Online Reorganisation with DBMS_REDEFINITION

Abstract

This paper demonstrates how to reorganize a main table with customer data of a Siebel system after a substantial amount of records have been deleted and row chaining is occurring.

Contents - Steps

Step 1 - Checks if online redefinition is possible

By executing DBMS_REDEFINITION.CAN_REDEF you can check whether the object is capable of being online redefined. If the table is not supported for redefinition, an error is reported.

There are 2 options for online redefinition:

- by using primary key/unique keys (default)
- by using Oracle rowid

```
set serveroutput on
begin
dbms_redefinition.can_redef_table(
        uname => 'SIEBEL',
        tname => 'S_CONTACT',
        options_flag => dbms_redefinition.cons_use_pk);
end;
/
```

Step 2 - Initial creation oft the interim table

Next, the interim table is created. This is the target segment of the table. This means, that if you wish to modify column list or partition the target table, this is where the definition has to be changed. In my example, I have added the string "_REDEF" to the original table name.

```
CREATE TABLE "SIEBEL"."S_CONTACT_REDEF"
   (  "ROW_ID" VARCHAR2(15 CHAR) NOT NULL ENABLE,
  "CREATED" DATE DEFAULT sysdate NOT NULL ENABLE,
  "CREATED_BY" VARCHAR2(15 CHAR) NOT NULL ENABLE,
  "LAST_UPD" DATE DEFAULT sysdate NOT NULL ENABLE,
  "LAST_UPD_BY" VARCHAR2(15 CHAR) NOT NULL ENABLE,
  "DCKING_NUM" NUMBER(22,7) DEFAULT 0,
  "MODIFICATION_NUM" NUMBER(10,0) DEFAULT 0 NOT NULL ENABLE,
  "CONFLICT_ID" VARCHAR2(15 CHAR) DEFAULT '0' NOT NULL ENABLE,
  "PAR_ROW_ID" VARCHAR2(15 CHAR) DEFAULT 'x' NOT NULL ENABLE,
  "ACTIVE_FLG" CHAR(1 CHAR) DEFAULT 'Y' NOT NULL ENABLE,
  "BU_ID" VARCHAR2(15 CHAR) DEFAULT '0-R9NH' NOT NULL ENABLE,
  "COURT_PAY_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "DISA_CLEANSE_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "DISP_IMG_AUTH_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "EMAIL_SR_UPD_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "EMP_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "FST_NAME" VARCHAR2(50 CHAR) NOT NULL ENABLE,
  "INVSTGTR_FLG" CHAR(1 CHAR) DEFAULT 'N' NOT NULL ENABLE,
  "LAST_NAME" VARCHAR2(50 CHAR) NOT NULL ENABLE,
  "WORK_PH_NUM" VARCHAR2(40 CHAR),
   SUPPLEMENTAL LOG GROUP "LP_S_CONTACT_REDEF" ("ROW_ID") ALWAYS,
   SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS,
   SUPPLEMENTAL LOG DATA (UNIQUE INDEX) COLUMNS,
   SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS
   ) PCTFREE 20 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
  STORAGE(INITIAL 134217728 NEXT 134217728 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "TSD128M_01" ;
```

Step 3 – Start of Redef

I suspected that the start of the redefinition would take a long time and consume many resources, so I have wrapped the command inside a procedure and executed

