PRACTICAS TEMA 5.
MONITORIZACIÓN Y AJUSTE TRADICIONAL.

5.1. Revisar los eventos del sistema y comprobar los más significativos. Comprobar el evento “latch free” por sesiones. Comprobar sesiones esperando por el evento “db file sequential read”. Comprueba el parámetro TIMED_STATISTICS.

5.2. Revisa las estadísticas del sistema más significativas. Comprueba el tamaño medio de la PGA de cada sesión. Revisa las lecturas lógicas y físicas y calcula el ratio de E/S. Comparar el uso de CPU para “SQL del sistema” (acceso al DD) sobre el total.

5.3. Ver el estado de ocupación de las partes más significativas de la Shared Pool.

5.4. Comprobar la contención en latches de la Shared Pool y Library Cache.

5.5. Comprobar el pinhitratio de la Library Caché, así como los reloads. Verificar el espacio libre de la Shared Pool, y el valor de open_cursors.

5.6. Ver el ratio de la Row Cache.

5.7. Comprobar si el sistema recomienda ampliar la SharedPool.

5.8. Detectar sentencias similares que usan literales. Verificar los parámetros cursor_sharing y session_cached_cursors. Asignar cursor_sharing=similar.

5.9. Instalar el paquete DBMS_SHARED_POOL. Comprobar paquetes que se pueden “fijar” en la SharedPool y hacerlo.

5.10. Comprobar sentencias que ocupan mucha memoria (>=10% de SharedPool).

5.11. Calcular el ratio de eficiencia de la Caché de Datos. Comprobar el parámetro db_cache_advice. Consultar si Oracle recomienda incrementar la Caché de Datos. Comprobar si hay contención en el latch “cache buffers lru chain”. Ver si hay esperas del tipo “write complete waits” o “free buffer waits”.

5.12. Comprobar el tamaño de la Cache de Redo. Ver si hay contención en los latches de redo. Verificar la estadística “redo log space requests”.

5.13. Comprobar los segmentos de rollback ONLINE. Verificar las extensiones que tiene cada uno, así como el espacio total y libre del tablespace que los contiene. Ver si hay contención en segmentos de rollback.

5.14. Instalar la utilidad STATSPACK. Crear snapshots y generar informe. Analizar el informe.

5.15. Probar autotrace de sql*plus.

5.16. Comprobar el plan de ejecución con el que se ha compilado una sentencia sql que ya se ha ejecutado, obteniendo los datos de la SHARED POOL (v$sql).

5.1. Revisar los eventos del sistema y comprobar los más significativos. Comprobar el evento “latch free” por sesiones. Comprobar sesiones esperando por el evento “db file sequential read”. Comprueba el parámetro TIMED_STATISTICS.

Solución:

```
SQL> set linesize 100
SQL> select rpad(event,30),total_waits,total_timeouts,time_waited,average_wait
     from v$system_event order by total_waits desc;
RPAD(EVENT,30)                 TOTAL_WAITS TOTAL_TIMEOUTS TIME_WAITED AVERAGE_WAIT
------------------------------ ----------- -------------- ----------- ------------
...
control file parallel write         141073              0       18513          .13
db file sequential read              30448              0       23774          .78
...
control file sequential read         14608              0         247          .02
log file parallel write               6563              0        3165          .48
db file scattered read                4386              0        8012         1.83
...
log file sequential read                24              0           2           .1
log file single write                   24              0           6          .26
log file switch completion              21              2         592        28.18
latch free                              21              0          39         1.84
...

(La columna “time_waited” está en centésimas de segundo)

SQL> select sid,total_waits,total_timeouts,time_waited,average_wait
      from v$session_event
      where event='latch free';
no rows selected

SQL> select event,sid,P1TEXT,P1,P2TEXT,P2,P3TEXT,P3,WAIT_TIME,SECONDS_IN_WAIT,STATE
      from v$session_wait
      where event like 'db file %';
no rows selected

SQL> show parameter timed_statistics
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
timed_statistics                     boolean     TRUE
```

Consulta la documentación para ver en qué consiste el evento “db file scattered read”. Lo puedes ver en “http://cursos.atica.um.es/oradoc102/server.102/b14237/waitevents003.htm#sthref4387”.

5.2. Revisa las estadísticas del sistema más significativas. Comprueba el tamaño medio de la PGA de cada sesión. Revisa las lecturas lógicas y físicas y calcula el ratio de E/S. Comparar el uso de CPU para “SQL del sistema” (acceso al DD) sobre el total.

