Oracle GRID Center

Verification of Oracle Database 11g for Data Warehousing Using Fujitsu SPARC Enterprise

—Performance Improvement Based on Data Segment Compression and ASM Utilization—

Creation Date: December 16, 2008

Version: 1.0

ORACLE

FUJITSU

THE POSSIBILITIES ARE INFINITE

A. Introduction

Since concluding an OEM agreement in 1989, Oracle Corporation Japan (“Oracle Japan”) and Fujitsu Limited (“Fujitsu”) have engaged in various joint efforts to provide solutions that deliver safety and security to clients, including system construction, joint verification, and post-installation support.

In November 2006, Oracle Japan established the Oracle GRID Center (http://www.oracle.co.jp/solutions/grid_center/index.html), featuring cutting-edge technology with the goal of creating next-generation grid-based business solutions that optimize corporate system infrastructures. Lending its full support to establish the Oracle GRID Center, Fujitsu engages in joint technical verification efforts with Oracle Japan at the Oracle GRID Center, drawing on its server and storage products.

As part of efforts to improve the performance of data warehousing systems, Oracle Japan and Fujitsu have recently performed verification tests at the Oracle GRID Center. These tests sought to confirm the validity of the Oracle Database 11g data segment compression function and the Automatic Storage Management rebalancing function using Fujitsu’s newest SPARC Enterprise series UNIX servers. The results of the verification tests are reported below.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

B. Table of Contents

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

C. Goals of Verification Tests

The verification tests described here seek to demonstrate the effectiveness of the Oracle Database 11g data segment compression function and the Automatic Storage Management (ASM) dynamic disk addition and data rebalancing functions to improve data warehousing (DWH) system performance. These tests involved Fujitsu's newest SPARC Enterprise UNIX servers.

The data segment compression function originally offered in Oracle 9i Database Release 2 is capable of achieving effective use of storage space by compressing huge data volumes in a DWH environment and reducing total data volumes. Since compression reduces data volumes, it also reduces disk I/O, the most costly operation in general database processing. This is expected to improve query performance.

However, compression also tends to generate CPU overhead. In verifying data segment compression, we examined the effects of the CPU overhead generated by compression on performance and changes in query performance resulting from reduced disk I/O.

Disks are added for databases to achieve two purposes: to meet growing data volumes and to improve performance by distributing data across additional disks. Adding disks involves complex, costly procedures such as data backup, disk addition/reconfiguration, data reloading, index reproduction, and reacquisition of statistical data. It's also highly difficult to perform these tasks without shutting down business databases, however briefly. ASM, the disk management layer of Oracle Database, permits the redistribution (rebalancing) of data to added disks through straightforward procedures that don't involve shutting down business databases. In verifying the ASM rebalancing function, as part of efforts to confirm the validity of the ASM rebalancing function, we checked query performance improvements resulting from the additional disks and from data rebalancing, as well as reductions in man-hours required to add the disks.

In these verification tests, we loaded a large volume of DWH data and executed complex, high-load queries repeatedly over an extended period. Since this subjected the system to constantly high loads, it also served as a stress test for the SPARC Enterprise and Oracle Database 11g combination during verification testing. The SPARC Enterprise series inherits the compatibility and reliability of the Fujitsu UNIX platform and Oracle Database, fostered through the development of the PRIMEPOWER family. Successful completion of these verification tests testifies to the sustained compatibility and reliability of the SPARC Enterprise and Oracle Database 11g pairing.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

D. Fujitsu SPARC Enterprise

The enterprise information systems used by corporations today consist of three server layers: database servers, application servers, and Web servers. Multiple purpose-specific servers are widely used to configure multilayer models.

Fujitsu SPARC Enterprise is a series of UNIX servers developed to meet the emerging needs of clients. It is ideal for modern enterprise systems in which each server is assigned a specific task, such as Web front-end and back-end operations. The SPARC Enterprise is a new UNIX server standard that features the technologies used in Fujitsu's PRIMEPOWER and Sun Microsystems' Sun Fire. The high reliability and high availability technologies accumulated by Fujitsu in developing mainframes, coupled with the advanced network computing technologies of Sun Microsystems, provide superb performance, business continuity, and virtualization functions.

The SPARC Enterprise series is equipped with the newest SPARC processors, including the SPARC64 VI, UltraSPARC T1, and UltraSPARC T2, and runs the Solaris OS, a global-standard version of UNIX. It retains binary compatibility with previous Solaris versions and enables seamless migration from the PRIMEPOWER or Sun Fire to the SPARC Enterprise.

To resolve various management issues that companies face, including timely management, business continuity, and TCO reductions, the SPARC Enterprise series encompasses a wide range of models for various business applications, including the M4000, M5000, M8000, and M9000—models suitable for back-end operations such as large-scale databases and mission-critical tasks—and the T1000, T2000, T5120, and T5220, which are ideal for Web front-end operations.

1. SPARC Enterprise M5000

The SPARC Enterprise M5000 is a midrange-class server packed with high performance, high reliability, and virtualization technologies previously found only in high-end servers. This model inherits the reliability of mainframes and incorporates a SPARC64 VI processor with improved performance. Each unit can implement up to 16 cores and 32 threads.

Back-end operations such as database and batch processing are characterized by large loads per transaction and individual transaction processing. The new SPARC64 VI high-performance processor has been developed for high-speed processing of high-load transactions. By combining powerful parallel command processing capabilities and high-precision command branch prediction technologies with newly-developed multi-core, multi-thread technologies, the SPARC Enterprise M5000 achieves remarkable performance. It also incorporates an improved system bus and the latest I/O interfaces to achieve optimal overall system performance.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Operations such as database and batch processing can result in major business repercussions if they stop functioning. Based on the design concept of mainframes, models in the SPARC Enterprise series feature various technologies that minimize down time, including error-prevention functions that help prevent server failures and malfunctions from suspending client operations; self-correcting functions that rectify or circumvent problems to ensure operational continuity; component redundancy; and hot-swapping capabilities.

Operations that cause high server loads vary over time, typically clustering during daylight hours, at nighttime, at the beginning of the month, or at the end of the month. Different server groups have typically been arranged for use at different peak times. The SPARC Enterprise M5000 can dynamically divide server resources and reconfigure them based on partitioning and dynamic reconfiguration functions, adding or removing resources as needed, allowing it to flexibly adapt to various business tasks.

[Features of the SPARC Enterprise M5000]

- Inherits the mainframe reliability and features of the SPARC64 VI processor with improved performance.
- Multi-core, multi-thread capabilities permit 2-core, 4-thread processing per processor.
- Revamped bus bandwidth and the latest I/O interfaces achieve high performance.
- Extensive data protection measures and redundancy design improve unit availability.
- Partitioning, DR, and COD functions enable flexible server operations.

2. SPARC Enterprise T2000

The SPARC Enterprise T2000 is equipped with an UltraSPARC T1 processor capable of executing 8 cores and 32 threads per processor.

Since Web front-end operations are characterized by simultaneous processing of many low-load transactions, the servers performing such tasks are required to offer high throughput performance to ensure rapid processing of a great number of transactions.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

The UltraSPARC T1 processor is a high-throughput processor developed to process large transaction volumes. It efficiently processes commands using up to 32 threads, making it ideal for applications that require parallel processing of multiple tasks such as Web front-end operations.

The SPARC Enterprise T2000 also saves space and energy. Its small physical footprint requires minimal floor space to deploy, and its energy-saving characteristics reduce operating costs. Since this model permits higher-density systems than conventional UNIX servers, it offers improved processing performance per unit of space or per unit of power consumption. Compared to comparable UNIX servers manufactured by other companies, the SPARC Enterprise T2000 realizes remarkable energy and space savings—consuming as little as half the power and requiring as little as 25% of the installation space. By concentrating operations in Solaris Containers, clients can consolidate growing number of servers to further reduce TCO.

[Features of SPARC Enterprise T2000]

- Equipped with the UltraSPARC T1 multi-core processor.
- Eco-server offering space-saving (2U size) and energy-saving features.
- Equipped with the latest I/O interfaces, including PCI Express and SAS disk.
- Disks, power supply units, and fans feature redundancy design and are hot-swappable.
- Remote monitoring and control of servers can be performed via LAN.

3. ETERNUS storage systems

Fujitsu offers a wide range of SAN-compatible disk arrays to meet an even wider range of needs, including the ETERNUS8000 enterprise disk array with the world-leading capacity of 2.04 PB (petabytes), the EXTERNUS4000 midrange disk array offering high cost-performance, and the ENTERNUS2000 entry-level disk array with its space-saving, energy-saving, and quiet operating characteristics.

Assuring the highest level of reliability for each part and system, the ETERNUS series is ideal for securely storing valuable data. In addition to excellent connectivity for a multi-platform environment, this series also realizes the integration of storage based on SAN, while the advanced copy function performs high-speed on-line data backup to dramatically reduce backup times.

The ETERNUS models store important corporate data both reliably and efficiently, providing optimal storage solutions to resolve various issues facing corporations, including increasingly stringent compliance requirements.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

3.1. ETERNUS4000 Model 500

The ETERNUS4000 Model 500 midrange disk array incorporates a high-speed 2-GHz dual-core processor and a 4-Gbps Fiber Channel interface to achieve high processing capabilities. Expandable to 313 TB, this model can meet large-scale storage integration needs and growing data volumes.

The main components like the controller, power supply, fan, and battery feature redundant design. In the event of a power outage, the internal battery provides power to write data in the cache memory to the hard disk to ensure data safety for an unlimited number of days. The RAID group can be configured with RAID6 with two parity disks to protect data from double disk failures. This high-reliability design supports uninterrupted operations around the clock 365 days a year.

