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](#_Setup_environment)
* Then [estimate the compression](#_Estimate_compression)
* Then [create the actual HCC tables](#_Creating_HCC_object)
* Check the [estimate vs real](#_Validate_real_compression)
* and [validate the rows](#_Validate_rows_inside)
* [Run some sample SQLs](#_SQL_Performance_Instrumentation) on HCC tables and instrument each run
* You can also run some DML on the tables and [validate the rows](#_Validate_rows_inside)
* Lastly [re-compress](#_Re-compress/Rebuild_HCC_table)

# 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)](#_View_hcc_estimate) |

# 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. |