An Oracle White Paper
August 2012

Advanced Compression Option (ACO) with Oracle Database 11g

Introduction

Enterprises are experiencing an explosion in the volume of data required to effectively run their businesses. This trend in data growth can be attributed to several key factors. Recent changes in the regulatory landscape, such as Sarbanes-Oxley and HIPAA, are contributing to this trend by mandating that enterprises retain large amounts of information for long periods of time.

Mass distribution of rich and multimedia content over the Internet, made possible through advancements in broadband technologies, also contributes to the growth in overall data volume. Further fueling the exponential trend in data growth is the advent of Web 2.0, with collaborative applications that encourage enormous amounts of user-generated content. Various estimates indicate that data volume is almost doubling every 2-3 years.

This sudden explosion in data volume presents a daunting management challenge for IT administrators. First and foremost are the spiraling storage costs: even though the cost per MB of storage has been declining dramatically in the last few years, the enormous growth in the volume of data that needs to be retained online makes storage one of the biggest cost elements of most IT budgets. In addition, application scalability and performance must continue to meet the demands of the business – even as data volumes explode.

Oracle Database 11g Release 1 introduced the Advanced Compression Option to help customers cope with these challenges. Innovations in Oracle compression technologies help customers reduce the resources and costs of managing large data volumes. In addition to OLTP Table Compression, the Advanced Compression Option includes a comprehensive set of compression capabilities to help customers maximize resource utilization and reduce costs by enabling compression for unstructured (SecureFiles deduplication and compression), backups (RMAN and Data Pump) and for Data Guard Redo Log network transport. The Advanced Compression Option also includes Oracle Total Recall, which increases security and reduces the cost of storing and accessing historical data.

Oracle Advanced Compression

The Oracle Database 11g Advanced Compression Option introduces a comprehensive set of compression capabilities to help customers maximize resource utilization and reduce costs. It allows IT administrators to significantly reduce their overall database storage footprint by enabling compression for all types of data – be it relational (table), unstructured (file), or backup data. Although storage cost savings are often seen as the most tangible benefit of compression, innovative technologies included in the Advanced Compression Option are designed to reduce resource requirements and technology costs for all components of your IT infrastructure, including memory and network bandwidth.

Compression for Table Data

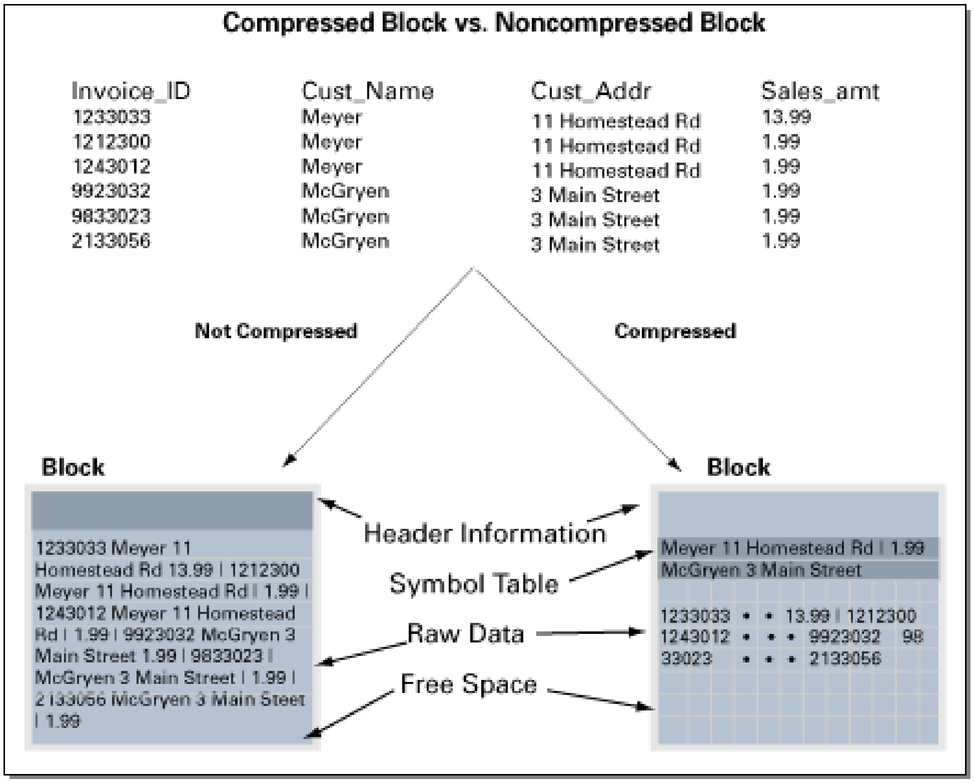
Oracle has been a pioneer in database compression technology. Oracle Database 9i introduced Basic Table Compression several years ago that compressed data that was loaded using bulk load operations. Oracle Database 11g Release 1 introduced a new feature called OLTP Table Compression that allows data to be compressed during all types of data manipulation operations, including conventional DML such as INSERT and UPDATE. In addition, OLTP Table Compression reduces the associated compression overhead of write operations making it suitable for transactional or OLTP environments as well. OLTP Table Compression, therefore, extends the benefits of compression to all application workloads.