The combined use of the advanced copy function of the hardware and the ETERNUS SF AdvancedCopy Manager software enables high-speed copy/backup during business operations. Multiple generations of backup data can be maintained and managed, while data can be transferred between remote disks to protect data from potential calamities.

[Features of ETERNUS4000 Model 500]

- High scalability up to 313 TB.
- High-speed 2-GHz dual-core processor and 4-Gbps Fiber Channel interface for high processing capabilities.
- High reliability design to support uninterrupted operation for 24 hours a day, 365 days a year.
- Advanced copy function for high-speed production of backups during business operation.
- Data can be transferred between remote disks to protect data against potential calamities.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

E. Oracle Database Functions Used in Our Verification Tests

The functions of the Oracle Database used in our verification tests are introduced below.

1. Data segment compression

The data segment compression function compresses data segments by eliminating duplicate values found in each data block. Duplicate values in a block are stored in the symbol table at the beginning of the block, and duplicate values in each record are replaced by small pointers that reference the actual value in the symbol table. This reduces the overall record length while increasing the number of records that can be stored in a block, effectively compressing data segments. In short, this function compresses data to reduce disk I/O, a typical bottleneck in DWH environments, thereby improving query response.

Until the release of the Oracle Database 10g Release 2 version, data segment compression functions compressed data during direct loading of data, mainly for DWH. In contrast, the Oracle Advanced Compression function, an optional feature in Oracle Database 11g Enterprise Edition, offers extended capacity to execute compression even for ordinary DML statements that do not call for direct loads, thereby providing the benefits of compression regardless of the data loading method.

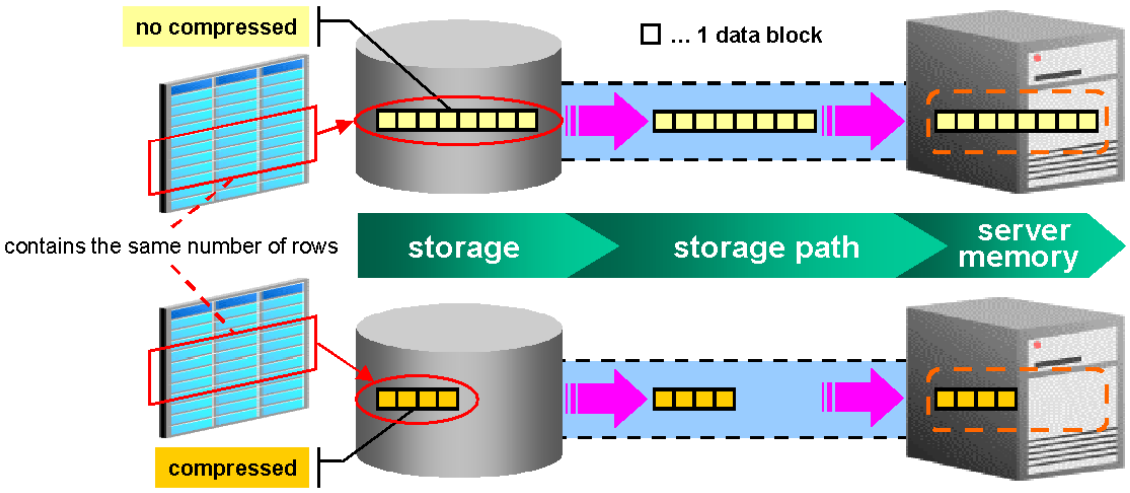

Figure 1 Data segment compression

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

2. Parallel query

The parallel query function executes the execution request of a single SQL using multiple processes. Since it can perform a full table scan, a full index scan, and sorting in parallel, it is especially effective in DWH environments characterized by accesses of large tables. Since multi-core, multi-thread CPUs in recent servers deliver significant multi-processing performance improvement, parallel queries allow effective use of server resources and achieve dramatic improvements in query response.

3. Automatic Storage Management

Oracle Database 10g features Automatic Storage Management (ASM), a function that manages the disks used for databases. In addition to multi-disk striping and mirroring functions, it provides a rebalancing function capable of dynamically changing disk configurations. Since this function allows the addition of disks without shutting down business databases, it can reduce disk management costs and improve performance by distributing disk loads.

4. External table

SQL*Loader has conventionally been used to load data from text data file sources.

The external table allows use of SQL to query data in OS files such as CSV files, just like table data in a database. The external table has certain restrictions, however. For example, only the SELECT statement can be executed for an external table; DML statements can’t be used, nor can indexes be generated. However, just like ordinary tables, external tables can be selected, merged, and sorted by the SELECT statement, and parallel queries can be performed. In short, by using external tables, data can be loaded while performing advanced conversion processing not possible with SQL*Loader.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

F. Verification Environment

Our verification testing featured a single database client, database server, and storage device. Processing of SQL and other functions for verification was performed on the database server.

1. System configuration

Figure 2 shows the schematics of the system configuration used in our verification tests. The client and database server were linked by 1000Base-T, while the database server and storage device were linked by four 4-Gbps Fiber Channels via an FC switch. The ETERNUS multipath driver was used to distribute loads to the four paths between the database server and storage device.

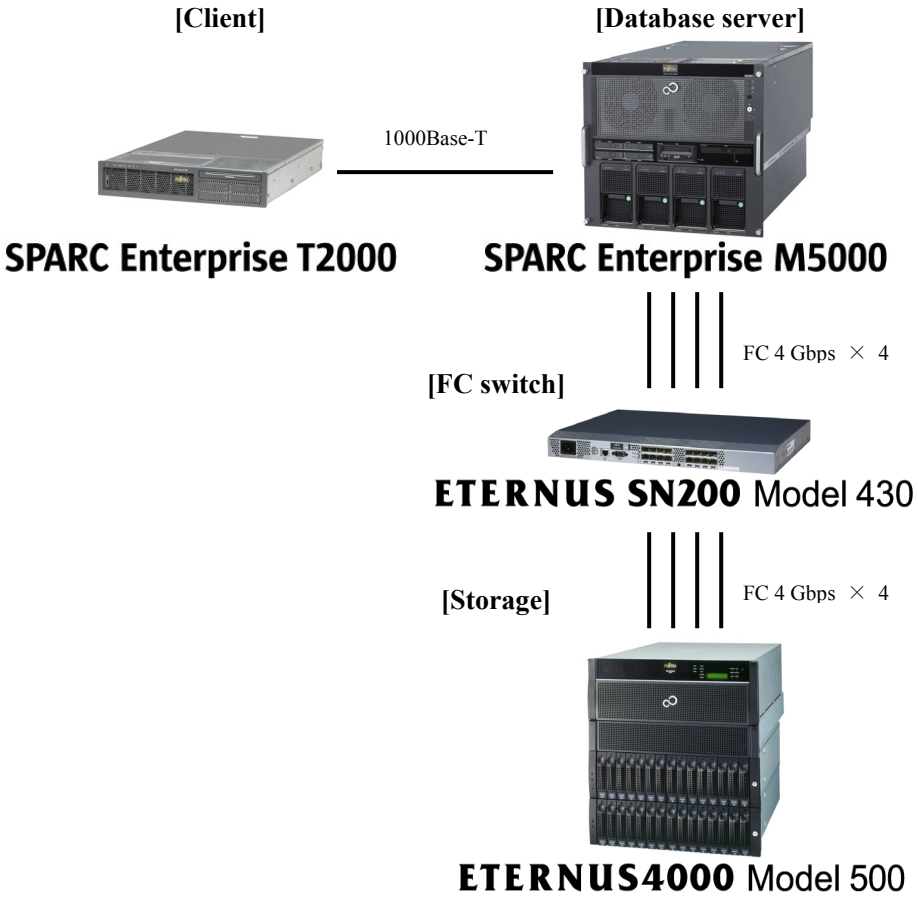

Figure 2 Schematics of system configuration

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Shown below are the equipment specifications and software used.

1.1. Database server

Hardware

Model	Fujitsu SPARC Enterprise M5000
CPU	SPARC64 VI, 2.15 GHz, 5-MB cache 4 CPUs, 8 cores, 16 threads
Memory	32 GB
Internal HDD	73-GB (10,000 rpm) SAS disk x 2
I/O Board	2
FC Card	Dual-channel 4-Gbps FC card x 2

Software

OS	Solaris 10 OS 8/07 (SunOS 5.10 Generic_127111-03)
Storage software	ETERNUS Multipath Driver 2.0.3
Database	Oracle Database 11g (11.1.0.6)

1.2. Storage

Model	Fujitsu ETERNUS 4000 Model 500
Controller (CM)	2
Memory	16 GB (8 GB/CM)
Channel adapter	4-Gbps 2-port FCCA x 2 (1 FCCA/CM)
Drive enclosure	4
Disk drive	300 GB (15,000 rpm) x 60

1.3. Client

Hardware

Model	Fujitsu SPARC Enterprise T2000
CPU	UltraSPARC T1, 1.2-GHz, 3-MB cache 1 CPU, 8 cores, 32 threads
Memory	32 GB
Internal HDD	73-GB (10,000 rpm) SAS disk x 2

Software

OS	Solaris 10 OS 8/07 (SunOS 5.10 Generic_127111-03)
Database client	Oracle Database 11g (11.1.0.6)

1.4. FC switch

Model	Fujitsu ETERNUS SN200 M430
Transfer capacity	4 Gbps
Number of ports	8

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

2. Storage configuration

Storage RAID groups, volumes, and ASM disk group configuration used in the verification testing are described below.

As shown in the Storage configuration diagram in Figure 3, a total of 14 RAID groups were established, with each RAID group consisting of RAID1+0 (four disk drives). Four 126-GB logic volumes were created only in RAID group #0000 which required the system area for the ETERNUS, and six 91-GB logic groups were created in each of the other RAID groups.

