Implementing Information Lifecycle Management (ILM) with Oracle Database 12*c*

Table of Contents

Introduction

Exponential increases in data volumes are putting enterprise IT infrastructures under severe pressure – from a cost, performance, scalability and manageability perspective. It has become imperative to employ more efficient ways of storing and managing data to meet the growing demands being placed on IT systems. Dramatic increases in storage volumes are evident in all types of applications, and enterprise applications are no exception.

Although most organizations have long regarded their data as one of their most valuable corporate assets, only recently has the amount of data under management become a major issue. Originally, data was used to help achieve operational goals to run the business, but as technology capabilities have grown, ever-larger databases have become feasible for both operational - OLTP - and analytical - Data Warehouse - applications.

Regulatory requirements are also changing how and why data is being retained, as many organizations are now required to retain and control much more information for much longer periods. These requirements often extend beyond structured data - typically stored in relational databases such as Oracle Database – to semi-structured and unstructured data such as medical images, videos, photos, contracts, documents, etc. The result is an explosion in the amount of data that organizations are required to obtain, organize, manage, and store securely (and safely), while still providing easy, scalable, and high-performance access.

Consequently, organizations are trying to store fast growing quantities of data for the lowest possible cost while meeting increasingly stringent regulatory requirements for data retention and protection. Oracle Database 12*c* contains a rich feature set that can help implement an Information Lifecycle Management (ILM) solution and meet these new data storage demands, including Data Partitioning, Advanced Row Compression, Hybrid Columnar Compression, Automatic Data Optimization, Heat Map, Direct NFS Client, Clonedb, SecureFiles, In-database Archiving, SecureFiles, and Database File System (DBFS).

Information Lifecycle Management

Information Lifecycle Management (ILM) is the practice of applying policies for the effective management of information throughout its useful life. ILM for Oracle Database includes every phase of information from its beginning to its end, and consists of the policies, processes, practices and tools used to align the business value of information with the most appropriate and cost effective IT infrastructure -- from the time information is created or acquired through its final disposition.

Generally speaking, there are **FIVE STEPS** to implement an ILM strategy:

1. **Define the Data Classes:** For the primary databases that drive your business, identify the types of data in each database and where it is stored, and then determine:

   - Which data is important, where it is, and what must be retained

   - How this data flows within the organization

   - What happens to this data over time and when is it no longer actively needed

   - The degree of data availability, and protection, that is needed

   - Data retention for legal and business requirements

2. **Create Logical Storage Tiers:** For the data classes that represent the different types of storage tiers available in your environment.

3. **Define a Lifecycle**: A Lifecycle definition describes how data migrates across logical storage tiers during its lifetime. A lifecycle definition comprises one or more lifecycle stages that select a logical storage tier, data attributes such as compression and read-only, and a duration for data residing on that lifecycle stage. To summarize, a lifecycle defines WHERE to store data, HOW to store data and HOW LONG data should be retained.

4. **Assign a Lifecycle to Database Tables/Partitions**

5. **Define and Enforce Compliance Policies**

Automating ILM with Oracle Database 12*c*

When implementing an ILM strategy with Oracle Database 11*g*, organizations typically use the Advanced Compression and Data Partitioning options to manually create and deploy a compression and storage tiering solution – a solution which requires organizations to have sharp insight into data access and usage patterns across applications and tables/partitions.

Based upon this insight, DBAs, along with their storage counterparts, can manually compress and/or move data based upon their best estimations regarding actual data usage, ideally trying to ensure that the most frequently accessed data remains on the highest performance storage.

What has become clear to many organizations, after implementing manual storage tiering solutions, is that the ideal ILM solution is automated and is not reliant upon the organization's best guess of data access and usage patterns, but instead uses data usage information maintained by the database. The ideal automation solution would provide policy-based classification of data based on usage, greatly simplifying the ILM implementation process.

Heat Map and Automatic Data Optimization

Two features were added to Oracle Database 12c to help automate ILM: Heat Map, and Automatic Data Optimization (ADO). Both features are part of the Oracle Advanced Compression option.

Heat Map

Heat Map automatically tracks usage information at the row and segment levels.[1] Data modification times are tracked at the row level and aggregated to the block level, and modification times, full table scan times, and index lookup times are tracked at the segment level. Heat Map enables a detailed view of how data is accessed, and how access patterns change over time. Programmatic access to Heat Map data is available through a set of PL/SQL table functions, as well as through data dictionary views. In addition, Oracle Enterprise Manager provides graphical representations of Heat Map data.