Solución:

Las estadísticas del sistema se pueden consultar en V$SYSSTAT.

```
SQL> select name,value from v$sysstat;
NAME                                                                  VALUE
---------------------------------------------------------------- ----------
logons cumulative                                                       113
logons current                                                           15
opened cursors cumulative                                            222072
opened cursors current                                                   53
...
CPU used by this session                                               9526
...
db block gets                                                        727314
...
consistent gets                                                     5805950
...
physical reads                                                       134646...
...
```

Las estadísticas sobre el uso de la PGA nos permiten hacer estimaciones sobre el consumo de memoria (RAM) de cada sesión:

```
SQL> select name,value from v$sysstat where name like 'session pga memory%';
NAME                                                                  VALUE
---------------------------------------------------------------- ----------
session pga memory                                                 60340248
session pga memory max                                            101365784
```

Las estadísticas de E/S nos permiten calcular el ratio de eficiencia de la caché de datos. Podemos ver el porcentaje de lecturas a disco sobre las lecturas lógicas:

```
SQL> select 100*f1/(r1+r2)
      from (select value f1 from v$sysstat where name='physical reads'),
           (select value r1 from v$sysstat where name='consistent gets'),
           (select value r2 from v$sysstat where name='db block gets');

100*F1/(R1+R2)
--------------
    2.06093003
```

También podemos calcular el tiempo de CPU que se lleva el “SQL del sistema” (acceso al DD) respecto al tiempo total de CPU:

```
SQL> select 100*cpu1/cpu2
      from (select value cpu1 from v$sysstat where name='recursive cpu usage'),
           (select value cpu2 from v$sysstat where name='CPU used by this session');
100*CPU1/CPU2
-------------
    79.687172
```

5.3. Ver el estado de ocupación de las partes más significativas de la Shared Pool.

Solución:

La vista V$SGASTAT nos muestra el estado de las distintas partes de la Shared Pool.

```
SQL> select * from v$sgastat
      where name in ('free memory','library cache','sql area','dictionary cache');
POOL        NAME                         BYTES
----------- -------------------------- ----------
shared pool sql area                     3755132
shared pool free memory                   680920
shared pool library cache                2765552
```

```
shared pool dictionary cache              1610880
```

5.4. Comprobar la contención en latches de la Shared Pool y Library Cache.

Solución:

Los latches son microbloqueos necesarios para la gestión de los recursos compartidos del sistema. En la vista V$LATCH tenemos información sobre el uso de latches. La columna MISSES indica los fallos que hay al intentar coger un latch; y SLEEPS indica el nº de veces que el proceso que intenta coger el latch pasa al estado "dormido".

```
SQL> select rpad(name,30),gets,misses,sleeps
      from v$latch
      where name like '%library cache%' or
           name like '%shared pool%';
RPAD(NAME,30)                      GETS     MISSES     SLEEPS
------------------------------ ---------- ---------- ----------
shared pool                        974654        250          6
library cache pin allocation         2227          0          0
library cache lock allocation        5346          0          0
library cache hash chains               0          0          0
library cache lock                 768836          7          0
shared pool simulator              992390          0          0
library cache                     1361966        123          3
shared pool sim alloc                 116          0          0
library cache pin                  587213          2          0
library cache load lock             56515          0          0
```

Como hay fallos (misses) con esperas (sleeps) en los latch "shared pool" y "library cache", habría q subir el tamaño de la Shared Pool (shared_pool_size).

5.5. Comprobar el pinhitratio de la Library Caché, así como los reloads. Verificar el espacio libre de la Shared Pool, y el valor de open_cursors.

Solución:

Podemos ver las estadísticas de la Library Caché en la vista V$LIBRARYCACHE. Los espacios correspondientes a SQL y PLSQL son:

```
SQL> select namespace,pinhitratio,reloads from v$librarycache
      where namespace in ('SQL AREA','TABLE/PROCEDURE','BODY','TRIGGER');

NAMESPACE       PINHITRATIO    RELOADS
--------------- ----------- ----------
SQL AREA          .872040822      11301
TABLE/PROCEDURE   .681956796      12562
BODY              .811509591         22
TRIGGER           .142857143          3
```

Se recomienda un pinhitratio cercano a 1, y reloads casi cero. Llama la atención espacialmente el nº de reloads en "sql area" y "table/procedure"; así como el bajo ratio en "trigger". Podemos plantearnos subir algo el tamaño de la SharedPool. Primero habría que ver si hay espacio libre en la SharedPool, de forma continuada; y también revisar open_cursors.

```
SQL> select SharedPool, LibraryCache, Libre
     from (select sum(bytes) SharedPool from v$sgastat where pool='shared pool'),
     (select bytes LibraryCache from v$sgastat where name='library cache'),
```

```
     (select bytes Libre from v$sgastat where name='free memory' and pool='shared pool');
SHAREDPOOL LIBRARYCACHE      LIBRE
---------- ------------ ----------
  20971520      2793664     764464

SQL> show parameter open_cursors
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
open_cursors                         integer     50
```

5.6. Ver el ratio de la Row Cache.

Solución:

Podemos ver los ratios de la Row Caché en V$ROWCACHE, y también calcular un “ratio medio”..

```
SQL> SELECT rpad(parameter,25), sum(gets), sum(getmisses)
           , 100*sum(gets - getmisses) / sum(gets) pct_succ_gets
           , sum(modifications) updates
       FROM V$ROWCACHE WHERE gets > 0 GROUP BY parameter;
RPAD(PARAMETER,25)         SUM(GETS) SUM(GETMISSES) PCT_SUCC_GETS    UPDATES
------------------------- ---------- -------------- ------------- ----------
dc_constraints                   171             67    60.8187135        171
dc_tablespaces                 60613             72    99.8812136          0
dc_tablespace_quotas               1              1             0          1
dc_awr_control                  7454              5    99.9329219         44
dc_object_grants                  74             68    8.10810811          0
dc_histogram_data              73382          21633    70.5200185       2594
dc_rollback_segments           33258              9    99.9729388         17
dc_sequences                      46             41    10.8695652         46
dc_usernames                    1298             47    96.3790447          0
dc_segments                    30725          14028    54.3433686        255
dc_objects                     89868           5463    93.9210843        334
dc_histogram_defs             106533          51025    52.1040429       3090
dc_table_scns                     12             12             0          0
dc_users                      102206            107    99.8953095          0
outstanding_alerts                 3              1    66.6666667          2
dc_object_ids                 138501           6718    95.1494935        133
dc_global_oids                  2081            311    85.0552619          0
dc_profiles                       33             10    69.6969697          0
18 rows selected.

SQL> SELECT SUM(GETS - GETMISSES - FIXED) * 100 / SUM(GETS) "Ratio Medio"
      FROM V$ROWCACHE;
Ratio Medio
-----------
 84.5682021
```

La recomendación es que dicho ratio ser >= 85%.

5.7. Comprobar si el sistema recomienda ampliar la SharedPool.

Solución:

Las estadísticas sobre posibles redimensionamientos de la SharedPool los podemos ver en V$SHARED_POOL_ADVICE.

```
SQL> select SHARED_POOL_SIZE_FOR_ESTIMATE SIZE_ESTIMATE,
      SHARED_POOL_SIZE_FACTOR SIZE_FACTOR,
      ESTD_LC_TIME_SAVED_FACTOR PARSE_SAVED_FACTOR
```

```
      from v$shared_pool_advice;
SIZE_ESTIMATE SIZE_FACTOR PARSE_SAVED_FACTOR
------------- ----------- ------------------
           40       .8333              .8568
           48           1                 1
           56      1.1667             1.046
           64      1.3333            1.1486
           72         1.5            1.1676
           80      1.6667            1.1691
           88      1.8333            1.1817
           96           2            1.1817
```

Según los datos anteriores la ganancia por subir la SharedPool de 48 a 64M (+33%) sería del 14.8% (de 1 a 1.1486), así q probaríamos con shared_pool_size=64M.

5.8. Detectar sentencias similares que usan literales. Verificar el parámetros cursor_sharing y session_cached_cursors. Asignar cursor_sharing=similar.

Solución:

Para detectar sentencias similares miraremos en V$SQLAREA:

```
SQL> SELECT substr(sql_text,1,40) "SQL", count(*), sum(executions) "TotExecs"
     FROM v$sqlarea
     GROUP BY substr(sql_text,1,40) HAVING count(*) > 5 ORDER BY 2;
SQL                                      COUNT(*)   TotExecs
---------------------------------------- ---------- ----------
SELECT substr(sql_text,1,40) "SQL", coun          7          9

SQL>  select  sql_text  from  v$sqlarea  where  substr(sql_text,1,40)  like  'SELECT
substr(sql_text,1,40) %';
SQL_TEXT
--------------------------------------------------------------------------------
SELECT  substr(sql_text,1,40)  "SQL",  count(*),  sum(executions)  "TotExecs"          FROM
v$sqlarea      GR
OUP BY substr(sql_text,1,40) HAVING count(*) > 20 ORDER BY 2

SELECT  substr(sql_text,1,40)  "SQL",  count(*),  sum(executions)  "TotExecs"          FROM
v$sqlarea      WH
ERE executions < 5      GROUP BY substr(sql_text,1,40) HAVING count(*) > 20 ORDER BY 2

SELECT  substr(sql_text,1,40)  "SQL",  count(*),  sum(executions)  "TotExecs"          FROM
v$sqlarea      GR
OUP BY substr(sql_text,1,40) HAVING count(*) > 10 ORDER BY 2
...

SQL> show parameter cursor_sharing
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
cursor_sharing                       string      EXACT

SQL> show parameter session_cached_cursors
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
session_cached_cursors               integer     20
SQL> alter system set cursor_sharing=similar;
System altered.
```

5.9. Instalar el paquete DBMS_SHARED_POOL. Comprobar paquetes que se pueden "fijar" en la SharedPool y hacerlo.

El paquete dbms_shared_pool se instala con el script $ORACLE_HOME/rdbms/admin/dbmspool.sql, bajo el usuario SYS.