The ASM disk groups were configured as shown in Table A.

Table A ASM disk group configuration

ASM disk group	RAID group	Application
SYS_DG	#0000	System, Sysaux, Undo table areas, Redo log, Control file
TEMP_DG	#0001–#000D	Temporary table area
COMP_DG	#000A–#000D	Table area for compressed data
NOCO_DG	#0006–#0009	Table area for non-compressed data

In our verification tests, we stored small-load data, such as System, Sysaux, Undo table areas, Redo log files, and control files, in RAID group #0000.

Since the temporary table area would require high I/O speed if sorting, table merge, and indexing could not be completed in available memory, logical volumes in the RAID groups other than #0000 were used one by one to distribute disk loads.

Generally, in order to simplify management and obtain good performance, we configure all the RAID groups as one ASM diskgroup. But in this test, to compare absence or presence of compression attribute under the same condition, we made above 4 ASM diskgroups and made RAID groups belong to each ASM disk group. This is because to eliminate differences which may be produced like outer and inner circumference of disk, deviation of disk controller, and more.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

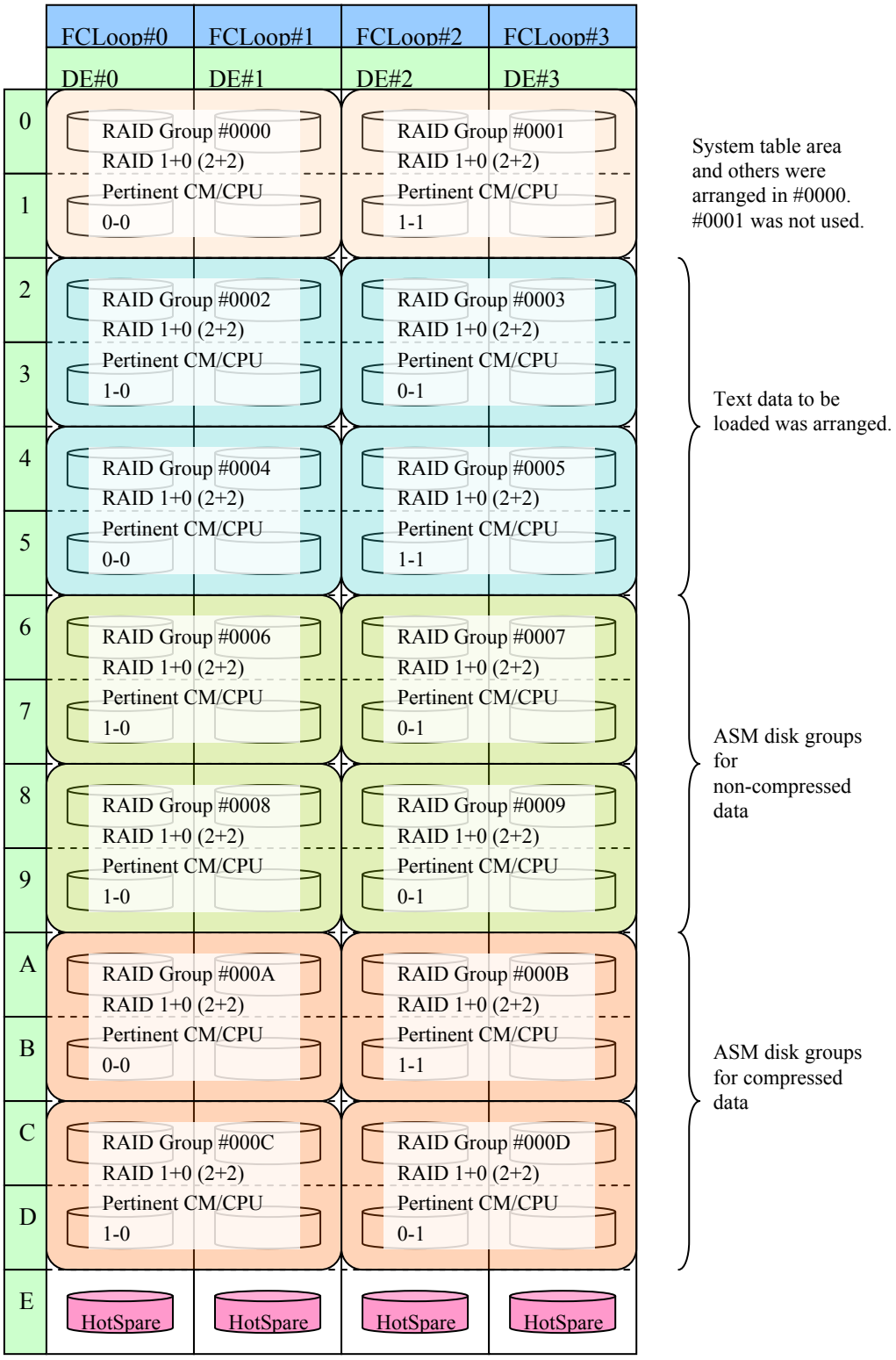

Figure 3 Storage configuration

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

3. Schema

The following shows the schema of the verification model used in the tests.

3.1. Overview of the schema

The verification schema assumed DWH. Table B illustrates the tables, number of rows, and non-compressed data size.

Table B

Table name	rows	size
T1	1,200,000,000	153GB
T2	160,000,000	22.8GB
T3	40,000,000	6GB
T4	5	192KB
T5	25	192KB
T6	2,000,000	352MB
T7	30,000,000	4.75GB
T8	300,000,000	34.48GB

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

3.2. Verification queries

We used 18 types of queries for DWH in our verification tests. Table C describes each of these queries.

Table C Verification queries

Query No.	Query description
1	This query calculates the sales amount, discount amount, and so on for each returned item flag and order status in a given two-month period.
2	This query finds the supplier whose product supply cost for an area is lowest.
3	This query fetches the total income from orders received in a one-day period from a specific client.
4	This query fetches the number of orders for each priority level from orders whose receipt date postdates the transaction confirmation date over a given three-month period.
5	This query calculates the income from each country in an area.
6	This query calculates total income from items for which each order was less than 25 pieces AND a discount rate of 0.3 to 0.5 was applied over a given one-year period.
7	This query calculates income from orders for which one of the supplier or client was located in country A AND the other was located in country B over a given two-year period.
8	This query calculates the French share of a product in the Europe market.
9	This query calculates profits for products matching the keyword in the search (“like” search in middle search).
10	This query calculates total amount for products returned over a given three-month period.
11	This query calculates total amount for products whose supply cost in Canada was 0.00002% or more of the total and the total supply cost of those products.
12	This query fetches the number of orders with high priority levels and the number of orders with low priority levels over a given one-year period.
13	Among clients ordering products that are not ordinary packaged products, this query fetches the number of non-packaged product orders placed by each applicable client.
14	This query calculates the sales of products over a one-year period matching the keyword in the search (“like” search in prefix search) based on product type.
15	This query produces a view of the income from each supplier over a given three-month period and searches for the supplier with the highest income in the view.
16	This query finds clients purchasing 300 or more of the same products within a single order.
17	This query fetches the number of orders in standby status from each supplier in Vietnam.
18	This query fetches the total amounts owed by clients who have outstanding balances exceeding the national average for outstanding balance.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

G. Details of Verification

In our verification tests, we loaded data into tables and executed queries to assess the performance of data segment compression and dynamic disk addition and data rebalancing by ASM. The specifics of loading data into tables and query execution are described below.

1. Loading data into tables

We produced the empty tables shown in the schema and an external table that references text data. Based on the external table, we performed a parallel direct data load for the empty tables. We used a separate application to generate the text data to be referenced by the external table, storing this data in four RAID groups differing from the RAID groups used for data files.

Since the tests used a 4-CPU, 8-core, 16-thread verification environment, parallelism was set to 16. Block size was 32 KB.

2. Executing queries

We executed the 18 types of queries listed in Table C Verification queries at 16 parallelism degree. As with loading data into tables, we selected this parallelism based on the 4-CPU, 8-core, 16-thread verification environment.

The details of verification of data segment compression and ASM rebalancing are described below.

3. Data segment compression

All tables used in our verification tests were produced by specifying COMPRESS FOR ALL OPERATIONS, a compression attribute of the CREATE TABLE statement. Using COMPRESS FOR ALL OPERATIONS allowed us to segment compression as specified by Oracle data segment compression without selecting a data insertion method. In the verification tests, we compared whether the absence or presence of the compression attribute would make any difference, not just in segment size, but with respect to data loading and query execution times.

4. Dynamic disk addition and data rebalancing by ASM

As explained in the section describing Automatic Storage Management, when disks are added to or removed from an ASM disk group, data rebalancing is performed for the ASM disks. In our verification tests, we verified whether the query performance would change when data loaded to one RAID group was rebalanced to two RAID groups or four RAID groups to improve I/O performance.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

H. Verification Results

The results of our verification tests are discussed below.

1. Data segment compression

1.1. Difference in segment size by compression

Figure 4 shows the relative values for total table size (size in non-compressed state = 1). Compressing the data used in the verification tests reduced its size to approximately 62% of the non-compressed data. (Note that the relative size of a compressed table depends on the definitions of columns and data content.)

The following discussion addresses differences between compressed and non-compressed tables.

Among the tables used in our verification tests, the T1 table exhibited the highest compression rates, while the T2 table exhibited the lowest. Figure 5 shows their relative sizes post-compression, with a value of "1" assigned to the size of the non-compressed data.

