Agenda

• Data Growth Challenges
• Advanced Compression Feature Overview
  • Table Compression
  • Unstructured Data Compression
  • Backup Compression
  • Data Guard Network Transport Compression
• Competitive Analysis
• Questions and Answers
Challenges

- Explosion in Data Volumes
  - Government Regulations (Sarbanes-Oxley, etc)
  - User Generated Content (Web 2.0)
  - Application Consolidation
- IT Managers Must Support Larger Volumes of Data with Limited Technology Budgets
  - Need to optimize storage consumption
  - Also maintain acceptable application performance
- Intelligent and Efficient Compression Technology can Help Address These Challenges
Introducing Advanced Compression Option

• Oracle Database 11g introduces a comprehensive set of compression capabilities
  • Structured/Relational data compression
  • Unstructured data compression
  • Compression for backup data
  • Network transport compression

• Reduces resource requirements and costs
  • Storage System
  • Network Bandwidth
  • Memory Usage
Table Compression

- Introduced in Oracle Database 9i Release 2
  - Compression during bulk load operations (Direct Load, CTAS)
  - Data modified using conventional DML not compressed
- Optimized compression algorithm for relational data
- Improved performance for queries accessing large amounts of data
  - Fewer IOs
  - Buffer Cache efficiency
- Data is compressed at the database block level
- Compression enabled at either the table or partition level
- Completely transparent to applications
OLTP Table Compression

- Oracle Database 11g extends table compression for OLTP data
  - Support for conventional DML Operations (INSERT, UPDATE, DELETE)
- New algorithm significantly reduces write overhead
  - Batched compression ensures no impact for most OLTP transactions
- No impact on reads
  - Reads may actually see improved performance due to fewer IOs and enhanced memory efficiency
OLTP Table Compression Process

Legend

- **Gray** | Header Data
- **Uncompr.** | Uncompressed Data
- **Free Space**
- **Compr.** | Compressed Data
Block-Level *Batch* Compression

- Patent pending algorithm minimizes performance overhead and maximizes compression
- Individual INSERT and UPDATEs do not cause recompression
- Compression cost is amortized over several DML operations
- Block-level (Local) compression keeps up with frequent data changes in OLTP environments
  - Competitors use static, fixed size dictionary table thereby compromising compression benefits
- Extends industry standard compression algorithm to databases
  - Compression utilities such as GZIP and BZ2 use similar adaptive, block level compression
OLTP Table Compression

Employee Table

<table>
<thead>
<tr>
<th>ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>Doe</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>3</td>
<td>John</td>
<td>Smith</td>
</tr>
<tr>
<td>4</td>
<td>Jane</td>
<td>Doe</td>
</tr>
</tbody>
</table>

Initially Uncompressed Block

Header

```
1•John•Doe 2•Jane•Doe 3•John•Smith 4•Jane•Doe
Free Space
```
OLTP Table Compression

Employee Table

<table>
<thead>
<tr>
<th>ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>Doe</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>3</td>
<td>John</td>
<td>Smith</td>
</tr>
<tr>
<td>4</td>
<td>Jane</td>
<td>Doe</td>
</tr>
</tbody>
</table>

Initially Uncompressed Block

Header

1•John•Doe,2•Jane•Doe,3•John•Smith,4•Jane•Doe

Free Space

INSERT INTO EMPLOYEE
VALUES (5, ‘Jack’, ‘Smith’);
COMMIT;
## OLTP Table Compression

### Employee Table

<table>
<thead>
<tr>
<th>ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>Doe</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>3</td>
<td>John</td>
<td>Smith</td>
</tr>
<tr>
<td>4</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>5</td>
<td>Jack</td>
<td>Smith</td>
</tr>
</tbody>
</table>