Solución:

Instalar el paquete.

```
SQL> connect / as sysdba
SQL> @$ORACLE_HOME/rdbms/admin/dbmspool.sql
```

Comprobar paquetes a fijar, por ejemplo q ocupen más de 100Kb.

```
SQL> set serveroutput on size 2000
SQL> exec dbms_shared_pool.sizes(100)
SIZE(K) KEPT   NAME
------- ------ ----------------------------------------------------------------
429        SYS.STANDARD                (PACKAGE)
```

Fijarlo.

```
SQL> exec dbms_shared_pool.keep('SYS.STANDARD');
PL/SQL procedure successfully completed.
```

Comprobarlo:

```
SQL> exec dbms_shared_pool.sizes(100)
SIZE(K) KEPT   NAME
------- ------ ----------------------------------------------------------------
429 YES    SYS.STANDARD                (PACKAGE)
```

5.10. Comprobar sentencias que ocupan mucha memoria (>=10% de SharedPool).

Solución:

En V$SQL tenemos información sobre las sentencias SQL en ejecución, incluyendo el consumo de memoria. Nos interesa controlar aquellas que ocupan el 10% o más de la Shared Pool.

```
SQL> show parameter shared_pool_size
NAME                 TYPE                            VALUE
-------------------- ------------------------------- ------------------------------
shared_pool_size     big integer                     0

SQL> select * from v$sgainfo;
NAME                                  BYTES RES
------------------------------ ---------- ---
Fixed SGA Size                     1259408 No
Redo Buffers                       2932736 No
Buffer Cache Size                 33554432 Yes
Shared Pool Size                  50331648 Yes
...
Free SGA Memory Available          8388608

SQL> selecT substr(sql_text,1,40) "Stmt", count(*),
                     sum(sharable_mem)    "Mem",
                    sum(users_opening)   "Open",
```

```
                sum(executions)      "Exec"
             FROM v$sql
            GROUP BY substr(sql_text,1,40)
           HAVING sum(sharable_mem) > 5000000;
no rows selected
```

No hay ninguna (si ponemos 50000, en lugar de 5000000, veremos q sale algo).

5.11. Calcular el ratio de eficiencia de la Caché de Datos. Comprobar el parámetro db_cache_advice. Consultar si Oracle recomienda incrementar la Caché de Datos. Comprobar si hay contención en el latch “cache buffers lru chain”. Ver si hay esperas del tipo “write complete waits” o “free buffer waits”.

Solución:

Lo calcularemos a partir de los datos de lecturas físicas y lógicas que podemos ver en V$SYSSTAT.

```
SQL>   select 100*(1 - (f1 - f2 - f3)/(r1 + r2 - f2 -f3)) Ratio
        from (select value f1 from v$sysstat where name='physical reads'),
             (select value f2 from v$sysstat where name='physical reads direct'),
             (select value f3 from v$sysstat where name='physical reads direct (lob)'),
             (select value r1 from v$sysstat where name='consistent gets'),
             (select value r2 from v$sysstat where name='db block gets');
     RATIO
----------
98.0134287
```

Las recomendaciones sobre incrementos de tamaño de la Caché de Datos los podemos ver en V$DB_CACHE_ADVICE, siempre que el parámetro db_cache_advice=on.

```
SQL> show parameter db_cache_advice
NAME                 TYPE                              VALUE
-------------------- --------------------------------- ------------------------------
db_cache_advice      string                            ON

SQL> set linesize 100
SQL> select id,name,size_for_estimate,size_factor,ESTD_PHYSICAL_READ_FACTOR
       from v$db_cache_advice;
         ID NAME                 SIZE_FOR_ESTIMATE SIZE_FACTOR ESTD_PHYSICAL_READ_FACTOR
---------- -------------------- ----------------- ----------- -------------------------
...
          3 DEFAULT                             24         .75                    1.0056
          3 DEFAULT                             28        .875                    1.0033
          3 DEFAULT                             32           1                         1
          3 DEFAULT                             36       1.125                     .9986
          3 DEFAULT                             40        1.25                     .9976
          3 DEFAULT                             44       1.375                      .997
          3 DEFAULT                             48         1.5                     .9957
          3 DEFAULT                             52       1.625                      .995
          3 DEFAULT                             56        1.75                     .9943
          3 DEFAULT                             60       1.875                     .9929
          3 DEFAULT                             64           2                     .9433
...
```

Como se observa, en este caso no merece la pena subir la caché, pues tenemos q duplicar su

tamaño (64Mb), para conseguir una mejora del 6%. Incluso podríamos reducirla a 28M, pues sólo empeoramos el 0.3%.