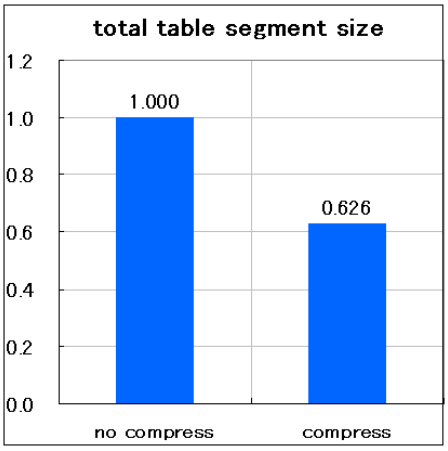

Figure 4
Relative values for total table size

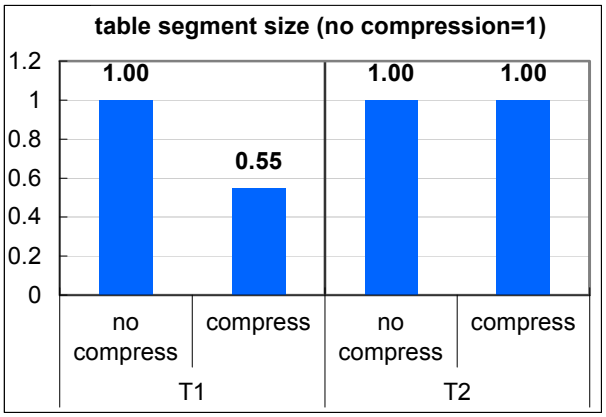

Figure 5
Relative sizes of the T1 and T2 tables

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

The column definitions for each table are given below

- T1 table

Number of column	Type	Restriction
c1	DATE	
c2	NUMBER	NOT NULL
c3	NUMBER	NOT NULL
c4	NUMBER	NOT NULL
c5	NUMBER	NOT NULL
c6	NUMBER	NOT NULL
c7	CHAR(1)	
c8	NUMBER	NOT NULL
c9	CHAR(1)	
c10	NUMBER	NOT NULL
c11	DATE	
c12	DATE	
c13	CHAR(10)	
c14	NUMBER	NOT NULL
c15	CHAR(25)	
c16	VARCHAR2(44)	

- T2 table

Name of column	Type	Restriction
c1	NUMBER	NOT NULL、PK
c2	NUMBER	NOT NULL、PK
c3	NUMBER	NOT NULL
c4	NUMBER	
c5	VARCHAR(199)	

The data segment compression feature compresses data by eliminating duplicate values in a database block. The duplicate values found in not only within a column but also in a database block will be compressed. In this test, to investigate the tendency of data, we compared the cardinality of the most compressed table T1 and the least compressed table T2. Cardinality is the number of distinct values for each "column" and the compression effect is not decided only by Cardinality. However, Cardinality can become one element which affects it to the compression effect.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

First, let's examine the T1 table which exhibited the highest compression rate. The results of calculations based on the number of blocks and number of rows showed an average of 239 rows stored per block when the T1 table was not compressed and an average of 438 rows stored per block when the T1 table was compressed. To study this data trend, we checked for duplicate values in each column of the first 438 rows, which is equivalent to an average T1 table block size. The results are shown below.

```
select count(*)                          count,
(abbrev. count(distinct COLUMNNAME) for each column )
from T1
where ...

COUNT   C1   C2 C3   C4   C5  C6 C7   C8 C9 C10  C11  C12 C13 C14 C15  C16
----- ---- ---- -- ---- ---- --- -- ---- -- --- ---- ---- --- --- --- ----
  438  388  111 11  438  438  50  3  438  2   9  381  391   7   7   4  434
```

Since the number of unique values in the columns in the 438 rows was less than 438, most columns featured duplicate values. C13 column and C15 column featured low cardinality, or number of column data types: only 7 or 4 unique values in the 438 rows. Since c13 and c15 were fixed-length rows of CHAR(10) and CHAR(25), the compression made it possible to reduce the area previously occupied by the data corresponding to the (10+25)*438 rows in the non-compressed table to the size of the actual data (10*7+25*4) stored in the symbol table and a pointer.

When the cardinality of table columns with relatively long column lengths is low, compression tends to result in greater reductions.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Next, let's examine the T2 table, which featured the lowest compression rates. The results of calculations based on the number of blocks and number of rows showed an average of 212 rows stored per block when the T2 table was either compressed or not compressed. To study this data trend, we checked for duplicate values in each column of the first 212 rows, which was equivalent to an average block size of the compressed T2 table. The results are given below.

```
select count(*)          count,
(abbrev. count(distinct COLUMNNAME) for each column )

from T2
where ...

COUNT          C2          C3          C4          C5
----- ---------- ---------- ---------- ----------
  212         212         212         208         212
```

In the 212 rows, we found 208 unique values in C4 column, meaning only four rows were duplicates. The remaining four columns each contained 212 unique values in the 212 rows, indicating the absence of duplication. In short, all columns featured very high cardinality. When all columns have high cardinality, it is difficult to compress data.

1.2. Loading data to tables

We compared the time required to load data into all of the tables used in the verification tests by performing and omitting data segment compression.

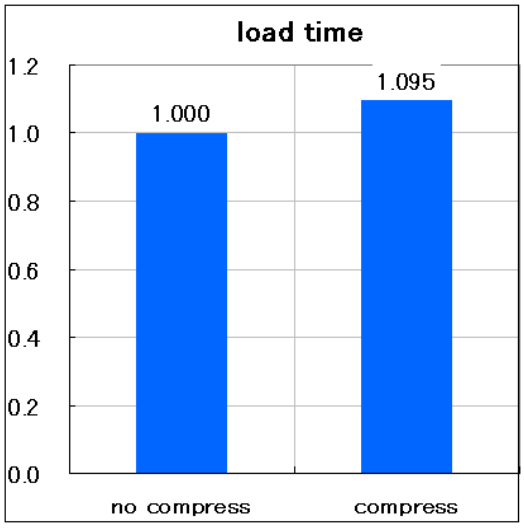

Figure 6 Relative value of total data load time

We produced an external table that referenced a text file used as the source data and performed a parallel direct load for empty tables while converting the data using the external table. In this operation, the ASM disk group storing the tables consisted of a single RAID group (RAID 1+0, with four disk drives).

Figure 6 gives the relative value of the data load time ("1.000" assigned to the time required to load non-compressed data).

Figure 7 shows the temporal changes in CPU usage during the loading of the T1 table (approx. 153 GB in non-compressed state), the largest of all the tables used in our verification tests. This data loading operation was not a simple loading operation like one

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

by SQL*Loader. It involved reading row data from the external table via the INSERT ... SELECT statement, conversion of row data, and direct insertions. During the first half ("TRANSFORMATION" time in Figure 7), CPU usage was about 60% for both non-compressed and compressed data. This time was used to convert the text data read via the external table (this process was described in the earlier section). The time required for conversion was about the same for non-compressed and compressed data, as shown in Figure 7.

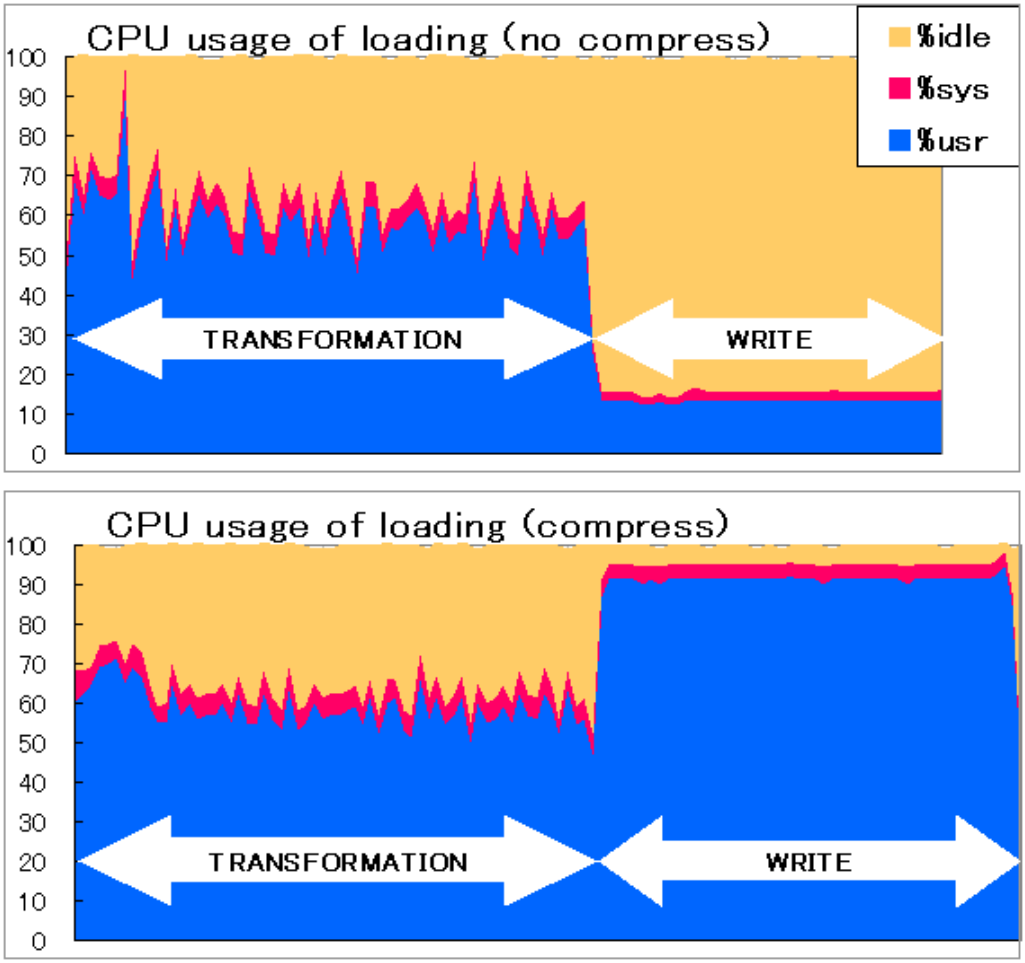

Figure 7 Temporal change in CPU usage during loading of T1 table

We believed that the difference in load times between compressed data and non-compressed data was attributable to the writing of data to a file ("WRITE" time in Figure 7) following data conversion process.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Figure 8 (one RAID group) and Figure 9 (two RAID groups) show the temporal changes in CPU usage during the loading of one of the tables used in the verification tests without data conversion. This indicates the difference in data load times only. The graphs on the top show the results of loading non-compressed data, while the graphs on the bottom show the results of loading compressed data.

