ORACLE 11G SECURE FILE LOBS PART 3 – STORAGE IMPROVEMENTS

Inderpal S. Johal, Data Softech Inc.

INTRODUCTION

Oracle 11g has re-architectured the LOB Storage format so as to provide performance improvement, optimized space utilization and high security. Don't think that LOBs types are changed as it is still CLOB, BLOB and BFILE but Oracle has improved the storage and it will discussed in this and coming paper in detail.

COMPARE THE SPACE UTILIZATION OF BASICFILE VS SECUREFILE LOBS

In order to check the performance of two types of LOBs format, we will perform the following

1. Create the Procedure to create Space Report
2. Execute the Report against BASICFILE Lob where we load DIFFERENT data 5 times.
3. Execute the Report against SECUREFILE Lob where we load DIFFERENT data 5 times
4. Execute the Report against BASICFILE Lob where we will load SAME data 5 times.
5. Execute the Report against SECUREFILE lob where we will load SAME data 5 times
6. Create Two comparison Report

Step 1 : Create the procedure to provide the Space utilization Report for LOBs storage

DBMS_USAGE package has two SPACE_USAGE procedures. The procedure used below is specifically meant for BASICFILE Lobs.

```
SQL> CREATE OR REPLACE PROCEDURE check_space_bf
    IS
   l_fs1_bytes NUMBER;
   l_fs2_bytes NUMBER;
   l_fs3_bytes NUMBER;
   l_fs4_bytes NUMBER;
   l_fs1_blocks NUMBER;
   l_fs2_blocks NUMBER;
   l_fs3_blocks NUMBER;
   l_fs4_blocks NUMBER;
   l_full_bytes NUMBER;
   l_full_blocks NUMBER;
   l_unformatted_bytes NUMBER;
   l_unformatted_blocks NUMBER;
   v_segname VARCHAR2(500);
BEGIN
   SELECT segment_name
   INTO v_segname
   FROM user_lobs
   WHERE table_name = 'RESUMES'
   AND column_name = 'RESUME';

   DBMS_SPACE.SPACE_USAGE(
      segment_owner      => 'SF_DEMO',
      segment_name       => v_segname,
      segment_type       => 'LOB',
      fs1_bytes          => l_fs1_bytes,
      fs1_blocks         => l_fs1_blocks,
      fs2_bytes          => l_fs2_bytes,
      fs2_blocks         => l_fs2_blocks,
      fs3_bytes          => l_fs3_bytes,
      fs3_blocks         => l_fs3_blocks,
      fs4_bytes          => l_fs4_bytes,
      fs4_blocks         => l_fs4_blocks,
      full_bytes         => l_full_bytes,
      full_blocks        => l_full_blocks,
      unformatted_blocks => l_unformatted_blocks,
      unformatted_bytes  => l_unformatted_bytes
   );
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' FS1 Blocks = '||l_fs1_blocks||' Bytes = '||l_fs1_bytes);
   DBMS_OUTPUT.PUT_LINE(' FS2 Blocks = '||l_fs2_blocks||' Bytes = '||l_fs2_bytes);
   DBMS_OUTPUT.PUT_LINE(' FS3 Blocks = '||l_fs3_blocks||' Bytes = '||l_fs3_bytes);
   DBMS_OUTPUT.PUT_LINE(' FS4 Blocks = '||l_fs4_blocks||' Bytes = '||l_fs4_bytes);
   DBMS_OUTPUT.PUT_LINE('Full Blocks = '||l_full_blocks||' Bytes = '||l_full_bytes);
   DBMS_OUTPUT.PUT_LINE('Unformatted Blocks = '||l_unformatted_blocks||' Bytes = '||l_unformatted_bytes);
   DBMS_OUTPUT.PUT_LINE('=============================================');
   DBMS_OUTPUT.PUT_LINE('Total Blocks = '||to_char(l_fs1_blocks + l_fs2_blocks +
                            l_fs3_blocks + l_fs4_blocks + l_full_blocks)||  ' ||  Total Bytes = '||
                            to_char(l_fs1_bytes + l_fs2_bytes +
                            l_fs3_bytes + l_fs4_bytes + l_full_bytes));
END;
/
Procedure created.
```

