HCC test case

By: Karl Arao

Objective

- A quick guide on how to do HCC compression

How to use this doc

- First, follow the setup the environment
- Then estimate the compression
- Then create the actual HCC tables
- Check the estimate vs real
- and validate the rows
- Run some sample SQLs on HCC tables and instrument each run
- You can also run some DML on the tables and validate the rows
- Lastly re-compress

Setup environment

- Create the two scripts and run them sequentially. The scripts will do the following:
    - Create the tablespaces TS_HCCTEST and TS_SCRATCH
    - Create the user HCCUSER
    - Create the table HCCTEST
    - Grow the HCCTEST table to 1.2GB

```
$ vi 1_cr_table.sh

sqlplus /nolog<<EOF
connect / as sysdba
drop user hccuser cascade;
drop tablespace ts_hcctest including contents and datafiles;
drop tablespace ts_scratch including contents and datafiles;

create bigfile tablespace ts_hcctest;
create bigfile tablespace ts_scratch;

create user hccuser identified by hccuser;
grant dba to hccuser;
grant select any dictionary to hccuser;
grant unlimited tablespace to hccuser;
alter user hccuser default tablespace ts_hcctest;
```

```
alter user hccuser temporary tablespace temp;

connect hccuser/hccuser
create table hcctable tablespace ts_hcctest parallel nologging as
select * from sys.dba_objects where rownum <= 10000;
commit;
exit
EOF
```

```
$ vi 2_datagrow.sh

(( n=0 ))
while (( n<10 ));do
(( n=n+1 ))
sqlplus -s /NOLOG <<! &
connect hccuser/hccuser;
set timing on
set time on
alter session enable parallel dml;
insert /*+ APPEND */ into hcctable select * from hcctable;
commit;
select /*+ parallel(32) */ count(*) from hcctable
exit;
!
wait
done
wait
```

Test data

- The data is simply a clone of the dba_objects

```
desc hccuser.hcctable

           Name                 Null?    Type
           ------------------ -------- ---------------
    1      OWNER                         VARCHAR2(30)
    2      OBJECT_NAME                   VARCHAR2(128)
    3      SUBOBJECT_NAME                VARCHAR2(30)
    4      OBJECT_ID                     NUMBER
    5      DATA_OBJECT_ID                NUMBER
    6      OBJECT_TYPE                   VARCHAR2(19)
```

```
     7       CREATED                        DATE
     8       LAST_DDL_TIME                  DATE
     9       TIMESTAMP                      VARCHAR2(19)
    10       STATUS                         VARCHAR2(7)
    11       TEMPORARY                      VARCHAR2(1)
    12       GENERATED                      VARCHAR2(1)
    13       SECONDARY                      VARCHAR2(1)
    14       NAMESPACE                      NUMBER
    15       EDITION_NAME                   VARCHAR2(30)
```

Estimate compression

- The compression efficiency can be estimated by using the DBMS_COMPRESSION package.
- The package creates a temporary object in the scratch tablespace (TS_SCRATCH) and from the initial tests the tables with 850GB and 75GB size created 343GB and 37GB scratch objects respectively (40%-49% of size).
    - So for a 2TB size table, at max the scratch object size would be around 1TB
    - A tablespace quota can be enforced on the test user to not consume too much space. Or a size limit can be enforced on the tablespace upon creation.

Create hcc_estimate script

```
$ vi hcc_estimate.sql

spool hcc_estimate.txt append
set serveroutput on
DECLARE
  l_blkcnt_cmp       BINARY_INTEGER;
  l_blkcnt_uncmp     BINARY_INTEGER;
  l_row_cmp          BINARY_INTEGER;
  l_row_uncmp        BINARY_INTEGER;
  l_cmp_ratio        NUMBER;
  l_comptype_str     VARCHAR2(100);
BEGIN
  FOR i IN (SELECT table_name
            FROM user_tables
            WHERE compression = 'DISABLED'
            AND table_name in (&1) -- put table names here
            ORDER BY table_name)
  LOOP
    FOR j IN 1..5
    LOOP
      dbms_compression.get_compression_ratio(
        -- input parameters
        scratchtbsname   => 'TS_SCRATCH',       -- scratch tablespace
        ownname          => user,            -- owner of the table
```

```
        tabname           => i.table_name,     -- table name
        partname          => NULL,             -- partition name
        comptype          => power(2,j),       -- compression algorithm
        -- output parameters
        blkcnt_cmp        => l_blkcnt_cmp,     -- number of compressed
blocks
        blkcnt_uncmp      => l_blkcnt_uncmp,   -- number of uncompressed
blocks
        row_cmp           => l_row_cmp,        -- number of rows in a
compressed block
        row_uncmp         => l_row_uncmp,      -- number of rows in an
uncompressed block
        cmp_ratio         => l_cmp_ratio,      -- compression ratio
        comptype_str      => l_comptype_str    -- compression type
      );
      dbms_output.put_line(i.table_name||' - '||'type:
'||l_comptype_str||' ratio: '||to_char(l_cmp_ratio,'99.999'));
    END LOOP;
  END LOOP;
END;
/
spool off
```