the procedure via DBMS_SCHEDULER during off-peak hours. Theoretically, it is possible to reorder the rows with the “orderby_cols” parameter. On a production system I would not recommend to reorder the rows of this main Siebel table without thorough testing.

```
CREATE OR REPLACE PROCEDURE MDDB."START_REDEF" is
begin
  execute immediate 'ALTER SESSION ENABLE PARALLEL DML';
  execute immediate 'ALTER SESSION FORCE PARALLEL DML PARALLEL 4';
  execute immediate 'ALTER SESSION FORCE PARALLEL QUERY PARALLEL 4';
  execute immediate 'ALTER SESSION SET DB_FILE_MULTIBLOCK_READ_COUNT=128';
  dbms_redefinition.start_redef_table(uname        => 'SIEBEL',
        orig_table   => 'S_CONTACT',
        int_table    => 'S_CONTACT_REDEF',
        options_flag => dbms_redefinition.cons_use_pk,
        orderby_cols => 'LAST_NAME,FST_NAME');
end start_redef;
/

BEGIN
sys.dbms_scheduler.create_job(
job_name => '"MD"."REDEF_S_CONTACT"',
job_type => 'STORED_PROCEDURE',
job_action => '"MD"."START_REDEF"',
start_date => sysdate+(6/24),
job_class => '"DEFAULT_JOB_CLASS"',
auto_drop => TRUE,
enabled => TRUE);
END;
/

select * from dba_scheduler_job_run_details where job_name =
'REDEF_S_CONTACT'
```

4 Creation of SQL for registration of constraints

The next step is to register the constraints. I have used the following dynamic SQL statement to generate a list of SQL commands for registration:

```
 select 'DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT
(SIEBEL,S_CONTACT,S_CONTACT_REDEF,3, SIEBEL,
||a.constraint_name||,||b.constraint_name||);' from
(select * from dba_cons_columns where table_name = 'S_CONTACT') a,
(select * from dba_cons_columns where table_name = 'S_CONTACT_REDEF') b
where a.column_name = b.column_name
```

```
BEGIN
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',3, 'SIEBEL', 'SYS_C00168731','SYS_C00169292');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',3, 'SIEBEL', 'SYS_C00168730','SYS_C00169291');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',3, 'SIEBEL', 'SYS_C00168729','SYS_C00169290');
…
END;
/
```

5 Copy of dependencies (Trigger, Grants)

With the next step, I have copied Triggers and Grants from the source table to the interim table.

```
set serveroutput on
DECLARE
   nr_errors PLS_INTEGER;
BEGIN
DBMS_REDEFINITION.COPY_TABLE_DEPENDENTS(
   uname                    => 'SIEBEL',
   orig_table               => 'S_CONTACT',
   int_table                => 'S_CONTACT_REDEF',
   copy_indexes             => 0,
   copy_triggers            => TRUE,
   copy_constraints         => FALSE,
   copy_privileges          => TRUE,
   ignore_errors            => FALSE,
   num_errors               => nr_errors,
   copy_statistics          => FALSE);
dbms_output.put_line(nr_errors);
END;
/
```

6 Creation of primary indexes

I have decided to manually create the indexes. Therefore in the previous command, the parameter copy_indexes is set to zero. I start with the “main” index which happens to be called “_P1” by Siebel. I have decided to add the postfix “_R” to the index names.

```
ALTER SESSION SET DB_FILE_MULTIBLOCK_READ_COUNT=128;
CREATE UNIQUE INDEX "SIEBEL"."S_CONTACT_P1_R" ON "SIEBEL"."S_CONTACT_REDEF"
("ROW_ID")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 NOLOGGING COMPUTE STATISTICS
  STORAGE(INITIAL 5242880 NEXT 5242880 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "TSI128M_01" PARALLEL 2;

ALTER INDEX "SIEBEL"."S_CONTACT_P1_R" NOPARALLEL;
```

7 Intermediate sync

It is important to create the unique index before executing the sync_interim_table command.

```
begin
dbms_redefinition.sync_interim_table(
        uname => 'SIEBEL',
        orig_table => 'S_CONTACT',int_table => 'S_CONTACT_REDEF');
end;
/
```

