ORACLE

Overview of Oracle SecureFiles

July 2025, Version 23ai

Public

Purpose Statement

This document provides an overview of features and enhancements included in release 23ai. It is intended solely to help you assess the business benefits of upgrading to 23ai and planning for the implementation and upgrade of the product features described.

Disclaimer

This document in any form, software, or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. 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, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Table of contents

Introduction

The nature of critical business information is changing rapidly and is no longer limited to structured text or numeric data. Unstructured content – images, audio, JSON (JavaScript Object Notation), video, PDF, Word documents etc. – is becoming pervasive. Web-based applications are forcing organizations to capture and manage more unstructured data such as photos, videos, and news clips more than ever before. Regulatory changes like Sarbanes-Oxley and the Health Insurance Portability and Accountability Act of 1996 (HIPAA) are further fueling this trend.

Mainstream web and business applications must integrate both relational or structured and unstructured content to provide users with a seamless and richer user experience. Traditionally, unstructured content has been stored in file systems due to their simplicity and performance. This choice forces organizations to compromise Oracle's security, scalability, transactional semantics, read consistency and high availability for file or unstructured data. However, this tradeoff comes at a cost, often prohibitive.

Different storage mechanisms for structured and unstructured data introduces disjoint security/audit models, difficulty in searching across all related enterprise content and disparate IT management procedures for backup and recovery – which can ultimately lead to lower ROI. SecureFiles is a feature of Oracle Database specifically engineered to deliver high performance for file or unstructured data comparable to that of traditional file systems while retaining the advantages of the Oracle Database.

SecureFiles removes the need for compromise by offering the 'best-of-both-worlds' architecture for storing unstructured content. SecureFiles is designed as a superset of the ANSI standard for large objects (LOBs) and offers easy migration from old LOBs or BasicFiles. Applications can transparently take advantage of industry-standard encryption for added security and intelligent storage features like compression and deduplication, for increased space savings and faster performance.

With SecureFiles, organizations can now manage all relational data, and associated file data in Oracle, using a single security/audit model, a unified backup and recovery process and perform seamless search across all information. Also, using SecureFiles as the backbone storage engine, Oracle's Database File System (DBFS) enables a user space file system to utilize large objects (SecureFiles lobs) -- providing database level data integrity and database level security to any files in the file system.

SecureFiles Overview

SecureFiles represents the core infrastructure in the Oracle Database for unstructured content management. SecureFiles typically delivers substantially improved performance, along with optimized storage, for unstructured data inside the Oracle Database. LOBs from Oracle Database 10g, and prior releases, are still supported using 'BasicFiles'.

SecureFiles includes numerous architectural enhancements for greatly improved performance, scalability, efficient storage, and easier manageability, including the following:

No LOB Index Contention

Older LOBs use a global per-segment B+ tree index to maintain the inodes, which are data structures that map the logical offsets to the physical disk blocks. The global B+ tree structure is used for both access and navigation into the LOBs, and for free space management in LOB segments. This causes contention and significant performance degradation in highly concurrent environments.

Unlike older LOBs, SecureFiles do not use the LOB index to manage such meta-data. Instead, "private" meta-data blocks are used that are co-located with the data blocks in the LOB segment.

This design removes the LOB index contention that existed with BasicFiles LOBs and greatly improves performance, especially under real world insert-delete-reclaim situations.

Write-Gather Cache (WGC)

SecureFiles WGC buffers data typically up to 4MB during write operations before flushing or committing to the underlying storage layer. This buffering of in-flight data allows for large contiguous space allocation and large disk I/O. Write performance is greatly improved due to reduced disk seek costs.

The WGC is maintained on a per-transaction basis. Oracle automatically determines if the WGC needs to be flushed to disk before the 4MB limit is reached. A 'commit' by the user also causes flushing of the WGC to disk.

The WGC size is 4MB and is not a user tunable parameter.