En la vista V$LATCH puedo ver si hay contención en “cache buffers lru chain”.

```
SQL> SELECT NAME,GETS,MISSES,SLEEPS FROM V$LATCH WHERE NAME='cache buffers lru chain';
NAME                                     GETS       MISSES     SLEEPS
---------------------------------------- ---------- ---------- ----------
cache buffers lru chain                      341187        154          3
```

En la vista V$BUFFER_POOL_STATISTICS puedo ver si hay esperas del tipo “free buffer wait” o “write complete wait”.

```
SQL> SELECT ID, NAME, FREE_BUFFER_WAIT, WRITE_COMPLETE_WAIT
       FROM V$BUFFER_POOL_STATISTICS;
        ID NAME                 FREE_BUFFER_WAIT WRITE_COMPLETE_WAIT
---------- -------------------- ---------------- --------------------
         3 DEFAULT                             0                    0
```

5.12. Comprobar el tamaño de la Cache de Redo. Ver si hay contención en los latches de redo. Verificar la estadística “redo log space requests”.

Solución:

El tamaño de la Caché de Redo lo determina el parámetro log_buffer.

```
SQL> show parameter log_buffer
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
log_buffer                           integer     2886656
```

Tendremos contención en los latches de redo si alguno de los siguientes ratios supera el 1%:

```
SQL> SELECT substr(name,1,20) latch,
           100*decode(gets, 0, 0, misses/gets) mis_ratio,
           100*decode(immediate_gets + immediate_misses,0,0,
                      immediate_misses/(immediate_gets + immediate_misses)) im_mis_ratio
       FROM v$latch
      WHERE name in ('redo allocation', 'redo copy');
LATCH                MIS_RATIO IM_MIS_RATIO
-------------------- ---------- ------------
redo copy                     0   .245571834
redo allocation       .00722064   .005709181
```

Finalmente, comprobemos la estadística “redo log space requests”:

```
SQL> select  * from v$sysstat where name='redo log space requests';
STATISTIC# NAME                                                  CLASS      VALUE
---------- ---------------------------------------------------- ---------- ----------
       141 redo log space requests                                       2         25
```

5.13. Comprobar los segmentos de rollback ONLINE. Verificar las extensiones que tiene cada uno, así como el espacio total y libre del tablespace que los contiene. Ver si hay contención en segmentos de rollback.

Solución:

En la vista DBA_ROLLBACK_SEGS podemos ver los segmentos de rollback, si están en línea, y a qué tablespace pertenecen. En V$ROLLSTAT (y V$ROLLNAME), podemos ver estadísticas de

tamaño, peticiones, extensión dinámica, etc.

```
SQL> select segment_name, tablespace_name, initial_extent,
            min_extents, max_extents,status
      from dba_rollback_segs;
SEGMENT_NAME                   TABLESPACE INITIAL_EXTENT MIN_EXTENTS MAX_EXTENTS STATUS
------------------------------ ---------- -------------- ----------- ----------- ------
SYSTEM                         SYSTEM             102400           1       32765 ONLINE
_SYSSMU1$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU2$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU3$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU4$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU5$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU6$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU7$                      UNDO_RBS           131072           2       32765 ONLINE
_SYSSMU8$                      UNDO_RBS           131072           2       32765 ONLINE

SQL> SELECT RPAD(NAME,10) NAME,EXTENTS,RSSIZE,GETS,WAITS,SHRINKS,EXTENDS,STATUS
       FROM V$ROLLNAME A, V$ROLLSTAT B
      WHERE A.USN=B.USN;
NAME          EXTENTS     RSSIZE      GETS      WAITS    SHRINKS    EXTENDS STATUS
------------ ---------- --------- ---------- ---------- ---------- ------- --------
SYSTEM              7     456704       1479          0          0       0 ONLINE
_SYSSMU1$           3    1177600       6722          0          3       3 ONLINE
_SYSSMU2$           2     129024       7714          1         11      32 ONLINE
_SYSSMU3$           2     129024       9242          0          4      12 ONLINE
_SYSSMU4$           3    1177600       6893          0          3       3 ONLINE
_SYSSMU5$           2     129024       6647          0          2       1 ONLINE
_SYSSMU6$           2     129024       9983          0          7      36 ONLINE
_SYSSMU7$           2     129024       7277          0          2       2 ONLINE
_SYSSMU8$           3    1177600       6494          0          2      11 ONLINE
```

Podemos consultar las extensiones y espacio ocupado de cada RS, en la vista DBA_EXTENTS; y el espacio libre del tablespace que los contiene, en DBA_FREE_SPACE.

```
SQL> SELECT rpad(segment_name,10) Rsname,count(*),sum(bytes)
      FROM DBA_EXTENTS WHERE TABLESPACE_NAME='UNDO_RBS' group by segment_name;
RSNAME      COUNT(*) SUM(BYTES)
---------- ---------- ----------
_SYSSMU5$          2     131072
_SYSSMU3$          2     131072
_SYSSMU2$          2     131072
_SYSSMU6$          2     131072
_SYSSMU1$          3    1179648
_SYSSMU4$          3    1179648
_SYSSMU8$          3    1179648
_SYSSMU7$          2     131072

SQL> select sum(bytes) from dba_free_space where tablespace_name='UNDO_RBS';
SUM(BYTES)
----------
  16711680
```

