Optimize Oracle UNDO Parameters

Overview

Starting in Oracle9i, rollback segments are re-named undo logs. Traditionally transaction undo information was stored in Rollback Segments until a commit or rollback statement was issued, at which point it was made available for overlaying.

Best of all, automatic undo management allows the DBA **to specify how long undo information should be retained after commit**, preventing "snapshot too old" errors on long running queries.

This is done by setting the *UNDO_RETENTION* parameter. The default is 900 seconds (5 minutes), and you can set this parameter to guarantee that Oracle keeps undo logs for extended periods of time.

Rather than having to define and manage rollback segments, you can simply define an Undo tablespace and let Oracle take care of the rest. Turning on automatic undo management is easy. All you need to do is create an undo tablespace and set *UNDO_MANAGEMENT = AUTO.*

However it is worth to tune the following important parameters

1. The size of the UNDO tablespace
2. The UNDO_RETENTION parameter

Calculate UNDO_RETENTION for given UNDO Tabespace

You can choose to allocate a specific size for the UNDO tablespace and then set the UNDO_RETENTION parameter to an optimal value according to the UNDO size and the database activity. If your disk space is limited and you do not want to allocate more space than necessary to the UNDO tablespace, this is the way to proceed. The following query will help you to optimize the UNDO_RETENTION parameter:

$$\text{OPTIMAL UNDO RETENTION} = \frac{\text{ACTUAL UNDO SIZE}}{\text{DB\_BLOCK\_SIZE} * \text{UNDO\_BLOCK\_PER\_SEC}}$$

Because these following queries use the V$UNDOSTAT statistics, run the queries only after the database has been running with UNDO for a significant and representative time!

Actual Undo Size

```
SELECT SUM(a.bytes) "UNDO_SIZE"
  FROM v$datafile a,
       v$tablespace b,
       dba_tablespaces c
 WHERE c.contents = 'UNDO'
   AND c.status = 'ONLINE'
   AND b.name = c.tablespace_name
   AND a.ts# = b.ts#;

 UNDO_SIZE
----------
  209715200
```

Undo Blocks per Second

```
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
      "UNDO_BLOCK_PER_SEC"
  FROM v$undostat;

UNDO_BLOCK_PER_SEC
------------------
        3.12166667
```

DB Block Size

```
SELECT TO_NUMBER(value) "DB_BLOCK_SIZE [KByte]"
 FROM v$parameter
WHERE name = 'db_block_size';

DB_BLOCK_SIZE [Byte]
--------------------
                4096
```

Optimal Undo Retention

209'715'200 / (3.12166667 * 4'096) = 16'401 [Sec]

Using Inline Views, you can do all in one query!

```
SELECT d.undo_size/(1024*1024) "ACTUAL UNDO SIZE [MByte]",
       SUBSTR(e.value,1,25) "UNDO RETENTION [Sec]",
       ROUND((d.undo_size / (to_number(f.value) *
       g.undo_block_per_sec))) "OPTIMAL UNDO RETENTION [Sec]"
  FROM (
       SELECT SUM(a.bytes) undo_size
          FROM v$datafile a,
               v$tablespace b,
               dba_tablespaces c
         WHERE c.contents = 'UNDO'
           AND c.status = 'ONLINE'
           AND b.name = c.tablespace_name
           AND a.ts# = b.ts#
       ) d,
       v$parameter e,
       v$parameter f,
       (
       SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
              undo_block_per_sec
         FROM v$undostat
       ) g
WHERE e.name = 'undo_retention'
  AND f.name = 'db_block_size'
/

ACTUAL UNDO SIZE [MByte]
------------------------
200

UNDO RETENTION [Sec]
--------------------
10800

OPTIMAL UNDO RETENTION [Sec]
----------------------------
16401
```

Calculate Needed UNDO Size for given Database Activity

If you are not limited by disk space, then it would be better to choose the UNDO_RETENTION time that is best for you (for FLASHBACK, etc.). Allocate the appropriate size to the UNDO tablespace according to the database activity:

```
UNDO SIZE = UNDO RETENTION * DB_BLOCK_SIZE * UNDO_BLOCK_PER_SEC
```

Again, all in one query:

```
SELECT d.undo_size/(1024*1024) "ACTUAL UNDO SIZE [MByte]",
       SUBSTR(e.value,1,25) "UNDO RETENTION [Sec]",
       (TO_NUMBER(e.value) * TO_NUMBER(f.value) *
       g.undo_block_per_sec) / (1024*1024)
      "NEEDED UNDO SIZE [MByte]"
  FROM (
       SELECT SUM(a.bytes) undo_size
         FROM v$datafile a,
              v$tablespace b,
              dba_tablespaces c
        WHERE c.contents = 'UNDO'
          AND c.status = 'ONLINE'
          AND b.name = c.tablespace_name
          AND a.ts# = b.ts#
       ) d,
      v$parameter e,
       v$parameter f,
       (
       SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
         undo_block_per_sec
         FROM v$undostat
       ) g
 WHERE e.name = 'undo_retention'
  AND f.name = 'db_block_size'
/

ACTUAL UNDO SIZE [MByte]
------------------------
200
UNDO RETENTION [Sec]
--------------------
10800
NEEDED UNDO SIZE [MByte]
------------------------
131.695313
```

The previous query may return a "NEEDED UNDO SIZE" that is less than the "ACTUAL UNDO SIZE". If this is the case, you may be wasting space. You can choose to resize your UNDO tablespace to a lesser value or increase your UNDO_RETENTION parameter to use the additional space.