19c ORACLE Database

Getting Started with Advanced Row Compression and Advanced Index Compression

ORACLE WHITE PAPER | FEBRUARY 2019

ORACLE

Table of Contents

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Oracle Database Compression

Oracle Advanced Compression includes a comprehensive set of compression capabilities to help organizations maximize resource utilization and reduce costs. It allows IT administrators to significantly reduce their overall database storage footprint, and improve performance, by enabling compression for all types of data, including:

Advanced Row Compression

Enables table data to be compressed during all types of data manipulation operations, including DML INSERT and UPDATE operations -- intelligent algorithm minimizes compression overhead during write operations, thereby making compression viable for both data warehouse and OLTP workloads.

Advanced LOB Compression

Provides compression for LOBS managed by Oracle SecureFiles – a high performance and powerful infrastructure for managing unstructured data such as images, documents, videos and more.

Advanced Index Compression

Reduces the size of all supported unique and non-unique indexes- automatically chooses the right compression per index block. Advanced Index Compression provides significant space savings while also improving performance for queries that are executed using indexes.

Compression Advisor

An easy way to get started, with Advanced Compression, is by using compression advisor. The "DBMS_COMPRESSION" PL/SQL package (commonly called compression advisor) gathers compression-related information within a database environment. This includes estimating the compressibility of both partitioned, and non-partitioned tables, and gathering row-level compression information on previously compressed tables/partitions. Compression advisor provides organizations with the storage reduction information needed to make compression-related usage decisions.

The output of running compression advisor is an estimation of the compression ratio for the specific table or partition that was the target of compression advisor. The output indicates the "COMPRESSION RATIO" presented as a number such as 2.1. This number indicates that, for this specific table or partition, the estimated compression ratio is 2.1x, which represents about a 50% reduction in the footprint of the table or partition should compression be enabled.

The compression ratio achieved in a given environment depends on the data being compressed, specifically the cardinality of the data. In general, organizations can expect to reduce their storage space consumption by a factor of 2x to 4x by using Advanced Row Compression. That is, the amount of space consumed by uncompressed data will be two to four times larger than that of the compressed data.

A version of compression advisor, which supports Oracle Database 9*i* Release 2 through 11g Release 1, is available on the Oracle Technology Network website (see here). This version only can report the compression ratio for data tables – those tables (and partitions) that would be targets for OLTP Table

Compression. Another version of the DBMS_COMPRESSION PL/SQL package is included with Oracle Database 11g Release 2 and above. This version can report the compression ratio for data tables (targets for Advanced Row Compression), LOBS using SecureFiles (targets for Advanced LOB Compression) and indexes (targets for Advanced Index Compression).

Both versions are free to use with Oracle Database Enterprise Edition.

Using Advisor – The GET_COMPRESSION_RATIO Procedure

In this document, we will focus on using the PL/SQL package named DBMS_COMPRESSION.GET_COMPRESSION_RATIO to estimate compression ratios based on analysis of a sample of data.

When using the GET_COMPRESSION_RATIO procedure, different constants are specified, as parameters, to selectively analyze different compression types, including: (see here)

Constant	Type	Value	Description
COMP_NOCOMPRESS	NUMBER	1	No compression
COMP_ADVANCED	NUMBER	2	Advanced row compression
COMP_QUERY_HIGH	NUMBER	4	High for query warehouse compression (Hybrid Columnar Compression)
COMP_QUERY_LOW	NUMBER	8	Low for query warehouse compression (Hybrid Columnar Compression)
COMP_ARCHIVE_HIGH	NUMBER	16	High archive compression (Hybrid Columnar Compression)
COMP_ARCHIVE_LOW	NUMBER	32	Low archive compression (Hybrid Columnar Compression)
COMP_BLOCK	NUMBER	64	Compressed block
COMP_LOB_HIGH	NUMBER	128	High compression level for LOB operations
COMP_LOB_MEDIUM	NUMBER	256	Medium compression level for LOB operations
COMP_LOB_LOW	NUMBER	512	Low compression level for LOB operations
COMP_INDEX_ADVANCED_HIGH	NUMBER	1024	High compression level for indexes
COMP_INDEX_ADVANCED_LOW	NUMBER	2048	Low compression level for indexes

Figure 1: DBMS_COMPRESSION Constants - Compression Types

Compression advisor typically provides fairly accurate estimates, of the actual compression results that may be obtained, after implementing compression.

Below are syntax examples are of the GET_COMPRESSION_RATIO procedures that analyze the compression ratio of a table, index or LOB and provides information about the compressibility of the object.

Syntax for GET_COMPRESSION_RATIO for an object (table or index, default is table):

```
DBMS_COMPRESSION.GET_COMPRESSION_RATIO (
 scratchtbsname          IN     VARCHAR2,
 ownname                 IN     VARCHAR2,
 objname                 IN     VARCHAR2,
 subobjname              IN     VARCHAR2,
 comptype                IN     NUMBER,
 blkcnt_cmp              OUT    PLS_INTEGER,
 blkcnt_uncmp            OUT    PLS_INTEGER,
 row_cmp                 OUT    PLS_INTEGER,
 row_uncmp               OUT    PLS_INTEGER,
 cmp_ratio               OUT    NUMBER,
 comptype_str            OUT    VARCHAR2,
 subset_numrows          IN     NUMBER  DEFAULT COMP_RATIO_MINROWS,
 objtype                 IN     PLS_INTEGER DEFAULT OBJTYPE_TABLE);
```