Automatic Data Optimization

Automatic Data Optimization (ADO) allows organizations to create policies for data compression and data movement, and to implement automatic tiering of compression and storage. Oracle Database evaluates ADO policies during the DBA-defined database maintenance window, and uses the information collected by Heat Map to determine which operations to execute. All ADO operations are executed automatically and in the background, with no user intervention required.

ADO policies can be specified at the segment or row level for tables and table partitions. In addition to being evaluated and executed automatically in the background during the maintenance window, policies can also be evaluated and executed anytime by a DBA, manually or via a script. ADO policies specify *what* conditions (of data access) will initiate an ADO operation – such as no access, or no modification, or creation time – and *when* the policy will take effect – for example, after *n* days or months or years. Custom conditions can also be created by the DBA, allowing other factors to be used to determine when to move or compress data.

Compression Tiering

ADO supports both row-level compression tiering and segment-level compression tiering. Policies can be created to compress inactive rows (with Advanced Row Compression) at the block level, or to compress entire segments either with Advanced Row Compression or Hybrid Columnar Compression. This provides the flexibility to create compression tiering, keeping a mixture of uncompressed and compressed data in the same segment, along with compressing entire segments with Advanced Row Compression or with the Query (Warehouse) or Archive compression levels of Hybrid Columnar Compression (HCC). All of the compression actions implemented by ADO are fully online, and don't block any queries or OLTP transactions.

Storage Tiering

ADO supports storage tiering at the tablespace level in the database. As a tablespace reaches a *percent used value* any segments that have qualifying ADO policies will be moved to a target tablespace as specified by the policy; the oldest segments will be moved first. Once the space in the original tablespace reaches a *percent free value* no more segments are moved. This is a one-way movement and utilizes the online partition move feature of Oracle Database 12*c*, which results in no application downtime.

Implementing ILM with Oracle Database 12*c*

At the core of an Oracle Database ILM solution is the ability to define multiple data classes and tiers of storage, and assign different portions of data to different tiers based on the desired cost, performance and security for each portion. This is enabled through the use of Data Partitioning, Advanced Row Compression and Hybrid Columnar Compression, which are briefly described below.

Data Partitioning

At the most basic level, IT administrators can implement an Information Lifecycle Management (ILM) strategy by partitioning data based on the age of the data, and then moving historical partitions to low-cost storage, while keeping more active partitions on high performance storage.

Data Partitioning allows a table, index or index-organized table (IOT) to be subdivided into pieces. Each piece of a database object is called a partition. Each partition has its own name, and may optionally have its own storage characteristics. From the perspective of a database administrator, a partitioned object has multiple pieces, which can be managed either collectively or individually. This gives the

---

[1] Database rows are stored in database blocks, which are grouped in extents. A segment is a set of extents that contains all the data for a logical storage structure within a tablespace, i.e. a table or partition.

administrator considerable flexibility in managing the partitioned object. However, from the perspective of the application, a partitioned table is identical to a non-partitioned table; no modifications to application queries are necessary when accessing a partitioned table.

It is not unusual for partitioning to improve the performance of queries or maintenance operations by an order of magnitude. Moreover, partitioning can greatly reduce the total cost of data ownership, enabling a "tiered archiving" approach of keeping older but still relevant information online on lower cost storage devices.

Data Partitioning requires a separate license. For more information on partitioning, please see this Technical White Paper

Advanced Row Compression

Advanced Row Compression, a feature of the Advanced Compression option, uses a unique compression algorithm specifically designed to work with database tables in all types of applications. The algorithm works by eliminating duplicate values within a database block, even across multiple columns.

The compression ratio achieved with a given data set depends on the nature of the data being compressed. In general, organizations can expect to reduce their storage space consumption 2x to 4x by using Advanced Row Compression. That is, the amount of space consumed by compressed data will be two to four times smaller than that of the same data without compression.

Advanced Compression requires a separate license. For more information, please see this Technical White Paper: [2]

Hybrid Columnar Compression

Hybrid Columnar Compression (HCC) enables higher levels of data compression and provides enterprises with tremendous cost savings. Average compression ratios can range from 6x to 15x depending on which Hybrid Columnar Compression level is implemented – real world customer benchmarks have resulted in storage savings of up to 50x and more.

Oracle's Hybrid Columnar Compression technology utilizes a combination of both row and columnar methods for storing data. While HCC compressed data can be modified using conventional Data Manipulation Language (DML) operations, such as INSERT, UPDATE and DELETE – HCC is best suited for application with no, or very limited DML operations.