It should be noted that Basic Table Compression is a base feature of Oracle Database 11g Enterprise Edition (EE). OLTP Table Compression is a part of the Oracle Advanced Compression option, which requires a license in addition to the Enterprise Edition.

OLTP Table Compression

Oracle's OLTP Table Compression uses a unique compression algorithm specifically designed to work with OLTP applications. The algorithm works by eliminating duplicate values within a database block, even across multiple columns. Compressed blocks contain a structure called a symbol table that maintains compression metadata. When a block is compressed, duplicate values are eliminated by first adding a single copy of the duplicate value to the symbol table. Each duplicate value is then replaced by a short reference to the appropriate entry in the symbol table.

Through this innovative design, compressed data is self-contained within the database block as the metadata used to translate compressed data into its original state is stored in the block. When compared with competing compression algorithms that maintain a global database symbol table, Oracle's unique approach offers significant performance benefits by not introducing additional I/O when accessing compressed data.



Compressed Block vs. Noncompressed Block

Invoice_ID	Cust_Name	Cust_Addr	Sales_amt
1233033	Meyer	11 Homestead Rd	13.99
1212300	Meyer	11 Homestead Rd	1.99
1243012	Meyer	11 Homestead Rd	1.99
9923032	McGryen	3 Main Street	1.99
9833023	McGryen	3 Main Street	1.99
2133056	McGryen	3 Main Street	1.99

Not Compressed
Compressed
Block
Block
1233033 Meyer 11 Homestead Rd 13.99 | 1212300 Meyer 11 Homestead Rd | 1.99 | 1243012 Meyer 11 Homestead Rd | 1.99 | 9923032 McGryen 3 Main Street 1.99 | 9833023 | McGryen 3 Main Street | 1.99 | 2133056 McGryen 3 Main Steet | 1.99
Header Information
Symbol Table
Raw Data
Free Space
Meyer 11 Homestead Rd | 1.99
McGryen 3 Main Street
1233033 • • 13.99 | 1212300
1243012 • • • 9923032 98
33023 • • • 2133056

Figure 1: Compressed Block vs. Non-compressed Block

Benefits of OLTP Table Compression

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

The benefits of OLTP Table Compression go beyond just on-disk storage savings. One significant advantage is Oracle's ability to read compressed blocks directly without having to first uncompress the block. Therefore, there is no measurable performance degradation for accessing compressed data. In fact, in many cases performance may improve due to the reduction in I/O since Oracle will have to access fewer blocks. Further, the buffer cache will become more efficient by storing more data without having to add memory.

