Automatic Data Optimization

Saving Space
and Improving Performance!

Erik Benner, Enterprise Architect
Who am I?

Erik Benner
@erik_benner
TalesFromTheDatacenter.com

Enterprise Architect
Ebenner@mythics.com

- Published Author
- IOUG Solaris SIG Leader
- RAC Attack Ninja
- First Version of Oracle…7
- Linux since 1992
- Solaris since 1996
- OEM 12c since 2011
- Oracle Database Appliance since 2011
- DB 12c in prelaunch 2013
Growth in Data Diversity and Usage

1,800 Exabytes of Data in 2011, 20x Growth by 2020

Today’s Drivers

- **Enterprise**
  - 45% per year growth in database data

- **Cloud**
  - 80% of new applications and their data

- **Regulation**
  - 300 exabytes in archives by 2015

Emerging Growth Factors

- **Mobile**
  - #1 Internet access device in 2013

- **Big Data**
  - Large customers top 50PB

- **Social Business**
  - $30B/year in commerce by 2015
I know how to divide. What I want to learn is how to divide and conquer. That could come in handy one day.
The Concept of Partitioning

Simple Yet Powerful

Large Table
Difficult to Manage

Partition
Divide and Conquer
Easier to Manage
Improve Performance

Composite Partition
Better Performance
More flexibility to match business needs

Transparent to applications
What Can Be Partitioned?

- Tables
  - Heap tables
  - Index-organized tables
- Indexes
  - Global Indexes
  - Local Indexes
- Materialized Views
Partition for Tiered Storage

<table>
<thead>
<tr>
<th>ORDERS TABLE (10 years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
</tr>
</tbody>
</table>

95% Less Active

Low End Storage Tier
2-3x less per terabyte

5% Active

High End Storage Tier
WHAT ARE YOU DOING?

I'M COMPRESSING MY BANANAS!
# Compression Techniques

<table>
<thead>
<tr>
<th>COMPRESSION TYPE:</th>
<th>SUITABLE FOR:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Compression</td>
<td>“Mostly read” tables and partitions in Data Warehouse environments or “inactive” data partitions in OLTP environments</td>
</tr>
<tr>
<td>Advanced Row Compression</td>
<td>Active tables and partitions in OLTP and Data Warehouse environments</td>
</tr>
<tr>
<td>Advanced LOB Compression and Deduplication</td>
<td>Non-relational data in OLTP and Data Warehouse environments</td>
</tr>
<tr>
<td>Advanced Network Compression and Data Guard Redo Transport Compression</td>
<td>All environments</td>
</tr>
<tr>
<td>RMAN/Data Pump Backup Compression</td>
<td>All environments</td>
</tr>
<tr>
<td>Index Key Compression</td>
<td>Indexes on tables for OLTP and Data Warehouse</td>
</tr>
<tr>
<td>Hybrid Columnar Compression – Warehouse Level</td>
<td>“Mostly read” tables and partitions in Data Warehouse environments</td>
</tr>
<tr>
<td>Hybrid Columnar Compression – Archive Level</td>
<td>“Inactive” data partitions in OLTP and Data Warehousing environments</td>
</tr>
</tbody>
</table>
Advanced Row Compression

• **Partition/table/tablesce data compression**
  – Support for conventional DML Operations (INSERT, UPDATE)
  – *Customers indicate that 2x to 4x compression ratio’s typical*

• **Significantly eliminates/reduces write overhead of DML’s**
  – Batched compression minimizes impact on transaction performance

• **“Database-Aware” compression**
  – Does not require data to be uncompressed – keeps data compressed in memory
  – Reads often see improved performance due to fewer I/Os and enhanced memory efficiency
Hybrid Columnar Compression

- **Hybrid Columnar Compressed Tables**
  - Compressed tables can be modified using conventional DML operations
  - Useful for data that is bulk loaded and queried

- **How it Works**
  - Tables are organized into Compression Units (CUs)
    - CUs are multiple database blocks
  - Within Compression Unit, data is organized by column instead of by row
    - Column organization brings similar values close together, enhancing compression

10x to 40x Reduction
REALLY!
Data Compression

- Reduce storage footprint, read compressed data faster

<table>
<thead>
<tr>
<th>Hot Data</th>
<th>Warm Data</th>
<th>Archive Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1010101011010100110101</td>
<td>1010101011010100110101</td>
<td>1010101011010100110101</td>
</tr>
<tr>
<td>1100001010001101101101</td>
<td>1010010010010000100010</td>
<td>1010010010010000100010</td>
</tr>
<tr>
<td>1010010010010000100010</td>
<td>010101101010101001101101</td>
<td>010101101010101001101101</td>
</tr>
<tr>
<td>010101101010101001101101</td>
<td>0001010010010000100100</td>
<td>0001010010010000100100</td>
</tr>
</tbody>
</table>

3X
Advanced Row Compression

10X
Columnar Query Compression

15X +
Columnar Archive Compression
Compression Benefits

• **Transparent: 100% Application Transparent**

• **Smaller: Reduces Footprint**
  – CapEx: Reduces server & storage costs for primary, standby, backup, test & dev databases ...
  – OpEx: Reduces heating, cooling, floor space costs ...
    • Additional ongoing savings over life of a database as database grows in size