DATA SOFTECH INC.
Complete Database Solution Provider

Step 2 : Check the Space Usage for BINARYFILE LOBs Table

```
SQL> conn sf_demo/oracle
Connected.
SQL> drop table resumes;
Table dropped.

SQL> CREATE TABLE resumes
  2  (id NUMBER, first_name VARCHAR2(15),
  3  last_name VARCHAR2(40), resume BLOB)
  4  LOB(resume) STORE AS BASICFILE
  5  (TABLESPACE tbs2)
  6  /
Table created.

SQL> timing start load_data
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy2.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy3.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy4.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy5.doc');
PL/SQL procedure successfully completed.

SQL> timing stop
timing for: load_data
Elapsed: 00:00:00.53

SQL> exec check_space_bf
FS1 Blocks = 0 Bytes = 0
FS2 Blocks = 0 Bytes = 0
FS3 Blocks = 0 Bytes = 0
FS4 Blocks = 0 Bytes = 0
Full Blocks = 1080 Bytes = 8847360
Unformatted Blocks = 7078 Bytes = 57982976
==============================================
Total Blocks = 1080 ||  Total Bytes = 8847360
PL/SQL procedure successfully completed.
```

Step 3 : Create the procedure to provide the Space utilization Report for LOBs storage

DBMS_USAGE package has two SPACE_USAGE procedure. The procedure used below is specifically meant for SECUREFILE Lobs and cannot be used for BASICFILE Lob.

```
SQL> CREATE OR REPLACE PROCEDURE check_space_sf
  2  IS
  3     l_segment_size_blocks NUMBER;
  4     l_segment_size_bytes NUMBER;
  5     l_used_blocks NUMBER;
  6     l_used_bytes NUMBER;
  7     l_expired_blocks NUMBER;
  8     l_expired_bytes NUMBER;
  9     l_unexpired_blocks NUMBER;
 10     l_unexpired_bytes NUMBER;
 11     v_segname varchar2(30);
 12  BEGIN
 13     SELECT segment_name
 14     INTO v_segname
 15     FROM user_lobs
 16     WHERE table_name = 'RESUMES'
 17     AND column_name = 'RESUME';
 18     DBMS_SPACE.SPACE_USAGE(
 19        segment_owner      => 'SF_DEMO',
 20        segment_name       => v_segname,
 21        segment_type       => 'LOB',
 22        segment_size_blocks => l_segment_size_blocks,
 23        segment_size_bytes  => l_segment_size_bytes,
 24        used_blocks         => l_used_blocks,
 25        used_bytes          => l_used_bytes,
 26        expired_blocks         => l_expired_blocks,
 27        expired_bytes          => l_expired_bytes,
 28        unexpired_blocks         => l_unexpired_blocks,
 29        unexpired_bytes          => l_unexpired_bytes
 30     );
 31     DBMS_OUTPUT.ENABLE;
 32     DBMS_OUTPUT.PUT_LINE(' Segment Blocks = '||l_segment_size_blocks||' Bytes = '||l_segment_size_bytes);
 33     DBMS_OUTPUT.PUT_LINE(' Used Blocks = '||l_used_blocks||' Bytes = '||l_used_bytes);
 34     DBMS_OUTPUT.PUT_LINE(' Expired Blocks = '||l_expired_blocks||' Bytes = '||l_expired_bytes);
 35     DBMS_OUTPUT.PUT_LINE(' Unexpired Blocks = '||l_unexpired_blocks||' Bytes = '||l_unexpired_bytes);
 36     DBMS_OUTPUT.PUT_LINE('=============================================');
 37  END;
 38  /
Procedure created.
```

Step 4 : Check the Space Usage for SECUREFILE LOBs Table where Resume Loaded are all Different

```
SQL> drop table resumes;
Table dropped.
```

PLEASE NOTE we are USING DEDUPLICATE Storage Option that means that we don't want to duplicate the data

```
SQL> CREATE TABLE resumes
  2  (id NUMBER, first_name VARCHAR2(15),
  3  last_name VARCHAR2(40), resume BLOB)
  4  LOB(resume) STORE AS SECUREFILE
  5  (TABLESPACE tbs1
  6  COMPRESS HIGH
  7  DEDUPLICATE)
  8  /
Table created.

SQL> timing start load_data
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy2.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy3.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy4.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy5.doc');
PL/SQL procedure successfully completed.
SQL> timing stop
timing for: load_data
Elapsed: 00:00:00.35

SQL> exec check_space_sf
Segment Blocks = 128 Bytes = 1048576
Used Blocks = 12 Bytes = 98304
Expired Blocks = 99 Bytes = 811008
Unexpired Blocks = 0 Bytes = 0
=============================================
PL/SQL procedure successfully completed.
```