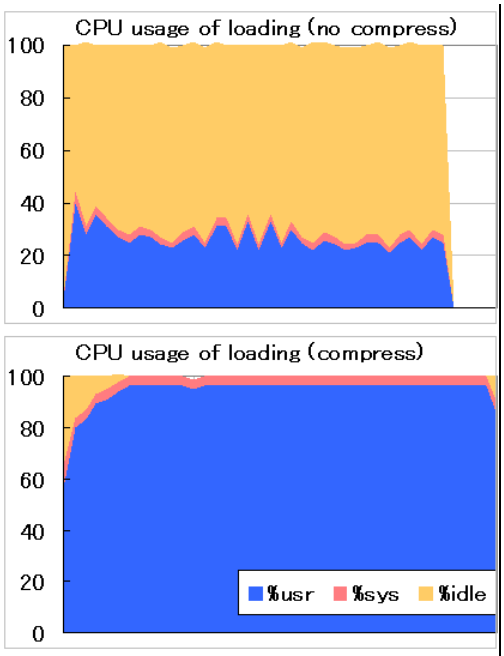

Figure 8 Temporal change in CPU usage (one RAID group)

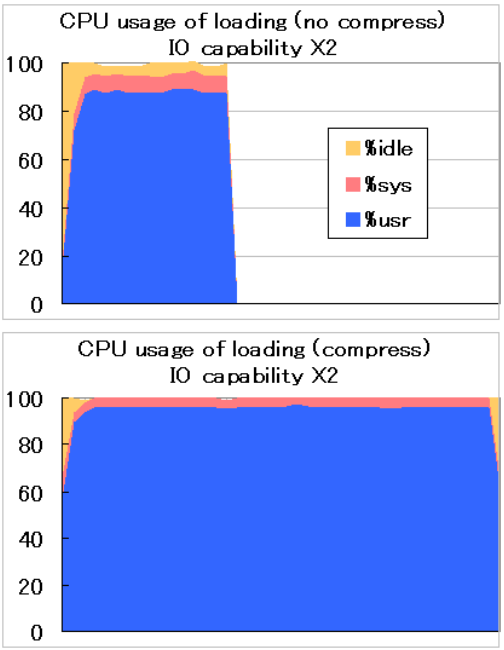

Figure 9 Temporal change in CPU usage (two RAID groups)

Figure 8 shows that compression increases CPU usage. Generally, data loading is a process that generates high CPU loads. However, for non-compressed data, the CPU use remains at approximately 30%. This means disk I/O is the likely cause of the bottleneck.

To clarify this issue, we doubled the number of disks used, which meant doubling disk I/O performance, and performed data loading without conversion. Figure 9 shows the results. This case resolves disk I/O, a bottleneck in loading non-compressed data, and increases CPU use. In terms of processing time, we observed greater differences between compressed data and non-compressed data for the results shown in Figure 9 than in Figure 8.

Based on these results, we concluded that differences in data load times between compression and non-compression were relatively small in environments in which disk I/O was a bottleneck and relatively large in environments in which a factor other than disk I/O—for example, CPU performance—becomes a bottleneck.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

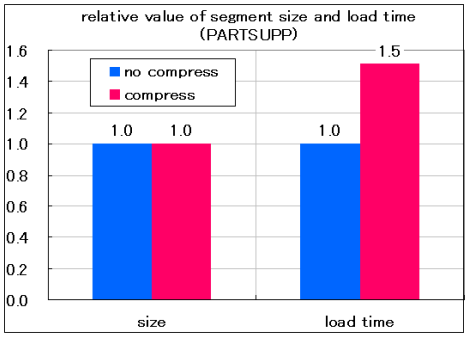

Figure 10 T2 table load times and relative size

Next, we focused on the time required to load data to the T2 table, the one with the lowest compression rate, and the relative size of this table. Figure 10 gives the relative values, with a value of "1" assigned to the load time and segment size of the non-compressed data. As Figure 10 shows, it took longer to load data to the table with a compression attribute.

This is because even non-compressed data was checked for duplicate values, generating overhead.

When applying segment compression, we must consider the "redundancy in the data" of each column in the target tables from the perspective of load times.

1.3. Query execution

The following discusses the effects of compression on query performance.

Figure 11 shows the execution times for each of 18 queries whose specifics are given in the section on Verification queries. The ASM disk group used to store the tables in our verification tests consisted of a single RAID group. Although execution times varied with query processing type, the graph shows that execution times were generally faster for queries involving compressed tables.

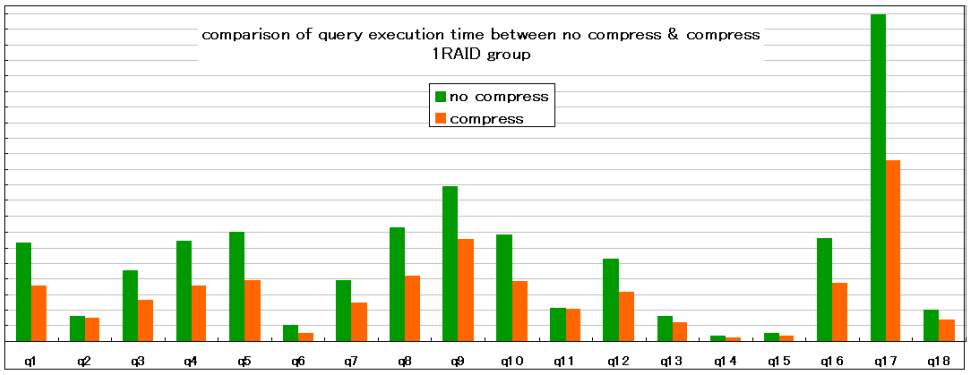

Figure 11 Query execution times (single RAID group)

Figure 12 shows the relative execution times for each query for compressed data. To show the effects of compression on each query, the baseline execution time for each query for the non-compressed data was set to "1."

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

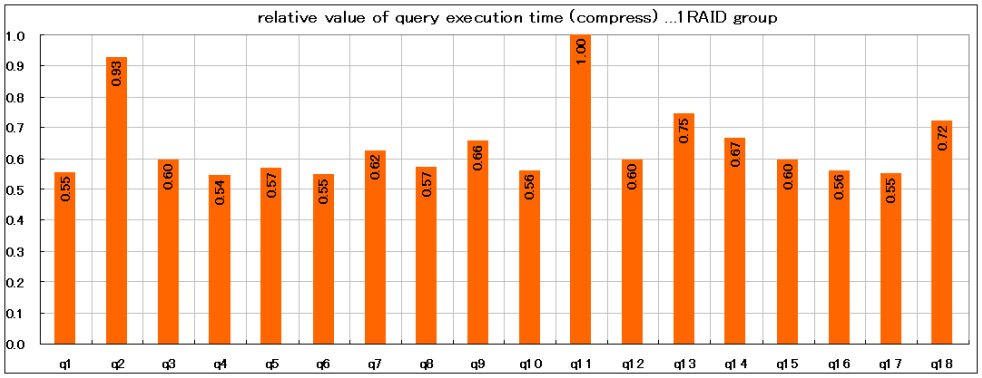

Figure 12 Relative query execution times (baseline query execution times for non-compressed data, single RAID group, set to "1.00")

Next, we compared actual disk I/O. To Figure 12, we added relative values for physical reads in query execution for the compressed table. The baseline number of physical reads in query execution for a non-compressed table was set to "1." Figure 13 shows the results.

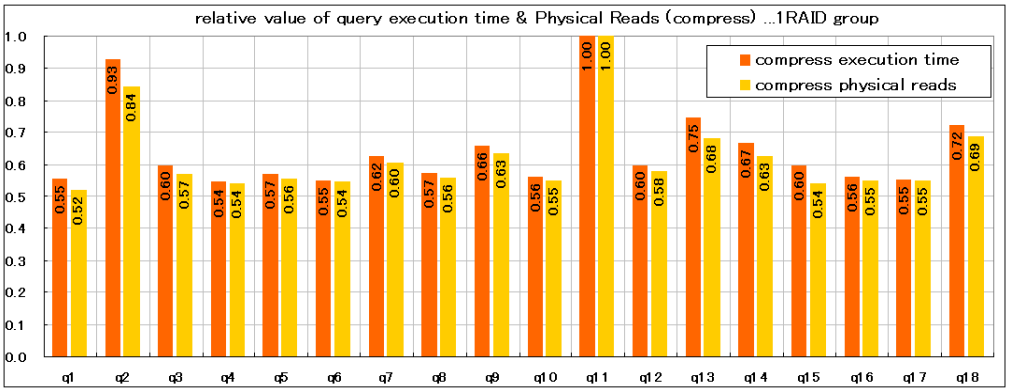

Figure 13 Relative values for query execution times and physical reads (baseline for query execution times and physical reads for non-compressed data set to "1.00")

The results for query execution times and physical reads showed similar trends, and the coefficient of correlation between the two was quite high—0.98724—strongly suggesting that faster queries are attributable to decreased I/O resulting from compression. Since compressing data segments increases the number of rows in a single block, loading the same number of rows with compressed data involves reading fewer blocks.