Minimal Performance Overhead

As stated above, OLTP Table Compression has no adverse impact on read operations. There is additional work performed while writing data, making it impossible to eliminate performance overhead for write operations. However, Oracle has put in a significant amount of work to minimize this overhead for OLTP Table Compression. Oracle compresses blocks in batch mode rather than compressing data every time a write operation takes place. A newly initialized block remains uncompressed until data in the block reaches an internally controlled threshold. When a transaction causes the data in the block to reach this threshold, all contents of the block are compressed. Subsequently, as more data is added to the block and the threshold is again reached, the entire block is recompressed to achieve the highest level of compression.

This process repeats until Oracle determines that the block can no longer benefit from further compression. Only transactions that trigger the compression of the block will experience the slight compression overhead. Therefore, a majority of OLTP transactions on compressed blocks will have the exact same performance as they would with uncompressed blocks.

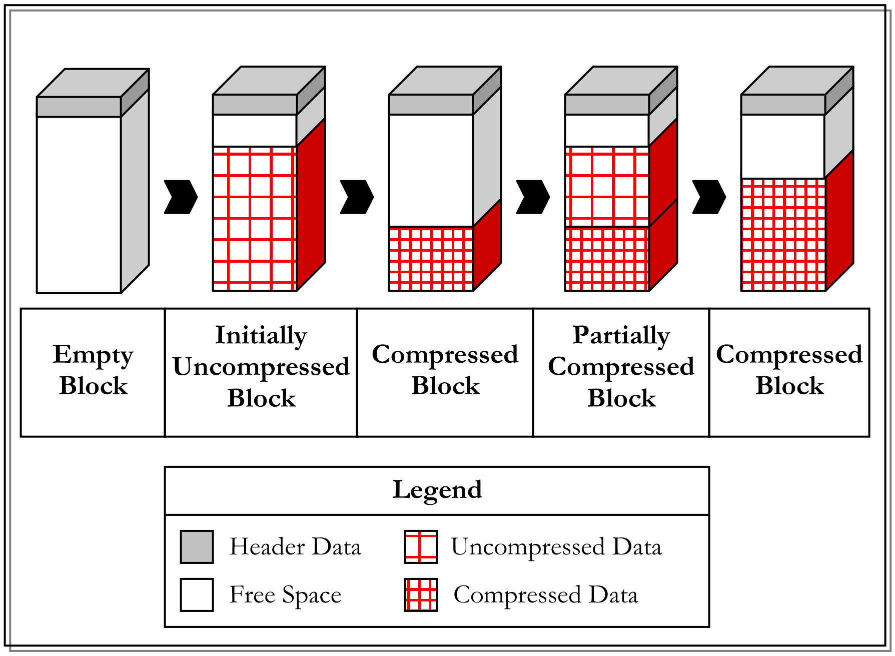


Figure 2 OLTP Table Compression Process

Migration and Best Practices

For new tables and partitions, enabling OLTP Table Compression is as easy as simply CREATEing the table or partition and specifying "COMPRESS FOR OLTP". See the example below:

```
CREATE TABLE emp (emp_id NUMBER, first_name VARCHAR2(128), last_name VARCHAR2(128))
COMPRESS FOR OLTP;
```

For existing tables and partitions, there are three recommended approaches to enabling OLTP Table Compression:

1. **ALTER TABLE … COMPRESS FOR OLTP**
   - This approach will enable OLTP Table Compression for all future DML -- however, the existing data in the table will remain uncompressed.
2. **Online Redefinition (DBMS_REDEFINITION)**
   - This approach will enable OLTP Table Compression for future DML and also compress existing data. Using DBMS_REDEFINITION keeps the table online for both read/write activity during the migration. Run DBMS_REDEFINITION in parallel for best performance.
   - Online redefinition will clone the indexes to the interim table during the operation. All the cloned indexes are incrementally maintained during the sync (refresh) operation so there is no interrupt in the use of the indexes during, or after, the online redefinition. The only exception is when online redefinition is used for redefining a partition -- the global index is invalidated and needs to be rebuilt after the online redefinition.
3. **ALTER TABLE … MOVE COMPRESS FOR OLTP**
   - This approach will enable OLTP Table Compression for future DML and also compress existing data. While the table is being moved it is online for read activity but has an exclusive (X) lock – so all DML will be blocked until the move command completes. Run ALTER TABLE…MOVE in parallel for best performance.
   - The ALTER TABLE...MOVE statement allows you to relocate data of a non-partitioned table, or of a partition of a partitioned table, into a new segment,

and optionally into a different tablespace. ALTER TABLE…MOVE COMPRESS compresses the data by creating new extents for the compressed data in the tablespace being moved to -- it is important to note that the positioning of the new segment can be anywhere within the datafile, not necessarily at the tail of the file or head of the file. When the original segment is released, depending on the location of the extents, it may or may not be possible to shrink the datafile.

- ALTER TABLE… MOVE will invalidate any indexes on the partition or table; those indexes will need to be rebuilt after the ALTER TABLE… MOVE. For partition moves, the use of ALTER TABLE… MOVE PARTITION with the UPDATE INDEXES clause will maintain indexes (it places an exclusive (X) lock so all DML will be blocked until the move command completes) – not available for non-partitioned tables.

Below are some best practices and considerations for the capabilities that are included as part of the Advanced Compression Option:

- The best test environment for each Advanced Compression capability is where you can most closely duplicate the production environment– this will provide the most realistic (pre- and post- compression) performance comparisons.

- Space usage reduction with OLTP Table Compression enabled gives the best results where the most duplicate data is stored (low cardinality). This is especially true for backups -- greater compression will result in less data backed up and hence shorter recovery time. Sorting data (on the columns with the most duplicates) prior to bulk loads may increase the compression ratio.

- Regarding whether or not to compress at the Tablespace level: For custom applications, we recommend compressing at the Tablespace level, but users should consider turning off compression on very high traffic or very small tables, such as tables used as queues. For commercial packaged applications, where typically the number of objects can be very large, the recommended approach is object selection instead of exclusion. Often the top hundred largest tables and indexes consume the majority of the database space. Compressing those objects, while excluding high traffic objects like tables used as queues, will give the majority of the compression benefits. Other objects can be compressed over time as needed. Note that prefix compression is included with Oracle

Database Enterprise Edition at no extra cost - it does not require licensing the Advanced Compression Option.

- Although CPU overhead is typically minimal, implementing Oracle Advanced Compression is ideal on systems with available CPU cycles, as compression will have additional, although minor overhead for some DML operations.

- Oracle Advanced Compression Advisor is a PL/SQL package that is used to estimate potential storage savings for OLTP Table Compression based on analysis of a sample of data. It provides a good estimate of the actual results that may be obtained after implementing Oracle Advanced Compression's OLTP Table compression feature. Oracle Advanced Compression Advisor, which supports Oracle Database 9*i* Release 2 through 11g Release 1, is available for free on the Oracle Technology Network website. The Advanced Compression Advisor is built in to Oracle Database 11g Release 2.

- OLTP Table Compression is NOT supported for use with tables that have more than 255 columns or that have LONG data types.

- Larger blocks don't always ensure higher OLTP Table Compression ratios. Testing with your own data is suggested if you want to determine if larger/smaller block sizes have an impact on your OLTP Table Compression ratio.

- LOBs are best stored in the Oracle Database as SecureFiles LOBs, and if the customer has licensed the Advanced Compression Option, they can use SecureFiles Compression and Deduplication to potentially reduce the amount of storage required for LOBs.

Compression for File Data

SecureFiles, a feature included in Oracle Database 11g, offers a 'best-of-both-worlds' architecture for storing unstructured content, such as documents, spreadsheets and XML files. SecureFiles is specifically engineered to deliver high performance for file data comparable to that of traditional file systems while retaining the advantages of the Oracle database.

