE      FALSE   IMMEDIATE
__shared_pool_size              58720256    FALSE      FALSE   IMMEDIATE
active_instance_count                       TRUE       FALSE   FALSE
aq_tm_processes                 0           TRUE       FALSE   IMMEDIATE
archive_lag_target              0           TRUE       FALSE   IMMEDIATE
asm_diskgroups                              TRUE       FALSE   IMMEDIATE
asm_diskstring                              TRUE       FALSE   IMMEDIATE
asm_power_limit                 1           TRUE       TRUE    IMMEDIATE
...partial output listing)
```

依赖于版本的列也是可用的。

技巧：

查询 V$PARAMETER 视图，将得到 init.ora 参数的当前值。它还显示了哪些 init.ora 参数已经改动了原始的默认值：ISDEFAULT = FALSE。它还显示了对于一个给定的会话，只能修改哪些参数(当 ISSES_MODIFIABLE = TRUE 时)。最后，它显

示了在不用关闭和重启数据库可以修改哪些参数(当 ISSYS_MODIFIABLE = IMMEDIATE 时)；而 ISSYS_MODIFIABLE = DEFERRED 说明该参数对所有新登录的，但当前未登录会话的用户有效。如果参数 ISSYS _MODIFIABLE =FALSE，则说明该实例必须关闭并重启，才能使设置生效。

12.1.8 测定数据的命中率(V$SYSSTAT)

查询 V$SYSSTAT 视图(如下程序清单所示)可以查看从内存中读取数据的频率。它提供了数据库中设置的数据块缓存区的命中率。这个信息可以帮助您判断系统何时需要更多的数据缓存(DB_CACHE_SIZE)，或者系统的状态何时调整得不佳(二者均将导致较低的命中率)。通常情况下,您应当确保读数据的命中率保持在 95%以上。将系统的命中率从 98%提高到 99%，可能意味着性能提高了 100%(取决于引起磁盘读操作的语句)。

```
select 1-(sum(decode(name, 'physical reads', value,0))/
(sum(decode(name, 'db block gets', value,0)) +
(sum(decode(name, 'consistent gets', value,0)))))
"Read Hit Ratio"
from       v$sysstat;
Read Hit Ratio
--------------
.996558641
```

在 Oracle 10g 中，也可以直接获得 V$SYSMETRIC 中的 AWR 信息:

```
select metric_name, value
from     v$sysmetric
where    metric_name = 'Buffer Cache Hit Ratio';
METRIC_NAME                                                      VALUE
---------------------------------------------------------- ----------
Buffer Cache Hit Ratio                                            100
```

上面程序清单中的命中率很高，但这并不意味着系统已经调整至最佳状态。很高的命中率也可能意味着查询使用了过度的索引。如果这个命中率低于 95%，您可能需要增加 init.ora 参数 DB_CACHE_SIZE，或者调整一些引起磁盘读取操作的查询(仅当这样做是可行的并且确实有效的情况下)。一种例外情况就是分布在不同块中的数据分布的极不平衡。如果不考虑这种可能性，那么命中率低于 90%几乎总意味着系统调整得很糟糕，要么就是某些人不切实际地设计，使每个数据块的数据都极不平衡。(参阅第 4 章，查看有关命中率的其他信息)。

如果需要，也可以使用新的 V$DB_CACHE_ADVICE 视图来帮助改变数据缓存的大小。下面程序清单的查询通过创建一个值的列表来展示较大的数据缓存和较小的数据缓存的不同效果。

```
column buffers_for_estimate format 999,999,999 heading 'Buffers'
```

```
column estd_physical_read_factor format 999.90 heading 'Estd Phys|Read Fact'
column estd_physical_reads format 999,999,999 heading 'Estd Phys| Reads'
SELECT size_for_estimate, buffers_for_estimate,
estd_physical_read_factor, estd_physical_reads
FROM    V$DB_CACHE_ADVICE
WHERE  name = 'DEFAULT'
AND    block_size =
(SELECT value
FROM V$PARAMETER
WHERE name = 'db_block_size')
AND    advice_status = 'ON';
Estd Phys Estd Phys
SIZE_FOR_ESTIMATE       Buffers Read Fact           Reads
----------------- ------------ --------- ------------
4              501        63.59      243,243,371
8            1,002        45.65     174,614,755
12            1,503         2.61        9,965,760
16            2,004         1.00         3,824,900
20            2,505          .76        2,909,026
24            3,006          .57        2,165,817
28            3,507          .41        1,555,860
32            4,008          .33        1,253,696
```

12.1.9 测定数据字典的命中率(V$ROWCACHE)

可以使用 V$ROWCACHE 视图(如程序清单所示)来发现对数据字典的调用是否有效地利用了通过 init.ora 参数 SHARED_POOL_SIZE 分配的内存缓存。这已经在第 4 章详细讨论过。这里的唯一目标就是复习 V$视图的访问方法。如果字典的命中率不高，系统的综合性能将大受影响。

```
select     sum(gets), sum(getmisses),(1 - (sum(getmisses) / (sum(gets)
+ sum(getmisses)))) * 100 HitRate
from       v$rowcache;
SUM(GETS)    SUM(GETMISSES)       HITRATE
--------     -------------     ----------
110673                 670      99.3982558
```

在 Oracle 10g 中，也可以直接获得 V$SYSMETRIC 中的 AWR 信息:

```
select metric_name, value
from     v$sysmetric
```

```
where    metric_name = 'Library Cache Hit Ratio';
METRIC_NAME                                                      VALUE
-----------------------------, ------------------  -----------
Library Cache Hit Ratio                                          99
```

推荐的命中率是 95%或者更高。如果命中率低于这个百分比，说明可能需要增加 init.ora 参数 SHARED_POOL_SIZE。但要记住，在 V$SGASTAT 视图中看到的共享池包括多个部分，而这里仅仅就是其中之一。注意：在大幅度使用公共同名的环境中，字典命中率可能难以超过 75%，即使共享池的尺寸很大。这是因为 Oracle 必须经常检查不存在的对象是否依旧存在。

12.1.10 测定共享 SQL 和 PL/SQL 的命中率(V$LIBRARYCACHE)

访问 V$LIBRARYCACHE 视图可以显示实际使用的语句(SQL 和 PL/SQL)访问内存的情况。如果 init.ora 的参数 SHARED_POOL_SIZE 设置得太小，内存中就没有足够的空间来存储所有的语句。如果共享池的碎片化现象很严重，较大的 PL/SQL 程序就无法加载到共享池中。如果不能有效地重用语句，则扩大共享池可能事与愿违，反而带来更多不利(参阅第 4 章的详细信息)。.

这里包括执行率(固定命中率)和重载命中率。推荐的固定对象的命中率是 95%以上，重载命中率应为 99%以上(低于 1%的重载次数)。如果语句先前已经经过分析，但共享池通常不够大，在分析其他的语句时无法在内存中保存这个语句，则在此时会出现重载。语句的主体被挤出内存(语句头仍然保存下来)；当需要再次使用该语句时，就将重载记录下来，并将语句主体再次加载到内存中。这种情况也会出现在语句的执行计划发生变动时。如果有任何一个命中率低于这些百分比，就说明应该更仔细地分析共享池的分布情况。下面的程序清单显示了如何查询上面讨论的所有信息。

查询 v$librarycache，看看是否重用 SQL ：

```
select     sum(pins) "Executions", sum(pinhits) "Hits",
((sum(pinhits) / sum(pins)) * 100) "PinHitRatio",
sum(reloads) "Misses", ((sum(pins) / (sum(pins)
+ sum(reloads))) * 100) "RelHitRatio"
from        v$librarycache;
Executions        Hits PinHitRatio      Misses RelHitRatio
---------- ---------- ----------- ---------- -----------
7002504    6996247    99.9106462         327    99.9953305
```

查询 v$sql_bind_capture，看看 average binds 是否大于 15 (issue):

```
select sql_id, count(*) bind_count
from     v$sql_bind_capture
where   child_number = 0
```

```
group  by sql_id
having count(*) > 20
order    by count(*);
SQL_ID          BIND_COUNT
------------- ----------
9qgtwh66xg6nz           21
```

查找有问题的 SQL 并修复它：

```
select   sql_text, users_executing, executions, users_opening, buffer_gets
from     v$sqlarea
where    sql_id = '9qgtwh66xg6nz'
order    by buffer_gets;
SQL_TEXT
--------------------------------------------------------------------------------
USERS_EXECUTING  EXECUTIONS  USERS_OPENING  BUFFER_GETS
---------------  ----------  -------------  -----------
update seg$ set type#=:4,blocks=:5,extents=:6,minexts=:7, maxexts=:8,
extsize
=:9,e
xtpct=:10,user#=:11,iniexts=:12,lists=decode(:13, 65535, NULL, :13),groups=
decod
e(:14, 65535, NULL, :14), cachehint=:15, hwmincr=:16, spare1=DECODE(:17,0,
NULL,:
17),scanhint=:18 where ts#=:1 and file#=:2 and block#=:3
0           90                0           690
```

查询 v$sql_bind_capture，看看 average binds 是否大于 15 (issue):