Space Management

SecureFiles segments require tablespaces managed with automatic segment space management (ASSM). Space management is optimized by using separate pools for Committed Free Space (CFS) and Uncommitted Free Space (UFS) to shield new allocation from the overhead of in-flight transactions.

An intelligent space manager provides fast allocation of large, contiguous disk blocks and efficient deletion by automatically reclaiming freed space. Space is also pre-allocated based on heuristics of recent demand, and this ensures no stalling and consequently no impact on foreground performance. Space management uses statistics for efficient space pre-allocation.

Reclamation of space is done using a combination of foreground and background. If a foreground cannot find free space in the Committed Free Space pool, it will start searching for reclaimable space in the Uncommitted Free Space pool. However, it only does the minimal work (reclaim the space just enough for allocation). Then the foreground pokes background to perform work that is more intensive. In addition, the background can do the work as part of space preallocation.

SecureFiles typically encounters less High-Water Mark (HWM) contention than BasicFiles.

Reduced Fragmentation

SecureFiles, unlike BasicFiles or older LOBs, uses a dynamic CHUNK (one or more contiguous Oracle blocks) size setting to maximize contiguous allocations on disk and reduce fragmentation.

The user-specified CHUNK value is used as a suggestion, along with other factors such as size of the SecureFiles and the availability of space in the segment, to determine the optimal CHUNK size. SecureFiles CHUNK is completely an internal concept.

A fixed CHUNK size setting causes internal fragmentation if it is too large, and poor write performance if it is too small. With a dynamic CHUNK size setting, large regions of disk can be allocated and deallocated in a single operation. SecureFiles is thus designed from the ground up to be a high performance and storage efficient solution for unstructured or file data of all sizes.

Intelligent Pre-Fetching

Read performance is enhanced by read-ahead or pre-fetching data from disk while data is sent over the network.

SecureFiles keeps track of access patterns and intelligently pre-fetches data before the request is made. Read latency is reduced by the overlap of the network roundtrip with the disk I/O for the pre-fetched data and throughput is greatly increased.

Network Layer

SecureFiles client/server network layer allows for high-speed data transfer between the client and server. This allows for reading and writing bulk data directly from the network socket without the need for any temporary staging. This provides for significantly higher read/write performance.

Easier Manageability

SecureFiles features self-tuning and intelligent space and memory management algorithms and consequently require lesser number of user-tuned parameters. Specifically, FREEPOOLS, FREELISTS, FREELIST GROUPS & PCTVERSION no longer need to be specified and are ignored for SecureFiles.

Not only are some of these parameters difficult to tune for unpredictable spikes in workloads, but also cannot be changed online.

SecureFiles maintains internal statistics to self-tune the space management algorithms and ensures high performance and scalability under diverse workloads.

Oracle Database File System (DBFS) Integration

DBFS creates a standard file system interface on top of files and directories that are stored in database tables. DBFS is like NFS in that it provides a shared network file system that looks like a local file system. Like NFS, there is a server component and a client component. In DBFS, the server is the Oracle Database. Files are stored as SecureFiles LOBs in a database table.

A set of PL/SQL procedures implement the file system access primitives such as create, open, read, write, and list directory. The implementation of the file system in the database is called the DBFS Content Store. The DBFS Content Store allows each database user to create one or more file systems that can be mounted by clients. Each file system has its own dedicated tables that hold the file system content.

SecureFiles Deduplication, Compression and Encryption

ENCRYPT, COMPRESS and DEDUPLICATE are available only for SecureFiles and do not apply to BasicFiles.

These features in SecureFiles can be enabled independently, or as a combination of one or more features. If all three features are enabled, Oracle will perform deduplication first and then compression followed by encryption (when tablespace-level encryption is enabled).

Deduplication

This feature eliminates multiple, redundant copies of SecureFiles data and is completely transparent to applications. Oracle automatically detects multiple, identical SecureFiles data and stores only one copy, thereby saving storage space. Deduplication not only simplifies storage management but also results in significantly better performance, especially for copy operations.