> **HCC Compression Levels**
>
> HCC *Warehouse compression* (also referred to as Query compression) has been optimized on Exadata storage to increase scan query performance by taking advantage of the smaller number of blocks on disk. HCC *Archive Compression* is optimized to maximize storage savings, typically achieving a compression ratio of 15:1 (15x).
>
> Hybrid Columnar Compression is available for use on Exadata, SuperCluster, Pillar Axiom, FS1, and Sun ZFS Storage Appliance (ZFSSA) storage hardware – no additional license is required to use HCC.

For more information about Hybrid Columnar Compression please see this Technical White Paper: [3]

Oracle Database 12*c* ILM Implementation Example

The remainder of this document will discuss how the features and options of Oracle Database 12*c* can be used to implement the **FIVE STEPS** of an ILM strategy that are defined earlier in this document. These steps will be reviewed as to how and where they fit into an Oracle Database ILM solution – as well as which Oracle Database features/options can be utilized.

Steps 1 to 3: Define Data Classes, Logical Storage Tiers and Information Lifecycle
Related Oracle Features/Options:
Data Partitioning, Advanced Compression and/or Hybrid Columnar Compression

---

[2] http://www.oracle.com/technetwork/database/focus-areas/storage/advanced-compression-whitepaper-130502.pdf
[3] http://www.oracle.com/technetwork/middleware/bi-foundation/ehcc-twp-131254.pdf

➢ **Define the Data Classes**

This step involves looking at all the data in your organization. This analysis requires organizations to understand which objects are associated with which applications, where those objects are located (on what class of storage), whether the objects have been compressed, and the granularity of the object (table vs. partition).

➢ **Create Logical Storage Tiers**

This step identifies and creates logical storage tiers, utilizing higher cost high performance storage and lower cost high capacity storage.

➢ **Define a Lifecycle**

The Lifecycle definition describes how data migrates across logical storage tiers during its lifetime. A lifecycle definition includes one or more lifecycle stages that select a logical storage tier, data attributes such as compression and/or read only, and a retention period for data residing on that lifecycle stage.

The lifecycle brings together the information/activities in **STEPS** 1 and 2 to allow DBA's to plan *WHERE* to store data (the logical storage tiers), *HOW* to store data (the data granularity and whether to compress the data) and *HOW LONG* data should be retained (which also helps determine how to compress the data).

Utilizing the planning from **STEP** 3, the diagram below (figure 1) shows how the most active data can be located on a high performance tier, and the less active / historical data on lower-cost tiers (and begins to associate compression levels to the various storage tiers).

Using Oracle Data Partitioning, the most active data partitions can be placed on faster, higher performance storage, while less active and historical data can be placed on lower cost storage. Data compression can also be applied as desired on a partition-by-partition basis. With this combination of features, the business is meeting all of its performance, reliability, and security requirements, but at a significantly lower cost than in a configuration where all data is located on one tier of storage.

In OLTP applications, organizations can use Advanced Row Compression (previously named OLTP Table Compression) for the most active tables/partitions, to ensure that newly added or updated data will be compressed as DML operations are performed against the active tables/partitions. For cold or historic data within the OLTP tables, organizations can use either Warehouse or Archive Hybrid Columnar Compression (assuming they are using storage that supports HCC).
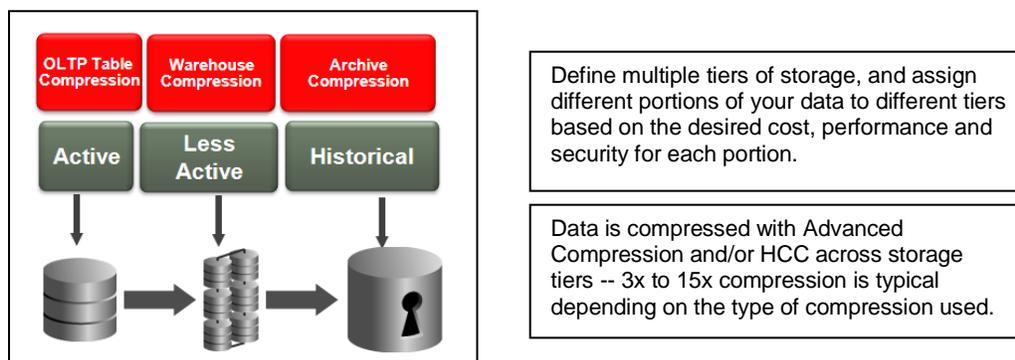

Figure 1. Compression and storage tiering.