Run hcc_estimate_script

- Enclose the single quotes with double quotes. The output will be spooled at hcc_estimate.txt

```
-- for a single table run
@hcc_estimate "'HCCTABLE'"

-- for multiple table run
@hcc_estimate "'HCCTABLE','KARLTEST1','KARLTEST2','KARLTEST3'"
```

View hcc_estimate

```
$ cat hcc_estimate.txt | grep ratio
HCCTABLE - type: "Compress For OLTP" ratio:   1.600
HCCTABLE - type: "Compress For Query Low" ratio:   3.700
HCCTABLE - type: "Compress For Query High" ratio:   9.200
HCCTABLE - type: "Compress For Archive Low" ratio:  10.700
HCCTABLE - type: "Compress For Archive High" ratio:  37.400
```

Creating HCC object

Using Create Table As Select (CTAS)

```
-- OLTP
CREATE TABLE "HCCUSER"."HCCTABLE_CTAS_OLTP"
LOGGING TABLESPACE "TS_HCCTEST"
COMPRESS FOR OLTP
AS SELECT * FROM "HCCUSER"."HCCTABLE";

-- QUERY LOW
CREATE TABLE "HCCUSER"."HCCTABLE_CTAS_QUERY_LOW"
LOGGING TABLESPACE "TS_HCCTEST"
COMPRESS FOR QUERY LOW
AS SELECT * FROM "HCCUSER"."HCCTABLE";

-- QUERY HIGH
CREATE TABLE "HCCUSER"."HCCTABLE_CTAS_QUERY_HIGH"
LOGGING TABLESPACE "TS_HCCTEST"
COMPRESS FOR QUERY HIGH
AS SELECT * FROM "HCCUSER"."HCCTABLE";

-- ARCHIVE LOW
CREATE TABLE "HCCUSER"."HCCTABLE_CTAS_ARCHIVE_LOW"
LOGGING TABLESPACE "TS_HCCTEST"
COMPRESS FOR ARCHIVE LOW
AS SELECT * FROM "HCCUSER"."HCCTABLE";

-- ARCHIVE HIGH
CREATE TABLE "HCCUSER"."HCCTABLE_CTAS_ARCHIVE_HIGH"
LOGGING TABLESPACE "TS_HCCTEST"
COMPRESS FOR ARCHIVE HIGH
AS SELECT * FROM "HCCUSER"."HCCTABLE";
```

Using CREATE TABLE from DBMS_METADATA

Extract DDL

```
-- extract info
set heading off
set echo off
set long 9999999
select dbms_metadata.get_ddl('TABLE','HCCTABLE','HCCUSER') from dual;

-- output
  CREATE TABLE "HCCUSER"."HCCTABLE"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS NOLOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST"
  PARALLEL
```

Create DDL

OLTP

```
-- modify the DDL (make sure to remove NOCOMPRESS and NOLOGGING, add
the ones highlighted in yellow)
  CREATE TABLE "HCCUSER"."HCCTABLE_OLTP"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
```

```
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST" COMPRESS FOR OLTP
  PARALLEL
/
```

Query Low

```
  CREATE TABLE "HCCUSER"."HCCTABLE_QUERY_LOW"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST" COMPRESS FOR QUERY LOW
  PARALLEL
/
```

Query High

```
  CREATE TABLE "HCCUSER"."HCCTABLE_QUERY_HIGH"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST" COMPRESS FOR QUERY HIGH
  PARALLEL
/
```