Finalmente, las estadísticas sobre contención en RS, las podemos ver en V$WAITSTAT. Si la hubiese en “undo header” tendríamos que crear más RS.

```
SQL> SELECT CLASS, COUNT FROM V$WAITSTAT
      WHERE CLASS like '%undo%';
CLASS                   COUNT
------------------ ----------
save undo block             0
save undo header            0
system undo header          0
system undo block           0
undo header                 1
undo block                  0
```

5.14. Instalar la utilidad STATSPACK. Crear snapshots y generar informe.

Solución:

Crear tablespace statspack_tsp con 100M en /u05/oradata/CURSOxy (crear dir. /u05/oradata/CURSOxy).
Crear tablespace statspack_temp con 2M en /u05/oradata/CURSOxy.
Instalar statspack con $ORACLE_HOME/rdbms/admin/spcreate.sql, indicando los datos que se piden: clave de usuario perfstat, tablespace por defecto, y tablespace temporal por defecto.

```
$ mkdir /u05/oradata/CURSOxy
$ chmod -R g+w /u05/oradata/CURSOxy


SQL> create tablespace statpack_tsp
      datafile '/u05/oradata/CURSOxy/statpack_tsp01.dbf' size 100M
      autoextend on next 10M maxsize 200M
      extent management local autoallocate
      segment space management auto;
Tablespace created.

SQL> create temporary tablespace statpack_temp
      tempfile '/u05/oradata/CURSOxy/statpack_temp01.dbf' size 2M
      autoextend on next 1M maxsize 10M;
Tablespace created.

SQL> @$ORACLE_HOME/rdbms/admin/spcreate.sql
... Installing Required Packages
...
... Creating PERFSTAT user ...
Choose the PERFSTAT user's password.
Not specifying a password will result in the installation FAILING
Specify PERFSTAT password
Enter value for perfstat_password: perfcursoXY
...
Choose the Default tablespace for the PERFSTAT user
---------------------------------------------------
...
Specifying the SYSTEM tablespace will result in the installation
FAILING, as using SYSTEM for performance data is not supported.

TABLESPACE_NAME                CONTENTS  STATSPACK DEFAULT TABLESPACE
------------------------------ --------- ----------------------------
SEGAUTO                        PERMANENT
STATPACK_TSP                   PERMANENT
SYSAUX                         PERMANENT *
TSP4K                          PERMANENT
USERS                          PERMANENT

Pressing <return> will result in STATSPACK's recommended default
tablespace (identified by *) being used.

Enter value for default_tablespace: STATPACK_TSP
...
Choose the PERFSTAT user's temporary tablespace.

Specifying the SYSTEM tablespace will result in the installation
FAILING, as using SYSTEM for the temporary tablespace is not recommended.
...
Specify PERFSTAT user's temporary tablespace.
Enter value for temporary_tablespace: STATPACK_TEMP
```

```
...
No errors.
NOTE:
SPCPKG complete. Please check spcpkg.lis for any errors.
```

Una vez instalado STATSACK, comprobamos si ha habido errores (en los ficheros *.lis):

```
SQL> !grep -i err *.lis
spcpkg.lis:No errors.
spcpkg.lis:No errors.
spcpkg.lis:SPCPKG complete. Please check spcpkg.lis for any errors.
spctab.lis:SPCTAB complete. Please check spctab.lis for any errors.
spcusr.lis:SPCUSR complete. Please check spcusr.lis for any errors.
```

Ahora podemos utilizar el paquete STATSPACK.
Primero compruebo si está activada la recolección de estadísticas:

```
SQL> show parameter timed_statistics
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
timed_statistics                     boolean     TRUE
```

Ahora me conecto como PERFSTAT (o usuario con acceso al paquete STATSPACK) y genero el primer snapshot (baseline) con nivel 10 (i_snap_level). Después, espero al menos 5 minutos para generar otro snapshot, pues para sacar un informe necesito al menos dos:

```
SQL> connect perfstat
Enter password:
Connected.

SQL> exec STATSPACK.SNAP(i_snap_level=>10)
PL/SQL procedure successfully completed.
(... LANZAR ALGUNAS CONSULTAS, COMO LAS DOS SIGUIENTES ...)
SQL> SELECT COUNT(*) FROM DBA_OBJECTS;
  COUNT(*)
----------
      5924
SQL> SELECT COUNT(*) FROM DBA_EXTENTS;
  COUNT(*)
----------
      2647
(... ESPERAR 5 MINUTOS DESDE QUE HICIMOS EL SNAPSHOT...)
SQL> exec STATSPACK.SNAP
PL/SQL procedure successfully completed.
```