Deduplication is done on per LOB segment basis; duplicates are not detected across multiple LOB segments. The `lob_storage_clause` allows for specifying deduplication at a partition level. However, duplicate detection does not span across partitions or subpartitions for partitioned SecureFiles columns.

The DEDUPLICATE keyword is used to enable Advanced LOB Deduplication checking for SecureFiles. Oracle uses a secure hash index to detect duplicate SecureFiles data and stores a single copy for all identical content. Deduplication is transparent to applications, reduces storage and simplifies storage management.

Advanced LOB Deduplication (previously named SecureFiles LOB Deduplication) is a feature of Oracle Advanced Compression.

Example:

Create a table with a SECUREFILE LOB column and LOB deduplication enabled on only one partition. Only LOBs that belong to partition 'p1' are deduplicated.

```
CREATE TABLE t1 ( REGION VARCHAR2(20), mydata BLOB)
         LOB(mydata) STORE AS SECUREFILE )
PARTITION BY LIST (REGION) (
         PARTITION p1 VALUES ('x', 'y')
         LOB(mydata) STORE AS SECUREFILE (
             DEDUPLICATE),
         PARTITION p2 VALUES (DEFAULT));
```

Compression

Oracle detects if SecureFiles data is compressible and will compress using industry standard compression algorithms. If the compression does not yield any savings or if the data is already compressed, SecureFiles will turn off compression for such LOBs. Compression not only results in significant savings in storage but typically also improved query performance by reducing I/O, buffer cache requirements, redo generation and encryption overhead.

Random access reads and writes to compressed SecureFiles are achieved without the need to decompress the entire file. Only the sub portion (compression unit) of the compressed file, corresponding to the logical offset being read or written, needs to be decompressed thus saving CPU and I/O.

There are three levels of Advanced LOB Compression: LOW, MEDIUM, and HIGH. By default, Advanced LOB Compression uses the MEDIUM level, which typically provides good compression with a modest CPU overhead of up to 3-5%. Advanced LOB Compression LOW is optimized for high performance. Advanced LOB Compression LOW maintains up to 80% of the compression achieved through MEDIUM, while utilizing up to 3x less CPU. Finally, Advanced LOB Compression HIGH achieves the highest storage savings but incurs the most CPU overhead.

The COMPRESS keyword is used to enable SecureFiles Advanced LOB Compression. SecureFiles compression is orthogonal to table or index compression. Setting table or index compression does not affect SecureFiles compression or vice versa. For partitioned tables, the `lob_storage_clause` is used for specifying compression at a partition level.

Advanced LOB Compression (previously named SecureFiles LOB Compression) is a feature of Oracle Advanced Compression.

Example:

Create a table with a SECUREFILE LOB column and LOB compression enabled on only one partition. Only LOBs that belong to partition p1 are compressed.

```
CREATE TABLE t1 ( REGION VARCHAR2(20), mydata BLOB)
         LOB(mydata) STORE AS SECUREFILE )
    PARTITION BY LIST (REGION) (
         PARTITION p1 VALUES ('x', 'y')
         LOB(mydata) STORE AS SECUREFILE (
               COMPRESS MEDIUM ),
         PARTITION p2 VALUES (DEFAULT));
```

Encryption

SecureFiles supports the use of Transparent Data Encryption (TDE) syntax. The database supports automatic key management for all SecureFiles columns within a table and transparently encrypts/decrypts data and backups. Applications require no changes and can take advantage of SecureFiles using TDE semantics.

The ENCRYPT and DECRYPT keywords are used to turn on or turn off SecureFiles encryption and optionally select the encryption algorithm to be used. Encryption is performed at the block level and all SecureFiles in the specified column are encrypted.

Example:

Create a table with SECUREFILE LOB column and LOB encryption enabled on only one partition. Only LOBs that belong to partition p1 are encrypted.

```
CREATE TABLE t1 ( REGION VARCHAR2(20), mydata BLOB)
      LOB(mydata) STORE AS SECUREFILE (
          CACHE
          ENCRYPT USING 'AES128' )
      PARTITION BY LIST (REGION) (
         PARTITION p1 VALUES ('x', 'y')
         PARTITION p2 VALUES (DEFAULT));
```

Transparent Data Encryption (TDE) is a feature of Oracle Advanced Security.

