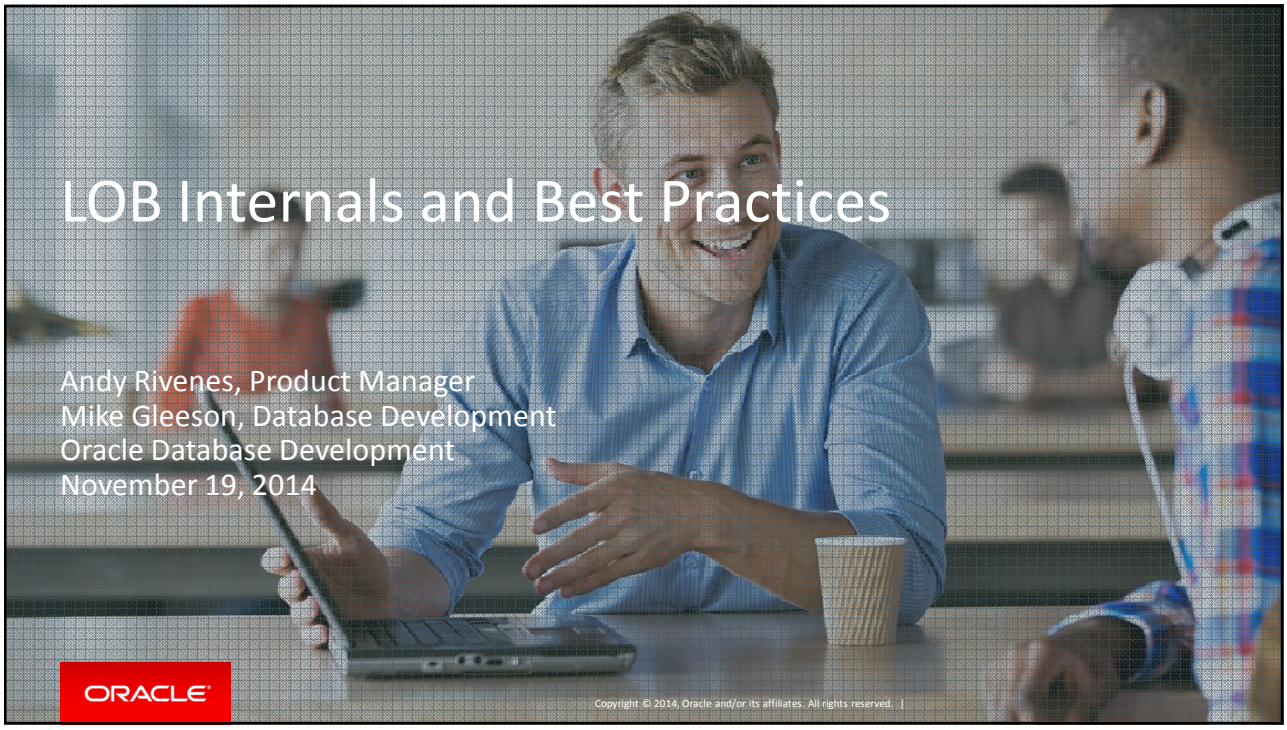
ORACLE


LOB Internals and Best Practices
Andy Rivenes, Product Manager
Mike Gleeson, Database Development
Oracle Database Development
November 19, 2014
ORACLE
Copyright © 2014, Oracle and/or its affiliates. All rights reserved. |

Safe Harbor Statement

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.