Ahora ya podemos generar el informe:

```
SQL> @$ORACLE_HOME/rdbms/admin/spreport.sql
...
Enter value for begin_snap: 1
...
Enter value for end_snap: 2
...
Snapshot       Snap Id     Snap Time      Sessions Curs/Sess Comment
~~~~~~~~    ---------- ------------------ -------- --------- -------------------
Begin Snap:          1 29-Jan-07 13:15:56       15       5.4
  End Snap:          2 29-Jan-07 13:17:40       15       6.8
   Elapsed:                1.73 (mins)

Cache Sizes                       Begin        End
~~~~~~~~~~~                  ---------- ----------
             Buffer Cache:        28M             Std Block Size:         2K
         Shared Pool Size:        52M
     Log Buffer:                   2,819K
```

```
Load Profile                            Per Second       Per Transaction
~~~~~~~~~~~~                       ---------------       ---------------
                  Redo size:             17,111.12          1,779,556.00
              Logical reads:                352.43             36,653.00
              Block changes:                 64.20              6,677.00
             Physical reads:                 35.24              3,665.00
            Physical writes:                 31.09              3,233.00
                 User calls:                  0.10                 10.00
                     Parses:                 13.85              1,440.00
                Hard parses:                  1.34                139.00
                      Sorts:                  6.13                637.00
                     Logons:                  0.00                  0.00
                   Executes:                 29.43              3,061.00
               Transactions:                  0.01

  % Blocks changed per Read:   18.22    Recursive Call %:                99.98
 Rollback per transaction %:    0.00       Rows per Sort:                12.03

Instance Efficiency Percentages
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
            Buffer Nowait %:  100.00       Redo NoWait %:   99.95
            Buffer  Hit   %:   89.99    In-memory Sort %:  100.00
            Library Hit   %:   83.66        Soft Parse %:   90.35
         Execute to Parse %:   52.96         Latch Hit %:   99.99
Parse CPU to Parse Elapsd %:   50.98     % Non-Parse CPU:   61.48

 Shared Pool Statistics        Begin   End
                               ------  ------
             Memory Usage %:   83.22   84.28
    % SQL with executions>1:   72.27   87.15
  % Memory for SQL w/exec>1:   66.21   78.80

Top 5 Timed Events                                                    Avg %Total
~~~~~~~~~~~~~~~~~~                                                   wait   Call
Event                                            Waits    Time (s)   (ms)   Time
----------------------------------------- ------------ ----------- ------ ------
db file sequential read                          1,387           6      4   74.5
CPU time                                                         1          16.4
db file scattered read                             107           0      2    3.1
log file parallel write                              9           0     24    2.6
log file switch completion                           2           0     55    1.3
...

End of Report ( sp_1_2.lst )
```

5.15. Probar autotrace de sqlplus.

Solución:

AUTOTRACE es una opción de sql*plus, que nos permite ver el plan de ejecución de una sentencia sql y/o las estadísticas asociadas a la misma:

- Plan de ejecución: nos dice cómo se va a ejecutar la sentencia (qué indices se usan o si se accede a la tabla completa, etc).
- Estadísticas: proporcionan datos como las lecturas lógicas y físicas necesarias para extraer los datos.