Archive Low

```
  CREATE TABLE "HCCUSER"."HCCTABLE_ARCHIVE_LOW"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
```

```
 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST" COMPRESS FOR ARCHIVE LOW
  PARALLEL
/
```

Archive High

```
  CREATE TABLE "HCCUSER"."HCCTABLE_ARCHIVE_HIGH"
   (    "OWNER" VARCHAR2(30),
        "OBJECT_NAME" VARCHAR2(128),
        "SUBOBJECT_NAME" VARCHAR2(30),
        "OBJECT_ID" NUMBER,
        "DATA_OBJECT_ID" NUMBER,
        "OBJECT_TYPE" VARCHAR2(19),
        "CREATED" DATE,
        "LAST_DDL_TIME" DATE,
        "TIMESTAMP" VARCHAR2(19),
        "STATUS" VARCHAR2(7),
        "TEMPORARY" VARCHAR2(1),
        "GENERATED" VARCHAR2(1),
        "SECONDARY" VARCHAR2(1),
        "NAMESPACE" NUMBER,
        "EDITION_NAME" VARCHAR2(30)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "TS_HCCTEST" COMPRESS FOR ARCHIVE HIGH
  PARALLEL
/
```

Insert SQLs

```
insert /*+ APPEND */ into hcctable_oltp select * from hcctable;
insert /*+ APPEND */ into hcctable_query_low select * from hcctable;
insert /*+ APPEND */ into hcctable_query_high select * from hcctable;
insert /*+ APPEND */ into hcctable_archive_low select * from hcctable;
insert /*+ APPEND */ into hcctable_archive_high select * from
hcctable;
commit;
```

Validate real compression vs estimate

```
-- first, gather stats on tables
exec dbms_stats.gather_schema_stats('HCCUSER');

-- compare the compressed tables vs the base uncompressed table
select segment_name, 0 ratio, round(sum(bytes)/1024/1024,2) size_mb
from user_segments
where segment_type = 'TABLE'
and segment_name = 'HCCTABLE'
group by segment_name
union all
SELECT comp.table_name, round(uncomp.blocks/comp.blocks,3) AS ratio,
seg.size_mb
FROM
  user_tables comp,
  (select * from user_tables where table_name = 'HCCTABLE') uncomp,
  (select segment_name, round(sum(bytes)/1024/1024,2) size_mb from
user_segments where segment_type = 'TABLE' group by segment_name) seg
WHERE comp.compression = 'ENABLED'
AND uncomp.compression = 'DISABLED'
AND seg.segment_name = comp.table_name
ORDER BY 2 ASC;

SEGMENT_NAME                     RATIO    SIZE_MB
------------------------------ ---------- ----------
HCCTABLE                              0    1233.06 <- base table
HCCTABLE_CTAS_OLTP                1.738        712 <- OLTP compressed
using CTAS
HCCTABLE_OLTP                     1.738        716 <- OLTP compressed
using DDL and INSERT/APPEND
HCCTABLE_QUERY_LOW                4.028        309 <- QUERY LOW
compressed using DDL and INSERT/APPEND
HCCTABLE_CTAS_QUERY_LOW            4.05        308 <- QUERY LOW
compressed using CTAS
HCCTABLE_CTAS_QUERY_HIGH          9.873        132 <- QUERY HIGH
compressed using CTAS
HCCTABLE_QUERY_HIGH               9.876        128 <- QUERY HIGH
compressed using DDL and INSERT/APPEND
HCCTABLE_ARCHIVE_LOW             11.365        112 <- ARCHIVE LOW
compressed using DDL and INSERT/APPEND
HCCTABLE_CTAS_ARCHIVE_LOW        11.365        112 <- ARCHIVE LOW
compressed using CTAS
HCCTABLE_CTAS_ARCHIVE_HIGH       39.847         31 <- ARCHIVE HIGH
compressed using CTAS
HCCTABLE_ARCHIVE_HIGH            39.927         31 <- ARCHIVE HIGH
```

```
compressed using DDL and INSERT/APPEND
```

These numbers can be compared to the estimate output (hcc_estimate.txt)

