**TRUNCATING TABLE**

**\* Let’s create a REPLICA of table EMP in SCOTT’s account, in order to show how truncating table works \***

SQL> **CREATE TABLE scott.myemp AS SELECT \* FROM scott.emp;**

Table created.

**\* CREATE TABLE AS command does NOT transfer the original PK, UK and FK constraints, just the NN and CK ones. So, we need to ADD manually any of these three. \***

SQL> **ALTER TABLE scott.myemp ADD CONSTRAINT myemp\_pk PRIMARY KEY (empno);**

Table altered.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'SCOTT' AND segment\_name = 'MYEMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

7 0 424 8

**\* This one shows that we have just ONE extent, because table MYEMP is in Tablespace USERS (Default one for user SCOTT), that is AUTOALLOCATED by system with Extents of 64K (or 8 Blocks).**

SQL> **ALTER TABLE scott.myemp ALLOCATE EXTENT (SIZE 120k);**

Table altered.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'SCOTT' AND segment\_name = 'MYEMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

7 0 424 8

7 1 440 8

7 2 448 8

**\* This one brings TWO new extents, because table MYEMP is in Tablespace USERS, that is AUTOALLOCATED with Extents of 64K ( or 8 Blocks).**

**It was asked for 120K (or 15 blocks), but we can get only Extents of 8 blocks, so It will be 2\*8=16 blocks, and the total is 1+2=3 extents \***

SQL> **ANALYZE TABLE scott.myemp ESTIMATE STATISTICS;**

Table analyzed.

SQL> **SELECT num\_rows, blocks HWM, empty\_blocks "Above HWM"**

**FROM dba\_tables**

**WHERE owner = 'SCOTT' AND table\_name ='MYEMP';**

NUM\_ROWS HWM Above HWM

---------- ---------- ----------

14 4 20

**🡪 Notice that HWM+ABOVE gives you Total # of Blocks (4+20=24)**

SQL> **TRUNCATE TABLE scott.myemp;**

Table truncated.

SQL> **ANALYZE TABLE scott.myemp ESTIMATE STATISTICS;**

Table analyzed.

SQL> **SELECT num\_rows, blocks HWM, empty\_blocks "Above HWM"**

**FROM dba\_tables**

**WHERE owner = 'SCOTT' AND table\_name ='MYEMP';**

NUM\_ROWS HWM Above HWM

---------- ---------- ----------

0 0 8

**\* TRUNCATE command removes ALL rows quickly and moves HWM to the far left position and also leaves ONE EMPTY EXTENT by default (unless, if table was created with STORAGE option and MINEXTENTS parameter was used) \***

**REORGANIZING (MOVING) TABLE EXTENTS**

SQL> **ALTER TABLE scott.myemp ALLOCATE EXTENT;**

Table altered.

**\* This one brings ONE new extent, so the total is 1+1=2 extents \***

SQL> **ALTER TABLE scott.myemp ALLOCATE EXTENT (SIZE 160K);**

Table altered.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'SCOTT' AND segment\_name = 'MYEMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

7 0 424 8

7 1 440 8

7 2 448 8

7 3 456 8

7 4 464 8

**\* This one brings THREE new extents, because table MYEMP is in Tablespace USERS, that is AUTOALLOCATED by system with Extents of 64K (or 8 Blocks).**

**It was asked for 160K (or 20 blocks)  3\*8 =24 blocks, so the total is 2+3=5 extents \***

**\* We will try to REORGANIZE the Extent situation for this table in SQL, meaning we will compress (MOVE) the Extents and hopefully reduce the # of allocated Extents.**

**This scenario involves 3 commands**

**1) MOVE the table extents**

**2) REBUILD the PK index**

**3) GATHER Fresh Statistics for the table \***

SQL> **ALTER TABLE SCOTT.MYEMP MOVE;**

Table altered.

SQL> **ALTER INDEX scott.myemp\_pk REBUILD ONLINE;**

Index altered.

SQL> **BEGIN DBMS\_STATS.GATHER\_TABLE\_STATS('SCOTT', 'MYEMP'); END;**

/

PL/SQL procedure successfully completed.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'SCOTT' AND segment\_name = 'MYEMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

7 0 472 8

**\* After doing these 3 steps, our table now has only ONE extent, and it used to have FIVE. \***

**SHRINKING TABLE CONTENT (Reducing # of Extents)**

SQL> **CONN tom/cat**

Connected.