• **Faster: Transactional, Analytics, DW**
  – Greater speedup from in-memory: 3-10x more data fits in buffer cache & flash cache
  – Faster queries
  – Faster backup & restore speeds

• **End-to-end Cost / Performance Benefits across CPU, DRAM, Flash, Disk & Network**
Automatic Data Optimization (ADO)

- ADO offers capability to move and/or compress data based on observed usage patterns.
- Uses heat maps to determine how often data has been accessed.
- Tracks exactly how data has been utilized (DML vs. query, single-block vs. table scan).
- Data usage patterns can be tracked at tablespace, segment, and row level.
## Automatic Data Optimization

**Simple Declarative SQL extension**

<table>
<thead>
<tr>
<th>Access Level</th>
<th>Optimization Features</th>
<th>Timeframe</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active</strong></td>
<td>Advanced Row Compression (2-4x)</td>
<td>row store compress advanced row after 2 days of no update</td>
</tr>
<tr>
<td></td>
<td>Affects ONLY candidate rows</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cached in DRAM &amp; FLASH</td>
<td></td>
</tr>
<tr>
<td><strong>Frequent Access</strong></td>
<td>Warehouse Compression (10x)</td>
<td>column store compress for query low after 1 week of no update</td>
</tr>
<tr>
<td></td>
<td>High Performance Storage</td>
<td></td>
</tr>
<tr>
<td><strong>Occasional Access</strong></td>
<td>Warehouse Compression (10x)</td>
<td>tier to lower cost tablespace</td>
</tr>
<tr>
<td></td>
<td>High Capacity Storage</td>
<td></td>
</tr>
<tr>
<td><strong>Dormant</strong></td>
<td>Archive Compression (15-40X)</td>
<td>column store compress for archive high after 6 months of no access</td>
</tr>
<tr>
<td></td>
<td>Archival Storage</td>
<td></td>
</tr>
</tbody>
</table>
Heat Maps: “How Hot Is It?”

ADO leverages heat maps to:

• Capture data *usage frequencies*
• Determine which *compression level* is most appropriate for *how* data is being used
• Determine which data could be *moved* from a high-performance *storage tier* to a lower-performance tier
• Decide when data should be moved between *different tablespaces or partitions* to limit possible *out-of-space* conditions
Heat Maps: An Example

Heat map shows heavy DML and queries:
Leave data uncompressed

After 7 days of more limited access:
Enable ADVANCED compression

After 30 days of only limited access:
Enable HCC QUERY LOW* compression

After 90 days of no access:
Enable HCC ARCHIVE HIGH* compression

* Requires Exadata Storage Servers, ZFS Appliance, or Pillar Axiom storage
How much space will we save?
DBMS_COMPRESSION

Compliments of Oracle

The DBMS_COMPRESSION package gathers compression-related information within a database environment. This includes tools for estimating compressibility of a table for both partitioned and non-partitioned tables, and gathering row-level compression information on previously compressed tables. This gives the user with adequate information to make compression-related decisions.

Built into 12c, can be downloaded for 11gR2

The Official docs

http://docs.oracle.com/database/121/ARPLS/d_compress.htm#ARPLS236
set serveroutput on

DECLARE
  blkcnt_cmp  PLS_INTEGER;
  blkcnt_uncmp  PLS_INTEGER;
  row_comp  PLS_INTEGER;
  row_uncmp  PLS_INTEGER;
  cmp_ratio  NUMBER;
  comptype  VARCHAR2(30);
BEGIN
  dbms_compression.get_compression_ratio('ADO', 'ADO', 'ADO_DEMO', NULL,
                                         32, blkcnt_cmp,
                                         blkcnt_uncmp, row_comp, row_uncmp, cmp_ratio, comptype);

  dbms_output.put_line('Block Count Compressed: ' || TO_CHAR(blkcnt_cmp));
  dbms_output.put_line('Block Count UnCompressed: ' || TO_CHAR(blkcnt_uncmp));
  dbms_output.put_line('Compression Type: ' || comptype);
END;
/

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMP_NOCOMPRESS</td>
<td>1</td>
<td>No compression</td>
</tr>
<tr>
<td>COMP_FOR OLTP</td>
<td>2</td>
<td>OLTP compression</td>
</tr>
<tr>
<td>COMP_FOR_QUERY_HIGH</td>
<td>4</td>
<td>High compression level for query operations</td>
</tr>
<tr>
<td>COMP_FOR_QUERY_LOW</td>
<td>8</td>
<td>Low compression level for query operations</td>
</tr>
<tr>
<td>COMP_FOR_ARCHIVE_HIGH</td>
<td>16</td>
<td>High compression level for archive operations</td>
</tr>
<tr>
<td>COMP_FOR_ARCHIVE_LOW</td>
<td>32</td>
<td>Low compression level for archive operations</td>
</tr>
</tbody>
</table>
Execution

SQL> @ado
Compression Advisor self-check validation successful. select count(*) on both
Uncompressed and EHCC Compressed format = 1000001 rows
Block Count Compressed: 8941
Block Count UnCompressed: 37986
Compression Type: "Compress Archive Low"

PL/SQL procedure successfully completed.