 3

Program Agenda

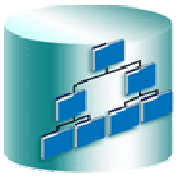
1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

 4

Oracle SecureFiles

Consolidated Secure Management of Data

- SecureFiles gives file system performance for files in the database
- Introduced with Oracle Database 11g Release 1
- Similar to BasicFile LOBs but much faster, and with more capabilities
  - Transparent encryption (with Advanced Security Option)
  - Compression, deduplication (with Advanced Compression Option)
  - Extends the security, reliability, and scalability of database to files
  - Superset of LOB interfaces allows easy migration from LOBs
- Enables consolidation of file data with associated relational data
  - Single security model
  - Single view of data
  - Single management of data



ORACLE

SecureFiles Innovations

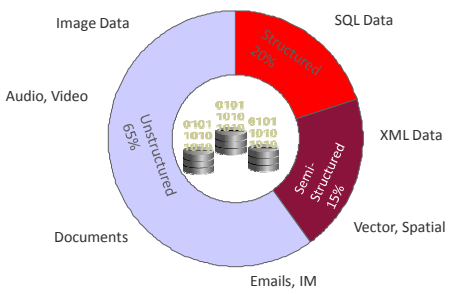
- Write Gather Cache
  - Cache above the storage layer buffers data up to 4MB during writes before flushing to disk
  - Allows for large contiguous space allocation for LOB data and reduced write latency
- Intelligent Pre-fetching
  - Improves read performance by pre-fetching LOB data from disk
  - Overlaps disk IO with network latency to improve throughput
- New Space Management routine
  - Automates new space allocation and “freed” space reclamation
  - Optimized chunk size reduces fragmentation
- No more High Water Mark contention as with old LOBs
  - Deletion and reuse of entire LOBs, not just individual chunks

ORACLE

Why are SecureFiles "Secure"?

- It's not "security", but safety
- File data is safest inside a database
  - Best protection from unauthorized and partial modification
  - Best and most granular access control
  - Best protection from hardware failure, disaster and human error
- Oracle Database is more fault tolerant and scalable than a file system
- Achieve all these benefits without any performance compromise

Enterprise Data



Yearly Data Growth

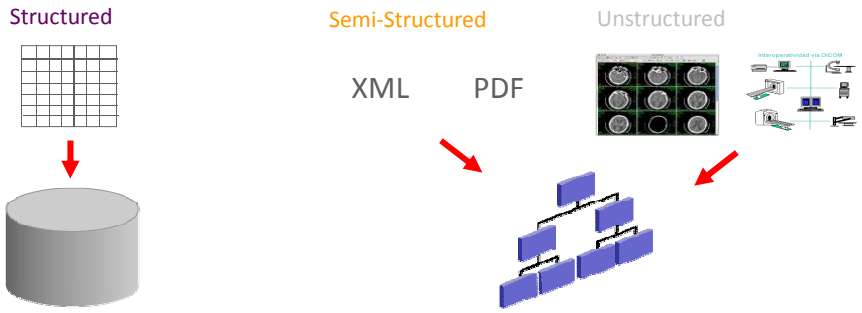
Data	Growth
Structured	15 - 20%
Semi & Unstructured	50 - 100%

* Gartner & IDC Estimates

- In a typical enterprise, Structured Data is ~20%
- Semi & Unstructured Data represents the other 80%
- Data growth is happening across the board!

Managing Information

- Organizations need to efficiently and securely manage all data



- Simplicity and performance of file systems makes it attractive to store file data in file systems, while keeping relational data in DB
- Mainstream DBs support ANSI-standard LOBs for storing file data inside DB – performance is a concern for many users

Files Belong with Relational Data

- Many Enterprise applications manipulate both files and relational data
  - Rich user experience, compliance, business integration
- This split compromises security, robustness, and management
  - Disjoint security and auditing models
  - Changes cannot be made atomically
  - Backup and recovery are fragmented
  - Search across relational data and files is difficult
  - Space management is complicated
  - Separate interfaces and protocols
  - Application architecture more complex

Two data managers for one application is one too many!