Syntax for GET_COMPRESSION_RATIO for LOBs:

```
DBMS_COMPRESSION.GET_COMPRESSION_RATIO (
 scratchtbsname          IN     VARCHAR2,
 tabowner                IN     VARCHAR2,
 tabname                 IN     VARCHAR2,
 lobname                 IN     VARCHAR2,
 partname                IN     VARCHAR2,
 comptype                IN     NUMBER,
 blkcnt_cmp              OUT    PLS_INTEGER,
 blkcnt_uncmp            OUT    PLS_INTEGER,
 lobcnt                  OUT    PLS_INTEGER,
 cmp_ratio               OUT    NUMBER,
 comptype_str            OUT    VARCHAR2,
 subset_numrows          IN     number DEFAULT COMP_RATIO_LOB_MAXROWS);
```

Syntax for GET_COMPRESSION_RATIO for all indexes on a table:

```
DBMS_COMPRESSION.GET_COMPRESSION_RATIO (
 scratchtbsname          IN     VARCHAR2,
 ownname                 IN     VARCHAR2,
 tabname                 IN     VARCHAR2,
 comptype                IN     NUMBER,
 index_cr                OUT    DBMS_COMPRESSION.COMPRECLIST,
 comptype_str            OUT    VARCHAR2,
 subset_numrows          IN     NUMBER DEFAULT COMP_RATIO_INDEX_MINROWS);
```

Usage Example: Advanced Row Compression

The example below runs compression advisor, for Advanced Row Compression (COMP_ADVANCED), on a small default table (SH.SALES) that is included with Oracle Database.

For additional information about using DBMS_COMPRESSION, please see the Oracle Database documentation here.

```
set serveroutput on
DECLARE
blkcnt_cnt pls_integer;
blkcnt_uncmp pls_integer;
row_cmp pls_integer;
row_uncmp pls_integer;
cmp_ratio pls_integer;
comptype_str varchar2(100);
BEGIN
DBMS_COMPRESSION.GET_COMPRESSION_RATIO ('USERS', 'SH', 'SALES', '',
 DBMS_COMPRESSION.COMP_ADVANCED, blkcnt_cmp, blkcnt_uncmp, row_cmp,
 row_uncmp, cmp_ratio, cmptype_str);
DBMS_OUTPUT.PUT_LINE('Block count compressed = '|| blkcnt_cmp);
DBMS_OUTPUT.PUT_LINE('Block count uncompressed = '|| blkcnt_uncmp);
DBMS_OUTPUT.PUT_LINE('Row count per block compressed = '|| row_cmp);
DBMS_OUTPUT.PUT_LINE('Row count per block uncompressed = '|| row_uncmp);
DBMS_OUTPUT.PUT_LINE('Compression type = '|| comptype_str);
DBMS_OUTPUT.PUT_LINE('Compression ratio = '|| cmp_ratio);
END;
 /
```

Figure 2: Example DBMS_COMPRESSION usage

It is important to note that compression advisor builds two temporary tables (for comparison purposes) as part of the estimation process for Advanced Row Compression (Hybrid Columnar Compression uses four tables). The temporary tables are created using the prefix 'cmp3$' and/or 'cmp4$' and are dropped by the compression advisor when no longer required. Although these temporary tables are removed after compression advisor completes, you will need free space equivalent to the sampling space specified.

The example below, the result of running the advisor code above, shows the type of output that is provided by compression advisor.

```
Block count compressed = 161
Block count uncompressed = 427
Row count per block compressed = 555
Row count per block uncompressed = 209
Compression type = "Compress Advanced"
Compression ratio = 2.65
```

Figure 3: Example DBMS_COMPRESSION output

The COMPRESSION RATIO, determined by compression advisor, and as indicated above is 2.65x in this example.

In general,

- OLTP Table Compression and Advanced Row Compression users can typically expect compression ratios in the range of 2x to 4x
- Hybrid Columnar Compression users can typically expect compression ratios in the range of 6x to 15x
- Advanced Index Compression users can typically expect compression ratios in the range of 2x to 4x
- Advanced LOB Compression users can typically expect compression ratios in the range of 2x to 3x

That is, for example, for users of Advanced Row Compression the amount of space consumed by uncompressed data will be two to four times larger than that of the compressed data. Note that the compression ratio achieved, in a given environment, depends on the nature of the data being compressed.

Compression Advisor Best Practices

- If you get this type of message when estimating Hybrid Columnar Compression:

  ORA-12801: error signaled in parallel query server P002
  ORA-64307: Exadata Hybrid Columnar Compression is not supported for tablespaces on this storage type
  Solution: Disable parallel processing for the session (set parallel_max_servers=0)
- Tables residing in uniform tablespaces can be compressed. However, the compression adviser has the restriction that the scratch tablespace cannot be uniform
- In earlier releases, Oracle did require 1M rows in a table for estimating HCC compression ratios with compression advisor – this restriction is lifted in Oracle Database release 12.1.0.2 and above.
- Outside advisor, here are no restrictions with Hybrid Columnar Compression in regards to the minimal amount of data needed (in tables/partitions) with HCC.
- For more information, and examples, about compression advisor, please see this MOS note:
  - How to Use DBMS_COMPRESSION.GET_COMPRESSION_RATIO in 12c (Doc ID 1589879.1)

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Hardware and Software, Engineered to Work Together