DBA

ONLINE TABLE REDEFINITION
MODIFYING STORAGE PARAMETERS

Introduction

When we delete the records from a table, the records from the table is deleted but the segment space is still occupied in the table. Thus, we have to reclaim those deleted segment space from the table. There is a feature in Oracle through with we could recover those segment space, after the records in the table are deleted, which is called **Online Table Redefinition**. Further there is a package in Oracle *DBMS_REDEFINITION* which does this. Below documentation provides a brief overview of how we can recoup the unused segment space which was once used by the table.

The code can be found on github.

To regain the segment space, we have following alternatives:

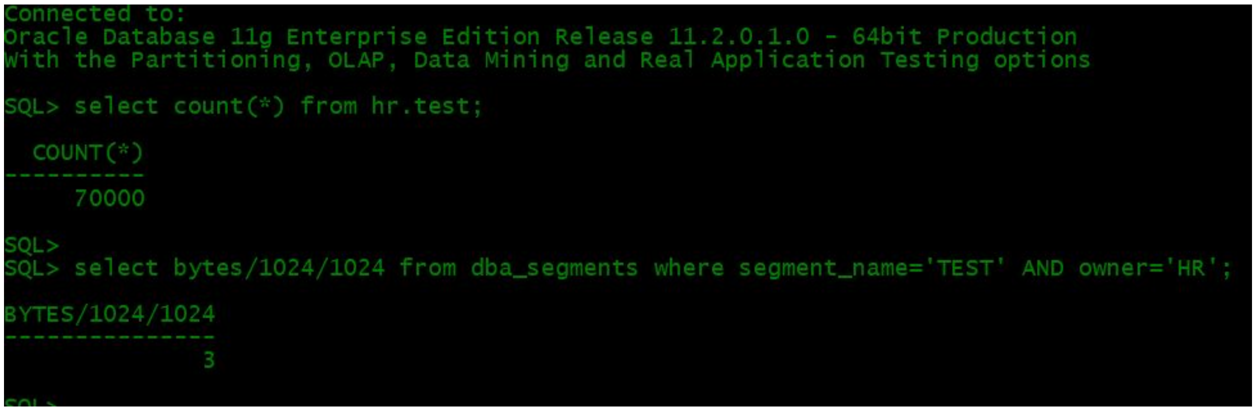
- Shrink the table
- Table redefinition

This documentation is about **table redefinition**.

For instance, We have a test table with 70000 rows. It's size in MB is calculated using following query :

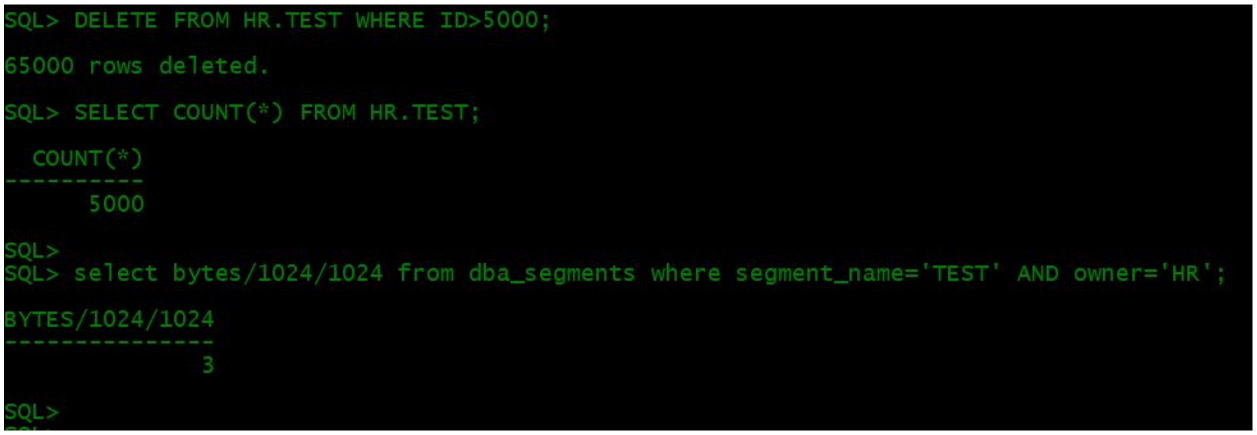
Select bytes/1024/1024 from dba_segments where segment_name='TEST' and owner='HR';