Compressing data reduces the number of blocks that need to be read to read the same number of rows, increasing I/O efficiency. We examined how improved I/O efficiency changes CPU use during query execution. For queries q1 and q13, Figure 14 (q1) and Figure 15 (q13) show examples of temporal changes in CPU use during execution. The ratio of physical reads for q1:q13 was approximately 4:1.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

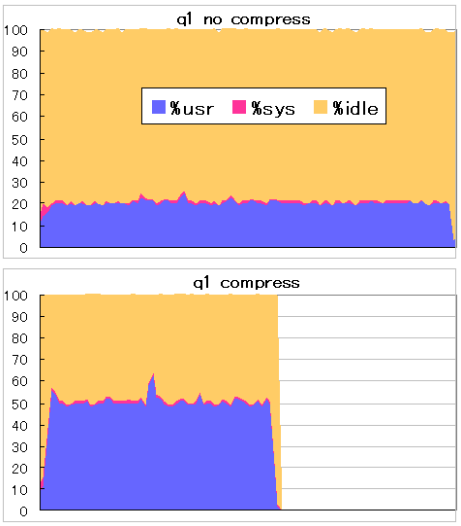

Figure 14 Temporal changes in CPU use during execution of q1 (top: non-compressed, bottom: compressed)

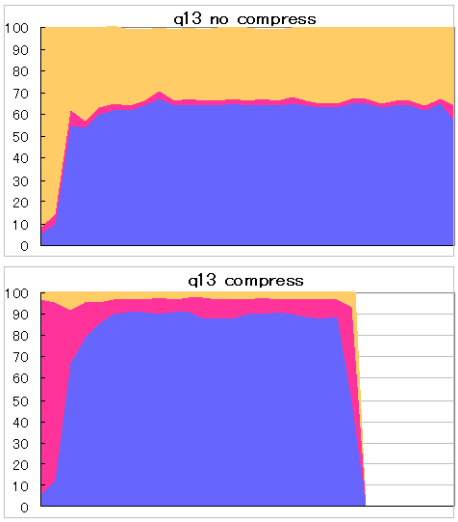

Figure 15 Temporal changes in CPU use during execution of q13 (top: non-compressed, bottom: compressed)

With both queries, query execution times decreased when data was compressed, while CPU use increased, indicating that compression reduces disk I/O standby times and increases CPU efficiency.

1.4. Overhead in query execution resulting from segment compression

The table data used in our verification tests described thus far was stored in a single RAID group, both for compressed and non-compressed data. Disk I/O performance tended to represent the bottleneck. To improve disk I/O performance, we stored the data in four RAID groups and performed a verification test to assess whether compression would affect query performance in the same way as with a single RAID group. Figure 16 shows the query execution times for four RAID groups.

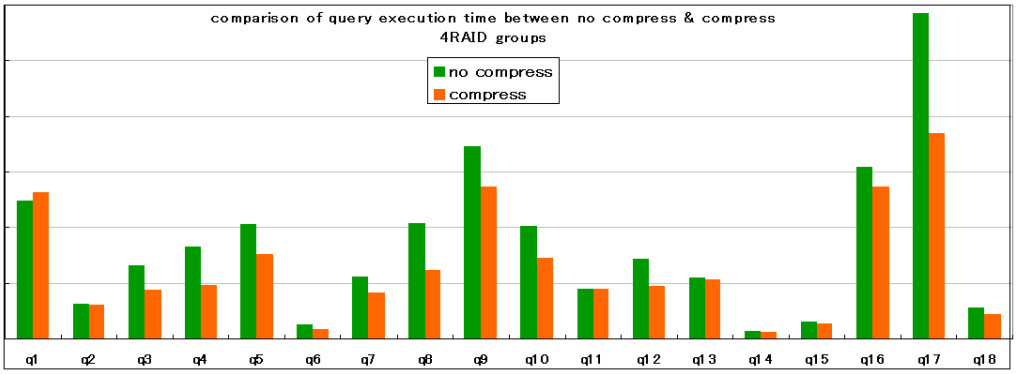

Figure 16 Query execution times (baseline query execution times for non-compressed data, four RAID groups, set to "1.00")

As with the results for the single RAID group in Figure 13, Figure 17 shows the relative values for query execution times and physical reads, with a baseline of "1" assigned to non-compressed data.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

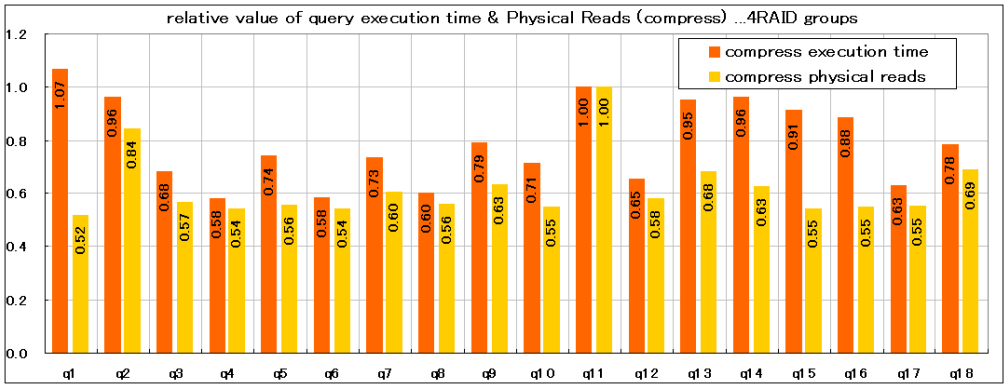

Figure 17 Relative values for query execution times and physical reads (baseline for query execution times and physical reads for non-compressed data, four RAID groups, set to "1.00")

Compared to the data obtained with a single RAID group (Figure 13), we saw marked differences in relative values for query execution times and physical reads. The calculated coefficient of correlation between the two was 0.4893, well below the value of 0.98724 obtained for data with a single RAID group. That is, the proportion of the disk I/O time to the query execution time was lower with four RAID groups, since four RAID groups offered higher disk I/O performance than a single RAID group configuration. For queries with extreme differences between the two, the bottleneck is a factor other than disk I/O performance (e.g., CPU performance).

According to Figure 16, with four RAID groups, for q1 only, query execution times for compressed data was slightly longer than for non-compressed data. Two factors help explain this result. One is the effect of CPU overhead caused by repeated pointer referencing of the symbol table. Since compression replaces some column data with a pointer to the symbol table, the symbol table must be referenced by the pointer. The overhead created by this process is all but negligible and causes no issues during normal operations, as indicated by the results for the execution of other queries. However, the q1 query involves calculating sums, average values, etc, which means that it must reference the symbol table repeatedly. Despite the low CPU overhead, when a query like q1 references the symbol table repeatedly, the effects of CPU overhead can become perceptible.

The second factor is that q1 is a computation-oriented query. For this reason, q1 tends to impose higher loads on the CPU than other queries. With other queries, response improvements achieved by reduced I/O are significant relative to CPU overhead, making CPU overhead a negligible factor. We believe the bottleneck in the case of q1 with four RAID groups was imposed by the CPU, not I/O, due to the two factors mentioned above, which increased CPU overhead and reduced response.

According to Figure 13, there were no differences in q11 execution times between compressed data and non-compressed data. q11 was a query for the T2 table, which was not compressed. As described in the example of q1, the overhead for a query for a compressed table arises when the pointer locates data. Since the non-compressed T2 table had no symbol table or pointer, although the table has a compression attribute, the data

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

access method is the same as for a query for a non-compressed table. For these reasons, processing that attempts to compress a table for which segment compression does not actually reduce size will not degrade query performance.

1.5. Summary of data segment compression

In our verification tests, we used the data segment compression function to successfully reduce overall segment size by about 40%. When there are more redundancies in the database block, compression effect is higher. Cardinality of a particular column is one of the factors that define the redundancy, but in this test, we confirmed that compression would significantly reduce table size when the table includes many columns of low cardinality or when columns of low cardinality are large. Because the data used by this verification is generated using a random number, compression effect is smaller as the overall trend. In the data set of a real system, it can be expected that it is larger.

Compressing a table generates a certain CPU overhead. The CPU overhead generated by data loads can be caused by compression-related operations, such as checking for duplicate values within each block and producing a symbol table.

On the other hand, compression increases the number of rows contained in each block, allowing more rows to be read in fewer disk I/O operations. We confirmed that compression reduced disk I/O standby times during query execution and improved processing efficiency, thereby reducing execution times.

Data segment compression reduces the query execution times, even if we account for the overhead during data load resulting from compression, serving as a very effective method to meet the needs of business users.

2. Dynamic disk addition and data rebalancing by ASM

2.1. Effect of adding ASM disks

One way to improve query performance in a DWH environment is to add disk drives to disperse I/O, thereby improving disk I/O performance. We increased the number of RAID groups used to store data from one to two, and then to four, seeking to identify how data rebalancing affects query execution times.

As shown in Figure 18, we added RAID groups to an ASM disk group comprised of a single RAID group and reconfigured the ASM disk group so that the ASM disk group used two RAID groups and four RAID groups. We then measured query performance.