```
select avg(bind_count) AVG_NUM_BINDS from
(select sql_id, count(*) bind_count
from v$sql_bind_capture
where child_number = 0
group by sql_id);
AVG_NUM_BINDS
-------------
3.35471698
```

技巧：

查询V$LIBRARYCACHE视图可以知道从内存中访问 SQL 和 PL/SQL 的频率的信息。固定命中率通常应该是 95%或更高，而重载的次数不应该超过 1%。查询 V$SQL_BIND_CAPTURE 视图，看看每个 SQL 绑定是否太高，是否需要 CURSOR_SHARING。

12.1.11 确定需要固定的 PL/SQL 对象

碎片化现象造成共享池中的可用空间均成为许多零散的片段，而没有足够大的连续空间，这是共享池中的普遍现象。消除共享池错误(参阅第 4 章和第 13 章以了解更多信息)的关键是理解哪些对象会引起问题。一旦知道了会引起潜在问题的 PL/SQL 对象，就可以在数据库启动时固定这个代码(这时共享池是完全连续的)。可以查询 V$DB_OBJECT_CACHE 视图，以决定哪个 PL/SQL 占用空间，而目前尚未固定下来。该查询只显示缓存中的现有语句。下面程序清单中的示例搜索那些所需空间大于 100KB 的对象。

```
select name, sharable_mem
from     v$db_object_cache
where    sharable_mem > 100000
and      type in ('PACKAGE', 'PACKAGE BODY',
'FUNCTION', 'PROCEDURE')
and      kept = 'NO';
NAME                                    SHARABLE_MEM
------------------------------- ------------
MGMT_JOB_ENGINE                          238590
DBMS_STATS_INTERNAL                      220939
STANDARD                                 435820
DBMS_RCVMAN                              354875
WK_CRW                                   183688
DBMS_BACKUP_RESTORE                      258495
DBMS_STATS                               446886
```

技巧：

通过 V$DB_OBJECT_CACHE 视图可以查看尚未固定且所需空间较大，可能会引起潜在问题的对象。

12.1.12 通过 V$SQLAREA 查找有问题的查询

V$SQLAREA 视图提供了一种识别有潜在问题或者需要优化的 SQL 语句的方法，从而可通过减少磁盘的访问来优化数据库的综合性能。disk_reads 表示系统中正在执行的磁盘读取量。它和执行次数结合起来(disk_reads/执行情况)就返回了 SQL 语句，构成每一条语句执行时主要的磁盘访问率。上面程序清单中的 disk_reads 设置为 100000，但它在最终的产品系统(依赖于数据库)中也可以设置得更大或更小，以便仅揭示系统中问题较大的语句。一旦确定问题以后，最上层的语句应当被重新检查并优化，以提高系统的综合性能。一般情况下，问题语句都没有使用索引，或者执行路径限制语句无法使用正确的索引。

下面程序清单中的查询有一个部分可能会造成误导，即 rds_exec_radio。它表示磁盘读操作的次数除以执行过程的数目所得的值。实际上，一条语句可以一次使用 100 个磁盘读操作来读取，然后再被强制清出内存(如果内存空间不足的话)。如果再次读取的话，那么它将再次进行 100 次磁盘读操作，而 rds_exec_radio 就是 100(100+100 次磁盘读操作，再除以 2 次执行过程)。但是，如果第二次读取它时，该语句正在内存中(内存空间足够)，那么磁盘读操作将为 0(第 2 次)，所以 rds_exec_radio 将是 50(100+0 次磁盘读操作，再除以 2 次执行过程)。在结果列表顶部的任何语句都存在问题，并需要被调整——定期调整！

注意：

下面的代码已经经过格式化处理，以方便阅读。

```
select      b.username username, a.disk_reads reads,
a.executions exec, a.disk_reads /decode
(a.executions, 0, 1,a.executions) rds_exec_ratio,
a.command_type, a.sql_text Statement
from        v$sqlarea a, dba_users b
where       a.parsing_user_id = b.user_id
and          a.disk_reads > 100000
order by    a.disk_reads desc;
USERNAME  READS   EXEC RDS_EXEC_RATIO STATEMENT
--------  -------  ----  --------------   --------------------------------
ADHOC1    7281934    1          7281934   select custno, ordno
from     cust, orders
ADHOC5    4230044    4          1057511   select ordno
from     orders
where   trunc(ordno) = 721305
ADHOC1      801715   2           499858 select   custno, ordno
from    cust
where   decode(custno,1,6) = 314159
```

在前面语句中的 DISK_READS 列可以用 BUFFER_GETS 列来代替，以提供关于 SQL 语句的信息，该 SQL 语句拥有的磁盘访问率不高(尽管它们通常是这样)，但拥有的内存访问率很高(高于所期望的值)。这些语句使用了大量的内存来分配给数据(DB _CACHE_SIZE)。问题不在于语句是在内存中执行的(这本身是好事)，而在于该语句独占了大量的内存。许多情况下，问题出在当 SQL 语句应做全表扫描或连接操作时，它却使用了一个索引。这种类型的 SQL 语句也可能牵涉到一个连接操作，该连接会强制使用一个并非所需的索引，或者使用多重索引以及强制合并索引或数据。记住，大部分的系统性能问题是由于糟糕的 SQL 或 PL/SQL 语句。

技巧：

V$SQLAREA 视图可发现有问题的查询(和用户)。

12.1.13 检查用户的当前操作及其使用的资源

将 V$SESSION 和 V$SQLTEXT 连接就可以显示目前每一个会话正在执行的 SQL 语句，如下面的程序清单所示。这在有些时候是极为有用的，例如 DBA 希望查看某一个给定的时间点上系统究竟执行了哪些操作。

```
select    a.sid, a.username, s.sql_text
from      v$session a, v$sqltext s
where     a.sql_address = s.address
and       a.sql_hash_value = s.hash_value
order by  a.username, a.sid, s.piece;
SID USERNAME    SQL_TEXT
--- ---------- ----------------------------------
11 PLSQL_USER update s_employee set salary = 10000
9  SYS          select a.sid, a.username, s.sql_text
9  SYS          from v$session a, v$sqltext
9  SYS          where a.sql_address = s.address
9  SYS          and a.sql_hash_value = s.hash_value
9  SYS          order by a.username, a.sid, s.piece
(...partial output listing)
```

SQL_TEXT 列显示了整个 SQL 语句,但整个语句在 V$SQLTEXT 视图中是作为 VARCHAR2(64)数据类型来存储的，所以跨越了多条记录。PIECE 列用于对语句排序。为了查看每个用户所使用的资源，可以直接使用下面程序清单中的查询。这个语句的目标是显示每个会话的物理磁盘命中率和内存命中率。这就非常容易发现哪些用户执行了大量的物理磁盘和内存读操作。

```
select      a.username, b.block_gets, b.consistent_gets,
b.physical_reads, b.block_changes, b.consistent_changes
from         v$session a, v$sess_io b
where       a.sid = b.sid
order by a.username;
USERNAME  BLOCK_GETS CONSISTENT_GETS PHYSICAL_READS BLOCK_ CHANGES
CONSISTENT_CHANGES
----------  ---------- --------------- -------------- --------------
---------
PLSQL_USER           39                  72                11
53          1
SCOTT                 11                  53                12
 0          0
SYS                    14                 409                26
    0          0
SYSTEM              8340              10197               291                25
58       419
```

技巧：

通过查询 V$SESSION、V$SQLTEXT 和 V$SESS_IO 可发现有问题的用户，并可发现一个给定的时间点上他们在执行什么操作。

12.1.14 查找用户正在访问的对象

一旦发现某些用户或者系统中的查询存在问题，查询 V$ACCESS 可以为您指出有潜在问题的对象(可能缺少索引)。当想修改一个特殊的对象，或者需要知道在一个给定的时间点上谁在使用该对象时，它也非常有帮助，如下面程序清单所示。

```
select a.sid, a.username, b.owner, b.object, b.type
from      v$session a, v$access b
where    a.sid = b.sid;
SID USERNAME OWNER       OBJECT                       TYPE
--- -------- -----       ---------------------    -------
8 SCOTT       SYS        DBMS_APPLICATION_INFO    PACKAGE
9 SYS         SYS        DBMS_APPLICATION_INFO PACKAGE
9 SYS         SYS        X$BH                         TABLE
10   SYSTEM      PUBLIC    V$ACCESS                    SYNONYM
10   SYSTEM      PUBLIC     V$SESSION                  SYNONYM
```

```
10  SYSTEM      SYS       DBMS_APPLICATION_INFO    PACKAGE
10  SYSTEM      SYS        V$ACCESS                    VIEW
10  SYSTEM      SYS        V$SESSION                   VIEW
10  SYSTEM      SYS        V_$ACCESS                   VIEW
```

该脚本显示了所有正被访问的对象，包括同名表、视图、已存储的源代码等。

技巧：

通过查询V$ACCESS视图可查看在给定的时间点上用户所访问的所有对象。这有助于查明有问题的对象，在想修改一个特定的对象时也很有用(查找谁在访问它)。然而，当系统有一个很大的共享池和数百个用户时，这个操作的开销将很大。

获得详细的用户信息

当正在测试一个新的或者升版的应用程序模块以决定其开销时，分析用户的统计数据的方法是极为有用的。当用户遇到性能问题时，它也向用户提供了一个窗口，因为它提供的统计数据涵盖了每个用户的各个方面。此外，在设置配置文件来限制特定用户时，也可将其作为一种指导。下面程序清单中的脚本将统计数据限制为必须有值的数据(b.value != 0)。注意只列出了 IMU，它只存在于 Oracle 10g 之中：