SQL> **SELECT TNAME FROM TAB;**

TNAME

----------------------------------------------------------------------

DEPT

EMP

CUSTOMER

EXCEPTIONS

SQL> **CONN / AS SYSDBA**

Connected.

SQL> **SELECT COUNT(\*) “# of rows” FROM tom.emp;**

# of rows

---------------

15

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'TOM' AND segment\_name = 'EMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

5 0 256 64

**\* This one shows that we have just ONE extent, because table EMP is in Tablespace MINE (Default one for user TOM), that is of UNIFORM size 512K (or 64 Blocks). \***

SQL> **ALTER TABLE tom.emp ALLOCATE EXTENT (SIZE 1M);**

Table altered.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'TOM' and segment\_name = 'EMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

5 0 256 64

5 1 640 64

5 2 704 64

**\* This one brings TWO new extents, because table EMP is in Tablespace MINE, that is of UNIFORM size 512K (or 64 Blocks).**

**It was asked for 1M  2\*512K = 1M, so the total is 1+2 = 3 extents \***

**\* We’ll try here to use option SHRINK to reduce # of extents used here. This scenario involves 2 steps:**

**1) Enable Row Movement for the table**

**2) SHRINK SPACE 🡪 that will Compact the table and Release freed space \***

SQL> **ALTER TABLE tom.emp ENABLE ROW MOVEMENT;**

Table altered.

SQL> **ALTER TABLE tom.emp SHRINK SPACE;**

Table altered.

SQL> **SELECT file\_id, extent\_id, block\_id, blocks**

**FROM dba\_extents**

**WHERE owner = 'TOM' AND segment\_name = 'NEW\_EMP';**

FILE\_ID EXTENT\_ID BLOCK\_ID BLOCKS

---------- ---------- ---------- ----------

5 0 256 64

**\* It is clear that table TOM.EMP shrank from 3 to only 1 extent \***

**REORGANIZING (COALESCING) TABLESPACE FREE SPACE**

SQL> **SELECT tablespace\_name, COUNT(\*) "# of Free Fragments",**

**SUM(bytes)/1024/1024 "Total Free Space (M)",**

**MAX(bytes)/1024/1024 "Largest Free Fragment"**

**FROM dba\_free\_space**

**GROUP BY tablespace\_name**

**ORDER BY 3 DESC;**

TABLESPACE\_NAME # of Free Fragments Total Free Space (M)

------------------------------ ------------------- -------------------

Largest Free Fragment

---------------------

SYSAUX 82 101

56

UNDOTBS1 6 81.6875

78

JOKE 2 17.8125

14.921875

INDX 1 8.8125

8.8125

SYSTEM 2 8.75

8

MINE 1 5

5

USERS 3 2.8125

2.5

7 rows selected.

**\* Let’s try to Reorganize the empty space in Tablespace USERS, that has 3 Fragments. This operation is much easier and quicker to perform in SQL compared to EM CC. We will use COALESCE option (similar to REORGANIZE=MOVE option for segments) \***

SQL> **ALTER TABLESPACE users COALESCE;**

Tablespace altered.

SQL> **SELECT tablespace\_name, COUNT(\*) "# of Free Fragments",**

**SUM(bytes)/1024/1024 "Total Free Space (M)",**

**MAX(bytes)/1024/1024 "Largest Free Fragment"**

**FROM dba\_free\_space**

**GROUP BY tablespace\_name**

**ORDER BY 3 DESC;**

TABLESPACE\_NAME # of Free Fragments Total Free Space (M)

------------------------------ ------------------- -------------------

Largest Free Fragment

---------------------

SYSAUX 82 101

56

UNDOTBS1 6 81.6875

78

JOKE 2 17.8125

14.921875

INDX 1 8.8125

8.8125

SYSTEM 2 8.75

8

MINE 1 5

5

USERS 3 2.8125

2.5

7 rows selected.

**\* We see that our Coalesce option did NOT make any difference, but it usually helps. \***

SQL> **EXIT**

Disconnected from Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.3.0.0.0