To isolate the relationship between disk I/O performance and query execution times, we omitted data segment compression during this testing.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

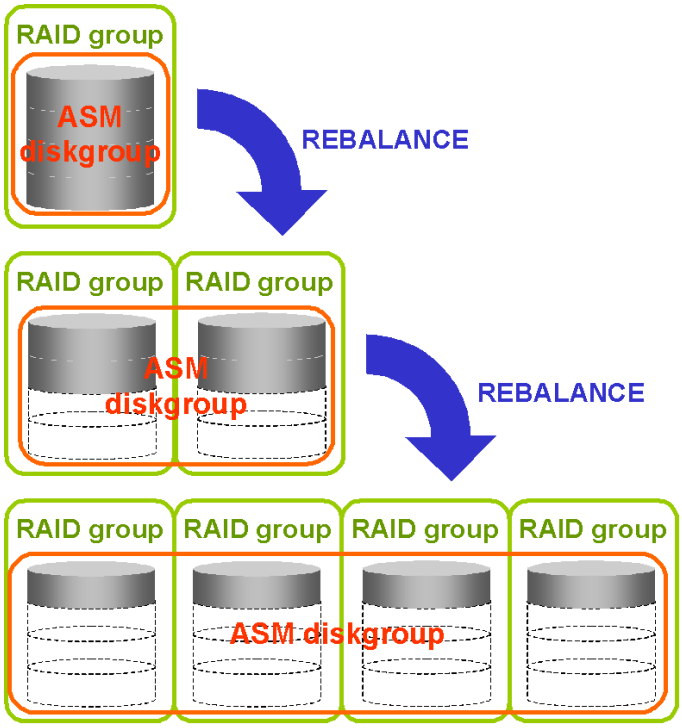

Figure 18 ASM disk addition and data rebalancing

Figure 19 shows query execution times for each disk configuration, which indicate that all queries tested tended to be faster when we increased the number of RAID groups.

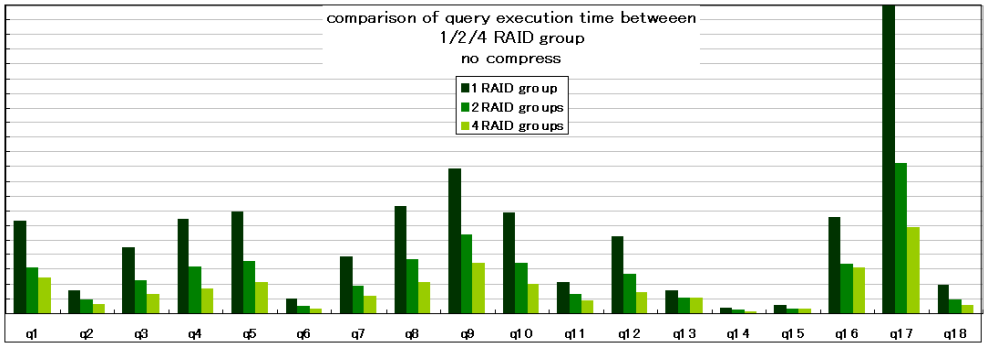

Figure 19 Query execution times

Shown below are execution times for each query with two RAID groups and with four RAID groups, relative to a baseline value of “1” set for query execution times with a single RAID group.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

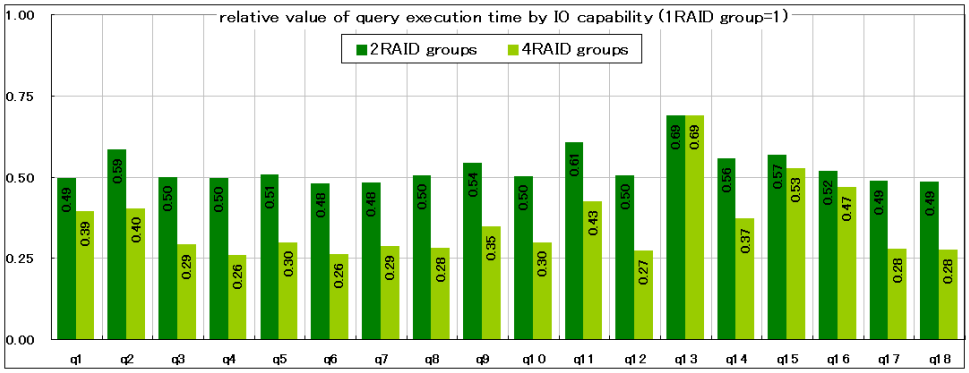

Figure 20 Relative query execution times

Compared to a single RAID group, two RAID groups improved execution times for all queries. Doubling the number of RAID groups also doubled disk I/O performance, reducing query execution times. However, comparing the results obtained with two RAID groups and four RAID groups showed varying reductions in query execution times.

To identify the factor causing the presence or absence of performance improvements, we compared q8, which exhibited near-linear improvements in execution times when we increased the number of RAID groups from one to two, and then to four; and q13, which showed no improvement when we increased the number of RAID groups. Shown below is the temporal change in CPU use during the execution of q8 and q13.

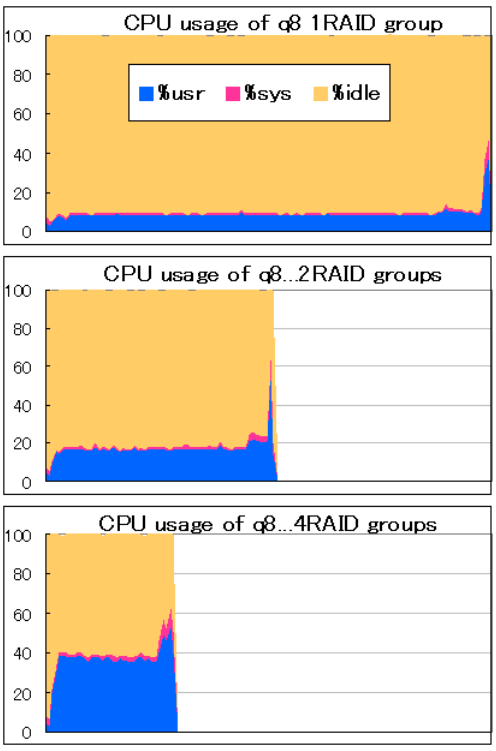

Figure 21 Temporal change in CPU use during execution of q8 (from top to bottom: single RAID group, two RAID groups, and four RAID groups)

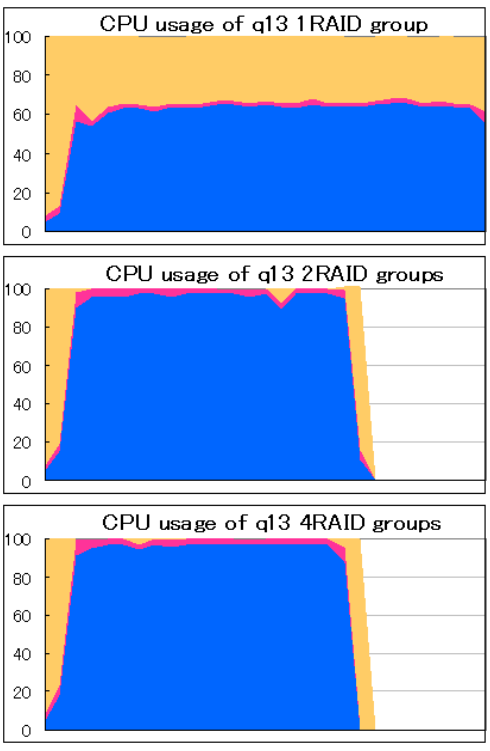

Figure 22 Temporal change in CPU use during execution of q13 (from top to bottom: single RAID group, two RAID groups, and four RAID groups)

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

As shown in Figure 21, CPU use during execution of q8 doubled when we increased the number of RAID groups from one to two. Since it showed some margin for CPU use even when four RAID groups are used, adding more RAID groups should further improve query performance.

As shown in Figure 22, CPU use during execution of q13 was approximately 60% with a single RAID group. Increasing the number of RAID groups to two boosted CPU use to 100%.

With q13, disk I/O performance constituted the bottleneck with a single RAID group during query execution. When we increased the number of RAID groups to two or more, however, CPU performance became the bottleneck. This appears to explain why increasing the number of RAID groups from two to four did not improve query performance.

2.2. ASM rebalancing costs

We can migrate data from a one-RAID-group environment to a two-RAID-group environment by preparing a two-RAID-group environment and reloading data, in addition to rebalancing the data stored on the ASM disks. Below, we consider ASM rebalancing and reloading with respect to time-based costs.

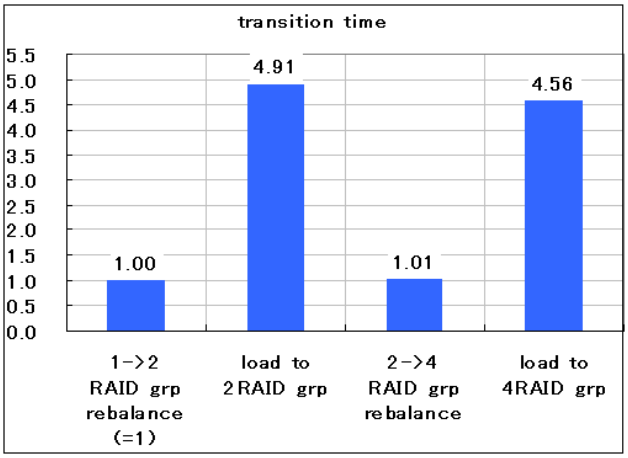

Figure 23 Relative times required for rebalancing and reloading

Figure 23 shows the relative time values required for rebalancing and reloading, with a baseline value of "1" assigned to the time required to rebalance data when we increase the number RAID groups from one to two. (This test omitted data segment compression.) The reload times include times for direct loading of data into empty tables and the time required to generate indices.

Actually reloading data to a new environment after adding disks to an existing environment must also include the time required to unload data from the existing environment, in addition to the two factors mentioned above. As Figure 23 shows, the time required to rebalance data when we increased the number of RAID groups from one to two in the verification environment was more or less equivalent to the time required to rebalance data when we increased the number of RAID groups from two to four.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

2.3. Summary of dynamic disk addition and data balancing via ASM

In general, clients add disks to a database system when capacity begins to run out. Since ASM rebalances the existing data during disk addition; disk I/O performance improves, alongside expanded capacity. In our verification testing, for queries in which disk I/O performance represents the bottleneck, we increased the number of RAID groups using ASM's dynamic disk addition and data rebalancing functions. Our observations indicated near-linear performance gains.