```
select      a.username, c.name, sum(b.value) value
from        v$session a, v$sesstat b, v$statname c
where       a.sid          = b.sid
and         b.statistic# = c.statistic#
and        b.value       != 0
group by name, username;
USERNAME                  NAME                                VALUE
-------------------    -------------------------  ----------

SYS                         DB time                                3690
redo size                          2143640
SYS                         redo size                            98008
user calls                                 28
SYS                         user calls                             337
IMU Flushes                                 1
SYS                        IMU Flushes                              2
IMU commits                                19
SYS                         IMU commits                              1
redo writes                              4443
redo entries                             8728
```

...etc.

12.1.15 使用索引

Oracle 9i 提供了监控索引使用的功能。这个新的视图表示索引是否被引用，但不能反映索引使用的频率。要监控的索引需要单独打开和关闭。可以使用 alter index 命令来初始化监控工作，然后通过对视图 V$OBJECT_USAGE 的查询来实现索引的跟踪。下面的代码提供了 V$OBJECT_USAGE 视图的描述。

```
SQL> desc v$object_usage
Name                  Null?     Type
---------------   -------- ----------------------------
INDEX_NAME          NOT NULL VARCHAR2(30)
TABLE_NAME          NOT NULL VARCHAR2(30)
MONITORING                   VARCHAR2(3)
USED                          VARCHAR2(3)
START_MONITORING             VARCHAR2(19)
END_MONITORING               VARCHAR2(19)
```

在监控任何索引前，这个视图没有任何记录：

```
select   *
from     v$object_usage;
no rows selected
```

开始监控 4 个索引：

```
alter index HRDT_INDEX1 monitoring usage;
alter index HRDT_INDEX2 monitoring usage;
alter index HRDT_INDEX3 monitoring usage;
alter index HRDT_INDEX4 monitoring usage;
```

现在视图显示 4 个索引的启动时间，但也同时指明它们并未被使用：

```
select index_name, table_name, monitoring, used,
start_monitoring, end_monitoring
from    v$object_usage;
INDEX_NAME  TABLE_NAME MON  USE START_MONITORING    END_MONITORING
----------- ---------- --- --- ------------------- -------------------
HRDT_INDEX1 HRS_DETAIL YES NO   10/13/2002 03:11:34
HRDT_INDEX2 HRS_DETAIL YES NO   10/13/2002 03:11:38
HRDT_INDEX3 HRS_DETAIL YES NO   10/13/2002 03:11:46
HRDT_INDEX4 HRS_DETAIL YES NO   10/13/2002 03:11:52
```

如果使用 HRDT_INDEX1 进行查询，视图则会显示该索引已经投入使用：

```
select index_name, table_name, monitoring, used,
```

```
start_monitoring, end_monitoring
from    v$object_usage;
INDEX_NAME  TABLE_NAME MON USE START_MONITORING      END_MONITORING
----------- ---------- --- --- ------------------- -------------------
HRDT_INDEX1 HRS_DETAIL YES YES 10/13/2002 03:11:34
HRDT_INDEX2 HRS_DETAIL YES NO  10/13/2002 03:11:38
HRDT_INDEX3 HRS_DETAIL YES NO  10/13/2002 03:11:46
HRDT_INDEX4 HRS_DETAIL YES NO  10/13/2002 03:11:52
```

结束了对 HRDT_INDEX4 的监控，视图现在会显示监控的结束时间：

```
alter index HRDT_INDEX4 nomonitoring usage;
select index_name, table_name, monitoring, used,
start_monitoring, end_monitoring
from    v$object_usage;
INDEX_NAME  TABLE_NAME MON USE START_MONITORING      END_MONITORING
----------- ---------- --- --- ------------------- -------------------
HRDT_INDEX1 HRS_DETAIL YES YES 10/13/2002 03:11:34
HRDT_INDEX2 HRS_DETAIL YE   NO   10/13/2002 03:11:38
HRDT_INDEX3 HRS_DETAIL YES NO 10/13/2002 03:11:46
HRDT_INDEX4 HRS_DETAIL NO   NO   10/13/2002 03:11:52 10/13/2002 03:16:01
```

技巧：

使用V$OBJECT_USAGE视图来查看索引是否已被使用。也许某些索引是不需要的。

12.1.16 确定锁定问题

确定锁定问题将有助于定位正在等待其他某些用户或者某些东西的用户。可以使用这个策略来确定当前被锁定在系统中的用户。这也使 DBA 们可以确认一个相关的 Oracle 进程是否真地被锁定了，还是仅仅运行得比较慢。您还能够识别当前的语句是否正在执行锁定用户的操作。下面的程序清单提供了一个确定锁定问题的示例。

注意：

在本书前一版中，这些语句没有经过调整(很令笔者尴尬)。

```
select   /*+ ordered */ b.username, b.serial#, d.id1, a.sql_text
from      v$lock d, v$session b, v$sqltext a
```

```
where     b.lockwait    = d.kaddr
and       a.address     = b.sql_address
and       a.hash_value = b.sql_hash_value;
USERNAME     SERIAL#          ID1    SQL_TEXT
-------- ---------- -----------  ------------------------------
AUTHUSER           53       393242   update emp set salary = 5000
```

您还需要识别在系统中是哪个用户造成了前一个用户被锁定的问题，如下面程序清单所示(通常情况下，这是因为当您靠近用户/开发人员的桌子时，他/她按下了 CTRL-ALT-DEL 而引起的问题)。

```
select    /*+ ordered */ a.serial#, a.sid, a.username, b.id1, c.sql_text
from       v$lock b, v$session a, v$sqltext c
where      b.id1 in
(select   /*+ ordered */ distinct e.id1
from       v$lock e, v$session d
where     d.lockwait     = e.kaddr)
and         a.sid            = b.sid
and        c.hash_value = a.sql_hash_value
and        b.request       = 0;
SERIAL# SID USERNAME   ID1       SQL_TEXT
------- --- -------- ------ ----------------------------------------
18        11 JOHNSON  393242 update authuser.emp set salary=90000
```

在上面的程序清单中，JOHNSON 如果忘记了写至关紧要的 WHERE 子句的话，就皆大欢喜了。但是，JOHNSON 锁定了这张表的授权用户。

也可以详细查看锁定，确切了解运行和中断情况。在第 9 章，我们介绍了代码块级调整；还描述了其中某些列，也执行了一些对 V$TRANSACTION 的查询(它显示所有最近运行的 DML [update/insert/delete]事务)。下面的程序清单中，有 4 个在相同信息块上同时运行的事务。这里没有有中断，因为 initrans 设置为同时处理相同代码块内的 4 个变更(至少设置为 4 ITL 槽)。如果有问题，在下面第 3 个查询中 LMODE 应该是 0,REQUEST 应该是 6 (TX6)。

4 位用户更新相同代码块中的不同行：

```
select /*+ ordered */ username, v$lock.sid, trunc(id1/power(2,16)) rbs,
bitand(id1,to_number('ffff','xxxx'))+0 slot,
id2 seq, lmode, request
from    v$lock, v$session
where  v$lock.type = 'TX'
and     v$lock.sid = v$session.sid;
USERNAME     SID       RBS      SLOT       SEQ    LMODE    REQUEST
-------- ------ --------  -------  -------  ------   -------
```

```
SCOTT        146          6        32        85        6          0
SCOTT        150          4        39     21557        6          0
SCOTT        151          5        34      1510        6          0
SCOTT        161          7        24        44        6          0
select xid, xidusn, xidslot, xidsqn, status, start_scn
from     v$transaction
order by start_scn;
XID                XIDUSN  XIDSLOT  XIDSQN  STATUS              START_SC
N
---------------- ------- ------- ------- ---------------- ----------
0600200055000000         6       32       85  ACTIVE               1657348
0
0400270035540000         4       39    21557   ACTIVE               165735
06
05002200E6050000         5       34     1510  ACTIVE               1657354
5
070018002C000000         7       24       44  ACTIVE               1657442
0
```

3 位用户试图更新相同的行：

```
select /*+ ordered */ username, v$lock.sid, trunc(id1/power(2,16)) rbs,
bitand(id1,to_number('ffff','xxxx'))+0 slot,
id2 seq, lmode, request
from   v$lock, v$session
where   v$lock.type = 'TX'
and     v$lock.sid = v$session.sid;
USERNAME       SID    RBS   SLOT          SEQ       LMODE     REQUEST
---------- ------ ----- ------ ---------- ---------- ----------
SCOTT          146     4      47        21557          0          6
SCOTT          150     4      47        21557          6          0
SCOTT          161     4      47        21557          0          6
select xid, xidusn, xidslot, xidsqn, status, start_scn
from   v$transaction
order by start_scn;
XID                XIDUSN  XIDSLOT      XIDSQN
STATUS              START_SCN
---------------- ------- -------- ---------- ---------------- ----------
```

04002F0035540000 4 47 21557
ACTIVE 16575501

阻止了两个用户：