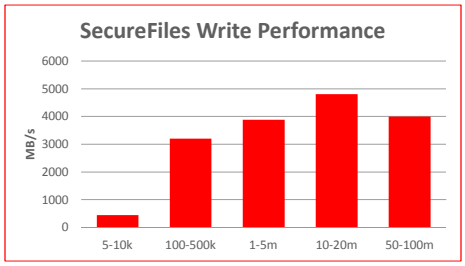
Program Agenda

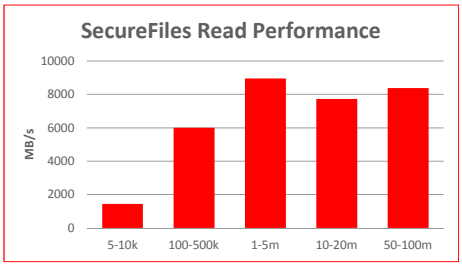
1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

ORACLE

SecureFiles Performance & Scalability on Exadata

- Small Documents: Extremely High Throughput
  - Load at 200 Million docs/hr, Read at 780 Million docs/hr
- Large Multimedia: Extremely High Bandwidth
  - Load at 4 GB/s and Read at 8 GB/s



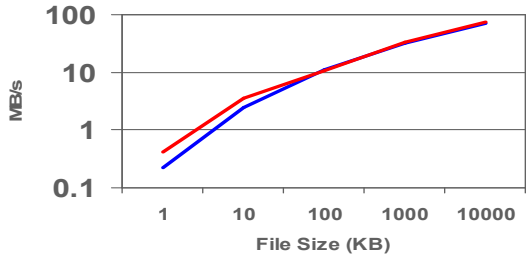


ORACLE

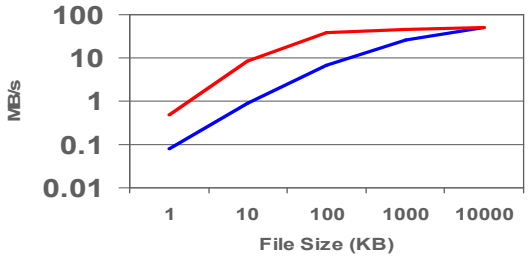
High Performance

SecureFiles Vs. NFS

File Read Performance



File Write Performance



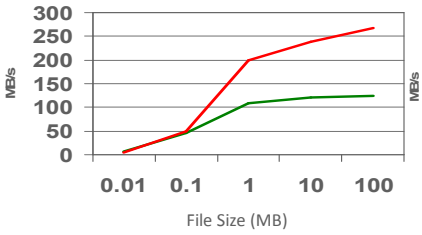
SecureFiles NFS

- SQLPlus file test, single stream, single host
- Using SecureFiles is faster across the board
  - 2x-3x faster for Queries, 6x for Inserts
  - Tests run using both SecureFiles and NFS/ext3 in metadata journaling only (default for NFS)
- Filesystem-like performance

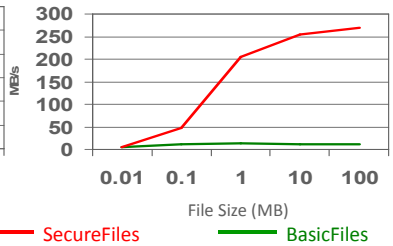
High Performance

SecureFiles Vs. BasicFiles

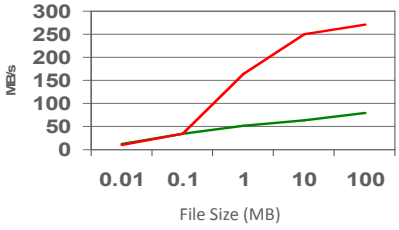
Writes: New Space



Writes: Reused Space



Read Performance



SecureFiles BasicFiles

SQLPlus file test: Concurrent Reads/Writes, OCI, 4 streams

- Adding files using new disk space: ***Up to 2x faster***
- Adding files reusing space: ***Up to 22x faster***
- Reads up to ***3x faster***

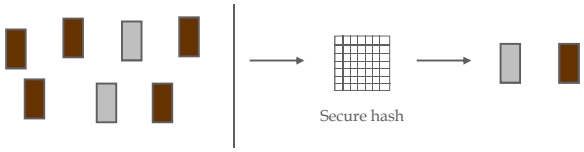
Program Agenda

1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

Advanced Features - Compression

- SecureFiles supports compression
- Huge storage savings
  - Industry standard compression algorithms
  - 2-3x compression for typical files (doc, pdf, xml)
- Auto-detects if SecureFiles data is compressible
  - Skips compression for already compressed data
  - Skips compression when space savings are minimal or zero
- Server-side compression
  - Allows for random reads and writes to SecureFiles data
- Three levels of compression
  - Compression Levels: LOW, MEDIUM (default), HIGH
  - More latency and CPU overhead for higher compression levels
- Part of the Advanced Compression Option