SecureFiles Shrink

SecureFiles is the default storage mechanism for LOBs with Oracle Database, Oracle strongly recommends SecureFiles for storing and managing LOBs.

The Oracle Database SecureFiles Shrink feature provides manual and automatic methods to free the unused space in SecureFiles LOB segments and release the space back to the containing tablespace.

This document provides an overview of both the Manual, and Automatic SecureFiles Shrink features.

Manual SecureFiles Shrink

A user can use an `ALTER TABLE ... SHRINK SPACE` statement to manually shrink a SecureFiles LOB segment. The user can also use tools, such as Segment Advisor or a PL/SQL procedure, such as `DBMS_SPACE.SPACE_USAGE` to return information about SecureFiles space usage before deciding on the SecureFiles LOB segments to shrink.

The following points are important when opting for the manual shrink method:

- The manual SecureFiles shrink operation is an online DDL with part of the operations being offline, where offline means concurrent DML are blocked until the shrink activity on the critical section ends. Concurrent DML statements do not fail with ORA-54 but are blocked.
- The manual SecureFiles shrink operation disregards any flavor of undo retention and treats it as if the retention is equal to none. The user cannot expect the LOB retention feature to provide the usual guarantees after invoking the shrink operation. The user may see the ORA-1555 snapshot too old error message in queries. Run the shrink operation with caution if this is a concern.

Users can apply the *`shrink_clause`* on SecureFiles LOB segments for Oracle Database release 21c and onward. There are two ways to invoke the *`shrink_clause`*:

This DDL targets a specific LOB column and all its partitions.

```
ALTER TABLE <table_name> MODIFY LOB <lob_column> SHRINK SPACE;
```

This DDL cascades the shrink operation for all the LOB columns and its partitions for the given table .

```
ALTER TABLE <table_name> SHRINK SPACE CASCADE;
```

Automatic SecureFiles Shrink

SecureFiles LOB segments can potentially become the largest consumer of space in a database. It has been the responsibility of users to identify the SecureFiles LOB segments with the largest amount of free space and invoke `ALTER TABLE ... SHRINK SPACE` or `ALTER TABLE ... MOVE` manually.

With Automatic SecureFiles Shrink, users can now expect the database to automatically identify candidate SecureFiles LOB segments based on a set of criteria and invoke `ALTER TABLE ... SHRINK SPACE` in the background.

Automatic SecureFiles Shrink is designed to minimize the functional and performance impact on concurrent workloads. While shrink is running automatically on a SecureFiles LOB segment, all DML and DDL statements involving the segment will continue to succeed. Space will be gradually freed up in the SecureFiles LOB segment and performance impact will be minimal.

Automatic SecureFiles Shrink does not have any effect on the BasicFiles LOBs and in-lined SecureFiles LOBs.

Automatic SecureFiles Shrink is not enabled by default. In on-prem environments, use the following command to enable the feature.

```
exec DBMS_SPACE.SECUREFILE_SHRINK_ENABLE();
```

To disable Automatic SecureFiles Shrink, use the following command to disable the feature.

```
exec DBMS_SPACE.SECUREFILE_SHRINK_DISABLE();
```

In Autonomous Cloud, users should contact their system administrator for enabling Automatic SecureFiles Shrink for their PDBs.