Más información sobre autotrace en "http://download.oracle.com/docs/cd/B19306_01/server.102/b14357/ch8.htm#i1037226".

```
SQL> connect system/systcursoXY
Enter password:
Connected.

(Ahora vamos a activar AUTORACE, de modo que sólo muestre el plan de ejecución de las
sentencias sql que lancemos posteriormente)
SQL> set autotrace traceonly explain
SQL> set linesize 100
SQL> select count(*) from dba_objects where status='VALID';
Execution Plan
----------------------------------------------------------

-------------------------------------------------------------------------------------------
| Id  | Operation                          | Name        | Rows  | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |             |     1 |     5 |    86   (2)|
|   1 |  SORT AGGREGATE                    |             |     1 |     5 |            |
|   2 |   VIEW                             | DBA_OBJECTS |    89 |   445 |    86   (2)|
|   3 |    UNION-ALL                       |             |       |       |            |
|*  4 |     FILTER                         |             |       |       |            |
|*  5 |      HASH JOIN                     |             |    99 |  7623 |    85   (2)|
|   6 |       TABLE ACCESS FULL            | USER$       |    27 |    81 |     3   (0)|
|*  7 |       TABLE ACCESS FULL            | OBJ$        |    99 |  7326 |    82   (2)|
|*  8 |      TABLE ACCESS BY INDEX ROWID   | IND$        |     1 |     7 |     2   (0)|
|*  9 |       INDEX UNIQUE SCAN            | I_IND1      |     1 |       |     1   (0)|
|  10 |     NESTED LOOPS                   |             |     1 |    16 |     1   (0)|
|  11 |      INDEX FULL SCAN               | I_LINK1     |     1 |    13 |     0   (0)|
|  12 |      TABLE ACCESS CLUSTER          | USER$       |     1 |     3 |     1   (0)|
|* 13 |       INDEX UNIQUE SCAN            | I_USER#     |     1 |       |     0   (0)|
-------------------------------------------------------------------------------------------
...

(Por defecto, el optimizador seleccionará el mejor plan de ejecución para obtener TODAS
las filas que puede devolver una consulta (ALL_ROWS). Ahora vamos a decirle que queremos
que escoja el mejor plan de ejecución para obtener pronto algún resultado. o las PRIMERAS
FILAS (FIRST_ROWS))
SQL> alter session set optimizer_mode=first_rows;
Session altered.

SQL> select count(*) from dba_objects where status='VALID';
Execution Plan
----------------------------------------------------------

-------------------------------------------------------------------------------------------
| Id  | Operation                          | Name        | Rows  | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |             |     1 |     5 |   182   (1)|
|   1 |  SORT AGGREGATE                    |             |     1 |     5 |            |
|   2 |   VIEW                             | DBA_OBJECTS |    89 |   445 |   182   (1)|
|   3 |    UNION-ALL                       |             |       |       |            |
|*  4 |     FILTER                         |             |       |       |            |
|   5 |      NESTED LOOPS                  |             |    99 |  7623 |   181   (1)|
|*  6 |       TABLE ACCESS FULL            | OBJ$        |    99 |  7326 |    82   (2)|
|   7 |       TABLE ACCESS CLUSTER         | USER$       |     1 |     3 |     1   (0)|
|*  8 |        INDEX UNIQUE SCAN           | I_USER#     |     1 |       |     0   (0)|
|*  9 |      TABLE ACCESS BY INDEX ROWID   | IND$        |     1 |     7 |     2   (0)|
|* 10 |       INDEX UNIQUE SCAN            | I_IND1      |     1 |       |     1   (0)|
|  11 |     NESTED LOOPS                   |             |     1 |    16 |     1   (0)|
|  12 |      INDEX FULL SCAN               | I_LINK1     |     1 |    13 |     0   (0)|
|  13 |      TABLE ACCESS CLUSTER          | USER$       |     1 |     3 |     1   (0)|
|* 14 |       INDEX UNIQUE SCAN            | I_USER#     |     1 |       |     0   (0)|
-------------------------------------------------------------------------------------------
...
```

```
(Ahora queremos ver sólo las estadísticas de acceso a disco)
SQL> set autotrace traceonly statistics
SQL> select count(*) from dba_objects where status='VALID';
Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
      22992  consistent gets
          0  physical reads
          0  redo size
        412  bytes sent via SQL*Net to client
        384  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          1  rows processed
```

Ya hemos visto como ver de forma automática el plan de ejecución de una sentencia, así como las estadísticas de E/S.
También podemos calcular el tiempo de repuesta al usuario con SET TIMING:

```
SQL> set autotrace off
SQL> set timing on
SQL> select count(*) from dba_objects where status='VALID';
  COUNT(*)
----------
      9932
Elapsed: 00:00:00.07
```

5.16. Comprobar el plan de ejecución con el que se ha compilado una sentencia sql que ya se ha ejecutado, obteniendo los datos de la SHARED POOL (v$sql).

Solución:

En V$SQL puedo ver las sentencias sql que ya se han ejecutado, de modo que puedo obtener el identificador (V$SQL.SQL_ID) de cualquier sentencia sql, para posteriormente consultar el plan de ejecución haciendo uso del paquete dbms_xplan (DBMS_XPLAN.DISPLAY_CURSOR(SQL_ID)).

```
SQL> connect system/systcursoXY
Enter password:
Connected.

SQL> select deptno,count(*) from scott.emp group by deptno;

    DEPTNO   COUNT(*)
---------- ----------
        30          6
        20          5
        10          3

SQL> select sql_id from v$sql where sql_text='select deptno,count(*) from scott.emp group
by deptno';

SQL_ID
-------------
cxwqkgckrcpd8


SQL> select * from table(dbms_xplan.display_cursor('cxwqkgckrcpd8'));
```

```
PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------
SQL_ID  cxwqkgckrcpd8, child number 0
-------------------------------------
select deptno,count(*) from scott.emp group by deptno

Plan hash value: 4067220884

-----------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |     7 (100)|          |
|   1 |  HASH GROUP BY     |      |     3 |     9 |     7  (15)| 00:00:01 |
|   2 |   TABLE ACCESS FULL| EMP  |    14 |    42 |     6   (0)| 00:00:01 |
-----------------------------------------------------------------------------


14 filas seleccionadas.
```