ORACLE 11G SECURE FILES - PART 1

Inderpal S. Johal, Data Softech Inc.

INTRODUCTION

Oracle has provided various features like Domain Indexes. Partitioning, Parallelism etc which can provide good performance to store and retrieve the unstructured data in LOBs inside the database. But if you are dealing with very big files like Medical X-rays or Satellite images or Multimedia files and stored them inside the database, you may not get the same performance as comparison to storing them in Operating System. Oracle 11g has redesign LOBs format to store the Multimedia data which will not only provide superior performance but also take less storage to store the data.

The new LOB format introduced in 11g is called `SECUREFILE` LOBs (or SecureFiles). You can continue to use LOBs format used in pre-11g (Oracle 8 to Oracle 10g) which is now called `BASICFILE` LOBS in 11g. The performance of `SECUREFILE` LOBs is significantly better than that of `BASICFILE` LOBs, especially for large media data. Oracle recommends to use `SECUREFILE` LOBs for storing media data whenever possible. SecureFile LOBs are identified by specifying storage keyword `SECUREFILE` with `CREATE TABLE` and similarly BasicFile LOBS are identified by specifying keyword `BASICFILE` in the `CREATE TABLE` SQL statement. So whenever you add a LOB column to a table, you can specify whether it should be created as `SECUREFILE` or `BASICFILE`. If you do not specify the storage type with LOB column storage, then it is created as `BASICFILE` as long as `DB_SECUREFILE` initialization parameter is not forcing to create them as `SECUREFILE`. I will discuss this new 11g initialization parameter later in this paper.

You can easily move the existing pre 11g LOBs to new 11g Secure file LOBs online using online Redefinition package and other options like CTAS/ITAS, Export/Import and the like.

TOPICS COVERED IN ORACLE 11G SECURE FILE LOBS

1. How to create `SECUREFILE` LOBs
2. How to move the old pre-11g LOBs or Oracle 11g `BASICFILE` LOBs into new 11g `SECUREFILE` LOBs format
3. Compare the Performance of `BASICFILE` LOBs and `SECUREFILE` LOBs
4. Compare the Storage Usage of `BASICFILE` LOBs and `SECUREFILE` LOBs
5. How to secure the `SECUREFILE` LOBs data
6. How to share the `SECUREFILE` LOBs data to avoid redundancy
7. How to Compress the `SECUREFILE` LOBs data to reduce the Storage

ENABLING SECUREFILES STORAGE

Check the Default setting of 11g Database

```
SQL> show parameter securefile
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_securefile                        string      PERMITTED
```

The `DB_SECUREFILE` initialization parameter allows DBAs to determine the usage of SecureFiles, where valid values are as shown below. PERMITTED is default setting of this parameter:

[ALWAYS | FORCE | **PERMITTED** | NEVER | IGNORE]

DB_SECUREFILE Values	Description
ALWAYS	

```
Change the DB_SECUREFILE initialization parameter to ALWAYS
SQL> ALTER SYSTEM SET db_securefile = 'ALWAYS';
System altered.

Create ASSM and NON-ASSM Tablespace and see if you can created SECUREFILE Lobs in it
SQL> CREATE TABLESPACE tbs1
  2  DATAFILE '/home/oracle/app/oradata/11gtest/tbs2.dbf' SIZE 150M REUSE
  3  EXTENT MANAGEMENT LOCAL  UNIFORM SIZE 1M
  4  SEGMENT SPACE MANAGEMENT MANUAL;
Tablespace created.
SQL> CREATE TABLESPACE tbs2
  2  DATAFILE '/home/oracle/app/oradata/11gtest/tbs2.dbf' SIZE 150M REUSE
  3  EXTENT MANAGEMENT LOCAL  UNIFORM SIZE 1M
  4  SEGMENT SPACE MANAGEMENT AUTO;
Tablespace created.
```