### Block

| Header |
|========|
| John=0 | Doe=1 | Jane=2 | Smith=3 |

| 1•John•Doe 2•Jane•Doe 3•John•Smith 4•Jane•Doe |

| Free Space |

### Local Symbol Table
## OLTP Table Compression

### Employee Table

<table>
<thead>
<tr>
<th>ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>Doe</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>3</td>
<td>John</td>
<td>Smith</td>
</tr>
<tr>
<td>4</td>
<td>Jane</td>
<td>Doe</td>
</tr>
<tr>
<td>5</td>
<td>Jack</td>
<td>Smith</td>
</tr>
</tbody>
</table>

### Compressed Block

**Header**

- John=0
- Doe=1
- Jane=2
- Smith=3

**Local Symbol Table**

- 1 0 1 2 2 1 3 0 3 4 2
- 1 5 5 6

**Free Space**
### OLTP Table Compression

#### Uncompressed Block

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>1•John•Doe 2•Jane•Doe 3•John•Smith 4•Jane•Doe 5•Jack•Smith</td>
</tr>
</tbody>
</table>

#### Compressed Block

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>John=0 Doe=1 Jane=2 Smith=3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>More Data Per Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>1•0•1 2•2•1 3•0•3 4•2</td>
</tr>
<tr>
<td>5•Jack•3</td>
</tr>
</tbody>
</table>

#### Local Symbol Table

- Free Space
Table Compression Syntax

OLTP Table Compression Syntax:

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

Direct Load Compression Syntax (default):

```sql
CREATE TABLE emp (  
    emp_id NUMBER  
, first_name VARCHAR2(128)  
, last_name VARCHAR2(128)  
) COMPRESS [FOR DIRECT_LOAD OPERATIONS];
```
OLTP Table Compression

Best Practices

• Compress your 10 Largest Tables
  • 80/20 Rule – 20% of your Tables Consume 80% of your Space
• Better Compression with Bigger Block Sizes
  • Higher Probability of Duplicate Values
• Sorted Data Improves Compression Ratio
  • Sort on columns with medium cardinality data
• B-Tree Index Compression
  • Validate Index and Review INDEX_STATS
    • INDEX_STATS.COMP_CMPR_COUNT
    • INDEX_STATS.COMP_CMPR_PCTSAVE
• Bitmap Indexes are Highly Compressed
  • Good for low and medium cardinality
OLTP Table Compression

Getting Started with Compression

- Compression Advisor (Coming Soon)
  - Shows projected compression ratio for uncompressed tables
  - Reports actual compression ratio for compressed tables
OLTP Table Compression

Getting Started with Compression

- Monitoring Compression
  - DBA_TABLES
    - COMPRESSION (ENABLED/DISABLED)
    - COMPRESS_FOR (ALL OPERATIONS / DIRECT LOAD OPERATIONS)

```sql
SQL> SELECT table_name, compression, compress_for
2  FROM user_tables
3  WHERE compression = 'ENABLED';

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>COMPRESS</th>
<th>COMPRESS_FOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>SALES_COMPRESS</td>
<td>ENABLED</td>
<td>DIRECT LOAD ONLY</td>
</tr>
<tr>
<td>Hz_PARAM_TAB_C</td>
<td>ENABLED</td>
<td>FOR ALL OPERATIONS</td>
</tr>
</tbody>
</table>

Elapsed: 00:00:00.29
SQL>
```
Table Compression Results
Real World Compression Overview

- Data from Oracle’s implementation of Oracle Applications
  - Compressed the 10 Largest Tables
    - Two copies of each table: Compressed and Uncompressed
  - Oracle Enterprise Linux
  - Oracle Database 11g Release 1

- Test Queries
  - Full Table Scan
  - Index Range Scan
  - DML Operations (Insert, Update, Delete)
Real World Compression Results

Storage Utilization

More than 70% Storage Savings
Real World Compression Results

Table Scan Performance

<table>
<thead>
<tr>
<th>Time (seconds)</th>
<th>No Compression</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>0.2</td>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td>0.3</td>
<td></td>
<td>0.4</td>
</tr>
</tbody>
</table>