Validate rows inside HCC table

Range of row_ids of a table

```
$ vi hcc_comptype_rows.sql

set lines 200
col owner format a15 head "Owner"
col tabname format a35 head "Table"
col myrowid format a20 head "RowId"
col comptype format a20 head "CompType"
set echo on

select '&&owner' owner, '&&table_name' tabname, rowid myrowid,
decode(dbms_compression.get_compression_type('&&owner','&&table_name',
rowid),
        1,'No Compression',
        2,'Basic/OLTP',
        4,'HCC Query High',
        8,'HCC Query Low',
        16,'HCC Archive High',
        32,'HCC Archive Low',
        64,'Block') comptype
from "&&owner"."&&table_name"
where &&predicate
/

Enter value for owner: HCCUSER
Enter value for table_name: HCCTABLE_OLTP
Enter value for predicate: rownum < 2

Owner           Table                                RowId
CompType
--------------- ------------------------------------ ------------------
-- --------------------
HCCUSER         HCCTABLE_OLTP                        AAAdb/AAAAABt8BAAA
Basic/OLTP
```

All row_ids of a table

```
$ vi hcc_comptype_all.sql

set lines 200
col comptype format a20 head "CompType"
col cnt format 999,999,999 head "#Rows"
col pct format 999.90 head "%ofTotal"
set echo on

select comptype,count(*) cnt,100*(count(*)/rowcount) pct
from (
 select '&&owner' owner, '&&table_name' tabname, rowid myrowid,

decode(dbms_compression.get_compression_type('&&owner','&&table_name',
rowid),
      1,'No Compression', 2,'Basic/OLTP', 4,'HCC Query High',
      8,'HCC Query Low', 16,'HCC Archive High', 32,'HCC Archive Low',
        64,'Block') comptype,
 (count(*) over ()) rowcount
from "&&owner"."&&table_name"
) group by comptype,rowcount
/


CompType                    #Rows %ofTotal
-------------------- ------------ --------
Basic/OLTP             10,240,000   100.00
```

SQL Performance Instrumentation

Install get_run_stats

```
conn hccuser/hccuser
@run_stats_create.sql
```

Execute SQL

- The parameter is the HCC compressed table where a bunch of select count(*) query will be executed. The parameter also serves as the TEST_NAME identifier in the get_run_stats table

- The script can be modified to run a custom SQL. In this case the parameter will just serve as a TEST_NAME identifier.

```
$ sh hcc_test.sh HCCTABLE_OLTP

PL/SQL procedure successfully completed.


  COUNT(*)
----------
         1


PL/SQL procedure successfully completed.
```

Query get_run_stats

- Below are the runs on HCCTABLE_QUERY_LOW AND HCCTABLE_OLTP. The data shown are delta of Elapsed time, session statistics, and wait events on each test.

```
conn hccuser/hccuser
@run_stats_hcc_query.sql

TEST_NAME            BEGIN_SNAP        END_SNAP          STAT_CLASS
NAME                                                            DELTA
-------------------- ----------------- ----------------- ----------------------------------------
----- ------------------------------------------------------------ ----------
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 ELAPSED
secs - elapsed time                                                44
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 User
secs - CPU used by this session                                 22.28
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 SQL
MB/s - cell physical IO bytes eligible for predicate offload   2128.875
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 Cache
MB/s - physical read total bytes                             2128.875
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 SQL
MB/s - cell physical IO interconnect bytes                 62.7857361
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 SQL
MB/s - cell IO uncompressed bytes                          6667.69141
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 SQL
cell CUs processed for uncompressed                             64155
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 SQL
cell CUs sent uncompressed                                      64155
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Debug
EHCC Archive CUs Decompressed                                       0
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Debug
EHCC Query Low CUs Decompressed                                     0
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 Debug
EHCC Query High CUs Decompressed                                    0
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Debug
EHCC CUs Decompressed                                               0
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Concurrency - os thread startup
TIME_WAITED_MICRO                                            42831598
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Idle - PX Deq: Execute Reply
TIME_WAITED_MICRO                                              631418
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Other - events in waitclass Other
TIME_WAITED_MICRO                                               59179
HCCTABLE_QUERY_LOW   20160420 23:41:48 20160420 23:42:32 Idle - SQL*Net message from client
```

```
TIME_WAITED_MICRO                                                  8337
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Commit - log file sync
TIME_WAITED_MICRO                                                   449
HCCTABLE_QUERY_LOW   20160420 23:42:32 20160420 23:42:32 Cluster - gc current multi block request
TIME_WAITED_MICRO                                                   242

HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 ELAPSED
secs - elapsed time                                                  37
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 User
secs - CPU used by this session                                   21.58
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 SQL
MB/s - cell physical IO bytes eligible for predicate offload 4940.79688
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 Cache
MB/s - physical read total bytes                          4940.79688
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 SQL
MB/s - cell physical IO interconnect bytes                1016.67155
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 SQL
MB/s - cell IO uncompressed bytes                         4940.79688
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 SQL
cell CUs processed for uncompressed                                  0
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 SQL
cell CUs sent uncompressed                                           0
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 Debug
EHCC CUs Decompressed                                                0
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Debug
EHCC Query Low CUs Decompressed                                      0
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Debug
EHCC Archive CUs Decompressed                                        0
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Debug
EHCC Query High CUs Decompressed                                     0
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Concurrency - os thread startup
TIME_WAITED_MICRO                                              35061464
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Idle - PX Deq: Execute Reply
TIME_WAITED_MICRO                                                695284
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Idle - PX Deq: Parse Reply
TIME_WAITED_MICRO                                                112633
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Other - events in waitclass Other
TIME_WAITED_MICRO                                                 52181
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Idle - SQL*Net message from client
TIME_WAITED_MICRO                                                  8383
HCCTABLE_OLTP        20160421 13:55:29 20160421 13:56:06 Idle - PX Deq: Join ACK
TIME_WAITED_MICRO                                                  7031
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Application - enq: KO - fast object
checkpoin TIME_WAITED_MICRO                                                  2435
HCCTABLE_OLTP        20160421 13:56:06 20160421 13:56:06 Cluster - gc current multi block request
TIME_WAITED_MICRO                                                  1146
```