LOB Segment Selection Criteria for Automatic SecureFiles Shrink

The Automatic SecureFiles Shrink task excludes the following SecureFiles LOB segments when choosing the SecureFiles LOB segments to shrink:

- The SecureFiles LOB segment is not an idle segment as per SecureFiles LOB segment Idle Time Limit
- The SecureFiles LOB segment does not contain extra free space greater than the pre-allocation threshold
- The SecureFiles LOB segment has RETENTION MAX, which means the segment keeps as many unexpired blocks as possible
- The SecureFiles LOB segment is currently being shrunk
- The SecureFiles LOB segment does not have enough expired free space that is no longer needed for lob retention requirement. Space that is still needed for lob retention is treated as used space
- The SecureFiles LOB segment has failed a previous shrink task. Previous shrink calls have failed to free space from the SecureFiles LOB segment. Automatic SecureFiles

Shrink identifies the SecureFiles LOB segments that it failed to shrink previously and avoids such segments

SecureFiles shrink supports existing advanced capabilities including Compression, Encryption, and Deduplication.

Migration to SecureFiles

In previous versions it was challenging to decide which BasicFiles LOBs to migrate to SecureFiles LOBs, and whether or not to compress the LOBs, especially considering that organizations often have multiple databases, with a large number of schemas, tables, and segments.

You can now use the SecureFiles Migration Utility to simplify the migration, and compression, of BasicFiles LOB segments to SecureFiles LOB segments. The SecureFiles Migration Utility automates several steps that were earlier performed manually. It also generates reports that help you decide which BasicFiles LOBs you want to migrate and compress.

Migration Utility Advantages

- No need not take the table or partition offline
- Perform the migration at the database, schema, table, or LOB segment level
- After migrating the data, you can also use the SecureFiles Migration Utility to compress the SecureFiles LOBs. (Advanced LOB Compression is a feature of Oracle Advanced Compression)

Migration Utility Disadvantages

- Additional storage, equal to the entire table or partition, required and all LOB segments must be available
- Global indexes must be rebuilt

The SecureFiles Migration Utility is the recommended method to migrate BasicFiles LOBs to SecureFiles. This utility encapsulates all the functionality offered by Online Redefinition and saves you the time and effort involved in manually running a series of API calls.

Interfaces

SecureFiles supports standard client interfaces including SQL, PL/SQL, OCI, ODBC and JDBC.

Get Started with Compression Advisor

Use the free Compression Advisor (DBMS_COMPRESSION) to estimate the storage space that you can save by enabling the compression feature for existing SecureFiles LOBs. It analyzes the compression ratio of a table, or an index, and provides information about compressibility of the object. You can provide various parameters to selectively analyze different compression types.

In Oracle Database 23ai, the Compression Advisor for LOBs procedure has been enhanced to estimate the compression ratio faster for both inline and out-of-line LOBs while using less space. Now you can also estimate the compression ratio for BasicFiles LOBs. This helps you decide upfront whether you want to compress the resultant SecureFiles LOB, before migrating BasicFiles LOBs to SecureFiles LOBs.

You can estimate the compression ratio at the LOB byte level and the time taken, in hours, to compress the LOB data in the table.

Estimate Savings with Advanced LOB Deduplication

Advanced LOB Deduplication enables Oracle Database to automatically detect duplicate LOB data, within a LOB column or partition, and conserve space by storing only one copy of the data. Note that Advanced LOB Deduplication is a feature of Advanced Compression.

You can estimate the space, that you can save, before enabling Advanced LOB Deduplication. This allows you to make an informed decision whether or not to enable LOB deduplication as well as decide whether you want to deduplicate the resultant SecureFiles LOB, before migrating BasicFiles LOBs to SecureFiles LOBs.

The GET_LOB_DEDUPLICATION_RATIO function estimates the storage space that you can save by enabling the deduplication feature, for existing SecureFiles LOBs, and returns the deduplication ratio.

Usage Note:

- The deduplication ratio is an approximate value, which is calculated based on the sampled rows in the LOB column. The actual space that you save when you enable deduplication for the complete table may be different

More Information

- See the *SecureFiles and Large Object's Developers Guide* for more information, and usage examples, for the SecureFiles Migration Utility
- See the *PL/SQL Packages and Types Reference* for more information about the DBMS_SECUREFILES.GET_LOB_DEDUPLICATION_RATIO function and Compression Advisor (DBMS_COMPRESSION)