8 Creation of other indexes

Now, the remaining indexes can be created. If the creation takes a long time, then it is advisable to run sync_interim_table before the next “create index” statement.

```
CREATE INDEX "SIEBEL"."S_CONTACT_DB01_X_R" ON "SIEBEL"."S_CONTACT_REDEF"
("CON_CD", "LAST_NAME", "FST_NAME")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 NOLOGGING COMPUTE STATISTICS
  STORAGE(INITIAL 5242880 NEXT 5242880 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "TSI128M_01" PARALLEL 2;

CREATE INDEX "SIEBEL"."S_CONTACT_DB02_X_R" ON "SIEBEL"."S_CONTACT_REDEF"
("CALL_FLG")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 NOLOGGING COMPUTE STATISTICS
  STORAGE(INITIAL 5242880 NEXT 5242880 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "TSI5M_01" PARALLEL 2;

CREATE INDEX "SIEBEL"."S_CONTACT_DB04_X_R" ON "SIEBEL"."S_CONTACT_REDEF"
("BIRTH_DT", "LAST_NAME", "FST_NAME")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 NOLOGGING COMPUTE STATISTICS
  STORAGE(INITIAL 5242880 NEXT 5242880 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "TSI128M_01" PARALLEL 2;


etc.

 ALTER INDEX .. NOPARALLEL;
```

9 Registration of indexes

Now with REGISTER_DEPENDENT_OBJECT, we can tell Oracle which index name on the interim table matches which index on the source table.

```
 select
'DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT(SIEBEL,S_CONTACT,S_CONTACT_RED
EF,2, SIEBEL, ||i1.index_name||,||i2.index_name||);'
from dba_indexes i1, dba_indexes i2
where i1.owner = i2.owner and
```

```
i2.index_name = i1.index_name||'_R'
and i1.owner = 'SIEBEL'
and i1.table_name = 'S_CONTACT' and
i2.table_name = 'S_CONTACT_REDEF'
BEGIN
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M15','S_CONTACT_M15_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M16','S_CONTACT_M16_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M17','S_CONTACT_M17_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M18','S_CONTACT_M18_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M19','S_CONTACT_M19_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M20','S_CONTACT_M20_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M21','S_CONTACT_M21_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M22','S_CONTACT_M22_R');
DBMS_REDEFINITION.REGISTER_DEPENDENT_OBJECT('SIEBEL','S_CONTACT','S_CONTACT
_REDEF',2, 'SIEBEL', 'S_CONTACT_M3','S_CONTACT_M3_R');
--- and more
END;
/
```

10 Intermediate sync

Now we are ready for one last intermediate sync.

```
begin
dbms_redefinition.sync_interim_table(uname => 'SIEBEL',orig_table =>
'S_CONTACT',int_table => 'S_CONTACT_REDEF');
end;
/
```