Prior to Oracle Database 12*c*, organizations must implement both the storage tiering and compression tiering of their data manually, based upon their knowledge of the database. With Oracle Database 12*c,* storage tiering and compression can be automated, reducing the requirement for organizations to have deep insights into their data access/usage patterns. This information can now be provided by the database itself, using the Advanced Compression Heat Map capability, and ILM policies can be enforced automatically by the database using Automatic Data Optimization (ADO).

Steps 4 and 5: Assign a Lifecycle to Tables/Partitions and Define and Enforce Compliance Policies

Related Oracle Features/Options:

Data Partitioning, Advanced Row Compression and/or Hybrid Columnar Compression, Automatic Data Optimization and Heat Map

Implementing an automated compression and storage tiering solution using Automatic Data Optimization and Heat Map is straightforward, as the example below will show.

This example enables an information lifecycle policy for the ORDERS table with the goal to first compress the table for OLTP activity (ACTIVE tier), and then when the table becomes less active (few or no modifications but active queries) compress the table using HCC QUERY compression (LESS ACTIVE tier). When the data is no longer being actively queried but must remain available for historical/compliance usage, then compress the table even further using HCC ARCHIVE compression and move to lower-cost storage (HISTORICAL tier).

To implement these compression and storage tiering plans, the database administrator creates the following ADO policies:

> ALTER TABLE orders ILM ADD POLICY
> COLUMN STORE **COMPRESS FOR QUERY HIGH SEGMENT**
> **AFTER 30 DAYS OF NO MODIFICATION**;

> ALTER TABLE orders ILM ADD POLICY
> COLUMN STORE **COMPRESS FOR ARCHIVE HIGH SEGMENT**
> **AFTER 90 DAYS OF NO MODIFICATION**;

> ALTER TABLE orders ILM ADD POLICY **tier to low_cost_store;**

For this example we assume that the orders table is defined with Advanced Row Compression enabled, so that rows are compressed at that level when they are first inserted (the ACTIVE tier of storage). The organization could have also chosen to perform their initial data loads without compression enabled (to eliminate the minimal CPU overhead related to compressing the table during the loads) and instead deferred Advanced Row Compression to some period of time after the initial loads.

For example, the ADO policies below enable Advanced Row Compression thirty days after the initial load, once the initial post-load activity has subsided. In this case the tables' blocks would be individually compressed when all the rows in the block have a modification time more than thirty days in the past.

> ALTER TABLE orders ILM ADD POLICY
> ROW STORE **COMPRESS ADVANCED ROW**
> **AFTER 30 DAYS OF NO MODIFICATION;**

Or, if the organization wanted to compress the data sooner, the next example creates a row-level ADO policy to automatically compress blocks in the table after no row in a given block has been modified for at least 3 days.

> ALTER TABLE orders ILM ADD POLICY
> ROW STORE **COMPRESS ADVANCED ROW**
> **AFTER 3 DAYS OF NO MODIFICATION**;

The next policy will be automatically enforced by the database at the segment level when Heat Map determines there have been no data modifications for 30 days. The policy is intended to change the compression level of the table (or partition) to a higher level of compression when the data is no longer being actively modified (LESS ACTIVE tier).

Changing the compression from Advanced Row Compression to Hybrid Columnar Compression (QUERY HIGH level) will occur during the first maintenance window when the specific ADO policy criteria has been met.

```
ALTER TABLE orders ILM ADD POLICY
COLUMN STORE COMPRESS FOR QUERY HIGH SEGMENT
AFTER 30 DAYS OF NO MODIFICATION;
```

In this example, if the orders table is partitioned, then the ADO policy will automatically be applied to each partition individually. Using Data Partitioning, organizations can configure multiple types of storage, creating a high performance tier with faster more expensive storage, and a high capacity tier with slower less expensive storage. Using the information gathered by Heat Map, Oracle Database will automatically evaluate ADO policies to determine when each table/partition is eligible to be moved to a lower cost storage tier.

In the next example, we assume that the orders table has been compressed earlier by ADO to the HCC QUERY HIGH level after 30 days of no modifications to the table (this could have easily also been a partition). QUERY HIGH compression typically provides a compression ratio of 6x to 10x and is intended for data that is actively queried, but not modified (or lightly modified). When this data further "cools down" additional storage and performance gains can be realized when ADO automatically moves the data, which is still needed for query purposes, but is now no longer being actively modified/queried, to HCC ARCHIVE HIGH level compression (HISTORICAL/COMPLAINCE tier).