So, its size as calculated is 3 MB.



```
Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select count(*) from hr.test;

  COUNT(*)
----------
     70000

SQL>
SQL> select bytes/1024/1024 from dba_segments where segment_name='TEST' AND owner='HR';

BYTES/1024/1024
---------------
              3
```

Now, we deleted 65000 rows from that table and calculated the size of table, which is still 3MB.



```
SQL> DELETE FROM HR.TEST WHERE ID>5000;

65000 rows deleted.

SQL> SELECT COUNT(*) FROM HR.TEST;

  COUNT(*)
----------
      5000

SQL>
SQL> select bytes/1024/1024 from dba_segments where segment_name='TEST' AND owner='HR';

BYTES/1024/1024
---------------
              3

SQL>
```

So this concludes that oracle does not frees the unused space automatically. In case of large database, it is a big problem. To solve this issue, we can use dbms_redefinition package. Oracle redefinition provides an interface to perform an online redefinition of tables.

Privilege to be granted:

grant execute on dbms_redefinition to HR;

grant select on dba_segments to HR;

Overview of the process:

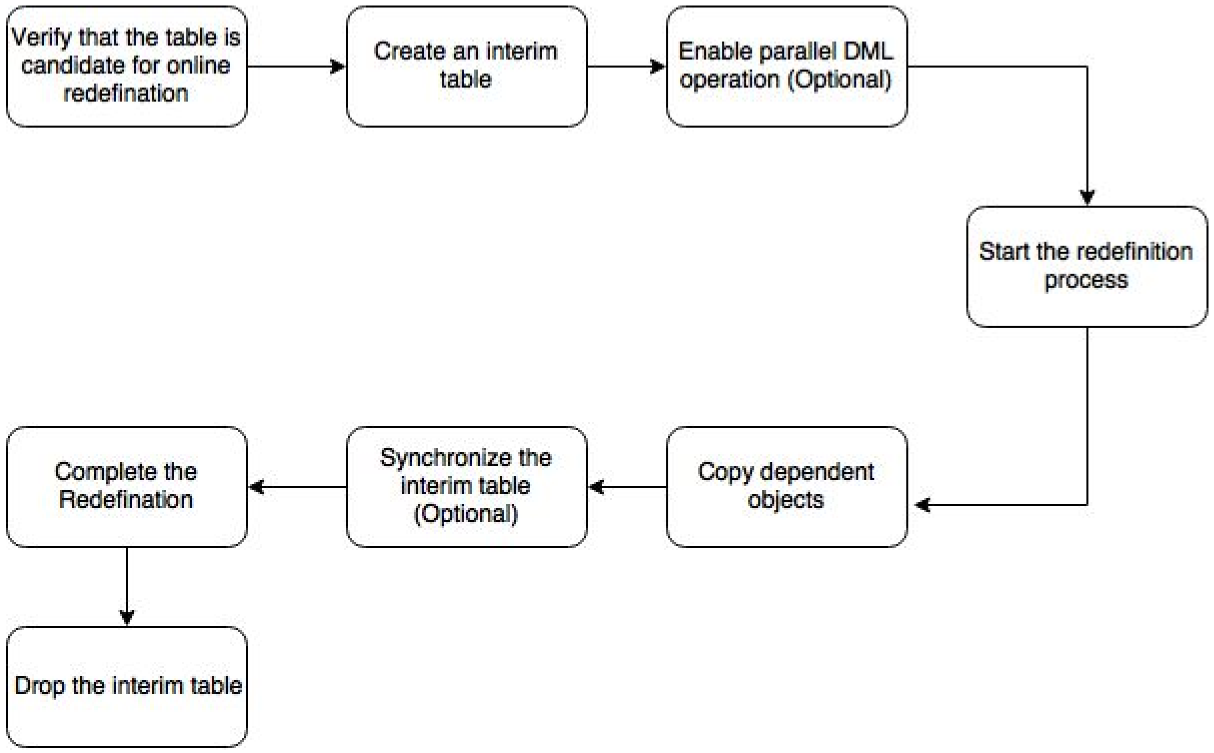

Verify that the table is candidate for online redefination
Create an interim table
Enable parallel DML operation (Optional)
Start the redefinition process
Copy dependent objects
Synchronize the interim table (Optional)
Complete the Redefination
Drop the interim table