SecureFiles is designed as a superset of the ANSI standard LOB data type and offers easy migration from existing BasicFile LOBs, the precursor to SecureFiles. With SecureFiles, organizations can now manage all relational data and associated file data in Oracle using a single security/audit model, a unified backup & recovery process, and perform seamless retrievals across all information.

The Advanced Compression Option of Oracle Database 11g includes compression and deduplication technologies that drastically reduce the storage footprint of SecureFiles data.

SecureFiles Deduplication

It is extremely common for applications to store exact replicas of files. A typical example is an email application where multiple users may receive the same attachment. SecureFiles Deduplication is an intelligent technology included with the Advanced Compression option that eliminates duplicate copies of SecureFiles data. Oracle stores one image of the SecureFiles data and replaces the duplicate copies with references to this image.

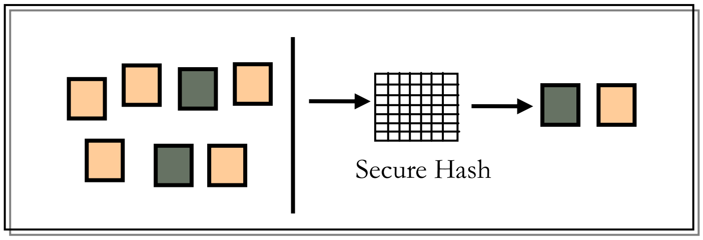


Figure 3 SecureFiles Deduplication

Consider an email application where 10 users receive an email with the same 1MB attachment. Without SecureFiles Deduplication, the system would store one copy of the file for each of the 10 users – requiring 10MB of storage. If the email application in our example had used SecureFiles with Deduplication, it would have stored the 1MB attachment just once. That's a 90% savings in storage requirements.

In addition to the storage savings, SecureFiles Deduplication also increases application performance. Specifically, write and copy operations are much more efficient since only references to the SecureFiles image are written. Further, read operations may improve if duplicate SecureFiles data already exists in the buffer cache.

SecureFiles Compression

The Advanced Compression Option of Oracle Database 11g provides yet another mechanism to control the size of your SecureFiles data. In addition to Deduplication discussed earlier, SecureFiles Compression utilizes industry standard compression algorithms to further minimize the storage requirements of SecureFiles data.

With SecureFiles Compression, typical files such as documents or XML files, experience a reduction of 2x to 3x times in size. Using built-in intelligence, SecureFiles Compression automatically avoids compressing data that would not benefit from compression – for instance a document that was compressed via a 3rd party tool before being inserted into the database as a SecureFiles file. Applications are still able to perform random reads and writes on compressed SecureFiles data since the compressed data is broken down into small chunks of data. This can vastly improve performance when compared with compressing entire files before inserting them into the database.

There are three levels of SecureFiles compression available: LOW, MEDIUM, and HIGH. By default, SecureFiles Compression uses the MEDIUM level, which typically provides good compression with a modest CPU overhead of 3-5%. SecureFiles Compression LOW, new in Oracle Database 11g Release 2, is optimized for high performance. SecureFiles Compression LOW actually maintains about 80% of the compression achieved through MEDIUM, while utilizing 3x less CPU. Finally, SecureFiles Compression HIGH achieves the highest storage savings but incurs the most CPU overhead.

For more information about SecureFiles and LOB storage, please see the *Oracle® Database SecureFiles and Large Objects Developer's Guide.*

Compression for Backup Data

In addition to compressing data stored inside the database, Oracle Advanced Compression also includes the capability to compress backed up data. Recovery Manager (RMAN) and Data Pump are the two most commonly used tools to backup the data stored inside an Oracle Database.