After 90 days of no modification being made to the data, the final ADO policies will be applied.

```
ALTER TABLE orders ILM ADD POLICY
COLUMN STORE COMPRESS FOR ARCHIVE HIGH SEGMENT
AFTER 90 DAYS OF NO MODIFICATION;


ALTER TABLE orders ILM ADD POLICY tier to lessactivetbs;
```

Continuing with the final stages of the organization's planned lifecycle management, with the final ADO policy criteria being satisfied, the data will now be compressed to the HCC ARCHIVE HIGH level and will be moved to lower cost storage. This allows active data to remain on higher performance tiers and also allows the historical data, which remains online, to still be accessed by applications as needed and ensures a smaller footprint for the historical data – ARCHIVE HIGH typically provides for the highest level of HCC compression, in the range of 10x to 15x or more.

In this simple ILM example Oracle Database automatically evaluated the ADO policies to determine when data was eligible to be moved to a higher compression level, and when data was eligible to be moved to a different tablespace. This ensures data accessibility and performance, while reducing the storage footprint even further – with no additional burden placed on database administrators or storage management staff.

Note that Hybrid Columnar Compression is only available with Oracle Database on Exadata or with specific Oracle Storage.

Additional Features for Implementing ILM with Oracle Database

Oracle Database 12*c* contains a rich set of capabilities and options that can be used to enhance and optimize an Information Lifecycle Management (ILM) solution, including:

> **Direct NFS Client (dNFS)**

Standard NFS client software, provided by the operating system, is not optimized for Oracle Database file I/O access patterns. With Oracle Database 11*g* and later releases, organizations can configure Oracle Database to access NFS V3 (or V4 with Oracle Database 12*c*) NAS devices directly using Oracle Direct NFS Client, rather than using the operating system kernel NFS client. Oracle Database will access files stored on the NFS server directly through the integrated Direct NFS Client, eliminating the overhead imposed by the operating system kernel NFS. These files are also accessible via the operating system kernel NFS client thereby allowing seamless administration. (See here for more information)

> **CloneDB**

CloneDB allows the creation of databases using "copy on write" technology to quickly create database clones. CloneDB requires that each clone database use the Direct NFS Client and that a backup of the source database be available on Direct NFS Client-mounted

storage. There must also be enough storage for that backup and for each database block that changes in each of the clone database(s). The time to create the clone is very quick because the source database's data is not copied to create the clone – only file header information is copied – and when creating multiple clones from a single source database the total storage required can be significantly less than with traditional full size database copies. This allows the organization to easily keep development, test and QA databases available and in sync quickly while using significantly less storage.

> **SecureFiles**

SecureFiles is designed to deliver high performance for files stored in Oracle Database, comparable to that of traditional file systems, while retaining the advantages of Oracle Database. SecureFiles was first introduced in Oracle Database 11*g*, and enables a major paradigm shift for storing and managing files. Traditionally, relational data is stored in a database while unstructured data is stored as files in file systems; with SecureFiles, you can store relational and file data together in Oracle Database and deliver high performance while also implementing a unified security model, a unified backup and recovery infrastructure, and enabling all the other features of Oracle Database for both structured and unstructured data. See also Database File System below.

> **Database File System (DBFS)**

Database File System (DBFS) implements a standard file system interface for files stored in Oracle Database. With this interface, storing files in the database is no longer limited to programs specifically written to use BLOB and CLOB programmatic interfaces - files in the database can now be transparently accessed using any operating system (OS) program that acts on files. DBFS makes it easy to store all your files in your database.

> **In-Database Archiving**

In-Database Archiving allows applications to archive rows within tables by marking them as inactive. This feature can meet compliance requirements for data retention while hiding archived data from current application usage. Archived rows can be displayed using SQL statements that specifically make them visible, and the rows can be re-activated if needed because they still reside in the original table. These archived rows can also be compressed to reduce their storage usage and can be incorporated into an ILM strategy at the segment level.

Conclusion

Information Lifecycle Management (ILM) enables organizations to understand how their data is accessed over time, and manage the data compression and storage tiering accordingly. However, most ILM solutions for databases lack two key capabilities – automatic classification of data and automatic data compression and movement across storage tiers.

The Heat Map and Automatic Data Optimization features of Advanced Compression with Oracle Database 12*c* provide comprehensive and automated ILM capabilities that minimize costs while maximizing performance. In combination with its comprehensive compression features and options, Oracle Database 12*c* provides an ideal platform for implementing Information Lifecycle Management for all of your database data.

ORACLE®