Advanced Features - Deduplication



- Enables storage of a single physical image for duplicate data
- Significantly reduces space consumption
- Dramatically improves writes and copy operations
- No adverse impact on read operations
  - May actually improve read performance for cache data
- Especially useful for content management, email applications and data archival applications
- Part of the Advanced Compression Option

Advanced Features - Encryption

- Extends Transparent Data Encryption (TDE) functionality to SecureFiles data
  - Data encrypted on disk
  - Key management completely transparent to applications
- Support for industry-standard encryption algorithms
  - 3DES168
  - AES192 (default)
  - AES256
- Unified security level for both file and relational data
- Part of the Advanced Security Option

SecureFiles Interfaces

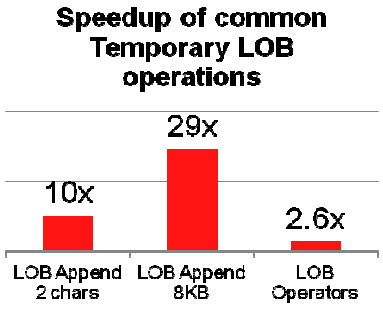
- SecureFiles is 100% backwards compatible with ANSI SQL 92 LOB interfaces
- Database clients use standard LOB interfaces
  - JDBC, ODBC, OCI, OCCI, .NET, PL/SQL and SQL

ORACLE

Oracle Database 12c: In-Memory LOB Queries & Updates

Speeds up string ops on LOBs & updates of Temporary LOBs

- In-Memory optimization to trade PGA memory for speed
- Uses in-memory working area for Temporary LOBs that are small
  - Automatically and transparently spills temporary LOB to disk / Flash Cache as LOB grows beyond a threshold
- Speeds up all LOB string operations
  - concatenate, append, substr, length, instr, compare, trim, like, replace, pad, nvl using SQL functions or DBMS_LOB package

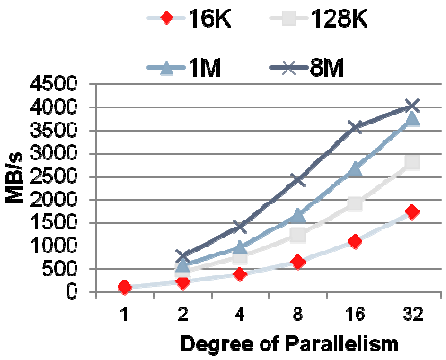


ORACLE

Oracle Database 12c: Parallel Load & Move LOBs

5x to 17x speedup in loading & moving SecureFiles

- Enhanced Parallelism for DML, CTAS & MOVE for SecureFiles
  - Intra-partition parallelism
  - Parallel move for Non-partitioned table
- Linear Scaling with degree of parallelism
- Helps exploit multi-core & I/O parallelism