Fig: Online Table Redefinition Process

1.Verify the table is candidate for online redefinition:

If the table is indexed with the primary key, then we can verify using primary key verification else we could use rowid verification.

Using rowid

DBMS_REDEFINITION.CAN_REDEF_TABLE ('HR', 'TEST', DBMS_REDEFINITION.CONS_USE_ROWID);

Using primary key

DBMS_REDEFINITION.CAN_REDEF_TABLE ('HR', 'TEST', DBMS_REDEFINITION.CONS_USE_PK);

2. Create an Interim Table:

In our stored procedure, we created an interim table of the same structure as of original TEST table, basically, to store all the contents of the TEST table. The sql script to do this is:

CREATE TABLE test_interim AS

*SELECT * FROM test WHERE 1=2;*

Here, above script creates an interim table, same structure as of table named TEST . We did this with the help of WHERE clause, where 1=2 specifies FALSE condition. That means, no records are inserted because we have all false condition to insert records into new table.

3. Enable parallel for large table(Optional):

In case if the table we are trying to redefine is large, then it is a good idea to enable parallelism. We can do this using the ALTER SESSION statement.

ALTER SESSION FORCE PARALLEL DML PARALLEL 8;

ALTER SESSION FORCE PARALLEL QUERY PARALLEL 8;

Here, 8 is the degree of parallelism. The degree of parallelism (DOP) is the number of parallel execution servers in one set. Here DOP = 8 means, a table is partitioned into 8 fragments and each fragments are concurrently processed.

4. Start redefinition process:

Next, we started the redefinition process using *start_redef_table* procedure of DBMS_REDEFINITION package. The parameters for the procedure were

- Schema Name
- Original Table Name
- Interim Table Name

SQL Script:

DBMS_REDEFINITION.start_redef_table(‘HR’, ‘TEST’, ‘TEST_INTERIM’);

5. Copy Dependent objects:

We copy all the records from the original table(TEST) into our interim table using *copy_table_dependents* procedure of DBMS_REDEFINITION package. This procedure clones the dependent objects of the table being redefined onto the interim table and registers the dependent objects. This procedure does not clone the already registered dependent objects.

The parameters for this procedure are:

- Schema name
- Original table name
- Interim table name

SQL Scripts:

DBMS_REDEFINITION.copy_table_dependents('HR', 'TEST', 'TEST_INTERIM');

6. Synchronize Interim Table(Optional):

Instead of *copy_table_dependents* procedure, we can use *sync_interim_table* procedure of DBMS_REDEFINITION package to synchronize the interim table with the original table.

The parameters for this procedure are:

- Schema name
- Original table name
- Interim table name

SQL Script:

DBMS_REDEFINITION.sync_interim_table('HR', 'TEST', 'TEST_INTERIM');

7. Complete redefinition process:

Finally, we complete the redefinition process using *finish_redef_table* procedure of DBMS_REDEFINITION package.

The parameters for this procedure are:

- Schema name

- Original table name
- Interim table name

SQL Script:

DBMS_REDEFINITION.finish_redef_table('HR', 'TEST', 'TEST_INTERIM');

8. Drop the interim table:

Now, after the segment space issue is solved, we can drop the interim table.

Interim Table was dropped simply using DROP TABLE statement.

SQL Script:

DROP TABLE test_interim;

References :

1. https://docs.oracle.com/cd/B28359_01/server.111/b28310/tables007.htm#ADMIN11668
2. https://oracle-base.com/articles/11g/online-table-redefinition-enhancements-11gr1