- It will only create LOBs as `SECUREFILE` in ASSM Tablespace else an error will be thrown.

```
SECUREFILE Lob can only be allowed in ASSM Tablespace else it fails ass shown below
SQL> CREATE TABLE resumes_non_assm
  2  ( first_name VARCHAR2(15),
  3         resume BLOB
  4  ) LOB(resume) STORE AS SECUREFILE
  5  (TABLESPACE tbs1);
CREATE TABLE resumes_non_assm
*
ERROR at line 1:
ORA-43853: SECUREFILE lobs cannot be used in non-ASSM tablespace "TBS1"
```

- Attempts to create all LOBs as `SECUREFILE` LOBs whether you specify it or not.

```
Create Database table without specifying STORE AS SECUREFILE or BASICFILE but in ASSM
SQL> CREATE TABLE resumes_assm
  2  (  Name VARCHAR2(15),
  3     resume BLOB )TABLESPACE tbs2;
Table created.

SECUREFILE in the following SQL statement shows that it is creates as SECUREFILE LOB
SQL> select table_name,column_name,tablespace_name,securefile from user_lobs where
column_name='RESUME';
TABLE_NAME           COLUMN_NAM TABLESPACE SEC
-------------------- ---------- ---------- ---
RESUMES_ASSM         RESUME     TBS2       YES
```

DATA SOFTECH INC.
Complete Database Solution Provider

- Any LOBS created in non-ASSM Tablespaces are created as `BASICFILE` LOBs

```
If you are not specifying SECUREFILE Keyword and Tablespace specified is non-ASSM
SQL> CREATE TABLE resumes_assm
  2  ( Name VARCHAR2(15),
  3    resume BLOB
  4  )TABLESPACE tbs1;
Table created.

LOBs column is created as BASICFILE means not SECUREFILE as shown below
SQL> select table_name,column_name,tablespace_name,securefile from user_lobs where
column_name='RESUME';
TABLE_NAME           COLUMN_NAM TABLESPACE SEC
-------------------- ---------- ---------- ---
RESUMES_ASSM         RESUME     TBS1       NO
```

FORCE

- Force all LOBs created now onwards as `SECUREFILE` LOBs
- If any of the `BASICFILE` LOBs is specified then it will be ignored and default `SECUREFILE` storage option are automatically used.

```
SQL> ALTER SYSTEM SET db_securefile = 'FORCE';
System altered.

Tablespace used is ASSM as shown below which is required for SECUREFILE Lobs
SQL> select tablespace_name,segment_space_management "ASSM" from dba_tablespaces
where tablespace_name='TBS2';
TABLESPACE ASSM
---------- ------
TBS1       AUTO

SQL> CREATE TABLE resumes_assm
  2  (  Name VARCHAR2(15),
  3     resume BLOB
  4  ) LOB(resume) STORE AS BASICFILE
  5  TABLESPACE tbs1;
Table created.

We can see even BASICFILE lobs is created as SECUREFILE as long as Tablespace is ASSM
SQL> select table_name,column_name,tablespace_name,securefile from user_lobs where
column_name='RESUME';
TABLE_NAME           COLUMN_NAM TABLESPACE SEC
-------------------- ---------- ---------- ---
RESUMES_ASSM         RESUME     TBS1       YES
```

- If LOB is created in non ASSM Tablespace, and error will be thrown.

```
What happen if Tablespace is now ASSM as shown below. It will fail with error
SQL> select tablespace_name,segment_space_management "ASSM" from dba_tablespaces
where tablespace_name='TBS2';
TABLESPACE ASSM
---------- ------
TBS2       MANUAL

SQL> CREATE TABLE resumes_assm
  2  (  Name VARCHAR2(15),
  3     resume BLOB
  5  ) LOB(resume) STORE AS BASICFILE
  6  TABLESPACE tbs2;
CREATE TABLE resumes_assm
*
ERROR at line 1:
ORA-43853: SECUREFILE lobs cannot be used in non-ASSM tablespace "TBS2"
```