Re-compress/Rebuild HCC table

- To re-compress/re-organize the table there are a couple of things that can be done
  - alter table move
  - CTAS
  - DBMS_REDEFINITION

```
alter table HCCTABLE_OLTP move;

Table altered.
```

Appendix

Scripts

Get tablespace compression type

```
$ vi hcc_tablespaces.sql

set lines 200
set pages 80
set echo on
col tablespace_name format a22 head 'Tablespace'
col compress_for format a20 head 'CompressType'
col def_tab_compression format a20 head 'CompSetting'
select tablespace_name,
       def_tab_compression,
       nvl(compress_for,'NONE') compress_for
from dba_tablespaces;

Tablespace             CompSetting          CompressType
---------------------- -------------------- --------------------
SYSTEM                 DISABLED             NONE
SYSAUX                 DISABLED             NONE
UNDOTBS1               DISABLED             NONE
TEMP                   DISABLED             NONE
USERS                  DISABLED             NONE
UNDOTBS2               DISABLED             NONE
EXAMPLE                DISABLED             NONE
TS_IOSATURATIONTOOLKIT DISABLED             NONE
IOPS                   DISABLED             NONE
SCRATCH                DISABLED             NONE
TS_HCCTEST             DISABLED             NONE
TS_SCRATCH             DISABLED             NONE

12 rows selected.
```

Get table compression type

```
$ vi hcc_tables.sql

select table_name,num_rows,blocks,compression,compress_for
from dba_tables
where owner='HCCUSER'
and compression = 'ENABLED';

TABLE_NAME                     NUM_ROWS     BLOCKS COMPRESS
COMPRESS_FOR
------------------------------ ---------- ---------- -------- --------
----
HCCTABLE_OLTP                   10240000      90807 ENABLED  OLTP
HCCTABLE_QUERY_HIGH             10240000      15981 ENABLED  QUERY
HIGH
HCCTABLE_QUERY_LOW              10240000      39186 ENABLED  QUERY
LOW
HCCTABLE_ARCHIVE_LOW            10240000      13888 ENABLED  ARCHIVE
LOW
HCCTABLE_ARCHIVE_HIGH           10240000       3953 ENABLED  ARCHIVE
HIGH
HCCTABLE_CTAS_OLTP              10240000      90801 ENABLED  OLTP
HCCTABLE_CTAS_QUERY_LOW         10240000      38973 ENABLED  QUERY
LOW
HCCTABLE_CTAS_QUERY_HIGH        10240000      15987 ENABLED  QUERY
HIGH
HCCTABLE_CTAS_ARCHIVE_LOW       10240000      13888 ENABLED  ARCHIVE
LOW
HCCTABLE_CTAS_ARCHIVE_HIGH      10240000       3961 ENABLED  ARCHIVE
HIGH

10 rows selected.
```