Compression is 2.5x Faster
Real World Compression Results

Table Scan Reads

Compression performs 3.5x fewer reads
Real World Compression Results

Index Range Scan Performance

Time (seconds)

No Performance Impact on Index Scans

No Compression  Compression
Real World Compression Results

DML Performance

Time (seconds)

Compression has less than 3% overhead
Introduction to SecureFiles

- SecureFiles is a new 11g feature designed to break the performance barrier keeping file data out of databases
- Next-generation LOB
  - Superset of LOB interfaces allows easy migration from LOBs
  - Transparent deduplication, compression, and encryption
  - Leverage the security, reliability, and scalability of database
- Enables consolidation of file data with associated relational data
  - Single security model
  - Single view of data
  - Single management of data
  - Scalable to any level using SMP scale-up or grid scale-out
SecureFiles 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
- Duplicate detection happens within a table, partition or sub-partition
- Specially useful for content management, email applications and data archival applications
SecureFiles Compression

- Huge storage savings
  - Industry standard compression algorithms
  - 2-3x compression for typical files (doc, pdf, xml)

- Allows for random reads and writes to Compressed SecureFile data

- Can be specified at a partition level

- Automatically detects if SecureFile data is compressible
  - Skips compression for already compressed data
  - Auto-turn off compression when space savings are minimal or zero

- Two levels of compression provide different compression ratios

- SecureFiles Compression is independent of table or index compression
SecureFiles Compression Syntax

Compression Syntax

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

Deduplication Syntax

```sql
CREATE TABLE images (  
  image_id NUMBER,  
  image BLOB)  
  LOB(image) STORE AS SECUREFILE  
(TABLESPACE lob_tbs DEDUPLICATE);
```
Want to Know More?

Technical Session

Next Generation Unstructured Data Management with SecureFiles
Date/Time: Tuesday, 11/13, 3:15 - 4:15 PM
Location: Moscone South 304

Demonstration

Oracle DEMOgrounds
Moscone West Exhibition Hall
Data Pump Compression

- Metadata compression available since Oracle Database 10g
- Oracle Database 11g extends compression to table data during exports
  - No need to decompress before import
- Single step compression of both data and metadata
  - Compressed data directly hits disk resulting in reduced disk space requirements
    - Internal tests reduced dump file size up to 75%
- Compression factor comparable to GNU gzip utility
- Application transparent
  - Complete Data Pump functionality available on compressed files
Backup Compression

- Fast RMAN Compression
  - Compresses the backup set contents before writing them to disk or tape
  - No extra decompression steps are required during recovery when you use RMAN compression
  - High performance, industry standard compression algorithm
    - 40% faster backup compression versus Oracle Database 10g
  - Suitable for fast, incremental daily backups
  - Reduces network usage
Backup Compression Syntax

FAST RMAN Compression Configuration

```sql
RMAN> CONFIGURE COMPRESSION ALGORITHM 'zlib';
```

RMAN Compression Syntax

```sql
RMAN> backup as COMPRESSED BACKUPSET database archivelog all
```

DataPump Syntax

```sql
PROMPT> expdp hr FULL=y
    DUMPFILE=dpump_dir:full.dmp COMPRESS
```

ORACLE
RMAN Compression Results
RMAN Compression Overview

- Data from Oracle’s implementation of Oracle Applications
  - 3.5 GB Database
  - Oracle Enterprise Linux
  - Oracle Database 11g Release 1
- Test 1: Slow I/O (16 MB/s)
  - 11g RMAN without Compression
  - 10g RMAN with Compression
  - 11g RMAN with FAST Compression
- Test 2: Fast I/O (200 MB/s)
  - 11g RMAN without Compression
  - 10g RMAN with Compression
  - 11g RMAN with FAST Compression
Backup Compression Results

Backup Size Comparison

Compression reduced backup size by 6x
Backup Compression

Backup Speed Comparison
Slow I/O (Tape)