RMAN makes a block-by-block backup of the database data, also known as a "physical" backup, which can be used to perform database, tablespace or block level recovery. Data Pump is used to perform a "logical" backup by offloading data from one or more tables into a flat file. Oracle Advanced Compression includes the capability to compress the backup data generated by both of these tools.

Recovery Manager (RMAN) Compression

The continuous growth in enterprise databases creates an enormous challenge to database administrators. The storage requirements for maintaining database backups and the performance of the backup procedures are directly impacted by database size. Oracle Advanced Compression includes RMAN compression technology that can dramatically reduce the storage requirements for backup data.

Due to RMAN's tight integration with Oracle Database, backup data is compressed before it is written to disk or tape and doesn't need to be uncompressed before recovery – providing an enormous reduction in storage costs.

There are three levels of RMAN Compression: LOW, MEDIUM, and HIGH. The amount of storage savings increases from LOW to HIGH, while potentially consuming more CPU resources.

Data Pump Compression

The ability to compress the metadata associated with a Data Pump job was first provided in Oracle Database 10g Release 2. In Oracle Database 11g, this compression capability has been extended so that table data can be compressed on export.

Data Pump compression is an inline operation, so the reduced dump file size means a significant savings in disk space. Unlike operating system or file system compression utilities, Data Pump compression is fully inline on the import side as well, so there is no need to uncompress a dump file before importing it. The compressed dump file sets are automatically decompressed during import without any additional steps by the Database Administrator.

Full Data Pump functionality is available using a compressed file. Any command that is used on a regular file will also work on a compressed file. Users have the following options to determine which parts of a dump file set should be compressed:

- **ALL** enables compression for the entire export operation.
- **DATA-ONLY** results in all data being written to the dump file in compressed format.
- **METADATA-ONLY** results in all metadata being written to the dump file in compressed format. This is the default.
- **NONE** disables compression for the entire export operation.

The reduction in dump file size will vary based on data types and other factors. Note that when importing using Data Pump, the CREATE TABLE statements will have compression clauses that match the definition in the export file. If a compression clause is missing, then the table inherits the COMPRESSION attributes of the tablespace where the table is stored.

For more information about Oracle Data Pump, please visit http://www.oracle.com/technology/products/database/utilities/index.html

Compression for Network Traffic

Oracle Data Guard provides the management, monitoring, and automation software infrastructure to create, maintain, and monitor one or more standby databases to protect enterprise data from failures, disasters, errors, and data corruptions. Data Guard maintains synchronization of primary and standby databases using redo data (the information required to recover a transaction). As transactions occur in the primary database, redo data is generated and written to the local redo log files.

Data Guard Redo Transport Services are used to transfer this redo data to the standby site(s). With Advanced Compression, redo data may be transmitted in a compressed format to reduce network bandwidth consumption and in some cases reduce transmission time of redo data. As of Oracle Database 11g Release 2, redo data can be transmitted in a compressed format when the Oracle Data Guard configuration uses either synchronous redo transport (SYNC) or asynchronous redo transport (ASYNC).

For more information about Oracle Data Guard, please visit http://www.oracle.com/technology/deploy/availability/index.html

Out-of-the-Box Compression Capabilities

Oracle Database 11g Enterprise Edition includes a number of compression capabilities, out-of-the-box, that do not require a separate licenses, these include:

HCC on Oracle Storage

Oracle's Hybrid Columnar Compression (HCC) technology is a new (to Oracle Database 11g Release 2 on Exadata) method for organizing data within a database block. HCC utilizes a combination of both row and columnar methods for storing data. A logical construct called the compression unit is used to store a set of HCC-compressed rows. When data is loaded, groups of rows are stored in columnar format, with the values for a given column stored and compressed together. After the column data for a set of rows has been compressed, it is fit into the compression unit. Storing column data together, with the same data type and similar characteristics, drastically increases the storage savings achieved from compression.