```
SELECT sid, blocking_session, username, blocking_session_status
FROM     v$session
WHERE    username='SCOTT'
ORDER BY blocking_session;
SID BLOCKING_SESSION    USERNAME  BLOCKING_SESSION_STATUS
------- ---------------- ---------- ----------------------
146                  150  SCOTT      VALID
161                  150  SCOTT      VALID
150                       SCOTT      NO HOLDER
```

12.1.17 关闭有问题的会话

一个用户可能已经运行了一些他/她也不想运行的东西，或者可能需要在工作时间终止一个有问题的查询，到晚上再运行。如果需要终止前面一节的操作的话，可以执行下面的程序清单中的语句(用于发现并终止会话)。

```
select   username, sid, serial#, program, terminal
from     v$session;
alter system kill session '11,18';
```

You can't kill your own session though:

```
alter system kill session '10,4';
*
ERROR at line 1:
ORA-00027: cannot kill current session
```

以上程序代码中参数的顺序是 SID，然后是 SERIAL#。确保可以使用 V$SESSION，因为它有许多有帮助的列。在 Oralce 的前一版本中，可终止当前的用户会话。令人高兴的是，您不再会意外地终止自己的会话，如上面的程序清单所示。

技巧：

确定锁定其他用户的用户并终止他们的会话(如果需要)。

12.1.18 查找使用多会话的用户

有些时候，用户喜欢使用多会话来一次完成多个任务，但这会引起问题。开发人员也会有同样的问题，如果他开发了一个创建了会派生大量进程的糟糕的应用程序。所有这些都可能降低系统的综合性能。用户名 NULL 是后台进程。下面的程序清单中对 V$SESSION 的查询显示了这几种类型的问题：

```
select      username, count(*)
from        v$session
group by username;
USERNAME      COUNT(*)
----------- --------
PLSQL_USER           1
SCOTT                 1
JOHNSON              9
SYS                   4
SYSTEM               1
14
```

在某些 O/S 平台上，如果一个用户开始了一个会话，然后重启 PC，通常当用户再次启动另一个会话时，原有的进程仍将在后台运行。如果用户在多终端或者多台 PC 上运行多个报表，这也将影响系统的综合性能。

注意：

V$SESSION 视图中用户名 NULL 的行是 Oracle 的后台进程。

技巧：

确定使用了多会话的用户，并判定是一个管理问题(用户使用了多终端)还是一个系统问题(会话没有被清除，或者产生了多余的进程)。

查询当前的配置文件

配置文件被限制于一个给定的模式(用户)。要查看系统中的配置文件，请执行下面程序清单中的查询。

```
select      substr(profile,1,10) Profile,
substr(resource_name,1,30) "Resource Name",
substr(limit,1,10) Limit
from        dba_profiles
group by    substr(profile,1,10), substr(resource_name,1,30),
substr(limit,1,10);
```

```
PROFILE   Resource Name                    LIMIT
-------   ------------------------------ ----------
DEFAULT    IDLE_TIME                      UNLIMITED
DEFAULT    PRIVATE_SGA                    UNLIMITED
DEFAULT    CONNECT_TIME                   UNLIMITED
DEFAULT    CPU_PER_CALL                   UNLIMITED
DEFAULT    COMPOSITE_LIMIT                UNLIMITED
DEFAULT    CPU_PER_SESSION                UNLIMITED
DEFAULT    SESSIONS_PER_USER              UNLIMITED
DEFAULT    PASSWORD_LIFE_TIME             UNLIMITED
DEFAULT    PASSWORD_LOCK_TIME             UNLIMITED
DEFAULT    PASSWORD_REUSE_MAX             UNLIMITED
DEFAULT    PASSWORD_GRACE_TIME            UNLIMITED
DEFAULT    PASSWORD_REUSE_TIME            UNLIMITED
DEFAULT    FAILED_LOGIN_ATTEMPTS          10
DEFAULT    LOGICAL_READS_PER_CALL         UNLIMITED
DEFAULT    PASSWORD_VERIFY_FUNCTION       NULL
DEFAULT     LOGICAL_READS_PER_SESSION     UNLIMITED
MONITORING IDLE_TIME                       DEFAULT
MONITORING PRIVATE_SGA                     DEFAULT
MONITORING CONNECT_TIME                    DEFAULT
MONITORING CPU_PER_CALL                    DEFAULT
MONITORING COMPOSITE_LIMIT                 DEFAULT
MONITORING CPU_PER_SESSION                 DEFAULT
MONITORING SESSIONS_PER_USER               DEFAULT
MONITORING PASSWORD_LIFE_TIME              DEFAULT
MONITORING PASSWORD_LOCK_TIME              DEFAULT
MONITORING PASSWORD_REUSE_MAX              DEFAULT
MONITORING PASSWORD_GRACE_TIME             DEFAULT
MONITORING PASSWORD_REUSE_TIME             DEFAULT
MONITORING FAILED_LOGIN_ATTEMPTS           UNLIMITED
MONITORING LOGICAL_READS_PER_CALL          DEFAULT
MONITORING PASSWORD_VERIFY_FUNCTION        DEFAULT
MONITORING LOGICAL_READS_PER_SESSION       DEFAULT
32 rows selected.
```

12.1.19 查找磁盘 I/O 问题

视图 V$DATAFILE、V$FILESTAT 和 V$DBA_DATA_FILES 提供了数据库中所有数据文件和磁盘的文件 I/O 活动信息。理想情况下，物理的读和写应当平均分布。如果没有合理的配置系统，其综合性能就会受到影响。下面程序清单中的脚本展示了实际的分布情况并可以很方便地判断出是否有不平衡的现象存在。第 3 章对这个问题进行了详细的讨论；本节只是显示快捷式查询，以了解其大致情况。

```
select   a.file#, a.name, a.status, a.bytes,
b.phyrds, b.phywrts
from     v$datafile a, v$filestat b
where    a.file# = b.file#;
```

下面两个程序清单中的查询提供了有关文件和数据分布问题的报告，并已改进其格式。第一个报告获得数据文件 I/O 的情况，第二个报告获得磁盘 I/O 的情况。

```
Set TrimSpool   On
Set Line        142
Set Pages        57
Set NewPage       0
Set FeedBack   Off
Set Verify      Off
Set Term         On
TTitle          Off
BTitle          Off
Clear Breaks
Break On Tablespace_Name
Column TableSpace_Name For A12      Head "Tablespace"
Column Name       For A45           Head "File Name"
Column Total      For 999,999,990   Head "Total"
Column Phyrds    For 999,999,990    Head "Physical|Reads "
Column Phywrts   For 999,999,990    Head "Physical| Writes "
Column Phyblkrd For 999,999,990    Head "Physical |Block Reads"
Column Phyblkwrt For 999,999,990   Head "Physical |Block Writes"
Column Avg_Rd_Time   For 90.9999999 Head "Average |Read Time|Per Block"
Column Avg_Wrt_Time For 90.9999999 Head "Average |Write Time|Per Block"
Column Instance           New_Value _Instance    NoPrint
Column Today              New_Value _Date        NoPrint
Select  Global_Name Instance, To_Char(SysDate,'FXDay, Month DD, YYYY HH:MI')
Today
From      Global_Name;
TTitle On
```

```
TTitle Left 'Date Run: ' _Date Skip 1-
Center 'Data File I/O' Skip 1 -
Center 'Instance Name: ' _Instance Skip 1
select       C.TableSpace_Name, B.Name, A.Phyblkrd +
A.Phyblkwrt Total, A.Phyrds, A.Phywrts,
A.Phyblkrd, A.Phyblkwrt
From         V$FileStat A, V$DataFile B, Sys.DBA_Data_Files C
where        B.File# = A.File#
and           B.File# = C.File_Id
order by    TableSpace_Name, A.File#
/
select object_name, statistic_name, value
from v$segment_statistics
where value > 100000
order by value;
OBJECT_NAME STATISTIC_NAME    VALUE
-------------------------------------
ORDERS       space allocated   96551
ORDERS       space allocated   134181
ORDERS       logical reads     140976
ORDER_LINES db block changes 183600
```

下面的程序清单获得磁盘 I/O:

```
Column TableSpace_Name For A12      Head "Tablespace"
Column Total        For 9,999,999,990 Head "Total"
Column Phyrds       For 9,999,999,990 Head "Physical|Reads "
Column Phywrts     For 9,999,999,990 Head "Physical| Writes "
Column Phyblkrd    For 9,999,999,990 Head "Physical |Block Reads"
Column Phyblkwrt   For 9,999,999,990 Head "Physical |Block Writes"
Column Avg_Rd_Time For 9,999,990.9999 Head "Average|Read Time  |Per Block"
Column Avg_Wrt_TimeFor 9,999,990.9999 Head "Average |Write Time|Per Block"
Clear Breaks
Break on Disk Skip 1
Compute Sum Of Total On Disk
Compute Sum Of Phyrds On Disk
Compute Sum Of Phywrts On Disk
Compute Sum Of Phyblkrd On Disk
Compute Sum Of Phyblkwrt On Disk
```