DATA SOFTECH INC.
Complete Database Solution Provider

PERMITTED	Allow DBAs to create SECUREFILE LOBs

NEVER

- Disallow SECUREFILE LOBs creation from now onwards.

```
SQL> ALTER SYSTEM SET db_securefile = 'NEVER';
System altered.

We will see that even you specify SECUREFILE, it will still be created as BASICFILE
SQL> CREATE TABLE resumes_assm
  2  (  Name VARCHAR2(15),
  3     resume BLOB
  5  ) LOB(resume) STORE AS SECUREFILE
  6  TABLESPACE tbs1;
Table created.

SQL> select table_name,column_name,tablespace_name,securefile from user_lobs where
column_name='RESUME';
TABLE_NAME           COLUMN_NAM TABLESPACE SEC
-------------------- ---------- ---------- ---
RESUMES_ASSM         RESUME     TBS1       NO
```

- All SECUREFILE LOBs are now created as BASICFILE LOBs
- All SECUREFILE specific storage options [COMPRESS,ENCRYPT,DEDUPLICATE] will throw an exception
- All BASICFILE LOBs default storage option are used automatically

```
If we specify any of the SECUREFILE option like DEDUPLICATE,COMPRESS,ENCRYPT etc it
will fail. We will compare the same with DB_SECUREFILE=IGNORE option later on
SQL> Create table resume
  2  (  name  varchar2(10),
  3     resume blob encrypt using 'AES128'
  4  ) lob(resume) store as SECUREFILE
  5  (DEDUPLICATE  COMPRESS);
(DEDUPLICATE  COMPRESS)
                     *
ERROR at line 5:
ORA-43854: use of a BASICFILE LOB where a SECUREFILE LOB was expected
```

IGNORE

- Disallow SECUREFILE LOBs same like NEVER option above
- Ignore any errors caused by SECUREFILE keyword as well as SECUREFILE options. Main difference as comparision to NEVER option which fails as shown earlier

```
SQL> ALTER SYSTEM SET db_securefile = 'IGNORE';
System altered.

Create Table as SECUREFILE lob, it will be created as BASICFILE
SQL> CREATE TABLE resumes_assm
  2  ( Name VARCHAR2(15),
  3    resume BLOB
  4  ) LOB(resume) STORE AS SECUREFILE
  5  TABLESPACE tbs1;
Table created.

Create Table as SECUREFILE lob with SECUREFILE option, it will be created as
BASICFILE and will not fail like happen in DB_SECUREFILE=NEVER
SQL> Create table resume_secopt
  2  (  name  varchar2(10),
  3     resume blob encrypt using 'AES128'
  4  ) lob(resume) store as SECUREFILE
  5  (DEDUPLICATE  COMPRESS);
Table created.
```

```
SQL> select table_name,column_name,tablespace_name,securefile from user_lobs where
column_name='RESUME';
TABLE_NAME           COLUMN_NAM TABLESPACE SEC
-------------------- ---------- ---------- ---
RESUMES_ASSM         RESUME     TBS1       NO
RESUME_SECOPT        RESUME     USERS      NO
```

The initialization parameter `DB_SECUREFILE` is dynamic and the scope is `ALTER SYSTEM` as already shown above

e.g

```
 ALTER SYSTEM SET db_securefile = 'ALWAYS';
```

Make sure that `Compatible` initialization parameter is set to 11g. If it is set to 10g, then LOB will work as it work in 10g and keyword `BASICFILE` is even not valid which behave like 10g LOBs. If you set Compatible to 11g, then you can continue to use LOBs functionality like 10g as long as you are using `BASICFILE` keyword and initialization parameter `DB_SECUREFILE` is set correctly as per above table.

Hopefully this will also cover the method to Create Database Table with SECUREFILE LOBs. I will cover more about securfile in coming Papers