Hybrid Columnar Compression has been extended to support Pillar Axiom and Sun ZFS Storage Appliance (ZFSSA) storage when used with Oracle Database Enterprise Edition 11.2.0.3

For more information about HCC, please see *Oracle Hybrid Columnar Compression Technical White Paper*[1]

Basic Table Compression

Oracle Database 9i introduced Basic Table Compression several years ago that compressed data that was loaded using bulk load operations. Basic Table Compression is a base feature of Oracle Database 11g Enterprise Edition (EE). Unlike Advanced Compression, Basic Compression does not apply compression to DML operations (INSERT/UPDATE) performed on the table after the initial bulk load. The formats on disk for Basic Compression and OLTP Table Compression are identical, so it is technically possible to convert from Basic to Advanced Compression simply by changing the storage definition on the table/partition.

RMAN Basic Compression

Oracle Recovery Manager (RMAN) includes a Basic Compression capability which enables RMAN to perform binary compression of backupsets.

Data Pump Metadata compression

The COMPRESSION parameter can be used to decrease the size of metadata written during Data Pump exports.

[1] http://www.oracle.com/technetwork/middleware/bi-foundation/ehcc-twp-131254.pdf

Index Compression

Index Key compression is a feature of Oracle Database that enables users to compress portions of the primary key column values in an index or index-organized table, which reduces the storage overhead of repeated values.

Key compression breaks the index key into a prefix entry (the grouping piece) and a suffix entry (the unique piece). Compression is achieved by sharing the prefix entries among the suffix entries in an index block. Only keys in the leaf blocks of a B-tree index are compressed. Key compression is done within an index block but not across multiple index blocks. Indexes can be compressed, independently of whether the underlying table is compressed.

Total Recall for Storing and Auditing Historical Data

Organizations need an efficient mechanism for long term data tracking and retention that doesn't involve application rewrites, 3rd party or handcrafted software solutions, or additional administrative overhead.

Oracle Total Recall is part of the Advanced Compression Option, and works with Oracle Database 11g, Enterprise Edition to help companies collect and store historical data in secure, tamper-proof tables while providing easy access to that information for existing applications. It provides an efficient, easy-to-use, and transparent solution for the long-term storage and auditing of historical data.

For more information about Oracle Total Recall, please see the *Oracle Total Recall* page on OTN[2]

Conclusion

The explosion in data volume being experienced by enterprises introduces significant challenges. Companies must quickly adapt to the changing business landscape without impacting the bottom line. IT managers need to efficiently manage their existing infrastructure to control costs, yet continue to deliver extraordinary application performance.

[2] http://www.oracle.com/technetwork/database/application-development/total-recall-1667156.html

The Advanced Compression Option of Oracle Database 11g provides a robust set of compression capabilities that enable IT managers to succeed in this complex environment. Using the Advanced Compression Option, enterprises can efficiently manage their increasing data requirements throughout all components of their data center – minimizing costs while continuing to achieve the highest levels of application performance.

Compression Syntax Examples

The examples below show the ease at which OLTP Table Compression and SecureFiles Compression/Deduplication can be enabled, for more detailed information please see the Oracle® Database SQL Language Reference.

OLTP Table Compression Syntax

```
CREATE TABLE emp (
      emp_id NUMBER
, first_name VARCHAR2(128)
, last_name VARCHAR2(128)
) COMPRESS FOR OLTP;
```

SecureFiles Deduplication Syntax

```
CREATE TABLE images (
image_id NUMBER,
image BLOB)
      LOB(image) STORE AS SECUREFILE
(TABLESPACE lob_tbs DEDUPLICATE);
```

SecureFiles Compression Syntax

```
CREATE TABLE images (
image_id NUMBER,
image BLOB)
      LOB(image)STORE AS SECUREFILE
(TABLESPACE lob_tbs COMPRESS);
```

ORACLE

Advanced Compression Option with Oracle
Database 11g
August 2012
Author: Gregg Christman
Contributing Authors: Kevin Jernigan

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

Hardware and Software, Engineered to Work Together