```
TTitle Left 'Date Run: ' _Date Skip 1-
Center 'Disk I/O' Skip 1 -
Center 'Instance Name: ' _Instance Skip 2
select       SubStr(B.Name, 1, 13) Disk, C.TableSpace_Name,
A.Phyblkrd + A.Phyblkwrt Total, A.Phyrds, A.Phywrts,
A.Phyblkrd, A.Phyblkwrt, ((A.ReadTim /
Decode(A.Phyrds,0,1,A.Phyblkrd))/100) Avg_Rd_Time,
((A.WriteTim / Decode(A.PhyWrts,0,1,A.PhyblkWrt)) /
100) Avg_Wrt_Time
from         V$FileStat A, V$DataFile B, Sys.DBA_Data_Files C
where        B.File# = A.File#
and           B.File# = C.File_Id
order by     Disk,C.Tablespace_Name, A.File#
/
Set FeedBack On
Set Verify     On
Set Term       On
Ttitle         Off
Btitle         Off
```

技巧：

视图 V$DATAFILE、V$FILESTAT 和 V$DBA_DATA_FILES 提供了数据库中所有数据文件和磁盘的文件 I/O 活动信息。确保数据文件和磁盘都处于合理的平衡中，以获得最佳的性能。

12.1.20 查找回滚段的内容

这个有帮助的查询显示了一个回滚段的实际等待数。可以显示回滚信息(包括自动撤消)。还可以从程序清单显示的视图中查询 Shrink 和 wrap 信息：

注意：

这个数据库中使用了自动或系统托管的撤消。

```
select    a.name, b.extents, b.rssize, b.xacts,
b.waits, b.gets, optsize, status
from       v$rollname a, v$rollstat b
where      a.usn = b.usn;
```

```
File Name      EXTENTS     RSSIZE      XACTS WAITS      GETS     OPTSIZE STATUS
------------ ------- -------- ------- ------   ------    -------------
SYSTEM                6      385024          0        0        164
 ONLINE
_SYSSMU1$             6     4317184          0        0       3947
ONLINE
_SYSSMU2$             6     3334144          0       ,
0       2985              ONLINE
_SYSSMU3$             7      450560          1        0        204
ONLINE
_SYSSMU4$             4      253952          0        0        244
 ONLINE
_SYSSMU5$            17     2088960          0        1       5426
ONLINE
_SYSSMU6$             7      450560          0        0       1070
ONLINE
_SYSSMU7$             3      188416          0        0        275
ONLINE
_SYSSMU8$             2      122880          0        0        182
ONLINE
_SYSSMU9$             2      122880          0        0        182
ONLINE
_SYSSMU10$            2      122880          0        0        182             0
NLINE
```

技巧：

查询 V$ROLLNAME、V$ROLLSTAT 和 V$TRANSACTION 视图可以提供用户如何使用回滚段和撤消表空间的信息。通常情况下，在一个时间点上不应让多个用户访问同一个回滚段(尽管这是被允许的)。

注意：

如果使用自动撤消管理，通常就不需要前一个查询。

数据库吧
database8.com

下面程序清单中的查询显示了整个系统在整体上的等待数。

```
Set TrimSpool  On
Set NewPage      0
```

```
Set Pages       57
Set Line        132
Set FeedBack  Off
Set Verify     Off
Set Term        On
Ttitle         Off
Btitle          Off
Clear Breaks
Column Event          For A40 Heading "Wait Event"
Column Total_Waits For 999,999,990 Head "Total Number| Of Waits "
Column Total_Timeouts For 999,999,990 Head "Total Number|Of TimeOuts"
Column Tot_Time     For 999,999,990 Head "Total Time|Waited "
Column Avg_Time     For 99,990.999 Head "Average Time|Per Wait "
Column Instance New_Value _Instance NoPrint
Column Today      New_Value _Date       NoPrint
select Global_Name Instance, To_Char(SysDate,
'FXDay DD, YYYY HH:MI') Today
from    Global_Name;
TTitle On
TTitle Left 'Date Run: ' _Date Skip 1-
Center 'System Wide Wait Events' Skip 1 -
Center 'Instance Name: ' _Instance Skip 2
Select     event, total_waits, total_timeouts,
(time_waited / 100) tot_time, (average_wait / 100)
Avg_time
from       v$system_event
order by  total_waits desc
/
Date Run: Friday     01, 2006 09:24
System Wide Wait Events
Instance Name: ORCL
Total Number Total Number Total Time Average
Time
Wait Event                    Of Waits    Of TimeOuts    Waited    Per Wait
---------------------- ------------ ------------ -------- ---------
db file sequential read     2,376,513              0    30,776
0.010
```

```
db file scattered read        136,602              0       6,069
0.040
rdbms ipc message              103,301         99,481    276,659
2.680
latch: redo writing             57,488              0          0
0.000
...etc...
```

12.1.21 检查空闲列表是否充足

如果使用多进程完成大量的插入操作，空闲列表(空闲的数据库数据块的列表)的默认值 1 可能是不够的。如果没有使用自动空间段管理(Automatic Space Segment Management，简称 ASSM)，您可能需要增加空闲列表，或者空闲列表组(请参阅第 14 章，以获得更多的信息)。为了检查空闲列表组的存储参数是否足够，请运行下面程序清单显示的报表。

```
Set TrimSpool On
Set Line 132
Set Pages 57
Set NewPage 0
Set FeedBack Off
Set Verify Off
Set Term Off
TTitle Off
BTitle Off
Column Pct Format 990.99 Heading "% Of |Free List Waits"
Column Instance New_Value _Instance NoPrint
Column Today New_Value _Date NoPrint
select Global_Name Instance, To_Char
(SysDate, 'FXDay DD, YYYY HH:MI') Today
from    Global_Name;
TTitle On
TTitle Left 'Date Run: ' _Date Skip 1-
Center 'Free list Contention' Skip 1 -
Center 'If Percentage is Greater than 1%' Skip 1 -
Center 'Consider increasing the number of free lists' Skip 1 -
Center 'Instance Name: ' _Instance
select    ((A.Count / (B.Value + C.Value)) * 100) Pct
from       V$WaitStat A, V$SysStat B, V$SysStat C
where      A.Class = 'free list'
```

```
and        B.Statistic# = (select Statistic#
from    V$StatName
where   Name = 'db block gets')
and       C.Statistic# =  (select Statistic#
from    V$StatName
where   Name = 'consistent gets')
/
Date Run: Friday 01, 2006 09:26
Free list Contention
If Percentage is Greater than 1%
Consider increasing the number of free lists
Instance Name: ORCL
% Of
Free List Waits
---------------
0.00
(of course... I' m using ASSM)
```

如果活动比率超过 1%，就需要增加空闲列表组了。

技巧：

在使用多进程完成大量的插入操作时，应确保有足够的空闲列表和空闲列表组。空闲列表的默认存储值是 1。如果您使用了 ASSM，Oracle 将为您管理这些参数，但是一个有大量数据交换的事务环境中，在应用 ASSM 前应经过仔细的测试。虽然如此，但通常最好使用 ASSM。

12.1.22 检查角色和权限设置

本节包含了多个 V$脚本，用于显示各种安全权限。下列程序清单中每个脚本的标题将简要说明了它们将检索的信息。输出结果依据系统不同可能很大，因此要小心运行。

根据用户名进行授权的对象级特权

```
select b.owner || '.' || b.table_name obj,
b.privilege what_granted, b.grantable,
a.username
from   sys.dba_users a, sys.dba_tab_privs b
where  a.username = b.grantee
order by 1,2,3;
```

根据被授权人进行授权的对象级特权

```
Select owner || '.' || table_name obj,
privilege what_granted, grantable, grantee
from    sys.dba_tab_privs
where   not exists
(select 'x'
from sys.dba_users
where username = grantee)
order by 1,2,3;
```

根据用户名进行授予的系统级特权

```
select   b.privilege what_granted,
b.admin_option, a.username
from     sys.dba_users a, sys.dba_sys_privs b
where    a.username = b.grantee
order by 1,2;
```

根据被授权人进行授予的系统级特权

```
select   privilege what_granted,
admin_option, grantee
from     sys.dba_sys_privs
where    not exists
(select 'x' from sys.dba_users
where username = grantee)
order by 1,2;
```

根据用户名授予的角色

```
select   b.granted_role ||
decode(admin_option, 'YES',
' (With Admin Option)',
null) what_granted, a.username
from     sys.dba_users a, sys.dba_role_privs b
where    a.username = b.grantee
order by 1;
```

根据被授权人授予的角色

```
select   granted_role ||
```

```
decode(admin_option, 'YES',
' (With Admin Option)', null) what_granted,
grantee
from     sys.dba_role_privs
where    not exists
(select 'x'
from sys.dba_users
where username = grantee)
order by 1;
```

用户名及已被授予的相应权限