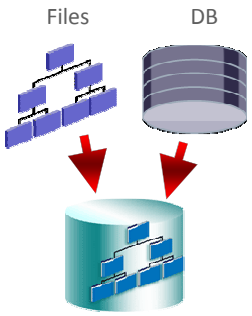
ORACLE

Program Agenda

1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

ORACLE

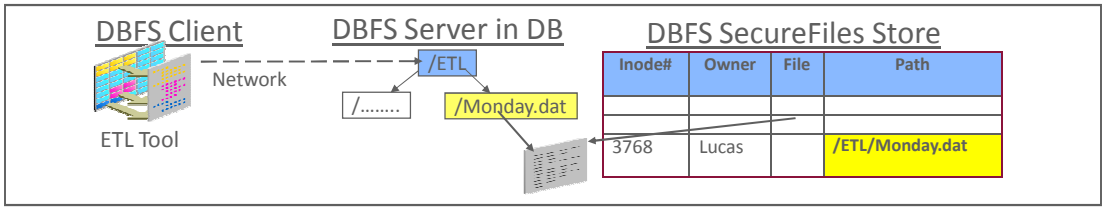
Files in the Database Reinvented



- Oracle Database 11g reinvented files in the database
- SecureFiles removes performance barrier to storing files in the database
- DBFS provides simple file system interface to files stored in the database
  - Enables existing file based tools to access database files
  - Familiar access through pathnames, directories, links
  - Files kept in a dedicated file store, or existing application tables
- Storing business data files inside the database is now simpler, faster, and more robust than storing them outside

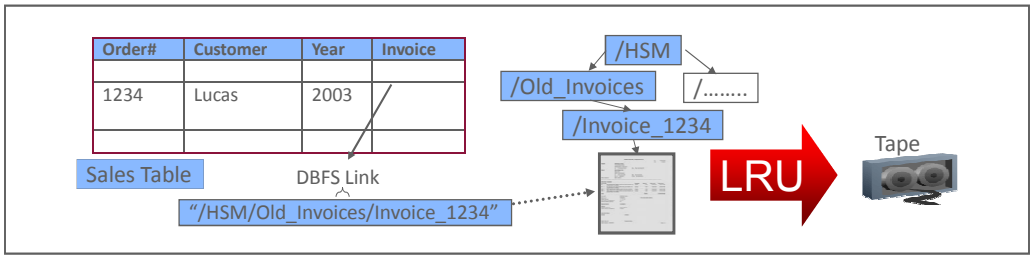
DBFS SecureFiles Store

- DBFS: stand-alone file systems in the database
  - Directory information stored in tables
  - Files stored in SecureFiles LOBs
- Used for operational application files such as ETL files, reports, etc. that are not in application tables
  - Provides unified data and file backup, DR, management



DBFS HSM Store

- A DBFS HSM store allows archiving files to tape
- Application migrates older files to HSM store (e.g. old invoices)
  - A DBFS Link replaces the LOB, LOB reads on links are transparent
  - A LOB can be easily migrated back to the table for updates
- HSM store: disk staging area for storing recently accessed files
  - Seldom accessed files are migrated to tape, brought back on reference



Rich Capabilities Inherited from DB

DBFS Capability	Provided By
Compression, Deduplication, Encryption	SecureFiles
Crash Tolerance	Atomic transactions, Logging
Mirroring, Striping, Online Add Storage	ASM
Disaster Recovery, Readable Remote Mirror	Data Guard
Consistent Backup	RMAN, Hot backup
Multi-Node Scalability, Transparent Failover	RAC
Impromptu Snapshots	Consistent Read
Restore to Point in Time	Flashback, Media Recovery
Retention / Compliance	Total Recall
Network Security	SSL

DBFS Client Interfaces

- Linux File System Client
- New in 12c: WebDAV
- Command Line Client

Program Agenda

1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

SecureFiles Internals

Overview

- SecureFiles support CLOB, BLOB and NCLOB data types
- A SecureFile object is a column in a table, there can be more than one SecureFiles LOB in a single table
- A SecureFile is made up of two parts:
  - LOBSEGMENT
  - LOBINDEX
- A SecureFile LOB can be stored in-line if less than approx. 4K bytes, or out-of-line
- A SecureFile object is a collection of variable sized chunks and each chunk is a set of contiguous database blocks

ORACLE

 29

SecureFiles Internals

Structure

- Each SecureFile object is identified by a LOB locator which points to the very first block of the SecureFile object
- The LOB locator is stored in the table column along with metadata about the SecureFile object. Metadata includes:
  - Length of the file
  - Whether the file is compressed, encrypted or de-duplicated
  - The starting block addresses and lengths if stored in-line