So here is Initial COMPARISION report where we have loaded 5 Different RESUMES into BASIFILE as well as SECUREFILE Lobs

Test Name	**BASICFILE Lobs [5 Different Resume Loaded]**	**SECUREFILE Lobs [5 Different Resume Loaded]**
Space Usage	Full Blocks = 1080 Bytes = 8847360	Used Blocks = 12 Bytes = 98304

Step 5 : Check the Space Usage for BINARYFILE LOBs Table when We Load the Same Resume 5 times

```
SQL> conn sf_demo/oracle
Connected.
SQL> drop table resumes;
Table dropped.

SQL> CREATE TABLE resumes
  2  (id NUMBER, first_name VARCHAR2(15),
  3  last_name VARCHAR2(40), resume BLOB)
  4  LOB(resume) STORE AS BASICFILE
  5  (TABLESPACE tbs2)
  6  /
Table created.

SQL> timing start load_data
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy2.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy3.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy4.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy5.doc');
PL/SQL procedure successfully completed.

SQL> timing stop
timing for: load_data
Elapsed: 00:00:00.53

SQL> exec check_space_bf
FS1 Blocks = 0 Bytes = 0
FS2 Blocks = 0 Bytes = 0
FS3 Blocks = 0 Bytes = 0
FS4 Blocks = 0 Bytes = 0
Full Blocks = 1200 Bytes = 9830400
Unformatted Blocks = 6958 Bytes = 56999936
==============================================
Total Blocks = 1200 ||  Total Bytes = 9830400
PL/SQL procedure successfully completed.
```

Step 6: Check the Space Usage for SECUREFILE LOBs Table where SAME Resume is loaded 5 times

```
SQL> drop table resumes;
Table dropped.

SQL> CREATE TABLE resumes
  2  (id NUMBER, first_name VARCHAR2(15),
  3  last_name VARCHAR2(40), resume BLOB)
  4  LOB(resume) STORE AS SECUREFILE
  5  (TABLESPACE tbs1
  6  COMPRESS HIGH
  7  DEDUPLICATE)
  8  /
Table created.

SQL> timing start load_data
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> exec write_lob('indy1.doc');
PL/SQL procedure successfully completed.
SQL> timing stop
timing for: load_data
Elapsed: 00:00:00.35

SQL> exec check_space_sf
Segment Blocks = 128 Bytes = 1048576
Used Blocks = 3 Bytes = 24576
Expired Blocks = 108 Bytes = 884736
Unexpired Blocks = 0 Bytes = 0
=============================================
PL/SQL procedure successfully completed.
```

So here is Initial COMPARISION report where we have loaded SAME RESUMES into BASIFILE as well as SECUREFILE Lobs
So here is Initial BASIC FILE CHART

Test Name	**BASICFILE Lobs [5 Same Resume Loaded]**	**SECUREFILE Lobs [5 Same Resume Loaded]**
Space Usage	Full Blocks = 1200 Bytes = 9830400	Used Blocks = 3 Bytes = 24576

DATA SOFTECH INC.
Complete Database Solution Provider

COMPARISON REPORT

When we have Loaded 5 Different RESUME, we can see that both BASICFILE as well as SECUREFILE has to load the data into the database. Still the Difference is very high in space utilization in BASICFILE lob as comparison to SECUREFILE Lob

Test Name	**BASICFILE Lobs** **[5 Different Resume Loaded]**	**SECUREFILE Lobs** **[5 Different Resume Loaded]**
Space Usage	Full Blocks = 1080 Bytes = 8847360	Used Blocks = 12 Bytes = 98304

When we have Loaded SAME RESUME 5 Times, we can see that only BASICFILE has to load this data 5 times while with SECUREFILE Storage clause we are using DEDUPLICATE and it means that duplicate will be avoided. The figures clearly indicates that the storage is now very much reduced in SECUREFILE as comparison to SECUREFILE Lobs

Test Name	**BASICFILE Lobs** **[5 Same Resume Loaded]**	**SECUREFILE Lobs** **[5 Same Resume Loaded]**
Space Usage	Full Blocks = 1200 Bytes = 9830400	Used Blocks = 3 Bytes = 24576

In the Final Part tomorrow, I will cover how to convert BASICFILE storage to SECUREFILE storage as well as cover how to Encrypt SECUREFILE Lobs data