```
select a.username,
b.granted_role || decode(admin_option,'YES',
' (With Admin Option)',null) what_granted
from     sys.dba_users a, sys.dba_role_privs b
where    a.username = b.grantee
UNION
select a.username,
b.privilege || decode(admin_option,'YES',
' (With Admin Option)', null) what_granted
from     sys.dba_users a, sys.dba_sys_privs b
where    a.username = b.grantee
UNION
select   a.username,
b.table_name || ' - ' || b.privilege
|| decode(grantable,'YES',
' (With Grant Option)',null) what_granted
from     sys.dba_users a, sys.dba_tab_privs b
where    a.username = b.grantee
order by 1;
```

技巧：

记录系统中已有的特权，这样您就可以应对各种类型的安全场景。

查询用户名及相应的配置文件、默认的表空间和临时表空间

```
Select   username, profile, default_tablespace,
```

```
temporary_tablespace, created
from      sys.dba_users
order by username;
```

12.1.23 等待事件 V$视图

本节包含一些显示等待事件的 V$ 脚本。从个人角度来说，我更喜欢使用 STATSPACK 报表、AWR 报表或企业管理器来查找等待事件。也就是说，有些很好的视图可以查看等待事件。Oracle 10gR2 中添加了一些新的视图，但最幸运的是在 V$SESSION_WAIT 中找到的东西现在在 V$SESSION 中可以找到。

马上该谁等待——查询 V$SESSION_WAIT / V$SESSION

```
select event, sum(decode(wait_time,0,1,0)) "Waiting Now",
sum(decode(wait_time,0,0,1)) "Previous Waits",
count(*) "Total"
from v$session_wait
group by event
order by count(*);
WAIT_TIME = 0 means that it's waiting
WAIT_TIME > 0 means that it previously waited this many ms
EVENT                          Waiting Now  Previous Waits Total
-------------------------- ------------ -------------- -------
db file sequential read        0              1                1
db file scattered read         2              0                2
latch free                       0              1                1
enqueue                          2              0                2
SQL*Net message from client   0            254              480
...
select event, sum(decode(wait_time,0,1,0)) "Waiting Now",
sum(decode(wait_time,0,0,1)) "Previous Waits",
count(*) "Total"
from v$session
group by event
order by count(*);
EVENT                            Waiting Now Previous Waits   Total
-------------------------- ------------ --------------    --------
db file sequential read        0              1                  1
db file scattered read         2              0                  2
latch free                       0              1                  1
```

```
enqueue                          2          0              2
SQL*Net message from client   0          254            480
...
```

马上该谁等待；SPECIFIC Waits——查询 V$SESSION_WAIT

```
SELECT /*+ ordered */ sid, event, owner, segment_name, segment_type,p1,p2,p3
FROM   v$session_wait sw, dba_extents de
WHERE  de.file_id = sw.p1
AND    sw.p2 between de.block_id and de.block_id+de.blocks - 1
AND    (event = 'buffer busy waits' OR event = 'write complete waits')
AND    p1 IS NOT null
ORDER BY event,sid;
```

谁在等待 - 最后 10 个等待数——查询 V$SESSION_WAIT_HISTORY

```
SELECT /*+ ordered */ sid, event, owner, segment_name, segment_type,p1,p2,p3
FROM   v$session_wait_history sw, dba_extents de
WHERE  de.file_id = sw.p1
AND    sw.p2 between de.block_id and de.block_id+de.blocks - 1
AND    (event = 'buffer busy waits' OR event = 'write complete waits')
AND    p1 IS NOT null
ORDER BY event,sid;
```

查找 P1, P2, P3 代表什么——查询 V$EVENT_NAME

```
col name for a20
col p1 for a10
col p2 for a10
col p3 for a10
select event#,name,parameter1 p1,parameter2 p2,parameter3 p3
from    v$event_name
where   name in ('buffer busy waits', 'write complete waits');
EVENT#          NAME                     P1           P2           P3
------------- -------------------- ---------- ---------- ----------
143 write complete waits file#        block#
145 buffer busy waits     file#        block#     id
```

会话开始后的所有等待数——查询 V$SESSION_EVENT

```
select sid, event, total_waits, time_waited, event_id
```

```
from     v$session_event
where    time_waited > 0
order    by time_waited;
SID          EVENT                                TOTAL_WAITS  TIME_WAITED
---------- ---------------------------- ----------- -----------
159 process startup                                 2             1
167 latch: redo allocation                        4             1
168 log buffer space                               2             3
166 control file single write                     5             4
...
```

类的所有会话等待数——查询 V$SESSION_WAIT_CLASS

```
select sid, wait_class, total_waits
from    v$session_wait_class;
SID WAIT_CLASS              TOTAL_WAITS
---------- ------------------- -----------
168 Other                                2
168 Concurrency                          1
168 Idle                             12825
168 User I/O                            12
168 System I/O                       4448
169 Other                                1
169 Idle                             12812
170 Idle                             13527
```

系统启动后的所有等待数——查询 V$SYSTEM_EVENT

```
select  event, total_waits, time_waited, event_id
from     v$system_event
where    time_waited > 0
order    by time_waited;
EVENT                                     TOTAL_WAITS
TIME_WAITED  EVENT_ID
---------------------------------- ----------- ----------- ----------
enq: TX - row lock contention                  1196        366837     310662678
enq: TM -
contention                               170         52074     668627480
```

```
     db file sequential
read                          17387          3163    2652584166
     control file parallel write                    12961         23117 4078387448
     db file scattered read                          4706         15762    506183215
     class slave
wait                                    20         10246    1055154682
```

类的系统等待数——查询 V$SYSTEM_WAIT_CLASS

```
select wait_class, total_waits
from    v$system_wait_class
order   by total_waits desc;
WAIT_CLASS              TOTAL_WAITS
-------------------- -----------
Idle                          161896
Other                          65308
System I/O                     24339
User I/O                       22227
Application                    1404
Commit                           524
Network                          522
Concurrency                      221
Configuration                     55
...
```

类的系统等待数——查询 V$ACTIVE_SESSION_HISTORY

```
--In the query below, the highest count session is leader in non-idle wait
events.
select session_id,count(1)
from    v$active_session_history
group   by session_id
order  by 2;
In the query below, find the SQL for the leader in non-idle wait events.
select c.sql_id, a.sql_text
from v$sql a, (select sql_id,count(1)
from v$active_session_history b where sql_id is not null
group by sql_id
order by 2 desc) c
```