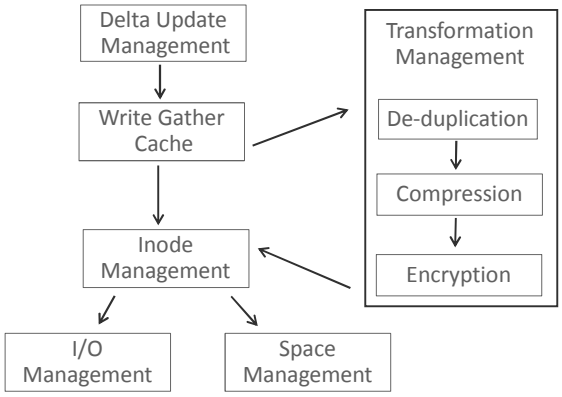
ORACLE

 30

SecureFiles Internals

Architecture

- Oracle SecureFiles architecture is layered into six major components:



SecureFiles Internals

Delta Updates

- Non-length preserving updates are supported through 'delta updates'
- Allows a user to specify:
  - Object to update
  - Content to change
  - Length of the content
  - Start and end offsets
- The delta update component maintains its own metadata structures to record the mapping of source and destination offsets for each of the deltas

SecureFiles Internals

Write Gather Cache

- The WGC buffers up to 4MB of in-flight data for every server process which results in fewer space management operations
- It also results in fewer, but larger contiguous space allocations
- Allows write performance throughput to scale up with the number of server processes

SecureFiles Internals

Transformation Management

- De-duplication – if enabled and a duplicate is found then a pointer to the original version is stored
- Compression – Automatically detects if compression makes sense and can use multiple file compression algorithms. Compression is performed in pieces to allow efficient random access of large files
- Encryption – Uses Transparent Data Encryption (TDE), supports automatic key management and encrypts/decrypts on database block size units

SecureFiles Internals

Inode Management

- Inode management tracks all chunks in the SecureFiles LOB
- Initiates on-disk storage and access operations
- Requests free space to store data flushed by the Write Gather Cache
- Free space is returned in variable chunk sizes up to 64M by the space management layer
- Inode metadata and on-disk space metadata is tracked within the LOB segment itself and changes modify Oracle data blocks in the buffer cache and are always logged in the redo logs

ORACLE

SecureFiles Internals

Space Management

- The space management layer allocates free disk space and de-allocates used space
- A background free space monitor manages the growth of segments
- Space is managed on a Committed Free Space (CFS) or Uncommitted Free Space (UFS) list.
- The space management layer supports ‘copy-on-write’ semantics
- Undo records do not need to be generated for larger update and overwrite operations, the previous version is available because of ‘copy-on-write’ semantics. This means rollback may not need to perform any I/O

ORACLE

SecureFiles Internals

I/O Management

- The I/O Management layer is responsible for satisfying I/O requests during reads and writes of SecureFile objects
- During writes the I/O manager processes chunks (direct writes) from the Inode Manager and buffers (cache writes) from the write gather cache
- The I/O manager supports read-ahead or pre-fetching of data from disk. It tracks access patterns and issues intelligent pre-fetching of chunks before the request is actually made

SecureFiles Internals

In-line LOB Storage

- Row Dump

```
tab 0, row 0, @0x1bf7
tl: 881 fb: --H-FL-- lb: 0x0  cc: 4
col  0: [ 2]  c1 02
col  1: [ 9]  68 73 63 5f 73 74 61 74 73
col  2: *NULL*
col  3: [861]
 00 54 00 01 01 0c 00 80 00 01 00 00 00 01 00 00 00 1a 48 a3 03 49 48 90 03
< data deleted >
LOB
Locator:
  Length:        84(861)
  Version:        1
  Byte Length:    1
  LobID: 00.00.00.01.00.00.00.1a.48.a3
  Flags[ 0x01 0x0c 0x00 0x80 ]:
    Type: BLOB
    Storage: SecureFile
    Characterset Format: IMPLICIT
    Partitioned Table: No
    Options: ReadWrite
  SecureFile Header:
    Length:   841
    Old Flag: 0x48 [ DataInRow SecureFile ]
    Flag 0:   0x90 [ INODE Valid ]
    Layers:
      Lengths Array: INODE:835
      INODE:
        01 00 03 3e 01 0a 4e 41 4d 45 20 20 20 20 20 20 20 20 20 20
< data deleted - the in-line LOB >
```