11g Compression reduces backup time by almost 3x
Backup Speed Comparison

Slow I/O (Tape)

11g Compression is almost 2.5x faster than 10g Compression

No Compression  10g Compression  11g Compression
Backup Compression

Backup Speed Comparison
Fast I/O (Disk)

11g Compression is almost 2.5x faster than 10g Compression
Oracle Data Guard

- **Redo Shipping**
  - Send redo data over network from primary to standby
  - Size of redo data typically small (transactional) and not network-bound

- **Gap Resolution**
  - After network outage – resynchronize standby
  - Size of redo data much larger
Network Compression

*Oracle Data Guard Redo Transport Services*

- Fast re-sync of standby database after network outages
- Lower bandwidth networks (<100Mbps)
  - 15-35% less time required to transmit 1 GB of data
  - Bandwidth consumption reduced up to 35%
- High bandwidth networks (>100 Mbps)
  - Compression will not reduce transmission time
  - But will reduce bandwidth consumption up to 35%
Competitive Analysis
## Competitive Analysis – DB2

<table>
<thead>
<tr>
<th>Oracle</th>
<th>IBM DB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block Level Compression</td>
<td>Table/Partition Level Symbol Table</td>
</tr>
<tr>
<td>- Adaptive / Dynamic Compression</td>
<td>- New data may not be compressed / offline rebuild symbol table</td>
</tr>
<tr>
<td>- Unlimited Compression Values</td>
<td>- Max 4K Values per Table</td>
</tr>
<tr>
<td>Zero uncompression overhead</td>
<td>Some uncompression overhead</td>
</tr>
<tr>
<td>- Minimal compression overhead</td>
<td>- More compression overhead</td>
</tr>
<tr>
<td>- Batched compression</td>
<td>- Transactional compression</td>
</tr>
<tr>
<td>Index Compression</td>
<td>None</td>
</tr>
<tr>
<td>Unstructured Data Compression</td>
<td>None</td>
</tr>
<tr>
<td>(SecureFiles/LOB)</td>
<td></td>
</tr>
</tbody>
</table>
Competitive Analysis – DB2

• Even without compression, Oracle needs less disk space than DB2:
  • Oracle uses variable length representation for numbers – DB2 uses fixed length
  • DB2 has more overhead for VARCHAR data

• SAP BW data needs 30-50% less disk space if stored in Oracle
# Competitive Analysis – Teradata

<table>
<thead>
<tr>
<th>Oracle</th>
<th>Teradata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block Level Compression</td>
<td>Field Level Compression</td>
</tr>
<tr>
<td>- Dynamic Compression Values</td>
<td>- Admin must define compression values</td>
</tr>
<tr>
<td>- Unlimited values</td>
<td>- Max 255 Compression Values</td>
</tr>
<tr>
<td>All Data Types Supported</td>
<td>Limited Data Type Support</td>
</tr>
<tr>
<td>- Minimal compression overhead</td>
<td>- More compression overhead</td>
</tr>
<tr>
<td>- Batched compression</td>
<td>- Transactional compression</td>
</tr>
<tr>
<td>Backup Compression</td>
<td>None</td>
</tr>
<tr>
<td>Unstructured Data Compression (SecureFiles/LOB)</td>
<td>None</td>
</tr>
</tbody>
</table>
Competitive Analysis

Compression: Oracle vs Competition
Top 10 Tables in an ERP Application
Summary

- Advanced Compression Option contains comprehensive data compression capabilities for all types of data
  - Structured, Unstructured, Backup, Network Transport
- Reduces storage consumption by 2 to 4 times
- Improves read performance
- Enhances memory, buffer cache utilization
- Complete application transparency
- Benefits diverse application workloads
Want More Information?

- Advanced Compression White Paper

- Advanced Compression Data Sheet

- Advanced Compression Tutorial

Demonstration

Oracle DEMOgrounds
Moscone West Exhibition Hall
ORACLE IS THE INFORMATION COMPANY