Dynamic disk addition and data rebalancing by ASM offer other benefits, including shorter required times and uninterrupted application operations. When a system is used continuously, growing data volumes are inevitable. When disk I/O performance reaches its limit due to increasing data volumes, slowing query performance, ASM makes it possible to add disks without halting applications to leverage disk I/O performance and improve query performance. However, if we used a method other than ASM for the data file memory area, rebalancing existing data after adding disks would require halting applications before data unloading, data reloading, reacquisition of statistical information, or index reproduction, which would in most cases incur large performance costs. Given these drawbacks, ASM is the preferred option.

3. Combined use of data segment compression and disk addition

So far, we have separately addressed improvements in query performance achieved by data segment compression and by ASM's dynamic disk addition function. We would expect combining both functions to result in still greater improvements in query performance.

Figure 24 shows query execution times for non-compressed data and compressed data using one, two, and four RAID groups. Increasing the number of RAID groups with ASM's dynamic disk addition and data rebalancing functions improves performance significantly with virtually all queries, whether or not the data in question is compressed.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

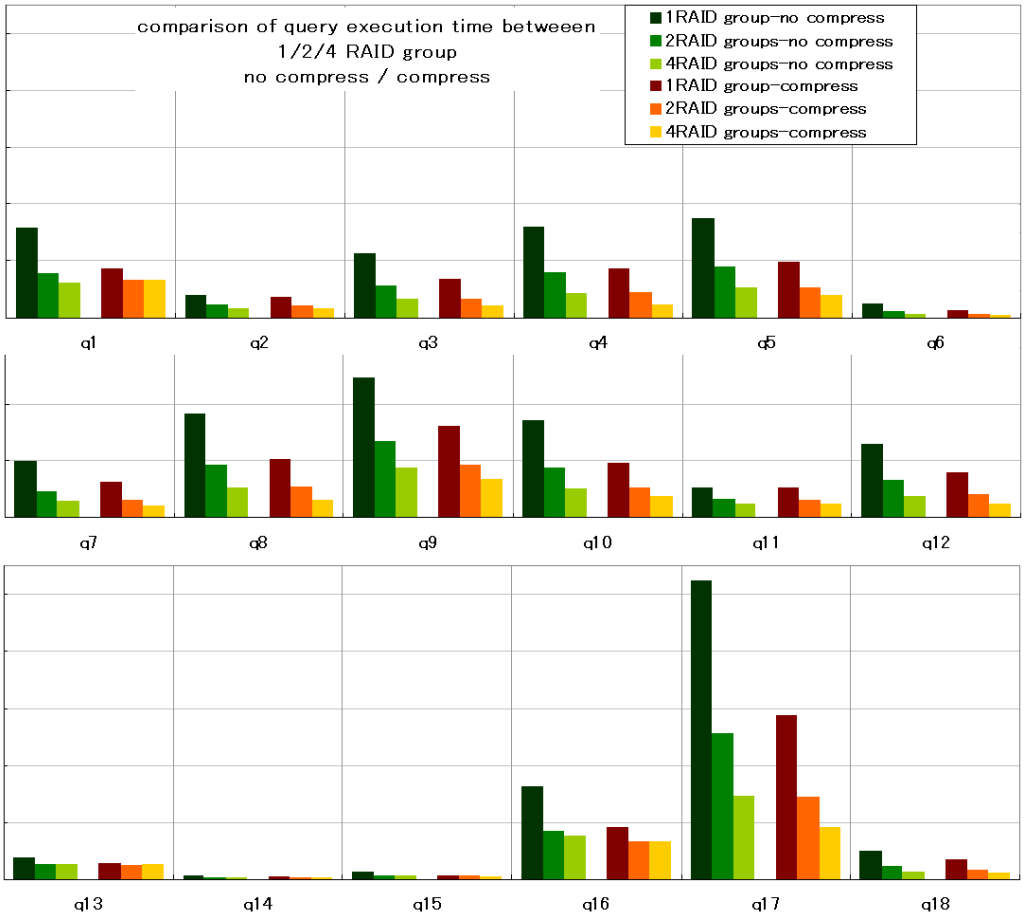

Figure 24 Combining data segment compression and disk addition

Although the data segment compression function and ASM's dynamic disk addition and data rebalancing functions improve query performance when used separately, the above results confirmed that their combined use would improve performance still further.

When disk I/O performance was low (for example, when data was stored on a single RAID group in our verification tests), query execution times involving a compressed table were approximately 60% of query execution times for a non-compressed table, indicating that data segment compression improves performance significantly even in environments characterized by low disk I/O performance.

Improvements in query performance attributable to compression were lower in cases of high disk I/O performance (e.g., when data was stored on four RAID groups) than when the disk I/O performance was low. Nevertheless, we confirmed that query execution times for compressed tables were shorter than for non-compressed tables.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

I. Conclusion

The results of our verification tests confirmed that data segment compression effectively reduces data volumes in a DWH environment. In addition to enabling more effective use of storage, data segment compression decreases the I/O required to read data, thereby improving query performance. Although data compression does generate some overhead during data loads, the query performance gains far exceed this drawback. Data segment compression unquestionably improves overall DWH system performance. By examining the data spread and other characteristics and selectively compressing tables most suitable for data compression, we can improve query performance while minimizing CPU overhead, thereby achieving significant improvements in overall system performance.

We also verified the effectiveness of ASM's dynamic disk addition and data rebalancing functions when adding disk drives to improve database performance. In an ASM environment, we can forego the various cumbersome procedures otherwise needed to add disk drives, such as data backups, disk addition/reconfiguration, backup data reloading, index reproduction, and reacquisition of statistical data. Through Oracle Enterprise Manager or via the command line for ASM disk addition, we can rebalance data to all disks in use to disperse I/O and improve database performance.

Data segment compression function and ASM's dynamic disk addition and data rebalancing functions are effective when used individually, but combining them improves performance still further. Our verification tests confirmed improvements in query performance following disk addition and data rebalancing with ASM, with and without data segment compression. We also verified shorter query execution times with compressed data, even if the number of RAID groups used remained constant.

These results indicate that Oracle Database 11g's data segment compression function and ASM's dynamic disk addition and data rebalancing functions improve the performance of DWH systems.

During our testing, the SPARC Enterprise and Oracle Database 11g operated stably and exhibited high performance at all times, requiring no system shutdowns or rebooting, despite running under constant high-load conditions involving high-volume data loading, repeated high-load search operations, and repeated tabulation processes, all of which are typical of DWH systems.

The SPARC Enterprise M5000 is equipped with the SPARC64 VI, the latest high-speed processor, for superb database processing performance. It also features extensive data protection and redundancy design to ensure high reliability. Combined with the Oracle Database 11g segment compression function, which improves performance when processing large data volumes, and with ASM's functions, the SPARC Enterprise M5000 realizes a high-performance, high-reliability system capable of continuously handling the increasing data volumes and processing requirements imposed by ever-growing databases.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

J. Reference Information

Oracle Japan Website

- Oracle Japan / Oracle GRID Center

  http://www.oracle.co.jp/solutions/grid_center/

Fujitsu Websites

- SPARC Enterprise UNIX Servers

  http://www.fujitsu.com/global/services/computing/server/sparcenterprise/

- ETERNUS Storage Systems

  http://www.fujitsu.com/global/services/computing/storage/system/

- Oracle Software

  http://software.fujitsu.com/jp/oracle/

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved
Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Oracle Corporation Japan

Oracle Aoyama Center

2-5-8, Kita Aoyama, Minato-ku, Tokyo

107-0061 Japan

FUJITSU LIMITED

Shiodome City Center, 1-5-2,

Higashi-Shimbashi, Minato-ku, Tokyo

105-7123, Japan

Copyright © 2008 Oracle Corporation Japan. All Rights Reserved.

Copyright © 2008 FUJITSU LIMITED, All Rights Reserved

Duplication prohibited

This document is provided for informational purposes only. The contents of the document are subject to change without notice. Neither Oracle Corporation Japan nor Fujitsu Limited warrant that this document is error-free, nor do they provide any other warranties or conditions, whether expressed or implied, including implied warranties or conditions concerning merchantability or fitness for a particular purpose. Oracle Corporation Japan specifically disclaims any liability with respect to this document. This document does not form any contractual obligations, either directly or indirectly. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without prior written permission from Oracle Corporation Japan.

This document is intended to provide technical information regarding the results of verification tests conducted at the Oracle GRID Center. The contents of the document are subject to change without notice to permit improvements. Fujitsu Limited makes no warranty regarding the contents of this document; nor does it assume any liability for damages resulting from or related to the document contents.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation in the United States and its subsidiaries and affiliates. Other product names mentioned are trademarks or registered trademarks of their respective companies.

UNIX is a registered trademark of The Open Group in the United States and other countries.

All SPARC trademarks are used under license from SPARC International, Inc. in the United States and other countries, and are registered trademarks of that company in the United States and other countries. Products bearing the SPARC trademark are based on an architecture developed by Sun Microsystems, Inc.

SPARC64 is used under license from SPARC International, Inc. in the United States, and is a registered trademark of that company.

Sun, Sun Microsystems, the Sun logo, Solaris, and all Solaris-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries, and are used under license from that company.

Other product names mentioned are the product names, trademarks, or registered trademarks of their respective companies.

Note that system names or product names in this document may not be accompanied by trademark notices (®, ™).