SecureFiles Internals

Out-of-line LOB Storage

- Row Dump

```
tab 0, row 1, @0x1b83
tl: 62 fb: --H-FL-- lb: 0x1  cc: 4
col  0: [ 2]  c1 03
col  1: [15]  4c 4f 42 5f 31 31 67 5f 75 6e 69 66 6f 72 6d
col  2: *NULL*
col  3: [38]
 00 54 00 01 01 0c 00 80 00 01 00 00 00 01 00 00 00 1a 48 a4 00 12 40 90 00
 0c 21 00 2a cf 01 00 01 01 80 00 eb 02
LOB
Locator:
  Length:        84(38)
  Version:        1
  Byte Length:    1
  LobID: 00.00.00.01.00.00.00.1a.48.a4
  Flags[ 0x01 0x0c 0x00 0x80 ]:
    Type: BLOB
    Storage: SecureFile
    Characterset Format: IMPLICIT
    Partitioned Table: No
    Options: ReadWrite
  SecureFile Header:
    Length:   18
    Old Flag: 0x40 [ SecureFile ]
    Flag 0:   0x90 [ INODE Valid ]
    Layers:
      Lengths Array: INODE:12
      INODE:
        21 00 2a cf 01 00 01 01 80 00 eb 02
end_of_block_dump
End dump data blocks tsn: 4 file#: 6 minblk 221 maxblk 221
```


SecureFiles Internals

Out-of-line LOB Storage

- LOBSEGMENT Block Dump

```
Block header dump:  0x018000eb
 Object id on Block? Y
 seg/obj: 0x18f75  csc: 0x00.571400  itc: 1  flg: E  typ: 5 - LOCAL LOBS
     fsl: 0  fnx: 0xffffffff ver: 0x01

 Itl           Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0x0004.00b.00000ddf  0x00000000.0000.00  -B--    0  fsc 0x0000.00000000
========
bdba     [0x018000eb]
kdlich   [0x7f5ecdf5024c 56]
  flg0   0x20 [ver=0 typ=data lock=n]
  flg1   0x00
  scn    0x0000.00571400
  lid    000000010000001a48a4
  rid    0x00000000.0000
kdlidh   [0x7f5ecdf50264 24]
  flg2   0x00 [ver=0 lid=short-rowid hash=n cmap=n pfill=n]
  flg3   0x00
  pskip  0
  sskip  0
  hash   00000000000000000000000000000000000000000000
  hwm    8060
  spr    0
  data   [0x7f5ecdf50280 52 8060]
0a 54 4b 50 52 4f 46 3a 20 52 65 6c 65 61 73 65 20 31 31 2e 32 2e 30 2e 33 2e
30 20 2d 20 44 65 76 65 6c 6f 70 6d 65 6e 74 20 6f 6e 20 54 68 75 20 46 65 62
< Data deleted >
```