```
where rownum <= 5
order by rownum;
```

技巧：

在 Oracle 10g 中 V$SESSION_WAIT 中的所有等待事件列现在都在 V$SESSION 中。因此，确保查询等待信息的 V$SESSION，因为它是一个更快的视图。V$ACTIVE_SESSION_HISTORY (ASH) 将许多重要统计数据合并为一个视图或一个报表(ASH 报表)。

12.1.24 一些主要的 V$视图种类

本节中根据视图的主要功能对它们进行了分类。这里的列表并不完整(V$ 视图和 X$表查询的完整列表请参阅附录 B)。您将经常需要将一类视图与另一类视图进行连接，以检索所需要的信息。可以像对其他 Oracle 视图一样对 V$视图进行查询，但要记住这些表中的信息变化很快。您可以将 V$视图中的信息导入先前已经创建好的表中，这样就可以花费一定时间编译数据，或者供以后分析使用，或者用于建立统计数据报表和基于数据库的不同条件产生警告。

目前市场上的绝大多数 DBA 监控工具都使用了 V$视图(和 X$表)。若不使用 DBA 监控工具来查询数据库信息，需要您对每个视图中存储的信息和如何正确地查询视图有深刻的了解。表 12-1 包含了一个根据它们的主要功能分类的 V$视图列表。列表中的视图分类与它们监控的操作相关联。这张列表不是完整的，仅包含了绝大多数常用的视图。有些视图因为 Oracle 版本的不同而不同。这些就是 TUSC V$ Poster 中包含的信息。

表 12-1 V$视图分类

类 别	描述和相关的 V$视图
顾问	与缓存顾问相关的信息 V$视图: V$PGA_TARGET_ADVICE、V$PGA_TARGET_ADVICE_HISTOGRAM、V$MTTR_TARGET_ADVICE、V$PX_BUFFER_ADVICE、V$DB_CACHE_ ADVICE、 V$SHARED_POOL_ADVICE、V$JAVA_POOL_ADVICE、V$STREAMS_POOL_ ADVICE (10.2)、V$SGA _ TARGET _ ADVICE (10.2)和 V$ADVISOR _ PROGRESS (10.2)
ASM	V$ASM_ALIAS(10.1)、V$ASM_CLIENT(10.1)、V$ASM_DISK(10.1)、V$ASM_DISK_ STAT(10.2)、V$ASM_DISKGROUP(10.1)、V$ASM_DISKGROUP_STAT(10.2)、V$ASM _FILE(10.1)、V$ASM _OPERATION(10.1)、V$ASM_ TEMPLATE(10.1)
备份/恢复	有关数据库备份和恢复的信息，包括以前的备份、归档日志、备份文件的状态以及恢复信息 V$视图: V$ARCHIVE, V$ARCHIVED_LOG, V$ARCHIVE_DEST, V$ARCHIVE_DEST_STATUS, V$ARCHIVE_GAP, V$ARCHIVE_PROCESSES, V$BACKUP, V$BACKUP_ASYNC_IO, V$BACKUP_CORRUPTION, V$BACKUP_DATAFILE,

	V$BACKUP_DEVICE,V$BACKUP_PIECE,V$BACKUP_REDOLOG,V$BACKUP_SET,V$BACKUP_SYNC_IO,V$BLOCK_CHANGE_TRACKING,V$COPY_OCORRUPTION,V$DATABASE_BLOCK_CORRUPTION,V$DATABASEINCARNATION,V$DATAFILE_COPY,V$DELETED_OBJECT,V$FAST_START_SERVERS,V$FAST_START_ TRANSACTIONS,V$INSTANCE_RECOVERY,V$MTTR_TARGET_ADVICE,V$PROXY_ARCHIVEDLOG,V$PROXY_DATAFILE,V$RMAN_CONFIGURATION,V$RECOVERY_FILE_STATUS,V$RECOVERY_LOG,V$RECOVERY_PROGRESS,V$RECOVERY_ STATUS,V$RECOVER_FILE,V$BACKUP_ARCHIVELOG_DETAILS(10.2),V$BACKUP_ARCHIVELOG_SUMMARY(10.2),V$BACKUP_CONTROLFILE_DETAILS(10.2), V$BACKUP_CONTROLFILE_SUMMARY(10.2),V$BACKUP_COPY_DETAILS(10.2),V$BACKUP_COPY_SUMMARY(10.2),V$BACKUP_FILES(10.1),V$BACKUP_PIECE_DETAILS(10.2),V$BACKUP_SET_DETAILS(10.2),V$BACKUP_SET_SUMMARY(10.2),V$BACKUP_SPFILE,V$BACKUP_SPFILE_DETAILS(10.2),V$BACKUP_SPFILE_ SUMMARY(10.2),V$DATAFILE_HEADER,V$FLASH_RECOVERY_AREA_USAGE(10.2),V$FLASHBACK_DATABASE_LOG(10.1),V$FLASHBACK_DATABASE_STAT(10.1),V$OBSOLETE_BACKUP_FILES,V$OFFLINE_RANGE,V$PROXY_ARCHIVELOG_DETAILS(10.2),V$PROXY_ARCHIVELOG_SUMMARY(10.2),V$PROXY_COPY_ DETAILS(10.2),V$PROXY_COPY_SUMMARY(10.2),V$RECOVERY_FILE_DEST(10,1),V$RESTORE_POINT(10.2),V$RMAN_BACKUPJOB_DETAILS(10.2),V$RMAN_ BACKUP_SUBJOB_DETAILS(10.2),V$RMAN_BACKUP_TYPE(10.2),V$RMAN_ ENCRYPTION_ALGORITHMS(10.2),V$RMAN_OUTPUT(10.1),V$RMAN_STATUS(10.1),V$UNUSABLE_BACKUPFILE_DETAILS(10.2)

类　别	描述和相关的 V$视图
缓存	有关各种缓存的信息，包括对象、库、游标和字典 V$ 视 图: V$ACCESS,V$BUFFER_POOL,V$BUFFER_POOL_STATISTICS,V$DB_ CACHE_ADVICE,VDB_OBJECT_CACHE,VJAVA_POOL_ADVICE,V$LIBRARYC- ACHE,V$LIBRARY_CACHE_MEMORY,V$PGASTAT,V$PGA_TARGET_ADVICE, V$PGA_TARGET_ADVICE_HISTOGRAM,V$ROWCACHE,V$ROWCACHE_PARENT, V$ROWCACHE_SUBORDINATE,V$SESSION_CURSOR_CACHE,VSGA,VSGASTAT, V$SGA_CURRENT_RESIZE_OPS,V$SGA_DYNAMIC_COMPONENTS,V$SGA_ DYNAMIC_FREE_MEMORY,VSGA_RESIZE_OPS,VSGAINFO,V$SHARED_POOL_ADVICE,V$SHARED_POOL_RESERVED,V$SYSTEM_CURSOR_CACHE, V$SUBCACHE,V$JAVA_LIBRARY_CACHE_MEMORY(10.1),V$PROCESS_MEMORY(10.2),V$SGA_TARGET_ADVICE(10.2)
缓存熔合/RAC	V$ACTIVE_INSTANCES,V$BH,V$CLUSTER_INTERCONNECTS(10.2),V$CON- FIGURED_INTERCONNECTS(10.2),VCR_BLOCK_SERVER,VCURRENT_BLOCK_SERVER(10.1),V$GC_ELEMENT,VGCSHVMASTER_INFO,V$GCSPFMAS_TER_INFO,V$

	GES_BLOCKING_ENQUEUE,V$GES_CONVERT_LOCAL,V$GES_CONVERT_ REMOTE,V$GES_ENQUEUE,V$GES_LATCH,V$GES_RESOURCE,V$GES_ STATISTICS,V$HVMASTER_INFO,V$INSTANCE_CACHE_TRANSFER,V$RESOURCE_LIMIT
控制文件	有关实例控制文件的信息 V$ 视 图: V$CONTROLFILE, V$CONTROLFILE_RECORD_SECTION
游标/SQL 语句	有关游标和 SQL 语句的信息，包括开放游标、统计数据和实际的 SQL 文本 V$ 视 图: V$OPEN_CURSOR,V$SQL,V$SQLAREA,V$SQLTEXT, V$SQLTEXT_WITH_NEWLINES,V$SQL_BIND_DATA,V$SQL_BIND_METADATA,V$SQL_CURSOR,V$SQL_OPTIMIZER_ENV,V$SQL_PLAN,V$SQL_PLAN_STATISTICS, V$SQL_PLAN_STATISTICS_ALL,V$SQL_REDIRECTION,V$SESSION_CURSOR_CACHE,VSQL_SHARED_CURSOR,VSQL_SHARED_MEMORY,V$SQL_ WORKAREA,V$SQL_WORKAREA_ACTIVE,V$SQL_WORKAREA_HISTOGRAM, V$SYSTEM_CURSOR_CACHEV$MUTEX_SLEEP(10.2),V$MUTEX_SLEEP_HISTORY(10.2) (专用于那些不确定的共享闩锁), V$SQL_BIND_ CAPTURE(10.1), V$SQL_ JOIN_FILTER(10.2),V$SQL_ AREA_ PLAN_HASH(10.2), V$SQLSTATS(10.2), V$SYS _OPTIMIZER_ENV(10.1), V$VPD_ POLICY

(续表)

类 别	描述和相关的 V$视图
数据库实例	有关实际数据库实例的信息 V$ 视 图: V$ACTIVE_INSTANCES,V$BGPROCESS,V$DATABASE, V$INSTANCE, V$PROCESS,V$SGA,V$SGASTAT,V$BLOCKING_QUIESCE(10.2),V$CLIENT_STATS (10.1) RAC 视图: VBH, VACTIVE_INSTANCES
直接路径操作	有关 SQL*Loader 直接加载产品项的信息 V$ 视 图: V$LOADISTAT, V$LOADPSTAT
分布式/异类服务	V$ 视 图: V$DBLINK,V$GLOBAL_TRANSACTION, V$GLOBAL_ BLOCKED_ LOCKS,V$HS_AGENT,V$HS_PARAMETER,V$HS_SESSION
固定视图	有关 V$表自身的信息 V$ 视 图: V$FIXED_TABLE,V$FIXED_VIEW_DEFINITION, V$INDEXED_FIXED_ COLUMN
通用信息	有关各种系统信息的通用信息 V$ 视 图:V$DBPIPES,V$CONTEXT,V$GLOBALCONTEXT,V$LICENSE, V$OPTION, V$RESERVED_WORDS,V$TIMER,V$TIMEZONE_NAMES,V$TYPE_SIZE,V$_ SEQUENCES,V$VERSIONV$DB_TRANSPORTABLE_PLATFORM(10.2), V$TRANSPORTABLE_PLATFORM(10.2)V$TRANSPORTABLE_PLATFORM(10.1)