11 Optimizer Statistics

Perhaps one of the most important steps is the creation of the optimizer statistics for the interim table. One way to gather stats is by setting COPY_STATS to yes when calling DBMS_REDEFINITION.COPY_TABLE_DEPENDENTS. However this results in ANALYZE TABLE COMPUTE STATISTICS, which is not how we want to gather stats. In fact, I have decided to clone the statistics from the source table to the interim table. This has several reasons but the main reason is that I want to avoid SQL Plan changes because of this reorganization. You can find a description for the stattab table fields in this document:
http://www.centrexcc.com/SQL%20Tuning%20with%20Statistics.ppt.pdf

```
drop table MDECKER.S_CONTACT_REDEF_STATTAB;
drop table MDECKER.S_CONTACT_STATTAB;
```

```
begin dbms_stats.create_stat_table(
        ownname => 'MDECKER',
        stattab => 'S_CONTACT_REDEF_STATTAB');
end;
/

begin dbms_stats.create_stat_table(
        ownname => 'MDECKER',
        stattab => 'S_CONTACT_STATTAB');
end;
/

begin
  dbms_stats.export_table_stats(ownname => 'SIEBEL',
                                tabname => 'S_CONTACT_REDEF',
                                stattab => 'S_CONTACT_REDEF_STATTAB',
                                cascade => true,
                                statown => 'MDECKER');
end;
/

begin
  dbms_stats.export_table_stats(ownname => 'SIEBEL',
                                tabname => 'S_CONTACT',
                                stattab => 'S_CONTACT_STATTAB',
                                cascade => true,
                                statown => 'MDECKER');
end;
/

SELECT * from MDECKER.S_CONTACT_STATTAB WHERE TYPE = 'T'

update MDECKER.S_CONTACT_STATTAB
set c1 = 'S_CONTACT_REDEF'
WHERE TYPE = 'T';


SELECT * from MDECKER.S_CONTACT_STATTAB WHERE TYPE = 'I'

update MDECKER.S_CONTACT_STATTAB
set c1 = c1||'_R'
WHERE TYPE = 'I';

SELECT *
  FROM DBA_TAB_COLS
 where TABLE_NAME IN ('S_CONTACT', 'S_CONTACT_REDEF')
   AND OWNER = 'SIEBEL'
   AND COLUMN_NAME LIKE 'SYS_%'

SELECT * from MDECKER.S_CONTACT_STATTAB WHERE TYPE = 'C' and C4 like
'SYS_%' order by c4

update MDECKER.S_CONTACT_STATTAB
set c1 = 'S_CONTACT_REDEF'
WHERE TYPE = 'C';

update MDECKER.S_CONTACT_STATTAB
set c4 = 'SYS_NC00288$'
WHERE TYPE = 'C'
and c4 like 'SYS_NC%';
```

```
commit;

begin
  dbms_stats.import_table_stats(ownname => 'SIEBEL',
                                tabname => 'S_CONTACT_REDEF',
                                stattab => 'S_CONTACT_STATTAB',
                                cascade => true,
                                statown => 'MDECKER');
end;
/
```

Checks:

```
select index_name, clustering_factor, num_rows, last_analyzed from
dba_ind_statistics where table_owner = 'SIEBEL' and table_name =
'S_CONTACT_REDEF'
MINUS
select index_name||'_R', clustering_factor, num_rows, last_analyzed from
dba_ind_statistics where table_owner = 'SIEBEL' and table_name =
'S_CONTACT'

select index_name||'_R', clustering_factor, num_rows, last_analyzed from
dba_ind_statistics where table_owner = 'SIEBEL' and table_name =
'S_CONTACT'
MINUS
select index_name, clustering_factor, num_rows, last_analyzed from
dba_ind_statistics where table_owner = 'SIEBEL' and table_name =
'S_CONTACT_REDEF'

select column_name, num_distinct, low_value, high_value, num_buckets,
last_analyzed from dba_tab_col_statistics  where owner = 'SIEBEL' and
table_name = 'S_CONTACT_REDEF'
MINUS
select column_name, num_distinct, low_value, high_value, num_buckets,
last_analyzed from dba_tab_col_statistics  where owner = 'SIEBEL' and
table_name = 'S_CONTACT'

select column_name, num_distinct, low_value, high_value, num_buckets,
last_analyzed from dba_tab_col_statistics  where owner = 'SIEBEL' and
table_name = 'S_CONTACT'
MINUS
select column_name, num_distinct, low_value, high_value, num_buckets,
last_analyzed from dba_tab_col_statistics  where owner = 'SIEBEL' and
table_name = 'S_CONTACT_REDEF'
```

12 Ending of Redefinition begin

Finally this is the last step of our redefinition process. This is the only point in time where an exclusive lock is obtained for a very short amount of time to rename the interim table to the target table and vice versa. I have successfully performed this reorganization on a mission critical database during normal daily activity without any problems. The final step took less than one second.

```
begin
dbms_redefinition.finish_redef_table(
```

```
        uname => 'SIEBEL',
        orig_table => 'S_CONTACT',int_table => 'S_CONTACT_REDEF');
end;
/
```

References:

Oracle Database 10.2 Documentation – Administrators Guide
http://download.oracle.com/docs/cd/B19306_01/server.102/b14231/tables.htm#i1006754