SecureFiles Internals

Space Usage

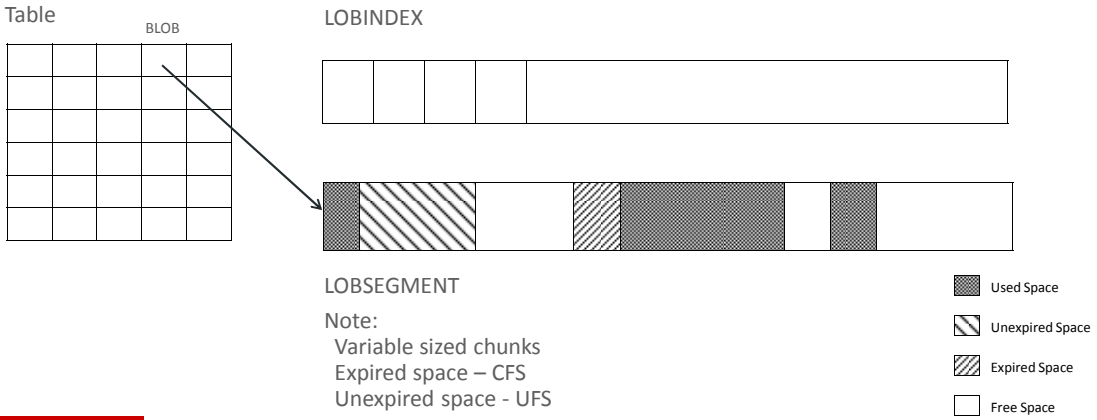
- Space usage for SecureFiles LOBs can be viewed using the DBMS_SPACE.SPACE_USAGE procedure – There are many script for formatting, I used Note: 861344.1

```
SQL> exec check_space_sf;
Segment Blocks = 152 Bytes = 1245184
Used Blocks = 2 Bytes = 16384
Expired Blocks = 83 Bytes = 679936
Unexpired Blocks = 0 Bytes = 0
=============================================

PL/SQL procedure successfully completed.

SQL>
```

 41

SecureFiles Internals

SecureFile Layout – Out-of-Line LOB



Note:
Variable sized chunks
Expired space – CFS
Unexpired space - UFS

 42

Program Agenda

1. What are SecureFiles?
2. Performance
3. Features
4. DBFS
5. Internals
6. Wrap Up

Migration to SecureFiles

- Easiest approach is to just enable SecureFiles on new partitions
  - Old data stays as LOBs
- Migrating existing data requires data movement
  - Table rebuild
    - Can be done at the partition level
  - Online Table Redefinition can be used to eliminate downtime
    - No need to take the table or partition offline
    - Additional storage equal to the entire table and all LOB segments must be available
    - Global indexes need to be rebuilt.
    - Recommend setting NOLOGGING storage attribute for destination SecureFile columns during migration to avoid performance problems with redo generation
    - If the destination table is partition, online redefinition can be done in parallel

SecureFiles Parameters

- Enable/Disable Storage In Row
- CACHE/NOCACHE/CACHE READS
  - CACHE READS supports direct writes
- LOGGING/NOLOGGING/FILESYSTEM_LIKE_LOGGING
  - CACHE and NOLOGGING is not a valid combination
- RETENTION – NONE, AUTO, MAX (requires maxsize)
  - Main reason for space usage complaints
  - Default is AUTO (retain UNDO sufficient for consistent reads)
- CHUNK – Ignored for SecureFiles

 45

Bugs

- Since Oracle 11.1 there have been a lot of SecureFiles bugs
- You should be using 11.2.0.4 or 12.1 to avoid most SecureFiles bugs

 46

Misconceptions

- Space Management – This always comes up. SecureFiles are not architected like tables. They're meant to have a "pool" of storage, much like a file system
- Storage usage – There is no shrink option, SecureFiles automatically manages space. Space is not meant to be returned to the tablespace
- Performance – You can't compare DBFS performance to native file systems, it should be compared to NFS
- Migration – You have to write to the SecureFiles format in some manner to migrate. It doesn't just happen. We recommend that you use online redefinition

ORACLE

References

- Database SecureFiles and Large Objects Developer's Guide
- Oracle SecureFiles System
- Oracle SecureFiles: Prepared for the Digital Deluge
- Note 861344.1 – 11g Advanced Compression – How to Check Space Occupied by LOB Compression
- Note 66431.1 – LOBS – Storage, Redo and Performance Issues
- Note 1453350.1 – How to Determine what storage is used in a LOBSEGMENT and should it be shrunk / reorganized?
- Note 1451124.1 – How to Shrink (make less sparse) a LOB (BASICFILE or SECUREFILE)?
- Note 386341.1 – How to determine the actual size of the LOB segments and how to free the deleted/unused space above/below the HWM

ORACLE

Hardware and Software
Engineered to Work Together

ORACLE

ORACLE