	V$SCHEDULER_RUNNING_JOBS(10.2)
I/O	有关 I/O 的信息，包括文件和统计数据 V$ 视 图: V$DBFILE,V$FILESTAT,V$WAITSTAT,V$TEMPSTAT, V$FILE_ HISTOGRAM(10.1),V$FILEMETRIC(10.1),V$FILEMETRIC_HISTORY(10.1), V$SYSAUX_OCCUPANTS(10.1)。V$TABLESPACE, V$TEMP_HISTOGRAM(10.1),V$TEMP _SPACE_HEADER, V$TEMPFILE, V$TEMPSEG_USAGE
闩锁/锁定	有关闩锁和锁定的信息 V$ 视 图:V$ENQUEUE_LOCK,V$ENQUEUE_STAT, V$EVENT_ NAME,V$FILE_ CACHE_TRANSFER,V$GLOBAL_BLOCKED_LOCKS,V$LATCH,V$LATCHHOLDER,V$ LATCHNAME,V$LATCH_CHILDREN,V$LATCH_MISSES,V$LATCH_PARENT, V$LOCK,V$LOCKED_OBJECT,V$RESOURCE,V$RESOURCE_LIMIT, V$TRANSACTION_ENQUEUE,V$_LOCK,V$_LOCK1,V$ENQUEUE_STATISTICS RAC 视 图:VCR_BLOCK_SERVER,VGCSHVMASTER_INFO, V$GCSPFMASTER _INFO,V$GC_ELEMENT,V$GES_BLOCKING_ENQUEUE,V$GES_ENQUEUE, V$HVMASTER_INFO,V$GES_LATCH, V$GES_RESOURCES

(续表)

类别	描述和相关的 V$视图
重做日志	有关重做日志的信息，包括统计信息和历史 V$ 视 图: VLOG, VLOGFILE, V$LOGHIST, V$LOG_HISTORY, V$THREAD (与 RAC 相关)
复制和物化视图	有关复制和物化(materialized)视图的信息 V$ 视 图: V$MVREFRESH , V$REPLPROP, V$REPLQUEUE
资源管理器	有关资源管理的信息 V$ 视 图: V$ACTIVE_SESSION_POOL_MTH, V$ACTIVE_SESSION_POOL_HISTORY, V$RSRC_CONS_GROUP_HISTORY(10.2),V$RSRC_CONSUMER_GROUP,V$RSRC_ CONSUMER_GROUP_CPU_MTH,V$RSRC_PLAN,V$RSRC_PLAN_CPU_MTH, V$RSRC_PLAN_HISTORY(10.2), V$RSRC_SESSION_INFO(10.2)
回滚段和撤消	有关回滚段的信息，包括统计数据和事务 V$ 视 图: V$ROLLNAME, V$ROLLSTAT, V$TRANSACTION, V$UNDOSTAT
安全/权限	有关安全的信息 V$ 视 图: V$ENABLEDPRIVS,V$PWFILE_USERS,VVPD_POLICY, VWALLET (10.2), 和 V$XML_AUDIT_TRAIL(10.2)
会话(包括某些复制信息和不同种类的服务)	有关会话的信息，包括访问对象、游标、进程和统计数据 V$ 视 图:V$ACTIVE_SESSION_HISTORY,V$MYSTAT,V$PROCESS, V$SESS_TIME _MODEL,V$SESSION,V$SESSION_CONNECT_INFO,V$SESSION_CURSOR_CACHE, V$SESSION_EVENT,V$SESSION_LONGOPS,V$SESSION_OBJECT_CACHE,V$SESSIO N_WAIT,V$SESSION_WAIT_CLASS,V$SESSION_WAIT_HISTORY,V$SESSTAT,V$SESS_I

	O,V$SES_OPTIMIZER_ENV,V$SESSMETRIC, 以及 V$CLIENT_STATS, 和 V$TSM_SESSIONS(10.2)
服务(在 10.1 中这些都是新的)	V$ACTIVE_SERVICES,V$SERV_MOD_ACT_STATS,V$SERVICE_EVENT, V$SERVICE_STATS, V$SERVICE_WAIT_CLASS, V$SERVICES
排序	有关排序的信息 V$视图:V$SORT_SEGMENT, V$TEMPSEG_USAGE, V$TEMP_EXTENT_MAP, V$TEMP_EXTENT_POOL,V$TEMP_HISTOGRAM(10.1),V$TEMP_SPACE_HEADER, V$TEMPFILE, V$TEMPSTAT
备用数据库 (Data Guard)	有关备用数据库的信息 V$视图: V$DATAGUARD_STATUS, V$LOGSTDBY, V$LOGSTDBY_STATS, V$MANAGED_STANDBY,V$STANDBY_LOG,V$DATAGUARD_CONFIG(10.1), V$DATAGUARD_STATS(10.2),V$LOGSTDBY_PROCESS(10.2),V$LOGSTDBY_ PROGRESS(10.2),V$LOGSTDBY_STATE(10.2),V$LOGSTDBY_TRANSACTION(10.2)

类别	描述和相关的 V$视图	
日志挖掘	有关日志挖掘(log miner)的信息 V$ 视 图: V$LOGMNR_CALLBACK,V$LOGMNR_CONTENTS, V$LOGMNR_ DICTIONARY,V$LOGMNR_LATCH,V$LOGMNR_LOGFILE,V$LOGMNR_LOGS, V$LOGMNR_PARAMETERS,V$LOGMNR_PROCESS,V$LOGMNR_REGION,V$LOGMNR_SESSION,V$LOGMNR_STATS,V$LOGMNR_TRANSACTION,V$LOGMNR_ DICTIONARY_LOAD(10.2)	
Metrics	有关 Metrics 的信息 (这在 Oracle 10g 中是全新的!) V$ 视 图:V$METRICNAME, V$SERVICEMETRIC,V$EVENTMETRIC, V$FILEMETRIC,V$FILEMETRIC_HISTORY,V$SERVICEMETRIC_HISTORY, V$SESSMETRIC,V$SYSMETRIC,V$SYSMETRIC_HISTORY,V$SYSMETRIC_SU MMARY,V$THRESHOLD_TYPES,V$WAITCLASSMETRIC,V$WAITCLASSMETRIC_HISTORY	
多线程/共享服务器	有关多线程和共享服务器的信息，包括连接、队列、调度和共享服务器 V$ 视 图: V$CIRCUIT, V$DISPATCHER, V$DISPATCHER_RATE, V$QUEUE, V$QUEUEING_MTH,V$REQDIST,V$SHARED_SERVER,V$SHARED_SERVER_ MONITOR,V$DISPATCHER_CONFIG(10.1)	
对象使用	有关对象使用和依赖的信息 V$ 视 图: V$OBJECT_DEPENDENCY, V$OBJECT_USAGE	
整个系统	有关整个系统性能的信息 V$ 视 图:V$GLOBAL_TRANSACTION, V$SHARED_POOL_RESERVED, V$RESUMABLE, V$SORT_SEGMENT, V$TEMPSEG_USAGE, V$STATNAME, V$SYS_ OPTIMIZER_ENV,V$SYS_TIME_MODEL,V$SYSSTAT,V$SYSTEM_CURSOR_CACHE, V$SYSTEM_EVENT,V$TEMPFILE,V$TEMPORARY_LOBS,V$TEMP_EXTENT_MAP, V$TEMP_EXTENT_POOL,V$TEMP_SPACE_HEADER,V$TRANSACTION, V$ALERT _TYPES(110.1),V$EVENT_HISTOGRAM(10.1),V$OSSTAT(10.1),V$SYSTEM_WAIT_ CLASS (10.1) ,V$TEMP_HISTOGRAM(10.1), V$XML_AUDIT_TRAIL	
并行查询	有关并行查询选项的信息 V$ 视 图:V$EXECUTION,V$PARALLEL_DEGREE_LIMIT_MTH, V$PQ_SESSTAT,	

	VPQ_SLAVE,VPQ_SYSSTAT,VPQ_TQSTAT,VPX_PROCE_SS, V$PX_PROCESS _SYSSTAT, V$PX_SESSION, V$PX_SESSTAT	
	有关各种 Oracle 参数的信息，包括初始化和每个会话的 NLS V$ 视 图: V$NLS_PARAMETERS,V$NLS_VALID_VALUES, V$OBSOLETE_ PARAMETER,V$PARAMETER,V$PARAMETER2,V$SPPARAMETER,V$SYS TEM_ PARAM_ETER,V$SYSTEM_PARAMETER2,V$PARAMETER_VALID_VALUE S(10.2)	

(续表)

类 别	描述和相关的 V$视图
文件映射接口	有关文件映射的信息 V$ 视 图: VMAP_COMP_LIST, VMAP_ELEMENT, V$MAP_EXT_ELEMENT, V$MAP _FILE,VMAP_FILE_EXTENT,VMAP_FILE_IO_STACK,V$MAP_LIBRARY 和 V$MAP _SUBELEMENT
流	有关流的信息 V$ 视 图: VAQ,VSTREAMS_APPLY_COORDINATOR, V$STREAMS_APPLY _READER,V$STREAMS_APPLY_SERVER,V$STREAMS_CAPTURE, V$BUFFERED _PUBLISHERS(10.1),V$BUFFERED_QUEUES(10.1), V$BUFFERED_SUBSCRIBERS (10.1),V$PROPAGATION_RECEIVER(10.1),V$PROPAGATION_SENDER(10.1), V$RULE(10.1),V$RULE_SET(10.1),V$RULE_SET_AGGREGATE_STATS(10.1), V$STREAMS_TRANSACTION(10.2)
统计数据	有关通用统计数据的信息 V$视图: V$SEGMENT_STATISTICS, V$SEGSTAT, V$SEGSTAT_NAME, V$STATISTICS_LEVEL, V$STATNAME, V$WAITSTAT
事务	有关通用事务的信息 V$ 视 图: V$GLOBAL_TRANSACTION,V$LOGSTDBY_TRANSACTION, V$RESU MABLE,V$STREAMS_TRANSACTION,V$TRANSACTION,V$TRANSACTION_ ENQUEUE

注意：

V$ROLLNAME 视图的创建和其他 V$视图有一些细微差别。V$ROLLNAME 视图是 X$表和 undo$表连接的结果。某些 V$timing 字段的值依赖于 init.ora 的 TIMED_STATISTICS 参数是否被设置为 TRUE；如果该参数值不是 TRUE，则这些域将没有计时信息。